



Weave

Business Integration Framework

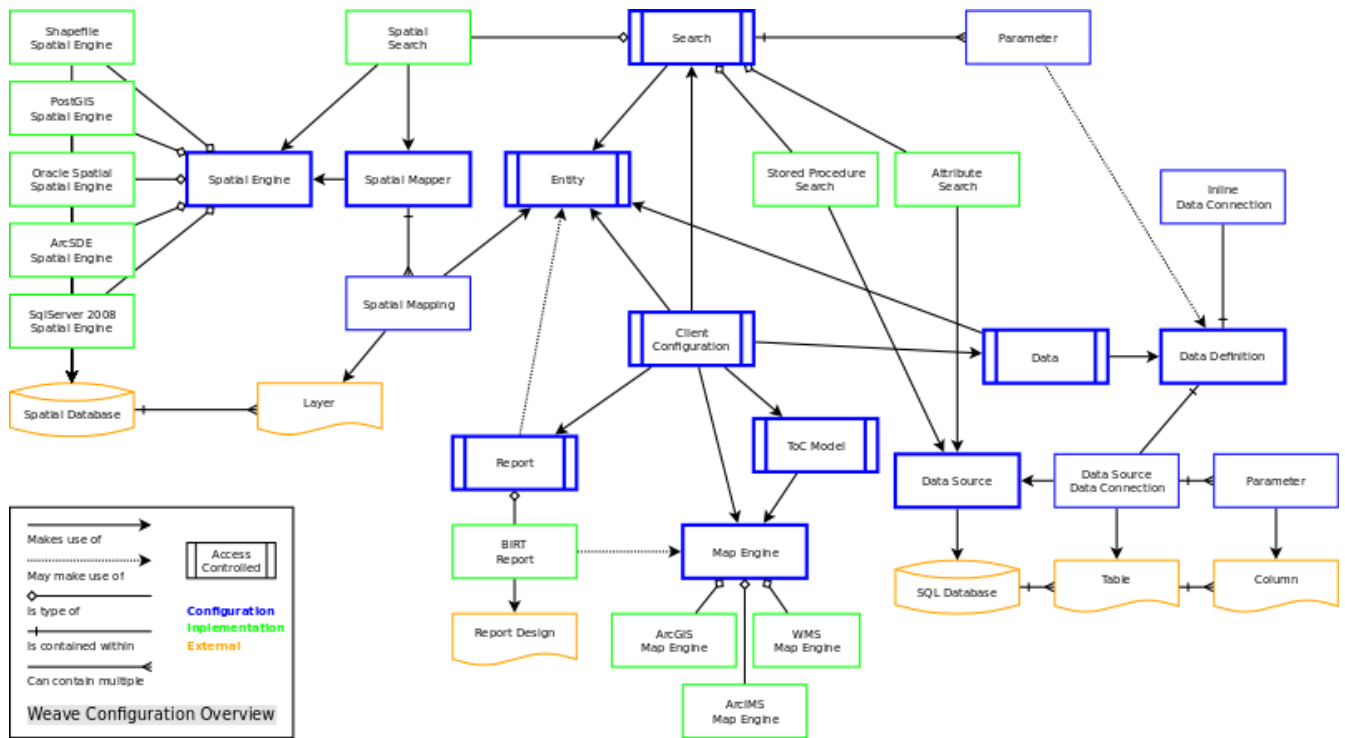
Weave System Administrator Guide Excerpt Feb 2022

1. Configuration Reference	4
1.1 Access Control List	4
1.2 Spatial Engine	8
1.2.1 Spatial Engine ArcSDE	10
1.2.2 Spatial Engine Geometryless	12
1.2.3 Spatial Engine MySQL	14
1.2.4 Spatial Engine OracleSpatial	15
1.2.5 Spatial Engine PostGIS	19
1.2.6 Spatial Engine Shapefile	23
1.2.7 Spatial Engine SQLServer	24
1.2.8 Spatial Engine WFS	28
1.2.9 Spatial Engine WFS-NG	29
1.2.10 Spatial Engine Cache	30
1.3 Spatial Mapper	31
1.4 Map Engine	36
1.4.1 Map Engine WMS	38
1.4.2 Map Engine ArcIMS	45
1.4.3 Map Engine ArcGIS (SOAP)	46
1.4.3.1 ArcGIS Server map caches	53
1.4.4 Map Engine ArcGIS (REST)	59
1.4.5 Tiled Map Engine	64
1.4.6 Map Engine Selection	73
1.4.7 Weave Map Engine	81
1.4.8 Dynamic Map Engine	85
1.4.9 Layer Filtering	87
1.5 Table Of Contents	92
1.6 Data Source	100
1.7 Data Definition	103
1.7.1 Aggregate Data Connection	105
1.7.2 Datasource Data Connection	105
1.7.3 Group Data Connection	115
1.7.4 Inline Data Connection	118
1.7.5 Spatial Data Connection	120
1.7.6 Spatial Intersection Data Definition	122
1.8 Data	124
1.9 Entity	126
1.10 Cache	127
1.11 Search	128
1.11.1 Attribute Search	129
1.11.2 Spatial Search	137
1.11.3 Stored Procedure Search	140
1.12 Report	144
1.12.1 Emailing Reports	146
1.13 Client	150
1.13.1 Client Layout	166
1.13.2 Client Views	171
1.13.2.1 Client Views Data	172
1.13.2.1.1 Detailed data formatting instructions	178
1.13.2.1.2 Common Grid Panel Configuration Options	182
1.13.2.2 Client Views Details	183
1.13.2.3 Client Views Entity Selector	186
1.13.2.4 Client Views HTML	193
1.13.2.5 Client Views Identify	193
1.13.2.6 Client Views Legend	194
1.13.2.7 Client Views Map	196
1.13.2.8 Client Views Projections	214
1.13.2.9 Client Views Quick Search	216
1.13.2.10 Client Views Search	216
1.13.2.11 Client Views Simple Map	217
1.13.2.12 Client Views ToC	219
1.13.3 Client Toolbars	222
1.13.4 Client Actions	226
1.13.4.1 Client Actions Adhoc Query	230
1.13.4.2 Client Actions Bookmarks	231
1.13.4.2.1 Client Actions Bookmarks - Standard Bookmark Action	231
1.13.4.2.2 Client Actions Bookmarks - Published Bookmark Action	234
1.13.4.2.3 Client Actions Bookmarks - Bookmark Manager	235
1.13.4.3 Client Actions Coordinate Zoom	235
1.13.4.4 Client Actions Feedback	237
1.13.4.5 Client Actions Export Grid	239
1.13.4.6 Client Actions Grid Export	240
1.13.4.7 Client Actions Help	241
1.13.4.8 Client Actions Identify	242
1.13.4.9 Client Actions Legend	243
1.13.4.10 Client Actions Map Tips	243
1.13.4.11 Client Actions Measure Polygon	247
1.13.4.12 Client Actions Measure Polyline	248

1.13.4.13 Client Actions Projections	249
1.13.4.14 Client Actions Redline	250
1.13.4.14.1 Client Actions Redline - Export to Shapefile	252
1.13.4.15 Client Actions Selection Identify	253
1.13.4.16 Client Actions Selection Undo	255
1.13.4.17 Client Actions Simple Report Menu	255
1.13.4.18 Client Actions Spatial Identify	256
1.13.4.19 Client Actions Toc Menu	259
1.13.4.20 Client Actions Transfer	260
1.13.4.21 Client Actions Upload	262
1.13.4.21.1 Client Actions Upload - Obsolete	270
1.13.4.22 Client Actions ZipNShip	274
1.13.4.23 Client Actions ZipNShip Envelope	276
1.13.5 Client Components	278
1.13.5.1 Client Components Entity	280
1.13.6 Client ContextMenus	281
1.13.7 Client Samples	283
1.13.8 Client Snippets	285
1.13.9 Client Storage	293
1.13.10 Client Plugins	295
1.13.10.1 Client Plugin Map Popup	295
1.13.11 Internationalisation and Localisation	296
1.13.11.1 Current Client i18n Resources	299
1.13.12 Client User Attributes	317
1.14 Pool	319
1.15 DMS	321
1.15.1 Database DMS	322
1.15.2 DMS related bundles	323
1.15.3 Document Upload	323
1.16 Indexing	324
1.16.1 Index Search Client Configuration	325
1.16.2 Index Search Server Configuration	327
1.16.3 Indexing and Quick Search Tool Best Practice	336
1.17 Editing	339
1.17.1 Non-spatial Editor	339

Configuration Reference

The following diagram shows the relationship between some of the core configuration items.



Larger Version: [Config Item Overview Large.png](#)

Access Control List

An ACL (Access Control List) provides restrictions on who can access a particular configuration item.

When a process initiated by the user attempts to access a restricted item (which is anything with an ACL attached) the groups that the user belongs are checked against the ACL to determine if the user should be given access to the item. See [Security](#) for details on how the groups that a user belongs to is determined.

ACL's can be setup to either deny access to everything and selectively enable access, or to allow access to everything and selectively deny access. The former is more secure but is a lot harder to implement as it requires that an ACL is attached to everything. Generally if access control is required on individual items it's easiest to alter the default ACL to allow access then selectively restrict access to those items that require it.

An ACL provides a list of groups that can either be denied access or allowed access and are processed linearly until there is a match that positively denies or grants approval. Also, there is a special group, `anonymous`, that can be used to allow or deny access to users that are not logged in.

Additionally, ACL's can reference other ACL's to provide a hierarchy with recursive checking performed to determine accessibility.

A special ACL with the id `acl.default` can be setup to determine what will happen if no ACL entry is found that can positively grant or deny permission.

ACL's can be setup either in-line, that is included directly in the item they're guarding access to, or can be configured individually and referenced by an item.

Important Note

There is no access control at all unless at least one ACL is defined

This means that a default installation will provide no access control to any items until at least one ACL is defined, even if it's just `default.acl`.

Once at least one ACL is defined then access control restrictions will be enabled. This means that if you are creating a public (or non-sensitive) installation then you do not need to worry about access control lists at all.

It is recommended that if you are going to be having any sensitive information then the first thing you do is enable the default ACL and set it to deny access to everyone.

As of Weave 2.5.25 it's possible to also list individual users in an ACL. Before then you could only reference groups that users belong to, but this has to be enabled by setting a system property (because of the security implications).

To do this you need to define the property in `... \weave\service\conf\wrapper.conf`, and/or `... \weave\startup.cmd`, depending upon how you start Weave.

For example in `wrapper.conf`, and the bottom of the file, there will already be a number of system properties defined, e.g.

```
wrapper.java.additional.4 = -Dfile.encoding=UTF-8
```

you just need to find the highest number, add one to it, and create a new line with the named property and value, so for the current default version of that file you'd add:

```
wrapper.java.additional.22 = -Dweave.enable.user.acl=true
```

You should be able to figure it out for `startup.cmd`, it's the same basic premise.

Once you've added the system property and restarted Weave you should be able to directly reference user names in the allow/deny entries in an ACL.

Namespace

`urn:com.cohga.server.acl#1.0`

Tags

acl

Properties

Name	Type	Required	Description
id	string	yes	unique identifier

Sub-tags

Name	Type	Cardinality
entry	<code>urn:com.cohga.server.acl#1.0:entry</code>	1..n

Content

None

entry

Properties

Name	Type	Required	Description
type	'allow', 'deny' or 'acl'	yes	decides if this entry should allow or deny access or is a reference to another ACL

Sub-tags

None

Content

The name of a group that the user belongs to, * to match any group, or a reference to another `urn:com.cohga.server.acl#1.0:acl`

Examples

Allow access to everything then remove access, this is the easiest setup but a mistake could allow access to restricted information

```

<!-- Allow access to anyone as default, then restrict the important
stuff -->
<!-- If no ACL is specified or none of the specified ACL's produce a
match -->
<!-- then acl.default will be used -->
<acl:acl id="acl.default">
    <entry type="allow">*</entry>
</acl:acl>

<!-- Only users with the ROLE_ADMINISTRATOR role get access to
'private' stuff -->
<!-- everyone else is explicitly denied -->
<acl:acl id="private">
    <entry type="allow">ROLE_ADMINISTRATOR</entry>
    <entry type="deny">*</entry>
</acl:acl>

<!-- ROLE_ADMINISTRATOR and ROLE_USER get access to 'internal' stuff
-->
<!-- everyone else is explicitly denied -->
<acl:acl id="internal">
    <entry type="allow">ROLE_ADMINISTRATOR</entry>
    <entry type="allow">ROLE_USER</entry>
    <entry type="deny">*</entry>
</acl:acl>

<!-- everyone gets access to roads and property -->
<entity:entity id="road">
    <label>Road</label>
</entity:entity>

<entity:entity id="property">
    <label>Property</label>
</entity:entity>

<!-- users matching the 'internal' acl get access to rates -->
<entity:entity id="rates">
    <label>Rates</label>
    <acl:acl id="internal"/>
</entity:entity>

<!-- users matching the 'private' acl get access to uers -->
<entity:entity id="users">
    <label>Users</label>
    <acl:acl id="private"/>
</entity:entity>

```

Deny access to everything then add access, most secure option, but a lot of work to ensure you attach an ACL to *everything*

```

<!-- Set deny as default, but now we have to make sure we set access
explicitly for everything -->
<!-- we don't really need to do this since it happens as soon as we
create an ACL, but for completeness... -->
<acl:acl id="acl.default">
    <entry type="deny">*</entry>
</acl:acl>

<!-- Create a private ACL, but fall back to acl.default -->
<!-- ROLE_ADMINISTRATOR will be allowed -->
<!-- anyone else will fall back to acl.default -->
<acl:acl id="private">
    <entry type="allow">ROLE_ADMINISTRATOR</entry>
</acl:acl>

<!-- Create an internal ACL, but fall back to acl.default -->
<!-- ROLE_ADMINISTRATOR and ROLE_USER will be allowed -->
<!-- anyone else will fall back to acl.default -->
<acl:acl id="internal">
    <entry type="allow">ROLE_ADMINISTRATOR</entry>
    <entry type="allow">ROLE_USER</entry>
</acl:acl>

<acl:acl id="anyone">
    <entry type="allow">*</entry>
</acl:acl>

<!-- now we have to explicitly grant access to roads and property -->
<entity:entity id="road">
    <label>Road</label>
    <acl:acl id="anyone"/>
</entity:entity>

<entity:entity id="property">
    <label>Property</label>
    <acl:acl id="anyone"/>
</entity:entity>

<entity:entity id="rates">
    <label>Rates</label>
    <acl:acl id="internal"/>
</entity:entity>

<entity:entity id="users">
    <label>Users</label>
    <acl:acl id="private"/>
</entity:entity>

```

Recommended default acl usage example for internal Weave instance

```

    <acl:acl id="acl.default">
      <!-- Setup the default acl so that users have to be
logged in before they can access the system by denying access to
anonymous users -->
      <entry type="deny">anonymous</entry>
      <!-- but still provide access to everything that
hasn't explicitly been denied with other acl's -->
      <entry type="allow">*</entry>
    </acl:acl>

    <!-- Attach this acl to items that only planners should have
access to -->
    <acl:acl id="planners">
      <entry type="allow">ROLE_PLANNERS</entry>
      <entry type="deny">*</entry>
    </acl:acl>

    <!-- Attach this acl to items that only engineers should have
access to -->
    <acl:acl id="engineers">
      <entry type="allow">ROLE_ENGINEERS</entry>
      <entry type="deny">*</entry>
    </acl:acl>

```

Spatial Engine

A spatial engine provides access to a service that can perform spatial operations, such as determining what entities fall within a polygon, but it also provides a spatial perspective to the entities that the system will use. For example providing details about the coordinate reference system that the entity is stored in.

This will generally be something like ArcSDE, Oracle Spatial, Shapefiles or WFS, and usually at least one will be defined.

Because there are different spatial engines available there are different configuration parameters for each one so the content of this tag is different depending upon the actual spatial engine implementation.

Namespace

urn:com.cohga.server.spatial.geotools#1.0

Tags

spatialengine

Properties

Name	Type	Required	Description
id	string	yes	unique identifier

Sub-tags

Depends on type of spatial engine

Content

Depends on type of spatial engine

Examples

Connecting to ArcSDE

```

<spatial:spatialengine id="arcsde">
  <dbtype>arcsde</dbtype>
  <server>hostname</server>
  <port>5151</port>
  <user>username</user>
  <password>password</password>
  <pool>
    <minConnections>2</minConnections>
    <maxConnections>5</maxConnections>
    <timeOut>10000</timeOut>
    <testOnBorrow>true</testOnBorrow>
    <testOnReturn>false</testOnReturn>
    <testWhileIdle>false</testWhileIdle>
    <timeBetweenEvictionRunsMillis>5000<
/timeBetweenEvictionRunsMillis>
    <whenExhaustedAction>block</whenExhaustedAction>
  </pool>
</spatial:spatialengine>

```

Connecting to Oracle Spatial

```

<spatial:spatialengine id="oracle">
  <dbtype>oracle</dbtype>
  <host>hostname</host>
  <port>1521</port>
  <user>username</user>
  <passwd>password</passwd>
  <schema>SCHEMA</schema>
  <instance>instance</instance>
</spatial:spatialengine>

```

Connecting to Microsoft SQL Server

```

<spatial:spatialengine id="sqlserver">
  <dbtype>sqlserver</dbtype>
  <host>hostname</host>
  <port>1433</port>
  <user>username</user>
  <passwd>password</passwd>
  <database>password</database>
  <schema>SCHEMA</schema>
  <geometrymetadatatable>v_geometry_columns</geometrymetadatatable>
</spatial:spatialengine>

```

Connecting to Microsoft SQL Server using JNDI

```
<spatial:spatialengine id="sqlserver">
  <dbtype>sqlserverjndi</dbtype>
  <jndi><![CDATA[ java:comp/env/jdbc/spatialDS ]]></jndi> <!-- Prior
to 2.5.28 this must be <jndiReferenceName>...</jndiReferenceName> -->
</spatial:spatialengine>
```

Connecting to a Shapefile directory

```
<spatial:spatialengine id="shapefile">
  <dbtype>shapefiledir</dbtype>
  <url><![CDATA[file:C:/data_dir/data]]></url>
  <memorymapped>true</memorymapped>
</spatial:spatialengine>
```

Connecting to WFS

```
<spatial:spatialengine id="wfs">
  <dbtype>wfs</dbtype>
  <url><![CDATA[http://hostname/wfs]]></url>
  <protocol>get</protocol>
  <username>username</username>
  <password>password</password>
  <timeout>5000</timeout>
  <bufferize>100000</bufferize>
  <gzip>true</gzip>
  <lenient>true</lenient>
</spatial:spatialengine>
```

Connecting to ArcGIS

```
<spatial:spatialengine id="arcgis">
  <dbtype>arcgis</dbtype>
  <url><![CDATA[https://services.arcgis.com/3vStCH7NDoBOZ5zn
/arcgis/rest/services/Potable_Water_Pipe/FeatureServer]]></url>
  <crs>EPSG:28356</crs>
</spatial:spatialengine>
```

Spatial Engine ArcSDE

A spatial engine provides access to a service that can perform spatial operations, one of the available spatial engines is for ArcSDE.

ArcSDE 10.3

Cohga does not recommend sites update beyond ArcSDE 10.2. ArcSDE 10.3 provides little additional functionality (in fact none that I can find) and ESRI has removed the ability for third party applications (like Weave) to connect to ArcSDE.

If you do migrate to ArcSDE 10.3 and wish to continue to use data managed by ArcSDE with Weave you will need to ensure that ArcSDE stores the data using the underlying database native geometry type and have Weave connect to the database directly.

Supporting Direct Connect is not a trivial matter, it requires a full installation of the ArcSDE client on the Weave server and modifications to the Weave startup (to include the ArcSDE native libraries in Weave). The current stable release of Weave contains libraries for ArcSDE version 9.3.1, so only support ArcSDE 9.3.1 and earlier for direct connect. However, the libraries for ArcSDE 10 are available and can be provided on request for sites attempting direct connect to ArcSDE 10.

The following links may help.

- [ArcSDE database connection properties - link to ESRI web site](#)
- [Setup requirements - link to GeoTools web site](#)

Namespace

urn:com.cohga.server.spatial.geotools#1.0

Tags

spatialengine

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
dbtype	'arcsde'	yes	The name of the driver to use, in this case ArcSDE
server	string	yes	The hostname or ip address of the ArcSDE server
port	string	yes	The port number that the ArcSDE server is running on or the direct connect description
instance	string	no	The specific database to connect to. Only applicable to certain databases. Value ignored if not applicable
user	string	yes	The userid used when connecting to ArcSDE
password	string	yes	The password used when connecting to ArcSDE
database.version	string	no	Which version to use, the default will be used if not set.
datastore.allowNonSpatialTables	boolean	no	Should non-spatial tables also be exposed.
datastore.allowUnregisteredTables	boolean	no	Should non-registered tables also be exposed.

Sub-tags

Name	type	cardinality
pool	urn:com.cohga.server.spatial.geotools#1.0:pool	0..1

Content

None

pool

Properties

Name	Type	Required	Default	Description
minConnections	number	no	2	The number of connection to create at startup.
maxConnections	number	no	6	The maximum number of objects that can be borrowed from the pool at one.
timeOut	number	no	500	The maximum amount of time to wait, in milliseconds, for an object when the pool is exhausted and whenExhaustedAction is 'block' (otherwise ignored).
minIdle	number	no	0	The minimum number of idle connections in the pool. This is always 0 now.
maxIdle	number	no	-1	The maximum number of idle connections in the pool. This is always -1 (no limit) now.
testOnBorrow	boolean	no	true	Should the connection be checked for validity when being borrowed from the pool.

testOnReturn	boolean	no	false	Should the connection be checked for validity when being returned to the pool.
testWhileIdle	boolean	no	false	Should the connection be checked for validity when just sitting in the pool.
timeBetweenEvictionRunsMillis	number	no	-1	Sets the number of milliseconds to sleep between runs of the idle connection evictor. This needs to be set if 'testWhileIdle' has been set to true.
whenExhaustedAction	'fail', 'grow' or 'block'	no	block	Sets the action to take when the pool is exhausted (the maximum number of "active" objects has been reached). This value is 'block' when timeOut > 0, or 'fail' when timeOut <= 0, and 'grow' is no longer supported.
minEvictableIdleTimeMillis	number	no	30000	The minimum number of milliseconds an connection can sit idle in the pool before it is eligible for eviction.
softMinEvictableIdleTimeMillis	number	no	-1	The minimum number of milliseconds an connection can sit idle in the pool before it is eligible for eviction with the extra condition that at least "minIdle" amount of connections remain in the pool. This is no longer supported/required.
lifo	boolean	no	true	If true then the last connection returned to the pool will be the next one handed out, if false then the oldest connection returned to the pool will be the next one handed out.

Sub-tags

None

Content

None

Examples

```

<geotools:spatialengine id="arcsde">
  <dbtype>arcsde</dbtype>
  <server>sddev</server>
  <port>5151</port>
  <user>gis</user>
  <password>b0rdumm</password>
  <pool>
    <minConnections>2</minConnections>
    <maxConnections>16</maxConnections>
    <timeout>1000</timeout>
    <testOnBorrow>true</testOnBorrow>
    <testOnReturn>true</testOnReturn>
    <testWhileIdle>false</testWhileIdle>
    <minEvictableIdleTimeMillis>180000<
  /minEvictableIdleTimeMillis>
  </pool>
  <database.version>PUBLIC</database.version>
  <datastore.allowNonSpatialTables>true</datastore.
allowNonSpatialTables>
</geotools:spatialengine>

```

Spatial Engine Geometryless

The "geometryless" spatial engine is intended to add support for database tables that have an x and y location.

The driver emulates a spatially enabled table by generating point geometry on the fly based on the values in the x and y columns in the underlying tables.

The driver tries to expose every table in the database that it's pointed to as a spatially enabled table.

Name	Type	Required	Default	Description
dbtype	String	Yes		Must be 'locationsxy'
driver	String	Yes		Java Class name of an installed JDBC driver
urlprefix	String	Yes		Complete JDBC URL for the database connection
user	String	Yes		Username to connect to the database
password	String	Yes		Password to connect to the database
xcolumn	String	Yes		Name of JDBC results column containing easting (x, longitude etc)
ycolumn	String	Yes		Name of JDBC results column containing northing (y, latitude etc)
geom_name	String	Yes		Name of the geometry attribute generated from the x,y columns. This is a virtual column that will be added to each table and will be a Point that is made from the values in <i>xcolumn and ycolumn</i> . The value used here doesn't have any external effects, it just provides the name of the column that will be created. examples are "shape" or "the_geom" or just "geom"
schema	String	no	none	Database schema
charset	String	no	none	Database character set
crs	String	no	none	A default CRS to use for the data contained within the tables

Example

```
<spatial:spatialengine id="test">
  <dbtype>locationsxy</dbtype>
  <driver>com.microsoft.sqlserver.jdbc.SQLServerDriver</driver>
  <urlprefix><![CDATA[jdbc:sqlserver://sqlserver2008;
instanceName=SQLEXPRESS;databaseName=GIS]]></urlprefix>
  <user>gis</user>
  <passwd>ENCSSJKXUMJBMTPGEJFGJZJ</passwd>
  <xcolumn>x</xcolumn>
  <ycolumn>y</ycolumn>
  <geom_name>shape</geom_name>
  <crs>EPSG:28355</crs>
  <primarykeymetadata>
    <table name="WkALR" column="fid" />
    <table name="WkPRN" column="fid" />
  </primarykeymetadata>
</spatial:spatialengine>
```

<Edit>

Spatial Engine MySQL

A spatial engine provides access to a service that can perform spatial operations, one of the available spatial engines is for MySQL.

Namespace

urn:com.cohga.server.spatial.geotools#1.0

Tags

spatialengine

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
dbtype	'mysql'	yes	The name of the driver to use, in this case MySQL
host	string	yes	The hostname or ip address of the MySQL server
port	number	yes	The port number that the MySQL instance is running on
database	string	yes	The database to connected to
user	string	yes	The userid used when connecting to the database
passwd	string	yes	The password used when connecting to the database
minconnections	integer	no	The minimum number of connections to open to the database
maxconnections	integer	no	The maximum number of connections to open to the database
validateconnections	boolean	no	Should database connections be checked on each access to see if they're still valid
namespace	string	no	The namespace prefix

Sub-tags

None

Content

None

Examples

```
<spatial:spatialengine id="mysql">
  <dbtype>mysql</dbtype>
  <host>mysqlhost</host>
  <port>3306</port>
  <database>spatial</database>
  <user>gis</user>
  <passwd>hak0rz</passwd>
</spatial:spatialengine>
```

Spatial Engine OracleSpatial

A spatial engine provides access to a service that can perform spatial operations, one of the available spatial engines is for Oracle Spatial.

ESRI GeoDatabase enabled Oracle databases are currently not supported by Weave as a spatial engine. If access is required it will have to be via ArcGIS server.

Namespace

urn:com.cohga.server.spatial.geotools#1.0

Tags

spatialengine

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
dbtype	'oracle'	yes	The name of the driver to use, in this case Oracle
host	string	yes	The hostname or ip address of the Oracle server
port	number	yes	The port number that the Oracle instance is running on
database	string	no	The database parameter for the connection url
username	string	yes	The userid used when connecting to the database
password	string	yes	The password used when connecting to the database
schema	string	no	The schema used when connecting to the database
minconnections	integer	no	The minimum number of pooled connection. Default 1
maxconnections	integer	no	The maximum number of open connections. Default 10
maxopenpreparedstatements	integer	no	Set the maximum number of prepared statements, -1 to disable prepared statement caching.
validateconnections	boolean	no	Check connection is alive before using it. Default false
fetchsize	integer	no	The number of records read with each interaction with the dbms. Default 1000
connectiontimeout	integer	no	The number of seconds the connection pool will wait before timing out attempting to get a new connection. Default 20 (seconds)
primarykeymetadatable	string	no	The optional table containing primary key structure and sequence associations. Can be expressed as 'schema.name' or just 'name'. Default none

maxopenpreparedstatements	integer	no	Maximum number of prepared statements kept open and cached for each connection in the pool. Set to 0 to have unbounded caching, to -1 to disable caching. Default 50
geometrymetadataatable	string	no	The optional table containing metadata about the geometry for the tables being accessed via this spatial engine.

Sub-tags

None

Content

None

Notes

No connection pool configuration is provided with this driver, the driver manages the connection pooling itself.

If database starts with '(' then the format of the connection url will be

```
jdbc:oracle:thin:@database
```

If database starts with '/' then the format of the connection url will be

```
jdbc:oracle:thin:@//host:portdatabase
```

else the format of the connection url will be

```
jdbc:oracle:thin:@host:port:database
```

Examples

Basic Oracle connection

```
<spatial:spatialengine id="oracle">
  <dbtype>oracle</dbtype>
  <host>oradev</host>
  <port>1521</port>
  <user>gis</user>
  <passwd>hak0rz</passwd>
  <schema>GIS</schema>
</spatial:spatialengine>
```

Basic Oracle connection with the password using the osgi encrypt command and a database specified

```
<spatial:spatialengine id="sdbeos">
  <dbtype>oracle</dbtype>
  <host>sdbeos</host>
  <port>1521</port>
  <database>staging</database>
  <user>data_viewer</user>
  <passwd>ENCSSJKXUMJBMTPEJFGCXQFXJJZUTMJZJ</passwd>
  <schema>ASSETS</schema>
</spatial:spatialengine>
```

Primary Key Metadata

If the primary key for a table can not be determined by Weave directly from the database then the information can be provided by creating a table or directly embedding the information in the spatial engine configuration.

SQL create statement for gt_pk_metadata table

```
CREATE TABLE gt_pk_metadata (
  table_schema VARCHAR(32),
  table_name VARCHAR(32) NOT NULL,
  pk_column VARCHAR(32) NOT NULL,
  pk_column_idx INTEGER,
  pk_policy VARCHAR(32),
  pk_sequence VARCHAR(64),
  unique (table_schema, table_name, pk_column),
  check (pk_policy in ('sequence', 'assigned', 'autogenerated'))
)
```

pk_policy can be 'assigned', 'sequence' or 'autogenerated' depending upon how the key column value is generated. The pk_sequence value only need to be set if the policy is 'sequence'.

'sequence' means that the value for the column is generated using a database sequence, and the 'pk_sequence' value must be set.

'autogenerated' means that the value for the column is generated by the database using another method.

'assigned' means that the value for the column is determined by the current maximum value +1, if the column is an integral type, or as a random string if the column is textual.

Column	Description
table_schema	Name of the database schema in which the table is located.
table_name	Name of the table to be published
pk_column	Name of a column used to form the feature IDs
pk_column_idx	Index of the column in a multi-column key. In case multi column keys are needed multiple records with the same table schema and table name will be used.
pk_policy	The new value generation policy, used in case a new feature needs to be added in the table ('assigned', 'sequence' or 'autogenerated').
pk_sequence	The name of the database sequence to be used when generating a new value for the pk_column.

Using primary key metadata configuration

You can also specify the above information directly in the spatial engine configuration.

Name	Description
name	The table name
schema	The optional table schema
column	The column in the the table that contains the unique id
policy	How the key is generated
sequence	The name of the database sequence to use if the policy is 'sequence'

```

<spatial:spatialengine id="sdbeos">
  <dbtype>oracle</dbtype>
  <host>sdbeos</host>
  <port>1521</port>
  <database>staging</database>
  <user>data_viewer</user>
  <passwd>ENCSSJKXUMJBMTPEJFGCXQCQFXJJZUTMJZJ</passwd>
  <schema>ASSETS</schema>
  <primarykeymetadata>
    <table schema="ASSETS" name="PARK" column="GID" policy="
autogenerated" />
    <table schema="ASSETS" name="PROPERTY" column="PARK_ID" type="
autogenerated" />
    <table schema="ASSETS" name="ROAD" column="OBJECTID" type="
autogenerated" />
  </primarykeymetadata>
</spatial:spatialengine>

```

Geometry metadata

The Oracle data store will, by default, look into the MDSYS.USER_SDO* and MDSYS.ALL_SDO* views to determine the geometry type and native SRID of each geometry column. Those views are automatically populated with information about the geometry columns stored in tables that the current user owns (for the MDSYS.USER_SDO* views) or can otherwise access (for the MDSYS.ALL_SDO* views).

There are a few hiccups in this process:

- if the connection pool user cannot access the tables (because impersonation is being used) the MDSYS views will be empty, making it impossible to determine either the geometry type and the native SRID
- the geometry type can be specified only while building the spatial indexes, as a index constraint, however such information is often not included when creating the indexes
- the views are populated dynamically based on the current user, if the database has thousands of tables and users the views can become very slow

The database administrator can address the above issues by manually creating a geometry metadata table describing each geometry column, and then indicate its presence among the Oracle data store connection parameter named *geometrymetadatatable* (either as a simple table name, or a schema qualified one). The table has the following structure (the table name is free, just indicate the one chosen in the data store connection parameter):

```

CREATE TABLE GEOMETRY_COLUMNS (
  F_TABLE_SCHEMA VARCHAR(30) NOT NULL,
  F_TABLE_NAME VARCHAR(30) NOT NULL,
  F_GEOMETRY_COLUMN VARCHAR(30) NOT NULL,
  COORD_DIMENSION INTEGER,
  SRID INTEGER NOT NULL,
  TYPE VARCHAR(30) NOT NULL,
  UNIQUE(F_TABLE_SCHEMA, F_TABLE_NAME, F_GEOMETRY_COLUMN),
  CHECK(TYPE IN ('POINT', 'LINESTRING', 'POLYGON', 'COLLECTION',
'MULTIPOINT', 'MULTILINESTRING', 'MULTIPOLYGON', 'GEOMETRY' ) )
);

```

When the table is present the store will first search it for information about each geometry column to be classified, and fall back on the MDSYS views only if such table does not contain any information.

Using geometry metadata configuration

As of Weave 2.5.16 it's possible to specify the information contained in the geometry metadata table directly in the spatial engine configuration.

Name	Description
name	The table name
schema	The optional table schema
type	The geometry type (point, linestring, curve, polygon, multipoint, multilinestring, multicurve, multipolygon)
srid	The geometry srid
dimension	The geometry dimension

```

<spatial:spatialengine id="sdbeos">
  <dbtype>oracle</dbtype>
  <host>sdbeos</host>
  <port>1521</port>
  <database>staging</database>
  <user>data_viewer</user>
  <passwd>ENCSSJKXUMJBMTPEJFGCXCFXJJZUTMJZJ</passwd>
  <schema>ASSETS</schema>
  <geometrymetadata>
    <table name="PARK" type="multipolygon" srid="28355" dimension="2"
  />
    <table name="PROPERTY" type="polygon" srid="28355" dimension="2"/>
    <table name="ROAD" type="linestring" srid="28355" dimension="2"/>
  </geometrymetadata>
</spatial:spatialengine>

```

Spatial Engine PostGIS

A spatial engine provides access to a service that can perform spatial operations, one of the available spatial engines is for PostGIS.

ESRI GeoDatabase enabled Postgres databases are not supported by Weave as a spatial engine. If access is required it will have to be via ArcGis server.

Namespace

urn:com.cohga.server.spatial.geotools#1.0

Tags

spatialengine

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier

dbtype	'postgis'	yes	The name of the driver to use, in this case PostGIS
host	string	yes	The hostname or ip address of the server
port	number	yes	The port number that the instance is running on
database	string	yes	The database to connected to
schema	string	no	The default schema to connected to, default is 'public'
user	string	no	The userid used when connecting to the database
passwd	string	no	The password used when connecting to the database
minconnections	integer	no	The minimum number of connections to open to the database
maxconnections	integer	no	The maximum number of connections to open to the database
validateconnections	boolean	no	Should database connections be checked on each access to see if they're still valid
namespace	string	no	The namespace prefix
loosebox	boolean	no	If set to 'true' if the Bounding Box should be 'loose', faster but not as deadly accurate
estimatedextent	boolean	no	set to true if the bounds for a table should be computed using the 'estimated_extent' function, but beware that this function is less accurate, and in some cases *far* less accurate if the data within the actual bounds does not follow a uniform distribution. It also relies on the fact that you have accurate table stats available. So it is a good idea to 'VACUUM ANALYZE' the postgis table

Sub-tags

None

Content

None

Examples

Basic PostGIS connection

```
<spatial:spatialengine id="postgis">
  <dbtype>postgis</dbtype>
  <host>mysqlhost</host>
  <port>5432</port>
  <database>spatial</database>
  <user>gis</user>
  <passwd>hak0rz</passwd>
</spatial:spatialengine>
```

Primary Key Table

For specifying primary key columns when Weave/GeoTools can not determine it directly.

SQL create statement for gt_pk_metadata table

```
CREATE TABLE gt_pk_metadata (
  table_schema VARCHAR(32),
  table_name VARCHAR(32) NOT NULL,
  pk_column VARCHAR(32) NOT NULL,
  pk_column_idx INTEGER,
  pk_policy VARCHAR(32),
  pk_sequence VARCHAR(64),
  unique (table_schema, table_name, pk_column),
  check (pk_policy in ('sequence', 'assigned', 'autogenerated'))
)
```

Column	Description
table_schema	Name of the database schema in which the table is located.
table_name	Name of the table to be published
pk_column	Name of a column used to form the feature IDs
pk_column_idx	Index of the column in a multi-column key. In case multi column keys are needed multiple records with the same table schema and table name will be used.
pk_policy	The new value generation policy, used in case a new feature needs to be added in the table ('assigned', 'sequence' or 'autogenerated'). Where: 'sequence' means that the value for the column is generated using a database sequence, and the 'pk_sequence' value must be set. 'autogenerated' means that the value for the column is generated by the database using another method. 'assigned' means that the value for the column is determined by the current maximum value +1, if the column is an integral type, or as a random string if the column is textual.
pk_sequence	The name of the database sequence to be used when generating a new value for the pk_column. The pk_sequence value only need to be set if the pk_policy is 'sequence'.

Using primary key metadata configuration

You can also specify the above information directly in the spatial engine configuration.

Name	Description
name	The table name
schema	The optional table schema
column	The column in the the table that contains the unique id
policy	How the key is generated
sequence	The name of the database sequence to use if the policy is 'sequence'

Using primary key metadata

```
<spatial:spatialengine id="postgis">
  <dbtype>postgis</dbtype>
  <host>mysqlhost</host>
  <port>5432</port>
  <database>spatial</database>
  <user>gis</user>
  <passwd>hak0rz</passwd>
  <primarykeymetadata>
    <table name="park" column="gid" policy="autogenerated"/>
    <table name="property" column="park_id" type="autogenerated"/>
    <table name="road" column="objectid" type="autogenerated"/>
  </primarykeymetadata>
</spatial:spatialengine>
```

Using geometry metadata configuration

As of Weave 2.5.16 it's possible to specify the information contained in the geometry metadata table directly in the spatial engine configuration.

Name	Description
name	The table name
schema	The optional table schema
type	The geometry type (point, linestring, polygon, multipoint, multilinestring, multipolygon)
srid	The geometry srid
dimension	The geometry dimension

Using geometry metadata table

```
<spatial:spatialengine id="postgis">
  <dbtype>postgis</dbtype>
  <host>mysqlhost</host>
  <port>5432</port>
  <database>spatial</database>
  <user>gis</user>
  <passwd>hak0rz</passwd>
  <geometrymetadata>
    <table schema="public" name="park" type="multipolygon" srid="
28355" dimension="2"/>
    <table schema="public" name="property" type="polygon" srid="
28355" dimension="2"/>
    <table schema="public" name="road" type="linestring" srid="28355"
dimension="2"/>
  </geometrymetadata>
</spatial:spatialengine>
```

Spatial Engine Shapefile

A spatial engine provides access to a service that can perform spatial operations, one of the available spatial engines is for a directory of Shapefiles.

The shapefile directory spatial engine will use sub-directories to give the shapefiles kind of schema. For example if the URL parameter points to a directory that contains a single shape file called `RIVERS` then the shapefile will be registered under the name `RIVERS`, but if the URL parameter points to a directory that contains another directory called `WATER` and under that directory there is a shapefile called `RIVERS` then the shapefile will be registered as `WATER.RIVERS`.

Namespace

urn:com.cohga.server.spatial.geotools#1.0

Tags

spatialengine

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
dbtype	'shapefiledir'	yes	The name of the driver to use, in this case Shapefile
url	string	yes	The URL that points to the directory containing the shapefiles
memorymapped	boolean	no	Enable/disable the use of memory-mapped io
createspatialindex	boolean	no	Enable/disable the automatic creation of spatial index
namespace	string	no	A namespace to associate with the shapefiles

Sub-tags

None

Content

None

Examples

```
<spatial:spatialEngine id="spatialengine.shapefile">
  <dbtype>shapefiledir</dbtype>
  <url><![CDATA[file:/home/sforbes/data/spatial/demo]]><
/urll>
  <memorymapped>>true</memorymapped>
  <createspatialindex>>true</createspatialindex>
</spatial:spatialEngine>
```

```
<spatial:spatialEngine id="spatialengine.shapefile">
  <dbtype>shapefiledir</dbtype>
  <url><![CDATA[file:G:\GIS\DATA\SHAPEFILES]]></url>
  <memorymapped>>true</memorymapped>
  <createspatialindex>>true</createspatialindex>
</spatial:spatialEngine>
```

Spatial Engine SQLServer

A spatial engine provides access to a service that can perform spatial operations, one of the available spatial engines is for SQLServer.

Weave does not currently support the SQL Server `geography` column type.

Namespace

urn:com.cohga.server.spatial.geotools#1.0

Tags

spatialengine

Properties

Name	Type	Req uired	Description
id	string	yes	Unique identifier
dbtype	'sqls erver'	yes	The name of the driver to use, in this case sqlserver
host	string	yes	The hostname or ip address of the SQL Server server
port	num ber	yes	The port number that the SQL Server instance is running on
database	string	no	The database to be connected to
instance	string	no	The SQLServer instance to be connected to
schema	string	no	The schema used when connecting to the database
user	string	yes	The userid used when connecting to the database
passwd	string	yes	The password used when connecting to the database
namespace	string	no	The namespace prefix

minconnections	integer	no	The minimum number of pooled connection. Default 1
maxconnections	integer	no	The maximum number of open connections. Default 10
validateconnections	boolean	no	Check connection is alive before using it. Default false
fetchsize	integer	no	The number of records read with each interaction with the dbms. Default 1000
connectiontimeout	integer	no	The number of seconds the connection pool will wait before timing out attempting to get a new connection. Default 20 (seconds)
primarykeymetadatatable	string	no	The optional table containing primary key structure and sequence associations. Can be expressed as 'schema.name' or just 'schema' (if the table name is 'GT_PK_METADATA'). Default none
maxopenpreparedstatements	integer	no	Maximum number of prepared statements kept open and cached for each connection in the pool. Set to 0 to have unbounded caching, to -1 to disable caching. Default 50
geometrymetadatatable	string	no	The options table containing geometry column metadata information.
usenativegeometryserialization	boolean	no	Use native SQL Server serialization, or WKB serialization. Default is false.
forcespatialindexusageviahints	boolean	no	When enabled, spatial filters will be accompanied by a WITH INDEX sql hint forcing the usage of the spatial index. Default is false.

Sub-tags

None

Content

None

Examples

Basic SQL Server connection

```
<spatial:spatialengine id="sqlserver">
  <dbtype>sqlserver</dbtype>
  <host>sqlhost</host>
  <port>1434</port>
  <user>gis</user>
  <passwd>hak0rz</passwd>
  <schema>GIS</schema>
  <minconnection>2</minconnections>
  <validateconnections>>true</validateconnections>
</spatial:spatialengine>
```

Row Identifiers

Weave needs to be able to uniquely identify each row in a particular table, normally it would do this using the primary key of the table, but sometimes a primary key has not been specified so this information must be provided another way.

This can be done by creating a primary key metadata table and populating it with a row for each table that will be exposed to Weave and then setting the `primarykeymetadatatable` attribute for the spatial engine.

SQL create statement for gt_pk_metadata table

```
CREATE TABLE GT_PK_METADATA (
  TABLE_SCHEMA varchar(255) NOT NULL,
  TABLE_NAME varchar(255) NOT NULL,
  PK_COLUMN varchar(255) NOT NULL,
  PK_COLUMN_IDX int NOT NULL,
  PK_POLICY varchar(255) NOT NULL,
  PK_SEQUENCE varchar(255),
  UNIQUE(TABLE_SCHEMA, TABLE_NAME, PK_COLUMN),
  CHECK(PK_POLICY IN ('ASSIGNED', 'SEQUENCE', 'AUTOGENERATED'))
)
```

Primary Key Metadata

If the primary key for a table can not be determined by Weave directly from the database then the information can be provided by creating a table or directly embedding the information in the spatial engine configuration.

Column	Description
table_schema	Name of the database schema in which the table is located.
table_name	Name of the table to be published
pk_column	Name of a column used to form the feature IDs
pk_column_idx	Index of the column in a multi-column key. In case multi column keys are needed multiple records with the same table schema and table name will be used.
pk_policy	The new value generation policy, used in case a new feature needs to be added in the table ('assigned', 'sequence' or 'autogenerated'). Where: 'sequence' means that the value for the column is generated using a database sequence, and the 'pk_sequence' value must be set. 'autogenerated' means that the value for the column is generated by the database using another method. 'assigned' means that the value for the column is determined by the current maximum value +1, if the column is an integral type, or as a random string if the column is textual.
pk_sequence	The name of the database sequence to be used when generating a new value for the pk_column. The pk_sequence value only need to be set if the pk_policy is 'sequence'.

Using primary key metadata configuration

You can also specify the above information directly in the spatial engine configuration.

Name	Description
name	The table name
schema	The optional table schema
column	The column in the the table that contains the unique id
policy	How the key is generated
sequence	The name of the database sequence to use if the policy is 'sequence'

```

<spatial:spatialengine id="sqlserver">
  <dbtype>sqlserver</dbtype>
  <host>sqlhost</host>
  <port>1434</port>
  <user>gis</user>
  <passwd>hak0rz</passwd>
  <schema>GIS</schema>
  <minconnection>2</minconnections>
  <validateconnections>true</validateconnections>
  <primarykeymetadata>
    <table schema="dbo" name="park" column="gid" policy="
autogenerated" />
    <table schema="dbo" name="property" column="park_id" type="
autogenerated" />
    <table schema="dbo" name="road" column="objectid" type="
autogenerated" />
  </primarykeymetadata>
</spatial:spatialengine>

```

Using the geometry metadata table

The SQL server data store can determine the geometry type and native SRID of a particular column only by data inspection, by looking at the first row in the table. Of course this is error prone, and works only if there is data in the table. The administrator can address the above issue by manually creating a geometry metadata table describing each geometry column. Its presence is indicated via the SQL Server datastore connection parameter named *geometrymetadatatable* (which may be a simple table name or a schema-qualified one). The table has the following structure (the table name is flexible, just specify the one chosen in the data store connection parameter):

Creating geometry metadata table

```

CREATE TABLE GT_GEOMETRY_METADATA (
  F_TABLE_SCHEMA VARCHAR(30) NOT NULL,
  F_TABLE_NAME VARCHAR(30) NOT NULL,
  F_GEOMETRY_COLUMN VARCHAR(30) NOT NULL,
  COORD_DIMENSION INTEGER,
  SRID INTEGER NOT NULL,
  TYPE VARCHAR(30) NOT NULL,
  UNIQUE(F_TABLE_SCHEMA, F_TABLE_NAME, F_GEOMETRY_COLUMN),
  CHECK(TYPE IN ('POINT', 'LINESTRING', 'POLYGON', 'COLLECTION',
'MULTIPOINT', 'MULTILINESTRING', 'MULTIPOLYGON', 'GEOMETRY') )
)

```

When the table is present the store first searches it for information about each geometry column to be classified, and falls back on data inspection only if the table does not contain any information.

The SRID column should contain the EPSG code for the projection that the data for the layer is stored in.

Using geometry metadata configuration

As of Weave 2.5.16 it's possible to specify the information contained in the geometry metadata table directly in the spatial engine configuration.

Name	Description
name	The table name

schema	The optional table schema
type	The geometry type (point, linestring, polygon, multipoint, multilinestring, multipolygon)
srid	The geometry srid
dimension	The geometry dimension

Using geometry metadata configuration

```
<spatial:spatialengine id="sqlserver">
  <dbtype>sqlserver</dbtype>
  <host>sqlhost</host>
  <port>1434</port>
  <user>gis</user>
  <passwd>hak0rz</passwd>
  <schema>GIS</schema>
  <minconnection>2</minconnections>
  <validateconnections>>true</validateconnections>
  <geometrymetadata>
    <table schema="dbo" name="park" type="multipolygon" srid="28355"
dimension="2"/>
    <table schema="dbo" name="property" type="polygon" srid="28355"
dimension="2"/>
    <table schema="dbo" name="road" type="linestring" srid="28355"
dimension="2"/>
  </geometrymetadata>
</spatial:spatialengine>
```

Spatial Engine WFS

A spatial engine provides access to a service that can perform spatial operations, one of the available spatial engines is for a WFS service.

Namespace

urn:com.cohga.server.spatial.geotools#1.0

Tags

spatialengine

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
dbtype	'wfs'	yes	The name of the driver to use, in this case WFS
url	string	yes	The URL that points to the WFS servers get capabilities request
protocol	string	no	'post' to use POST requests, 'get' to use GET or 'auto' or not set for auto
encoding	string	no	The character encoding to use for the XML requests sent to the WFS server
username	string	no	The userid used when connecting to the WFS server
password	string	no	The password used when connecting to WFS server
timeout	integer	no	Time in ms to timeout a request, default 3000ms
buffersize	integer	no	The number of features to buffer, default 10

gzip	boolean	no	Indicates that gzip compression should be performed when communicating with the WFS server, default is 'true'
lenient	boolean	no	Indicates that the spatial engine should try and be forgiving of the data it receives from the WFS server, default is 'false'

Sub-tags

None

Content

None

Examples

```

<spatial:spatialengine id="spatialengine.wfs">
  <dbtype>wfs</dbtype>
  <url><![CDATA[http://vmrunout:8080/geoserver/ows]]><
/urll>
  <protocol>get</protocol>
  <username>username</username>
  <password>password</password>
  <timeout>60000</timeout>
  <bufferize>100000</bufferize>
  <gzip>>false</gzip>
  <lenient>>true</lenient>
</spatial:spatialengine>

```

Spatial Engine WFS-NG

This is an update of the existing WFS spatial engines that performs better than the previous version.

Available from Weave 2.5.16.

Namespace

urn:com.cohga.server.spatial.geotools#1.0

Tags

spatialengine

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
dbtype	'wfs-ng'	yes	The name of the driver to use, in this case WFS
url	string	yes	The URL that points to the WFS servers get capabilities request
protocol	string	no	'post' to use POST requests, 'get' to use GET or 'auto' or not set for auto
encoding	string	no	The character encoding to use for the XML requests sent to the WFS server
username	string	no	The userid used when connecting to the WFS server
password	string	no	The password used when connecting to WFS server
timeout	integer	no	Time in ms to timeout a request, default 3000ms
bufferize	integer	no	The number of features to buffer, default 10
gzip	boolean	no	Indicates that gzip compression should be performed when communicating with the WFS server, default is 'true'
lenient	boolean	no	Indicates that the spatial engine should try and be forgiving of the data it receives from the WFS server, default is 'false'

Sub-tags

None

Content

None

Examples

```

<spatial:spatialengine id="spatialengine.wfsng">
  <dbtype>wfs-ng</dbtype>
  <url><![CDATA[http://vrunout:8080/geoserver/ows]]><
/url>

  <protocol>get</protocol>
  <username>username</username>
  <password>password</password>
  <timeout>60000</timeout>
  <bufferize>100000</bufferize>
  <gzip>>false</gzip>
  <lenient>>true</lenient>
</spatial:spatialengine>
    
```

Spatial Engine Cache

This is a virtual spatial engine that provides caching for other spatial engines.

Available from Weave 2.6.4.

Namespace

urn:com.cohga.server.spatial.geotools#1.0

Tags

spatialengine

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
dbtype	'cache'	yes	The name of the driver to use, in this case "cache"
datasource	string	yes	The id of the spatial engine that will provide the data for this spatial engine
maxAgeSeconds	integer	no	How many seconds should the data in the cache be kept for before it is re-fetched from the source spatial engine. If this is not set then the data will be read once when first accessed and not updated after that.
autoRenew	boolean	no	Should the data be fetched periodically in the background? If this is set to false the underlying data will be re-fetched at the time it's access if it is older than maxAgeSeconds seconds old. If this isn't set or is set to true then the data will be periodically be checked if it is older than maxAgeSeconds seconds old and re-fetched and replace the existing cached data.
evictorPeriodicitySeconds	integer	no	The number of seconds to wait between background checks to see if data is exposed, default is 300 seconds.
evictorParallelism	integer	no	The number of expired layers to fetch at one from the server during a background load when the data has expired, default is 1.

Sub-tags

None

Content

None

Examples

```

<!-- create the source for the data to be cached
      in this case we're using an WFS service -->
<spatial:spatialengine id="spatialengine.wfsng.source">
  <dbtype>wfs-ng</dbtype>
  <url><![CDATA[http://vmrunout:8080/geoserver/ows]]></url>
  <protocol>get</protocol>
  <username>username</username>
  <password>password</password>
  <timeout>60000</timeout>
  <bufferize>100000</bufferize>
  <gzip>>false</gzip>
  <lenient>>true</lenient>
</spatial:spatialengine>

<!-- create the cache referring to the WFS spatial engine
      this is what you should refer in the rest of the config -->
<spatial:spatialengine id="spatialengine.wfsng">
  <dbtype>cache</dbtype>
  <datasource>spatialengine.wfsng.source</datasource>
  <maxAgeSeconds>3600</maxAgeSeconds>
</spatial:spatialengine>

```

Spatial Mapper

A spatial mapper links an Entity to a Spatial Engine. This linkage is not required and in fact, may not be possible if there is no spatial information associated with the entity.

If there is spatial information that relates to the entity and you want the users to be able to perform spatial operations (select by polygon, etc.) on the entity then a link should be set up.

The mapping is actually between the entity and an object that is managed by a spatial engine. The type of object will be different depending upon the spatial engine. For example, if the spatial engine is ArcSDE then the mapping will be between the entity and an ArcSDE Layer, for Oracle Spatial it will be between the entity and a spatially enabled table.

When adding a definition for a spatial mapping, the setup will be the same regardless of the spatial engine type but the "name" of the spatial object may be different. For example, a shapefile based spatial engine will refer to the spatial objects based on the filename which may be case sensitive depending upon the underlying operating system, but an Oracle Spatial engine will refer to a table name which may need to be fully qualified when a schema.

The spatial mapping also specified which column in the underlying table is used to uniquely identify an entity, this is set with the `key` attribute. Because the key used by Weave to identify an entity may be different from that used by the underlying spatial entity to identify a feature there has to be a mapping performed between the Weave keys and the spatial engine feature ids.

Generally, this mapping is performed once at system startup and doesn't change unless Weave notices a change in the count of the available rows in the underlying table (which is checked every five minutes). This is assuming that the underlying keys/feature ids don't change and only the addition/removal of a row in the table would have an effect on the key mapping.

However there may be times when this is not suitable, for example, if the table is very large it can take a long time to read all of the key mappings and they can take up a large amount of memory, or if the key mappings do actually change frequently, so this initial key mapping can be disabled and the mapping between the Weave keys and the spatial engine feature ids can be performed on-the-fly each time Weave needs to transform a key to a feature id, or a feature id to a key. Obviously, this will have an impact on performance, since it requires a trip to the spatial engine for each transform, rather than having it done once and the transform performed in memory.

The initial loading of the key/fid mapping can be disabled, for either all spatial mappings in a spatial mapper or for a single spatial mapping, by setting the `dynamic` flag in the configuration.

In addition, if dynamic mapping is turned on you can indicate that the values retrieved from the spatial engine for a mapping should be kept in memory to avoid the cost of querying the spatial engine next time the mapping is required, this is done by setting the `cache` flag in the

configuration. This can be useful if you want to avoid the longer startup time but the key/fid mappings are still fairly static (so it's safe to have them loaded once). This provides a middle ground, if suitable, between a static, loaded at startup, key/fid mapping, and a dynamic, performed each time, key/fid mapping.

Previously a single spatial mapper could be used with multiple spatial engines, by setting the spatial engine attribute for each mapping, but that is no longer possible.

If you have multiple spatial engines you need to have multiple spatial mappers configured, one for each spatial engine.

Setting the spatial engine at the mapping level is no longer supported, it must be set once at the top level for the spatial mapper.

Namespace

urn:com.cohga.server.spatial.mapper#1.0

Tags

mapper

Properties

Name	Type	Required	Description
id	string	yes	unique identifier
spatialEngine	ref urn:com.cohga.server.spatial.geotools#1.0:spatialEngine	yes	The identifier of the spatial engine that the mappings should link to
crs	string	no	The CRS to override for entities that don't provide an explicit value
filter	string	no	Provide a CQL formatted filter to apply when performing spatial operations with this mapper
dynamic	boolean	no	Should the key/fid transform be performed on each lookup
cache	boolean	no	Should dynamically looked up key/fid transforms be cached in memory

Sub-tags

Name	Type	Cardinality	Description
mapping	urn:com.cohga.server.spatial.mapper#1.0:mapping	1..n	A collection of mappings that link entities to spatial items

Content

None

Notes

- The CRS property does not perform any transform, it just tells the server how the actual data is stored in cases where the underlying spatial engine does not supply this information or supplied an incorrect value.

mapping

Properties

Name	Type	Required	Description
entity	ref urn:com.cohga.server.entity#1.0:entity	yes	The identifier of the entity that this mapping applies to
table	string	yes	The name of the spatial item within the spatial engine that this mapping applies to
key	string	no	The attribute of the spatial item that uniquely identifies an entity
crs	string	no	Provide a Coordinate Reference System if the underlying spatial engine doesn't provide the correct information
filter	string	no	Provide a CQL formatted filter to apply when performing spatial operations with this mapping

dynamic	boolean	no	Should the key/fid transform be performed on each lookup
cache	boolean	no	Should dynamically looked up key/fid transforms be cached in memory
unique	boolean	no	Tells Weave that the key values will be unique, so that it doesn't have to check for itself. Weave can optimise the handling of spatial selections if the keys are unique, if you know the keys are unique you can set this to true and Weave will not need to pre-check for itself.

Sub-tags

Name	Type	Cardinality	Description
mapping	urn:com.cohga.server.spatial.mapper#1.0:mapping	0..n	If more than one underlying feature maps to the entity then multiple sub-mapping tags can be used to specify additional mappings

Content

None

Notes

- The CRS property does not perform any transformation, it just tells the server how the actual data is stored in cases where the underlying spatial engine does not supply this information or supplied an incorrect value.
- Only 1 level of nesting is supported for sub-mappings.
- Only table and key are allowed in sub-mapping tags.
- Only table is required, key can be inferred from the parent mapping.

Examples

Basic spatial mapping setup

```

<mapper:mapper id="mapper.arcsde">
  <spatialEngine>spatialengine.arcsde</spatialEngine>
  <mapping>
    <entity>property</entity>
    <table>PROPERTY</table>
    <key>PID</key>
  </mapping>
  <mapping>
    <entity>roads</entity>
    <key>RD_NAME</key>
    <mapping>
      <table>MAJOR_ROADS</table>
    </mapping>
    <mapping>
      <table>MINOR_ROADS</table>
    </mapping>
  </mapping>
  <mapping>
    <entity>suburbs</entity>
    <table>SUBURBS</table>
    <key>LOC_NAME</key>
  </mapping>
  <mapping>
    <entity>council_buildings</entity>
    <table>BUILDINGS</table>
  </mapping>
</mapping>
<mapper:mapper id="mapper.oracle">
  <spatialEngine>spatialengine.oracle</spatialEngine>
  <mapping>
    <entity>drainage</entity>
    <table>DRAINAGE</table>
  </mapping>
</mapper:mapper>

```

If the data in the underlying spatial engine can change over time then the dynamic and cache flags should be set to disable caching and enable dynamic lookup of the id's. This is especially relevant if the id's change when records are added or deleted for example so should always be performed when the entity is editable.

Spatial mapping for dynamic data

```
<mapper:mapper id="mapper.edit">
  <spatialEngine>spatialengine.arcsde</spatialEngine>
  <mapping>
    <entity>grafitti</entity>
    <table>GRAFITTI</table>
    <key>PID</key>
    <dynamic>true</dynamic>
    <cache>false</cache>
  </mapping>
</mapper:mapper>
```

Mapping with a filter

```
<mapper:mapper id="mapper.main">
  <spatialEngine>spatialengine.arcsde</spatialEngine>
  <mapping>
    <entity>property</entity>
    <table>PROPERTY</table>
    <key>PID</key>
    <filter><![CDATA[status = 'C']]></filter>
  </mapping>
</mapper:mapper>
```

Filters

Simple filters

```
<filter>lga_code == 0</filter>

<filter><![CDATA[lga_code <> 0]]></filter>

<filter><![CDATA[planno = 'LP129263']]></filter>

<filter><![CDATA[planno LIKE 'LP%']]></filter>

<filter><![CDATA[planno NOT LIKE 'LP%']]></filter>

<filter>further_de IS NULL</filter>

<filter>further_de IS NOT NULL</filter>
```

Geometry filters

```
<filter><![CDATA[CROSS(the_geom, LINESTRING(332312 5815856, 333688 5823904))]]></filter>
```

```
<filter><![CDATA[CROSS(the_geom, LINESTRING(332312 5815856, 333688 5823904)) OR CROSS(the_geom, LINESTRING(331312 5814856, 332688 5822904))]]></filter>
```

```
<filter><![CDATA[BBOX(the_geom, 332312,5815856,333688,5816856)]]></filter>
```

Other Filters

Geometry filter, only works with units stored in the database

```
<filter><![CDATA[DWITHIN(the_geom, POINT(331899 5818553), 500, meters)]]></mapper:filter>
```

Geometry filter, only works with units stored in the DB

```
<filter><![CDATA[BEYOND(the_geom, POINT(331899 5818553), 500, meters)]]></filter>
```

Geometry filter with collection. Does not work in PostGis as it does not like geometry collections, not tested in other spatial engines

```
<filter><![CDATA[INTERSECT(the_geom, GEOMETRYCOLLECTION(LINESTRING(332312 5815856, 333688 5823904), LINESTRING(333312 5815856, 334688 5823904)))]></filter>
```

Map Engine

At least one Map Engine must be setup for the server to be able to generate and display map images for the client.

The map engine configuration is different depending upon which map engine you are connecting to.

Weave currently supports the following map engines:

- [WMS](#)
- [ArcIMS](#)
- [ArcGIS Server via Web Service](#)
- [Tiled map engine](#)
- [Selection map engine](#)
- [Weave map engine](#)
- [Dynamic map engine](#)

Refining the layer a map engine exposes

It's possible that you want to make use of a map engine that contains dozens or even hundreds of layers where you actually only want to expose only a handful of those layers to Weave, for example if you have a WMS map service that's provided by a government department and covers the entire country but you only want to make use of the layers that cover your particular state (assuming the WMS service exposes separate layers for separate states), or if you have a map service that exposes both vector and raster based layers but you're rather split them up in Weave so you have two separate Weave map engine, one that's just vector layers and one that's just raster.

You can do this one of two ways in Weave, depending on it being easier to remove the layers you don't want or if it's easier to include just the layers that you do want, but you should use only one of those methods and not both. If you want to only list the layers to include in the final map engine you can add a `layers` tag to the map engine that has one or more individual `layer` tags within for each layer that you want to include, alternatively if you want to remove layers you'd add a `remove` tag between the `layers` and `layer` tags, e.g.

Include only two layers from a map engine

```
<wms:mapengine id="test">
  ...
  <layers>
    <layer>sa_temp_2012</layer>
    <layer>sa_temp_2016</layer>
  </layers>
</wms:mapengine>
```

Remove four layers from a map engine

```
<wms:mapengine id="test">
  ...
  <layers>
    <remove>
      <layer>nt_temp_2012</layer>
      <layer>wa_temp_2012</layer>
      <layer>nt_temp_2016</layer>
      <layer>wa_temp_2016</layer>
    </remove>
  </layers>
</wms:mapengine>
```

Note that as of Weave 2.5.20 it's possible to specify a wildcard using `*` for the layer id's, e.g.

```
<wms:mapengine id="test">
  ...
  <layers>
    <remove>
      <layer>nt_temp_*</layer>
      <layer>wa_temp_*</layer>
    </remove>
  </layers>
</wms:mapengine>
```

Attribution (Watermark)

To display a text overlay on the map you can add an attribution (e.g. Watermark) control to the map then include an attribution property for one of your client map engines.

Adding attribution to the map view

```
<client:config id="...">
  ...
  <view id="com.cohga.html.client.map.mapView">
    <control id="com.cohga.client.mapctrl.attribution"/>
    ...
    <mapengine id="raster">
      ....
    </mapengine>

    <mapengine id="vector">
      ....
      <options>

        <attribution>Copyright(c) 2010</attribution>
        ....
      </options>
    </mapengine>
  </view>
</client:config>
```

Map Engine WMS

The minimum setup required for a WMS map engine is the capabilities URL for the service.

Optionally you can specify the format of the image generated (gif, jpeg, png, etc.) if the background of the image should be transparent or not, and if not, the colour of the background.

You are also able to change the colours or styling Weave uses to colour selections.

A WMS Tester tool is available from [here](#). It can help resolve issues with WMS servers.

Namespace

urn:com.cohga.server.map.wms#1.0

Tags

mapengine

Properties

Name	Type	Required	Default	Description
id	string	yes		Unique identifier
url	string	yes		The capabilities URL for the WMS service. You should not include the SERVICE, REQUEST or VERSION parameters in the URL, just use the base URL, these parameters will be added automatically based on the type of request that is being made. Note: WMS version 1.3.0 may be forced on by including the URL VERSION parameter, however, this is not recommended and Weave will use WMS version 1.1.1 by default.
format	string	no	first format reported by map engine	Default format that the map engine will request from the WMS server if one is not specified.

transparent	boolean	no	false	Determines if the areas of the image not covered by spatial data are rendered as transparent or as a solid colour (as set by the background property). Enabling transparency requires an image format that supports it (jpeg and png24 do not).
background	integer	no	white	The RGB value for the image background if transparency is not enabled
username	string	no		A username for connecting to the server.
password	string	no		A password for connecting to the server. This should be encrypted using 'encrypt' at the osgi prompt.
qgis	boolean	no		Force the map engine to be treated as though it were QGIS. Weave will try to determine if the underlying WMS service is provided by QGIS and perform additional optimisations and expose additional capabilities if it is. Sometimes Weave might not be able to determine this correctly, in which case you can set this property to true in the configuration and the map engine will be treated as though it were being served from QGIS. If Weave incorrectly determines that the WMS service is QGIS when it is not, then you can set this property to <code>false</code> to disable the additional functionality. (Note: Weave does not automatically identify QGIS 3.4 hence the need to set <code>qgis=true</code> .)
geoserver	boolean	no		Force the map engine to be treated as though it were GeoServer. Weave will try and determine if the underlying WMS service is provided by GeoServer and perform additional optimisations and expose additional capabilities if it is. Sometimes Weave might not be able to determine this correctly, in which case you can set this property to true in the configuration and the map engine will be treated as though it were being served from GeoServer. If Weave incorrectly determines that the WMS service is GeoServer when it is not, then you can set this property to <code>false</code> to disable the additional functionality.
arcgis	boolean	no		Force the map engine to be treated as though it were ArcGIS. Weave will try and determine if the underlying WMS service is provided by ArcGIS and perform additional optimisations and expose additional capabilities if it is. Sometimes Weave might not be able to determine this correctly, in which case you can set this property to true in the configuration and the map engine will be treated as though it were being served from ArcGIS. If Weave incorrectly determines that the WMS service is ArcGIS when it is not, then you can set this property to <code>false</code> to disable the additional functionality.
arcims	boolean	no		For those still using ArcIMS I'm sure you can figure out what this property does by reading the three properties before this one
disableTests	boolean	no	false	Set to true to disable the background checks to validate WMS connections
testInterval	integer	no	60,000	Milliseconds between tests when the WMS connection is working (minimum 15,000)
testIntervalWhileFailing	integer	no	15,000	Milliseconds between tests when the WMS connection is failing (minimum 5,000)
testFailingDuration	integer	no	300,000	How long before a failing WMS connection is determined to have failed (minimum 60,000)
testIntervalWhenFailed	integer	no	600,000	Milliseconds between tests when the WMS connection has failed (minimum 120,000)
cql	boolean	no	false	Set to true if the map engine supports filtering using CQL. If the map engine is one that is not directly supported by Weave but does support using CQL to filter layers then setting this to true will enable the support for layer filtering using CQL.
cqlparameter	string	no	cql_filter	The name of the parameter to add to the request URL if CQL filtering support was enabled by setting cql to true
cqlseparator	string	no	;	The filter separator to add to the request if CQL filtering support was enabled by setting cql to true

Sub-tags

Name	Type	Cardinality	Description
layers	#layers	0..1	Refine what layers are included in the map description.

sort	#sort	0..1	
styles	#style	0..1	
crs	#crs	0..n	<p>Refine what CRSs the map engine lists as being supported, as opposed to including all of them (as there might be thousands of CRSs that the WMS server says that it supports).</p> <p>Without any <code>crs</code> tags a map engine will report all the projections the WMS server says it supports. By adding one or more <code>crs</code> tag you can refine this list so that the map engine only includes those projections that you're actually interested in using (which helps to improve performance).</p>
tokens	#tokens	0..1	Additional tokens to add to the request. The tokens available are dependent on the WMS server implementation and are generally used to support implementation specific extensions to the WMS specification.

layers

Sub-tags

Name	Type	Cardinality
layer	#layer	0..n
remove	#remove	0..1

You should include one `remove` tag or one or more `layer` tags, not both.

remove

Sub-tags

Name	Type	Cardinality
layer	#layer	1..n

sort

Properties

Name	Type	Required	Description
method	"firsttolast" or "lasttofirst"	no	Default is <code>firsttolast</code> . It specifies if the layer list specifies layers to be sorted from top to bottom or from bottom to top.

Sub-tags

Name	Type	Cardinality
layer	#layer	2..n

layer

Content

The id of the layer

You do not need to include all layers in the layer list, only those you want moved, which is where the `method` property comes in. If you want to list a handful of layers to draw on top then just list them and use `firsttolast` (or don't set the `method`). If you only want to list a handful of layers to move to the bottom on the map then just list them and use `lasttofirst`.

styles

Properties

None.

Sub-tags

Name	Type	Cardinality
style	#style	1..n

style

Properties

Name	Type	Required	Description
layer	string	yes	The id of the layer, from the map engine, to override the style for
name	string	yes	The name of the style to use

crs

Properties

Name	Type	Required	Description
force	boolean	no	Ensure the CRS is added to the list of available CRSs even if the underlying server doesn't indicate that it supports the EPSG code

Content

The EPSG code of the coordinate reference system to add

tokens

Tokens provide a way for the administrator to include additional parameters in the URL sent to the WMS server

Properties

Name	Type	Required	Default	Description
tokenJoinDelimiter	string	no	;	value to use to join multiple tokens with the same name, since 2.6.7

Sub-tags

Name	Type	Cardinality
token	#token	1..n

token

Properties

Name	Type	Required	Default	Description
name	string	yes		The name of the token
value	string	yes		The value of the token
legendOnly	boolean	no	false if layer is empty true if layer is not empty	Should this token only be applied to requests that are generating a legend
layer	string	no		Comma separated list of layer id's for which this token should be included, can be used to further restrict when the token is sent, and allows for different tokens to be sent for different layers. Only applies if legendOnly is true
prefix	string	no		Value to be prepended to the token value, since 2.6.7
infix	string	no	or	Value to be used to join multiple user attributes, since 2.6.7
suffix	string	no		Value to be appended to the token value, since 2.6.7
default	string	no		Value to be used if user attribute substitution does not provide a value, since 2.6.7

Examples

Filtering layers to only include specific layers

```
<wms:mapengine id="test">
  <url>http://server/wms/test</url>
  <layers>
    <layer>point1</layer>
    <layer>point2</layer>
    <layer>line1</layer>
    <layer>line2</layer>
    <layer>polygon1</layer>
    <layer>polygon2</layer>
  </layers>
</wms:mapengine>
```

Filtering layers to remove specific layers

```
<wms:mapengine id="test">
  <url>http://server/wms/test</url>
  <layers>
    <remove>
      <layer>point2</layer>
      <layer>line2</layer>
      <layer>polygon2</layer>
    </remove>
  </layers>
</wms:mapengine>
```

Sorting layers

```
<wms:mapengine id="test">
  <url>http://server/wms/test</url>
  <sort>
    <layer>point1</layer>
    <layer>point2</layer>
    <layer>line1</layer>
    <layer>line2</layer>
    <layer>polygon1</layer>
    <layer>polygon2</layer>
  </sort>
</wms:mapengine>
```

Sorting layers by moving one layer to the bottom

```
<wms:mapengine id="test">
  <url>http://server/wms/test</url>
  <sort method='lasttofirst'>
    <layer>photo</layer>
  </sort>
</wms:mapengine>
```

Applying styles

```
<wms:mapengine id="test">
  <url>http://server/wms/test</url>
  <styles>
    <style layer="property" name="bw_property"/>
    <style layer="road" name="bw_road"/>
  </styles>
</wms:mapengine>
```

Adding authentication

```
<wms:mapengine id="test">
  <url>http://username:password@server/wms/test</url>
</wms:mapengine>
```

Refine the list of supported CRS's for the map engine

```
<wms:mapengine id="test">
  <url>http://server/wms/test</url>
  <crs>EPSG:4326</crs>
  <crs>EPSG:28355</crs>
  <crs>EPSG:3857</crs>
</wms:mapengine>
```

Adding tokens

```
<wms:mapengine id="test">
  <url>http://server/wms/test</url>
  <tokens>
    <token name="VIEWPARAMS" value="{user.filter.status}" prefix="
STATUS:" infix=","/>
    <token name="VIEWPARAMS" value="USER:{user.id}"/>
    <token name="BACKFILL" value="true" legendOnly="true" layer="ward,
busroutes"/> <!-- BACKFILL will be true for ward and busroutes -->
    <token name="BACKFILL" value="false" legendOnly="true"/> <!--
BACKFILL will be false for any other layers -->
  </tokens>
</wms:mapengine>
```

As of version 1.3.7 of the WMS bundle, the password can be encrypted using the OSGi encrypt command.

WMS 1.3.0

Don't "upgrade" to WMS 1.3.0 unless you really have to.

Version 2.15.16 of the `com.cohga.server.map.wms` bundle (released as part of Weave 2.5.21) improves support for WMS 1.3.0 by supporting map services that require their axis order to be swapped.

Weave disables the use of WMS 1.3.0 when communicating with a WMS server unless it's specifically asked to, which can be done by including a `VERSION` parameters in the map engine `url` attribute, e.g.

Enabling support for WMS 1.3.0

```
<wms:mapengine id="test">
  <url>http://server/wms/test?VERSION=1.3.0</url>
</wms:mapengine>
```

This is typically all you need to do to enable support for WMS 1.3.0. It is possible that you may be using a coordinate reference system that requires requests sent to the server to reverse the order of x, y coordinates. It is updated 2.14.16 of the Weave WMS map engine bundle that allows you to tell Weave to change the order of the coordinates when sending requests to the server. This is done by setting `swapAxis` to `true` in the map engine config, e.g.

Swapping coordinate order for a WMS map engine

```
<wms:mapengine id="test">
  <url>http://server/wms/test?VERSION=1.3.0</url>
  <swapAxis>true</swapAxis>
</wms:mapengine>
```

Once this is done Weave will send the bounding box parameter in `miny, minx, maxy, maxx` order, rather than `minx, miny, maxx, maxy`.

Note that when `swapAxis` is `true` swapping the axis order is done for all map requests sent to the server, so if you're trying to use a map service using a coordinate reference system that requires swapping of the coordinates, for example, EPSG:3006, along with a coordinate

reference system that doesn't, for example, EPSG:3857, then you will need to create two map engines, one with `swapAxis` set to true and one without, and use the appropriate map engine for the different clients (since it's the `crs` attribute in the clients' map view that determines what coordinate reference system will be used, and it's the coordinate reference system in WMS 1.3.0 that determines the axis order).

Further to this, the change between 2.14.16 and 2.15.16 adds the ability for Weave to determine automatically if it should swap the axis or not when working with WMS 1.3, removing the need to use the `swapAxis` setting at all and the need to have two map engines configured if you need to use both swapped and un-swapped axis (you still need to include `VERSION=1.3.0` in the URL, to enable WMS 1.3 support, but that too may be removed eventually).

To enable the automatic determination of whether to swap the axis or not, you must update your startup setting to remove the `org.geotools.referencing.forceXY` setting. This is done by editing `startup.cmd` (or `startup.sh`) or editing `wrapper.conf` and removing the option that sets this property to `true`, otherwise, Weave will always think that the axis order is x, y regardless of the projection used.

Note that setting `swapAxis` to `false` will force x, y ordering of the axis regardless of what the projection information says.

Metromap

When using Metromap as a WMS map engine, there are two different methods available to provide the API key in the URL you have to configure for the map engine. One format sets the API key as a parameter and the other method embeds the API key within the URL path, [see this page for more info](#).

The format with the API key set as a query parameter will *not* work with Weave, you have to use the format where the API key is provided as part of the URL path as shown below.

Required format for Metromap URL

```
https://api.metromap.com.au/api-link/{YOUR_API_KEY}/wms
```

Map Engine ArcIMS

The ArcIMS map engine connector allow for a direct connection to an ArcIMS map service.

The ArcIMS map engine also allows the map service used to generate the map image to be switched depending upon a list of layers that the user has checked, this allows for switching between a PNG and JPEG image if aerial photography is enabled for example.

Namespace

urn:com.cohga.server.map.arcims#1.0

Tags

mapengine

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
service	string	yes	The name of the map service to use
host	string	no	The hostname of the ArcIMS server, defaults to localhost
port	integer	no	The port that ArcIMS is listening on, defaults to 80
alias	string	no	The start portion of the connection url, defaults to /servlet/com.esri.esrimap.Esrimap?client=weave
custom	string	no	Any custom parameter that should be added to the url
format	string	no	Informs the Weave server what format the map service has been created
username	string	no	The username, if required

password	string	no	The password, if required
crs	string	no	Override, or set, the CRS used for this mapengine. Use this if the service doesn't have FILTERCOORDSYS and FEATURECOORDSYS set, it's not required if they're set correctly
altservice	string	no	The name of the alternative map service to use if any of <code>altlayers</code> are visible
altlayers	string	no	A comma separated list of layer ids that if visible will result in <code>altservice</code> being used to generate the map rather than <code>service</code>
altformat	string	no	The format of the alternate map service
dpi	integer	no	Tell the server if the ArcIMS DPI value has been changed from the default

Notes

If an alternate map service is used the underlying AXL file must have the same layers in the same order as the original map services AXL file. Ideally they should be made from the same AXL file, but they can be different, for example if you want different renderers for the two services.

The possible values for the format are:

- jpg
- jpeg
- gif
- png

Default is png

Examples

```
<arcims:mapengine id="mapengine.arcims">
  <host>arcimssrv.example.com</host>
  <service>MyMapService</service>
  <format>png8</format>
  <altservice>MyMapService_JPG</altservice>
  <altlayers>aerial_2000,aerial_2006</altlayers>
  <altformat>jpg</altformat>
</arcims:mapengine>
```

Map Engine ArcGIS (SOAP)

The ArcGIS Server Web Service map engine component provides a means for the Weave server to generate maps using ArcGIS Server via its web service API. This has the advantage over the direct ArcGIS Server map engine because it does not require any additional components to be installed on the server while still providing the same level of functionality (hence the direct connect component being made obsolete).

For information about using ArcGIS Server map caches with Weave see the page [ArcGIS Server map caches](#).

Namespace

urn:com.cohga.server.map.arcgis.ws#1.0

Tags

mapengine

Properties

N	T	R	D	Description
a	y	e	e	
m	p	e	f	
	q	u	a	
		u	u	
		ir		
		ed	lt	

id	string		Unique identifier for this map engine, will be used by other items to refer to this map engine
url	string		The URL used to connect to the ArcGIS server, e.g. http://vmbreakout:8399/arcgis/services/Topo250/MapServer
map	string	the identifier map	The name of the map within the map service to connect to if there is more than one. This is not the map service name, which is included the URL, but the name of a map within the .mxd file itself. Generally this does not need to be set at all.
map	boolean	false	Should a tile cache be used (if it's available). If this is false or not set then any tile cache generated for the map service WON'T be used
crs	string		Override the coordinate reference system supplied by ArcGIS
user	string		A username to connect to the server as
password	string		A password to connect to the server as, should be encrypted using 'encrypt' at the OSGi prompt
use	boolean	false	Use the layer name to refer to individual layers, especially useful when building a ToC model since the names don't change as often as the id. You should ensure that the layer name is unique if you set this to true. If not set or set to false then the layer identifiers for each layer (which are used in ToC models, amongst other things) will be taken from the order that the layer appears within the .mxd file, which can change if layers are added/removed from the .mxd file. If set to true then the label that appears in the .mxd file will be used as the identifier instead
transparent	boolean	false	If true then ArcGIS is asked to generate an image that is transparent where no map data is drawn
direct	boolean	false	If true then Weave will send the URL's returned from ArcGIS directly to the client rather than proxying them on behalf of ArcGIS. This will help performance if ArcGIS is accessed internally and the clients can access the URL's the ArcGIS returns.
referrer	string		Referer header value when sending HTTP requests to ArcGIS server (since com.cohga.server.map.arcgis.ws bundle version 2.38.48) This is useful if ArcGIS server filters all requests by header value by comparing the referer domain against a list of white-listed domains.
cache	boolean	false	Should Weave cache the tiles it retrieves from ArcGIS. This only applies to ArcGIS map engines using a map cache and can be enabled if the back-end service is slow to return the tiles, for example, if it's not hosted internally.
cache	integer	-1	How many minutes should Weave keep the cached ArcGIS tiles for before they're considered stale and should be re-fetched? This only applies to ArcGIS map engines using a map cache and only if <code>cachetiles</code> is set to true. -1 means the tiles are kept forever.

ignore	boolean	true	Should Weave ignore the groups when working with ArcGIS map documents.
			When Weave works with map layers it only deals with the layers themselves, groups don't come into play until you start creating a ToC model, and even then the grouping in ToC models is not used when referring to map layers (they're only used internally on the client to turn on/off a collection of individual layers). Normally this works fine for ArcGIS as Weave will ensure that if a layer is "visible", then all of its parent groups (on the ArcGIS side) will also be turned on. This ensures that the layer is actually visible (as a layer won't be displayed if it's visible but one or more of its parents aren't). This way you can setup your AGS map document however you want (with or without groups) and have the Weave ToC model be the single source of group related information for the ToC panel, and then Weave will ensure that the map layers that need to be drawn are drawn.
			However, sometimes it's desirable to have Weave recognize the groups (that's the ArcGIS map document groups, not ToC model groups) and have them treated as layers. By setting <code>ignoregroups</code> to <code>false</code> Weave will include entries in the map engine description for the groups as well as for the layers. In this case the ToC model and the ArcGIS map document groups will, in combination, be responsible for determining whether a layer is visible or not.

