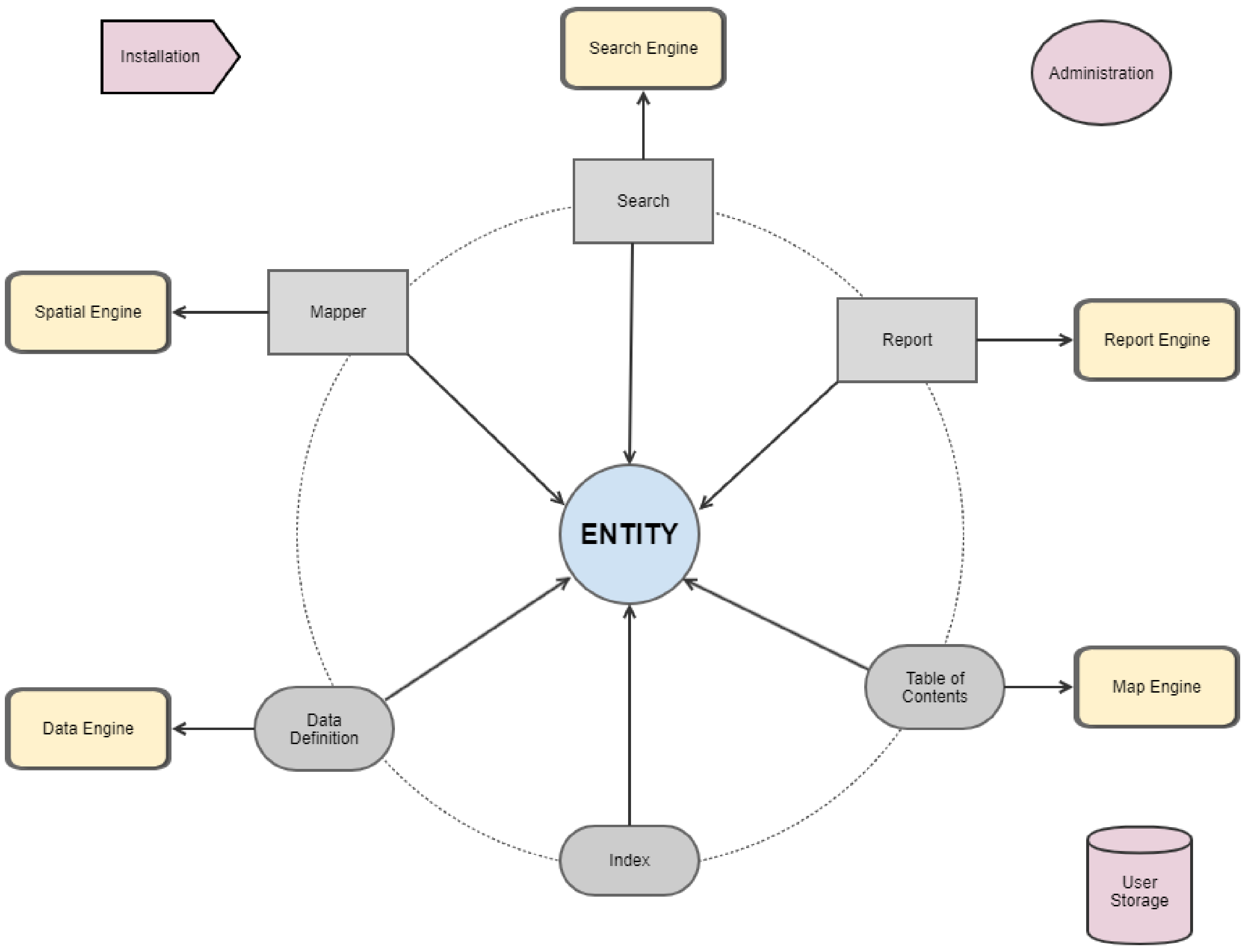
Weave System Administrator Guide Excerpt Feb 2022

1. How-to Pages . . . 3
- 1.1 How to Add a Column to a Data Grid . . . 5
- 1.2 How to Add a Layer to the Table of Contents . . . 6
- 1.3 How to Add a New Active Layer . . . 8
- 1.4 How to Add a New Data Grid . . . 10
- 1.5 How to Add a New Search . . . 13
- 1.6 How to Clear Your Browser Cache . . . 16
- 1.7 How to Create a Quick Search . . . 18
- 1.8 How to Create an Image Slider . . . 19
- 1.9 How to Debug a BIRT Report . . . 21
- 1.10 How to Edit an A3 Portrait BIRT Report . . . 22
- 1.11 How to Encrypt Passwords in Configuration Files . . . 24
- 1.12 How to fix a PKIX Path Building Failed Error . . . 25
- 1.13 How to Get the Most From Weave Logging . . . 29
- 1.14 How to Include an Image in a Data Grid or Map Tip . . . 31
- 1.15 How to Include Special Characters in Weave Configuration Files . . . 32
- 1.16 How to Increase the Spatial Operation Limit . . . 33
- 1.17 How to Install a New Version of Weave . . . 34
- 1.18 How to Install a Weave Patch . . . 40
- 1.19 How to Log a Support Ticket for Best Results . . . 41
- 1.20 How to Maintain Bookmarks When a Map Engine Changes . . . 41
- 1.21 How to Make a Layer Selectable . . . 43
- 1.22 How to Make and Use Snippets . . . 43
- 1.23 How to Migrate Existing Storage Parameters to Database Storage . . . 45
- 1.24 How to Run a Weave Hotfix . . . 46
- 1.25 How to Run a Weave Software Upgrade . . . 47
- 1.26 How to Serve Additional Web Content . . . 48
- 1.27 How to Set a System Property . . . 50
- 1.28 How to Set up Non-spatial Editing . . . 52
- 1.29 How to Set up Spatial Editing using the Weave Edit Extension . . . 53
- 1.30 How to Store Redlines and Bookmarks in a Database . . . 54
- 1.31 How to Test Your Weave Application Integrations . . . 56
- 1.32 How to Update Weave Help Content . . . 57
- 1.33 How to Upgrade to Java 8 . . . 58
- 1.34 How to Use a List in an Attribute Search . . . 59
- 1.35 How to Use a Shapefile in Weave . . . 61
- 1.36 How to View the Developer Console . . . 63
- 1.37 How to Get a Weave Support Dump using a URI . . . 68

How-to Pages

Find a How-to Page by its Component

The diagram below links common tasks in Weave administration to their related components.



Find a How-to Page by Keyword

A-C

- activelayer
- administration
- api
- arcgis
- birt
- bookmark
- cdata
- certificate
- configuration

D-G

- database
- datadefinition
- datagrid
- debug
- dropdownlist
- edit
- editing
- encrypt
- entity
- error
- grid

H-K

- help
- hotfix
- http
- image
- index
- installation
- java
- jetty
- jvm

L-O

- layer
- limit
- list
- listbox
- logfile
- logging
- mapengine
- maplayers
- maptip
- non-spatial

P-R

- password
- patch
- query
- quicksearch
- redline
- report

S

- search
- select
- selectable
- shapefile
- slider
- snippet
- spatial
- spatialengine
- ssl
- storage
- support
- syntax
- system

T-Z

- tableofcontents
- testing
- toc
- upgrade
- user
- userstorage
- web

How-to Pages Listing

Click here to expand...

- How to Add a Column to a Data Grid
- How to Add a Layer to the Table of Contents
- How to Add a New Active Layer
- How to Add a New Data Grid
- How to Add a New Search
- How to Clear Your Browser Cache
- How to Create a Quick Search
- How to Create an Image Slider
- How to Debug a BIRT Report
- How to Edit an A3 Portrait BIRT Report
- How to Encrypt Passwords in Configuration Files
- How to fix a PKIX Path Building Failed Error
- How to Get a Weave Support Dump using a URI
- How to Get the Most From Weave Logging
- How to Include an Image in a Data Grid or Map Tip
- How to Include Special Characters in Weave Configuration Files
- How to Increase the Spatial Operation Limit
- How to Install a New Version of Weave
- How to Install a Weave Patch
- How to Log a Support Ticket for Best Results
- How to Maintain Bookmarks When a Map Engine Changes
- How to Make a Layer Selectable
- How to Make and Use Snippets
- How to Migrate Existing Storage Parameters to Database Storage
- How to Run a Weave Hotfix
- How to Run a Weave Software Upgrade

- How to Serve Additional Web Content
- How to Set a System Property
- How to Set up Non-spatial Editing
- How to Set up Spatial Editing using the Weave Edit Extension
- How to Store Redlines and Bookmarks in a Database
- How to Test Your Weave Application Integrations
- How to Update Weave Help Content
- How to Upgrade to Java 8
- How to Use a List in an Attribute Search
- How to Use a Shapefile in Weave
- How to View the Developer Console

How to Add a Column to a Data Grid

When you have attributes being displayed in a Data Grid you can add or remove any field/column of the data at any time. This requires you to change the relevant Data Definition.

Step-by-step guide

1. Make note of the *Active Layer* and the field from that layer you want to add to the existing *Data Grid*. If you have more than one report used in the *Data Grid* for that *Active Layer*, then also note the name of the report.

2. Open your `entity.xml` file. Search for the name of the *Active Layer* you identified above (it will appear as the `label`). Note the value of the `id` XML Attribute.

3. Open your `data.xml` file for editing. Search for the value of the `id` identified in the step above. Find the section that defines the Data Definition you want to modify.

4. Add a new line to the `datasourcedataconnection` section so you have a new `parameter` line. The easiest way to do this is to copy and paste an existing line and then modify it. Ensure that you update the `type, name, label` and `column` parameters to match those of the field you are adding to the Data Definition.

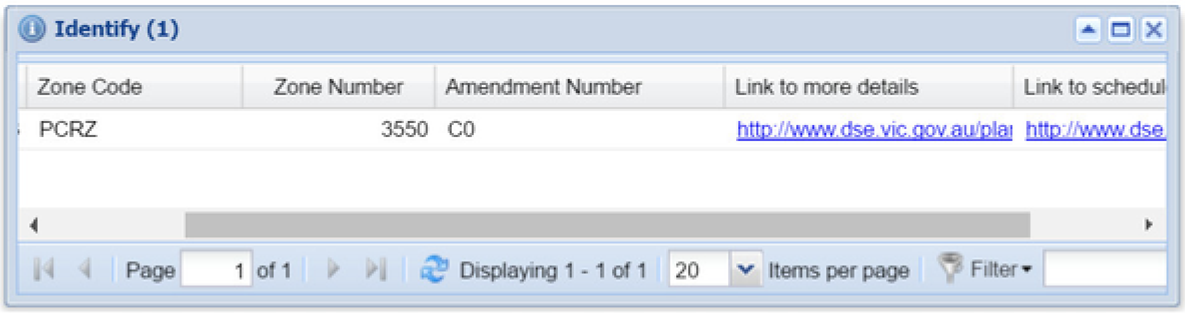
   In the example below the field being added is called `amd_num` and is specified as:

   ```
   <parameter type='string' name='zoneamendmt_num' label='Amendment
   Number' column='amd_num'/>
   ```

   Once incorporated into the existing Data Definition it it will look like:

```
<data:datadefinition id='dd_zone'>
        <datasourcedataconnection key='ogc_fid' table='mann.
zones' datasource='gis'>
                <parameter type='interger' name='ogc_fid'
label='FID' column='ogc_fid'/>
                <parameter type='string' name='zone_code'
label='Zone Code' column='zone_code'/>
                <parameter type='interger' name='zone_num'
label='Zone Number' column='zone_num'/>
                <parameter type='string' name='zoneamendmt_num'
label='Amendment Number' column='amd_num'/>
                <parameter type='url' name='details' label='Link
to more details' column='details'/>
                <parameter type='url' name='schedule'
label='Link to schedule' column='schedule'/>
        </datasourcedataconnection>
</data:datadefinition>
```

5. Once this change has been made, you may need to refresh your browser to see this attribute appear in your *Data Grid* window.



Zone Code	Zone Number	Amendment Number	Link to more details	Link to schedul
PCRZ	3550	C0	http://www.dse.vic.gov.au/plai	http://www.dse

Also refer to the following sections of the Weave System Administrator Guides:

- Data Definition

Related articles

- How to Add a Column to a Data Grid
- How to Add a New Data Grid
- How to Include an Image in a Data Grid or Map Tip

How to Add a Layer to the Table of Contents

When you have a new Map Layer available you can add it to your *Table of Contents*.

Step-by-step guide

1. If you know the details of the layer you want to add to your *Table of Contents* then open the `toc.xml` and edit this file. If you are familiar with the layer you want to display, you can use the other layers defined in this file as a guide and add in as many `entry` XM L Tag lines that you need.

2. If you're not sure what layers are available in the Map Engine you can use the Weave Administration Tool *Console* or Weave's OSGi console to assist you (using the Console will save you some typing and reduces the likelihood of a typing mistake). In the console, run the `memd` command with the details of the Map Engine you want to report on. If you're not sure of the Map Engine name, either refer to the `mapengine.xml` file or use the *Map Engine Tool* from the Administration Tool.

```
memd toc <mapengine_name>

e.g.:
        memd toc mapengine.main
```

3. The `memd` command will generate XML that will be used in the `toc.xml` file. Depending on your logging properties, the results from this command will either be reported in the Console (where you can simply copy and paste the text) or in the `C:\weave\logs\weave.log` file (go to the bottom of the file and copy the section after the line “MapEngine ToC Configuration Generator”).

4. Add the XML generated from the step above to the `toc.xml` file. The layers are assigned their XML Attribute `id` with the prefix of “l_” followed by the Map Engine name, followed by the name of the layer in the Map Engine. Use this same naming convention for any further layers added. Change the `checked` XML Attribute to `true` if you want the layer to be displayed by default when Weave starts.

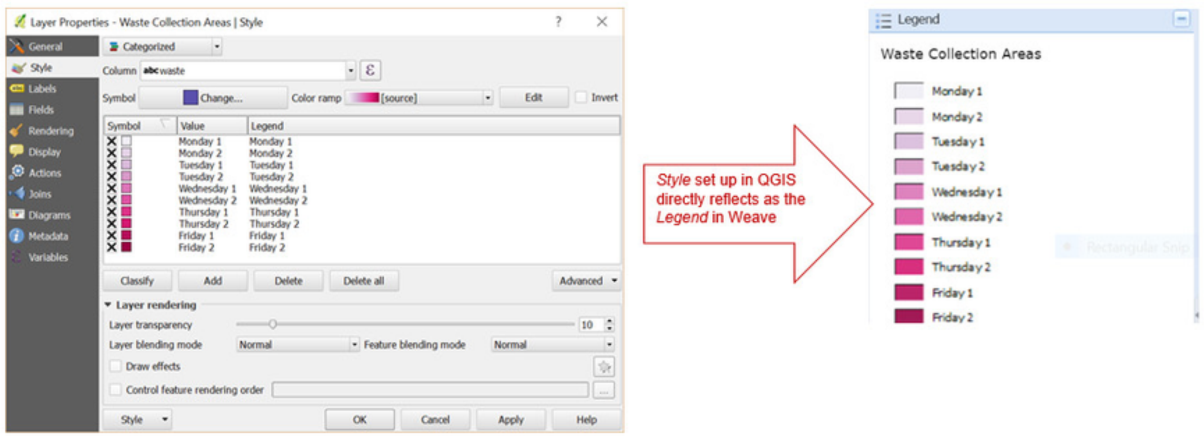
```
<toc:model id='toc.main'>
        <mapengine>mapengine.main</mapengine>
        <entry id='l_mapengine.main_waste' layer='waste'
label='Waste Collection Areas' checked='false'/>
        <entry id='l_mapengine.main_property' layer='property'
label='Property' checked='false'/>
        ...
        <entry id='l_mapengine.main_counres' layer='counres'
label='Council Reserves' checked='false'/>
</toc:model>
```

Note: As you add more layers, ensure that you every ToC entry has an id (whether the entry is a group, folder or layer). The id should be unique across the whole ToC model. This also applies if you’re using the same layer multiple times in a ToC model, i.e. maybe you have layers in groups and for ease of use you’ve put the same layer in multiple groups.

5. If the layer that you want to add to Weave is not reported as being part of the Map Engine then:
   a. If you are the author of the mapping data then add the layer to one of your existing Map Engines.
   b. Or contact the author of the mapping data and get them to add the spatial layer.

   Changes to a Map Engine used by Weave will be automatically picked up by Weave.

6. The legend as displayed in the *Legend* panel in Weave is controlled by the Map Engine. If you want to change the ordering of items in the legend or naming of the legend items, etc., this needs to done in the system that is producing the map (e.g. ArcMap, QGIS, GeoServer). There is no Weave configuration that will change the appearance of the legend.



Categorized
Column: waste
Symbol: Change...   Color ramp: [source]   Edit   Invert

Symbol	Value	Legend
✕	Monday 1	Monday 1
✕	Monday 2	Monday 2
✕	Tuesday 1	Tuesday 1
✕	Tuesday 2	Tuesday 2
✕	Wednesday 1	Wednesday 1
✕	Wednesday 2	Wednesday 2
✕	Thursday 1	Thursday 1
✕	Thursday 2	Thursday 2
✕	Friday 1	Friday 1
✕	Friday 2	Friday 2

Layer rendering
Layer transparency: 10
Layer blending mode: Normal   Feature blending mode: Normal
Draw effects
Control feature rendering order

Waste Collection Areas
Monday 1
Monday 2
Tuesday 1
Tuesday 2
Wednesday 1
Wednesday 2
Thursday 1
Thursday 2
Friday 1
Friday 2

Also refer to the following sections of the Weave System Administrator Guides:

- Table Of Contents

Related articles

- How to Add a Layer to the Table of Contents
- How to Create an Image Slider
- How to Maintain Bookmarks When a Map Engine Changes
- How to Add a New Data Grid
- How to Add a New Active Layer

How to Add a New Active Layer

When you have a layer that you want to graphically select from, and report on its attributes, it needs to be set up as an *Active Layer*. This article assumes that you already have some *Active Layers* configured in Weave.

Step-by-step guide

1. If the layer you want to add as a new *Active Layer* is in the *Table of Contents (TOC)*, make a note of the name shown in the Weave *TOC* (on the *Layers* panel).
    - Generally, an *Active Layer* is related to a *Map Layer* from the *TOC* but this does not need to be the case. In the extreme, you can have just one raster base map in your *TOC* but you will still be able to do a graphical selection and have the selection appear over the raster map. In this case it will appear as though features on the raster map have been selected, but in fact it is the vector layer, that covers the same geographic area as the raster map, that is used in the graphical selection (through the Spatial Engine).

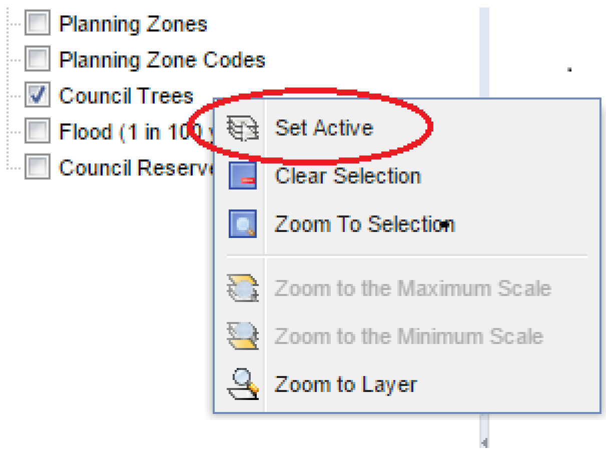
2. Open your `toc.xml` file for editing. Search for the name of the *Map Layer* you identified above (it will appear as the `label` XML Attribute). Note the value of the `layer` XML Attribute.

3. While you have the `toc.xml` open, add an XML Attribute called `entity` to the end of that entry line.

   In the example below the XML Attribute `entity` is added to the `counciltrees` layer and is set to `counciltrees`:

```
<entry id='l_mapengine.main_counciltrees' layer='counciltrees'
label='Council Trees' checked='false' entity='counciltrees'/>
```

   Adding this `entity` attribute sets the relationship between the *Active Layer* and the *Map Layer* that is drawn from the *TOC* and means the *Map Layer* in the *TOC* will be highlighted when the corresponding layer is selected from the Active Layer drop-down list. To have the *Set Active* option available on the *TOC Context Menu* (right mouse click), ensure you have the `weave.toc.selectLayer` item enabled in the `client_main.xml` file. This will most likely have been done by Cohga during your Weave installation.

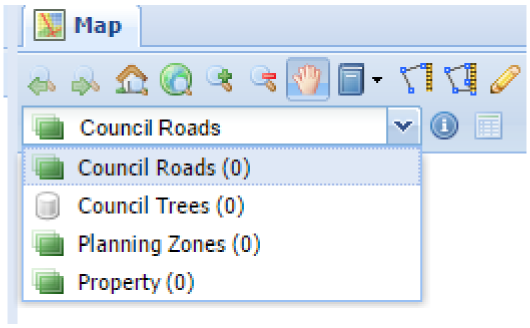


4. Open your `entity.xml` file for editing. Add an XML Element for this entity. The value of the (XML Attribute) `id` for the entity (shown in the example code below) should match the (XML Attribute) `entity` from the `toc.xml` file (shown in Step 3). In our example the value of the `id` is "`counciltrees`".

```
<entity:entity id='counciltrees'>
     <label>Council Trees</label>
</entity:entity>
```

   The `label` in the `entity.xml` doesn't have to match the `label` in the `toc.xml` file but unless there's a good reason why these should be different, it's recommended to keep them the same to avoid confusion for the user.

5. If you refresh your browser you'll now see the *Active Layer* has been added to the Active Layer drop-down list (i.e. `Council Trees`) . However it will have a different icon to the other Active Layers in the list as it hasn't been linked to a spatial database.



6. Stay in the `entity.xml` file. You now need to specify where the spatial data for the selection will be coming from by adding another `mapping` XML Entity that links the *Active Layer* to its relevant Spatial Engine. Add this to the `mapper` XML Element that is reading from the required Spatial Engine,and include the table name, key field, etc. as shown below.

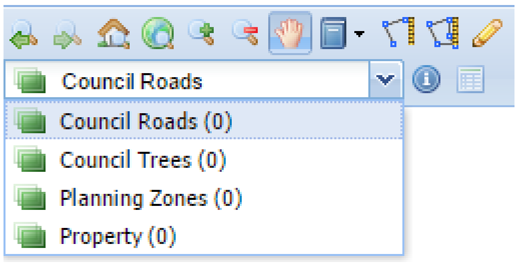
```
<mapping>
        <entity>counciltrees</entity>
        <spatialengine>postgis</spatialengine>
        <table>counciltrees</table>
        <key>ogc_fid</key>
        <dynamic>true</dynamic>
</mapping>
```

   You can also set a `filter` XML Tag so that only features that meet certain criteria will be considered for selection (see Spatial Mapper).

   If there are no Active Layers from the Spatial Engine you need to access already defined in the `entity.xml` file, then you'll need to add a new tag for that Spatial Engine. For simplicity, give this mapper an id of `mapper.<spatialengine_id>` as shown below.

```
<mapper:mapper id='mapper.postgis'>
        <mapping>
                <entity>counciltrees</entity>
                <spatialengine>postgis</spatialengine>
                <table>counciltrees</table>
                <key>ogc_fid</key>
                <dynamic>true</dynamic>
        </mapping>
</mapper:mapper>
```

If you refresh your browser you'll now see that the icon for that Active Layer (i.e. `Council Trees`) has changed to a layer-like icon. This icon indicates that the Active Layer is now a spatial layer.



7. If you now run a spatial selection, the selected features will be highlighted in the *Map Window*. The number of features selected will be reported as the "Selection Size" in the *Status Bar* at the bottom right corner of the Weave window.

Also refer to the following sections of the Weave System Administrator Guides:

- Entity
- Spatial Mapper

Related articles

- How to Add a Layer to the Table of Contents
- How to Add a New Active Layer
- How to Add a New Data Grid
- How to Add a New Search
- How to Create a Quick Search

How to Add a New Data Grid

When you have made a layer selectable (by defining it as an *Active Layer*) you will normally want to create a *Data Grid* to display the attributes of the layer or to display data from related tables.

Step-by-step guide

1. From the Weave client, make note of the *Active Layer* for which you want to create a *Data Grid*. Note also the `datasource` value (from the `datasource.xml` file), database schema name and table name.

2. Open your `data.xml` file for editing. This file will contain other Data Grids that have been configured for your site.

3. A new Data Definition needs to be created as this is what dictates which fields will be shown in the Data Grid. If you are familiar with the fields you want in the Data Grid, you can use the other Data Definitions as a guide and add in as many `parameter` XML Entity lines that you need. However an easier way is to use the Administration Tool Console window or Weave's OSGi console (this will save you some typing and reduces the likelihood of a typing mistake). In the console, run the `dbmd` command with the details of the table you want to report on.

```
dbmd conf <datasource> <table> <schema>
        (schema name is not needed if table name is unique in
the database)
e.g.:
        dbmd conf gis counciltrees
```

4. Copy the XML generated at Step 3 into the `data.xml` file. Notice there are some `FIX_ME` values that need to be updated, i.e. you will need to give the Data Definition an identifier (with the `id` XML Attribute) and supply the key field (with the `key` XML Attribute). This is shown in the example below.

Raw XML as output from the dbmd command:

```
<data:datadefinition id='FIX_ME'>
        <datasourcedataconnection key='FIX_ME' table='mann.
counciltrees' datasource='gis'>
                <parameter name='fid' label='FID' column='fid'
type='int'/>
        <parameter name='tree_no' label='Tree No.'
column='tree_no' type='float'/>
                <parameter name='species' label='Species'
column='species'/>
                <parameter name='height' label='Height'
column='height'/>
        <parameter name='dbh' label='DBH' column='dbh'/>
        <parameter name='hazard' label='Hazard' column='hazard'/>
        <parameter name='condition' label='Condition'
column='condition'/>
        </datasourcedataconnection>
</data:datadefinition>
```

Updated XML ready for use:

```
<data:datadefinition id='counciltrees'>
        <datasourcedataconnection key='fid' table='mann.
counciltrees' datasource='gis'>
                   <parameter name='fid' label='FID'
column='fid' type='int' />
                <parameter name='tree_no' label='Tree No.'
column='tree_no' type='float'/>
        <parameter name='species' label='Species'
column='species'/>
        <parameter name='height' label='Height' column='height'/>
        <parameter name='dbh' label='DBH' column='dbh'/>
        <parameter name='hazard' label='Hazard' column='hazard'/>
        <parameter name='condition' label='Condition'
column='condition'/>
        </datasourcedataconnection>
</data:datadefinition>
```

5. The output from the `dbmd` command reports on all the fields from the table so delete or comment out any lines that you do not want to include in the Data Grid. If you want all the fields to appear in the Data Grid, then you can delete all the parameter lines. By default, all fields from the table will be displayed in the Data Grid when no fields are given as shown below.

```
<data:datadefinition id='counciltrees'>
        <datasourcedataconnection key='fid' table='mann.
counciltrees' datasource='gis'/>
</data:datadefinition>
```

6. Some Weave practitioners use the “`dd_`” prefix as their naming convention for Data Definitions. This is not necessary as long as there are no other Data Definitions with the same `id` value. However if your Weave configuration is using the "`dd_`" prefix, or

another naming convention, then for consistency it is best to continue using this convention.

7. Add a `data:data` XML Element to link the Data Definition (defined above) to the Entity (from the `entity.xml` file) as shown below.

