Template Project Development

[TemplateProjectDevelopment.htm - uqlpayne 29 Jul 2013]

Contents

- Requirements
- Template LabEquipment
  - Configuring
  - Running
- Template LabServer
  - Configuring
  - Running
- Dummy ServiceBroker
  - Configuring
  - Running
- Template LabEquipment
  - Configuring
  - Running

Requirements

- The *NetBeans IDE 7.3.1 + GlassFish Server Open Source Edition 4.0* is installed and running.
- The *UQ-iLab-BatchLabServer-Java* GitHub repository has been downloaded as a ZIP file from:

  `https://github.com/uqlpayne/UQ-iLab-BatchLabServer-Java`

- The *Template LabServer Database* has been created and configured (see TemplateLabServerDatabase.pdf).
- The *Dummy ServiceBroker Database* has been created and configured (see DummyServiceBrokerDatabase.pdf).

Template LabEquipment

Configuring

- In the NetBeans IDE, open the *LabEquipment* web application project in the source code folder *UQ-iLab-BatchLabServer-Java/Template* including its required projects.
- In the *Template_LabEquipment* project, go to the *Web Pages/WEB-INF* folder and open the *ConfigProperties.xml* file to see what's in it. A template of this file is provided in the *ConfigProperties_template.xml* file.

  No changes need to be made to the *ConfigProperties.xml* file at this stage because it already contains the information necessary while working in the development environment.
- Open the *web.xml* file which is also in the same folder just to see what's in it. No changes need to be made to this file.

Running

- Run the *Template_LabEquipment* web application project and watch the *Output* window in the IDE. You should see the project being built, then the GlassFish Server starting and deploying the web application.
- In the *Template_LabEquipment* project, go to *Web Services*, right-click *LabEquipmentService*

and select *Test Web Service*. The *LabEquipmentService Web Service Tester* page will open in the web browser.

Click the *getLabEquipmentStatus* button. This will throw an exception but have a look at the *Output* window in the IDE. You should see logging information showing the LabEquipment service starting and then running.

- An exception was thrown because the web service tester did not provide the LabServer's GUID and passkey.

  Open the *ConfigProperties.xml* file, change the value for the *Authenticating* key to *false* and save the file.

  Run the *Template_LabEquipment* web application project again and then run the web service tester again. When the *getLabEquipmentStatus* button is clicked this time, no exception is thrown. Instead, you will see a *SOAP Response* showing that the LabEquipment is *online*.

  Open the *ConfigProperties.xml* file, change the value for the *Authenticating* key back to *true* and save the file.

Template LabServer

Configuring

- In the NetBeans IDE, open the *LabServer* web application project in the source code folder *UQ-iLab-BatchLabServer-Java/Template* including its required projects.
- In the *Template_LabServer* project, go to the *Web Pages/WEB-INF* folder and open the *ConfigProperties.xml* file to see what's in it. A template of this file is provided in the *ConfigProperties_template.xml* file.

  No changes need to be made to the *ConfigProperties.xml* file at this stage because it already contains the information necessary while working in the development environment.
- Open the *web.xml* file which is also in the same folder just to see what's in it. No changes need to be made to this file.

Running

- Run the *Template_LabServer* web application project and watch the *Output* window in the IDE. You should see the project being built and then the GlassFish Server deploying the web application.
- In the *Template_LabServer* project, go to *Web Services*, right-click *LabServerWebService* and select *Test Web Service*. The *LabServerWebService Web Service Tester* page will open in the web browser.

  Click the *getLabStatus* button. This will throw an exception but look at the *Output* window in the IDE. You should see logging information showing the LabServer service starting and then running.

  An exception was thrown because the web service tester did not provide the ServiceBroker's GUID and passkey.
- When the *Template_LabServer* web application project was run, the LabServer's Home page was opened in the web browser. Log into the LabServer with the username *manager* and password *ilab*. This account was created when the *LabServer_Defaults.sql* database script was executed.
- Click on the *Self Registration* menu item to display the web page for configuring the LabServer.

  When the LabServer completes an experiment, an email can be sent to a specified email

address or addresses. Under *Optional Information*, enter your email address in the *Completed Email:* and *Failed Email:* fields. Multiple email addresses can be specified by separating them with a semicolon. No email is sent if no email addresses are specified.

The LabServer's web service can be tested by clicking the *Test* button. A *GetLabStatus* web service call is made to the *Service Url* specified. A message is displayed showing the status of the LabServer and LabEquipment similar to the message displayed on the LabClient's *Status* page.

The *Authenticate* checkbox enables ServiceBroker authentication and should remain checked. If unchecked then the *Web Service Tester* can be used to make a web service call to the LabServer without throwing an exception.

- Click on the *Lab Equipment* menu item to display the web page for configuring the LabEquipment Service implementations or units. It is possible to configure multiple identical units in a *farm* to provide load sharing and reduced response times under heavy load. Generally, a LabServer will only use one LabEquipment unit.

  In the development environment, it is useful to have multiple LabEquipment units configured and then enable or disable them as required. For example, one LabEquipment unit might be implemented in Java while another is implement in Microsoft .NET. Alternatively, one LabEquipment unit might reside on the local computer and another on a remote server.

  The LabEquipment's web service can be tested by clicking the *Test* button. A *GetLabEquipmentStatus* web service call is made to the *Service Url* specified. A message is displayed showing the status of the LabEquipment unit. page.

