Reuben Steenekamp

Exam No

141113020948

PAT

FancyShader

Reddam House Constantia

2014

Table of Contents

Contents

[Project Specifications 6](#_Toc399713995)

[Summary 6](#_Toc399713996)

[Specifications of Program Function 6](#_Toc399713997)

[Specifications of User Interface 6](#_Toc399713998)

[Browser 6](#_Toc399713999)

[Editor 7](#_Toc399714000)

[Exit confirmation dialog 8](#_Toc399714001)

[Login 8](#_Toc399714002)

[Register 8](#_Toc399714003)

[Detail Editor 8](#_Toc399714004)

[Detail Viewer 8](#_Toc399714005)

[Common Elements 8](#_Toc399714006)

[Specification of Help 8](#_Toc399714007)

[Specifications of Data Storage 9](#_Toc399714008)

[Hardware and Software requirements 9](#_Toc399714009)

[Server 9](#_Toc399714010)

[Client 9](#_Toc399714011)

[System Design Document 9](#_Toc399714012)

[User Interface Design 9](#_Toc399714013)

[Login 9](#_Toc399714014)

[Register 10](#_Toc399714015)

[Browser 10](#_Toc399714016)

[Detail Editor 12](#_Toc399714017)

[Detail Viewer 12](#_Toc399714018)

[Editor 13](#_Toc399714019)

[Exit confirmation 14](#_Toc399714020)

[Sequencing 14](#_Toc399714021)

[Client 14](#_Toc399714022)

[Server 24](#_Toc399714023)

[Class Design 24](#_Toc399714024)

[Exclusions and the reasoning 24](#_Toc399714025)

[Class Diagram 25](#_Toc399714026)

[Persistent Storage Design 26](#_Toc399714027)

[Explanation of Storage Design 26](#_Toc399714028)

[Why SQL 26](#_Toc399714029)

[Why Apache Derby Embedded 26](#_Toc399714030)

[Why This Table Design 26](#_Toc399714031)

[CODING and Technical Documentation 27](#_Toc399714032)

[Program Commenting 27](#_Toc399714033)

[Separation of UI from working code 27](#_Toc399714034)

[Inter-Code communication (Typed Methods and Parameters) 27](#_Toc399714035)

[General Techniques 28](#_Toc399714036)

[Persistent storage/Querying 29](#_Toc399714037)

[Data validation, exception handling, error messages 29](#_Toc399714038)

[Fulfilment of Specifications 29](#_Toc399714039)

[User Experience report 30](#_Toc399714040)

[Source Code 30](#_Toc399714041)

[FancyShaderClient\src\fancyshader\client\FancyShaderClient.java 30](#_Toc399714042)

[FancyShaderClient\src\fancyshader\client\FancyShaderRemote.java 31](#_Toc399714043)

[FancyShaderClient\src\fancyshader\client\gui\Browser.fxml 32](#_Toc399714044)

[FancyShaderClient\src\fancyshader\client\gui\BrowserController.java 33](#_Toc399714045)

[FancyShaderClient\src\fancyshader\client\gui\ConfirmExit.fxml 39](#_Toc399714046)

[FancyShaderClient\src\fancyshader\client\gui\ConfirmExitController.java 40](#_Toc399714047)

[FancyShaderClient\src\fancyshader\client\gui\ConsumerWithException.java 41](#_Toc399714048)

[FancyShaderClient\src\fancyshader\client\gui\DetailEditor.fxml 42](#_Toc399714049)

[FancyShaderClient\src\fancyshader\client\gui\DetailEditorController.java 42](#_Toc399714050)

[FancyShaderClient\src\fancyshader\client\gui\DetailViewer.fxml 45](#_Toc399714051)

[FancyShaderClient\src\fancyshader\client\gui\DetailViewerController.java 46](#_Toc399714052)

[FancyShaderClient\src\fancyshader\client\gui\Editor.fxml 47](#_Toc399714053)

[FancyShaderClient\src\fancyshader\client\gui\EditorController.java 48](#_Toc399714054)

[FancyShaderClient\src\fancyshader\client\gui\Helper.java 52](#_Toc399714055)

[FancyShaderClient\src\fancyshader\client\gui\Login.fxml 54](#_Toc399714056)

[FancyShaderClient\src\fancyshader\client\gui\LoginController.java 55](#_Toc399714057)

[FancyShaderClient\src\fancyshader\client\gui\MainController.java 56](#_Toc399714058)

[FancyShaderClient\src\fancyshader\client\gui\Register.fxml 63](#_Toc399714059)

[FancyShaderClient\src\fancyshader\client\gui\RegisterController.java 64](#_Toc399714060)

[FancyShaderClient\src\fancyshader\client\gui\Splash.fxml 66](#_Toc399714061)

[FancyShaderClient\src\fancyshader\client\viewer\DemoAnimator.java 66](#_Toc399714062)

[FancyShaderClient\src\fancyshader\client\viewer\DemoRunner.java 68](#_Toc399714063)

[FancyShaderClient\src\fancyshader\client\viewer\DemoViewer.java 72](#_Toc399714064)

[FancyShaderClient\resources\about.html 73](#_Toc399714065)

[FancyShaderClient\resources\browser.html 74](#_Toc399714066)

[FancyShaderClient\resources\editor.html 75](#_Toc399714067)

[FancyShaderClient\resources\main.css 76](#_Toc399714068)

[FancyShaderClient\build.xml 76](#_Toc399714069)

[FancyShaderLib\src\fancyshader\entity\Account.java 76](#_Toc399714070)

[FancyShaderLib\src\fancyshader\entity\AuthDetail.java 80](#_Toc399714071)

[FancyShaderLib\src\fancyshader\entity\ShaderDemo.java 83](#_Toc399714072)

[FancyShaderLib\src\fancyshader\service\FancyShaderInternalException.java 87](#_Toc399714073)

[FancyShaderLib\src\fancyshader\service\FancyShaderLoginException.java 88](#_Toc399714074)

[FancyShaderLib\src\fancyshader\service\FancyShaderRemoteService.java 88](#_Toc399714075)

[FancyShaderLib\src\fancyshader\service\FancyShaderService.java 89](#_Toc399714076)

[FancyShaderLib\src\fancyshader\service\FancyShaderSessionExpiredException.java 92](#_Toc399714077)

[FancyShaderLib\src\META-INF\persistence.xml 93](#_Toc399714078)

[FancyShaderServer\src\fancyshader\server\FancyShaderRemoteServiceImpl.java 93](#_Toc399714079)

[FancyShaderServer\src\fancyshader\server\FancyShaderServer.java 95](#_Toc399714080)

[FancyShaderServer\src\fancyshader\server\FancyShaderServiceImpl.java 96](#_Toc399714081)

[FancyShaderServer\src\fancyshader\server\FancyShaderSession.java 102](#_Toc399714082)

[FancyShaderServer\src\fancyshader\server\FancyShaderSessionSingleton.java 104](#_Toc399714083)

[FancyShaderServer\fancyshaderdb.sql 105](#_Toc399714084)

[Generated Javadocs 106](#_Toc399714085)

[Technical Documentation 106](#_Toc399714086)

[External code 106](#_Toc399714087)

[Explanation of Critical Algorithms 106](#_Toc399714088)

[Rendering of Shader 106](#_Toc399714089)

[Hashing and Salting of User Account Passwords 107](#_Toc399714090)

[Advanced Techniques 108](#_Toc399714091)

[Testing 109](#_Toc399714092)

[Test plan and results 109](#_Toc399714093)

[Login 109](#_Toc399714094)

[Register 111](#_Toc399714095)

[Search 113](#_Toc399714096)

# Project Specifications

## Summary

There are groups of people who strive to bring art, math and programming together in the form of procedurally generated visual displays. These people identify themselves as the demo scene. The programs they create are demos.

FancyShader is a utility used to browse, view, edit and share demos programmed as a shader in GLSL. Shaders were created with the intent to provide a programmable element in the transition from rasterized fragments in the 3D rendering pipeline of graphics cards to the final pixel colours outputted to the screen, but it has an alternative use case within the demo scene.

## Specifications of Program Function

The PAT consists of three NetBeans projects. An executable client and server, and a library containing the common elements between them.

The client utilizes the server’s interface and exposes its functionality to the user interface through several dialogs. A user can register and login. A logged in user can view publicly available demos as well as their own. They can also create, edit and fork demos. They can only edit a demo if they are the creator of the demo. Any demo visible to the user can be forked into a new demo, copying the source code. If they select a demo, it is marked as visited by them. They can also express delight by marking a demo as delightful. They can reverse the action by marking it as undelightful afterwards.

## Specifications of User Interface

### Browser

The browser window consists of:

* A menu bar
  + An account menu
    - A login button
      * Opens modal login dialog
      * Enabled only when user is logged out
    - A register button
      * Opens modal register dialog
    - A logout button
      * Enabled only when user is logged in
  + A demo menu
    - A create button
      * Opens modal demo detail editor dialog which in turn opens the demo editor window for the newly created demo upon completion
      * Enabled only when user is logged in
    - An edit button
      * Opens modal demo editor window for the selected demo
      * Enabled only when user is logged in, a demo is selected and the selected demo was created by the user
    - A fork button
      * Opens modal demo detail editor dialog which in turn opens the demo editor window for the newly forked demo upon completion
      * Enabled only when user is logged in and a demo is selected
    - An info button
      * Opens modal detail viewer dialog
      * Enabled only when user is logged in and a demo is selected
    - An delightful/undelightful button
      * Marks the selected demo as delightful/undelightful
      * Will show the opposite state to the demo current state. It reads undelightful if the demo is marked delightful and delightful if the demo is marked undelightful.
  + A run menu
    - See: Common Elements
  + A help menu
    - See: Common Elements
* A search bar
  + Refreshes the list when changed
* A list
  + Contains the results of the search
  + The selected item is displayed on the demo canvas
* A demo canvas
  + See: Common Elements

### Editor

The editor window consists of:

* A menu bar
  + A file menu
    - A save button
      * Saves the changes to the demo source code
    - An edit details button
      * Opens the modal detail editor dialog
    - A delete button
      * Deletes the demo and closes the editor window
  + A run menu
    - See: Common Elements
  + A help menu
    - See: Common Elements
* A code area
  + Area for user to write the demo source code
  + Displays the result on the demo canvas as the code changes
* A demo canvas
  + See: Common Elements

### Exit confirmation dialog

The exit confirmation dialog is shown when a close request occurs on an editor window with unsaved changes. In the dialog there are save, exit and cancel buttons. Save saves and closes the dialog and editor. Exit only closes the dialog and editor. Cancel only exits the dialog and not the editor.

### Login

The login dialog is used for the user to login by providing their username and password. It will only close if the login is successful, or the user cancels the dialog. Errors are displayed on the top label.

### Register

The register dialog is user for the user to create a new account. A username and password must be provided and be of sufficient length. The account cannot be registered if the input is not valid or there is an existing user with the same username. Server and validation errors are displayed on the top label.

### Detail Editor

The detail editor dialog is used to set the metadata of a demo. The title, tags and availability can be changed. The changes cannot be saved if the input is not valid or there is an existing demo with the same title. Server and validation errors are displayed on the top label

### Detail Viewer

The detail viewer dialog is used to view the metadata and statistics of a demo. The title, tags, creator, date of creation, number of visits and number of delights are shown.

### Common Elements

Elements present across multiple dialogs:

* Demo canvas
  + Compiles demo source code written in GLSL and displays the result
* Run menu
  + A play button
    - Play/Resume the demo animation
    - Enabled when demo animation is paused
  + A pause button
    - Pause the demo animation
    - Enabled when demo animation is playing
  + A reset button
    - Reset the animation to the beginning
* Help menu
  + A help contents button
    - Opens web browser page containing contextual help
  + An about button
    - Opens web browser page containing information about the program

## Specification of Help

Help is provided through the menu item **Help 🡪 Help contents**. When pressed, a web browser page containing information relevant to the specific window will open. Help contents are available on the browser and editor windows. The browser specific page instructs how to register an account, login to an account, create a demo, edit a demo and fork a demo. The editor specific page gives an overview of GLSL and instructs how to edit the details of a demo. An about page containing general information about the program and contact details is also available through **Help 🡪 About**.

## Specifications of Data Storage

## Hardware and Software requirements

### Server

#### Hardware

1GB RAM

5GB available disk space

#### Software

Java Runtime Environment 8

### Client

#### Hardware

1GB RAM

500MB available disk space

Graphics card compatible with OpenGL 2.0 or later

#### Software

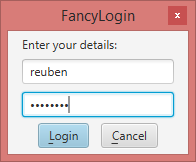
Java Runtime Environment 8

# System Design Document

## User Interface Design

The following interfaces were constructed using a combination of JavaFX Scene Builder 2.0 and editing the FXML files directly.

### Login

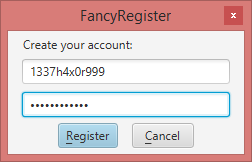


The FXML describing this GUI is available in the document at this section:

FancyShaderClient\src\fancyshader\client\gui\Login.fxml

|  |  |
| --- | --- |
| **XPath** | **Description** |
| \*[@fx:id=“topLabel”] | Displays initial message and further error messages |
| \*[@fx:id=“usernameField”][[1]](#footnote-1) | Field to enter username |
| \*[@fx:id=“passwordField”]1 | Field to enter password |
| \*[@fx:id=“loginButton”] | Attempts to login and close the dialog |
| \*[@fx:id=“cancelButton”] | Closes the dialog |

### Register

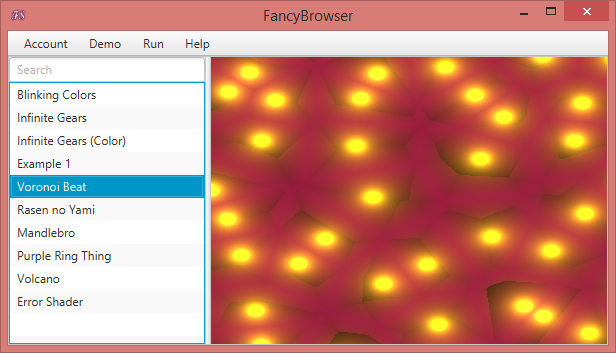


The FXML describing this GUI is available in the document at this section:

FancyShaderClient\src\fancyshader\client\gui\Register.fxml

|  |  |
| --- | --- |
| **XPath** | **Description** |
| \*[@fx:id=“topLabel”] | Displays initial message and further error messages |
| \*[@fx:id=“usernameField”]1 | Field to enter username |
| \*[@fx:id=“passwordField”]1 | Field to enter password |
| \*[@fx:id=“registerButton”] | Attempts to register and close the dialog |
| \*[@fx:id=“cancelButton”] | Closes the dialog |

### Browser

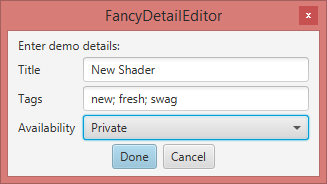


The FXML describing this GUI is available in the document at this section:

FancyShaderClient\src\fancyshader\client\gui\Browser.fxml

|  |  |
| --- | --- |
| **XPath** | **Description** |
| MenuBar | The menu bar |
| Menu[@text=”\_Account”] | The menu containing items pertaining to the user session and accounts |
| \*[@fx:id=“loginMenuItem”] | Opens a login dialog |
| \*[@fx:id=“registerMenuItem”] | Opens a register dialog |
| \*[@fx:id=“logoutMenuItem”] | Attempts to logout |
| Menu[@text=”\_Demo”] | The menu containing items pertaining to the selected demo or the creation of new demos |
| \*[@fx:id=“createMenuItem”] | Opens a demo creation dialog |
| \*[@fx:id=“editMenuItem”] | Opens an editor window for the selected demo in \*[@fx:id=”demoList”] |
| \*[@fx:id=“infoMenuItem”] | Opens a demo viewer dialog for the selected demo in [@fx:id=“demoList |
| \*[@fx:id=“delightfulMenuItem”] | Toggles the delightful status of the selected demo in \*[@fx:id=”demoList”] |
| Menu[@text=”\_Run”] | The menu containing items pertaining to the animation in the preview DemoViewer [@fx:id=“viewer |
| \*[@fx:id=“playMenuItem”] | Plays the animation of \*[@fx:id=”viewer”] |
| \*[@fx:id=“pauseMenuItem”] | Pauses the animation of \*[@fx:id=”viewer”] |
| \*[@fx:id=“resetMenuItem”] | Starts the animation of \*[@fx:id=”viewer”] from the beginning |
| Menu[@text=”\_Help”] | The menu containing items pertaining to the help |
| MenuItem[@text=”Help \_Contents”] | Opens the help contents page for the demo browser in the default web browser |
| MenuItem[@text=”\_About”] | Opens the about page for the program in the default web browser |
| \*[@fx:id=“searchField”] | The search bar |
| \*[@fx:id=“demoList”] | The result list containing the demos accessible to the user matching the search query |
| \*[@fx:id=“viewer”] | The DemoViewer displaying the currently selected demo in \*[@fx:id=”demoList”] |

### Detail Editor

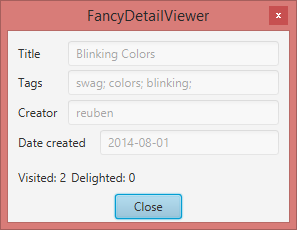


The FXML describing this GUI is available in the document at this section:

FancyShaderClient\src\fancyshader\client\gui\DetailEditor.fxml

|  |  |
| --- | --- |
| **XPath** | **Description** |
| \*[@fx:id=”topLabel”] | Displays initial message and further error messages |
| Label[@text=”Title”] | Label for \*[@fx:id=”titleField”] |
| \*[@fx:id=”titleField”] | Field to enter title |
| Label[@text=”Tags”] | Label for \*[@fx:id=”tagField”] |
| \*[@fx:id=”tagField”] | Field to enter semicolon delimited tags |
| Label[@text=”Availability”] | Label for \*[@fx:id=”availabilityBox”] |
| \*[@fx:id=”availabilityBox”] | Choice box to select availability |
| \*[@fx:id=”doneButton”] | Attempts to commit the changes and close the dialog |
| \*[@fx:id=”cancelButton”] | Closes the dialog without committing changes |

### Detail Viewer

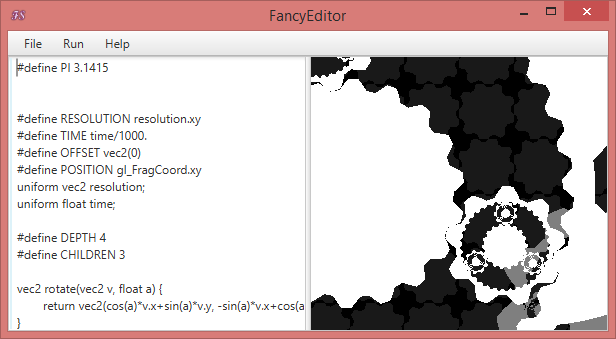


The FXML describing this GUI is available in the document at this section:

FancyShaderClient\src\fancyshader\client\gui\DetailViewer.fxml

|  |  |
| --- | --- |
| **XPath** | **Description** |
| Label[@text=”Title”] | Label for \*[@fx:id=”titleField”] |
| \*[@fx:id=”titleField”] | Field to display title |
| Label[@text=”Tags”] | Label for \*[@fx:id=”tagField”] |
| \*[@fx:id=”tagField”] | Field to display semicolon delimited tags |
| Label[@text=”Creator”] | Label for \*[@fx:id=”creatorField”] |
| \*[@fx:id=”creatorField”] | Field to display the creator |
| Label[@text=”Date created”] | Label for \*[@fx:id=”dateCreatedField”] |
| \*[@fx:id=”dateCreatedField”] | Field to display the date created |
| \*[@fx:id=”infoLabel”] | Label to display the number of users that have visited and the number of users that are delighted |
| \*[@fx:id=”closeButton”] | Closes the dialog |

### Editor

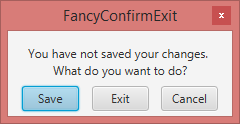


The FXML describing this GUI is available in the document at this section:

FancyShaderClient\src\fancyshader\client\gui\Editor.fxml

|  |  |
| --- | --- |
| **XPath** | **Description** |
| MenuBar | The menu bar |
| Menu[@text=”\_File”] | The menu containing items pertaining to the saving, modification and deletion of the demo |
| MenuItem[@text=”\_Save”] | Saves the changes made in the code area |
| MenuItem[@text=”\_Edit details”] |  |
| Menu[@text=”\_Run”] | The menu containing items pertaining to the animation in the preview DemoViewer #viewer |
| \*[@fx:id=”playMenuItem”] | Plays the animation of \*[@fx:id=”viewer”] |
| \*[@fx:id=”pauseMenuItem”] | Pauses the animation of \*[@fx:id=”viewer”] |
| \*[@fx:id=”resetMenuItem”] | Starts the animation of \*[@fx:id=”viewer”] from the beginning |
| Menu[@text=”\_Help] | The menu containing items pertaining to the help |
| MenuItem[@text=”Help \_Contents”] | Opens the help contents page for the demo browser in the default web browser |
| MenuItem[@text=”\_About”] | Opens the about page for the program in the default web browser |
| \*[@fx:id=”codeArea”] | The section to edit the GLSL code of the demo |
| \*[@fx:id=”viewer”] | The DemoViewer displaying the result of the code in \*[@fx:id=”codeArea”] |

### Exit confirmation



The FXML describing this GUI is available in the document at this section:

FancyShaderClient\src\fancyshader\client\gui\ConfirmExit.fxml

|  |  |
| --- | --- |
| **XPath** | **Description** |
| \*[@fx:id=”topLabel”] | Displays initial message and further error messages |
| \*[@fx:id=”saveButton”] | Attempts to commit the changes and close the dialog as well as the editor window |
| \*[@fx:id=”exitButton”] | Closes the dialog as well as the editor without committing changes |
| \*[@fx:id=”cancelButton”] | Closes the dialog without closing the editor |