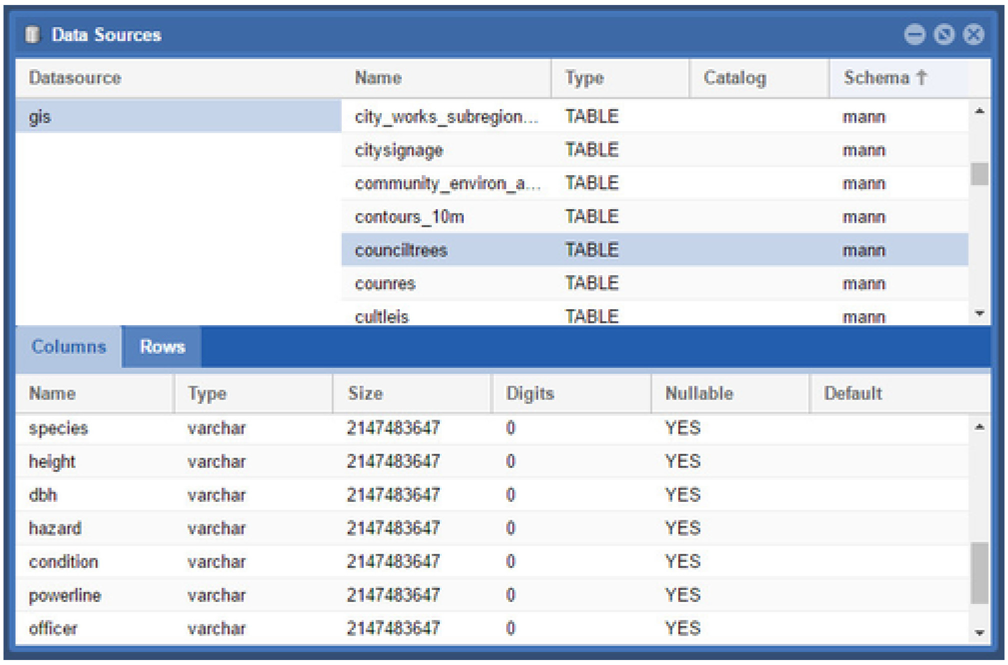
```
<data:data id="counciltrees" label="Council Trees" entity="
counciltrees" datadefinition="counciltrees" />
```

8. This has now set up a Data Definition that will be used in the *Data Grid* when a *Search* is run or the *Identify* tool is used.

9. Often, you'll want to display fields from more than one table or database in your *Data Grid*, particularly if you want to link spatial features to data stored in an aspatial database. This can be done by adding a `from` and `where` clause in the `data:datadefinition` XML Tag before listing the fields to retrieve. It's best to fully qualify the field in the `column` XML Attribute when reading from more than one table, i.e. `<schema>.<table>.<field>.` In the example below additional fields are being read from the same database but a different schema and table (i.e. datasource = `gis`, schema = `cid`, table = `treedetails`). (Note, you can only use a table from a second database if that table is *known* to the first database.)

```
<data:datadefinition id='counciltrees'>
        <datasourcedataconnection key='fid' table='mann.
counciltrees' datasource='gis'>
                <from table="cid.treedetails"/>
                <where clause="cid.treedetails.fid=mann.
counciltrees.fid"/>
                           <parameter name='fid' label='FID'
column='fid' type='int' />
                        <parameter name='tree_no' label='Tree
No.' column='tree_no' type='float'/>
                        ...
            <parameter name='condition' label='Condition'
column='condition'/>
                        <parameter name='priority'
label='Priority' column='priority'/>
       </datasourcedataconnection>
</data:datadefinition>
```

10. The *Data Sources Tool* in the Administration Tool reports on the tables that you can use in Data Definitions (and thus Data Grids). It is useful to explore data and check field names, etc. for data you wish to include in a Data Grid.



Datasource	Name	Type	Catalog	Schema ↑
gis	city_works_subregion...	TABLE		mann
	citysignage	TABLE		mann
	community_environ_a...	TABLE		mann
	contours_10m	TABLE		mann
	counciltrees	TABLE		mann
	counres	TABLE		mann
	cultleis	TABLE		mann

Name	Type	Size	Digits	Nullable	Default
species	varchar	2147483647	0	YES	
height	varchar	2147483647	0	YES	
dbh	varchar	2147483647	0	YES	
hazard	varchar	2147483647	0	YES	
condition	varchar	2147483647	0	YES	
powerline	varchar	2147483647	0	YES	
officer	varchar	2147483647	0	YES	

Also refer to the following sections of the Weave System Administrator Guides:

- Datasource Data Connection

Related articles

- How to Add a Column to a Data Grid
- How to Add a New Data Grid
- How to Include an Image in a Data Grid or Map Tip

How to Add a New Search

When you have made a layer selectable (by defining it as an *Active Layer*) you can now define a Search that will run on that layer.

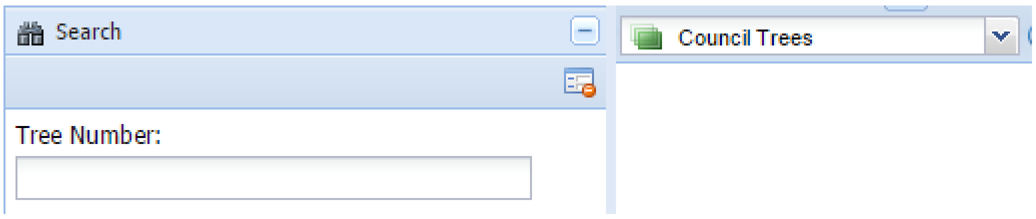
Step-by-step guide

1. Make note of the Active Layer from the Weave client for which you want to define a *Search*.
2. Open your `entity.xml` and search for the Active Layer name. It will appear as the `label` in an `entity` XML Tag. Make note of the value of the `id` XML Attribute and search for it within the same file. It will appear in a `mapping` XML Tag. Make a note of the values of the `spatialengine`, `table` and `key` XML Tags.
3. Ensure the *Search* panel is visible somewhere in the Weave application by adding the `<view id='com.cohga.html.client.main.searchView'>` item to the `client_main.xml` file. This will most likely have been done during your Weave installation.
4. During your Weave installation, some searches may have been configured in the `search.xml`. In order to get the correct structure of the search XML, the easiest thing to do is to copy an existing `search:attribute` XML Element, and then edit it.
5. Modify the XML Attribute values so that they reflect your database and the fields you want to search. Keep in mind that databases have different rules regarding the case of field names (PostgreSQL uses lower case, Oracle uses upper case, SQL Server uses mixed case, etc.).
6. Start with a simple search where the input is typed in by the Weave user as shown in the example below.

```
<search:attribute id='counciltrees' entity='counciltrees'
displayname='Council Trees by Details' datasource='gis'
table='mann.counciltrees' key='fid'>
        <search:parameter id='treenum' promptText='Tree Number'
dataType='float' alignment='right' helptext='Type in the tree
```

```
number' column='tree_no' />
</search:attribute>
```

The resulting search in Weave will appear as shown below:



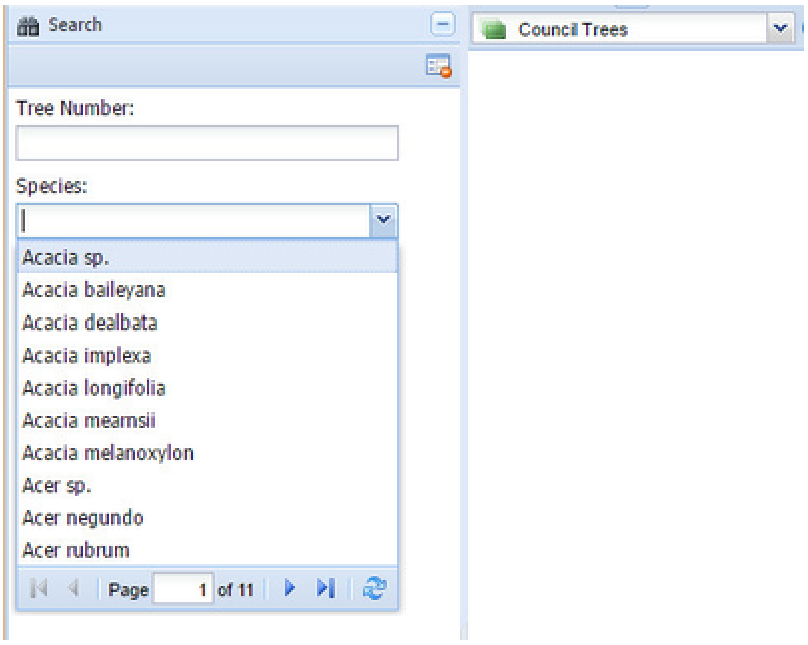
7. If you want to have a drop-down list as part of your Search, the `controlType` XML Attribute of "`listbox`" needs to be included as shown below. The content of the list is specified by the `dataset` XML Attribute (which will be created in the next step).

```
<search:attribute id='counciltrees' entity='counciltrees'
displayname='Council Trees by Details' datasource='gis'
table='mann.counciltrees' key='fid'>
        <search:parameter id='treenum' promptText='Tree Number'
dataType='float' alignment='right' helptext='Type in the tree
number' column='tree_no' />
        <search:parameter id='treespecies' label='Species'
controlType='listbox' column='species' dataset='specieslist'
labelcolumn='label' valuecolumn='value' />
</search:attribute>
```

8. The above step adds a drop-down list to the *Search* panel but the list won't have any values in it. To populate the list, a Data Definition needs to be created in the `data.xml` file. The Data Definition specifies which values will be in the list.

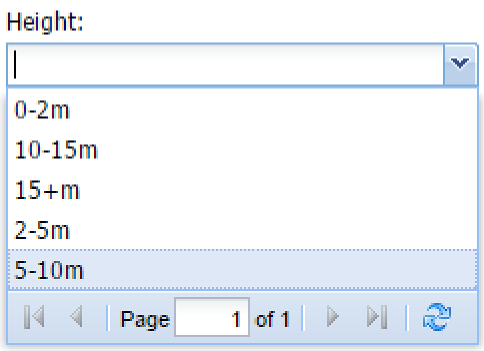
```
<data:datadefinition id='specieslist>
        <datasourcedataconnection datasource='gis'
table='counciltrees' prefix='DISTINCT'>
                <parameter type='string' name='label'
label='Label' column='species' />
                <parameter type='string' name='value'
label='Value' column='species' />
        </datasourcedataconnection>
</data:datadefinition>
```

9. You will need to refresh your Weave browser to see the new list in the *Search* panel.



10. The *Config Tool* in the Administration Tool reports on all the valid Data Definitions created so you can use it to find and view the definitions of all the Data Definitions. The Data Definition you created for your drop-down list should be shown, together with others used in *Data Grids*. Press the *Open* button to view the XML for it.

11. In some cases, the values in a list may appear in the incorrect order, e.g. they are stored as a string but they are quantitative rather than qualitative, as shown in the example below.

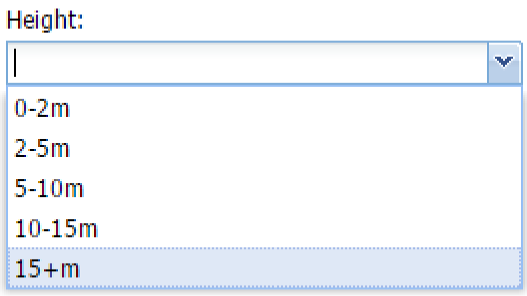


In order to get the list items displayed in a different order, you can set up this search parameter as a value list. This requires some configuration of the XML file but it will produce a better result for the Weave user and is a good alternative to an automatic list if the number of possible values is small.

12. Add the following code to your `search.xml` file.

```
<search:parameter id="treeheight" label="Height" controlType="
listbox" column="height">
        <list label="0-2m" value="0-2m"/>
        <list label="2-5m" value="2-5m"/>
        <list label="5-10m" value="5-10m"/>
        <list label="10-15m" value="10-15m"/>
        <list label="15+m" value="15+m"/>
</search:parameter>
```

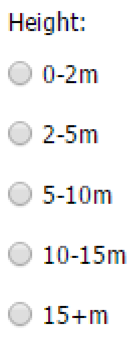
The resulting list in Weave will appear as shown below:



13. You could also use radio buttons to get the listing in the correct order. This works for a short list and five items is probably the maximum number of items you'd want to use for this type of list.

```
<search:parameter id="treeheight" label="Height" controlType="
radiobutton" column="height">
        <list label="0-2m" value="0-2m"/>
        <list label="2-5m" value="2-5m"/>
        <list label="5-10m" value="5-10m"/>
        <list label="10-15m" value="10-15m"/>
        <list label="15+m" value="15+m"/>
</search:parameter>
```

The resulting list in Weave will appear as shown below:



Also refer to the following sections of the Weave System Administrator Guides:

- Attribute Search
- Spatial Search

Related articles

- How to Add a Layer to the Table of Contents
- How to Add a New Active Layer
- How to Add a New Data Grid
- How to Add a New Search
- How to Create a Quick Search

How to Clear Your Browser Cache

When you are getting unexpected results from Weave, it can be useful to clear your browser's cache. The browser may be holding on to (caching) information that you don't want it to. This can be the case when you receive "invalid session" errors or "502 - Bad Gateway" errors.

An alternative to clearing your browser cache is to use Incognito/Private windows when browsing. However, please note that if you have multiple Incognito/Private windows/tabs, browser cache storage will be shared across those incognito windows/tabs (i.e. closing one incognito window while others are open does not wipe out the cache storage).

This page explains how to clear the cache for the most popular web browsers.

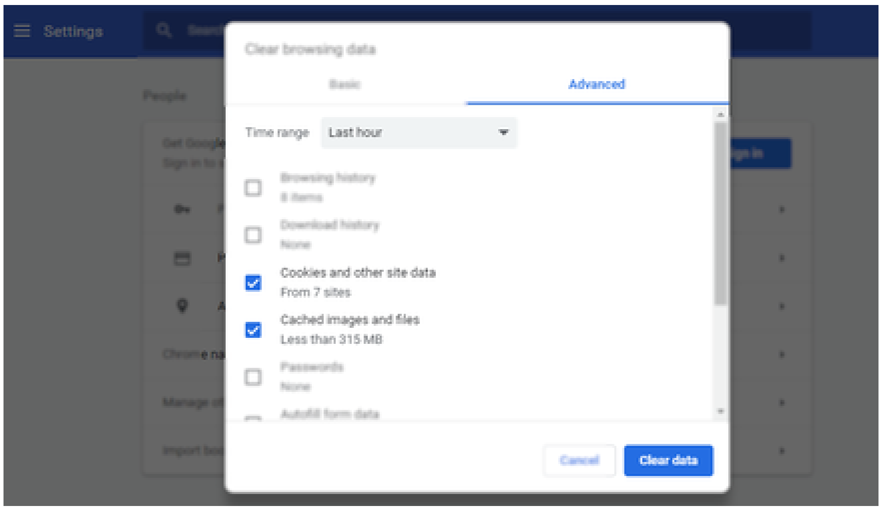
Step-by-step guide

1. For IE 11:
    a. Open the `Tools` menu (Alt-x).
    b. Go to `Internet options`.
    c. In the `General` tab, in the `Browsing history` section, press the `Delete` button.

2. For Edge:
    a. Open `Settings and more` menu (Alt-x).
    b. Go to Settings.
    c. In the `Clear browsing data`, press the `Choose what to clear` button.
    d. Select all the options and press the `Clear` button.

3. For Mozilla Firefox:
    a. Open menu > Options > Privacy & Security.
    b. In the `Cached Web Content` section, press the `Clear Now` button.

4. For Google Chrome:
    a. Click on the Chrome menu button (⋮).
    b. Select 'More tools' 'Clear browsing data' (keyboard shortcut: Ctrl+Shift+Del) .
    c. Select the 'Advanced' tab.
    d. Select an appropriate time range from the drop down list.
    e. Check:
        i. Cookies and other site data
        ii. Cached images and files
    f. Press the '`Clear data'` button.



Clear browsing data

Basic | Advanced

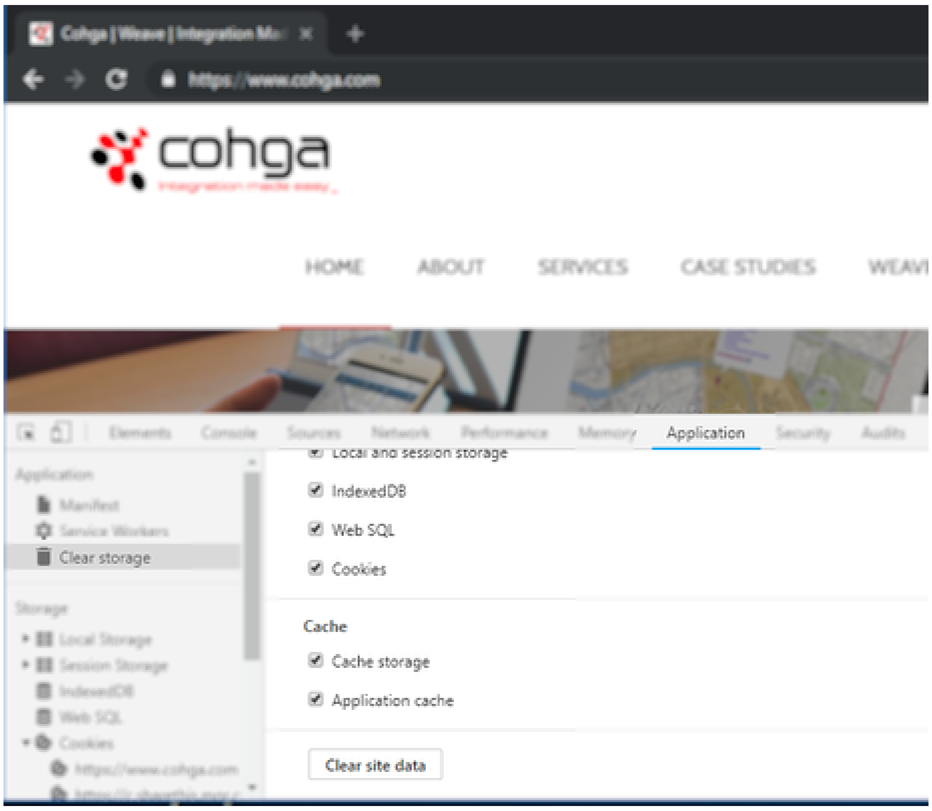
Time range: Last hour

- [ ] Browsing history — 6 items
- [ ] Download history — None
- [x] Cookies and other site data — From 7 sites
- [x] Cached images and files — Less than 315 MB
- [ ] Passwords — None
- Autofill form data

Note: If your problem persists after clearing history for the 'Last hour', select 'All time' from the 'Time range' drop down as an older cookie may be interfering with the workflow.

Chrome Advanced: Only clear cache and cookie data associated with the current website:

a. Press F12 key to open Developer Tools window
b. Navigate to 'Application' tab
c. Select 'Clear storage' node on the left hand side
d. Make sure all items on the right hand side pane are selected
e. Click 'Clear site data' button.



Related articles

- How to Get the Most From Weave Logging
- How to Clear Your Browser Cache
- How to Debug a BIRT Report

How to Create a Quick Search

When you want to create a flexible search option you can create a Quick Search. A Quick Search looks across a range of layers for a match against entered criteria. The Quick Search function is independent of the Active Layer. However before it will work, a Quick Search requires an Index to be created.

Step-by-step guide

1. Open the `client_main.xml` file for editing. Ensure the *Quick Search* text box is visible somewhere in the Weave application by searching for "`indexcombo`". If it's not there, add the following XML to your file:

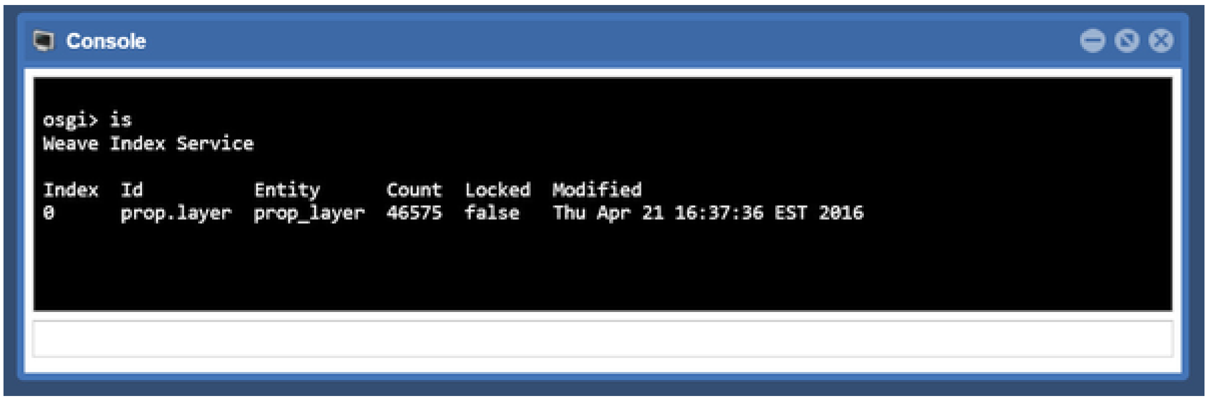
```
<item component="weave.indexcombo" minScale="2500" all="true"/>
```

2. An Index is linked to an Entity so the Entity should exist in the `entity.xml` file. If it doesn't exist, add it to the `entity.xml` file.

3. The details of the Quick Search itself need to be added to the `index.xml` file. This specifies what results you want returned. The process of setting up an Index (and thus a Quick Search) is well documented in the Weave Configuration Reference (Indexing). Please refer to it as there are many parameters that can be set on your Index, and to achieve the desired result it is necessary to understand the way the Index is created and the parameters that can be defined.

4. Though an Index is defined in the `index.xml` file the Index itself is not automatically created, and is created manually or at a set time. Once you've defined an Index, it can be created manually using the Weave Administration Tool Console or Weave's OSGi console.

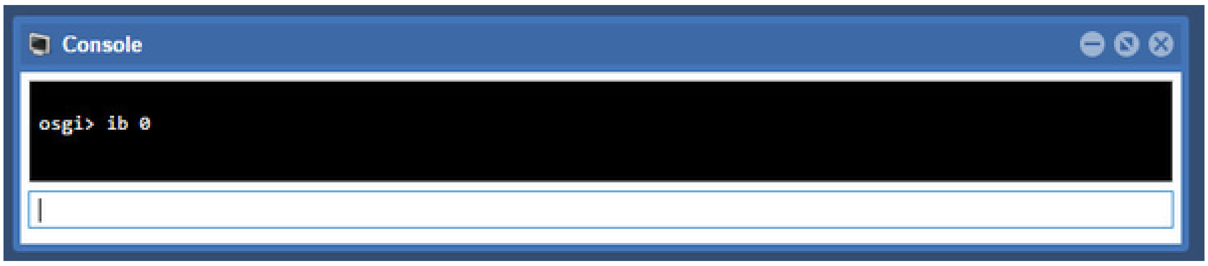
5. Open the console window and type the command `is` to return a list of all indexes.



```
osgi> is
Weave Index Service

Index  Id          Entity      Count  Locked  Modified
0      prop.layer  prop_layer  46575  false   Thu Apr 21 16:37:36 EST 2016
```

6. Note the Index number that is of interest (the number under the `Index` heading). Then use the command `ib <space separated list of index ids>` to rebuild the index of interest. The index will be built in the background so check the Weave log file for the results of the index build.



```
osgi> ib 0
```

Also refer to the following sections of the Weave System Administrator Guides:

- Indexing

Related articles

- How to Add a New Active Layer
- How to Add a New Search
- How to Create a Quick Search
- How to Use a List in an Attribute Search

How to Create an Image Slider

There are many things to consider when adding an Image Slider in order to get the result you desire. Also affecting the workings of your slider is what is serving the images and whether they have the ability to set image transparency. These considerations are covered in other sections of the Weave Configuration Reference.

The instructions below cover how to add a simple image slider when you have a number of vector layers in your Table of Contents and are looking to provide the user the ability to turn on one of three images via an Image Slider. This is by no means the limit of what can be done, it's merely an example of what can be done.

Step-by-step guide

1. Add the `imageTransSlider` component to your client XML file. This is added to the Map View and will use existing Map Engines.

```
                  <view id="com.cohga.html.client.map.mapView">
                          <label>Map</label>
                          <location>center</location>
                          ...

  <statusbar>
                                  <item>Image Slider</item>
                                  <item></item>
```

```
                                <item component="weave.
compassImageTransSlider">
                                        <width>120</width>
                                        <value>0</value>
                                        <increment>10</increment>
                                        <disabled>false<
/disabled>
                                        <mapEngine>mapengine.
google.satellite</mapEngine>
                                </item>
                                <item>Google Satellite</item>
                        </statusbar>
                        ...
                        <mapEngine id="mapengine.google.
satellite" />
                        <mapEngine id="mapengine.main">
                                <options>
                                        ...
                                </options>
                        </mapEngine>
                        ...
                </view>
```

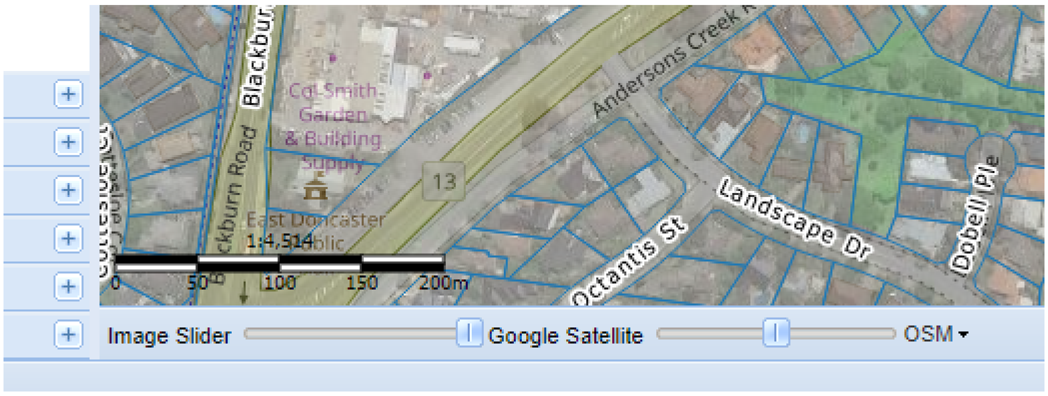
2. Primary image slider will transition between the layers in your ToC and the Map Engine given in the `imageTransSlider`. More information about the Image Slider can be found here.

3. To add a secondary imagery slider more configuration is required.

```
                <view id="com.cohga.html.client.map.mapView">
                        <label>Map</label>
                        <location>center</location>
                        ...

<statusbar>
                                <item>Image Slider</item>
                                <item></item>
                                <item component="weave.
compassImageTransSlider">
                                        <width>120</width>
                                        <value>0</value>
                                        <increment>10</increment>
                                        <disabled>false<
/disabled>
                                        <mapEngine>mapengine.
google.satellite</mapEngine>
                                </item>
                                <item>Google Satellite</item>
                                <item><
/item>
                                <item component="weave.
compassSecImageTransSlider">
                                        <width>120</width>
                                        <value>0</value>
                                        <increment>10</increment>
                                        <mapEngine>mapengine.
google.alt</mapEngine>
                                </item>
                                <item component="weave.
compassSecImageChooser">
                                        <layer id="0" text="
Google Maps Alt" mapengine="mapengine.google.alt" isDefault="
true"/>
                                        <layer id="0" text="
Google Maps Alt Terrain" mapengine="mapengine.google.alt.
terrain" />
                                        <layer id="0" text="OSM"
mapengine="mapengine.osm" />
                                </item>
                        </statusbar>
                        ...
                        <mapEngine id="mapengine.google.
satellite" />
                        <mapEngine id="mapengine.google.alt" />
                        <mapEngine id="mapengine.google.alt.
terrain" />
                        <mapEngine id="mapengine.osm" />
                        <mapEngine id="mapengine.main">
                                <options>
                                        ...
                                </options>
                        </mapEngine>
                        ...
                </view>
```

This will result in two sliders as shown below.



Also refer to the following sections of the Weave System Administrator Guides:

- Client Views Map

Related articles

- How to Create an Image Slider

How to Debug a BIRT Report

When you have a BIRT Report that is not giving the output you were expecting, it can be useful to obtain debugging information from BIRT.

Step-by-step guide

1. Create a file called `birt.properties` file in the `..\weave\platform\workspace` directory (e.g. `C:\weave_dev\platform\workspace`) and add the line below:

   File to enable BIRT logging

   ```
   logLevel=FINE
   ```

2. Restart Weave.
3. This will enable BIRT logging which will be written to the `..\weave\platform\workspace\.birt\logs` folder (e.g. `C:\weave_dev\platform\workspace\.birt\logs`).
4. Run the problematic BIRT report and check the newly created log file for errors.
5. To turn off this logging, either comment out the line as shown below (that way you have the file for the next time you need to use it) and restart Weave,

   File to disable BIRT logging

   ```
   #logLevel=FINE
   ```

   or delete the `birt.properties` file and restart Weave.

