SPK
and Associates

www.spkaa.com
Ph: 888-310-4540

SPK and Associates
900 E Hamilton Ave, Ste.100
Campbell, CA 95008

A Six Step Setup Process for Incorporating CSS and jQuery into Your Electric Commander Plugin

Let's suppose you've been tasked with developing a custom dashboard for your Electric Commander installation to show off some real-time stats about your software development process. If you read my previous article about displaying Commander metrics in a GWT dashboard, this shouldn't be a big deal to you. But suppose Management wants to use this dashboard to impress potential investors. Raw numbers are great, but something with a little style and flair can make it awesome! No problem! Keep reading as I demonstrate, in three easy steps each, how you can set up your Eclipse IDE to incorporate custom CSS files and third-party GWT libraries (like GWT's jQuery port) to inject some spice into your Electric Commander GWT dashboards!

David Hubbell
SPK Software Engineer

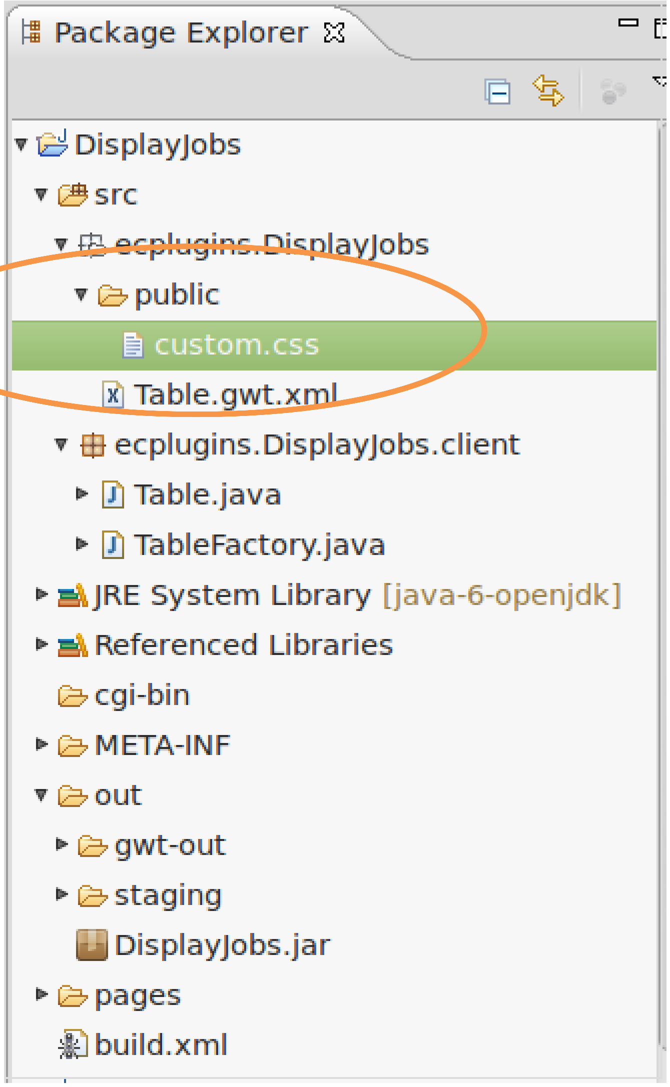
Three easy steps for configuring Eclipse to use custom CSS files in your GWT plugin:

1)

The first thing we need to do is open up our Commander Plugin project in Eclipse and, using the package explorer, create a folder named “public” in the location indicated below:

"*<projectName>*/src/ecPlugins.*<projectName>*/**public**"

Inside the “public” folder will be where you place your custom CSS files.

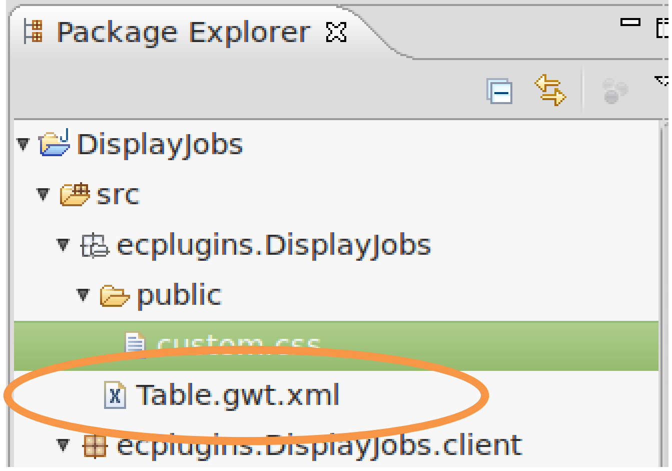


```
Package Explorer
▼ DisplayJobs
  ▼ src
    ▼ ecplugins.DisplayJobs
      ▼ public
          custom.css
        Table.gwt.xml
    ▼ ecplugins.DisplayJobs.client
      ▶ Table.java
      ▶ TableFactory.java
  ▶ JRE System Library [java-6-openjdk]
  ▶ Referenced Libraries
    cgi-bin
  ▶ META-INF
  ▼ out
    ▶ gwt-out
    ▶ staging
      DisplayJobs.jar
  ▶ pages
    build.xml
```