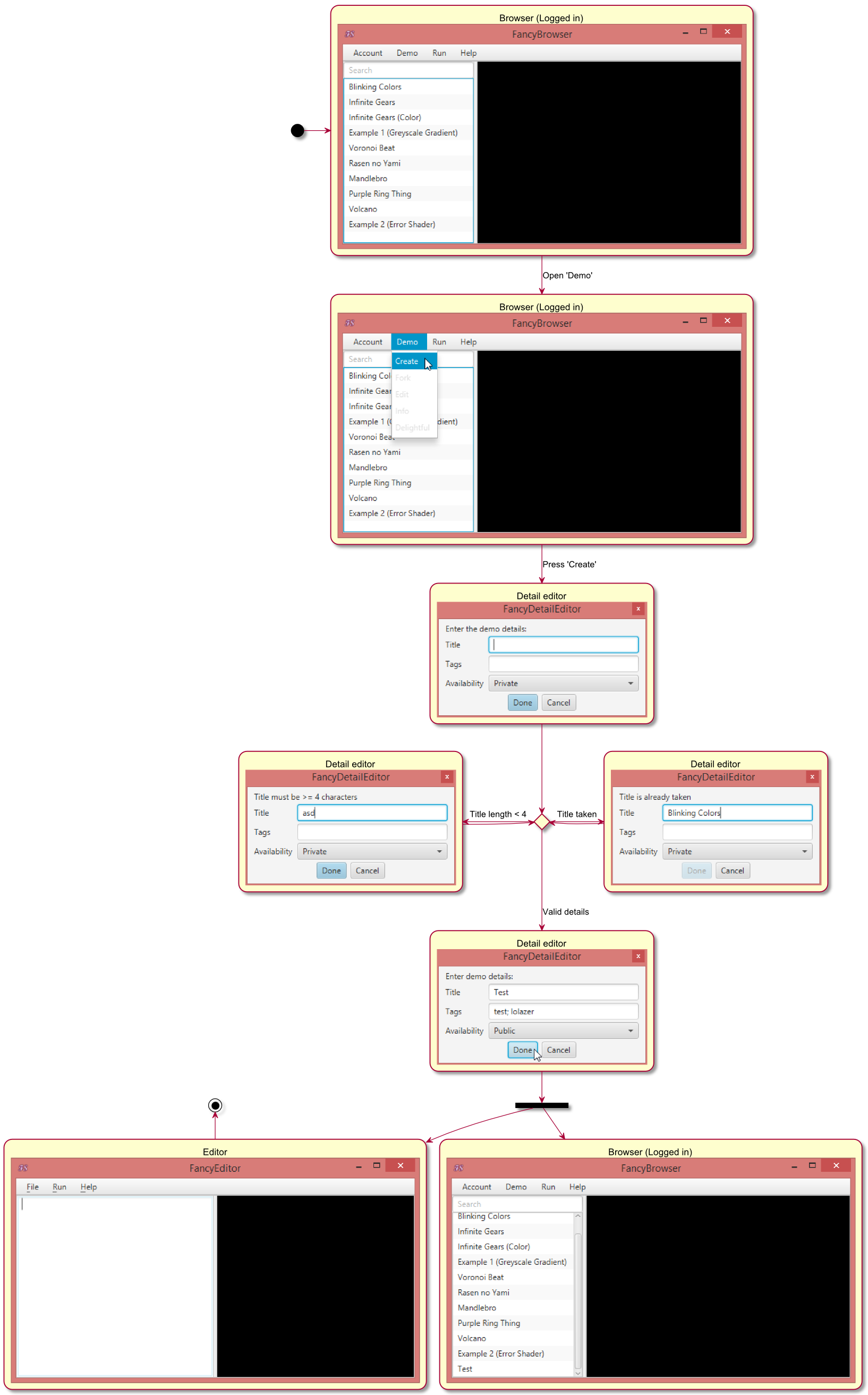
## Sequencing

### Client

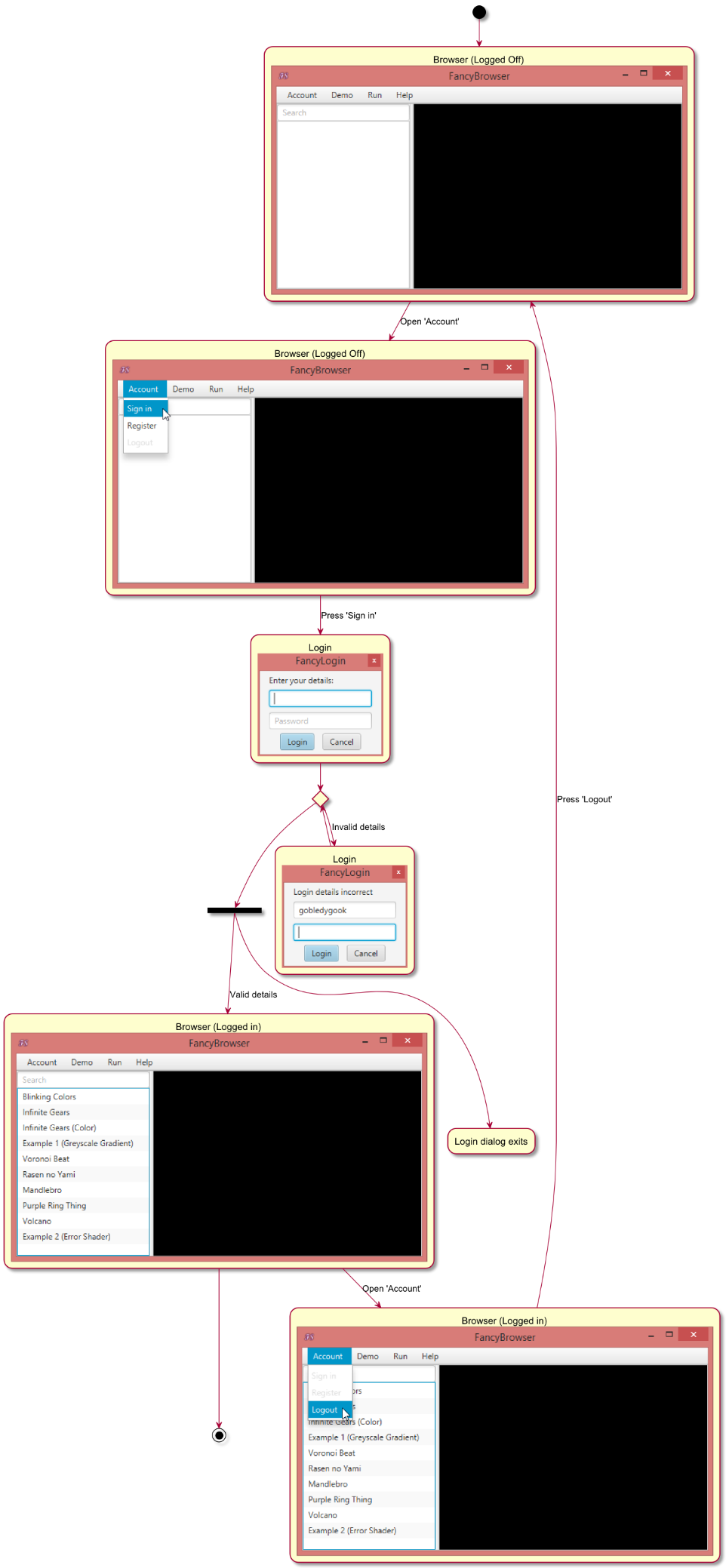
#### On startup

The client entry point is the **FancyShaderClient**#*main* method which in turn launches the JavaFX Application entry point **FancyShaderClient**#*start*. The method starts a splash screen. It then tries to fetch the server’s remote registry and obtains the **FancyShaderRemoteService**. The client then waits for 1 second to allow the splash screen to be seen. If the service was not retrieved, then the program exits with an error code. Otherwise a **MainController** instance is created with the retrieved service, and the **MainController**#*createBrowser* method is invoked. Further action takes place in JavaFX controller methods upon user interaction.

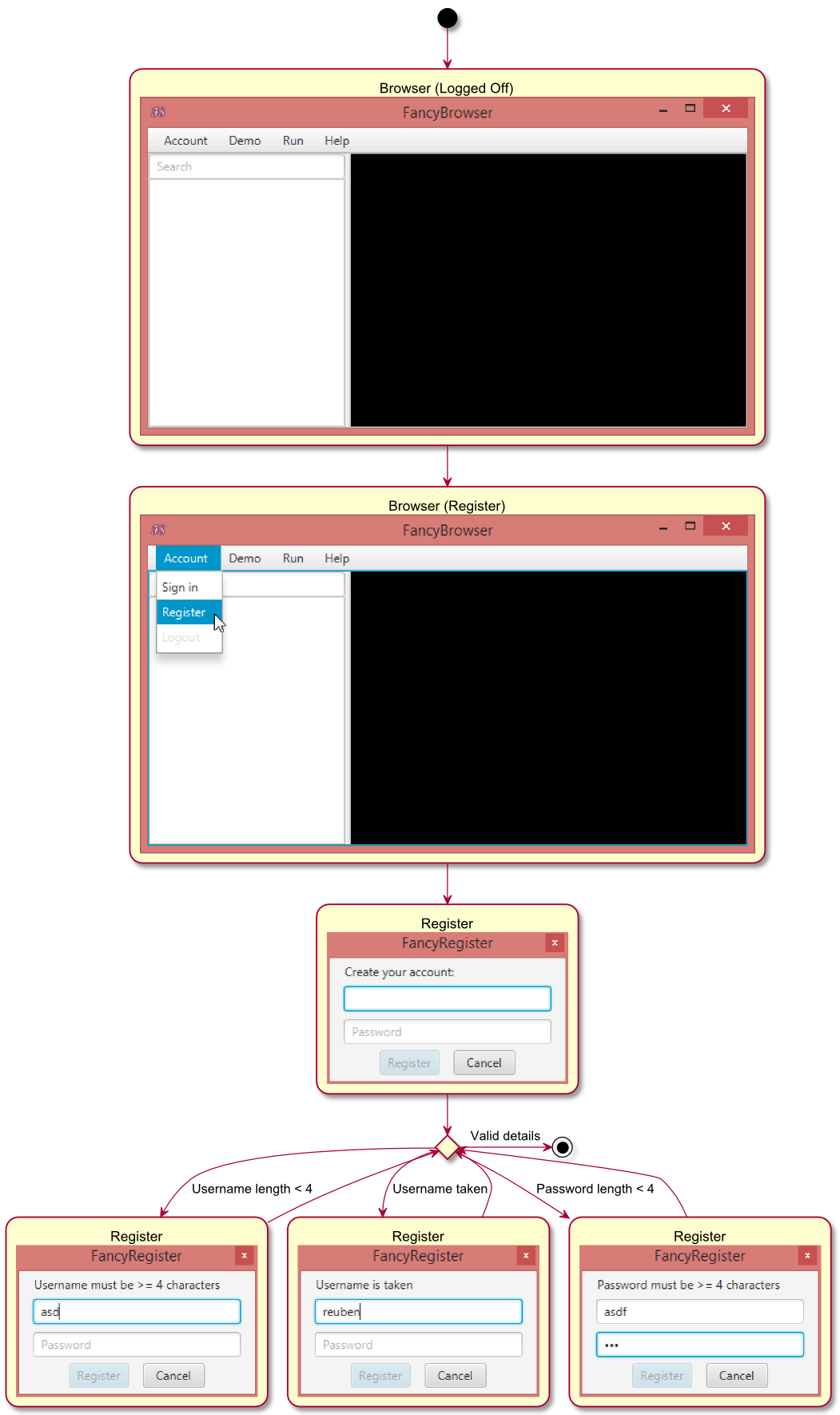
#### Creation of Demo



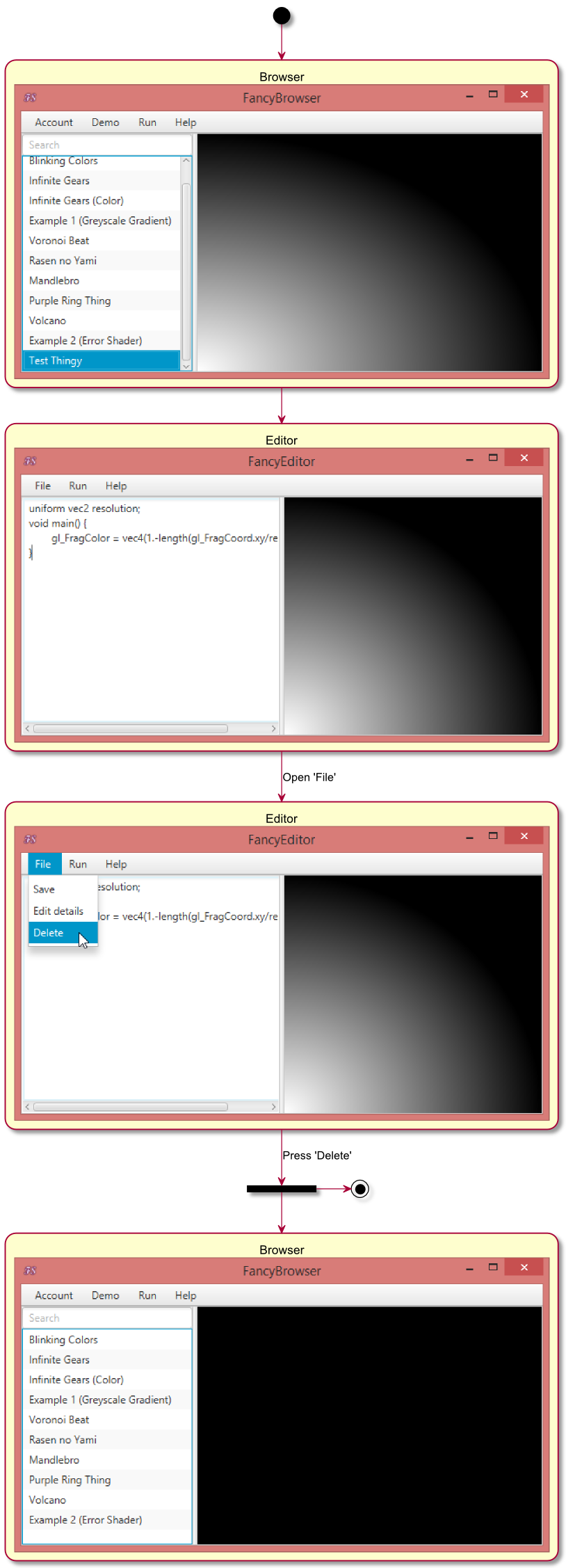
#### Sign in / Logout



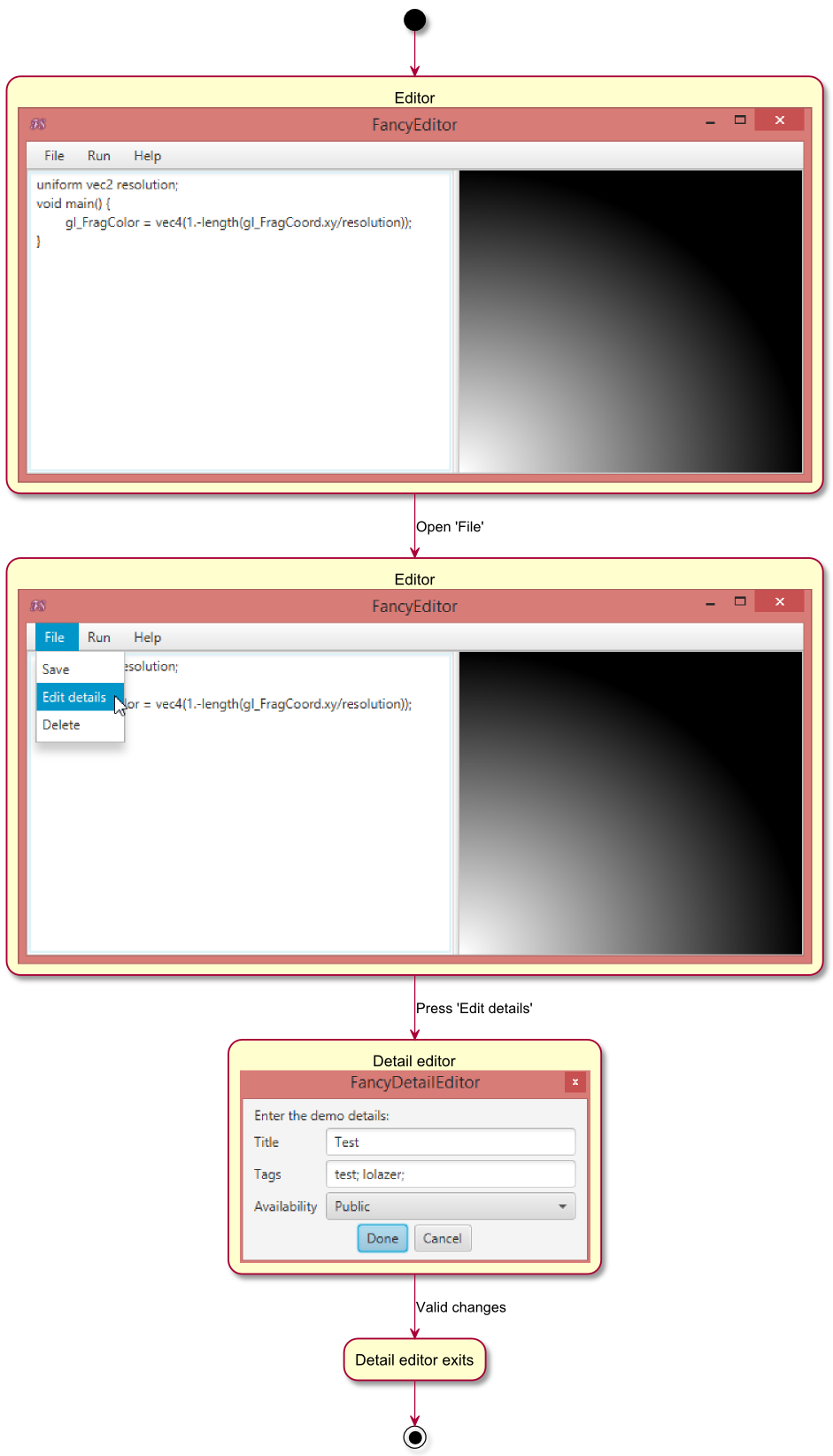
#### Register



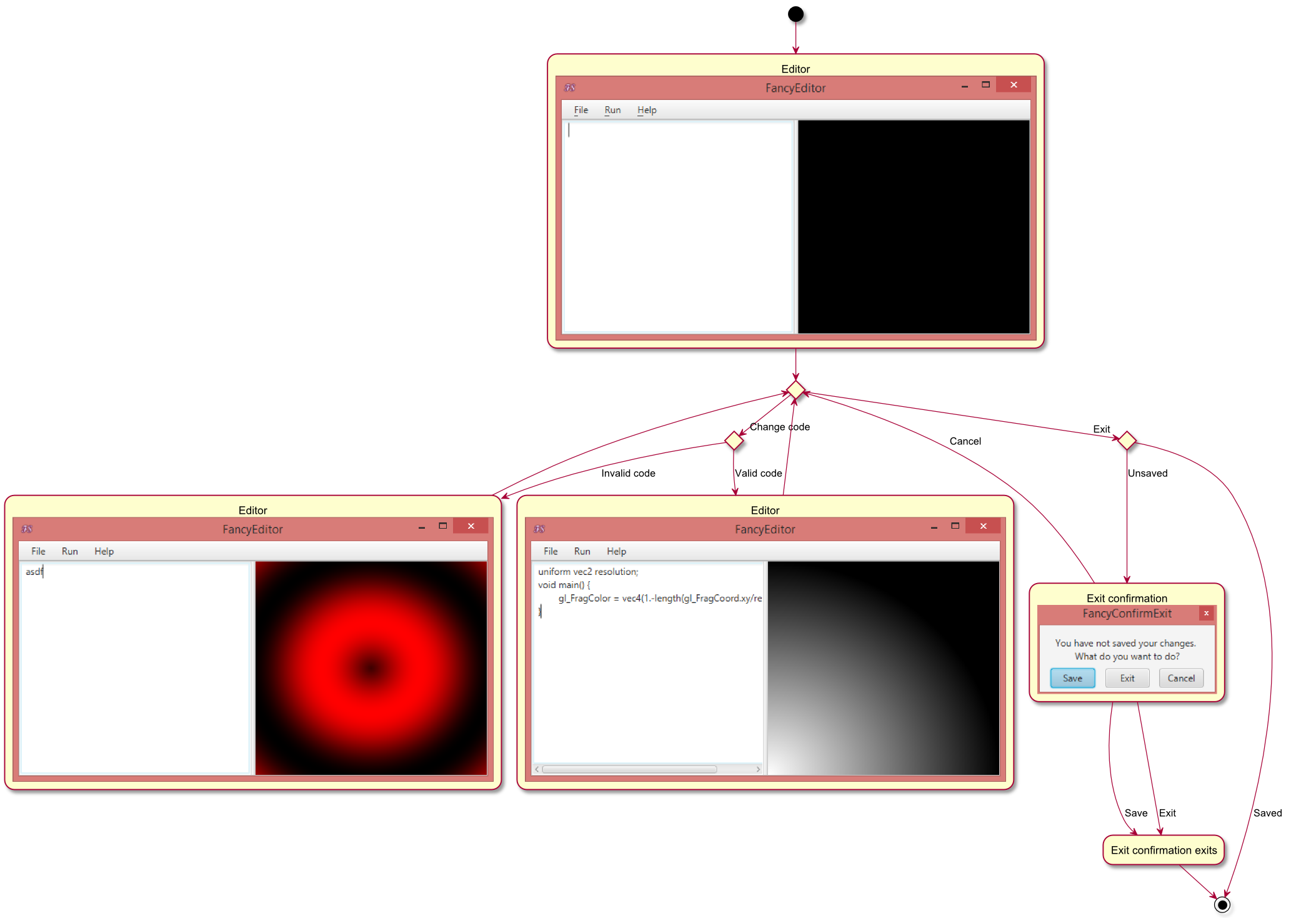
#### Deletion of demo



#### Edit details



#### Edit



### Server

#### On startup

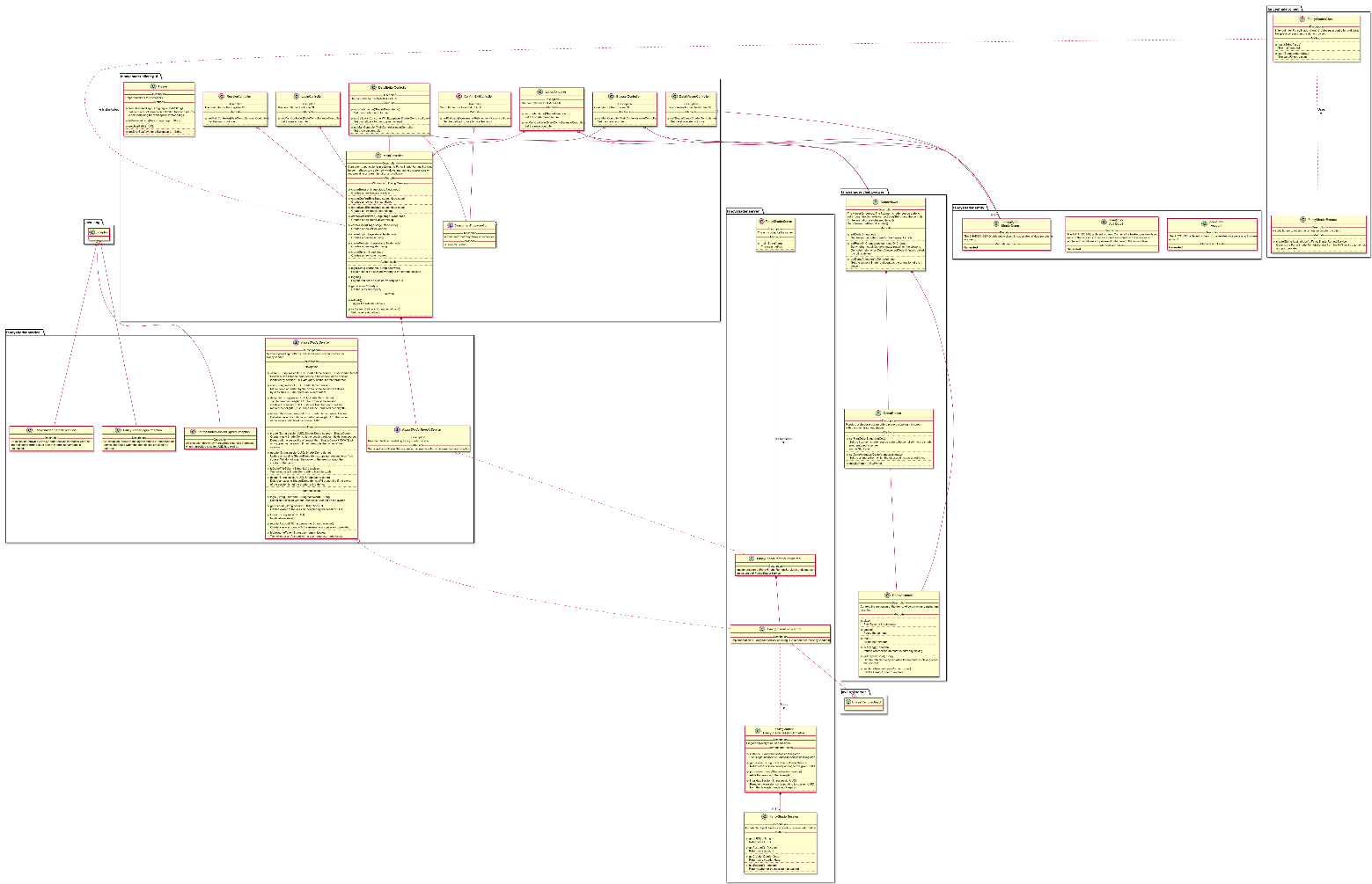
The server entry point is the **FancyShaderServer**#*main* method. The method creates a new instance of **FancyShaderServiceRemoteImpl** and binds it to the host machines RMI registry. Further action takes place upon remote invocation of any of the **FancyShaderServiceRemote** methods.