Also refer to the following sections of the Weave System Administrator Guides:

- BIRT

Related articles

- How to Debug a BIRT Report
- How to Edit an A3 Portrait BIRT Report

How to Edit an A3 Portrait BIRT Report

There is currently a bug with the version of BIRT Weave 2.5 which means it's not possible to edit some reports through the *BIRT Report Designer*. So if you are using Weave 2.5 or greater, and have an A3 Portrait report, follow the steps below in order to fix your report so that it can be edited with the *BIRT Report Designer*.

This BIRT bug is not related to any mapping component so will apply to all A3 Portrait BIRT reports, including those that don't have map elements.

Step-by-step guide

For an existing report:

1. When you attempt to open a report that has an A3 portrait size set, BIRT will give an error (*"Failed to create the part's controls"*). So you can create a new A3 portrait report however once closed, it cannot be opened in BIRT.
2. To be able to modify this report through BIRT you need to change the size of the page using a text editor.
3. Navigate to your BIRT reports folder (e.g. `C:\weave\platform\workspace\reports`).
4. Open the relevant `.rptdesgn` file in a text editor.
5. Search for "a3" in the file.
6. This will generally appear in the "Simple MasterPage" section.
7. Replace that with the lines below, or something similar depending on what else you have defined in your Simple MasterPage section.

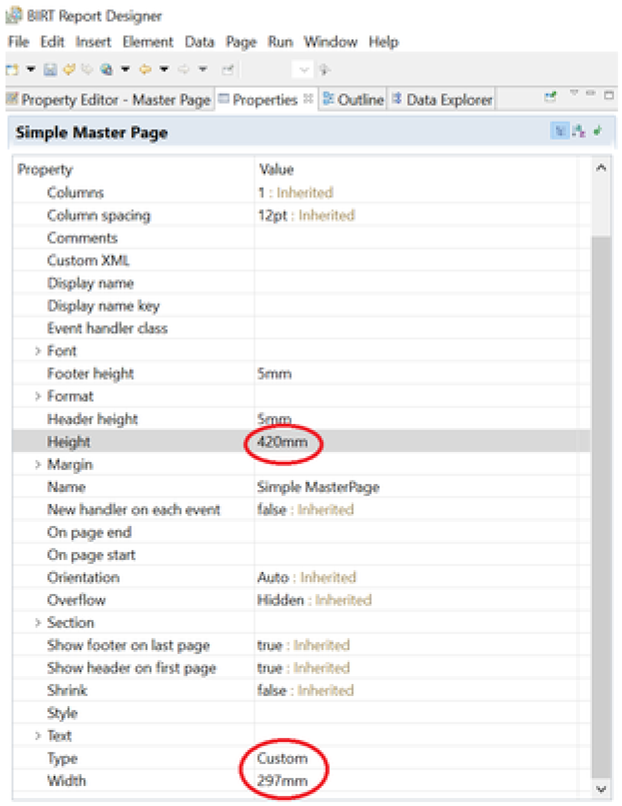
BIRT report file extract

```
<page-setup>
     <simple-master-page name="Simple MasterPage" id="2">
         <property name="type">custom</property>
         <property name="orientation">portrait</property>
         <property name="height">420mm</property>
         <property name="width">297mm</property>
          ...
     </simple-master-page>
</page-setup>
```

8. The report should now be able to be opened with the *BIRT Report Designer* and edited as per other reports.

For a new report:

1. Start the *BIRT Report Designing* by double clicking the `birt.cmd` file in your Weave folder.
2. Choose `File > New > New Report` and give the report a name and press `Finish`.
3. Open the `Properties` window (`Windows > Show View > Property Editor`).
4. Click on the `Advanced ` tab.
5. In the *Simple Master Page* section, scroll to the `Type` option. Even though you want an A3 size report, don't chose A3 from the list of options, choose "Custom".
6. Or in the `General` tab under the `Type` option, choose "Custom".
7. Then set the `Width` parameter to "297mm" and the `Height` parameter to "420mm" as per the image below.


BIRT Report Designer
File Edit Insert Element Data Page Run Window Help
Property Editor - Master Page
Properties
Outline
Data Explorer

Simple Master Page

Property	Value
Columns	1 : Inherited
Column spacing	12pt : Inherited
Comments	
Custom XML	
Display name	
Display name key	
Event handler class	
› Font	
Footer height	5mm
› Format	
Header height	5mm
Height	420mm
› Margin	
Name	Simple MasterPage
New handler on each event	false : Inherited
On page end	
On page start	
Orientation	Auto : Inherited
Overflow	Hidden : Inherited
› Section	
Show footer on last page	true : Inherited
Show header on first page	true : Inherited
Shrink	false : Inherited
Style	
› Text	
Type	Custom
Width	297mm

8. Save the report file and close the *BIRT Report Designer*.
9. The report should now be able to be opened with the *BIRT Report Designer* and edited as per other reports.

Also refer to the following sections of the Weave System Administrator Guides:

- BIRT
- Report

Related articles

- How to Debug a BIRT Report
- How to Edit an A3 Portrait BIRT Report

How to Encrypt Passwords in Configuration Files

When you need to include sensitive passwords in your config files and want to ensure that they're encrypted so they can't be easily seen be users looking at the configuration files.

Step-by-step guide

1. Use the osgi console `encrypt` command to encrypt a clear text password
   a. Go to to the osgi console and type `encrypt <password>` and press return e.g.

Encrypting a password

```
osgi> encrypt My$upaSekrutP@sswurd
ENCKPXBCTMGGMKBGKPK
```

2. Copy the encrypted password from the console and paste it back into the configuration file replacing the original password

Before encrypting password

```
<username>bob</username>
<password>My$upaSekrutP@sswurd<password>
```

After encrypting password

```
<username>bob</username>
<password>ENCKPXBCTMGGMKBGKPK<password>
```

The encryption key used to encrypt the passwords is stored in the file called `private.key` located in the Weave folder (e.g. `C:\weave\platform\workspace`).

This file is generated the first time Weave starts and will be unique for each Weave instance, which means that encrypting the same password with different instances of Weave would result in different encrypted text.

This means that if you want to re-use the same configuration files on multiple Weave instances, for example development and production, then you need to either:

- Copy the `private.key` file from the server where the passwords were encrypted to the other server, so the encryption /decryption key is the same on both instances.
- Extract the passwords from being directly included in the XML and `set` them in a separate XML file that you `include` into your config and encrypt the passwords on both Weave instances and have the different passwords set in the password XML file.

Additionally, for extra security the permissions on that file should be set to only allow the users running the Weave instance to read the `private.key` file.

Also refer to the following sections of the Weave System Administrator Guides:

- Client Actions
- Table Of Contents

Related articles

- How to Encrypt Passwords in Configuration Files
- Encrypting content in the security.xml file

How to fix a PKIX Path Building Failed Error

If you're seeing errors in the weave.log file like:

PKIX exception output

```
com.esri.arcgisws.runtime.exception.ArcGISWebServiceException: sun.
security.validator.ValidatorException: PKIX path building failed: sun.
security.provider.certpath.SunCertPathBuilderException: unable to
find valid certification path to requested target
```

it's likely cause by Weave trying to communicate with an external service that uses a certificate that Weave doesn't know about, for example if the service uses a self-signed certificate.

This could happen if you're connecting to an ArcGIS server that you're hosting within your own organisation where it's been installed and setup using its own certificate (which is the default when ArcGIS Server is installed).

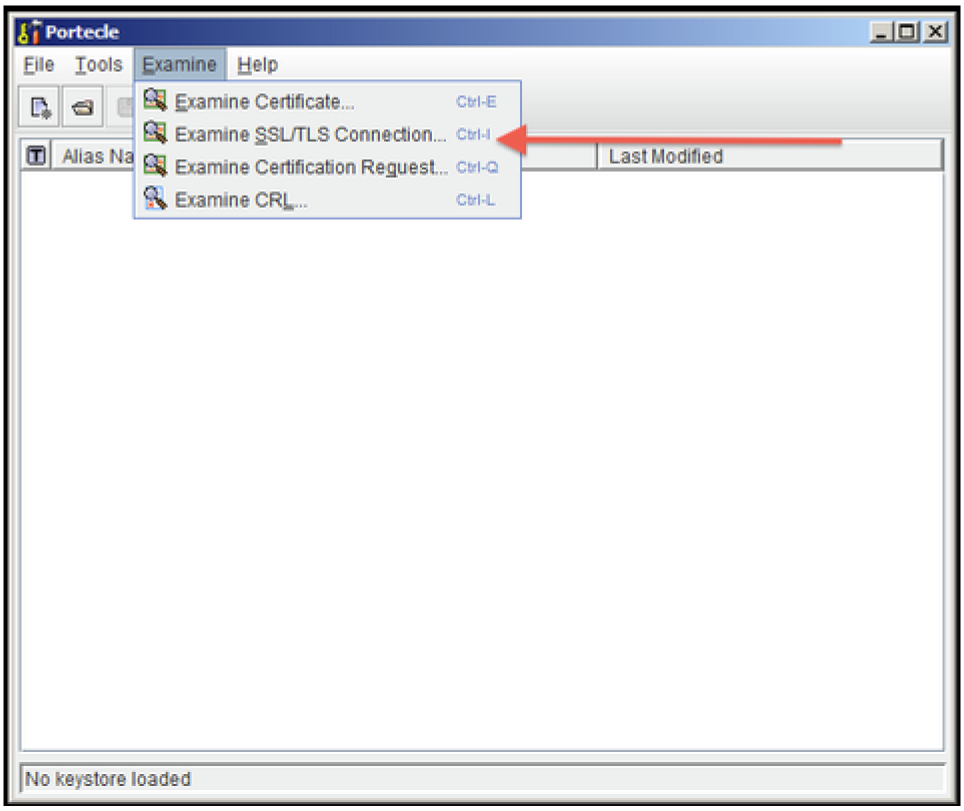
To resolve this problem the Weave certificate store needs to be updated to include information about the certificate that the server is presenting. This can be done manually using the `openssl` tool but it can also be done using a graphical tool Portecle.

Step-by-step guide

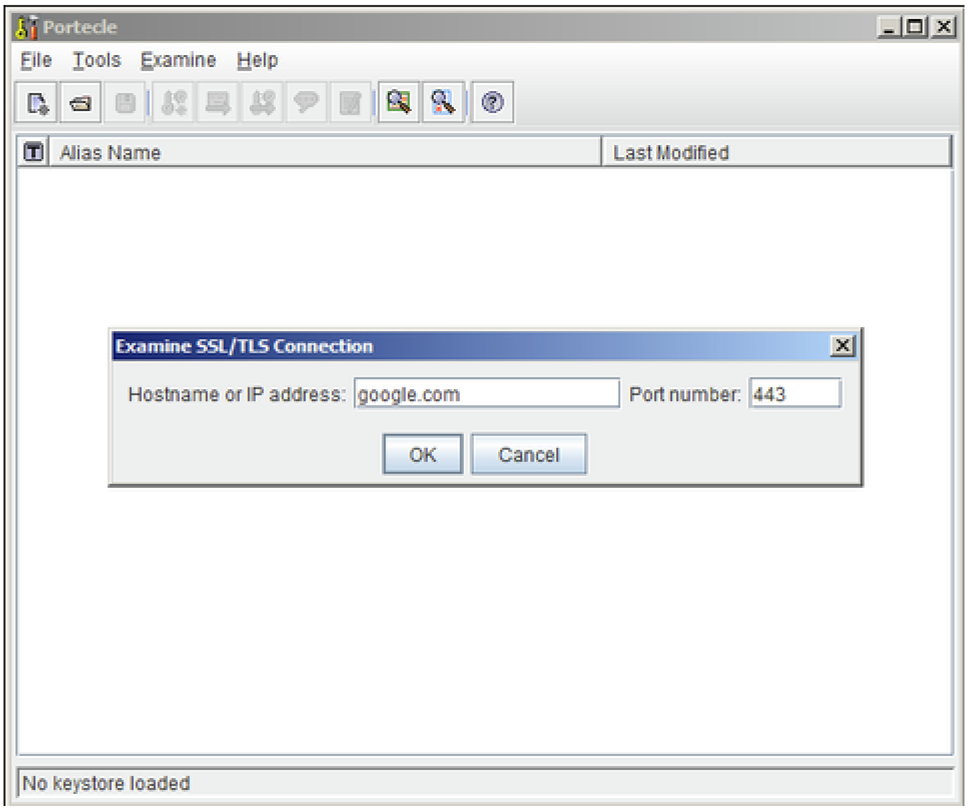
1. Download and install the Portecle application onto the server that runs Weave. For this example we'll download the `portecle.zip` file and assume it's been unzipped into the `c:\temp` directory (the current version of Portecle at the time of writing is 1.11, which is included in the paths within the `.zip` file).
2. Start the Portecle application in the same way you'd start a Weave updater. You may be able to double click on the `portecle.jar` file but if that doesn't work you'll need open a command prompt and start it manually, e.g. assuming Weave is installed at `c:\weave\` and Portecle has been extracted to `c:\temp\portecle-1.11\` you can open `cmd.exe` and run `java.exe` with the `-jar` option and the path to the `portecle.jar` file

```
C:\Users\sforbes> c:\weave\jre\bin\java.exe -jar c:
\temp\portecle-1.11\portecle.jar
```

3. Select the **Examine** menu and then click **Examine SSL/TLS Connection**:

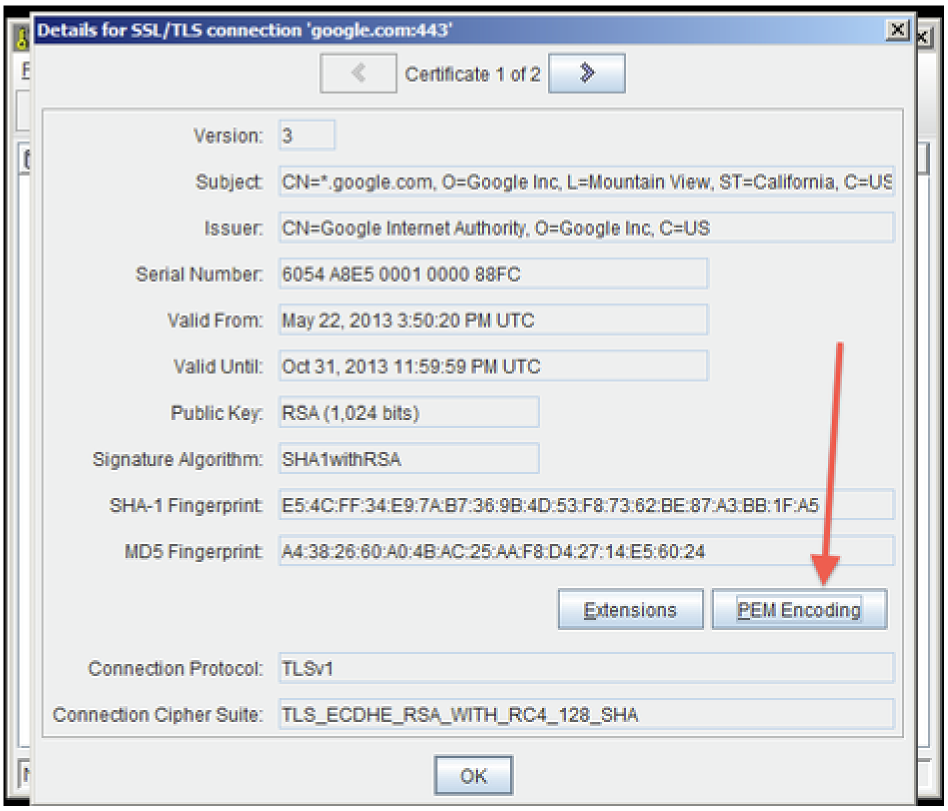

Portecle
File Tools Examine Help
Examine Certificate... Ctrl-E
Examine SSL/TLS Connection... Ctrl-I
Examine Certification Request... Ctrl-Q
Examine CRL... Ctrl-L
Alias Na
Last Modified
No keystore loaded

4. Enter the SSL Host and Port of the target system. In this example we're looking at google.com but it'll likely be the name of your ArcGIS Server host, the `weave.log` file should provide the information just before the PKIX exception (note that if the information in the log does not report a port number then it's is probably 443):



Hostname or IP address: google.com Port number: 443

5. Wait for it to load, then select the public certificate and click on PEM (you will likely only have a single certificate to choose from, but this screen shot shows two available):

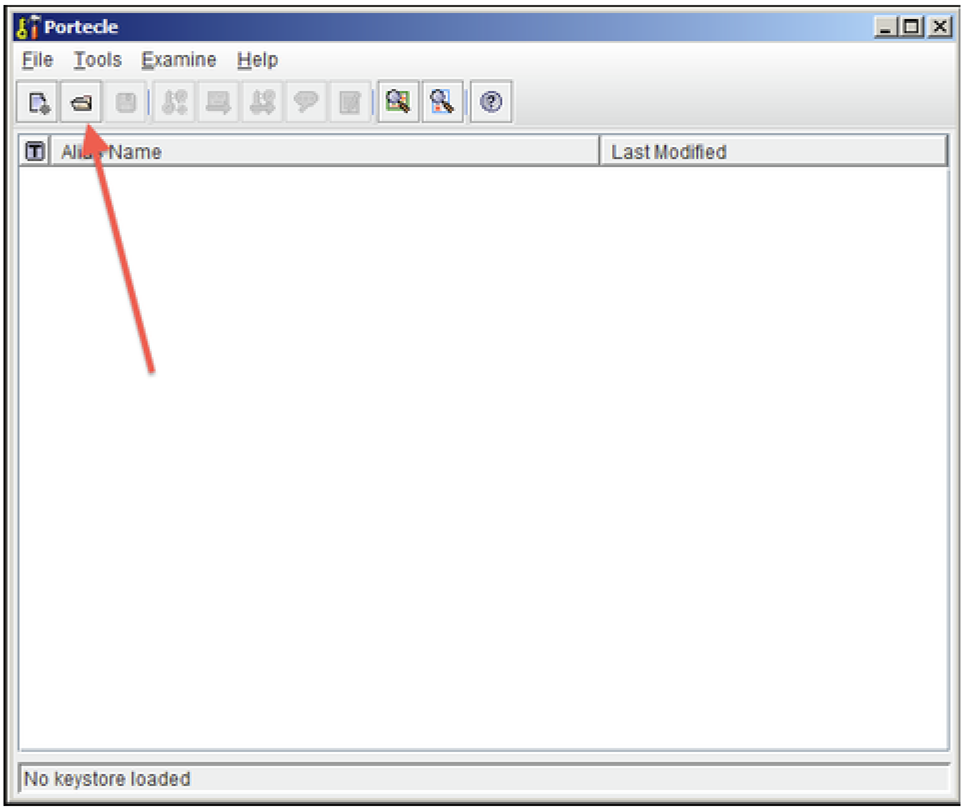

Details for SSL/TLS connection 'google.com:443'

Certificate 1 of 2

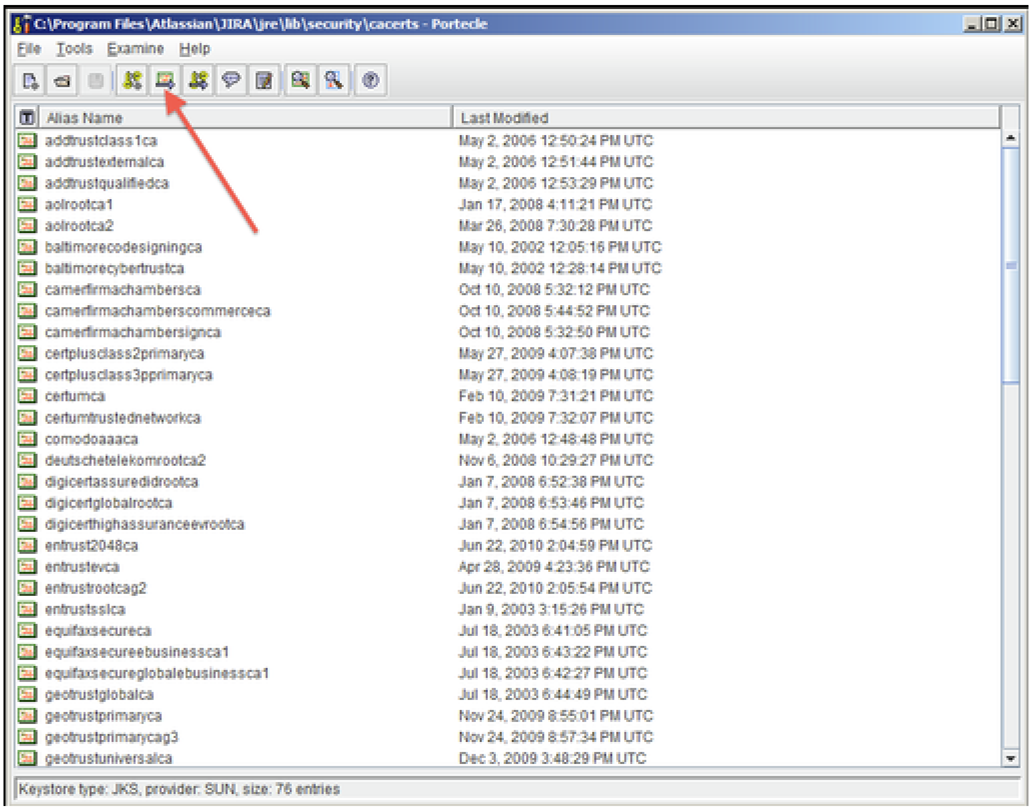
Version: 3
Subject: CN=*.google.com, O=Google Inc, L=Mountain View, ST=California, C=US
Issuer: CN=Google Internet Authority, O=Google Inc, C=US
Serial Number: 6054 A8E5 0001 0000 88FC
Valid From: May 22, 2013 3:50:20 PM UTC
Valid Until: Oct 31, 2013 11:59:59 PM UTC
Public Key: RSA (1,024 bits)
Signature Algorithm: SHA1withRSA
SHA-1 Fingerprint: E5:4C:FF:34:E9:7A:B7:36:9B:4D:53:F8:73:62:BE:87:A3:BB:1F:A5
MD5 Fingerprint: A4:38:26:60:A0:4B:AC:25:AA:F8:D4:27:14:E5:60:24
Connection Protocol: TLSv1
Connection Cipher Suite: TLS_ECDHE_RSA_WITH_RC4_128_SHA

6. Export the certificate and save it to a file.

7. Go back to the main screen and select the **Open an existing keystore from disk** option, and select the **`cacerts`** file from the Weave Java runtime (the default password is **`changeit`**), for example **`C:\weave\jre\lib\security\cacerts`**:


No keystore loaded

8. Select the **Import a trusted certificate into the loaded keystore** button:

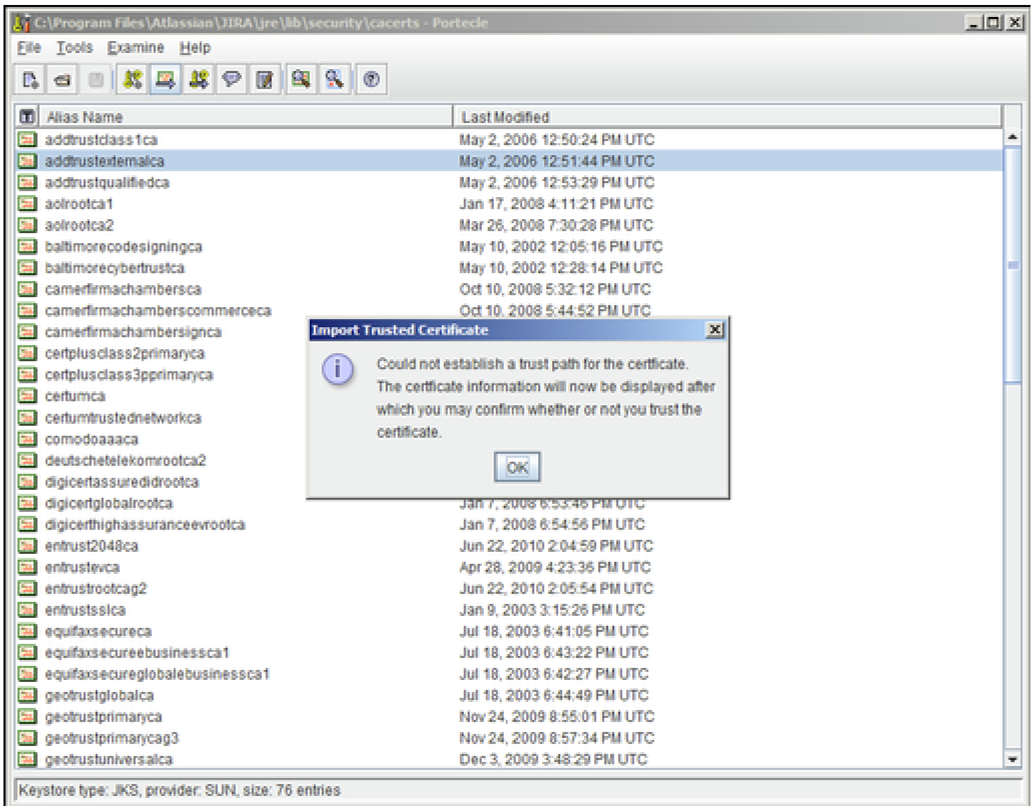

C:\Program Files\Atlassian\JIRA\jre\lib\security\cacerts - Portecle

File Tools Examine Help

Alias Name	Last Modified
addtrustclass1ca	May 2, 2006 12:50:24 PM UTC
addtrustexternalca	May 2, 2006 12:51:44 PM UTC
addtrustqualifiedca	May 2, 2006 12:53:29 PM UTC
aolrootca1	Jan 17, 2008 4:11:21 PM UTC
aolrootca2	Mar 26, 2008 7:30:28 PM UTC
baltimorecodesigningca	May 10, 2002 12:05:16 PM UTC
baltimorecybertrustca	May 10, 2002 12:28:14 PM UTC
camerfirmachambersca	Oct 10, 2008 5:32:12 PM UTC
camerfirmachamberscommerceca	Oct 10, 2008 5:44:52 PM UTC
camerfirmachambersignca	Oct 10, 2008 5:32:50 PM UTC
certplusclass2primaryca	May 27, 2009 4:07:38 PM UTC
certplusclass3pprimaryca	May 27, 2009 4:08:19 PM UTC
certumca	Feb 10, 2009 7:31:21 PM UTC
certumtrustednetworkca	Feb 10, 2009 7:32:07 PM UTC
comodoaaaca	May 2, 2006 12:48:48 PM UTC
deutschetelekomrootca2	Nov 6, 2008 10:29:27 PM UTC
digicertassuredidrootca	Jan 7, 2008 6:52:38 PM UTC
digicertglobalrootca	Jan 7, 2008 6:53:46 PM UTC
digicerthighassuranceevrootca	Jan 7, 2008 6:54:56 PM UTC
entrust2048ca	Jun 22, 2010 2:04:59 PM UTC
entrustevca	Apr 28, 2009 4:23:36 PM UTC
entrustrootcag2	Jun 22, 2010 2:05:54 PM UTC
entrustsslca	Jan 9, 2003 3:15:26 PM UTC
equifaxsecureca	Jul 18, 2003 6:41:05 PM UTC
equifaxsecureebusinessca1	Jul 18, 2003 6:43:22 PM UTC
equifaxsecureglobalebusinessca1	Jul 18, 2003 6:42:27 PM UTC
geotrustglobalca	Jul 18, 2003 6:44:49 PM UTC
geotrustprimaryca	Nov 24, 2009 8:55:01 PM UTC
geotrustprimarycag3	Nov 24, 2009 8:57:34 PM UTC
geotrustuniversalca	Dec 3, 2009 3:48:29 PM UTC

Keystore type: JKS, provider: SUN, size: 76 entries

9. Select the certificate that was saved in Step 6 and confirm that you trust it, giving it an appropriate alias and verifying that it should be added:


C:\Program Files\Atlassian\JIRA\jre\lib\security\cacerts - Portecle

File Tools Examine Help

Alias Name	Last Modified
addtrustclass1ca	May 2, 2006 12:50:24 PM UTC
addtrustexternalca	May 2, 2006 12:51:44 PM UTC
addtrustqualifiedca	May 2, 2006 12:53:29 PM UTC
aolrootca1	Jan 17, 2008 4:11:21 PM UTC
aolrootca2	Mar 26, 2008 7:30:28 PM UTC
baltimorecodesigningca	May 10, 2002 12:05:16 PM UTC
baltimorecybertrustca	May 10, 2002 12:28:14 PM UTC
camerfirmachambersca	Oct 10, 2008 5:32:12 PM UTC
camerfirmachamberscommerceca	Oct 10, 2008 5:44:52 PM UTC
camerfirmachambersignca	
certplusclass2primaryca	
certplusclass3pprimaryca	
certumca	
certumtrustednetworkca	
comodoaaaca	
deutschetelekomrootca2	
digicertassuredidrootca	
digicertglobalrootca	Jan 7, 2008 6:53:46 PM UTC
digicerthighassuranceevrootca	Jan 7, 2008 6:54:56 PM UTC
entrust2048ca	Jun 22, 2010 2:04:59 PM UTC
entrustevca	Apr 28, 2009 4:23:36 PM UTC
entrustrootcag2	Jun 22, 2010 2:05:54 PM UTC
entrustsslca	Jan 9, 2003 3:15:26 PM UTC
equifaxsecureca	Jul 18, 2003 6:41:05 PM UTC
equifaxsecureebusinessca1	Jul 18, 2003 6:43:22 PM UTC
equifaxsecureglobalebusinessca1	Jul 18, 2003 6:42:27 PM UTC
geotrustglobalca	Jul 18, 2003 6:44:49 PM UTC
geotrustprimaryca	Nov 24, 2009 8:55:01 PM UTC
geotrustprimarycag3	Nov 24, 2009 8:57:34 PM UTC
geotrustuniversalca	Dec 3, 2009 3:48:29 PM UTC

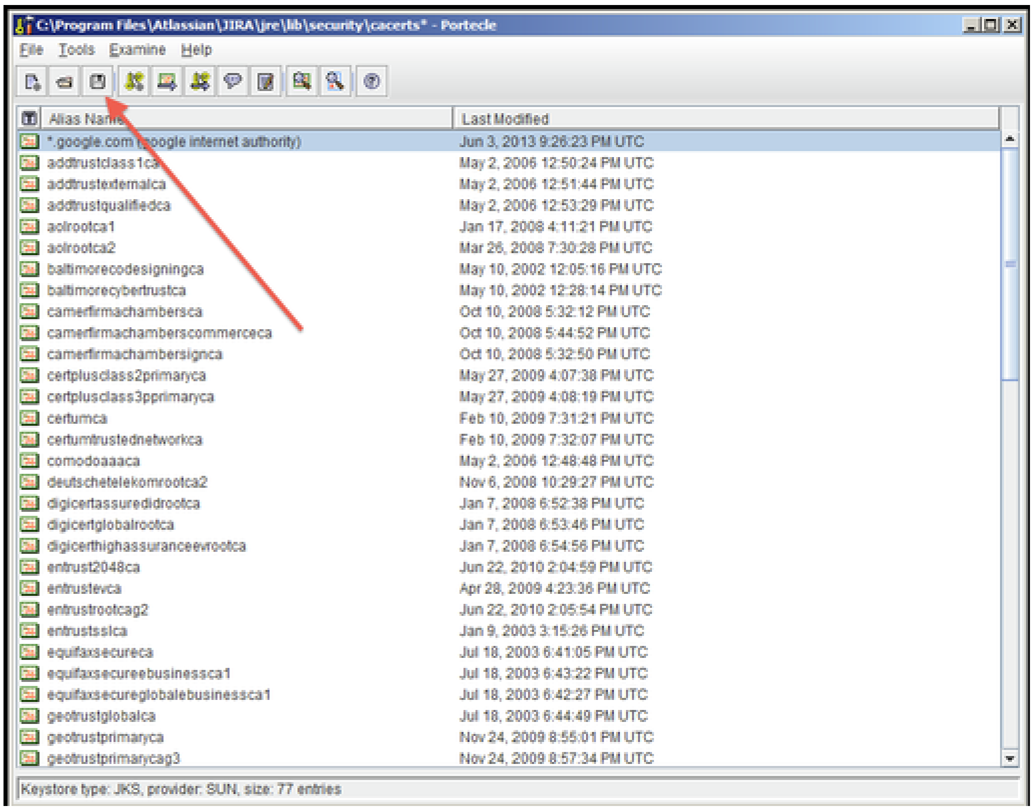
Import Trusted Certificate

Could not establish a trust path for the certficate. The certficate information will now be displayed after which you may confirm whether or not you trust the certificate.