2)

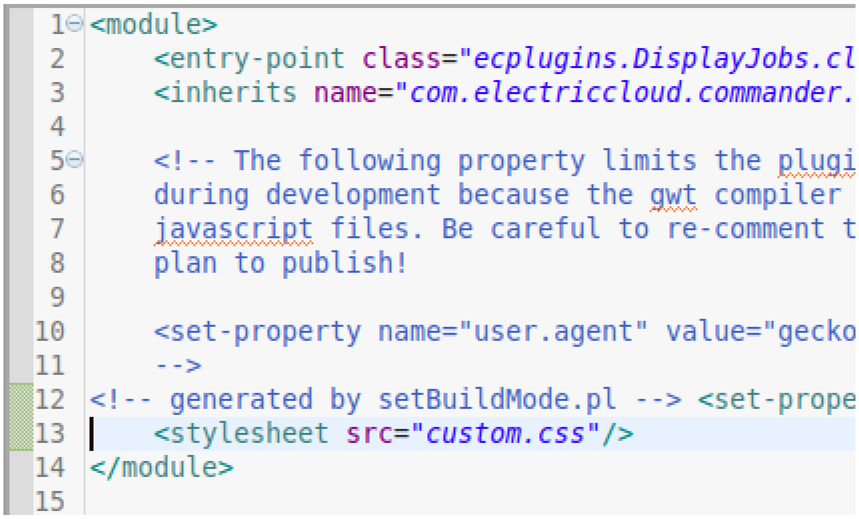
Next, edit the “`<componentName>.gwt.xml`” file located in the directory:

"`<projectName>/src/ecPlugins.<projectName>`"



Add the line:

<stylesheet src="custom.css"**/>**



```
<module>
    <entry-point class="ecplugins.DisplayJobs.cl
    <inherits name="com.electriccloud.commander.

    <!-- The following property limits the plugi
    during development because the gwt compiler
    javascript files. Be careful to re-comment t
    plan to publish!

    <set-property name="user.agent" value="gecko
    -->
<!-- generated by setBuildMode.pl --> <set-prope
    <stylesheet src="custom.css"/>
</module>
```

This tells Eclipse to include a link to your custom CSS file when it compiles your project and generates the plugin’s .jar file.

3)

Start adding style tags to your various GWT objects. For more information on this see [Google's Developer's Guide]

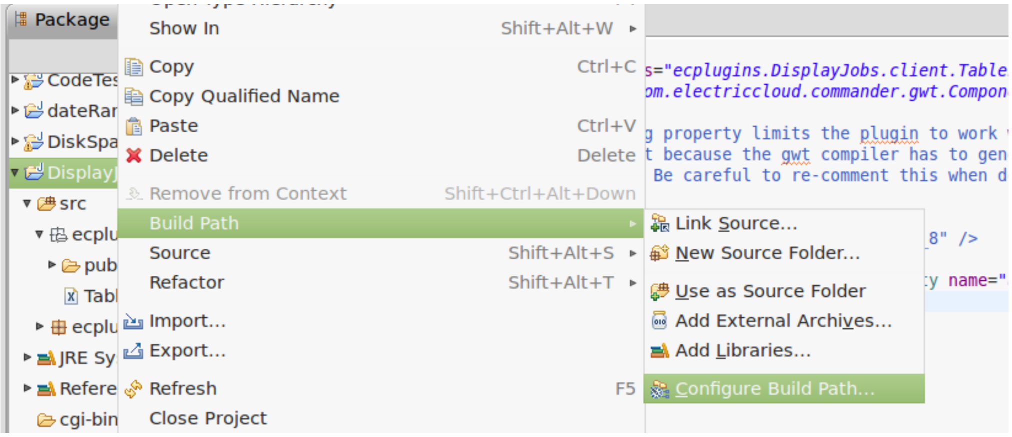
Three easy steps for configuring Eclise to use third-party GWT libraries in your GWT plugin:

In order to make use of external jar libraries, we need to tell Eclipse where to find the jar and then edit a couple of files in order to make sure GWT compiles everything correctly.