## Class Design

### Exclusions and the reasoning

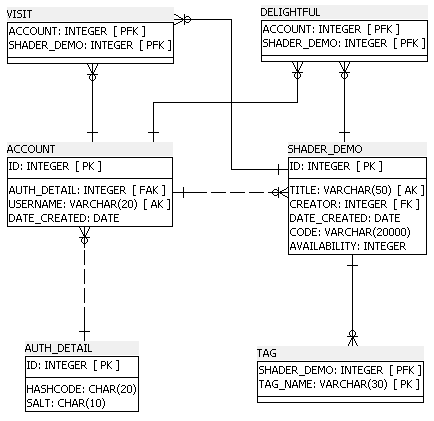
In the class design there are methods and fields which I have explicitly left out, specifically those with private attribute modifiers, those with @FXML or @Override annotations and those generated inside the entity classes from the databases. Private fields and methods are representative of the implementation used and do not reflect the class design. @FXML fields and methods are representative of the design of the FXML document that requires their presence and not the class design. @Override methods are implementing behaviours described previously and should not be restated. Fields and methods generated in entity classes are representative of the design of the database and not the class design.

### Class Diagram



In the case that this image is malformed, note that it is bundled with the documentation and also available [here](http://i.imgur.com/nzFpC0a.jpg)

## Persistent Storage Design



## Explanation of Storage Design

FancyShaderServer uses the Apache Derby SQL database to store all user data and creations.

### Why SQL

The relational model of SQL allows for easy establishment of relationships between tables and efficient joins. While it could be more efficient to use an alternate NoSQL database, it would have been complete overkill for the purposes of this program. With SQL it is straightforward to get the demos that a specific user liked, or the demos with a specific tag.

### Why Apache Derby Embedded

Apache Derby is a lightweight and portable database. There are no prerequisites for setting it up. As it is embedded, there is no need for a separate database server to be run alongside the FancyShaderServer.

### Why This Table Design

If you enumerate the objects you want (visits, delightfuls, accounts, shaders, tags), the relationships you want between them and follow common sense design patterns you will arrive at the architechure no matter what. The oddity being table AUTH\_DETAIL, which from a database point of view does not need to be separated from ACCOUNT. However, from a program point of view, these tables are represented entity objects, and it is not secure having user authorization details being attached automatically along-side more vanilla details such as their creation date.

# CODING and Technical Documentation

## Program Commenting

An example of commented code (FancyShaderServer\src\fancyshader\server\FancyShaderServiceImpl.java):

52 @Override

53 public String login(String username, String password) throws FancyShaderLoginException, FancyShaderInternalException {

54 try {

55 //Obtain the corresponding Account with the input username

56 TypedQuery<Account> accountQuery = entityManager.createNamedQuery("Account.findByUsername", Account.class)

57 .setParameter("username", username);

58 Account account = accountQuery.getSingleResult();

59

60 // The AuthDetail in Account is not fetched automatically by JPA to avoid scrubbing Account of the AuthDetail later on when

61 // exposing Account through the FancyShaderService. We do not want users to have free access to password hashes of other users,

62 // in the event that they chose easy to guess passwords.

63 AuthDetail detail = entityManager.find(AuthDetail.class, account.getAuthDetail());

64

65 String hash = getHash(password + detail.getSalt()); //Get the hash of the salted input password

66 if (hash.equals(detail.getHashcode())) { //Check if the hash of the salted input password is the same as the hash stored in the AuthDetail

67 return FancyShaderSession.create(account).getUUID(); //Create a session and return the UUID

68 } else {

69 throw new FancyShaderLoginException(); //The passwords do not match

70 }

71 } catch (NoResultException e) {

72 throw new FancyShaderLoginException(); //There is no Account with the input username

73 }

74 }

75

An example of javadocs is available at Generated Javadocs

## Separation of UI from working code

The UI layout is specified in FXML files. The files also reference a controller class. This controller class contains dependency injected properties and event handler methods. This allows the UI layout to be independent of the code. Logic present in controller classes is solely for validation.

The handling of GLSL compilation is done by the DemoViewer and animation logic by the DemoAnimator. DemoViewer is embedded as a component in the Browser and Editor FXML, and thus its logic is not present in the UI code, but its methods expose behaviours to allow the controllers to change the source code, and play, pause and reset the animation.

The client/server architecture further decouples the UI from working code, as all data persistence and querying occurs server-side, and the UI code only has to call the service methods.

## Inter-Code communication (Typed Methods and Parameters)

All Client/Server intercommunication is done through the Remote Method Invocation of the methods of the common interface **FancyShaderRemoteService**. The client invokes a method on a proxy class created from the server’s registry. The proxy class in turn marshals the call stack frame and sends it to the server. The server-side method is invoked with the parameters and the return frame is marshalled and sent back to the client. If communication fails than **RemoteException** is thrown.

For example (FancyShaderClient\src\fancyshader\client\gui\BrowserController.java):

163 @FXML

164 public void onCreatePressed(ActionEvent event) {

165 // Open a detail editor dialog

166 mainController.createDetailEditor(new Stage(), new ShaderDemo(), rootPane, (demo)->{

167 // When done

168

169 // Create a new demo using FancyShaderRemoteService

170 ShaderDemo createdDemo = mainController.getService().create(mainController.getSessionProperty().getValue(), demo);

171

172 // Trigger a refresh

173 mainController.refresh();

174

175 // Open an editor window for the new demo

176 mainController.createEditor(new Stage(), createdDemo, rootPane);

177 });

178 }

179

## General Techniques

Indentation present

Switch statements used where appropriate (FancyShaderClient\src\fancyshader\client\gui\EditorController.java):

73 // Open a confirm exit dialog if there is unsaved work, otherwise just exit

74 if(unsavedWork) {

75 mainController.createConfirmExit(new Stage(), rootPane, (result)->{

76 switch(result){

77 case SAVE:

78 save();

79 exitWindow();

80 break;

81 case EXIT:

82 exitWindow();

83 break;

84 default:

85 throw new AssertionError(result.name());

86

87 }

88 });

89 }

90 else exitWindow();

All variables are named with some indication of purpose

Common procedures factored out (FancyShaderClient\src\fancyshader\client\gui\DetailEditorController.java):

128 exitDialog();

and

141 exitDialog();

with

135 private void exitDialog() {

136 ((Stage) rootPane.getScene().getWindow()).close();

137 }

138

## Persistent storage/Querying

JPQL queries are located in the NamedQuery anonatations of the entity classes:

* FancyShaderLib\src\fancyshader\entity\Account.java
* FancyShaderLib\src\fancyshader\entity\AuthDetail.java
* FancyShaderLib\src\fancyshader\entity\ShaderDemo.java

They are invoked through the EntityManager. For example (FancyShaderServer\src\fancyshader\server\FancyShaderServiceImpl.java):

55 //Obtain the corresponding Account with the input username

56 TypedQuery<Account> accountQuery = entityManager.createNamedQuery("Account.findByUsername", Account.class)

57 .setParameter("username", username);

58 Account account = accountQuery.getSingleResult();

59

## Data validation, exception handling, error messages

All dialogs re-evaluate the validity of the input whenever the input fields are changed. If the input is invalid, the user cannot proceed. The dialog is updated using a method.

In the register dialog the method is as such:

61 public void updateDialog() {

62 // Validate the fields, update the top label to show the message and

63 // disable the register button if not valid

64 String text = TOP\_LABEL\_TEXT;

65 boolean valid = false;

66 try {

67 if (usernameField.getText().length() < 4) {

68 text = "Username must be >= 4 characters";

69 } else if (mainController != null && mainController.getService().isUsernameTaken(usernameField.getText())) {

70 text = "Username is taken";

71 } else if (passwordField.getText().length() < 4) {

72 text = "Password must be >= 4 characters";

73 } else valid = true;

74 topLabel.setText(text);

75 } catch (FancyShaderInternalException | RemoteException e) {

76 topLabel.setText(Helper.getErrorText(e));

77 Logger.getLogger(RegisterController.class.getName()).log(Level.WARNING, null, e);

78 }

79 registerButton.setDisable(!valid);

80 }

In the detail editor dialog it is as such:

84 private void updateDialog() {

85 // Validate the fields, update the top label to show the message and

86 // disable the done button if not valid

87 String text = TOP\_LABEL\_TEXT;

88 boolean doneEnabled = true;

89 try {

90 if (titleField.getText().length() < 4) {

91 text = "Title must be >= 4 characters";

92 }

93 if (!titleField.getText().equals(previousTitle) && mainController != null && mainController.getService().isDemoTitleTaken(titleField.getText())) {

94 text = "Title is already taken";

95 doneEnabled = false;

96 }

97 } catch (RemoteException | FancyShaderInternalException ex) {

98 text = Helper.getErrorText(ex);

99 Logger.getLogger(DetailEditorController.class.getName()).log(Level.SEVERE, null, ex);

100 doneEnabled = false;

101 }

102 topLabel.setText(text);

103 doneButton.setDisable(!doneEnabled);

104

105 }

106

## Fulfilment of Specifications

All specifications have been fulfilled

## User Experience report

The program has been compiled and packaged into jar files and a README.txt contains instructions on how to run the program on any platform that supports Java.

Note that FancyShaderClient is the executable that is meant to be distributed, while the server is intended to be run by the distributor. However for demonstration purposes the server and client are run on the same machine.

## Source Code

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\FancyShaderClient.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client;

18

19 import fancyshader.client.gui.MainController;

20 import fancyshader.service.FancyShaderRemoteService;

21 import java.util.Map;

22 import java.util.Timer;

23 import java.util.TimerTask;

24 import java.util.logging.Level;

25 import java.util.logging.Logger;

26 import javafx.application.Application;

27 import javafx.application.Platform;

28 import javafx.stage.Stage;

29

30 /\*\*

31 \* Entry point for FancyShader client. Creates main controller and starts the splash screen then the demo browser.

32 \* @author Reuben Steenekamp

33 \*/

34 public class FancyShaderClient extends Application {

35 @Override

36 public void start(Stage primaryStage){

37 FancyShaderRemoteService service = null;

38 Stage splash = new Stage();

39 MainController.createSplash(splash);

40 try {

41 Map<String,String> parameters = getParameters().getNamed();

42 String host = parameters.get("host");

43 int port = Integer.parseInt(parameters.get("port"));

44 service = FancyShaderRemote.create(host, port);

45 } catch (Exception e) {

46 Logger.getLogger(FancyShaderClient.class.getName()).log(Level.WARNING, null, e);

47 }

48 MainController mainController = new MainController(service);

49 Timer timer = new Timer();

50 final FancyShaderRemoteService serviceRef = service;

51 timer.schedule(new TimerTask(){

52 @Override

53 public void run() {

54 Platform.runLater(()->{

55 splash.close();

56 if(serviceRef == null)

57 System.exit(1);

58 else {

59 mainController.createBrowser(primaryStage, null);

60 primaryStage.onCloseRequestProperty().setValue((e)->{

61 System.exit(0);

62 });

63 }

64 });

65 }

66 }, 1000);

67 }

68

69 /\*\*

70 \* @param args the command line arguments

71 \*/

72 public static void main(String[] args) {

73 launch(args);

74 }

75

76 }

77

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\FancyShaderRemote.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client;

18

19 import fancyshader.service.FancyShaderRemoteService;

20 import java.rmi.Remote;

21 import java.rmi.registry.LocateRegistry;

22

23 /\*\*

24 \* A static library to obtain the remote service for the client

25 \* @author Reuben Steenekamp

26 \*/

27 public class FancyShaderRemote {

28 /\*\*

29 \* obtains the {@link fancyshader.service.FancyShaderRemoteService}

30 \* from the RMI registry at the {@code host} and {@code port} provided.

31 \* @param host the host of the RMI registry

32 \* @param port the port of the RMI registry

33 \* @return the remote service

34 \* @throws Exception

35 \*/

36 public static FancyShaderRemoteService create(String host, int port) throws Exception {

37 Remote remote = LocateRegistry.getRegistry(host, port).lookup("FancyShaderRemoteService");

38 return (FancyShaderRemoteService)remote;

39 }

40 }

41

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\Browser.fxml |

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

2 <!--

3 Copyright (C) 2014 Reuben Steenekamp

4

5 This program is free software: you can redistribute it and/or modify

6 it under the terms of the GNU General Public License as published by

7 the Free Software Foundation, either version 3 of the License, or

8 (at your option) any later version.

9

10 This program is distributed in the hope that it will be useful,

11 but WITHOUT ANY WARRANTY; without even the implied warranty of

12 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

13 GNU General Public License for more details.

14

15 You should have received a copy of the GNU General Public License

16 along with this program. If not, see <http://www.gnu.org/licenses/>.

17 -->

18

19 <?import java.lang.\*?>

20 <?import java.util.\*?>

21 <?import javafx.scene.\*?>

22 <?import javafx.scene.control.\*?>

23 <?import javafx.scene.layout.\*?>

24 <?import javafx.scene.canvas.\*?>

25 <?import javax.media.opengl.awt.\*?>

26

27 <BorderPane xmlns:fx="http://javafx.com/fxml/1" fx:id="rootPane" prefHeight="300.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/8" fx:controller="fancyshader.client.gui.BrowserController">

28 <center>

29 <SplitPane dividerPositions="0.3328" focusTraversable="true" prefHeight="312.0" prefWidth="422.0">

30 <items>

31 <BorderPane prefHeight="200.0" prefWidth="200.0">

32 <center>

33 <ListView fx:id="demoList" prefHeight="200.0" prefWidth="200.0" />

34 </center>

35 <top>

36 <TextField fx:id="searchField" prefWidth="200.0" promptText="Search" />

37 </top>

38 </BorderPane>

39 <BorderPane prefHeight="200.0" prefWidth="200.0">

40 <center>

41 <fancyshader.client.viewer.DemoViewer fx:id="viewer"/>

42 </center>

43 </BorderPane>

44 </items>

45 </SplitPane>

46 </center>

47 <top>

48 <MenuBar BorderPane.alignment="CENTER">

49 <menus>

50 <Menu text="\_Account">

51 <items>

52 <MenuItem fx:id="loginMenuItem" onAction="#onLoginPressed" text="\_Sign in" />

53 <MenuItem fx:id="registerMenuItem" onAction="#onRegisterPressed" text="\_Register"/>

54 <MenuItem fx:id="logoutMenuItem" onAction="#onLogoutPressed" text="\_Logout" />

55 </items>

56 </Menu>

57 <Menu text="\_Demo">

58 <items>

59 <MenuItem fx:id="createMenuItem" onAction="#onCreatePressed" text="\_Create" />

60 <MenuItem fx:id="forkMenuItem" onAction="#onForkPressed" text="\_Fork" />

61 <MenuItem fx:id="editMenuItem" onAction="#onEditPressed" text="\_Edit" />

62 <MenuItem fx:id="infoMenuItem" onAction="#onInfoPressed" text="\_Info" />

63 <MenuItem fx:id="delightfulMenuItem" onAction="#onDelightfulPressed" text="\_Delightful" />

64 </items>

65 </Menu>

66

67 <Menu text="\_Run">

68 <items>

69 <MenuItem fx:id="playMenuItem" onAction="#onPlayPressed" text="\_Start"/>

70 <MenuItem fx:id="pauseMenuItem" onAction="#onPausePressed" text="\_Pause"/>

71 <MenuItem fx:id="resetMenuItem" onAction="#onResetPressed" text="\_Reset"/>

72 </items>

73 </Menu>

74 <Menu text="\_Help">

75 <items>

76 <MenuItem fx:id="helpMenuItem" onAction="#onHelpContentsPressed" text="Help \_Contents" />

77 <MenuItem fx:id="aboutMenuItem" text="\_About" onAction="#onAboutPressed" />

78 </items>

79 </Menu>

80 </menus>

81 </MenuBar>

82 </top>

83 </BorderPane>

84

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\BrowserController.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.gui;

18

19 import fancyshader.client.viewer.DemoViewer;

20 import fancyshader.entity.ShaderDemo;

21 import fancyshader.service.FancyShaderSessionExpiredException;

22 import java.awt.Desktop;

23 import java.io.IOException;

24 import java.net.URI;

25 import java.net.URISyntaxException;

26 import java.rmi.RemoteException;

27 import java.util.List;

28 import java.util.logging.Level;

29 import java.util.logging.Logger;

30 import javafx.beans.property.ReadOnlyObjectProperty;

31 import javafx.collections.ObservableList;

32 import javafx.event.ActionEvent;

33 import javafx.fxml.FXML;

34 import javafx.scene.control.ListView;

35 import javafx.scene.control.MenuItem;

36 import javafx.scene.control.TextField;

37 import javafx.scene.layout.Pane;

38 import javafx.stage.Stage;

39

40 /\*\*

41 \* The controller for the Browser GUI

42 \* @author Reuben Steenekamp

43 \*/

44 public class BrowserController {

45 private MainController mainController;

46 /\*\*

47 \* Set the main controller

48 \* @param mainController the main controller to set

49 \*/

50 public void setMainController(MainController mainController) {

51 this.mainController = mainController;

52 Runnable callback = ()->{

53 updateDemoList();

54 };

55 mainController.setRefreshCallback(callback);

56

57 selectedItemProperty.addListener((observable, oldValue, newValue) -> { updateSelected(); });

58 updateSelected(); // Initial update

59 }

60

61 private ObservableList<ShaderDemo> demos;

62

63 @FXML private Pane rootPane;

64 @FXML private TextField searchField;

65

66 @FXML private ListView<ShaderDemo> demoList;

67 private ReadOnlyObjectProperty<ShaderDemo> selectedItemProperty;

68

69 @FXML private DemoViewer viewer;

70

71 public void initialize() {

72 demos = demoList.getItems();

73

74 searchField.textProperty().addListener((observable, oldValue, newValue)-> {

75 // On search field change

76

77 updateDemoList();

78 });

79

80 selectedItemProperty = demoList.getSelectionModel().selectedItemProperty();

81

82 // Start the shader from the beginning when it changes, as some shaders may consist of one time events

83 viewer.setResetOnChange(true);

84

85 // Set the viewer to play initially

86 viewer.getDemoAnimator().play();

87 }

88

89 public void updateDemoList() {

90 // Remove previous items

91 demos.clear();

92

93 // Check if there is a user session

94 if(mainController!=null && mainController.getSessionProperty().getValue()!=null) {

95 try {

96 // Fetch the demos using FancyShsaderRemoteService

97 List<ShaderDemo> resultDemos = mainController.getService().search(mainController.getSessionProperty().getValue(), searchField.getText());

98 // Add to the observable list

99 // will automatically update demoList`

100 demos.addAll(resultDemos);

101 } catch (FancyShaderSessionExpiredException | RemoteException ex) {

102 // If the fetching fails, then the list will be empty as it had been cleared.

103 // This is the behaviour that we want

104 Logger.getLogger(BrowserController.class.getName()).log(Level.WARNING, null, ex);

105 }

106 }

107

108 //Since the items in the list have changed, the selection defaults to null and must be updated

109 updateSelected();

110 }

111

112 public void updateSelected() {

113 ShaderDemo selected = selectedItemProperty.getValue();

114 if(selected != null) {

115 try {

116 mainController.getService().visit(mainController.getSessionProperty().getValue(), selectedItemProperty.getValue());

117 } catch (FancyShaderSessionExpiredException | RemoteException ex) {

118 Logger.getLogger(BrowserController.class.getName()).log(Level.SEVERE, null, ex);

119 }

120 }

121

122 // Set the viewer's code to the selected demo's.

123 // If the selected is null, then the code is null

124 viewer.setCode(selected == null ? null : selected.getCode());

125

126 // Menu items change state when the selected item is changed

127 updateMenuItems();

128 }

129

130 @FXML private MenuItem loginMenuItem;

131 @FXML private MenuItem registerMenuItem;

132 @FXML private MenuItem logoutMenuItem;

133 @FXML private MenuItem createMenuItem;

134 @FXML private MenuItem forkMenuItem;

135 @FXML private MenuItem editMenuItem;

136 @FXML private MenuItem infoMenuItem;

137 @FXML private MenuItem delightfulMenuItem;

138 @FXML private MenuItem playMenuItem;

139 @FXML private MenuItem pauseMenuItem;

140

141 @FXML

142 public void onLoginPressed(ActionEvent event) {

143 // Open a login dialog

144 mainController.createLogin(new Stage(), rootPane);

145 }

146

147 @FXML

148 public void onRegisterPressed(ActionEvent event) {

149 // Open a register dialog

150 mainController.createRegister(new Stage(), rootPane);

151 }

152

153 @FXML

154 public void onLogoutPressed(ActionEvent event) {

155 // Attempt to logout

156 try {

157 mainController.logout();

158 } catch (RemoteException ex) {

159 Logger.getLogger(BrowserController.class.getName()).log(Level.SEVERE, null, ex);

160 }

161 }

162

163 @FXML

164 public void onCreatePressed(ActionEvent event) {

165 // Open a detail editor dialog

166 mainController.createDetailEditor(new Stage(), new ShaderDemo(), rootPane, (demo)->{

167 // When done

168

169 // Create a new demo using FancyShaderRemoteService

170 ShaderDemo createdDemo = mainController.getService().create(mainController.getSessionProperty().getValue(), demo);

171

172 // Trigger a refresh

173 mainController.refresh();

174

175 // Open an editor window for the new demo

176 mainController.createEditor(new Stage(), createdDemo, rootPane);

177 });

178 }

179

180 @FXML

181 public void onForkPressed(ActionEvent event) {

182 // Get the selected value which is guaranteed to be non-null

183 // As the fork menu button would not be enabled otherwise

184 ShaderDemo selected = selectedItemProperty.getValue();

185

186 String code = selected.getCode();

187

188 mainController.createDetailEditor(new Stage(), selected, rootPane, (demo)->{

189 // When done

190

191 // Set the code of the demo

192 demo.setCode(code);

193

194 // Create a new demo using the FancyShaderRemoteService

195 ShaderDemo forkedDemo = mainController.getService().create(mainController.getSessionProperty().getValue(), demo);

196

197 // Trigger a refresh

198 mainController.refresh();

199

200 // Open an editor window for the new demo

201 mainController.createEditor(new Stage(), forkedDemo, rootPane);

202 });

203 }

204

205 @FXML

206 public void onInfoPressed(ActionEvent e) {

207 // Get the selected value which is guaranteed to be non-null

208 // As the info menu button would not be enabled otherwise

209 ShaderDemo selected = selectedItemProperty.getValue();

210

211 // Open a detail viewer dialog for the selected demo

212 mainController.createDetailViewer(new Stage(), selected, viewer);

213 }

214

215 @FXML

216 public void onDelightfulPressed(ActionEvent e) {

217 try {

218 // Get the selected value which is guaranteed to be non-null

219 // As the delightful menu button would not be enabled otherwise

220 ShaderDemo selected = selectedItemProperty.getValue();

221

222 String sessionUUID = mainController.getSessionProperty().getValue();

223

224 // Toggle the delightful status using the FancyShaderRemoteService

225 mainController.getService().delightful(sessionUUID, selected);

226

227 // The text of the delightful menu item has changed as it had toggled

228 // so the menu items need to be updated

229 updateMenuItems();

230 } catch (FancyShaderSessionExpiredException | RemoteException ex) {

231 Logger.getLogger(BrowserController.class.getName()).log(Level.SEVERE, null, ex);

232 }

233 }

234

235 @FXML

236 public void onEditPressed(ActionEvent event) {

237 // Open an editor window for the selected item which is guaranteed to be non-null