OK

Keystore type: JKS, provider: SUN, size: 76 entries

10. Save the Key Store to disk:


C:\Program Files\Atlassian\JIRA\jre\lib\security\cacerts* - Portecle
File
Tools
Examine
Help

Alias Name	Last Modified
*.google.com (google internet authority)	Jun 3, 2013 9:26:23 PM UTC
addtrustclass1ca	May 2, 2006 12:50:24 PM UTC
addtrustexternalca	May 2, 2006 12:51:44 PM UTC
addtrustqualifiedca	May 2, 2006 12:53:29 PM UTC
aolrootca1	Jan 17, 2008 4:11:21 PM UTC
aolrootca2	Mar 26, 2008 7:30:28 PM UTC
baltimorecodesigningca	May 10, 2002 12:05:16 PM UTC
baltimorecybertrustca	May 10, 2002 12:28:14 PM UTC
camerfirmachambersca	Oct 10, 2008 5:32:12 PM UTC
camerfirmachamberscommerceca	Oct 10, 2008 5:44:52 PM UTC
camerfirmachambersignca	Oct 10, 2008 5:32:50 PM UTC
certplusclass2primaryca	May 27, 2009 4:07:38 PM UTC
certplusclass3pprimaryca	May 27, 2009 4:08:19 PM UTC
certumca	Feb 10, 2009 7:31:21 PM UTC
certumtrustednetworkca	Feb 10, 2009 7:32:07 PM UTC
comodoaaaca	May 2, 2006 12:48:48 PM UTC
deutschetelekomrootca2	Nov 6, 2008 10:29:27 PM UTC
digicertassuredidrootca	Jan 7, 2008 6:52:38 PM UTC
digicertglobalrootca	Jan 7, 2008 6:53:46 PM UTC
digicerthighassuranceevrootca	Jan 7, 2008 6:54:56 PM UTC
entrust2048ca	Jun 22, 2010 2:04:59 PM UTC
entrustevca	Apr 28, 2009 4:23:36 PM UTC
entrustrootcag2	Jun 22, 2010 2:05:54 PM UTC
entrustsslca	Jan 9, 2003 3:15:26 PM UTC
equifaxsecureca	Jul 18, 2003 6:41:05 PM UTC
equifaxsecureebusinessca1	Jul 18, 2003 6:43:22 PM UTC
equifaxsecureglobalebusinessca1	Jul 18, 2003 6:42:27 PM UTC
geotrustglobalca	Jul 18, 2003 6:44:49 PM UTC
geotrustprimaryca	Nov 24, 2009 8:55:01 PM UTC
geotrustprimarycag3	Nov 24, 2009 8:57:34 PM UTC

Keystore type: JKS, provider: SUN, size: 77 entries

11. Restart Weave and verify that the external service can now be connected to.

If your Weave server is running on a server where you cannot run a graphical application then you can copy the `cacerts` file to another PC, follow these instructions, then copy the file back to the Weave server.

Related articles

Connecting to SSL services

Unable to Connect to SSL Services due to PKIX Path Building Failed

- How to Get the Most From Weave Logging
- How to fix a PKIX Path Building Failed Error
- How to Clear Your Browser Cache

How to Get the Most From Weave Logging

Weave logs can give you a lot of information to help you deal with problems if they arise in your Weave installation. Here are a few pointers that can help you get the most out of your Weave logging.

As of Weave 2.6.7, the logging system for Weave has changed from log4j to Logback. So while this page may refer to log4j (instead of Logback) and `logging.properties` (instead of `logging.xml`), the information is still relevant and has been extended to reflect the change in logging library, Refer to the Administration Guide Logging page for more information.

Step-by-step guide

1. In most cases the first step in resolving issues with Weave involves looking at the `weave.log` file (in the `..\weave\logs\` folder) and searching for any `ERROR` or `WARN` messages that may be related to the problem being experienced. This file keeps a running log of what Weave is doing at any moment, and by triggering the action that's causing the problem and then looking at the last few lines that were written to the log file, it may be possible to determine the cause of the problem.
    a. This is assuming that logging to `weave.log` is enabled, which it was by default prior to Weave 2.6.7. As of Weave 2.6.7, by default `weave.log` output is only generated when Weave is run from a service/daemon, if it is not then the output will be sent to the console window. But this can be changed by editing the default `logging.xml` file.
    b. As of Weave 2.6.7, detailed log information is always available via the Log Tool in the Admin Tool and also via the "dump logs" OSGi console command, independent of the `weave.log` file.
2. If you have administration access to the Weave server then you can change the logging level yourself. This is done by changing a parameter in the `logging.properties` files (in the ..`\weave\platform\workspace` folder) to one of the settings as shown

below. Enabling the DEBUG log level is important when trying to identify the cause of a problem, and can be done on a running system without having to restart Weave (though you have wait for up to 30 seconds for the change to be registered).

Logging for problem debugging and installation

```
log4j.rootCategory=DEBUG, stdout, file, admin
```

Logging for normal operation

```
log4j.rootCategory=INFO, stdout, file, admin
```

For Weave running as a service output to log file and Admin Tool only

```
log4j.rootCategory=INFO, file, admin
```

   a. For Weave 2.6.7 and later, this is done in `logging.xml` and the root log level is always set to debug so you must set the log level within each appended to change the level. The root category tag lists the logging appenders that are used to generate the different types of log output and by default use configuration options to try and determine the best type of log output depending upon where Weave is running, e.g. from the console, as a service, within an IDE. But this can be changed to remove the conditions and allow you to directly choose which types of log output your require.

3. If your initial investigation of the `weave.log` file doesn't reveal anything then the following method should be followed:
   1. Ensure that the logging level in the `...\weave\platform\workspace\logging.properties` file is set to `DEBUG` (this includes both the `log4j.rootCategory` and `log4j.appender.file.Threshold` settings).
   2. Stop Weave.
   3. Delete `...\weave\logs\weave.log`.
   4. Start Weave.
   5. Perform the action that causes the problem.
   6. Log a Support Ticket and include the `weave.log` file to us (after compressing it if it's large).

   This method asks for Weave to be stopped and the log file cleaned out so that we can ensure that the log file covers as little as possible *other* than what is related to the problem since log files can grow quite large. If the problem is being experienced on a production system then it's not essential that the log file be cleaned out since Weave cannot be stopped. But, enabling the `DEBUG` log level is important, and can be done without having to restart Weave.
   a. This still applies for Weave 2.6.7 and later, it's just the changes are made to `logging.xml`, not `logging.properties`
4. Use the *Log Tool* in the Weave Administration Tool to view the debug, information, warning and error statements from your Weave while it's in operation.
5. By default the Weave log file will not contain the date of the entry written in the log file, just the time, as there is a new log file created every day. However if you need the date of the log entry recorded in your log file then you can make this change in the `logging.properties` files (in the ..`\weave\platform\workspace` folder) as shown below.

   Default log file parameters

```
# Define a logger that sends it's output to a rolling log file
log4j.appender.file=org.apache.log4j.DailyRollingFileAppender
log4j.appender.file.Threshold=DEBUG
log4j.appender.file.File=../logs/weave.log
log4j.appender.file.Encoding=UTF-8
log4j.appender.file.layout=org.apache.log4j.PatternLayout
log4j.appender.file.layout.ConversionPattern=%d{ABSOLUTE} %-5p [%
t] %x %c "%m"%n
```

   Change the `ConversionPattern` setting (in the last line) as shown below.

   Log file with date for each entry

```
# Define a logger that sends it's output to a rolling log file
log4j.appender.file=org.apache.log4j.DailyRollingFileAppender
log4j.appender.file.Threshold=DEBUG
log4j.appender.file.File=../logs/weave.log
log4j.appender.file.Encoding=UTF-8
log4j.appender.file.layout=org.apache.log4j.PatternLayout
log4j.appender.file.layout.ConversionPattern=%d %-5p [%t] %x %c
"%m"%n
```

The same change can be applied to other logging methods you have defined in your `logging.properties` file (e.g. `log4j.appender.stdout, log4j.appender.email`).

   a. This is no longer the default in Weave 2.6.7 and later, the full date and time of the log event is recorded in the log file.

6. If you are running a search and it's not giving you the results you are expecting then it's good to see the actual SQL command that is being sent to the database. Search for "NativeSQL" in your log file, copy the SQL statement, and run it in your database query software or in the Data Source SQL tool in the Weave Administration Tool.
7. If you are having problems on the Weave client side (browser) you can get more information returned from the browser. To do this add ";debug" to the startup URL (e.g. `http://server1:8080/weave/main.html;debug`). Pressing the `F12` key, in Internet Explorer, once Weave is open, will open up the browser console and display any client side errors. Restarting Weave with the normal URL will return it to the standard way of operating.

Learn to love your log files!

Related articles

- How to Get the Most From Weave Logging
- How to fix a PKIX Path Building Failed Error
- How to Clear Your Browser Cache

How to Include an Image in a Data Grid or Map Tip

When you have images associated with records in a table it is possible to include the image in a Data Grid or Map Tip. In order for the image to be included in either of these, it must be part of the Data Definition used.

Step-by-step guide

1. Identify which table and field contains the image name and location, or just the image name. This will be needed so that every feature has its corresponding file link.
2. In your Data Definition, add a parameter for the image file link. It's best to enclose the file path with a CDATA section to ensure all characters are interpreted correctly. Weave will need to serve the image using a web server so put in the correct reference to your images through the web server, e.g. `http://servername/path/image_name`
3. If you do not have a web server you can use Weave's web server. However for this to work you need to store your images in a folder recognised by the Weave web server. For example you could create an "`images`" folder in your `webapps` folder (`C:\weave\webapps\images`) and put all your linking images in there.
4. As mentioned in Step 3, if you have put your images in a Weave web server folder, and the name of the image is stored in a field called `tree_photo`, then to include those images in a Data Definition you could use the code below.

```
<parameter type="string" name="imageparam" label="Image">
        <column>
                <![CDATA['<img src="http://localhost:8080/images
/'||tree_photo||'" height="30" width="30" />']]>
        </column>
</parameter>
```

If the image file extension is not stored in the field of the image name, you can append it to the file name as shown below.

```
<parameter type="string" name="imageparam" label="Image">
        <column>
                <![CDATA['<img src="http://localhost:8080/images
/'||tree_photo||'.jpg" height="30" width="30" />']]>
        </column>
</parameter>
```

5. Alternatively you could add a link to the image to your Data Definition, rather than displaying the image itself. In this case you set the parameter of `type` as "url" as shown below.

```
<parameter type="url" name="imagelink" label="Image Link"
column="'http://localhost:8080/images/'||tree_photo"/>
```

6. If you do not want to use a web server you could also use the `file:` reference to add a link to your images in a Data Definition, but this only works if the user is using Internet Explorer as their browser. If you do want to try the `file:` reference, two alternatives for this are shown below, where `mycomp` is the name of the computer where the images are stored in the folder `C:/images/flora.`

```
<parameter type='url' name="link1" label="UNC File Link" text="
open">
        <column>
                <![CDATA['file://mycomp/c$/images/flora/tree1.
png']]>
        </column>
</parameter>

<parameter type='url' name="link2" label="UNC Directory Link"
text="open">
        <column>
                <![CDATA['file://mycomp/c$/images/flora/']]>
        </column>
</parameter>
```

There are security implications with allowing web sites to access your local file system via the `file:` reference so in some environments this may not work and you may have to use the `http:` reference as described in Step 5.

For further information on Data Definitions: Datasource Data Connection

Related articles

- How to Serve Additional Web Content
- How to Include an Image in a Data Grid or Map Tip

How to Include Special Characters in Weave Configuration Files

When you need to use special characters in your Weave configuration files it's necessary to use the CDATA XML construct so that all the characters are correctly interpreted.

All text in an XML file will be interpreted by an XML parser. Some characters (<, >, &, !) have a special meaning in XML so if they are used in an XML file their markup language meaning will be used rather than their literal character meaning. In order to use the literal character meaning of special characters the text is put in a CDATA section. CDATA sections can be used anywhere character data would be used, and anything within the CDATA section will be ignored by the XML parser (that is, the XML markup meaning will be ignored).

Step-by-step guide

1. CDATA sections are used a lot when specifying file paths and locations to ensure all characters in the path and file name are interpreted correctly:

   ```
   <wms:url><![CDATA[http://services.land.vic.gov.au/catalogue
   /publicproxy/guest/dv_geoserver/wms]]></wms:url>
   ```

   Or when setting filters:

   ```
   <filter><![CDATA[plan_code <> 0]]></filter>
   ```

2. CDATA sections begin with the string

   ```
   <![CDATA[
   ```

3. CDATA sections end with the string

   ```
    ]]>
   ```

4. An alternative to using CDATA sections is to use the markup characters to get the resulting character you need. However this can get messy, using CDATA sections makes your XML files much more readable and reduces the likelihood of mistakes.

   For example, the same `filter` parameter given in Step 1 can be written as:

   ```
   <filter>plan_code &lt;&gt; 0</filter>
   ```

Further reading: https://en.wikipedia.org/wiki/CDATA

Related articles

- How to Set a System Property
- How to Increase the Spatial Operation Limit
- How to Include Special Characters in Weave Configuration Files

How to Increase the Spatial Operation Limit

Some spatial operations can take a very long time to execute and if the user creates a large selection and attempts to perform one of those operations against the selection this could result in high CPU usage for an extended period while the operation is performed. Usually, the browser has timed out before the operation completes and, making matters worse, the user may re-try the operation causing even more load to be placed on the server.

To try and mitigate this risk, Weave implements a limit on the number of entities that can be selected before one of these operations is performed. The default for this limit is 200 but can be changed by setting a system property named `weave.spatial.operation.limit` to the new limit, or `0` to disable it.

Instructions

1. Edit `...\weave\service\conf\wrapper.conf` and add an additional system property
   a. Locate the last additional property in the file (they are usually defined at the end of the file)
      i. The "additional properties" are those listed in the form `wrapper.java.additional.## = ...`
   b. Find the number of the last set additional property, and increase it by one
   c. Create a new additional property (with the new incremented property number) and set the value to `-Dweave.spatial.operation.limit=###`
      i. Where `###` is the new value you want to use for the limit (or `0`)
2. If you use `startup.cmd` or `startup.sh` to start Weave then you need to do the same thing for those files but the way the properties are set is different
   a. For those files the `JAVA_OPTS` variable must be updated.
   b. There are a number of existing properties in those files that should make it clear how to add an additional property. Follow the same set out as those existing properties for this new property.

Example wrapper.conf line Assuming previous additional property was number 21 and 500 is the new limit.

```
wrapper.java.additional.22 = -Dweave.spatial.operation.limit=500
```

Example for startup.cmd

```
SET JAVA_OPTS=%JAVA_OPTS% -Dweave.spatial.operation.limit=500
```

Example for startup.sh

```
JAVA_OPTS="$JAVA_OPTS -Dweave.spatial.operation.limit=500"
```

Related articles

- How to Get a Weave Support Dump using a URI
- How to Get the Most From Weave Logging
- How to Serve Additional Web Content
- How to Set a System Property
- How to fix a PKIX Path Building Failed Error

How to Install a New Version of Weave

If you already have a version of Weave running, you can upgrade it to the latest version released, or you can perform a completely new installation. This page covers how to perform a new installation.

⚠ This page is valid if you are already running a 2.5 version of Weave and are looking to install a later minor release (a point release).

Step-by-step guide

1. Download the Installer file as directed by the new release email from the Weave Support System or from the Latest Downloads page.

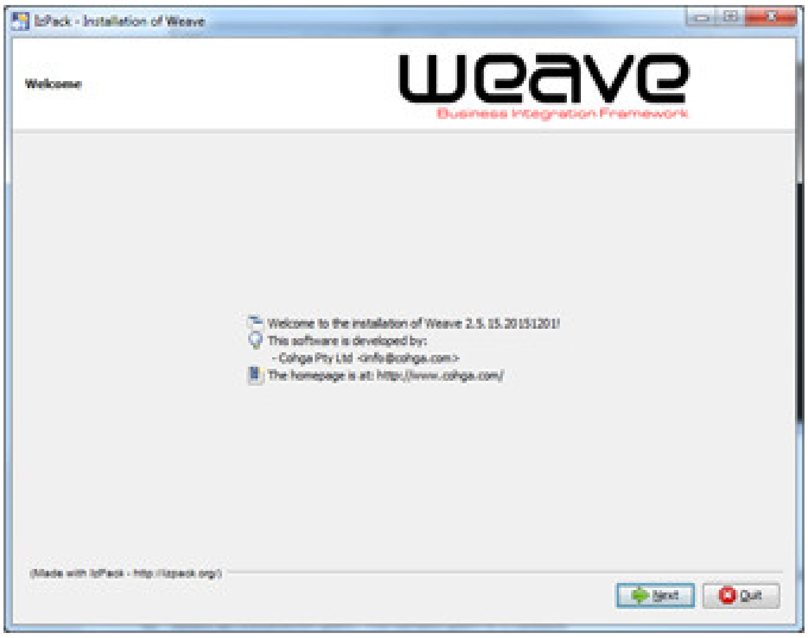
2. The Installer file is a `.jar` file so it needs Java to run. If you don't have Java installed but have a previous version of Weave installed, you can use the Java executable that is installed with that Weave (e.g. `c:\weave\jre\bin\java.exe`).

3. It is recommended that you install the new Weave release into a new folder so you can still have your current version running. You can then remove/rename the previous installation once you are satisfied that the new installation is ready. The steps below give you an example of how this would be done in a Windows environment.

   Current Weave folders = `c:\weave`
   New Weave folders = `c:\weave-2-5-18`
   Weave downloaded file = `c:\downloads\weave-installer-2.5.18.jar`

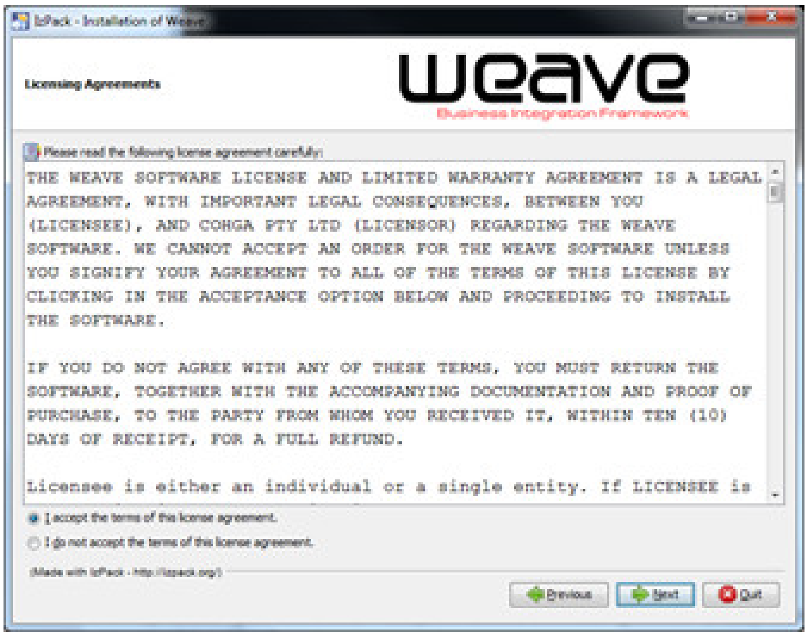
   - On a Windows Command Line, run:
     ```
     cd c:\weave\jre\bin
     java -jar c:\downloads\weave-installer-2.5.18.jar c:\weave-2-5-18
     ```
     The last parameter in the line directly above (`c:\weave-2-5-18`) is optional, if given this will be set the default folder into which Weave will be installed, if not given the default location will be `c:\weave`. You also get a chance to change this during the installation process.

4. **Welcome:** Running the Installer `.jar` file will open up the Weave installation wizard. Click *Next* on the initial screen.


IzPack - Installation of Weave
Welcome
weave
Business Integration Framework
(Made with IzPack - http://izpack.org/)
Next
Quit

Welcome to the installation of Weave 2.5.15.20151201!
This software is developed by:
- Cohga Pty Ltd <info@cohga.com>
The homepage is at: http://www.cohga.com/

5. **License Agreements:** Click on "I accept the terms of the license agreement" to accept the license agreement and click *Next.*


IzPack - Installation of Weave
Licensing Agreements
weave
Business Integration Framework
(Made with IzPack - http://izpack.org/)
Previous
Next
Quit

Please read the following license agreement carefully:

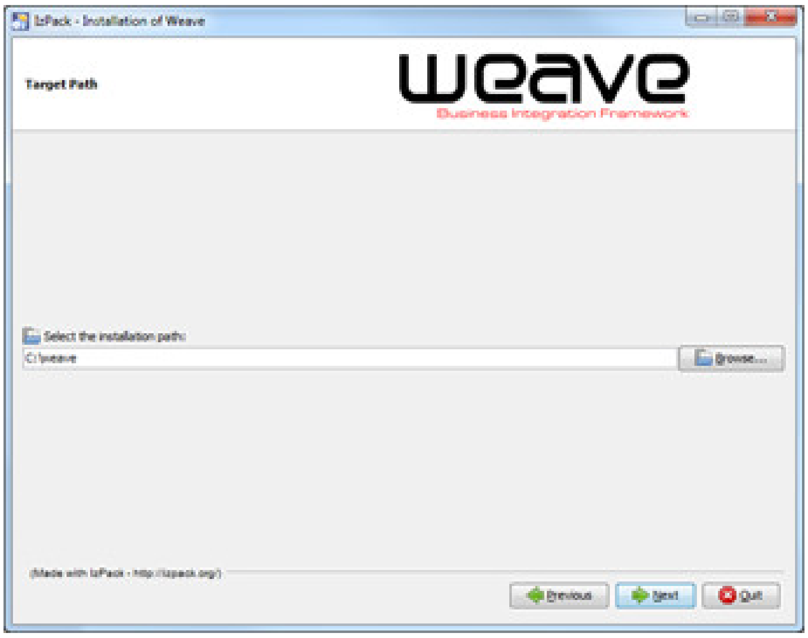
```
THE WEAVE SOFTWARE LICENSE AND LIMITED WARRANTY AGREEMENT IS A LEGAL
AGREEMENT, WITH IMPORTANT LEGAL CONSEQUENCES, BETWEEN YOU
(LICENSEE), AND COHGA PTY LTD (LICENSOR) REGARDING THE WEAVE
SOFTWARE. WE CANNOT ACCEPT AN ORDER FOR THE WEAVE SOFTWARE UNLESS
YOU SIGNIFY YOUR AGREEMENT TO ALL OF THE TERMS OF THIS LICENSE BY
CLICKING IN THE ACCEPTANCE OPTION BELOW AND PROCEEDING TO INSTALL
THE SOFTWARE.

IF YOU DO NOT AGREE WITH ANY OF THESE TERMS, YOU MUST RETURN THE
SOFTWARE, TOGETHER WITH THE ACCOMPANYING DOCUMENTATION AND PROOF OF
PURCHASE, TO THE PARTY FROM WHOM YOU RECEIVED IT, WITHIN TEN (10)
DAYS OF RECEIPT, FOR A FULL REFUND.

Licensee is either an individual or a single entity. If LICENSEE is
```

[✓ I accept the terms of this license agreement.]
I do not accept the terms of this license agreement.

6. **Target Path:** Select an installation path and click *Next*. The default path is `C:\weave` as per Step 3.