Assuming you have already downloaded a third-party GWT library, like gwtquery, and that it's located somewhere accessible in your file system, let's proceed with the following:

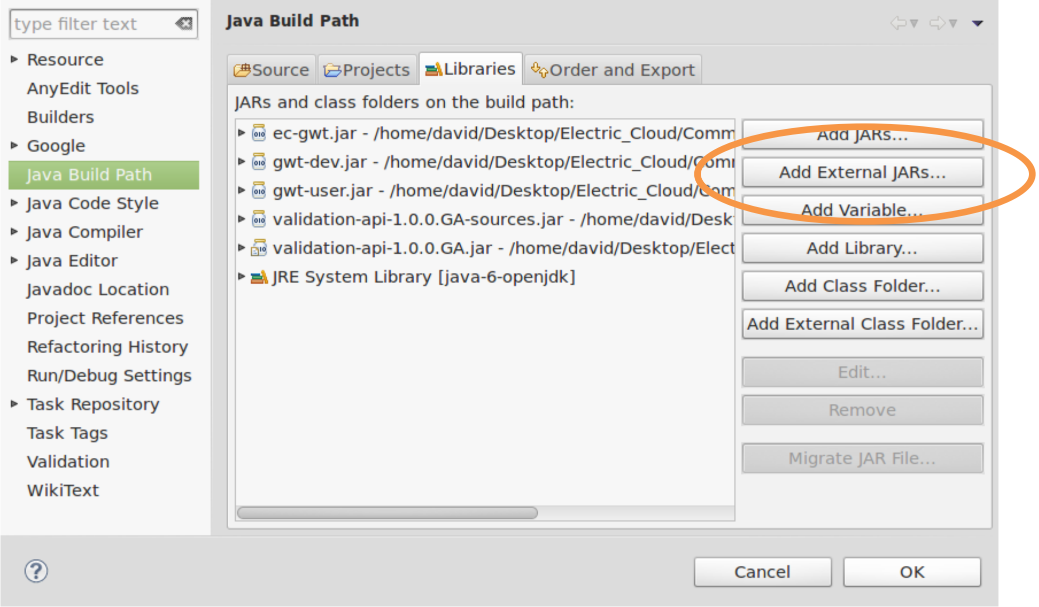
1)

Enter your project's build-path configuration screen by right-clicking your project in the Package Explorer and selecting **"Build Path → Configure Build Path..."** .



```
s="ecplugins.DisplayJobs.client.Table
om.electriccloud.commander.gwt.Compon
g property limits the plugin to work
t because the gwt compiler has to gen
 Be careful to re-comment this when d
8" />
y name="
```

Navigate to the "**Libraries**" tab and click the "**Add External JARs**" button.



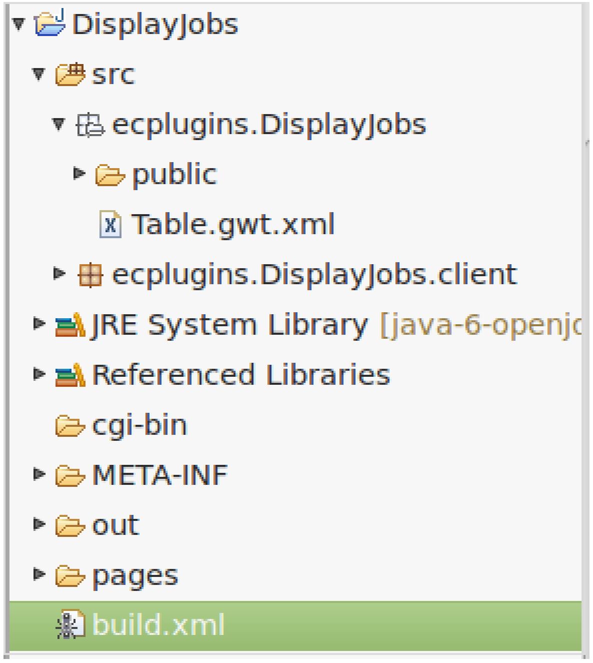
JARs and class folders on the build path:

- ec-gwt.jar - /home/david/Desktop/Electric_Cloud/Comm
- gwt-dev.jar - /home/david/Desktop/Electric_Cloud/Comm
- gwt-user.jar - /home/david/Desktop/Electric_Cloud/Com
- validation-api-1.0.0.GA-sources.jar - /home/david/Deskt
- validation-api-1.0.0.GA.jar - /home/david/Desktop/Elect
- JRE System Library [java-6-openjdk]

Locate the .jar file and press **"OK"**.