238 // As the edit menu button would not be enabled otherwise

239 mainController.createEditor(new Stage(), selectedItemProperty.getValue(), rootPane);

240 }

241

242

243 @FXML

244 public void onPlayPressed(ActionEvent e) {

245 viewer.getDemoAnimator().play();

246

247 // The play menu item is disabled when the viewer is playing

248 // so the menu items need to be updated

249 updateMenuItems();

250 }

251

252 @FXML

253 public void onPausePressed(ActionEvent e) {

254 viewer.getDemoAnimator().pause();

255

256

257 // The pause menu item is disabled when the viewer is paused

258 // so the menu items need to be updated

259 updateMenuItems();

260 }

261

262 @FXML

263 public void onResetPressed(ActionEvent e) {

264 viewer.getDemoAnimator().reset();

265 }

266

267 @FXML

268 public void onAboutPressed(ActionEvent event) {

269 // Open the about page in the default browser

270 if(Desktop.isDesktopSupported()) {

271 try {

272 Desktop.getDesktop().browse(new URI("\"file://"+Helper.getUserPath().getPath().replace(" ", "%20")+"/resources/about.html\""));

273 } catch (URISyntaxException | IOException ex) {

274 Logger.getLogger(BrowserController.class.getName()).log(Level.WARNING, null, ex);

275 }

276 }

277 }

278

279 @FXML

280 public void onHelpContentsPressed(ActionEvent event) {

281 // Open the help page in the default browser

282 if(Desktop.isDesktopSupported()) {

283 try {

284 Desktop.getDesktop().browse(new URI("file://"+Helper.getUserPath().getPath().replace(" ", "%20")+"/resources/browser.html"));

285 } catch (URISyntaxException | IOException ex) {

286 Logger.getLogger(BrowserController.class.getName()).log(Level.WARNING, null, ex);

287 }

288 }

289

290 }

291

292

293

294 public void updateMenuItems() {

295 boolean loggedIn = mainController != null && mainController.getSessionProperty().getValue()!=null;

296 boolean selectedItem = loggedIn && selectedItemProperty != null && selectedItemProperty.getValue() != null;

297 String sessionUUID = mainController.getSessionProperty().getValue();

298

299 // Determine whether the user is the creator of the selected item using FancyShaderRemoteService

300 boolean ownerSelectedItem = false;

301 try {

302 ownerSelectedItem = selectedItem && mainController.getService().getAccount(sessionUUID).equals(selectedItemProperty.getValue().getCreator());

303 } catch (RemoteException | FancyShaderSessionExpiredException e) {

304 Logger.getLogger(BrowserController.class.getName()).log(Level.WARNING, null, e);

305 }

306

307 // Determine whether the user is delighted by the selected item using FancyShaderRemoteService

308 boolean delightedSelectedItem = false;

309 try {

310 delightedSelectedItem = selectedItem && mainController.getService().isDelighted(sessionUUID, selectedItemProperty.getValue());

311 } catch (FancyShaderSessionExpiredException | RemoteException ex) {

312 Logger.getLogger(BrowserController.class.getName()).log(Level.SEVERE, null, ex);

313 }

314

315 loginMenuItem.setDisable(loggedIn);

316 registerMenuItem.setDisable(loggedIn);

317 logoutMenuItem.setDisable(!loggedIn);

318 createMenuItem.setDisable(!loggedIn);

319 playMenuItem.setDisable(viewer.getDemoAnimator().isPlaying());

320 pauseMenuItem.setDisable(!viewer.getDemoAnimator().isPlaying());

321 editMenuItem.setDisable(!ownerSelectedItem);

322 forkMenuItem.setDisable(!selectedItem);

323 infoMenuItem.setDisable(!selectedItem);

324 delightfulMenuItem.setDisable(!selectedItem);

325 delightfulMenuItem.setText(delightedSelectedItem ? "Undelightful" : "Delightful");

326

327 }

328

329

330 }

331

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\ConfirmExit.fxml |

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

2

3 <!--

4 Copyright (C) 2014 Reuben Steenekamp

5

6 This program is free software: you can redistribute it and/or modify

7 it under the terms of the GNU General Public License as published by

8 the Free Software Foundation, either version 3 of the License, or

9 (at your option) any later version.

10

11 This program is distributed in the hope that it will be useful,

12 but WITHOUT ANY WARRANTY; without even the implied warranty of

13 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

14 GNU General Public License for more details.

15

16 You should have received a copy of the GNU General Public License

17 along with this program. If not, see <http://www.gnu.org/licenses/>.

18 -->

19

20 <?import java.lang.\*?>

21 <?import java.util.\*?>

22 <?import javafx.scene.\*?>

23 <?import javafx.scene.control.\*?>

24 <?import javafx.scene.layout.\*?>

25

26 <AnchorPane id="AnchorPane" fx:id="rootPane" prefHeight="85.0" prefWidth="224.0" xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1" fx:controller="fancyshader.client.gui.ConfirmExitController">

27 <children>

28 <Label fx:id="topLabel" layoutX="15.0" layoutY="5.0" prefHeight="51.0" prefWidth="196.0" text="You have not saved your changes. What do you want to do?" textAlignment="CENTER" wrapText="true" AnchorPane.leftAnchor="14.0" AnchorPane.rightAnchor="14.0" />

29 <Button layoutX="84.0" layoutY="55.0" mnemonicParsing="true" onAction="#onExitPressed" prefHeight="25.0" prefWidth="57.0" text="E\_xit" />

30 <Button cancelButton="true" layoutX="153.0" layoutY="55.0" mnemonicParsing="true" onAction="#onCancelPressed" prefHeight="25.0" prefWidth="57.0" text="\_Cancel" />

31 <Button defaultButton="true" layoutX="14.0" layoutY="55.0" mnemonicParsing="true" onAction="#onSavePressed" prefHeight="25.0" prefWidth="57.0" text="\_Save" />

32 </children>

33 </AnchorPane>

34

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\ConfirmExitController.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17

18 package fancyshader.client.gui;

19

20 import java.util.logging.Level;

21 import java.util.logging.Logger;

22 import javafx.event.ActionEvent;

23 import javafx.fxml.FXML;

24 import javafx.scene.control.Label;

25 import javafx.scene.layout.Pane;

26 import javafx.stage.Stage;

27

28 /\*\*

29 \* The controller for the ConfirmExit GUI.

30 \* @author Reuben Steenekamp

31 \*/

32 public class ConfirmExitController {

33 @FXML private Pane rootPane;

34 @FXML private Label topLabel;

35 public static enum Result {

36 SAVE,

37 EXIT

38 }

39 private ConsumerWithException<Result> callback;

40 /\*\*

41 \* Set the callback to use to give the result

42 \* @param callback the callback to use

43 \*/

44 public void setCallback(ConsumerWithException<Result> callback) {

45 this.callback = callback;

46 }

47

48 private void doCallback(Result result) {

49 // Try and perform the callback

50 // show the error if an exception occurs

51 try {

52 callback.accept(result);

53 exitDialog();

54 } catch (Exception ex) {

55 topLabel.setText(Helper.getErrorText(ex));

56 Logger.getLogger(ConfirmExitController.class.getName()).log(Level.SEVERE, null, ex);

57 }

58 }

59 private void exitDialog() {

60 ((Stage)rootPane.getScene().getWindow()).close();

61 }

62 @FXML

63 public void onSavePressed(ActionEvent e) {

64 doCallback(Result.SAVE);

65 }

66

67 @FXML

68 public void onExitPressed(ActionEvent e) {

69 doCallback(Result.EXIT);

70 }

71

72 @FXML

73 public void onCancelPressed(ActionEvent e) {

74 exitDialog();

75 }

76 }

77

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\ConsumerWithException.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17

18 package fancyshader.client.gui;

19

20 /\*\*

21 \* A consumer that may throw an exception

22 \* @author Reuben Steenekamp

23 \*/

24 public interface ConsumerWithException<T> {

25 public void accept(T value) throws Exception;

26 }

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\DetailEditor.fxml |

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

2 <!--

3 Copyright (C) 2014 Reuben Steenekamp

4

5 This program is free software: you can redistribute it and/or modify

6 it under the terms of the GNU General Public License as published by

7 the Free Software Foundation, either version 3 of the License, or

8 (at your option) any later version.

9

10 This program is distributed in the hope that it will be useful,

11 but WITHOUT ANY WARRANTY; without even the implied warranty of

12 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

13 GNU General Public License for more details.

14

15 You should have received a copy of the GNU General Public License

16 along with this program. If not, see <http://www.gnu.org/licenses/>.

17 -->

18

19 <?import java.lang.\*?>

20 <?import java.util.\*?>

21 <?import javafx.scene.\*?>

22 <?import javafx.scene.control.\*?>

23 <?import javafx.scene.layout.\*?>

24

25 <AnchorPane fx:id="rootPane" prefHeight="145.0" prefWidth="311.0" xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1" fx:controller="fancyshader.client.gui.DetailEditorController">

26 <children>

27 <TextField onKeyReleased="#onTitleFieldChanged" fx:id="titleField" layoutX="75.0" layoutY="26.0" prefHeight="25.0" prefWidth="208.0" AnchorPane.leftAnchor="75.0" AnchorPane.rightAnchor="10.0" />

28 <TextField fx:id="tagField" layoutX="75.0" layoutY="55.0" prefHeight="25.0" prefWidth="208.0" AnchorPane.leftAnchor="75.0" AnchorPane.rightAnchor="10.0" />

29 <ChoiceBox fx:id="availabilityBox" layoutX="75.0" layoutY="84.0" prefHeight="25.0" prefWidth="208.0" AnchorPane.leftAnchor="75.0" AnchorPane.rightAnchor="10.0" />

30 <Label layoutX="10.0" layoutY="30.0" text="Title" AnchorPane.leftAnchor="10.0" />

31 <Label layoutX="10.0" layoutY="59.0" prefHeight="17.0" prefWidth="34.0" text="Tags" AnchorPane.leftAnchor="10.0" />

32 <Label layoutX="10.0" layoutY="88.0" text="Availability" AnchorPane.leftAnchor="10.0" />

33 <Button fx:id="doneButton" defaultButton="true" layoutX="104.0" layoutY="113.0" onAction="#onDonePressed" text="Done" />

34 <Button cancelButton="true" layoutX="155.0" layoutY="113.0" onAction="#onCancelPressed" text="Cancel" />

35 <Label fx:id="topLabel" layoutX="13.0" layoutY="6.0" text="Enter the demo details:" AnchorPane.leftAnchor="10.0" AnchorPane.rightAnchor="10.0" />

36 </children>

37 </AnchorPane>

38

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\DetailEditorController.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.gui;

18

19 import fancyshader.entity.ShaderDemo;

20 import fancyshader.service.FancyShaderInternalException;

21 import java.rmi.RemoteException;

22 import java.util.logging.Level;

23 import java.util.logging.Logger;

24 import javafx.event.ActionEvent;

25 import javafx.fxml.FXML;

26 import javafx.scene.control.Button;

27 import javafx.scene.control.ChoiceBox;

28 import javafx.scene.control.Label;

29 import javafx.scene.control.TextField;

30 import javafx.scene.input.KeyEvent;

31 import javafx.scene.layout.Pane;

32 import javafx.stage.Stage;

33

34 /\*\*

35 \* The controller for the DetailEditor GUI

36 \*

37 \* @author Reuben Steenekamp

38 \*/

39 public class DetailEditorController {

40

41 @FXML

42 private Pane rootPane;

43 @FXML

44 private Label topLabel;

45 @FXML

46 private TextField titleField;

47 @FXML

48 private TextField tagField;

49 @FXML

50 private ChoiceBox<String> availabilityBox;

51 @FXML

52 private Button doneButton;

53

54 private String previousTitle = null;

55 private ConsumerWithException<ShaderDemo> callback;

56 private MainController mainController;

57

58 public void initialize() {

59 availabilityBox.getItems().addAll("Private", "Public");

60 }

61

62 /\*\*

63 \* Set the shader demo to use

64 \* @param demo the shader demo to use

65 \*/

66 public void setShaderDemo(ShaderDemo demo) {

67 previousTitle = demo.getTitle();

68 titleField.setText(previousTitle == null ? "" : previousTitle);

69 String joinedTags = Helper.toDelimitedTags(demo.getTags());

70 tagField.setText(joinedTags);

71 availabilityBox.getSelectionModel().select(demo.getAvailability());

72 }

73

74 /\*\*

75 \* Set the callback to use to give the result

76 \* @param callback the callback to use

77 \*/

78 public void setCallback(ConsumerWithException<ShaderDemo> callback) {

79 this.callback = callback;

80 }

81

82 public static final String TOP\_LABEL\_TEXT = "Enter demo details:";

83

84 private void updateDialog() {

85 // Validate the fields, update the top label to show the message and

86 // disable the done button if not valid

87 String text = TOP\_LABEL\_TEXT;

88 boolean doneEnabled = true;

89 try {

90 if (titleField.getText().length() < 4) {

91 text = "Title must be >= 4 characters";

92 }

93 if (!titleField.getText().equals(previousTitle) && mainController != null && mainController.getService().isDemoTitleTaken(titleField.getText())) {

94 text = "Title is already taken";

95 doneEnabled = false;

96 }

97 } catch (RemoteException | FancyShaderInternalException ex) {

98 text = Helper.getErrorText(ex);

99 Logger.getLogger(DetailEditorController.class.getName()).log(Level.SEVERE, null, ex);

100 doneEnabled = false;

101 }

102 topLabel.setText(text);

103 doneButton.setDisable(!doneEnabled);

104

105 }

106

107 @FXML

108 public void onTitleFieldChanged(KeyEvent e) {

109 updateDialog();

110 }

111

112 @FXML

113 public void onDonePressed(ActionEvent e) {

114 String title = titleField.getText();

115 ShaderDemo demo = new ShaderDemo();

116 int availability = availabilityBox.getSelectionModel().getSelectedIndex();

117 demo.setTitle(title);

118 demo.setTags(Helper.fromDelimitedTags(tagField.getText()));

119 demo.setAvailability(availability);

120 doCallback(demo);

121 }

122

123 private void doCallback(ShaderDemo demo) {

124 // Try and perform the callback

125 // show the error if an exception occurs

126 try {

127 callback.accept(demo);

128 exitDialog();

129 } catch (Exception ex) {

130 topLabel.setText(Helper.getErrorText(ex));

131 Logger.getLogger(DetailEditorController.class.getName()).log(Level.SEVERE, null, ex);

132 }

133 }

134

135 private void exitDialog() {

136 ((Stage) rootPane.getScene().getWindow()).close();

137 }

138

139 @FXML

140 public void onCancelPressed(ActionEvent e) {

141 exitDialog();

142 }

143

144 /\*\*

145 \* Set the main controller

146 \* @param mainController the main controller to set

147 \*/

148 public void setMainController(MainController mainController) {

149 this.mainController = mainController;

150 }

151 }

152

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\DetailViewer.fxml |

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

2 <!--

3 Copyright (C) 2014 Reuben Steenekamp

4

5 This program is free software: you can redistribute it and/or modify

6 it under the terms of the GNU General Public License as published by

7 the Free Software Foundation, either version 3 of the License, or

8 (at your option) any later version.

9

10 This program is distributed in the hope that it will be useful,

11 but WITHOUT ANY WARRANTY; without even the implied warranty of

12 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

13 GNU General Public License for more details.

14

15 You should have received a copy of the GNU General Public License

16 along with this program. If not, see <http://www.gnu.org/licenses/>.

17 -->

18

19 <?import java.lang.\*?>

20 <?import java.util.\*?>

21 <?import javafx.scene.\*?>

22 <?import javafx.scene.control.\*?>

23 <?import javafx.scene.layout.\*?>

24

25 <AnchorPane fx:id="rootPane" prefHeight="191.0" prefWidth="281.0" xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1" fx:controller="fancyshader.client.gui.DetailViewerController">

26 <children>

27 <Label layoutX="10.0" layoutY="14.0" text="Title" AnchorPane.leftAnchor="10.0" AnchorPane.rightAnchor="225.0" />

28 <TextField fx:id="titleField" disable="true" layoutX="54.0" layoutY="10.0" prefHeight="25.0" prefWidth="208.0" AnchorPane.leftAnchor="60.0" AnchorPane.rightAnchor="10.0" />

29 <Label layoutX="10.0" layoutY="43.0" prefHeight="17.0" prefWidth="34.0" text="Tags" AnchorPane.leftAnchor="10.0" AnchorPane.rightAnchor="225.0" />

30 <TextField fx:id="tagField" disable="true" layoutX="54.0" layoutY="39.0" prefHeight="25.0" prefWidth="208.0" AnchorPane.leftAnchor="60.0" AnchorPane.rightAnchor="10.0" />

31 <Label layoutX="10.0" layoutY="73.0" prefHeight="17.0" prefWidth="34.0" text="Creator" AnchorPane.leftAnchor="10.0" AnchorPane.rightAnchor="225.0" />

32 <TextField fx:id="creatorField" disable="true" layoutX="54.0" layoutY="69.0" prefHeight="25.0" prefWidth="208.0" AnchorPane.leftAnchor="60.0" AnchorPane.rightAnchor="10.0" />

33 <Label layoutX="17.0" layoutY="103.0" prefHeight="17.0" prefWidth="75.0" text="Date created" AnchorPane.leftAnchor="10.0" AnchorPane.rightAnchor="194.0" />

34 <TextField fx:id="dateCreatedField" disable="true" layoutX="92.0" layoutY="99.0" prefHeight="25.0" prefWidth="177.0" AnchorPane.leftAnchor="92.0" AnchorPane.rightAnchor="10.0" />

35 <Button fx:id="closeButton" cancelButton="true" defaultButton="true" layoutX="107.0" layoutY="163.0" onAction="#onClosePressed" prefHeight="25.0" prefWidth="50.0" text="Close" AnchorPane.leftAnchor="107.0" AnchorPane.rightAnchor="107.0" />

36 <Label fx:id="infoLabel" layoutX="11.0" layoutY="138.0" prefHeight="17.0" prefWidth="263.0" text="Views and delightful" AnchorPane.leftAnchor="10.0" AnchorPane.rightAnchor="10.0" />

37 </children>

38 </AnchorPane>

39

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\DetailViewerController.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.gui;

18

19 import fancyshader.entity.ShaderDemo;

20 import java.text.DateFormat;

21 import java.text.SimpleDateFormat;

22 import javafx.event.ActionEvent;

23 import javafx.fxml.FXML;

24 import javafx.scene.control.Label;

25 import javafx.scene.control.TextField;

26 import javafx.scene.layout.Pane;

27 import javafx.stage.Stage;

28

29 /\*\*

30 \* The controller for the DetailViewer GUI

31 \* @author Reuben Steenekamp

32 \*/

33 public class DetailViewerController {

34 private ShaderDemo shaderDemo;

35

36 @FXML Pane rootPane;

37 @FXML TextField titleField;

38 @FXML TextField tagField;

39 @FXML TextField creatorField;

40 @FXML TextField dateCreatedField;

41 @FXML Label infoLabel;

42

43 /\*\*

44 \* Set the shader demo to use

45 \* @param shaderDemo the shader demo to use

46 \*/

47 public void setShaderDemo(ShaderDemo shaderDemo) {

48 this.shaderDemo = shaderDemo;

49 initFields();

50 }

51

52 private void initFields() {

53 titleField.setText(shaderDemo.getTitle());

54 tagField.setText(Helper.toDelimitedTags(shaderDemo.getTags()));

55 creatorField.setText(shaderDemo.getCreator().getUsername());

56 DateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");

57 dateCreatedField.setText(formatter.format(shaderDemo.getDateCreated()));

58 infoLabel.setText("Visited: "+(shaderDemo.getVisitors()==null?0:shaderDemo.getVisitors().size())+"\tDelighted: "+(shaderDemo.getVisitors()==null?0:shaderDemo.getDelighted().size()));

59 }

60

61 @FXML

62 public void onClosePressed(ActionEvent e) {

63 ((Stage)rootPane.getScene().getWindow()).close();

64 }

65 }

66

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\Editor.fxml |

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

2 <!--

3 Copyright (C) 2014 Reuben Steenekamp

4

5 This program is free software: you can redistribute it and/or modify

6 it under the terms of the GNU General Public License as published by

7 the Free Software Foundation, either version 3 of the License, or

8 (at your option) any later version.

9

10 This program is distributed in the hope that it will be useful,

11 but WITHOUT ANY WARRANTY; without even the implied warranty of

12 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

13 GNU General Public License for more details.

14

15 You should have received a copy of the GNU General Public License

16 along with this program. If not, see <http://www.gnu.org/licenses/>.

