Tapestry5GWTIntegration

Tapestry 5 GWT Integration

A version of the source as a maven project for Eclipse is available here: t5gwt.jar.

Download the Source code and Project This article is on making use of both of these technologies. http://www.phy6.net/downloads/T5GWT/myapp.png

Reader Requirements

*Basic knowledge of T5 features
*Know how to create the hello world archetype of both T5 and GWT.
*very basic Maven knowledge
*Suggest you read through the first GWT tutorial, linked from the wiki page.

Technologies Used

*Tapestry 5 (downloaded for you by Maven!)
*GWT 1.5.3 (downloaded for you by Maven!)
*maven-googlewebtoolkit2-plugin (aka maven-gwt, This is the one from com.totsp.gwt)
*Maven 2.0.9 (or similar version) on classpath
*Eclipse Ganymede (The big J2EE one)
*Tomcat 6.0.18 (or similar--probably works fine with Jetty, etc but I have not tried)
*M2Eclipse plugin for Eclipse

GWT and Tapestry 5 (Thanks to Martin Kersten)

GWT and Tapestry have in common: Both are Web Techs, Both have a client and server side

GWT vs. Tapestry:
*GWT does not claim nor wants to be a full WebApp multipage framework but a WebApp singlepage framework (GWT Apps lose state when page refreshes or changes - beside using gears)
*GWT is mostly about describing JavaScript using Java whereas Tapestry is about describing HTML using Java and Templates.
*GWT is all about AJAX (yes I know there are certain areas where no AJAX is used at all for instance the view of client side information stores) and Tapestry is about HTML with some AJAX if you like to. -> Tapestry uses plain 'old' HTML Requests with some XHR where as GWT talks only XHR (untrue if you are geek and use GWT to extrem of cause 🙂).
*GWT uses a single servlet endpoint serving AJAX remote calls (XML, JSON) and Tapestry has multi-pages with IOC (SOA) and can perform all three kinds of AJAX requests (JSON, XML, HTML).
*GWT can be embedded within Tapestry pages but a Tapestry page can not be embedded within GWT pages (beside IFrames). You will need your own JavaScript to let a tapestry page talking to GWT and vise versa.

Lets get you set up

You can download the whole project using the link above. Here's the POM for you to look at:

full pom.xml

```
<project
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://maven.apache.org/POM/4.0.0">
        <modelVersion>4.0.0</modelVersion>
        <groupId>org.example</groupId>
        <artifactId>myapp</artifactId>
        <version>1.0.0-SNAPSHOT</version>
        <packaging>war</packaging><!--
        First use clean compile gwt:compile, and afterwards you can just run the compile.cmd that gets
generated.
        Remember to refresh your source directory before serving this, since the GWT compilation outputs into a
source dir!
        --><name>myapp Tapestry 5 Application</name>
        <dependencies>
            <!--  GWT deps (from central repo) -->
                <dependency>
                        <groupId>com.google.gwt</groupId>
                        <artifactId>gwt-servlet</artifactId>
                        <version>${gwtVersion}</version>
                        <scope>runtime</scope>
                </dependency>
                <dependency>
                        <groupId>com.google.gwt</groupId>
                        <artifactId>gwt-user</artifactId>
```

```
                <version>${gwtVersion}</version>
                <scope>provided</scope>
        </dependency>
        <dependency>
                <groupId>com.google.gwt</groupId>
                <artifactId>gwt-dev</artifactId>
                <version>${gwtVersion}</version>
                <classifier>${platform}-libs</classifier>
                <type>zip</type>
                <scope>provided</scope>
        </dependency>
        <dependency>
                <groupId>com.google.gwt</groupId>
                <artifactId>gwt-dev</artifactId>
                <version>${gwtVersion}</version>
                <classifier>${platform}</classifier>
                <scope>provided</scope>
        </dependency>
        <dependency>
                <groupId>log4j</groupId>
                <artifactId>log4j</artifactId>
                <version>1.2.14</version>
        </dependency>
        <dependency>
                <groupId>junit</groupId>
                <artifactId>junit</artifactId>
                <version>4.1</version>
                <scope>test</scope>
        </dependency>
        <dependency>
                <groupId>org.apache.tapestry</groupId>
                <artifactId>tapestry-core</artifactId>
                <version>${tapestry-release-version}</version>
        </dependency>
        <!--
                A dependency on either JUnit or TestNG is required, or the surefire
                plugin (which runs the tests) will fail, preventing Maven from
                packaging the WAR. Tapestry includes a large number of testing
                facilities designed for use with TestNG (http://testng.org/), so it's
                recommended.
        -->
        <dependency>
                <groupId>org.testng</groupId>
                <artifactId>testng</artifactId>
                <version>5.1</version>
                <classifier>jdk15</classifier>
                <scope>test</scope>
        </dependency>
</dependencies>
<build>
        <finalName>myapp</finalName>
        <resources>
                <resource>
                        <directory>src/main/resources</directory>
                        <includes>
                                <include>**</include>
                        </includes>
                        <excludes>
                                <exclude>hibernatecfg/*</exclude>
                                <exclude>hibernate.cfg.xml</exclude>
                                <exclude>log4j.properties</exclude>
                                <exclude>spy.properties</exclude>
                                <exclude>version.properties</exclude>
                                <exclude>**/.svn/**</exclude>
                        </excludes>
                </resource>
                <resource>
                        <directory>src/main/java</directory>
                        <includes>
                                <include>**/*</include>
                        </includes>
```

```
                                <excludes>
                                        <exclude>**/*.java</exclude>
                                        <exclude>**/.svn/**</exclude>
                                </excludes>
                        </resource>
                </resources>
                <plugins>
<!--  configure the GWT-Maven plugin -->
                        <plugin>
                                <groupId>com.totsp.gwt</groupId>
                                <artifactId>maven-googlewebtoolkit2-plugin</artifactId>
                                <version>2.0-beta26</version>
                                <configuration>
                                        <logLevel>INFO</logLevel>
                                        <compileTargets>
                                                <value>se.pmdit.tutorial.t5gwt.gwt.Sample</value>
                                                <value>se.pmdit.tutorial.t5gwt.gwt.ComplexSample</value>
                                                <value>se.pmdit.tutorial.t5gwt.gwt.StockWatcher</value>
                                        </compileTargets>
                                        <runTarget>/myapp/Start</runTarget>
                                        <style>DETAILED</style>
                                        <noServer>false</noServer>
                                        <extraJvmArgs>-Xmx512m</extraJvmArgs>
                                        <!--
                                                this parameter is VERY important with automatic mode - has to
                                                match the version in your declared deps
                                        -->
                                        <!--
                                                if this is set incorrect, or left out and default does not match
                                                (default is 1.5.3) you will have mysterious errors
                                        -->
                                        <gwtVersion>${gwtVersion}</gwtVersion>
                                        <!-- This puts the GWT output (javascripts) into the source tree -->
                                        <output>src/main/webapp</output >
                                </configuration>
                                <executions>
                                        <execution>
                                                <goals>
                                                        <!--<goal>mergewebxml</goal>
                                                        --><!--<goal>i18n</goal>
                                                        -->
                                                        <goal>compile</goal>
                                                        <!--<goal>test</goal>
                                                -->
                                                </goals>
                                        </execution>
                                </executions>
                        </plugin>
                        <!--  Use the dependency plugin to unpack gwt-dev-PLATFORM-libs.zip -->
                        <!--
                                (this is a replacement for the old "automatic" mode - useful if you
                                don't have GWT installed already, or you just want a maven way to
                                handle gwt deps)
                        -->
                        <plugin>
                                <groupId>org.apache.maven.plugins</groupId>
                                <artifactId>maven-dependency-plugin</artifactId>
                                <executions>
                                        <execution>
                                                <id>unpack</id>
                                                <phase>compile</phase>
                                                <goals>
                                                        <goal>unpack</goal>