2)

Open the project's build.xml file



Find the “`<path id='extras'/>`“ tag and insert the following `pathelement` tag below it, modifying it to fit your particular needs, obviously:

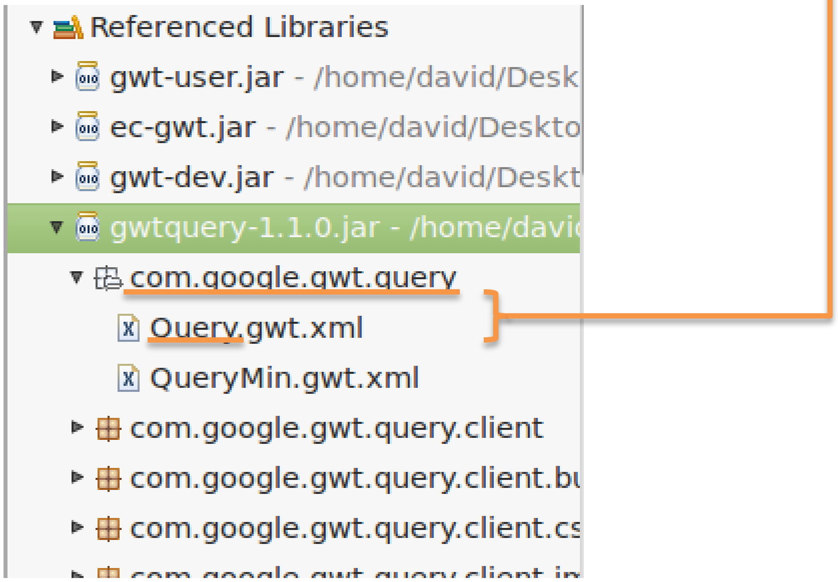
```
...
<path id='extras'/>
  <pathelement  location="/home/david/eclipseWorkspace/lib/gwtquery-1.1.0.jar"/>
...
```

3)

Open the project's “`<componentName>.gwt.xml`” file, find the section containing any existing “`<inherits ... >`” tags and add the following:

```
...
  <inherits name="com.google.gwt.query.Query"/>
  <inherits name="com.google.gwt.query.QueryMin"/>
...
```

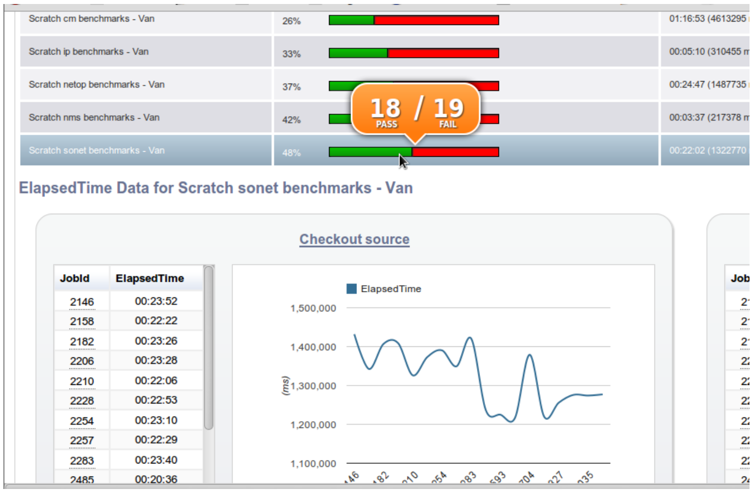
The section in quotes corresponds to the location of the jar's “<componentName>.gwt.xml” file:



- Referenced Libraries
  - gwt-user.jar - /home/david/Desk
  - ec-gwt.jar - /home/david/Deskto
  - gwt-dev.jar - /home/david/Deskt
  - gwtquery-1.1.0.jar - /home/davi
    - com.google.gwt.query
      - Query.gwt.xml
      - QueryMin.gwt.xml
    - com.google.gwt.query.client
    - com.google.gwt.query.client.bu
    - com.google.gwt.query.client.cs
    - com.google.gwt.query.client.in

Now you’re ready to make use of all the features afforded by the gwtquery library! Repeat this process for any of the other many GWT libraries out there.

For additional support using the gwtquery library, check out the online javadoc.



Scratch cm benchmarks - Van	26%	01:16:53 (4613295
Scratch ip benchmarks - Van	33%	00:05:10 (310455 m
Scratch netop benchmarks - Van	37%	00:24:47 (1487735
Scratch nms benchmarks - Van	42%	00:03:37 (217378 m
Scratch sonet benchmarks - Van	48%	00:22:02 (1322770

ElapsedTime Data for Scratch sonet benchmarks - Van

Checkout source

JobId	ElapsedTime
2146	00:23:52
2158	00:22:22
2182	00:23:26
2206	00:23:28
2210	00:22:06
2228	00:22:53
2254	00:23:10
2257	00:22:29
2283	00:23:40
2485	00:20:36