17 -->

18

19 <?import javafx.geometry.\*?>

20 <?import javafx.scene.canvas.\*?>

21 <?import java.lang.\*?>

22 <?import java.util.\*?>

23 <?import javafx.scene.\*?>

24 <?import javafx.scene.control.\*?>

25 <?import javafx.scene.layout.\*?>

26

27 <BorderPane xmlns:fx="http://javafx.com/fxml/1" fx:id="rootPane" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="0.0" minWidth="0.0" prefHeight="300.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/8" fx:controller="fancyshader.client.gui.EditorController">

28 <top>

29 <MenuBar BorderPane.alignment="CENTER">

30 <menus>

31 <Menu text="\_File">

32 <items>

33 <MenuItem onAction="#onSavePressed" text="\_Save" />

34 <MenuItem onAction="#onEditDetailsPressed" text="\_Edit details" />

35 <MenuItem onAction="#onDeletePressed" text="\_Delete" />

36 </items>

37 </Menu>

38 <Menu text="\_Run">

39 <items>

40 <MenuItem fx:id="playMenuItem" onAction="#onPlayPressed" text="\_Start"/>

41 <MenuItem fx:id="pauseMenuItem" onAction="#onPausePressed" text="\_Pause"/>

42 <MenuItem fx:id="resetMenuItem" onAction="#onResetPressed" text="\_Reset"/>

43 </items>

44 </Menu>

45 <Menu text="\_Help">

46 <items>

47 <MenuItem onAction="#onHelpContentPressed" text="Help \_content" />

48 <MenuItem onAction="#onAboutPressed" text="\_About" />

49 </items>

50 </Menu>

51 </menus>

52 </MenuBar>

53 </top>

54 <center>

55 <SplitPane dividerPositions="0.5" prefHeight="430.0" prefWidth="439.0" BorderPane.alignment="CENTER">

56 <items>

57 <TextArea fx:id="codeArea" SplitPane.resizableWithParent="false" />

58 <BorderPane>

59 <center>

60 <fancyshader.client.viewer.DemoViewer fx:id="viewer" />

61 </center>

62 </BorderPane>

63 </items>

64 </SplitPane>

65 </center>

66 </BorderPane>

67

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\EditorController.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.gui;

18

19 import fancyshader.client.viewer.DemoViewer;

20 import fancyshader.entity.ShaderDemo;

21 import fancyshader.service.FancyShaderSessionExpiredException;

22 import java.awt.Desktop;

23 import java.io.IOException;

24 import java.net.URI;

25 import java.net.URISyntaxException;

26 import java.rmi.RemoteException;

27 import java.util.logging.Level;

28 import java.util.logging.Logger;

29 import javafx.event.ActionEvent;

30 import javafx.fxml.FXML;

31 import javafx.scene.control.MenuItem;

32 import javafx.scene.control.TextArea;

33 import javafx.scene.layout.Pane;

34 import javafx.stage.Stage;

35

36 /\*\*

37 \* The controller for the Editor GUI

38 \*

39 \* @author Reuben Steenekamp

40 \*/

41 public class EditorController {

42 private MainController mainController;

43 private ShaderDemo shaderDemo;

44

45 @FXML private Pane rootPane;

46 @FXML private TextArea codeArea;

47 @FXML private DemoViewer viewer;

48

49 @FXML private MenuItem playMenuItem;

50 @FXML private MenuItem pauseMenuItem;

51 @FXML private MenuItem resetMenuItem;

52

53 private boolean unsavedWork = false;

54

55 public void initialize() {

56 updateMenuItems();

57 }

58

59 /\*\*

60 \* Set the main controller

61 \* @param mainController the main controller to set

62 \*/

63 public void setMainController(MainController mainController) {

64 this.mainController = mainController;

65 viewer.getDemoAnimator().play();

66

67 ((Stage)rootPane.getScene().getWindow()).onCloseRequestProperty().set((e)->{

68 // On an request to close

69

70 // Vito the close request

71 e.consume();

72

73 // Open a confirm exit dialog if there is unsaved work, otherwise just exit

74 if(unsavedWork) {

75 mainController.createConfirmExit(new Stage(), rootPane, (result)->{

76 switch(result){

77 case SAVE:

78 save();

79 exitWindow();

80 break;

81 case EXIT:

82 exitWindow();

83 break;

84 default:

85 throw new AssertionError(result.name());

86

87 }

88 });

89 }

90 else exitWindow();

91 });

92 codeArea.textProperty().addListener((observable, oldValue, newValue)-> {

93 unsavedWork = true;

94 updatePreview();

95 });

96 }

97

98 private void save() {

99 try {

100 // Set the code to that of the code area

101 shaderDemo.setCode(codeArea.getText());

102

103 // Update the demo using FancyShaderRemoteService

104 mainController.getService().update(mainController.getSessionProperty().getValue(), shaderDemo);

105

106 // Clear the unsaved work flag

107 unsavedWork = false;

108

109 // Trigger a refresh, as the change in code must be reflected in the browser

110 mainController.refresh();

111 } catch (FancyShaderSessionExpiredException | RemoteException ex) {

112 Logger.getLogger(EditorController.class.getName()).log(Level.SEVERE, null, ex);

113 }

114 }

115

116 private void exitWindow() {

117 ((Stage)rootPane.getScene().getWindow()).close();

118 }

119

120 /\*\*

121 \* Set the shader demo to use

122 \* @param shaderDemo the shader demo to use

123 \*/

124 public void setShaderDemo(ShaderDemo shaderDemo) {

125 this.shaderDemo = shaderDemo;

126 codeArea.setText(shaderDemo.getCode());

127 unsavedWork = false;

128 }

129

130 private void updatePreview() {

131 if(shaderDemo != null)

132 viewer.setCode(codeArea.getText());

133 }

134

135

136 @FXML

137 public void onAboutPressed(ActionEvent e) {

138 if(Desktop.isDesktopSupported()) {

139 try {

140 Desktop.getDesktop().browse(new URI("file://"+Helper.getUserPath().getPath().replace(" ", "%20")+"/resources/about.html"));

141 } catch (URISyntaxException | IOException ex) {

142 Logger.getLogger(BrowserController.class.getName()).log(Level.WARNING, null, ex);

143 }

144 }

145 }

146

147 @FXML

148 public void onHelpContentPressed(ActionEvent e) {

149 if(Desktop.isDesktopSupported()) {

150 try {

151 Desktop.getDesktop().browse(new URI("file://"+Helper.getUserPath().getPath().replace(" ", "%20")+"/resources/editor.html"));

152 } catch (URISyntaxException | IOException ex) {

153 Logger.getLogger(BrowserController.class.getName()).log(Level.WARNING, null, ex);

154 }

155 }

156 }

157

158 @FXML

159 public void onSavePressed(ActionEvent e) {

160 save();

161 }

162

163 @FXML

164 public void onEditDetailsPressed(ActionEvent e) {

165 // Open a detail editor

166 mainController.createDetailEditor(new Stage(), shaderDemo, rootPane, (ShaderDemo input)->{

167 // When done

168

169 // Set the details of the demo

170 shaderDemo.setTitle(input.getTitle());

171 shaderDemo.setTags(input.getTags());

172 shaderDemo.setAvailability(input.getAvailability());

173

174 // Update the demo using FancyShaderRemoteService

175 mainController.getService().update(mainController.getSessionProperty().getValue(), shaderDemo);

176

177 // Trigger a refresh, as the change in the name of the demo must be reflected in the browser

178 mainController.refresh();

179 });

180 }

181

182 @FXML

183 public void onDeletePressed(ActionEvent e) {

184 try {

185 // Delete the demo using FancyShaderRemoteService

186 mainController.getService().delete(mainController.getSessionProperty().getValue(), shaderDemo);

187

188 // Trigger a refresh, as the removal of the demo must be reflected in the browser

189 mainController.refresh();

190 exitWindow();

191 } catch (RemoteException | FancyShaderSessionExpiredException ex) {

192 Logger.getLogger(EditorController.class.getName()).log(Level.SEVERE, null, ex);

193 }

194 }

195

196 @FXML

197 public void onPlayPressed(ActionEvent e) {

198 viewer.getDemoAnimator().play();

199

200 // The play menu item is disabled when the viewer is playing

201 // so the menu items need to be updated

202 updateMenuItems();

203 }

204

205 @FXML

206 public void onPausePressed(ActionEvent e) {

207 viewer.getDemoAnimator().pause();

208

209 // The pause menu item is disabled when the viewer is paused

210 // so the menu items need to be updated

211 updateMenuItems();

212 }

213

214 @FXML

215 public void onResetPressed(ActionEvent e) {

216 viewer.getDemoAnimator().reset();

217 }

218

219 public void updateMenuItems() {

220 playMenuItem.setDisable(viewer.getDemoAnimator().isPlaying());

221 pauseMenuItem.setDisable(!viewer.getDemoAnimator().isPlaying());

222 }

223 }

224

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\Helper.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.gui;

18

19 import fancyshader.service.FancyShaderInternalException;

20 import fancyshader.service.FancyShaderLoginException;

21 import fancyshader.service.FancyShaderSessionExpiredException;

22 import java.net.MalformedURLException;

23 import java.net.URL;

24 import java.rmi.RemoteException;

25 import java.util.Arrays;

26 import java.util.Collections;

27 import java.util.HashSet;

28 import java.util.List;

29 import java.util.Set;

30 import java.util.logging.Level;

31 import java.util.logging.Logger;

32

33 /\*\*

34 \* Helper functions for controllers

35 \*

36 \* @author Reuben Steenekamp

37 \*/

38 public class Helper {

39

40 /\*\*

41 \* Takes in a list of semicolon delimited tags and returns a set containing

42 \* the whitespace trimmed tags

43 \* @param tags the tags to be parsed

44 \* @return the set of tags

45 \*/

46 public static Set<String> fromDelimitedTags(String tags) {

47 List<String> tagList = Arrays.asList(tags.split(";"));

48 tagList.replaceAll(String::trim);

49 return new HashSet<>(tagList);

50 }

51

52 /\*\*

53 \* Takes in a set of tags and returns a semicolon delimited String of the tags

54 \* @param tagSet the set of tags

55 \* @return the semicolon delimited tags

56 \*/

57 public static String toDelimitedTags(Set<String> tagSet) {

58 Set<String> tags = tagSet != null ? tagSet : Collections.EMPTY\_SET;

59 String joinedTags = "";

60 for (String tag : tags) {

61 joinedTags += tag + "; ";

62 }

63 return joinedTags;

64 }

65

66 /\*\*

67 \* @return the path to the system property "user.dir"

68 \*/

69 public static URL getUserPath() {

70 try {

71 return new URL("file:///" + System.getProperty("user.dir"));

72 } catch (MalformedURLException ex) {

73 Logger.getLogger(MainController.class.getName()).log(Level.SEVERE, null, ex);

74 return null;

75 }

76 }

77

78 /\*\*

79 \* Takes in an exception and returns a user friendly message to display

80 \* @param e the exception

81 \* @return the user friendly message

82 \*/

83 public static String getErrorText(Exception e) {

84 String text = "An error occurred";

85 try {

86 throw e;

87 } catch (RemoteException ex) {

88 text = "Error communicating with server";

89 } catch (FancyShaderInternalException ex) {

90 text = "An internal server error has occurred";

91 } catch (FancyShaderLoginException ex) {

92 text = "Login details incorrect";

93 } catch (FancyShaderSessionExpiredException ex) {

94 text = "Your session has expired";

95 } catch (Exception ex) {}

96 return text;

97 }

98 }

99

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\Login.fxml |

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

2 <!--

3 Copyright (C) 2014 Reuben Steenekamp

4

5 This program is free software: you can redistribute it and/or modify

6 it under the terms of the GNU General Public License as published by

7 the Free Software Foundation, either version 3 of the License, or

8 (at your option) any later version.

9

10 This program is distributed in the hope that it will be useful,

11 but WITHOUT ANY WARRANTY; without even the implied warranty of

12 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

13 GNU General Public License for more details.

14

15 You should have received a copy of the GNU General Public License

16 along with this program. If not, see <http://www.gnu.org/licenses/>.

17 -->

18

19 <?import java.lang.\*?>

20 <?import java.util.\*?>

21 <?import javafx.scene.\*?>

22 <?import javafx.scene.control.\*?>

23 <?import javafx.scene.layout.\*?>

24

25 <AnchorPane id="AnchorPane" fx:id="rootPane" prefHeight="123.0" prefWidth="180.0" xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1" fx:controller="fancyshader.client.gui.LoginController">

26 <children>

27 <TextField fx:id="usernameField" layoutX="14.0" layoutY="28.0" promptText="Username" AnchorPane.leftAnchor="14.0" AnchorPane.rightAnchor="14.0" />

28 <PasswordField fx:id="passwordField" layoutX="14.0" layoutY="61.0" promptText="Password" AnchorPane.leftAnchor="14.0" AnchorPane.rightAnchor="14.0" />

29 <Button defaultButton="true" layoutX="33.0" layoutY="92.0" mnemonicParsing="true" onAction="#onLoginPressed" text="\_Login" AnchorPane.leftAnchor="30.0" AnchorPane.rightAnchor="99.0" />

30 <Label fx:id="topLabel" layoutX="15.0" layoutY="5.0" prefHeight="17.0" prefWidth="149.0" text="Enter your details:" textAlignment="CENTER" AnchorPane.leftAnchor="14.0" AnchorPane.rightAnchor="14.0" />

31 <Button cancelButton="true" layoutX="96.0" layoutY="92.0" mnemonicParsing="true" onAction="#onCancelPressed" text="\_Cancel" AnchorPane.leftAnchor="93.0" AnchorPane.rightAnchor="30.0" />

32 </children>

33 </AnchorPane>

34

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\LoginController.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.gui;

18

19 import fancyshader.service.FancyShaderInternalException;

20 import fancyshader.service.FancyShaderLoginException;

21 import java.net.URL;

22 import java.rmi.RemoteException;

23 import java.util.ResourceBundle;

24 import java.util.logging.Level;

25 import java.util.logging.Logger;

26 import javafx.event.ActionEvent;

27 import javafx.fxml.FXML;

28 import javafx.fxml.Initializable;

29 import javafx.scene.control.Label;

30 import javafx.scene.control.PasswordField;

31 import javafx.scene.control.TextField;

32 import javafx.scene.layout.Pane;

33 import javafx.stage.Stage;

34

35 /\*\*

36 \* The controller for the Login GUI

37 \*

38 \* @author Reuben Steenekamp

39 \*/

40 public class LoginController{

41 private MainController mainController;

42

43 /\*\*

44 \* Set the main controller

45 \* @param mainController the main controller to set

46 \*/

47 public void setMainController(MainController mainController) {

48 this.mainController = mainController;

49 }

50

51 @FXML private Pane rootPane;

52 @FXML private TextField usernameField;

53 @FXML private PasswordField passwordField;

54 @FXML private Label topLabel;

55

56

57 @FXML

58 public void onLoginPressed(ActionEvent e) {

59 try {

60 mainController.login(usernameField.getText(), passwordField.getText());

61 } catch (FancyShaderInternalException | FancyShaderLoginException | RemoteException ex) {

62 topLabel.setText(Helper.getErrorText(ex));

63 Logger.getLogger(LoginController.class.getName()).log(Level.FINEST, null, ex);

64 return;

65 } finally {

66 // Clear the password field and set it as the focus

67 passwordField.setText("");

68 passwordField.requestFocus();

69 }

70 exitDialog();

71 }

72

73 @FXML

74 public void onCancelPressed(ActionEvent e) {

75 exitDialog();

76 }

77

78 private void exitDialog() {

79 ((Stage)rootPane.getScene().getWindow()).close();

80 }

81 }

82

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\MainController.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.gui;

18

19 import fancyshader.entity.ShaderDemo;

20 import fancyshader.service.FancyShaderInternalException;

21 import fancyshader.service.FancyShaderLoginException;

22 import fancyshader.service.FancyShaderRemoteService;

23 import java.io.IOException;

24 import java.net.URL;

25 import java.rmi.RemoteException;

26 import java.util.logging.Level;

27 import java.util.logging.Logger;

28 import javafx.beans.property.Property;

29 import javafx.beans.property.SimpleObjectProperty;

30 import javafx.fxml.FXMLLoader;

31 import javafx.scene.Node;

32 import javafx.scene.Parent;

33 import javafx.scene.Scene;

34 import javafx.scene.image.Image;

35 import javafx.scene.paint.Color;

36 import javafx.stage.Modality;

37 import javafx.stage.Stage;

38 import javafx.stage.StageStyle;

39

40 /\*\*

41 \* Controller responsible for providing the {@link fancyshader.service.FancyShaderRemoteService},

42 \* helper methods to create new windows, managing the user session and providing a refresh function

43 \* and callback.

44 \* @author Reuben Steenekamp

45 \*/

46 public final class MainController {

47

48 private Property<String> sessionProperty;

49 private FancyShaderRemoteService service;

50 private Runnable refreshCallback;

51

52 public MainController(FancyShaderRemoteService service) {

53 this.service = service;

54 this.sessionProperty = new SimpleObjectProperty<>(null);

55 sessionProperty.addListener((observable, oldValue, newValue) -> {

56 refresh();

57 });

58 }

59

60 /\*\*

61 \* Set the refresh callback

62 \* @param refreshCallback the callback to set

63 \*/

64 public void setRefreshCallback(Runnable refreshCallback) {

65 this.refreshCallback = refreshCallback;

66 }

67

68 /\*\*

69 \* Create a new browser window

70 \* @param stage the stage to use

71 \* @param node the parent

72 \*/

73 public void createBrowser(Stage stage, Node node) {

74 try {

75 URL url = getClass().getResource("Browser.fxml");

76 FXMLLoader loader = new FXMLLoader(url);

77 Parent root = loader.load();

78

79 BrowserController browserController = loader.getController();

80

81 Scene scene = new Scene(root);

82

83 stage.setScene(scene);

84 stage.getIcons().add(new Image(getClass().getResource("FancyShaderIcon.png").toExternalForm()));

85 stage.setResizable(true);

86 stage.setTitle("FancyBrowser");

87

88 browserController.setMainController(this);

89

90 if(node!=null) {

91 stage.initOwner(node.getScene().getWindow());

92 }

93

94 stage.show();

95 } catch (IOException ex) {

96 Logger.getLogger(MainController.class.getName()).log(Level.SEVERE, null, ex);

97 }

98 }

99

100 /\*\*

101 \* Create a new editor window

102 \* @param stage the stage to use

103 \* @param demo the demo to edit

104 \* @param node the parent

105 \*/

106 public void createEditor(Stage stage, ShaderDemo demo, Node node) {

107 try {

108 URL url = getClass().getResource("Editor.fxml");

109 FXMLLoader loader = new FXMLLoader(url);

110 Parent root = loader.load();

111

112 EditorController editorController = loader.getController();

113

114 Scene scene = new Scene(root);

115

116 stage.setScene(scene);

117 stage.getIcons().add(new Image(getClass().getResource("FancyShaderIcon.png").toExternalForm()));

118 stage.setResizable(true);

119 stage.setTitle("FancyEditor");

120

121 editorController.setMainController(this);

122 editorController.setShaderDemo(demo);

123

124 if(node!=null) {

125 stage.initOwner(node.getScene().getWindow());

126 }

127 stage.show();

128

129 } catch (IOException ex) {

130 Logger.getLogger(MainController.class.getName()).log(Level.SEVERE, null, ex);

131 }

132 }

133

134 /\*\*

135 \* Create a new exit confirmation dialog

136 \* @param stage the stage to use

137 \* @param node the parent

138 \* @param callback the callback upon save/exit

139 \*/

140 public void createConfirmExit(Stage stage, Node node, ConsumerWithException<ConfirmExitController.Result> callback) {

141 try {

142 URL url = getClass().getResource("ConfirmExit.fxml");

143 FXMLLoader loader = new FXMLLoader(url);

144 Parent root = loader.load();

145

146 ConfirmExitController confirmExitController = loader.getController();

147

148

149 Scene scene = new Scene(root);

150

151 stage.setScene(scene);

152 stage.getIcons().add(new Image(getClass().getResource("FancyShaderIcon.png").toExternalForm()));

153 stage.setResizable(false);

154 stage.initStyle(StageStyle.UTILITY);

155 stage.setTitle("FancyConfirmExit");

156

157 confirmExitController.setCallback(callback);

158

159 if(node!=null) {

160 stage.initModality(Modality.APPLICATION\_MODAL);

161 stage.initOwner(node.getScene().getWindow());

162 }

163 stage.show();

164

165 } catch (IOException ex) {

166 Logger.getLogger(MainController.class.getName()).log(Level.SEVERE, null, ex);

167 }

168 }

169

170 /\*\*

171 \* Create a new detail editor dialog

172 \* @param stage the stage to use

173 \* @param demo the demo to edit details of

174 \* @param node the parent

175 \* @param callback the callback upon completion of the dialog

176 \*/

177 public void createDetailEditor(Stage stage, ShaderDemo demo, Node node, ConsumerWithException<ShaderDemo> callback) {

178 try {

179 URL url = getClass().getResource("DetailEditor.fxml");

180 FXMLLoader loader = new FXMLLoader(url);

181 Parent root = loader.load();

182

183 DetailEditorController detailEditorController = loader.getController();

184 detailEditorController.setCallback(callback);

185

186

187 Scene scene = new Scene(root);

188

189 stage.setScene(scene);

190 stage.getIcons().add(new Image(getClass().getResource("FancyShaderIcon.png").toExternalForm()));

191 stage.setResizable(false);

192 stage.initStyle(StageStyle.UTILITY);

193 stage.setTitle("FancyDetailEditor");

194

195

196 detailEditorController.setMainController(this);

197 detailEditorController.setShaderDemo(demo);

198

199 if(node!=null) {

200 stage.initOwner(node.getScene().getWindow());

201 }

202 stage.show();

203

204 } catch (IOException ex) {

205 Logger.getLogger(MainController.class.getName()).log(Level.SEVERE, null, ex);

206 }

207 }

208

209 /\*\*

210 \* Create a detail viewer window

211 \* @param stage the stage to use

212 \* @param demo the demo to view details of

213 \* @param node the parent

214 \*/

215 public void createDetailViewer(Stage stage, ShaderDemo demo, Node node) {

216 try {

217 URL url = getClass().getResource("DetailViewer.fxml");

218 FXMLLoader loader = new FXMLLoader(url);

219 Parent root = loader.load();

220

221 DetailViewerController detailViewerController = loader.getController();

222

223 Scene scene = new Scene(root);

224

225 stage.setScene(scene);

226 stage.getIcons().add(new Image(getClass().getResource("FancyShaderIcon.png").toExternalForm()));

227 stage.setResizable(false);

228 stage.initStyle(StageStyle.UTILITY);

229 stage.setTitle("FancyDetailViewer");

230

231

232 detailViewerController.setShaderDemo(demo);

233

234 if(node!=null) {

235 stage.initOwner(node.getScene().getWindow());

236 }

237 stage.show();

238

239 } catch (IOException ex) {

240 Logger.getLogger(MainController.class.getName()).log(Level.SEVERE, null, ex);

241 }

242 }

243

244 /\*\*

245 \* Create a login dialog

246 \* @param stage the stage to use

247 \* @param node the parent

248 \*/

249 public void createLogin(Stage stage, Node node) {

250 try {

251 URL url = getClass().getResource("Login.fxml");

252 FXMLLoader loader = new FXMLLoader(url);

253 Parent root = loader.load();

254

255 LoginController loginController = loader.getController();

256

257

258 Scene scene = new Scene(root);

259

260 stage.setScene(scene);

261 stage.getIcons().add(new Image(getClass().getResource("FancyShaderIcon.png").toExternalForm()));

262 stage.setResizable(false);

263 stage.initStyle(StageStyle.UTILITY);

264 stage.setTitle("FancyLogin");

265

266 loginController.setMainController(this);

267

268 if(node!=null) {

269 stage.initModality(Modality.APPLICATION\_MODAL);

270 stage.initOwner(node.getScene().getWindow());

271 }

272 stage.show();

273

274 } catch (IOException ex) {

275 Logger.getLogger(MainController.class.getName()).log(Level.SEVERE, null, ex);

276 }

277 }

278

279 /\*\*

280 \* Create a new register dialog

281 \* @param stage the stage to use

282 \* @param node the parent

283 \*/

284 public void createRegister(Stage stage, Node node) {

285 try {

286 URL url = getClass().getResource("Register.fxml");

287 FXMLLoader loader = new FXMLLoader(url);

288 Parent root = loader.load();

289

290 RegisterController registerController = loader.getController();

291

292

293 Scene scene = new Scene(root);

294

295 stage.setScene(scene);

296 stage.getIcons().add(new Image(getClass().getResource("FancyShaderIcon.png").toExternalForm()));

297 stage.setResizable(false);

298 stage.initStyle(StageStyle.UTILITY);

299 stage.setTitle("FancyRegister");

300

301 registerController.setMainController(this);

302

303 if(node!=null) {

304 stage.initModality(Modality.APPLICATION\_MODAL);

305 stage.initOwner(node.getScene().getWindow());

306 }

307 stage.show();

308

309 } catch (IOException ex) {

310 Logger.getLogger(MainController.class.getName()).log(Level.SEVERE, null, ex);

311 }

312 }

313

314 /\*\*

315 \* Create a splash screen

316 \* @param stage the stage to use

317 \*/

318 public static void createSplash(Stage stage) {

319 try {

320 URL url = MainController.class.getResource("Splash.fxml");

321 FXMLLoader loader = new FXMLLoader(url);

322 Parent root = loader.load();

323

324 Scene scene = new Scene(root);

325 scene.setFill(Color.TRANSPARENT);

326

327 stage.setScene(scene);

328 stage.getIcons().add(new Image(MainController.class.getResource("FancyShaderIcon.png").toExternalForm()));

329 stage.setResizable(false);

330 stage.initStyle(StageStyle.TRANSPARENT);

331

332 stage.show();

333

334 } catch (IOException ex) {

335 Logger.getLogger(MainController.class.getName()).log(Level.SEVERE, null, ex);

336 }

337 }

338

339 /\*\*

340 \* Trigger the refresh callback

341 \*/

342 public void refresh() {

343 refreshCallback.run();

344 }

345

346 /\*\*

347 \* Login using the {@link FancyShaderRemoteService} and set the sessionProperty to the created session

348 \* @param username

349 \* @param password

350 \* @throws FancyShaderLoginException

351 \* @throws FancyShaderInternalException

352 \* @throws RemoteException

353 \*/

354 public void login(String username, String password) throws FancyShaderLoginException, FancyShaderInternalException, RemoteException {

355 String uuid = getService().login(username, password);

356 getSessionProperty().setValue(uuid);

357 }

358

359 /\*\*

360 \* Logout using the {@link FancyShaderRemoteService} and set the sessionProperty to null

361 \* @throws RemoteException

362 \*/

363 public void logout() throws RemoteException {

364 getService().logout(getSessionProperty().getValue());

365 getSessionProperty().setValue(null);

366 }

367

368 /\*\*

369 \* @return the sessionProperty

370 \*/

371 public Property<String> getSessionProperty() {

372 return sessionProperty;

373 }

374

375 /\*\*

376 \* @return the service

377 \*/

378 public FancyShaderRemoteService getService() {

379 return service;

380 }

381 }

382

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\Register.fxml |

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

2 <!--

3 Copyright (C) 2014 Reuben Steenekamp

4

5 This program is free software: you can redistribute it and/or modify

6 it under the terms of the GNU General Public License as published by

7 the Free Software Foundation, either version 3 of the License, or

8 (at your option) any later version.