- Click on the *ServiceBrokers* menu item to display the web page for configuring ServiceBroker access to this LabServer. There is only one ServiceBroker listed, *localhost*. The *localhost* ServiceBroker was the one that allowed the *Test* button on the *Self Registration* page to get the lab status of the LabServer. If the *Allowed* checkbox was unchecked then a message would have been displayed stating that LabServer access was denied.

- Click on the *My Account* menu item to display the web page for the LabServer Manager's user information. Only one user account exists and cannot be deleted although all information other than the username can be changed.

  It is advisable to change the password particularly if the LabServer is deployed outside of the development environment.

- Click on the *Logout* menu item to log out of the LabServer.

Dummy ServiceBroker

Configuring

- In the NetBeans IDE, open the *ServiceBroker* web application project in the source code folder *UQ-iLab-BatchLabServer-Java/Dummy* including its required projects.

- In the *Dummy_ServiceBroker* project, go to the *WEB-INF* folder and open the *ConfigProperties.xml* file to see what's in it. There is no template of this file provided.

  No changes need to be made to the *ConfigProperties.xml* file.

  Note: A GUID has already been provided for the ServiceBroker and is for use in the development environment only. This ServiceBroker simply provides pass-through methods to allow the LabClient to communicate with the LabServer. It will not be deployed to an application web server.

  There are three LabServers configured for the ServiceBroker. Each must have a different GUID and WebService Url. *LabServer0* is configured for a Microsoft .NET implementation hence the non-Java WebService Url.

- Open the *web.xml* file which is also in the same folder just to see what's in it. No changes need to be made to this file.

Running

- Run the *Dummy_ServiceBroker* web application project and watch the *Output* window in the IDE. You should see the project being built and then the GlassFish Server deploying the web application.
- In the *Dummy_ServiceBroker* project, go to *Web Services*, right-click *ServiceBrokerService* and select *Test Web Service*. The *ServiceBrokerService Web Service Tester* page will open in the web browser.

  Click the *getLabStatus* button. This will throw an exception but have a look at the *Output* window in the IDE. You should see logging information showing the ServiceBroker service starting and then running.
- An exception was thrown because the web service tester did not provide the LabClient's coupon Id and coupon passkey.

  Open the *ConfigProperties.xml* file, change the value for the *Authenticating* key to *false* and save the file.

  Run the *Dummy_ServiceBroker* web application project again and then run the web service tester again. When the *getLabStatus* button is clicked this time, an exception is thrown again. But this time, it is because the LabServer's GUID was not provided in the parameter input box beside the button.

  Copy the LabServer's GUID from *LabServer1* (TemplateLabServer) in the *ConfigProperties.xml* file and paste into the parameter input box beside the *getLabStatus* button.

  Click the *getLabStatus* button. You will see a *SOAP Response* showing that the LabServer is *online*.
- Open the *ConfigProperties.xml* file, change the value for the *Authenticating* key back to *true* and save the file. Run the *Dummy_ServiceBroker* web application project again.

Template LabClient

Configuring

- In the NetBeans IDE, open the *LabClient* web application project in the source code folder *UQ-iLab-BatchLabServer-Java/Template* including its required projects.
- In the *Template_LabClient* project, go to the *Web Pages/WEB-INF* folder and open the *ConfigProperties.xml* file to see what's in it. A template of this file is provided in the *ConfigProperties_template.xml* file.

  No changes need to be made to the *ConfigProperties.xml* file.
- Open the *web.xml* file which is also in the same folder just to see what's in it. No changes need to be made to this file.
- Open the *Properties* dialog window for the *Template_LabClient* project and select the *Run* category. Check that the following *Relative URL:* is specified:

  ```
  /LabClient.do?CouponId=12345&Passkey=qwerty
  ```

  The URL request parameters provide the coupon Id and coupon passkey required by the Dummy ServiceBroker. A real ServiceBroker provides these request parameters (with different values) when it launches the LabClient.

  If values for *ServiceUrl*, *LabServerId*, or *MultiSubmit* are specified in the URL request

parameters along with *CouponId* and *Passkey*, these override the values set in the *ConfigProperties.xml* file. This way, multiple ServiceBrokers can launch the same LabClient but with different parameters.

Running

- Run the *Template_LabClient* web application project and watch the *Output* window in the IDE. You should see the project being built and then the GlassFish Server deploying the web application.
- The LabClient's Home page is opened in the web browser. If all is well, you should see the title *Template* at the top of the page and a version number near the top-right of the page.
- Click on the *Status* menu item to display the web page for LabServer's status which should be *Online*.
- Click on the *Setup* menu item to display the web page for specifying the experiment setup to run on the LabServer.

  For the *Setup*, select *Setup for Equipment*. Enter the value *500* in the input box next to *SomeParameter* and click the *Validate button*. A message is displayed stating that the experiment specification valid and will take 22 seconds to execute.

  Click the *Submit* button to submit this experiment specification to the LabServer. A message is displayed stating that the submission was successful and the experiment number.
- Click on the *Status* menu item to display the web page for the experiment status. Enter the experiment number in the input box next to *Experiment Id* (if the number is not already there) and click the *Check* button. A message is displayed showing the status of the experiment.

  Periodically click the *Check* button until the experiment has completed.
- Click on the *Results* menu item to display the web page for the experiment results. Enter the experiment number in the input box next to *Experiment Id* (if the number is not already there) and click the *Retrieve* button. A message is displayed showing the status of the experiment.

  Also displayed is the information about the experiment including the experiment setup and experiment results. The result for this experiment is shown as *SomeResult* and will be a randomly-generated number in the range of 0 to 500.
- The results can be saved to a comma-separated-value (CSV) file by clicking the *Save button*.

That's it.