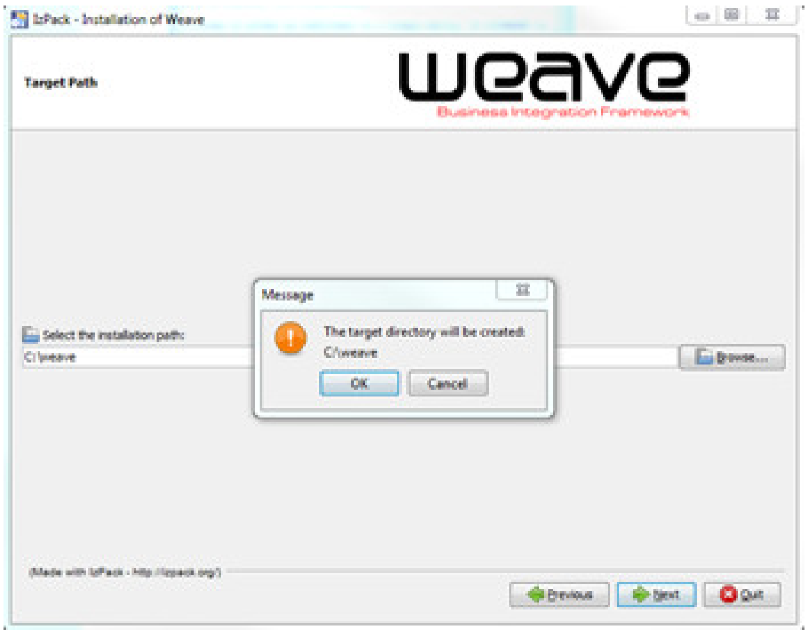

IzPack - Installation of Weave

Target Path

Select the installation path:

C:\weave

7. Click *OK* in the resulting window to create the Weave directory, then click *Next.*


IzPack - Installation of Weave

Target Path

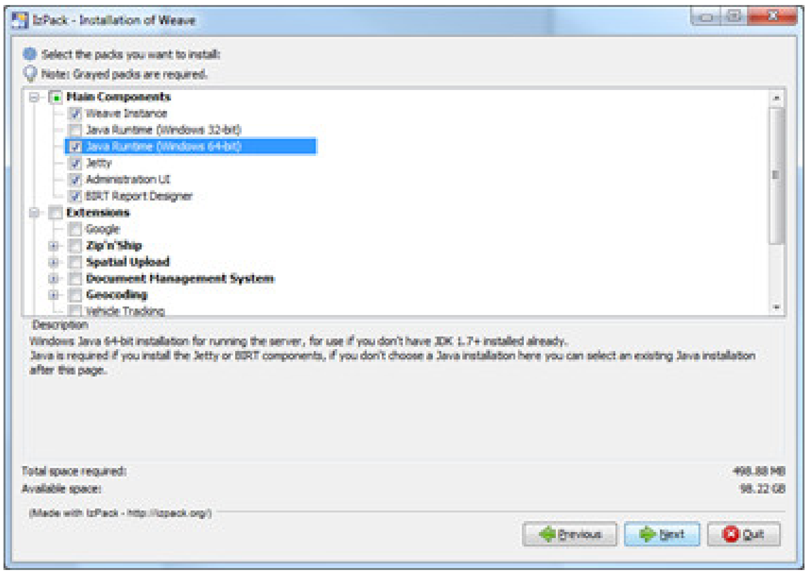
Select the installation path:

C:\weave

Message

The target directory will be created:
C:\weave

8. **Installation Packages:** You can select the components to install. Choose the *Extensions* that you wish to have access to, and click *Next.*



Select the packs you want to install:
Note: Grayed packs are required.

- Main Components
  - ✓ Weave Instance
  - Java Runtime (Windows 32-bit)
  - ✓ Java Runtime (Windows 64-bit)
  - ✓ Jetty
  - ✓ Administration UI
  - ✓ BIRT Report Designer
- Extensions
  - Google
  - **Zip'n'Ship**
  - **Spatial Upload**
  - **Document Management System**
  - **Geocoding**
  - Vehicle Tracking

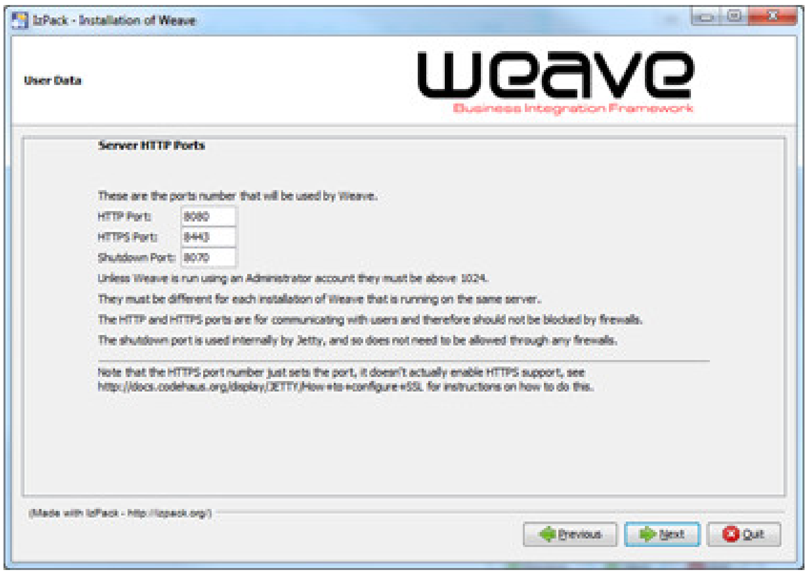
Description
Windows Java 64-bit installation for running the server, for use if you don't have JDK 1.7+ installed already.
Java is required if you install the Jetty or BIRT components, if you don't choose a Java installation here you can select an existing Java installation after this page.

Total space required:	498.88 MB
Available space:	98.22 GB

(Made with IzPack - http://izpack.org/)

From Weave 2.6.4, some items that were previously optional to install have been included as part of the *Main Components* item so don't need to be individually selected. This includes the *Google*, *Zip'n'Ship* and *Spatial Upload* items. Also two new extensions have been added - *System Metrics* and *Online Help*.

9. **User Data:** Specify the port numbers for Weave to connect with. Please make sure that these ports are not already in use by other applications. Click *Next.*


IzPack - Installation of Weave
weave
Business Integration Framework
Previous
Next
Quit

User Data

Server HTTP Ports

These are the ports number that will be used by Weave.

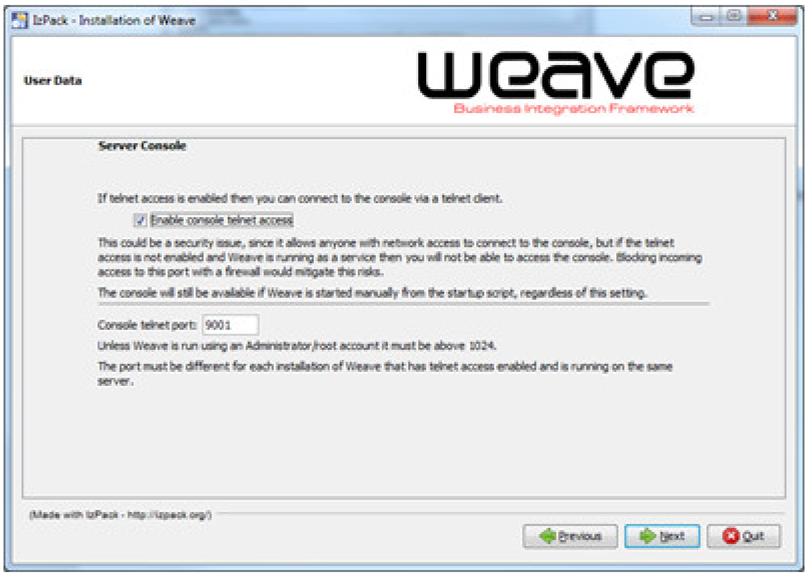
HTTP Port:	8080
HTTPS Port:	8443
Shutdown Port:	8070

Unless Weave is run using an Administrator account they must be above 1024.
They must be different for each installation of Weave that is running on the same server.
The HTTP and HTTPS ports are for communicating with users and therefore should not be blocked by firewalls.
The shutdown port is used internally by Jetty, and so does not need to be allowed through any firewalls.

Note that the HTTPS port number just sets the port, it doesn't actually enable HTTPS support, see http://docs.codehaus.org/display/JETTY/How+to+configure+SSL for instructions on how to do this.

(Made with IzPack - http://izpack.org/)

10. If needed, you can configure *telnet* access to Weave's OSGi console, and click *Next.*


IzPack - Installation of Weave
User Data
weave
Business Integration Framework
(Made with IzPack - http://izpack.org/)
Previous
Next
Quit

Server Console

If telnet access is enabled then you can connect to the console via a telnet client.

[✓] Enable console telnet access

This could be a security issue, since it allows anyone with network access to connect to the console, but if the telnet access is not enabled and Weave is running as a service then you will not be able to access the console. Blocking incoming access to this port with a firewall would mitigate this risks.

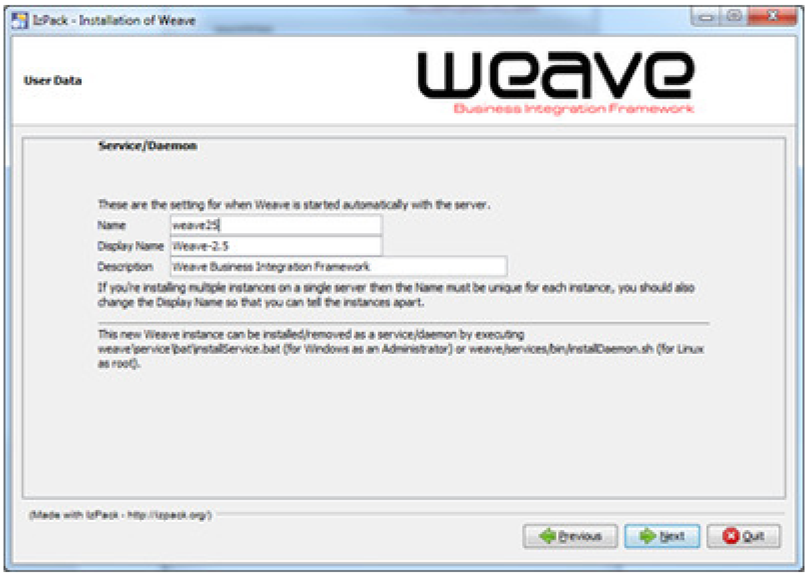
The console will still be available if Weave is started manually from the startup script, regardless of this setting.

Console telnet port: 9001

Unless Weave is run using an Administrator/root account it must be above 1024.

The port must be different for each installation of Weave that has telnet access enabled and is running on the same server.

11. The following screen shows the service settings for Weave. Click *Next* to proceed.


IzPack - Installation of Weave
User Data
weave
Business Integration Framework
(Made with IzPack - http://izpack.org/)
Previous
Next
Quit

Service/Daemon

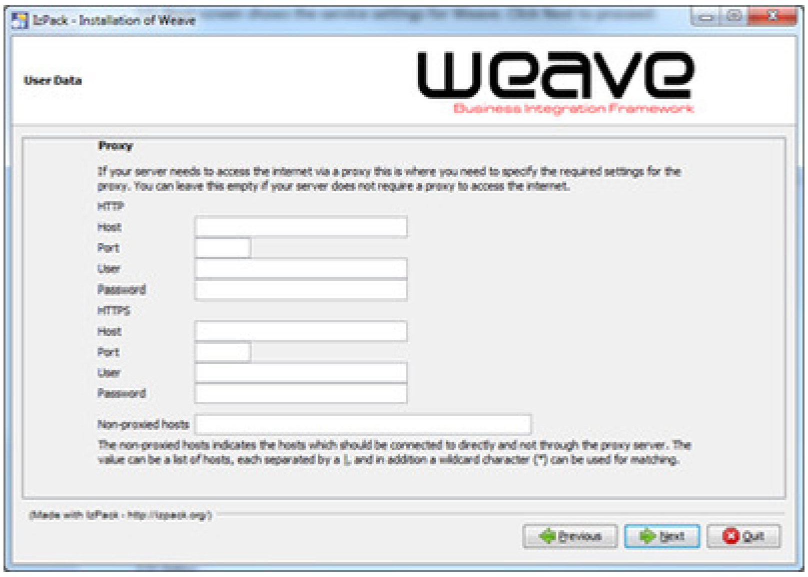
These are the setting for when Weave is started automatically with the server.

Name	weave25
Display Name	Weave-2.5
Description	Weave Business Integration Framework

If you're installing multiple instances on a single server then the Name must be unique for each instance, you should also change the Display Name so that you can tell the instances apart.

This new Weave instance can be installed/removed as a service/daemon by executing weave\service\bat\installService.bat (for Windows as an Administrator) or weave/services/bin/installDaemon.sh (for Linux as root).

12. In the following screen, configure the proxy settings if applicable, and click *Next.*


IzPack - Installation of Weave
User Data
weave
Business Integration Framework
(Made with IzPack - http://izpack.org/)
Previous
Next
Quit

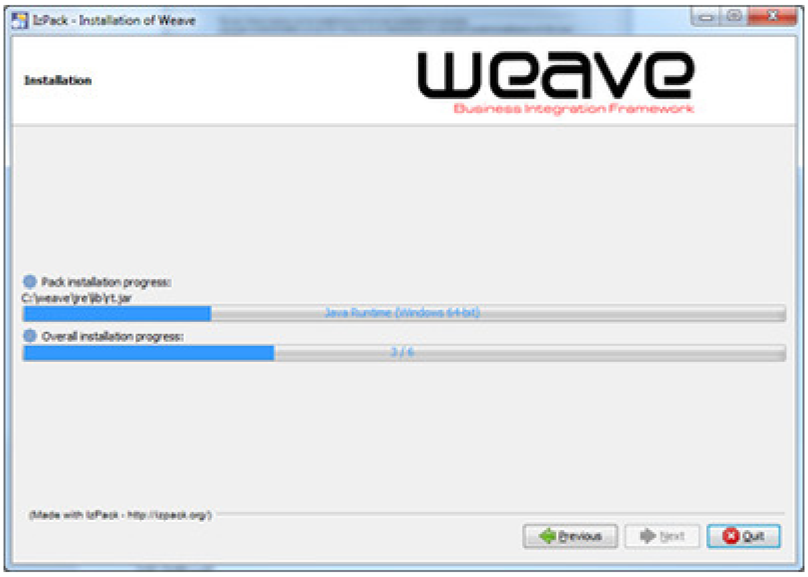
Proxy

If your server needs to access the internet via a proxy this is where you need to specify the required settings for the proxy. You can leave this empty if your server does not require a proxy to access the internet.

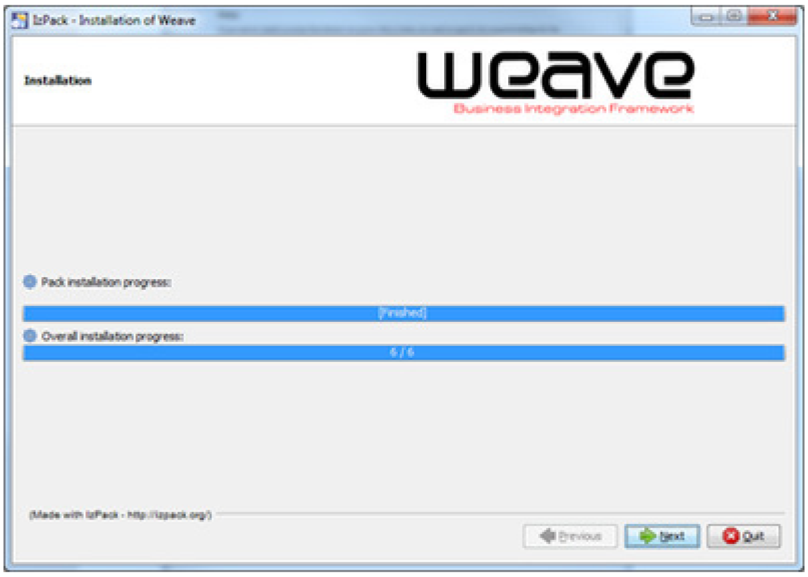
HTTP	
Host	
Port	
User	
Password	
HTTPS	
Host	
Port	
User	
Password	
Non-proxied hosts	

The non-proxied hosts indicates the hosts which should be connected to directly and not through the proxy server. The value can be a list of hosts, each separated by a |, and in addition a wildcard character (*) can be used for matching.

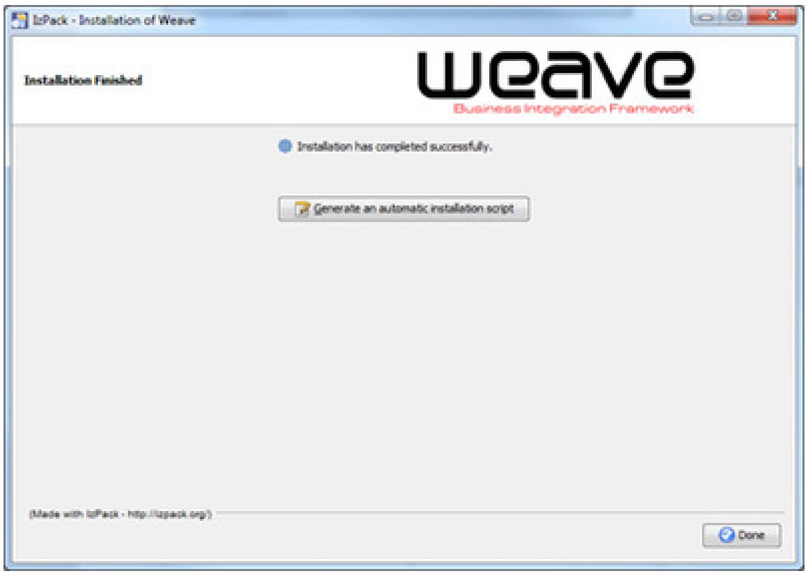
13. The Weave installation will now begin.


IzPack - Installation of Weave
Installation
weave
Business Integration Framework
Pack installation progress:
C:\weave\jre\lib\rt.jar
Java Runtime (Windows 64-bit)
Overall installation progress:
3 / 6
(Made with IzPack - http://izpack.org/)
Previous
Next
Quit

14. When the installation is complete, click on *Next* to proceed.


IzPack - Installation of Weave
Installation
weave
Business Integration Framework
Pack installation progress:
[Finished]
Overall installation progress:
6 / 6
(Made with IzPack - http://izpack.org/)
Previous
Next
Quit

15. Once the installation is finished you can chose to generate an automatic installation script with all your settings for future use by clicking the button. Else click *Done* to exit the wizard.


IzPack - Installation of Weave
Installation Finished
weave
Business Integration Framework
Generate an automatic installation script
(Made with IzPack - http://izpack.org/)
Done

Installation has completed successfully.

16. To complete the installation process, you have to carry out a few minor steps as described below.

17. Copy your existing configuration XML files to the new installation (e.g. `c:\weave\platform\workspace\config`).

18. Copy other configuration files that need to be carried over from your old installation to the new one (e.g. `C:\weave\platform\workspace\reports, C:\weave\platform\workspace\icons, C:\weave\platform\workspace\logging.properties, C:\weave\platform\workspace\config.xml`, `C:\weave\platform\workspace\users.properties`, etc.).

19. If you have installed particular JDBC drivers for your existing Weave, copy them into the new installation (`c:\weave\platform\workspace\jdbc`).

20. If you have had functions written specifically for your environment, you'll need to copy the customised bundles (binary files) to the new installation (`C:\weave\platform\plugins`). Also copy the `config.ini` from your existing installation as this file lists the bundles to start when Weave starts up (or edit the new `config.ini` file to include your custom bundles).

21. Once you have copied all the appropriate files to your new installation folders:
    a. Start Weave manually by running `c:\weave-2-5-18\startup.cmd` (once you've seen that the new Weave can be started manually you can set Weave to run as a service).
        i. Starting Weave for the first time using `startup.cmd` will create files owned by the same user that runs `startup.cmd`, if you later install Weave as a service you may need to manually change the permissions on the Weave installation directory to provide full control to the user the service will run as.
    b. In the Weave client, click the *About* button as this will give you the software version, confirm that is corresponds with the version you've upgraded to (or check the `c:\weave-2-5-18\version.txt` file).

The same process should be followed to start the installation process when installing any additional components that come with their own installer, e.g. the editing extensions, the interoperability extensions or the report designer

For:

- Technical details, refer to: Latest Downloads
- What's New, refer to: What's New

Related articles

- How to Install a New Version of Weave
- How to Upgrade to Java 8
- How to Run a Weave Hotfix
- How to Install a Weave Patch
- How to Run a Weave Software Upgrade

How to Install a Weave Patch

If you have encountered a problem with Weave, you may be provided with a "patch" to rectify just this particular issue.

Installing a patch will keep you on the same version of Weave, it will just update the files specific to that problem.

Step-by-step guide

1. Download the patch from the link provided in your support ticket; this will generally be in the form of a `.jar` file.
2. Copy this `.jar` file into the `plugins` directory (`..\weave\platform\plugins`).
    - You can leave the old `.jar` file in the `plugins` directory as Weave will look for the latest file when running.
3. Stop Weave.
4. Delete everything in the `...\weave\platform\configuration` directory except for the file `config.ini` file and `felix.fileinstall` directory (if it exists).
5. Start Weave.
6. Test, and confirm that this fixes your issue.
7. Notify Weave Support Team of the results of your testing.

If this patch is relevant to other Weave customers (i.e. not site specific) it will be rolled out as part of a regular update or hotfix.

Related articles

- How to Install a New Version of Weave

- How to Upgrade to Java 8
- How to Run a Weave Hotfix
- How to Install a Weave Patch
- How to Run a Weave Software Upgrade

How to Log a Support Ticket for Best Results

When you have a problem with Weave you can create a support ticket via the on-line support system available at support.cohga.com or you can also lodge support requests by email to support@cohga.com. Working out the cause of a problem can sometimes be difficult so the way to get the best result for all (problem report and problem solver) is to provide the most relevant information. This reduces the time that may be spent asking for required information and will result in a faster resolution of the issue. Clearly defining the problem, supplying supporting information, screenshots and configuration files is very helpful to the support team.

Step-by-step guide

1. If you are having a problem figuring out how to configure a particular component of Weave, ensure that you include the configuration item you are having a problem with. Include any relevant files containing configuration items in your support request, or if you're including a Support Dump, provide a reference to the configuration items that you're having problems with so that we can locate it within the configuration files.

2. If you are experiencing an error in the Weave client, it would help to include:
   a. Either a screen capture of the error message or the words of the error message (exactly as it appears) if a simple text description doesn't clearly explain what's happening.
   b. Any output generated in the browser developer console, and ensure that the client is started with `;debug` appended to the URL, e.g. `/weave/main.html;debug`
      i. The developer console can generally be accessed by pressing `F12`, do this then reload the browser and switch to the "Console" tab and copy the generated output. More details on using the developer console can be found here.
      ii. **Note:** A Support Dump *does not* include the information from the browser console, so even if you include a Support Dump please perform steps a and b above to include the browser-specific information.
   c. If you're not going to include a Support Dump, include as much of the `weave.log` file as possible (up to 8MB in size, preferably compressed if over 2MB).
   d. And provide a rough time when you performed the operation that showed the problem, so it can be cross-referenced in the `weave.log` file.

3. If you are having a problem with a BIRT report ensure that you include:
   a. A copy of the original report design as well as the final report showing the problem.
   b. Include as much of the `weave.log` file as possible (up to 8MB in size, preferably compressed if over 2MB) from just after generating the report.
   c. **Note:** A Support Dump *does* include any report design files and the `weave.log` file, so you just need to indicate the name of the report that is having the issue if you provide a Support Dump.

4. Reread your support ticket or email before sending it and ensure that it clearly explains what you actually mean, remember we're not seeing everything that you are seeing. If necessary, revise the text of your request, perhaps supplementing it with references to the relevant configuration items (if you're including a Support Dump) or the actual configuration items (if you're not including a Support Dump).

5. If you are finding that something is not working as you expect, ensure that the part of the `weave.log` file that you are providing is related to the issue. Include as much of the `weave.log` file as possible (up to 8MB in size, preferably compressed if over 2MB).

6. **If you are running Weave version 2.4.15 or later you can execute a dump command at the OSGi prompt (or use the *Support Tool* from the Weave Administration Tool) that will produce a .zip file containing a collection of relevant support information that would greatly assist the Weave Support Team in resolving your issue.**

More Weave Support information at the Weave Support page.

Related articles

- How to Get a Weave Support Dump using a URI
- How to View the Developer Console
- How to Log a Support Ticket for Best Results

How to Maintain Bookmarks When a Map Engine Changes

When you change a Map Engine any Bookmarks which are based on the Map Engine may no longer work. This may happen when a Bookmark is used to save the layers on/off status in your Table of Contents. If you are planning to change your Map Engine you need to keep in mind that existing Bookmarks may no longer reference the correct layers and so will not show the layers that they should; the layers may not even exist, or their id's may have changed in the new Map Engine. At version 2.4.25 Weave now uses entry ids so that you may change layer names without breaking the linkage made with existing bookmarks. The following notes will explain this in more detail.

Step-by-step guide

1. The key to maintaining a functioning Bookmark is to keep the `entry id` for layers that are used (or might be used) in Bookmarks consistent. So if you are changing the name of a Map Engine, or even the name of the layer (as displayed in the Table of Contents) this won't matter as long as you don't change the entry id's in the `toc.xml`. This is shown in the example below.

   As an example from a toc.xml file

   ```
   <entry id='l_mapengine.grids_250m_grid' layer='250m Grid'
   label='250m Grid' checked='false'/>
   ```

   The QGis project file, Geoserver Workspace, or ArcMap Document that is setting the display layers can now be changed and the layer name changed as long as the entry id is not changed. Following the example above, the layer is now called `grid_250m`, but the id has been left as `l_mapengine.grids_250m_grid`. If the entry was changed as shown below, any Bookmarks referring to that layer would continue to work.

   Updated toc.xml file

   ```
   <entry id='l_mapengine.grids_250m_grid' layer='grid_250m'
   label='250m Grid' checked='false'/>
   ```

2. If you are using layers and groups, an `entry id` should be used for both. The standard is to prefix group ids with a "g_" and layer ids with a "l_" , followed by Map Engine name, followed by layer name. This makes it easier to read and generally gives you a unique id across all your layers in all ToC models.

   Using entry id for layers and groups

   ```
   <entry id='g_mapengine.main_contour' label='Contours' exclusive="
   true">
           <entry id='l_mapengine.main_contours50cm'
   layer='Contours 50cm' label='Contours 50cm' checked='false'
   entity='lyr_contours50cm'/>
           <entry id='l_mapengine.main_contours1m' layer='Contours
   1m' label='Contours 1m' checked='false'/>
   </entry>
   ```

3. Weave uses entry ids so that changes can be made to the Map Engines and the bookmarks that are saved will continue to work. An entry id should not be changed once it's been set unless there is no requirement that any existing bookmarks should work.

4. This method of storing Bookmarks has not always been used in Weave. Prior to Weave version 2.4.25, layer information stored in bookmarks was based on the Map Engine and layer id from the ToC model. So when a Bookmark was opened the Map Engine and Layer id were read and Weave would use this information to locate the Table of Contents entry that had to be turned off or on. This would result in problems if layers were renamed.

   The solution to this was to allow for a unique id to be set in the Table of Contents configuration for each entry and by doing so, remove the dependency on the Map Engine and Layer id. This meant that the underlying layer could be renamed, or even moved to a different Map Engine, and the saved bookmarks would still work. (This explains why the entry id concept was introduced to the Weave ToC model.)

   Unfortunately this method of saving Bookmarks relies on the entry id's being added to the ToC model before they're saved. If this was not done then Weave goes back to the old method for a interpreting bookmarks and there's currently no way to "fix" this for older bookmarks.

Also refer to the following sections of the Weave System Administrator Guides:

- Client Actions
- Table Of Contents

Related articles

- How to Create an Image Slider
- How to Maintain Bookmarks When a Map Engine Changes
- How to Migrate Existing Storage Parameters to Database Storage
- How to Store Redlines and Bookmarks in a Database

How to Make a Layer Selectable

When you want to make a Layer selectable, you need to need to set it up as as an *Active Layer*.

Step-by-step guide

Follow the steps described on the How to Add a New Active Layer page. Once the Layer is set up as an *Active Layer* then graphical and attribute searches can be run on that Layer.

Also refer to the following sections of the Weave System Administrator Guides:

- Entity
- Spatial Mapper

Related articles

- How to Add a Layer to the Table of Contents
- How to Add a New Active Layer
- How to Add a New Data Grid
- How to Make a Layer Selectable

How to Make and Use Snippets

A snippet is useful when you want to reuse a piece of client configuration in multiple client configurations. This can be useful when you have more than one client XML file but have common elements across them. Note that snippets can *only* be used for Weave client configuration items and not for Weave server configuration items.

Note that the following examples create a separate file to contain the snippets that are used but this is not a requirement, snippets are just like any other configuration item and can be added to any valid XML configuration file.