9

10 This program is distributed in the hope that it will be useful,

11 but WITHOUT ANY WARRANTY; without even the implied warranty of

12 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

13 GNU General Public License for more details.

14

15 You should have received a copy of the GNU General Public License

16 along with this program. If not, see <http://www.gnu.org/licenses/>.

17 -->

18

19 <?import java.lang.\*?>

20 <?import java.util.\*?>

21 <?import javafx.scene.\*?>

22 <?import javafx.scene.control.\*?>

23 <?import javafx.scene.layout.\*?>

24

25 <AnchorPane id="AnchorPane" fx:id="rootPane" prefHeight="123.0" prefWidth="236.0" xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1" fx:controller="fancyshader.client.gui.RegisterController">

26 <children>

27 <TextField fx:id="usernameField" layoutX="14.0" layoutY="28.0" onKeyReleased="#onUsernameChanged" promptText="Username" AnchorPane.leftAnchor="14.0" AnchorPane.rightAnchor="14.0" />

28 <PasswordField fx:id="passwordField" layoutX="14.0" layoutY="61.0" onKeyReleased="#onPasswordChanged" promptText="Password" AnchorPane.leftAnchor="14.0" AnchorPane.rightAnchor="14.0" />

29 <Button fx:id="registerButton" defaultButton="true" layoutX="48.0" layoutY="92.0" mnemonicParsing="true" onAction="#onRegisterPressed" prefHeight="25.0" prefWidth="62.0" text="\_Register" AnchorPane.leftAnchor="50.0" AnchorPane.rightAnchor="126.0" />

30 <Label fx:id="topLabel" layoutX="15.0" layoutY="5.0" prefHeight="17.0" prefWidth="149.0" text="Create your account:" textAlignment="CENTER" AnchorPane.leftAnchor="15.0" AnchorPane.rightAnchor="16.0" />

31 <Button fx:id="cancelButton" cancelButton="true" layoutX="125.0" layoutY="92.0" mnemonicParsing="true" onAction="#onCancelPressed" prefHeight="25.0" prefWidth="62.0" text="\_Cancel" AnchorPane.leftAnchor="124.0" AnchorPane.rightAnchor="50.0" />

32 </children>

33 </AnchorPane>

34

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\RegisterController.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.gui;

18

19 import fancyshader.service.FancyShaderInternalException;

20 import java.net.URL;

21 import java.rmi.RemoteException;

22 import java.util.ResourceBundle;

23 import java.util.logging.Level;

24 import java.util.logging.Logger;

25 import javafx.event.ActionEvent;

26 import javafx.fxml.FXML;

27 import javafx.fxml.Initializable;

28 import javafx.scene.control.Button;

29 import javafx.scene.control.Label;

30 import javafx.scene.control.PasswordField;

31 import javafx.scene.control.TextField;

32 import javafx.scene.input.KeyEvent;

33 import javafx.scene.layout.Pane;

34 import javafx.stage.Stage;

35

36 /\*\*

37 \* The controller for the Register GUI

38 \*

39 \* @author Reuben Steenekamp

40 \*/

41 public class RegisterController {

42

43 private MainController mainController;

44

45 /\*\*

46 \* Set the main controller

47 \* @param mainController the main controller to set

48 \*/

49 public void setMainController(MainController mainController) {

50 this.mainController = mainController;

51 }

52

53 @FXML private Pane rootPane;

54 @FXML private TextField usernameField;

55 @FXML private PasswordField passwordField;

56 @FXML private Label topLabel;

57 @FXML private Button registerButton;

58

59 private final static String TOP\_LABEL\_TEXT = "Create your account:";

60

61 public void updateDialog() {

62 // Validate the fields, update the top label to show the message and

63 // disable the register button if not valid

64 String text = TOP\_LABEL\_TEXT;

65 boolean valid = false;

66 try {

67 if (usernameField.getText().length() < 4) {

68 text = "Username must be >= 4 characters";

69 } else if (mainController != null && mainController.getService().isUsernameTaken(usernameField.getText())) {

70 text = "Username is taken";

71 } else if (passwordField.getText().length() < 4) {

72 text = "Password must be >= 4 characters";

73 } else valid = true;

74 topLabel.setText(text);

75 } catch (FancyShaderInternalException | RemoteException e) {

76 topLabel.setText(Helper.getErrorText(e));

77 Logger.getLogger(RegisterController.class.getName()).log(Level.WARNING, null, e);

78 }

79 registerButton.setDisable(!valid);

80 }

81

82 @FXML

83 public void initialize() {

84 registerButton.setDisable(true);

85 }

86

87 @FXML

88 public void onUsernameChanged(KeyEvent e) {

89 updateDialog();

90 }

91

92 @FXML

93 public void onPasswordChanged(KeyEvent e) {

94 updateDialog();

95 }

96

97 @FXML

98 public void onRegisterPressed(ActionEvent e) {

99 try {

100 mainController.getService().registerAccount(usernameField.getText(), passwordField.getText());

101 } catch (FancyShaderInternalException | RemoteException ex) {

102 topLabel.setText(Helper.getErrorText(ex));

103 return;

104 }

105 passwordField.setText("");

106 passwordField.requestFocus();

107 ((Stage) rootPane.getScene().getWindow()).close();

108 }

109

110 @FXML

111 public void onCancelPressed(ActionEvent e) {

112 ((Stage)rootPane.getScene().getWindow()).close();

113 }

114

115 }

116

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\gui\Splash.fxml |

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

2 <!--

3 Copyright (C) 2014 Reuben Steenekamp

4

5 This program is free software: you can redistribute it and/or modify

6 it under the terms of the GNU General Public License as published by

7 the Free Software Foundation, either version 3 of the License, or

8 (at your option) any later version.

9

10 This program is distributed in the hope that it will be useful,

11 but WITHOUT ANY WARRANTY; without even the implied warranty of

12 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

13 GNU General Public License for more details.

14

15 You should have received a copy of the GNU General Public License

16 along with this program. If not, see <http://www.gnu.org/licenses/>.

17 -->

18

19 <?import java.lang.\*?>

20 <?import java.net.\*?>

21 <?import java.util.\*?>

22 <?import javafx.scene.\*?>

23 <?import javafx.scene.control.\*?>

24 <?import javafx.scene.layout.\*?>

25 <?import javafx.scene.image.\*?>

26

27 <AnchorPane id="rootPane" prefHeight="0.0" prefWidth="800.0" styleClass="mainFxmlClass" xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1">

28 <children>

29 <ImageView fx:id="imageView">

30 <image>

31 <Image url="@FancyShaderLogo.png" />

32 </image>

33 </ImageView>

34 </children>

35 </AnchorPane>

36

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\viewer\DemoAnimator.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.viewer;

18

19 import com.jogamp.opengl.util.Animator;

20

21 /\*\*

22 \* Controls the animation of the demo. Allows playing, pausing and resetting.

23 \* @author Reuben Steenekamp

24 \*/

25 public class DemoAnimator {

26 private long previousElapsed;

27 private DemoRunner runner;

28 private Animator animator;

29 private boolean firstSet = false;

30

31 public DemoAnimator() {

32 previousElapsed = 0;

33 animator = new Animator();

34 animator.setUpdateFPSFrames(1, null);

35 animator.start();

36 }

37

38 /\*\*

39 \* Set the {@link DemoRunner} to animate.

40 \* @param runner the {@link DemoRunner} to set

41 \*/

42 public void setDemoRunner(DemoRunner runner) {

43 boolean wasPlaying = isPlaying();

44 if(wasPlaying) pause();

45 if(this.runner != null && !firstSet) {

46 firstSet = true;

47 animator.remove(this.runner.getGLJPanel());

48 runner.setDemoAnimator(null);

49 }

50

51 this.runner = runner;

52

53 if(runner != null) {

54 runner.setDemoAnimator(this);

55 animator.add(runner.getGLJPanel());

56 }

57 if(wasPlaying) play();

58 }

59

60 /\*\*

61 \* Play/Resume the animator

62 \*/

63 public void play() {

64 animator.resume();

65 }

66

67 /\*\*

68 \* Pause the animator

69 \*/

70 public void pause() {

71 bankTime();

72 animator.pause();

73 }

74

75 /\*\*

76 \* Reset the animator

77 \*/

78 public void reset() {

79 bankTime();

80 previousElapsed = 0;

81 if(!animator.isAnimating()) {

82 play();

83 pause();

84 }

85 }

86

87 /\*\*

88 \* @return whether the animator is currently playing

89 \*/

90 public boolean isPlaying() {

91 return !animator.isPaused();

92 }

93

94 /\*\*

95 \* Get the total time elapsed while the animator is playing since the last reset

96 \* @return the time elapsed

97 \*/

98 public long getElapsedTime() {

99 //bankTime();

100 return previousElapsed+animator.getTotalFPSDuration();

101 }

102

103 private void bankTime() {

104 previousElapsed += animator.getTotalFPSDuration();

105 animator.resetFPSCounter();

106 }

107 }

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\viewer\DemoRunner.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.viewer;

18

19 import java.nio.ByteBuffer;

20 import java.nio.IntBuffer;

21 import java.nio.charset.Charset;

22 import javax.media.opengl.GL2;

23 import javax.media.opengl.GLAutoDrawable;

24 import javax.media.opengl.GLCapabilities;

25 import javax.media.opengl.GLEventListener;

26 import javax.media.opengl.GLProfile;

27 import javax.media.opengl.awt.GLJPanel;

28

29 /\*\*

30 \* Renders a shader program onto a plane occupying the screen with the given fragment shader

31 \* @author Reuben Steenekamp

32 \*/

33 public class DemoRunner implements GLEventListener{

34

35 private final String VERT\_CODE = "void main(){ gl\_Position = gl\_ModelViewProjectionMatrix \* gl\_Vertex; }";

36 private final String ERROR\_CODE = "uniform float time; uniform vec2 resolution; void main() { vec2 pos = gl\_FragCoord.xy/resolution-.5; pos\*=10.; float mul = .5\*(1.+sin(length(pos)-time/1000.)); gl\_FragColor = mul\*vec4(1,0,0,0); }";

37 private String fragCode = null;

38 private boolean shadersReady = false;

39 private GLJPanel panel;

40 private GLProfile profile;

41 private GL2 gl;

42 private boolean error;

43 private String vertLog, fragLog, programLog;

44 private int timeLocation, resolutionLocation;

45 private DemoAnimator animator;

46

47 public DemoRunner() {

48 profile = GLProfile.get(GLProfile.GL2);

49 panel = new GLJPanel(new GLCapabilities(profile));

50 addEventListeners();

51 }

52

53 private void addEventListeners() {

54 panel.addGLEventListener(this);

55 }

56

57

58 private int program = -1;

59

60 private boolean getLinkAndValidateStatus(int program) {

61 IntBuffer status = IntBuffer.allocate(1);

62 gl.glGetProgramiv(program, GL2.GL\_LINK\_STATUS, status);

63 if(status.get() == GL2.GL\_FALSE) return false;

64 status.rewind();

65 gl.glGetProgramiv(program, GL2.GL\_VALIDATE\_STATUS, status);

66 if(status.get() == GL2.GL\_FALSE) return false;

67 return true;

68 }

69 private String getInfoLog(int object, boolean program) {

70 IntBuffer logLength = IntBuffer.allocate(1);

71 if(program)

72 gl.glGetProgramiv(object, GL2.GL\_INFO\_LOG\_LENGTH, logLength);

73 else

74 gl.glGetShaderiv(object, GL2.GL\_INFO\_LOG\_LENGTH, logLength);

75 ByteBuffer logBytes = ByteBuffer.allocate(logLength.get(0));

76 gl.glGetInfoLogARB(object, logBytes.capacity(), null, logBytes);

77 String log = Charset.forName("US-ASCII").decode(logBytes).toString();

78 return log.trim();

79 }

80

81 @Override

82 public void init(GLAutoDrawable drawable) {

83 gl = panel.getGL().getGL2();

84 }

85

86 @Override

87 public void dispose(GLAutoDrawable drawable) {

88 gl.glDeleteProgram(program);

89 }

90

91 private void initShaders(GL2 gl, String code) {

92 program = gl.glCreateProgramObjectARB();

93 if(program<1) throw new RuntimeException("Shader program could not be created");

94

95 int vertShader = gl.glCreateShaderObjectARB(GL2.GL\_VERTEX\_SHADER),

96 fragShader = gl.glCreateShaderObjectARB(GL2.GL\_FRAGMENT\_SHADER);

97

98

99 gl.glShaderSourceARB(vertShader, 1, new String[]{VERT\_CODE}, null);

100 gl.glCompileShader(vertShader);

101 gl.glAttachObjectARB(program, vertShader);

102

103 gl.glShaderSourceARB(fragShader, 1, new String[]{code}, null);

104 gl.glCompileShader(fragShader);

105 gl.glAttachObjectARB(program, fragShader);

106

107 gl.glLinkProgramARB(program);

108

109 vertLog = getInfoLog(vertShader, false);

110 fragLog = getInfoLog(fragShader, false);

111 programLog = getInfoLog(program, true);

112 error = !getLinkAndValidateStatus(program);

113 if(error || fragLog.length() > 0 || programLog.length() > 0) {

114 gl.glDeleteProgram(program);

115 gl.glDeleteObjectARB(vertShader);

116 gl.glDeleteObjectARB(fragShader);

117 if(code == (Object)ERROR\_CODE)

118 program = 0;

119 else

120 initShaders(gl, ERROR\_CODE);

121 return;

122 }

123

124 timeLocation = gl.glGetUniformLocationARB(program, "time");

125 resolutionLocation = gl.glGetUniformLocationARB(program, "resolution");

126 }

127

128 @Override

129 public void display(GLAutoDrawable drawable) {

130

131 if(!shadersReady) {

132 if(fragCode == null)

133 return;

134 initShaders(gl, fragCode);

135 shadersReady = true;

136 }

137

138 gl.glClear(GL2.GL\_COLOR\_BUFFER\_BIT | GL2.GL\_DEPTH\_BUFFER\_BIT);

139

140

141

142 gl.glMatrixMode(GL2.GL\_MODELVIEW);

143 gl.glLoadIdentity();

144 gl.glTranslatef(0, 0, -1); // Not certain if necesary

145

146

147 gl.glUseProgramObjectARB(program);

148 if(timeLocation > -1)

149 gl.glUniform1f(timeLocation, animator.getElapsedTime());

150 if(resolutionLocation > -1)

151 gl.glUniform2f(resolutionLocation, drawable.getWidth(), drawable.getHeight());

152

153

154 gl.glBegin(GL2.GL\_TRIANGLES);

155 gl.glVertex3i(-1,-1, 0);

156 gl.glVertex3i( 1,-1, 0);

157 gl.glVertex3i( 1, 1, 0);

158

159 gl.glVertex3i( 1, 1, 0);

160 gl.glVertex3i( -1,-1, 0);

161 gl.glVertex3i(-1, 1, 0);

162 gl.glEnd();

163

164 gl.glUseProgramObjectARB(0);

165 }

166

167 @Override

168 public void reshape(GLAutoDrawable drawable, int x, int y, int width, int height) {}

169

170 /\*\*

171 \* Get the GLJPanel that the Runner renders on.

172 \* @return the GLJPanel

173 \*/

174 public GLJPanel getGLJPanel() {

175 return panel;

176 }

177

178 /\*\*

179 \* Set the fragment shader source code to be compiled into the shader program used to render on the GLJPanel

180 \* @param fragCode the fragment shader source code to be set

181 \*/

182 public void setFragCode(String fragCode) {

183 this.fragCode = fragCode;

184 this.shadersReady = false;

185 }

186

187 /\*\*

188 \* Set the animator from which the elapsed time can be retrieved

189 \* @param animator the {@link DemoAnimator} to set

190 \*/

191 public void setDemoAnimator(DemoAnimator animator) {

192 this.animator = animator;

193 }

194

195 }

196

|  |
| --- |
| FancyShaderClient\src\fancyshader\client\viewer\DemoViewer.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.client.viewer;

18

19 import javafx.embed.swing.SwingNode;

20 import javafx.scene.layout.AnchorPane;

21 import javax.swing.JComponent;

22

23 /\*\*

24 \* The viewer for demos. The fragment shader source code is set through the {@link DemoViewer#setCode(String)} and the controls for the animation

25 \* are obtained through {@link DemoViewer#getDemoAnimator()}.

26 \* @author Reuben Steenekamp

27 \*/

28 public class DemoViewer extends AnchorPane{

29 private DemoRunner runner;

30 private SwingNode node;

31 private DemoAnimator animator;

32 private boolean resetOnChange = false;

33

34 public DemoViewer() {

35 super();

36 node = new SwingNode();

37 AnchorPane.setLeftAnchor(node, 0.);

38 AnchorPane.setRightAnchor(node, 0.);

39 AnchorPane.setTopAnchor(node, 0.);

40 AnchorPane.setBottomAnchor(node, 0.);

41 getChildren().add(node);

42

43 animator = new DemoAnimator();

44

45 runner = new DemoRunner();

46 animator.setDemoRunner(runner);

47

48 JComponent panel = runner.getGLJPanel();

49 node.setContent(panel);

50

51 setCode(null);

52 }

53

54 /\*\*

55 \* Set whether to call {@link DemoAnimator#reset()} on the viewer's {@link DemoAnimator} when {@link DemoViewer#setCode(String)} is called.

56 \* It is off by default.

57 \* @param resetOnChange whether to reset on change.

58 \*/

59 public void setResetOnChange(boolean resetOnChange) {

60 this.resetOnChange = resetOnChange;

61 }

62

63 /\*\*

64 \* Set the source code to use for the fragment shader

65 \* @param code the source code

66 \*/

67 public void setCode(String code) {

68 if(code == null) code = "";

69 if(resetOnChange)

70 animator.reset();

71 runner.setFragCode(code);

72 runner.getGLJPanel().repaint();

73 }

74

75 /\*\*

76 \* Get the demo animator that controls the animation of the viewer.

77 \* @return the demo animator

78 \*/

79 public DemoAnimator getDemoAnimator() {

80 return animator;

81 }

82 }

83

|  |
| --- |
| FancyShaderClient\resources\about.html |

1 <!DOCTYPE html>

2 <!--

3 To change this license header, choose License Headers in Project Properties.

4 To change this template file, choose Tools | Templates

5 and open the template in the editor.

6 -->

7 <html>

8 <head>

9 <title>FancyShader About</title>

10 <meta charset="UTF-8">

11 <meta name="viewport" content="width=device-width, initial-scale=1.0">

12 <link rel="stylesheet" href="main.css"/>

13 </head>

14 <body>

15 <center><img alt="FancyShader" src="FancyShaderLogo.png"/></center>

16 <h1>About</h1>

17 <h2>What</h2>

18 <article>

19 <p>

20 There are groups of people who strive to bring art, math and programming

21 together in the form of procedurally generated visual displays. These people

22 identify themselves as the demo scene. The programs they create are demos.

23 </p>

24

25 <p>

26 FancyShader is a utility used to browse, view, edit and share demos programmed

27 as a shader in GLSL. Shaders were created with the intent to provide a programmable

28 element in the transition from rasterized fragments in the 3D rendering pipeline of

29 graphics cards to the final pixel colours outputted to the screen, but it has an alternative

30 use case within the demo scene.

31 </p>

32 </article>

33 <h2>Why</h2>

34 <article>Because it is fun</article>

35 <h2>Who</h2>

36 <article>

37 Reuben Steenekamp (<a href="mailto:reuben.steenekamp+fancyshader@gmail.com">Contact me</a>)

38 </article>

39 </body>

40 </html>

41

|  |
| --- |
| FancyShaderClient\resources\browser.html |

1 <!DOCTYPE html>

2 <!--

3 To change this license header, choose License Headers in Project Properties.

4 To change this template file, choose Tools | Templates

5 and open the template in the editor.

6 -->

7 <html>

8 <head>

9 <title>FancyShader Help</title>

10 <meta charset="UTF-8">

11 <meta name="viewport" content="width=device-width, initial-scale=1.0">

12 <link rel="stylesheet" href="main.css"/>

13 </head>

14 <body>

15 <center><img alt="FancyShader" src="FancyShaderLogo.png"/></center>

16 <h1>Create an account and login</h1>

17 <ol>

18 <li>Open <b>Account &#8594 Register</b>.</li>

19 <li>Enter the details and press <b>Done</b>.</li>

20 <li>Open <b>Account &#8594 Login</b> and fill in your details to login with your new account.</li>

21 </ol>

22 <h1>Create a new demo</h1>

23 <ol>

24 <li>Login to your account if you have not already.</li>

25 <li>Open <b>Demo &#8594 Create</b>.</li>

26 <li>Enter the details and press <b>Done</b> to open the editor.</li>

27 <li>Open <b>Help &#8594 Help contents</b> for help on the editor.</li>

28 </ol>

29 <div>

30 <b>Note:</b> Demos with availability set to public can be seen by anyone. Set it to private if you only want yourself to see it.

31 If you want to work on a demo and show it when it is complete, you can change the availability later under <b>File &#8594 Edit</b> details in the editor.

32 </div>

33 <h1>Fork an existing demo</h1>

34 <ol>

35 <li>Login to your account if you have not already.</li>

36 <li>Select the demo that you want to fork from in the list.</li>

37 <li>Open <b>Demo &#8594 Fork</b>.</li>

38 <li>Enter the details and press <b>Done</b> to open the editor.</li>

39 </ol>

40 <h1>Edit your own demo</h1>

41 <ol>

42 <li>Login to your account if you have not already.</li>

43 <li>Select the demo that you want to edit in the list.</li>

44 <li>Open <b>Demo &#8594 Edit</b> to open the editor.</li>

45 </ol>

46 <div>

47 <b>Note: </b> A demo must have been created through your account in order for you to edit it. If you want to edit someone else's demo, fork it instead.

48 </div>

49 </body>

50 </html>

51

|  |
| --- |
| FancyShaderClient\resources\editor.html |

1 <!DOCTYPE html>

2 <!--

3 To change this license header, choose License Headers in Project Properties.

4 To change this template file, choose Tools | Templates

5 and open the template in the editor.

6 -->

7 <html>

8 <head>

9 <title>FancyShader Help</title>

10 <meta charset="UTF-8">

11 <meta name="viewport" content="width=device-width, initial-scale=1.0">

12 <link rel="stylesheet" href="main.css"/>

13 </head>

14 <body>

15 <center><img alt="FancyShader" src="FancyShaderLogo.png"/></center>

16 <h1>Designing a shader</h1>

17 <p>

18 FancyShader uses GLSL. The GLSL in FancyShader describes a program

19 that runs for each pixel on the preview canvas with the pixel coordinate

20 as the input and the pixel color as the output. For example the GLSL program:

21 </p>

22 <code>

23 uniform vec2 resolution; <br/>

24 void main() { <br/>

25 &nbsp;&nbsp;&nbsp;&nbsp;// normalize the coordinates from pixel dimensions to unit dimensions <br/>

26 &nbsp;&nbsp;&nbsp;&nbsp;vec2 position = gl\_FragCoord.xy/resolution; <br/>

27 &nbsp;&nbsp;&nbsp;&nbsp;// GLSL colors are represented as 4-dimensional vectors containing red, blue, green and opacity values ranging from 0.0 to 1.0 <br/>

28 &nbsp;&nbsp;&nbsp;&nbsp;gl\_FragColor = vec4(position.x); <br/>

29 }

30 </code>

31 <p>

32 is a grayscale gradient that goes from black to white as one goes from left to right. This is a public demo <b>Example 1 (Greyscale Gradient)</b>.