Sub-tags

Name	Type	Cardinality
pool	urn:com.cohga.server.pool#1.0:pool	0..1
layers	urn:layers	0..1
legend	urn:legend	0..1

Content

None

layers

Properties

None

Sub-tags

Name	Type	Cardinality
remove	urn:remove	0..1
layer	urn:layer	0..n

Content

None

remove

Properties

None

Sub-tags

Name	Type	Cardinality
layer	urn:layer	0..n

Content

None

layer

Properties

Name	Type	Required	Description
id	number	no	The id of the layer to match
name	string	no	The name of the layer to match

Sub-tags

None

Content

The id or name, depending upon the `username` property, of the layer to match

Notes

- The `layer` tag support two formats, one using the `id` and/or `name` attributes and the other using the tags `content`
- Either one or both of the `id` and/or `name` attributes must be set OR tag `content` must be provided
- If the content of the `layer` tag is used then the value of the `username` property determines if the content should match the layer id or the layer name
- Both the `id` and `name` attributes can be set, but they must match both values for the layer that you're trying to filter or they won't match the layer
- The `pool` tag when used with ArcGIS Server also support a `maxAgeMillis` property which specifies how long the connection to ArcGIS Server should be kept for, this should be less that the value set for "The maximum time a client can use a service" in ArcGIS, note however that this check only occurs during connection validation, so `testOnBorrow`, `testOnReturn` and/or `testWhileIdle` must be set for the pool. [ArcGIS Server Settings - Pooling and Processes](#)

legend

Alter the display of the legend

Properties

None

Sub-tags

Name	Type	Cardinality
alias	urn:aliases	0..1

Content

None

aliases

List the new labels to be displayed for a layer

Properties

None

Sub-tags

Name	Type	Cardinality
alias	urn:alias	0..n

Content

None

alias

A single label change for a layer

Properties

Name	Type	Required	Description
id	string	yes	The id of the layer to change
label	string	yes	The label to display

Sub-tags

None

Content

None

Examples

Simple ArcGIS definition

```
<arcgisws:mapengine id="arcgis.topo">  
  <url>http://vmbreakout:8399/arcgis/services/Topo250/MapServer<  
/url>  
</arcgisws:mapengine>
```

ArcGIS definition overriding CRS returned, asking for a transparent image and using the layer names, rather than order, as the unique identifier for a layer

```
<arcgisws:mapengine id="arcgis.roads">  
  <url>http://vmbreakout:8399/arcgis/services/Roads/MapServer<  
/url>  
  <crs>EPSG:20255</crs>  
  <transparent>true</transparent>  
  <username>true</username>  
</arcgisws:mapengine>
```

ArcGIS definition using a tile cache and connection pooling

```

<arcgisws:mapengine id="arcgis.base">
  <url>http://vmbreakout:8399/arcgis/services/Base/MapServer<
/url>
  <map>Layers</map>
  <crs>EPSG:20255</crs>
  <transparent>true</transparent>
  <username>true</username>
  <mapcache>true</mapcache>
  <pool:pool>
    <maxActive>16</maxActive>
    <minIdle>2</minIdle> <!-- always keep at least 2
connection open -->
    <maxIdle>8</maxIdle>
    <testOnBorrow>true</testOnBorrow>
    <testWhileIdle>true</testWhileIdle> <!-- enable
background tests so that "old" (i.e. maxAgeMillis) connections can be
cleaned up in the background -->
    <timeBetweenEvictionRunsMillis>30000<
/timeBetweenEvictionRunsMillis>
    <numTestsPerEvictionRun>16</numTestsPerEvictionRun> <!
-- test all the connection on each run -->
    <minEvictableIdleTimeMillis>300000<
/minEvictableIdleTimeMillis> <!-- close a connection that's been idle
for five minutes -->
    <whenExhaustedAction>block</whenExhaustedAction> <!--
ensure we don't create more than maxActive connections -->
    <maxAgeMillis>565000</maxAgeMillis> <!-- close
anything that's been open for a the max allowed time from AGS
(assuming 600 seconds here) minus a little more than the
timeBetweenEvictionRunsMillis value to ensure it's removed before the
AGS setting-->
  </pool:pool>
</arcgisws:mapengine>

```

ArcGIS definition providing only a single specific layer based on the layer id

```

<arcgisws:mapengine id="arcgis.roads">
  <url>http://vmbreakout:8399/arcgis/services/Base/MapServer<
/url>
  <layers>
    <layer id="0"/>
  </layers>
</arcgisws:mapengine>

```

ArcGIS definition providing only a single specific layer based on the layer name

```

<arcgisws:mapengine id="arcgis.roads">
  <url>http://vmbreakout:8399/arcgis/services/Base/MapServer<
/ url>
  <layers>
    <layer name="Aerial Photo"/>
  </layers>
</arcgisws:mapengine>

```

ArcGIS definition providing only a single specific layer based on the layer id, using an alternate format

```

<arcgisws:mapengine id="arcgis.roads">
  <url>http://vmbreakout:8399/arcgis/services/Base/MapServer<
/ url>
  <layers>
    <layer>0</layer>
  </layers>
</arcgisws:mapengine>

```

ArcGIS definition providing only a single specific layer based on the layer name, using an alternate format

```

<arcgisws:mapengine id="arcgis.roads">
  <url>http://vmbreakout:8399/arcgis/services/Base/MapServer<
/ url>
  <username>>true</username>
  <layers>
    <layer>Aerial Photo</layer>
  </layers>
</arcgisws:mapengine>

```

ArcGIS definition removing a specific layers based on the layer id

```

<arcgisws:mapengine id="arcgis.roads">
  <url>http://vmbreakout:8399/arcgis/services/Base/MapServer<
/ url>
  <layers>
    <remove>
      <layer id="12"/>
      <layer id="13"/>
    </remove>
  </layers>
</arcgisws:mapengine>

```

ArcGIS definition removing a specific layers based on the layer name, using an alternate format


```

<arcgisws:mapengine id="arcgis.roads">
  <url>http://vmbreakout:8399/arcgis/services/Base/MapServer<
/urll>
  <username>>true</username>
  <layers>
    <remove>
      <layer>Property (Pending)</layer>
      <layer>Property (Historical)</layer>
    </remove>
  </layers>
</arcgisws:mapengine>

```

Changing labels for legend

```

<arcgis:mapengine id="mapengine.vector.ags">
  <url>http://vmbreakout:8399/arcgis/services/Base/MapServer<
/urll>
  <legend>
    <aliases>
      <alias id="Boundary_Suburbs" label="Suburbs"/>
      <alias id="Property_BodyCorporate" label="
Body Corporate"/>
    </aliases>
  </legend>
</arcgis:mapengine>

```

ArcGIS Server map caches

Weave 2.4.1 adds support for client side map caches, previously Weave utilised ArcGIS Server map caches only on the server.

If you're currently using a map engine that's pointing to ArcGIS server then the client will automatically utilise an available map cache once you've upgraded to Weave 2.4.1, there should be no changes required to the client configuration.

There are a couple of requirements that must be met before this can occur though:

- The client configuration must be setup to utilise fixed scales and they must match those of the ArcGIS server map cache.
- The users browser must be able to directly connect to ArcGIS server (unless you're using a pre-generated tile cache being served directly via a web server).

If the version of ArcGIS server you're running supports it then the update will work with:

- Fused and multi-layer map caches
- Dynamic and pre-generated map caches
- Compressed and exploded map caches

There is additional information for configuring tiled map engines at [Tiled Map Engine](#). Specifically the sections of tile subsets and server resolutions.

Configuring the server side map engine

The use of the client side map cache still requires that a server side map engine be setup. The Weave server will interrogate the server to see if it supports a map cache and if it does will dynamically alter the client configuration for the map engine to make use of the map cache.

The server side map engine is also used to generate map images when required on the server, for example when embedding in a report.

The following sample configurations show standard ArcGIS Server server side map engine configurations.

They have been trimmed to keep the samples brief, but would normally contain at least additional connection pooling information and will usually contain other configuration options.

A map service with a fused map cache.

```
<arcgisws:mapserver id="raster.fused">
  <url>http://arcserver:8399/arcgis/services/AerialPhotography
/MapServer</url>
  <format>image/jpeg</format>
  <transparent>>false</transparent>
  <background>white</background>
  <selection>>false</selection>
  <acetate>>false</acetate>
</arcgisws:mapserver>
```

A map service with a multi-layer map cache

```
<arcgisws:mapserver id="raster.multilayer">
  <url>http://arcserver:8399/arcgis/services/Water/MapServer<
/url>
  <format>image/png</format>
  <transparent>>true</transparent>
  <selection>>false</selection>
  <acetate>>false</acetate>
</arcgisws:mapserver>
```

A map service with no map cache

```
<arcgisws:mapserver id="vector">
  <url>http://arcserver:8399/arcgis/services/Main/MapServer<
/url>
  <format>image/png</format>
  <transparent>>true</transparent>
</arcgisws:mapserver>
```

Map engines with a fused map cache will always be registered with the Weave server as having a single layer, with the id '0' and the name 'image', regardless of what's in the .mxd file.

Map engines with multi-layer map caches or no map caches will take their layer information from the .mxd file, but the existing 'username' setting still applies and tells Weave to register the layers using the layer name as the id, rather than the layer id itself which changes if layers are added/removed from the map service.

Sample Toc model

The following sample show how the three map engines would need to be configured in a toc model (this assumes that the vector map engine has three layers, the multilayer two layers and the fused layer always has one, and because we didn't set 'username' to true in the map engine config the layer ids are numeric indexes)

```
<toc:model id="sample">
  <mapengine>vector</mapengine>
  <entry label="Roads" layer="0"/>
  <entry label="Properties" layer="1"/>
  <entry label="Suburbs" layer="2"/>
  <entry label="Water Valves" mapengine="raster.multilayer"
layer="0"/>
  <entry label="Water Pipes" mapengine="raster.multilayer"
layer="1"/>
  <entry label="Aerial Photography" mapengine="raster.fused"
layer="0"/>
</toc:model>
```

Configuring the client side map engine

Previously if you had these three map engines included in your client configuration then they would all communicate with the Weave server to generate their map images.

Previously to utilise the map cache on the server the 'mapcache' property must be set to true for the map engine, that is no longer required as the 'mapcache' property defaults to true. If you wish to disable use of the map cache then it must be explicitly set to false

The configuration for the client would be something similar to

```

<client:config id="sample">
  <!-- standard config items appear here -->
  <view id="com.cohga.html.client.map.mapView">
    <!-- standard map view config items appear here -->
    <!-- we must set the scale to match the map caches -->
    <scales>
      <scale>1000</scale>
      <scale>10000</scale>
      <scale>100000</scale>
      <scale>1000000</scale>
    </scales>
    <!-- now add the three map engines -->
    <mapEngine id="raster.fused">
      <options>
        <transitionEffect>resize<
/transitionEffect>
          <alpha>true</alpha>
        </options>
      </mapEngine>
    <mapEngine id="raster.multilayer"/>
      <options>
        <transitionEffect>resize<
/transitionEffect>
          <alpha>true</alpha>
        </options>
      </mapEngine>
    <mapEngine id="vector">
      <options>
        <transitionEffect>resize<
/transitionEffect>
          <alpha>true</alpha>
          <ratio>1.2</ratio>
        </options>
      </mapEngine>
    </view>
  </client:config>

```

This would display the three map engines and the user will be able to turn any of the six layers on and off. That should be enough, if map caches are enabled the user will utilise them. If the map caches were deleted for the map engines then the client will still work, but it will communicate with the Weave server to generate its images.

Aligning a tile cache

This section on aligning tile caches is no longer applicable, as of release 2.5.22, manually aligning tile caches should not be required. If your tiles do not line up there is likely a separate configuration issue.

Sometimes the tile cache will not align exactly with the vector data, this can be corrected by specifying a `tileOffset`, in either the server map engine or client map engine configs. The `tileOffset` specifies an x and y shift that should be performed on the tiles before being displayed.

If we wanted to fix the fused map cache in the server configuration then we'd change it to

```

<arcgisws:mapserver id="raster.fused">
  <url>http://arcserver:8399/arcgis/services/AerialPhotography
/MapServer</url>
  <format>image/jpeg</format>
  <transparent>>false</transparent>
  <background>white</background>
  <selection>>false</selection>
  <acetate>>false</acetate>
  <tileOffset x="-15.5" y="6"/>
</arcgisws:mapserver>

```

to do it in the client we'd change the client config to be

```

<mapEngine id="raster.fused">
  <options>
    <transitionEffect>resize</transitionEffect>
    <alpha>>true</alpha>
    <tileOffset x="-15.5" y="6"/>
  </options>
</mapEngine>

```

Unfortunately you'll need to figure out the correct values for x and y by trial and error (the measure distance tool in the Weave client can help here).

Generally setting the `tileOffset` in the server config would be better since it only needs to be done once.

Directly serving tiles

If you've pre-generated the tiles from a fused cache and have the tiles published via web server then you can alter the client configuration to take advantage of this (this could also be used for a multi-layer map cache, but you'd only be able to point to one of the sub-layers).

Firstly you must specify a base URL that gives the location of the tiles, secondly you need to tell the client to use a direct connection rather than talking to ArcGIS server and finally you need to tell the client what format the tiles are stored in.

Currently this must all be performed in the client configuration, so for our fused map engine we'd need to change it to something like the following:

```

<mapEngine id="raster.fused">
  <url>http://arcserver/arcgiscache/AerialPhotography/Layers
/_alllayers</url>
  <options>
    <useArcGISServer>>false</useArcGISServer>
    <type>jpg</type>
    <transitionEffect>resize</transitionEffect>
    <alpha>>true</alpha>
    <tileOffset x="-15.5" y="6"/>
  </options>
</mapEngine>

```

This provides the best performance of all the client side map caching options, and with the proper use of expiry times in the web server can result in very fast map redraws.

Disabling client side map caching

If you want to disable the caching on the client you need to explicitly set the map engine type to 'weave'.

As part of the process of setting up the client side map tiling the Weave server will set the client map engine type to either 'agstiled' or 'agsrest' (depending upon the type of map cache created in AGS, fused vs. multi layer) UNLESS it's already explicitly set to something. We can take advantage of this to set the map type to 'weave', which is the default if not specified anyway, thereby stopping the alterations from taking place.

Disabling the use of client side tile caching.

```
<mapEngine id="raster.fused">
  <type>weave</type>
  <options>
    <type>jpg</type>
    <transitionEffect>resize</transitionEffect>
    <alpha>>true</alpha>
  </options>
</mapEngine>
```

forcing the client to contact the Weave server to generate map images.

Locally caching tiles

If you're accessing a remote AGS map engine and the response time from that engine is slow then you can turn on local caching of the tiles so that the Weave server will keep a local copy of each tile that's downloaded, this will improve the response the next time the tile is needed.

Enabling local caching of tiles

```
<arcgisws:mapserver id="vector">
  <url>http://arcserver:8399/arcgis/services/Main/MapServer<
/urll>
  <format>image/png</format>
  <transparent>>true</transparent>
  <cachetiles>>true</cachetiles>
</arcgisws:mapserver>
```

The tiles will be stored locally under the Weave directory at ...\\weave\\platform\\workspace\\agscache unless a system properties called `agscache.location` is set pointing to another directory location (allowing one local cache to be reused by multiple Weave instances).

Tiles can be set to expire after a certain period if it's believed that the underlying tiles will change over time, this is done by setting a `cacheexpiry` value to the number of minutes that a tile should be valid for.

Expiring tiles after 1 week

```
<arcgisws:mapserver id="vector">
  <url>http://arcserver:8399/arcgis/services/Main/MapServer<
/urll>
  <format>image/png</format>
  <transparent>>true</transparent>
  <cachetiles>>true</cachetiles>
  <cacheexpiry>10800</cacheexpiry>
</arcgisws:mapserver>
```

If you receive errors from ArcGIS Server related to invalid chunk sizes, when generating maps on the server add `<fetchThreads>1</fetchThreads>` to the server side `arcgisws:mapserver` map engine configuration

Map Engine ArcGIS (REST)

The ArcGIS REST map engine is similar to the ArcGIS Soap map engine but it utilises the ArcGIS REST API to communicate with ArcGIS, rather than the SOAP API used by the ArcGIS SOAP mapengine.

Namespace

urn:com.cohga.server.map.arcgis.ws#1.0

Tags

mapengine

Properties

Name	Type	Required	Default	Description
id	string	yes		Unique identifier for this map engine, will be used by other items to refer to this map engine
url	string	yes		The URL used to connect to the ArcGIS server, e.g. http://vmbreakout:6080/arcgis/rest/services/Topo250/MapServer
mapcache	boolean	no	false	Should a tile cache be used (if it's available). If this is false or not set then any tile cache generated for the map service won't be used
username	string	no		A username to connect to the server as
password	string	no		A password to connect to the server as, should be encrypted using 'encrypt' at the OSGi prompt
useaname	boolean	no	false	Use the layer name to refer to individual layers, especially useful when building a ToC model since the names don't change as often as the id. You should ensure that the layer name is unique if you set this to true. If not set or set to false then the layer identifiers for each layer (which are used in ToC models, amongst other things) will be taken from the order that the layer appears within the .mxd file, which can change if layers are added/removed from the .mxd file. If set to true then the label that appears in the .mxd file will be used as the identifier instead
transparent	boolean	no	false	If true then ArcGIS is asked to generate an image that is transparent where no map data is drawn
layers	url:n:layers	no		Alter the layer information from ArcGIS
legend	url:n:legend	no		Alter the display of the legend
direct	boolean	no	false	If true then Weave will send the URL's returned from ArcGIS directly to the client rather than proxying them on behalf of ArcGIS. This will help performance if ArcGIS is accessed internally and the clients can access the URL's the ArcGIS returns.

refer	string	no	<p>Referer header value when sending HTTP requests to ArcGIS server (since com.cohga.server.map.arcgis.ws bundle version 2.38.48)</p> <p>This is useful if ArcGIS server filters all requests by header value by comparing the referer domain against a list of whitelisted domains.</p> <p>It is probably better to set the referer value in the tokenservice config, since chances are that it is really just required as part of authentication token generation, and not just general request filtering, but if you are performing request filtering in ArcGIS based on the referer header then you should set it here (and probably in the token service config too unless you're using IP or Request IP).</p>
ignoregroups	boolean	no	<p>Should Weave ignore the groups when working with ArcGIS map documents.</p> <p>When Weave works with map layers it only deals with the layers themselves, groups don't come into play until you start creating a ToC model, and even then the grouping in ToC models is not used when referring to map layers (they're only used internally on the client to turn on/off a collection of individual layers). Normally this works fine for ArcGIS as Weave will ensure that if a layer is "visible", then all of its parent groups (on the ArcGIS side) will also be turned on. This ensures that the layer is actually visible (as a layer won't be displayed if it's visible but one or more of its parents aren't). This way you can setup your AGS map document however you want (with or without groups) and have the Weave ToC model be the single source of group related information for the ToC panel, and then Weave will ensure that the map layers that need to be drawn are drawn.</p> <p>However, sometimes it's desirable to have Weave recognize the groups (that's the ArcGIS map document groups, not ToC model groups) and have them treated as layers. By setting <code>ignoregroups</code> to <code>false</code> Weave will include entries in the map engine description for the groups as well as for the layers. In this case the ToC model and the ArcGIS map document groups will, in combination, be responsible for determining whether a layer is visible or not.</p>
username	string	no	<p>Username used to generate tokens. You should not set this, but instead include it in the <code>tokenservice</code> configuration</p>
password	string	no	<p>Password used to generate tokens. You should not set this, but instead include it in the <code>tokenservice</code> configuration</p>
tokenurl	string	no	<p>This can be a simple string providing the token service URL to override the one provided by ArcGIS, in which case you will need to set username and password properties above too, but you should use this as a sub-tag and include all the required information inside of that instead.</p>

Sub-tags

Name	Type	Cardinality
layers	urn:layers	0..1
legend	urn:legend	0..1
tokenservice	urn:tokenservice	0..1

Content

None

layers

Properties

None

Sub-tags

Name	Type	Cardinality
remove	urn:remove	0..1
layer	urn:layer	0..n

Content

None

remove

Properties

None

Sub-tags

Name	Type	Cardinality
layer	urn:layer	0..n

Content

None

layer

Properties

Name	Type	Required	Description
id	number	no	The id of the layer to match
name	string	no	The name of the layer to match

Sub-tags

None

*Content*The id or name, depending upon the `username` property, of the layer to match*Notes*

- The `layer` tag support two formats, one using the `id` and/or `name` attributes and the other using the tags content
- Either one or both of the `id` and/or `name` attributes must be set OR tag content must be provided
- If the content of the `layer` tag is used then the value of the `username` property determines if the content should match the layer id or the layer name
- Both the `id` and `name` attributes can be set, but they must match both values for the layer that you're trying to filter or they won't match the layer

legend

Alter the display of the legend

Properties

None

Sub-tags

Name	Type	Cardinality
alias	urn:aliases	0..1

Content

None

aliases

List the new labels to be displayed for a layer

Properties

None

Sub-tags

Name	Type	Cardinality
alias	urn:alias	0..n

Content

None

alias

A single label change for a layer

Properties

Name	Type	Required	Description
id	string	yes	The id of the layer to change
label	string	yes	The label to display

Sub-tags

None

Content

None

tokenservice

Provide the configuration information to allow Weave to talk to ArcGIS to generate tokens for authentication

Properties

Name	Type	Required	Description
username	string	no	Used with password when requesting a token from ArcGIS
password	string	no	Used with username when requesting a token from ArcGIS
token	string	no	Can be provided instead of username and password and will be used when requesting a token from ArcGIS
url	string	no	Override the token generation endpoint that ArcGIS provides
timeout	integer	no	Number of seconds to timeout connections and requests sent to the token generation endpoint. Default is 0, which mean use the default system timeout. This sets both <code>connectiontimeout</code> and <code>readtimeout</code> so use those rather than this property if you want different values for each.
connectiontimeout	integer	no	Number of seconds to timeout the connections to the token service
readtimeout	integer	no	Number of seconds to timeout requests sent to the token service

Notes

Any additional properties specified in the token service config will be included in the request sent to ArcGIS to generate the tokens, this can include things like `referer` if the token service requires a referer value.

Either username and password must be provided, or just token.

For the `url` parameter:

- if the value starts with "https://" or "http://" (you should really not use http://), the value will be used as is
- if the value starts with "/", then "https:" will be prepended to the value
- if otherwise, the value will be prepended with the schema, host and port used for the map service itself
- if the value is not set, Weave will query ArcGIS (based on the map engine URL) for the URL to use.

Examples:

Service URL	Token Service Value	Final URL
N/A	https://example.com/arcgis/tokens/	https://example.com/arcgis/tokens/
N/A	http://example.com/arcgis/tokens/	https://example.com/arcgis/tokens/
N/A	https://example.com:6443/arcgis/tokens/	https://example.com:6443/arcgis/tokens/
N/A	http://example.com:8080/arcgis/tokens/	http://example.com:8080/arcgis/tokens/

N/A	//example.com:6443/arcgis/tokens/	https://example.com:6443/arcgis/tokens/
N/A	//example.com:8080/arcgis/tokens/	https://example.com:8080/arcgis/tokens/
https://example.com/arcgis/rest/map/MapServer	/arcgis/tokens/	https://example.com/arcgis/tokens/
http://example.com/arcgis/rest/map/MapServer	/arcgis/tokens/	http://example.com/arcgis/tokens/
https://example.com:6443/arcgis/rest/map/MapServer	/arcgis/tokens/	https://example.com:6443/arcgis/tokens/
http://example.com:8080/arcgis/rest/map/MapServer	/arcgis/tokens/	http://example.com:8080/arcgis/tokens/

Examples

Simple ArcGIS definition

```
<arcgisws:mapengine id="arcgis.topo">
  <url>http://vmbreakout:6080/arcgis/rest/services/Topo250/MapServer</url>
</arcgisws:mapengine>
```

ArcGIS definition overriding CRS returned, asking for a transparent image and using the layer names, rather than order, as the unique identifier for a layer

```
<arcgisws:mapengine id="arcgis.roads">
  <url>http://vmbreakout:6080/arcgis/rest/services/Roads/MapServer</url>
  <crs>EPSG:28355</crs>
  <transparent>true</transparent>
  <username>true</username>
</arcgisws:mapengine>
```

Configuring token generation

```

<arcgisws:mapengine id="example1">
  <url>https://arcgis.lan/arcgis/rest/services/Roads/MapServer</url>
  <username>weave</username>
  <password>ENCAFDHASNADGQAEDAS</password>
</arcgisws:mapengine>

<arcgisws:mapengine id="example2">
  <url>https://arcgis.lan/arcgis/rest/services/Roads/MapServer</url>
  <tokenservice>
    <username>weave</username>
    <password>ENCAFDHASNADGQAEDAS</password>
  </tokenservice>
</arcgisws:mapengine>

<arcgisws:mapengine id="example3">
  <url>https://arcgis.lan/arcgis/rest/services/Roads/MapServer</url>
  <tokenservice>

<token>D9Tk_kRQbq4bkJQSM0gSkKjy7OGSfT05rtafQ77Ha0gEmRw5dc6vhTSN5MG5vmd
__-T0tMSaLZxlUf8ZLZGrDmNbnBjdl_nqt5y_U7HA4yA.</token>
  </tokenservice>
</arcgisws:mapengine>

<arcgisws:mapengine id="example4">
  <url>https://arcgis.lan/arcgis/rest/services/Roads/MapServer</url>
  <tokenservice>
    <url>/sharing/rest/generateToken/</url>
    <username>weave</username>
    <password>ENCAFDHASNADGQAEDAS</password>
    <referer>https://weave.lan/weave/main.html</referer>
  </tokenservice>
</arcgisws:mapengine>

<arcgisws:mapengine id="example5">
  <url>https://arcgis.lan/sharing/rest/services/Roads/MapServer<
/ur
l>
  <tokenservice>
    <url>https://portal.arcgis.lan/sharing/rest/generateToken/<
/ur
l>

<token>D9Tk_kRQbq4bkJQSM0gSkKjy7OGSfT05rtafQ77Ha0gEmRw5dc6vhTSN5MG5vmd
__-T0tMSaLZxlUf8ZLZGrDmNbnBjdl_nqt5y_U7HA4yA.</token>
    <serverUrl>https://arcgis.lan/arcgis</serverUrl>
  </tokenservice>
</arcgisws:mapengine>

```

The tiled map engine is used to support cached map services that provide tiles to the browser to draw the maps, as opposed to generating a complete map image.

Current Restrictions

When using a tiled map engine the scales/resolutions used by the client must match exactly those provided by the tiles, which also implies that you can not mix tiled map engines that have different resolutions for their tiles.

As of 2.5.28 this is not necessarily true, it's now possible to specify the resolutions of the server tiles separately and Weave will choose the closest.

If more than one crs, format or layer from a tiled map service is needed then multiple map engines must be created, each specifying a different crs, format or layer. Each tiled map engine can only represent a single map layer.

Performance/Quality improvements

As of 2.6.4 you can specify if a tiled map engine should aim for performance or quality when re-projecting tiles and also how many tiles should be fetched at in parallel.

`reprojectMode` can be set to `default`, `speed` or `quality`, and defaults to `default` if not set, which is a balance between the two (but `speed` is still significantly quicker than both `default` and `quality`).

`fetchThread`, which defaults to 1, specifies how many tiles will be fetched at once from the back end tile provider. More threads mean better performance, but it currently defaults to 1 since it's a new feature.

```
<tiled:mapengine id="osm">
  <type>osm</type>
  <format>image/png</format>
  <url>//a.tile.openstreetmap.org/${z}/${x}/${y}.png</url>
  <url>//b.tile.openstreetmap.org/${z}/${x}/${y}.png</url>
  <url>//c.tile.openstreetmap.org/${z}/${x}/${y}.png</url>
  <reprojectMode>speed</reprojectMode>
  <fetchThreads>8</fetchThreads>
</tiled:mapengine>
```

GeoWebCache

Example tiled map engine configuration

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns='urn:com.cohga.server.config#1.0' xmlns:tiled="urn:com.cohga.server.map.tiled#1.0">

  <tiled:mapengine id="mapengine.tiled">
    <type>gwc</type>
    <url>http://mapserver/geowebcache/service/wms?
TILED=true</url>
    <username>user</username>
    <password>ENCAZEPAMMBZAAUTUPMABFJEBZJAKFCXGME<
/password>
    <crs>EPSG:3857</crs>
    <format>image/jpeg</format>
    <layer>aerialphoto_2014_10_15</layer>
  </tiled:mapengine>
```

```
</config>
```

The above example shows how to create a GeoWebCache tiled map engine.

The `type` attribute defaults to "gwc" ("geowebcache" can also be used) and determines how the tiles are handled by the client.

The `url` points to where the capabilities document for the tile service can be downloaded from. Generally with GeoWebCache tiled services it's a regular WMS url with the `TILED=true` parameter included.

The `username` and `password` are optional (the password had been encrypted with the `encrypt` command at the osgi console) and should be used if the service requires authentication.

The `crs` parameter is required and specifies what coordinate reference system the tiles are projected in.

The `format` parameter is required and specifies what image format the tiles are presented as.

The `layer` parameter is required and specifies the layer within the service that should be used.

OpenStreetMap

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns='urn:com.cohga.server.config#1.0' xmlns:tiled="urn:com.
cohga.server.map.tiled#1.0">

    <tiled:mapengine id="mapengine.tiled">
        <type>osm</type>
        <url>http://a.tile.openstreetmap.org/{z}/{x}/{y}.
png</url>
        <url>http://b.tile.openstreetmap.org/{z}/{x}/{y}.
png</url>
        <url>http://c.tile.openstreetmap.org/{z}/{x}/{y}.
png</url>
        <format>image/png</format>
        <layer>0</layer>
        <label>image</label>
        <!-- The items shown here are the defaults and are
just included to show what can be changed.
            Do not just cut and paste the following
unless you intend changing it, and do not
            change it unless you know what you are doing. -->
        <tile>
            <resolutions>156543.
03390625,78271.516953125,39135.7584765625,19567.87923828125,...<full
list missing for brevity>...,0.5971642833948135</resolutions>
            <tileWidth>256</tileWidth>
            <tileHeight>256</tileHeight>
        </tile>
        <envelope crs="EPSG:3857" minx="-20037508.34" miny="
-20037508.34" maxx="20037508.34" maxy="20037508.34" />
    </tiled:mapengine>

</config>
```

The above example shows how to create a OpenStreetMap (OSM) tiled map engine.

The `type` attribute defaults to "osm" ("openstreetmap" can also be used) and determines how the tiles are handled by the client.

The `url` points to where the tiles can be downloaded from, and must include place holders for the x, y and z values.

The `format` parameter is optional and specifies what image format the tiles are presented as. The default format is "image/jpeg".

The `layer` parameter is optional and specifies what will be used as the layer id. The default is "0".

The `label` parameter is optional and specifies what will be used as the layer name. The default is "image".

The `tile` parameter is optional and specifies settings related to the tiles provided. The default is set according to the OSM spec, and should only be changed if you're using your own OSM tiles.

The `envelope` parameter is optional and specifies the extent of the tiles. The default is set according to the OSM spec, and should only be changed if you're using your own OSM tiles.

Here

Please refer to the Here Developer Portal for more information about licensing and pricing <https://developer.here.com/>

See the documentation for the Map Tile REST Service offered by Here https://developer.here.com/documentation/map-tile/dev_guide/topics/introduction.html

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns='urn:com.cohga.server.config#1.0' xmlns:tiled="urn:com.cohga.server.map.tiled#1.0">

  <?set HERE_APP_ID=YOUR_APP_ID?>
  <?set HERE_APP_CODE=YOUR_APP_CODE?>

  <tiled:mapengine id="mapengine.here.normal.day">
    <type>osm</type>
    <url><![CDATA[https://1.base.maps.api.here.com/maptile/2.1
/maptile/newest/normal.day/{z}/{x}/{y}/256/png8?
app_id=${HERE_APP_ID}&app_code=${HERE_APP_CODE}]]></url>
    <url><![CDATA[https://2.base.maps.api.here.com/maptile/2.1
/maptile/newest/normal.day/{z}/{x}/{y}/256/png8?
app_id=${HERE_APP_ID}&app_code=${HERE_APP_CODE}]]></url>
    <format>image/png</format>
    <layer>0</layer>
    <label>Here Streetmap</label>
  </tiled:mapengine>

  <tiled:mapengine id="mapengine.here.satellite.day">
    <type>osm</type>
    <url><![CDATA[https://1.aerial.maps.api.here.com/maptile/2.1
/maptile/newest/satellite.day/{z}/{x}/{y}/256/png8?
app_id=${HERE_APP_ID}&app_code=${HERE_APP_CODE}]]></url>
    <url><![CDATA[https://2.aerial.maps.api.here.com/maptile/2.1
/maptile/newest/satellite.day/{z}/{x}/{y}/256/png8?
app_id=${HERE_APP_ID}&app_code=${HERE_APP_CODE}]]></url>
    <format>image/png</format>
    <layer>0</layer>
    <label>Here Satellite</label>
  </tiled:mapengine>

  <tiled:mapengine id="mapengine.here.hybrid.day">
    <type>osm</type>
    <url><![CDATA[https://1.aerial.maps.api.here.com/maptile/2.1
```

```

/maptile/newest/hybrid.day/{z}/{x}/{y}/256/png8?
app_id=${HERE_APP_ID}&app_code=${HERE_APP_CODE}]]></url>
    <url><![CDATA[https://2.aerial.maps.api.here.com/maptile/2.1
/maptile/newest/hybrid.day/{z}/{x}/{y}/256/png8?
app_id=${HERE_APP_ID}&app_code=${HERE_APP_CODE}]]></url>
    <format>image/png</format>
    <layer>0</layer>
    <label>Here Hybrid</label>
</tiled:mapengine>

<tiled:mapengine id="mapengine.here.terrain.day">
    <type>osm</type>
    <url><![CDATA[https://1.aerial.maps.api.here.com/maptile/2.1
/maptile/newest/terrain.day/{z}/{x}/{y}/256/png8?
app_id=${HERE_APP_ID}&app_code=${HERE_APP_CODE}]]></url>
    <url><![CDATA[https://2.aerial.maps.api.here.com/maptile/2.1
/maptile/newest/terrain.day/{z}/{x}/{y}/256/png8?
app_id=${HERE_APP_ID}&app_code=${HERE_APP_CODE}]]></url>
    <format>image/png</format>
    <layer>0</layer>
    <label>Here Terrain</label>
</tiled:mapengine>

</config>

```

Google

You can also access Google map tiles using the same algorithm as OpenStreetMap.

The following shows how to create a map engine to represent each of the display types that Google provides tiles for. Once this is done, the map engines can be treated like any other tiled map engine and be added to a toc model and a client config.

The URLs contain special characters so these need to be escaped, or CDATA can be used to ensure they are interpreted correctly. Examples of both are shown below.

Note if you notice Google tiles using a different language then add `&hl=en` to the end of the URL (for English)

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns='urn:com.cohga.server.config#1.0' xmlns:tiled="urn:com.
cohga.server.map.tiled#1.0">

    <tiled:mapengine id="google.standard">
        <type>osm</type>
        <format>image/png</format>
        <url><![CDATA[/mt0.google.com/vt/lyrs=m&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[/mt1.google.com/vt/lyrs=m&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[/mt2.google.com/vt/lyrs=m&x=${x}&y=${y}
&z=${z}]]></url>
    </tiled:mapengine>

    <tiled:mapengine id="google.satellite">

```



```

        <type>osm</type>
        <format>image/png</format>
        <url><![CDATA[//mt0.google.com/vt/lyrs=s&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[//mt1.google.com/vt/lyrs=s&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[//mt2.google.com/vt/lyrs=s&x=${x}&y=${y}
&z=${z}]]></url>
    </tiled:mapengine>

    <tiled:mapengine id="google.alt">
        <type>osm</type>
        <format>image/png</format>
        <url><![CDATA[//mt0.google.com/vt/lyrs=r&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[//mt1.google.com/vt/lyrs=r&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[//mt2.google.com/vt/lyrs=r&x=${x}&y=${y}
&z=${z}]]></url>
    </tiled:mapengine>

    <tiled:mapengine id="google.terrain">
        <type>osm</type>
        <format>image/png</format>
        <url><![CDATA[//mt0.google.com/vt/lyrs=t&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[//mt1.google.com/vt/lyrs=t&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[//mt2.google.com/vt/lyrs=t&x=${x}&y=${y}
&z=${z}]]></url>
    </tiled:mapengine>

    <tiled:mapengine id="google.alt.terrain">
        <type>osm</type>
        <format>image/png</format>
        <url><![CDATA[//mt0.google.com/vt/lyrs=p&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[//mt1.google.com/vt/lyrs=p&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[//mt2.google.com/vt/lyrs=p&x=${x}&y=${y}
&z=${z}]]></url>
    </tiled:mapengine>

    <tiled:mapengine id="google.roads">
        <type>osm</type>
        <format>image/png</format>
        <url><![CDATA[//mt0.google.com/vt/lyrs=h&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[//mt1.google.com/vt/lyrs=h&x=${x}&y=${y}
&z=${z}]]></url>
        <url><![CDATA[//mt2.google.com/vt/lyrs=h&x=${x}&y=${y}
&z=${z}]]></url>
    </tiled:mapengine>

```

```

        <tilde:mapengine id="google.hybrid">
            <type>osm</type>
            <format>image/png</format>
            <url><![CDATA[//mt0.google.com/vt/lyrs=y&x=${x}&y=${y}
&z=${z}]]></url>
            <url><![CDATA[//mt1.google.com/vt/lyrs=y&x=${x}&y=${y}
&z=${z}]]></url>
            <url><![CDATA[//mt2.google.com/vt/lyrs=y&x=${x}&y=${y}
&z=${z}]]></url>
        </tilde:mapengine>

</config>

```

Note: Before using the Google Map Service, please ensure you read through the Google Maps [Terms of Service](#) and ensure that you have correctly setup licensing with Weave.

ArcGIS

There are two ArcGIS specific methods of providing tile cache support to Weave clients. The method outlined here is specifically provided to support exposing a local compressed ArcGIS map service tile cache, i.e. a tile cache created by ArcGIS where the tiles are stored in the compressed format (with `.bundle`, or with `.bundle` and `.bundlex` files) on the local file system of the Weave server. If this is not your situation then you should use the ArcGIS specific map engine to expose the tiles (see [Map Engine ArcGIS \(REST\)](#) or [Map Engine ArcGIS \(SOAP\)](#)).

For the ArcGIS tiles map engine the `type` value should be set to `arcgis` and a `tilingScheme` value must be provided that points to the local `conf.xml` file.

Note this requires Weave 2.6.6 or later.

Exposing compressed ArcGIS tile cache

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns='urn:com.cohga.server.config#1.0' xmlns:tilde="urn:com.
cohga.server.map.tiled#1.0">

    <tilde:mapengine id="ags.tiled.aerial10cm">
        <type>arcgis</type>
        <tilingScheme>/data/spatial/raster/aerial10cm/Layers
/conf.xml</tilingScheme>
    </tilde:mapengine>

</config>

```

Client

Once the map engine is created it must be added to a ToC model and client map view the same as any other map engine. The client map engine configuration should specify only the map engine id. Weave provides all of the parameters required to configure the map engine and it obtains these by processing the 'capabilities document' of the map service.

Example showing embedding of tiled map engine in a client map view

```

<view id='com.cohga.html.client.map.mapView'>
    <mapEngine id="mapengine.tiled"/>
    <mapEngine id="mapengine.vector"/>

```

```

    <mapEngine id="mapengine.selection">
      <options>
        <selection>true</selection>
      </options>
    </mapEngine>
    <!-- resolutions should match tiled map engine -->
    <resolutions>156543.
03390625,78271.516953125,39135.7584765625,19567.87923828125,...full
list missing for brevity...,0.5971642833948135</resolutions>
    <!-- CRS should match tiled map engine -->
    <crs>EPSG:3857</crs>
  </view>

```

Tile Subsets

It's possible to use a subset of tiles in the client that does not cover the entire zoom range available in that tile set. For example, if you're using a Google tile set and the corresponding resolutions, there are on average 20 different zoom levels available. However you might wish to include your own local tile set that cover a subset of those 20 zoom levels. Or, if you're using the Google or OpenStreetMap tile sets, you might want to limit the user to a smaller zoom range and not allow them to zoom out beyond a certain level. Both of these situations can be catered for in Weave.

To include your local tile set you need to configure the map engine to tell it at which zoom level it should start at and how many zoom levels it covers, otherwise it will assume that it starts at zoom level 0 and has 20 zoom levels available (assuming that you've setup the client with the default Google zoom levels).

This is done by setting the `numZoomLevels` and `zoomOffset` options for the map engine. Also you may need to reduce the number of resolutions listed for the map view.

For example, if we wanted to update the map view example above so that the user can only access the first 8 zoom levels, we set `numZoomLevels` to 8 and set `zoomOffset` to 12 (20, the default number of zoom levels for the map view, minus 8) and then change the client map view resolutions list to only include the resolutions we want to enable.

Tile subset example

```

<view id='com.cohga.html.client.map.mapView'>
  <mapEngine id="mapengine.tiled">
    <options>
      <numZoomLevels>8</numZoomLevels>
      <zoomOffset>12</zoomOffset>
    </options>
  </mapEngine>
  <mapEngine id="mapengine.vector"/>
  <mapEngine id="mapengine.selection">
    <options>
      <selection>true</selection>
    </options>
  </mapEngine>
  <!-- resolutions should match tiled map engine -->
  <resolutions>38.218514137268066, 19.109257068634033,
9.554628534317017, 4.777314267158508, 2.388657133579254,
1.194328566789627, 0.5971642833948135, 0.298582142</resolutions>
  <!-- CRS should match tiled map engine -->
  <crs>EPSG:3857</crs>
</view>

```

Alternatively, if you still wanted to allow 8 zoom levels but wanted to skip the first two lowest zoom levels so the user can't zoom in too far, then you'd still set `numZoomLevels` to 8, but you'd set `zoomOffset` to 10 (20, minus 8, minus 2) and set the map view resolutions to match the 8 zoom levels (skipping the two lowest resolutions).

Alternate tile subset example

```
<view id='com.cohga.html.client.map.mapView'>
  <mapEngine id="mapengine.tiled">
    <options>
      <numZoomLevels>8</numZoomLevels>
      <zoomOffset>10</zoomOffset>
    </options>
  </mapEngine>
  <mapEngine id="mapengine.vector"/>
  <mapEngine id="mapengine.selection">
    <options>
      <selection>true</selection>
    </options>
  </mapEngine>
  <!-- resolutions should match tiled map engine -->
  <resolutions>152.87405654907226, 76.43702827453613,
38.218514137268066, 19.109257068634033, 9.554628534317017,
4.777314267158508, 2.388657133579254, 1.194328566789627</resolutions>
  <!-- CRS should match tiled map engine -->
  <crs>EPSG:3857</crs>
</view>
```

Server Resolutions

It is now possible to specify the resolutions provided by the tiles separately from those used by the client, and then have Weave choose the closest.

Note:

- This will cause the displayed tiles to not look as crisp as they would if the correct resolutions are used.
- The tiles must still be in the same projection as the client.

serverResolutions example

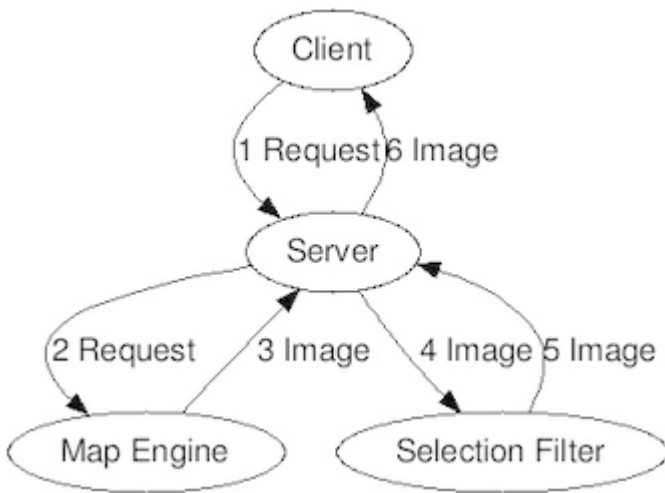
```
<view id='com.cohga.html.client.map.mapView'>
  <mapEngine id="mapengine.tiled">
    <options>
      <serverResolutions>160000,
80000,40000,20000,...full list missing for brevity...,0.5<
/serverResolutions>
    </options>
  </mapEngine>
  <mapEngine id="mapengine.vector"/>
  <mapEngine id="mapengine.selection">
    <options>
      <selection>true</selection>
    </options>
  </mapEngine>
  <resolutions>156543.
```

```
03390625,78271.516953125,39135.7584765625,19567.87923828125,...full
list missing for brevity...,0.5971642833948135</resolutions>
  <!-- CRS should match tiled map engine -->
  <crs>EPSG:3857</crs>
</view>
```

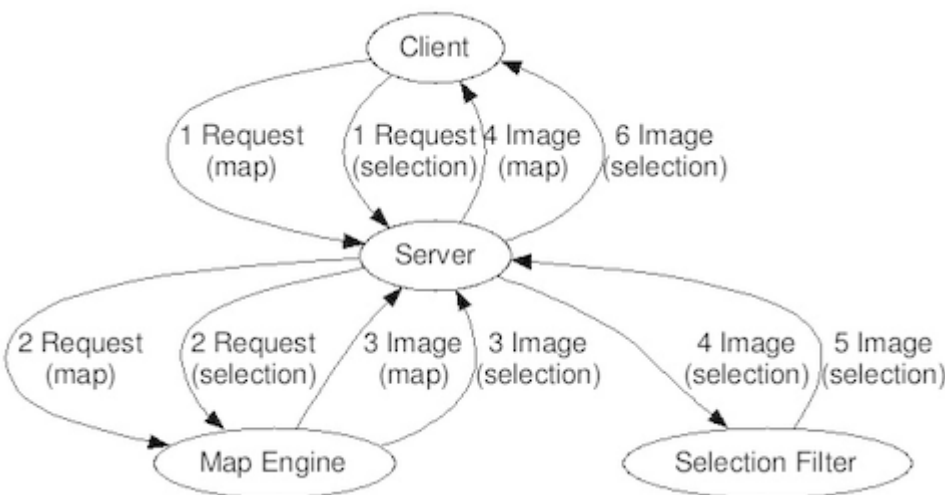
Map Engine Selection

Selection Rendering Introduction

Originally in Weave the drawing of the current selection was always done as part of the rendering the map image, this was because originally there was only a single map layer on the client. In this situation each update of the map image included a step, on the server, where the current selection was drawn on top of the map image generated by the underlying map engine before passing the image back to the browser for display.

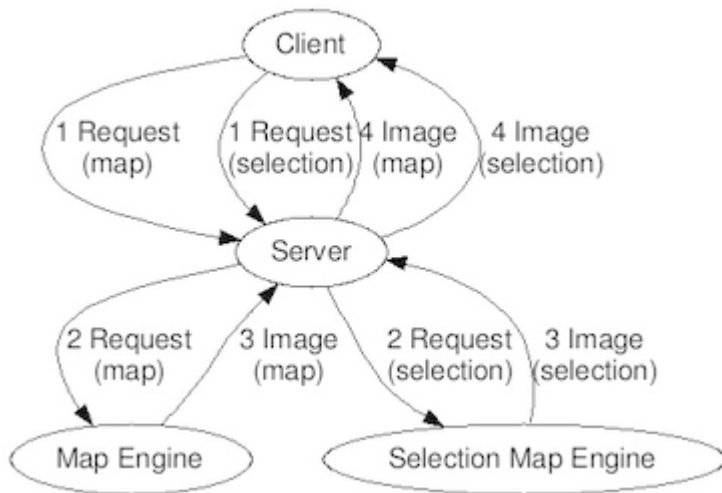


Once support for more than one map engine was available on the client this process was supplemented by configuring the client as such that the original map image was untouched and the current selection was drawn as a completely separate image, where both images were sent to the browser which displayed the selection image on top of the original map image.



This provided improved performance for the client since it no longer required an entire map redraw if just the selection was changed, and was all done through configuration. The configuration basically involved creating a dummy map engine, based on the original map engine, that never ended up drawing any of its layers but instead went through the process outlined above of having the selection drawn on the (empty) image it generated.

But this was only ever a hack since it relied upon the use of a dummy map engine configured through guesswork. Now however this dummy map engine can be replaced by a map engine specifically designed to draw a separate selection layer on top of other layer in the client browser, it's this map engine, the Selection Map Engine that is described here.



Configuration of the server map engine

The drawing of the current selections on the map is performed in Weave by a separate bundle `com.cohga.server.map.selection`. This way the other map engines are kept as simple as possible, just drawing an image of a specific size, covering a specific area with a specific list of layers visible.

Because all of the selection drawing is handled by the same plugin it means that the configuration for every map engine is the same when it comes to selections.

Creating a selection map engine follows the same process as other map engines and just has a new namespace of its own when it comes to configuration, obviously the content of the configuration for the selection map engine is different since it serves a different purpose, but we explain that configuration here.

Basic configuration to setup a selection map engine on the server

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:selection="urn:com.cohga.server.map.selection#1.0">

  <selection:mapengine id="selection">
    <!-- Selection specific configuration if required -->
  </selection:mapengine>

</config>

```

The above configuration describes just what's required to setup the server side portion of the selection map engine. The client must also be configured to make use of this map engine and that is described later.

Beyond that it's also possible to change some other settings that effect selection drawing.

Customising the selection map engine

Setting a maximum scale

The maximum scale beyond which selections won't be drawn can be set with the `maxscale` value.

Setting maximum scale for selections to be drawn at

```
<selection:mapengine id="selection">
  <maxscale>50000</maxscale>
</selection:mapengine>
```

Limiting the drawing time

You can also limit the amount of time that is spent drawing the selection and limit the number of features that are ever drawn for the selection.

By setting a `timeout` value, in seconds, you can specify that once the plugin has spent that many seconds drawing features then it should stop, regardless of how many it has drawn. Currently, the default is 10 seconds. As of Weave 2.5.28, it's possible to add a 's' or 'ms' suffix to the timeout value to specify if the units are seconds or milliseconds, e.g. `<timeout>500ms</timeout>`.

By setting a `limit` value you can specify that the plugin should draw a maximum of that many features. There is no default value.

Setting time and selection size limits

```
<selection:mapengine id="selection">
  <limit>5000</limit>
  <timeout>5</timeout>
</selection:mapengine>
```

Both of these setting can help ensure a speedy map response even if the user has a large number of selected features.

Other optimizations

Additionally, if you really want to tweak the performance of the selection rendering you can disable the smoothing of the drawing by turning off anti-aliasing, which may gain a few milliseconds. The default is to anti-alias the selection drawing.

Turning off antialiasing

```
<selection:mapengine id="selection">
  <antialiasing>false</antialiasing>
</selection:mapengine>
```

Also, since the selected features are drawn in batches you can alter the number of features that are drawn in each batch. The default batch size is 1000.

Increasing this value will reduce the overhead that's required for each batch but sometimes there are limits on the underlying infrastructure (database or spatial engine) that limit the number of features that can be retrieved at once. You can try increasing this value and selecting more than that many features and see if the selection is still drawing, if it is then it's safe to use the new limit value, otherwise it should be reduced until a safe value is determined. There's little point in setting this value to more than the `limit` value, if set.

Changing number of features rendered in each batch

```
<selection:mapengine id="selection">
  <batchsize>2500</batchsize>
</selection:mapengine>
```

Since there are a large number of possible image formats (in regards to bits-per-pixel, colour palettes, transparency support, etc) the selection drawing plugin will generate a new image of a known format and draw the selection on top, this way a consistent image format is produced at the end (this is mainly to handle problems experienced with earlier versions of Internet Explorer).

You can specify the default format for the images generated by the selection map engine with the `format` property.

Changing the format of the generated image

```
<selection:mapengine id="selection">
  <format>image/png32</format>
</selection:mapengine>
```

By default the selection map engine reports that it support 32 and 8 bit png and gif, with 32 bit png being the default if no format is specified.

Prior to version 1.2.3 of the selection map engine bundle the default image format was png8 which would result in solid filled polygons (if the default fill styles were used), in version 1.2.3 of the bundle this was changed to png32. It's recommended that users of versions before 1.2.3 explicitly set the format to png32.

8 bit png and gif are similar in size and image quality, 32 bit png produces the best image quality but are larger than the 8 bit formats, gif is recommended if you're sure that all of your clients will be using IE6 (in which case you should also ensure that your fill styles are all fully transparent).

Changing colours

The rendering of the selections can also be altered by adjusting the line and fill colours and transparency, the line widths, and the point symbology.

Because Weave can display selections for more than one entity at a time you can change the symbology for both the *active* entity and the *inactive* entities independently. This allows for the inactive selections to be drawn in a slightly less visible manner. If you wish to have the selections drawn the same regardless of the active entity then you don't need to include the *active* and *inactive* tags and just include one set of setting for each feature type (point, line and/or polygon).

This is done by adding a *point*, *line* and/or *polygon* tag to an *override* tag (the *override* tag is optional as of version 1.3.3 of the bundle).

CSS style rendering properties

Name	Type	Description
strokecolour	colour	The colour of the line to draw lines
strokewidth	number	The width of the lines to draw
strokeopacity	number (0-1)	The opacity of the lines drawn
fillcolour	colour	The colour used when filling selections
fillopacity	number (0-1)	The opacity used when filling selections
mark	'square', 'circle', 'triangle', 'star', 'cross' or 'x'	The type of marker to draw point selections with
marksize	number	The size of the marker used to draw point selections
markrotation	number (0-360)	The angle of rotation use to draw point selection markers with
markopacity	number (0-1)	The opacity of point selection markers

- Colours can be specified as either an RGB value, as decimal or hex (e.g., #ff0000), or as one of black, blue, cyan, darkgray, darkgrey, gray, grey, green, lightgray, lightgrey, magenta, orange, pink, red, white, yellow.
- The word 'colour' or 'color' and hyphens can be used in the property names, for example the following are all equivalent, 'strokecolour', 'strokecolor', 'stroke-colour' and 'stroke-color'.
- When setting line properties only the stroke settings are read.
- When setting polygon properties only stroke and fill settings are read.
- When setting point properties stroke, fill and marker settings are read.
- You don't need to include point, line and polygon unless you want to change all renderers, you only need to include the one(s) you want to change, and you only need to include the properties that you want to change from the defaults

Changing the basic rendering styles of point, lines and polygons

```
<selection:mapengine id="selection">
  <override>
```



```

<point>
  <active>
    <strokecolour>white</strokecolour>
    <strokewidth>1</strokewidth>
    <strokeopacity>0</strokeopacity>
    <fillcolour>cyan</fillcolour>
    <filloppacity>1</filloppacity>
    <mark>circle</mark>
    <marksize>12</marksize>
  </active>
  <inactive>
    <strokecolour>cyan</strokecolour>
    <strokewidth>2</strokewidth>
    <strokeopacity>1</strokeopacity>
    <fillcolour>white</fillcolour>
    <filloppacity>0</filloppacity>
    <mark>circle</mark>
    <marksize>12</marksize>
  </inactive>
</point>
<line>
  <active>
    <strokecolour>cyan</strokecolour>
    <strokewidth>3</strokewidth>
    <strokeopacity>0.75</strokeopacity>
  </active>
  <inactive>
    <strokecolour>#00ffff</strokecolour>
    <strokewidth>2</strokewidth>
    <strokeopacity>0.25</strokeopacity>
  </inactive>
</line>
<polygon>
  <active>
    <strokecolour>#00ffff</strokecolour>
    <strokewidth>3</strokewidth>
    <strokeopacity>0.75</strokeopacity>
    <fillcolour>#00ffff</fillcolour>
    <filloppacity>0.125</filloppacity>
  </active>
  <inactive>
    <strokecolour>#00ffff</strokecolour>
    <strokewidth>2</strokewidth>
    <strokeopacity>0.25</strokeopacity>
    <fillcolour>#00ffff</fillcolour>
    <filloppacity>0.025</filloppacity>
  </inactive>
</polygon>
</override>
</selection:mapengine>

```

Overriding selection drawing type for a specific entity

Occasionally if you're using a spatial engine that does not support specific geometry types, `point`, `line` and `polygon`, but instead just a generic `geometry` type (this includes SQL Server) then you may need to tell the selection map engine what type of renderer to use for a particular entity. This is particularly important if you're seeing "Unable to make suitable draw style for ..." messages in the `weave.log` file (note that another page that may be relevant in this situation, and is probably a better solution is [this page](#) if you're using SQL Server)

This can be done by setting the geometry type for an entity as follows:

Overriding the rendering type for an entity

```
<selection:mapengine id="selection">
  <override>
    <entity id="property">polygon</entity>
    <entity id="road">line</entity>
    <entity id="trafficlight">point</entity>
  </override>
</selection:mapengine>
```

You can use `point` or `multipoint`, `line`, `multiline`, `linestring` or `multilinestring`, and `polygon` or `multipolygon` (all values will be accepted, but they'll be interpreted as 'point', 'line' or 'polygon' regardless).

This then sets the geometry type for the entities selection, and the renderer for that particular type will be used to draw the selections.

Completely overriding the selection drawing for a specific entity

It's also possible to specify the rendering directly for a specific entity by using the entity id instead of `point`, `line` or `polygon`, for example:

Overriding an entities rendering using CSS properties

```
<selection:mapengine id="selection">
  <override>
    <entity id="property">
      <active>
        <strokecolour>green</strokecolour>
        <strokewidth>3</strokewidth>
        <strokeopacity>0.75</strokeopacity>
        <fillcolour>green</fillcolour>
        <fillopaicity>0.125</fillopaicity>
      </active>
      <inactive>
        <strokecolour>green</strokecolour>
        <strokewidth>2</strokewidth>
        <strokeopacity>0.25</strokeopacity>
        <fillcolour>green</fillcolour>
        <fillopaicity>0.025</fillopaicity>
      </inactive>
    </entity>
  </override>
</selection:mapengine>
```

Advanced selection rendering

Additionally it's possible to specify the selection rendering using SLD ([Styled Layer Descriptor](#)).

This can be done either by specifying the SLD directly in the config file or by referencing a file containing the SLD from the config file.

Referencing a SLD file

```
<selection:mapengine id="selection">
  <override>
    <entity id="property">property.sld</entity>
  </override>
</selection:mapengine>
```

The contents of `property.sld`, which should be in the `workspace` directory, should contain valid SLD XML to describe the renderer to use for the property entity.

Alternatively the SLD can be embedded directly in the configuration file (but is not recommended)

Embedding a SLD file

```
<selection:mapengine id="selection">
  <override>
    <entity id="property"><![CDATA[
      <StyledLayerDescriptor version="1.0.0" xsi:schemaLocation="
http://www.opengis.net/sld StyledLayerDescriptor.xsd" xmlns="
http://www.opengis.net/sld" xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://www.w3.
org/2001/XMLSchema-instance">
        <NamedLayer>
          <Name>property</Name>
          <UserStyle>
            <!-- Rest of SLD content here -->
          </UserStyle>
        </NamedLayer>
      </StyledLayerDescriptor>
    ]]></entity>
  </override>
</selection:mapengine>
```

There's an open source tool that can be used to create SLD files available at [AtlasStyler SLD editor](#).

Optional 'override' tag in definition

Additionally, the `<override>` tags are now optional.

The following two examples are equivalent:

Referencing a SLD file with override tag

```
<selection:mapengine id="selection">
  <override>
    <entity id="property">property.sld</entity>
  </override>
</selection:mapengine>
```

Referencing a SLD file without override tag

```
<selection:mapengine id="selection">
  <entity id="property">property.sld</entity>
</selection:mapengine>
```

Drawing selections with a separate map engine on the client

Using the selection map engine requires that we add the selection map engine to the client configuration so that it is drawn on top of the existing layers.

New 'selection' map engine added to client

```
<client:config id="...">
  ...
  <view id="com.cohga.html.client.map.mapView">
    ...
    <mapEngine id="raster">
      <options>
        <selection>false</selection>
        <alpha>false</alpha>
      </options>
    </mapEngine>

    <mapEngine id="vector">
      <options>
        <selection>false</selection>
        <alpha>true</alpha>
      </options>
    </mapEngine>

    <mapEngine id="selection">
      <options>
        <!-- tell the client map engine that this layer
is for handing selections -->
        <selection>true</selection>
        <!-- ensure the the client knows that this map
engine supports transparency -->
        <alpha>true</alpha>
      </options>
    </mapEngine>
  </view>
</client:config>
```

The final step is to switch the toc model to use the new selection map engine for the selection related entries by adding a mapengine tag to the Selection entry.

```
<toc:model id="toc.main">
  <!-- set the default map engine for entries that don't have
one set -->
  <mapengine>vector</mapengine>
```

```

    <entry label="Selection" exclusive="true">
      <!-- change the map engine for every entry in this group
to use the selection map engine-->
      <mapengine>selection</mapengine>
      <entry layer="_selection.active" label="Active" checked="
true"/>
      <entry layer="_selection.all" label="All" checked="false"
/>
      <entry label="Selected" checked="false">
        <!-- individual selection layer here -->
        ...
      </entry>
    </entry>

    <!-- regular layer entries here -->
    ...

    <entry label="Raster" exclusive="true">
      <!-- use the raster map engine for every entry in here -->
      <mapengine>raster</mapengine>
      ...
    </entry>
  </toc:model>

```

Weave Map Engine

As of Weave 2.5 there is a built in map engine that is able to generate map images based on spatial data that has been registered using a [Spatial Engine](#).

The configuration for the Weave map engine consists of two parts, the first part defines what data will be made available to be drawn by the map engine, and the second part determines how that data will be drawn.

Currently all of the data that's drawn by a Weave map engine must come from the same spatial engine, but you can have multiple Weave map engines configured.

And the map engine also only supports vector data, it does **not** support raster data.

The namespace used for the Weave map engine is `com.cohga.server.map.weave`.

Example Weave map engine

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:weave="urn:com.
cohga.server.map.weave#1.0">

    <weave:mapengine id="vector">
      <spatialengine>gis</spatialengine>
      <format>image/png32</format>
      <layers>
        <layer id="contours01" label="Contours 1m"
layer="GIS.CONTOURS_01M" style="contour"/>
        <layer id="contours02" label="Contours 2m"
layer="GIS.CONTOURS_02M" style="contour"/>
        <layer id="contours05" label="Contours 5m"

```

```

layer="GIS.CONTOURS_05M" style="contour"/>
    <layer id="contours10" label="Contours 10m"
layer="GIS.CONTOURS_10M" style="contour"/>
    </layers>
<styles>
    <style id="contour">
        <type>line</type>
        <stroke-color>#853111</stroke-color>
        <stroke-width>2</stroke-width>
    </style>
</styles>
</weave:mapengine>

</config>

```

The above example creates a map engine with four layers that all use the same simple rendering.

Note that to use this map engine in a client you still need to create a [Table Of Contents](#) and add it as a map engine to the [Map View](#). This map engine is treated exactly the same as the other map engines (e.g. ArcIMS, WMS, ArcGIS Server, etc)

You can specify one or more filters for each `layer` that can be used to refine what data is retrieved from the underlying table, e.g.

Filtering data using CQL

```

<layer id="active_incidents" label="Active Incidents" layer="GIS.
INCIDENTS" style="incidents">
    <filter>STATUS = 'ACTIVE'</filter>
</layer>
<layer id="inactive_incidents" label="Inactive Incidents" layer="GIS.
INCIDENTS" style="incidents">
    <filter><![CDATA[STATUS <> 'ACTIVE']]</filter>
</layer>

```

Here we've split a single spatial table into two layers, note the use of the `<![CDATA[. . .]>` to enclose the second filter, because it contains special XML characters (`<` and `>`).

You can use multiple `filter` tags for each layer, and they will be combined using an AND, that is the data must satisfy all filters to be displayed.

The filters are specified using the [CQL](#) query language, which is similar to SQL.

The `styles` section can define multiple named styles, which are then referenced using the `style` attribute of the individual `layer` in the `layers` section.

The example above uses a simple inline style definition that uses CSS like attributes to define how the vector data should be drawn. It's also possible to use an external [SLD](#) (Styled Layer Descriptor) file to describe how the layer should be displayed, if you require more advanced styling. In this case you can just reference the SLD file using the `layer style` attribute, e.g.

Using an external SLD file to style a layer

```

    <layer id="contours01" label="Contours 1m"
layer="GIS.CONTOURS_01M" style="sld\contours.sld"/>

```

In this example there should be a file located at `weave\platform\workspace\sld\contours.sld` that contains the XML definition for the style.

Example SLD file

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<StyledLayerDescriptor version="1.0.0" xmlns="http://www.opengis.net
/sld" xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://www.w3.
org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/sld http://schemas.
opengis.net/sld/1.0.0/StyledLayerDescriptor.xsd">
  <NamedLayer>
    <Name>Contour</Name>
    <UserStyle>
      <Title>Contour</Title>
      <FeatureTypeStyle>
        <Rule>
          <LineSymbolizer>
            <Stroke>
              <CssParameter name="stroke">#853111</CssParameter>
              <CssParameter name="stroke-width">2</CssParameter>
            </Stroke>
          </LineSymbolizer>
        </Rule>
      </FeatureTypeStyle>
    </UserStyle>
  </NamedLayer>
</StyledLayerDescriptor>
```

This SLD file does not do much more than the inline style shown in the first example, but this page isn't a tutorial on SLD.

Alternatively, when referring to an external SLD file it can also be specified in the style itself, rather than the layer, so the layer can reference one of the inline styles, and the style then points to the external SLD file, e.g.

```
<styles>
  <style id="contour">sld\contours.sld</style>
</styles>
```

This way if you use the same SLD file for multiple layers you can just reference a single inline style and have that style refer to the SLD file.

Inline Styles

When specifying a style within the Weave map engine config directly you need to specify an id for the style, so that it can be referenced by the layer, and also what type of features the style will be applied to, `point`, `line` (or `linestring`) or `polygon`, and you do this by setting the `type` attribute for the style.

Additionally you can specify a maximum and minimum scale range for the style using `minScale` (or `minscale`) and `maxScale` (or `maxscale`).

Finally, you can also specify a label (or title) and a description (or abstract) which can be used when generating the legend.

After that it's just a case of setting the appropriate attributes that define how you want the data rendered.

The following are the different values that can be used with an inline style definition, not that both "colour" and "color" can be used, and the hyphen is also optional, so "stroke-colour" and "strokecolor" can both be used.

When specifying colours you can either use hex notation with RRGGBB and prefix the value with a #, for example #ff0000 for red, or you can use a pre-defined colour from the list `black`, `blue`, `cyan`, `darkgray`, `darkgrey`, `gray`, `grey`, `green`, `lightgray`, `lightgrey`, `magenta`, `orange`, `pink`, `red`, `white`, `yellow`.

The mark type can be one of square, circle, triangle, star, cross, arrow or x.

When specifying a style for lines you can specify the stroke attributes, for a polygons both stroke and fill attributes, and for points stroke, fill and mark are required.

Stroke

Name	Default
stroke-colour	orange
stroke-width	3
stroke-opacity	1

Fill

Name	Default
fill-colour	orange
fill-opacity	0.125

Mark

Name	Default
mark-type	circle
mark-size	10
mark-opacity	1
mark-rotation	0

Joining Tables

From 2.6.4 it's possible to join to another table if the underlying spatial engine is using a database, i.e. this won't work for shapefiles for example.

This would for example provide the ability to filter the rows displayed on the map based on the value in a column in a table different from the one containing the geometry. To join the additional table you need to add a `<join>` tag with the information about the other table (which must be in the same database) and how it is joined to the source table. Additionally you can set the type of join, inner or outer, and add a filter for the extra table.

```
<layer id="my_active_incidents" label="My Active Incidents" layer="
GIS.INCIDENTS" style="incidents">
  <join>
    <type>inner</type> <!-- this is the default, and will
result in only the rows that are in both tables being displayed -->
    <table>GIS.INCIDENT_DETAILS</table> <!-- The name of
the table to join to GIS.INCIDENTS -->
    <where>ID = ID</where> <!-- The columns in the two
tables to join, must always be an equals -->
    <filter>STATUS = 'ACTIVE'</filter> <!-- An additional
filter to further reduce the rows that are displayed -->
    <filter>USERID = '${user.id}'</filter> <!-- Another filter,
this time selecting rows that belong to the current user -->
  </join>
</layer>
```

Join

N	Ty	C	D	Description
---	----	---	---	-------------

name	type	arity	default	
table	string	1..1		The name of the table to join to the source table
where	string	1..1		A description of the join between the two columns. This should always be a X = Y type of expression where X is the column in the source table and Y is the column in the extra table. The rows from the two tables will be matched based on the values in the X column in the source table having the same value as the Y column in the target table.
type	'outer' or 'inner'	0..1	'inner'	Should the join between the two tables be an inner join, only rows in both tables are returned, or an outer join, rows in the source table are still returned even if they don't have a matching row in the extra table
filter	string	0..n		This is a ECQL expression that will be applied to the extra table to filter the rows that should be included, this includes support for user attributes.

Dynamic Map Engine

The dynamic map engine is a virtual map engine that provides additional functionality on top of other map engines, for example load balancing and fail over support.

When a dynamic map engine is configured, it is setup so that it points to one or more other map engines then anything that previously pointed to the original map engine is changed to point to the new map engine, or the old map engine is renamed and the dynamic map engine is given the id of the old map engine then nothing else needs to change.

Depending upon how the dynamic map engine is configured it can provide the following different functionality:

- Redirect
 - If there's only one sub-map engine, then any request to the map engine will be sent directly to the sub-map engine.
- Fail-over
 - If "failover" is set to true for the map engine, then the first sub-map engine listed will be used until it fails, then the second sub-map engine will be used instead.
- Alternating styles
 - If "altlayers" is set by providing a list of layer id's, when none of those layers are requested, the first sub-map engine will be used, but if any of those layers are requested then the second sub-map engine will be used.
- Load balancing
 - If there is more than one sub-map engine listed and "failover" and "altlayers" are not set then incoming requests will be spread across the listed sub-map engines.
- Stitching
 - If there is only one sub-map engine and "maxwidth" and "maxheight" are set, then any request for a map over that size will be broken up into multiple requests to ensure the underlying map engine will generate the requested image size.

Dynamic map engine configuration

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0"
  xmlns:wms="urn:com.cohga.server.map.wms#1.0"
  xmlns:dynamic="urn:com.cohga.server.map.dynamic#1.0">

  <wms:mapengine id="wms1">
    <url>http://server1.local/wms</url>
    <!-- other WMS configuration items -->
  </wms:mapengine>
  <wms:mapengine id="wms2">
    <url>http://server2.local/wms</url>
    <!-- other WMS configuration items -->
  </wms:mapengine>

  <!-- will cycle requests for 'loadbalanced' between 'wms1'
and 'wms2' -->
```

```

    <!-- you can list as many map engines as you need but they must
    all be configured identically -->
    <dynamic:mapengine id="loadbalanced">
        <mapengine>wms1</mapengine>
        <mapengine>wms2</mapengine>
    </dynamic:mapengine>

    <!-- will send requests for 'failover' to 'wms1' until that
    falls over, in which case it'll use 'wms2' -->
    <!-- you can list as many map engines as you need but they must
    all be configured identically -->
    <dynamic:mapengine id="failover">
        <failover>true</failover>
        <mapengine>wms1</mapengine>
        <mapengine>wms2</mapengine>
    </dynamic:mapengine>

    <!-- will send requests for 'alternate' to 'wms1' unless the
    user turns on an aerial photo layer, in which case it will use 'wms2'
    -->
    <!-- you can only list two map engines -->
    <!-- Only available since 2.5.16 -->
    <dynamic:mapengine id="alternate">
        <altlayers>aerial_photo_2001,aerial_photo_2008,
aerial_photo_2015</altlayers>
        <mapengine>wms1</mapengine>
        <mapengine>wms2</mapengine>
    </dynamic:mapengine>

    <!-- any request that has a width greater than 4096 or a
    height greater than 4096 pixels will be broken up into multiple
    requests -->
    <!-- this example is assuming that the wms1 map engine has
    been configured to generate an image less than 4096 x 4096 pixels -->
    <!-- only available since 2.6.4 -->
    <dynamic:mapengine id="stitching">
        <maxwidth>4096</maxwidth>
        <maxheight>4096</maxheight>
        <mapengine>wms1</mapengine>
    </dynamic:mapengine>

    <!-- will redirect all requests for 'redirect' to 'wms1' -->
    <!-- this allows us to use "redirect" as the map engine id in
    the rest of the configuration -->
    <!-- but allow us to switch between different real map
    engines for testing, development, etc -->
    <dynamic:mapengine id="redirect">
        <mapengine>wms1</mapengine>
    </dynamic:mapengine>

</config>

```

In the above examples, any configuration item that previously references `wms1` or `wms2` map engines would now reference `redirect`, `loadbalanced`, `failover`, `alternate` or `stitching` and operate the same as before with the benefits of the particular function

configured. Plus, dynamic map engines can also reference other dynamic map engines, so you can combine load balancing and fail over, for example.

Why would I use the Redirect Dynamic Map Engine?

In most cases, you wouldn't use it. The `redirect` dynamic map engine is probably only useful to the Cohga Support Team because they often have many different map engines that they need to test and don't want to have to update all the relevant configuration files each time they need to switch map engines.

However here is a real life example of it in use. The "main" vector map engine is called "mapengine.vector" and there are a number of other map engines, roughly containing the same layers, called "mapengine.vector.wms.geoserver", "mapengine.vector.wms.qgis", "mapengine.vector.arcgis.soap.100", "mapengine.vector.arcgis.rest.140", etc. By just changing the sub-map engine id it's possible to switch to test a different map engine implementation (e.g. for problem solving a Support issue). In this example, there is also a toc model, called "toc.vector" that has a single toc entry that is changed to the toc model that corresponds to the chosen map engine. This way all the other config files just reference "mapengine.vector" and "toc.vector" making testing different types of map engines easy.

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:dynamic="urn:com.cohga.server.map.dynamic#1.0">

  <?set vector.mapengine=wms.qgis?>

    <dynamic:mapengine id="mapengine.vector">
      <mapengine>mapengine.vector.${vector.mapengine}<
/ mapengine>
    </dynamic:mapengine>

    <toc:model id="toc.vector">
      <entry toc="toc.vector.${vector.mapengine}"/>
    </toc:model>

  </config>
```

Layer Filtering

As of Weave 2.6.6, it's possible to define filters that can be associated with a map layer. The filters are used when the layer is drawn to refine the set of features that will be rendered for the layer. Additionally, a new client panel allows the user to dynamically alter the filter, giving them the ability to dynamically refine the features displayed on the map.

This is similar to the search panel where the user can enter certain criteria to select a set of entities, but in this case, the input the user enters is used to refine the set of features displayed on the map. For example, if the map contains a layer representing the public reserves, a filter could be created for the reserves layer that allows the user to display only those reserves containing a specific facility or facilities, i.e. only those reserves with barbecues and/or public toilets.

The panel to include in the client to allow the user to alter the filters has the id `weave.layerFilter`.

The client UI for the filtering contains a couple of new controls that were not previously available with other input forms, the slider and multi-slider. These controls are useful for layer filtering because they provide an easier way to constrain a numeric value to a certain range. But they're also different in that they force a value to be chosen, that is a filter that uses a slider for a value will always have that value applied when displaying the layer. This can be important when the table behind the layer contains a number of "sets" of data but only one should be displayed at a time, and the column behind the parameter is used to separate out the data into the various sets. An example of this could be the year when the data in the table was generated and the user would only want to see a single year at a time, or for a layer with flood levels and you only want to show a single depth.

In addition to the filter panel, there is also a ToC panel context menu item, `weave.toc.filterLayer`. This can be added to the ToC panel context menu to allow the user to filter a layer directly. It's recommended that the layer filtering capability is provided either in the filter panel or context menu in a client and not both (as providing both could result in display synchronisation issues).

Finally, there is a new flag available for the ToC panel, `showFilterIcon`, which when set to true will place an indicator icon next to a layer in the ToC that currently has a filter applied.

Current limitations

There can be only one filter associated with a particular map layer, this is unlikely to change.

The filtering only affects the display of the layer on the map, it plays no part for example if the user tries to spatially select entities that are associated with that layer (this may change depending upon demand) or when displaying a legend.

Configuration

Namespace

urn:com.cohga.weave.map.filter#1.0

Tags

layer

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier for this filter
label	string	yes	Text to be displayed to the user to represent this filter
mapengine	string	yes	The identifier of the map engine the layer to filter belongs to
layer	string	yes	The identifier of the layer in the map engine this filter applies to

Sub-tags

Name	Type	Cardinality
parameter	urn:com.cohga.weave.map.filter#1.0:parameter	1..n

parameter

Properties

Name	Type	Required	Default	Description
id	string	yes		A unique identifier for the parameter
label	string	yes		The prompt text displayed when user input the parameter value
column	string	yes		The name of the column within the layer that this parameter references.
controlType	'text', 'list', 'radio', 'check', 'multicheck', 'silder', 'multislider'	no	'text'	The UI control to use when displaying the parameter
dataType	'boolean', 'float', 'integer', 'string'	no	'string'	The data type for the parameter
allowBlank	boolean	no	true	Give the user the choice of an empty value in the listbox (as opposed to a null value)
value	any	no		The default value of the parameter (except multislider fields)
dataSet	ref urn:com.cohga.server.data.database#1.0:datadefinition	no		Where to get the values for a listbox
labelColumn	string	no		Column in the datadefinition that supplies the label of the value to show the user
valueColumn	string	no		

				Column in the datadefinition that supplies the value of the value to use in the SQL
scalarparametertype	string	no	'simple'	'simple' or 'multi-value' to determine of more than one value can be selected from a list.
width	integer	no		Set the width of the field
minValue	number	yes - for slider and multislider		The minimum value allowed for a numeric field.
maxValue	number	yes - for slider and multislider		The maximum value allowed for a numeric field.
leftValue	number	no		The initial minimum value for a multislider
rightValue	number	no		The initial maximum value for a multislider
increment	number	no		The increment to use for fields that support it, the units are dependant upon the field type.
trueValue	any	no		The value that equates to "true" in the underlying table, only suitable for checkboxes
falseValue	any	no		The value that equates to "false" in the underlying table, only suitable for checkboxes
decimalPrecision	number	no		The precision of any numeric fields

Sub-tags

Name	Type	Cardinality
list	urn:com.cohga.weave.map.filter#1.0:list	1..1 - only for radio and multicheck fields

list

Properties

Name	Type	Required	Description
value	string	yes	Value to be used in the filter
label	string	yes	Text to be displayed to the user to represent this filter
checked	string	no	Should this check box be initially checked. Only for list within multicheck field

Examples

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0"
  xmlns:mapfilter="urn:com.cohga.weave.map.filter#1.0"
  xmlns:data="urn:com.cohga.server.data.database#1.0">

  <!-- provide the values for the title names list -->
  <data:datadefinition id="titles.names">
    <datasourcedataconnection datasource="main" table="TITLES">
```

```

prefix="DISTINCT">
  <parameter name="title_name" label="Title Name" column="
TITLENAMENAME"/>
  </datasourcedataconnection>
</data:datadefinition>

<!-- create a simple list parameter to filter on title name -->
<mapfilter:layer id="titles">
  <label>Titles</label>
  <mapengine>main</mapengine>
  <layer>titles</layer>
  <parameter id="name" label="Name" controlType="list"
scalarParameterType="multi-value"
  dataSet="titles.names" column="TITLENAMENAME"/>
</mapfilter:layer>

</config>

```

```

<!-- parameter examples -->

<!-- a parameter representing a floating point value between 2.5 and
26 with an increment of 0.5
and default value of 13 -->
<parameter id="level">
  <label>Level (m)</label>
  <controlType>slider</controlType>
  <dataType>float</dataType>
  <column>Level_mGH</column>
  <minValue>2.5</minValue>
  <maxValue>26</maxValue>
  <value>13</value>
  <increment>0.5</increment>
  <decimalPrecision>1</decimalPrecision>
</parameter>

<!-- a parameter representing an integer value between 2011 and 2019
inclusive
with a default minimum of 2015 and default maximum of 2018 -->
<parameter id="study_year">
  <label>Study Year</label>
  <controlType>multisliderr</controlType>
  <dataType>int</dataType>
  <column>Study_year</column>
  <minValue>2011</minValue>
  <maxValue>2019</maxValue>
  <leftValue>2015</leftValue>
  <rightValue>2018</rightValue>
</parameter>

<!-- multiple grouped check boxes. Note in this case all checked
boxes start as checked
but if none of the check boxes are checked all items would be

```

```

displayed as well -->
  <parameter id="location">
    <label>Location</label>
    <controlType>multicheck</controlType>
    <column>Location</column>
    <dataType>string</dataType>
    <list label="Catchment, Todd" value="Todd Catchment" checked="
true"/>
    <list label="River, Adelaide" value="Adelaide River" checked="
true"/>
    <list label="River, McArthur" value="McArthur River" checked="
true"/>
  </parameter>

<!-- single check box. Note in this case since we've set a different
true value
when this check box is checked the filter applied will be
"Location='Adelaide River'"
rather than "Location=true" as would be the case if trueValue were
not set -->
  <parameter id="location">
    <label>Adelaide River Only</label>
    <controlType>check</controlType>
    <column>Location</column>
    <dataType>string</dataType>
    <trueValue>Adelaide River</trueValue>
  </parameter>

<!-- radio button -->
  <parameter id="location">
    <label>Location</label>
    <controlType>radio</controlType>
    <column>Location</column>
    <dataType>string</dataType>
    <value>Adelaide River</value>
    <list label="Catchment, Todd" value="Todd Catchment"/>
    <list label="River, Adelaide" value="Adelaide River"/>
    <list label="River, McArthur" value="McArthur River"/>
  </parameter>

<!-- list box -->
  <parameter id="location">
    <label>Location</label>
    <controlType>list</controlType>
    <dataSet>locations</dataSet>
    <column>Location</column>
    <dataType>string</dataType>
  </parameter>

<!-- a basic text box. Note this format is discouraged as the user
has no way of knowing
if the value they're entering is valid and may result in nothing
being displayed -->
  <parameter id="location">

```

```

    <label>Location</label>
    <controlType>text</controlType>
    <column>Location</column>
    <dataType>string</dataType>
  </parameter>

```

Client

```

<client:config id="filtering">
  <title>Example Filtering client</title>

  <!-- other config here -->

  <view id="weave.layerFilter">
    <location>east</location>
  </view>

  <!-- other config here -->

  <view id="com.cohga.html.client.map.tocView">
    <location>west</location>
    <label>Layers</label>
    <showFilterIcon>true</showFilterIcon>
    <contextmenu>
      <item action="weave.toc.filterLayer"/>
    </contextmenu>
  </view>

</client:config>

```

Table Of Contents

It is possible to arrange the table of contents that appears on the client by setting up a TOC model.

This model allows you to categorize and organize the layers that are available from a map engine.

For historical reasons all top level toc models, i.e. those that are directly included in a client config, should start their id with `toc.`, e.g.