                                                </goals>
                                                <configuration>
                                                        <artifactItems>
                                                                <artifactItem>
                                                                        <groupId>com.google.gwt</groupId>
                                                                        <artifactId>gwt-dev</artifactId>
                                                                        <version>${gwtVersion}</version>
                                                                        <classifier>${platform}-libs<
```

```
/classifier>
                                                                        <type>zip</type>
                                                                        <overWrite>false</overWrite>
                                                                        <outputDirectory>${settings.
localRepository}/com/google/gwt/gwt-dev/${gwtVersion}</outputDirectory>
                                                                    </artifactItem>
                                                                </artifactItems>
                                                            </configuration>
                                                    </execution>
                                            </executions>
                                    </plugin>
                                    <!--
                                            If you want to use the target/web.xml file mergewebxml produces,
                                            tell the war plugin to use it. Also, exclude what you want from the
                                            final artifact here.
                                    -->
                                    <plugin>
                                            <groupId>org.apache.maven.plugins</groupId>
                                            <artifactId>maven-war-plugin</artifactId>
                                            <configuration><!--
                                                    <webXml>src/main/webapp/WEB-INF/web.xml</webXml>
                                                    --><warSourceExcludes>.gwt-tmp/**</warSourceExcludes>
                                            </configuration>
                                    </plugin>
                                    <!--  tell the compiler we can use 1.5 -->
                                    <plugin>
                                            <groupId>org.apache.maven.plugins</groupId>
                                            <artifactId>maven-compiler-plugin</artifactId>
                                            <configuration>
                                                    <source>1.5</source>
                                                    <target>1.5</target>
                                            </configuration>
                                    </plugin>
                            </plugins>
                    </build>
<!--  profiles (with activation per platform) -->
                    <profiles>
                            <profile>
                                    <id>gwt-dev-windows</id>
                                    <properties>
                                            <platform>windows</platform>
                                    </properties>
                                    <activation>
                                            <activeByDefault>true</activeByDefault>
                                            <os>
                                                    <family>windows</family>
                                            </os>
                                    </activation>
                            </profile>
                            <profile>
                                    <id>gwt-dev-mac</id>
                                    <properties>
                                            <platform>mac</platform>
                                    </properties>
                                    <activation>
                                            <activeByDefault>false</activeByDefault>
                                            <os>
                                                    <family>mac</family>
                                            </os>
                                    </activation>
                            </profile>
                            <profile>
                                    <id>gwt-dev-linux</id>
                                    <properties>
                                            <platform>linux</platform>
                                    </properties>
                                    <activation>
                                            <activeByDefault>false</activeByDefault>
                                            <os>
                                                    <name>linux</name>
                                            </os>
```

```
                        </activation>
                </profile>
        </profiles>
        <reporting>
                <!--
                        Adds a report detailing the components, mixins and base classes
                        defined by this module.
                -->
                <plugins>
                        <plugin>
                                <groupId>org.apache.tapestry</groupId>
                                <artifactId>tapestry-component-report</artifactId>
                                <version>${tapestry-release-version}</version>
                                <configuration>
                                        <rootPackage>org.example.myapp</rootPackage>
                                </configuration>
                        </plugin>
                </plugins>
        </reporting>
        <repositories>
                <!--
                        This can be commented out when the tapestry-release-version is a not
                        a snapshot. The non-snapshot Tapestry artifacts are distributed
                        through the central repository at ibiblio.
                -->
                <repository>
                        <id>tapestry-snapshots</id>
                        <url>http://tapestry.formos.com/maven-snapshot-repository/
                        </url>
                </repository>
                <repository>
                        <id>codehaus.snapshots</id>
                        <url>http://snapshots.repository.codehaus.org
                        </url>
                </repository>
                <repository>
                        <id>OpenQA_Release</id>
                        <name>OpenQA Release Repository</name>
                        <url>http://archiva.openqa.org/repository/releases/
                        </url>
                </repository>
                <repository>
                        <id>gwt-maven</id>
                        <url> http://gwt-maven.googlecode.com/svn/trunk/mavenrepo/</url>
                </repository>
        </repositories>
        <pluginRepositories>
                <!--
                        As above, this can be commented out when access to the snapshot
                        version of a Tapestry Maven plugin is not required.
                -->
                <pluginRepository>
                        <id>tapestry-snapshots</id>
                        <url>http://tapestry.formos.com/maven-snapshot-repository/
                        </url>
                </pluginRepository>
                <pluginRepository>
                        <id>gwt-maven-plugins</id>
                        <url> http://gwt-maven.googlecode.com/svn/trunk/mavenrepo/</url>
                </pluginRepository>
        </pluginRepositories>
        <properties>
                <gwtVersion>1.5.3</gwtVersion>
                <tapestry-release-version>5.0.18</tapestry-release-version>
        </properties>
</project>
```