33 One can see that <b>gl\_FragCoord</b> was not the only input. The uniform <b>resolution</b> was also inputted, which is a vector

34 containing the preview canvas pixel dimensions. Another such uniform provided by FancyShader is <b>time</b> which is a <b>float</b>

35 rather than a <b>vec2</b>. This allows for animation to be introduced.

36 </p>

37

38 <p>

39 If the code entered is invalid and does not compile, the error shader will be shown. This can be seen in the public demo <b>Example 2 (Error Shader)</b>

40 </p>

41 <h2>Edit shader details</h2>

42 <ol>

43 <li>Open <b>File &#8594 Edit details</b></li>

44 <li>Change the details and press <b>Done</b> to save</li>

45 </ol>

46 </body>

47 </html>

|  |
| --- |
| FancyShaderClient\resources\main.css |

1 /\*

2 To change this license header, choose License Headers in Project Properties.

3 To change this template file, choose Tools | Templates

4 and open the template in the editor.

5 \*/

6 /\*

7 Created on : 30 Jul 2014, 8:09:26 AM

8 Author : 8bitb\_000

9 \*/

10

11 body {

12 background: #e5c1e5;

13 }

14 h1, h2 {

15 font-family: "Brush Script MT", cursive;

16 color: blueviolet;

17 text-shadow: 0px 0px 5px rgb(140, 70, 200);

18 }

19

|  |
| --- |
| FancyShaderClient\build.xml |

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

2 <project name="FancyShaderClient" default="default" basedir="." xmlns:fx="javafx:com.sun.javafx.tools.ant">

3 <description>Builds, tests, and runs the project FancyShaderClient.</description>

4 <import file="nbproject/build-impl.xml"/>

5 <target name="-post-jfx-jar">

6 <copydir src="${basedir}/resources" dest="${dist.dir}/resources"/>

7 </target>

8 </project>

9

|  |
| --- |
| FancyShaderLib\src\fancyshader\entity\Account.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.entity;

18

19 import java.io.Serializable;

20 import java.util.Date;

21 import java.util.Set;

22 import javax.persistence.Basic;

23 import javax.persistence.CascadeType;

24 import javax.persistence.Column;

25 import javax.persistence.Entity;

26 import javax.persistence.FetchType;

27 import javax.persistence.GeneratedValue;

28 import javax.persistence.GenerationType;

29 import javax.persistence.Id;

30 import javax.persistence.JoinColumn;

31 import javax.persistence.JoinTable;

32 import javax.persistence.ManyToMany;

33 import javax.persistence.NamedQueries;

34 import javax.persistence.NamedQuery;

35 import javax.persistence.OneToMany;

36 import javax.persistence.Table;

37 import javax.persistence.Temporal;

38 import javax.persistence.TemporalType;

39 import javax.xml.bind.annotation.XmlRootElement;

40 import javax.xml.bind.annotation.XmlTransient;

41

42 /\*\*

43 \* The ACCOUNT table entity class. Encapsulates data pertaining to a user account.

44 \* @author Reuben Steenekamp

45 \*/

46 @Entity

47 @Table(name = "ACCOUNT")

48 @XmlRootElement

49 @NamedQueries({

50 @NamedQuery(name = "Account.findAll", query = "SELECT a FROM Account a"),

51 @NamedQuery(name = "Account.findById", query = "SELECT a FROM Account a WHERE a.id = :id"),

52 @NamedQuery(name = "Account.findByUsername", query = "SELECT a FROM Account a WHERE a.username = :username"),

53 @NamedQuery(name = "Account.findByDateCreated", query = "SELECT a FROM Account a WHERE a.dateCreated = :dateCreated")})

54 public class Account implements Serializable {

55 private static final long serialVersionUID = 1L;

56 @Id

57 @GeneratedValue(strategy = GenerationType.IDENTITY)

58 @Basic(optional = false)

59 @Column(name = "ID")

60 private Integer id;

61 @Basic(optional = false)

62 @Column(name = "USERNAME")

63 private String username;

64 @Basic(optional = false)

65 @Column(name = "DATE\_CREATED")

66 @Temporal(TemporalType.DATE)

67 private Date dateCreated;

68 @JoinTable(name = "DELIGHTFUL", joinColumns = {

69 @JoinColumn(name = "ACCOUNT", referencedColumnName = "ID")}, inverseJoinColumns = {

70 @JoinColumn(name = "SHADER\_DEMO", referencedColumnName = "ID")})

71 @ManyToMany(fetch = FetchType.EAGER)

72 private Set<ShaderDemo> delights;

73 @JoinTable(name = "VISIT", joinColumns = {

74 @JoinColumn(name = "ACCOUNT", referencedColumnName = "ID")}, inverseJoinColumns = {

75 @JoinColumn(name = "SHADER\_DEMO", referencedColumnName = "ID")})

76 @ManyToMany(fetch = FetchType.EAGER)

77 private Set<ShaderDemo> visits;

78 @Basic(optional = false)

79 @Column(name ="AUTH\_DETAIL")

80 private int authDetail;

81

82 @OneToMany(cascade = CascadeType.ALL, mappedBy = "creator", fetch = FetchType.EAGER)

83 private Set<ShaderDemo> creations;

84

85 public Account() {

86 }

87

88 public Account(Integer id) {

89 this.id = id;

90 }

91

92 public Account(Integer id, String username, Date dateCreated) {

93 this.id = id;

94 this.username = username;

95 this.dateCreated = dateCreated;

96 }

97

98

99

100 @Override

101 public int hashCode() {

102 int hash = 0;

103 hash += (getId() != null ? getId().hashCode() : 0);

104 return hash;

105 }

106

107 @Override

108 public boolean equals(Object object) {

109 // TODO: Warning - this method won't work in the case the id fields are not set

110 if (!(object instanceof Account)) {

111 return false;

112 }

113 Account other = (Account) object;

114 if ((this.getId() == null && other.getId() != null) || (this.getId() != null && !this.id.equals(other.id))) {

115 return false;

116 }

117 return true;

118 }

119

120 @Override

121 public String toString() {

122 return "fancyshader.entity.Account[ id=" + getId() + " ]";

123 }

124

125 /\*\*

126 \* @return the id

127 \*/

128 public Integer getId() {

129 return id;

130 }

131

132 /\*\*

133 \* @param id the id to set

134 \*/

135 public void setId(Integer id) {

136 this.id = id;

137 }

138

139 /\*\*

140 \* @return the username

141 \*/

142 public String getUsername() {

143 return username;

144 }

145

146 /\*\*

147 \* @param username the username to set

148 \*/

149 public void setUsername(String username) {

150 this.username = username;

151 }

152

153 /\*\*

154 \* @return the date created

155 \*/

156 public Date getDateCreated() {

157 return dateCreated;

158 }

159

160 /\*\*

161 \* @param dateCreated the date created to set

162 \*/

163 public void setDateCreated(Date dateCreated) {

164 this.dateCreated = dateCreated;

165 }

166

167 /\*\*

168 \* @return the set of demos delighted by

169 \*/

170 public Set<ShaderDemo> getDelights() {

171 return delights;

172 }

173

174 /\*\*

175 \* @param delights the set of demos delighted by to set

176 \*/

177 public void setDelights(Set<ShaderDemo> delights) {

178 this.delights = delights;

179 }

180

181 /\*\*

182 \* @return the set of demos visited

183 \*/

184 public Set<ShaderDemo> getVisits() {

185 return visits;

186 }

187

188 /\*\*

189 \* @param visits the set of demos visited to set

190 \*/

191 public void setVisits(Set<ShaderDemo> visits) {

192 this.visits = visits;

193 }

194

195 /\*\*

196 \* @return the AuthDetail id

197 \*/

198 public int getAuthDetail() {

199 return authDetail;

200 }

201

202 /\*\*

203 \* @param authDetail the AuthDetail id to set

204 \*/

205 public void setAuthDetail(int authDetail) {

206 this.authDetail = authDetail;

207 }

208

209 /\*\*

210 \* @return the set of demos created by the account

211 \*/

212 public Set<ShaderDemo> getCreations() {

213 return creations;

214 }

215

216 /\*\*

217 \* @param creations the set of demos created by the account to set

218 \*/

219 public void setCreations(Set<ShaderDemo> creations) {

220 this.creations = creations;

221 }

222

223 }

224

|  |
| --- |
| FancyShaderLib\src\fancyshader\entity\AuthDetail.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.entity;

18

19 import java.io.Serializable;

20 import javax.persistence.Basic;

21 import javax.persistence.Column;

22 import javax.persistence.Entity;

23 import javax.persistence.GeneratedValue;

24 import javax.persistence.GenerationType;

25 import javax.persistence.Id;

26 import javax.persistence.NamedQueries;

27 import javax.persistence.NamedQuery;

28 import javax.persistence.Table;

29 import javax.xml.bind.annotation.XmlRootElement;

30

31 /\*\*

32 \* The AUTH\_DETAIL table entity class. Contains the hashed passwords and salts for the accounts. They are stored

33 \* are stored on the one-to-one join table to avoid leaks of password data through the service layer.

34 \* @author Reuben Steenekamp

35 \*/

36 @Entity

37 @Table(name = "AUTH\_DETAIL")

38 @XmlRootElement

39 @NamedQueries({

40 @NamedQuery(name = "AuthDetail.findAll", query = "SELECT a FROM AuthDetail a"),

41 @NamedQuery(name = "AuthDetail.findById", query = "SELECT a FROM AuthDetail a WHERE a.id = :id"),

42 @NamedQuery(name = "AuthDetail.findByHashcode", query = "SELECT a FROM AuthDetail a WHERE a.hashcode = :hashcode"),

43 @NamedQuery(name = "AuthDetail.findBySalt", query = "SELECT a FROM AuthDetail a WHERE a.salt = :salt")})

44 public class AuthDetail implements Serializable {

45 private static final long serialVersionUID = 1L;

46 @Id

47 @GeneratedValue(strategy = GenerationType.IDENTITY)

48 @Basic(optional = false)

49 @Column(name = "ID")

50 private Integer id;

51 @Basic(optional = false)

52 @Column(name = "HASHCODE")

53 private String hashcode;

54 @Basic(optional = false)

55 @Column(name = "SALT")

56 private String salt;

57

58

59 public AuthDetail() {

60 }

61

62 public AuthDetail(Integer id) {

63 this.id = id;

64 }

65

66 public AuthDetail(Integer id, String hashcode, String salt) {

67 this.id = id;

68 this.hashcode = hashcode;

69 this.salt = salt;

70 }

71

72

73

74 @Override

75 public int hashCode() {

76 int hash = 0;

77 hash += (getId() != null ? getId().hashCode() : 0);

78 return hash;

79 }

80

81 @Override

82 public boolean equals(Object object) {

83 // TODO: Warning - this method won't work in the case the id fields are not set

84 if (!(object instanceof AuthDetail)) {

85 return false;

86 }

87 AuthDetail other = (AuthDetail) object;

88 if ((this.getId() == null && other.getId() != null) || (this.getId() != null && !this.id.equals(other.id))) {

89 return false;

90 }

91 return true;

92 }

93

94 @Override

95 public String toString() {

96 return "fancyshader.entity.AuthDetail[ id=" + getId() + " ]";

97 }

98

99 /\*\*

100 \* @return the id

101 \*/

102 public Integer getId() {

103 return id;

104 }

105

106 /\*\*

107 \* @param id the id to set

108 \*/

109 public void setId(Integer id) {

110 this.id = id;

111 }

112

113 /\*\*

114 \* @return the password hashcode

115 \*/

116 public String getHashcode() {

117 return hashcode;

118 }

119

120 /\*\*

121 \* @param hashcode the password hashcode to set

122 \*/

123 public void setHashcode(String hashcode) {

124 this.hashcode = hashcode;

125 }

126

127 /\*\*

128 \* @return the salt

129 \*/

130 public String getSalt() {

131 return salt;

132 }

133

134 /\*\*

135 \* @param salt the salt to set

136 \*/

137 public void setSalt(String salt) {

138 this.salt = salt;

139 }

140

141 }

142

|  |
| --- |
| FancyShaderLib\src\fancyshader\entity\ShaderDemo.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.entity;

18

19 import java.io.Serializable;

20 import java.util.Arrays;

21 import java.util.Collections;

22 import java.util.Date;

23 import java.util.HashSet;

24 import java.util.List;

25 import java.util.Set;

26 import javax.persistence.Basic;

27 import javax.persistence.CollectionTable;

28 import javax.persistence.Column;

29 import javax.persistence.ElementCollection;

30 import javax.persistence.Entity;

31 import javax.persistence.FetchType;

32 import javax.persistence.GeneratedValue;

33 import javax.persistence.GenerationType;

34 import javax.persistence.Id;

35 import javax.persistence.JoinColumn;

36 import javax.persistence.ManyToMany;

37 import javax.persistence.ManyToOne;

38 import javax.persistence.NamedQueries;

39 import javax.persistence.NamedQuery;

40 import javax.persistence.Table;

41 import javax.persistence.Temporal;

42 import javax.persistence.TemporalType;

43 import javax.xml.bind.annotation.XmlRootElement;

44

45 /\*\*

46 \* The SHADER\_DEMO table entity class. Encapsulates all data pertaining to a demo.

47 \* @author Reuben Steenekamp

48 \*/

49 @Entity

50 @Table(name = "SHADER\_DEMO")

51 @XmlRootElement

52 @NamedQueries({

53 @NamedQuery(name = "ShaderDemo.search", query = "SELECT s FROM ShaderDemo s WHERE (LOWER(s.title) LIKE CONCAT('%',LOWER(:q),'%') OR EXISTS (SELECT t FROM s.tags t WHERE t in :qWs)) AND (s.creator = :account OR s.availability > 0)"),

54 @NamedQuery(name = "ShaderDemo.findAll", query = "SELECT s FROM ShaderDemo s"),

55 @NamedQuery(name = "ShaderDemo.findById", query = "SELECT s FROM ShaderDemo s WHERE s.id = :id"),

56 @NamedQuery(name = "ShaderDemo.findByTitle", query = "SELECT s FROM ShaderDemo s WHERE s.title = :title"),

57 @NamedQuery(name = "ShaderDemo.findByDateCreated", query = "SELECT s FROM ShaderDemo s WHERE s.dateCreated = :dateCreated"),

58 @NamedQuery(name = "ShaderDemo.findByCode", query = "SELECT s FROM ShaderDemo s WHERE s.code = :code"),

59 @NamedQuery(name = "ShaderDemo.findByAvailability", query = "SELECT s FROM ShaderDemo s WHERE s.availability = :availability")})

60 public class ShaderDemo implements Serializable {

61 private static final long serialVersionUID = 1L;

62 public static final int PRIVATE = 0;

63 public static final int PUBLIC = 1;

64

65 @Id

66 @GeneratedValue(strategy = GenerationType.IDENTITY)

67 @Basic(optional = false)

68 @Column(name = "ID")

69 private Integer id;

70 @Basic(optional = false)

71 @Column(name = "TITLE")

72 private String title;

73 @Basic(optional = false)

74 @Column(name = "DATE\_CREATED")

75 @Temporal(TemporalType.DATE)

76 private Date dateCreated;

77 @Basic(optional = false)

78 @Column(name = "CODE")

79 private String code;

80 @Basic(optional = false)

81 @Column(name = "AVAILABILITY")

82 private int availability;

83 @ManyToMany(mappedBy = "delights", fetch = FetchType.EAGER)

84 private Set<Account> delighted;

85 @ManyToMany(mappedBy = "visits", fetch = FetchType.EAGER)

86 private Set<Account> visitors;

87 @ElementCollection(fetch = FetchType.EAGER)

88 @CollectionTable(

89 name="TAG",

90 joinColumns=@JoinColumn(name="SHADER\_DEMO")

91 )

92 @Column(name="TAG\_NAME")

93 private Set<String> tags;

94 @JoinColumn(name = "CREATOR", referencedColumnName = "ID")

95 @ManyToOne(optional = false, fetch = FetchType.EAGER)

96 private Account creator;

97

98 public ShaderDemo() {

99 }

100

101 public ShaderDemo(Integer id) {

102 this.id = id;

103 }

104

105 public ShaderDemo(Integer id, String title, Date dateCreated, String code, int availability) {

106 this.id = id;

107 this.title = title;

108 this.dateCreated = dateCreated;

109 this.code = code;

110 this.availability = availability;

111 }

112

113

114

115 @Override

116 public boolean equals(Object object) {

117 // TODO: Warning - this method won't work in the case the id fields are not set

118 if (!(object instanceof ShaderDemo)) {

119 return false;

120 }

121 ShaderDemo other = (ShaderDemo) object;

122 if ((this.getId() == null && other.getId() != null) || (this.getId() != null && !this.id.equals(other.id))) {

123 return false;

124 }

125 return true;

126 }

127

128 @Override

129 public String toString() {

130 return getTitle();

131 }

132

133 /\*\*

134 \* @return the id

135 \*/

136 public Integer getId() {

137 return id;

138 }

139

140 /\*\*

141 \* @param id the id to set

142 \*/

143 public void setId(Integer id) {

144 this.id = id;

145 }

146

147 /\*\*

148 \* @return the title

149 \*/

150 public String getTitle() {

151 return title;

152 }

153

154 /\*\*

155 \* @param title the title to set

156 \*/

157 public void setTitle(String title) {

158 this.title = title;

159 }

160

161 /\*\*

162 \* @return the creation date

163 \*/

164 public Date getDateCreated() {

165 return dateCreated;

166 }

167

168 /\*\*

169 \* @param dateCreated the creation date to set

170 \*/

171 public void setDateCreated(Date dateCreated) {

172 this.dateCreated = dateCreated;

173 }

174

175 /\*\*

176 \* @return the code

177 \*/

178 public String getCode() {

179 return code;

180 }

181

182 /\*\*

183 \* @param code the code to set

184 \*/

185 public void setCode(String code) {

186 this.code = code;

187 }

188

189 /\*\*

190 \* @return the availability

191 \*/

192 public int getAvailability() {

193 return availability;

194 }

195

196 /\*\*

197 \* @param availability the availability to set

198 \*/

199 public void setAvailability(int availability) {

200 this.availability = availability;

201 }

202

203 /\*\*

204 \* @return the set of accounts delighted by the shader

205 \*/

206 public Set<Account> getDelighted() {

207 return delighted;

208 }

209

210 /\*\*

211 \* @param delighted the set of accounts delighted by the shader to set

212 \*/

213 public void setDelighted(Set<Account> delighted) {

214 this.delighted = delighted;

215 }

216

217 /\*\*

218 \* @return the visitors to the shader

219 \*/

220 public Set<Account> getVisitors() {

221 return visitors;

222 }

223

224 /\*\*

225 \* @param visitors the visitors to set

226 \*/

227 public void setVisitors(Set<Account> visitors) {

228 this.visitors = visitors;

229 }

230

231 /\*\*

232 \* @return the tags

233 \*/

234 public Set<String> getTags() {

235 return tags;

236 }

237

238 /\*\*

239 \* @param tags the tags to set

240 \*/

241 public void setTags(Set<String> tags) {

242 this.tags = tags;

243 }

244

245 /\*\*

246 \* @return the creator

247 \*/

248 public Account getCreator() {

249 return creator;

250 }

251

252 /\*\*

253 \* @param creator the creator to set

254 \*/

255 public void setCreator(Account creator) {

256 this.creator = creator;

257 }

258

259 }

260

|  |
| --- |
| FancyShaderLib\src\fancyshader\service\FancyShaderInternalException.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.service;

18

19 /\*\*

20 \* An exception thrown by FancyShader service interfaces when an internal server error occurs so that internal server state is not leaked.

21 \* @author Reuben Steenekamp

22 \*/

23 public class FancyShaderInternalException extends Exception {}

24

|  |
| --- |
| FancyShaderLib\src\fancyshader\service\FancyShaderLoginException.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.service;

18

19 /\*\*

20 \* An exception thrown in the {@code login} methods of FancyShader service interfaces when the login details provided are incorrect

21 \* @author Reuben Steenekamp

22 \*/

23 public class FancyShaderLoginException extends Exception {}

24

|  |
| --- |
| FancyShaderLib\src\fancyshader\service\FancyShaderRemoteService.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.service;

18

19 import fancyshader.entity.Account;

20 import fancyshader.entity.ShaderDemo;

21 import java.rmi.Remote;

22 import java.rmi.RemoteException;

23 import java.util.List;

24

25 /\*\*

26 \* Remote interface paralleling {@link FancyShaderService}

27 \* @author Reuben Steenekamp

28 \*/

29 public interface FancyShaderRemoteService extends Remote {

30 public String login(String username, String password) throws FancyShaderLoginException, FancyShaderInternalException, RemoteException;

31 public void logout(String sessionUUID) throws RemoteException;

32 public void registerAccount(String username, String password) throws FancyShaderInternalException, RemoteException;

33

34

35 public Account getAccount(String sessionUUID) throws FancyShaderSessionExpiredException, RemoteException;

36 public List<ShaderDemo> search(String sessionUUID, String q) throws FancyShaderSessionExpiredException, RemoteException;

37

38 public ShaderDemo create(String sessionUUID, ShaderDemo source) throws FancyShaderSessionExpiredException, RemoteException;

39

40 public void update(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException, RemoteException;

41

42 public void delightful(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException, RemoteException;

43 public void visit(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException, RemoteException;

44

45 public boolean isUsernameTaken(String username) throws FancyShaderInternalException, RemoteException;

46

47 public void delete(String uuid, ShaderDemo shaderDemo) throws FancyShaderSessionExpiredException, RemoteException;

48

49 public boolean isDemoTitleTaken(String title) throws FancyShaderInternalException, RemoteException;

50

51 public boolean isDelighted(String sessionUUID, ShaderDemo value) throws FancyShaderSessionExpiredException, RemoteException;

52

53 }

54

|  |
| --- |
| FancyShaderLib\src\fancyshader\service\FancyShaderService.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.service;

18

19 import fancyshader.entity.Account;

20 import fancyshader.entity.ShaderDemo;

21 import java.util.List;

22

23 /\*\*

24 \* Interface providing methods for Client-Server communication in FancyShader.

25 \* @author Reuben Steenekamp

26 \*/

27 public interface FancyShaderService {

28 /\*\*

29 \* Establish a session with the respective {@link fancyshader.entity.Account} as the owner

30 \* @param username the account username

31 \* @param password the account password

32 \* @return the session UUID of the established session

33 \* @throws FancyShaderLoginException

34 \* @throws FancyShaderInternalException

35 \*/

36 public String login(String username, String password) throws FancyShaderLoginException, FancyShaderInternalException;

37

38 /\*\*

39 \* Invalidates a session

40 \* @param sessionUUID session UUID of the session to invalidate

41 \*/

42 public void logout(String sessionUUID);

43

44 /\*\*

45 \* Get the owner of the session identified by the {@code sessionUUID}

46 \* @param sessionUUID the session UUID of the session from which the owner is retrieved

47 \* @return the account with which the session was created

48 \* @throws FancyShaderSessionExpiredException

49 \*/

50 public Account getAccount(String sessionUUID) throws FancyShaderSessionExpiredException;

51

52 /\*\*

53 \* Create a new account with the {@code username} and {@code password} provided.

54 \* @param username the username to use

55 \* @param password the password to use

56 \* @throws FancyShaderInternalException

57 \*/

58 public void registerAccount(String username, String password) throws FancyShaderInternalException;

59

60 /\*\*

61 \* Search all the {@link fancyshader.entity.ShaderDemo} demos in the context

62 \* of the session identified by {@code sessionUUID} with {@code query} as the search parameter.