```

<toc:model id='toc.main'>
rather than
<toc:model id='main.toc'>

```

Nesting Update

A recent update to the toc model, since release 2.4.10, allows a toc model to be directly included within another toc model. Previously it was only possible to include another toc model within a group in a toc model.

For example, a previous nested toc model would look like this:

```

<toc:model id="subtoc">
  <mapengine>default</mapengine>
  <entry id="layer1" label="Layer 1" layer="layer1"/>
</toc:model>

```



```
<toc:model id="toc.main">
  <mapengine>default</mapengine>
  <entry id="group1" label="Group 1" toc="subtoc"/>
  <entry id="layer2" label="Layer 2" layer="layer2"/>
</toc:model>
```

which would result in a toc like:

```
+Group 1
  Layer 1
  Layer 2
```

The following is now possible:

```
<toc:model id="subtoc"> <!-- same as previous example -->
  <mapengine>default</mapengine>
  <entry id="layer1" label="Layer 1" layer="layer1"/>
</toc:model>

<toc:model id="toc.main">
  <mapengine>default</mapengine>
  <entry id="subtoc1" toc="subtoc"/> <!-- note no label set -->
  <entry id="layer2" label="Layer 2" layer="layer2"/>
</toc:model>
```

which would result in a toc like:

```
Layer 1
Layer 2
```

The original structure can now also be accomplished like this:

```
<toc:model id="subtoc">
  <mapengine>default</mapengine>
  <entry label="Group 1">
    <entry id="layer1" label="Layer 1" layer="layer1"/>
  </entry>
</toc:model>

<toc:model id="toc.main"> <!-- same as previous example -->
  <mapengine>default</mapengine>
  <entry id="subtoc1" toc="subtoc"/>
  <entry id="layer2" label="Layer 2" layer="layer2"/>
</toc:model>
```

Virtual Layers

There can be special virtual layers added to a map engine by additional bundles that provide extra functionality (e.g. an acetate layer or selection drawing), and these layers need to be taken into consideration when creating a table of contents for a map. Virtual layers will be covered later in this document.

Lockable layers

You just need to create a duplicate of the layer with the word UNLOCKED at the start. For example, if you have a layer called `property` in the mxd, arcims or wms you would create another called `UNLOCKEDproperty`. There is no need to add the unlocked layer to the toc mode. When the user right clicks the layer, Weave checks to see if this layer is available in the mapengine. If it is, the Unlock Layer menu item will be enabled.

You can add it to the toc view contextmenu with the following.

```
<view id='com.cohga.html.client.map.tocView'>
  <label>Layers</label>
  <location>west</location>
  <contextmenu>
    <item action="weave.toc.lockLayer" text="Lock Layer"/>
  </contextmenu>
</view>
```

Namespace

urn:com.cohga.server.map.toc#1.0

Tags

model

Properties

Name	Type	Required	Description
id	string	yes	unique identifier, should start with 'toc.' for top level toc models
mapengine	ref urn:com.cohga.server.map.toc#1.0:mapengine	no	The default map engine to associate with the entries included in this toc model if they don't set their own map engine. This value is not required if the this toc model only includes other toc models, it is required if the toc model includes layers or groups that don't set their own map engine

Sub-tags

Name	Type	Cardinality
entry	urn:com.cohga.server.map.toc#1.0:entry	1..n

Content

None

entry

Properties (for a 'layer')

Name	Type	Required	Description
id	string	no	Not "required" but you should set it. It uniquely identifies an entry when saving a toc model, for example in a bookmark, which make your life a lot easier if you ever want to re-arrange or rename your map layers, if it is not specified Weave will generate one for you (which could result in there being multiple toc entries with the same id, which can cause problems).

			It should be set it to a unique value for every entry, it doesn't really matter what the value is, just make it unique across all toc model entries (a combination of map engine id and layer name would work if you have a lot of layers). This value should never be changed once it has been set, or any bookmarks that reference this entry will be invalidated.
layer	string	yes	The identifier for the layer within the map engine
label	string	yes	Identifier displayed to the user for the entry
map engine	ref urn:com.cohga.server.map.toc#1.0:mapEngine	no	Override the default map engine for this layer
checked	boolean	no	Determines the entry should be turned on or off by default
visible	boolean	no	Should the entry be displayed in the TOC at all
description	string	no	A description to display when the mouse hovers over the layer
entity	ref urn:com.cohga.server.entity#1.0:entity	no	The id of an entity that should be associated with this layer, by associating an entity with a layer additional functionality is available.
style	string	no	For map engines that support multiple styles for a layer this value can be used to change the default style. Note that adding the <code>weave.toc.styleLayer</code> action to the toc panel context menu allows the user to switch styles on the fly.
acl	ref urn:com.cohga.server.acl#1.0:acl	no	A reference to an ACL to attach to the layer
minScale	number	no	Specifies a minimum scale below which the entry in the toc will not be enabled. Note that this may not affect the display of the layer in the map, just the entry in the toc panel.
maxScale	number	no	Specifies a maximum scale above which the entry in the toc will not be enabled. Note that this may not affect the display of the layer in the map, just the entry in the toc panel.

Properties (for a 'toc')

Name	Type	Required	Description
toc	ref urn:com.cohga.server.map.toc#1.0:acl	yes	A toc model to include in this one
label	string	no	The label for a group to include the toc within, if not set the toc will be included directly within the parent toc model
acl	ref urn:com.cohga.server.acl#1.0:acl	no	A reference to an ACL to attach to the toc

Properties (for a 'group')

Name	Type	Required	Description
id	string	no	Not "required" but you should set it. It uniquely identifies an entry when saving a toc model, for example in a bookmark, which make your life a lot easier if you ever want to re-arrange or rename your map layers, if it is not specified Weave will generate one for you (which could result in there being multiple toc entries with the same id, which can cause problems). It should be set it to a unique value for every entry, it doesn't really matter what the value is, just make it unique across all toc model entries. This value should never be changed once it has been set, or any bookmarks that reference this entry will be invalidated.
label	string	yes	Identifier displayed to the user for the group

mapengine	ref urn:com.cohga.server.map.toc#1.0:mapEngine	no	Override the default map engine for this group
checked	boolean	no	Determines the group should be turned on or off by default
description	string	no	A description to display when the mouse hovers over the group
expandable	boolean	no	Will the user be able to expand/collapse the group in the TOC
expanded	boolean	no	Will the group start out already expanded
exclusive	boolean	no	Only one item within the group can be checked at a time
folder	boolean	no	Should the expand/collapse control be hidden for this group
entity	ref urn:com.cohga.server.entity#1.0:entity	no	The id of an entity that should be associated with this group, by associating an entity with a group additional functionality is available.
acl	ref urn:com.cohga.server.acl#1.0:acl	no	A reference to an ACL to attach to the group
minScale	number	no	Specifies a minimum scale below which the entry in the toc will not be enabled. Note that this may not affect the display of the layer in the map, just the entry in the toc panel.
maxScale	number	no	Specifies a maximum scale above which the entry in the toc will not be enabled. Note that this may not affect the display of the layer in the map, just the entry in the toc panel.

Sub-tags (for a 'group')

Name	Type	Cardinality
entry	urn:com.cohga.server.map.toc#1.0:entry	1..n

Content

None

Examples

A Simple ToC

```
<toc:model id="toc.basic">
  <mapengine>mapengine.wms</mapengine>
  <entry id="g_roads" label="Roads" expandable="false">
    <entry id="mainroads" layer="mainroads" label="Main
Roads" />
    <entry id="roads" layer="roads" label="Roads" />
  </entry>
  <entry id="buildings" layer="buildings" label="Council
Buildings" checked="false" />
  <entry id="easements" layer="easement" label="Easements" />
  <entry id="water" label="Water" expandable="true" folder="
true">
```

```

        <entry id="drainage" layer="drainage" label="
Drainage" checked="false"/>
        <entry id="pipes" label="Pipes" checked="false"
expandable="true">
            <entry id="waterpipe" layer="water_pipe"
label="Water" checked="false"/>
            <entry id="sewerpipe" layer="sewer_pipe"
label="Sewer" checked="false"/>
        </entry>
        <entry id="streams" label="Streams" checked="false"
expandable="false">
            <entry id="hydro" layer="hydro" label="Hydro"
/>
            <entry id="mainhydro" layer="main_hydo"
label="Main Hydro"/>
        </entry>
    </entry>
    <entry id="g_property" label="Property" checked="true"
expandable="false">
        <entry id="property_dissolve" layer="
property_dissolve" label="Property Dissolve"/>
        <entry id="property" layer="property" label="Property"
/>
    </entry>
    <entry id="suburb" layer="suburb" label="Suburbs"/>
    <entry id="bounds" layer="mccbound" label="Municipal Boundary"
/>
</toc:model>

```

A ToC included in another ToC

```

<toc:model id="aerial">
    <mapengine>raster</mapengine>
    <entry id="aerial2006" label="2006" layer="aerial2006"/>
    <entry id="aerial1996" label="1986" layer="aerial1996"
checked="false"/>
    <entry id="aerial1986" label="1966" layer="aerial1986"
checked="false"/>
</toc:model>

<toc:model id="toc.main">
    <mapengine>vector</mapengine>
    <entry id="boundary" label="Boundaries" layer="boundary"/>
    <entry id="property" label="Properties" layer="property"/>
    <entry id="aerial_photography" label="Aerial Photography"
exclusive="true">
        <toc>aerial</toc>
    </entry>
</toc:model>

```

A selection ToC directly included in another ToC

```

<toc:model id="selection">
  <mapengine>selection</mapengine>
  <entry id="selections" label="Selection" exclusive="true">
    <entry id="selection.active" label="Active" layer="_selection.active"/>
    <entry id="selection.all" label="All" layer="_selection.all" checked="false"/>
    <entry id="selection.other" label="Other" checked="false">
      <entry id="selection.property" label="Properties" layer="_selection.property"/>
      <entry id="selection.boundary" label="Boundaries" layer="_selection.boundary"/>
    </entry>
  </entry>
</toc:model>

<toc:model id="toc.main">
  <mapengine>vector</mapengine>
  <entry toc="selection"/>
  <entry id="boundary" label="Boundaries" layer="boundary" entity="boundary"/>
  <entry id="property" label="Properties" layer="property" entity="property"/>
  <entry id="aerial_photography" label="Aerial Photography" exclusive="true" toc="aerial"/>
</toc:model>

```

A ToC with an ACL where one group of users get a 5cm aerial photo layer and everyone else 50cm.

```

<toc:model id="aerial">
  <mapengine>raster</mapengine>
  <entry id="aerial_5cm" label="Aerial Photo" layer="aerial_5cm" acl="planners"/>
  <entry id="aerial_50cm" label="Aerial Photo" layer="aerial_50cm" acl="!planners"/> <!-- Note the use of ! to invert the acl -->
</toc:model>

<toc:model id="toc.main">
  <mapengine>vector</mapengine>
  <entry id="boundary" label="Boundaries" layer="boundary"/>
  <entry id="property" label="Properties" layer="property"/>
  <entry id="aerial" toc="aerial"/>
</toc:model>

```

Selection Layers

One group of virtual layers that's almost always available are those representing the users current selections. When selections are enabled in Weave each map engine (unless configured otherwise) gets additional layers added to it, one representing the selection for the active entity, one representing the selection for all entities, and one each representing the current selection for the individual entities.

Because we're looking at a table of contents at the moment I'll assume that you're interested on how to add these layers to a table of contents (as opposed to other ways of having these layers displayed).

And to do this you need to add an entry to your toc model for each of the virtual layers you wish to enable. The virtual layers themselves are given special layer ids and just need to be added like any other layer.

The simplest way to add the selection to the map is to turn on the layer that represents the selection for the active entity, and to do that you would add the following layer definition to your toc model:

```
<entry id="selection.active" label="Selection" layer="
_selection.active"/>
```

Note that you can also use just "_selection" as an alternative for "_selection.active", and the values used for the labels in these entries is not important and can be set to anything you think appropriate.

The above example will add an entry to the table of contents allowing the user to turn on and off the active selection.

If you wanted to always draw the current selection and not allow the user the option of turning it off the you would use the following:

```
<entry id="selection.active" label="Selection" layer="
_selection.active" checked="true" visible="false"/>
```

An alternative to displaying just the active selection is to display the selections for every layer (keep in mind this could be slow). And, this is done using the "_selection.all" virtual layer, for example:

```
<entry id="selection.all" label="Selections" layer="
_selection.all"/>
```

This will draw the selection for the current entity in a more pronounced style and the other entities in a less pronounced style, unless changed in the map engine configuration.

The other virtual selection layers are given a layer id that relates to the entity id that they're associated with, for example to add a selection layer for a 'road' entity you would use:

```
<entry id="selection.road" label="Road Selection" layer="
_selection.road"/>
```

And this can be applied for as many entities as you wish to appear in the table of contents.

You can use groups in the table of contents to setup a quite complex selection drawing structure, allowing the user to choose between drawing the current selection, drawing all selections or drawing selections for individual entities. This would be done as follows:

```
<entry label="Selection" exclusive="true">
  <entry id="selection.active" layer="_selection.
active" label="Active" checked="true"/>
  <entry id="selection.all" layer="_selection.all"
label="All" checked="false"/>
  <entry id="selection.other" label="Selected" checked="
false">
    <entry id="selection.property" layer="
_selection.properties" label="Properties" checked="false"/>
    <entry id="selection.road" layer="_selection.
roads" label="Roads" checked="false"/>
    <entry id="selection.suburb" layer="
```

```

_selection.suburbs" label="Suburbs" checked="false"/>
      </entry>
    </entry>

```

This way the user will have full control over which selections are drawn.

Data Source

A Data Source provides the information that the Weave server requires to connect to a databases.

Weave uses [JDBC](#), a standard Java database API, to interact with databases and the data source configuration is used to provide Weave with the required JDBC settings to be able to connect and interact with a database.

Most of the settings required for a data source are specific to the database, and JDBC driver, that you are connecting to and the documentation for the driver should be consulted to determine the correct values, especially for the `driver` and `url` parameters.

Time filtering with SQL Server

If when filtering data from a SQL Server you can resolve this by setting `sendTimeAsDatetime` to `false` as a parameter in the connection URL, e.g.

```

<url>jdbc:sqlserver://sqls2014:1433;databaseName=GIS;sendTimeAsDatetime=false</url>

```

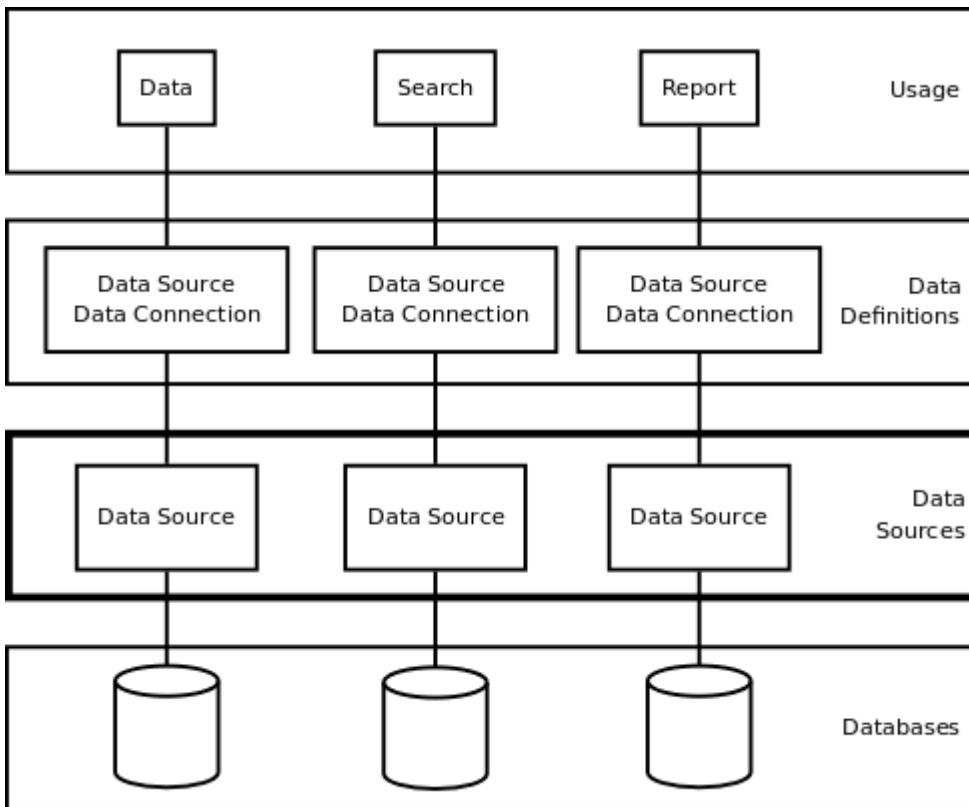
Timestamps with Oracle

The Oracle JDBC driver by default does not correctly follow the JDBC specification when it comes to handling timestamp columns, to force the Oracle JDBC driver to be compliant you should set the system property `oracle.jdbc.J2EE13Compliant` to `true`.

Weave will try and work around the problem (and generate a warning pointing to this page) but at the cost of performance overhead.

Generally, only read access is required when connecting to the database, some custom modules may require write access.

Connecting pooling can be enabled for the data source, so that rather than connecting to the database each time it needs to perform an operation the Weave server will instead create a collection of connections and keep them in a pool to be used when required.



Namespace

urn:com.cohga.server.datasource.jdbc#1.0

Tags

datasource

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
driver	string	no	The class name of the JDBC driver. There are different JDBC drivers for different databases, and even more than one for some databases, so this value will depend upon what JDBC driver you're using. This value is required if jndi is not set.
url	string	no	The connection URL. This is driver specific and will depend upon the database you're connecting to an the JDBC driver you're using. This value is required if jndi is not set.
username	string	no	The userid used to connect to the database.
password	string	no	The password of the userid used to connect to the database
jndi	string	no	A JNDI resource name for a datasource created by the underlying application server. If specified the other settings here are ignored.

Sub-tags

Name	Type	Cardinality
pool	ref urn:com.cohga.server.pool#1.0	0..1

Content

None

Notes

- The database connection can be created in one of two ways. Either Weave is responsible for connecting to the database and managing the connection, or the underlying application server (Jetty, Tomcat, WebSphere, etc) is responsible for the connection.
- If Weave is to be responsible for managing the connection then the appropriate JDBC driver must be installed in the `platform\workspace\jdbc` directory, and at least the `driver` and `url` settings must be specified in the Weave datasource configuration.
- If the application server is responsible for managing the connections then the appropriate JDBC driver must be installed in the application server and a JNDI database connection created in the application server, and the `jndi` value set to the resource name for the application server datasource in the Weave datasource configuration.
- The `pool` tag also support a `maxAgeMillis` property which specifies a limit on how long the database connection should be kept open for, note however that this check only occurs during connection validation, so `testOnBorrow`, `testOnReturn` and/or `testWhileIdle` must be set for the pool.

Examples

Connecting to Oracle

```
<jdbc:datasource id="datasource.oradev">
  <driver>oracle.jdbc.driver.OracleDriver</driver>
  <url><![CDATA[ jdbc:oracle:thin:@1521:oradev:ORCL ]]></url>
  <username>read</username>
  <password>ManD0lAuh</password>
```

```

    <pool:pool>
      <maxActive>15</maxActive>
      <minIdle>2</minIdle>
      <maxIdle>2</maxIdle>
      <testOnBorrow>true</testOnBorrow>
      <timeBetweenEvictionRunsMillis>60000<
/timeBetweenEvictionRunsMillis>
      <minEvictableIdleTimeMillis>1200000<
/minEvictableIdleTimeMillis>
      <whenExhaustedAction>grow</whenExhaustedAction>
    </pool:pool>
  </jdbc:datasource>

```

Connecting to Postgres

```

<jdbc:datasource id="datasource.postgis">
  <driver>org.postgresql.Driver</driver>
  <url><![CDATA[ jdbc:postgresql://postgis21:5432/mann ]]></url>
  <username>gis</username>
  <password>gispassword</password>
  <schema>mann</schema>
  <pool:pool>
    <minIdle>2</minIdle>
    <maxIdle>4</maxIdle>
    <testOnBorrow>true</testOnBorrow>
    <testOnReturn>true</testOnReturn>
    <testWhileIdle>true</testWhileIdle>
    <timeBetweenEvictionRunsMillis>15000<
/timeBetweenEvictionRunsMillis>
    <minEvictableIdleTimeMillis>120000<
/minEvictableIdleTimeMillis>
    <whenExhaustedAction>grow</whenExhaustedAction>
  </pool:pool>
</jdbc:datasource>

```

Connecting to SQL Server (using the Microsoft SQL Server JDBC driver)

```

<jdbc:datasource id="sqlserver2012">
  <driver>com.microsoft.sqlserver.jdbc.SQLServerDriver</driver>
  <url><![CDATA[ jdbc:sqlserver://sqls2012dev;databaseName=GIS ]]>
></url>
  <username>gisadmin</username>
  <password>ENCTXTJPCQPAJUCKKMABFJEBZJAKFCXGME</password>
  <pool:pool>
    <maxActive>4</maxActive>
    <minIdle>2</minIdle>
    <maxIdle>2</maxIdle>
    <testOnBorrow>true</testOnBorrow>
    <testOnReturn>true</testOnReturn>
    <testWhileIdle>true</testWhileIdle>
    <timeBetweenEvictionRunsMillis>60000<

```

```

/timeBetweenEvictionRunsMillis>
    <minEvictableIdleTimeMillis>1200000<
/minEvictableIdleTimeMillis>
    <whenExhaustedAction>block</whenExhaustedAction>
</pool:pool>
</jdbc:datasource>
    
```

Connecting to a directory of dBase tables

```

<jdbc:datasource id="datasource.dbf">
    <driver>com.hxtt.sql.dbf.DBFDriver</driver>
    <url><![CDATA[ jdbc:dbf:///u1/data/dbf/ ]]></url>
</jdbc:datasource>
    
```

Exposing a JNDI resource declared elsewhere as a data source to Weave

```

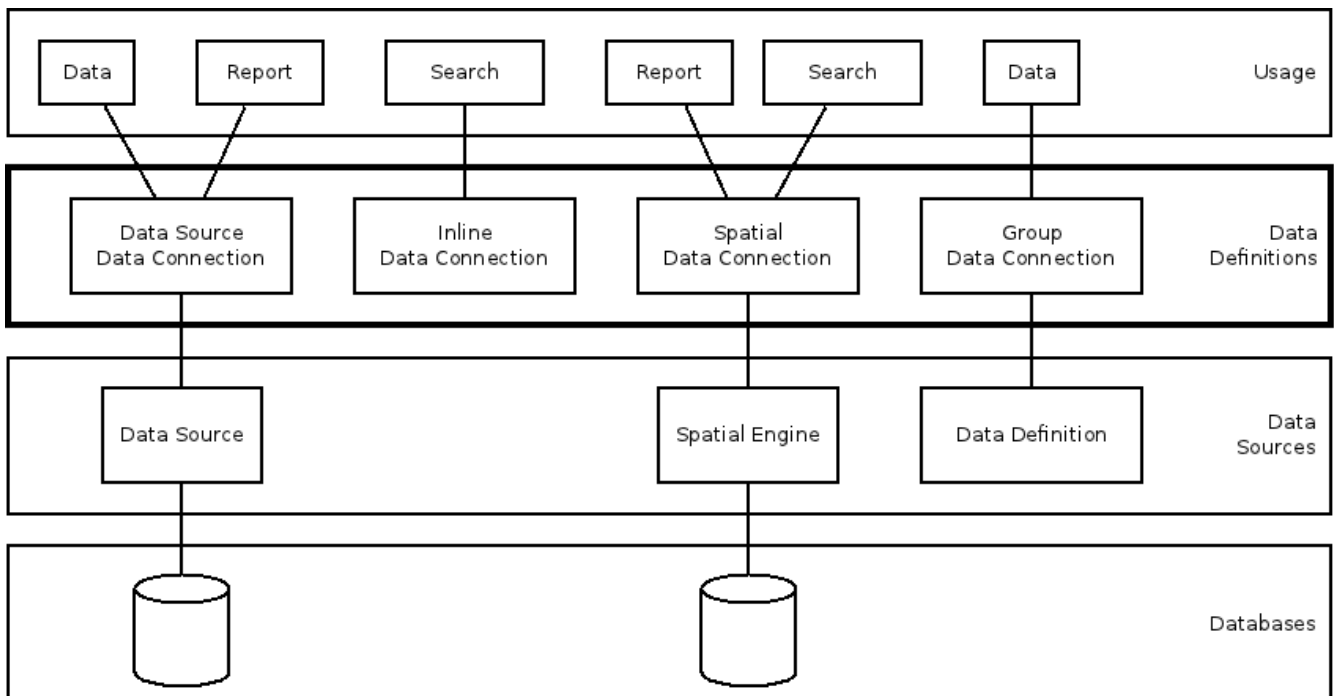
<jdbc:datasource id="datasource.jndi">
    <jndi>java:comp/env/jdbc/uat</jndi>
</jdbc:datasource>
    
```

Data Definition

A Data Definition represents a set of data that the Weave server can use in various ways, for example to provide a list of values for the user when entering data, or as a tabular dataset to display textual data about a selected entity. Basically anywhere that Weave needs some textual data it will retrieve it via a data definition.

There are different types of data definition depending upon where the data is retrieved from, including data definitions that are virtual data definitions that use other data definitions to supply the data which they then filter in some way and provide as a new data definition.

The data definition itself is just a container for a data connection, the data definition provides an id and a container for a data connection and it's the data connection that describes the source of the underlying data and how that data is obtained.



Namespace

urn:com.cohga.server.data.database#1.0

Tags

datadefinition

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier for referencing the data definition

Sub-tags

Name	Type	Cardinality
datasourcedataconnection	urn:com.cohga.server.data.database#1.0:datasourcedataconnection	0..1
inlinedataconnection	urn:com.cohga.server.data.database#1.0:inlinedataconnection	0..1
spatialdataconnection	urn:com.cohga.server.data.database#1.0:spatialdataconnection	0..1
groupdataconnection	urn:com.cohga.server.data.database#1.0:groupdataconnection	0..1

Content

None

Notes

- Only one of the data connection types can be included in a data definition.

Examples

Sample data definition using a database

```
<data:datadefinition id="roadDetails">
  <datasourcedataconnection datasource="datasource.main" table="
ROADS" key="RD_NAME" prefix="DISTINCT">
    <from table="PLANS"/>
    <where clause="PLANS.ID=ROADS.PLAN_ID"/>
    <parameter name="name" label="Name" column="RD_NAME"/>
    <parameter name="suffix" label="Suffix" column="
RD_SUFFIX"/>
    <parameter name="type" label="Type" column="RD_TYPE"/>
    <parameter name="altname" label="Alt. Name" column="
ALT_NAME"/>
    <parameter name="altsuffix" label="Alt. Suffix"
column="ALT_SUFFIX"/>
    <parameter name="alttype" label="Alt. Type" column="
ALT_TYPE"/>
  </datasourcedataconnection>
</data:datadefinition>
```

A simple inline data definition

```

<data:datadefinition id="connectionType">
  <inlinedataconnection>
    <parameter type="string" name="key" label="Key" />
    <parameter type="string" name="label" label="Label" />
    <row>
      <cell>U</cell>
      <cell>Unverified</cell>
    </row>
    <row>
      <cell>V</cell>
      <cell>Verified</cell>
    </row>
  </inlinedataconnection>
</data:datadefinition>

```

Aggregate Data Connection

An aggregate data connection allows you to pull data from multiple tables like an SQL union operation would do. That is the data generated for the data definition would be the rows from multiple tables returned one after the other. The aggregate data connection is constructed from a list of other existing data definitions which must all have exactly the same parameters.

aggregatedataconnection

Properties

Name	Type	Cardinality	Description
datadefinition	urn:com.cohga.server.data.database#1.0	1..n	Id of a data definitions that will supply the source data that this data definition will provide

Sub-tags

None

Content

None

```

<data:datadefinition id="list.wells">
  <aggregatedataconnection>
    <datadefinition>list.wells.1</datadefinition>
    <datadefinition>list.wells.2</datadefinition>
  </aggregatedataconnection>
</data:datadefinition>

```

Datasource Data Connection

datasourcedataconnection

Properties

Name	Type	Required	Description
datasource	ref urn:com.cohga.server.datasource.jdbc#1.0:datasource	yes	reference to the data source that this data definition should use to generate its data

table	string	yes	The name of the table that provides the data
key	string	no	If this data definition is generating data relating to an entity then this is the column name that contains the entities id
prefix	string	no	The prefix value will be inserted into the generated SQL statement immediately after the SELECT, for example DISTINCT
trim	boolean	no	If <code>true</code> the generated SQL will ensure that the id's are compared against a trimmed version of the id stored in the database. This helps with systems like Pathway, where some tables store padded id's
keytable	string	no	An optional database table that will be used to store entity keys temporarily when performing the required SQL to generate the data for this data connection. If this option isn't set then the SQL used to generate the data will create the data in batches, which can cause issues with sorting, optionally you can create a simple two column table, with an index on the userid column, and set this value to the name of this table, then Weave will insert the entity id's into this table temporarily and use this table in a SQL join when generating the data. This will resolve any sorting issues, and is more important if caching is also disabled for the data connection.
keycolumn	string	no	The column in <code>keytable</code> that contains the entity id, default is <code>key</code>
usercolumn	string	no	The column in <code>keytable</code> that contains the userid, default is <code>userid</code>

Sub-tags

Name	Type	Cardinality
parameter	urn:com.cohga.server.data.database#1.0:parameter	0..n
cache	urn:com.cohga.server.cache#1.0:cache	0..1
from	urn:com.cohga.server.data.database#1.0:from	0..n
where	urn:com.cohga.server.data.database#1.0:where	0..n
sort	urn:com.cohga.server.data.database#1.0:sort	0..n
join	urn:com.cohga.server.data.database#1.0:join	0..n
options	urn:com.cohga.server.data.database#1.0:options	0..1

Content

None

Notes

- The cache settings are only used for data source data connections that have a key specified, since it's the key that's used as the cache index
- If no `cache` tag is specified then the data will still be cached, but it will use the default cache configuration, to disable caching you need to set `disable` to `true` inside the cache definition (see example below).
- You can completely disable the caching (for all data definitions that don't have cache setting explicitly set) by setting the startup parameter `weave.cache.default` to `true` (in `startup.cmd`, `startup.sh` and/or `weave-service.conf`).
- If no parameters are specified then the server will generate parameters based on the columns in the underlying table, in this case you should also **not** include any `from` tags since joins are not currently supported, `where` tags are ok though.

parameter

Properties

Name	Type	Req uired	Default	Description
column	string	yes		The name of the column within the table that this parameter references (can be an sql function also)
type	'string', 'numeric', 'date' or 'url'	no	'string'	An indicator of how the data should be displayed on the client
name	string	no	lowercase value of column	A unique identifier for the parameter
label	string	no	"pretty" value of column	A user displayable label for the column
text	string	no		Only when type is 'url'. Specifies text to be displayed to the user instead of the actual url contents

textcolumn	string	no		Only when type is 'url'. Specifies the column that contains text to be displayed to the user instead of the actual url contents
format	string	no	"date", "time" or "datetime"	When the type attribute is date this value can force the value to be formatted as as specific date type.

Sub-tags

Name	Type	Cardinality
from	urn:com.cohga.server.data.database#1.0:from	0..n
where	urn:com.cohga.server.data.database#1.0:where	0..n

Content

None

Notes

- If no label is specified then it will be generated by formatting the name, unless no name is supplied, then it will be generated by formatting the column
- Formatting involves converting any _'s to spaces, converting the first letter and any letter after a space to upper case and converting every other letter to lower case, e.g. "BOMB_DISPOSAL_METHOD" becomes "Bomb Disposal Method"
- Only one of text or textcolumn should be specified when type is 'url'

from

Properties

Name	Type	Required	Description
table	string	yes	An additional table to include in the generated SQL

Sub-tags

None

Content

None

where

Properties

Name	Type	Required	Description
clause	string	yes	An additional clause to include in the generated SQL
uppercase	boolean	no	If the clause uses parameter substitution should the value be converted to upper-case before being substituted

Sub-tags

None

Content

None

sort

Properties

Name	Type	Required	Description
parameter	string	yes	Name of parameter to sort on
direction	string	no	ASC or DESC, ASC is default

Sub-tags

None

Content

None

join

Only available in 2.5.11 or later

properties

Name	Type	Required	Description
table	string	yes	The table to join to
on	string	yes	The clause used to join the table
type	string	no	The type of join to use
clause	string	no	Additional clauses that can be added to the join

options

Properties

Name	Type	Required	Description
rpadd	number	no	If the key value needs to be padded with spaces on the right to make it match the database, this specifies the total length of the final padded value. For example <code>rpadd = 10</code> would change a key value from "value" to "value...." (where each . is a space).
lpadd	number	no	If the key value needs to be padded with spaces on the left to make it match the database, this specifies the total length of the final padded value. For example <code>lpadd = 10</code> would change a key value from "value" to "....value" (where each . is a space).
trim	boolean	no	If a key value needs to be trimmed of spaces before being used setting this to <code>true</code> will cause that to be done.
maxrows	number	no	Limit the number of rows that will be extracted from the database when generating the data.
failOnSubstitutionError	boolean	no	If set to <code>true</code> , or not set at all, this flag will force the generation of the data to fail if there is an issue with performing parameter substitution. If this is set to <code>false</code> then a where clause that fails parameter substitution will be ignored. The default value for this flag if it is not set is <code>true</code> .
timeout	number	no	The number of milliseconds before the SQL should be cancelled. Default 120000, can be disabled by setting to 0, can be changed globally by setting the system property <code>weave.data.timeout</code> . Also, the system property <code>weave.data.minimumtimeout</code> specifies a minimum value, currently 5000, that if the timeout is set greater than 0 but less than this value it will be increased to be this value.

Sub-tags

None

Content

None

Examples

Basic data definition where output is based on columns in the underlying table

```
<data:datadefinition id="roadDetails">
  <datasourcedataconnection datasource="datasource.main" table="
ROADS" key="RD_NAME" prefix="DISTINCT"/>
</data:datadefinition>
```


In the following examples the || operator is used to concatenate strings, this operator is Oracle specific, and may require using a different operator for other databases, for example +

Sample data definition with caching parameters explicitly set

```
<data:datadefinition id="roadDetails">
  <datasourcedataconnection datasource="datasource.main" table="
ROADS" key="RD_NAME" prefix="DISTINCT">
    <from table="PLANS"/>
    <where clause="PLANS.ID=ROADS.PLAN_ID"/>
    <parameter name="name" label="Name" column="RD_NAME"/>
    <parameter name="suffix" label="Suffix" column="
RD_SUFFIX"/>
    <parameter name="type" label="Type" column="RD_TYPE"/>
    <parameter name="altname" label="Alt. Name" column="
ALT_NAME"/>
    <parameter name="altsuffix" label="Alt. Suffix"
column="ALT_SUFFIX"/>
    <parameter name="alttype" label="Alt. Type" column="
ALT_TYPE"/>
    <parameter type="integer" name="class" label="Class"
column="CLASS_CODE"/>
    <parameter type="url" name="plan" label="Plan"
column="'http://imgsvr:8080/gis/documents/plan_' || PLANS.PLAN_CODE || '.
pdf'" text="Open"/>
    <cache>
      <maxElementsInMemory>500</maxElementsInMemory>
      <eternal>>false</eternal>
      <timeToIdleSeconds>60</timeToIdleSeconds>
      <timeToLiveSeconds>300</timeToLiveSeconds>
      <overflowToDisk>>true</overflowToDisk>
      <maxElementsOnDisk>20000</maxElementsOnDisk>
      <diskPersistent>>false</diskPersistent>
      <diskExpiryThreadIntervalSeconds>120<
/diskExpiryThreadIntervalSeconds>
      <memoryStoreEvictionPolicy>LRU<
/memoryStoreEvictionPolicy>
    </cache>
  </datasourcedataconnection>
</data:datadefinition>
```

Sample data definition with caching disabled

```
<data:datadefinition id="roadDetails">
  <datasourcedataconnection datasource="datasource.main" table="
ROADS" key="RD_NAME" prefix="DISTINCT">
```

```

        <from table="PLANS"/>
        <where clause="PLANS.ID=ROADS.PLAN_ID"/>
        <parameter name="name" label="Name" column="RD_NAME"/>
        <parameter name="suffix" label="Suffix" column="
RD_SUFFIX"/>
        <parameter name="type" label="Type" column="RD_TYPE"/>
        <parameter name="altname" label="Alt. Name" column="
ALT_NAME"/>
        <parameter name="altsuffix" label="Alt. Suffix"
column="ALT_SUFFIX"/>
        <parameter name="alttype" label="Alt. Type" column="
ALT_TYPE"/>
        <parameter type="integer" name="class" label="Class"
column="CLASS_CODE"/>
        <parameter type="url" name="plan" label="Plan"
column="'http://imgsvr:8080/gis/documents/plan_' || PLANS.PLAN_CODE || '.
pdf'" text="Open"/>
        <cache disable="true"/>
    </datasourcedataconnection>
</data:datadefinition>

```

Sample data definition with caching parameters supplied by a cache configuration (so the cache definition can be reused)

```

<data:cache id="test.cache">
    <maxElementsInMemory>500</maxElementsInMemory>
    <eternal>false</eternal>
    <timeToIdleSeconds>60</timeToIdleSeconds>
    <timeToLiveSeconds>300</timeToLiveSeconds>
    <overflowToDisk>true</overflowToDisk>
    <maxElementsOnDisk>20000</maxElementsOnDisk>
    <diskPersistent>false</diskPersistent>
    <diskExpiryThreadIntervalSeconds>120<
/diskExpiryThreadIntervalSeconds>
    <memoryStoreEvictionPolicy>LRU</memoryStoreEvictionPolicy>
</data:cache>

<data:datadefinition id="roadDetails">
    <datasourcedataconnection datasource="datasource.main" table="
ROADS" key="RD_NAME" prefix="DISTINCT">
        <from table="PLANS"/>
        <where clause="PLANS.ID=ROADS.PLAN_ID"/>
        <parameter name="name" label="Name" column="RD_NAME"/>
        <parameter name="suffix" label="Suffix" column="
RD_SUFFIX"/>
        <parameter name="type" label="Type" column="RD_TYPE"/>
        <parameter name="altname" label="Alt. Name" column="
ALT_NAME"/>
        <parameter name="altsuffix" label="Alt. Suffix"
column="ALT_SUFFIX"/>
        <parameter name="alttype" label="Alt. Type" column="
ALT_TYPE"/>

```

```

        <parameter type="integer" name="class" label="Class"
column="CLASS_CODE"/>
        <parameter type="url" name="plan" label="Plan"
column="'http://imgsvr:8080/gis/documents/plan_' || PLANS.PLAN_CODE || '.
pdf'" text="Open"/>
        <cache id="test.cache"/>
    </datasourcedataconnection>
</data:datadefinition>

```

A data definition for supplying a list of values (no key is defined), probably for a cascading input parameter for a search

```

<data:datadefinition id="suburbRoads">
    <datasourcedataconnection datasource="datasource.main" table="
PROPERTY" prefix="DISTINCT">
        <parameter name="suburb" label="Suburb" column="
PRSUB_NAME"/>
        <parameter name="roadname" label="Road Name" column="
PRROD_NAME"/>
    </datasourcedataconnection>
</data:datadefinition>

```

Another data definition for supplying a list of values, probably for a triple field cascading input parameter for a search

```

<data:datadefinition id="suburbRoadType">
    <datasourcedataconnection datasource="datasource.main" table="
PROPERTY" prefix="DISTINCT">
        <parameter name="suburb" label="Suburb" column="
PRSUB_NAME"/>
        <parameter name="roadname" label="Road Name" column="
PRROD_NAME"/>
        <parameter name="roadtype" label="Road Type" column="
PRROD_TYPE"/>
    </datasourcedataconnection>
</data:datadefinition>

```

Set a timeout for a data definition

```

<data:datadefinition id="suburbRoadType">
    <datasourcedataconnection datasource="datasource.main" table="
PROPERTY" prefix="DISTINCT">
        <parameter name="suburb" label="Suburb" column="
PRSUB_NAME"/>
        <parameter name="roadname" label="Road Name" column="
PRROD_NAME"/>

```

```

        <parameter name="roadtype" label="Road Type" column="
PRROD_TYPE" />
        <options>
            <timeout>15000</timeout>
        </options>
    </datasourcedataconnection>
</data:datadefinition>

```

More on Caching

To disable the caching all together you'd do the following

```

<data:datadefinition id="...">
    <datasourcedataconnection ...>
        ...
        <cache disable="true"/>
    </datasourcedataconnection>
</data:datadefinition>

```

To modify the default caching options

```

<data:datadefinition id="...">
    <datasourcedataconnection ...>
        ...
        <cache>
            <setting>value</setting>
            <setting>value</setting>
            <setting>value</setting>
        </cache>
    </datasourcedataconnection>
</data:datadefinition>

```

Where the `<settings>` you can change are identified at [caching](#)

You can set the default cache settings with

```

<data:cache>
    <setting>value</setting>
    <setting>value</setting>
    <setting>value</setting>
</data:cache>

```

and create a cache definition that can be re-used

```

<data:cache id="test">
    <setting>value</setting>
    <setting>value</setting>

```

```

        <setting>value</setting>
    </data:cache>

    <data:datadefinition id="...">
        <datasourcedataconnection ...>
            ...
            <cache id="test"/>
        </datasourcedataconnection>
    </data:datadefinition>

```

Note that prior to Weave 2.5.18 the default cache setting for a datasource data connection was to cache the information *forever*, as of 2.5.18 the data will only be cached for 5 minutes by default.

Using a temporary table

Using a temporary table

```

<data:datadefinition id="roadDetails">
    <datasourcedataconnection datasource="datasource.main" table="
ROADS" key="RD_NAME" prefix="DISTINCT" keytable="ROADS_SELECTION"
keycolumn="RD_NAME" usercolumn="UID">
        <from table="PLANS"/>
        <where clause="PLANS.ID=ROADS.PLAN_ID"/>
        <parameter name="name" label="Name" column="RD_NAME"/>
        <parameter name="suffix" label="Suffix" column="
RD_SUFFIX"/>
        <parameter name="type" label="Type" column="RD_TYPE"/>
        <parameter name="altname" label="Alt. Name" column="
ALT_NAME"/>
        <parameter name="altsuffix" label="Alt. Suffix"
column="ALT_SUFFIX"/>
        <parameter name="alttype" label="Alt. Type" column="
ALT_TYPE"/>
        <parameter type="integer" name="class" label="Class"
column="CLASS_CODE"/>
        <parameter type="url" name="plan" label="Plan"
column="'http://imgsvr:8080/gis/documents/plan_' || PLANS.PLAN_CODE || '.
pdf'" text="Open"/>
        <cache disable="true"/>
    </datasourcedataconnection>
</data:datadefinition>

```

This example will utilise a temporary table called `ROADS_SELECTION` which contains two columns `RD_NAME` who's data type matches the `RD_NAME` column in the `ROADS` table, and `UID` which is a `varchar` column long enough to contain 32 random characters, and there should be a non-unique index on the `UID` column.

Note: that "userid" and "key" are the default names for the columns in the temporary table, they can be called anything but if they're not "userid" and/or "key" then you'd need to set the `keycolumn` and `usercolumn` values in the data definition to match the names of the columns in the temporary table, as has been done above since the column names are actually "RD_NAME" and "UID" rather than "userid" and "key". You always need to provide the `keytable` parameter so there's no default for that.

Sorting data

You can add <sort> tags to specify what column(s) to sort by by default, e.g.

```
<data:datadefinition id="roadDetails">
  <datasourcedataconnection datasource="datasource.main" table="
ROADS" key="RD_NAME" prefix="DISTINCT">
    <from table="PLANS"/>
    <where clause="PLANS.ID=ROADS.PLAN_ID"/>
    <parameter name="name" label="Name" column="RD_NAME"/>
    <parameter name="suffix" label="Suffix" column="
RD_SUFFIX"/>
    <parameter name="type" label="Type" column="RD_TYPE"/>
    <parameter name="altname" label="Alt. Name" column="
ALT_NAME"/>
    <parameter name="altsuffix" label="Alt. Suffix"
column="ALT_SUFFIX"/>
    <parameter name="alttype" label="Alt. Type" column="
ALT_TYPE"/>
    <sort parameter="type"/>
    <sort parameter="name"/>
  </datasourcedataconnection>
</data:datadefinition>
```

New Join Syntax

As of Weave 2.5.11 it's possible to specify a table join using join specific tag (rather than having to use `from` and `where` tags).

This is done with the `join` tag, which takes 3 parameters, `table` which is required and is the name of the table to join on, `on` which is also required and is the join clause, and finally `type` which isn't required and will specify the join type to use (will default to an inner join).

Note that there currently is no checking on the value specified for the `type` parameter, the valid specified for `type` will be placed directly into the generated SQL, so you should ensure that you're using a valid join type for your database ("inner", "left", "right", "full", etc).

Join syntax example

```
<data:datadefinition id="roadDetails">
  <datasourcedataconnection datasource="datasource.main" table="
ROADS" key="RD_NAME" prefix="DISTINCT">
    <join type="left" table="PLANS" on="PLANS.ID=ROADS.
PLAN_ID">
      <clause>PLANS.ACTIVE=TRUE</clause>
    </join>
    <parameter name="name" label="Name" column="RD_NAME"/>
    <parameter name="suffix" label="Suffix" column="
RD_SUFFIX"/>
    <parameter name="type" label="Type" column="RD_TYPE"/>
    <parameter name="altname" label="Alt. Name" column="
ALT_NAME"/>
    <parameter name="altsuffix" label="Alt. Suffix"
column="ALT_SUFFIX"/>
    <parameter name="alttype" label="Alt. Type" column="
ALT_TYPE"/>
    <parameter type="integer" name="class" label="Class"
column="CLASS_CODE"/>
```

```

        <parameter type="url" name="plan" label="Plan"
column=" 'http://imgsvr:8080/gis/documents/plan_' || PLANS.PLAN_CODE || '.
pdf' " text="Open"/>
    </datasourcedataconnection>
</data:datadefinition>

```

The above example shows joining the `PLANS` table. Using the previous version of the `roadDetails` data definition would not return any rows where there was not a match between the `ROADS` and `PLANS` table, using the newer syntax it's possible to generate SQL that will return rows from `ROADS` even if there is no match in the `PLANS` table.

Note that the above configuration for the `plan` parameter is not currently smart enough to generate a value value for `column`, since it does not take into account the fact that the `PLANS.PLAN_CODE` value will be `null` if there is no matching row in the `PLANS` table. The `column` attribute should be updated to use the appropriate SQL syntax for your database to check for a `null` value and handle it accordingly (i.e. generate a link to a "no plan available" page).

The SQL generated for the previous version (using `from` and `where` tags) would be of the form:

```

SELECT columns
FROM primary_table, secondary_table
WHERE primary_table.column = secondary_table.column

```

Using a `join` tag (without a `type` specified) the SQL generated would be of the form:

```

SELECT columns
FROM primary_table JOIN secondary_table ON primary_table.column =
secondary_table.column

```

If the `type` were specified as "left" then the SQL generated would be of the form:

```

SELECT columns
FROM primary_table left JOIN secondary_table ON primary_table.column
= secondary_table.column

```

Group Data Connection

groupdataconnection

Properties

Name	Type	Required	Description
datadefinition	urn:com.cohga.server.data.database#1.0	yes	Id of a data definition that will supply the source data that this data definition will group

Sub-tags

Name	Type	Cardinality
parameter	parameter	1..n

Content

None

parameter

Properties

Name	Type	Required	Default	Description
type	'string', 'int', 'date' or 'url'	no	'string')	An indicator of how the data should be displayed on the client
name	string	no	lowercase value of column	A unique identifier for the parameter
label	string	no	'pretty' value of column	A user displayable label for the column
column	ref	no	'pretty' value of column	A user displayable label for the column

Sub-tags

None

Content

None

Notes

- The `column` attribute either references a parameter in the source data definition or it uses one of the following formulas
 - `sum(parameter)` - The average value of the parameter
 - `avg(parameter)` - The average value of the parameter
 - `min(parameter)` - The average value of the parameter
 - `max(parameter)` - The average value of the parameter
 - `count()` - The count of the records in the group
 - `rownum()` - The row number of the record
 - `date()` - The current date
- If no `label` is specified then it will be generated by formatting the `name`, unless no `name` is supplied, then it will be generated by formatting the `column`
- Formatting involves converting any `_`'s to spaces, converting the first letter and any letter after a space to upper case and converting every other letter to lower case, e.g. "BOMB_DISPOSAL_METHOD" becomes "Bomb Disposal Method"

Examples

Given the following data definition describing a table that contains a unique 'id' for each row, a non-unique 'group' for each row, a 'value' and a 'date' when that value was recorded.

```
<data:datadefinition id="test.source">
  <datasourcedataconnection datasource="test.
datasource" table="TESTDATA" key="ID">
    <parameter name="group" label="Group" column="
GROUP" />
    <parameter name="date" label="Date" column="
DATE" type="date" />
    <parameter name="value" label="Value" column="
VALUE" type="integer" />
  </datasourcedataconnection>
</data:datadefinition>
```

representing this table

ID	GROUP	DATE	VALUE
0	A	1-jan-2000	1
1	A	2-jan-2000	3
2	A		1

		3-jan-2000	
3	B	1-jan-2000	2
4	B	2-jan-2000	5
5	B	3-jan-2000	3
6	C	1-jan-2000	4
7	C	2-jan-2000	2
8	C	3-jan-2000	2
9	C	4-jan-2000	1

This example

```

    <!-- Create a new data definition that groups all of the
values -->
    <data:datadefinition id="test.group">
      <groupdataconnection datadefinition="test.source">
        <parameter name="sum"      label="Sum"
column="sum(value)"/>
        <parameter name="avg"      label="Average"
column="avg(value)"/>
        <parameter name="min"      label="Minimum"
column="min(value)"/>
        <parameter name="max"      label="Maximum"
column="max(value)"/>
        <parameter name="count"    label="Count"
column="count(1)"/>
      </groupdataconnection>
    </data:datadefinition>

```

will generate this

su m	av g	mi n	ma x	coun t
24	2.4	1	5	10

This example

```

    <!-- Create a new data definition that groups the values
based on their 'group' value -->
    <data:datadefinition id="test.group">
      <groupdataconnection datadefinition="test.source">
        <parameter name="group"    label="Group"
column="group"/>
        <parameter name="sum"      label="Sum"
column="sum(value)"/>
        <parameter name="avg"      label="Average"
column="avg(value)"/>
        <parameter name="min"      label="Minimum"

```

```

column="min(value)"/>
                                <parameter name="max"      label="Maximum"
column="max(value)"/>
                                <parameter name="count"   label="Count "
column="count(1)"/>
                                <group column="group"/>
                                </groupdataconnection>
                                </data:datadefinition>

```

will generate this

group	sum	avg	min	max	count
A	5	1.66	1	3	3
B	10	3.33	2	5	3
C	9	2.25	1	4	4

This example

```

<!-- Create a new data definition that groups the values
based on their 'date' value -->
<data:datadefinition id="test.group">
  <groupdataconnection datadefinition="test.source">
    <parameter name="group"   label="Group "
column="date"/>
    <parameter name="sum"     label="Sum"
column="sum(value)"/>
    <parameter name="avg"    label="Average"
column="avg(value)"/>
    <parameter name="min"   label="Minimum"
column="min(value)"/>
    <parameter name="max"   label="Maximum"
column="max(value)"/>
    <parameter name="count" label="Count "
column="count(1)"/>
    <group column="date"/>
  </groupdataconnection>
</data:datadefinition>

```

will generate this

group	sum	avg	min	max	count
1-jan-2000	7	2.33	1	4	3
2-jan-2000	10	3.33	2	5	3
3-jan-2000	6	2	1	3	3
4-jan-2000	1	1	1	1	1

Inline Data Connection

inlinedataconnection

Properties

None

Sub-tags

Name	Type	Cardinality
parameter	urn:com.cohga.server.data.database#1.0:parameter	1..n
row	urn:com.cohga.server.data.database#1.0:row	1..n

Content

None

parameter

Properties

Name	Type	Required	Description
name	string	yes	A unique identifier for the parameter
type	'string', 'numeric', 'date' or 'uri'	yes	An indicator of how the data should be displayed on the client
label	string	yes	A user displayable label for the column

Sub-tags

None

Content

None

Notes

row

Properties

None

Sub-tags

Name	Type	Cardinality
cell	urn:com.cohga.server.data.database#1.0:cell	1..n

Content

None

Notes

- The number of cells should match the number of parameters.

cell

Properties

None

Sub-tags

None

Content

The value to be used for this cell, or null if null should be returned as this cells value. Note that you can also use `xsi:nil="true"` if you have the `xsi` namespace setup as described in [configuration](#).

Examples

A simple inline data definition

```
<data:datadefinition id="connectionType">
  <inlinedataconnection>
    <parameter type="string" name="key" label="Key" />
    <parameter type="string" name="label" label="Label" />
    <row>
      <cell>U</cell>
      <cell>Unverified</cell>
    </row>
    <row>
      <cell>V</cell>
      <cell>Verified</cell>
    </row>
  </inlinedataconnection>
</data:datadefinition>
```

A simple inline data definition using null

```
<data:datadefinition id="customerType">
  <inlinedataconnection>
    <parameter type="string" name="label" label="Label" />
    <parameter type="string" name="key" label="Key" />
    <row>
      <cell>Domestic</cell>
      <cell>D</cell>
    </row>
    <row>
      <cell>Commercial</cell>
      <cell>C</cell>
    </row>
    <row>
      <cell>Unknown</cell>
      <cell>null</cell>
    </row>
  </inlinedataconnection>
</data:datadefinition>
```

Spatial Data Connection

spatialdataconnection

Properties

Name	Type	Required	Description

entity	ref urn:com.cohga.server.entity#1.0	yes	Id if the entity that this data definition should use to generate its data. The data is generated based on the Spatial Mapping specified for the entity
--------	---	-----	---

Sub-tags

Name	Type	Cardinality
parameter	parameter	1..n

Content

None

parameter

Properties

Name	Type	Required	Description
column	string	yes	The name of the column within the table that this parameter references (can be an sql function also)
type	'string', 'numeric', 'date' or 'url'	no (default 'string')	An indicator of how the data should be displayed on the client
name	string	no (default lowercase value of column)	A unique identifier for the parameter
label	string	no (default 'pretty' value of column)	A user displayable label for the column
text	string	no	Only when type is 'url'. Specifies text to be displayed to the user instead of the actual url contents
textcolumn	string	no	Only when type is 'url'. Specifies the column that contains text to be displayed to the user instead of the actual url contents

Sub-tags

None

Content

None

Notes

- If no `label` is specified then it will be generated by formatting the `name`, unless no `name` is supplied, then it will be generated by formatting the `column`
- Formatting involves converting any `_`'s to spaces, converting the first letter and any letter after a space to upper case and converting every other letter to lower case, e.g. "BOMB_DISPOSAL_METHOD" becomes "Bomb Disposal Method"
- Only one of `text` or `textcolumn` should be specified when type is 'url'

Examples

A simple spatial data definition generating `name` and `label` attributes based on the `column`

```
<data:datadefinition id="graffiti">
  <spatialdataconnection entity="graffiti">
    <parameter column="ID"/> <!-- name will be 'id',
label 'Id' -->
    <parameter column="DESCRIPTION"/> <!-- name will be
'description', label 'Description' -->
    <parameter label="Created on" column="CREATED" type="
datetime"/> <!-- name will be 'created' -->
    <parameter label="Modified on" column="MODIFIED"
type="datetime"/> <!-- name will be 'modified' -->
    <parameter name="user" label="User" column="USERID"/>
```

```

    </spatialdataconnection>
</data:datadefinition>

```

Spatial Intersection Data Definition

The new Spatial Intersection Data Definition will allow you to create a data definition that returns data in a different table from an entity based on a spatial intersection between the two.

The updated bundle can be [downloaded from here](#)

The implementation was developed under the assumption that it'll be used in BIRT reports to generate spatial drill down type reports and that there will generally be only one (or at worst a handful of) parent entities, and is not optimised to be used on large numbers of parent entities.

I'm making a note this because while you can create a <data:data> tag to allow display of the data generated on the client, which may be handy for testing/debugging, but if the user selects hundreds of parent entities and tries to display the joined data the process could take a long time to complete.

So it's recommended that, initially at least, this only be used in BIRT reports where a small number of parent entities can be enforced.

This is because the data is generated by iterating over the source entities and performing a spatial join for each row returned.

The basic spatial intersection data definition is something like the following:

Table join with output rows pulled from child table

```

<?xml version="1.0" encoding="UTF-8"?>
<config      xmlns = "urn:com.cohga.server.config#1.0" xmlns:data =
"urn:com.cohga.server.data.database#1.0">
    <data:datadefinition id="spatial.join">
        <spatialintersectiondataconnection entity="
parentEntityId" table="childTableName"/>
    </data:datadefinition>
</config>

```

You'll want to choose appropriate values for "spatial.join", "parentEntityId" and "childTableName".

What this will do is create a new data definition that you can use in a BIRT report to display data from a spatial table called "childTableName" (which would be something like "FLOOD_ZONES" or "WASTE_COLLECTION_DAY") based on the selected entities in the "parentEntityId" entity (which would be something like "property" or "parcel") and it does this based on the intersection of the geometry in the childTableName table and the parentEntityId entities.

Currently the child table needs to be in the same location as the underlying spatial table of the parent entity when specifying a child table directly.

By default the data definition will return all attributes from the underlying spatial table, but when it's added to BIRT you can select only those columns that you actually want to display, so that's not a big issue.

If you want to limit the columns returned then you can add "parameter" tags to describe the columns you want returned, e.g.

Table join with output columns specified directly

```

<?xml version="1.0" encoding="UTF-8"?>
<config      xmlns = "urn:com.cohga.server.config#1.0" xmlns:data =
"urn:com.cohga.server.data.database#1.0">
    <data:datadefinition id="spatial.join">
        <spatialintersectiondataconnection entity="
parentEntityId" table="childTableName">
            <parameter name="childColumn1" label="Child

```

```

Column 1" column="CHILD_COLUMN1"/>
                <parameter name="childColumn2" label="Child
Column 2" column="CHILD_COLUMN2"/>
            </spatialintersectiondataconnection>
        </data:datadefinition>
    </config>

```

So then only the 2 columns listed will be returned.

Additionally you can specify a child entity rather than a child table if you already have an entity that represents the table that you're trying to join with, e.g.

Entity join with output columns pulled from child entity

```

<?xml version="1.0" encoding="UTF-8"?>
<config          xmlns = "urn:com.cohga.server.config#1.0" xmlns:data =
"urn:com.cohga.server.data.database#1.0">
    <data:datadefinition id="spatial.join">
        <spatialintersectiondataconnection entity="
parentEntityId" targetentity="childEntityId"/>
    </data:datadefinition>
</config>

```

or with parameters also defined directly

Entity join with output columns defined

```

<?xml version="1.0" encoding="UTF-8"?>
<config          xmlns = "urn:com.cohga.server.config#1.0" xmlns:data =
"urn:com.cohga.server.data.database#1.0">
    <data:datadefinition id="spatial.join">
        <spatialintersectiondataconnection entity="
parentEntityId" targetentity="childEntityId">
            <parameter name="childColumn1" label="Child
Column 1" column="CHILD_COLUMN1"/>
            <parameter name="childColumn2" label="Child
Column 2" column="CHILD_COLUMN2"/>
        </spatialintersectiondataconnection>
    </data:datadefinition>
</config>

```

Finally it's also possible to specify a buffer that can be applied to the parent geometry before using it to select the child rows, e.g.

Table join with buffer specified for source geometry

```

<?xml version="1.0" encoding="UTF-8"?>
<config          xmlns = "urn:com.cohga.server.config#1.0" xmlns:data =

```

```

"urn:com.cohga.server.data.database#1.0">
  <data:datadefinition id="spatial.join">
    <spatialintersectiondataconnection entity="
parentEntityId" table="childTableName" buffer="10" bufferUnits="m" />
  </data:datadefinition>
</config>

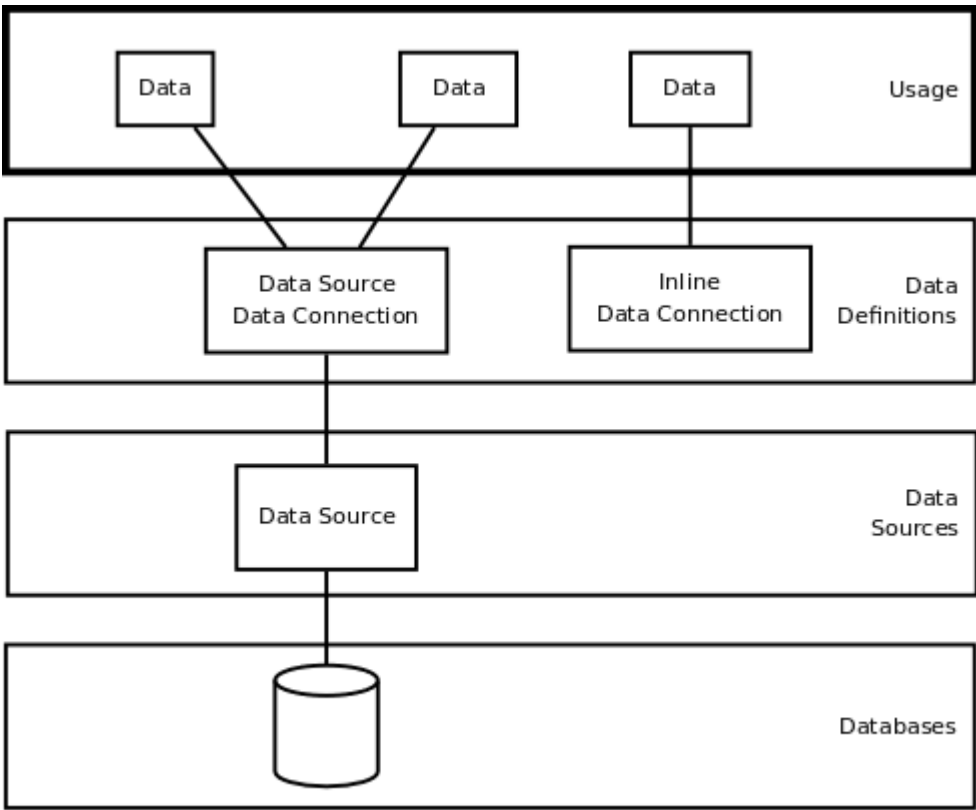
```

In the above examples "parentEntityId" is the name of an entity that the data is being generated for, based on the selection for that entity, for example if it were "property" then the data will appear as though it's directly related to the "property" entity, even though it's coming from something different, for example a flood area polygon table. The attributes that are returned come from the "childTableName" table or the table underlying the "childEntityId" entity (as specified by the spatial mapper associated with the child entity).

For information on how to use this data definition in a BIRT report to generate a parent-child drill down report please see the section on Linking Data Sets at [Adding Data to BIRT Reports](#), as the process of creating a parent-child report is exactly the same as described there.

Data

The Data configuration item is the last piece of the puzzle in providing textual data display to the user, with the [Data Source](#) providing the repository information, the [Data Definition](#) providing the description of the data, and finally the Data providing the display label and entity to link to.



A [Data Definition](#) describes a source of data that can be used in a number of places through out Weave, but to allow this data to be displayed directly to the user (via a data grid view, etc) then we need to provide some extra information to allow Weave to know how to present the data to the user (i.e. give it a user readable label) and tell it which entity the data relates to (so Weave knows which rows to return results for).

Each Data configuration provides a selectable entry for the users within any grid data views to allow them to display data from the associated [Data Definition](#).

Without a Data tag there would be no way for the user to directly display the information represented by a [Data Definition](#).

Name	Suffix	Type	Alt. Name	Alt. Suffix
GREENRDIGE		AVENUE		
GREENRIDGE		AVENUE		
HAMPDEN		COURT		
KOLOR		WAY		
NEWLYN		CLOSE		
CRICKLEWOOD		DRIVE		
BIRCHGROVE		CRESCENT		
VICTORIA		STREET	DONCASTER-	MORDIALLOC
VICTORIA		STREET		
CHELMSFORD		AVENUE		
CASTLEWOOD		PLACE		

Users may still be able to indirectly display information from a [Data Definition](#) if that [Data Definition](#) is used by another item that the user has access to, for example if the [Data Definition](#) presents a list of values for an search.

Namespace

urn:com.cohga.server.data.database#1.0

Tags

data

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
label	string	yes	Text to display to the user to represent this data set
datadefinition	ref urn:com.cohga.server.data.database#1.0:datadefinition	yes	The data definition that that data will present
entity	ref urn:com.cohga.server.entity#1.0:entity	yes	The entity that this data should be linked to
acl	ref urn:com.cohga.server.acl#1.0:acl	no	A reference to an ACL to attach to the entity

Sub-tags

Name	Type	Cardinality
acl	urn:com.cohga.server.acl#1.0:acl	0..1

Content

None

Examples

```

<jdbc:datasource id="datasource.main">
  <!-- connection details for data source -->
</jdbc:datasource>

<data:datadefinition id="dd_roadDetails">
  <datasourcedataconnection datasource="datasource.main" table="
ROADS" key="RD_NAME" prefix="DISTINCT">
    <parameter name="name" label="Name" column="RD_NAME"/>
    <parameter name="suffix" label="Suffix" column="
RD_SUFFIX"/>
    <parameter name="type" label="Type" column="RD_TYPE"/>
    <parameter name="altname" label="Alt. Name" column="
ALT_NAME"/>
    <parameter name="altsuffix" label="Alt. Suffix"
column="ALT_SUFFIX"/>
    <parameter name="alttype" label="Alt. Type" column="
ALT_TYPE"/>
    <parameter name="class" label="Class" column="
CLASS_CODE" type="integer"/>
  </datasourcedataconnection>
</data:datadefinition>

<data:data id="roadDetails" label="Road Details" datadefinition="
dd_roadDetails" entity="roads"/>

```

Data with ACL

```

<data:data id="roadDetails" label="Road Details" datadefinition="
dd_roadDetails" entity="roads">
  <acl id="public"/>
</data:data>

```

Entity

An entity provides a basic reference to an item that will be searchable and reportable within the system.

Functionally it does not provide much more than access control and a label for display to the user, it is up to other items in the configuration to reference the entity to provide added functionality.

For example for an entity to be "spatially enabled" rather than adding spatial information to the entity you create a separate spatial mapper item that references the entity.

Namespace

urn:com.cohga.server.entity#1.0

Tags

entity

Properties

Name	Type	Required	Description
------	------	----------	-------------

id	string	yes	unique identifier
label	string	yes	User presented label for the entity
publish	string	no	"true": display entity to users "false": hide entity from users e.g.: in the Entity drop down
acl	ref urn:com.cohga.server.acl#1.0:acl	no	A reference to an ACL to attach to the entity

Sub-tags

Name	Type	Cardinality
acl	urn:com.cohga.server.acl#1.0:acl	0..1

Content

None

Notes

- An ACL can either be defined in-line or referenced indirectly, but only one should be used (the in-line version will take priority)

Examples

A simple entity with a label

```
<entity:entity id="suburb">
  <label>Suburb</label>
</entity:entity>
```

An entity that's only visible to users who are members of the 'public' ACL

```
<entity:entity id="road">
  <label>Road</label>
  <acl>public</acl>
</entity:entity>
```

A entity with an ACL defined in-line, rather than referenced like the previous example

```
<entity:entity id="property">
  <label>Property</label>
  <acl:acl>
    <entry type="deny">anonymous</entry>
    <entry type="allow">*</entry>
  </acl:acl>
</entity:entity>
```

Cache

The cache configuration provides a way to tune the setting used to setup the cache associated with various other configuration items, for example a [Data Source based Data Definition](#).

The cache tag itself is not used as a top level configuration item but is instead embedded within other tags.

Namespace

urn:com.cohga.server.cache#1.0

Tags

cache

Properties

Name	Type	Required	Description
maxElementsInMemory	number	no	The maximum number of elements in memory, before they are evicted
maxElementsOnDisk	number	no	The maximum number of elements to hold on Disk
memoryStoreEvictionPolicy	'LRU', 'LFU', or 'FIFO'	no	Policy used to determine which items to remove, default is LRU
overflowToDisk	boolean	no	Whether to use the disk store
eternal	boolean	no	Whether the elements in the cache are eternal, i.e. never expire
timeToLiveSeconds	number	no	The default amount of time to live for an element from its creation date
timeToIdleSeconds	number	no	The default amount of time to live for an element from its last accessed or modified date
diskPersistent	boolean	no	Whether to persist the cache to disk between JVM restarts
diskExpiryThreadIntervalSeconds	number	no	How often to run the disk store expiry thread. A large number of 120 seconds plus is recommended

Sub-tags

None

Content

None

Examples

```
<cache:cache>
  <maxElementsInMemory>50</maxElementsInMemory>
  <memoryStoreEvictionPolicy>LRU</memoryStoreEvictionPolicy>
  <eternal>false</eternal>
  <timeToIdleSeconds>0</timeToIdleSeconds>
  <timeToLiveSeconds>300</timeToLiveSeconds>
  <overflowToDisk>true</overflowToDisk>
  <maxElementsOnDisk>2000</maxElementsOnDisk>
  <diskPersistent>false</diskPersistent>
  <diskExpiryThreadIntervalSeconds>120<
/diskExpiryThreadIntervalSeconds>
</cache:cache>
```

Disable caching

```
<cache:cache disable="true"/>
```

Search

A search allows the user to locate entities based on textual attributes of the entity. The user is presented with an input form where they can fill out information required to identify the entities to locate and those entities will be used to update the current selection for that entity.

Current implementations perform those searches either on a database, using either SQL or Stored Procedures, or on a spatial engine.

- [Attribute Search](#)
- [Stored Procedure Search](#)
- [Spatial Search](#)

Attribute Search

An attribute search allows the server to locate entities using SQL.

Wildcards

When searching text columns the default SQL generated will use an 'equals' expression rather than a 'like' expression in the where clause, unless the user includes a '%' in the search value.

You can override this by not setting a "column" in the parameter definition but instead adding a "where" clause to the parameter that defines exactly the SQL you want included, so for example:

```
<search:attribute ...>
...
<parameter id="text_param">
  <column>TEXT_COLUMN</column>
  ...
</parameter>
</search:attribute>
```

which would become (unless the user entered a value that included a %)

```
WHERE TEXT_COLUMN = ...
```

in the generated SQL.

You should instead use:

```
<search:attribute ...>
...
<parameter id="text_param">
  <where clause="TEXT_COLUMN like '${text_param}%'"/>
  ...
</parameter>
</search:attribute>
```

which uses `${text_param}` as a place marker for the value the user entered for the parameter named 'text_param', and would become

```
WHERE TEXT_COLUMN like '...%'
```

To be safe, it is recommended that the where clause is wrapped in CDATA tags, so the where clause would be

```
<where>
  <clause><![CDATA[TEXT_COLUMN like'${text_param}%']]></clause>
</where>
```

Namespace

urn:com.cohga.server.search.database#1.0

Tags

attribute

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier for this search
entity	ref urn:com.cohga.server.entity#1.0:entity	yes	The id of the entity that this search will be searching for
label	string	yes	Text to be displayed to the user to represent this search
description	string	no	Description of the search that could be displayed to the user to explain this search
datasource	ref urn:com.cohga.server.datasource.jdbc#1.0:datasource	yes	The data source that should be connected to perform the search
table	string	yes	The name of the table in the datasource that is the root of the generated SQL
key	string	yes	The column in table that represents the unique id of the entity being searched for
acl	ref urn:com.cohga.server.acl#1.0:acl	no	A reference to an ACL to attach to the search

Sub-tags

Name	Type	Cardinality
from	urn:com.cohga.server.search.database#1.0:from	0..n
where	urn:com.cohga.server.search.database#1.0:where	0..n
parameter	urn:com.cohga.server.search.database#1.0:parameter	1..n
combination	urn:com.cohga.server.search.database#1.0:combination	0..n
acl	urn:com.cohga.server.acl#1.0:acl	0..1
cache	urn:com.cohga.server.cache#1.0:cache	0..1
options	urn:com.cohga.server.search.database#1.0:options	0..1

Notes

An ACL can either be defined in-line or referenced indirectly, but only one should be used (the in-line version will take priority)

parameter

Properties

Name	Type	Required	Default	Description
id	string	yes		A unique identifier for the parameter
label	string	yes		The prompt text displayed when user input the parameter value

column	string	no		The name of the column within the table that this parameter references, if this isn't provided then a <code>where</code> sub-tag must be added to this parameter to describe how the parameter should be included in the generated SQL.
displayname	string	no	label	Provides a user-friendly name for the element
helptext	string	no		Additional text to display for the parameter to explain how to use the parameter
hidden	boolean	no	false	Hides the parameter from the parameter UI
alignment	'left', 'center', 'right', 'auto'	no	'auto'	How the items should appear in the UI
controltype	'listbox', 'checkbox', 'radiobutton', 'textbox'	no	'textbox'	The suggested type of UI control to use when displaying the parameter
datatype	'any', 'boolean', 'datetime', 'decimal', 'float', 'integer', 'string'	no	'string'	The data type for the parameter
allownull	boolean	no	false	Whether a null value is allowed for this parameter
allowblank	boolean	no	true	Give the user the choice of an empty value in the listbox (as opposed to a null value)
allownewvalues	boolean	no	false	Allow the user to enter values not in the listbox already. Note that setting this to true will convert the listbox into a combo box, also note that the default was true before 2.5.30.
defaultvalue	any	no		The default value of the parameter
displayformat	string	no		the formatting instructions for the parameter value within the parameter UI
dataset	ref urn:com.cohga.server.data.database#1.0:datadefinition	no		Where to get the values for a listbox
labelcolumn	string	no		Column in the datadefinition that supplies the label of the value to show the user
valuecolumn	string	no		Column in the datadefinition that supplies the value of the value to use in the SQL
uppercase	boolean	no	false	Should the value be converted to upper case in the generated SQL
scalarpаметertype	string	no	'simple'	'simple' or 'multi-value' to determine if more than one value can be selected from a list.
matchtype	string	no	'start'	'start', 'middle' or 'end' to determine if list filtering is performed by matching against the start of the list entry, the middle or the end of the label.
width	integer	no		Set the width of the field
pagesize	integer	no	10	Number of items to show on a list page, 0 disables paging
autoselect	boolean	no	true	Should the first value in a list be pre-selected
minvalue	any	no		The minimum value allowed for a field. Available from 2.5.28.
maxvalue	any	no		The maximum value allowed for a field. Available from 2.5.28.
minlength	integer	no		The minimum length allowed for a field. Available from 2.5.28.
maxlength	integer	no		The maximum length allowed for a field. Available from 2.5.28.
increment	integer	no		The increment to use for fields that support it, the units are dependant upon the field type. Available from 2.5.28 and currently only for time fields.
trueValue	any	no		The value that equates to "true" in the underlying table, only suitable for checkboxes
falseValue	any	no		The value that equates to "false" in the underlying table, only suitable for checkboxes

Sub-tags

Name	Type	Cardinality
------	------	-------------

from	urn:com.cohga.server.data.database#1.0:from	0..n
where	urn:com.cohga.server.data.database#1.0:where	0..n
parameter	urn:com.cohga.server.data.database#1.0:parameter	0..n
list	urn:com.cohga.server.search.database#1.0:list	0..n

Notes

- If the controltype is listbox then a dataset must be provided, this dataset will provide the values to be displayed in the listbox.
- If the listbox dataset only contains 1 column then that column will supply both the label and the value, if it contains two values then the first column will supply the label and the second the value.
- If a parameter contains another parameter then you are defining a cascading parameter, where setting the first sub-parameter will enable, and filter, the second parameter, and setting the second parameter will do the same for the third, etc.
- In a cascading parameter all sub-parameters must be of listbox type, and the dataset should be set in the outer parameter, not the sub parameters.
- In a cascading parameter only one level of nesting should be used.
- In a cascading parameter the dataset should supply the columns for all of the parameters, and valuecolumn properties should be set for each sub-parameter, and labelcolumn should be also set for all parameters if a different label is to be displayed to the user.
- If the controltype is radiobutton then 2 or more list entries must be provided to populate the buttons. If no defaultvalue is set, the first entry will be taken as the default.

combination

Properties

None

Sub-tags

Name	Type	Cardinality
from	urn:com.cohga.server.data.database#1.0:from	0..n
where	urn:com.cohga.server.data.database#1.0:where	1..n

from

Properties

Name	Type	Required	Description
table	string	yes	An additional table to include in the generated SQL

where

Properties

Name	Type	Required	Description
clause	string	yes	An additional clause to include in the generated SQL.
uppercase	boolean	no	If the clause uses parameter substitution should the value be converted to upper-case before being substituted

list

Properties

Name	Type	Required	Description
value	string	yes	The value used if this list item was chosen
label	string	yes	The label to display for this list item

options

Properties

--	--	--	--

Name	Type	Required	Description
limit	integer	no	Place a limit on the number of items that will be returned

Examples

Comprehensive search example

```

<data:datadefinition id="dd_roadtypes">
  <datasourcedataconnection datasource="datasource.main" table="
ROAD" prefix="DISTINCT">
    <parameter type="string" name="roadname" column="
ROADNAME" />
    <parameter type="string" name="roadname_code" column="
ROADNAME_CODE" />
    <parameter type="string" name="roadtype" column="
ROADTYPE" />
    <parameter type="string" name="roadtype_code" column="
ROADTYPE_CODE" />
  </datasourcedataconnection>
</data:datadefinition>

<search:attribute id="property.address.entry">
  <entity>property</entity>
  <label>Address by Entry</label>
  <description>Locate a property or properties by their street
address</description>
  <datasource>datasource.main</datasource>
  <table>PID_LINK</table>
  <key>PID</key>
  <from>
    <table>PROPERTY</table>
  </from>
  <where>
    <clause>LINK.ID=PROPERTY.ID</clause>
  </where>
  <parameter id="unit">
    <label>Unit</label>
    <column>UNIT</column>
  </parameter>
  <parameter id="house_from">
    <label>House From</label>
    <datatype>integer</datatype>
    <column>PROP_NO</column>
  </parameter>
  <parameter id="house_to">
    <label>House To</label>
    <datatype>integer</datatype>
    <column>PROP_NO</column>
  </parameter>
  <parameter id="suburb">
    <label>Suburb</label>
    <controltype>listbox</controltype>
    <allownull>true</allownull>

```

```

        <column>SUBURB</column>
        <datadefinition>dd_suburbs</datadefinition>
    </parameter>
    <parameter id="road">
        <dataset>dd_roadtypes</dataset>
        <parameter id="roadname">
            <label>Road Name</label>
            <controlype>listbox</controlype>
            <column>ROADNAME_CODE</column>
            <valuecolumn>roadname_code</valuecolumn>
            <labelcolumn>roadname</labelcolumn>
        </parameter>
        <parameter id="roadtype">
            <label>Road Type</label>
            <controlype>listbox</controlype>
            <column>ROADTYPE_CODE</column>
            <valuecolumn>roadtype_code</valuecolumn>
            <labelcolumn>roadtype</labelcolumn>
        </parameter>
    </parameter>
    <parameter id="postcode">
        <label>Postcode</label>
        <datatype>integer</datatype>
        <where>
            <clause><![CDATA[POSTCODE=${postcode}]]><
/clause>
            </where>
        </parameter>
        <combination>
            <where>
                <clause><![CDATA[PROP_NO >= ${house_from} and
PROP_NO <= ${house_to}]]></clause>
            </where>
        </combination>
        <cache>
            <maxElementsInMemory>50</maxElementsInMemory>
            <eternal>>false</eternal>
            <timeToIdleSeconds>0</timeToIdleSeconds>
            <timeToLiveSeconds>300</timeToLiveSeconds>
            <overflowToDisk>>true</overflowToDisk>
            <maxElementsOnDisk>2000</maxElementsOnDisk>
            <diskPersistent>>false</diskPersistent>
            <diskExpiryThreadIntervalSeconds>120<
/diskExpiryThreadIntervalSeconds>
            <memoryStoreEvictionPolicy>LRU<
/memoryStoreEvictionPolicy>
        </cache>
        <options>
            <limit>2000</limit>
        </options>
    </search:attribute>

```

Searching on a date or a range of dates

```

<search:attribute id="property.by_sale_date">
  <entity>property</entity>
  <displayName>by Sale Date</displayName>
  <description>Locate a property by the date of its sale<
/description>
  <dataSource>datasource.property</dataSource>
  <table>PROP_SALE</table>
  <key>PROPERTYID</key>
  <parameter id="startdate">
    <promptText>Start Date</promptText>
    <controlType>text-box</controlType>
    <dataType>date</dataType>
    <column>SALEDATE</column>
  </parameter>
  <parameter id="enddate">
    <promptText>End Date</promptText>
    <controlType>text-box</controlType>
    <dataType>date</dataType>
    <column>SALEDATE</column>
  </parameter>
  <combination>
    <where>
      <clause><![CDATA[SALEDATE between
${startdate} and ${enddate}]]></clause>
    </where>
  </combination>
</search:attribute>

```

The trueValue/falseValue settings in the checkbox in the following example only work for Weave 2.5+

Search with checkbox and radiobutton

```

<search:attribute id="certificate.by_status">
  <entity>certificate</entity>
  <displayName>by Status</displayName>
  <description>Locate a certificate by its current status<
/description>
  <dataSource>datasource.main</dataSource>
  <table>CERTIFICATES</table>
  <key>ID</key>
  <parameter id="current">
    <label>Current</label>
    <controlType>checkbox</controlType>
    <column>IS_CURRENT</column>
    <trueValue>Y</trueValue>
    <falseValue xsi:nil="true"/>
  </parameter>
  <parameter id="state">

```

```

        <label>State</label>
        <controlType>radio-button</controlType>
        <column>STATE</column>
        <defaultValue>P</defaultValue>
        <list>
            <label>Pending</label>
            <value>P</value>
        </list>
        <list>
            <label>Registered</label>
            <value>R</value>
        </list>
        <list>
            <label>Active</label>
            <value>A</value>
        </list>
        <list>
            <label>Closed</label>
            <value>C</value>
        </list>
        <list>
            <label>Unknown</label>
            <value xsi:nil="true" />
        </list>
    </parameter>
</search:attribute>

```

Date parameters in SQL Server

The date being passed from the client to the server is in the following format *yyyy-mm-ddThh:mi:ss.mmm* (no spaces). If you are overriding the where clause of a parameter you will need to convert the date string from the client into a SQL Server date. To do this use the CONVERT function using style 126. For more information see the following page

<https://msdn.microsoft.com/en-AU/library/ms187928.aspx>

As an example here are two date parameters in a single search that override the where clause.