Step-by-step guide

1. Create an XML file for your snippets. This file can have any name and will depend on the naming convention you have adopted for your XML files. In the following example we will use `client_snippets.xml` (as a reminder that snippets can only be used for anything in the client configuration file). It is good practice to store this file in the same location as your other client XML files.

2. An element that you might want to create a snippet for is the coordinate extents used by Weave (since this is likely to be common across all your Weave clients). In your original client configuration file, this may have looked like:

client.xml

```
...
        <extents>
                <initial crs="EPSG:28355" minx="329090" miny="
5813180" maxx="350260" maxy="5826500" />
                <full crs="EPSG:28355" minx="329090" miny="
5813180" maxx="350260" maxy="5826500" />
```

```
                <limit crs="EPSG:28355" minx="325557" miny="
5811130" maxx="352957" maxy="5826257" />
        </extents>
...
```

This same text is now put into your snippets file in a slightly altered format. The sample snippets file below contains two snippets, the first one is the one that defines all the coordinate extents.

client_snippets.xml

```
<?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="default_extents">
                <initial crs="EPSG:28355" minx="329090" miny="
5813180" maxx="350260" maxy="5826500" />
                <full crs="EPSG:28355" minx="329090" miny="
5813180" maxx="350260" maxy="5826500" />
                <limit crs="EPSG:28355" minx="325557" miny="
5811130" maxx="352957" maxy="5826257" />
        </client:extents>

        <client:highlight id="highlight_style">
                <marker>
                        <!-- more config goes here -->
                </marker>
                <vector>
                        <!-- and more config that we're not
showing -->
                </vector>
        </client:highlight>

</config>
```

3. You now need to tell Weave to read this file, so the snippets file needs to be included in the `config.xml` file.

config.xml

```
<?include client_snippets.xml?>
```

4. The last part is telling your client file that it will have to go elsewhere to get all the parameters it needs for setting up with Weave GUI. This is done by setting a `ref` tag ("default_extents") which matches the `id` from the snippets file.

   So the client XML file previously had the following lines to set the coordinate extents:

client.xml

```
...
        <extents>
                <initial crs="EPSG:28355" minx="329090.0" miny="
5813180.0" maxx="350260.0" maxy="5826500.0" />
                <full crs="EPSG:28355" minx="329090.0" miny="
5813180.0" maxx="350260.0" maxy="5826500.0" />
                <limit crs="EPSG:28355" minx="325557.735" miny="
5811130.676" maxx="352957.634" maxy="5826257.148" />
        </extents>
...
```

But these five lines have now been replaced with just:

client.xml

```
...
        <extents ref="default_extents"/>
...
```

5. In summary, Weave reads the client XML file and gets to the `<extents>` section and was expecting some coordinate values here but is directed elsewhere to find these values (by the `ref` tag). Weave knows about the `client_snippets.xml` file because it's been included it in the `config.xml` file, so it goes searching and finds a matching `id` in the `client_snippets.xml` file and reads the parameters from that file.

   The key part is that in the snippets file you have to make sure the type matches what you would have had in the client XML file, the "id" can be anything as long as the same name in used in the client XML file, but the "extent" from "client:extents" is called the type and it is defined by Weave, and was used in the client XML file.

For further details on snippets refer to: Client Snippets

Related articles

- How to Make and Use Snippets

How to Migrate Existing Storage Parameters to Database Storage

The Weave Server can store information regarding the user's Weave usage, for example the bookmarks that a user saves. This information is stored on the server so that when the user begins a new session, Weave can access these details. There are also some system-wide details that Weave stores and retrieves. As an alternative to storage in the Weave Server, this information can be stored in a database (being any data source configured within Weave).

If you are using Weave's internal system for managing storage parameter, but want to move to database storage, then it's likely that you'll want to migrate your existing storage parameters. To migrate these parameters follow the steps below.

Step-by-step guide

1. At the OSGi prompt, use the `ustorage save` and `storage save` commands to write the user storage and server storage parameters out to text files, e.g.

   ```
   ustorage save c:/temp/user-store-dump.txt
   storage save c:/temp/server-store-dump.txt
   ```

   If a location is specified (as shown above), the contents will be saved to that file otherwise the content will be saved to an internal file location.

2. Switch over configuration so that a database is used for storage. For details about configuring Weave to use a database for user storage refer to this How-to page.

3. At the OSGi prompt, use the `ustorage load` and `storage load` commands to read the user storage and server storage parameters from the text files created in Step 1, e.g.

   ```
   ustorage load c:/temp/user-store-dump.txt
   storage load c:/temp/server-store-dump.txt
   ```

   If a location is specified (as shown above), the contents will be loaded from that file otherwise the content will be loaded from an internal file location.

4. Open Weave and check that the previously stored parameters (such as Bookmarks, save Redlines, etc.) are available.

Also refer to the following sections of the Weave System Administrator Guides:

- Client Storage

Related articles

- How to Maintain Bookmarks When a Map Engine Changes
- How to Migrate Existing Storage Parameters to Database Storage
- How to Store Redlines and Bookmarks in a Database

How to Run a Weave Hotfix

If you already have a version of Weave running and have encountered a problem, you may be asked to run a "hotfix" if it's been determined that your issue has already been fixed and was released as part of a hotfix. A hotfix contains one or more bug fixes that have been found since the last release.

Running a hotfix is similar to running a Weave update; the steps to follow are provided below.

⚠ This page is valid if you are already running a 2.5 version of Weave.

Step-by-step guide

1. Download the hotfix from the link provided in your support ticket or from the Weave download page (in the *Hotfixes* section, choose the hotfix for the version of Weave you are running).
2. Stop the existing Weave service.
3. Delete your `weave.log (..\weave\logs)` file so you can start with a clean log file for the hotfix testing.
4. Optionally make a backup of Weave plugins and features folders (in case you need to roll-back to your original Weave). The hotfix will just replace the files in the folders it needs to update.
5. Then either
   a. Run the hotfix jar file without any extra parameters and I'll work the same way as the Weave installer and you have to choose the current Weave directory, e.g.
      `java -jar c:\downloads\weave-hotfix-2.5.21.9.jar`

      Or:
   b. Run it with the existing Weave directory as a command line parameter and it'll run, update the directory and exit without any interaction.
      `java -jar c:\downloads\weave-hotfix-2.5.21.9.jar c:\weave-2-5-21-dev`

      If you don't have java installed on the server then you'll need to use Weave's own java to run the hotfix so it'll be something like:
      ```
      cd c:\weave-2-5-21-dev\jre\bin
      java -jar c:\downloads\weave-hotfix-2.5.21.9.jar c:\weave-2-5-21-dev
      ```
6. Restart your Weave service and test the fix.

It's best to apply the hotfix to your Weave development environment (DEV) and test it, before applying to your Weave production environment (PROD) but generally hotfixes are minor changes and just update one or two files in the plugins directory.

The best practice to adopt for hotfixes is to apply them when they are released for the version you are running. Hotfixes are generally small fixes and you'll generally be better off when you install them.

The fixes provided in each hotfix are documented on the What's New page.

Related articles

- How to Install a New Version of Weave
- How to Upgrade to Java 8
- How to Run a Weave Hotfix
- How to Install a Weave Patch
- How to Run a Weave Software Upgrade

How to Run a Weave Software Upgrade

If you already have a version of Weave running, you can upgrade it to the latest version released.

⚠ This page is valid if you are already running a 2.5 version of Weave and are looking to upgrade to a later minor release (a point release).

You should perform the upgrade as the same user that performed the initial installation, or who is the owner of the Weave installation directory, otherwise the updater may not be able to modify/delete existing files.

Step-by-step guide

1. Download the *Updater* file as directed by the new release email from the Weave Support System or from the Latest Downloads page .

2. If you're running Weave as a service, stop the existing Weave service. If Weave is not running as a service, shut it down (e.g. run `c:\weave\shutdown.cmd`).

3. The Updater file is a .jar file so it needs Java to run. If you don't have Java installed you can use the Java executable that was installed with your current installation of Weave (e.g. `C:\weave\jre\bin`).

4. It is recommended that you make a copy of your existing Weave installation before the upgrade. You can then remove the backup once you are satisfied that the new installation is ready.
   The updater will change the content of the `...\weave\platform` directory, but it will not touch any of your configuration files (it will also update `...\weave\version.txt`).

   The steps below give you an example of how this would be done in a Windows environment (assuming the current version is 2.5.18).

   Current Weave folder = `c:\weave`
   Copied Weave backup folder = `c:\weave-2-5-18`
   Weave downloaded updater file = `c:\downloads\weave-updater-2.5.29.jar`

   a. Copy existing Weave folder, `c:\weave`, to backup Weave folder, `c:\weave-2-5-18`
   b. Delete all the log files from the `C:\weave\logs` folder.
   c. On a Windows Command Line, run:
   ```
   cd c:\weave\jre\bin
   java -jar c:\downloads\weave-updater-2.5.29.jar c:\weave
   ```
   This will perform an automated update, alternatively you can use the Weave update wizard which will take you through the update (including path to Weave instance to upgrade) if you don't include the Weave directory in the command line. For example:
   ```
   cd c:\weave\jre\bin
   java -jar c:\downloads\weave-updater-2.5.29.jar
   ```
5. If you're running Weave as a service then restart the service, or use `startup.cmd` if you're not, and test that the updated instance starts and works correctly.
6. Once you have tested the new version of Weave you can remove the old backup.

For:

1. Technical details, refer to: Latest Downloads
2. What's New, refer to: What's New

Related articles

- How to Install a New Version of Weave
- How to Upgrade to Java 8
- How to Run a Weave Hotfix

- How to Install a Weave Patch
- How to Run a Weave Software Upgrade

How to Serve Additional Web Content

Static Directory

As of Weave 2.6.7 anything stored in the `...\weave\platform\workspace\static\` directory will be available for anyone to access via the URL `http(s)://<hostname>:<port>/weave/static/`.

This is provided as a quick and easy method to serve additional web content. You can store any content here, HTML files, images, stylesheets, either directly or in sub-directories.

Missing Files

If a user requests a file that does not exist, the underlying application server will handle it as a regular "404 not found" response. If you wish to provide your own content in this situation you can create a `404.html` or `404.htm` file and store it in the same directory and the content from that file will be returned instead. In addition to `404.html` or `404.htm` you can also provide other 404 files with different extensions, e.g. `404.png`, that will be provided to the user if they request a file of that type and it doesn't exist. The files with different extensions will take precedent over the HTML versions if one is found. Finally, the search for the not found file proceeds up the file hierarchy until a match is found, first for a match with the same file type, then for the html/htm version, this way you can provide a single file to be used by any sub-directories.

Access Control

Out of the box there is no access control on the content served beyond what may be configured with the general Weave security infrastructure, e.g. security.xml, SAML, etc, but there is support for `.htaccess` files to be included in a directory or one of its parents. The `htaccess` file format is a standard that provides configuration options for web server content but only the parts related to access control are supported in this situation. Weave will look for a `.htaccess` file in the same directory as the requested content and proceed up the file hierarchy until it finds one. If it doesn't find a `.htaccess` file there is no additional access control on the content, if a `.htaccess` file is found but it is invalid then access will be denied to all users.

The options supported in the `.htaccess` file are:

- require - Only one of the following is allowed, and if it is not provided then the user plays no part in determining if the content is accessible.
  - **`require user`** `*username* [*username2*] ... [*usernameN*]`
    - The username of the user requesting the content must be one of the listed usernames
  - **`require group`** `*acl* [*acl2*] ... [*aclN*]`
    - The user requesting the content must pass one of the listed Weave configured ACL's
  - **`require valid-user`**
    - The user requesting the content must be logged in, and cannot be an anonymous user

- allow from/deny from - These allow you to restrict access based on IP address. The listed IP addresses can be a partial IP address, e.g. "192.168.0." or "172.16." to match a range of IP addresses, only one "allow from" and one "deny from" is allowed.
  - **`allow from`** `*ip* [*ip2*] ... [*ipN*]`
  - **`allow from all`**
  - **`deny from`** `*ip* [*ip2*] ... [*ipN*]`
  - **`deny from all`**

- order - Set the order in which the allow and deny checks are performed, the allow/deny lists are processed in this order with the last one that matches the users IP address winning.
  - **`order allow,deny`**
    - The allow directives are evaluated before the deny directives
    - This is the default if not specified
    - You can use this to allow access to everyone, **`allow from all`**, then selectively deny access, **`deny from 192.168.0.12`**
  - **`order deny,allow`**
    - The deny directives are evaluated before the allow directives
    - You can use this to deny access to everyone, **`deny from all`**, then selectively add access, **`allow from 192.168.`**

- satisfy - This indicates if both user and IP checks must pass or either one must pass. This only applies if both user and IP address checks are configured.
  - **`satisfy any`**
    - If the IP address check passes *or* the user check passes then the user can access the content
    - This is the default if not specified
  - **`satisfy all`**
    - If the IP address check passes *and* the user check passes then the user can access the content

No other configuration options will be processed in the `.htaccess` file.

Jetty

This page applies to Weave 2.5 which is using Jetty 8.

For Weave 2.6, which is using Jetty 9, refer to this page Configuring Static Content Deployment

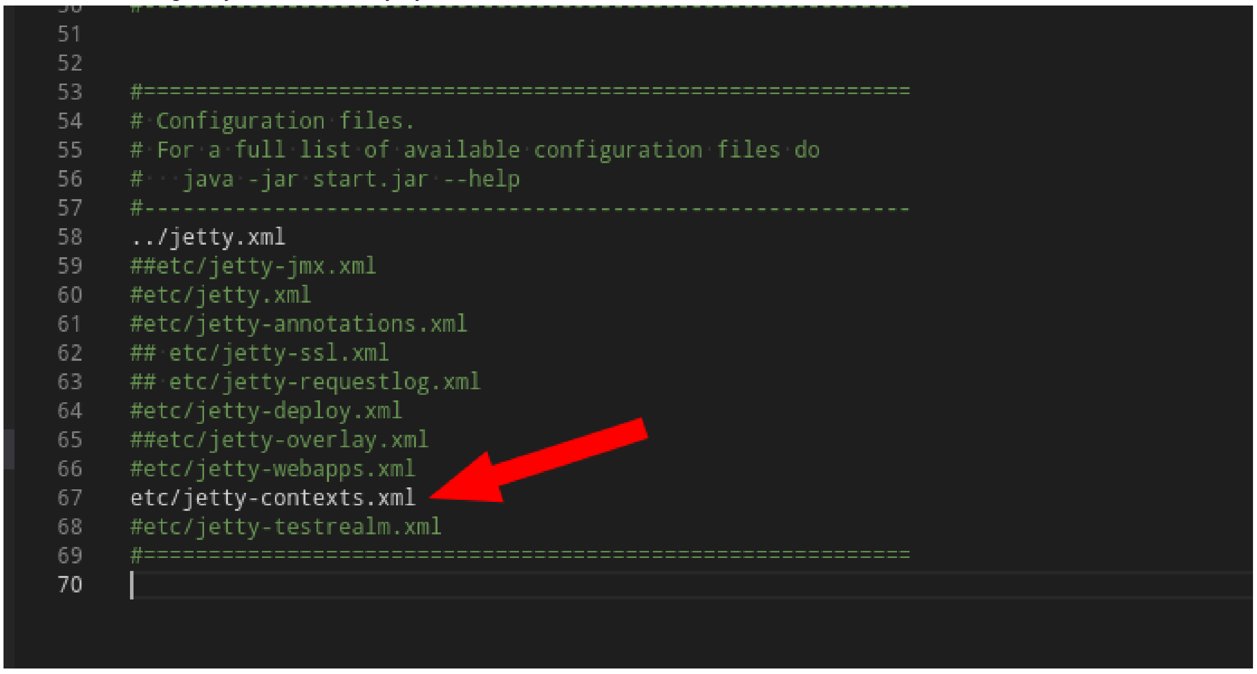
In a default Weave installation Jetty is used as the underlying Web Application server, which includes a web server. This web server can be leveraged to serve additional content besides that required by Weave (for example if you need to host images for embedding within a Weave client).

The content will be served and available to anyone who can access the server and has no access restriction (by default). Additionally, the Weave server itself has nothing to do with the content, it is being served directly by Jetty as a web server.

Instructions

How to setup a new context:

1. Edit `...\weave\jetty.ini` to enable jetty-contexts



```
#===========================================================
# Configuration files.
# For a full list of available configuration files do
#   java -jar start.jar --help
#-----------------------------------------------------------
../jetty.xml
##etc/jetty-jmx.xml
#etc/jetty.xml
#etc/jetty-annotations.xml
## etc/jetty-ssl.xml
## etc/jetty-requestlog.xml
#etc/jetty-deploy.xml
##etc/jetty-overlay.xml
#etc/jetty-webapps.xml
etc/jetty-contexts.xml
#etc/jetty-testrealm.xml
#===========================================================
```

2. Create a directory to server the content from, in this case it will be within the Weave installation directory, e.g.
   a. `mkdir /opt/weave-2.5/public`
   b. `mkdir c:\weave-2.5\public`

3. Create a new xml file in the `...\weave\jetty\contexts\` directory to serve the content
   a. For example, create one called `public.xml` to serve the directory we just created

Example context file

```
<?xml version="1.0"  encoding="ISO-8859-1"?>
<!DOCTYPE Configure PUBLIC "-//Jetty//Configure//EN"
"http://www.eclipse.org/jetty/configure.dtd">

<Configure class="org.eclipse.jetty.server.handler.
ContextHandler">
  <Set name="contextPath">/public</Set> <!-- the base URL
to serve the content from -->
  <Set name="resourceBase"><SystemProperty name="jetty.
home" default="."/>/../public/</Set> <!-- the location of
the files, in this case relative to the jetty directory -->
  <Set name="handler">
    <New class="org.eclipse.jetty.server.handler.
```

```
ResourceHandler">
      <Set name="cacheControl">max-age=3600,public</Set> <!
-- some caching options, change as appropriate -->
    </New>
  </Set>
</Configure>
```

   b. Here we set the `resourceBase` value as being relative to the jetty directory, since we know we created it under the Weave directory, but you can change it to an absolute path if the content is stored elsewhere.

4. Access the content from the same server and port that Weave is accessed from, but with the new context name, in our example `/public`, e.g. assuming a file called `image1.png` is stored at `...\weave\public\image1.png`
   a. http://weaveserver:8080/public/image1.png

Related articles

- How to Serve Additional Web Content
- How to Include an Image in a Data Grid or Map Tip

How to Set a System Property

This content applies to Weave 2.5.

Weave 2.6 system properties should be set in a file in the `...\weave\jetty_base\start.d\` directory.

There are various internal settings within Weave that globally alter the way the server functions. These settings can be altered by setting a "system property" which is read when the server starts up.

Occasionally you'll have to set a system property to alter the way the server behaves, to address a support ticket for example, and here we'll describe how that can be done.

Depending upon how you start Weave there are two places where system properties can be set. If you start Weave as a service the properties are set in `...\weave\service\conf\wrapper.conf`, and if you start Weave via `startup.cmd` or `startup.sh` the properties are set directly in that file.

No matter where the system property is defined, `wrapper.conf` or `startup.cmd/startup.sh`, what we will accomplish here is the passing of a new command line parameter to the Weave server process. The command line parameter we're trying to pass to the server process is specified in the format `-Dname=value`. The `-D` tells the Java process that we're defining a new property, `name` is the name of the property we're creating and `value` is what we want to set it to. As we do not start the Weave server process manually and rely on the service wrapper or `startup.cmd/startup.sh` to do it for us, what we're outlining here is how we tell them to pass the new command line parameter to the process when it starts.

The setting of a system property, as described here, is only a subset of the things we can do with command line parameters set in the `wrapper.conf` or `startup.cmd/startup.sh`, but this How-to page will only cover setting system properties.

Instructions

Setting a system property in `wrapper.conf`

1. Open `...\weave\service\conf\wrapper.conf` in the text editor of your choice.
2. Locate the last additional property in the file (they are usually defined at the end of the file)
   a. The "additional properties" are those listed in the form `wrapper.java.additional.## = ...`
3. Find the number of the last set additional property, and increase it by one.
4. Create a new additional property, by duplicating an existing one, increment the property number, and set the value to `-D`*`name`*`=`*`value.`*
   a. Note that it is important that you increment the property number, in the example below we use 22, which is assuming the previous largest property number was 21, but it might be different in your `wrapper.conf`.
5. Save the file and restart Weave.

Setting the weave.spatial.operation.limit system property to 500

```
wrapper.java.additional.22 = -Dweave.spatial.operation.limit=500
```

Setting a system property in `startup.cmd`

1. Open `...\weave\startup.cmd` in the text editor of your choice.
2. Locate the last `SET JAVA_OPTS` line in the file (they are set through the file and the last one would usually be defined near the end of the file).
   a. The properties are those listed in the form `SET JAVA_OPTS=%JAVA_OPTS% -D...=...`
   b. There are a number of these lines through the file, that concatenate different system properties into a single `JAVA_OPTS` variable that is then passed to the Java process at the end of the file.
3. Create a new `SET JAVA_OPTS` line after the last one with the format:
   a. `SET JAVA_OPTS=%JAVA_OPTS% -Dname=value`
4. Save the file and restart Weave.

Setting the weave.spatial.operation.limit system property to 500

```
SET JAVA_OPTS=%JAVA_OPTS% -Dweave.spatial.operation.limit=500
```

Setting a system property in `startup.sh`

1. Open `...\weave\startup.sh` in the text editor of your choice.
2. Locate the last `JAVA_OPTS="$JAVA_OPTS ..."` line in the file (they are set through the file and the last one would usually be defined near the end of the file).
   a. The properties are those listed in the form `JAVA_OPTS="$JAVA_OPTS -D...=..."`
   b. There are a number of these lines through the file, that concatenate different system properties into a single `JAVA_OPTS` variable that is then passed to the Java process at the end of the file.
3. Create a new `JAVA_OPTS` line after the last one with the format:
   a. `JAVA_OPTS="$JAVA_OPTS -Dname=value"`
4. Save the file and restart Weave.

Setting the weave.spatial.operation.limit system property to 500

```
JAVA_OPTS="$JAVA_OPTS -Dweave.spatial.operation.limit=500"
```

Example of customised wrapper.conf with weave.spatial.operation.limit set to 500
... there's more content before here

```
#********************************************************************
# genConfig: further Properties generated by genConfig
#********************************************************************
placeHolderSoGenPropsComeHere=
wrapper.java.app.jar = .\/start.jar
wrapper.app.parameter.1 = --ini=..\/jetty.ini
wrapper.java.additional.1 = -server
wrapper.java.additional.2 = -XX:PermSize=64M -XX:MaxPermSize=192M
wrapper.java.additional.3 = -XX:+HeapDumpOnOutOfMemoryError
wrapper.java.additional.4 = -Dfile.encoding=UTF-8
wrapper.java.additional.5 = -Dhttp.agent=weave/2.5 -Dhttpclient.useragent=weave/2.5
wrapper.java.additional.6 = -Djava.awt.headless=true
wrapper.java.additional.7 = -Djava.net.preferIPv4Stack=true
wrapper.java.additional.8 = -Dorg.eclipse.jetty.server.Request.maxFormContentSize=1000000
wrapper.java.additional.9 = -Dorg.geotools.referencing.forceXY=true
wrapper.java.additional.10 = -Dorg.apache.jasper.compiler.disablejsr199=true
wrapper.java.additional.11 = -Dosgi.clean=true
# set the console port for telneting into Weave.
wrapper.java.additional.12 = -Dosgi.console=9001
wrapper.java.additional.13 = -Dweave.spatial.operation.limit=500
```

Example of customised startup.cmd with weave.spatial.operation.limit set to 500

```
... there's more content before here

REM Force coordinate axis order for compatability
SET JAVA_OPTS=%JAVA_OPTS% -Dorg.geotools.referencing.forceXY=true

REM Uncomment the line below to allow a debugger to be attached
REM SET JAVA_OPTS=%JAVA_OPTS% -Xdebug -Xnoagent -Djava.compiler=NONE -Xrunjdwp:transport=dt_socket,server=y,
suspend=n,address=5000

REM Set Jetty options
SET JAVA_OPTS=%JAVA_OPTS% -DSTOP.PORT=%{jetty.shutdown.port} -DSTOP.KEY=seekrit

SET JAVA_OPTS=%JAVA_OPTS% -Dweave.spatial.operation.limit=500

cd jetty

"..\jre\bin\java.exe" %JAVA_OPTS% -jar start.jar --ini=..\jetty.ini

popd
```

Example of customised startup.sh with weave.spatial.operation.limit set to 500

```
... there's more content before here

# Force coordinate axis order for compatability
JAVA_OPTS="$JAVA_OPTS -Dorg.geotools.referencing.forceXY=true"

# Uncomment the line below to allow a debugger to be attached
#JAVA_OPTS="$JAVA_OPTS -Xdebug -Xnoagent -Djava.compiler=NONE -Xrunjdwp:transport=dt_socket,server=y,suspend=n,
address=5000"

# Set Jetty options
JAVA_OPTS="$JAVA_OPTS -DSTOP.PORT=%{jetty.shutdown.port} -DSTOP.KEY=seekrit"

JAVA_OPTS="$JAVA_OPTS -Dweave.spatial.operation.limit=500"

cd jetty

../jre/bin/java $JAVA_OPTS -jar start.jar --ini=../jetty.ini

cd ..
```

Related articles

- How to Get a Weave Support Dump using a URI
- How to Get the Most From Weave Logging
- How to Serve Additional Web Content
- How to Set a System Property
- How to fix a PKIX Path Building Failed Error

How to Set up Non-spatial Editing

Weave can be used for the editing of non-spatial data. The non-spatial data can be standalone tables, related table or entity tables.

This page will explain how to set up Weave for the editing of standalone tables. It is a simple example and many more elements can be added by referring to the Weave Administration Guide.

Although the editing functions require an additional Weave licence there are many advantages to using Weave for editing as well as an enquiry and reporting tool. User's will have the advantage of using the a system with which they are familiar, rather than introducing a new, more complex application to them.

Step-by-step guide

1. Add the Editor grid to your Weave client.
   a. Add the following to your client XML file and set the `<location>` depending on where you want the grid to appear in the Weave window:

Extract from client XML file

```
<view id="editor.panel.simplegrid">
        <label>Edit Table</label>
        <location>center</location>
        <type>table</type>
</view>
```

2. Create a new XML file for non-spatial editing. It can be called anything, e.g. `edittable.xml`.
3. Update you `config.xml` file to include the newly created XML file so that it is read by Weave.

```
<?include edittable.xml?>
```

4. In `edittable.xml` you need to put in details of the table you want to edit. You can either do this by basing it on the samples given in the Weave Administration Guide or use the Weave OSGi or Weave Admin Tool Console tool to help. There are a number of “editor” commands that you can use to check your edit setup. You can also generate an editor config for the table named using the `editor conf` command as shown below.

```
editor conf <datasource> <table>
```

5. Use the output from the “editor” command (above) in your `edittable.xml` file. Add additional tags if needed.
6. Restart Weave.
7. Open the *Console Tool* in the Weave *Admin Tool*. If you type in `editor` the Editor that you just set up should be listed (if it's all set up correctly).
8. Once you've done all that, open your Weave client and you should have an editing grid appear.
9. Double click on one of the cells to update the cell, or use the *Update* button. Or use the *Create* and *Delete* buttons.

These instructions assume you have installed the Weave Edit bundles as per the download page.

Also refer to the following section of the Weave System Administrator Guides for more information:

- Non-spatial Editing Introduction

Related articles

- How to Set up Non-spatial Editing
- How to Set up Spatial Editing using the Weave Edit Extension

How to Set up Spatial Editing using the Weave Edit Extension

Weave can be used for the editing of spatial data. This allows the editing of geometry (spatial) and attributes (non-spatial) for an entity.

This page will provide a simple example of an editing setup however many more elements can be added. A full description of this setup is in the Weave Administration Guide. The editing setup is completely configurable so in order to get a meaningful result you need to have a good understanding of the workflow required for the editing operations.

Although the editing functions require the Weave Edit Extension licence, there are many advantages to using Weave for editing as well as your enquiry and reporting tool. Users will have the advantage of using the system with which they are familiar, rather than introducing a new, more complex application to them.