For now, I manually make additions to the web.xml if a servlet mapping is needed. So far this example only uses one, for the Stock Prices.

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
>
        <session-config>
                <session-timeout> 30 </session-timeout>
        </session-config>
        <welcome-file-list>
                <welcome-file>Start</welcome-file>
        </welcome-file-list>
        <display-name>Tapestry 5 With GWT</display-name>
        <context-param>
                <param-name>tapestry.app-package</param-name>
                <param-value>se.pmdit.tutorial.t5gwt.tapestry</param-value>
        </context-param>
        <filter>
                <filter-name>app</filter-name>
                <filter-class>org.apache.tapestry5.TapestryFilter</filter-class>
        </filter>
        <filter-mapping>
                <filter-name>app</filter-name>
                <url-pattern>/*</url-pattern>
        </filter-mapping>
        <servlet>
                <servlet-name>stock</servlet-name>
                <servlet-class>se.pmdit.tutorial.t5gwt.gwt.server.StockPriceServiceImpl</servlet-class>
        </servlet>
        <servlet-mapping>
                <servlet-name>stock</servlet-name>
                <url-pattern>/se.pmdit.tutorial.t5gwt.gwt.StockWatcher/stockPrices</url-pattern>
        </servlet-mapping>
</web-app>
```

Adding GWT Targets in the POM

Notice that for each entrypoint you want to have compiled, you add it here.

```
...
                                        <compileTargets>
                                              <value>se.pmdit.tutorial.t5gwt.gwt.Sample</value>
                                              <value>se.pmdit.tutorial.t5gwt.gwt.ComplexSample</value>
                                              <value>se.pmdit.tutorial.t5gwt.gwt.StockWatcher</value>
                                       </compileTargets>
...
```

The StockWatcher Entrypoint

```
package se.pmdit.tutorial.t5gwt.gwt.client;

import com.google.gwt.core.client.EntryPoint;
import com.google.gwt.i18n.client.Dictionary;
import com.google.gwt.user.client.Timer;
import com.google.gwt.user.client.ui.RootPanel;
/**
 *
 * @author djue
 *
 */
public class StockWatcher implements EntryPoint {

	private static final int REFRESH_INTERVAL = 5000; // ms

	public void onModuleLoad() {

		// Find all the instances in this DOM where this entry point should be
		// created.
		// The dictionary is created by
		Dictionary gwtComponents = Dictionary.getDictionary("gwtComponents");
		if (gwtComponents != null) {
			String str[] = gwtComponents.get("stockwatcher").split(",");
			for (int i = 0; i < str.length; i++) {
				createUniqueModule(str[i]);
			}
		}

	}

	/**
	 * @param string
	 */
	private void createUniqueModule(String string) {

		// // add the main panel to the HTML element with the id "stockList"
		// RootPanel.get("stockList"+string).add(mainPanel);

		final StockWatcherPanel swp = new StockWatcherPanel();
		// add the main panel to the HTML element with the id "stockList"
		RootPanel.get("stockList" + string).add(swp);

		// setup timer to refresh list automatically
		Timer refreshTimer = new Timer() {
			public void run() {
				swp.refreshWatchList();
			}
		};
		refreshTimer.scheduleRepeating(REFRESH_INTERVAL);

		// move cursor focus to the text box
		swp.newSymbolTextBox.setFocus(true);