```

<parameter id="dp_appdatefrom" promptText="Application Date From:"
dataType="date">
    <where>
        <clause><![CDATA[APPLICATION_DATE]=CONVERT
(DATE, '$dp_appdatefrom$',126)]]></clause>
    </where>
</parameter>
<parameter id="dp_appdateto" promptText="Application Date To:"
dataType="date">
    <where>
        <clause><![CDATA[APPLICATION_DATE<=CONVERT
(DATE, '$dp_appdateto$',126)]]></clause>
    </where>
</parameter>

```

Spatial Search

A spatial search is similar to an attribute search, but uses the results of the search to then perform a geometry intersection search on another entity, which is the real target of the search.

So a spatial search works like this:

- The user enters the search parameters and submits the search
- The server searches for the source entities that match the search parameters
- The geometry of the found source entities are then used in a spatial search for the target entities
- The selection for the target entities is then updated based on the results of the spatial search

The spatial search has been extended to provide support for performing an attribute search against a spatial engine. That is a spatial search can now be used to perform the same function as an attribute search, but rather than performing the operation against a database table it is performed against a spatial engine layer.

None of the operations outlined above in "So a spatial search works like this" is performed, instead the operation more closely follows what an attribute search performs, i.e. finding rows in a layer that have attributes that match what a user has entered.

The configuration of the spatial attribute search follows more closely that of the attribute search. Because it is using the underlying API of the spatial engine, rather than SQL as the attribute search does (although some spatial engines also use SQL as their API), the support and configuration may be different depending upon the level of support provided by the underlying spatial engine. For example, a spatial search operating on a spatial engine that is pointing to a directory of shapefiles may operate differently from one that is pointing to ArcSDE. This is similar to the differences in an attribute search that is pointing to an Oracle database compared to one that is pointing to SQL Server.

Namespace

urn:com.cohga.server.search.spatial#1.0

Tags

spatial

Properties

Name	Type	Required	Description
id	string	yes	unique identifier for this search
entity	ref urn:com.cohga.server.entity#1.0:entity	yes	The id of the entity that this search will be searching for
sourceEntity (or sourceentity)	ref urn:com.cohga.server.entity#1.0:entity	yes	The id of the entity that the search parameters apply to
displayName (or displayname)	string	yes	Text to be displayed to the user to represent this search
description	string	no	Description of the search that could be displayed to the user to explain this search
acl	ref urn:com.cohga.server.acl#1.0:acl	no	A reference to an ACL to attach to the search

Sub-tags

Name	Type	Cardinality
parameter	urn:com.cohga.server.search.database#1.0:parameter	1..n
acl	urn:com.cohga.server.acl#1.0:acl	0..1
cache	urn:com.cohga.server.cache#1.0:cache	0..1

Content

None

Notes

An ACL can either be defined in-line or referenced indirectly, but only one should be used (the in-line version will take priority)

parameter

Properties

Name	Type	Required	Default	Description
id	string	yes		A unique identifier for the parameter
label	string	yes		The prompt text displayed when user input the parameter value
column	string	yes		The name of the column within the table that this parameter references
displayname	string	no	label	Provides a user-friendly name for the element
helptext	string	no		Additional text to display for the parameter to explain how to use the parameter
hidden	boolean	no	false	Hides the parameter from the parameter UI
alignment	'left', 'center', 'right', 'auto'	no	'auto'	How the items should appear in the UI
controltype	'listbox', 'checkbox', 'radiobutton', 'textbox'	no	'textbox'	The suggested type of UI control to use when displaying the parameter
datatype	'any', 'boolean', 'datetime', 'decimal', 'float', 'integer', 'string'	no	'string'	The data type for the parameter
allownull	boolean	no	false	Whether a null value is allowed for this parameter
allowblank	boolean	no	true	Give the user the choice of an empty value in the listbox (as opposed to a null value)
allownewvalues	boolean	no	false	Allow the user to enter values not in the listbox already
defaultvalue	any	no		The default value of the parameter
displayformat	string	no		The formatting instructions for the parameter value within the parameter UI
dataset	ref urn:com.cohga.server.data.database#1.0:datadefinition	no		Where to get the values for a listbox
labelcolumn	string	no		Column in the datadefinition that supplies the label of the value to show the user
valuecolumn	string	no		Column in the datadefinition that supplies the value of the value to use in the SQL
uppercase	boolean	no	false	Should the value be converted to upper case in the generated SQL
autoSelect	boolean	no	true	Should the first value in a list be pre-selected

Sub-tags

Name	Type	Cardinality
parameter	urn:com.cohga.server.data.database#1.0:parameter	0..n

Notes

- If the controltype is listbox then a dataset must be provided. This dataset will provide the values to be displayed in the listbox.
- If the listbox dataset only contains 1 column then that column will supply both the label and the value, if it contains two values then the first column will supply the label and the second the value.
- If a parameter contains another parameter then you are defining a cascading parameter, where setting the first sub-parameter will enable, and filter, the second parameter, and setting the second parameter will do the same for the third, etc.
- In a cascading parameter all sub-parameters must be of listbox type, and the dataset should be set in the outer parameter, not the sub parameters.
- In a cascading parameter only one level of nesting should be used.
- In a cascading parameter the dataset should supply the columns for all of the parameters, and valuecolumn properties should be set for each sub-parameter, and labelcolumn should be also set for all parameters if a different label is to be displayed to the user.

Spatial operation types

By default Weave will use a spatial intersection operation when performing the comparison between the source and target geometries. This can be changed by specifying a "spatialOperation" option in the config, see the example below.

The available options for the spatial operation setting are:

- intersect - The geometries have at least one point in common
- contains - One geometry contains another
- disjoint - The geometries have no points in common

- crosses - The geometries do more than touch, they actually overlap edges
- touches - The geometries only touch edges and do not overlap in any way
- within - One geometry is completely within another (no touching edges)

Examples

```

<data:datadefinition id="dd_wards">
  <datasourcedataconnection datasource="datasource.main" table="
WARDS" prefix="DISTINCT">
    <parameter name="name" column="WARD_NAME"/>
    <parameter name="code" column="WARD_CODE"/>
  </datasourcedataconnection>
</data:datadefinition>

<search:spatial id="property.byward">
  <displayname>by Ward Boundary</displayname>
  <description>Locate a property or properties by ward<
/description>
  <entity>property</entity>
  <sourceentity>ward</sourceentity>
  <parameter id="ward">
    <dataset>dd_wards</dataset>
    <label>Ward Name</label>
    <controltype>listbox</controltype>
    <column>CODE</column>
    <valuecolumn>code</valuecolumn>
    <labelcolumn>name</labelcolumn>
  </parameter>
</search:spatial>

```

Search with buffer

```

<search:spatial id="property.byward">
  <displayname>by Ward Boundary</displayname>
  <description>Locate a property or properties by ward<
/description>
  <entity>property</entity>
  <sourceentity>ward</sourceentity>
  <parameter id="ward">
    <dataset>dd_wards</dataset>
    <label>Ward Name</label>
    <controltype>listbox</controltype>
    <column>CODE</column>
    <valuecolumn>code</valuecolumn>
    <labelcolumn>name</labelcolumn>
  </parameter>
  <options:options>
    <!-- version 1.4.105 of com.cohga.server.search.
database and earlier -->
    <buffer:distance>-10</buffer:distance>
    <!-- version 1.5.105 of com.cohga.server.search.
database and later -->

```

```

        <buffer>-10</buffer>
        <bufferUnits>m</bufferUnits>
    </options:options>
</search:spatial>

```

Changing the spatial operation type

```

<search:spatial id="property.byward">
    <displayname>by Ward Boundary</displayname>
    <description>Locate a property or properties by ward<
/description>
    <entity>property</entity>
    <sourceentity>ward</sourceentity>
    <parameter id="ward">
        <dataset>dd_wards</dataset>
        <label>Ward Name</label>
        <controltype>listbox</controltype>
        <column>CODE</column>
        <valuecolumn>code</valuecolumn>
        <labelcolumn>name</labelcolumn>
    </parameter>
    <options:options>
        <spatialOperation>within</spatialOperation>
    </options:options>
</search:spatial>

```

Performing an "attribute" search against a spatial engine

```

<search:spatial id="ward">
    <displayname>by Ward Name</displayname>
    <description>Locate a ward by name</description>
    <entity>ward</entity>
    <parameter id="ward">
        <dataset>dd_wards</dataset>
        <label>Ward Name</label>
        <controltype>listbox</controltype>
        <column>CODE</column>
        <valuecolumn>code</valuecolumn>
        <labelcolumn>name</labelcolumn>
    </parameter>
</search:spatial>

```

Stored Procedure Search

A stored procedure search allows the server to locate entities using a stored procedure in a database.

The stored procedure must provide the server with a list of the identifiers that it should use to update the users selection.

There are three possible ways that the stored procedure can accomplish this:

- return the list as the result of the stored procedure
- return the list as an output parameter

- populate a table

If a `table` and `key` are provided in the configuration then the third method will be used, otherwise one of the first two methods will be used. The second method will be used for Oracle (since it doesn't support the first method) and the first method for all other database (unless a special option, `resultAsParameter`, is set to indicate that the second method should be used for non-Oracle databases).

In all cases the parameters the user enters will be passed as input parameters to the stored procedure, in the order they are listed in the configuration.

It is up to the stored procedure to determine if the user did not provide a value for a particular parameter, as the server will always attempt to set a value for every parameter defined in the configuration. If the user did not supply a value then `null` will be passed (note that the administrator can specify that a value is required for each parameter in the configuration to ensure that values are populated with a non-null value).

Returning results directly

If the database supports returning results directly then the list of id's to use for the selection can be returned in this manner.

Returning results indirectly

If the database does not support returning results from a stored procedure then the results can be provided via an output parameter (and the stored procedure returns nothing). This is the default for Oracle (unless a `table` and `key` are set).

Returning results via a table

If the stored procedure wishes to return results via a table then the `table` and `key` parameters must be set in the configuration, and they will specify the table name and the column which contain the id's used to update the selection.

It is the responsibility of the stored procedure to clear any rows belonging to the `userid` that it does not wish to return. That is, the server will not clear any old records from the table before calling the stored procedure.

In addition to this the stored procedure must also store a `userid` in the table to indicate which rows in the table correspond to which user. The value that should be stored in that column will be passed to the stored procedure as the first parameter (unless a special option, `userIdAtEnd`, is set to indicate it is the last parameter). The other input parameters for the search will follow the `userid`. Also, note that the name of the `userid` column can be changed by setting the `userIdColumn` spacial option.

Namespace

`urn:com.cohga.server.search.database#1.0`

Tags

procedure

Properties

Name	Type	Required	Description
id	string	yes	unique identifier
entity	ref urn:com.cohga.server.entity#1.0:entity	yes	The id of the entity that this search will be searching for
label	string	yes	Text to be displayed to the user to represent this search
description	string	no	Description of the search that could be displayed to the user to explain this search
datasource	ref urn:com.cohga.server.datasource.jdbc#1.0:datasource	yes	The data source that should be connected to to perform the search
procedure	string	yes	Name of the stored procedure to execute
table	string	no	The name of the table in the datasource that the stored procedure will populate to provide the resultant ids
key	string	no	The column in table that the stored procedure will populate to provide the resultant ids
acl	ref urn:com.cohga.server.acl#1.0:acl	no	A reference to an ACL to attach to the search

Sub-tags

Name	Type	Cardinality
parameter	<code>urn:com.cohga.server.search.database#1.0:parameter</code>	1..n
acl	urn:com.cohga.server.acl#1.0:acl	0..1
cache	urn:com.cohga.server.cache#1.0:cache	0..1

Content

None

Notes

An ACL can either be defined in-line or referenced indirectly, but only one should be used (the in-line version will take priority).

Table and key must both be included if one is

parameter

Properties

Name	Type	Required	Default	Description
id	string	yes		A unique identifier for the parameter
label	string	yes		The prompt text displayed when user input the parameter value
column	string	yes		The name of the column within the table that this parameter references
displayname	string	no	label	Provides a user-friendly name for the element
helptext	string	no		Additional text to display for the parameter to explain how to use the parameter
hidden	boolean	no	false	Hides the parameter from the parameter UI
alignment	'left', 'center', 'right', 'auto'	no	'auto'	How the items should appear in the UI
controltype	'listbox', 'checkbox', 'radiobutton', 'textbox'	no	'textbox'	The suggested type of UI control to use when displaying the parameter
datatype	'any', 'boolean', 'datetime', 'decimal', 'float', 'integer', 'string'	no	'string'	The data type for the parameter
allownull	boolean	no	false	Whether a null value is allowed for this parameter
allowblank	boolean	no	true	Give the user the choice of an empty value in the listbox (as opposed to a null value)
allownewvalues	boolean	no	false	Allow the user to enter values not in the listbox already
defaultvalue	any	no		The default value of the parameter
displayformat	string	no		the formatting instructions for the parameter value within the parameter UI
datadefinition	ref urn:com.cohga.server.data.database#1.0:datadefinition	no		Where to get the values for a listbox
labelcolumn	string	no		Column in the datadefinition that supplies the label of the value to show the user
valuecolumn	string	no		Column in the datadefinition that supplies the value of the value to use in the SQL
uppercase	boolean	no	false	Should the value be converted to upper case in the generated SQL

Sub-tags

Name	Type	Cardinality
parameter	urn:com.cohga.server.data.database#1.0:parameter	0..n

Notes

- If the controltype is listbox then a datadefinition must be provided, this datadefinition will provide the values to be displayed in the listbox.
- If the listbox datadefinition only contains 1 column then that column will supply both the label and the value. If it contains two values then the first column will supply the label and the second the value.
- If a parameter contains another parameter, then you are defining a cascading parameter where setting the first sub-parameter will enable, and filter, the second parameter, and setting the second parameter will do the same for the third, etc.
- In a cascading parameter all sub-parameters must be of listbox type, and the datadefinition should be set in the outer parameter, not the sub parameters.
- In a cascading parameter only one level of nesting should be used.
- In a cascading parameter the datadefinition should supply the columns for all of the parameters, and valuecolumn properties should be set for each sub-parameter, and labelcolumn should be also set for all parameters if a different label is to be displayed to the user.

Examples

Stored procedure that returns result directly

Assume a database that supports returning a selection (in this case SQL Server)

```
CREATE PROCEDURE RoadByNameDirect @roadname varchar(20) AS
  SELECT DISTINCT id FROM roads WHERE name=@roadname
```

Stored procedure that returns result as a parameter

In this case Oracle requires the REF CURSOR type to be defined in a separate package (rather than being referenced directly in the parameters). Well this is what I have found, YMMV, either way the parameter must be a reference to a cursor.

```
CREATE PACKAGE cursor_types AS
  TYPE cursorType IS REF CURSOR;
END;

CREATE PROCEDURE RoadByNameParameter(roadname IN roads.name%TYPE
  DEFAULT NULL, results OUT cursor_types.cursorType) IS
BEGIN
  OPEN results FOR
    SELECT DISTINCT id
    FROM roads
    WHERE name = roadname;
END;
```

Stored procedure that returns results in a table

```
CREATE PROCEDURE RoadByNameTable(user IN road_table.userid%TYPE,
  roadname IN roads.name%TYPE DEFAULT NULL) IS
BEGIN
  DELETE FROM road_table
    WHERE road_table.userid = user;

  INSERT INTO road_table(userid, id)
    SELECT userid, id
    FROM roads
    WHERE name = roadname;
  COMMIT;
END;
```

```
<search:procedure id="road_direct">
  <entity>road</entity>
  <displayName>by Stored Procedure (Direct)</displayName>
  <description>Stored procedure returns result directly<
/description>
  <dataSource>ds.main</dataSource>
  <procedure>RoadByNameDirect</procedure>
  <parameter id="roadname">
    <parameter:label>Road Name</parameter:label>
```

```

        </parameter>
</search:procedure>

<search:procedure id="road_parameter">
  <entity>road</entity>
  <displayName>by Stored Procedure (Parameter)</displayName>
  <description>Stored procedure returns result as output
parameter</description>
  <dataSource>ds.main</dataSource>
  <procedure>RoadByNameParameter</procedure>
  <parameter id="roadname">
    <label>Road Name</label>
  </parameter>
</search:procedure>

<search:procedure id="road_table">
  <entity>road</entity>
  <displayName>by Stored Procedure (Table)</displayName>
  <description>Stored procedure populates a table</description>
  <dataSource>ds.main</dataSource>
  <procedure>RoadByNameTable</procedure>
  <table>road_table</table>
  <key>id</key>
  <parameter id="roadname">
    <label>Road Name</label>
  </parameter>
</search:procedure>

```

Report

A report provides a link to a report design that the server can generate and display to the user.

Currently **BIRT** is the default report engine, and the BIRT designer should be used to create the report designs. To aid integration between Weave and BIRT there are two plugins available for BIRT that provide additional functionality, one to provide a map component for embedding a map in your report, and another for connecting BIRT to a Weave data definition which can then be used to supply data to the standard BIRT reporting components.

www.amazon.com provides additional support for developing BIRT reports as well as the online help available within the BIRT designer itself.

Once the reports designs are created they need to be copied to the Weave workspace directory (or a sub-directory, or referenced with a fully qualified filename) and a report entry added to the Weave configuration to reference the report design.

Namespace

urn:com.cohga.server.report.birt#1.0

Tags

report

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier
label	string	yes	User presented label for the report
design	string	yes	The name of the report design file
description	string	no	A description to show to the user
entity	string	no	The name of the entity that this report should be associated with

enablesFor	string	no	An expression that indicates the number of selected entities required for the report
group	string	no	User presented label for groups. Reports with same group label will appear together within the same sub menu

Sub-tags

Name	Type	Cardinality
acl	urn:com.cohga.server.acl#1.0:acl	0..1

Content

None

Notes

- An ACL can either be defined in-line or referenced indirectly, but only one should be used (the in-line version will take priority)
- enablesFor can be one of:

!	size = 0
?	size = 0 or 1
+	size >= 1
*	size >= 0
n	size = n
n+	size >= n
n-	size >= n
-m	size <= m
n-m	n <= size <= m

reports

Properties

Name	Type	Required	Description
id	string	yes	Unique identifier pre-pended to report design name to generate report id
dir	string	yes	Directory name containing report to be added, either relative to workspace or absolute
entity	string	no	The name of the entity that these reports should be associated with
enablesFor	string	no	An expression that indicates the number of selected entities required for the reports

Sub-tags

Name	Type	Cardinality
acl	urn:com.cohga.server.acl#1.0:acl	0..1

Content

None

Notes

- An ACL can either be defined in-line or referenced indirectly, but only one should be used (the in-line version will take priority)
- The label and description for the reports will be extracted from the report design itself
- Both entity and group attributes can be specified together. When both attributes present, groups sub menus are created inside entity sub menu within the reports menu.

Examples

```

<birt:report id="property_report" label="Property Report" entity="
property" group="Special Reports" design="proprpt.rptdesign"
description="Provides a full property report for the selected
property" enablesFor="1"/>

<birt:report id="pothole_report" label="Potholes" design="potholes.
rptdesign" description="Summary report of currently reported potholes"
>
    <acl:acl>
        <entry type="deny">anonymous</entry>
        <entry type="allow">*</entry>
    </acl:acl>
</birt:report>

<birt:reports id="property_reports" dir="reports/property" entity="
property"/>

```

Emailing Reports

With release 2.4.20 of Weave there has been some basic report emailing capability added, which is expected to be expanded upon over time.

At the moment the capabilities are limited to the simple report menu and simple report action, and is *not* supported by reports generated via the report view.

Additionally the emailing is currently limited to sending the email to a fixed address, this is because the initial use case for this operation was to submit information to a single entity, rather than being a general purpose emailing facility.

Additional functionality will hopefully come as the UI/configuration changes are worked out relating to how this will be configured and where the email options will fit in with the existing report generation UI.

To configure the ability to send an email containing a report requires up to three things to be setup:

1. The setting for a SMTP server that will be used to send the emails.
2. Optional settings for the email itself, for things like the from address, to address, email subject, etc.
3. The simple report action and/or simple report menu must be updated to include the email information, either directly, in which case the previous step isn't required or indirectly by referencing the information setup in the previous step

Setting up a SMTP server

To be able to send any emails from Weave you must provide Weave with the details of a mail server that will be used to send the emails. This is done using the `com.cohga.server.mail` configuration options

Basic mail server setup

```

<?xml version="1.0" encoding="UTF-8"?>

<config          xmlns="urn:com.cohga.server.config#1.0" xmlns:mail="
urn:com.cohga.server.mail#1.0">

    <mail:smtp>
        <host>mail.sforbes.net</host>
        <username>sforbes</username>
        <password>ENCEAUFJCUABCFXEMFQABFJEBZJAKFCXGME<
/Password>
    </mail:smtp>

```

```
</config>
```

The `smtp` config requires at least the `host` property, to provide the hostname or ip address of the mail server. It may also requires the `user` name and `password` setting if the mail server requires authentication before it will send any email (the password can be encrypted using the `encrypt` command and the `osgi` prompt).

Additionally any setting supported by the Java Mail API may also be specified, by removing the `mail.smtp.` prefix from the setting and including it in the `smtp` config.

The available Java Mail API setting are available [here](#).

There should only ever be one SMTP server configured for Weave.
Multiple components that support emailing will all use the same SMTP server to send their emails.

Setting up email settings

If you're going to have more than one report supporting emailing then it's probably a good option to set the default options for the email in a separate report email configuration item and then reference it in the report definitions. Alternatively you can skip this and apply those settings directly to the report item in the client config.

The default values that can be setup in an email config are the `from` address, the `to` addresses, the `cc` addresses, the `bcc` addresses, the `email subject` and the `email message`.

Basic emails settings

```
<?xml version="1.0" encoding="UTF-8"?>

<config      xmlns="urn:com.cohga.server.config#1.0" xmlns:report="
urn:com.cohga.server.report#1.0">

    <report:email id="feedback">
        <from>noreply@sforbes.net</from>
        <to>feedback@sforbes.net</to>
    </report:email>

</config>
```

The minimum settings required are a `from` and `to` address.

You can specify multiple `to` addresses (and `cc` and `bcc` addresses) by either including multiple tags or comma separating the addresses in one tag.

There are a couple of special markers that can be used in the addresses, they are `${userid}` and `${domain}`.

They allow you to substitute the current `userid` and the servers domain into the address. Note that the domain is guessed based on the canonical host name of the Weave server, and defaults to 'local' if it can't be determined.

So you could use the following to try and guess the `from` address of the user

From address from userid and hosts domain

```
<from>${userid}@${domain}</from>
```

or

From address from userid with a fixed domain

```
<from>${userid}@sforbes.net</from>
```

It's possible to use the `${userid}` (and `${domain}`) value in the `to` address, which would allow the user to email the report to themselves, but this hadn't been tested. I just thought of it while writing the above documentation.

The default `subject` used to send the email is "Weave Report", which can be changed by either setting a `subject` property in the `email` config or by setting the `report.email.subject` i18n resource value.

Similarly the default `message`, which is only included when a report is sent as an attachment as would be the case if the report was generated as a PDF document, is "A report has been generated and sent to you", and can be changed by setting the `message` property or by setting the `report.email.message` i18n resource value.

Setting subject and message

```
<?xml version="1.0" encoding="UTF-8"?>

<config      xmlns="urn:com.cohga.server.config#1.0" xmlns:report="
urn:com.cohga.server.report#1.0">

    <report:email id="feedback">
        <from>noreply@sforbes.net</from>
        <to>feedback@sforbes.net</to>
        <subject>Feedback</subject>
        <message>The following has been submitted as a
feedback report</message>
    </report:email>

</config>
```

Setting subject and message using i18n resources

```
<?xml version="1.0" encoding="UTF-8"?>

<config      xmlns="urn:com.cohga.server.config#1.0" xmlns:report="
urn:com.cohga.server.report#1.0" xmlns:client="urn:com.cohga.html.
client#1.0">

    <client:resources>
        <resource id="report.email.subject">Feedback<
/resource>
        <resource id="report.email.message">The following has
been submitted as a feedback report</resource>
    </client:resources>

    <client:resources lang="it">
        <resource id="report.email.subject">Retroazione<
/resource>
        <resource id="report.email.message">È stata
```



```

presentata la seguente come un rapporto di feedback</resource>
  </client:resources>

  <report:email id="feedback">
    <from>noreply@sforbes.net</from>
    <to>feedback@sforbes.net</to>
  </report:email>

</config>

```

In the case of using the i18n resources the language used to choose the values will be based on the user using Weave, not the person receiving the email

Updating the report component

Once the smtp server and email configuration has been setup it's just a matter of referencing the email config in a simple report action or menu

Simple report action to email a report

```

<item action="com.cohga.client.action.simplereport">
  <email>feedback</email>
  <format>html</format>
  <report>br_map</report>
  <label>Feedback</label>
  <tooltip>
    <title>Feedback</title>
    <text>Send a feedback report using the
current map</text>
  </tooltip>
</item>

```

The above report item will email the `br_map` report in HTML format directly to the user listed in the feedback email config. If the report has any parameters then the user will be asked to enter the parameters before the report is sent, but if the report doesn't have any parameters then the report will be sent straight away. Similarly, because the report config specifies a format that format will be used and the user not asked to pick a particular format, but if the format wasn't included the user would be asked to select the format before the report was emailed.

Specifying email properties directly in the report action

```

<item action="com.cohga.client.action.simplereport">
  <email>
    <from>${userid}@sforbes.net</from>
    <to>feedback@sforbes.net</from>
  </email>
  <format>html</format>
  <report>br_map</report>
  <label>Feedback</label>
  <tooltip>
    <title>Feedback</title>

```

```

                <text>Send a feedback report using the
current map</text>
            </tooltip>
        </item>

```

The above example does not require the `email` config at all, the values are directly set in the report action, using the default values for those properties that are not set.

Extending email properties

```

        <item action="com.cohga.client.action.simplereport">
            <email id="feedback">
                <from>${userid}@sforbes.net</from>
                <to>feedback@sforbes.net</from>
            </email>
            <format>html</format>
            <report>br_map</report>
            <label>Feedback</label>
            <tooltip>
                <title>Feedback</title>
                <text>Send a feedback report using the
current map</text>
            </tooltip>
        </item>

```

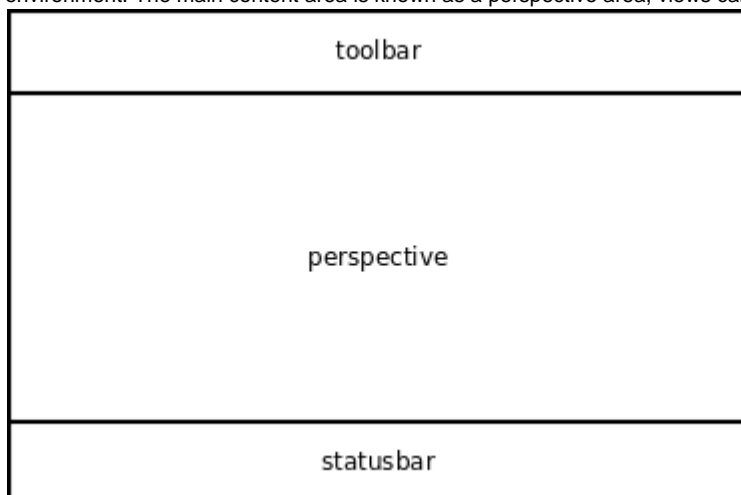
The above example does use the `feedback` `email` config, and replaces the values from it with those included in the report item, using those from the email config where the aren't set directly.

Client

Administering the client at a basic level is all done through the configuration xml file. The configuration allows multiple clients with different components depending on the user base of the client.

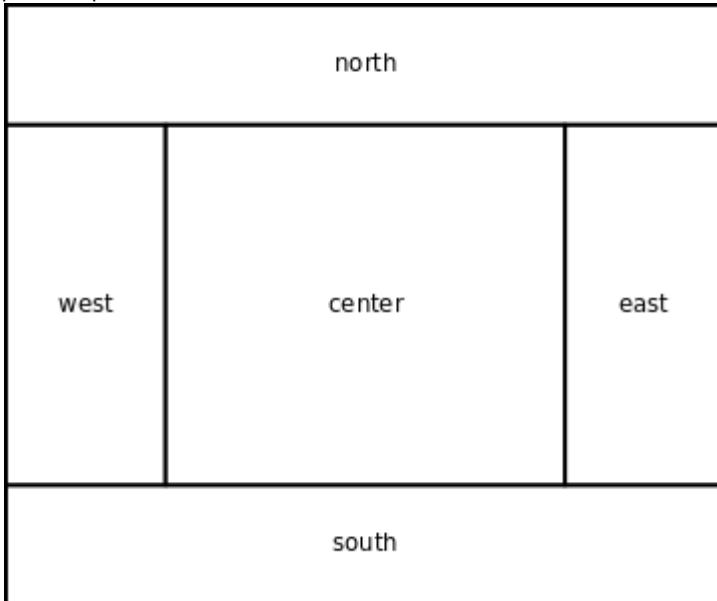
Layouts

The client is made up of three main parts, the toolbar, statusbar and the main content area. The toolbar is located at the top and the statusbar at the bottom. The administrator is free to move actions and components around the interface in order to customise the environment. The main content area is known as a perspective area, views can be added to a border layout within this area.



A border layout defines 5 regions that act as containers on the screen for views. The five areas are North (top), South (bottom), West (left), East (right) and Center (middle). Center is the only container that will dynamically resize all borders to fit within the area, all other regions only resize in one direction. For example, East and West Regions will resize their height depending on the size of the Application Window.

North and South regions have fixed heights and will only resize their width depending on the size of the Application Window. This does not mean that the user does not have the ability to resize the views if allowed by the administrator, just that the initial size of the non-center panels is pre-determined.



Main Views can be added to a particular region. For example you might want the TOC and Search Views to be added to the West region. When multiple views are added to a region then they are added as tabs given users the ability to move between the Views. An alternative options allows the administrator to configure the region to use an accordion type layout rather than a tab based one.

The client now also allow for nesting regions within a region, so now each region can also be sub-divided into north, south, east, west and center. Also, regions can now be named, so they can be referred to by a logical name when adding views rather than having to refer to their physical location. See the section on [Layouts](#) for more information.

Both the Statusbar and Toolbar support adding both Actions and Components to them. Toolbars can also be added to some Views, for example the [Map View](#). See the [Toolbars](#) and [Actions](#) sections for more information.

Client Configuration

Weave supports the ability to configure multiple clients from one instance. Each Client needs to have a unique ID that is used in the url when opening the client.

```

...
<client:config id="main">
  ...
</client:config>

<client:config id="police">
  ...
</client:config>
...

```

In the above example there are two clients defined and each has a different id. When using the client open a browser and point to

`http://server:port/weave/main.html`

or

`http://server:port/weave/police.html`

Additionally, if there is more than one client configuration and the user views

`http://server:port/weave`

they will be presented with a list of the available client configurations.

Access Control

By adding an ACL to the client configuration you can restrict access to the entire client configuration based on the access level of the user.

For example to restrict the `police` configuration to only those users who are in the police access control list you would change the configuration to:

```
...
<client:config id="police">
  <acl id="acl.police">
    ...
</client:config>
...
```

this assumes that elsewhere in your configuration you have the ACL for `acl.police`, something like:

```
<acl:acl id="acl.police">
  <entry type="allow">ROLE_POLICE</entry>
  <entry type="deny">*</deny>
</acl:acl>
```

You should consult the section on [Access Control Lists](#) for more information.

Alternatively, you can add an ACL to individual items within the client configuration to have those items only available to users that have the appropriate level of access.

For example, to provide a generic client that has a single button that should only be available to users in a certain group you could either create two separate client configurations, one with the button for the selected users and the other without the button for everyone else, or you could create one configuration with the button but attach an ACL to the button.

```
...
<client:config id="main">
  ...
  <toolbar>
    ...
    <item action="au.gov.police.ShowSpeedCameras">
      <acl id="acl.police"/>
    </item>
    ...
  </toolbar>
  ...
</client:config>
...
```

This way anyone can access the `main` client configuration, but only those users that login and pass the `acl.police` ACL will be able to see the `au.gov.police.ShowSpeedCameras` button.

The ACL can be attached to anything within the client configuration, not just buttons/actions. When a user does not pass the ACL then the entire content of the tag containing the ACL tag is removed.

Additionally, you can have the same item added twice, each with a mutually exclusive ACL attached so that all users see one version and others see another.

This can be used for example to alter the map area that a user is allowed to view, if you have multiple groups of users, where each one should only be able to view part of the overall map then you could setup multiple extent tags for the map view and attach different ACLs to each one so that in the end each user only receives one of the available extents and that's the area they're limited to.

Structure

The actual content of the client configuration should follow the same basic outline regardless of the final layout of the client.

The absolute minimum requirements for a client configuration is shown below (where `CONFIGID`, `TITLE` and `LABEL` are configured by you)

```
<client:config id="CONFIGID">
  <title>TITLE</title>
  <perspective>
    <label>LABEL</label>
  </perspective>
</client:config>
```

Obviously, this creates a pretty useless client experience as it creates single center region with nothing in it.

So the next this we could do would be to add content to the default center region that was created. We do this by adding Views to the perspective. A View is responsible for presenting its content the the user in whatever way is appropriate, it could be a map image, a text input form or just a fixed image. What's shown in the region is different depending upon the id of the view that's added to the perspective, so for now we won't bother looking at the actual content of the view tags.

```
<client:config id="CONFIGID">
  <title>TITLE</title>
  <perspective>
    <label>LABEL</label>
    <view id="VIEWID">
      VIEWCONTENT
    </view>
  </perspective>
</client:config>
```

Once we've added the view we would expect to see the content of the view displayed in the center region.

From there we can add additional views to show different content to the user.

```
<client:config id="CONFIGID">
  <title>TITLE</title>
  <perspective>
    <label>LABEL</label>
    <view id="VIEWID1">
      VIEW1CONTENT
    </view>
    <view id="VIEWID2">
      VIEW2CONTENT
    </view>
  </perspective>
</client:config>
```

The user will then be able to switch between displaying the two views because by default views are added to the center region and that region is configured to display multiple views using tabs.

This brings us on to layouts, which are covered in more detail at [Layouts](#), but we'll cover them here to show where they fit within the client configuration.

Say we wanted to move one of our views to be to the left of the other one, rather than having them both within the same region. To do this we'd add a layout to the perspective and then configure each of the views to be placed in a specific region within the layout. Remember at the moment there's a default layout created for us that has a single center region defined.

```
<client:config id="CONFIGID">
  <title>TITLE</title>
```

```

<perspective>
  <label>LABEL</label>
  <layout>
    <west width="WIDTH">
      <center>
    </center>
  </layout>
  <view id="VIEWID1" location="west">
    VIEW1CONTENT
  </view>
  <view id="VIEWID2">
    VIEW2CONTENT
  </view>
</perspective>
</client:config>

```

Once we have the new layout in place the first view will appear on the left and the second on the right, with the left view taking up an amount of room determined by the value of WIDTH and the second filling in the rest. The second view will still go into the center region since that's the default if none is explicitly set, and there must always be a center view defined within a layout.

From there we can fill out the client configuration with some other components. We can add a toolbar and statusbar (as mentioned earlier).

```

<client:config id="CONFIGID">
  <title>TITLE</title>
  <toolbar>
    TOOLBARCONTENT
  </toolbar>
  <statusbar>
    STATUSBARCONTENT
  </statusbar>
  <perspective>
    <label>LABEL</label>
    <layout>
      <west width="WIDTH">
        <center>
      </center>
    </layout>
    <view id="VIEWID1" location="west">
      VIEW1CONTENT
    </view>
    <view id="VIEWID2">
      VIEW2CONTENT
    </view>
  </perspective>
</client:config>

```

These will always be shown at the top and bottom of the page irrespective of the layout. If you have more than one perspective then a perspective switcher component would be an obvious choice to include in the top-level toolbar or statusbar. Note that perspectives can also have their own toolbar and/or statusbar, as can views (but it's up to the implementation of the view to provide support for it).

We can also include a set of default values at the same level as perspectives to provide default values for various components that are included on the client, for example indicating the initial map extent to use if a map view doesn't specify it explicitly, or determining if a report should be opened in a new browser window.

```

<client:config id="CONFIGID">
  <title>TITLE</title>

```

```

<toolbar>
  TOOLBARCONTENT
</toolbar>
<statusbar>
  STATUSBARCONTENT
</statusbar>
<perspective>
  <label>LABEL</label>
  <layout>
    <west width="WIDTH">
      <center>
    </center>
  </layout>
  <view id="VIEWID1" location="west">
    VIEW1CONTENT
  </view>
  <view id="VIEWID2">
    VIEW2CONTENT
  </view>
</perspective>
<defaults>
  DEFAULTCONTENT
</defaults>
</client:config>

```

Again the contents of the default section are determined by what's being set, but generally, there could be a report section, a map section and a section for entities.

Aside from that, there are a couple of other minor items that could be added, one is a description, that's shown to the user if they get the option to choose between multiple client configurations. Another is a debug flag that indicates that the client should be loaded in debug mode which helps when modifying the client configuration. And finally, it's possible to specify a theme to be used for the configuration.

```

<client:config id="CONFIGID">
  <title>TITLE</title>
  <description>DESCRIPTION</description>
  <debug>true</debug>
  <theme>THEMENAME</theme>
  <toolbar>
    TOOLBARCONTENT
  </toolbar>
  <statusbar>
    STATUSBARCONTENT
  </statusbar>
  <perspective>
    <label>LABEL</label>
    <layout>
      <west width="WIDTH">
        <center>
      </center>
    </layout>
    <view id="VIEWID1" location="west">
      VIEW1CONTENT
    </view>
    <view id="VIEWID2">
      VIEW2CONTENT
    </view>
  </perspective>
</client:config>

```

```

</perspective>
<defaults>
  DEFAULTCONTENT
</defaults>
</client:config>

```

Defaults

You can set various default options in the `defaults` section in the client configuration. For example, the default report format to generate, which entity to have active initially or which data to display when the user open a data grid.

```

<client:config id="CONFIGID">
  ...
  <perspective>
    ..
  </perspective>
  <defaults>
    <entities>
      <entity id="property" isDefault="true">
        <data>data.property.detail</data> <!-- set the default data
definition for properties -->
        <search>search.property.byaddress</search> <!-- set the
default search for properties -->
      </entity>
      <entity id="roads">
        <data>data.road.detail</data> <!-- set the default data
definition for roads -->
        <search>search.road.byname</search> <!-- set the default
search for roads -->
      </entity>
      <entity id="parks" remove="true"/> <!-- remove parks as being
an available entity -->
      <entity id="lights" remove="true"/> <!-- remove lights as being
an available entity -->
    </entities>
    <report>
      <format>html</format>
      <openExternal>true</openExternal>
      <openWithScript>true</openWithScript>
      <timeout>240</timeout>
    </report>
  </defaults>
</client:config>

```

In the above example we've set the property entity as the default, and set the initial search for properties to be `search.property.byaddress` and the initial data to be `data.property.detail`. For roads we've set the search to `search.road.byname` and the data to `data.road.detail`. We've also removed the parks and lights entities from being available to the user. The user will have access to all the other available entities, but they will have the default settings (for default search, data definition, etc).

We've also specified that the default report format will be `html`, `format`, report will open in a new window (or tab) rather than a window embedded within the weave client itself, `openExternal`, and it'll use JavaScript to open the window, bypassing the intermediate 'click to open' popup window, `openWithScript`, and finally increased the report timeout to 240 seconds (from the default of 120 seconds), `timeout`.

You can specify exactly what entities are available on the client for a user by specifying the name of a user attribute that contains the id's of the entities that should be available.

Setting entities based on a user attribute

```

<client:config id="CONFIGID">
  ...
  <perspective>
    ..
  </perspective>
  <defaults>
    <entities userattribute="entities"/>
    ....
  </defaults>
</client:config>

```

In the above example the user attribute named `entities` should contain one or more entity id's and the user will have access to only those entities. In this situation you can still specify individual entity settings, for example the default search, within the `entities` tag (they just haven't been shown in this example), but you can also specify that information directly within the entity configuration rather than here in the default section.

It's also possible to have the list of entities listed in the `entities` section refine the available entities directly. By default specifying multiple `entity` tags within the `entities` tag just modifies the settings for those entities, but by setting `filter` to `true` for in the `entities` tag you can further specify that these entities listed will be the only ones available to the client.

Refining what entities are available

```

<client:config id="CONFIGID">
  ...
  <perspective>
    ..
  </perspective>
  <defaults>
    <entities filter="true"> <!-- user will only have access to
property, roads and parks -->
      <entity id="property" isDefault="true">
        <data>data.property.detail</data>
        <search>search.property.byaddress</search>
      </entity>
      <entity id="roads">
        <data>data.road.detail</data>
        <search>search.road.byname</search>
      </entity>
      <entity id="parks"/>
      <entity id="lights"/>
    </entities>
    ....
  </defaults>
</client:config>

```

In this example users will *only* have access to the property, roads, parks and lights entities, with some additional settings being applied to the property and roads.

Icons

You can create custom icons that can then be referred to in the client configuration by creating an `icons` directory within the `...` `\weave\platform\workspace` directory, then any `.gif` or `.png` files within there will be available to use anywhere an `iconCls` attribute

can be set. Note that if the `icons` directory doesn't exist when the server is started it will need to be restarted before icons are available for use.

The icons aren't available directly, instead Weave will create a CSS rule to reference the image, and it's the CSS rule that you reference in the config. The name of the rule is based on the file name, but is given an `icon-` prefix, so for example `test.png` would create a rule named `.icon-test` which can then be referenced as `icon-test` in the config file.

It's assumed that if there are two icons with the same name, and one a `.png` and the other `.gif`, then the `.gif` will be used for Internet Explorer and the `.png` for other browsers, this is because of the poor support for `.png` images in earlier IE versions. In this case, the icon is still referred to be the same name in the config files, the rule just ensures that the correct image is chosen at runtime.

Note that this is no longer strictly true and Internet Explorer has improved the level of support for `.png` images, so you should always use `.png` images rather than `.gif`

To see what current icons are available for use, open the HTML page `/weave/styles/core.css` on the Weave instance e.g. `http://localhost:8080/weave/styles/core.css` and any line starting with `.icon-` will be an icon available for use.

The actual bitmaps for most of these will be contained in the `com.cohga.client.weave.main_*.jar` bundle (which is just a `.zip` file), and they're stored under the `/client/resources/images/icons/` directory inside that file.

Not all bitmaps are in that one file, others may be supplied by other bundles, but they should all be in the same directory, but any decent browser should provide an option to directly peruse the icons anyway.

Pretty much any component can have its icon set by setting the `iconCls` property to one of these icons, e.g. to use `.icon-edit` in a toolbar:

```
<item action="weave.toggleToolbar" text="Edit" iconCls="icon-edit">
  ...
</item>
```

If you want to remove the icon for a component set the `iconCls` property to be blank, e.g. to remove the icon from the Legend panel in your layout:

```
<view id="com.cohga.client.panel.legend" iconCls="">
  <label>Legend</label>
  <location>west</location>
  ...
</view>
```

Additionally, for the text/labelling, it can be a bit inconsistent, but a general run of thumb is to set a 'text', or 'label', property to add/alter the default text for a button/panel/component/control.

Aliases

As of version 2.22.9 of the `com.cohga.client.weave` bundle it's possible to specify an alias for a client configuration.

```
<client:config id="client">
  <alias id="alias"/>

  <!-- Other config items here -->
</client:config>
```

At the minimum this provides exactly the same client configuration under an additional url, so the above configuration would be available at `/weave/client.html`
and
`/weave/alias.html`

This by itself isn't much, but by allowing you to publish the same client config under different url's you can provide different security contexts for them.

That is the `security.xml` file can now specify different security filters to be applied to the same client config (this is because the security chain is based on the initial URL).

This means it's possible to have a single client config, and point one group of users to one url to access it, where they'll need to login with a username and password, and have another group of people access the same client config, via a different url, where NTLM authentication is used.

The security implications described above rely on changes also being made to the `security.xml`, and will probably require advanced configuration of the `security.xml` file.

Additionally you can override some of the configurations in the client config. For example:

```
<client:config id="external">
  <alias id="internal">
    <title>Internal Client</title>
    <description>Client for access by internal users</description>
    <license xsi:nil="true"/>
  </alias>

  <title>External Client</title>
  <description>Client for access by external users</description>
  <license>
    <title>License</title>
    <text>By clicking OK you agree ....</text>
  </license>

  <!-- Other config items here -->
</client:config>
```

would be the equivalent of:

```
<client:config id="external">
  <title>External Client</title>
  <description>Client for access by external users</description>
  <license>
    <title>License</title>
    <text>By clicking OK you agree ....</text>
  </license>

  <!-- Other config items here -->
</client:config>

<client:config id="internal">
  <title>Internal Client</title>
  <description>Client for access by internal users</description>

  <!-- Other config items here -->
</client:config>
```

This example will alter the title and description for the `internal` version of the client config, plus it'll remove the license screen that the external user normally would have to click on.

You can also specify an ACL in the alias, but it doesn't affect the alias itself, as is the case if an ACL is attached to any other item in a client config, rather it'll replace the ACL for the client as a whole.

This means the following:

```
<client:config id="external">
  <alias id="internal">
    <title>Internal Client</title>
    <description>Client for access by internal users</description>
    <license xsi:nil="true"/>
    <acl id="internal"/>
  </alias>

  <title>External Client</title>
  <description>Client for access by external users</description>
  <license>
    <title>License</title>
    <text>By clicking OK you agree ....</text>
  </license>
  <acl id="external"/>

  <!-- Other config items here -->
</client:config>
```

is equivalent to:

```
<client:config id="external">
  <title>External Client</title>
  <description>Client for access by external users</description>
  <license>
    <title>License</title>
    <text>By clicking OK you agree ....</text>
  </license>
  <acl id="external"/>

  <!-- Other config items here -->
</client:config>

<client:config id="internal">
  <title>Internal Client</title>
  <description>Client for access by internal users</description>
  <acl id="internal"/>

  <!-- Other config items here -->
</client:config>
```

Some other things you could do would be to set the `publish` flag to false in the alias configuration so that the aliased client configuration won't show up in the users list of available configs (but they can still access it using a direct url), which is handy for client configurations that are used for third party integration.

```

<client:config id="main">
  <alias id="pathway">
    <title>Pathway Client</title>
    <description>Client to be used when called from Pathway<
/description>
    <license xsi:nil="true"/>
    <publish>false</publish>
  </alias>

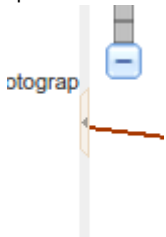
  <title>Default Client</title>
  <description>General client for use by users</description>
  <license>
    <title>License</title>
    <text>By clicking OK you agree ....</text>
  </license>

  <!-- Other config items here -->
</client:config>

```

Adjusting the panel collapse buttons

The buttons used to expand and collapse the panel can be too small for some users, but since these buttons are just images they can be updated to use a different image.



Here is a bundle that contains a set of replacement buttons, [com.cohga.client.weave.themes.collapse_1.0.0.jar](#), that you can use to alter the appearance.



Installing the new expand/collapse button theme

To use the updated buttons you need to download the [com.cohga.client.weave.themes.collapse_1.0.0.jar](#) file and copy it to the ... \weave\platform\plugins\ directory.

Once you've done that you have to edit the ... \weave\platform\configuration\config.ini file to add the file you just downloaded to the list of bundles that Weave will start automatically each time Weave is restarted.

The image below shows the config.ini file after it's been edited, the line that needs to be added has been highlighted in the image. In this case, the line has been added after the lines that list the grey and olive themes that ship with Weave, the line that's added to config.ini is the line containing com.cohga.client.weave.themes.collapse@:start,\

```

com.cohga.client.weave.snap@:start,\
com.cohga.client.weave.coordzoom@:start,\
com.cohga.client.weave.themes.grey@:start,\
com.cohga.client.weave.themes.olive@:start,\
com.cohga.client.weave.themes.collapse@:start,\
com.cohga.client.quicksearch@:start.\

```

After downloading and installing the file, and editing `config.ini` you should restart the Weave instance to ensure the new bundle is started and available for use.

Using the new expand/collapse theme in your client

After you have the new theme bundle installed and started you have to add the theme to the client config, along with any existing theme you're currently using.

The image below shows an existing client config that has been changed to add the line highlighted below. Note that your client config will **not** match the example below, the one below is just an example to show you where in *your* client config(s) you'd need to add the highlighted line to enable the larger expand/collapse buttons.

In the following example, the client config is already setup to use the `grey` theme, you may not be using a theme at all, in which case you won't have an existing `theme` entry and you should just add a new `theme` tag for the expand/collapse buttons.

```
<client:config id="collapse">
  <title>Collapse Button Example</title>
  <debug>true</debug>
  <theme>grey</theme>
  <theme>collapse</theme>
  <perspective>
    <label>Main</label>
    <lavout>
```

As of Weave 2.5.18 this bundle is already included and setup so all you need to do is the last step to add the `collapse` theme to your client config.

Licence agreement, startup tips and splash screens

It's possible to have the client display a licence that the user must agree to before continuing or to display some general text, by adding one of the following configs to the client file.

Displaying a licence agreement

```
<client:config id="main">
  <licence>
    <title>Licence Agreement</title> <!-- "Licence
Agreement" is the default so this is not strictly required -->
    <text>This is where you would insert the agreement
text you wish the user to confirm.</text> <!-- this must be set -->
    <url>http://example.com</url> <!-- optional link if
the user does not agree -->
    <usetop>true</usetop> <!-- Since Weave 2.5.26: When
Weave is embedded inside an IFrame, indicate that the top-level
window has to be changed to the given URL. -->
  </licence>

  <!-- rest of client config goes here -->
</client:config>
```

Licence Agreement



This is where you would insert the agreement text you wish the user to confirm.

Displaying a startup tip

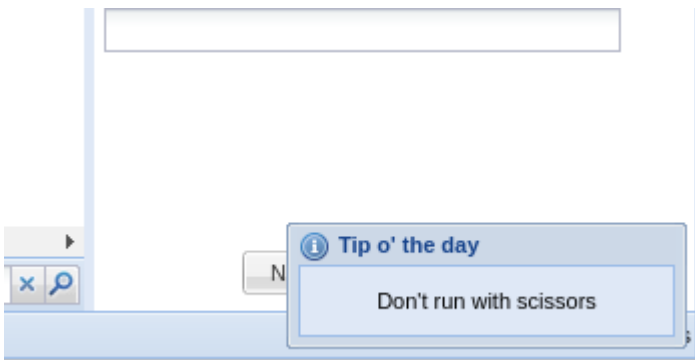
```

<client:config id="main">
  <tip>
    <delay>1000</delay> <!-- how long before the tip
should popup, default is 1000ms so this is not strictly required -->
    <title>Tip o' the day</title> <!-- title for the tip
window -->
    <text>Don't run with scissors</text> <!-- required
text for the tip -->
    <time>3000</time> <!-- how long the tip should show --
  >
  </tip>

  <!-- rest of client config goes here -->

</client:config>

```

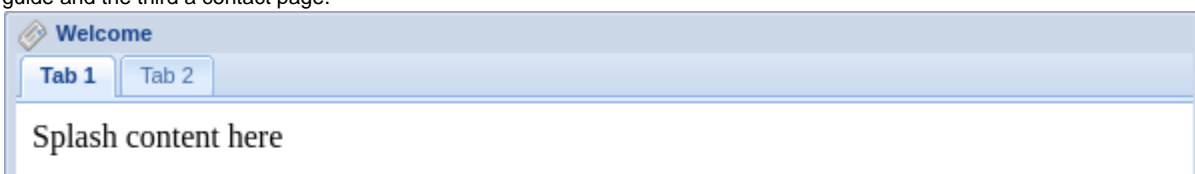


Splash screens

New in 2.6.7 is the splash screen, which has the combined function of the licence and startup tip. A splash screen provides a way to display information to the user at startup similar in appearance to the licence agreement above but provides additional options to customise the experience.



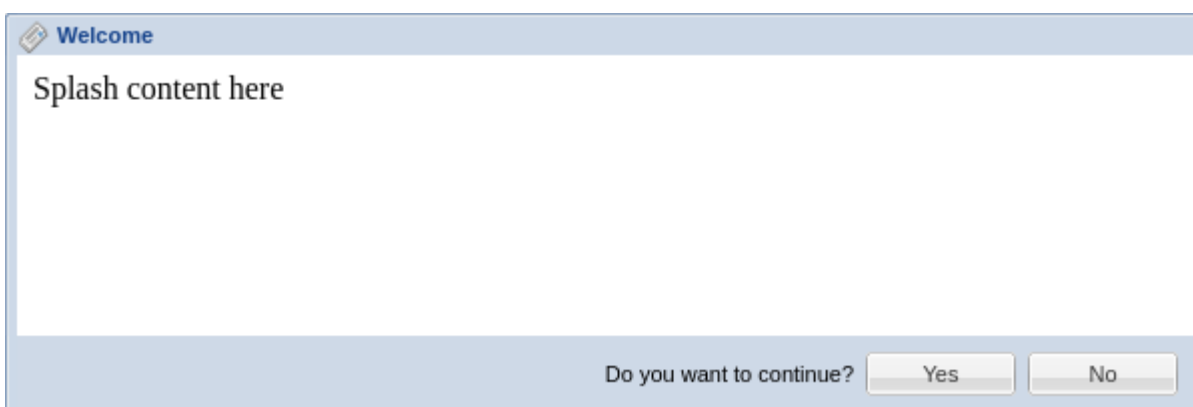
One major difference between the splash screens and the licence panel/tips, is that you have to supply HTML content to be displayed. The content can either be embedded directly in the configuration or provided as a URL pointing to a web page to display. Note that this isn't such an issue now because 2.6.7 also provides a convenient method of publishing your own custom HTML directly from Weave by storing the HTML in the `...\weave\platform\workspace\static\` directory. That HTML will then be available at the URL `http://example.com/weave/static/`. This allows much more control and flexibility with the content of the window, and because it directly supports multiple tabs, you can provide separate sections with different content. For example, the first tab can be a welcome text, the second a help guide and the third a contact page.





By providing a redirect URL as part of the configuration you can change the simple *Close* button into a choice between *Yes* and *No* and make this a replacement for the licence panel (as shown below).

```
<plugin id="weave.splash">
  <html>Splash content here</html>
  <redirectUrl>https://www.cohga.com/</redirectUrl>
</plugin>
```

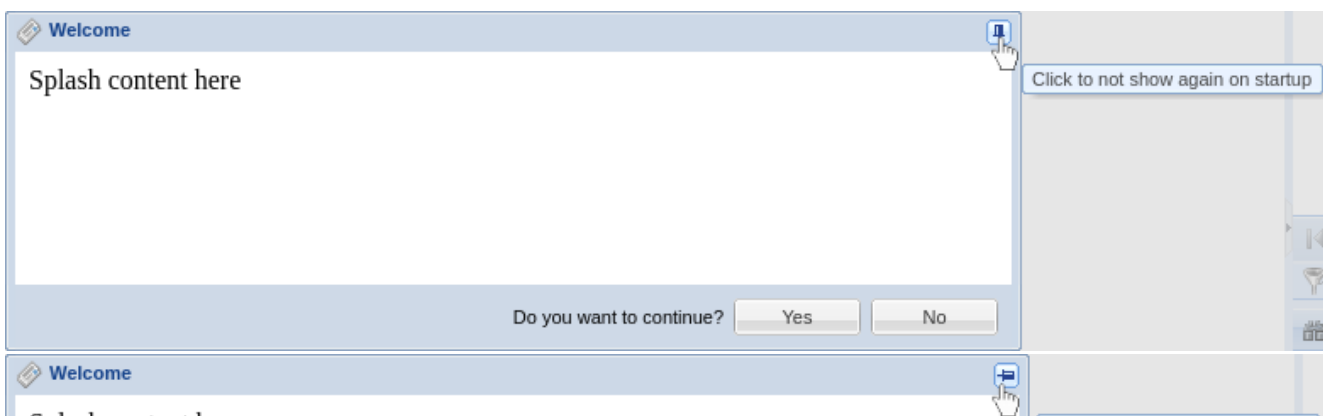


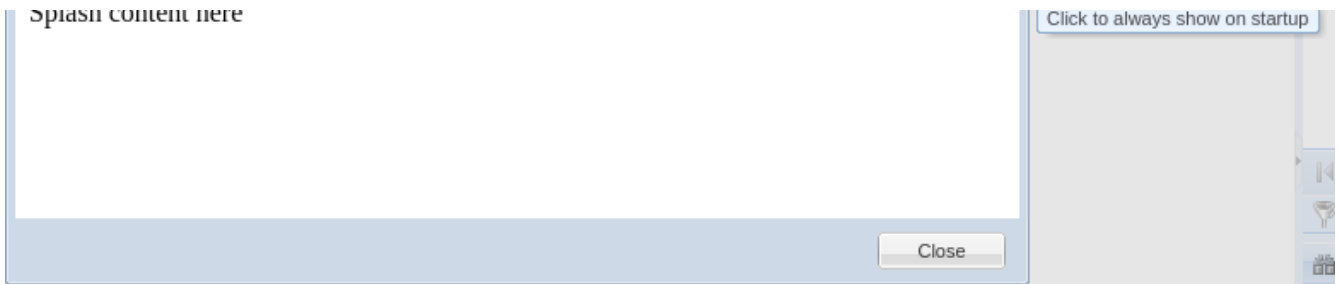
If the user clicks *Yes*, the panel will close. If they click *No*, they will be redirected to where ever you set the redirect URL.

In addition, you can provide the user with an option not to display the splash screen again. This can be used in combination with a button added to a toolbar/statusbar to allow the user to open the panel again if they want (as shown below).

```
<plugin id="weave.splash">
  <html>Splash content here</html>
  <redirectUrl>https://www.cohga.com/</redirectUrl>
  <allowHide>true</allowHide>
</plugin>

<toolbar>
  <item action="weave.showsplash" />
</toolbar>
```





Finally, you have the option to configure the splash panel to display only once. So once it's closed the first time it won't display on startup again unless you provide access to the button to re-show the splash panel (see toolbar item above) or update the config to give the panel a different id.

```
<plugin id="weave.splash">
  <html>Splash content here</html>
  <redirectUrl>https://www.cohga.com/</redirectUrl>
  <showOnce>true</showOnce>
</plugin>
```

Configuration for the splash panel is provided by a plugin, and the various configuration options for the panel are configured within the plugin. Some configuration examples are shown below.

Show a basic splash page with the content from a static web page (served by Weave from the content at . . .
 \weave\platform\workspace\static\welcome.html)

```
<plugin id="weave.splash">
  <url>static/welcome.html</url>
</plugin>
```

Show a basic splash page with the content from a static web page (served by Weave from the content at . . .
 \weave\platform\workspace\static\information.html) that the user only sees once.

Additionally, it has a `splashId` set which can be changed if the content of the `information.html` file changes and you want to force the panel to display again if it's been hidden. Note that the `splashId` defaults to "startup", so if you didn't set a `splashId` at the start but now want to force the page to display again you can just add a new `splashId` (that isn't "startup").

```
<plugin id="weave.splash">
  <url>static/information.html</url>
  <splashId>splash20211119</splashId>
  <title>Important News</title>
  <iconCls>icon-information</iconCls>
  <showOnce>true</showOnce>
</plugin>
```

Show a basic licence page with the content from a static web page (served by something else) and a custom window size.

```
<plugin id="weave.splash">
  <height>400</height>
  <width>600</width>
  <url>https://cohga.com/weave/licence.html</url>
  <redirectUrl>https://cohga.com/weave/</redirectUrl>
</plugin>
```

Show a basic splash page with embedded HTML that allows the user to stop it displaying at the start of each session.

```
<plugin id="weave.splash">
  <height>100</height>
  <html><![CDATA[
    <div style="padding: 15px; font-size: 15pt;
background: #dee9f7; height: 100%; text-align: center">
      Welcome to the wonderful world of Weave
    </div>
  ]]></html>
  <allowHide>true</allowHide>
</plugin>
```

Show a splash page with multiple tabs.

```
<plugin id="weave.splash">
  <tabs>
    <tab url="static/welcome.html" title="Welcome"/>
    <tab url="static/help.html" title="Help"/>
    <tab url="static/contact.html" title="Contact"/>
  </tabs>
</plugin>
```

Finally, the show splash button can be added to a toolbar and would allow the user to display the page at any time.

```
<item action="weave.showsplash" />
```

If you have a different `splashId` specified in the plugin configuration you have to also provide that.

```
<item action="weave.showsplash" splashId="splash20211119" />
```

If you want to specify a different tab on a multi-tab splash panel you can also do that, for example to show the Help tab in the above configuration when the button is pressed you can use the following (the tab number starts from 0).

```
<item action="weave.showsplash" tab="1" />
```

Further Reading

- [Toolbars](#)
- [Actions](#)
- [Views](#)
- [Layout](#)
- [Context Menus](#)
- [Samples](#)
- [Snippets](#)

Client Layout

Layouts are used within a client configuration to describe regions on the page that are available for embedding content, with the actual content being provided by [Views](#). Each perspective in a client configuration has its own layout, allowing for multiple page layouts to be available within a single client configuration.

Each region within the layout can have a different `type`, where the type determines how the content added to the region is displayed, `tab` is the default if not set and the available types are:

Type	Description
tab	Multiple views can be added to the region with tabs being added to the region to allow the user to switch between the contained content
accordion	Multiple views can be added to the region with collapsible labels added to the region to allow the user to switch between the contained content
fit	One view can be added to the regions and the view will be sized to take up the entire available area
border	No views can be added to the region, instead the region is given its own layout and views are added to that layout

If no layout is configured in the client then a single `center` region will be created. The type of this default center region will be either `tab` or a `fit` depending upon the number of views that are added to it, if there's only one view then it'll be `fit`, but if there's more than one view then it'll be `tab`.

Once the layout has been configured the [Views](#) can be added to the client configuration and the `location` set in the view determines which region in the layout the view will be added to. If no `location` is set for a view then it will be added to the `center` region.

Nested Layouts

I'll try and give a quick overview on how to create a nested layout structure in the client.

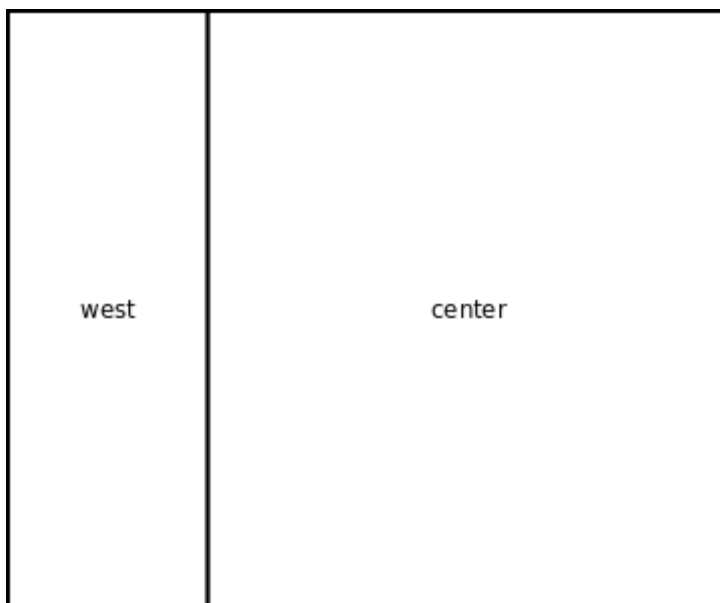
By default each client perspective is broken into five regions, called `north`, `south`, `east`, `west` and `center`. These regions can be further sub-divided by changing their `type` to `border`, then you can apply a `layout` to the divided region.

One thing to remember is that every layout always must have a `center` region and this `center` region will fill up the space remaining from the other regions in the layout.

So the first thing is you need to do is set the `type` to `border` for the region you want to split, then the split has the same options as the main layout tag (i.e. it now has a `center`, `north`, `south`, `west` and `east` regions)

So if we have the following layout

```
<layout>
  <center type="tab" />
  <west width="300" type="tab" />
</layout>
```



which looks like

Note that we didn't provide any width or height information for the `center` region above but we did for the `west` region, this is because the size of the `center` region is always determined automatically based on the size of the `east`, `west`, `north` and `south` regions it shares its layout with.

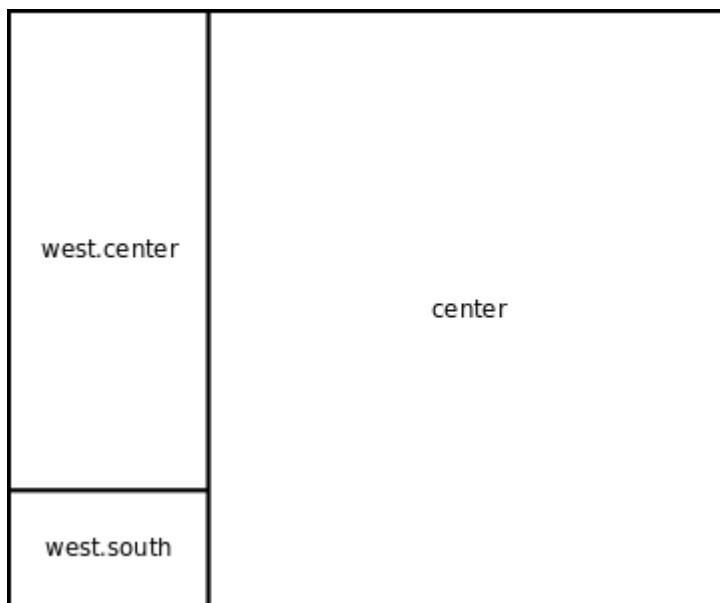
So you should always set a width for `east` and `west`, and a height for `north` and `south` regions.

The next thing we do is to change the `west` type to `border` (which is temporarily invalid, because a `border` type always needs the layout content)

```
<layout>
  <center type="tab" />
  <west width="300" type="border" />
</layout>
```

So the next thing we have to do is set the layout content for the west region

```
<layout>
  <center type="tab" />
  <west width="300" type="border">
    <layout>
      <center type="tab" />
      <south type="fit" height="200" />
    </layout>
  </west>
</layout>
```



So basically what used to be `west` is now `west.center` and we have a new `west.south` region (we could also add a `north`, `east` and/or `west`)

then when it's time to refer to the `location` in the view you now have

- `center`
- `west.center`
- `west.south`

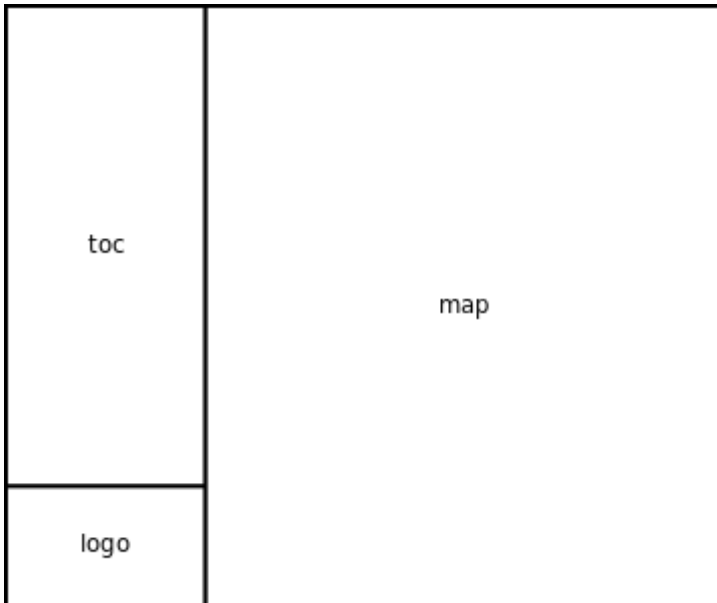
as the available locations (as opposed to `center` and `west` as in the original example)

From here we could name the regions so later on we can refer to them using easy to remember names (plus we can change our layout without having to make sure we change the locations for any views that we've added to the region)

```

<layout>
  <center name="map" type="tab"/>
  <west width="300" type="border">
    <layout>
      <center name="toc" type="tab"/>
      <south name="logo" type="fit" height="200"/>
    </layout>
  </west>
</layout>

```



This can be applied recursively and to each of the five regions available in a layout

```

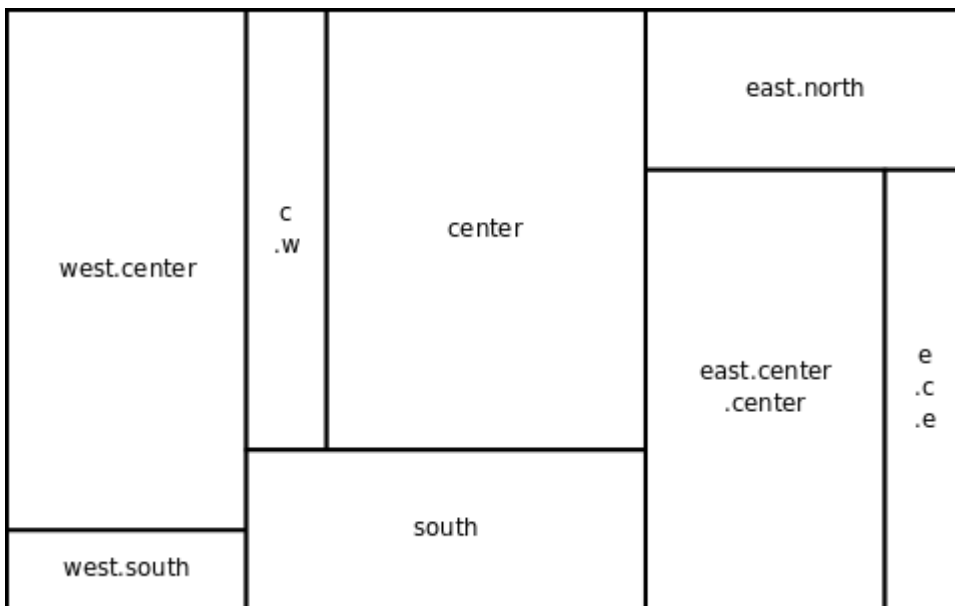
<layout>
  <center type="border">
    <layout>
      <center type="tab"/>
      <west type="fit" width="100"/>
    </layout>
  </center>
  <west width="300" type="border">
    <layout>
      <center type="tab"/>
      <south type="fit" height="100"/>
    </layout>
  </west>
  <south type="tab" height="200"/>
  <east width="400" type="border">
    <layout>
      <center type="border">
        <layout>
          <east type="fit" width="100"/>
          <center type="tab"/>
        </layout>
      </center>
    </layout>
  </east>
</layout>

```

```

        <north type="fit" height="200"/>
    </layout>
</east>
</layout>

```



Also, you should set a default height for north and south regions and a default width for east and west.

Further for north, south, east and west a minSize and maxSize should be provided if you set split to true. If split is false or not set then the region can't be resized.

You can also set collapsible to true to allow for easy collapsing of the region

```

<layout>
    <center name="map" type="tab"/>
    <west width="300" type="border" split="true" minSize="100"
maxSize="500" collapsible="true">
        <layout>
            <center name="toc" type="tab"/>
            <south name="logo" type="fit" height="200"/>
        </layout>
    </west>
</layout>

```

And then finally you can add a toolbar and statusbar to each layout

```

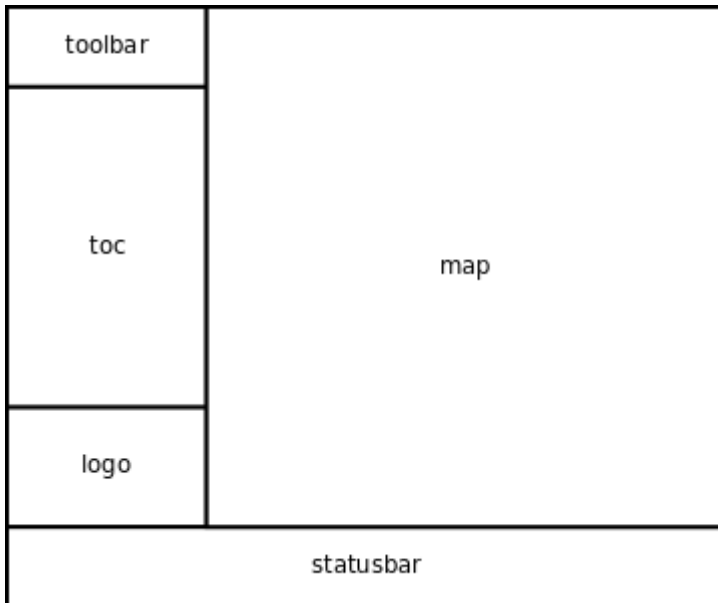
<layout>
    <center name="map" type="tab"/>
    <west width="300" type="border" collapsible="true">
        <layout>
            <toolbar>
                <item action="..." />
            </toolbar>
            <center name="toc" type="tab"/>
            <south name="logo" type="fit" height="200"/>
        </layout>
    </west>

```

```

        <statusbar>
            <item component="..." />
        </statusbar>
    </layout>

```



Finally, keep in mind that the `toolbar` and `statusbar` added to the layout is in addition to the global `toolbar` and `statusbar` that are outside of the layout (not that you have to define any of them if you don't want to).

Client Views

Views are what that the user sees with the Weave client. They're setup in the client config section of the config file.

Available Views

- [Geocode](#) - Search for a property using a geocoding engine
- [Data](#) - Display attribute information about the currently selected entities
- [Identify](#) - Display information about interactively selected entities
- [Projections](#) - Display mouse location in different coordinate systems
- [Report](#) - Choose a report to generate
- [Search](#) - Search for an entity
- [Map](#) - Display a map image
- [Table of Contents](#) - Determine what's displayed on the map
- [HTML](#) - Embed HTML directly in page
- [Details](#) - Custom display of entity details
- [Image](#) - Embed an image directly in page
- [Legend](#) - Embed a pre-rendered legend image
- [Quick Search](#) - Quickly find and display an entity
- [Entity Selector](#) - Choose active entity

Geocode	com.cohga.html.client.geocodeView
Data	com.cohga.html.client.main.gridView
Identify	com.cohga.html.client.main.identifyView
Projection	com.cohga.html.client.main.projView
Report	com.cohga.html.client.main.reportView
Search	com.cohga.html.client.main.searchView
Map	com.cohga.html.client.map.mapView
Table of Contents	com.cohga.html.client.map.tocView
HTML	com.cohga.html.client.view
Details	com.cohga.client.panel.details

Image	com.cohga.client.panel.image
Legend	com.cohga.client.panel.legend
Quick Search	com.cohga.view.quicksearch
Entity Selector	com.cohga.html.client.main.entitySelectorView

Client Views Data

Provides a means of displaying attribute information relating to the currently selected entities.

ID

com.cohga.html.client.main.gridView

Sub-tags

name	type	cardinality	default	description
label	String	0..1		Label to display in tab
location	String	1..1		Which region to add the view to
toolbar	grid actions	0..n		A collection of tools to add to the panel
format	format definitions	0..1		A collection of formatting options
autoShow OnUpdate	boolean	0..1	false	Should the grid panel tab be set active to active when the selection or active entity changes? Requires at least version 3.92.x of com.cohga.client.weave.main, which was first included in Weave 2.5.29. Note this defaults to <code>true</code> for the identify tool grid panel when it's configured to not use a popup window.

See [Common grid panel configuration options](#) for other grid related options

Positioning Highlight Markers

When you click on a row in the data grid the map will highlight the geometry related to the selected row. Normally the highlighting geometry is determined by the geometry associated with the entity that the row corresponds to, but it's also possible to include a coordinate within the data definition that the data grid is displaying and have a marker placed at that coordinate instead.

For example, a land parcel entity would generally be represented as a polygon boundary. But if you have water meter records associated with a parcel you could include a coordinate in the water meter data definition so that when a user clicks on a row in the data grid a marker is placed at the location of the water meter on the property rather than having the property boundary highlighted.

To replace the default highlight geometry with a marker in this way you should include a numeric `x` and `y` (lower case) parameters in your data definition, which contains the coordinate for the point. You can also include a `crs` parameter that provides the EPSG code for the projection that the coordinate is stored in; the coordinates will be assumed to be in the same coordinate system as the underlying map if this is not included. Note that you can also use `latitude` and `longitude` or `lat` and `lon` as the field names, and in his case, if the `crs` parameter isn't included it's assumed to be `WGS84`. It's suggested that a `crs` parameter always be included to explicitly provide the CRS and so allowing the same data definition to be used regardless of what CRS is used in the underlying map.

Data definition with coordinate

```
<data:datadefinition id="water_meter">
  <datasourcedataconnection datasource="db1" table="
WATER_METER" key="COMPKEY">
    <parameter column="COMPKEY" />
    <parameter column="CATEGORY" />
    <parameter column="TYPE" />
    <parameter column="OTHER" />
    <parameter name="x" label="X" column="LOCATION_X" type="
float" />
    <parameter name="y" label="Y" column="LOCATION_Y" type="
```



```
float" />
        <parameter name="crs" label="CRS" column="'EPSG:28355'" />
    </datasourcedataconnection>
</data:datadefinition>
<data:data id="water_meter" datadefinition="water_meter" entity="
water_meter" label="Water Meter"/>
```

On a final note, it's possible to hide the additional parameters by setting `hidden` to `true` when formatting the columns, as described in the examples below.

There was a bug, fixed in 2.5.22.1, that caused the marker not to be displayed if the `crs` parameter was not included and you were using `x` and `y` (as opposed to `latitude` and `longitude` or `lat` and `lon`).

Examples

Basic grid view

```
<view id='com.cohga.html.client.main.gridView'>
    <label>Identify</label>
    <location>south</location>
</view>
```

Formatting a grid view

```
<view id="com.cohga.html.client.main.gridView">
    <label>Data</label>
    <location>center.south</location>
    <toolbar>
        <!-- toolbar items go here -->
    </toolbar>

    <!-- override column widths -->
    <format>
        <!-- set column widths for columns in a specific data
-->
        <data id="ar_property_details">
            <column id="PropNo" width="20"/>
            <column id="OwnerName" width="200"/>
        </data>
        <data id="ar_road_details">
            <column id="Status" width="10"/>
            <column id="Old_Status" hidden="true"/>
        </data>
        <!-- set column width globally for columns in any
data -->
        <column id="RoadName" width="200"/>
        <column id="RoadType" width="50"/>
        <column id="Suburb" width="250"/>
        <column id="oid" hidden="true"/>
```

```

    </format>
</view>

```

Formatting with snippet (for re-use of formatting)

```

<client:format id="custom">
  <!-- set column widths for columns in a specific data -->
  <data id="ar_property_details">
    <column id="PropNo" width="20"/>
    <column id="OwnerName" width="200"/>
  </data>
  <data id="ar_road_details">
    <column id="Status" width="10"/>
  </data>
  <!-- set column width globally for columns in any data -->
  <column id="RoadName" width="200"/>
  <column id="RoadType" width="50"/>
  <column id="Suburb" width="250"/>
</client:format>

<client:config id="test">
  <!-- there would be other config items here -->
  <perspective>
    <!-- there would be other config items here -->
    <view id="com.cohga.html.client.main.gridView">
      <label>Data</label>
      <location>center.south</location>
      <toolbar>
        <!-- toolbar items go here -->
      </toolbar>

      <!-- override formatting -->
      <format ref="custom"/>
    </view>
  </perspective>
</client:config>

```

Setting formatting in defaults section

Alternate default formatting of grid and align option as of version 2.4.14

```

<client:config id="test">
  <!-- there would be other config items here -->
  <perspective>
    <!-- there would be other config items here -->
    <view id="com.cohga.html.client.main.gridView">
      <label>Data</label>
      <location>center.south</location>
      <toolbar>
        <!-- toolbar items go here -->
      </toolbar>
    </view>
  </perspective>
</client:config>

```

```

        </toolbar>
    </view>
</perspective>
<defaults>
    <data>
        <!-- set column widths for columns in a
specific data -->
        <data id="ar_property_details">
            <column id="PropNo" width="20" align="
right"/>
            <column id="OwnerName" width="200"/>
        </data>
        <data id="ar_road_details">
            <column id="Status" width="10"/>
        </data>
        <!-- set column width globally for columns in
any data -->
        <column id="RoadName" width="200"/>
        <column id="RoadType" width="50"/>
        <column id="Suburb" width="250"/>
    </data>
</defaults>
</client:config>

```

Localising the grid, old format

```

<client:config id="example">
    <!-- there should be other config items here -->

    <view id="com.cohga.html.client.main.gridView">
        <label>Data</label>
        <location>center.south</location>
        <emptyMsg>%nodata.text</emptyMsg>
    </view>

    <!-- there should be other config items here -->
</client:config>

<client:resources>
    <resource id="nodata.text">No data to display</resource>
</client:resources>

<client:resources lang="ru">
    <resource id="nodata.text">    </resource>
</client:resources>

<client:resources lang="sv">
    <resource id="nodata.text">Inga data för att visa</resource>
</client:resources>

<client:resources lang="it">
    <resource id="nodata.text">Nessun dato da visualizzare<

```

```

/resource>
</client:resources>

```

Localising the grid, new format

```

<client:config id="example">
  <!-- there should be other config items here -->

  <view id="com.cohga.html.client.main.gridView">
    <label>Data</label>
    <location>center.south</location>
    <!-- note, no text properties specified here -->
  </view>

  <!-- there should be other config items here -->
</client:config>

<client:resources>
  <resource id="grid.data.emptyMsg">No data to display<
/resource>
</client:resources>

<client:resources lang="ru">
  <resource id="grid.data.emptyMsg">  </resource>
</client:resources>

<client:resources lang="sv">
  <resource id="grid.data.emptyMsg">Inga data för att visa<
/resource>
</client:resources>

<client:resources lang="it">
  <resource id="grid.data.emptyMsg">Nessun dato da visualizzare<
/resource>
</client:resources>

```

118n resources

id	default
grid.data.emptyMsg	No data to display
grid.page.displayMsg	Displaying {0} - {1} of {2}
grid.page.beforePage	Page
grid.page.afterPage	of {0}
grid.page.first	First Page
grid.page.prev	Previous Page
grid.page.next	Next Page
grid.page.last	Last Page
grid.page.refresh	Refresh

grid.page.items

Items per page

Date and Time formatting

You can also change the default date, time and date/time format by setting adding `dateformat`, `timeformat` and/or `datetimeformat` entries to the `<data>` section of the `<defaults>` section in the client config.

