AcegiSpringJava5

NOTE: This information applies to an older version of Tapestry. For current information on Tapestry's Spring support, see http://tapestry.apache.org//integrating-with-spring-framework.html.

The integration of Tapestry, Acegi and Spring seems to be a hot topic.

I have created a simple example (this example uses the Tapestry-Acegi module) that shows how that can be done. The complete source code can be found here: http://www.zedlitz.de/tapestry-acegi.zip (8kB) (This example uses Tapestry 4.0.1)

If you have Maven2 installed you can go into the directoy tapestry-acegi and enter the command "mvn jetty:run". Then you can access the application via this URL: http://localhost:8080/tapestry-acegi/app

html pages

First I needed two pages (one secured):

`src/main/webapp/Home.html`

```
<html>
  <head>
    <title>tapestry-acegi: normal page</title>
  </head>
  <body>
    <h1>tapestry-acegi: normal page</h1>
    <p><a href="Secured.html" jwcid="@PageLink" page="Secured">secured page</a></p>
  </body>
</html>
```

`src/main/webapp/Secured.html`

```
<html>
  <head>
    <title>tapestry-acegi: secured page</title>
  </head>
  <body>
    <h1>tapestry-acegi: secured page</h1>
    <p><a href="Home.html" jwcid="@PageLink" page="Home">back to homepage</a></p>
  </body>
</html>
```

Java classes

Each of these pages is accompanied by a (trivial) Java class:

`src/main/java/de/zedlitz/tapestry/acegi/Home.java`

```
package de.zedlitz.tapestry.acegi;
public abstract class Home extends org.apache.tapestry.html.BasePage { }
```

`src/main/java/de/zedlitz/tapestry/acegi/Secured.java`

```
package de.zedlitz.tapestry.acegi;

@org.acegisecurity.annotation.Secured("ROLE_USER")
public abstract class Secured extends org.apache.tapestry.html.BasePage { }
```

This is the only place where you have to specify that this page is secured! 🙂

Tapestry configuration

I have to tell Tapestry in which package to find the classes:

`src/main/webapp/WEB-INF/tapestry-acegi.application`

```
<?xml version="1.0"?>
<!DOCTYPE application PUBLIC
  "-//Apache Software Foundation//Tapestry Specification 4.0//EN"
  "http://jakarta.apache.org/tapestry/dtd/Tapestry_4_0.dtd">

<application>
    <meta key="org.apache.tapestry.page-class-packages" value="de.zedlitz.tapestry.acegi"/>
</application>
```

web.xml

A standard webapplication setup for Tapestry and Spring:

`src/main/webapp/WEB-INF/web.xml`

```
<!DOCTYPE web-app
       PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
       "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>
     <display-name>tapestry-acegi</display-name>

     <servlet>
        <servlet-name>tapestry-acegi</servlet-name>
        <servlet-class>org.apache.tapestry.ApplicationServlet</servlet-class>
        <load-on-startup>0</load-on-startup>
     </servlet>

     <servlet-mapping>
         <servlet-name>tapestry-acegi</servlet-name>
         <url-pattern>/app</url-pattern>
     </servlet-mapping>

     <context-param>
         <param-name>contextConfigLocation</param-name>
         <param-value>classpath:/applicationContext.xml</param-value>
     </context-param>

     <listener>
         <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
     </listener>
</web-app>
```

Spring configuration

The Spring configuration is also not spectacular. Only one bean, the "memoryAuthenticationDao" is defined. In the example the username and password is specified here. But of course you can put any complex bean here to retrieve user information.