Step-by-step guide

1. Add the Editor panel to your Weave client.
   a. Add the following to your client XML file and set the `<location>` depending on where you want the panel to appear in the Weave window:

```
<view id="com.cohga.client.panel.edit">
        <label>Edit</label>
        <location>west</location>
</view>
```

2. Add the Edit plugin to client config XML file.

```
<view id="com.cohga.html.client.map.mapView">
        <label>Map</label>
        <location>center</location>

        <!-- Register the edit plugin with the map view -->
        <plugin id="weave.edit"/>
        …
        …
</view>
```

3. Create a new XML file for the spatial editing. It can be called anything, e.g. `edit.xml`.
4. Update your `config.xml` file to include the newly created XML file so that it is read by Weave.

```
<?include edit.xml?>
```

5. In `edit.xml` you need to put in the details of the layer you want to edit. You can do this by basing it on the samples given in the Weave Administration Guide.
6. Restart Weave.
7. Once you have done the above steps, open your Weave client and you should have an editing panel appear. This panel provides the actions required to initiate editing and the tools required to perform the spatial edit operations.
8. To ensure your editing is able to operate:
   a. If the underlying spatial engine is a database, ensure that the permissions on the database are correct so that edits can be made to the relevant table.
   b. You may need to specify the geometry metadata for any edit table so Weave knows its projection. This is done by adding an entry in the `spatialengine.xml` file or the geometry metadata table.
   c. To ensure that newly created entities are available for use once created, ensure that the spatial mapping has `dynamic` set to `true` for the entity.

These instructions assume you have installed the Weave Edit bundles as per the download page.

Also, refer to the following section of the Weave System Administrator Guides for more information:

- Spatial Editing Introduction

Related articles

- How to Set up Non-spatial Editing
- How to Set up Spatial Editing using the Weave Edit Extension

How to Store Redlines and Bookmarks in a Database

By default Weave stores information about Redlines, Bookmarks etc. in an internal Weave data source (`system.datasource`) however this information can be stored in a database instead. The advantage of storing this information in a database is that it stores it independently of the Weave installation. Also it allows items to be deleted that have been created by a specific Weave user (for example, if someone leaves the organisation).

Step-by-step guide

1. Define a Weave data source for the database where you want to store the Redlines, Bookmarks etc.
    a. If Weave is running from the command line, add a reference to the data source in the `startup.cmd` file to set the database that should be used for storage.

    Extract from startup.cmd file

    ```
    REM Set data storage to SQL Server DB instead of Weave
    internal DB
    SET JAVA_OPTS=%JAVA_OPTS% -Dstorage.datasource=gis_db
    ```

    where "gis_db" is an `id` in the `datasource.xml` file as:

    Extract from datasource.xml file

    ```
    <jdbc:datasource id="gis_db">

            <jdbc:driver>com.microsoft.sqlserver.jdbc.
    SQLServerDriver</jdbc:driver>
            <jdbc:url>
                    <![CDATA[jdbc:sqlserver://localhost:1433;
    databaseName=weave]]>
            </jdbc:url>
        ...
        ...
    </jdbc:datasource>
    ```

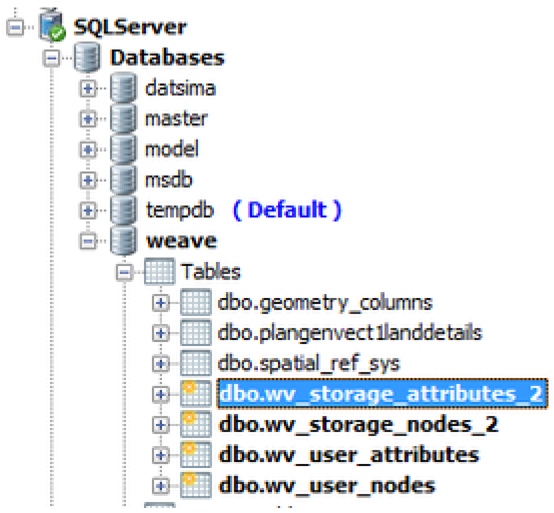
    b. If Weave is running as a Windows service, add a reference to the data source in the `service/conf/wrapper.conf` file.

    Extract from wrapper.conf file

    ```
    # Set data storage to SQL Server DB instead of Weave
    internal DB
    wrapper.java.additional.22 = -Dstorage.datasource=gis_db
    ```

2. Stop the Weave service and clean out the Eclipse folders in the `../weave/platform/configuration` folder:
    a. org.eclipse.core.runtime
    b. org.eclipse.equinox.app
    c. org.eclipse.osgi
    d. org.eclipse.update

3. Start the Weave service. Once Weave is started the `wv_*` tables will be created. These tables will be empty:
    a. wv_storage_attributes
    b. wv_user_attributes
    c. wv_user_nodes



SQLServer
- Databases
  - datsima
  - master
  - model
  - msdb
  - tempdb (Default)
  - weave
    - Tables
      - dbo.geometry_columns
      - dbo.plangenvect1landdetails
      - dbo.spatial_ref_sys
      - dbo.wv_storage_attributes_2
      - dbo.wv_storage_nodes_2
      - dbo.wv_user_attributes
      - dbo.wv_user_nodes

4. Open the Weave application and add a new Bookmark. This Bookmark will be saved into the database and you will be able to see it by looking into the `wv_user_nodes` table.

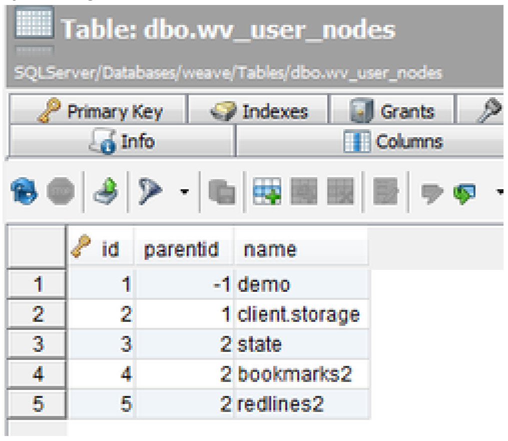


Table: dbo.wv_user_nodes

SQLServer/Databases/weave/Tables/dbo.wv_user_nodes

	id	parentid	name
1	1	-1	demo
2	2	1	client.storage
3	3	2	state
4	4	2	bookmarks2
5	5	2	redlines2

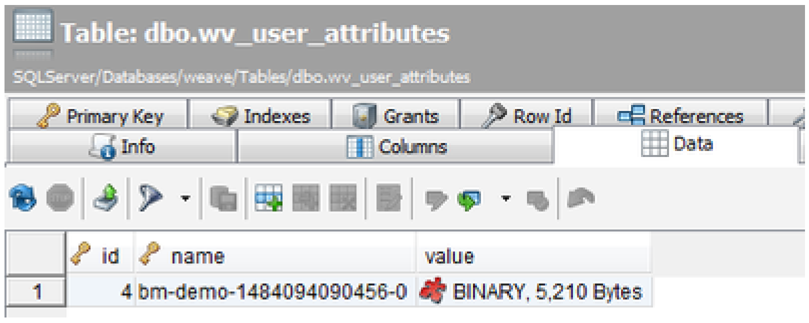


Table: dbo.wv_user_attributes

SQLServer/Databases/weave/Tables/dbo.wv_user_attributes

	id	name	value
1	4	bm-demo-1484094090456-0	BINARY, 5,210 Bytes

5. The following steps will ensure that everything is working correctly.
   a. Stop the Weave service.
   b. Repeat Step 2 to make sure that Weave will read the Bookmark from the database.
   c. Start the Weave service.
   d. Open the Weave application and the saved Bookmark will be available.

Also refer to the following sections of the Weave System Administrator Guides:

- Client Storage

Related articles

- How to Maintain Bookmarks When a Map Engine Changes
- How to Migrate Existing Storage Parameters to Database Storage
- How to Store Redlines and Bookmarks in a Database

How to Test Your Weave Application Integrations

When you are using Weave with an external application, it's sometimes useful to see what parameters are being passed to or from Weave to the external application. This can help if you're not getting the behaviour you were expecting and you need to debug the application integration.

Step-by-step guide

1. Download *DebugView* which is an application that lets you view debug output on your local computer.
2. Run *DebugView* on the PC where the Weave Client is running and not on the Weave server.
3. Start the Weave client on your computer.
4. Once *DebugView* is running, make a selection in Weave and use the application integration button (e.g. if you are linking to Pathway, select a property or parcel and press the link to Pathway button).
5. The *DebugView* window (or log file) will display debug messages sent from or to the Weave client. These messages should include instances of "Weave" and the application you are linking to (e.g. "Pathway"). In the example given below, Weave is sending this message to Pathway: `Send(DISP,0,parc,,N,9800)`

   Sample DebugView report

   ```
   00000007        19.35297966        [3348] Weave.Pathway:
   Initialize v1.1.2
   00000008        19.35331726        [3348] Weave.Pathway: Getting
   Pathway.Application
   00000009        19.35530472        [3348] Weave.Pathway: Got
   Pathway.Application
   00000010        19.35533524        [3348] Weave.Pathway: Pathway
   is available
   00000011        19.35545540        [3348] Weave.Pathway: Send
   (DISP,0,parc,,N,9800)
   00000012        19.35548782        [3348] Weave.Pathway: Getting
   Pathway.Application
   00000013        19.35642242        [3348] Weave.Pathway: Got
   Pathway.Application
   00000014        19.35664368        [3348] Weave.Pathway:
   Sent
   ```

6. The debug information that is reported will help you determine whether the correct details are being sent to or from the Weave client.

Futher information on passing parameters to other applications from Weave refer to: Application Integration

Related articles

- How to Get the Most From Weave Logging
- How to Test Your Weave Application Integrations
- How to Clear Your Browser Cache

How to Update Weave Help Content

Weave Help can be customised for your project/implementation of Weave rather than using the default Weave Help pages.

Instructions

1. Create a temporary directory that has the same name as the help content bundle, e.g. `c:\temp\com.cohga.client.weave.help.content_x.y.z\`
   where: x.y.z will be the version number of the bundle, not "x.y.z"
2. Copy the `platform\plugins\com.cohga.client.weave.help.content_x.y.z.jar` file into the directory you just created.
3. Change the file extension from *jar* to *zip.*

4. Extract the zip file into the directory. This should create `client\` and `META-INF\` directories.
5. Delete the original .zip file.
6. You can now start modifying the help pages as standard HTML pages. Each modification will require a number of steps as the menu, icon and page content may need to be created/modified.
7. Firstly we'll add a new item to the menu by modifying the `HelpContent.js` file under the `client\js\actions\` folder. To add an entry (e.g. *Example*) to the *Miscellaneous* menu, the following entry needs to be added to the JavaScript file:

```
Weave.HelpMgr.add("Miscellaneous", "Example", 'resources/weave
/help/html/misc/Example.html', 'icon-map');
```

8. The icon used for this new menu item is specified at the end of the entry added above. The icons used are the same as any icons used in Weave. Refer to the *Icons* section on this page for further details on icon use.
9. The new entry requires a corresponding HTML file and it will also use an image on the page. The HTML file (e.g. `Example.html`) is placed under the `client\resources\weave\help\html\` folder or sub-folder. The image (e.g. `example.jpg`) is placed under the `client\resources\weave\help\images\misc\` folder or sub-folder.
10. The recommended process for creating your customised help page is to copy an existing help HTML page and modify it. The result could be as shown below.

```
<div class="body-wrap">
  <h1>Example header</h1>
  <div class="description">
    Help text goes here.
  </div>
  <div class="help-image">
    <img src="resources/weave/help/images/misc/example.jpg"/>
  </div>
</div>
```

11. Once you've modified the help files as much as you require, move the entire updated `com.cohga.client.weave.content_x.y.z\` directory to the `platform\plugins\` directory and delete the original `com.cohga.client.weave.content_x.y.z.jar` file.
    a. For Weave 2.6 and later versions, this updated `com.cohga.client.weave.content_x.y.z\` directory could be moved into the `platform\custom\plugins\` directory.
12. Delete all files from the configuration directory (`c:\weave\platform\configuration`) except the `config.ini` and `feliex.fileinstall` files.
13. Restart the Weave service.
14. Open the Weave client to verify that the new help content is available.

Refer to the Weave Configuration Reference for details on how to set up the Help tool:
Client Actions Help

Related articles

- How to Update Weave Help Content

How to Upgrade to Java 8

This page only applies to Weave 2.5 as Weave 2.6 is already using Java 8

Weave 2.5.x ships with Java 7 (the current version of Java when Weave 2.5.0 was first released) but it also works with Java 8 (as of July 2019 it does not work with Java 9 or later) and there are occasions where Weave should be updated to use Java 8 which can be done manually by following this process.

Because of a licensing change by Oracle for their Java version 1.8.0.191 is the latest version of the Oracle JRE (Java Runtime Environment) that can be used without a license from Oracle (unless it's for development or personal use) so this how-to will outline how to use that version, however you can use later versions from Oracle, if you obtain the correct license from them. Alternatively, and as we've done with Weave 2.6, you could use a JRE from another vendor that does not come with any licensing restrictions (such as OpenJDK).

This page covers upgrading to Oracle Java 8 for 64-bit computers only.

This page assumes that you're upgrading from the JRE installed as part of the Weave installation, which will be in the `jre` directory where Weave is installed.

If you did not select to install the JRE when you installed Weave then this process will be different.

Instructions

1. Download the updated JRE archive
    a. Windows
    b. Linux
2. Extract the downloaded archive to the Weave installation root directory
    a. For example if Weave is installed at `d:\weave-2.5\` you would extract it directly into that directory, it will create a new `jre1.8.0_191` directory at the location where it is extracted, in our example at `d:\weave-2.5\jre1.8.0_191\`.
3. Ensure Weave is not running
4. Rename the existing `jre` directory, for example to `jre1.7.0_45`
    a. Continuing our example this would rename `d:\weave-2.5\jre\` to `d:\weave-2.5\jre1.7.0_45\`
5. Rename the new `jre1.8.0_191` directory to `jre`
    a. Continuing our example this would rename `d:\weave-2.5\jre1.8.0_191\` to `d:\weave-2.5\jre\`
6. Restart Weave and test

Alternative
If you already have a Java 8 JRE (or JDK) available on the server running Weave you can use that rather than downloading and installing the above JRE.

To do this you should edit `startup.cmd` or `startup.sh` and alter the `java` command, near the bottom of the file, to point to the `java` executable in the alternative Java 8 JRE.

If Weave is running as a service then the `wrapper.java.command` value in `service\conf\wrapper.conf` should be updated to point to the `java` executable in the alternative Java 8 JRE.

Note: that if you have spaces in the path for the alternative Java 8 JRE you may need to wrap the value in double quotes.

You might notice that when you start Weave after upgrading to Java 8 it will complain about two parameters that are no longer supported by Java 8, but were included in the startup parameters for Weave.

These are just warning and will not cause any issues but if you wish to remove them then edit `startup.cmd` (or `startup.sh`) and delete the parameters `-XX:PermSize=64M -XX:MaxPermSize=192M` form the `JAVA_OPTS` value (it should be near the top).

Also they should be removed from the `wrapper.conf` file, in the Weave `service\conf\` directory, if Weave is run as a service.

By default they're set near the end of the file as one of the `wrapper.java.additional.#` properties, and you should remove the two startup parameters, `-XX:PermSize=64M -XX:MaxPermSize=192M`.

If this results in the particular additional parameter no longer having any value (as by default these two settings are provided as a single additional parameter so removing them will remove the entire additional parameter) you should completely remove that additional parameter and re-number the following entries so that their numbering continues to be sequential. For example the current version of `wrapper.conf` has these parameters set by `wrapper.java.additional.2`, so you would completely delete that line and then change `wrapper.java.additional.3` to `wrapper.java.additional.2, wrapper.java.additional.4` to `wrapper.java.additional.3`, etc.

Related articles

- How to Install a New Version of Weave
- How to Upgrade to Java 8
- How to Run a Weave Hotfix
- How to Install a Weave Patch
- How to Run a Weave Software Upgrade

How to Use a List in an Attribute Search

When you have an existing query in the *Search* panel of Weave, you can define search parameters as lists rather than free-form text or check boxes. A list will perform an "exact" search in your database so can make searching easier.

Step-by-step guide

1. Identify which *Active Layer* and field on that layer you want to have appear as a list in your *Search* panel.
2. Add a Data Definition to your `data.xml` file. Set the XML Attributes of `name` and `label` as shown in the example below and set the `column` XML Attribute to the column in the table to be read as the values in the list. The XML Attributes of `name` and `label` only need to be configured if you want to one column to be displayed in the list but a different column to be used in the search.
3. Add a comment in your XML file to indicate that the following section is used to define Query or Search Lists. This will make it easy to identify all your query list definitions in your XML file.

   The example below creates a list of Planning Zone codes.

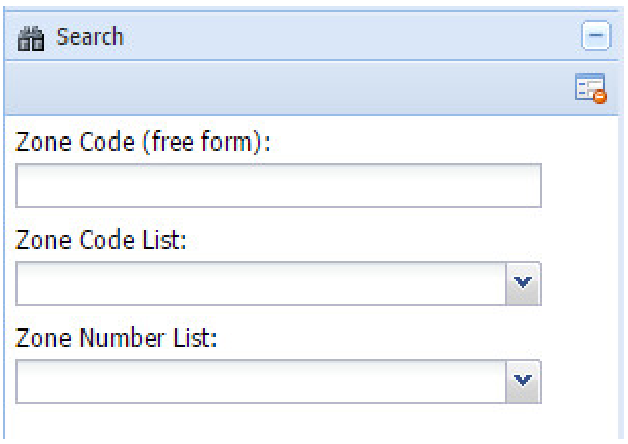
   ```
   <!--Query Lists-->
   <data:datadefinition id='zonecodes'>
           <datasourcedataconnection datasource='gis' table='mann.
   zones' prefix='DISTINCT'>
             <parameter type='string' name='label' label='Label'
   column='zone_code' />
             <parameter type='string' name='value' label='Value'
   column='zone_code' />
           </datasourcedataconnection>
   </data:datadefinition>
   ```

4. In your `search.xml` file, identify the section of code that relates to the existing *Search* you will be modifying.
5. Add a search parameter definition that will make use of the list defined in Step 3. The key in part of this definition is: `controlType='listbox'`. And ensure that the `dataset` parameter is set to the list `id` defined above.

   The example below uses the list of Planning Zone codes created above.

   ```
   <search:parameter id='zonescodelist' label='Zone Code List'
   controlType='listbox' column='mann.zones.zone_code'
   dataset='zonecodes' labelcolumn='label' valuecolumn='value'/>
   ```

6. You may need to refresh your browser to see this list in your *Search* panel.



Also refer to the following sections of the Weave System Administrator Guides:

- Attribute Search
- Spatial Search
- Stored Procedure Search

Related articles

- How to Add a New Search
- How to Use a List in an Attribute Search

How to Use a Shapefile in Weave

Cohga does not recommend using shapefiles in your Weave setup however they can be useful for use in the short term. Our view is always that corporate data should be stored, managed, accessed and edited through an RDBMS however we believe that shapefiles have a place.

Things to note when using shapefiles:

1. They are not recommended for broad scale corporate use.
2. The best performance will be achieved when the shapefiles are on the Weave server.

When using shapefiles, they can be used in Weave's Mapping Engine, Spatial Engine and Data Source. The example on this page covers all these items though they do not all need to be used - pick and chose the parts you need for your setup.

When using shapefiles, we use a spatial data connection to expose the tables via the Spatial Engine, rather than using a datasource. This will give the same result as a datasource as the Spatial Engine reads the shapefile's DBF file to access the attributes.

Step-by-step guide

1. If you want to use the shapefile in the Map View as a layer then:
   a. Add the shapefile to the map service so that it will be rendered, e.g. to the ArcMap document, QGIS project, or GeoServer workspace, then set symbology and publish map service.
   b. Add the layer to the Table of Contents (ToC), as you would for any other layer.

2. To use the shapefile as a Spatial Engine:
   a. Create a new Spatial Engine for the shapefile.

Spatial Engine

```
<spatialengine id="spatialengine.shapefile">
        <dbtype>shapefiledir</dbtype>
        <url><![CDATA[file:C:\Data\WeaveData\shapefiles]]><
/url>
        <memorymapped>true</memorymapped>
        <createspatialindex>true</createspatialindex>
</spatialengine>
```

   b. Add an entity and spatial mapper.

Entity and Spatial Mapper

```
<entity id="drainagepits">
        <label>Drainage Pits</label>
</entity>
<mapper id="mapper.shapefile">
        <spatialengine>spatialengine.shapefile<
/spatialengine>
        <mapping>
                <entity>drainagepits</entity>
                <table>am_drainagepits</table>
                <key>fid</key>
        </mapping>
</mapper>
```

c. Attach the entity to the entry in the ToC (otherwise the entity will exist but not be connected to a layer).

Table of Contents

```
<toc:model id='toc.main'>
        <entry toc='toc.selection'/>
        <mapengine>mapengine.main</mapengine>
        ...
        <entry id='l_mapengine.testing_drainage_pits'
layer='Drainage Pits (Shapefile)' label='Drainage Pits'
checked='false' entity='drainagepits'/>
        ...
</toc:model>
```

d. Note, if the projection of the data is not being picked up then you might need to change the Spatial Mapper to explicitly state the projection.

```
<mapper id="mapper.shapefile">
        <spatialengine>spatialengine.shapefile<
/spatialengine>
        <crs>EPSG:4326</crs>
        <mapping>
                <entity>drainagepits</entity>
                <table>am_drainagepits</table>
                <key>fid</key>
        </mapping>
</mapper>
```

3. And finally, to use the shapefile as a Data Source to access its attributes:
    a. Create a new Data Definition for the entity.

Data Definition

```
<datadefinition id="dd_drainagepit">
        <spatialdataconnection entity="drainagepits">
        </spatialdataconnection>
</datadefinition>

<data id="drainagepits" label="Drainage Pits" entity="
drainagepits" datadefinition="dd_drainagepit"/>
```

To begin with, don't add any `<parameter>` tags, just let all the attributes appear in the Data Grid. Once the shapefile is working as expected, add a Data Definition as for any other Entity as shown below.

Data Definition

```
<data:datadefinition id='dd_drainagepit'>
        <spatialdataconnection entity="drainagepits">
                <parameter name='unitid'   label='Unit
Id'   column='unitid'/>
                <parameter name='planno'   label='Plan
Number'   column='planno'/>
                <parameter name='sheet'    label='Sheet'
column='sheet'/>
                <parameter name='pitno'    label='Pit
Number'    column='pitno'/>
                <parameter name='type'     label='Type'
column='type'/>
                <parameter name='comment1' label='Comment
A' column='comment1'/>
                <parameter name='comment2' label='Comment
B' column='comment2'/>
                <parameter name='status'   label='Status'
column='status'/>
         </spatialdataconnection>
</data:datadefinition>
<data:data id="drainagepits" label="Drainage Pits" entity="
drainagepits" datadefinition="dd_drainagepit" />
```

Also refer to the following sections of the Weave System Administrator Guides:

- Spatial Engine Shapefile
- Spatial Mapper

Related articles

Content by label

There is no content with the specified labels

How to View the Developer Console

Sometimes an error can occur in the Weave client that is not recorded in the `weave.log` file, this is especially likely to happen if you've been asked to test updates to a new piece of client-side functionality. For example, if you've been sent an updated plugin file and you've installed the plugin but now the client won't start, or the item with the new functionality is now not working at all. In these situations, it is possible that the error is logged by the browser on the clients' computer, rather than on the server that's running Weave. This article will provide information on how you can open the developer console and view any additional log information that may be generated there, which you can then provide to Cohga to help resolve the problem.

Debug Mode

Normally, when the Weave client is started, the JavaScript code running on the browser is compressed to make it quicker to download and run the code. This results in less useful information being written to the console which is a good thing, unless you are trying to debug an issue. To try and avoid this you should start the client in “debug mode”, which will sometimes provide information that can itself be used to resolve a problem, even without producing additional console output.

To enable debug mode when starting the Weave client you just need to add `;debug` to the end of the URL you usually use to start Weave.

For example, if the usual Weave startup URL is:

```
http://example.com/weave/main.html
```

then to start the client in debug mode you’d use:

```
http://example.com/weave/main.html;debug
```

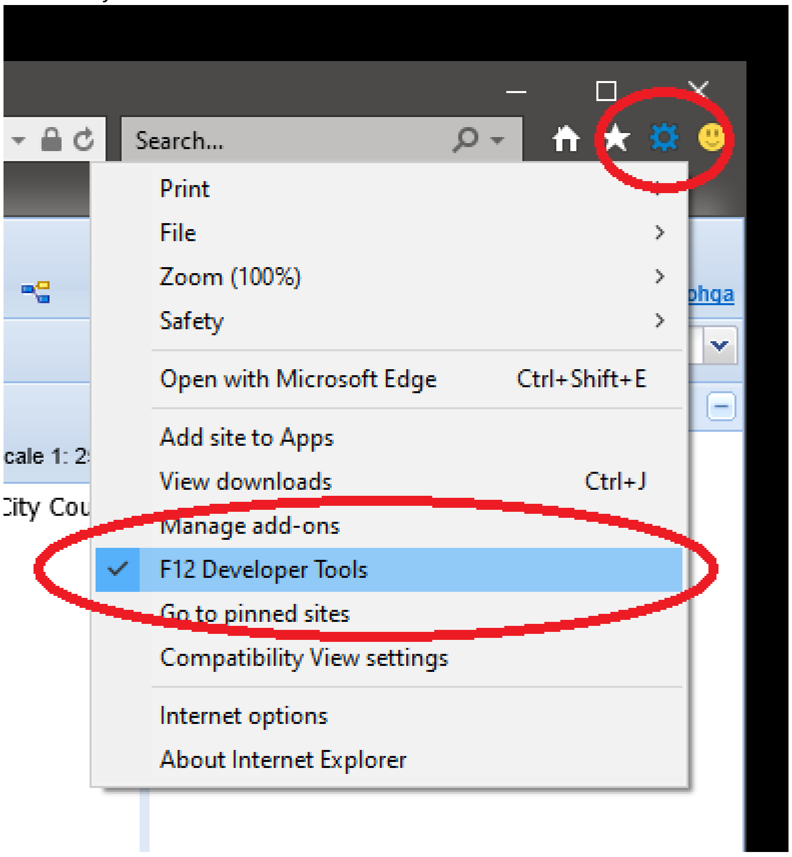
Note that if you have to login you may be directed away from the Weave client page to a login page and then redirected back to the Weave client page once you’ve logged in and the `;debug` string at the end of the URL has been removed. If this is the case just manually add it back in the URL bar, and this time you should not be redirected to a login page, since you’re already logged in, but instead the client should start in debug mode. If you can’t get debug mode to work you can still perform the rest of the things outlined in this page to at least see if there’s an error occurring.

Opening the Developer Tools

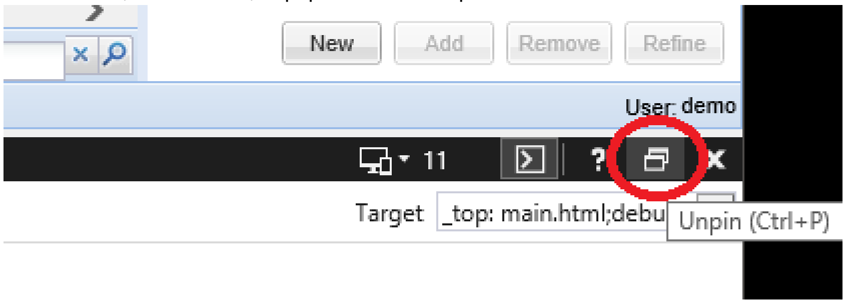
Depending upon your browser there may be a different way to open the developer tools. Note that it’s possible for the developer tools to have been disabled by your organization.