```
<client:config id="test">
  ...
  <defaults>
    ...
    <data>
      <dateformat>Y-m-d</dateformat>
      <timeformat>H:i:s</timeformat>
      <datetimeformat>Y-m-d H:i:s</datetimeformat>
    </data>
    ...
  </defaults>
</client:config>
```

Place-holders

name	value	format
d	Day of Month	Numeric, 2 digits, left padded
D	Day of Week	Text, long
j	Day of Month	Numeric
l	Day of Week	Text, short
N	Day of Week	Numeric, 1=Monday, 7=Sunday
S	Day of Month Suffix	Text, 'st', 'nd', 'rd', 'th'
w	Day of Week	Numeric, 0=Sunday, 6=Saturday
z	Day of Year	Numeric
W	Week of Year	Numeric, 2 digits, left padded
F	Month	Text, long
m	Month	Numeric, 2 digits, left padded, January=01
M	Month	Text, short
n	Month	Numeric, January=1
t	Days in Month	Numeric
Y	Year	Numeric, 4 digits
y	Year	Numeric, 2 digits
a	am/pm	Text
A	AM/PM	Text
g	Hour	12 Hour, Numeric
G	Hour	24 Hour, Numeric
h	Hour	12 Hour, 2 digits, left padded
H	Hour	24 Hour, 2 digits, left padded
i	Minute	Numeric, 2 digits, left padded

s	Second	Numeric, 2 digits, left padded
u	Milli-Second	Numeric, 3 digits, left padded
T	Timezone	String

Displaying record count

```
<client:config id="example">
  <!-- there should be other config items here -->

  <view id="com.cohga.html.client.main.gridView">
    <label>Data</label>
    <location>center.south</location>
    <displayMsg>Displaying {0} - {1} of {2}</displayMsg>
  </view>

  <!-- there should be other config items here -->
</client:config>
```

26 KENMAN CLOSE	TEMPLESTOWE
3 KENMAN CLOSE 3106	
11 LEANE DRIVE	ELTHAM
PO BOX 1	DONCASTER VIC 3108

Displaying 1 - 10 of 87

Detailed data formatting instructions

As at Weave 2.5.18 you can specify client formatting options for a data definition directly in the `<data:data>` tag for each data definition. This is outlined at the end of this page.

The following notes outline how to format columns in a data grid.

Firstly let's assume that you have the following, or something like it in your defaults section in your client config:

Initial client config defaults section

```
<defaults>
  <entity id="lyr_suburbs />
  <entity id="lyr_properties" isDefault="true">
    <search>aq_property_by_address_auth</search>
    <data>ar_property_address_auth</data>
  </entity>
  <entity id="lyr_parcel">
</defaults>
```

This is just setting some options for some entities, and there may even be other material in there that relates to reports for example.

Elsewhere in your config files you would have a data definition and data configuration that looks like this:

Data configuration ready to have it formatting changed

```
<data:datadefinition id="__dd__ar_properties_sde">
  <datasourcedataconnection datasource="ds_sde" table="
PROPERTIES_ACTIVE_REGION" key="PROPERTY_NUMBER" prefix="DISTINCT">
    <parameter type="string" name="sp_propno" label="Parcel Number"
column="PROPERTY_NUMBER" />
    <parameter type="string" name="sp_houseno" label="House Number"
column="HOUSE_NUMBER" />
    <parameter type="string" name="sp_roadname" label="Road Name"
column="STREET_NAME" />
    <parameter type="string" name="sp_pubprop" label="Public/State
Property" column="PUBLIC_PROPERTIES" />
    <parameter type="list" name="lp_landclass" label="Council Land
Classification" column="PUBLIC_LAND_IDENTIFIER" list="vl_props_active"
/>
    <parameter type="string" name="sp_propdesc" label="Property
Description" column="PROPERTY_NAME" />
  </datasourcedataconnection>
</data:datadefinition>
<data:data id="ar_properties_sde" label="GIS - Property Details"
entity="lyr_properties" datadefinition="__dd__ar_properties_sde" />
```

Now let us say that you want to set the width for the column that displays the `sp_houseno` parameter to 20 pixels and align it to the right, and you want to alter the width for the column that displays the `sp_propdesc` parameter to 200 pixels, just for when you display the data for the `ar_properties_sde` data, i.e. when the user selects the "GIS - Property Details" value in the grid panel combo box.

To do this you first need to add a `<data>` section to the defaults (Add it to the end of the defaults section quoted above), e.g .

Defaults section with empty data configuration section

```
<defaults>
  <entity id="lyr_suburbs" />
  <entity id="lyr_properties" isDefault="true">
    <search>aq_property_by_address_auth</search>
    <data>ar_property_address_auth</data>
  </entity>
  <entity id="lyr_parcel" />

  <data>
    <!-- This is the new data section that I was referring to above,
at the moment it doesn't do anything, I'm just showing where it goes,
note that I've added it just inside of the defaults tag, and not
within an entity tag -->
  </data>
</defaults>
```

Once you have the new `<data>` section inside of the `<defaults>` section, which is inside of your `client:config`, you can add a sub-data tag to configure an individual grid. In this example we are going to be setting the column definitions outlined above relating to the `sp_houseno` and `sp_propdesc` columns in the `ar_properties_sde` grid.

To do this we need to add yet another `<data>` tag inside of the one we just added, and give it an id that matched the grid we want to edit. In our example this will be `ar_properties_sde`. We can do this multiple times to change the display of multiple grids, but for now we will just do it for one grid, like this:

Defaults section with empty data configuration section for a single grid

```
<defaults>
  <entity id="lyr_suburbs />
  <entity id="lyr_properties" isDefault="true">
    <search>aq_property_by_address_auth</search>
    <data>ar_property_address_auth</data>
  </entity>
  <entity id="lyr_parcel" />

  <data>
    <data id="ar_properties_sde">
      <!-- This is the new sub-data section that I was just referring
to, it still doesn't do anything, I'm just showing where it goes,
what we add inside of this will alter the ar_properties_sde data
display -->
      </data>
    </data>

  </defaults>
```

To change the individual column formatting options for each parameter we want to change, we have to add a `<column>` tag to the `<data>` tag we just added, (the one with the id attribute, not the one without the id attribute). In our example we are only changing 2 columns, `sp_houseno` and `sp_propdesc`. The rest of the parameters defined in `_dd_ar_properties_sde` when displayed using the `ar_properties_sde` grid will use the default values.

Defaults section with a data configuration section to change to columns

```
<defaults>
  <entity id="lyr_suburbs />
  <entity id="lyr_properties" isDefault="true">
    <search>aq_property_by_address_auth</search>
    <data>ar_property_address_auth</data>
  </entity>
  <entity id="lyr_parcel" />

  <data>
    <data id="ar_properties_sde">
      <column id="sp_houseno" width="20" align="right"/>
      <column id="sp_propdesc" width="200"/>
    </data>
  </data>

  </defaults>
```

If you wanted to change other parameters in that grid you could add additional `<column>` tags to the `<data>` tag (the one with the id attribute, not the one without the id attribute).

If you wanted to change the parameter display for a column in a data definition other than the `ar_properties_sde` grid you would create a new `<data>` tag with the id of the data you want to change. For example, if we have another data definition and a corresponding `<data>` (this is the data tag related to the data definition, not the one we are adding here) tag with the id `ar_conquest_council_building_details`, you could add a `<data id="ar_conquest_council_building_details">...</data>` tag to the top data tag we just added (the one without the id), like so:

Defaults section with multiple data configuration section to alter multiple grid

```
<defaults>
  <entity id="lyr_suburbs />
  <entity id="lyr_properties" isDefault="true">
    <search>aq_property_by_address_auth</search>
    <data>ar_property_address_auth</data>
  </entity>
  <entity id="lyr_parcel" />

  <data>
    <data id="ar_properties_sde">
      <column id="sp_houseno" width="20" align="right"/>
      <column id="sp_propdesc" width="200"/>
    </data>
    <data id="ar_conquest_council_building_details">
      <!-- you would add some column tags here like the one above, I
haven't done this so this won't actually do anything, but you get
idea -->
    </data>
  </data>

</defaults>
```

If you have the same parameter used in multiple data definitions and you want to change all of them at once, then add a `<column>` tag at the same level as the `<data>` tags (the ones with the id's). For example, if you wanted to change all of the parameters named `strnum` and `strname` you would do the following:

Defaults section with multiple data configuration section to alter multiple grid and globally altering some columns

```
<defaults>
  <entity id="lyr_suburbs />
  <entity id="lyr_properties" isDefault="true">
    <search>aq_property_by_address_auth</search>
    <data>ar_property_address_auth</data>
  </entity>
  <entity id="lyr_parcel" />

  <data>
    <data id="ar_properties_sde">
      <column id="sp_houseno" width="20" align="right"/>
      <column id="sp_propdesc" width="200"/>
    </data>
    <data id="ar_conquest_council_building_details">
      <!-- you would add some column tags here like the one above, I
```

```

haven't done this so this won't actually do anything, but you get
idea -->
    </data>
    <!-- this is where I'm changing the grid display for some
parameter globally, every data definition that's displayed to the
user with a parameter with these names will be formatted according to
what we do here -->
    <column id="strnum" width="50" align="right"/>
    <column id="strname" width="100"/>
</data>

</defaults>

```

Once that is done, you should see the change in the column display for those columns we have changed.

Formatting in 2.5.18+

As of Weave 2.5.18 you can specify the same sort of formatting information described above directly within the `<data:data>` tag that's associated with a data definition, e.g.

Formatting a data definition

```

<data:datadefinition id="properties_details">
  <datasourcedataconnection datasource="ds_sde" table="
PROPERTIES_ACTIVE_REGION" key="PROPERTY_NUMBER" prefix="DISTINCT">
    <parameter type="string" name="sp_propno" label="Parcel Number"
column="PROPERTY_NUMBER" />
    <parameter type="string" name="sp_houseno" label="House Number"
column="HOUSE_NUMBER" />
    <parameter type="string" name="sp_roadname" label="Road Name"
column="STREET_NAME" />
    <parameter type="string" name="sp_pubprop" label="Public/State
Property" column="PUBLIC_PROPERTIES" />
    <parameter type="list" name="lp_landclass" label="Council Land
Classification" column="PUBLIC_LAND_IDENTIFIER" list="vl_props_active"
/>
    <parameter type="string" name="sp_propdesc" label="Property
Description" column="PROPERTY_NAME" />
  </datasourcedataconnection>
</data:datadefinition>

<data:data id="ar_property_details" label="Property Details" entity="
lyr_properties" datadefinition="properties_details">
  <columns>
    <column id="sp_propno" align="right" width="40"/>
    <column id="sp_houseno" align="right" width="40"/>
    <column id="sp_roadname" width="100"/>
  </columns>
</data:data>

```

Common Grid Panel Configuration Options

This page describes configuration options available for all client grid panels.

These include:

- [Client Views Data](#)
- [Client Views Identify](#)
- [Client Actions Identify](#)
- [Client Actions Spatial Identify](#)
- [Client Actions Selection Identify](#)

Name	Type	Cardinality	Default	i18n Resource	Description
toolbar	grid actions	0..n			A collection of tools to add to the panel
emptyMsg	string	0..1	"No data to display"	grid.data.emptyMsg	Text to display if there are no results
displayMsg	string	0..1	"Displaying {0} - {1} of {2}"	grid.page.displayMsg	Paging toolbar text showing which rows are being displayed and how many rows are available in total. If set then this text will be displayed at the end of the grid status bar. Requires at least version 2.16.x of <code>com.cohga.client.weave.main</code>
beforePageText	string	0..1	"Page"	grid.page.beforePage	Label before the page number
afterPageText	string	0..1	"of {0}"	grid.page.afterPage	Text after the page number
firstText	string	0..1	"First Page"	grid.page.first	Tooltip for the first page button
prevText	string	0..1	"Previous Page"	grid.page.prev	Tooltip for previous page button
nextText	string	0..1	"Next Page"	grid.page.next	Tooltip for next page button
lastText	string	0..1	"Last Page"	grid.page.last	Tooltip for last page button
refreshText	string	0..1	"Refresh"	grid.page.refresh	Tooltip for the refresh button
itemsText	string	0..1	"Items per page"	grid.page.items	Text for items per page
pageSize	integer	0..1	20		The number of results to display per page
showPageSize	boolean	0..1	true		Should the page size selector be shown? Requires at least version 2.16.x of <code>com.cohga.client.weave.main</code>
textSelect	boolean	0..1	false		Can the user select the text?

Client Views Details

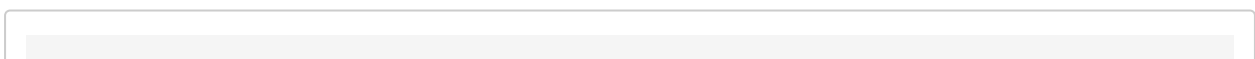
The Details View can display information related to a selected entity, like the [Data View](#), but it can format the data in a more pleasing manner by specifying a template to be used to display the data, this way advanced HTML formatting can be used to pretty up the display of the information.

The details view will only display one record at a time, but provided a paging toolbar for navigating between records if there is more than one available.

Id

`com.cohga.client.panel.details`

The following is a basic example of the details panel



Details panel generating default templates

```
<view id="com.cohga.client.panel.details">
  <location>west</location>
  <label>Details</label>
  <detail>
    <data>ar_road_details</data>
  </detail>
  <detail>
    <data>ar_property_details</data>
  </detail>
</view>
```

The above example will add the details view and setup the display of details for roads and properties.

Any entity that you want to be displayed in this panel must have a `details` entry in the details panel configuration, and only one `detail` entry per entity is supported.

The details panel understands a `data` tag, which already includes the `entity`, `label` and `datadefinition` (a template is defined automatically, unless it's also provided), alternatively these values can be specified directly.

The previous example did not include a template, so the Weave client will generate a simple default template that will list each attribute from the underlying data, using the label to annotate each entry. If you want more control over the output then you can set a template that contains HTML marker with place holders that mark where the associated attribute should be placed.

Details panel with a template

```
<view id="com.cohga.client.panel.details">
  <location>west</location>
  <label>Details</label>
  <loadingText>Loading...</loadingText>
  <detail>
    <data>ar_owner_details</data>
    <template><![CDATA[
The property located at<br/>
<b>{address}</b><br/>
is currently owned by<br/>
{owner}<br/>
of<br/>
{owner_address}
<hr>
<i>Links</i>
{owner_link}<br/>
{occupier_link}<br/>
{property_link}<br/>
]]></template>
  </detail>
</view>
```

If you use a template then there is no need to specify a `label` if you use the `entity/datadefinition` format (as opposed to the `data` format) to configure the details, since the label is just used in the default template that the Weave client will generate if

no template is set.

If you still want a label at the top if the details then you'll need to put it there yourself if you use a template.

You can set the text to be displayed when data is being loaded, when there are no records to display and if an error occurs when retrieving the data by setting the `loadingText`, `emptyText` and `errorText` properties in the details panel. Also, the tooltip content can be altered by setting a tooltip element.

Details panel with modified text

```
<view id="com.cohga.client.panel.details">
  <location>west</location>
  <label>Details</label>
  <emptyText>No results</emptyText>
  <loadingText>Please wait</loadingText>
  <errorText>Sorry, the server was unable to load data<
/errorText>
  <tooltip>
    <title>More Details</title>
    <text>Display details about the selected
object</text>
  </tooltip>
  <detail>
    <data>ar_road_details</data>
  </detail>
  <detail>
    <data>ar_property_details</data>
  </detail>
</view>
```

If you use complex templates then it may be worth using snippets to move the templates out of the configuration of the details panel itself (and possibly even store them in a separate configuration file)

Details panel with a template

```
<client:template id="owner_details">
<![CDATA[
The property located at<br/>
<b>{address}</b><br/>
is currently owned by<br/>
{owner}<br/>
of<br/>
{owner_address}
<hr>
<i>Links</i>
{owner_link}<br/>
{occupier_link}<br/>
{property_link}<br/>
]]>
</client:template>

<client:config id="example">
```

```

<!-- more client configuration goes here -->
<view id="com.cohga.client.panel.details">
  <location>west</location>
  <label>Details</label>
  <loadingText>Loading...</loadingText>
  <detail>
    <entity>property</entity>
    <datadefinition>dd_owner_details<
/datadefinition>
    <template ref="owner_details"/>
  </detail>
</view>
<!-- more client configuration goes here -->
</client:config>

```

Client Views Entity Selector

The Entity Selector panel provides a view that shows all of the current entities and allows the user to select the active entity. It support grouping of the entities and tools to clear the selection for the entity and zoom to the selection.

This view can be referenced using the id 'com.cohga.html.client.main.entitySelectorView', for historical reasons, or 'weave.entitySelectorView'.

A usable implementation of this view was only made available at Weave 2.4.11.

Basic panel definition

```

<view id="weave.entitySelectorView">
  <location>east</location>
</view>

```

Entities	
Entity	Size
Assets	21
Bus Stops	4
Council Buildings	12
Council Carparks	0
Development Applications	0
Drainage	0
Edit (ArcSDE)	0
Edit (Checkbox)	0
Edit (Oracle)	0
Edit (Shapefile)	0
Edit (Sweden)	0
Graffiti	0
Hydrants	0
Owners	0
Property	1468
Roads	37
Roads (Edit)	0
Suburbs	0

Panel with right click context menu

```
<view id="weave.entitySelectorView">
  <location>east</location>
  <contextmenu>
    <item action="weave.toc.zoomSelection" text="Zoom to
Selection"/>
    <item action="weave.toc.clearSelection" text="Clear
Selection"/>
  </contextmenu>
</view>
```

Entities	
Entity	Size
Assets	21
Bus Stops	4
Council Buildings	12
Council Carparks	0
Development Applications	0
Drainage	0
Edit (ArcSDE)	0
Edit (Checkbox)	0
Edit (Oracle)	0
Edit (Shapefile)	0
Edit (Sweden)	0
Graffiti	0
Hydrants	0
Owners	0
Property	0
Roads	0
Roads (Edit)	0
Suburbs	0

Zoom to Selection
Clear Selection

Panel with right click context menu and toolbar

```

<view id="weave.entitySelectorView">
  <location>east</location>
  <toolbar>
    <item action="com.cohga.html.client.map.select.
clearAction"/>
    <item action="com.cohga.html.client.map.select.
clearAllAction"/>
    <item action="com.cohga.html.client.map.select.
zoomAction"/>
  </toolbar>
  <contextmenu>
    <item action="weave.toc.zoomSelection" text="Zoom to
Selection"/>
    <item action="weave.toc.clearSelection" text="Clear
Selection"/>
  </contextmenu>
</view>

```


Entity	Size
Assets	21
Bus Stops	8
Council Buildings	44
Council Carparks	0
Development Applications	0
Drainage	0

Panel with refined set of entities

```
<view id="weave.entitySelectorView">
  <location>east</location>
  <entity>property</entity>
  <entity>owners</entity>
  <entity>roads</entity>
</view>
```

Entity	Size
Property	76
Owners	0
Roads	217

The ordering of the entities in the list is also determined by the entity list.

Panel with refined set of entities, using a different formatting, and custom icons



















```
<view id="weave.entitySelectorView">
  <location>east</location>
  <entities>
    <entity id="property" iconCls="icon-house"/>
    <entity id="owners" iconCls="icon-group"/>
    <entity id="roads" iconCls="icon-lorry"/>
  </entities>
</view>
```

Entities	
Entity	Size
 Property	76
 Owners	0
 Roads	217

Both forms of entity lists are supported, that is <entity> tags directly listed or embedded within an <entities> tag, this allows for the group of entities to be referenced as a whole using a snippet and included using <entities id="entitylist"/>. Additionally both entity formats are supported, that is <entity>id</entity> and <entity attribute="value"/>, the first format is the equivalent of <entity id="id"/>, but the second format allows you to specify additional attributes.

Defaults section modified to alter appearance of entities

```
<defaults>
  <entities>
    <entity id="property" iconCls="icon-house"/>
    <entity id="owners" iconCls="icon-group"/>
    <entity id="roads" iconCls="icon-lorry"/>
  </entities>
</defaults>
```

Entities	
Entity	Size
 Assets	21
 Bus Stops	8
 Council Buildings	44
 Council Carparks	0
 Development Applications	0
 Drainage	0
 Edit (ArcSDE)	0
 Edit (Checkbox)	0
 Edit (Oracle)	0
 Edit (Shapefile)	0
 Edit (Sweden)	0
 Graffiti	0
 Hydrants	0
 Owners	0
 Property	76
 Roads	217
 Roads (Edit)	0
 Suburbs	0

Icons listed in the defaults section will be used by other controls also, including the entity selector combo box and the spatial drill-down tool.

Defaults section entities customised and grouped

```
<defaults>
  <entities>
    <entity id="property" iconCls="icon-house" group="
Main"/>
    <entity id="owners" iconCls="icon-group" group="Main"
/>
    <entity id="roads" iconCls="icon-lorry" group="Main"/>
  </entities>
</defaults>
```

Entities	
Entity ▲	Size
Main	
Owners	0
Property	76
Roads	217
Other	
Assets	21
Bus Stops	8
Council Buildings	44
Council Carparks	0
Development Applications	0
Drainage	0
Edit (ArcSDE)	0
Edit (Checkbox)	0
Edit (Oracle)	0
Edit (Shapefile)	0
Edit (Sweden)	0
Graffiti	0
Hydrants	0
Roads (Edit)	0
Suburbs	0

The title of the group for ungrouped items "Other" by default, but can be changed by setting the 'defaultGroupName' property for the entity selector view, and this value can be set via a resource.

Changing group name for ungrouped entities

```
<view id="weave.entitySelectorView">
  <location>east</location>
  <defaultGroupName>Ungrouped</defaultGroupName>
</view>
```

Entities	
Entity ▲	Size
Main	
👤 Owners	0
🏠 Property	76
🚌 Roads	217
Ungrouped	
📁 Assets	21
📁 Bus Stops	8
📁 Council Buildings	44
📁 Council Carparks	0

Custom icons can be created as described [here](#)

Panel with all groups collapsed on start up (apart from Active Layer)

```
<view id="weave.entitySelectorView">
  <location>east</location>
  <startCollapsed>true</startCollapsed>
</view>
```

Client Views HTML

If you want to display external HTML content in a layout region then you can add a view like the following

```
<view id='com.cohga.html.client.view'>
  <location>center</location>
  <title>Title</title>
  <html><![CDATA[
    <iframe src="http://server/file.htm" width="100%"
height="100%" frameborder="0" />
  ]]></html>
</view>
```

This will display the content retrieved from <http://server/file.htm> in the center layout region.

Client Views Identify

Provides a means of displaying attribute information relating to the currently selected entities.

There is a corresponding [action](#) that must be used to drive the contents of this view.

It is not recommended to use this view in the same panel (`location`) as the [Client Views Data](#) as the results could be misleading to users if they are not aware that a selection returns attributes in a Grid View, while the Identify tools return attributes in an Identify View.

ID

com.cohga.html.client.main.identifyView

Sub-tags

Name	type	cardinality	description
label	String	0..1	Label to display in tab
location	String	1..1	Which region to add the view to

Example

```
<view id='com.cohga.html.client.main.identifyView' >
  <label>Identify</label>
  <location>south</location>
</view>
```

See [Common grid panel configuration options](#) for other grid related options

Client Views Legend

Provides a means of displaying a legend, supporting both static legend images and dynamically generated legends. The dynamic legends support map engines that generate bit map images, e.g. ArcIMS, and those that generate a hierarchy, e.g. ArcGIS Server and WMS.

There is a corresponding [action](#) that can be used to display the legend in a popup window.

ID

com.cohga.client.panel.legend

Sub-tags

Na me	type	c a r d i n a l i t y	description
lab el	String	0. .1	Label to display in tab
ma pE ngi ne	string	0. .n	Map engine(s) to use to generate the dynamic legends, all available map engines will be used if none specified
re mo ve	list	0. .1	List of layer ids, comma separated, to remove from the dynamic legends. Note that this is the layer id as the associated map engine would report it, it has nothing to do with any the id used in a toc model entry. As of 2.5.11 you can use the format "layerid/mapengine", rather than just "layerid" if the layer id is not unique across all map engines, also you can use "*/mapengine" to remove all layers from a map engine.
sh ow Mo de	'visibl e', 'all', 'chec ked'	0. .1	What type of dynamic legend to generate, default is 'visible'. 'all' will display all layers regardless of their map visibility or toc state, 'visible' will only display layers that are currently displayed on the map, 'checked' will display only layers that are checked in the toc regardless of the map scale and visibility of the layer on the map.
sta tic Le ge nd	static Lege nd	0. .n	Static images to add to legend
ext ent On ly	boole an	0. .1	If set to true (and you're using QGIS WMS server) then the legend can generate content just related to the current view extent, default value is false.
sel ect	'on', 'off'	0. .1	If set to "off" then selection entries will not be displayed in the legend, the default is "on". This is available at 2.5.28. Prior to 2.5.28 remove can be used, e.g. <remove>*/mapengine.selection</remove>.

ion
Mo
de*StaticLegend Tag*

Name	type	cardinality	description
url	String	1..1	URL of image to display
location	'top' or 'bottom'	0..1	Should the legend image be before the dynamic legends or after, defaults to 'top'
minScale	Number	0..1	Minimum scale range to display layer at
maxScale	Number	0..1	Maximum scale range to display layer at

Example

Display a legend using all map engines

```
<view id='com.cohga.client.panel.legend'>
  <label>Legend</label>
</view>
```

Display a legend using only a single map engine

```
<view id='com.cohga.client.panel.legend'>
  <label>Legend</label>
  <mapEngine>mapengine.vector</mapEngine>
</view>
```

Dynamically generate a legend for a single map engine and remove a couple of layers

```
<view id='com.cohga.client.panel.legend'>
  <label>Legend</label>
  <mapengine>mapengine.main</mapengine>
  <remove>hillshade,elevation</remove>
</view>
```

Dynamically generate a legend for a map engine and remove a couple of layers and add two static images on at the top and one at the bottom

```
<view id='com.cohga.client.panel.legend'>
  <label>Legend</label>
  <mapengine>mapengine.main</mapengine>
  <remove>hillshade,elevation</remove>
  <staticLegend url="http://server/images/markers.png" />
  <staticLegend url="http://server/images/aerial.png"
location='bottom' />
</view>
```

Remove map engine specific layers from legend. All layers from mapengine.vector and elevation from mapengine.raster.

```
<view id='com.cohga.client.panel.legend'>
  <label>Legend</label>
  <remove>*/mapengine.vector,elevation/mapengine.raster</remove>
</view>
```

Client Views Map

The Map View is used to provide the user with a means of navigating the data using a map.

ID

com.cohga.html.client.main.mapView

Sub-tags

Name	Type	Cardinality	Description
label	String	0..1	Label to display in tab
location	String	1..1	Which region to add the view to
toolbar	toolbar	0..1	Actions and components to show at top of map
scalebar	toolbar	0..1	Actions and components to show at bottom of map
contextmenu	context menu	0..1	The definition for the right click menu
showloading	boolean	0..1	If a progress bar is included in the client setting this value to true should enable the progress bar when requesting map images from Weave.
scales	list of scale	0..1	List of fixed scales to use for the map. If this value is set then the user will only be able to zoom to these fixed scales
toc	table of contents	0..1	A reference to a toc model to represent the layers in the map, generally this value should be set and is only optional to allow for a quick initial setup for testing purposes
extents	list of map extents	1..1	Contains the initial, full and limit extents of the map
crs	String	0..1	Coordinate reference system to use for the map
control	map control	0..n	An additional map component to be placed on the map
mapEngine	map engine	1..n	A source for data to display on the map
highlight	highlight	0..1	The default style to use to highlight an individual entity (for example when selected in a data grid)
minScale	Number	0..1	The minimum scale to zoom to when the map is zoomed into the boundary of an object. Useful for example when you may have small object to be zoomed to and you don't want the map scale to be set to 1:20 just to match the bounds of the object. By setting this value the map will zoom to the minScale value instead.
units	String	0..1	The map units that are to be used.
displayCrs	String	0..1	The CRS to use when displaying coordinates to the user, defaults to the same value as crs.

Scales

A list of scales can be added to the map which will prevent the user from zooming to arbitrary scales, that it they'll be limited to the values provided by this list.

The list is provided in the map view as a list of `<scale>` values within the `<scales>`


```
<scales>
  <scale>1000</scale>
  <scale>2000</scale>
  <scale>5000</scale>
  <scale>10000</scale>
  <scale>20000</scale>
  <scale>50000</scale>
  <scale>100000</scale>
  <scale>200000</scale>
  <scale>500000</scale>
</scales>
```

or as a comma separated list of values within the `<scales>`

```
<scales>1000,2000,5000,10000,20000,50000,100000,200000,500000</scales>
```

Table of Contents

A map view may be associated with a [Table Of Contents](#) model to provide the user with a more user friendly listing of the layers available in the map.

If no table of contents is linked explicitly with the map then Weave will create a generic one providing the user with a simple list of the available layers.

```
<toc ref="toc.vectors" />
```

Extents

The administrator can set the initial extent (used when the client starts and when the user uses the 'home extent' button), full extent (used when the user uses the 'full extent' button) and limit extent (used to restrict the users movement).

```
<extents>
  <initial minx="327098" miny="5811358" maxx="351971" maxy="
5827675" />
  <full minx="327098" miny="5811358" maxx="351971" maxy="
5827675" />
  <limit minx="320000" miny="5810000" maxx="360000" maxy="
5840000" />
</extents>
```

Note that these values can also be set in the `defaults` section of the client config, but the values set in the map view take precedence.

Coordinate Reference System

You can specify a default coordinate reference system that you wish to be used for the map. Obviously it should be one that is supported by the map engine, and preferably (for performance reasons) the default for the map engine.

The default if not set will be EPSG:4326.

```
<crs>EPSG:20255</crs>
```

Map Controls

Keyboard

com.cohga.client.mapctrl.keyboard

Adds keyboard control to the map panel (when the map has focus) allowing the user to pan and zoom the map using the keyboard

```
<control id="com.cohga.client.mapctrl.keyboard"/>
```

Scale Line

com.cohga.client.mapctrl.scaleLine

Display the current scale on the map as a line

```
<control id="com.cohga.client.mapctrl.scaleLine"/>
```

Display the current scale on the map as a line

Scale

com.cohga.client.mapctrl.scale

Display the current scale on the map as text

```
<control id="com.cohga.client.mapctrl.scale"/>
```

Scale Bar

weave.mapctrl.scalebar

Display the current scale or scale text, or both on the map

```
<control id="weave.mapctrl.scalebar">
  <!-- For simple scale text only -->
  <simple>false</simple>
  <!-- For display of scale text & scale bar -->
  <showText>true</showText>
  <!-- Controls the appearance of the scale bar, maximum is 2 -->
  >
  <lines>1</lines>
  <!-- Default units -->
  <units>m</units>
  <!-- Allow menu to change units by right mouse click over
scale bar -->
  <showMenu>true</showMenu>
  <!-- For placement of scale bar -->
  <bottom>5</bottom>
  <top>0</top>
  <left>10</left>
  <right>0</right>
</control>
```

Mouse Position

com.cohga.client.mapctrl.mousePosition

Display the current cursor position on the map

```
<control id="com.cohga.client.mapctrl.mousePosition" />
```

additional properties

name	default	description
numDigits	5	Number of digits of precision to display
granularity	10	Maximum number of pixels the mouse can move in one go before the position isn't updated
prefix	""	Text to place before the coordinates
suffix	""	Text to place after the coordinates
separator	", "	Text to place between the coordinates
halo	false	Set 'true' to enable white halo around text
background	false	Set 'true' to enable white background around text

Pan/Zoom Bar

com.cohga.client.mapctrl.panZoomBar

Add a control to the map to allow the user to directly pan and zoom the map

```
<control id="com.cohga.client.mapctrl.panZoomBar" />
```

Overview Map

com.cohga.client.mapctrl.overviewMap

Display an overview map on the map

```
<control id="com.cohga.client.mapctrl.overviewMap">
  <width>180</width>
  <height>70</height>
  <mapEngine id="mapengine.overview">
    <options>
      <initialLayerState>on</initialLayerState>
    </options>
  </mapEngine>
</control>
```

additional properties

name	default	description
minRatio	8	The ratio of the overview map resolution to the main map resolution at which to zoom farther out on the overview map
maxRatio	32	The ratio of the overview map resolution to the main map resolution at which to zoom farther in on the overview map

width	180	The default width in pixels of the overview map window
height	180	The default height in pixels of the overview map window

ratios

The ratio values are used to determine if the overview map extent should be changed.

From a coding perspective we divide the current overview map resolution by the main map resolution and compare the values to the min and max resolutions to determine if the overview map should adjust its extent.

```
ratio = ovmap.resolution / map.resolution;
needsAdjustment = (ratio <= minRatio) | (ratio > maxRatio)
```

The appropriate values are dependent upon the scales set for the main map engine in the map view. If you want to stop the zooming try setting minRatio to an large value and maxRatio to small value.

```
<maxRatio>-10000</maxRatio>
<minRatio>10000</minRatio>
```

Attribution

com.cohga.client.mapctrl.attribution

This control requires at least version 2.4.15.

Add the 'attribution' control then add an 'attribution' option to one or more client map engines.

```
<control id="com.cohga.client.mapctrl.attribution"/>

<mapEngine id="mapengine.vector">
  <options>
    <attribution>Data courtesy of ...</attribution>
  </options>
</mapEngine>
```

Touch

com.cohga.client.mapctrl.touch

Enable touch navigation for the map panel.

```
<control id="com.cohga.client.mapctrl.touch"/>
```

Highlight

You can alter the style used to highlight an individual entity (only from 2.5.18 onward) by adding a `highlight` tag to the map view which has the following properties

Highlight

Name	Type	Description
vector	vector	The styling for polygon and line based entities
marker	marker	The styling for point based entities

buffer	vector	The styling for buffers
autoClear	boolean	Should the highlights be automatically removed after a fixed delay or when the map is next refreshed, default true (from 2.5.23)
autoClearDelay	integer	Number of milliseconds to wait before automatically removing the highlights if autoClear is enabled (from 2.5.23)

Vector

Name	Type	Description
strokeColor	string	The colour for the outline (this is a CSS colour so may have to be forced to be a string if the config file reader determines that the value you have set is an integer)
strokeOpacity	float	The opacity for the outline, 0.0 - 1.0
strokeWidth	integer	The width of the outline
fillColor	string	The colour of the fill (this is a CSS colour so may have to be forced to be a string if the config file reader determines that the value you have set is an integer)
fillOpacity	float	The opacity of the fill, 0.0 - 1.0

Marker

Name	Type	Description
icon	string	A URL pointing to the icon to use
width	int	The width of the icon
height	int	The height of the icon
offsetx	int	The x offset for the hotspot within the icon
offsety	int	The y offset for the hotspot within the icon

```

<highlight>
  <marker>
    <icon>resources/images/markers/w-marker-red.png</icon>
    <width>24</width>
    <height>30</height>
    <offsetx>-8</offsetx>
    <offsety>-28</offsety>
  </marker>
  <vector>
    <strokeWidth>3</strokeWidth>
    <strokeColor xsd:type="xsd:string">#d27316</strokeColor>
    <strokeOpacity>0.75</strokeOpacity>
    <fillColor>red</fillColor>
    <fillOpacity>0.25</fillOpacity>
  </vector>
  <buffer>
    <strokeWidth>3</strokeWidth>
    <strokeColor>orange</strokeColor>
    <fillOpacity>0</fillOpacity>
  </buffer>
  <autoClear>>false</autoClear>
</highlight>

```

The map engine configures the actual source for the map image. Different types of map engines can be specified and more than one map engine can be used.

Previously one of the map engines needed to be specified as the "base layer" by setting `isBaseLayer` to true for that particular map engine, this is no longer the case.

There is no specific user controllable base layer as part of the map view any more, it is maintained internally by Weave.

Multiple map engines

A single map engine can represent a single map layer, for example the aerial photography from 1968, or multiple map layers, for example all of the vector data related to water utilities. They can also be combined in the client configuration so that multiple images are overlaid to combine the data from more than one map engine.

Example of using three map engine to display raster, vector and a separate selection layer.

```
<view id="com.cohga.html.client.map.mapView">
  ...
  <mapEngine id="mapengine.raster">
    <options>
      <alpha>>false</alpha>
    </options>
  </mapEngine>
  <mapEngine id="mapengine.vector"/>
  <mapEngine id="mapengine.selection">
    <options>
      <selection>>true</selection>
    </options>
  </mapEngine>
</view>
```

Note that to properly use the above configuration you may also need to update your toc model or add an image slider. Also, there are issues with transparency that will need to be considered (especially if you're unlucky enough to have to support Internet Explorer 6).

A toc model suitable for the above map engine configuration would set the map engine for the selection and raster layers to the same engines as configured above, rather than leaving them assigned to the original map engine.

For example assuming that we previously had a single map engine, `mapengine.main`, that contained raster and vector layers and was also used to draw the selection. In this case the toc model and client config could be something like:

```
<client:config id="client.main">
  ...
  <view id="com.cohga.html.client.map.mapView">
    ...
    <mapEngine id="mapengine.main"/>
  </view>
  ...
</client:config>

<toc:model id="toc.main">
  <mapengine>mapengine.main</mapengine>
  <entry layer="_selection.active" label="Selection"/>
  <entry layer="property" label="Property" entity="property"/>
  <entry layer="suburbs" label="Suburbs" entity="suburbs"/>
  <entry layer="photography" label="Photography" checked="false"
/>
</toc:model>
```

But if we wanted to switch to the client configuration with the three map engines then our toc model would become something like:

```
<toc:model id="toc.main">
  <mapengine>mapengine.vector</mapengine>
  <entry layer="_selection.active" label="Selection" mapengine="
mapengine.selection"/>
  <entry layer="property" label="Property" entity="property"/>
  <entry layer="suburbs" label="Suburbs" entity="suburbs"/>
  <entry layer="photography" label="Photography" checked="
false" mapengine="mapengine.raster"/>
</toc:model>
```

Which makes the default map engine `mapengine.vector`, and changed the selection layers map engine to `mapengine.selection` and the aerial photography switched to `mapengine.raster`.

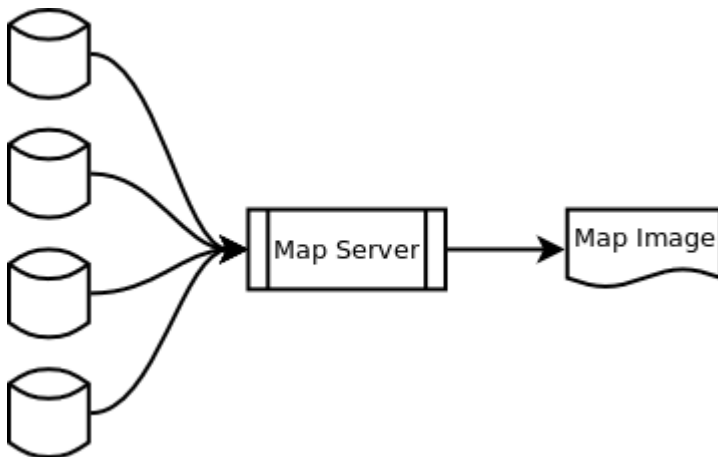
Then on the client side the map panel will now contain three separate map layers, one for each map engine, which may improve performance because it cuts down on the number of layers that must be drawn on the server when the user toggles a single layers visibility.

For example the selection layers are drawn a lot quicker in the above example because when the selection is changed only the selection layers need to be redrawn, not all of the other layers that are now in the `mapengine.vector` layer.

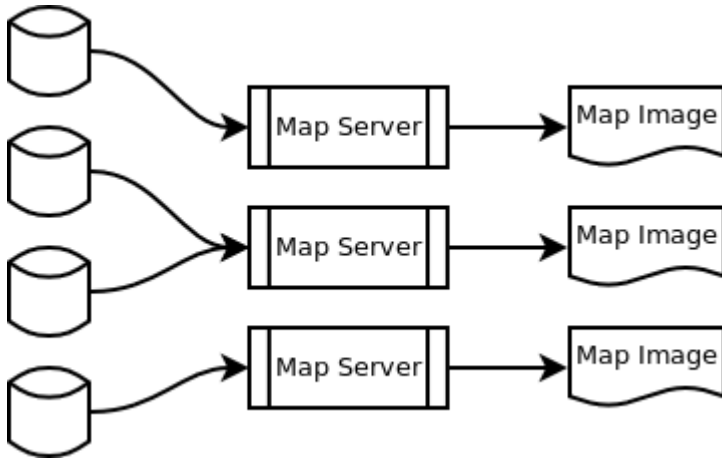
Transparency and multiple map engines

This may also make it clearer as to why transparency is now an issue, previously all of the map layers were drawn on the map server at once and the map server knew how to overlay the individual layers to ensure that lower layers show through the transparency of higher layers, but now that overlaying is performed on the client, so if the client doesn't support transparency properly (IE6 with 8-bit PNG images) or if the map engine doesn't provide full transparency support (ArcIMS) then the map panel on the client can look terrible. Sometimes that can be resolved by switching the image generation to 24 or 32 bit PNG or disabling anti-aliasing.

The following 2 images show the difference between using a single map engine, where the map server is retrieving the data, generating a single map image, and sending that to the client for display, and multiple map engines, where multiple images are sent to the client where they're overlaid.



In the above situation, since one map server is generating the map image it can produce the best image, making proper use of anti-aliasing, transparency, both full and partial. But because the image includes all map layers a change to any of the layer requires a whole new image to be drawn, which could mean that every map update is slow(er).



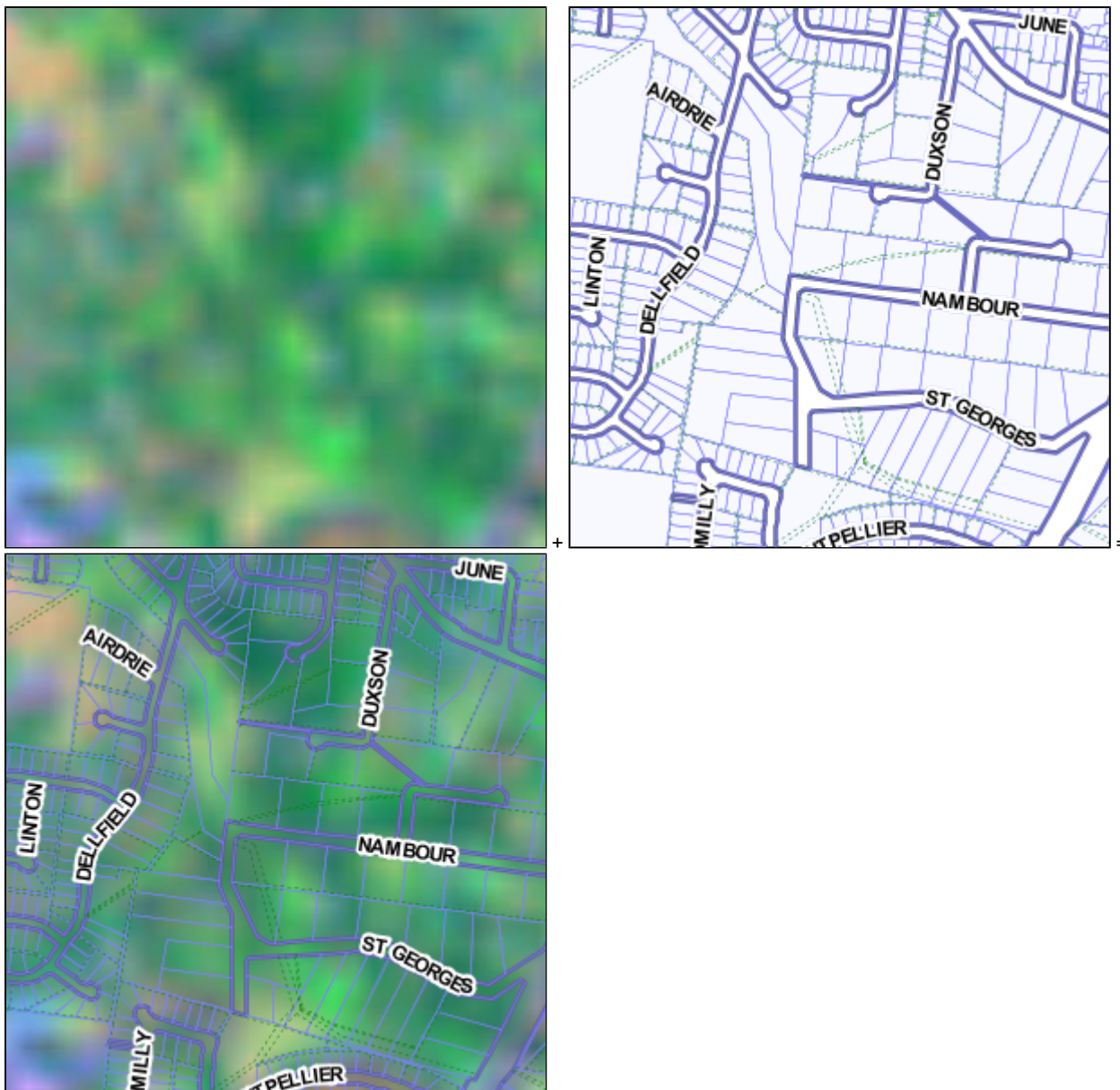
In this situation multiple map servers are generating the images, which means that a change to one of the layers will result on the request for a a map image that has fewer layers to draw, which should result in quicker response times. Of course this is the case if you're toggling the visibility of a single layer, if you're panning or zooming the map then the same number of map layers will need to be drawn, but there could still be a performance advantage if the map servers are running on multiple physical servers, or a server is spare capacity, since they'll be able to generate the map images at the same time (3 map servers generating 3 images with 2 layers each will be quicker that 1 map server generating 1 image with 6 layers).

The down side to this arrangement is that the map server can't optimise the final image, and further, if the map server doesn't support transparency properly then any anti-aliasing around transparent areas can look terrible. Also, some map engines (ArcIMS I'm looking at you) don't support partial transparency, so that if you, for example, specify a 50% transparent yellow fill for a polygon layer and there's no layers below that layer then rather than producing a final image that has the area filled with a transparent yellow fill it will instead produce an opaque fill that is 50% yellow and 50% whatever the background colour is set to. Which means that when you overlay the image on top of another image rather than getting the bottom image showing through with a yellow tint, you instead don't see the bottom image at all and just see the yellow/background colour image.

Opacity and multiple map engines

This is assuming that the top image doesn't have its opacity altered to be less that 100%, which is a different issue all together. Since the map engine can have it's opacity set to less that 100% in the client configuration it's possible to overlay a map image from a map server that doesn't support transparency at all and still have the underlying map image show through, for example setting the opacity to 0.5 (50%) would show a mix of 50% of the underlying image and 50% of the top image. The problem where is that the overall image appears washed out.

The images below show combining a raster image with a vector image that properly supports transparency (generated by GeoServer) with the vector images' opacity set to 1 (100%)



compared to combining a raster image with a vector that doesn't support transparency, but this time with the vector opacity set to 0.5 (50%)

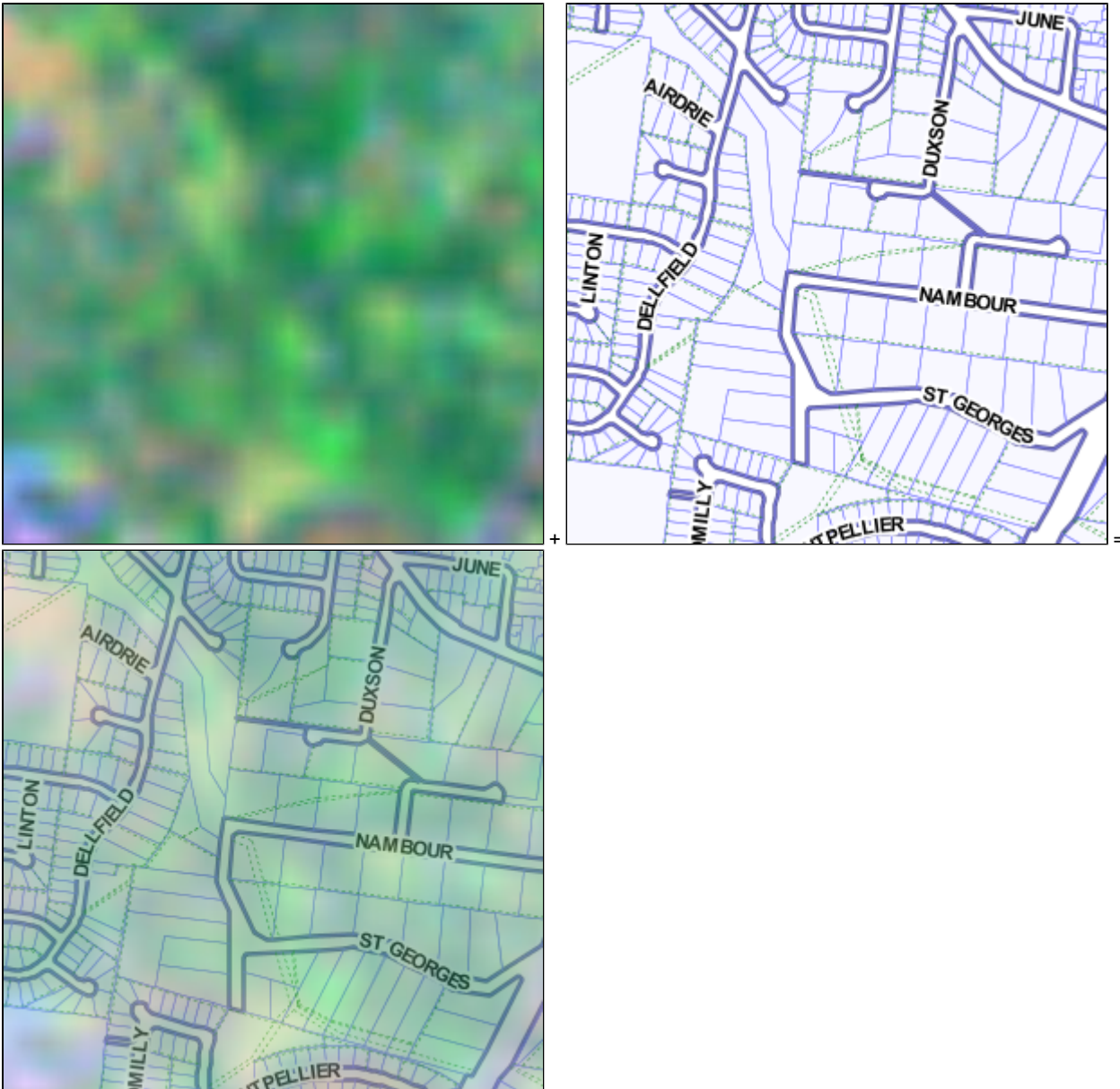
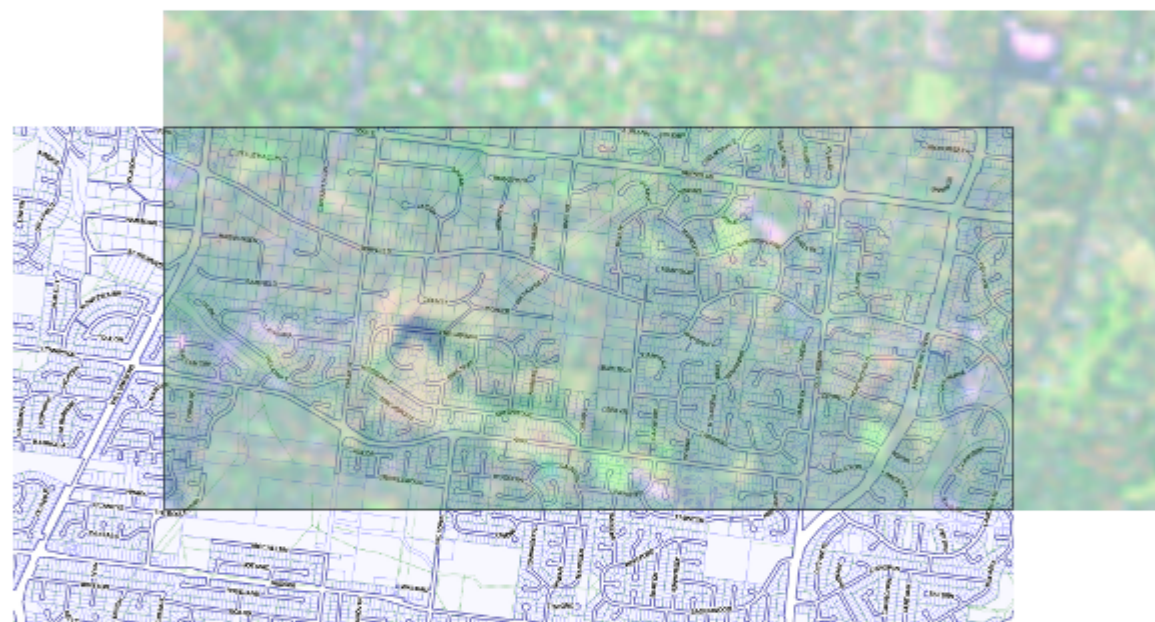
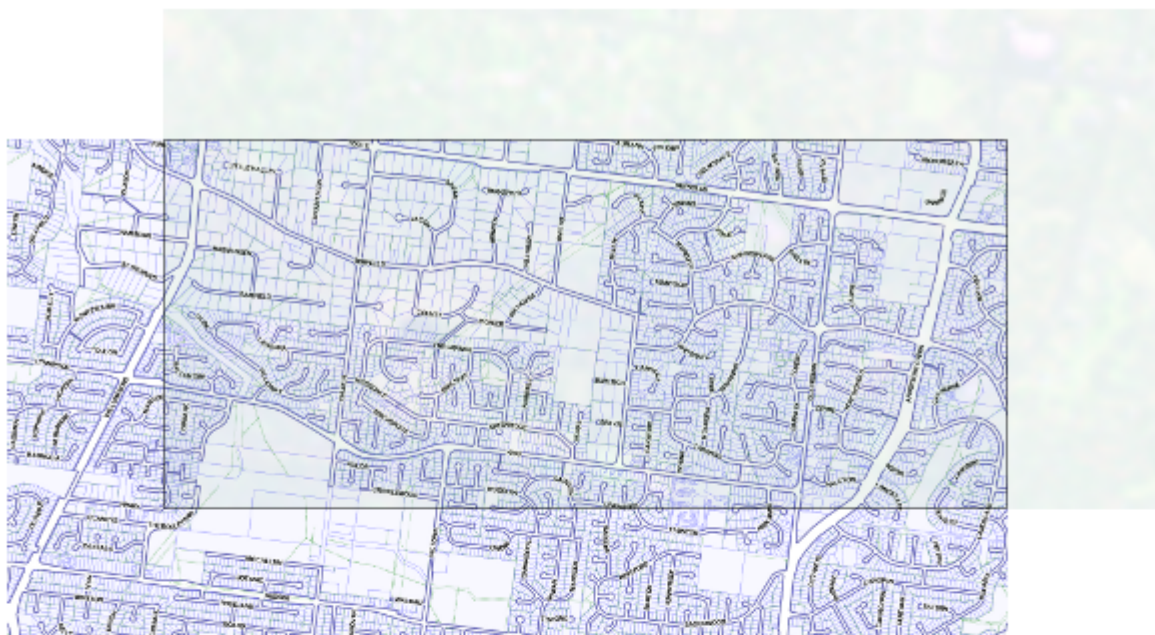


Image Slider

An alternative for when you have multiple overlaid map engines on the client is that you can use a slider to transition between layers. The image transition slider allows the user to dynamically adjust the opacity of a single map engine.

Two options when using a single image slider is that if you have map engines that support transparency then you can transition the underlying image and still have the overlaying image visible, or if you don't have full transparency support then you can use it to transition the overlaying image and completely replace the underlying image.

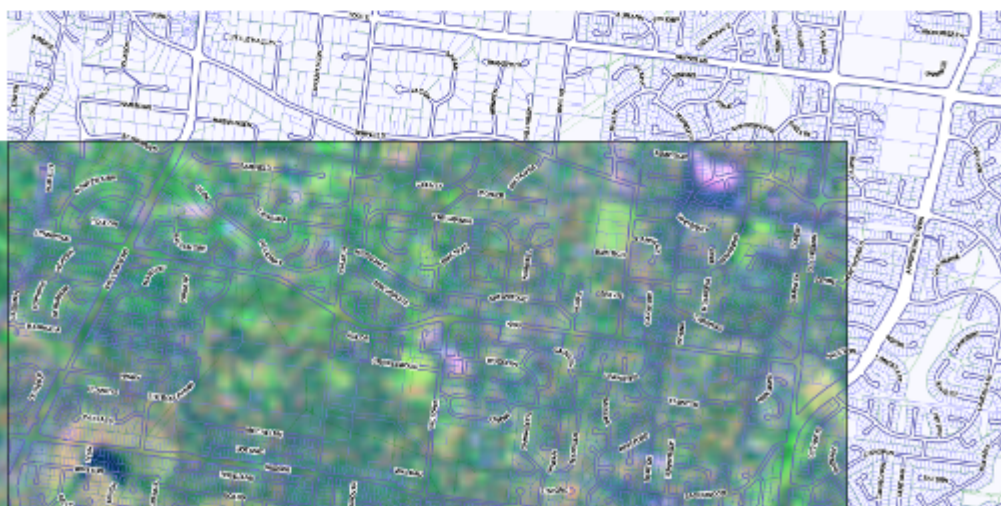
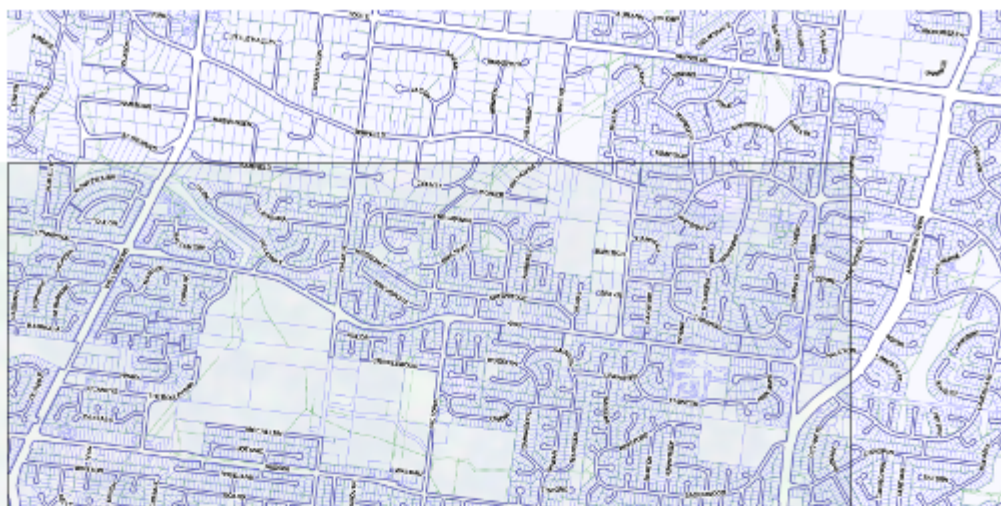
The following 3 images show an example when the transition slider is attached to a raster image map engine that's overlaid on top of a vector map engine that does **not** support transparency (this could also be reversed with the vector map engine on top and the transition slider attached to it to accomplish exactly the same thing). The top image shows the slider at 10%, the second at 50% and the third at 100% (when the slider is a 0% you'll only see the vector image).





You can see that the vector image is gradually replaced entirely by the raster image (the area where the two images intersect is what the client would expect to see). And you can imagine if the map engines were reversed, with the raster below the vector and the slider attached to the vector, then the process would be the same, but with the vector layer replacing raster rather than the raster replacing the vector (since the vector image is opaque). If the map engines were reversed but the slider still attached to the raster map engine then there would be no visible effect from the slider since the opaque vector image on top would be occluding the raster image anyway, regardless of its opacity.

The following 3 images show an example with the transition slider again attached to the raster image, but in this case the raster map engine is below the vector map engine, but in this case the vector map engine supports transparency, so as the slider progresses from 10% to 50% to 100% the vector layer is still visible and the raster that shows through the transparent parts of the vector image (and again when the slider is at 0% you'll only see the vector image).





In this case the raster image is gradually exposed through the vector image. If this configuration was kept (vector on top, raster below and image slider attached to raster image) but the vector images' map server didn't support transparency then moving the slider would have no visible effect, since the layer would be hiding the raster layer no matter what opacity the raster layer is set to.

The simplest example of this is to transition between a vector and raster layer

```
<toc:model id="toc.main">
  <mapengine>mapengine.vector</mapengine>
  <entry layer="_selection.active" label="Selection" mapengine="
mapengine.selection"/>
  <entry layer="property" label="Property" entity="property"/>
  <entry layer="suburbs" label="Suburbs" entity="suburbs"/>
</toc:model>

<client:config id="client.main">
  ...
  <view id="com.cohga.html.client.map.mapView">
    ...
    <statusbar>
      <item>Raster</item>
      <item component="com.cohga.html.client.
components.imageTransSlider" mapEngine="mapengine.vector"/>
      <item>Vector</item>
    </statusbar>
    ...
    <mapEngine id="mapengine.raster">
      <options>
        <alpha>>false</alpha>
      </options>
    </mapEngine>
    <mapEngine id="mapengine.vector"/>
    <mapEngine id="mapengine.selection">
      <options>
        <selection>>true</selection>
      </options>
    </mapEngine>
  </view>
</client:config>
```

Here we've kept the map engine configuration in the client the same as we have in the earlier example but we've added the image trans slider to allow the user to fade in and out the vector image. This example assumes that the vector image is opaque, if it were transparent you could attach the slider to the raster image instead and you'd always have the vector image visible.

You may have noticed that the Photography layer has been removed from the toc model, which means that the only way for the user to view the aerial photography is to use the slider, since it doesn't make a lot of sense to allow the user to toggle the display of the layer via the toc panel and also use the slider to change its visibility.

If you had multiple raster layers that you wish to attach to the slider you could create something like the following toc model

```
<toc:model id="toc.main">
  <mapengine>mapengine.vector</mapengine>
  <entry layer="_selection.active" label="Selection" mapengine="
mapengine.selection"/>
```

```

    <entry layer="property" label="Property" entity="property"/>
    <entry layer="suburbs" label="Suburbs" entity="suburbs"/>
    <entry label="Imagery" folder="true" exclusive="true">
      <mapengine>mapengine.raster</mapengine>
      <entry layer="aerial_2000" label="Photography 2000"
checked="true"/>
      <entry layer="aerial_2008" label="Photography 2008"
checked="false"/>
      <entry layer="dem" label="Elevation" checked="false"/>
      <entry layer="melways" label="Melways" checked="false"
/>
    </entry>
  </toc:model>

```

Then depending upon the ordering of the map engines in the client configuration you can attach the slider to the vector or raster map engines while also giving the user the ability to choose which raster layer to display.

Client side map engines

All of the map engines configured in the client in the previous examples use the Weave server to generate the map images on behalf of the client, that is the Weave client is communicating with the Weave server to generate the map images and the Weave server is calling the map engine to create the images which are then sent back to the client.

An alternative to this is to have the client directly calling the map engine to generate the images, and bypassing the Weave server all together. Unfortunately at this stage ArcIMS and ArcGIS Server are NOT directly supported in this configuration but can be used if accessed via WMS, which is supported on the client.

One reason for doing this would be to improve performance as it removes the extra layer of interaction with the Weave server, but it does mean that the client will require direct access to the map engine, whereas previously since it was the Weave server that was supplying the client with the images no access was required to the actual map engine by the client. So while performance may improve security will decrease and configuration complexity will increase.

Another limitation of this method is that as far as the user is concerned there is only a single layer coming from the map engine, even if the image generated by the map engine is composed of multiple feature layers. This means that if you add multiple map engines to the client configuration all pointing to client side map engines then the toc model must be setup to include a single layer entry for each engine, and the user can only display or hide the whole layer, not individual layers within the map engine.

One issue that arises when using a map engine that only the client knows about is when it comes time to generate a report with a map embedded in it. Currently this requires a duplicate map engine to be configured for the server to use and the client map engine configuration setup to tell Weave that it should use this map engine when it needs to generate a map on the server.

So for example lets look at what we'd need to setup the client with a very simple WMS map engine

```

<client:config id="client.main">
  ...
  <view id="com.cohga.html.client.map.mapView">
    ...
    <mapEngine id="mapengine.wms">
      <type>wms</type>
      <url><![CDATA[http://columbo.nrlssc.navy.mil
/ogcwms/servlet/WSServlet/Evansville_IN_Maps.wms]]></url>
      <options>
        <alpha>false</alpha>
      </options>
    </mapEngine>
  </view>
</client:config>

```

All we've done here is set the map engine type to wms (the default is 'weave') and provided it with the url that it need to access the server. From there the client will directly interact with the WMNS server and not Weave when it need to draw a map image.

From here if we want the user to be able to generate a report containing a map then we need to alter the client map engine configuration to tell it how the server can access this map engine.

To do this we need to first add a new map engine configuration pointing to the WMS server then alter the client map engine configuration to point to this new map engine

```
<wms:mapengine id="mapengine.wms">
  <url><![CDATA[http://columbo.nrlssc.navy.mil/ogcwms/servlet
/WMServlet/Evansville_IN_Maps.wms]]></url>
  <layers>
    <layer>0:2</layer>
  </layers>
</wms:mapengine>

<client:config id="client.main">
  ...
  <view id="com.cohga.html.client.map.mapView">
    ...
    <mapEngine id="mapengine.wms">
      <type>wms</type>
      <url><![CDATA[http://columbo.nrlssc.navy.mil
/ogcwms/servlet/WMServlet/Evansville_IN_Maps.wms]]></url>
      <options>
        <alpha>false</alpha>
        <server>
          <mapengine>mapengine.wms<
/
mapengine>
          <layers>0:2</layers>
        </server>
      </options>
    </mapEngine>
  </view>
</client:config>
```

Here we've created a new server side map engine that reflects the map engine used by the client and then setup the client to tell the server to use this map engine when it needs to generate a map image. We've also restricted the layers available through that map service to a single layer. The id used for the layers is taken from the WMS map service, in the example url I used 0:2 is the id for a 2005 aerial photography layer.

Tiled map engines

Te client side map stuff is all well and good but the best reason for using a client side map engine is that the client can then use a tiled map service. This is a map service where the map is generated in smaller tiles rather than as a single whole image, this provides large advantages in performance because once an individual tile is cached in the client browser they don't need to download or generate that tile again, which results in significant speed improvements for the client compared to generating a complete map image each time the user alter the map display.

The down side to tiled map engines is that they are fixed images and the user can not turn on and off the display of sub-layers within the map image. Whilst this generally isn't an issue for aerial photography type map images or some other raster based layers it can be a problem with vector layers.

For example if you have a map engine composed of three vector layers and you want to give the user the option to turn on and off the display of each of the layer individually you would need to generate 8 tile caches. One with no layers, three for when only one layer is visible, three for when the combinations of two layers are visible and one for when all layers are visible. If you wanted to add a new layer to the mix then you'd need 16 tile caches, another layer would mean 32 tile caches, I'm sure you can see where this is going.

For this reason if you have a large number of vector layers and you want to provide the user with the ability to selectively display or hide some of the layers then you'll probably still need to use a non-tiled weave layer. But if you have a limited number of layers, wish to create a very simple map environment or just want to add a single raster layer then tiles may be an option.

The easiest (and least useful) way to setup a tile cache is to add the `singleTile` flag to a WMS client side map engine and set it to `false`. This will tell the client to send requests to the WMS server asking for small tiles rather than as a single large image. But unless the WMS map engine is specifically configured to handle requests for tiles the performance of this setup will be severely compromised.

Alternatively the client map engine can be configured to use a specialized tile caching mechanism instead of WMS. This is where the map engine supplying the tiles is specifically designed and configured to supply image tiles rather than imposing this on a map engine that wasn't specifically designed for it like WMS.

Weave currently support (at least) two different tiling mechanisms [TileCache](#) and [TMS](#). These two tiling mechanisms use specially crafted urls to determine which tile to fetch and mainly differ in the formatting of the url.

Setting up and tiled map engine using one of these mechanisms is similar to the WMS map engine but the `type` should be `tilecache` or `TMS`, and both still require the `url` parameter.

```
<client:config id="client.main">
  ...
  <view id="com.cohga.html.client.map.mapView">
    ...
    <mapEngine id="mapengine.tiled">
      <type>tilecache</type>
      <url><![CDATA[http://tileserver:8080
/tilecache]]></url>
      <layername>aerial2008</layername>
      <options>
        <alpha>>false</alpha>
        <singleTile>>false</singleTile>
        <format>image/jpeg</format>
      </options>
    </mapEngine>
  </view>
</client:config>
```

In the case of the `TileCache` version it also requires a `layername` parameter. This is used when constructing the url and allows the same tile cache to server multiple sets of tiles.

You'll also notice that we set `singleTile` to `false` (which isn't really necessary since a tile cache layer already knows that it's generated using tiles) and changed the format (the default is `image/png` for `TileCache` tiles). For a `TMS` tile cache the `layername` isn't required

Depending upon the actual tile cache implementation used to provide the tiles there may be two options available, one to generate the tiles as requested and the other to pre-generate the tiles. The first option has the advantage that only the tiles actually viewed by the users are ever generated, thereby reducing disk storage, but the performance will be slow the first time the tiles are generated. The second option has the advantage that it's lightning fast right from the start, but has the disadvantage that the whole tile cache needs to be generated before it's available even those tiles that will never be seen by any users and therefore uses more disk storage (up front, over time the other option can also result in large disk usage as most of the map areas is viewed).

It can take many hours of processing to pre-generate a tile cache and use hundreds of gigabytes of disk space, depending upon how large an area is covered and how many levels of tiles are generated.

Currently Weave provides commands to build [TileCache](#) formatted tile cache for any map engine supported by Weave.

Map Engine Options

Below is a list of the options that are available for use when setting up a map engine

Option	Description
refreshInterval	Used to automatically refresh the map engine. Should be set to a positive integer that specifies the number of second between refreshes. If not set then no map refresh will occur.

transitionEffect	Determines how the browser displays the transitions between map images when a zoom in or out has been performed. Setting this option to <code>force</code> will ensure that the previous map image will be scaled and displayed while waiting for the new map image. If this option isn't set then a blank map image will be displayed while waiting for the new map image to arrive.
singleTile	Indicates that this map engine generates its display by requesting a single map image from the server, as opposed to requesting multiple smaller tiled map images. Default is <code>true</code> but this generally does not need to be set explicitly since it is determined from the map engine type.
selection	Indicates that this map engine will display selections and should be set to <code>true</code> for any map engine that will be displaying selections, default is <code>false</code> . Only one map engine should have this set to true.
alpha	Indicates to the browser if this map engine has an alpha channel, default is <code>true</code> , set it to <code>false</code> if you know the map engine will always be opaque, for example for an aerial photo layer covering the entire map area.
ratio	Indicates how big the map image (the buffer) should be compared to the size of the map panel itself, i.e. a value of 1.0 will generate requests for maps that are the same size as the map panel, a value of 2.0 will generate images that are twice the size of the map panel. This allows the user to pan the map without having to request a new map image from the server, at the cost of generating a bigger (and therefore slower) map image on the first request. The default is 1.5
isBaseLayer	This is obsolete and should not be set as an option as it is controlled internally by Weave, which will set it to <code>false</code> if it is set anyway.
minScale	The minimum scale at which the client will draw the map engine (for supported map engines).
maxScale	The maximum scale at which the client will draw the map engine (for supported map engines).

Client Views Projections

Provides a means of displaying the current cursor position in different projections.

Note that you need to know the Proj4 definition of the alternate projections you wish to display.

ID

`com.cohga.html.client.main.projView`

Sub-tags

Name	type	cardinality	description
label	String	0..1	Label to display in tab
location	String	1..1	Which region to add the view to
projection	String	1..n	The definitions and labels of the projections to display
server	Boolean	0..1	If the transformation need should be undertaken on the server

Example

```

<view id='com.cohga.html.client.main.projView'>
  <label>Projections</label>
  <location>west</location>
  <projection id="EPSG:28354" label="GDA94/MGA Zone 54" />
  <projection id="EPSG:28355" label="GDA94/MGA Zone 55" />
  <projection id="EPSG:28356" label="GDA94/MGA Zone 56" />
  <projection id="EPSG:20254" label="AGD66/AMG Zone 54" />
  <projection id="EPSG:20255" label="AGD66/AMG Zone 55" />
  <projection id="EPSG:20256" label="AGD66/AMG Zone 56" />
<!--
The following three entries show examples of how a custom transform
can be specified.
This does NOT have to be done if you want to use any of these three
projections, it's just done here to show you how you could do it.
-->
  <projection id="EPSG:4283" label="GDA94">
    <def>+proj=longlat +ellps=GRS80 +towgs84=0,
0,0,0,0,0,0 +no_defs</def>
  </projection>
  <projection id="EPSG:3111" label="VicGrid94">
    <def>+proj=lcc +lat_1=-36 +lat_2=-38 +lat_0=-37
+lon_0=145 +x_0=2500000 +y_0=2500000 ellps=GRS80 +towgs84=0,
0,0,0,0,0,0 +units=m +no_defs</def>
  </projection>
  <projection id="EPSG:3112" label="GDA94/Geoscience Australia
Lamberts" />
    <def>+proj=lcc +lat_1=-18 +lat_2=-36 +lat_0=0
+lon_0=134 +x_0=0 +y_0=0 +ellps=GRS80 +towgs84=0,0,0,0,0,0,0 +units=m
+no_defs</def>
  </projection>
</view>

```

Coordinate transformations using the server

```

<!-- This example shows a client defined in GDA2020 and reporting
coordinates in GDA94 -->
<view id='com.cohga.html.client.main.projView'>
  <label>Projections</label>
  <location>west</location>
  <!-- The following definitions will do the transformation on
the server using the NTV2 transformation from GDA2020 to GDA94 -->
    <projection id="EPSG:28354" label="GDA94/MGA Zone 54"
server="true" />
    <projection id="EPSG:28355" label="GDA94/MGA Zone 55" server="
true" />
    <projection id="EPSG:28356" label="GDA94/MGA Zone 56" server="
true" />
    <projection id="EPSG:3111" label="GDA94/VicGrid94"
server="true" />
    <projection id="EPSG:4283" label="GDA94" server="true"
/>

```

```

        <!-- The following two definitions will do the
transformation in the browser as the map datum is already in GDA2020
-->
        <projection id="EPSG:7844" label="GDA2020" />
        <projection id="EPSG:7855" label="GDA2020/MGA Zone 55"
/>
</view>

```

Client Views Quick Search

The Quick Search panel provides the supporting components, both client and server, for adding a new panel to the client that allows the user to quickly find an entity and see details about that entity.

This tool is a combination of the index combo and the data grid. It is useful for a Weave instance that has a cut down client with few tools.

```

<view id="com.cohga.view.quicksearch">
  <label>Quick Search</label>
  <location>west</location>
  <minscale>1000</minscale>
  <detail>
    <data>ar_road_details</data>
  </detail>
  <detail>
    <entity>property</entity>
    <datadefinition>dd_property_details</datadefinition>
    <template><![CDATA[
      <b>Property</b><br/>
      {PropNo} {StreetName} {StreetType}
    ]]></template>
  </detail>
</view>

```

Client Views Search

This is the basic search input form.

ID

com.cohga.html.client.main.searchView

Sub-tags

name	type	cardinality	default	description
label	String	0..1	Search	Label to display in tab
location	String	1..1		Which region to add the view to
searchComboWidth	Integer	0..1	250	The width of the search combo
searchComboListWidth	Integer	0..1	250	The width of the search combo popup list

Example

Basic search view

```
<view id='com.cohga.html.client.main.searchView'>
  <label>Find</label>
  <location>east</location>
</view>
```

Localisation

It's possible to localise the text used in the search panels so they're displayed in the local language of the user, this localisation is performed using the standard localisation mechanism outlined at [Internationalisation and localisation](#).

This was only made available for the search panel in version 2.24 of the com.cohga.client.weave.main bundle

The edit panel provides a set of resource id's that it understands and if they're set for the users locale they will be used instead of the defaults values.

id	default
search.error.validation	The search has fields that are required to be entered. Please fix the errors marked in red.
search.reset.tooltip	Reset the Form
new	New
add	Add
remove	Remove
refine	Refine

The last four are global resources that are used in places other than the search panel.

The following resources item can be used to replace the text for all of the items

```
<client:resources>
  <search>
    <error>
      <validation>The search has fields that are
required to be entered. Please fix the errors marked in red.<
/validation>
    </error>
    <reset>
      <tooltip>Reset the Form</tooltip>
    </reset>
  </search>
  <new>New</new>
  <add>Add</add>
  <remove>Remove</remove>
  <refine>Refine</refine>
</client:resources>
```

Client Views Simple Map

The Simple Map View is a cut down map view that allows you to add additional map views to your client config.

It is currently an experimental client component and has a number of limitations.

- It can't be the first map view displayed, for now it needs to be used in a tab region and have the main map tab be the first visible one and this new map view on a different tab.
- There are no tool bars or status bars available, which also means that you can't use any of the existing tools, zoom, pan, select, etc, so if you want the users to be able to pan/zoom around the map you'll need to add controls to the view (`com.cohga.client.mapctrl.navigation` or `com.cohga.client.mapctrl.panZoomBar`).
 - The main map view and the simple map view should remain in sync when panning/zooming around.
 - Other map controls may or may not work (they should but who knows how they'll interact with the other map view).
- You have no control over what layers are displayed on the map aside from specifying the map engines to be used within the map, all layers within the map engines listed in the view will be turned on.
 - You can control exactly what layers are displayed by setting up the required map engines with just the layers you want to have displayed.
 - Setting `isBaseLayer` to true will place the map engine into a group where only one of the group can be turned on at a time.
- You can't use map engines in this view that are used in the main map view, because when the simple map view is shown it turns on all the layers in the underlying map engine, which causes the layers to all be turned on in the main map view if it's using the same map engine.

Example client view config

```
<view id="com.cohga.html.client.map.simpleMapView">
  <label>Imagery</label>
  <location>center</location>
  <control id="weave.mapctrl.keyboard"/>
  <control id="com.cohga.client.mapctrl.touch"/>
  <control id="com.cohga.client.mapctrl.navigation"/>
  <control id="com.cohga.client.mapctrl.mousePosition"/>
  <control id="com.cohga.client.mapctrl.panZoomBar"/>
  <control id="com.cohga.client.mapctrl.layerSwitcher"
maximized="true" baseLabel="Raster" dataLabel="Vector"/>
  <mapengine id="mapengine.floods">
    <options>
      <label>Floods</label>
      <isBaseLayer>true</isBaseLayer>
    </options>
  </mapengine>
  <mapengine id="mapengine.planning">
    <options>
      <label>Planning</label>
      <isBaseLayer>true</isBaseLayer>
    </options>
  </mapengine>
  <mapengine id="mapengine.rail">
    <options>
      <label>Rail</label>
    </options>
  </mapengine>
  <crs>EPSG:28355</crs>
  <scales>
    <scale>500</scale>
    <scale>1000</scale>
    <scale>2000</scale>
    <scale>4000</scale>
    <scale>8000</scale>
    <scale>16000</scale>
    <scale>32000</scale>
    <scale>64000</scale>
    <scale>125000</scale>
    <scale>250000</scale>
  </scales>
</view>
```

```

</scales>
<extents>
    <initial crs="EPSG:28355" minx="327098" miny="
5811358" maxx="351971" maxy="5827675"/>
    <full crs="EPSG:28355" minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
    <limit crs="EPSG:28355" minx="320000" miny="5810000"
maxx="360000" maxy="5840000"/>
</extents>
</view>

```

The current version of the component can be downloaded from [here](#).

The file should be downloaded and copied to the `...\weave\platform\plugins` directory, an entry added to the `config.ini` file in the `...\weave\platform\configuration` directory, and then a new view added to the client config.

Client Views ToC

Provides a means of displaying the current layer and gives the user the ability to turn on/off the layer visibility.

The ToC view provides a placeholder to display the layers that are available on the map, but it doesn't actually describe the layers that are available, to do that you need to set up a [toc model](#).

ID

`com.cohga.html.client.main.tocView`

`weave.toc`

Sub-tags

Name	type	cardinality	default	description
label	String	0..1		Label to display in tab
location	String	1..1		Which region to add the view to
toolbar	toolbar	0..1		A toolbar of items to appear at the top of the view
statusbar	statusbar	0..1		A toolbar of items to appear at the bottom of the view
contextmenu	contextmenu	0..1		a right-click context menu
checkParentOnVisible	boolean	0..1	false	Should the parent group also be turned on when a layer is turned on?
turnOnActive	boolean	0..1	false	Should the associated layer be turned on when the active entity is changed? Only available since 2.5.26
allowExclusiveOff	boolean	0..1	true < 2.6.5 false >= 2.6.5	Should it be possible to turn off the active child in an exclusive ToC group? Only available since 2.6.6 and 2.6.5.2
activateEntity	boolean	0..1	false	Should the associated entity be set active when the layer is turned on? Only available since 2.6.6
showFilterIcon	boolean	0..1	false	Should a filter icon indicator be displayed for layers that are currently filtered? Only available since 2.6.6

Example ToC view will all available items

```

<view id="com.cohga.html.client.map.tocView">
    <label>Layers</label>
    <location>west</location>
    <toolbar>
        <item action="weave.toc.expandAll"/>
        <item action="weave.toc.collapseAll"/>
    </toolbar>
</view>

```

```

        <item action="weave.toc.save"/>
        <item action="weave.toc.load"/>
        <item>-&gt;</item>
        <item action="weave.toc.favourites"/>
        <item component="weave.toc.filter"/>
    </toolbar>
    <contextmenu>
        <item action="weave.toc.styleLayer"/>
        <item action="weave.toc.selectLayer"/>
        <item action="weave.toc.zoomToLayer"/>
        <item action="weave.toc.zoomSelection"/>
        <item action="weave.toc.zoomToMinScale"/>
        <item action="weave.toc.zoomToMaxScale"/>
        <item action="weave.toc.clearSelection"/>
        <item action="weave.toc.lockLayer"/>
        <item action="weave.toc.favourite"/>
        <item action="weave.toc.checkAll"/>
        <item action="weave.toc.uncheckAll"/>
        <item action="weave.toc.filterLayer"/>
    </contextmenu>
</view>

```

Toolbar and status bar Items

`weave.toc.expandAll`

Expands all the groups/folders within the ToC view.

`weave.toc.collapseAll`

Collapses all the groups/folders within the ToC view.

`weave.toc.save`

Saves the current ToC state so it can later be restored, note that only a single ToC state can be saved at once, using the save button will replace the currently saved ToC state.

`weave.toc.load`

Load the previously saved ToC state.

`weave.toc.filter`

Refine the layers listed in the ToC.

`weave.toc.favourites`

Toggle the display of just the favourite layers, i.e. those that have been marked as favourites using the `weave.toc.favourite` action list below. Available from 2.6.5.

Context Menu Items

`weave.toc.styleLayer`

If the layer supports multiple styles then this allows the user to switch between the available styles.

`weave.toc.selectLayer`

If the layer is associated with an entity this allows the user to change the active entity. Requires that the `entity` attribute is specified for the layer in the toc model.

`weave.toc.zoomToLayer`

Changes the current map extent to match that of the layer.

`weave.toc.zoomSelection`

If the layer is associated with an entity this allows the user to change the map view based on the extent of the current selection for that entity. Requires that the `entity` attribute is specified for the layer in the toc model.


```
weave.toc.zoomToMinScale
```

Change the current map extent to match the minimum scale range for the layer (if a minimum scale for the layer can be determined by Weave).

```
weave.toc.zoomToMaxScale
```

Change the current map extent to match the maximum scale range for the layer (if a maximum scale for the layer can be determined by Weave).

```
weave.toc.clearSelection
```

If the layer is associated with an entity this allows the user to clear the selection for the entity. Requires that the `entity` attribute is specified for the layer in the toc model.