`src/main/resources/applicationContext.xml`

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

<!DOCTYPE beans PUBLIC
    "-//SPRING//DTD BEAN//EN"
    "http://www.springframework.org/dtd/spring-beans.dtd">

<beans>
    <bean id="memoryAuthenticationDao" class="org.acegisecurity.userdetails.memory.InMemoryDaoImpl">
        <property name="userMap">
            <value>
                tester=secret,ROLE_USER,ROLE_SUPERVISOR
            </value>
        </property>
    </bean>
</beans>
```

hivemodule.xml

Now the interesting part - how to put it together:

`src/main/resources/META-INF/hivemodule.xml`

```
<module id="de.zedlitz.tapestry.acegi" version="1.0.0">
    <contribution configuration-id="hivemind.ApplicationDefaults">
        <default
           symbol="hivemind.acegi.dao.passwordEncoder"
           value="org.acegisecurity.providers.encoding.PlaintextPasswordEncoder"/>
       <default symbol="hivemind.acegi.dao.systemWideSalt" value="" />
    </contribution>

    <implementation service-id="hivemind.acegi.dao.UserDetailsService">
        <invoke-factory service-id="hivemind.lib.SpringLookupFactory">
          <lookup-bean name="memoryAuthenticationDao" />
        </invoke-factory>
    </implementation>

    <contribution configuration-id="hivemind.acegi.AccessDecisionVoters">
       <voter object="instance:org.acegisecurity.vote.RoleVoter" />
    </contribution>
</module>
```

pom.xml

`pom.xml`

```
<?xml version="1.0"?>
<project>
   <modelVersion>4.0.0</modelVersion>
   <groupId>de.zedlitz</groupId>
   <artifactId>tapestry-acegi</artifactId>
   <packaging>war</packaging>
   <name>tapestry-acegi</name>
   <version>1.0-SNAPSHOT</version>
   <description>a very simple example that shows how to integrate Tapestry4 and Acegi</description>
   <build>
      <finalName>tapestry-acegi</finalName>
      <plugins>
         <plugin>
            <artifactId>maven-compiler-plugin</artifactId>
            <configuration>
               <source>1.5</source>
               <target>1.5</target>
            </configuration>
         </plugin>
```

```
         <plugin>
            <groupId>org.mortbay.jetty</groupId>
            <artifactId>maven-jetty-plugin</artifactId>
         </plugin>
      </plugins>
   </build>

   <dependencies>
      <dependency>
         <groupId>tapestry</groupId>
         <artifactId>tapestry-annotations</artifactId>
         <version>4.0.1</version>
      </dependency>
      <dependency>
         <groupId>tapestry</groupId>
         <artifactId>tapestry</artifactId>
         <version>4.0.1</version>
      </dependency>
      <dependency>
         <artifactId>acegi-security-tiger</artifactId>
         <groupId>org.acegisecurity</groupId>
         <version>1.0.1</version>
      </dependency>
      <dependency>
      <groupId>com.javaforge.tapestry</groupId>
      <artifactId>tapestry-acegi</artifactId>
      <version>0.1-SNAPSHOT</version>
         </dependency>
      <dependency>
         <groupId>org.springframework</groupId>
         <artifactId>spring</artifactId>
         <version>1.2.7</version>
      </dependency>
      <dependency>
         <groupId>com.javaforge.hivemind</groupId>
         <artifactId>hivemind-acegi-dao</artifactId>
         <version>0.1-SNAPSHOT</version>
      </dependency>
      <dependency>
         <groupId>com.javaforge.tapestry</groupId>
         <artifactId>tapestry-spring</artifactId>
         <version>0.1.2</version>
      </dependency>
      <dependency>
         <groupId>hivemind</groupId>
         <artifactId>hivemind-lib</artifactId>
         <version>1.1.1</version>
      </dependency>
   </dependencies>

   <repositories>
      <repository>
         <id>carmanconsulting</id>
         <name>carmanconsulting</name>
         <url>http://www.carmanconsulting.com/mvn/</url>
      </repository>
   </repositories>
</project>
```

next parts of the tutorial

- AcegiSpringJava5Part2 - Who am I? Displaying the name of the authentication user.
- AcegiSpringJava5FormBased - An idea of how a form based login could work

tips and tricks

- You can change the realm name used for HTTP basic authentication by adding these lines to{{src/main/resources/META-INF/hivemodule.xml}}:

```
    <contribution configuration-id="hivemind.ApplicationDefaults">
        <default symbol="tapestry.acegi.realmName" value="Member area" />
    </contribution>
```