	}

}
```

The StockWatcherComponent

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package se.pmdit.tutorial.t5gwt.tapestry.components;

import org.apache.tapestry5.Asset;
import org.apache.tapestry5.ComponentResources;
import org.apache.tapestry5.RenderSupport;
import org.apache.tapestry5.annotations.Environmental;
import org.apache.tapestry5.annotations.IncludeJavaScriptLibrary;
import org.apache.tapestry5.annotations.Path;
import org.apache.tapestry5.annotations.Property;
import org.apache.tapestry5.ioc.annotations.Inject;

/**
 *
 * @author djue
 */
@IncludeJavaScriptLibrary("context:se.pmdit.tutorial.t5gwt.gwt.StockWatcher/se.pmdit.tutorial.t5gwt.gwt.
StockWatcher.nocache.js")
public class StockWatcherComponent extends GWTEntryPointComponent {
        @Environmental
        private RenderSupport renderSupport;
        @Inject
        private ComponentResources resources;

        @Inject
        @Path("context:se.pmdit.tutorial.t5gwt.gwt.StockWatcher/images/GoogleCode.png")
        @Property
        private Asset banner;

        void setupRender() {
                renderSupport.addScript("GWTComponentController.add('stockwatcher','" + resources.
getCompleteId() + "')");
        }
        // If you wanted you could output the DIV from here, then you would not
        // need the template file.
}
```

Make special note of this line:

```
...
renderSupport.addScript("GWTComponentController.add('stockwatcher','" + resources.getCompleteId() + "')");
...
```

It's kind of like adding it to a HashMap<String,String>. (It's not really a HashMap, but later your Entrypoint will look up values associated with a key; The key will be 'stockwatcher', and the values will be the T5 generated ids)

The StockWatcherComponent.tml

Notice the div where the id becomes stockList+component id. This allows a unique and independent stock list to be created.

```
<div xmlns:t="http://tapestry.apache.org/schema/tapestry_5_0_0.xsd">
Stock Watcher Component
<div id="${resources.completeId}"></div>
<img src="${banner}" />
<h1>Stock Watcher</h1>
<div id="stockList${resources.completeId}"></div>
</div>
```

Note that this:

```
...
<div id="${resources.completeId}"></div>
...
```

Is the same id that you added in the "HashMap" above.

General Tips

In general it can be very easy to get GWT working with Tapestry. The complexity comes in when you want GWT to be a component, and you consider that a component may exist many times on one page. (think of dialog boxes). In order for the components to behave properly, you must somehow provide a unique ID to the Javascript object that GWT will create.

In your GWT code, make the Entrypoint class do as little as possible. I've read this elsewhere as a GWT mini-pattern, but it helps alot here! Even references to your GWT services can be put into the separate class that your Entrypoint will instantiate. In fact the Entrypoints I show here do pretty much the same thing:

- They load a dictionary and find all the values for a certain "key"
- For each value, a new instance of something is created

For example, there will be a String like this:

```
mydialog,mydialog1,mydialog2
```

These are the unique id's that Tapestry will create for you. Some included Javascript (from the other GWT tutorial) will copy these unique id's into a Javascript variable. The compiled GWT Javascript will then read that list of id's in your Entrypoint (as shown later)

So instead of having all kinds of code in your entrypoint class, refactor it into a separate class that your entrypoint will add to the RootPanel.

The other tip is that when developing GWT by itself, you usually have a HTML file that runs your Javascript. You can still have those if that's what you want, but when using a GWT entrypoint as a compnent, you put the HTML into a Tapestry component/template.

The best way to understand this is to get this sample app working and play with it. You'll see that the component TML is minimal, and in some cases, could be done away with altogether.

Some unusual things about this

The GWT is configured in the POM to output it's compiled directories into your src/webapp. (If someone knows a way to use Maven Assembly or an Ant copy that will work better, please add it to this wiki) This will require a refresh of your source dir after compilation, or Eclipse won't see the GWT compiled code. If Eclipse does not see the new files, then your Tomcat instance won't see it either if you are launching Tomcat from Eclipse.

I have not included the gwt compiled directories in the Zip file, because they should be generated for you when you run:

```
mvn clean compile gwt:compile
```

clean will delete the Target directory compile will take care of Java compilation like your T5 code gwt:compile will take care of your GWT generated code (the Javascript)

You could also launch tomcat afterwards using Maven, but I find this less thank friendly.

You can leave "Build Automatically" checked in your Eclipse project settings, and Tomcat should restart when you make changes on the Tapestry side. But you'll need to restart Tomcat otherwise.