```
weave.toc.lockLayer
```

If the layer is lockable this allows the user to toggle the state of the locking for the layer. A lockable layer is a layer that has the ability to disable the scale ranges that are applied to a layer's display, allowing a user to view the layer outside of its normal scale ranges. To create a lockable layer you need to duplicate the layer in the underlying map engine, rename the new layer to include `UNLOCKED` as a prefix and remove any scale ranges that are attached to the symbology for the new layer. Then when Weave attempts to display the layer in an unlocked state it will display the unlocked version of the layer rather than the original.

Note:

- You should not include the unlocked version of the layer in the ToC model.
- If the underlying layer name contains a colon (which some WMS map engines do) the `UNLOCKED` prefix should be after the colon, e.g. `topp:property` should use `topp:UNLOCKEDproperty`.

```
weave.toc.checkAll
```

Allows you to check all child entries of a toc group with a single click.

```
weave.toc.uncheckAll
```

Allows you to uncheck all child entries of a toc group with a single click.

```
weave.toc.favourite
```

Mark a layer as a favourite, which would then be displayed if the user selects the `weave.toc.favourites` action listed above. Available from 2.6.5.

```
weave.toc.filterLayer
```

Allow the user to filter the layer, note that the filter itself must be defined separately and only layers that have an associated filter will have this entry enabled. Available from 2.6.6. See [Layer Filtering](#) for more information.

Layer Transparency Slider

The `weave.toc.layerTransparency` context menu item was added at 2.5.21, this item adds a new slider to the context menu allowing you to directly change the transparency of the underlying map engine. Note that the slider is attached to individual layers in the ToC panel but it changes the transparency of the entire map engine that the layer belongs to and not just the layer associated with the ToC entry.

You can either specify the `mapEngine` and `slider` settings within a `mapEngines` and `sliders` section, as in the example or specify them directly under the item.

You can also specify `width` and `interval`, either at the action level or the slider level, to specify the width of the slider and the movement interval of the slider.

Note that the tool currently only works on a layer entry in a ToC, and will not be available for a group, even if that group is set as not expandable. This has been resolved and will be available from 2.5.23.

add slider to all map engines

```
<contextmenu>
  <item action="weave.toc.layerTransparency" />
</contextmenu>
```

add slider to specified map engines

```

<contextmenu>
  <item action="weave.toc.layerTransparency">
    <mapEngines>
      <mapEngine>mapengine.raster</mapEngine>
    </mapEngines>
  </item>
</contextmenu>

```

add slider to specified map engines and configure individual sliders

```

<contextmenu>
  <item action="weave.toc.layerTransparency">
    <sliders>
      <slider mapengine="mapengine.vector" value="70"/>
      <slider mapengine="mapengine.raster" value="30"/>
    </sliders>
  </item>
</contextmenu>

```

Client Toolbars

A toolbar is a set of actions or components that are placed relative to each other. A toolbar can be placed inside any view, inside a layout region or at the application level.

An action can be rendered as a button or a tool.

A button is activated when a user clicks on it which fires an event. This could be opening up a window to present an input form, changing the map extent or opening a url with some parameters.

A tool is used to interact with a particular view on the screen. The user click the tool and it becomes active, the user then does some work with the view and the action then fires an event. This could be clicking on the zoomin tool, drawing a box an then the map changes its extent.

A component is a rendered piece of HTML code that presents custom features that and action cannot. e.g. a combobox with a list of active entities, a list of scales, a loading indicator or a scalebar. Components are rendered using javascript and HTML.

A separator can be added to a toolbar by adding `<item>-</item>`, items can be right aligned by adding `<item>-></item>` and a toolbar can be split using `<item>|</item>` which will split the toolbar/statusbar over multiple lines.

To be valid xml the `>` in `->` should be escaped, so you should really use either `<item>-></item>` or `<item><![CDATA[->]]></item>`, but the Weave config file reader is a bit more forgiving and will allow you to use `->`, just remember that you may have problems if you try and validate the xml file.

Also, simple text can be added to a toolbar using `<item>Label</item>`.

The default tool for a map panel must be in the toolbar attached to the map view, it can't be in a toolbar defined in a layout region.

A toolbar will be placed at the top of the area that it's been added to. To move it to the bottom add a statusbar instead, they're exactly the same as a toolbar.

Namespace

urn:com.cohga.html.client#1.0

Tags

toolbar | statusbar

Properties

Name	Type	Required	Description
scale	string	no	'small', 'medium' or 'large', default is 'small'. Adjusts the size of the buttons in the toolbar/statusbar. This value can also be set at the top level in a client config to set it for every toolbar/statusbar in the client.
iconAlign	string	no	'top', 'bottom', 'left' or 'right', default is 'left'. Adjusts the alignment on the images within the buttons in the toolbar/statusbar. This value can also be set at the top level in a client config to set it for every toolbar/statusbar in the client.

Sub-tags

Name	Type	Cardinality	Description
item	Client Actions	0..*	A place holder for either an Action or Component

Content

None

Examples

A Basic toolbar definition

```
<toolbar>
  <item action="com.cohga.html.client.actions.zoomInAction"/>
  <item action="com.cohga.html.client.actions.zoomOutAction"/>
  <item>-</item>
  <item action="com.cohga.html.client.map.initialExtentAction"/>
  <item action="com.cohga.html.client.map.fullExtentAction"/>
  <item action="com.cohga.html.client.map.refreshMapAction"/>
  <item>-></item> <!-- move this off to the right -->
  <item component="com.cohga.html.client.components.
loadingComponent"/>
  <item>|</item> <!-- put the next items on their own line -->
  <item action="com.cohga.html.client.actions.reportAction"/>
  <item action="com.cohga.html.client.actions.gridAction"/>
</toolbar>
```

Adding a toolbar to a layout region

```
<client:config>
  <perspective>
    <layout>
      <west>
        <name>toc</name>
        <type>fit</type>
      <west>
        <center>
          <name>map</toolbar>
          <type>fit</type>
        <toolbar>
```

```

        <item action="com.cohga.dms.ShowSelected"/>
        <item action="com.cohga.dms.ShowAllSelected"/>
    </toolbar>
</center>
</layout>
....
</perspective>
</client:config>

```

Adding a toolbar to the application

```

<client:config>
  <toolbar>
    <item action="com.cohga.dms.ShowSelected"/>
    <item action="com.cohga.dms.ShowAllSelected"/>
  </toolbar>
  <perspective>
    <layout>
      <west>
        <name>toc</name>
        <type>fit</type>
      </west>
      <center>
        <name>map</toolbar>
        <type>fit</type>
      </center>
    </layout>
    ....
  </perspective>
</client:config>

```

Here is the sample of the select tools inside the toggle button

Creating a sub-toolbar

```

<item action="weave.toggleToolbar">
  <iconCls>icon-selectfeatures</iconCls>
  <toolbar>
    <item action="com.cohga.html.client.map.select.rect.newAction"/>
    <item action="com.cohga.html.client.map.select.rect.addAction"/>
    <item action="com.cohga.html.client.map.select.rect.removeAction"
  />
    <item action="com.cohga.html.client.map.select.rect.refineAction"
  />
    <item>-</item>
    <item action="com.cohga.html.client.map.select.circle.newAction"/>
    <item action="com.cohga.html.client.map.select.circle.addAction"/>
    <item action="com.cohga.html.client.map.select.circle.
removeAction"/>
    <item action="com.cohga.html.client.map.select.circle.

```

```

refineAction"/>
  <item>-</item>
  <item action="com.cohga.html.client.map.select.polyline.newAction"
/>
  <item action="com.cohga.html.client.map.select.polyline.addAction"
/>
  <item action="com.cohga.html.client.map.select.polyline.
removeAction"/>
  <item action="com.cohga.html.client.map.select.polyline.
refineAction"/>
  <itemitem>
  <item action="com.cohga.html.client.map.select.polygon.newAction"
/>
  <item action="com.cohga.html.client.map.select.polygon.addAction"
/>
  <item action="com.cohga.html.client.map.select.polygon.
removeAction"/>
  <item action="com.cohga.html.client.map.select.polygon.
refineAction"/>
  <itemitem>
  <item action="com.cohga.html.client.map.select.clearAction"/>
  <item action="com.cohga.html.client.map.select.clearAllAction"/>
  <itemitem>
  <item action="com.cohga.html.client.map.select.zoomAction"/>
  <itemitem>
  <item action="com.cohga.html.client.actions.bufferAction"/>
</toolbar>
</item>

```

And maybe a Navigation Group

```

<item action="weave.toggleToolbar">
  <iconCls>icon-world</iconCls>
  <align>t1-bl</align>
  <pressed>>true</pressed>
  <toolbar>
    <item action="weave.previousExtent"/>
    <item action="weave.nextExtent"/>
    <item action="weave.initialExtent"/>
    <item action="weave.fullExtent"/>
    <item action="weave.zoomIn"/>
    <item action="weave.zoomOut"/>
    <item action="weave.pan" isDefault="true"/>
    <item action="weave.refreshMap"/>
  </toolbar>
</item>

```

Note that the `align` setting sets the position of the toolbar relative to the button. The first part, `t1` in this example, is the position point in the toolbar, and the second part, `bl` in this example, is the position point in the button. So in this example the top left of the toolbar will align with the bottom left of the button.

t1:: The top left corner
t:: The center of the top edge
tr:: The top right corner

l:: The center of the left edge
 c:: In the center of the element
 r:: The center of the right edge
 bl:: The bottom left corner
 b:: he center of the bottom edge
 br:: The bottom right corner

Client Actions

Action are ways that the user interacts with the Weave client. They're setup in the client config section of the config file.

Available Actions

- [Projections](#)
- [Identify](#)
- [Spatial Identify](#)
- [Legend](#)
- [Zip'n'Ship](#)
- [Simple Report Menu](#)
- [Map Tips](#)
- [Transfer](#)
- [Redline](#)

The above list is not definitive and is in the process of being updated.

The following is a list of the actions that are currently available. Please note that the version of Weave you have installed may not have access to these actions as you may be using earlier version. Please download the current release of Weave to ensure you have the actions as listed below.

Buttons

The actions require a single click to initiate their action

Name	id	alternate id	Description
Help	com.cohga.client.actions.help	weave.help	Display the help content.
Zoom To Scale		weave.zoomToScale	Zoom to a predetermined scale. The scale to zoom to is set by a <code>scale</code> attribute.
Open Log Window	com.cohga.client.actions.logAction		Display log output in a new window.
Zip'n'Ship Envelope	com.cohga.client.actions.zipshipenvelope	weave.zipshipenvelope	Export data via a wizard.
Zip'n'Ship	com.cohga.client.actions.zipship	weave.zipship	Export data via a menu.
Simple Report	com.cohga.client.action.simplereport		Directly print a report.
Simple Report Menu	com.cohga.client.action.simplereportmenu		Provide a menu with a selection of reports that can be generated.
Refresh Map	com.cohga.html.client.map.refreshMapAction	weave.refreshMap	Refresh the current map.
Zoom to Selection	com.cohga.html.client.map.select.zoomAction	weave.selectZoom	Zoom to the currently selected entities.
Fixed Zoom Out		weave.fixedZoomOut	Zoom out by a fixed amount.
Fixed Zoom In		weave.fixedZoomIn	Zoom in by a fixed amount.
Previous Extent	com.cohga.html.client.map.previousExtentAction	weave.previousExtent	Zoom back to the previously set map extent.
Next Extent	com.cohga.html.client.map.nextExtentAction	weave.nextExtent	Zoom forward to the previously set map extent (after Previous Extent is used)
Clear Selection	com.cohga.html.client.map.select.clearAction	weave.clear	Clear the selection of the active entity.
Clear All Selections	com.cohga.html.client.map.select.clearAllAction	weave.clearAll	Clear the selections for all entities.
Zoom to Full Extent	com.cohga.html.client.map.fullExtentAction	weave.fullExtent	Zoom out the the full extent of the map.
Zoom to Initial Extent	com.cohga.html.client.map.initialExtentAction	weave.initialExtent	Zoom to the extent the client started with.

Spatial Selection	com.cohga.html.client.actions.spatialSelectAction	weave.spatialselect	Perform a range of spatial operations.
Buffer Selection	com.cohga.html.client.actions.bufferAction	weave.buffer	Select entities within a certain distance of the currently selected entities. Superseded by Spatial Selection.
Spatial Intersection	com.cohga.html.client.actions.intersect	weave.intersect	Perform a spatial intersection between two layers. Superseded by Spatial Selection.
Selection Identify	com.cohga.html.client.map.selectionIdentify	weave.selectionIdentify	Display a spatial related to entities selected via the map.
Selection Undo	com.cohga.html.client.map.select.UndoAction	weave.selection.Undo	Undo the last selection operation.
About	com.cohga.client.actions.about		Show a window with information about the current client.
Legend	com.cohga.client.action.legend		Show a map legend in a separate window.
Metadata	com.cohga.client.action.metadata		Show some map metadata in a separate window.
Get	com.cohga.selection.Get		Update the current selection from an external system. Requires additional external configuration.
Put	com.cohga.selection.Put		Sends the current selection to an external system, Requires additional external configuration.
Coordinate Zoom		weave.coordZoom	Display a window allowing the user to zoom to a specific coordinate.
Logout	com.cohga.client.actions.logoutAction		End the current user session. Not useful when using Windows integrated authentication.

Tools

These actions require interaction with the map once selected

Name	id	alternate id
Identify	com.cohga.html.client.actions.identifyAction	weave.identify
Select Spatial	com.cohga.html.client.map.select.rect.newAction	
Select Spatial	com.cohga.html.client.map.select.rect.addAction	
Select Spatial	com.cohga.html.client.map.select.rect.removeAction	
Select Spatial	com.cohga.html.client.map.select.rect.refineAction	
Select Spatial	com.cohga.html.client.map.select.circle.newAction	
Select Spatial	com.cohga.html.client.map.select.circle.addAction	
Select Spatial	com.cohga.html.client.map.select.circle.removeAction	
Select Spatial	com.cohga.html.client.map.select.circle.refineAction	
Select Spatial	com.cohga.html.client.map.select.polyline.newAction	
Select Spatial	com.cohga.html.client.map.select.polyline.addAction	
Select Spatial	com.cohga.html.client.map.select.polyline.removeAction	
Select Spatial	com.cohga.html.client.map.select.polyline.refineAction	
Select Spatial	com.cohga.html.client.map.select.polygon.newAction	
Select Spatial	com.cohga.html.client.map.select.polygon.addAction	
Select Spatial	com.cohga.html.client.map.select.polygon.removeAction	
Select Spatial	com.cohga.html.client.map.select.polygon.refineAction	
Spatial Identify	com.cohga.html.client.map.spatialIdentify	weave.spatialIdentify
Pan	com.cohga.html.client.map.panAction	weave.pan
Zoom Out	com.cohga.html.client.map.zoomoutAction	weave.zoomOut
Zoom In	com.cohga.html.client.map.zoominAction	weave.zoomIn
Measure Polygon	com.cohga.client.actions.map.measurePolygon	weave.measurePolygon

Measure Polyline	com.cohga.client.actions.map.measurePolyline	weave.measurePolyline
Pan/Select	com.cohga.html.client.map.panSelectAction	weave.panSelect
Zoom in/Select	com.cohga.html.client.map.zoomInSelectAction	weave.zoomInSelect
Zoom out/Select	com.cohga.html.client.map.zoomOutSelectAction	weave.zoomOutSelect

Redline

These actions provide interaction with the redline component of the map

Name	id	alternate id
Modify	com.cohga.client.actions.redline.modify	weave.redline.modify
Point	com.cohga.client.actions.redline.pointAction	weave.redline.point
Polyline	com.cohga.client.actions.redline.polylineAction	weave.redline.polyline
Polygon	com.cohga.client.actions.redline.polygonAction	weave.redline.polygon
Delete	com.cohga.client.actions.redline.delete	weave.redline.delete
Delete All	com.cohga.client.actions.redline.deleteall	weave.redline.deleteall
Stroke Menu	com.cohga.client.actions.redline.strokeColourMenu	weave.redline.stroke
Font Menu	com.cohga.client.actions.redline.familyMenu	weave.redline.font
Fill Menu	com.cohga.client.actions.redline.fillColourMenu	weave.redline.fill
Save	com.cohga.client.actions.redline.save	weave.redline.save
Load	com.cohga.client.actions.redline.load	weave.redline.load
Toolbar	com.cohga.client.actions.redline	weave.redline
Import geometry from selected entity	com.cohga.client.actions.redline.entity	weave.redline.entity
Export to Shapefile	com.cohga.client.actions.redline.exportMarkersAction	weave.redline.exportMarkers
Export to KML	com.cohga.client.actions.redline.export.kml	weave.redline.exportKml
Buffer currently selected entity	com.cohga.client.actions.redline.bufferAction	weave.redline.buffer
Buffer selected redline		weave.redline.bufferSelected
Buffer all redlines		weave.redline.bufferAll
Circle by Radius		weave.redline.circleByRadius

Note: not all items are enabled by default when you add a redline toolbar. To add additional tools from the list above you should add a new tool to the `com.cohga.client.actions.redline` item, e.g.

```
<item action="com.cohga.client.actions.redline">
  <tool>entity</tool>
  <tool>buffer</tool>
  <tool>bufferAll</tool>
  <tool>bufferSelected</tool>
</item>
```

The name to use for the tool is the last part of the alternate id in the table above.

The Delete, Delete All and Load items can have an optional `warning` parameter to prompt a user before deleting/loading redlines, e.g.

```
<item action="com.cohga.client.actions.redline">
  <item action="weave.redline.delete" warning="true"/>
  <item action="weave.redline.deleteall" warning="true"/>
</item>
```



```

        <item action="weave.redline.load" warning="true" />
        ...
    </item>

```

Context Menu

These items can be added to the right click context menu. Note that some of the buttons can also be added, it's just that these are specific to the context menu. Some examples are provided [here](#).

Name	id	alternate id
Quick Identify	weave.quickIdentify	
Open Url	weave.openUrlToc	

Note: Quick Identify has an optional radius parameter, which defaults to 1 (which is based on the client map units, there is currently no way to set the units explicitly).

Grid

These relate to the grid data display

Name	id	alternate id
Display Grid Window	com.cohga.html.client.actions.gridWindowAction	
Zoom To	com.cohga.html.client.grid.zoomAction	
Pan To	com.cohga.html.client.grid.panAction	
Export	com.cohga.html.client.grid.exportAction	
Refine Selection	com.cohga.html.client.grid.refine	weave.grid.refine
Remove Selected	com.cohga.html.client.grid.removeAction	weave.grid.remove
Export Grid Data	com.cohga.html.client.grid.exportGridAction	weave.grid.exportgriddata

Other

These actions have more advanced interactions

Name	id	alternate id
Simple Report Menu	com.cohga.client.action.simplereportmenu	
ToC Menu	com.cohga.client.action.toc	
Map Tips	weave.maptips	
Bookmark Menu	com.cohga.html.client.map.bookmark	weave.bookmark
Published Bookmarks Menu	com.cohga.html.client.map.publishedbookmark	weave.publishedbookmark
Projection Display	com.cohga.client.actions.projAction	weave.projections
Perspective Selector	com.cohga.client.actions.perspectiveMenu	

Document Management System

Name	id	alternate id
Show All Documents	com.cohga.dms.ShowAllSelected	
Show Selected Documents	com.cohga.dms.ShowSelected	
Upload Document	com.cohga.dms.UploadSelected	

Pathway Integration

Name	id	alternate id
------	----	--------------

Display	com.cohga.pathway.Display	
Create Link	com.cohga.pathway.CreateLink	
Display Other	com.cohga.pathway.DisplayOther	
Return	com.cohga.pathway.Return	
Send Other Query	com.cohga.pathway.SendQueryOther	
Send Query	com.cohga.pathway.SendQuery	

Asset Master Integration

Name	id	alternate id
Asset Master	com.cohga.client.actions.assetmaster	
Asset Master Menu	com.cohga.client.actions.assetMasterMenu	

Interoperability

These actions are intended to be used by external systems when controlling Weave from third party applications

Name	id	alternate id
Zoom Selection	com.cohga.ZoomSelection	
Set Selection	com.cohga.SetSelection	
Get Selection	com.cohga.GetSelection	
Set Active Entity	com.cohga.SetActiveEntity	
Begin Update	com.cohga.BeginUpdate	
End Update	com.cohga.EndUpdate	

Internal

These action don't provide any GUI but are used by other code to perform an action

Name	id	alternate id
Set Selection	weave.setSelection	
Get Selection	weave.getSelection	
Search	com.cohga.client.actions.search	

Client Actions Adhoc Query

Provides a means to search on fields in a custom manner. It provides more flexibility than the Search Panel but also requires detailed knowledge of the data and search methods so it may not be suited to all Weave users.

For a description of Adhoc search/Advanced search Weave tool, please refer [Adhoc Query / Advanced Search](#)

ID

com.cohga.html.client.actions.adhocQuery

Sub-tags

Name	type	cardinality	default	description
label	String	0..1	none	Label to display in button
windowTitle	String	0..1	"Identify"	The title of the popup window

windowWidth	Integer	0..1	610	The width of the popup window
windowHeight	Integer	0..1	400	The height of the popup window

Examples

```
<item action="weave.adhocQuery">
  <windowTitle>Advanced Search</windowTitle>
  <windowWidth>550</windowWidth>
  <windowHeight>590</windowHeight>
  <tooltip>
    <title>Advanced Search</title>
    <text>Advanced search for complex data searches</text>
  </tooltip>
</item>
```

Client Actions Bookmarks

There are three bookmark tools available:

1. [Standard Bookmark Action](#) - Provides a single customisable toolbar action to allow users to create, manage and use bookmarks
2. [Published Bookmark Action](#) - Provides a cut down menu showing just published bookmarks
3. [Bookmark Manager](#) - Provides a UI for managing shared and published bookmarks

Client Actions Bookmarks - Standard Bookmark Action

ID

weave.bookmark

Sub-tags

Name	Type	Cardinality	Default	Description
extent	boolean	0..1	true	Should the current extent be includable in the bookmark?
layers	boolean	0..1	true	Should the current ToC content be includable in the bookmark?
redlines	boolean	0..1	true	Should the current redlines be includable in the bookmark?
entity	boolean	0..1	true	Should the current active entity be includable in the bookmark?
selection	boolean	0..1	true	Should the selection for the active entity be includable in the bookmark?
categories	boolean	0..1	true	Disable the use of categories. You only need to set this attribute (to false) if you have configured the bookmarks to include categories but you want to include one of the bookmark tools in a client without the categories. If you have not configured a data definition for the categories in the bookmark config then this flag will not do anything, regardless of it being set to true or false.
groups	boolean	0..1	true	Disable the use of groups, you only need to set this attribute (to false) if you have configured the bookmarks to include groups but you want to include one of the bookmark tools in a client without the groups.

				If you have not configured a data definition for the groups in the bookmark config then this flag will not do anything, regardless of it being set to true or false.
baseUrl	URL	0..1		Overwrite base URL when generating a shared bookmark. It will use the URL used to start the client if not specified.
create	boolean	0..1	true	Should the "Create Bookmark" option be included in the menu?
publish	boolean	0..1	true - if the user is logged in false - if the user is anonymous	Should the "Publish Bookmark" option be included in the menu? Note that this also determines if the publishing functions are included in the Bookmark Manager unless the <code>publisher</code> flag is used to override it.
publisher	boolean	0..1	taken from <code>publish</code>	Should the publishing functionality be included in the Bookmark Manager?
published	boolean	0..1	true	Should the published bookmarks themselves be included in the menu?
share	boolean	0..1	true - if the user is logged in false - if the user is anonymous	Should the "Share Bookmark" option be included in the menu?
manage	boolean	0..1	true - if the user has persistent storage available false - if the user does not have persistent storage available	Should the "Manage Bookmarks" option be included in the menu?
warning	boolean	0..1	true	Should delete actions be verified?
grouped	boolean	0..1	false	Changes the way groups are displayed This pre-dates the addition of the <code>groups</code> and <code>categories</code> properties (added in 2.5.29) and switches between grouping bookmarks based on the user id or just including the user id in the text.

- Persistent storage - If the user is logged in they will have persistent storage available. The information is sent to the server for storage against their user id, and so their bookmarks will be available from any browser the user logs in from. If, however, the user is not logged in, they may still have persistent storage available if their web browser supports it, which these days is more than likely. But it means that their bookmarks will only be available if they use the same browser that they created the bookmark in.

Examples

Grouping published bookmarks

To group published bookmarks under sub-menus you need to create one or two [data definitions](#), depending upon how many levels you're after, that contain the list of groups that should be available. And then create a `bookmark:config` item to tell Weave that it should use those data definitions when handing the categories and/or groups.

Categories are the top level groupings and the groups are the second level.

The following is an example of two inline data definitions that will contain the data definition used for the bookmark samples following it. They do not have to be inline data definitions, any data definition type will work, but their use here makes it easier to follow this example.

Bookmark grouping data definitions

The following example configuration file shows an entire file containing all the parts required to setup the bookmark category and grouping configuration which will be used by any of the bookmark tools you add to the client. Note that this example is assuming that you'll be using both categories and groups, if you're only using one you only need to define one data definition and include that in the `bookmarks:config` (either the `categories` or `groups` tag).

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0"
        xmlns:bookmarks="urn:com.cohga.weave.client.
bookmarks#1.0"
        xmlns:data="urn:com.cohga.server.data.database#1.0"
```

```

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">

<bookmarks:config>
  <categories>bookmark_categories</categories>
  <groups>bookmark_groups</groups>
</bookmarks:config>

<data:datadefinition id="bookmark_categories">
  <inlinedataconnection>
    <parameter type="string" name="label" label="
Label"/>
    <row>
      <cell>Corporate</cell>
    </row>
    <row>
      <cell>Environment</cell>
    </row>
    <row>
      <cell>Front Desk</cell>
    </row>
    <row>
      <cell>Projects</cell>
    </row>
  </inlinedataconnection>
</data:datadefinition>

<data:datadefinition id="bookmark_groups">
  <inlinedataconnection>
    <parameter type="string" name="label" label="
Label"/>
    <row>
      <cell>Assets</cell>
    </row>
    <row>
      <cell>External</cell>
    </row>
    <row>
      <cell>Finance</cell>
    </row>
    <row>
      <cell>Internal</cell>
    </row>
    <row>
      <cell>Property</cell>
    </row>
    <row>
      <cell>Works</cell>
    </row>
  </inlinedataconnection>
</data:datadefinition>
</config>

```

Bookmark action examples

These examples are just snippets showing what would need to be added to a client config to include the bookmark tool in a client toolbar. The category and grouping information is provided by the configuration example above, but the tools can also be configured using the attributes defined in the table above.

Default bookmark tool (if the bookmark config contains settings for categories and/or group they will be used, if it doesn't then they won't be used)

```
<item action="weave.bookmark" />
```

Disable the use of categories in the tool (only of use if you've configured a data definition for categories in the bookmark config but don't want to use it)

```
<item action="weave.bookmark" categories="false" />
```

Disable any folders in bookmarks (only of use if you've defined data definition for categories and groups in the bookmark config but don't want to use them)

```
<item action="weave.bookmark" categories="false" groups="false" />
```

Client Actions Bookmarks - Published Bookmark Action

ID

weave.publishedbookmark

Sub-tags

Name	Type	Cardinality	Default	Description
categories	boolean	0..1	true	Set to false to disable categories, you only need to set this attribute (to false) if you have configured the bookmarks to include categories but you want to include one of the bookmark tools in a client without the categories. If you have not configured a data definition for the categories in the bookmark config then this flag will not do anything, regardless of it being set to true or false.
groups	boolean	0..1	true	Set to false to disable groups, you only need to set this attribute (to false) if you have configured the bookmarks to include groups but you want to include one of the bookmark tools in a client without the groups. If you have not configured a data definition for the groups in the bookmark config then this flag will not do anything, regardless of it being set to true or false.
grouped	boolean	0..1	false	Change the way groups are displayed This pre-dates the addition of the groups and categories properties (added in 2.5.29) and switches between grouping bookmarks based on the user id or just including the user id in the text.

The categories and groups used for this tool are configured via separate configuration items. This tool is directly configurable enough just to turn off the use of categories and groups, to configure the categories and groups themselves requires a separate configuration, an example of which can be seen on this page [Client Actions Bookmarks - Standard Bookmark Action](#)

Examples

Bookmark action examples

Default bookmark tool (categories and/or groups will be used if they're defined in the bookmark config, or they won't be used if they're not defined)

```
<item action="weave.publishedbookmark" />
```

Disable categories folder in bookmark menu (thus is only useful if you've defined the categories data definition in the bookmark config but don't want to use it for this particular instance of the tool)

```
<item action="weave.publishedbookmark" categories="false" />
```

Disable any folders in bookmarks (this is only useful if you've defined the categories and groups data definition in the bookmark config but don't want to use them for this particular instance of this tool)

```
<item action="weave.publishedbookmark" categories="false" groups="false" />
```

Client Actions Bookmarks - Bookmark Manager

ID

weave.bookmark.manage

Sub-tags

Name	Type	Cardinality	Default	Description
categories	boolean	0..1	true	Disable categories, you only need to set this attribute (to false) if you have configured the bookmarks to include categories but you want to include one of the bookmark tools in a client without the categories. If you have not configured a data definition for the categories in the bookmark config then this flag will not do anything, regardless of it being set to true or false.
groups	boolean	0..1	true	Disable groups, you only need to set this attribute (to false) if you have configured the bookmarks to include groups but you want to include one of the bookmark tools in a client without the groups. If you have not configured a data definition for the groups in the bookmark config then this flag will not do anything, regardless of it being set to true or false.

Client Actions Coordinate Zoom

The coordinate zoom function allows a user to enter a coordinate and the map will zoom to that location for a specified zoom width.

A marker will be shown at the location on the map if specified in the configuration.

ID

weave.coordZoom

Sub-tags

Name	Type	Cardinality	Default	Description
windowTitle	String	0..1	Enter Map Coordinate	Tile in the coordinate zoom panel

windowWidth	Integer	0..1	470	Width of the coordinate zoom panel
windowHeight	Integer	0..1	375	Height of the coordinate zoom panel
zoomProjLabel	String	0..1	Select the projection	Label for the combo box of projections
zoomWidthLabel	String	0..1	Select the zoom width	Label for the combo box of zoom widths
showMarker	Boolean	0..1	false	Should a marker be placed at the zoom coordinate
markerColor	String	0..1	'green'	Colour of the marker, choices are blue, brown, dark, green, orange, purple, red, silver or yellow. markerColour can also be used.

Examples

```

<item action="weave.coordZoom">
  <showMarker>true</showMarker> <!-- place a marker at the
coordinate, default is false -->
  <markerColor>red</markerColor> <!-- default is 'green' -->
  <windowTitle>Zoom to Map Coordinate</windowTitle>
  <windowWidth>500</windowWidth>
  <windowHeight>400</windowHeight>
  <title>Zoom to Map Coordinate</title>
  <outOfRangeMsg>The Location is not within the Map Extent.<
/outOfRangeMsg>
  <tooltip>
    <title>Zoom to Map Coordinate</title>
    <text>Enter a map coordinate and zoom to it.</text>
  </tooltip>
  <bodyText>
  <![CDATA[<p>Select a projection and then enter the geographic
position where you want to centre the map. Select the width of
the map that you wish to show, then press the "OK" button.
The map will zoom to the extent that you entered. The position
shown is the current map centre.]]>
  </bodyText>
  <zoomProjLabel>Select projection</zoomProjLabel>
  <zoomWidthLabel>Select the zoom width</zoomWidthLabel>
  <zoomWidths>
    <width value="1" units="km" />
    <width value="1" units="mi" />
    <width value="10" units="km" />
    <width value="10" units="mi" />
    <width value="25" units="km" />
    <width value="25" units="mi" />
  </zoomWidths>
  <zoomWidthDefault>2</zoomWidthDefault>
  <projections>
    <!-- The following are the default labels and don't
need to set here, they're just here to show that the can be set -->
    <!-- The values can also be set for each projection
tag below -->
    <lonlabel>Longitude</lonlabel>
    <latlabel>Latitude</latlabel>
    <deglabel>Degrees</deglabel>
    <minlabel>Minutes</minlabel>

```



```

        <seclabel>Seconds</seclabel>
        <xlabel>X</xlabel>
        <ylabel>Y</ylabel>
        <projection id="EPSG:4283" type="DD" label="GDA94 Lat
/Long (Decimal)"/>
        <projection id="EPSG:4283" type="DDM" label="GDA94 Lat
/Long (DM)"/>
        <projection id="EPSG:4283" type="DMS" label="GDA94 Lat
/Long (DMS)"/>
        <projection id="EPSG:28354" label="MGA Zone 54
(GDA94)"/>
        <projection id="EPSG:28355" label="MGA Zone 55
(GDA94)"/>
        <projection id="EPSG:3111" isDefault="true" label="
VicGrid94 (GDA94)">
                <xlabel>Easting</xlabel> <!-- override "X" -->
                <ylabel>Northing</ylabel> <!-- override "Y" --
>
        </projection>
        <projection id="EPSG:20254" isNTV2="true" label="AMG
Zone 54 (AGD66)"/>
        <projection id="EPSG:20255" isNTV2="true" label="AMG
Zone 55 (AGD66)"/>
    </projections>
</item>

```

Client Actions Feedback

Allows a user to send feedback to the site administrator.

Note that for this action to work the action must be added to the client configuration but also feedback and mail server configurations must be available.

ID
weave.emailFeedback
Namespace
com.cohga.server.feedback

Tags
config

Name	Type	Cardinality	Description
classification	type	1..1	A list of different types of feedback the user can provide
to	string	1..1	The email address the feedback should be sent to
emailDomain	string	0..1	Set a default From address email domain. If this value is set the From email address will be made based on the user id and this domain (unless the user is anonymous). If this value is not set, or the user is anonymous, the user will have to enter a From address when submitting feedback. Setting this value only makes sense if users must log in to access the system.
subjectPrefix	string	0..1	A prefix to include in the email subject. This value, followed by ":", will be prepended to the Subject value the user enters, allowing the submitted emails to be filtered more easily.
includeMap	boolean	0..1	Set the default state of the "Include Map" input field, default it true.

type

Name	Type	Cardinality	Description
type	string	1..n	The type of feedback the user is providing, e.g. Bug Report, Data Issue, etc.

Example feedback and email configuration

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:feedback="urn:
com.cohga.server.feedback#1.0" xmlns:mail="urn:com.cohga.server.
mail#1.0">

    <feedback:config>
        <to>feedback@cohga.com</to>
        <emailDomain>cohga.com</emailDomain>
        <subjectPre>WEAVE</subjectPre>
        <classification>
            <type>Bug Report</type>
            <type>Data Issue</type>
            <type>Feedback</type>
            <type>Help</type>
            <type>Map Request</type>
            <type>Other</type>
        </classification>
    </feedback:config>

    <mail:smtp>
        <host>mail.cohga.com</host>
        <username>weave@cohga.com</username>
        <password>ENCEAUFJCUABCFXEMFQEBFJEBZJAKFCXGME<
/ password>
    </mail:smtp>

</config>
```

Example client configuration

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.
cohga.html.client#1.0">

    <client:config id="feedback_example">
        <!-- other configuration items -->

        <toolbar>
            <item action="weave.emailFeedback"/> <!--
note there are nothing to configure here, it's done in the feedback
config -->

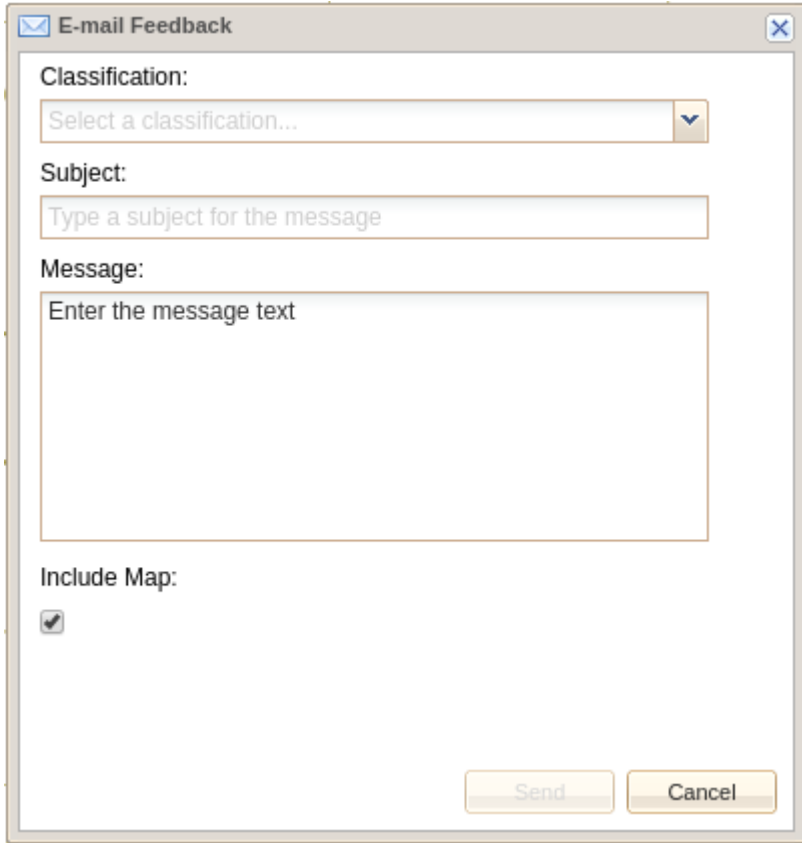
            <!-- other configuration items -->
```

```

        </toolbar>

        <!-- other configuration items -->
    </client:config>

</config>
    
```



Client Actions Export Grid

The export action for the grid panel allows the user to save data in the grid to a number of different formats. The formats supported by this tool are all provided by separate plugins. Formats could be HTML, CSV, Excel, GeoPackage etc. based on available plugins.

Check the [Export plugin](#) for more details on how to configure export formats.

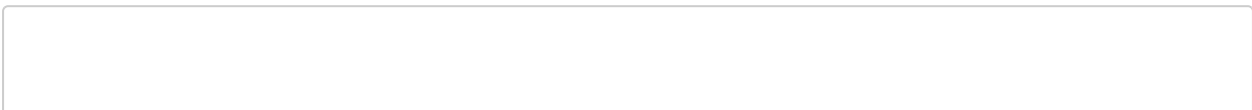
ID

com.cohga.html.client.grid.exportGridAction Or weave.grid.exportgriddata

Properties

Name	Description
text	The label to display to the user.
iconCls	The id of an icon to display for the export item.
tooltip	This will change the tooltip display and should contain title and text properties.
welcome	Text that displayed in the starting card of the wizard. Use a CDATA section if the the text content is HTML.
license	Text that displayed in the license card of the wizard. Use a CDATA section if the the text content is HTML.

Examples



```
<item action="com.cohga.html.client.grid.exportGridAction">
  <welcome><![CDATA[This is the text to be displayed on the <b>first<
  /b> wizard screen]]></welcome>
  <license><![CDATA[These are the terms and contitions that you
  <b>must</b> agree to before downloading this data.]]></license>
</item>
```

Client Actions Grid Export

The export action for the grid panel allows the user to save the attributes displayed in the grid to a number of different formats. The formats supported by this tool are all provided by separate plugins, with HTML, CSV and Excel being the currently supplied plugins.

More often than not the export actions are added to a menu that is itself included in the grid panel, rather than the button being directly embedded.

The HTML export supports an external .css file allowing for the HTML output to be prettied up a little.

This is accomplished by creating a file named 'export_html.css' saving it in the workspace directory, the content of this file will be embedded directly in the generated HTML document as a stylesheet.

The following is an example of a basic [export_html.css](#) file that will change the background colours for the table.

```
table { background-color: #beccdd; }
tr.even { background-color: #ffffff; }
tr.odd { background-color: #deecfd; }
```

ID

com.cohga.html.client.grid.exportAction

Properties

name	description
text	The label to display to the user.
iconCls	The id of an icon to display for the export item.
format	The format to export the data in, this must correspond to one of the supported formats based on the installed plugins, currently 'html', 'excel' or 'csv'.
tooltip	This will change the tooltip display and should contain title and text properties.

Examples

```
<item action="com.cohga.client.actions.menuAction">
  <text>Export</text>
  <iconCls>icon-database_go</iconCls>
  <item action="com.cohga.html.client.grid.exportAction"
  >
    <text>HTML</text>
    <format>html</format>
    <iconCls>icon-page_white_code</iconCls>
    <tooltip>
      <title>HTML</title>
      <text>Export the data to a HTML page<
  /text>
    </tooltip>
```

```

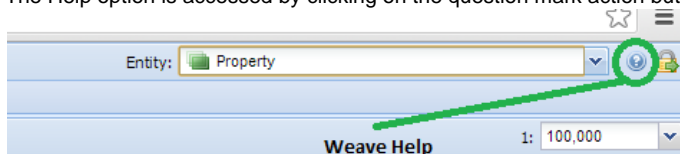
        </item>
        <item action="com.cohga.html.client.grid.exportAction"
    >
            <text>Excel</text>
            <format>excel</format>
            <iconCls>icon-page_white_excel</iconCls>
            <tooltip>
                <title>Excel</title>
                <text>Export the data to an Microsoft
Excel spreadsheet</text>
            </tooltip>
        </item>
        <item action="com.cohga.html.client.grid.exportAction"
    >
            <text>CSV</text>
            <format>csv</format>
            <iconCls>icon-page_white_wrench</iconCls>
            <tooltip>
                <title>CSV</title>
                <text>Export the data to a comma
separated text file</text>
            </tooltip>
        </item>
        <item action="com.cohga.html.client.grid.exportAction"
    >
            <text>GeoPackage</text>
            <format>gpkg</format>
            <iconCls>icon-page_white_world</iconCls>
            <tooltip>
                <title>GeoPackage</title>
                <text>Export the data to a GeoPackage
file</text>
            </tooltip>
        </item>
    </item>

```

Client Actions Help

Weave provides a default Help option that can be added to the Weave Map client. This has standard help pages but can be customised for your project.

The Help option is accessed by clicking on the question mark action button in the Map toolbar.



Prerequisites

- The Weave Help function requires two bundles which are:

```
com.cohga.client.weave.help (version 1.2.5 at the time of
writing)
```

```
com.cohga.client.weave.help.content (version 1.0.2 at the time
of writing)
```

- Customised help pages are added to the `com.cohga.client.weave.help.content` bundle as this bundle contains the help content.

Updating the config.ini file

In order for Weave to start the help bundles, an entry needs to be added to the `platform\configuration\config.ini` file for each bundle.

These two entries may already exist in `config.ini` but in case they do not, add the following lines to the end of the file and make sure that the preceding line (last line in the current file) includes a comma and backslash:

```
com.cohga.client.weave.help@:start,\
com.cohga.client.weave.help.content@:start
```

Map client configuration

Open your map client XML file and add the following code snippet to the toolbar definition. Save the file and refresh the browser.

```
<item action="weave.help">
  <startPage>resources/weave/help/html/index2.html</startPage>
  <windowTitle>Weave Help</windowTitle>
</item>
```

Note that the `startPage` value is not required (nor is the `windowTitle` value), and if not set will use the default start page. But you can change the value if you want the user to see a different page when first opening the help, for example if you add custom content and want to include your own custom start page.

Client Actions Identify

Provides a means of displaying attribute information related to entities that the user selects by clicking directly on the map (rather than via the current selection).

By default the Identify action requires the use of an [Identify view](#) to display the results of the identify operation, but it's possible, by setting a `useDialog` property, to tell the Identify action to display the results in a popup dialog box.

ID

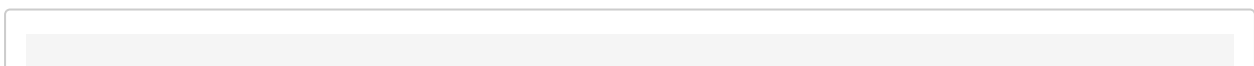
```
com.cohga.html.client.actions.identifyAction
```

Sub-tags

Name	type	cardinality	default	description
label	String	0..1	none	Label to display in button
useDialog	Boolean	0..1	true	true if the results should be displayed in a popup dialog box, false if in a Identify view
windowTitle	String	0..1	"Identify"	The title of the popup window
windowWidth	Integer	0..1	610	The width of the popup window
windowHeight	Integer	0..1	400	The height of the popup window

See [Client Views Data](#) for other i18n options and [Common Grid Panel Configuration Options](#) for other grid related options.

Examples



Basic identify tool with different window title

```
<view id='com.cohga.html.client.actions.identifyAction'>
  <label>Information</label>
  <windowTitle>Information</windowTitle>
</view>
```

Client Actions Legend

Provides a means of displaying a legend in a popup window.

See the [view](#) for details on how to configure the legend content.

ID

com.cohga.client.action.legend

Sub-tags

Name	type	cardinality	description
title	String	0..1	Text to display as the window title, default 'Legend'
width	Integer	0..1	The width of the window, default 300
height	Integer	0..1	The height of the window, default 500

Example

Dynamically generate a legend for an map engine and remove a couple of layers and add two static images on at the top and one at the bottom

```
<item action='com.cohga.client.action.legend'>
  <mapengine>mapengine.main</mapengine>
  <remove>hillshade,elevation</remove>
  <staticLegend url="http://server/images/markers.png" />
  <staticLegend url="http://server/images/aerial.png"
location='bottom' />
</item>
```

See the [view](#) for more samples.

Client Actions Map Tips

Map tips provide the user with the ability to quickly retrieve information about entities underneath the mouse cursor.

They use data definitions to supply the data, meaning that the data available to the user can come from any provider, not just the attributes attached to the underlying spatial feature.

The map tip tool can query any entity and is not dependent upon the active entity. The user chooses from the map tip menu what data they're interested in and when they move the mouse around the map that's the data they will be shown.

ID

weave.maptips

Properties

Name	Type	Required	Default	Description
minScale	number	no	0	The default minimum scale below which no map tips will be shown to the user.

	er			
maxScale	number	no	infinity	The default maximum scale above which no map tips will be shown to the user.
radius	number	no	3	The default radius, in map units, of the area to be searched, centred on the mouse location.
pressed	boolean	no	false	This sets the initial state of the map tips. Setting it to 'true' will turn on map tips at startup.
multipleMaximum	number	no	1	If the search returns more than this many results then <code>multipleText</code> will be shown to the user instead of the results. If these many or fewer results are available then they're all shown.
multipleText	string	no	There are too many features at that location	The text to show to the user if more than <code>multipleMaximum</code> results are available.
tip	tip	yes		A tip tag must be created for each set of data that's to be made available to the user for display in a map tip.
autoSize	boolean	no	false	true/false indicating if the popup window should try and adjust its size.
minSize	size	no	105,10	The minimum width and height in pixels that the popup window should use when auto-sizing.
maxSize	size	no	1200,660	The maximum width and height in pixels that the popup window should use when auto-sizing.
size	size	no	200,200	The default width and height in pixels that the popup windows should use.
hideDelay	number	no	2000	The number of milliseconds to wait before hiding the popup window.
selectedOnly	boolean	no	false	Restrict the map tip to the selected features only.
selectedOnlyMenu	boolean	no	false	Add a menu to the map tips tool to allow the user to turn on or off the <code>selectedOnly</code> flag.
visibleOnly	boolean	no	false	Restrict the map tip to display only if the associated layer is turned on in the ToC. Only available since 2.5.26
visibleOnlyMenu	boolean	no	false	Add a menu to the map tips tool to allow the user to turn on or off the <code>visibleOnly</code> flag. Only available since 2.5.26
showGeometry	boolean	no	true	Should the geometry of the related entity be drawn. Only available since 2.5.26

Sub-Tags

tip

Name	Type	Required	Description
minScale	number	no	The minimum scale below which no map tips will be shown to the user.
maxScale	number	no	The maximum scale above which no map tips will be shown to the user.
radius	number	no	The radius, in map units, of the area to be searched, centred on the mouse location.
isDefault	boolean	no	Should be set to 'true' in the tip that should be enabled by default.
data	Data	yes, if <code>datadefinition</code> is not set	The id of a Data configuration that provides the data for this particular tip.
datadefinition	Data Definition	yes, if <code>data</code> is not set	The id of a Data Definition configuration that provides the data for this particular tip.
entity	Entity	yes, if <code>datadefinition</code> is set	The id of the Entity that the Data Definition should generate the data for.

label	string	yes, if <code>datadefinition</code> is set no, if <code>data</code> is set	The label to display as this tip in the menu. Will override the label provided by the <code>Data</code> if <code>data</code> is specified rather than <code>datadefinition</code> .
autoSize	boolean	no	Indicating if the popup window should try and adjust its size
minSize	size	no	The minimum width and height in pixels that the popup window should use when auto-sizing.
maxSize	size	no	The maximum width and height in pixels that the popup window should use when auto-sizing.
size	size	no	The default width and height in pixels that the popup windows should use
selectedOnly	boolean	no	Restrict the map tip to the selected features only.
visibleOnly	boolean	no	Restrict the map tip to display only if the associated layer is turned on in the ToC.
group	string	no	A map tip with a group set will be placed under a sub-menu where the text of the sub-menu item will be taken from the group value, if a group isn't specified then the map tip will appear directly in the menu. Only available since 2.5.26
template	string	no	Used to apply HTML formatting to the tip for display in the popup.

size

Name	Type	Required	Description
width	number	yes	The width of the "size", in pixels
height	number	yes	The height of the "size" in pixels

Notes

- Only one of `data` or `datadefinition` should be set for a tip, but one of them must be set.
- The default values for tips are taken from the matching attribute in the parent tag.

Examples

```
<item action="weave.maptips">
  <tip>
    <data>property_details</data>
    <isDefault>true</isDefault>
  </tip>
  <tip>
    <data>road_details</data>
  </tip>
</item>
```

```
<item action="weave.maptips">
  <multipleMaximum>2</multipleMaximum>
  <multipleText>Too many features</multipleText>
  <minScale>1000</minScale>
  <maxScale>50000</maxScale>
  <radius>3</radius>
  <tip>
    <data>property_details</data>
    <isDefault>true</isDefault>
  </tip>
  <tip>
```

```

        <data>road_details</data>
    </tip>
</item>

```

```

<item action="weave.maptips">
    <pressed>true</pressed>
    <tip>
        <label>Property</label>
        <entity>property</entity>
        <datadefinition>property_details</datadefinition>
        <isDefault>true</isDefault>
        <minScale>1000</minScale>
        <maxScale>10000</maxScale>
    </tip>
    <tip>
        <datadefinition>road_details</datadefinition>
        <entity>roads</entity>
        <label>Road</label>
        <radius>5</radius>
        <minScale>250</minScale>
        <maxScale>1000</maxScale>
    </tip>
</item>

```

Adjusting the size of the popup

```

<item action="weave.maptips">
    <autoSize>true</autoSize>
    <minSize width="100" height="100"/>
    <maxSize width="200" height="400"/>
    <size width="150" height="150"/>
    <tips>
        <tip label="Property" entity="property"
datadefinition="property_details" isDefault="true"/>
        <tip label="Image" entity="property" datadefinition="
property_images">
            <maxSize width="800" height="800"/>
        </tip>
    </tips>
</item>

```

Formatting the text in the popup

```

<item action="weave.maptips">
    <tip>
        <data>property_details</data>
        <template><![CDATA[<div><h1><i>ADDRESS DETAILS:</i><

```

```

/h1>{address_num} {street_name} {street_type}<br>{suburb}</div>]]<
/template>
</tip>
</item>

```

Map tips with groups

```

<item action="weave.maptips">
  <tip>
    <data>property_details</data>
    <isDefault>true</isDefault>
    <group>Property</group>
  </tip>
  <tip>
    <data>property_summary</data>
    <group>Property</group>
  </tip>
  <tip>
    <data>property_owner</data>
    <group>Property</group>
  </tip>
  <tip>
    <data>road_details</data>
  </tip>
  <tip>
    <data>serwer_details</data>
    <group>Sewer</group>
  </tip>
  <tip>
    <data>sewer_summary</data>
    <group>Sewer</group>
  </tip>
</item>

```

The above example will result in a menu like the following

- Property
 - Property Details
 - Property Summary
 - Property Owner
- Road Details
- Sewer
 - Sewer Details
 - Sewer Summary

Client Actions Measure Polygon

ID

com.cohga.client.actions.map.measurePolygon

weave.measurePolygon

Properties

Name	Type	Cardinality	Default	Description	i18n resource
windowWid	integ	0..1	250	The width of the popup window	

th	er				
windowHeight	integer	0..1	100	The height of the popup window	
windowIconCls	string	0..1	icon-measure_polygon	The icon of the popup window	
windowTitle	string	0..1	Measure Polygon	The title of the popup window	weave.measurePolygon.window.title
iconCls	string	0..1	icon-measure_polygon	The icon of the button	
tooltipTitle	string	0..1	Measure Polygon	The title for the tooltip	weave.measurePolygon.tooltip.title
tooltipTitle	string	0..1	Measure a polygon on the map	The text for the tooltip	weave.measurePolygon.tooltip.text
	string		Segment Length	The "segment length" text	weave.measurePolygon.segmentLength
	string		Bearing	The "bearing" text	weave.measurePolygon.bearing
	string		Total Area (Approx)	The "total area" text	weave.measurePolygon.totalArea
hideSegment	boolean	0..1	false	Hide the segment value	
hideBearing	boolean	0..1	false	Hide the bearing value	
hideTotal	boolean	0..1	false	Hide the total value	
crs	string	0..1		Change the CRS to use when calculating the area	
digits	integer	0..1	3	Sets the number of digits to display after the decimal separator. Setting '0' drops the decimal separator and displays whole numbers only.	
minHa	integer	0..1	10000	The size of the polygon area where it will switch to displaying hectares	
minSqKm	integer	0..1	1000000	The size of the polygon area where it will switch to displaying square km	
minKm	integer	0..1	10000	The length of the polygon boundary where it will switch to displaying km	

Client Actions Measure Polyline

ID

com.cohga.client.actions.map.measurePolyline
weave.measurePolyline

Properties

Name	Type	Cardinality	Default	Description	i18n resource
windowWidth	integer	0..1	250	The width of the popup window	
windowHeight	integer	0..1	100	The height of the popup window	
windowIconCls	string	0..1	icon-measure_polyline	The icon of the popup window	
windowTitle	string	0..1	Measure Polyline	The title of the popup window	weave.measurePolyline.window.title
iconCls	string	0..1	icon-measure_polyline	The icon of the button	
tooltipTitle	string	0..1	Measure Polyline	The title of the tooltip	weave.measurePolyline.tooltip.title
tooltipText	string	0..1	Measure a polyline on the map	The text of the tooltip	weave.measurePolyline.tooltip.text
	string		Segment Length	The "segment length" text	weave.measurePolyline.

					segmentLength
	string		Bearing	The "bearing" text	weave.measurePolyline.bearing
	string		Total Length	The "total length" text	weave.measurePolyline.totalLength
hideSegment	boolean	0..1	false	Hide the segment value	
hideBearing	boolean	0..1	false	Hide the bearing value	
hideTotal	boolean	0..1	false	Hide the total value	
crs	string	0..1		Change the CRS to use when calculating the length	
digits	integer	0..1	3	Set the number of digits to display after the decimal separator. Setting '0' drops the decimal separator and displays whole numbers only.	
minKm	integer	0..1	10000	The length of the line where it will switch to displaying km. This applies to the current segment length as well as the total length.	

Client Actions Projections

Provides a means of displaying the current cursor position in different projections.

Note that you need to know the Proj4 definition of the alternate projections you wish to display.

ID

com.cohga.client.actions.projAction

Sub-tags

Name	type	cardinality	description
label	String	0..1	Label to display in tab
projection	String	1..n	The definitions and labels of the projections to display

Example

```
<item action="com.cohga.client.actions.projAction">
  <projection id="EPSG:28354" label="GDA94/MGA Zone 54"/>
  <projection id="EPSG:28355" label="GDA94/MGA Zone 55"/>
  <projection id="EPSG:28356" label="GDA94/MGA Zone 56"/>
  <projection id="EPSG:20254" label="AGD66/AMG Zone 54"/>
  <projection id="EPSG:20255" label="AGD66/AMG Zone 55"/>
  <projection id="EPSG:20256" label="AGD66/AMG Zone 56"/>
```

<!--

The following three entries show examples of how a custom transform can be specified.

This does NOT have to be done if you want to use any of these three projections, it's just done here to show you how you could do it.

-->

```
  <projection id="EPSG:4283" label="GDA94">
    <def>+proj=longlat +ellps=GRS80 +towgs84=0,
0,0,0,0,0,0 +no_defs</def>
  </projection>
  <projection id="EPSG:3111" label="VicGrid94">
    <def>+proj=lcc +lat_1=-36 +lat_2=-38 +lat_0=-37
+lon_0=145 +x_0=2500000 +y_0=2500000 ellps=GRS80 +towgs84=0,
```

```

0,0,0,0,0,0 +units=m +no_defs</def>
  </projection>
  <projection id="EPSG:3112" label="GDA94/Geoscience Australia
Lamberts" />
      <def>+proj=lcc +lat_1=-18 +lat_2=-36 +lat_0=0
+lon_0=134 +x_0=0 +y_0=0 +ellps=GRS80 +towgs84=0,0,0,0,0,0 +units=m
+no_defs</def>
  </projection>
</item>

```

Client Actions Redline

ID

com.cohga.client.actions.redline

weave.redline

Properties

Name	Type	Cardinality	Default	Description	i18n resource
tooltip\title	string	0..1	Redline	The title for the tooltip.	redline.tooltip.title
tooltip\text	string	0..1	Add elements to the map	The text for the tooltip.	redline.tooltip.text
title	string	0..1	Redline	The title of the redline toolbar window.	redline.title
hideOnClick	boolean	0..1	false	Should menus automatically hide when an item within is clicked?	
scale	string	0..1	'small'	The size of the icons in the toolbar, 'small', 'medium' or 'large'.	
geometryTool	string	0..n		A list of the geometry tools that should be included in the redline toolbar. point, polyline, polygon, circle, square, rectangle, marker and text	
settingTool	string	0..n		A list of the settings menus that should be included in the redline toolbar. fill, stroke, font, markerMenu	
tool	string	0..n		A list of additional tools that should be included in the redline toolbar.	
multilineLabel	boolean	0..1	false	Should the label field be a multi-line input field.	

Available tools

Id	Name	Enabled ¹	Description
circle	Circle	yes	Add a circle to the map
circleByRadius	Circle by Radius		Define a circle for a specified radius
defaults	Save Settings	yes ²	Save the current settings as the defaults
delete	Delete	yes	Delete the currently selected redline element
deleteall	Delete All	yes	Delete all redline elements
enitty	Get		Create a redline item from the currently selected entity
fill	Fill Settings	yes	Change the fill colour and transparency of the selected feature
font	Font Settings	yes	Change the Font colour and style of the selected features text
load	Load	yes ²	Load and manage your saved redlines
marker	Marker	yes	Add a marker to the map
markerMenu	Marker Settings	yes	Change the marker colour
modify	Modify	yes	Click on the redline to modify
point	Point	yes	Add a point to the map

polygon	Polygon	yes	Add a polygon to the map
polyline	Polyline	yes	Add a polyline to the map
rectangle	Rectangle	yes	Add a rectangle to the map
label	Label	yes	Add a label to a redline element
save	Save	yes ²	Save the current set of redlines
snap	Snap Settings	yes	Alter the current snapping settings
square	Square	yes	Add a square to the map
stroke	Line Settings	yes	Change the line colour and transparency of the selected feature
text	Text	yes	Add text to the map
buffer	Buffer		Generate a buffer around the selected features
bufferAll	Buffer All Redlines		Generate a buffer around all redline objects
bufferSelected	Buffer Selected Redline		Generate a buffer around the selected redline object
exportMarkers	Export to Shapefile		Export all redlines to Shapefile
exportKml	Export to KML		Export all redlines to KML
coordsCreate ³	Create Geometry		Create some geometry by entering the coordinates.
coordsUpdate ³	Update Geometry		Update some geometry by editing the coordinates.
upload ³	Upload Geometry		Create some geometry by uploading it from a file.
spatialSelect ⁴	Spatial Select		Select entities from redline geometry
manage ^{5,6}	Organise	yes	List and organise redlines

1. This tool is enabled by default, if an entry is not marked as enabled it will need to be added with a `tool` entry in the redline items configuration (see the Properties section above)
2. Only if storage is available for the user
3. Weave 2.6.4 or later. Requires [spatial upload extension](#) to be installed, which is done automatically as part of a 2.6.4 update.
4. Weave 2.6.6 or later
5. Weave 2.6.7 or later
6. This tool does not currently affect text, which is always displayed on top of all geometries.

Customise available markers

You can replace the content of the marker menu to change the available markers, beyond the default ref, green and blue ones.

Changing available markers

```
<item action="weave.redline">
  <!--
    rest of config goes here
  -->
  <!-- this replaces the configuration of the maker menu and
the markers that it supports -->
  <item action="weave.redline.markerMenu">
    <!-- these are the defaults -->
    <!-- you can also use brown, dark, orange, purple,
silver and yellow -->
    <marker graphic="resources/images/markers/w-marker-
green.png" text="Green" />
    <marker graphic="resources/images/markers/w-marker-
red.png" text="Red" />
    <marker graphic="resources/images/markers/w-marker-
blue.png" text="Blue" />

    <!-- these are extras and the graphics, sizeX, sizeY,
offsetX and offsetY are dependent upon the image -->
```

```

        <!-- this is assuming there is bundle installed in
Weave that is exposing these images -->
        <!-- Note these icons currently will only display on
the client, not in reports, etc -->
        <marker graphic="custom/markers/marker1.png" sizeX="
18" sizeY="18" offsetX="-9" offsetY="-9" text="Marker 1" />
        <marker graphic="custom/markers/marker2.png" sizeX="
18" sizeY="18" offsetX="-9" offsetY="-9" text="Marker 2" />
        <marker graphic="custom/markers/marker3.png" sizeX="
18" sizeY="18" offsetX="-9" offsetY="-9" text="Marker 3" />

        <!-- this is assuming example.com is serving up icons
(via http and https) -->
        <!-- Note these icons currently will only display on
the client, not in reports, etc -->
        <marker graphic="//example.com/custom/markers/marker4.
png" sizeX="18" sizeY="18" offsetX="-9" offsetY="-9" text="Marker 4"
/>
        <marker graphic="//example.com/custom/markers/marker5.
png" sizeX="18" sizeY="18" offsetX="-9" offsetY="-9" text="Marker 5"
/>
        <marker graphic="//example.com/custom/markers/marker6.
png" sizeX="18" sizeY="18" offsetX="-9" offsetY="-9" text="Marker 6"
/>
    </item>
</item>

```

Client Actions Redline - Export to Shapefile

The `com.cohga.client.redline.export` bundle allows you to export the current set of redlines to a shapefile.

To add the export to shapefile tool to the redline toolbar, assuming you have the above bundle installed and started, you have to add a new `tool` tag to the redline item in the client configuration, e.g.

Example of adding export markers tool to redline toolbar

```

<item action="com.cohga.client.actions.redline">
    <item action="weave.redline.exportMarkers" defaultProj="EPSG:
28355">
        <projections>
            <projection id="EPSG:4326" label="WGS 84"/>
            <projection id="EPSG:4283" label="Geographic
GDA 1994"/>
            <projection id="EPSG:28355" label="GDA 1994
MGA Zone 55"/>
            <projection id="EPSG:3111" label="VicGrid
1994 GDA 1994"/>
        </projections>
        <description>Some extra text for the file<
/description>
    </item>
</item>

```


The above example shows a couple of extra configuration items available for the export tool, `projections` and `description`.

Client Actions Selection Identify

Provides a means of displaying attribute information spatially related to the currently selected entities.

This action operates similarly to the [Spatial Identify](#) tool, but rather than using geometry that the user creates by hand it uses the geometry of the currently selected entities.

ID

```
com.cohga.html.client.map.selectionIdentify
```

```
weave.selectionIdentify
```

Note: there is also a `selectionIdentify2` version of this tool that fetches data from the server in batches, 4 entities at a time, which can start returning data to the user quicker than this original version. This version fetches all the data at once which can take a long time to display anything to the user.

Sub-tags

Name	Type	Cardinality	Default	Description	i18n Resource
label	String	0..1		Label to display as button text	
entity	entity	0..n		Optional list of entities to refine/sort the entity list	
defaultEntity	String	0..1		The id of the entity to have displayed first	
windowTitle	String	0..1	Selection Identify	The title of the popup window	weave.selectionIdentify.window.title
windowWidth	Integer	0..1	800	The width of the popup window	
windowHeight	Integer	0..1	500	The height of the popup window	
useVisibleEntities	Boolean	0..1	false	Use just the entities that are visible (based on ToC)	
tooltipTitle	String	0..1	Selection Identify	The title for the tooltip	weave.selectionIdentify.tooltip.title
tooltipText	String	0..1	Identify features that intersect the current selection	The text for the tooltip	weave.selectionIdentify.tooltip.text
bufferDistance	Number	0..1	0	Distance to buffer around selection before doing intersection operation	
bufferUnits	String	0..1	m	Units for the <code>bufferDistance</code> value	
bufferCrs	String	0..1	CRS of map	Coordinate reference system to use when performing the buffer generation	
spatialOperation	String	0..1	intersect	What type of spatial operation should be used INTERSECT, OVERLAPS, CONTAINS or WITHIN	
includeActive	Boolean	0..1	false	Should the source entity type also be returned in the results?	
ignoreScale	Boolean	0..1	false	Should the current scale be ignored when determining which entities to check?	
entityListWidth	Integer	0..1	200	The initial width of the entity list. Only available from 2.5.28.	
entityListWidthMin	Integer	0..1	175	The minimum width of the entity list. Only available from 2.5.28.	
entityListWidthMax	Integer	0..1	400	The maximum width of the entity list. Only available from 2.5.28.	
include				List of entities that the tool should be enabled for. Only available from 2.6.4	
exclude				List of entities that the tool should not be enable for. Only available from 2.6.4	

Notes:

Only one of include or exclude should be used, and they can be specified in multiple ways:

1. As a comma separated value - quick and easy when specifying entities by hand.
 - a. `<include>property, reserve</include>`
 - b. `<exclude>lga, suburb</exclude>`
2. As individual entries - useful when setting based on user attributes
 - a. `<include>property</include><include>reserve</include>`
 - b. `<exclude>lga</exclude><exclude>suburb</exclude>`
3. As a sub-list - useful when using a snippet
 - a. `<include><entity>property</entity><entity>reserve</entity></include>`
 - b. `<exclude><entity>lga</entity><entity>suburb</entity></exclude>`

Entity

Name	Type	Cardinality	Description
id	string	1..1	The unique entity id for this entity
minScale	number	0..1	Min zoom scale for this entity to be reported for
maxScale	number	0..1	Max zoom scale for this entity to be reported for
isDefault	boolean	0..1	Set this entity as the default (alternative to defaultEntity above)
data	string	0..n	A list of the data definition that should be available for the entity

See [Common grid panel configuration options](#) for other grid related options.

Example of tools configuration

```

                    <item action="weave.
selectionIdentify" useVisibleEntities="true" ignoreScale="true">
                    <!-- disable the
tools if lga or suburbs is the active entity -->
                    <exclude>
                                <entity>lga<
/entropy>
                    <entity>suburbs</entity>
                                </exclude>
                    <!--
be specified as
                                <exclude>lga,
suburb</exclude>
                                -->
                    <!-- which entities
should be interrogate -->
                    <entities>
                                <entity id="
property" maxScale="12000">
                    <data>property_details</data>
                    <data>property_spatial</data>
                    <data>property_owners</data>
                                </entity>
                    <entity>roads<

```

```

/entity>

<entity>suburbs</entity>

<entity>council_buildings</entity>
                                </entities>
</item>

```

Client Actions Selection Undo

Allows the user to undo the previous change to the current selection. Note that it only goes back one selection, so using the tool multiple times will only alternate between two selection sets.

ID

weave.selection.Undo

Properties

Name	Type	Default	Description
zoom	boolean	false	Should the map zoom to the selection?
active	boolean	false	Should the entity be set as the active entity?
turnon	boolean	false	Should the corresponding layer, if there is one, be turned on in the ToC?

Examples

```
<item action="weave.selection.Undo" zoom="true" turnon="true"/>
```

Client Actions Simple Report Menu

Provides a means of displaying a menu with reports that the user can quickly generate.

Note that for BIRT reports the report menu uses existing report configuration items to populate the menu, it does not point directly to the BIRT report definition files.

ID

com.cohga.client.action.simplereportmenu

Sub-tags

Name	type	cardinality	description
label	String	0..1	Label to display in button
format	String	0..1	The format to generate the report in (can also be set per-report)
reports	report list	1..1	The list of reports to include in the menu

Example

```

<item action="com.cohga.client.action.simplereportmenu">
  <label>Quick Reports</label>
  <tooltip>
    <title>Quick Report</title>
    <text>Generate a preset report</text>
  </tooltip>

```

```

        <!-- Set a default format, the user can't choose one
if you do this -->
        <format>pdf</format>

        <reports>
            <report>
                <report>map_portrait_a4</report>
            </report>

            <!-- Overwrite the report label -->
            <report>
                <label>A4 Landscape Map Report</label>
                <report>map_landscape_a4</report>
            </report>

            <!-- Only enable the entry for a particular
entity -->

            <report>
                <entity>property</entity>
                <label>Property Detail</label>
                <report>property_details</report>
            </report>

            <!-- set the report parameter rather than the
user having to enter it -->
            <report>
                <entity>property</entity>
                <label>Property Detail - 1:5000<
/label>
                <report>property_details</report>
                <!-- assumes the report has a
parameter called 'scale' that the user usually inputs -->
                <parameters>
                    <scale>5000</scale>
                </parameters>
            </report>
        </reports>
    </item>

```

Client Actions Spatial Identify

Provides a means of displaying attribute information relating to all of the entities that a user selects when clicking on the map.

Compared to the standard Identify action the Spatial Identify doesn't use the active entity as the source for the entities to display information about, but instead searches through every entity to find any that intersect the shape drawn by the user. Which means of course that it can take longer to display its information than the regular Identify action.

The Spatial Identify action always displays its information in a popup window.

ID

com.cohga.html.client.map.spatialIdentify

Sub-tags

Name	Type	Cardinality	Default	Description

label	String	0..1		Label to display in button, default 'Spatial Identify'
defaultEntity	String	0..1		The id of the entity to be displayed the first time the tool is used. This is more of a suggestion, if there are no entities of that type where the user clicks then a different entity will be the one displayed on first use.
entity	entity	0..n		Optional list of entities to refine/sort the entity list
useVisibleEntities	Boolean	0..1	false	As of 2.5.22 rather than specify the entities that are to be used, the tool can look up the visible entities configured in the toc. You must add the entity attribute to each layer in the toc for this function to work.
windowTitle	String	0..1	"Spatial Identify"	The title of the popup window
windowWidth	Integer	0..1	800	The width of the popup window
windowHeight	Integer	0..1	500	The height of the popup window
highlight	highlight	0..1		Styling to apply to the highlighted entity
entityListWidth	Integer	0..1	220	the initial width of the entity list
entityListWidthMin	Integer	0..1	175	the minimum width of the entity list
entityListWidthMax	Integer	0..1	400	the maximum width of the entity list
timeout	integer	0..1	30000	number of milliseconds to timeout the request

Entity

Name	Type	Required	Description
id	string	yes	The unique entity id for this entity
minScale	number	no	Min zoom scale for this entity to be reported for
maxScale	number	no	Max zoom scale for this entity to be reported for
isDefault	boolean	no	Set this entity as the default

Highlight

Name	Type	Description
vector	vector	The styling for polygon and line based entities
marker	marker	The styling for point based entities

Vector

Name	Type	Description
strokeColor	string	The colour for the outline
strokeOpacity	float	The opacity for the outline
strokeWidth	integer	The width of the outline
fillColor	string	The colour of the fill
fillOpacity	float	The opacity of the fill

Marker

Name	Type	Description
icon	string	A URL pointing to the icon to use
width	int	The width of the icon
height	int	The height of the icon

offsetx	int	The x offset for the hotspot within the icon
offsety	int	The y offset for the hotspot within the icon

See [Client Views Data](#) for other i18n options and [Common grid panel configuration options](#) for other grid related options

Example

Default with labelled button

```
<item action="com.cohga.html.client.map.spatialIdentify">
  <label>Identify</label>
</item>
```

Change the default entity

```
<item action="com.cohga.html.client.map.spatialIdentify"
defaultEntity="roads">
  <label>Identify</label>
</item>
```

Refine and alter the entities

```
<item action="com.cohga.html.client.map.spatialIdentify">
  <entity id="property" maxScale="12000" />
  <entity id="road" isDefault="true" />
  <entity id="suburb" minScale="25000" />
</item>
```

Note that prior to Weave 2.5.18 there is a bug that stopped the following example from working, and from 2.5.18 on-ward the `highlight` section can be added to the `map view` anyway which allows it to be set once and applied to all the tools that highlight an entity so it doesn't need to be set for each tool individually.

Changing the highlight styles

```
<item action="com.cohga.html.client.map.spatialIdentify">
  <highlight>
    <marker>
      <icon>resources/images/markers/w-marker-red.
png</icon>
      <width>24</width>
      <height>30</height>
      <offsetx>-8</offsetx>
      <offsety>-28</offsety>
    </marker>
    <vector>
      <strokeWidth>3</strokeWidth>
```

```

        <strokeColor xsd:type="xsd:string">#d27316<
/strokeColor>
        <strokeOpacity>0.75</strokeOpacity>
        <fillColor>red</fillColor>
        <fillOpacity>0.25</fillOpacity>
    </vector>
</highlight>
</item>

```

You can also change the tool which is used when identifying features on the map. By default the tool used is a circle. Valid options are

- DragCircle (default)
- DragRectangle
- Polyline
- Polygon
- Point

e.g. to change the tool to a DragRectangle

Changing the map action

```

<item action="com.cohga.html.client.map.spatialIdentify">
    <mapAction>DragRectangle</mapAction>
</item>

```

As of 2.5.22 you can let the toc drive the layers that will be interrogated when using the spatial identify tool.

Changing the map action

```

<item action="com.cohga.html.client.map.spatialIdentify">
    <useVisibleEntities>true</useVisibleEntities>
</item>

```

Client Actions Toc Menu

The toc menu action allows the user to alter the layer display via a menu.

Minimum required configuration for the action

```

<item action='com.cohga.client.action.toc' toc='toc.main' />

```

The action will present a menu representing the hierarchical structure of the toc model.

There's also an option to embed the structure for the menu within the configuration itself, but this has limitations compared to the other version, since it doesn't support turning on/off groups.

Alternate configuration where content is embedded in configuration

```

<item action="com.cohga.client.action.toc">
    <mapengine>mapengine.vector.ags</mapengine>
    <entry label="Water" checked="true">
        <entry layer="Drainage" label="Drainage" checked="

```

```

false"/>
    <entry label="Pipes" checked="true">
    <entry layer="Water Pipes" checked="false"/>
        <entry layer="Sewer Mains" label="Sewer
Mains" checked="false"/>
    </entry>
    <entry label="Streams" checked="false">
        <entry layer="Streams" label="Hydro"/>
        <entry layer="Main Streams" label="Main Hydro"
/>
    </entry>
</entry>
<entry layer="Property" label="Property" checked="true"/>
</item>

```

Client Actions Transfer

These actions provide support to allow selections to be transferred between Weave and other systems.

This bundle provides the server side API and the client side components to use it, as well as a database transfer implementation.

The included database transfer implementation support transferring selection via database tables.

The database transfer implementation will write, at least, a userid and key value to an administrator specified table, where, by default, the userid column will be named 'userid' but the key column must be specified in the config.

The configuration requires at least a 'datasource', 'table' and 'column'. 'usercolumn' can be used to change the userid column name.

id's can be altered when being send/retrieved from the other system by using the 'get' and 'put' tags, which take one of 'lpad', 'rpad' or 'trim' setting to alter their behaviour.

- trim='true' will strip leading/trailing spaced from the keys.
- lpad=[width] will left pad the id's with [width] spaces.
- rpad=[width] will right pad the id's with [width] spaces.

The 'get' option is performed after retrieving the id's from the other system, and the 'put' option is performed before sending the id's to the other system.

The database transfer implementation also requires a list of the entities that it can transfer, which can optionally specify a different column than the default key by setting the 'key' attribute.

Also, it can specify a list of additional values that can should be sent to the other system along with the userid and key value by adding a 'value' tag with the name of the column and the value to set.

You can also specify when the transfer table should be cleared, by setting the 'cleanwhen' value to 'before', 'after' or 'both' which correspond to clearing the table before sending to the other system, after getting from the other system or both. The default is never.

Additionally, all of the settings can be set at the top level, where they'll provide defaults, or can be set within each entity where they'll overwrite the defaults.

The Get and Put actions require a 'system' property to tell the button what system to transfer the selection to/from, the system property should be set to the id of a transfer service that's previously been configured (currently only a database transfer service is available).

The GetMenu and PutMenu action create a menu of all of the systems available, and therefore don't require the system property to be set for the item in the client configuration. Obviously GetMenu and PutMenu are only useful if you have more than one system registered and want to save screen real estate by not having multiple buttons.

The database transfer service can be registered multiple times with different configurations, for example if you wanted to setup a transfer between a Pathway test and production servers then you can register two externaldb services with different datasources pointing to the different databases.

Then the Get and Put actions can be added multiple times, maybe with an ACL attached, that use the different id's for the test and production Pathways.