Internet Explorer

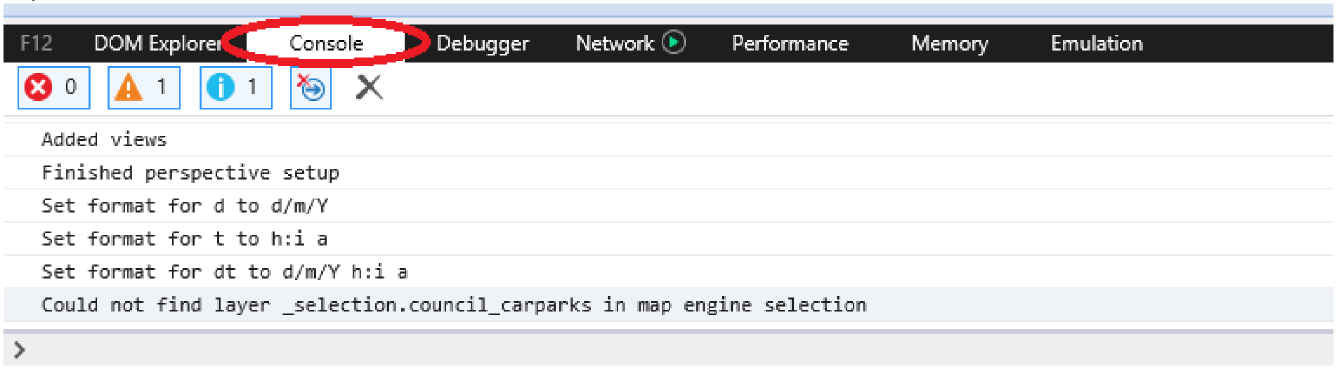
You can access the developer tools either via the *Setting* menu in the top right, under the *F12 Developer Tools* menu item, or by pressing the F12 key.



By default the developer tools will open up at the bottom of the existing browser window. You can use the “unpin” option in the developer tools title bar, or Control-P, to pop it out into a separate window that can be easier to view.



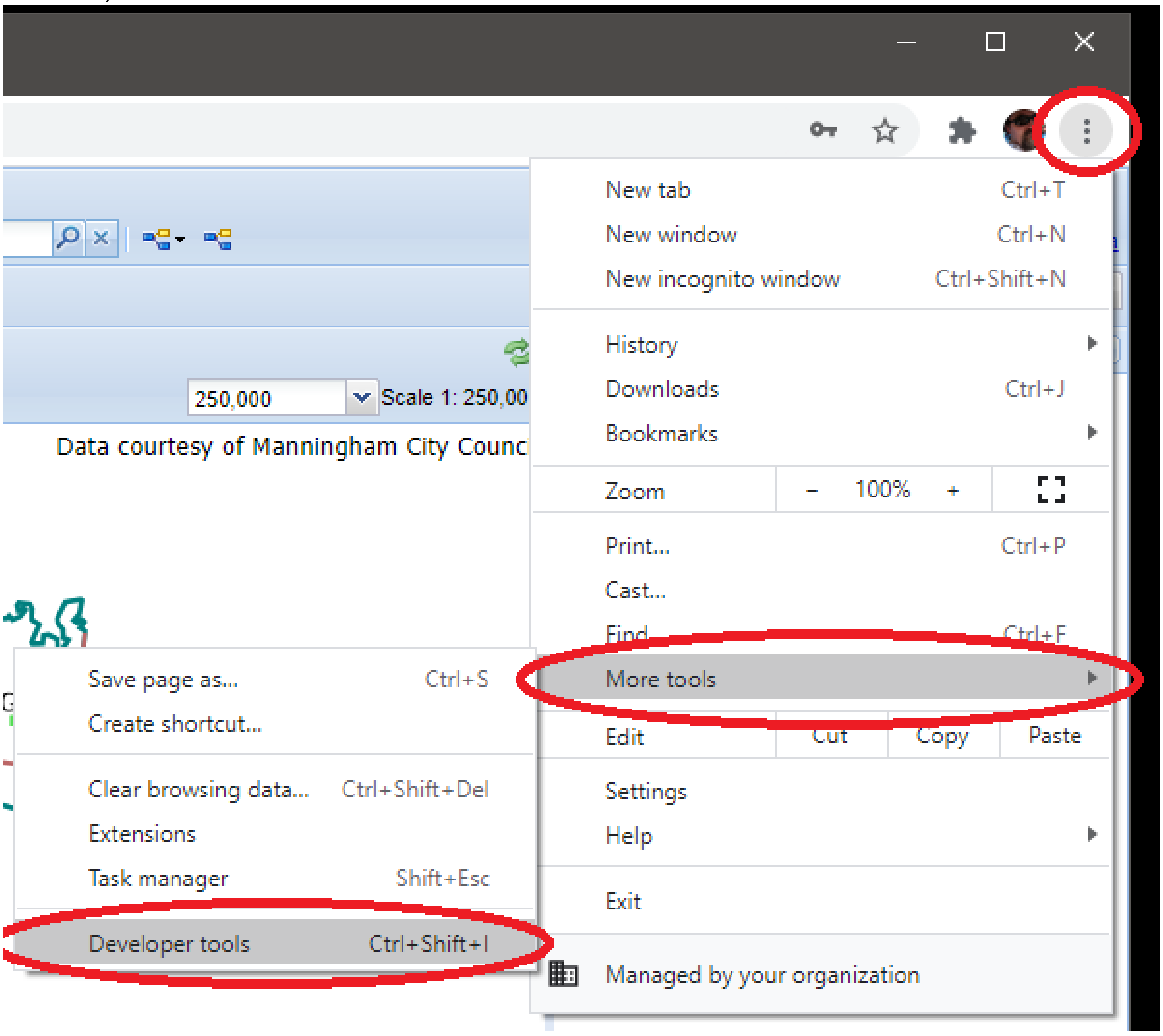
To view the console output from here you should switch to the *Console* tab. If the tab is empty reload the client and you should see some output.



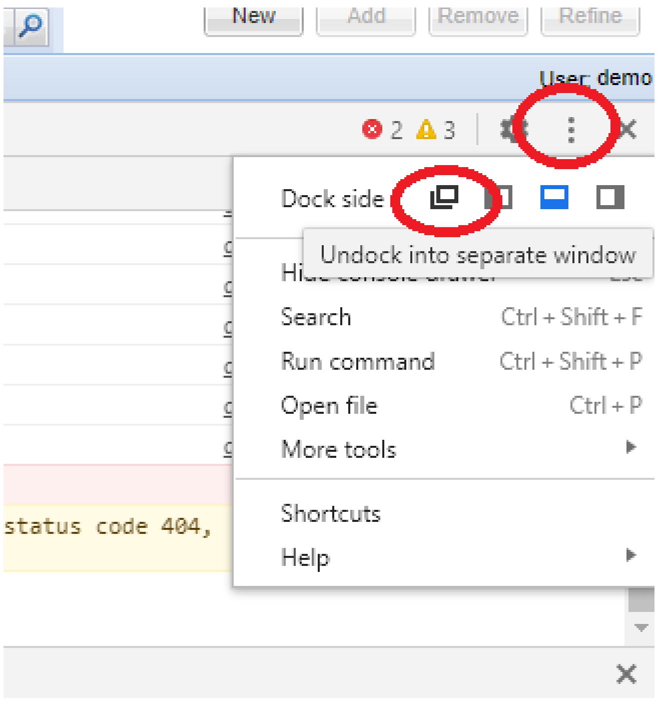
```
Added views
Finished perspective setup
Set format for d to d/m/Y
Set format for t to h:i a
Set format for dt to d/m/Y h:i a
Could not find layer _selection.council_carparks in map engine selection
```

Google Chrome

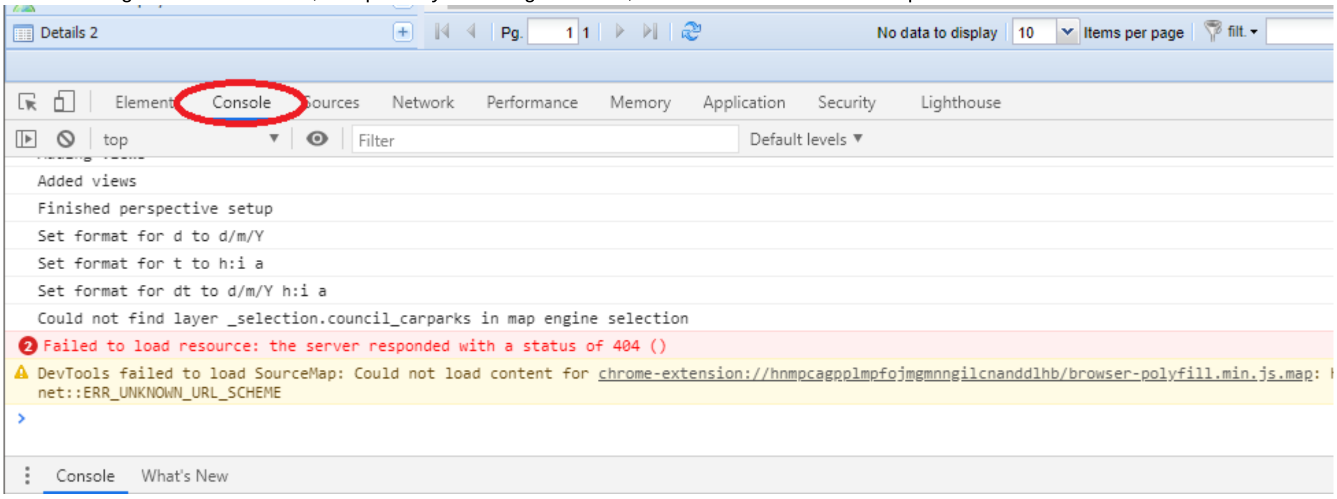
You can access the developer tools in Chrome again from the *Settings* menu in the top right, but F12 and Control-Shift-I (that's the letter, not the number) also work.



Again, Chrome will open the developer tools at the bottom of the browser window, and you can pop it out to a separate window.



And switching to the *Console* tab, and possibly reloading the client, should show the console output we're after.



```
Added views
Finished perspective setup
Set format for d to d/m/Y
Set format for t to h:i a
Set format for dt to d/m/Y h:i a
Could not find layer _selection.council_carparks in map engine selection
2 Failed to load resource: the server responded with a status of 404 ()
DevTools failed to load SourceMap: Could not load content for chrome-extension://hnmpcagpplmpfojmgmnngilcnanddlhb/browser-polyfill.min.js.map:
net::ERR_UNKNOWN_URL_SCHEME
```

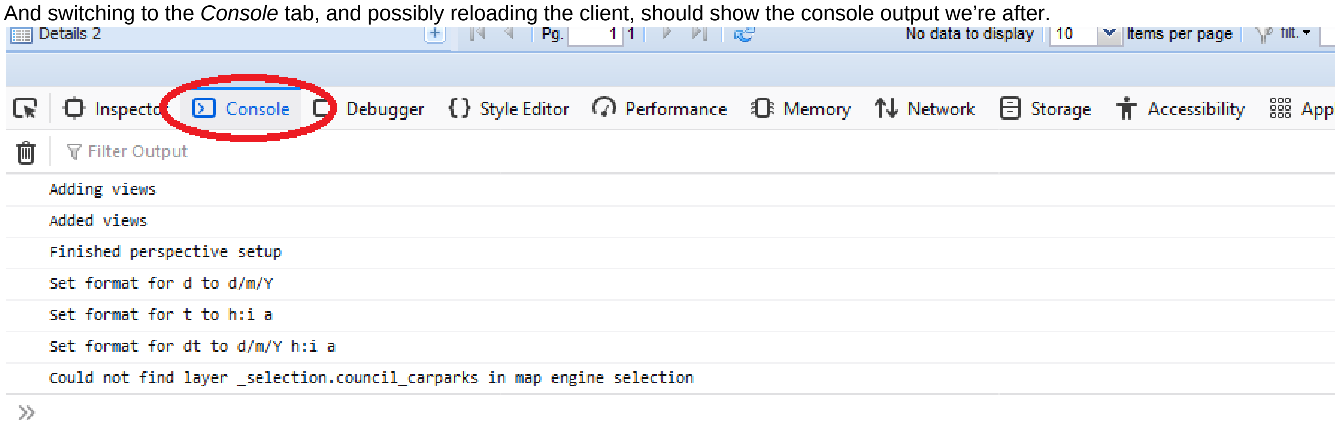
Microsoft Edge

The newest version of Edge operates in virtually the same way as Chrome (so refer to the instructions above).

Mozilla Firefox

Firefox uses Control-Shift-K to open the *Web Console*, or you can access it via the hamburger menu in the top right.

And switching to the *Console* tab, and possibly reloading the client, should show the console output we're after.



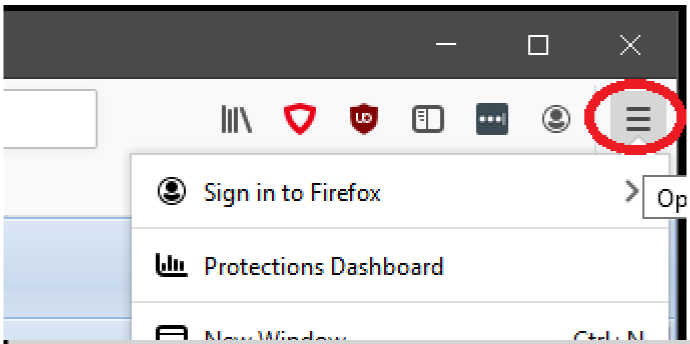
```
Adding views
Added views
Finished perspective setup
Set format for d to d/m/Y
Set format for t to h:i a
Set format for dt to d/m/Y h:i a
Could not find layer _selection.council_carparks in map engine selection
```

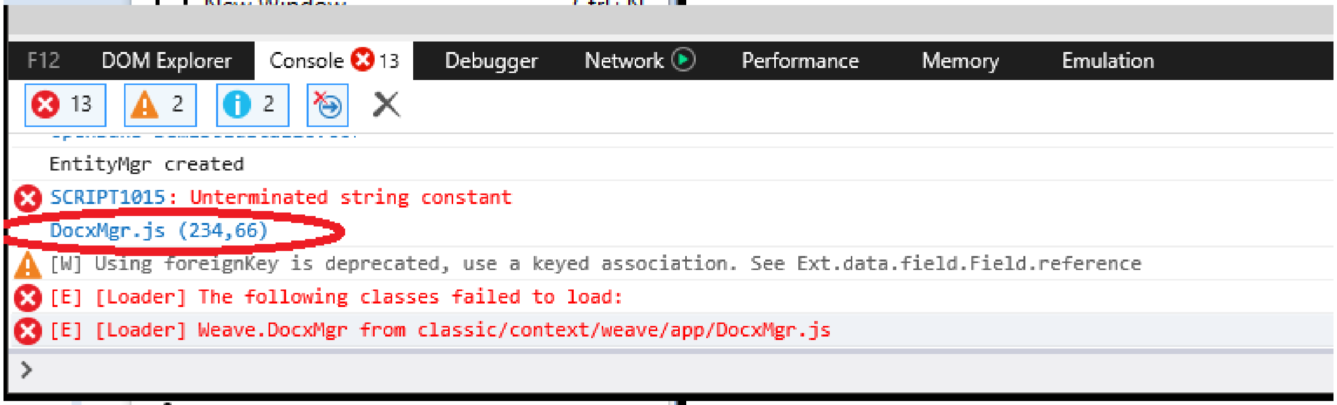
Displaying the Code

Sometimes the issue you're experiencing is related to errors in the Weave client JavaScript code. In this situation the developer console will generally output an error message that contains a link to open the offending code. Clicking on the link and sending the code, as a screenshot to Cohga can help resolve the issue.

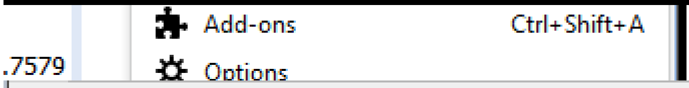
Note: If you're going to do this it is especially important that debug mode be enabled.

Internet Explorer

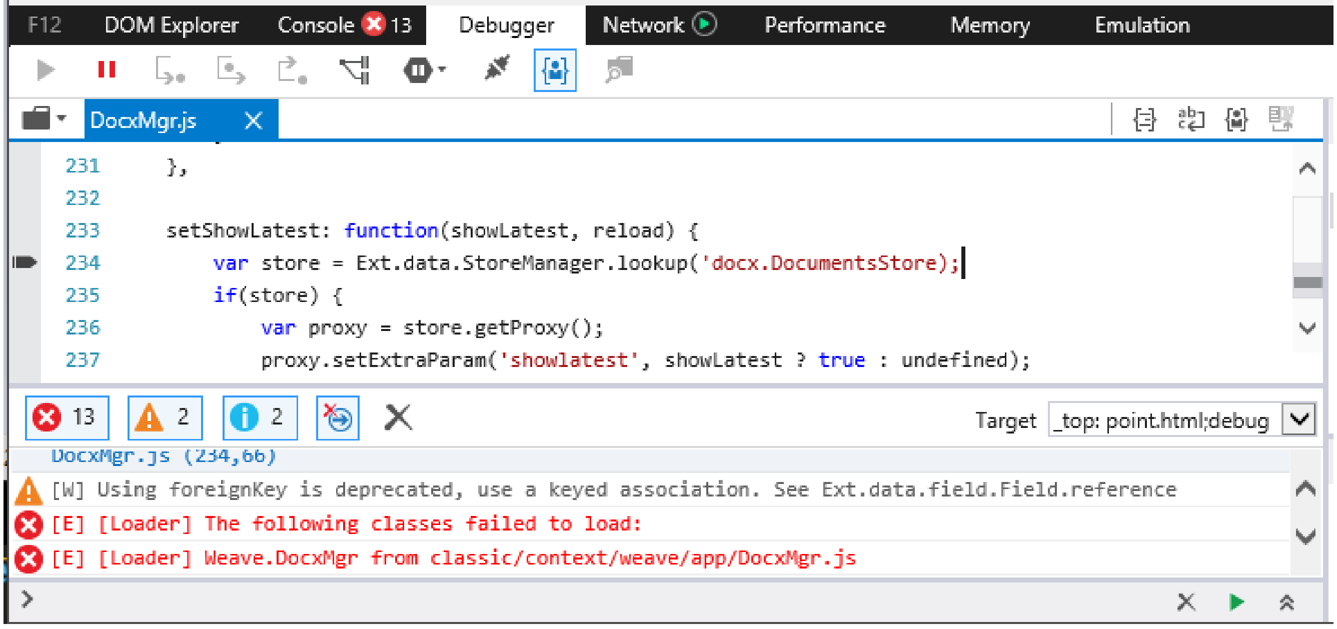



F12 DOM Explorer Console 13 Debugger Network Performance Memory Emulation
13 2 2

```
EntityMgr created
SCRIPT1015: Unterminated string constant
DocxMgr.js (234,66)
[W] Using foreignKey is deprecated, use a keyed association. See Ext.data.field.Field.reference
[E] [Loader] The following classes failed to load:
[E] [Loader] Weave.DocxMgr from classic/context/weave/app/DocxMgr.js
```



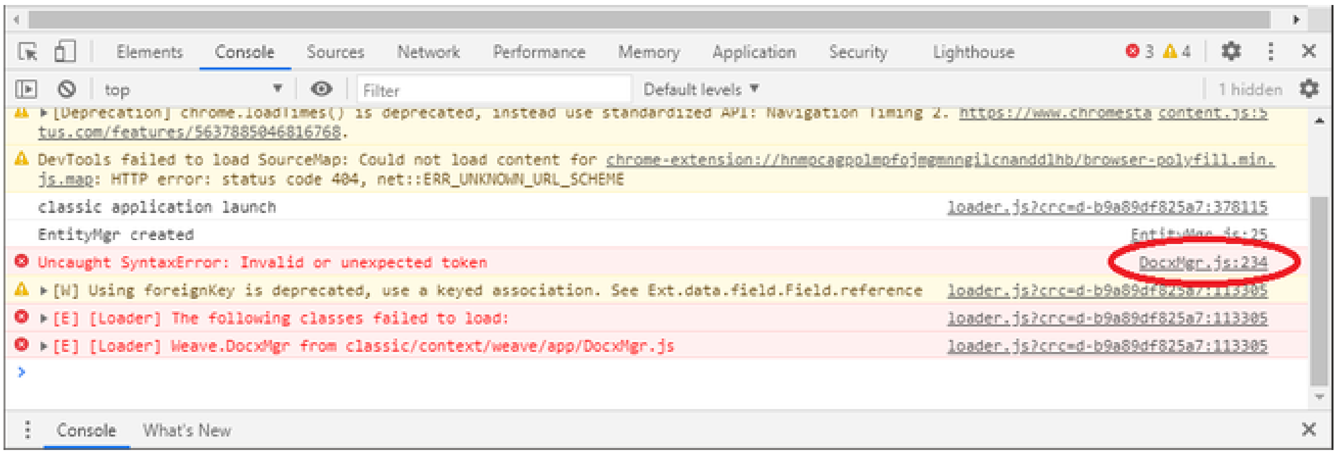
Clicking on the highlighted text above will switch to the Debugger tab and show the offending code


F12 DOM Explorer Console 13 Debugger Network Performance Memory Emulation
DocxMgr.js
13 2 2
Target _top: point.html;debug

```
231     },
232
233     setShowLatest: function(showLatest, reload) {
234         var store = Ext.data.StoreManager.lookup('docx.DocumentsStore);
235         if(store) {
236             var proxy = store.getProxy();
237             proxy.setExtraParam('showlatest', showLatest ? true : undefined);
```

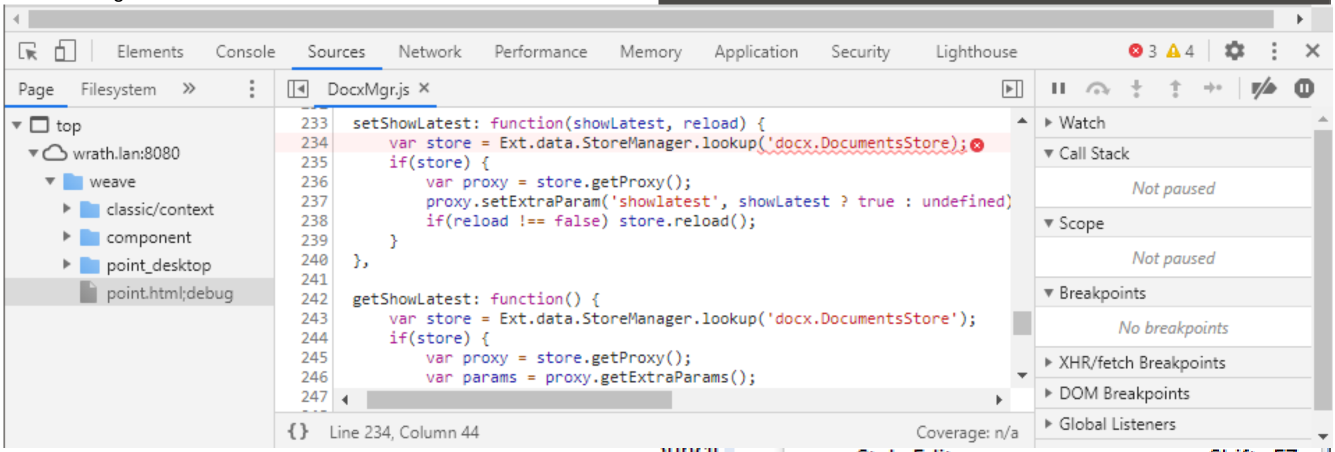
```
DocxMgr.js (234,66)
[W] Using foreignKey is deprecated, use a keyed association. See Ext.data.field.Field.reference
[E] [Loader] The following classes failed to load:
[E] [Loader] Weave.DocxMgr from classic/context/weave/app/DocxMgr.js
```

Google Chrome


Elements Console Sources Network Performance Memory Application Security Lighthouse
3 4
top
Filter
Default levels
1 hidden
Console What's New

```
▸[Deprecation] chrome.loadTimes() is deprecated, instead use standardized API: Navigation Timing 2. https://www.chromesta content.js:5
tus.com/features/5637885046816768.
DevTools failed to load SourceMap: Could not load content for chrome-extension://hnmpcagpplmpfojmgmnngilcnanddlhb/browser-polyfill.min.
js.map: HTTP error: status code 404, net::ERR_UNKNOWN_URL_SCHEME
classic application launch                                              loader.js?crc=d-b9a89df825a7:378115
EntityMgr created                                                                        EntityMgr.js:25
Uncaught SyntaxError: Invalid or unexpected token                                        DocxMgr.js:234
▸[W] Using foreignKey is deprecated, use a keyed association. See Ext.data.field.Field.reference loader.js?crc=d-b9a89df825a7:113305
▸[E] [Loader] The following classes failed to load:                     loader.js?crc=d-b9a89df825a7:113305
▸[E] [Loader] Weave.DocxMgr from classic/context/weave/app/DocxMgr.js    loader.js?crc=d-b9a89df825a7:113305
```

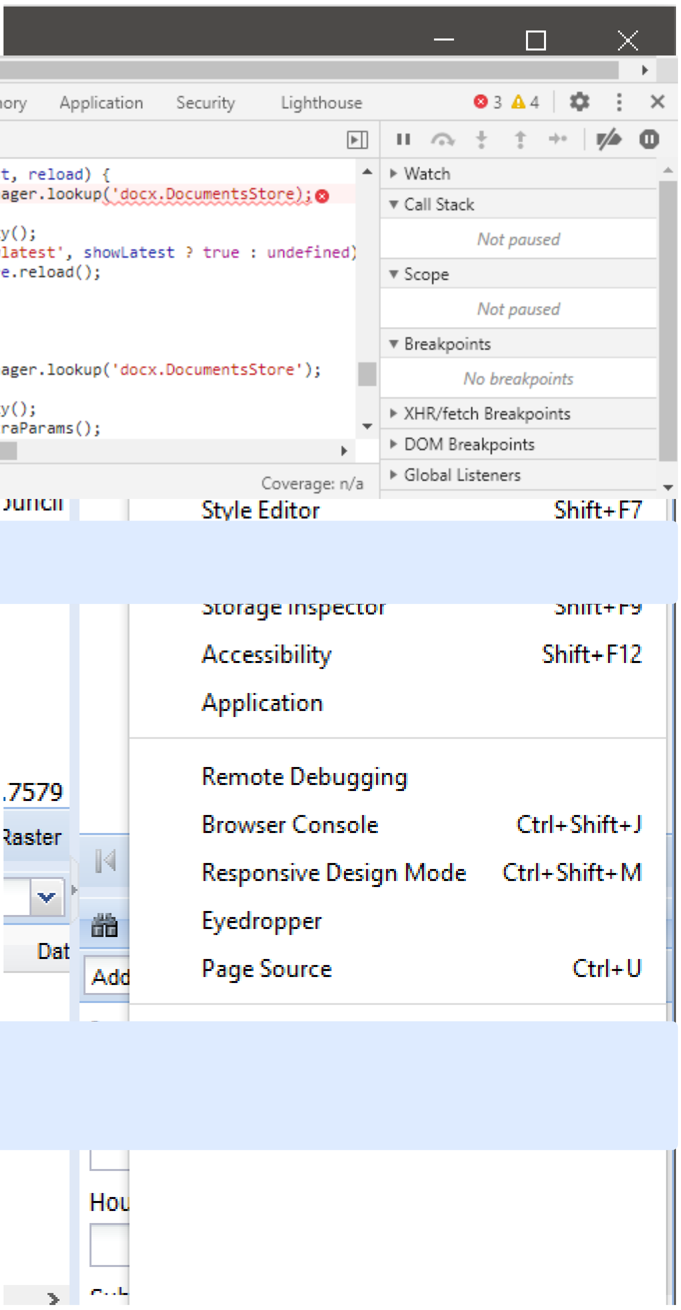
Clicking on the link above will open the Sources tab and highlight the offending code



```
setShowLatest: function(showLatest, reload) {
    var store = Ext.data.StoreManager.lookup('docx.DocumentsStore');
    if(store) {
        var proxy = store.getProxy();
        proxy.setExtraParam('showlatest', showLatest ? true : undefined)
        if(reload !== false) store.reload();
    }
},

getShowLatest: function() {
    var store = Ext.data.StoreManager.lookup('docx.DocumentsStore');
    if(store) {
        var proxy = store.getProxy();
        var params = proxy.getExtraParams();
```

Line 234, Column 44   Coverage: n/a



More Weave Support information at the Weave Support pag
e.

Related articles

- How to Log a Support Ticket for Best Results
- How to Get a Weave Support Dump using a URI
- How to View the Developer Console

How to Get a Weave Support Dump using a URI

If you are running Weave version 2.4.15 or later, depending on your access restriction configuration, you may be able to get a Weave support dump from just typing a URI into your browser.

The URI follows the following format:

```
https://<website_domain>:<port>/weave
/services/admin/support/package
```

Related articles

- How to Log a Support Ticket for Best Results