ID

```

com.cohga.selection.Get
com.cohga.selection.Put
com.cohga.selection.GetMenu
com.cohga.selection.PutMenu

```

Sub-tags

Name	type	cardinality	description
		1..1	

system	String	The id of the provider to use
--------	--------	-------------------------------

Example

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0"
  xmlns:externaldb="urn:com.cohga.selection.transfer.db#1.0"
  xmlns:client="urn:com.cohga.html.client#1.0">

  <externaldb:transfer id="pathway">
    <name>Pathway</name>
    <datasource>datasource.pathway</datasource>
    <table>PTH_GIST</table>
    <column>GISREF</column>
    <put lpad="15"/>
    <get trim="true"/>
    <clearwhen>both</clearwhen>
    <entity id="property">
      <key>PRUPIX</key>
    </entity>
  </externaldb:transfer>

  <externaldb:transfer id="pathway.test">
    <name>Pathway</name>
    <datasource>datasource.pathway.test</datasource>
    <table>PTH_GIST</table>
    <column>GISREF</column>
    <put lpad="15"/>
    <get trim="true"/>
    <clearwhen>both</clearwhen>
    <entity id="property">
      <key>PRUPIX</key>
    </entity>
  </externaldb:transfer>

  <externaldb:transfer id="hansen">
    <acl>acl.hansen</acl>
    <name>Hansen</name>
    <datasource>datasource.hansen</datasource>
    <table>HANSEN</table>
    <column>COMPKEY</column>
    <key>COMPKEY</key>
    <clearwhen>before</clearwhen>
    <entity id="council_buildings">
      <value name="COMPTYPE">BLDG</value>
    </entity>
    <entity id="council_carparks">
      <value name="COMPTYPE">CARP</value>
    </entity>
  </externaldb:transfer>
```

```

    <client:config id="test">
      ...
      <item action="com.cohga.selection.Get" system="
pathway">
          <tooltip title="Pathway" text="Get from
Pathway" />
      </item>
      ...
    </client:config>
  </config>

```

com.cohga.selection.Get
com.cohga.selection.Put

Client Actions Upload

Configuration

The namespace for the upload parameters is urn:com.cohga.spatial.upload#1.0, so the following should be added to the list of namespaces at the top of the config file:

```
xmlns:upload="urn:com.cohga.spatial.upload#1.0"
```

Then you can create a new top level configuration item to contain the customisation to the upload parameters.

The following upload parameter examples are each showing examples of changing one part of the upload configuration, when creating for your configuration for the upload there should only be a single <upload:parameters/> item

Upload parameters, containing all the available options and their default values, if you don't create an upload:parameters configuration item this is what you'll get

```

<upload:parameters>
  <cleaning maximumFileAgeInHours="96" checkDelayInHours="6" />
  <projections label="Projection" />
  <csv xlabel="X Field Column" ylabel="Y Field Column" />
  <drawing timeout="60" />
</upload:parameters>

```

Upload parameters, overriding default projection list

```

<upload:parameters>
  <projections>
    <entry value="EPSG:4326">Geographic (WGS84)</entry>
    <entry value="EPSG:4283">Geographic (GDA94)</entry>
    <entry value="EPSG:20254">MGA Zone 54 (AGD66)</entry>
    <entry value="EPSG:20255">MGA Zone 55 (AGD66)</entry>
    <entry value="EPSG:28354">MGA Zone 54 (GDA94)</entry>
    <entry value="EPSG:28355" default="true">MGA Zone 55
(GDA94)</entry>
    <entry value="EPSG:3111">VicGrid94 (GDA94)</entry>
  </projections>
</upload:parameters>

```

Basic button to add to toolbar in client config using default values and including a label for the button

```
<item action="com.cohga.spatial.upload.uploadFile" text="
Upload" />
```

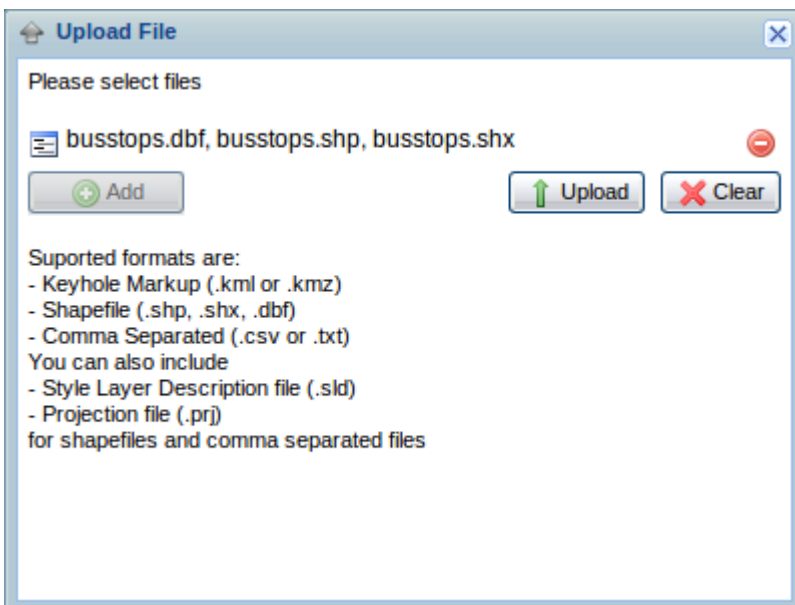
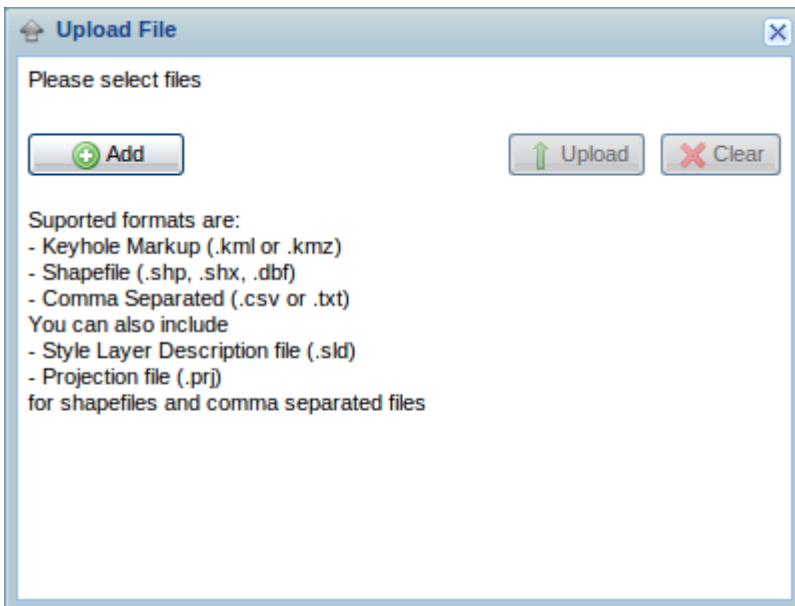
Advanced button to add to toolbar in client config with default values, if you don't override any of these values they are what you'll get

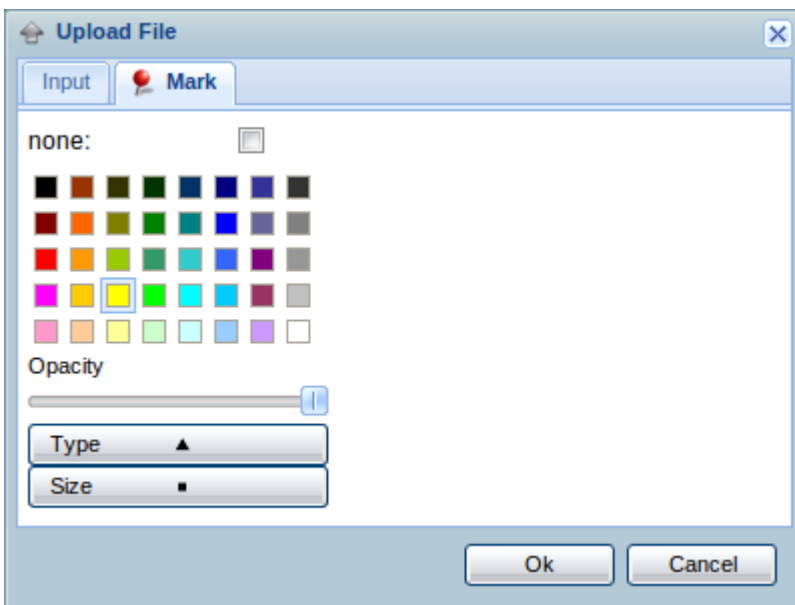
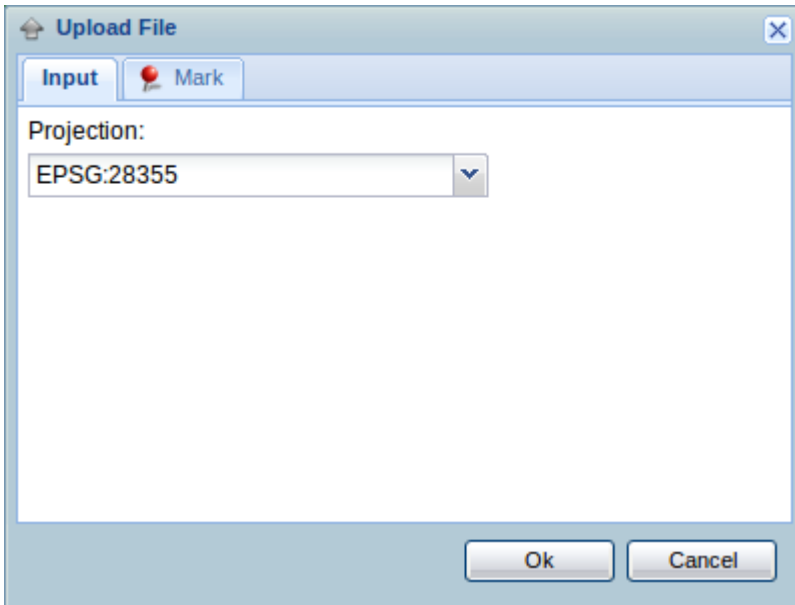
```
<item action="com.cohga.spatial.upload.uploadFile">
  <maxFileCount>5</maxFileCount>
  <windowTitle>Upload File</windowTitle>
  <windowWidth>400</windowWidth>
  <windowHeight>300</windowHeight>
  <addText>Add</addText>
  <addTooltip>Add local files</addTooltip>
  <clearAllText>Clear</clearAllText>
  <clearAllTooltip>Clear all uploading files<
/clearAllTooltip>
  <uploadText>Upload</uploadText>
  <uploadTooltip>Upload All Files</uploadTooltip>
  <stopText>Stop</stopText>
  <stopTooltip>Stop uploading</stopTooltip>
  <step1PromtText>Please select files</step1PromtText>
  <supportedFormats><![CDATA[
Supported formats are:<p>
<ol>
  <li>- Keyhole Markup (.kml or .kmz)</li>
  <li>- Shapefile (.shp, .shx, .dbf)</li>
  <li>- Comma Separated (.csv or .txt)</li>
</ol>
You can also include
<ol>
  <li>- Style Layer Description file (.sld)</li>
  <li>- Projection file (.prj)</li>
</ol>
for shapefiles and comma separated files
]]></supportedFormats>
<tocGroupLabel>Upload</tocGroupLabel>
<tocGroupDescription>Uploaded Layers<
/tocGroupDescription>
  <pointsLabel>Points</pointsLabel>
  <linesLabel>Lines</linesLabel>
  <polygonsLabel>Polygons</polygonsLabel>
  <markLabel>Mark</markLabel>
  <fillLabel>Fill</fillLabel>
  <strokeLabel>Stroke</strokeLabel>
</item>
```

Note that you only need to include any of the above setting if they're different from what's listed above.

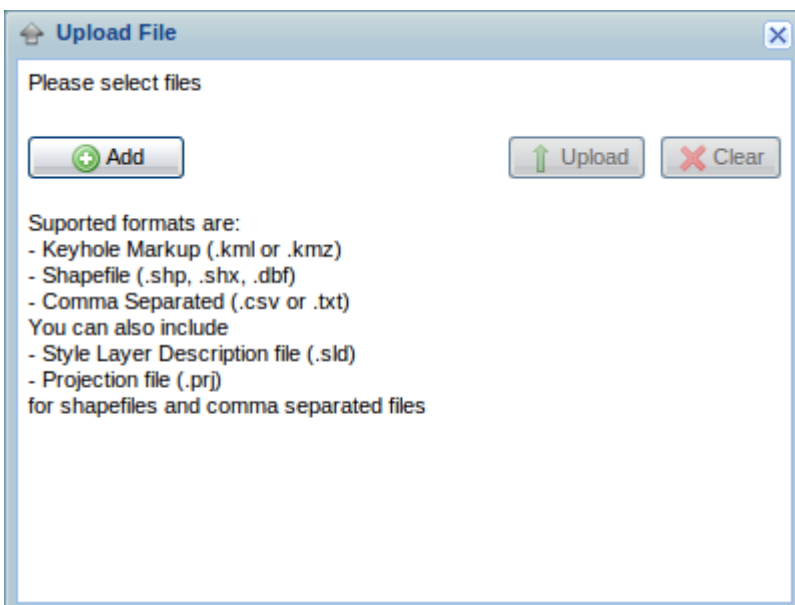
The actual default for maxFileCount is 1, but you should probably set it to 5.

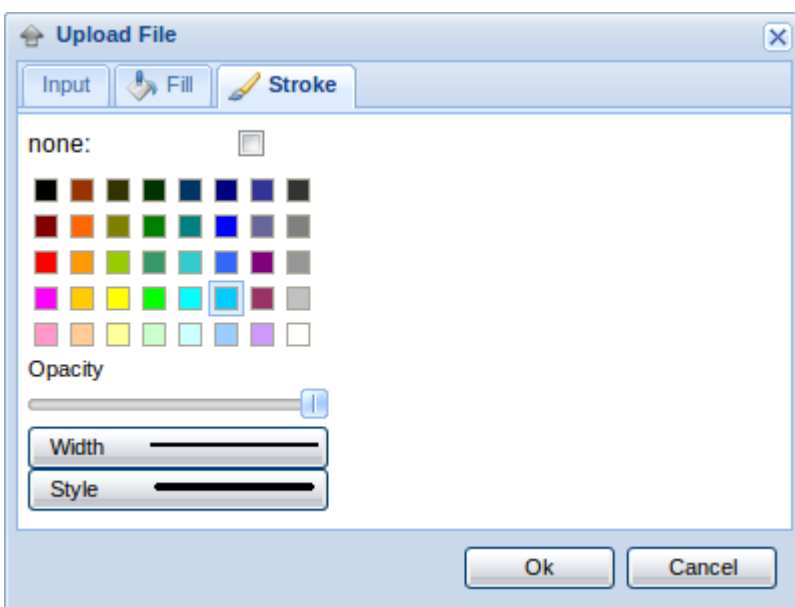
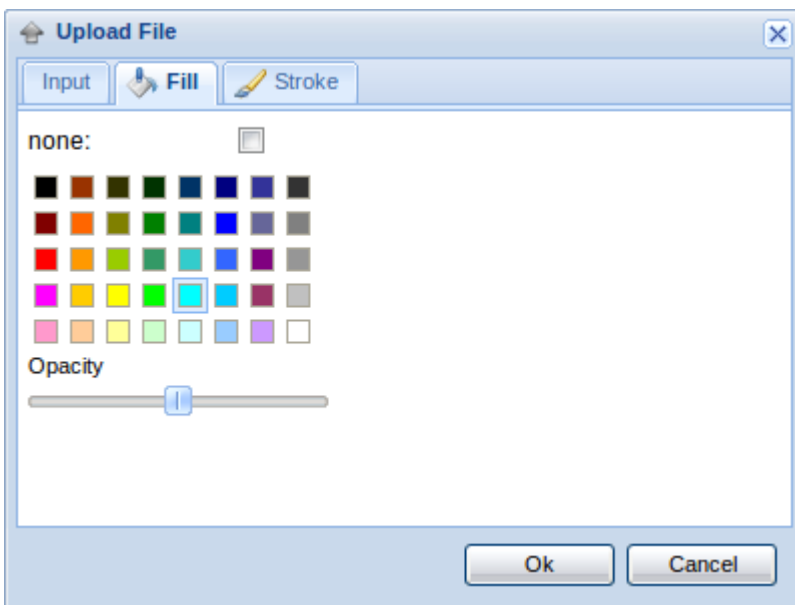
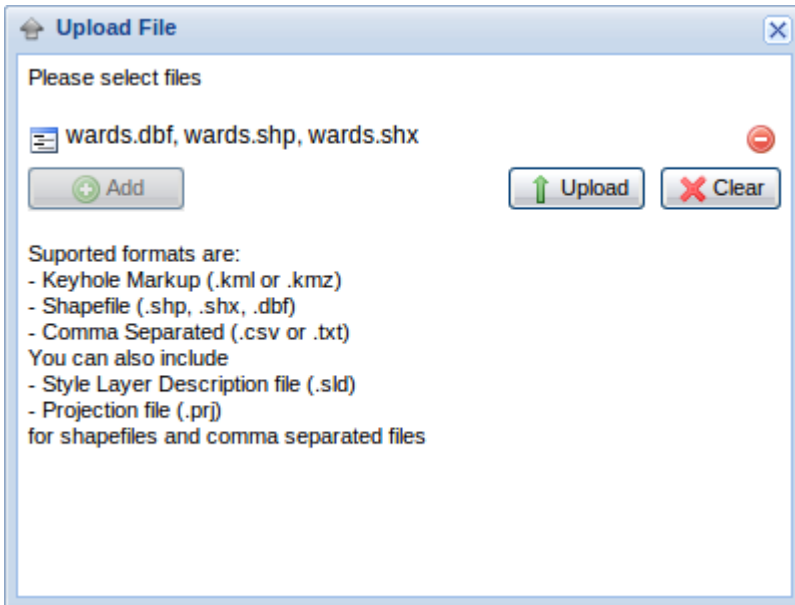
Uploading a point shapefile



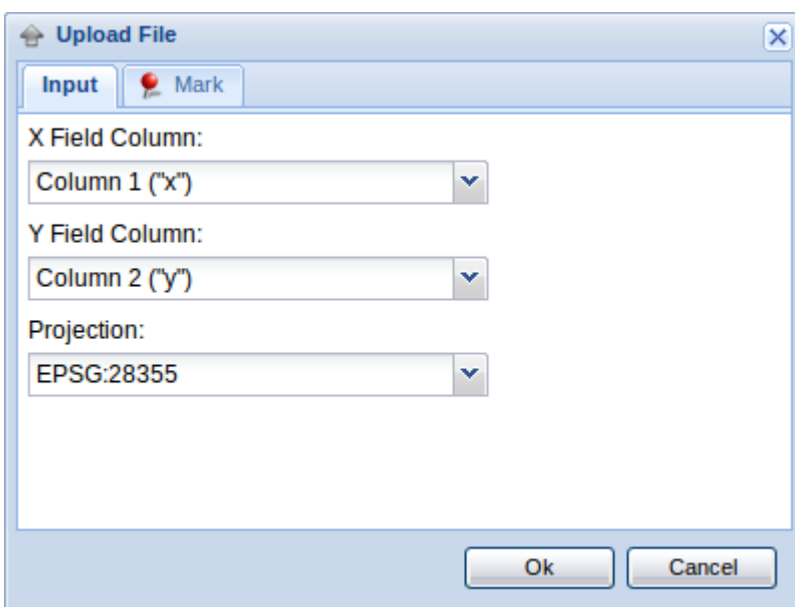
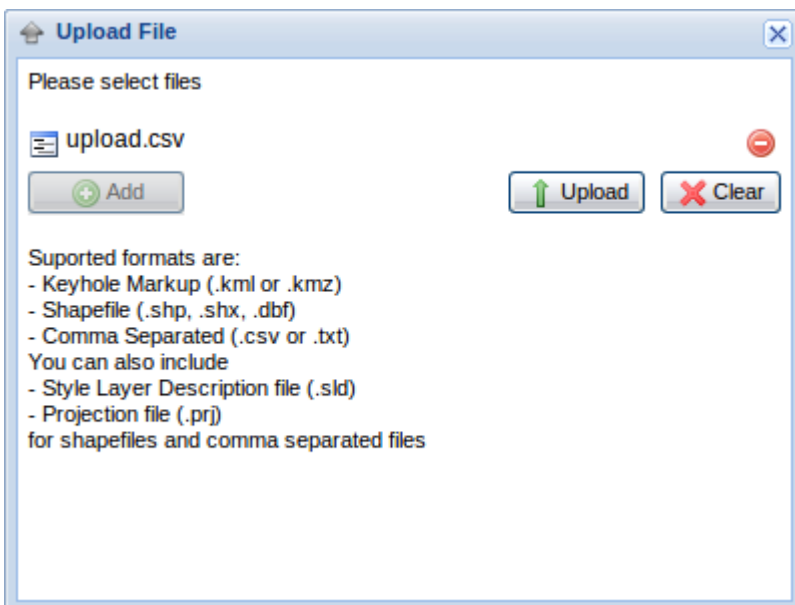
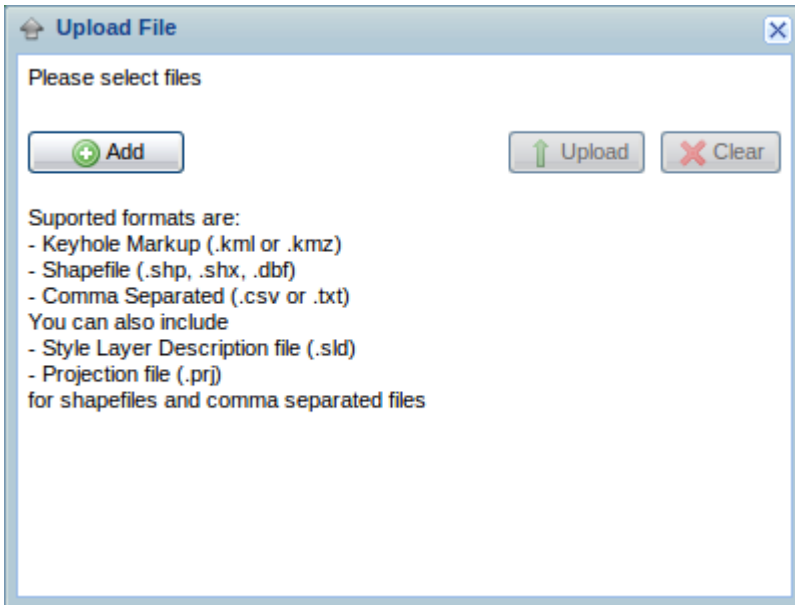


Uploading a polygon shapefile

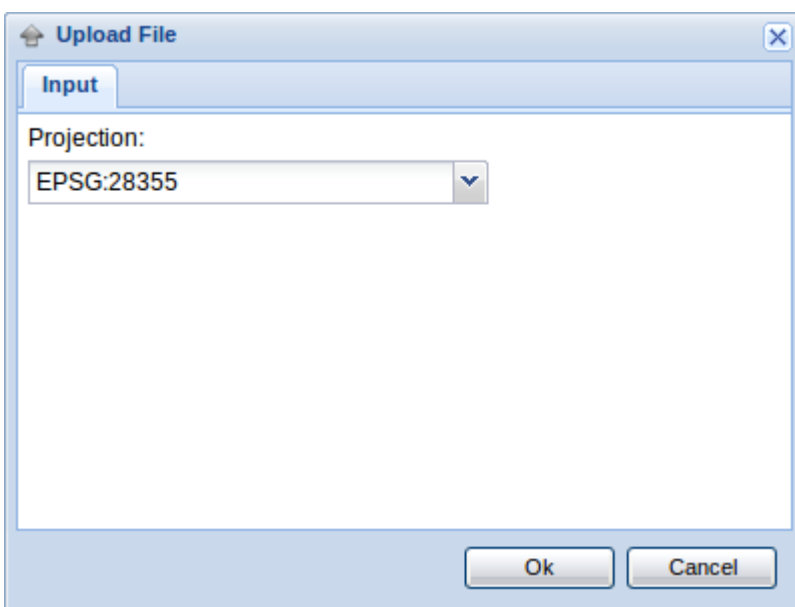
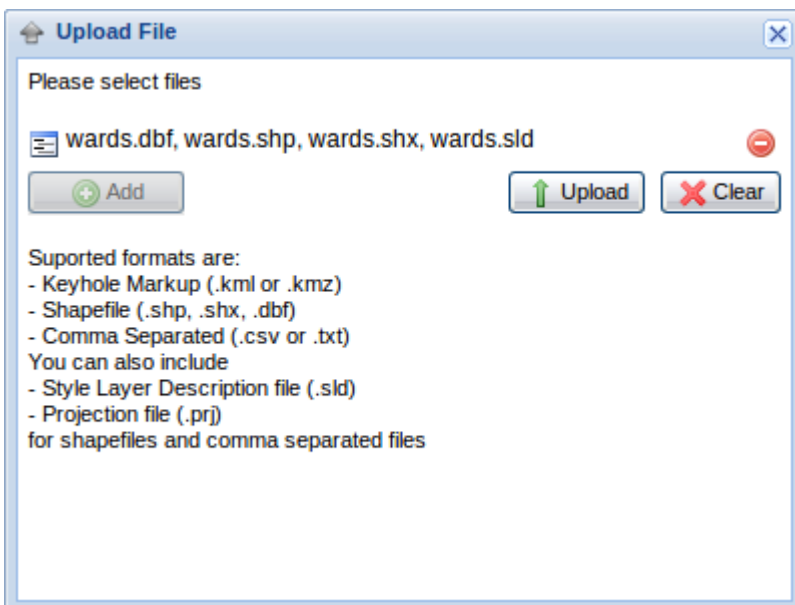
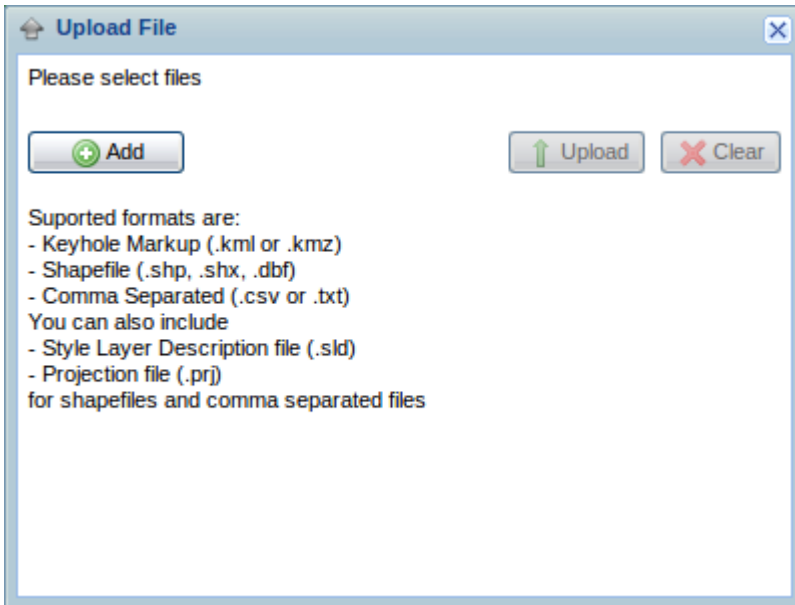




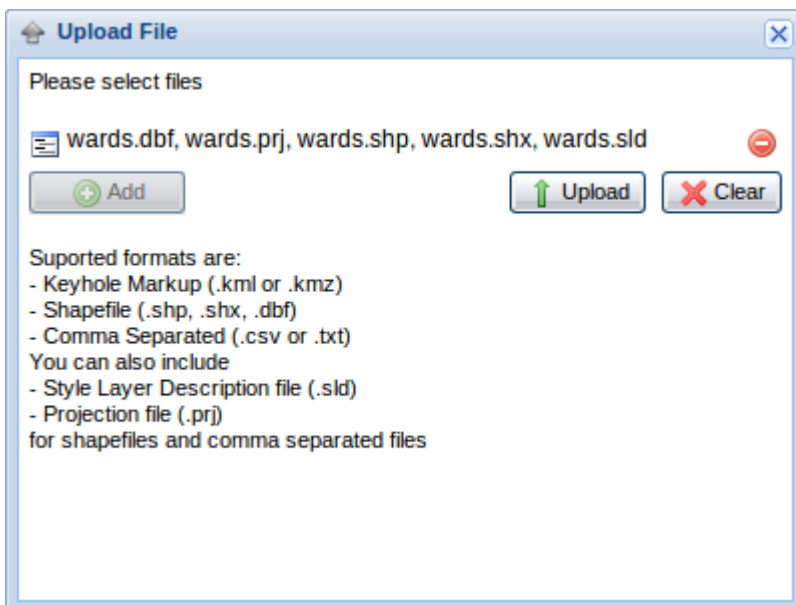
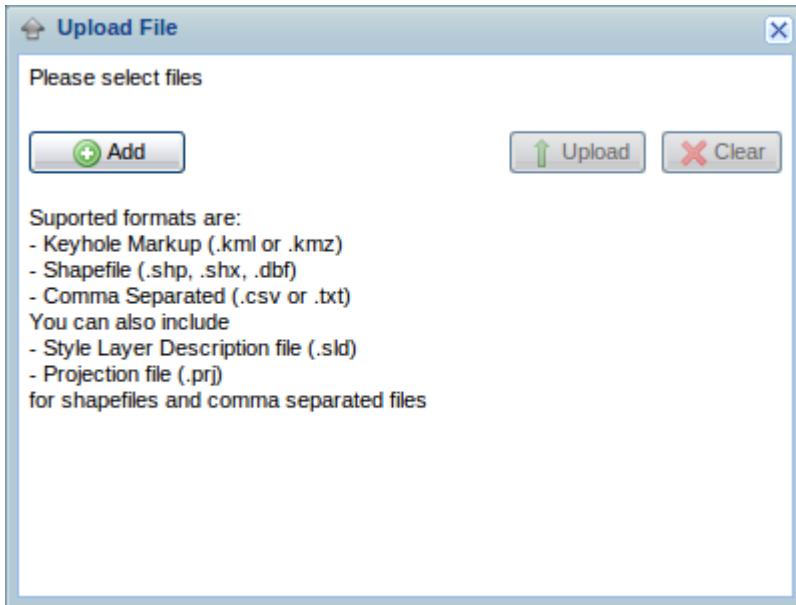
Uploading a .csv file



Upload a shapefile with a style layer description



Upload a shapefile with a style layer description and a projection



Changes from older version

- No dummy map engine required.
 - **This version does not require that you create a dummy `mapengine.upload` map engine, and if you have one it *must* be removed.**
 - The map engine is created and registered automatically.
 - The client does not *need* to reference the `mapengine.upload` map engine.
 - If the client map view does not reference the `mapengine.upload` map engine it will automatically added on top of the map, which may obscure selections and redlines, so you may want to add the `mapengine.upload` map engine to the client map view configuration anyway.
- Layer names are automatically generated.
 - The name of the layer to include in the table of contents is generated based on information in the uploaded file or the base filename.
- Multiple file upload.
 - On browsers that support it, basically anything except Internet Explorer, you can select multiple files to be uploaded at once, using the Control and Shift keys which choosing files in the file selector dialog box.
- .kml improvements.
 - .kml files compressed in a .kmz file are now supported.
 - Improved style support for .kml files.
 - Polygon, line and point (including graphic markers) styles embedded in the .kml file are now used.
 - Point markers that reference external graphics that are hosted on third party web sites will require that the Weave server has internet access to download the images.
 - Graphic markers embedded in .kmz file and referenced from point styles in the .kml file are now supported.
- .csv improvements.
 - .prj file support.
 - If you also upload a .prj file the projection of the data will be based on the projection information in the .prj file, so you don't have to choose the projection from a list.

- .sld file support.
 - If you also upload a .sld file the rendering of the data will be based on the style layer description contained in the .sld file.
 - The .sld file can reference attributes contained in the .csv file.
- .shp improvements.
 - .prj file support.
 - If you also upload a .prj file the projection of the data will be based on the projection information in the .prj file, so you don't have to choose the projection from a list.
 - .sld file support.
 - If you also upload a .sld file the rendering of the data will be based on the style layer description contained in the .sld file.
 - This requires that a .shx and .dbf file are also uploaded if the style contains references to feature attributes.
- Style improvements.
 - The user has more control over the style that uploaded data is drawn.
 - If the uploaded data does not contain a .sld file, or does not have any embedded style information, then the user is asked to provide suitable fill/stroke/marker information, not just a single colour.
- Simplified configuration.
 - No configuration of upload parameters is required.
 - You can provide configuration of upload parameters to change the projections available to the user.
 - You no longer can configure the colours or markers.
- .gml not currently supported.
 - The existing gml implementation has not been migrated to the new version.

Client Actions Upload - Obsolete

This information is for the previous upload bundle, pre-3.0.0, and is no longer supported.

The following contains the new bundles that need to be extracted and started using the usual procedures.

<https://dl.dropbox.com/u/6745119/plugins/com.cohga.spatial.upload.zip>

Then a dummy map engine must be setup to draw the uploaded layers (hopefully this will be simplified in future).

This map engine should be added to the client configurations map view.

The default name for the map engine is `mapengine.upload`, and would need to be explicitly specified in the upload configuration if a different name is used.

Example dummy upload map engine

```
<wms:mapengine id="mapengine.upload">
  <url><![CDATA[http://localhost:8081/geoserver/wms]]><
/urll>
  <format>image/png</format>
  <transparent>true</transparent>
  <layers>
    <layer>dummy</layer>
  </layers>
  <selection>>false</selection>
  <acetate>>false</acetate>
</wms:mapengine>
```

The namespace for the upload parameters is `urn:com.cohga.spatial.upload#1.0`, so the following should be added to the list of namespaces at the top of the config file:

```
xmlns:upload= "urn:com.cohga.spatial.upload#1.0"
```

Upload parameters

```
<upload:parameter tempPath="temp" destinationDir="dest"
memFileSizeThreshold="1" sizeMax="5">
  <projections label="Choose Projection">
```

```

        <entry value="EPSG:4283">Geographic (GDA94)<
/entry>
        <entry value="EPSG:20254">MGA Zone 54 (AGD66)<
/entry>
        <entry value="EPSG:20255">MGA Zone 55 (AGD66)<
/entry>
        <entry value="EPSG:28354">MGA Zone 54 (GDA94)<
/entry>
        <entry value="EPSG:28355" default="true">MGA
Zone 55 (GDA94)</entry>
        <entry value="EPSG:3111">VicGrid94 (GDA94)<
/entry>
    </projections>

    <colors label="Choose Colour">
        <entry value="#FF0000" default="true">Red<
/entry>
        <entry value="#FF7F7F">Light Red</entry>
        <entry value="#008000">Green</entry>
        <entry value="#209C20">Light Green</entry>
        <entry value="#0000FF">Blue</entry>
        <entry value="#ADD8E6">Light Blue</entry>
        <entry value="#00FFFF">Cyan</entry>
        <entry value="#FF00FF">Magenta</entry>
        <entry value="#808080">Grey</entry>
        <entry value="#D3D3D3">Light Grey</entry>
    </colors>

    <layername label="Input Layer Name" />

    <csv xLabel="X Field Column" yLabel="Y Field Column"
/>

    <cleaning maximumFileAgeInHours="96"
checkDelayInHours="6" />

    <drawing>
        <markShape>circle</markShape>
        <markShapeSize>8</markShapeSize>
        <strokeWidth>1</strokeWidth>
        <timeout>60</timeout>
    </drawing>
</upload:parameter>

```

Valid markShape values are:

- cross
- circle
- triangle
- X
- star
- arrow
- hatch
- square

Basic button to add to toolbar in client config using default values

```
<item action="com.cohga.spatial.upload.uploadFile">
```

Advanced button to add to toolbar in client config with default values customised

```
<item action="com.cohga.spatial.upload.uploadFile">
  <mapEngine>mapengine.upload</mapEngine>
  <windowTitle>Upload File</windowTitle>
  <windowWidth>400</windowWidth>
  <windowHeight>160</windowHeight>
  <addText>Add</addText>
  <addTooltip>Add local files</addTooltip>
  <clearAllText>Clear</clearAllText>
  <clearAllTooltip>Clear all uploading files<
/clearAllTooltip>
  <uploadText>Upload</uploadText>
  <uploadTooltip>Upload All Files</uploadTooltip>
  <stopText>Stop</stopText>
  <stopTooltip>Stop uploading</stopTooltip>
  <step1Heading>Step 1: Upload file(s)</step1Heading>
  <step1PromtText><![CDATA[<p>Please select files</p>]]
></step1PromtText>
  <step2Heading>Step 2: Input parameters</step2Heading>
  <tocGroupLabel>Upload</tocGroupLabel>
  <tocGroupDescription>Uploaded Layers<
/tocGroupDescription>
</item>
```

Alternative customised button

```
<item action="com.cohga.spatial.upload.uploadFile">
  <text>Import CSV File</text>
  <mapEngine>mapengine.upload</mapEngine>
  <windowTitle>Import CSV file as a New Layer<
/windowTitle>
  <windowWidth>400</windowWidth>
  <windowHeight>300</windowHeight>
  <addText>Browse</addText>
  <addTooltip>Browse Local Files</addTooltip>
  <clearAllText>Clear</clearAllText>
  <clearAllTooltip>Clear All Files</clearAllTooltip>
  <uploadText>Import</uploadText>
  <uploadTooltip>Import All Files</uploadTooltip>
  <step1Heading>Step 1: Import CSV file</step1Heading>
  <step2Heading>Step 2: Choose parameters</step2Heading>
```

```

        <step1PromtText><![CDATA[<p>Please select the CSV
file to import. (.csv or .txt and 5MB limit).</p><p>&nbsp;</p>]]><
/step1PromtText>
        <tocGroupLabel>Uploads</tocGroupLabel>
        <tocGroupDescription>Uploaded Layers<
/tocGroupDescription>
    </item>

```

Example ToC view with shapefile and csv upload

```

<view id='com.cohga.html.client.map.tocView'>
    <label>Layers</label>
    <location>west</location>
    <toolbar>
        <item>-&gt;</item>
        <item action="com.cohga.client.actions.menuAction">
            <iconCls>icon-database_go</iconCls>
            <text>Add Data</text>
            <item action="com.cohga.spatial.upload.
uploadFile">
                <tocId>toc.main</tocId>
                <mapEngine>mapengine.upload<
/MapEngine>
                <text>Import CSV File</text>
                <windowTitle>Import CSV file as a New
Layer</windowTitle>
                <windowWidth>400</windowWidth>
                <windowHeight>400</windowHeight>
                <addText>Browse</addText>
                <addTooltip>Browse Local Files<
/addTooltip>
                <clearAllText>Clear</clearAllText>
                <clearAllTooltip>Clear All Files<
/clearAllTooltip>
                <uploadText>Import</uploadText>
                <uploadTooltip>Import All Files<
/uploadTooltip>
                <step1Heading>Step 1: Import CSV file<
/step1Heading>
                <step2Heading>Step 2: Choose
parameters</step2Heading>
                <step1PromtText><![CDATA[<p>Please
select the CSV file to import. (.csv or .txt and 5MB limit).<
/p><p>&nbsp;</p>]]></step1PromtText>
                <tocGroupLabel>Uploads</tocGroupLabel>
                <tocGroupDescription>Uploaded Layers<
/tocGroupDescription>
            </item>
            <item action="com.cohga.spatial.upload.
uploadFile">
                <maxFileCount>4</maxFileCount>

```

```

        <tocId>toc.main</tocId>
        <mapEngine>mapengine.upload<
    /mapEngine>

    <text>Import Shapefile</text>
    <windowTitle>Import Shapefile as a
New Layer</windowTitle>

    <windowWidth>400</windowWidth>
    <windowHeight>300</windowHeight>
    <addText>Browse</addText>
    <addTooltip>Browse Local Files<
    /addTooltip>

    <clearAllText>Clear</clearAllText>
    <clearAllTooltip>Clear All Files<
    /clearAllTooltip>

    <uploadText>Import</uploadText>
    <uploadTooltip>Import All Files<
    /uploadTooltip>

    <step1Heading>Step 1: Select .shp, .
dbf and .shx files</step1Heading>
    <step2Heading>Step 2: Choose
parameters</step2Heading>

    <step1PromtText><![CDATA[<p>Please
select the shapefile to import. (.shp, .shx and .dbf and 5MB limit).<
/p><p>&nbsp;</p>]]></step1PromtText>

    <tocGroupLabel>Uploads</tocGroupLabel>
    <tocGroupDescription>Uploaded Layers<
    /tocGroupDescription>
    </item>
    </item>
</toolbar>
</view>

```

Client Actions ZipNShip

Provides options to export data from the current selection to a zip file in various formats

ID

weave.zipnship

Notes

This is a simplified version of the [ZipNShip Envelope](#) action and only exports the current selection.

The tool provides a menu with options to allow the user to select which format(s) they want to export the selection as, and if they want to export the selection for just the currently active entity or for all entities that have a selection.

The `crs` attribute determines what coordinate reference system to output the data in and if it's not specified then the original coordinate reference system of the underlying data will be used.

An example of the item required to add the new tool to a toolbar

```

<toolbar>
    ...
    <item action="weave.zipnship" crs="EPSG:4326"/>
    ...
</toolbar>

```

The entities/entity tags can refine which entities the tool should be enabled for.

An example of refining the available entities

```
<toolbar>
  ...
  <item action="weave.zipnship">
    <entities>
      <entity>property</entity>
      <entity>assets</entity>
      <entity>utilities</entity>
    </entities>
  </item>
  ...
</toolbar>
```

Allowing the user to choose the output projection, as an alternative to setting the crs as in the first example above

Allowing the user to choose the output projection

```
<toolbar>
  ...
  <item action="com.cohga.client.actions.zipnship">
    <projections>
      <entry value="EPSG:4326">Geographic (WGS84)<
/entry>
      <entry value="EPSG:7844">Geographic (GDA2020)<
/entry>
      <entry value="EPSG:4283">Geographic (GDA94)<
/entry>
      <entry value="EPSG:4203">Geographic (AGD84)<
/entry>
      <entry value="EPSG:4202">Geographic (AGD66)<
/entry>
      <entry value="EPSG:7855" default="true">MGA
Zone 55 (GDA2020)</entry>
      <entry value="EPSG:28355">MGA Zone 55 (GDA94)<
/entry>
      <entry value="EPSG:20355">MGA Zone 55 (AGD84)<
/entry>
      <entry value="EPSG:20255">MGA Zone 55 (AGD66)<
/entry>
      <entry value="EPSG:7899">VicGrid (GDA2020)<
/entry>
      <entry value="EPSG:3111">VicGrid (GDA94)<
/entry>
    </projections>
  </item>
  ...
</toolbar>
```

Client Actions ZipNShip Envelope

Provides options to export data from the current extent to a zip file in various formats

ID

weave.zipnshipenvelope

Notes

The `layer` tags in a `group` that has `raster` set to `true` refer to layer ids from the underlying map service.

When `raster` isn't set (or is set to `false`) then they're mappers configured through a spatial mapper tag. This way you can add new layers to be exported as vectors by defining a new spatial mapper, without the need to setup a corresponding entity tag.

`welcome` and `license` are optional, the panel won't show if they're not set.

If only one `size` is set then the size selection panel won't show.

A `group` with `raster` set to `but` with no layers will generate an image representing the current map as the user sees it.

The available vector and raster formats are determined by the zipnship client bundles that are installed.

Any files you drop into `...platform\workspace\zipnship` will be included in the generated zip file.

Example

```
<item action="weave.zipnshipenvelope">
  <tooltip>
    <title>Export</title>
    <text>Set you scale between 2000 and 15000 to
be able to export</text>
  </tooltip>
  <minScale>2000</minScale>
  <maxScale>15000</maxScale>
  <welcome>This is the text to be displayed on the
first wizard screen</welcome>
  <license><![CDATA[These are the terms and contitions
that you <b>must</b> agree to before downloading this data.<br
/>Further information will be contained within the generated
content.]]></license>
  <size width="400" height="400"/>
  <size width="800" height="600"/>
  <size width="1024" height="768"/>
  <size width="1200" height="1000"/>
  <size width="1600" height="1600"/>
  <group id="base" label="Base Layers" disabled="true"
checked="true">
    <layer>property</layer>
    <layer>roads</layer>
    <metadata><![CDATA[Base layers metadata goes
here]]></metadata>
  </group>
  <group id="pipes" label="Pipes Layers">
    <layer>drainage</layer>
    <metadata><![CDATA[Pipe Layers related
<i>metadata</i> goes here]]></metadata>
  </group>
  <group id="contours" label="Contours" maxScale="3000">
```



```

        <layer>manningham.contour</layer>
        <layer>manningham.con2m</layer>
        <layer>manningham.con5m</layer>
        <layer>manningham.con10m</layer>
    </group>
    <group id="aerial" label="Aerial Photo" checked="
true" raster="true">
        <layer>manningham.aerial</layer>
        <metadata><![CDATA[More metadata can be
specified here]]></metadata>
    </group>
    <group id="current" label="Current Map" raster="true"
/>
</item>

```

```

<item action="weave.zipnshipenvelope">
    <acl:acl id="acl.public"/>
    <tooltip>
        <title>Export Vector</title>
        <text>Export Vector Data to file</text>
    </tooltip>
    <maxScale>5000</maxScale>
    <welcome><![CDATA[Welcome text here]]></welcome>
    <license><![CDATA[License text here]]></license>
    <size width="1024" height="768"/>
    <group id="base" label="Base Layers" disabled="true" checked="
true">
        <layer>property</layer>
        <layer>road</layer>
        <metadata><![CDATA[Metadata text here]]></metadata>
    </group>
    <group id="water" label="Water Layers">
        <layer>asset_ww</layer>
        <layer>asset_sw</layer>
        <layer>asset_ws</layer>
        <metadata><![CDATA[Metadata text here]]></metadata>
    </group>
    <group id="other" label="Other Layers">
        <layer>park</layer>
        <layer>education</layer>
        <metadata><![CDATA[Metadata text here]]></metadata>
    </group>
    <group id="aerial" label="Aerial Photography" raster="true">
        <layer>photo250to500</layer>
        <layer>photo500to2500</layer>
        <layer>photo2500to10000</layer>
    </group>
</item>

```

Grouping entires into categories

```

<item action="weave.zipnshipenvelope">
  <group id="base" label="Base Layers" disabled="true" checked="
true" category="Main">
    <layer>property</layer>
    <layer>road</layer>
  </group>
  <group id="water" label="Water Layers" category="Main">
    <layer>asset_ww</layer>
    <layer>asset_sw</layer>
    <layer>asset_ws</layer>
  </group>
  <group id="other" label="Other Layers" category="Other">
    <layer>park</layer>
    <layer>education</layer>
  </group>
  <group id="aerial" label="Aerial Photography" raster="true"
category="Other">
    <layer>photo250to500</layer>
    <layer>photo500to2500</layer>
    <layer>photo2500to10000</layer>
  </group>
</item>

```

Client Components

Components are additional ways that the user can interact with the Weave client and also provide a way for the client to display information to the user.

They generally provide a more complex interaction with Weave than a simple action does. They're setup in the client config section of the config file.

Available Components

name	id	altername id
Loading	com.cohga.html.client.components.loadingComponent	
Status	com.cohga.html.client.components.statusMessageComponent	
Scale Selector	com.cohga.html.client.components.scaleSelector	
Entity	com.cohga.html.client.components.entitySelectorComponent	weave.entitySelector
Scale Bar	com.cohga.html.client.components.scalebar	
Image Slider	com.cohga.html.client.components.imageTransSlider	
Single Image Slider ¹	com.cohga.html.client.components.singleImageTransSlider com.cohga.html.client.components.singleImageChooser	weave.singleImageTransSlider weave.singleImageChooser
Index Combo	weave.indexcombo	

1. Weave 2.6.7 or later

Examples

Scale bar

```

<item component="com.cohga.html.client.components.scalebar">
  <!-- For simple scale text only (text mode) -->

```

```

<simple>true</simple>
<!-- Location of scale text in text mode -->
<height>10</height>
<!-- For display of scale text & scale bar -->
<showText>true</showText>
<!-- Controls the appearance of the scale bar, maximum is 2 --
>

<lines>1</lines>
<!-- Default units -->
<units>m</units>
<!-- Enable menu to change units by right mouse click over
scale bar -->
<showMenu>true</showMenu>
</item>

```

Single image slider

```

<!-- The following is included in the Weave client file -->
<view id="com.cohga.html.client.map.mapView">
  <label>Map</label>
  <location>center</location>
  ...
  ...

  <statusbar>
    ...
    <item>ToC Layers</item>
    <item component="weave.singleImageTransSlider">
      <width>150</width>
      <value>1</value>
      <sliderId>slider1</sliderId>
    </item>
    <item component="weave.singleImageChooser">
      <sliderId>slider1</sliderId>
      <layer id="0" text="Google Maps Satellite"
mapengine="google.satellite" isDefault="true"/>
      <layer id="0" text="Google Maps Standard"
mapengine="google.standard"/>
      <layer id="0" text="OSM" mapengine="osm"/>
    </item>
  </statusbar>
  ...
</view>

<!-- With matching layers in the ToC model -->
  <entry id='l_mapengine.basemap_google_standard'
layer='0' label='Google Standard' mapengine='google.standard'
visible='false' checked='false' />
  <entry id='l_mapengine.basemap_google_satellite'
layer='0' label='Google Satellite' mapengine='google.satellite'
visible='false' checked='false' />
  <entry id='l_mapengine.basemap_osm' layer='0'

```

```
label='OpenStreetMap' mapengine='osm' visible='false' checked='false'
/>
```

Client Components Entity

The Entity Selector component provides the user with the ability to change the currently active entity.

ID

```
com.cohga.html.client.components.entitySelectorComponent
weave.entitySelector
```

Sub-tags

name	type	cardinality	default	description
label	string	0..1		Label to display to the left of the component
entity	string	0..n	all	Limit the display to only the listed entities. See this page for examples on how to refine the listed entities.
entities	entity[]	0..1		A wrapper for providing multiple entity entries (helpful when used as a snippet)
showIcons	boolean	0..1	true	Should icons be shown representing the different types of entities
showCount	boolean	0..1	true	Should the count of selected entities be shown in the list
showNonSpatial	boolean	0..1	true	Should non-spatial entities be include in the list or only entities that have a spatial component
width	integer	0..1	200	The width of the combo box
listWidth	integer	0..1	200	The width of the combo box popup list

Example

Basic setup

```
<item component="com.cohga.html.client.components.
entitySelectorComponent" label="Entity:" />
```

Simplified setup

```
<item component="com.cohga.html.client.components.
entitySelectorComponent" label="Entity:" showNonSpatial="false"
showIcons="false" showCount="false" />
```

Refining available entities

```
<item component="com.cohga.html.client.components.
entitySelectorComponent">
  <entities>
    <entity>property</entity>
    <entity>owners</entity>
    <entity>roads</entity>
  </entities>
</item>
```

Client ContextMenus

Below are a couple of samples of context menus that can be added to the Weave client configuration.

The contentmenu tag should be added under the `tocView` or `mapView`, for example:

```
<view id="com.cohga.html.client.map.tocView">
  <label>Layers</label>
  <location>west</location>
  <contextmenu>
    <item action="weave.toc.openUrl">
      <text>Metadata</text>
      <url>/resources/metadata/{layername}.htm</url>
      <external>true</external>
    </item>
  </contextmenu>
</view>
```

Map

```
<contextmenu>
  <item action="weave.zoomToScale">
    <text>Street Level</text>
    <scale>2500</scale>
  </item>
  <item action="weave.zoomToScale">
    <text>Suburb Level</text>
    <scale>25000</scale>
  </item>
  <item action="weave.zoomToScale">
    <text>Council Level</text>
    <scale>130000</scale>
  </item>
  <item>-</item>
  <item action="weave.previousExtent">
    <text>Previous Extent</text>
  </item>
  <item action="weave.nextExtent">
    <text>Next Extent</text>
  </item>
  <item action="weave.initialExtent">
    <text>Initial Extent</text>
  </item>
  <item action="weave.fullExtent">
    <text>Full Extent</text>
  </item>
  <item>-</item>
  <item action="weave.fixedZoomIn">
    <text>Zoom In</text>
  </item>
  <item action="weave.fixedZoomOut">
    <text>Zoom Out</text>
```

```

</item>
<item>--</item>
<item action="weave.selectZoom">
    <text>Zoom To Selection</text>
</item>
<item>--</item>
<item action="weave.clear">
    <text>Clear Selection</text>
</item>
<item action="weave.clearAll">
    <text>Clear All</text>
</item>
<item action="weave.coords">
    <text>Coordinates</text>
    <projection id="EPSG:4283" label="Geographic (GDA94)"
/>
    <projection id="EPSG:4326" label="WGS84"/>
    <projection id="EPSG:28355" label="GDA94 / MGA Zone
55"/>
    <projection id="EPSG:3111" label="VicGrid94 (GDA94)"/>
    <projection id="EPSG:28355" label="AMG Zone 55"/>
</item>

<!-- Link to an external system via a URL with an embedded id
-->
    <item action="weave.quickLink" entity="sewer_pipes" filter="
sewer_pipe.plan_no" url="https://iswva02:1548/viewer.asp?
type=embedd&id={id}" text="Sewer Plans" iconCls="icon-sewer_app">
        <tooltip title="Sewer Plans" text="Show the plans for
the clicked sewer"/>
    </item>
</contextmenu>

```

Table of Contents

```

<contextmenu>
    <item action="weave.toc.selectLayer">
        <text>Set Active</text>
    </item>
    <item action="weave.toc.clearSelection">
        <text>Clear Selection</text>
    </item>
    <item action="weave.toc.zoomSelection">
        <text>Zoom To Selection</text>
    </item>
    <item>--</item>
    <item action="weave.toc.zoomToMaxScale">
        <text>Zoom to the Maximum Scale</text>
    </item>
    <item action="weave.toc.zoomToMinScale">
        <text>Zoom to the Minimum Scale</text>
    </item>

```

```

    <item action="weave.toc.zoomToLayer">
        <text>Zoom to Layer</text>
    </item>
    <item action="weave.toc.checkAll"/>
    <item action="weave.toc.uncheckAll"/>
    <item>-</item>
    <item action="weave.toc.openUrl">
        <text>Metadata</text>
        <external>true</external>
        <iconCls>icon-tag-green</iconCls>
        <layer label="Mineral Tenements"> <!-- use toc entry
label to specify matching layer -->
            <url><![CDATA[http://metadata.nre.vic.gov.au
/metadata/anzlic_report.cfm?dataset_name=MINTEN]]></url>
        </layer>
        <layer id="l_property"> <!-- use toc entry id to
specify matching layer -->
            <url><![CDATA[http://metadata.nre.vic.gov.au
/metadata/anzlic_report.cfm?dataset_name=PROPERTY]]></url>
        </layer>
        <layer mapengine="nre" layer="roads"> <!-- use
mapengine/layer id to specify matching layer -->
            <url><![CDATA[http://metadata.nre.vic.gov.au
/metadata/anzlic_report.cfm?dataset_name=ROADS]]></url>
        </layer>
        <url><![CDATA[http://metadata.nre.vic.gov.au/metadata
/anzlic_report.cfm]]></url> <!-- optionally specify fallback for
other layers not listed above -->
    </item>
</contextmenu>

```

Client Samples

Below you'll find various samples for different client configurations.

Very simple map based client

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.
cohga.html.client#1.0">

    <client:config id="map">
        <debug>>false</debug>
        <title>Basic Map Client</title>
        <description>A very basic client showing a map and
legend</description>
        <perspective>
            <text>Main</text>
            <layout>
                <west>
                    <name>legend</name>

```

```

                <type>fit</type>
                <width>200</width>
                <minSize>175</minSize>
                <maxSize>400</maxSize>
                <split>true</split>
            </west>
            <center>
                <type>fit</type>
            </center>
        </layout>

        <view id="com.cohga.client.panel.legend">
            <label/>
            <location>legend</location>
        </view>

        <view id="com.cohga.html.client.map.mapView">
            <label/>
            <location>center</location>
            <toolbar>
                <item action="weave.
previousExtent"/>
                <item action="weave.
nextExtent"/>
                <item action="weave.
initialExtent"/>
                <item action="weave.
fullExtent"/>
                <item action="weave.zoomIn"
isDefault="true"/>
                <item action="weave.zoomOut"/>
                <item action="weave.pan"/>
                <item action="weave.
refreshMap"/>
                <item>-></item>
                <item component="com.cohga.
html.client.components.loadingComponent"/>
            </toolbar>

            <control id="weave.mapctrl.keyboard"/>
            <control id="com.cohga.client.mapctrl.
mousePosition"/>
            <control id="com.cohga.client.mapctrl.
navigation"/>

            <mapEngine id="mapengine.main">
                <options>
                    <opacity>1.0</opacity>
                    <isBaseLayer>true<
/isBaseLayer>
                <transitionEffect>resize</transitionEffect>
                <singleTile>true<
/singleTile>

```



```

                                <selection>>false<
/selection>

                                <ratio>1.2</ratio>
                                </options>
                                </mapEngine>
                                <crs>EPSG:20255</crs>
                                <extents>
                                <initial minx="327098" miny="
5811358" maxx="351971" maxy="5827675"/>
                                <full minx="327098" miny="
5811358" maxx="351971" maxy="5827675"/>
                                <limit minx="320000" miny="
5810000" maxx="360000" maxy="5840000"/>
                                </extents>
                                </view>
                                </perspective>
                                </client:config>

</config>

```

Client Snippets

Snippets and References

If you're running version 1.2.0 or later of the `com.cohga.client.weave` bundle you'll be able to take advantage of `snippets`, which are a way of reusing a piece of client configuration in multiple client configurations.

Previously the client configuration bundle only understood a single tag, which was `config`, and was used to define separate client configurations. Version 1.2.0 of `com.cohga.client.weave` introduces support for other tags allowing you to create little bits of configuration that can be reused within a `config` tag.

This allows you to centralize common configuration items from multiple client configurations that you want to be the same for all client configurations, for example setting the extents for map views, or ensuring the same label for every ToC view, creating common layouts. Something like this can also be performed using the [set](#) processing instruction, but this method makes it much easier.

Snippets can only be used with client configuration items

To define a new snippet all you need to do is create the XML tags as you would if you were including them directly in the client configuration but rather than including them within the client configuration you create the tag at the top level within the `config.xml` file.

For example, lets assume that we want to create a snippet for the map extents in a map view, that way we can have multiple client configurations all with the same extents and only one location to set them, making it easier to change all of the clients at once. So, if we currently have something like the following:

Example, all be it incomplete, showing some client configurations

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.
cohga.html.client#1.0">

    <client:config id="test1">
        <view id="com.cohga.html.client.map.mapView">
            <extents>
                <initial minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>

```

```

                                <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
                                </extents>
                            </view>
                        </client:config>

                        <client:config id="test2">
                            <view id="com.cohga.html.client.map.mapView">
                                <extents>
                                    <initial minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
                                    <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
                                </extents>
                            </view>
                        </client:config>

                        <client:config id="test3">
                            <view id="com.cohga.html.client.map.mapView">
                                <extents>
                                    <initial minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
                                    <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
                                </extents>
                            </view>
                        </client:config>

</config>

```

To create a new snippet for the extents in the above sample we would extract the part of the configuration we want to make common, like this:

Common extents extracted into a snippet with the id 'defaults'

```

...
    <client:extents id="defaults">
        <initial minx="327098" miny="5811358" maxx="351971"
maxy="5827675" />
        <full minx="327098" miny="5811358" maxx="351971"
maxy="5827675" />
    </client:extents>
...

```

This would create a new snippet, called `defaults` of type `extents`. The id, `defaults` in the above example, isn't important and we can use anything that makes sense here (it's just used to refer to a specific snippet later when we want to use it). The type, `extents` in the above example, is important however and must be the same as the target tag that we're actually creating the snippet for, and it's the combination of the id and type that's used later on to include the snippet within a client configuration.

At the simplest to use the snippet within the client configuration simply requires a `reference` to the snippet to be added in place of the original tags, so to reference the new snippet we created you would replace the original extents tags with the following:

Referencing the new snippet

```
<extents ref="defaults"/>
```

Anywhere this is found within the client configuration it will be replaced with the contents of the snippet we defined earlier.

So to complete our example I'll show the complete client configurations that will replace the original using snippets:

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.cohga.html.client#1.0">

    <client:extents id="defaults">
        <initial minx="327098" miny="5811358" maxx="351971"
maxy="5827675"/>
        <full minx="327098" miny="5811358" maxx="351971"
maxy="5827675"/>
    </client:extents>

    <client:config id="test1">
        <view id="com.cohga.html.client.map.mapView">
            <extents ref="defaults"/>
        </view>
    </client:config>

    <client:config id="test2">
        <view id="com.cohga.html.client.map.mapView">
            <extents ref="defaults"/>
        </view>
    </client:config>

    <client:config id="test3">
        <view id="com.cohga.html.client.map.mapView">
            <extents ref="defaults"/>
        </view>
    </client:config>

</config>
```

Refining snippets

Beyond just including the snippet in the client configuration you can also alter the snippet by including extra sub-tags in the reference. For example if we wanted to setup one client configuration to have a different initial extent and another to add a limit extent we could do the following:

Altering included snippets

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.cohga.html.client#1.0">

    <client:extents id="defaults">
```

```

                <initial minx="327098" miny="5811358" maxx="351971"
maxy="5827675" />
                <full minx="327098" miny="5811358" maxx="351971"
maxy="5827675" />
            </client:extents>

            <client:config id="test1">
                <view id="com.cohga.html.client.map.mapView">
                    <extents ref="defaults" />
                </view>
            </client:config>

            <client:config id="test2">
                <view id="com.cohga.html.client.map.mapView">
                    <extents ref="defaults">
                        <initial minx="327000" miny="5811000"
maxx="330000" maxy="5813000" />
                    </extents>
                </view>
            </client:config>

            <client:config id="test3">
                <view id="com.cohga.html.client.map.mapView">
                    <extents ref="defaults">
                        <limit minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
                    </extents>
                </view>
            </client:config>

</config>

```

this would be the equivalent of

Altered example shown not using snippets

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.
cohga.html.client#1.0">

    <client:config id="test1">
        <view id="com.cohga.html.client.map.mapView">
            <extents>
                <initial minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
                <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
            </extents>
        </view>
    </client:config>

```

```

<client:config id="test2">
  <view id="com.cohga.html.client.map.mapView">
    <extents>
      <initial minx="327000" miny="5811000"
maxx="330000" maxy="5813000"/>
      <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
    </extents>
  </view>
</client:config>

<client:config id="test3">
  <view id="com.cohga.html.client.map.mapView">
    <extents>
      <initial minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
      <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
      <limit minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
    </extents>
  </view>
</client:config>

</config>

```

One thing you can't currently do is to remove a sub-tag from a snippet via the reference.

Nested snippets

Snippets can include references to other snippets and references can include references to other snippets. For example we could've created our extents snippet as follows

```

...
  <client:initial id="default" minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>

  <client:full id="default" minx="327098" miny="5811358" maxx="
351971" maxy="5827675"/>

  <client:limit id="default" minx="320000" miny="5810000" maxx="
360000" maxy="5840000"/>

  <client:extents id="defaults">
    <initial ref="default"/>
    <full ref="default"/>
    <limit ref="default"/>
  </client:extents>
...

```

And if we wanted to we could create a separate snippet for a "custom" extent

```

...
    <client:initial id="custom" minx="328000" miny="5811000"
maxx="331000" maxy="5817000" />
...

```

Then in our "custom" client configuration we could use the following to accomplish the same as we did previously

```

...
    <client:config id="custom">
        <view id="com.cohga.html.client.map.mapView">
            <extends ref="defaults">
                <initial ref="custom" />
            </extends>
        </view>
    </client:config>
...

```

Replacing content with an id attributes

Because snippets use an `id` attribute to uniquely identify a snippet, along with the type, you may be wondering how to create a snippet out of a configuration item that includes an `id` attribute.

For example if we want to create a common map view that we include in multiple clients how can we do it since the `view` tag has an `id`, and when we create the snippet the `id` which is used to identify which particular view is to be used will now become the identifier for the snippet. So if we tried to extract the map view from the following examples:

Example with two client configurations

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.cohga.html.client#1.0">

    <client:config id="test1">
        <view id="com.cohga.html.client.map.mapView">
            <extends>
                <initial minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
                <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
            </extends>
        </view>
    </client:config>

    <client:config id="test2">
        <view id="com.cohga.html.client.map.mapView">
            <extends>
                <initial minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
                <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675" />
            </extends>
        </view>
    </client:config>

```

```

        </view>
    </client:config>

</config>

```

we might come up with the following, which would **not** work.

Extracting map view into a snippet the wrong way

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.cohga.html.client#1.0">

    <client:view id="com.cohga.html.client.map.mapView">
        <extents>
            <initial minx="327098" miny="5811358" maxx="351971" maxy="5827675"/>
            <full minx="327098" miny="5811358" maxx="351971" maxy="5827675"/>
        </extents>
    </client:view>

    <client:config id="test1">
        <view ref="com.cohga.html.client.map.mapView"/>
    </client:config>

    <client:config id="test2">
        <view ref="com.cohga.html.client.map.mapView"/>
    </client:config>

</config>

```

The reason the above code does not work is that the code that processes the snippets does not know that the `id` for the view tag is important, and just treats it as an identifier used to provide the link between the snippet and the reference to include it. But as part of the including process it's the content of the snippet, which does not include the `id`, that replaces the reference, thereby losing the `id` as part of the include process.

To help explain why the above example doesn't work the following is the equivalent of the above example if snippets were not used, which show that the view tags do not have an `id`

Example with two client configurations

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.cohga.html.client#1.0">

    <client:config id="test1">
        <view>
            <extents>

```

```

                                <initial minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
                                <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
                                </extents>
                                </view>
                                </client:config>

                                <client:config id="test2">
                                <view>
                                <extents>
                                <initial minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
                                <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
                                </extents>
                                </view>
                                </client:config>

</config>

```

The way to fix this problem is to include the `id` in the reference, as follows:

Example with two client configurations

```

<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.
cohga.html.client#1.0">

    <client:view id="map">
        <extents>
            <initial minx="327098" miny="5811358" maxx="
351971" maxy="5827675"/>
            <full minx="327098" miny="5811358" maxx="
351971" maxy="5827675"/>
        </extents>
    </client:view>

    <client:config id="test1">
        <view ref="map" id="com.cohga.html.client.map.mapView"
/>
    </client:config>

    <client:config id="test2">
        <view ref="map" id="com.cohga.html.client.map.mapView"
/>
    </client:config>

</config>

```


Possible alternative format

This is not yet implemented, it is just an idea for a refinement of the snippet syntax to help with snippets that use an id

While updating this documentation an alternate format for the snippet occurred to me that may help with snippets that include id's.

Possible future syntax for snippets

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:client="urn:com.cohga.html.client#1.0">

    <client:snippet id="map">
        <view id="com.cohga.html.client.map.mapView">
            <extents>
                <initial minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
                <full minx="327098" miny="5811358"
maxx="351971" maxy="5827675"/>
            </extents>
        </view>
    </client:snippet>

    <client:config id="test1">
        <view ref="map"/>
    </client:config>

    <client:config id="test2">
        <view ref="map"/>
    </client:config>

</config>
```

This has the added bonus that the `snippet` can contain more than one item, although if that's actually useful is as yet unclear.

Client Storage

The Weave server can store information on behalf of the user, for example, the bookmarks that a user saves. This information is stored on the server so that when the user begins a new session it is available for access.

Previously this information was stored using an API provided by the OSGi framework that Weave is built upon. This implementation was file-based with the content stored under the `platform\configuration` directory. The OSGi API appeared to have issues with performance and corruption when used under load, plus the settings would be lost if the configuration directory was cleaned out (which is sometimes required).

To overcome the issues with the OSGi implementation it was replaced by another implementation that uses a database to store the information. By default, the database used is the 'system' database, which is a local HSQLDB database created and managed by Weave for use when the Weave server itself requires a database for internal activities. The content for the system database is stored in the `platform\workspace\.storage` directory.

Alternatively, the storage database can be switched to any data source configured within Weave, with support for Oracle, SQL Server, DB2, Postgres and SQLite.

The reason this is limited to those databases (plus HSQLDB) is that the database storage implementation will create the tables and indexes as required if they do not already exist, and since this process varies depending upon the underlying database, different SQL must be generated and used for each database type (which is different from the rest of Weave which attempts to use generic SQL that can be used on any database, and not just those four).

To change the storage database to a different data source the Java system property `storage.datasource` should be set to provide the id of a data source registered within Weave. In a standard Weave installation, this can be done in the `startup.cmd` and/or `weave-service`.

conf files. Other installations may have other methods to set system properties. If the `storage.datasource` property isn't specified then Weave will use the value `"system.datasource"`, which is the id of the internal HSQLDB database.

startup.cmd line to change storage database, assumes a Weave datasource with the id 'ds.sqlserver'

```
SET JAVA_OPTS=%JAVA_OPTS% -Dstorage.datasource=ds.sqlserver
```

The older OSGi implementation and the newer database implementation both support using the `ustorage` command at the `osgi` prompt to import/export the data they contain.

This allows migration from the older implementation to the newer, or from the newer database implementation that uses the internal system database to an external database.

So to ensure your users settings are not lost you should execute the `'ustorage save'` command before setting the `storage.datasource` value, then `'ustorage load'` after setting it. This will ensure that previous values are migrated to the new database.

Note that if your organisation changes a `userid`, or changes all users `userid`'s, it will break the link between the user and the information stored for that user in Weave, since all the information stored for an individual user is linked to their `userid`.

Currently, the only way to rectify this is to use the `ustorage save` command, edit the exported file by hand to change the `userid`'s and then use `ustorage load` to re-load the updated content.

Manually Creating Tables

These tables will be created automatically by Weave the first time it tries to access them. The following information is provided for informational purposes or, if for some reason, you need to create the tables manually.

SQL Server

```
CREATE TABLE wv_user_nodes(id INT IDENTITY(1,1) NOT NULL, parentid INT NOT NULL, name NVARCHAR(50) NOT NULL, CONSTRAINT wv_user_nodes_PK PRIMARY KEY(id))
```

```
CREATE TABLE wv_user_attributes(id INT NOT NULL, name NVARCHAR(50) NOT NULL, value VARBINARY(MAX) NOT NULL, CONSTRAINT wv_user_attributes_PK PRIMARY KEY (id, name))
```

Oracle

```
CREATE TABLE wv_user_nodes(id INTEGER PRIMARY KEY, parentid INTEGER NOT NULL, name VARCHAR(50) NOT NULL)
```

```
CREATE TABLE wv_user_attributes(id INTEGER NOT NULL, name VARCHAR(50) NOT NULL, value BLOB NOT NULL, CONSTRAINT wv_user_attributes_PK PRIMARY KEY (id, name))
```

```
CREATE SEQUENCE wv_user_nodes_SQ
```

DB2

```
CREATE TABLE wv_user_nodes(id INTEGER GENERATED ALWAYS AS IDENTITY PRIMARY KEY, parentid INTEGER NOT NULL, name VARCHAR(50) NOT NULL)
```

```
CREATE TABLE wv_user_attributes(id INTEGER NOT NULL, name VARCHAR(50) NOT NULL, value BLOB NOT NULL, PRIMARY KEY (id, name))
```

Postgres

```
CREATE TABLE wv_user_nodes(id SERIAL, parentid INTEGER NOT NULL, name VARCHAR(50) NOT NULL, PRIMARY KEY(id))
```

```
CREATE TABLE wv_user_attributes(id INTEGER NOT NULL, name VARCHAR(50) NOT NULL, value BYTEA NOT NULL, PRIMARY KEY (id, name))
```

HSQLDB

```
CREATE CACHED TABLE wv_user_nodes(id INTEGER GENERATED ALWAYS AS IDENTITY PRIMARY KEY, parentid INTEGER NOT NULL, name VARCHAR(50) NOT NULL)
```

```
CREATE CACHED TABLE ww_user_attributes(id INTEGER NOT NULL, name VARCHAR(50) NOT NULL, value OTHER NOT NULL,
PRIMARY KEY (id, name))
```

SQLite

```
CREATE TABLE ww_user_nodes(id INTEGER PRIMARY KEY AUTOINCREMENT, parentid INTEGER NOT NULL, name VARCHAR(50)
NOT NULL)
```

```
CREATE TABLE ww_user_attributes(id INTEGER NOT NULL, name TEXT NOT NULL, value BLOB NOT NULL, PRIMARY KEY (id, name))
Client Plugins
```

Plugins are additional functionality that can be added to a client to provide enhancement to the way the client works, without necessarily requiring any user interactions.

Available Plugins

Name	Id	Description
Map Popup	weave.mappopup	Show a popup when an entity is selected. Available from version 2.5.28.
Map Aliases	weave.aliases	Manages client handling of map engines that are just references to other map engines. <i>Internal plugin that should not be added manually. It will be added automatically if required.</i>
/wiki/spaces/weavedocs1/pages/741769265	weave.edit	Adds functionality required to support spatial editing.
Messaging	weave.messaging	Adds functionality required to support messaging operations between users and from the server to users.
Weave Hub	weave.hub	Adds functionality required to support communications between Weave and third party systems.

Client Plugin Map Popup

The Map Popup tool provides the ability to display a map popup when there is one feature selected on the map. The feature may have been selected by any supported selection method (i.e. Quick Search, graphical selection, attribute selection, etc.)

The tool uses data definitions to supply the data, meaning that the data available to the user can come from any provider, not just the attributes attached to the underlying spatial feature.

This plugin is available from Weave 2.5.28.

ID

weave.mappopup

Properties

name	description
multipleMaximum	If the selection returns more data results than this value, the default being 1, then <code>multipleText</code> will be shown to the user instead of the results. If this many or fewer results are available then they're all shown. This can only be set at the action level. This is used when one spatial feature returns more than one row in the Data Grid.
multipleText	The text to show to the user if more than <code>multipleMaximum</code> results are available. The default text is 'There are too many records at this location'. This can only be set at the action level.
tip	A tip tag must be created for each set of data that's to be made available to the user for display in a map popup. This is only available at the action level.
data	The name of a Data configuration that provides the data for this particular popup. This is only available at the tip level.
data definition	The name of a Data Definition configured that provides the data for this particular popup. This is only available at the tip level.
entity	The Entity that the data definition for the popup is associated with. This is only available at the tip level.
template	Used to apply HTML formatting to the tip for display in the popup.

Notes

- Only one of `data` or `datadefinition` should be set.
- If `datadefinition` is set then `entity` should also be set.
- If `data` is set then `entity` does not need to be set (and will be ignored).

Examples

Simple popup example

```
<view id="com.cohga.html.client.map.mapView">
  ...
  <plugin id="weave.mappopup">
    <multipleMaximum>2</multipleMaximum>
    <multipleText>This features has too many details to
be show</multipleText>
    <tips>
      <tip>
        <data>property_details</data>
      </tip>
      <tip>
        <datadefinition>road_details<
/datadefinition>
        <entity>roads</entity>
      </tip>
    </tips>
  </plugin>
  ...
</view>
```

Example with text formatting

```
<view id="com.cohga.html.client.map.mapView">
  ...
  <plugin id="weave.mappopup">
    <multipleMaximum>5</multipleMaximum>
    <multipleText>This features has too many details to
be shown</multipleText>
    <tips>
      <tip>
        <data>property_details</data>
        <template><![CDATA
[<div><h1><i>ADDRESS DETAILS:</i></h1>{street_number} {street_name}
{street_type}<br>{suburb_name}</div>]]></template>
      </tip>
    </tips>
  </plugin>
  ...
</view>
```

When configuring client items in Weave it's possible to set the text displayed to the user for various items directly, this is generally done by setting a 'label', 'text' or 'tooltip' type of properties for the items.

Previously when these values were set they would completely replace the value displayed for every user, however version 2.4.7 of Weave introduced a way to set these values indirectly and have a different value provided to the user based on their location.

Currently this localisation method is only available for properties set within a client config, so for example it's not yet possible to specify localised labels for an entity or a search using this method.
Note that as of version 2.4.12 this is no longer the case, and any item that's sent to the client will allow this.

The actual values to be displayed to the users are set in a separate client resources configuration item, where you outline the text to be displayed to the user and give each piece of text a unique id, then reference that id where previously you'd use the text directly.

Referencing the text item is performed by prefixing the texts id with % where you'd usually put the text, for example

Original Search panel definition contained within a larger client:config item

```
<view id="com.cohga.html.client.main.searchView">
  <label>Search</label>
  <location>west</location>
</view>
```

would be become

Localised Search panel definition

```
<view id="com.cohga.html.client.main.searchView">
  <label>%search.label</label>
  <location>west</location>
</view>
```

To actually define the text to be used you need to create a separate client `resources` item

Setting the default resource value for search label

```
<client:resources>
  <resource id="search.label">Search</resource>
</client:resources>
```

The client resources item is not embedded within the client config, and is a separate top level configuration item in its own right and would be included at the same level as the client config item.

You can create as many resources configuration items as you like, this allows you to either keep one single large resources section or break them up and locate them close to the items that they're used in.
Keeping them in a separate file and including them into your configuration is also a useful way to reduce clutter.

The above example is intended to show the syntax required to configure text for localisation, and in itself hasn't really provided us with any advantage, since all we've done here is to replace the default text displayed for every client and that would've been easier just to replace the text directly.

However, the real point here is to allow the text to be changed for each user based on their preferred language, and the way we do that is to create another client resources item that contains the alternate text and specify what locale the user should ask for for that text to be used. To do this we duplicate the existing resources item and set a `lang` property for the resources, for example:

Setting alternate resource values for search label

```
<client:resources lang="sv">
  <resource id="search.label">Hitta</resource>
</client:resources>

<client:resources lang="ru">
  <resource id="search.label"></resource>
</client:resources>
```

Here we've set alternate values for the search label for Swedish and Russian users, with no change required to the configuration for the search panel and nothing required from the user.

The original resource item didn't set a `lang` property, so it will be the default value for any properties that haven't specifically been overwritten by locale specific versions, so someone from Italy would see the English label for the search panel. The default value could have just as easily have been specified in Italian and an English specific resource could have been created (using 'en' and the language) if the site is intended primarily for Italian speaking users

Some newer client components will define their own set of resources that can be set to localise text, rather than having to set a property in the configuration of the component and then specify that the property should be replaced by a specific resource.

Alternate configuration layout

If you need to set a lot of resources it can become quite verbose to set each item with a separate `resource` entry in a `resources` item, so as an alternative way of creating resources you can build fullstop separated items by nesting tags, for example

Alternate resource syntax

```
<client:resources>
  <search>
    <label>Search</label>
    <tooltip>
      <title>Search</title>
      <text>Search for an entity</text>
    </tooltip>
  </search>
</client:resources>
```

This will define `search.label`, `search.tooltip.title` and `search.tooltip.text`

Some default resources

The following resources are defined by default within Weave and will be used internally to represent these values. Setting them in a resources item will allow you to change these values where ever they're used within Weave.

name	value
open	Open
close	Close
ok	Ok
cancel	Cancel

generate	Generate
new	New
add	Add
remove	Remove
refine	Refine

Not all component currently utilise these resources, but this will change over time as we go back and update these components to utilise these resources rather than hard coding the text within the components.

Locale override

As of Weave 2.4.11 the user browser local can be specified by adding a 'locale' parameter to the startup url. If this is not set then the browsers default locale will be used to determine any language resources to load.

e.g. <http://server:8080/weave/main.html?locale=sv>
or http://server:8080/weave/index.html?locale=en_AU

Resource property files

Version 2.15.8 of the `com.cohga.client.weave` bundles adds support for resources defined in external properties files, rather than just in `<resources>` tags in config files.

All `.properties` files in a `i18n` directory under the `workspace` directory will be processed for loading of resources. The format of the files should follow the standard Java properties file format, e.g. each line should be of the format `'name=value'`

If file name include an underscore, `'_'`, in the base name then the characters after the underscore will be the locale for the resources, the part of the name before the underscore is ignored. e.g. `test_sv.properties` will contain Swedish resources. No underscore means no locale for the resources so they'll be used as default values.

The file should be stored with UTF-8 encoding, although if the file contains a BOM (byte order mark) it can use other UTF encodings.

This encoding is different from "standard" Java properties files which assume an ISO-8859-1 encoding, but that would mean that a lot of non-ascii characters would have to be encoded using `"\uXXXX"` format in the file which would be too much of a pain to manage for people doing the translations.

Finally, it's possible to zip up the contents of the `i18n` directory and have that used instead of the `i18n` directory itself.

Other text

By default Weave only performs `i18n` substitution of `%` prefixed values that are contained within the client config, so if you need to internationalise other items, for example map layer names, this will not happen by default.

You can enable `i18n` processing of all responses sent from the server to the client, which includes stuff like the layer names from map engines or toc model entries, you need to set the system property `weave.i18n.all` to `true`, by adding `-Dweave.i18n.all=true` to `startup.cmd` and/or `weave-service.conf`.

This way you can setup your underlying map engine (ArcIMS, ArcGIS, etc) to return `"%map.layer.property"`, or for the toc model layer label, for the layer name for a Property layer, and by setting up the appropriate resources for the `map.layer.property` value the layer will appear in a localised name for the user.

Console

By default Windows may not correctly display UTF-8 characters in the OSGi console.

To fix this you need to change the default font for the Command Prompt to a true type font (for example Consolas or Lucinda Console) and then execute `chcp 65001`

This should be done before running `startup.cmd`, from the same command prompt window where you just entered the `chcp` command. Alternatively you could edit the `startup.cmd` file to include the `chcp 65001` command and set the command prompt as the default.

Current Client i18n Resources

The following is a reference for the i18n resources that can be set for a client

Global Resources

The following are global resources that can be set to change the text for common items (although not everything uses them yet)

id	Default
----	---------

open	Open
close	Close
ok	OK
cancel	Cancel
yes	Yes
no	No
new	New
add	Add
remove	Remove
refine	Refine
generate	Generate

id	Default
readytoopen	Ready to open...
clicktoopen	click here to open the link

Client Resources

The following items list the i18n resources that can be set using `client:resources` to alter the display of the component directly. The first line is the component id, the following lines are the names of the resources and their value if not set. When there are 2 client id's they're aliases for the same item.

These are all just set either in a `client:resources` tag or a `resources.properties` file, and they will automatically be used.

For example:

```

<client:resources lang="it">
  <detailsaction>
    <tooltip>
      <title>Ashowa da detaila</title>
      <text>Displaya da detaila view, eh<
/text>
    </tooltip>
  </detailsaction>
</client:resources>

```

or

```

<client:resources lang="it">
  <resource id="detailsaction.tooltip.title">Ashowa da
detaila</resource>
  <resource id="detailsaction.tooltip.text">Displaya da
detaila view, eh</resource>
</client:resources>

```

or in a `resources.properties` file


```

detailsaction.tooltip.title=Ashowa da detaila
detailsaction.tooltip.text=Displaya da detaila view, eh

```

com.cohga.html.client.grid.details

id	Default
detailsaction.tooltip.title	Show Details
detailsaction.tooltip.text	Display a detail view

com.cohga.client.action.legend

id	Default
legendaction.tooltip.title	Map Legend
legendaction.tooltip.text	Display a legend for the map layers

com.cohga.client.action.metadata

id	Default
metadataaction.tooltip.title	Metadata
metadataaction.tooltip.text	Display metadata for the map layers

com.cohga.client.action.toc

id	Default
tocaction.tooltip.title	Layers
tocaction.tooltip.text	Change the display of map layers

com.cohga.client.action.toctoggle

id	Default
toctoggle.tooltip.title	Toggle Layer
toctoggle.tooltip.text	Toggle the display of a map layer

weave.coordZoom

id	Default
coordzoom.tooltip.title	Coordinate Zoom
coordzoom.tooltip.text	Zoom to a given coordinate
coordzoom.bodyText	
coordzoom.zoomWidthLabel	Select the zoom width
coordzoom.zoomProjLabel	Select the projection
coordzoom.windowTitle	Enter Map Coordinate
coordzoom.units.km	kilometre
coordzoom.units.kms	kilometres
coordzoom.units.m	metre
coordzoom.units.ms	metres
coordzoom.units.mi	mile
coordzoom.units.mis	miles
coordzoom.units.ft	foot
coordzoom.units.fts	feet

coordzoom.units.nm	nautical mile
coordzoom.units.nms	nautical miles
coordzoom.lonLabel	Longitude
coordzoom.latLabel	Latitude
coordzoom.xLabel	X
coordzoom.yLabel	Y
coordzoom.degLabel	Deg
coordzoom.minLabel	Min
coordzoom.secLabel	Sec
coordzoom.error.outOfRange	The location could not be shown as it does not reside within the boundary of the Map
coordzoom.error.mapUnitsNotMetres	Map units are in feet or degrees not metres
coordzoom.error.invalidCoordinates	The coordinate values are not valid
coordzoom.error.invalidRange	Value must be between {0} and {1}

com.cohga.html.client.grid.exportAction

id	Default
exportaction.tooltip.title	Export Data
exportaction.tooltip.text	Export the Grid Data into the specified format

com.cohga.html.client.main.gridView com.cohga.html.client.main.identifyView

id	Default
grid.page.displayMsg	Displaying {0} - {1} of {2}
grid.data.emptyMsg	No data to display
grid.page.beforePage	Page
grid.page.afterPage	of {0}
grid.page.first	First Page
grid.page.prev	Previous Page
grid.page.next	Next Page
grid.page.last	Last Page
grid.page.refresh	Refresh
grid.page.items	Items per page

com.cohga.html.client.main.searchView

id	Default
search.error.validation	The search has fields that are required to be entered. Please fix the errors marked in red.
search.reset.tooltip	Reset the Form

weave.snap

id	Default
snap.title	Snapping
snap.tooltip	Enable snapping

weave.snap.settings com.cohga.snap.settings.view

id	Default
snap.settings.title	Snap Settings
snap.settings.tooltip	Alter the current snapping settings
snap.settings.node	Node
snap.settings.vertex	Vertex
snap.settings.edge	Edge
snap.settings.type	Snap Type
snap.settings.entity	Snap Entity
snap.settings.tolerance	Snap Tolerance (m)
snap.settings.save	Save Defaults

weave.emailFeedback

id	Default
feedback.email.tooltip.title	Feedback
feedback.email.tooltip.text	Send feedback to the system administrator
feedback.email.success.title	Feedback Submitted
feedback.email.success.text	Your feedback has been sent
feedback.email.window.title	Email support & feedback
feedback.email.window.send	Send
feedback.email.from.label	E-mail From
feedback.email.from.empty	Enter a valid email address
feedback.email.classification.label	Classification
feedback.email.classification.empty	Select a classification...
feedback.email.subject.label	Subject
feedback.email.subject.empty	Type a subject for the message
feedback.email.message.label	Message
feedback.email.message.empty	Enter the message text

com.cohga.client.panel.edit

id	Default
edit.create.text	Create
edit.create.tooltip	Create a new entity
edit.create.notify	Entity successfully created
edit.update.text	Update
edit.update.tooltip	Edit the selected entity
edit.update.notify	Entity successfully edited
edit.delete.text	Delete
edit.delete.tooltip	Delete the selected entity
edit.delete.notify	Entity successfully deleted
edit.submit.text	Submit
edit.submit.tooltip	Submit changes
edit.cancel.text	Cancel

edit.cancel.tooltip	Cancel changes
edit.cancel.modified.title	Unsaved Changes
edit.cancel.modified.message	Do you want to discard the changes you have made?
edit.polygon.text	Polygon
edit.polygon.tooltip	Add a polygon
edit.line.text	Line
edit.line.tooltip	Add a line
edit.point.text	Point
edit.point.tooltip	Add a point
edit.modify.text	Modify
edit.modify.tooltip	Click on an item to modify it
edit.remove.text	Remove
edit.remove.tooltip	Remove the currently selected item
edit.import.text	Import
edit.import.tooltip	Import geometry from another entity
edit.import.error.tooManyGeometries	There are too many geometries in the source
edit.import.error.tooManyPoints	There are too many points in the source
edit.import.error.tooManyLinestrings	There are too many lines in the source
edit.import.error.tooManyPolygons	There are too many polygons in the source
edit.reset.text	Reset
edit.reset.tooltip	Reset the form fields

com.cohga.html.client.map.bookmark
weave.bookmark

bookmark.tooltip.title	Bookmark
bookmark.tooltip.text	Work with bookmarks
bookmark.name	Bookmark Name
bookmark.saveExtent	Include Extent
bookmark.saveLayers	Include Layers
bookmark.saveRedlines	Include Redlines
bookmark.saveEntity	Include Entity
bookmar.saveSelection	Include Selection
bookmark.emptyText	No bookmarks to display
bookmark.title	Bookmark Manager
bookmark.addBookmark	Add Bookmark
bookmark.manageBookmarks	Manage Bookmarks
bookmark.replacePrompt	Do you want to replace the existing bookmark?
bookmark.removeAllText	Remove All
bookmark.removeAllPrompt	This will delete all of your bookmarks do you want to continue?
bookmark.removeSelectedText	Remove Selected
bookmark.removeSelectedPrompt	This will delete the selected bookmark do you want to continue?
bookmark.publish	Publish

bookmark.published	Published
bookmark.private	Private
bookmark.mamage.title	System Bookmar Manager
bookmark.id	Id
bookmark.user	User
bookmark.update	Update
bookmark.editBookmark	Edit Boomkark
bookmark.publishBookmark	Publish Bookmark
bookmark.edit	Edit
bookmark.created	Created
bookmark.time	Executed
bookmark.count	Count
bookmark.category	Category
bookmark.group	Group
bookmark.shareBookmark	Share Bookmark
bookmark.share	Share
bookmark.shared	Shared
bookmark.description	Description
bookmark.descriptionHelp	To help keep track of bookmark
publishbookmark.tooltip.title	Published Bookmarks
publishbookmark.tooltip.text	Open a published bookmark
managebookmark.tooltip.title	Manage Bookmarks
managebookmark.tooltip.text	Manage bookmarks that have been published or shared

com.cohga.client.actions.redline
weave.redline

id	Default
redline.tooltip.title	Redline
redline.tooltip.text	Add elements to the map
redline.title	Redline
redline.name	Redline Name
redline.transparency	Transparency
redline.none	none
redline.load.tooltip.title	Load
redline.load.tooltip.text	Load and manage your saved redlines
redline.load.title	Manage Redlines
redline.load.noRedlines	No redlines are currently available
redline.load.removeAllPrompt	This will delete all of your saved redlines, do you want to continue?
redline.load.removeSelectedPrompt	This will delete the selected saved redline, do you want to continue?
redline.load.replacePrompt	This will replace all of your existing redline elements, do you want to continue?
redline.save.tooltip.title	Save

redline.save.tooltip.text	Save the current set of redlines
redline.save.title	Save Redline
redline.save.prompt	Please enter a name to save the redline elements
redline.save.replacePrompt	Do you want to replace the existing redline?
redline.circle.tooltip.title	Circle
redline.circle.tooltip.text	Add a circle to the map
redline.defaults.tooltip.title	Save Settings
redline.defaults.tooltip.text	Save the current settings as the defaults
redline.delete.tooltip.text	Delete the currently selected redline element
redline.delete.confirmTitle	Delete
redline.delete.confirmPrompt	This will delete the selected redline element, do you want to continue?
redline.deleteAll.tooltip.title	Delete All
redline.deleteAll.tooltip.text	Delete all redline elements
redline.deleteAll.confirmTitle	Delete All
redline.deleteAll.confirmPrompt	This will delete all of your redline elements, do you want to continue?
redline.entity.tooltip.title	Get
redline.entity.tooltip.text	Create a redline item from the currently selected entity
redline.fillMenu.tooltip.title	Fill Settings
redline.fillMenu.tooltip.text	Change the fill colour and transparency of the selected feature
redline.fontMenu.tooltip.title	Font Settings
redline.fontMenu.tooltip.text	Change the Font colour and style of the selected features text
redline.fontMenu.family.label	Family
redline.fontMenu.family.serif	Serif
redline.fontMenu.family.sansserif	Sans-Serif
redline.fontMenu.family.monospace	Monospace
redline.fontMenu.size.label	Size
redline.fontMenu.size.small	Small
redline.fontMenu.size.medium	Medium
redline.fontMenu.size.large	Large
redline.fontMenu.weight.label	Weight
redline.fontMenu.weight.lighter	Lighter
redline.fontMenu.weight.normal	Normal
redline.fontMenu.weight.bold	Bold
redline.fontMenu.weight.bolder	Bolder
redline.marker.tooltip.title	Marker
redline.marker.tooltip.text	Add a marker to the map
redline.markerMenu.tooltip.title	Marker Settings
redline.markerMenu.tooltip.text	Change the marker colour
redline.markerMenu.red	Red
redline.markerMenu.green	Green
redline.markerMenu.blue	Blue

redline.modify.tooltip.title	Modify
redline.modify.tooltip.text	Click on the a redline to modify
redline.point.tooltip.title	Point
redline.point.tooltip.text	Add a point to the map
redline.polygon.tooltip.title	Polygon
redline.polygon.tooltip.text	Add a polygon to the map
redline.polyline.tooltip.title	Polyline
redline.polyline.tooltip.text	Add a polyline to the map
redline.rectangle.tooltip.title	Rectangle
redline.rectangle.tooltip.text	Add a rectangle to the map
redline.square.tooltip.title	Square
redline.square.tooltip.text	Add a square to the map
redline.strokeMenu.tooltip.title	Line Settings
redline.strokeMenu.tooltip.text	Change the line colour and transparency of the selected feature
redline.strokeMenu.width.label	Line Width
redline.strokeMenu.style.label	Line Style
redline.export.tooltip.title	
redline.export.tooltip.text	Export all redlines to Shapefile
redline.export.title	Export Redline
redline.export.projection	Projection
redline.export.description	Description
redline.export.export	Export
redline.export.clear	Clear
redline.export.kml.tooltip.title	Export to KML
redline.export.kml.tooltip.text	Export all redlines to KML
redline.export.kml.title	Export Redline
redline.layers.title	Organise Redlines
redline.layers.tooltip.title	Organise
redline.layers.tooltip.text	List and organise redlines
redline.layers.area	Area
redline.layers.length	Length

Document Management System

id	Default
dms.selected.tooltip.title	Show Documents
dms.selected.tooltip.text	Display any documents attached to the active selected entities
dms.all.tooltip.title	Show Documents for all
dms.all.tooltip.text	Display any documents attached to all selected entities
dms.remove.title	Remove Document
dms.remove.prompt	Are you sure you want to remove this document?
dms.remove.success	The document has been removed

dms.remove.failure	The document could not be removed
dms.upload.tooltip.title	Upload Document
dms.upload.tooltip.text	Upload a document and attach it to the selected entity
dms.upload.selectfile	Select a file
dms.upload.file	File
dms.upload.description	Description
dms.upload.upload	Upload
dms.upload.uploading	Uploading your file...
dms.upload.success	Successfully uploaded file to the server
dms.upload.failure	Upload to the server failed
dms.text.display	Display
dms.text.remove	Remove

Help

id	Default
help.tooltip.title	Help
help.tooltip.text	Click to open the Help window
help.title	Help
help.filter	Filter
help.expandAll	Expand All
help.collapseAll	Collapse All

Updates for Weave 2.5

weave.buffer (or com.cohga.html.client.actions.bufferAction)

id	Default
weave.buffer.tooltip.title	Buffer
weave.buffer.tooltip.text	Create a Buffer around the selected features
weave.buffer.window.title	Buffer the Selected Features
weave.buffer.layer.label	Layer
weave.buffer.distance.label	Distance (m)
weave.buffer.distance.invalid	Please enter a valid Buffer Distance
weave.buffer.distance.blank	Please enter a Buffer Distance

weave.coords

id	Default
weave.coords.tooltip.title	Coordinate Display
weave.coords.tooltip.text	Display current coordinates in a different projection
weave.coords.window.title	Coordinate Display

weave.fixedZoomIn

id	Default
weave.fixedZoomIn.tooltip.title	Zoom In
weave.fixedZoomIn.tooltip.text	Zoom in on the Map

weave.fixedZoomOut

id	Default
weave.fixedZoomOut.tooltip.title	Zoom Out
weave.fixedZoomOut.tooltip.text	Zoom out on the Map

weave.fullExtent (or com.cohga.html.client.map.fullExtentAction)

id	Default
weave.fullExtent.tooltip.title	Full Extent
weave.fullExtent.tooltip.text	Return to the Full Extent

weave.spatialIdentify (or com.cohga.html.client.map.spatialIdentify)

id	Default
weave.spatialIdentify.tooltip.title	Spatial Identify
weave.spatialIdentify.tooltip.text	Click on the map and drag a circle to identify features
weave.spatialIdentify.window.title	Spatial Identify

weave.identify (or com.cohga.html.client.actions.identifyAction)

id	Default
weave.identify.tooltip.title	Identify
weave.identify.tooltip.text	Click on the map and drag a circle to identify the features
weave.identify.window.title	Identify

weave.initialExtent (or com.cohga.html.client.map.initialExtentAction)

id	Default
weave.initialExtent.tooltip.title	Initial Extent
weave.initialExtent.tooltip.text	Return to the Initial Extent

weave.intersect (or com.cohga.html.client.actions.intersect)

id	Default
weave.intersect.tooltip.title	Intersect
weave.intersect.tooltip.text	Select features in this layer that Intersect selected features in another
weave.intersect.layer.label	Layer
weave.intersect.window.title	Intersect the Selected Features

weave.maptips

id	Default
weave.maptips.tooltip.title	MapTips
weave.maptips.tooltip.text	Hover the mouse over a location and get information about the point
weave.maptips.multipleText	There are too many features at that location

weave.gridWindow (or com.cohga.html.client.actions.gridWindowAction)

id	Default
weave.gridWindow.tooltip.title	Grid
weave.gridWindow.tooltip.text	Generate a Grid Report for the Active Entity
weave.gridWindow.window.title	Data

weave.grid.export (or com.cohga.html.client.grid.exportAction)

id	Default
weave.grid.export.tooltip.title	Export Data
weave.grid.export.tooltip.text	Export the Grid Data into the specified format

weave.grid.pan (or com.cohga.html.client.grid.panAction)

id	Default
weave.grid.pan.tooltip.title	Pan
weave.grid.pan.tooltip.text	Pan to the selected feature in the Grid
weave.grid.pan.text	Pan

weave.grid.refine (or com.cohga.html.client.grid.refine)

id	Default
weave.grid.refine.tooltip.title	Refine Selection
weave.grid.refine.tooltip.text	Refine the current selection in the Grid
weave.grid.refine.text	Refine

weave.grid.remove (or com.cohga.html.client.grid.removeAction)

id	Default
weave.grid.remove.tooltip.title	Remove Selection
weave.grid.remove.tooltip.text	Remove the current selection in the Grid
weave.grid.remove.text	Remove

weave.grid.zoom (or com.cohga.html.client.grid.zoomAction)

id	Default
weave.grid.zoom.tooltip.title	Zoom To
weave.grid.zoom.tooltip.text	Zoom to the selected feature in the Grid
weave.grid.zoom.text	Zoom

weave.measurePolygon (or com.cohga.client.actions.map.measurePolygon)

id	Default
weave.measurePolygon.tooltip.title	Measure Polygon
weave.measurePolygon.tooltip.text	Measure a polygon on the map
weave.measurePolygon.window.title	Measure Polygon
weave.measurePolygon.segmentLength	Segment Length
weave.measurePolygon.bearing	Bearing
weave.measurePolygon.totalArea	Total Area (Approx)

weave.measurePolyline (or com.cohga.client.actions.map.measurePolyline)

id	Default
weave.measurePolyline.tooltip.title	Measure Polyline
weave.measurePolyline.tooltip.text	Measure a polyline on the map
weave.measurePolyline.window.title	Measure Polyline
weave.measurePolyline.segmentLength	Segment Length
weave.measurePolyline.bearing	Bearing

weave.measurePolyline.totalLength	Total Length
-----------------------------------	--------------

weave.nextExtent (or com.cohga.html.client.map.nextExtentAction)

id	Default
weave.nextExtent.tooltip.title	Next Extent
weave.nextExtent.tooltip.text	Go to the Next Extent

weave.previousExtent (or com.cohga.html.client.map.previousExtentAction)

id	Default
weave.previousExtent.tooltip.title	Previous Extent
weave.previousExtent.tooltip.text	Go to the Previous Extent

weave.pan (or com.cohga.html.client.map.panAction)

id	Default
weave.pan.tooltip.title	Pan
weave.pan.tooltip.text	Click and drag the map to a new location

weave.projections (or com.cohga.client.actions.projAction)

id	Default
weave.projections.tooltip.title	Coordinate Display
weave.projections.tooltip.text	Display current coordinates in a different projection
weave.projections.window.title	Coordinate Display

weave.refreshMap (or com.cohga.html.client.map.refreshMapAction)

id	Default
weave.refreshMap.tooltip.title	Refresh
weave.refreshMap.tooltip.text	Refresh the Map

weave.clear (or com.cohga.html.client.map.select.clearAction)

id	Default
weave.clear.tooltip.title	Clear Selection
weave.clear.tooltip.text	Click to clear the current selection

weave.clearAll (or com.cohga.html.client.map.select.clearAllAction)

id	Default
weave.clearAllAction.tooltip.title	Clear All Selections
weave.clearAllAction.tooltip.text	Click to clear all selections

weave.selectZoom (or com.cohga.html.client.map.select.zoomAction)

id	Default
weave.selectZoom.tooltip.title	Zoom Selection
weave.selectZoom.tooltip.text	Click to zoom to the current selection
weave.selectZoom.errorTitle	Information
weave.selectZoom.errorMessage	No Spatial Features found matching selection

weave.selectZoomAll (or com.cohga.html.client.map.select.zoomAllAction)

--	--

id	Default
weave.selectZoomAll.tooltip.title	Zoom All Selection
weave.selectZoomAll.tooltip.text	Click to zoom to the current selection of all layers

weave.spatialselect (or com.cohga.html.client.actions.spatialSelectAction)

id	Default
weave.spatialselect.tooltip.title	Spatial Select
weave.spatialselect.tooltip.text	Select entities based on their spatial relationship to other entities
weave.spatialselect.window.title	Selected Entities
weave.spatialselect.label.withindistanceof	are within a distance of
weave.spatialselect.label.intersect	intersect
weave.spatialselect.label.closestto	is closest to
weave.spatialselect.label.nextto	are next to
weave.spatialselect.label.target	Target
weave.spatialselect.label.distance	Distance (m)
weave.spatialselect.text.distance.invalid	Please enter a valid Buffer Distance
weave.spatialselect.text.distance.blank	Please enter a Buffer Distance
weave.spatialselect.label.type	Type
weave.spatialselect.label.source	Source

weave.zoomIn (or com.cohga.html.client.map.zoominAction)

id	Default
weave.zoomIn.tooltip.title	Zoom In
weave.zoomIn.tooltip.text	Click on the map and drag a rectangle

weave.zoomOut (or com.cohga.html.client.map.zoomoutAction)

id	Default
weave.zoomOut.tooltip.title	Zoom Out
weave.zoomOut.tooltip.text	Click on the map and drag a rectangle

weave.toc.clearSelection

id	Default
weave.toc.clearSelection.text	Clear Selection

weave.toc.collapseAll

id	Default
weave.toc.collapseAll.tooltip.title	Collapse All
weave.toc.collapseAll.tooltip.text	Collapse all the Groups to hide all sub groups and layers
weave.toc.collapseAll.text	

weave.toc.expandAll

id	Default
weave.toc.expandAll.tooltip.title	Expand All
weave.toc.expandAll.tooltip.text	Expand all the Groups to hide all sub groups and layers
weave.toc.expandAll.text	

weave.toc.load

id	Default
weave.toc.load.tooltip.title	Load
weave.toc.load.tooltip.text	Retrieve the previously saved state
weave.toc.load.text	

weave.toc.save

id	Default
weave.toc.save.tooltip.title	Save
weave.toc.save.tooltip.text	Store the current state
weave.toc.save.text	

weave.toc.lockLayer

id	Default
weave.toc.lockLayer.text	Lock Layer
weave.toc.unlockLayer.text	Unlock Layer

weave.toc.selectLayer

id	Default
weave.toc.selectLayer.text	Select Layer

weave.toc.filter

id	Default
weave.toc.filter.emptyText	Filter
weave.toc.filter.filteringText	Filtering

weave.toc.zoomToLayer

id	Default
weave.toc.zoomToLayer.text	Zoom To Layer

weave.toc.zoomToMaxScale

id	Default
weave.toc.zoomToMaxScale.text	Zoom to Max. Scale

weave.toc.zoomToMinScale

id	Default
weave.toc.zoomToMinScale.text	Zoom to Min. Scale

weave.legendWindow (or com.cohga.client.action.legend)

id	Default
weave.legendWindow.tooltip.title	Map Legend
weave.legendWindow.tooltip.text	Display a legend for the map layers
weave.legendWindow.window.title	Legend

weave.metadataWindow (or com.cohga.client.action.metadata)

id	Default

weave.metadataWindow.tooltip.title	Metadata
weave.metadataWindow.tooltip.text	Display metadata for the map layers
weave.metadataWindow.window.title	Metadata

weave.tocMenu (or com.cohga.client.action.toc)

id	Default
weave.tocMenu.tooltip.title	Layers
weave.tocMenu.tooltip.text	Change the display of map layers

weave.tocWindow (or weave.tocWindowAction)

id	Default
weave.tocWindow.tooltip.title	Table of Contents
weave.tocWindow.tooltip.text	View the Table of Contents in a new floating window
weave.tocWindow.window.title	Table of Contents

weave.toggleToc (or com.cohga.client.action.toctoggle)

id	Default
weave.toggleToc.tooltip.title	Toggle Layer
weave.toggleToc.tooltip.text	Toggle the display of a map layer

weave.legend (or com.cohga.client.panel.legend)

id	Default
weave.legend.tooltip.title	Legend
weave.legend.tooltip.text	Display a legend for the map layers
weave.legend.emptyText	No legend is currently available

licence agreement

id	Default
weave.licence.title	Licence Agreement
weave.licence.text	No licence text defined. Please add one to the licence

paging toolbar

id	Default
weave.paging.itemsPerPage	Items per page

grid

id	Default
grid.filter.text	Filter
grid.filter.tooltip	Type text to filter and press Enter

weave.mapctrl.scalebar

id	Default
weave.mapctrl.scalebar.tooltip.title	Scalebar Units
weave.mapctrl.scalebar.tooltip.text	Right click to change the Units
weave.mapctrl.scalebar.label	Scale 1:
weave.mapctrl.scalebar.metres	Metres

weave.mapctrl.scalebar.kilometres	Kilometres
weave.mapctrl.scalebar.feet	Feet
weave.mapctrl.scalebar.miles	Miles

weave.entitySelectorView

id	Default
weave.entitySelectorView.title	Entities
weave.entitySelectorView.tooltip.title	Entities
weave.entitySelectorView.tooltip.text	Manage the current entities
weave.entitySelectorView.other.text	Other
weave.entitySelectorView.entity.text	Entity
weave.entitySelectorView.size.text	Size
weave.entitySelectorView.group.text	Group

weave.grid (or com.cohga.html.client.main.gridView)

id	Default
weave.grid.title	Data

weave.report (or com.cohga.html.client.main.reportView)

id	Default
weave.report.title	Report

weave.search (or com.cohga.html.client.main.searchView)

id	Default
weave.search.title	Search

weave.toc (or com.cohga.html.client.map.tocView)

id	Default
weave.toc.title	Layers

selection

id	Default
weave.select.new.tooltip.title	New Selection
weave.select.add.tooltip.title	Add Selection
weave.select.remove.tooltip.title	Remove Selection
weave.select.refine.tooltip.title	Refine Selection
weave.select.rect.tooltip.text	Click on the map and drag a rectangle
weave.select.circle.tooltip.text	Click on the map and drag a circle
weave.select.polyline.tooltip.text	Click on the map and drag a polyline
weave.select.polygon.tooltip.text	Click on the map and drag a polygon
weave.select.point.tooltip.text	Click on the map to select a location

weave.uploadFile (or com.cohga.spatial.upload.uploadFile)

id	Default
uploadFile.tooltip.title	Upload

uploadFile.tooltip.text	Upload a file as a layer on the map
uploadFile.windowTitle	Upload File
uploadFile.addText	Add
uploadFile.addTooltip	Add local files
uploadFile.clearAllText	Clear
uploadFile.clearAllTooltip	Clear all uploading files
uploadFile.uploadText	Upload
uploadFile.uploadTooltip	Upload all files
uploadFile.stopText	Stop
uploadFile.stopTooltip	Stop uploading
uploadFile.step1PromtText	<p>Please select files</p>
uploadFile.supportedFormats	<p>Suported formats are:<p> - Keyhole Markup (.kml or .kmz) - Shapefile (.shp, .shx, .dbf) - Comma Separated (.csv or .txt) You can also include - Style Layer Description file (.sld) - Projection file (.prj) for shapefiles and comma separated files</p>
uploadFile.tocGroupLabel	Upload
uploadFile.tocGroupDescription	Uploaded Layers
uploadFile.inputLabel	Input
uploadFile.pointsLabel	Points
uploadFile.linesLabel	Lines
uploadFile.polygonsLabel	Polygons
uploadFile.markLabel	Mark
uploadFile.fillLabel	Fill
uploadFile.strokeLabel	Stroke

weave.zipnship

id	default
zipnship.tooltip.title	Zip'n'Ship
zipnship.tooltip.text	Extract the selected features
zipnship.menu.active.label	Active Layer
zipnship.menu.active.tooltip	Only export the selected features from the active entity
zipnship.menu.all.label	All Layers
zipnship.menu.all.tooltip	Export the selected features from all entities that have a selection
zipnship.menu.export.label	Export
zipnship.menu.export.tooltip	Perform the export

weave.zipnshipenvelope

id	default
zipnshipenvelope.tooltip.title	Zip'n'Ship

zipnshipenvelope.tooltip.text	Extract the features from the current extent
-------------------------------	--

Other pending changes

id	Default
datacombo.emptyText	No Data Available
searchcombo.emptyText	No Searches Available
indexcombo.emptyText	Quick search
indexcombo.loading	Searching...
indexcombo.tooltip.title	Quick Search
indexcombo.tooltip.text	Type here to search
indexcombo.tooltip2.title	Redisplay
indexcombo.tooltip2.text	Redisplay search results
indexcombo.tooltip3.title	Reset
indexcombo.tooltip3.text	Reset search results
details.tooltip.title	Details
details.tooltip.text	Display details about a selected entity
details.emptyText	No details to display
details.loading	Loading...
details.loadError	Unable to load data
entityselector.tooltip.title	Entities
entityselector.tooltip.text	Manage the current entities
entityselector.defaultGroupName	Other
entityselector.entityLabel	Entity
entityselector.sizeLabel	Size
entityselector.groupLabel	Group
grid.filter.text	Filter
grid.filter.tooltip	Type text to filter and press Enter

Export to KML

Client User Attributes

From Weave 2.6.5 onwards it's possible to customise a client based on attributes associated with the user.

User attributes are custom values that are associated with a user, generally being read from a database table but can be provided in other ways. These values can be referenced in a client configuration to provide the value for any attribute in the configuration.

For example, if a user, or group of users, should be presented with an initial Table of Contents (ToC) for the map when they start a Weave client then the "toc" tag in that client configuration can be setup in such a way that the value for the name of the toc model to use can be provided by a user attribute.

```
<toc ref="{user.toc}" />
```

Note this example assumes that a user attribute called "toc" has been associated with *all* users.

In the above example if the value for the "toc" attribute for the current user is "main" then the above would be the equivalent of:

```
<toc ref="main"/>
```

Default Values

If you can't, or don't want to, set a value for an attribute for all users you can specify a default value for the attribute directly in the config. For example, if we wanted to use "main" as the default toc model, but have some users use something different, then you would use the following format:

```
<toc ref="\${user.toc|main}"/>
```

If one user had the value "custom" for the toc attribute, it would be equivalent to:

```
<toc ref="custom"/>
```

If another user did not have a toc attribute at all it would be equivalent to:

```
<toc ref="main"/>
```

Converting Values

Weave will try and guess the type of the value, either the attribute value or the default value, and convert the value to that type using the same rules that the config file reader uses when initially reading the values from the XML files. As such, there may be times when the conversion is incorrect. For example, if the value of the attribute is "1234", Weave will convert that to the number 1234. So:

```
<tag value="\${user.number}"/>
```

would result in the following object:

```
{tag: {value: 1234}}
```

If the value is supposed to be a string and not a number, then you can override this by appending a colon followed by a format. For example, if the value of the `user.number` attribute is 1234 but you need `tag.value` to be a string you can explicitly specify the type of the value. So:

```
<tag value="\${user.number:string}"/>
```

would result in the following object:

```
{tag: {value: "1234"}}
```

This is the same but provides a default in case the user does not have a `number` attribute:

```
<tag value="\${user.number|1234:string}"/>
```

Note that type conversion only applies when the entire value is replaced by the user attribute. If the user attribute replacement is only part of the final value then it will always be treated as a string and the format will be ignored. For example:

```
<item text="Number: ${user.number:float}"/>
```

would result in the following object:

```
{item: {text: "Number: 1234"}}
```

Pool

The pool configuration provides a way to tune the setting used to setup the connection pooling for other configuration items. For example a [Data Source](#) and [ArcGIS Server Map Engine](#).

The pool tag itself is not used as a top level configuration item but is instead embedded within other tags.

A pool provides a number of configurable parameters:

- `maxActive` controls the maximum number of objects that can be borrowed from the pool at one time. When non-positive, there is no limit to the number of objects that may be active at one time. When `maxActive` is exceeded, the pool is said to be exhausted. If `whenExhaustedAction` is 'grow' this value is not used.
 - `maxIdle` controls the maximum number of objects that can sit idle in the pool at any time. When negative, there is no limit to the number of objects that may be idle at one time.
 - `whenExhaustedAction` specifies the behavior when the pool is exhausted:
 - When `whenExhaustedAction` is 'fail', pool will throw an Exception.
 - When `whenExhaustedAction` is 'grow', pool will create a new object and return it, essentially making `maxActive` meaningless.
 - When `whenExhaustedAction` is 'block', pool will block. If a positive `maxWait` value is supplied, the pool will block for at most that many milliseconds, after which an Exception will be thrown. If `maxWait` is non-positive, the server method will block indefinitely. You *should* always set a positive `maxWait` value if you use 'block'.
 - When `testOnBorrow` is set, the pool will attempt to validate each object before it is returned from the pool. Objects that fail to validate will be dropped from the pool, and a different object will be borrowed.
 - When `testOnReturn` is set, the pool will attempt to validate each object before it is returned to the pool. Objects that fail to validate will be dropped from the pool.
- Optionally, one may configure the pool to examine and possibly evict objects as they sit idle in the pool. This is performed by an "idle object eviction" thread, which runs asynchronously. The idle object eviction thread may be configured using the following attributes:
- `timeBetweenEvictionRunsMillis` indicates how long the eviction thread should sleep before "runs" of examining idle objects. When non-positive, no eviction thread will be launched.
 - `minEvictableIdleTimeMillis` specifies the minimum amount of time that an object may sit idle in the pool before it is eligible for eviction due to idle time. When non-positive, no object will be dropped from the pool due to idle time alone.
 - `testWhileIdle` indicates whether or not idle objects should be validated. Objects that fail to validate will be dropped from the pool.
 - `maxAgeMillis` is only available for ArcGIS map engine pools and specifies how long Weave should keep and connection to the server open for, it should be less than the "The maximum time a client can use a service" setting in ArcGIS Server. [ArcGIS Server Settings - Pooling and Processes](#)

When Weave makes a request to map engine or database it needs to establish a connection to the resource and creating a connection each time Weave needs to make an interaction with the resource will cause a delay and use additional memory as the number of connections increases, to mitigate this Weave uses an object 'pool' where each request shares a pre-allocated connection object with other requests.

Using a pre-allocated collection of resource connections provides two benefits to Weave, firstly it means that the overhead of creating a new connection on each request is removed, resulting in better response times for the user.

And, secondly, it means that Weave can use less resources on the backend server by limiting the total number of connections that can be allocated at once, so for example if 50 requests are being processed at the same time then without connection pooling Weave would need to create 50 connections to the backend resource, but with a connection pool that's limited to a maximum of 10 connections then only 10 connections will be created to the backend service. The downside of this is that user requests will have to wait until a connection becomes available in the pool before it can proceed.

Determining the correct setting for a connection pool is not something that can be easily done without knowledge about how the backend service is configured and possibly other information about the general IT infrastructure.

Namespace

```
urn:com.cohga.server.pool#1.0
```

Tags

pool

Properties

Name	Type	Required	Default	Description
maxActive	number	no	8	The maximum number of objects that can be borrowed from the pool at one time. The cap on the total number of active instances from my pool. Use a negative value for an infinite number of instances.
maxIdle	number	no	8	The maximum number of idle connections in the pool. The cap on the number of "idle" instances in the pool. Use a negative value to indicate an unlimited number of idle instances.
maxWait	number	no	1000	The maximum amount of time to wait, in milliseconds, for an object when the pool is exhausted and whenExhaustedAction is 'block' (otherwise ignored).
minIdleTimeMillis	number	no	18000	The minimum number of milliseconds an connection can sit idle in the pool before it is eligible for eviction. Sets the minimum amount of time an object may sit idle in the pool before it is eligible for eviction by the idle object evictor (if any). When non-positive, no objects will be evicted from the pool due to idle time alone.
minIdle	number	no	0	The minimum number of idle connections in the pool. Sets the minimum number of objects allowed in the pool before the evictor thread (if active) spawns new objects. (Note no objects are created when: numActive + numIdle >= maxActive).
numTestsPerEvictionRun	number	no	3	The number of idle objects to examine per run within the idle object eviction thread (if any). Sets the max number of objects to examine during each run of the idle object evictor thread (if any). When a negative value is supplied, $\text{ceil}(\text{numIdle})/\text{abs}(\text{numTestsPerEvictionRun})$ tests will be run. I.e., when the value is -n, roughly one nth of the idle objects will be tested per run.
softMinEvictableIdleTimeMillis	number	no	-1	The minimum number of milliseconds an connection can sit idle in the pool before it is eligible for eviction with the extra condition that at least "minIdle" amount of connections remain in the pool. Sets the minimum amount of time an object may sit idle in the pool before it is eligible for eviction by the idle object evictor (if any), with the extra condition that at least "minIdle" amount of object remain in the pool. When non-positive, no objects will be evicted from the pool due to idle time alone.
testOnBorrow	boolean	no	false	Should the connection be checked for validity when being borrowed from the pool. When true, objects will be validated before being returned to the pool. If the object fails to validate, it will be dropped from the pool, and we will attempt to borrow another.
testOnReturn	boolean	no	false	Should the connection be checked for validity when being returned to the pool. When true, objects will be validated before being returned to the pool.
testWhileIdle	boolean	no	false	Should the connection be checked for validity when just sitting in the pool. When true, objects will be validated by the idle object evictor (if any). If an object fails to validate, it will be dropped from the pool.
timeBetweenEvictionRunsMillis	number	no	-1	Sets the number of milliseconds to sleep between runs of the idle connection evictor. Sets the number of milliseconds to sleep between runs of the idle object evictor thread. When non-positive, no idle object evictor thread will be run.
whenExhaustedAction	'fail', 'grow' or 'block'	no	block	Sets the action to take when the pool is exhausted (the maximum number of "active" objects has been reached)
lifo	boolean	no	false	"last in first out" if true then the last connection used should be the next one allocated from the pool, or if false then the oldest connection will be the next one given out. Prior to Weave 2.5.21 this setting was not available and defaulted to true.

Sub-tags

None

Content

None

Examples

```

<pool:pool>
  <minIdle>4</minIdle>
  <maxIdle>8</maxIdle>
  <testOnBorrow>true</testOnBorrow>
  <testOnReturn>false</testOnReturn>
  <testWhileIdle>true</testWhileIdle>
  <timeBetweenEvictionRunsMillis>60000<
/timeBetweenEvictionRunsMillis>
  <minEvictableIdleTimeMillis>300000</minEvictableIdleTimeMillis>
  <whenExhaustedAction>grow</whenExhaustedAction>
</pool:pool>

```

DMS

Weave has the ability to retrieve a list of documents associated with a given entity, display that list to the user, and allowing the user to select a document from that list to be displayed.

The determination of what documents are associated with an entity and the method of displaying the documents are provided by additional modules that are specific to different document management systems.

Currently there is one implementation available that will retrieve a list of documents from a database table and open those document in a new browser window.

- [Database DMS](#)

Installation

Installation of the DMS functionality requires at least four bundles, three that provide the core functionality and at least one for each type of DMS to be used a backend.

The core functionality is provided by

com.cohga.client.dms:: The actions that can be added to the client to interact with the DMS

com.cohga.server.dms.core:: The server side functionality that accepts the client action requests and communicates with the configured document management systems

com.cohga.server.dms:: The API that needs to be implemented by a document management system to interact with Weave

Configuration

Server

Configuration of the server side of the document management system integration is dependant upon the document management system that's in use, and the documentation for each plugin should be consulted.

Client

Regardless of the backend document management system configuring the client to display the related documents is the same.

There are two new actions available to the client when the core DMS bundle is active, they are:

com.cohga.dms.ShowSelected:: Show documents attached to the selected entities for the active entity

com.cohga.dms.ShowAllSelected:: Show the documents attached to the selected entities for all entities

These can be added to the client by adding one or both of the actions to a toolbar in the client.

```

<client:config id="main">
  ...
  <toolbar>
    <item action="com.cohga.dms.ShowSelected"/>
    <item action="com.cohga.dms.ShowAllSelected"/>
    ...
  </toolbar>
  ...
</client:config>

```

Database DMS

This document management system module retrieves a list of document associated with an entity from a database.

The information required when configuring the DB DMS is provided by a data definition, the `dms` tag links an entity to the data definition which is then used to retrieve a list of attributes to display to the user along with a URL that can be used to display the document in question.

Any number of attributes can be retrieved from the database for display to the user.

Namespace

`urn:com.cohga.server.dms.provider.db#1.0`

Tags

`dms`

Properties

Name	Type	Required	Description
<code>id</code>	string	yes	unique identifier
<code>entity</code>	ref urn:com.cohga.server.entity#1.0	yes	The entity that this DMS will provide documents for
<code>name</code>	string	yes	The name of this DMS
<code>datadefinition</code>	ref urn:com.cohga.server.data#1.0:datadefinition	yes	A reference to a data definition that will supply the attributes and URL link for each document
<code>acl</code>	ref urn:com.cohga.server.acl#1.0:acl	no	A reference to an ACL to attach to the dms. Only available since Weave 2.5.16.

Sub-tags

Name	Type	Cardinality
<code>acl</code>	urn:com.cohga.server.acl#1.0:acl	0..1

Content

None

Notes

- The `name` is used because there may be more than one DMS attached to an entity type and this helps the user differentiate between them

Examples

A simple example that uses a table that has a description and url column

```
<data:datadefinition id="property_doclink">
  <datasourcedataconnection datasource="docsource" table="
PROPERTY_DOCLINK" key="PID">
    <parameter column="description"/>
    <parameter column="url"/>
  </datasourcedataconnection>
</data:datadefinition>

<dbdms:dms id="old_docs">
  <name>Uploaded Documents</name>
  <entity>property</entity>
  <datadefinition>property_doclink</datadefinition>
</dbdms:dms>
```

Creating a list of documents to display properties in google maps.

The data definition constructs the url by concatenating a the first part of a google maps search to a pre-formatted address column from the database.

```
<data:datadefinition id="property_google">
  <datasourcedataconnection datasource="docsource" table="
PROPERTY_DOCLINK" key="PID">
    <parameter name="url" type="url" text="Display in
Google Maps" url="'http://maps.google.com.au/?q=' |address" />
    <parameter column="address" />
  </datasourcedataconnection>
</data:datadefinition>

<dbdms:dms id="google">
  <name>Google Maps</name>
  <entity>property</entity>
  <datadefinition>property_google</datadefinition>
</dbdms:dms>
```

DMS related bundles

There are a handful of com.cohga.server.dms and a couple of com.cohga.client.dms bundles, they provide the support for document retrieval and upload, and are each pretty small.

- com.cohga.server.dms
 - Base DMS API for retrieving documents attached to an entity, which can be used with anything that can provide a link to a document, not just our internal one.
- com.cohga.server.dms.core
 - Base implementation that provides the required support for the above API. Using the API to retrieve documents, building the information required by the client, handing requests, etc.
- com.cohga.client.dms
 - Provides the base client side support for retrieving documents attached to an entity. The buttons to retrieve documents for the selected entities.
- com.cohga.server.dms.provider.file
 - A file system based document provider that can handle documents stored in a structure on the file system, It builds an internal URL that links to the files and provides that to the client.
- com.cohga.server.dms.provider.db
 - A database based document provider that relies on a database table that contains links to documents.
- com.cohga.server.dms.upload
 - The DMS upload API, it provides the API that something needs to implement to allow storage of documents.
- com.cohga.server.dms.upload.code
 - The base implementation that uses the API to store the documents.
- com.cohga.client.dms.upload
 - The client side of the base upload implementation. Provides the forms for uploading a document.
- com.cohga.server.dms.storage.file
 - An implementation of the storage API that used a file system database to record the information about uploaded documents. It also implements the document download API so that uploaded documents can be retrieved.
- com.cohga.server.dms.storage.db
 - An implementation of the storage API that uses and external database to record information about uploaded documents, it also implements the document download API to allow retrieval of uploaded documents.

Only one of com.cohga.server.dms.storage.file or com.cohga.server.dms.storage.db should ever be installed, as they both provide the same function, just in different ways.

Document Upload

If you want to use the Weave document management system to store the documents then you need to tell Weave where it should store the documents.

There are two storage providers that are available to manage the storage of the documents, one that stores the documents on the file system and the other that stores them in a database.

You can only have one document storage provider, so you should only configure one option.

File document storage configuration

File system example storage configuration

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:storagefile="
urn:com.cohga.dms.storage.document.file#1.0">
  <storagefile:config>
    <directory>d:\\weave\\documents</directory>
  </storagefile:config>
</config>
```

There's a single configuration option for the file system document storage and that's the name of a directory where the files will be stored.

Database documents storage configuration

Database example storage configuration

```
<?xml version="1.0" encoding="UTF-8"?>

<config xmlns="urn:com.cohga.server.config#1.0" xmlns:storagedb="urn:
com.cohga.dms.storage.document.db#1.0">
  <storagedb:storage>
    <datasource>datasource.dms</datasource>
  </storagedb:storage>
</config>
```

There's a single configuration option for the database document storage and that's the name of an existing datasource.

Indexing

The Weave quick search function allows you to pre-index your data for faster searching and provides the user with a simple single input box with which to enter the search criteria.

The indexing allows the user to search across multiple types of entities with a single input box, which provides a drop-down list of the top matches for the user to choose from. Note that this limits the results of the search to a single entity, which is different from the existing search functionality which allows for multiple entities to be selected at once but only of a single type (point, line, polygon).

The indexing is setup in two parts. On the server the indexes must be configured to tell Weave which tables to search for each entity, and on the client the input box must be placed somewhere in the user interface.

Installation

To use the indexing functionality in Weave you need to install and start the `com.cohga.server.index` bundle, which at the time of writing was at version 1.0.28 (which added the scheduling).

Namespace

```
urn:com.cohga.server.index#1.0
```

[Index search client configuration](#) - client configurations to enable index search UI tools on the Weave standard client

[Index search server configuration](#) - server configurations required to enable indexing entities on the back end

Further reading

[Indexing and Quick Search Tool Best Practice](#)

Index Search Client Configuration

The following describe the Weave configurations required to enable index search UI tools in standard Weave client.

Adding the index search input box to the client involves adding a new item to the UI, and that new item is called 'weave.indexcombo'. If we assume that we have a toolbar configured for our client and wish to add the index search box to that toolbar then we'd update the client configuration to something like:

```
<toolbar>
  <item component="weave.indexcombo" />
</toolbar>
```

Once we reload the client then the index combo will appear in our user interface and the user could start typing within the input box and will receive a list of the matching entities from the server. Since we have not setup any indexes on the server as yet nothing will be found. We will look at how to setup the indexes later in this document.

By default the index combo will only search for entities that match the currently active entity. To have the index search over all entities we can set a flag in the item to tell the client to ask for all matches, and if the user chooses an item that is not the same type as the currently active entity, then the active entity will be changed when the user chooses that entity. To do this we add a 'all' flag to the item tag and set its value to true, so we would then have:

```
<toolbar>
  <item component="weave.indexcombo" all="true" />
</toolbar>
```

There are other flags that can be added to the item to alter the way the index combo works:

Name	Type	Description
width	string	Sets the width of the input box. This should be a standard HTML width setting, for example "100%" would fill the whole toolbar, "250px" would be 250 pixels wide.
minScale	number	When the user chooses an result from the list the map will be zoomed to the extent of the chosen entity, this setting ensures that the map won't be zoomed in beyond the given scale.
pageSize	number	This value changes the number of items that will be presented to the user as the results of the search, for example "8" will tell the client to display the top 8 results for the search.
doSelect	boolean	The Weave client can be instructed to not update the current selection when the user chooses a result from the list, this allows the index input box to be used as a simple find tool without actually interacting with the active selection. The default for this is <code>true</code> so selecting a result from the list will alter the current selection.
clearOthers	boolean	If <code>doSelect</code> is set not to <code>false</code> then when this flag is set to <code>true</code> it tells the Weave client that it should clear the selection for every other entity when the user chooses a result, if it is set to <code>false</code> (or not set) then the client won't clear the selection for the other entities.
doMarker	boolean	If set to <code>true</code> (the default) this flag tells the Weave client to display a marker at the center of the chosen entity, if set to <code>false</code> then the client will not display a marker.
doGeometry	boolean	If set to <code>false</code> (the default) this flag tells the Weave client not to display the geometry of the selected feature that the user chooses, if <code>true</code> then the Weave client will display the geometry of the entity chosen. This is more useful if the drawing of selections is disabled in the ToC, otherwise the geometry will be drawn twice.
doMarkers	boolean	If set to <code>true</code> (the default) this flag tells the client to display a marker for each item in the list once it completes the search, but before the user chooses a particular entity. If set to <code>false</code> then no markers will be displayed (until the user chooses a particular entity if <code>doMarker</code> is set to true).

doShowIcons	boolean	Display icons next to the search results, default is false.
doTurnOnLayer	boolean	Should the layer associated with the entity, via the ToC model, be turned on when a result is selected? Default is false. Available from release 2.5.26.
type	'wildcard', 'fuzzy' or 'exact'	Change the type of search that's performed, default is 'wildcard'. 'exact' performs the search using the search term, 'wildcard' appends an * to the end of the search term and 'fuzzy' appends a ~ to the end of the search term.
all	boolean	Default is <code>false</code> , when set to <code>true</code> indicates that all indexes should be searched, not just the indexes associated with the active entity, or indexes listed in the <code>index</code> attribute.
index	string	A comma separated list of index names that should be searched. When this value is set only the indexes listed will be searched, rather than all indexes (if <code>all</code> is set to <code>true</code>) or indexes associated with the active entity (if <code>all</code> isn't set to <code>true</code>). Note, this can also be a list of entities, rather than indexes, in which case <i>all</i> indexes associated with the listed entities will be searched.
highlight	boolean	Should the results list highlight the part of the text that was matched in the search? Default is false.
emptyText	string	The text to show in the field when nothing has been entered. i18n resource name is "indexcombo.emptyText", and default value "Quick search".
loadingText	string	The text to show in the field when the content is being loaded. i18n resource name is "indexcombo.loading", and default value "Searching...".
autoFocus	boolean	Should the index field get focus when the client starts up, default is true.

Beyond these settings it is also possible to change the stroke and fill of the geometry (if `doGeometry` is true) by adding a 'geometryStyle' tag to the configuration. The `geometryStyle` tag can set the fill and/or stroke used to draw the entity geometry when it is drawn on the client, but it will not change the style when drawing the selection.

```
<toolbar>
  <item component="weave.indexcombo" all="true">
    <geometryStyle>
      <strokeOpacity>0.75</strokeOpacity>
      <strokeColor>#0000ff</strokeColor>
      <strokeWidth>2</strokeWidth>
      <fillOpacity>0</fillOpacity>
    </geometryStyle>
  </item>
</toolbar>
```

In addition, you can alter the tooltips that are displayed when the user enters the input area, and when they hover over the search button.

```
<toolbar>
  <item component="weave.indexcombo" all="true">
    <geometryStyle>
      <strokeOpacity>0.75</strokeOpacity>
      <strokeColor>#0000ff</strokeColor>
      <strokeWidth>2</strokeWidth>
      <fillOpacity>0</fillOpacity>
```

```

</geometryStyle>
<tooltip>
  <title>Quick Search</title>
  <text>Type here to search</text>
</tooltip>
<tooltip2>
  <title>Quick Search</title>
  <text>Redisplay the last search results</text>
</tooltip2>
</item>
</toolbar>

```

Index Search Server Configuration

The following describe the configuration changes required to enable Weave indexes on the server in order to allow fast indexed search.

For the client to have something to search for, indexes must be configured and entities must be indexed on the server. To provide the type of performance required for interactive searching like this, the indexes must be pre-generated and used for the search rather than actually searching through the database directly each time the user performs a search. To do this Weave creates a free text index from information that you provide it from at least one data definition.

To understand how the index needs to be setup it could be useful to describe how Weave performs the searching. So starting from when the client has typed in some search terms into the search input box on the client, after waiting for a fraction of a second the client will send the contents of the search input box to the server (and then waits for the list of matches to be returned from the server for display), but it's what happens when the input gets to the server that we're interested in for now.

Now when the text arrives at the server it breaks the text into individual words by separating them based on the spaces in the text, and it's these words, individually called "keywords", that Weave will actually be using to search through the index. This leads us to the first important part of building our index, which is generating a list of keywords for each entity.

As part of building the index Weave creates what's referred to as a "document" for each and every entity that you want to be searched for (and index is just a collection of documents). These documents contain the unique id of the entity, so the server can know what's been found, but also has a list of keywords associated with each entity. It's these keywords that are matched against what the user types in that determines what's returned as the results of the search.

The actual contents of the keyword fields in each document are obtained by Weave from a data definition that you have to setup. This data definition links each entity id to a list of columns in the database that contain the keywords that need to be attached to the document, and it's up to you to know what columns it is that are appropriate for the search. For example you could attach registration numbers to dogs, owner names to properties, names to roads or business names to local businesses. Weave and the index builder don't particularly care what content you include in the keywords (apart from some smarts that we'll look at later) and it basically just adds the text to the documents and later when performing the search uses it's smarts to match the user supplied keywords with those included in each document.

So we've seen how the keywords the user supplies and the index builder has associated with with each document tells Weave what it is that you've searched for, but after that we need to display the results to the user and to do this we use 'display' fields from the database. This is done basically the same way as when we associated keywords with each entity when adding them to the documents, but in this case the contents of the display fields aren't indexed when performing the search, instead they're just returned to the client and used in the input box drop-down list to display the results to the user. This is again done by associating a data definition with the index.

Before we have a look at an actual index definition the last piece of information we need to supply is the entity that the index is being generated for, which is done by adding an 'entity' tag to the index definition. By telling the index which entity it's associated with we're also providing it with the other piece of information that it needs to get started, and that's the actual geometry that goes along with each document. This information provides the index with the extent and centroid of each entity, by utilizing a spatial mapper that's associated with the entity, which are stored in the document for each entity along with the entity id, list of keywords and display fields (along with some other supporting fields) to allow most of the information the user required to be quickly available after the search.

If we look at an example of creating a simple index for roads where the user can search for a road in a suburb then we would need some supporting information available. Firstly we need the actual entity that we're going to be searching for and the spatial mapper that provided its geometry and would be something like:

```

<!-- Create our roads entity -->
<entity:entity id="roads">
  <label>Roads</label>
</entity:entity>

<!-- Link the roads entity to the ROADS layer in our spatial engine
(not shown) -->

```

```
<mapper:mapper id="roads.mapper">
  <spatialEngine>spatialEngine</spatialEngine>
  <mapping>
    <entity>roads</entity>
    <table>ROADS</table>
    <key>ROAD_ID</key>
  </mapping>
</mapper:mapper>
```

Next we need a data definition that supplies the keywords and display fields, which could be separate data definitions, but we'll use the same one. An example of what our data definition may look like is as follows:

```
<!-- Provide a road name, type and suburb based on ROAD_ID from the
ROADS table -->
<data:datadefinition id="dd_index_roads">
  <datasourcedataconnection datasource="datasource" key="ROAD_ID">
    <prefix>DISTINCT</prefix>
    <from table="ROADS"/>
    <parameter name="name" column="NAME"/>
    <parameter name="type" column="TYPE"/>
    <parameter name="suburb" column="SUBURB"/>
  </datasourcedataconnection>
</data:datadefinition>
```

Using the above information a simple index definition would look something like this:

```
<index:entity id="index.roads">
  <entity>roads</entity>
  <display>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>Road: ${name} ${type}</level1>
    <level2>Suburb: ${suburb}</level2>
  </display>
  <keywords>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>${name} ${type}</level1>
    <level2>${suburb}</level2>
  </keywords>
</index:entity>
```

So what we end up with here is an index called 'index.roads' which indexes 'roads' entities based on road 'name', 'type' and 'suburb', and displaying the road and suburb details to the user.

Sorting

Further to this we can now (as of version 1.1.0 of the index bundle) also sort the results.

To do this you add a `<sort>` tag to the configuration to tell the index builder what information to attach to each indexed document that's used to determine the sort order.

Sorting does not effect which results are returned to the user, the weighting of the individual documents still does that, the sorting just determines the order in which the results are shown to the user.

When adding sorting to your indexes it's best to either add sort information to all indexes or none if you're searching across all entities (if you've configured the search functionality on the client to only search the active entity then this doesn't apply).

This is because the searching operation is different if sorting is involved compared to when it isn't, and if you're trying to search over multiple indexes (or entities) then as to not impose additional overhead in having to search twice (once for sorted indexes once for non-sorted) Weave will only search sorted indexes or non-sorted indexes. When performing a search across all entities the server will first look for all indexes that have sorting configured and use just those for the search, unless there are indexes that are configured for sorting in which case it will use all of the indexes (and assume that none are sorted). What this means is that if you're sorting across all entities and only some of your indexes have sorting configured then only those indexes will be searched and none of the entities in your non-sorted indexes will be found.

Anyway, to add sorting to a index is the same as adding `display` and `keywords`, but is limited to a single level. So when you add a sort to an index an extra processing step is performed to iterate over the data definition configured for the sort and construct a sort field for each indexed document that will then be used during the searching to order the returned results (as opposed to the default ordering which is based on the weighing of the found documents).

So if we wanted our roads sorted so they're ordered by the suburb they're in followed by the road name then we could do the following:

```
<index:entity id="index.roads">
  <entity>roads</entity>
  <display>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>Road: ${name} ${type}</level1>
    <level2>Suburb: ${suburb}</level2>
  </display>
  <keywords>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>${name} ${type}</level1>
    <level2>${suburb}</level2>
  </keywords>
  <sort>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>${suburb} ${name}</level1>
  </sort>
</index:entity>
```

Since the level in the sort works the same as the display and keywords it means that we can add additional text to the sort, and this can be used to our advantage to ensure that the order of the different types of entities found can be displayed to the user in a certain order. For example if we search enabled suburbs, roads and property addresses then we could use the sort field to ensure that suburbs are always listed first then followed by roads and then properties (regardless of how "well" the actual results match the search). To do this with our previous example we could change the `level1` tag of the sort to

```
<level1>0020 ${suburb} ${name}</level1>
```

and then make sure that our sort for suburbs had the level prefix set to `0010` and properties set to `0030`. This way we'll ensure that the sorting order of the suburbs, roads and properties will always be returned in that order, and the original sorting we specified (suburb and road name in the previous example) will be used within those groups.

A further example of this can be seen if you have address data that stores house numbers as a separate field in your data, then when creating the sort field for your property addresses you would add the house number to the end of the sort field to ensure that the properties are returned in house number order (but remember to left pad the field in the data definition with 0's to ensure the sort is performed numerically rather than alphabetically)

Detail

In detail, what happens with this information when it comes time for Weave to build the index is that the index builder will iterate over each and every feature returned by the spatial mapper associated with the entity indicated by the index. For each feature it finds it creates a new document in the index and to that document it attaches the entity type, the entity id, the entity centroid and the entity extent.

After it's created a document for each available entity it then processes the display definition to add the fields to the document that will be displayed to the user. It does this by iterating over the data definition set in the display configuration and using the `level1` and `level2` information to substitute the fields retrieved from the data definition. It does this by replacing the `${}` values with the matching parameter from the data definition and then using that text as the content of the field to be stored in the document. As you can see from the example above any text can be included in the display configuration, including HTML, in the example above "Road: " and "Suburb: " are examples of additional text that will be sent to the user. And the values from the data definition will replace the markers created using the `${}` syntax.

A display configuration is limited to two level tags, that is you can only specify level1 and level2.

The index builder then processes the keyword fields using the same process it used for the display fields, but in this case there can be up to 5 levels set (we only use 2 in the example above). Again, you can add your own text to the keywords (which hasn't been done in the example above) but it doesn't make sense to include HTML in there, since this is the text that's going to be searched for a match. This could be done for example to add the word 'ROAD' to the keywords index to allow the user to help narrow down the search to just roads if there were other indexes setup for other entities by including 'road' in the search field (I know that's a bad example since road type will include 'road' anyway but you get the idea).

Finally an additional run through is performed if sorting is configured for the index.

Weighting

The levels in the keywords gives us our introduction to weighting in the index. Weighting allows you to set higher priority to some database fields compared to others, and it does this by giving lower numbered levels a higher weighting than higher numbered levels. That is if a document is found that has a match in the level1 field it will be returned higher in the list than another document that may have the same value but in the level2 field, which will be returned before another document that has the same value in the level3 field, and so on.

From our example above we can see that road name and road type will be given the same weighting, the highest, and suburb will be given a slightly lower weighting. It's also possible to attach a weighting explicitly to the index as a whole by adding a 'weight' tag to the index definition that contains a number that's used to multiply the weighting, with the default being 1.0. So by setting a weight of 2.0 the documents in the index will be twice as likely to be returned than another index that has the same values but has the default weighting of 1.0, or setting the weight to 0.5 will halve the chance of those documents being returned.

As of version 1.8.17 of the com.cohga.server.index bundle you now have the ability to specify the weight values for individual records. As can be seen in the example below, the administrator can define a *weights* element inside the index comprising of a datadefinition and a value. The value is sourced from the datadefinition in this case *weight*. The weight will be applied to the feature to increase its score value when searching through the index.

```
<index:entity id="index.roads">
  <entity>roads</entity>
  <display>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>Road: ${name} ${type}</level1>
    <level2>Suburb: ${suburb}</level2>
  </display>
  <keywords>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>${name} ${type}</level1>
    <level2>${suburb}</level2>
  </keywords>
  <weights>
    <datadefinition>dd_index_roads</datadefinition>
    <value>${weight}</value>
  </weights>
  <sort>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>${suburb} ${name}</level1>
  </sort>
</index:entity>
```

Keyword Smarts

As mentioned before there are some smarts built into the index builder when processing the keywords, the first is synonyms and the second is number range expansion.

Synonyms

Synonyms allow you to specify a text file that provides alternate keywords for those found in the database. This can be done for simple things like including "STREET", "STR" and "ST" as keywords when the database provides just "ST" (or just "STR" or just "STREET"), then the user can use either of those values to search for streets. Weave supplies a text file with a list of common street types synonyms as part of the indexer installation that can be used with any index.

Synonyms can also be used to provide completely different words as synonyms, for example 'pharmacist' can be set as a synonym for 'chemist', unlikely to be of much use in our roads example (unless you wanted to setup synonyms list of the street names) but handy if you want people to be able to search for business types and want to catch the different business types that a business could be.

There are two formats of synonym files, one where each of the synonyms refer to the same word, for example in our street abbreviation file, where ST, STR and STREET are abbreviations for the same word. And the other for when the words are alternatives for the same word in one direction but may not apply in the reverse direction, for example 'revoke' and 'abandon' could be a synonym for 'vacate' but 'revoke' shouldn't be a synonym for 'abandon'.

The first format has a single line for each group of synonyms with the alternatives separated by a comma, for example:

```
ST , STR , STREET
```

And the second format has the original word followed by an equals sign and a space separated list of alternatives, for example:

```
abandon=vacate
revoke=vacate
vacate=abandon revoke
```

To add synonyms to an index you must add at least one 'synonyms' tag to the index and rebuild the index.

```
<index:entity id="index.roads">
  <entity>roads</entity>
  <display>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>Road: ${name} ${type}</level1>
    <level2>Suburb: ${suburb}</level2>
  </display>
  <keywords>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>${name} ${type}</level1>
    <level2>${suburb}</level2>
  </keywords>
  <synonyms>street.txt</synonyms>
</index:entity>
```

Number Range Expansion

When adding keywords to a document the index builder will look for keywords that look like number ranges and expands those to include the individual numbers within the range. This way if the database contains "11-14" as one of the fields then the index builder will include "11-14" as one of the keywords, but it will also include "11", "12", "13" and "14" as separate keywords (at the same weighting as the original keyword). This is done to help the user find those likely matches when they search for part of the number range or a number within the range.

In fact the range expansion is more complex than that and can handle a wide range of different formats, including some of the following examples:

Original	Additional
12A	12
1/12	12
1/12A	12A 12 1/12
1A/12A	1/12A 12A 12 1/12
1A/12	1/12 12

10-14	10 14 11 12 13
1/10-14	10-14 10 14 11 12 13 1/10 1/14 1/11 1/12 1/13
1A/10-14	1/10-14 10-14 10 14 11 12 13 1/10 1/14 1/11 1/12 1/13 1A/10 1A/14 1A/11 1A/12 1A/13

Number range expansion is automatic and does not require any changes to the index definition to be enabled, which also means that at the moment it can't be disabled, but that may change in the future.

Scheduling Updates

Because the index is built from the data that's available at the time it's built it may become stale over time and require rebuilding. This can be done manually at the OSGi console, (more on that later) or setup in the index definition using a schedule defined using a format similar to the [Cron](#) format.

By adding a schedule tag you can indicate to Weave when the index can be rebuilt down to the millisecond, have it rebuilt at certain times each day or on certain days of the week (or a combination of these).

Unit	Range
milliseconds	0-999
seconds	0-59
minutes	0-59
hours	0-23
day of week	1-7 (sunday-saturday)
day of month	1-31
month	1-12

Schedule	Description
0 0 30 2	will run at 2:30am each day
0 0 30 2,14	will run at 2:30am and 2:30pm each day
0 0 30 2,8,14,20	will run at 2:30am, 8:30am, 2:30pm and 8:30pm each day
0 0 30 2 3	will run at 2:30am each Tuesday
0 0 30 2 * 1	will run at 2:30am on the first of each month (do not use [*])
0 0 30 2 * 1 2	will run at 2:30am on the first of February each year (do not use [*])
0 0 30 2 5 * 2	will run at 2:30am on each Thursday of February each year (do not use [*])
0 0 30 2 4 1 2	will run at 2:30am on each Wednesday and on the first of February each year
0 0 15,45	will run every half hour at quarter past and quarter to

^{*} It appears that using * for the day of week or day of month may cause issues (like the index continually being built).

Note that since building the indexes can be CPU intensive you should stagger the rebuilding so that you don't try and rebuild more than one index at a time.

Scheduling an index build for 2:30am each day

```
<index:entity id="index.roads">
  <entity>roads</entity>
  <display>
    <datadefinition>dd_index_roads</datadefinition>
    <level1>Road: ${name} ${type}</level1>
    <level2>Suburb: ${suburb}</level2>
  </display>
  <keywords>
    <datadefinition>dd_index_roads</datadefinition>
  </keywords>
</index:entity>
```



```

    <level1>${name} ${type}</level1>
    <level2>${suburb}</level2>
</keywords>
<schedule>0 0 30 2</schedule>
</index:entity>

```

Command Line

The indexing in Weave provides a number of commands that can be used at the OSGi prompt to work with the indexes.

Command	Parameters	Description
is		return a list of all indexes
ib	[<index#> <indexId>]	rebuild an index or all indexes if no index is specified
ik	[<index#> <indexId>]	update keyword fields for an index or all indexes if no index is specified
id	[<index#> <indexId>]	update display fields for an index or all indexes if no index is specified
ig	[<index#> <indexId>]	update geometry field for an index or all indexes if no index is specified
io	[<index#> <indexId>]	update sort field for an index or all indexes if no index is specified
iu	[<index#> <indexId>]	unlock an index or all indexes if no index is specified
ir	[<index#> <indexId>]	remove an index or all indexes if no index is specified
it	"<search terms>" id:<entityKey> [<entityId> <indexId>] [<limit>]	test index

<> substitute, [] = optional, | alternate

<index#> is the value listed in the "Index" column of the is command, and is used to indicate which index to perform the operation on

<indexId> is the value listed in the "Id" column of the is command, and is used to indicate which index to perform the operation on

"<search terms>" is the text to search for, enclosed in double quotes if it contains spaces

id:<entityKey> is the text id: followed by the key value of a specific entity (not a type of entity), e.g. id:45142

<entityId> is the id (or type) of entity to search for, and is the value listed in the "Entity" column of the is command

Update: The ib, ik, id, ig, iu, io and ir commands now also accept a list of space separated index ids indexes or no index id parameters to perform the operation on all indexes.

Also, if multiple commands are submitted at once they'll be queued up so that only one command is performed at a time (this also goes for commands that are triggered through a schedule). This is to ensure that the server isn't overloaded with building indexes (you can imagine if you had 10 indexes and happened to type ib in the console and triggered the concurrent build of 10 indexes).

At the OSGi console you can use 'is' to see what indexes are currently registered in Weave

```

osgi> is
Weave Index Service

Index  Id          Entity          Count  Locked  Modified
0      idx.roads    roads           N/A    N/A     N/A
1      idx.property property        796142 false   12/01/16 11:00

```

From this we can see that the 'idx.roads' index has not actually been built, from here we can use the 'ib' command to build the index (assuming we haven't setup a schedule that would build the index for us)

```

osgi> ib

```

```

osgi> ib 0
Building index for 0
Processing index idx.roads
Indexing non-unique features from ROADS based on ROAD_ID
...
Total time to build index index.roads 32468ms

osgi> is
Weave Index Service

Index  Id          Entity          Count   Locked   Modified
0      idx.roads      roads          16131  false   14/02/16 10:56
1      idx.property  property      796142 false   12/01/16 11:00

```

And then we can actually test our index without having to start the client

```

osgi> it "cameo ct" 2
Start search results for cameo ct
Start result 0
Score: 14.37126
Index: idx.roads
Entity: roads
Id: RD000800
Display 1: <B>Cameo</B> <B>Ct</B>
Display 2: BULLEEN
Keyword 1: CAMEO
Keyword 2: COURT CT CRT
Keyword 3: BULLEEN
Sort: 025 BULLEEN COURT CAMEO
Centroid: {"crs": "EPSG:28355", "x":331951.86744797, "y":
5818594.860681824}
Envelope: {"crs": "EPSG:28355", "minx":331945.5843279641, "miny":
5818544.714681777, "maxx":331958.1505679758, "maxy":5818645.006681871}
End result 0

Start result 1
Score: 21.21312
Index: idx.property
Entity: property
Id: 12395
Display 1: Mr R Dunhill
Display 2: 1 <B>CAMEO</B> CRT, BULLEEN
Keyword 1: 1 CAMEO CRT BULLEEN COURT CT
Keyword 2: Mr R Dunhill
Keyword 3: LOT 21 LP83371
Sort: 050 BULLEEN CRT CAMEO 000001
Centroid: {"crs": "EPSG:28355", "x":331921.4042969137, "y":
5818564.7704825485}
Envelope: {"crs": "EPSG:28355", "minx":331903.6789254192, "miny":
5818553.782011072, "maxx":331940.9090216262, "maxy":5818578.51987176}
End result 1

```

End search results

```
osgi> it id:12395 idx.property
Start search results for id:12395
Start result 0
Score: 21.21312
Index: idx.property
Entity: property
Id: 12395
Display 1: Mr R Dunhill
Display 2: 1 <B>CAMEO</B> CRT, BULLEEN
Keyword 1: 1 CAMEO CRT BULLEEN COURT CT
Keyword 2: Mr R Dunhill
Keyword 3: LOT 21 LP83371
Sort: 050 BULLEEN CRT CAMEO 000001
Centroid: {"crs": "EPSG:28355", "x":331921.4042969137, "y":
5818564.7704825485}
Envelope: {"crs": "EPSG:28355", "minx":331903.6789254192, "miny":
5818553.782011072, "maxx":331940.9090216262, "maxy":5818578.51987176}
End result 0
End search results
```

Manually updating indexes

The server status page contains links for initiating an index update, and the 'build' link in that page can be accessed by an external application to start an index build.

The link to start an index build would <http://hostname:8080/weave/server/index/build/<indexid>>

Troubleshooting

The osgi console has the ability to perform index searches, using the 'it' (index test) command, and it show more details about the results, it also does it at a slightly lower level than the client.

You should run any test you do through this command to see what's actually going on.

You may want to check out http://lucene.apache.org/core/old_versioned_docs/versions/2_9_4/queryparsersyntax.html for details on the search syntax when using the 'it' command.

Note: By default the Weave index search adds an * to the end of the search term, so to replicate what the client is doing you should also include an * at the end of the search term when using the *it* command.

There's also a standalone tool you can download <http://www.getopt.org/luke/> that will allow you to open and look at the index directly.

Note: One thing to keep in mind is to change the *Analyzer* to the *StandardAnalyzer* from the *KeywordAnalyzer*, in the *Analysis* tab under the Search tab if you do any searches.

The Documents tab is handy because it allows you to cycle through the stored documents directly and see exactly what's stored for each one.

You need to make sure you're using synonyms if you're storing things like street types where the database contains 'RD' but the user may type in 'road' or 'rd'.

Also, punctuation can cause problems if used in a keyword fields, meaning the user typing 'road' won't necessarily match the keywords if it's derived from the value '123 Main Road, Smallville'.

Updates for Weave 2.5

As of Weave 2.5 the way indexes are built has changed.

Previously they were generated based on the geometry first, now they're generated based on the attributes first, but only if the data definitions used for the keywords, display, sort and fields is the same.

To switch to the older method of generating indexes you can set

```
<geometryFirst>true</geometryFirst>
```

as an option within the index config (and ensure all data definitions are the same). This flag was introduced in Weave 2.5.4.

Additionally, since all data definition are the same they can be set once at the top level of the index config, rather than duplicated within each section.

The index builder doesn't like it if there is a large change in the number of items in the index between one build and the next, since it normally indicates some sort of error, so rather than replacing the index with "bad" content it aborts the update. There are however situations where this is not an error, in which can you can turn off the check by adding `<check>false</check>` to the index definition.

If the key column values are unique for each entity in the spatial table(s) you can add `<unique>true</unique>` to the index config to provide a hint to the index builder that this is the case and it can optimize the building of the indexes.

Auto-building indexes

As of Weave 2.5.29, indexes will be built when they're created if their content does not already exist. Also indexes that aren't built will be built when Weave starts. This can be disabled by setting the system property `weave.index.autobuild` to `false` or adding `<autobuild>false</autobuild>` to the index config.

Coordinate Reference System

You can specify what CRS to use for the stored centroid and envelope by setting a `crs` tag in the index config. If you don't specify a `crs` it will be the same as the original geometry.

Indexing and Quick Search Tool Best Practice

This page provides guidelines for setting up and using indexes in Weave. It should be used in conjunction with the [Indexing](#) page which goes into technical detail about indexes and how they can be configured. Each Weave implementation is unique so the information provided below may not be relevant to all sites at all times.

The Quick Search engine is a fully integrated part of your core Weave installation and it is not a separate installation or server system.

The Quick Search tool (`weave.indexcombo`) in the Weave Client uses indexes that are created through configuration XML files.

Once created, the indexes can be updated manually or can be scheduled to update on a regular basis. Often this update will be after hours when no-one is using the Weave server and each index can have it's own update schedule.

Indexes can be complex and need to be set up correctly in order to get the best results. If you do not set parameters appropriate to your index then your Quick Search will return the records you are after *some* of the time, but other times it will return, what seem to be, totally unrelated records.

We do encourage you to experiment with creating a new index, or get a better understanding of your existing index(es), and the following notes will provide you some handy tips. If you get stuck then you can also get advice from Cohga as we have set up a variety of indexes for different purposes.

- The Quick Search tool does not replace the Search Panel (`main.searchView`). When you have a number of different fields to be searched simultaneously or want to find an *exact* match to the input criteria, the Quick Search tool is what you should be using. Its ability to use lists, convert text case, add wildcards, etc. makes it the best tool for finding exactly what you want. The Search Panel is provided for more structured or forms based queries.
- Indexes that are used in the Quick Search use Lucene which is an open-source text search engine library (written in Java). There is a wealth of information on the internet about Lucene but some of the characteristics of the library listed by its creator ([Apache](#)) are:
 - Scalable, High-Performance Indexing
 - over 150GB/hour on modern hardware
 - small RAM requirements
 - index size roughly 20-30% the size of text indexed
 - Powerful, Accurate and Efficient Search Algorithms
 - ranked searching - best results returned first
 - many powerful query types: phrase queries, wildcard queries, proximity queries, range queries
 - sorting by any field
 - multiple-index searching with merged results
 - allows simultaneous update and searching
 - fast, memory-efficient and typo-tolerant suggesters
 - Cross-Platform Solution
 - Available as Open Source software

[Source: <https://lucene.apache.org/core/>]

While Lucene is a valuable library for inclusion in Weave, without understanding how it works you will not get the search results you were expecting.

- One of the advantages of Lucene is its ability to do a "sounds like" query. So a user can search for "tara" and the returned records will include "nara", "sara", and a search for "bell" will return "jells", "bella", "wells", etc. This type of search is not possible in the Search Panel. Lucene indexing also allows you to have a list of synonyms so that common alternative terms (e.g. drive, dr, dv, dve) can be taken into consideration in the search without the user having to be aware of this.
- Weave indexes can do "sounds like" searches because these phrases sound the same or have a similar sequence of characters. The result of this, however, is that the Quick Search tool does not work well when searching for numbers. Therefore a good way to enable users to easily search for numbers is to add a character prefix to the numbers in the keyword definition.

In the configuration example below we have several numbers that can be searched on, namely `land_no`, `property_no`, and `proclaim_link`. We also have a `property_address` field that will contain a number at the start.

Original

```
<keywords>
  <datadefinition>dd.xxxx</datadefinition>
  <level1>
    ${land_no}
    ${property_no}
    ${proclaim_link}
    ${plan}
    ${property_address}
    ${land_address}
    ${owners}
  </level1>
</keywords>
```

When a user types "15" into the search box looking for `land_no` "15", the index may also return the `land_no` matching "15", `property_no` matching "15" and the `proclaim_link` matching "15". What will also be returned are all the addresses that potentially start with "15".

So if you want to allow the user to easily search for `land_no` in the above example, a simple trick is to append "ln" before the land number.

Enhanced

```
<keywords>
  <datadefinition>dd.xxxx</datadefinition>
  <level1>
    ln${land_no} ${land_no}
    ${property_no}
    ${proclaim_link}
    ${plan}
    ${property_address}
    ${land_address}
    ${owners}
  </level1>
</keywords>
```

Now if a user wants to search for `land_no` "15" they can simply type "ln15" into the search box and it will find the correct land number. You will notice in the example above that the `land_no` field is duplicated, this is done to allow the user to still type in a `land_no` in any way and will work well for longer sequences of numbers which are usually unique or will return only a few matches.

- Multiple indexes can be used together so data from different tables and databases can be searched for in the one search - this is something that is not possible in the Search Panel as each search in the Search Panel is linked to one Active Layer.
- A Quick Search can be linked to the Active Layer or it can be set up to search across all indexes so it ignores the Active Layer. In this way, a Quick Search can be used in your Weave client to assist novice users. It can provide a simple tool for users who are new to the concept of "active layers or selectable layers", and those who, even once experienced with this concept, are likely to struggle to remember it because of their infrequent use of the Weave client. The Quick Search tool offers one place for users to type in their search term compared to the Search Panel which can provide many, sometimes confusing, options for text entry for the web mapping novice.
- You can add weighting and sorting to your index results. In many cases it is not necessary to use either of these options and if you are new to the process of creating indexes for Weave then it might be best to avoid using them. If used incorrectly they can skew the results of your index and also slow down the index creation and search time. If you want to experiment with these parameters then wait until your existing index(es) are running smoothly, and you are confident in testing the index through the OSGi console or the Weave Admin Tool.
- When setting up the Quick Search tool in your client XML file, you can specify an XML Attribute of `type` which dictates *how* the search will be performed (this has nothing to do with how the index is created, just how the index is used within the Weave Client). The possible `type` values are 'wildcard', 'fuzzy' or 'exact'. When you specify one of these search methods, it is not the only search method used, and it is better thought of as the "starting point" for the search. This is shown in Figure 1, with the `type` on the left, the search start point in the middle and the resulting search string on the right.
- The search methods shown in Figure 1 are only used in the Quick Search tool (which is why they are specified in the client XML file rather than the index XML file). They do not form part of the indexing process. It is important to remember this when testing your index (through the OSGi console or the *Index Tester* in the Admin Tool) as, in order to mimic the Quick Search tool, you will need to add the wildcard (*) or fuzzy (~) characters to your search string.

There were errors rendering macro:

- An unknown error occurred.
- The [Indexing](#) page outlines all elements required to build and schedule the rebuilding of an index; Figure 2 summarises this.

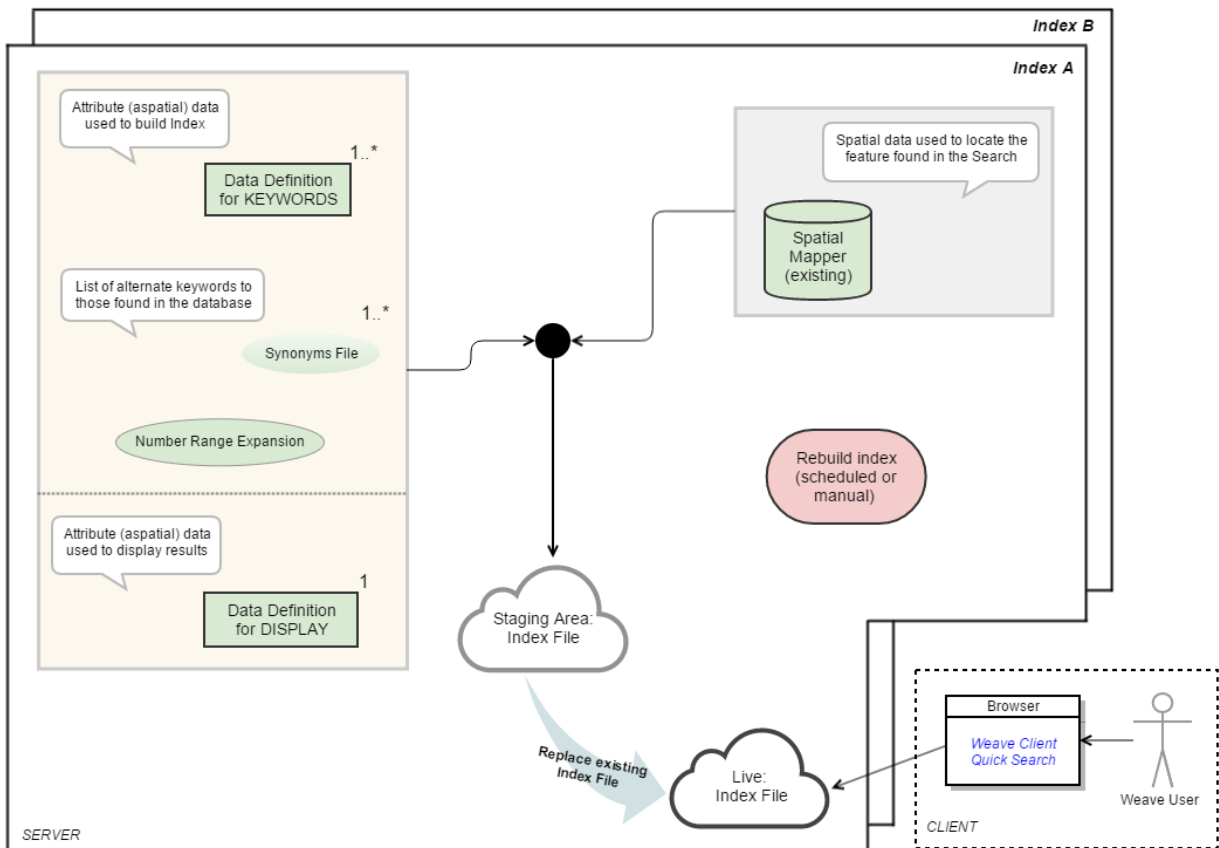


Figure 2: Creating and Accessing Indexes

Editing

Non-spatial Editor

The content here is a work in progress and may not be complete or accurate.

Namespace

urn:com.cohga.server.editor#1.0

Tags

entity

Properties

Name	Type	Required	Description
id	string	yes	unique identifier.
acl	ref urn:com.cohga.server.acl#1.0:acl	no	Optional ACL to restrict access.
entity	ref urn:com.cohga.server.entity#1.0:entity	yes	The entity that this edit config relates to.
label	string	yes	The label that is displayed to the user for this edit.
datasource	ref urn:com.cohga.server.datasource.jdbc#1.0:datasource	yes	The datasource that contains the table to edit.
table	string	yes	The name of the table within the datasource that is to be edited.
key	string	no	The column in the table that will uniquely identify each row, the primary key will be used if not set.
entityTable	string	no	Specify a different table for the entity id column from the one that's being edited. Requires a <code>join</code> to be configured to link this table to the source table.
entityKey	string	no	Specify a different column for the entity id from the key column specified here. If <code>entityTable</code> is specified the column must be in that table.
join	ref urn:com.cohga.server.editor#1.0:join	no	A join expression for linking the source table and other tables (generally for joins to refine what rows are available, e.g. linking to a status table).
sort	ref urn:com.cohga.server.editor#1.0:sort	no	Order by clauses to use when retrieving data for display.
parameters	boolean	no	If set to <code>true</code> the table will be examined to determine the information about what columns can be edited and the <code>parameter</code> tags override/supplement this information. If not set or set to <code>false</code> the <code>parameter</code> tags will determine what columns can be edited.
parameter	ref urn:com.cohga.server.editor#1.0:parameter	no	Describes one column to be edited, can be specified multiple times.
retireColumn	string	no	Column to use to "retire" a record rather than deleting the row when deleting a record. Must be a timestamp column type in the database. When this column is specified then deleting a record won't delete the record but will instead record the date the record was retired in this column. Rows with a value in this column won't be included when displaying data from this table to the user.
history	boolean	no	When set to <code>true</code> indicates that a history of the changes made to this table will be recorded in a separate history table. The schema for the history table must match the source table (with the addition of the user and time columns).
historyTable	string	no	The name of the table to record historical data, defaults to the table name plus the suffix <code>"_hst"</code> .
historyUserColumn	string	no	The column in the history table to record the user who made the change, defaults to <code>"modified_user"</code>
historyTimeColumn	string	no	The column in the history table to record the time when the change was made, defaults to <code>"modified_time"</code> . Must be a timestamp column type in the database.

join

Properties

Name	Type	Required	Description
type	LEFT, INNER, RIGHT	no	Type of join to perform, default is INNER
left	string or object	yes	The left table and column in the join, formatted as TABLE.COLUMN. <left>TABLE.COLUMN</left> can also be specified using table and column sub-tags, e.g. <left><table>TABLE</table><column>COLUMN</column></left>
right	string or object	yes	The right table and column in the join, formatted as TABLE.COLUMN. <right>TABLE.COLUMN</right> can also be specified using table and column sub-tags, e.g. <right><table>TABLE</table><column>COLUMN</column></right>

sort

Properties

Name	Type	Required	Description
	string	yes	The column name to sort by. Note this isn't an attribute of the tag, it's the content, e.g. it's <sort>CREATION_DATE</sort>
direction	ASC or DESC	no	Sort direction, <sort direction="DESC">CREATION_DATE</sort>

parameter

Properties

Name	Type	Required	Description
displayName	string	no	No idea.
helpText	string	no	Description of the input field.
promptText	string	no	The label for the field (can also be called <code>label</code> rather than <code>promptText</code>)
acl	ref urn:com.cohga.server.acl#1.0:acl	no	An ACL to restrict access to the parameter.
hidden	boolean	no	Is the input field displayed?
id	string	yes	A unique id for the parameter
alignment	'left', 'center', 'right' or 'auto'	no	How to align the input.
controlType	'listbox', 'checkbox', 'radiobutton', 'textbox', 'textarea' or 'money'	no	How the input of the field should be presented.
dataType	'any', 'boolean', 'datetime', 'date', 'time', 'decimal', 'float', 'integer', 'string'	no	What is the underlying data type for the column?
allowBlank	boolean	no	Can this field be left blank?
allowNewValues	boolean	no	Is a combo box limited to the defined values or can the user enter their own values?
allowNull	boolean	no	Does the database allow this value to be null?
defaultValue	string	no	The initial value to populate a field with.
displayFormat	string	no	No idea.
fixedOrder	boolean	no	No idea, but apparently you can set it to true or false.
concealValue	boolean	no	Should the input value be hidden when the user types it in (no currently supported in the client).
list	list of values	no	
scalarParameter Type	'simple' or 'multi-value'	no	If set to <code>multi-value</code> then more than one item can be selected from a list box.
pageSize	integer	no	Number of entries in a list box to display per-page, set to 0 to disable paging.
width	integer	no	Change the width of the input field.

matchType	'start', 'middle', end or 'auto'	no	When searching a list box based on the text the user is typing this determines how the text is used, default is start.
autoSelect	boolean	no	If <code>true</code> then the first value in a list box is preselected.
forceSelection	boolean	no	If <code>true</code> then a value must be chosen from a list box.
typeAhead	boolean	no	If set to <code>true</code> typing in a list box field refines the available values.
minLength	integer	no	The minimum number of characters required for a field.
maxLength	integer	no	The maximum number of characters required for a field.
minValue		no	The minimum value allowed for a field.
maxValue		no	The maximum value allowed for a field.
increment		no	For fields that support it this is the value of the difference between subsequent values, e.g. it's 15 for time fields
uppercase	boolean	no	Should the value be converted to upper-case before being inserted into the database.
sequence	string	no	If the value for the parameter is generated from a database sequence then this is the name.
autogenerated	boolean	no	Is the value for the parameter generated by the database.
trueValue		no	The value stored in the database that indicates that the value is "true".
falseValue		no	The value stored in the database that indicates that the value is "false".
readOnlyOnUpdate	boolean	no	If set to <code>true</code> then the value can not be changed when the record is being updated.
readOnlyOnInsert	boolean	no	If set to <code>true</code> then the value can not be changed when the record is being created.
dataSet	ref urn:com.cohga.server.data.database#1.0	no	Source data definition for list box content
labelColumn	string	no	Parameter in <code>dataSet</code> that contains the label for the list box content
valueColumn	string	no	Parameter in <code>dataSet</code> that contains the value for the list box content
sortColumn	labelColumn or valueColumn or one of 'label', 'value', 'none', 'defined' or 'default'	no	How the content of the listbox should be sorted: <ul style="list-style-type: none"> • label - sort based on content of the label • value - sort based on the content of the value • none - no sorting • defined - sorting is provided by the data definition • default - whatever happens when this value isn't set Additionally the value could be the same as the <code>labelColumn</code> or <code>valueColumn</code> attribute
value	string	no	The value that should be written to the column, the user will not be able to change this value.
autoExpand	boolean	no	Should a list box be expanded immediately when the user enters the field?
readOnly	boolean	no	If set to <code>true</code> then the value can not be changed.