63 \* @param sessionUUID the sessionUUID of session providing context for the search

64 \* @param query the search parameter

65 \* @return the result list of {@link fancyshader.entity.ShaderDemo} demos

66 \* @throws FancyShaderSessionExpiredException

67 \*/

68 public List<ShaderDemo> search(String sessionUUID, String query) throws FancyShaderSessionExpiredException;

69

70 /\*\*

71 \* Create a new {@link fancyshader.entity.ShaderDemo} demo, copying relevant fields from {@code source}.

72 \* Demos with {@code accessability} of greater than {@code fancyshader.entity.ShaderDemo.PRIVATE} will only

73 \* appear in the search if their creator is the owner of the session.

74 \* @param sessionUUID the session UUID of the session providing the context for the creation of the {@link fancyshader.entity.ShaderDemo}

75 \* @param source the {@link fancyshader.entity.ShaderDemo} from which the fields are copied

76 \* @return the {@link fancyshader.entity.ShaderDemo} created

77 \* @throws FancyShaderSessionExpiredException

78 \*/

79 public ShaderDemo create(String sessionUUID, ShaderDemo source) throws FancyShaderSessionExpiredException;

80

81 /\*\*

82 \* Update an existing {@link fancyshader.entity.ShaderDemo} demo, copying relevant fields from {@code source}.

83 \* Will do nothing if the owner of the session is not the creator of the demo.

84 \* @param sessionUUID the session UUID of the session providing the context for the updating of the {@link fancyshader.entity.ShaderDemo}

85 \* @param demo the {@link fancyshader.entity.ShaderDemo} from which the fields are copied and the {@code id} of the demo to update is obtained.

86 \* @throws FancyShaderSessionExpiredException

87 \*/

88 public void update(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException;

89

90 /\*\*

91 \* Delete an existing {@link fancyshader.entity.ShaderDemo} demo.

92 \* Will do nothing if the owner of the session is not the creator of the demo.

93 \* @param uuid the session UUID of the session providing the context for the deletion of the {@link fancyshader.entity.ShaderDemo}

94 \* @param demo the {@link fancyshader.entity.ShaderDemo} to be deleted

95 \* @throws FancyShaderSessionExpiredException

96 \*/

97 public void delete(String uuid, ShaderDemo demo) throws FancyShaderSessionExpiredException;

98

99 /\*\*

100 \* Toggle {@code demo} as delightful for the owner of the session identified by {@code sessionUUID}.

101 \* If {@code state} is true then {@code demo} will be marked as delightful, else {@code demo} will be unmarked as delightful.

102 \* @param sessionUUID the session UUID of the session providing the context for the operation

103 \* @param demo the demo to mark or unmark

104 \* @throws FancyShaderSessionExpiredException

105 \*/

106 public void delightful(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException;

107

108 /\*\*

109 \* Get whether a given demo is marked as delightful for the owner of the session identified by {@code sessionUUID}

110 \* @param sessionUUID the session UUID of the session providing the context for the operation

111 \* @param demo the demo that is checked

112 \* @return whether {@code demo} is marked as delightful

113 \* @throws FancyShaderSessionExpiredException

114 \*/

115 public boolean isDelightful(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException;

116

117 /\*\*

118 \* Marks {@code demo} as visited by the owner of the session identified by {@code sessionUUID}. The operation is idempotent.

119 \* @param sessionUUID the session UUID of the session providing the context for the operation

120 \* @param demo the demo to mark

121 \* @throws FancyShaderSessionExpiredException

122 \*/

123 public void visit(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException;

124

125 /\*\*

126 \* Test whether an {@link fancyshader.entity.Account} with a username {@code username} exists.

127 \* @param username the username to test

128 \* @return whether such an {@link fancyshader.entity.Account} exists

129 \* @throws FancyShaderInternalException

130 \*/

131 public boolean isUsernameTaken(String username) throws FancyShaderInternalException;

132

133 /\*\*

134 \* Test whether a {@link fancyshader.entity.ShaderDemo} with a title {@code title} exists.

135 \* @param title the title to test

136 \* @return whether such a {@link fancyshader.entity.ShaderDemo} exists.

137 \* @throws FancyShaderInternalException

138 \*/

139 public boolean isDemoTitleTaken(String title) throws FancyShaderInternalException;

140

141

142 }

143

|  |
| --- |
| FancyShaderLib\src\fancyshader\service\FancyShaderSessionExpiredException.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.service;

18

19 /\*\*

20 \* An exception thrown by FancyShader service interfaces when a provided session UUID has expired

21 \* @author Reuben Steenekamp

22 \*/

23 public class FancyShaderSessionExpiredException extends Exception {

24

25 public FancyShaderSessionExpiredException() {

26 }

27

28 }

29

|  |
| --- |
| FancyShaderLib\src\META-INF\persistence.xml |

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

2 <persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.org/xml/ns/persistence/persistence\_2\_1.xsd">

3 <persistence-unit name="FancyShaderPU" transaction-type="RESOURCE\_LOCAL">

4 <provider>org.eclipse.persistence.jpa.PersistenceProvider</provider>

5 <class>fancyshader.entity.Account</class>

6 <class>fancyshader.entity.AuthDetail</class>

7 <class>fancyshader.entity.ShaderDemo</class>

8 <properties>

9 <property name="javax.persistence.jdbc.url" value="jdbc:derby:../db/FancyShaderDB;create=true"/>

10 <property name="javax.persistence.jdbc.driver" value="org.apache.derby.jdbc.EmbeddedDriver"/>

11 <property name="javax.persistence.jdbc.user" value="root"/>

12 <property name="javax.persistence.jdbc.password" value="root"/>

13 </properties>

14 </persistence-unit>

15 </persistence>

16

|  |
| --- |
| FancyShaderServer\src\fancyshader\server\FancyShaderRemoteServiceImpl.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.server;

18

19 import fancyshader.entity.Account;

20 import fancyshader.entity.ShaderDemo;

21 import fancyshader.service.FancyShaderInternalException;

22 import fancyshader.service.FancyShaderLoginException;

23 import fancyshader.service.FancyShaderRemoteService;

24 import fancyshader.service.FancyShaderService;

25 import fancyshader.service.FancyShaderSessionExpiredException;

26 import java.rmi.RemoteException;

27 import java.rmi.server.UnicastRemoteObject;

28 import java.util.List;

29

30 /\*\*

31 \* Implementation of {@link fancyshader.service.FancyShaderRemoteService} that delegates

32 \* to an instance of {@link fancyshader.service.FancyShaderService}

33 \* @author Reuben Steenekamp

34 \*/

35 public class FancyShaderRemoteServiceImpl extends UnicastRemoteObject implements FancyShaderRemoteService {

36 private FancyShaderService service;

37 public FancyShaderRemoteServiceImpl() throws RemoteException {

38 super(0);

39 service = new FancyShaderServiceImpl();

40 }

41 public FancyShaderRemoteServiceImpl(FancyShaderService service) throws RemoteException {

42 super(0);

43 this.service = service;

44 }

45

46 @Override

47 public String login(String username, String password) throws FancyShaderLoginException, FancyShaderInternalException, RemoteException {

48 return service.login(username, password);

49 }

50

51 @Override

52 public void logout(String sessionUUID) throws RemoteException {

53 service.logout(sessionUUID);

54 }

55

56 @Override

57 public void registerAccount(String username, String password) throws FancyShaderInternalException, RemoteException {

58 service.registerAccount(username, password);

59 }

60

61 @Override

62 public Account getAccount(String sessionUUID) throws FancyShaderSessionExpiredException, RemoteException {

63 return service.getAccount(sessionUUID);

64 }

65

66 @Override

67 public List<ShaderDemo> search(String sessionUUID, String q) throws FancyShaderSessionExpiredException, RemoteException {

68 return service.search(sessionUUID, q);

69 }

70

71 @Override

72 public ShaderDemo create(String sessionUUID, ShaderDemo source) throws FancyShaderSessionExpiredException, RemoteException {

73 return service.create(sessionUUID, source);

74 }

75

76 @Override

77 public void update(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException, RemoteException {

78 service.update(sessionUUID, demo);

79 }

80

81 @Override

82 public void delightful(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException, RemoteException {

83 service.delightful(sessionUUID, demo);

84 }

85

86 @Override

87 public void visit(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException, RemoteException {

88 service.visit(sessionUUID, demo);

89 }

90

91 @Override

92 public boolean isUsernameTaken(String username) throws FancyShaderInternalException, RemoteException {

93 return service.isUsernameTaken(username);

94 }

95

96 @Override

97 public void delete(String uuid, ShaderDemo shaderDemo) throws FancyShaderSessionExpiredException, RemoteException {

98 service.delete(uuid, shaderDemo);

99 }

100

101 @Override

102 public boolean isDemoTitleTaken(String title) throws FancyShaderInternalException, RemoteException {

103 return service.isDemoTitleTaken(title);

104 }

105

106 @Override

107 public boolean isDelighted(String sessionUUID, ShaderDemo value) throws FancyShaderSessionExpiredException, RemoteException {

108 return service.isDelightful(sessionUUID, value);

109 }

110 }

111

|  |
| --- |
| FancyShaderServer\src\fancyshader\server\FancyShaderServer.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.server;

18

19 import java.rmi.RemoteException;

20 import java.rmi.registry.LocateRegistry;

21 import java.rmi.registry.Registry;

22 import java.rmi.server.RemoteServer;

23 import java.util.logging.Level;

24 import java.util.logging.Logger;

25 import javax.persistence.PersistenceException;

26

27 /\*\*

28 \* The entry point for the server

29 \* @author Reuben Steenekamp

30 \*/

31

32 public class FancyShaderServer {

33 public static void main(String[] args) throws Exception{

34 try {

35 int PORT = 1099;

36

37 Registry registry = LocateRegistry.createRegistry(PORT);

38 RemoteServer server = new FancyShaderRemoteServiceImpl();

39 registry.rebind("FancyShaderRemoteService", server);

40 } catch (PersistenceException | RemoteException e) {

41 Logger.getLogger(FancyShaderServer.class.getName()).log(Level.SEVERE, null, e);

42 javax.swing.JOptionPane.showMessageDialog(null, e.toString());

43 System.exit(1);

44 }

45 }

46 }

47

|  |
| --- |
| FancyShaderServer\src\fancyshader\server\FancyShaderServiceImpl.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.server;

18

19 import fancyshader.entity.Account;

20 import fancyshader.entity.AuthDetail;

21 import fancyshader.entity.ShaderDemo;

22 import fancyshader.service.FancyShaderInternalException;

23 import fancyshader.service.FancyShaderLoginException;

24 import fancyshader.service.FancyShaderService;

25 import fancyshader.service.FancyShaderSessionExpiredException;

26 import java.security.MessageDigest;

27 import java.security.NoSuchAlgorithmException;

28 import java.security.SecureRandom;

29 import java.util.Arrays;

30 import java.util.Collections;

31 import java.util.Date;

32 import java.util.HashSet;

33 import java.util.List;

34 import java.util.Set;

35 import java.util.logging.Level;

36 import java.util.logging.Logger;

37 import javax.persistence.EntityManager;

38 import javax.persistence.EntityTransaction;

39 import javax.persistence.NoResultException;

40 import javax.persistence.Persistence;

41 import javax.persistence.TypedQuery;

42

43 /\*\*

44 \* Implementation of {@link fancyshader.service.FancyShaderService} using JPA to connect to FancyShaderDB

45 \*

46 \* @author Reuben Steenekamp

47 \*/

48 public class FancyShaderServiceImpl implements FancyShaderService {

49

50 EntityManager entityManager = Persistence.createEntityManagerFactory("FancyShaderPU").createEntityManager();

51

52 @Override

53 public String login(String username, String password) throws FancyShaderLoginException, FancyShaderInternalException {

54 try {

55 //Obtain the corresponding Account with the input username

56 TypedQuery<Account> accountQuery = entityManager.createNamedQuery("Account.findByUsername", Account.class)

57 .setParameter("username", username);

58 Account account = accountQuery.getSingleResult();

59

60 // The AuthDetail in Account is not fetched automatically by JPA to avoid scrubbing Account of the AuthDetail later on when

61 // exposing Account through the FancyShaderService. We do not want users to have free access to password hashes of other users,

62 // in the event that they chose easy to guess passwords.

63 AuthDetail detail = entityManager.find(AuthDetail.class, account.getAuthDetail());

64

65 String hash = getHash(password + detail.getSalt()); //Get the hash of the salted input password

66 if (hash.equals(detail.getHashcode())) { //Check if the hash of the salted input password is the same as the hash stored in the AuthDetail

67 return FancyShaderSession.create(account).getUUID(); //Create a session and return the UUID

68 } else {

69 throw new FancyShaderLoginException(); //The passwords do not match

70 }

71 } catch (NoResultException e) {

72 throw new FancyShaderLoginException(); //There is no Account with the input username

73 }

74 }

75

76 @Override

77 public void logout(String sessionUUID) {

78 FancyShaderSessionSingleton.instance.invalidateSession(sessionUUID);

79 }

80

81 @Override

82 public void registerAccount(String username, String password) throws FancyShaderInternalException {

83 String salt = getSalt();

84 String hash = getHash(password + salt); //Get the hash of the salted password

85

86 //Initialize the Account, except its AuthDetail id which is yet to be established

87 Account account = new Account();

88 account.setUsername(username);

89 account.setDateCreated(new Date());

90 account.setDelights(new HashSet());

91 account.setVisits(new HashSet());

92

93 //Initialize the AuthDetail

94 AuthDetail detail = new AuthDetail();

95 detail.setHashcode(hash);

96 detail.setSalt(salt);

97

98 EntityTransaction transaction = entityManager.getTransaction();

99 transaction.begin();

100 try {

101 entityManager.persist(detail); //Add the AuthDetail to the entity graph

102 entityManager.flush(); //Ensure that all operations run so that detail is on the entity graph before proceeding

103 account.setAuthDetail(detail.getId()); //Initialize the AuthDetail id of the Account

104 entityManager.persist(account); //Add the Account to the entity graph

105 if (transaction.isActive()) {

106 transaction.commit();

107 }

108 } catch (Exception e) {

109 Logger.getLogger(this.getClass().getName()).log(Level.WARNING, null, e);

110 if (transaction.isActive()) {

111 transaction.rollback();

112 }

113 throw new FancyShaderInternalException();

114 }

115

116 }

117

118 /\*\*

119 \* Obtains a cryptographically random salt

120 \*

121 \* @return the salt

122 \*/

123 private static String getSalt() {

124 SecureRandom random = new SecureRandom();

125 byte[] saltBytes = new byte[10];

126 random.nextBytes(saltBytes);

127 return new String(saltBytes);

128 }

129

130 /\*\*

131 \* Obtains the SHA-1 hash of the {@code value}

132 \*

133 \* @param value the item to be hashed

134 \* @return the hash

135 \* @throws FancyShaderInternalException

136 \*/

137 private static String getHash(String value) throws FancyShaderInternalException {

138 MessageDigest hasher;

139 try {

140 hasher = MessageDigest.getInstance("SHA-1");

141 } catch (NoSuchAlgorithmException ex) {

142 Logger.getLogger(FancyShaderRemoteServiceImpl.class.getName()).log(Level.SEVERE, null, ex);

143 throw new FancyShaderInternalException();

144 }

145 return new String(hasher.digest(value.getBytes()));

146 }

147

148 @Override

149 public List<ShaderDemo> search(String sessionUUID, String q) throws FancyShaderSessionExpiredException {

150 Account account = getAccount(sessionUUID);

151

152 //Get the lowercase words of the search query

153 List<String> qWs = Arrays.asList(q.toLowerCase().split(" "));

154 qWs.replaceAll(String::trim);

155

156 //Prepare the query

157 TypedQuery<ShaderDemo> searchQuery = entityManager.createNamedQuery("ShaderDemo.search", ShaderDemo.class)

158 .setParameter("q", q)

159 .setParameter("qWs", qWs)

160 .setParameter("account", account);

161

162 return searchQuery.getResultList();

163 }

164

165 @Override

166 public ShaderDemo create(String sessionUUID, ShaderDemo source) throws FancyShaderSessionExpiredException {

167 Account account = getAccount(sessionUUID);

168

169 //Initialize the ShaderDemo

170 ShaderDemo shaderDemo = new ShaderDemo();

171 shaderDemo.setTitle(source.getTitle());

172 shaderDemo.setCreator(account);

173 shaderDemo.setDateCreated(new Date());

174 shaderDemo.setCode(source.getCode() == null ? "" : source.getCode());

175 shaderDemo.setTags(canonicalizeTags(source.getTags()));

176 shaderDemo.setAvailability(source.getAvailability());

177

178 //Add the ShaderDemo to the entity graph

179 EntityTransaction transaction = entityManager.getTransaction();

180 transaction.begin();

181 entityManager.persist(shaderDemo);

182 if (transaction.isActive()) {

183 transaction.commit();

184 }

185

186 return shaderDemo;

187 }

188

189 @Override

190 public void update(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException {

191 Account account = getAccount(sessionUUID);

192

193 //Obtain the corresponding managed entity with the id of the inputted ShaderDemo

194 ShaderDemo managed = entityManager.find(ShaderDemo.class, demo.getId());

195 if (!managed.getCreator().equals(account)) {

196 return; //The owner of the session is not the creator of the ShaderDemo and therefore cannot update it

197 }

198

199 //Update the managed entity's fields

200 managed.setTitle(demo.getTitle());

201 managed.setCode(demo.getCode());

202 managed.setTags(canonicalizeTags(demo.getTags()));

203 managed.setAvailability(demo.getAvailability());

204

205 //Commit the changes

206 EntityTransaction transaction = entityManager.getTransaction();

207 transaction.begin();

208 entityManager.persist(managed);

209 if (transaction.isActive()) {

210 transaction.commit();

211 }

212 }

213

214 /\*\*

215 \* Change the given tags into a canonical form

216 \*

217 \* @param tags the tags to canonicalize

218 \* @return the canonical tags

219 \*/

220 private Set<String> canonicalizeTags(Set<String> tags) {

221 if (tags == null) {

222 tags = Collections.EMPTY\_SET;

223 }

224 Set<String> canonicalTags = new HashSet<>();

225 for (String tag : tags) {

226 canonicalTags.add(tag.toLowerCase());

227 }

228 return canonicalTags;

229 }

230

231 @Override

232 public void delightful(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException {

233 Account account = getAccount(sessionUUID);

234

235 //Obtain the corresponding managed entity with the id of the inputted ShaderDemo

236 ShaderDemo managed = entityManager.find(ShaderDemo.class, demo.getId());

237

238 //Toggle the demo's delightful status

239 if (account.getDelights().contains(managed)) {

240 account.getDelights().remove(managed); //The demo is marked and must be unmarked

241 } else {

242 account.getDelights().add(managed); //The demo is unmarked and must be marked

243 }

244 //Commit the changes

245 EntityTransaction transaction = entityManager.getTransaction();

246 transaction.begin();

247 entityManager.persist(account);

248 if (transaction.isActive()) {

249 transaction.commit();

250 }

251 }

252

253 @Override

254 public void visit(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException {

255 Account account = getAccount(sessionUUID);

256

257 //Obtain the corresponding managed entity with the id of the inputted ShaderDemo

258 ShaderDemo managed = entityManager.find(ShaderDemo.class, demo.getId());

259

260 //Add visit to account

261 if (!account.getVisits().contains(managed)) {

262 account.getVisits().add(managed);

263 }

264

265 //Commit the changes

266 EntityTransaction transaction = entityManager.getTransaction();

267 transaction.begin();

268 entityManager.persist(account);

269 if (transaction.isActive()) {

270 transaction.commit();

271 }

272 }

273

274 @Override

275 public Account getAccount(String sessionUUID) throws FancyShaderSessionExpiredException {

276 //Get the Account stored in the session

277 Account sessionAccount = FancyShaderSessionSingleton.instance.getSession(sessionUUID).getAccount();

278

279 //Obtain the corresponding managed entity with the id of the session Account

280 return entityManager.find(Account.class, sessionAccount.getId());

281 }

282

283 @Override

284 public void delete(String sessionUUID, ShaderDemo shaderDemo) throws FancyShaderSessionExpiredException {

285 //Obtain the corresponding managed entity with the id of the inputted ShaderDemo

286 ShaderDemo managed = entityManager.find(ShaderDemo.class, shaderDemo.getId());

287

288 if (!managed.getCreator().equals(getAccount(sessionUUID))) {

289 return; //The owner of the session is not the creator of the demo and therefore cannot delete it

290 }

291 //Remove the demo

292 EntityTransaction transaction = entityManager.getTransaction();

293 transaction.begin();

294 entityManager.remove(managed);

295 if (transaction.isActive()) {

296 transaction.commit();

297 }

298 }

299

300 @Override

301 public boolean isUsernameTaken(String username) throws FancyShaderInternalException {

302 try {

303 //Attempt to obtain the corresponding Account with the inputted username

304 entityManager.createNamedQuery("Account.findByUsername").setParameter("username", username).getSingleResult();

305

306 //No error occurred. The Account exists and the username is taken.

307 return true;

308 } catch (NoResultException e) {

309 //No result obtained. The Account does not exist and the username is available.

310 return false;

311 } catch (Exception e) {

312 //Some other error occured

313 throw new FancyShaderInternalException();

314 }

315 }

316

317 @Override

318 public boolean isDemoTitleTaken(String title) throws FancyShaderInternalException {

319 try {

320 //Attempt to obtain the corresponding ShaderDemo with the inputted title

321 entityManager.createNamedQuery("ShaderDemo.findByTitle").setParameter("title", title).getSingleResult();

322

323 //No error occurred. The ShaderDemo exists and the title is taken.

324 return true;

325 } catch (NoResultException e) {

326 //No result obtained. The ShaderDemo does not exist and the title is available.

327 return false;

328 } catch (Exception e) {

329 //Some other error occured

330 throw new FancyShaderInternalException();

331 }

332 }

333

334 @Override

335 public boolean isDelightful(String sessionUUID, ShaderDemo demo) throws FancyShaderSessionExpiredException {

336 if (demo == null)

337 return false;

338 Account account = getAccount(sessionUUID);

339

340 //Obtain the corresponding managed entity with the id of the inputted ShaderDemo

341 ShaderDemo managed = entityManager.find(ShaderDemo.class, demo.getId());

342

343 //Return whether the demo is contained in the owner of the sessions's delights

344 return account.getDelights().contains(managed);

345 }

346 }

347

|  |
| --- |
| FancyShaderServer\src\fancyshader\server\FancyShaderSession.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.server;

18

19 import fancyshader.entity.Account;

20 import java.io.Serializable;

21 import java.util.Date;

22 import java.util.UUID;

23

24 /\*\*

25 \* Immutable object class encapsulating session information.

26 \* @author Reuben Steenekamp

27 \*/

28 public class FancyShaderSession implements Serializable {

29 public static final long EXPIRE = 2\*7\*24\*60\*60; // 2 weeks in seconds;

30

31 public static FancyShaderSession create(Account account) {

32 FancyShaderSession session = new FancyShaderSession(account);

33 FancyShaderSessionSingleton.instance.putSession(session);

34 return session;

35 }

36

37 private Account account;

38 private String uuid;

39 private Date creationDate;

40

41 private FancyShaderSession(Account account){

42 this.uuid = UUID.randomUUID().toString();

43 this.creationDate = new Date();

44 this.account = account;

45 }

46

47

48 /\*\*

49 \* @return whether the session has expired

50 \*/

51 public boolean hasExpired() {

52 return (new Date()).toInstant().getEpochSecond()-getCreationDate().toInstant().getEpochSecond() > EXPIRE;

53 }

54

55 /\*\*

56 \* @return the account

57 \*/

58 public Account getAccount() {

59 return account;

60 }

61

62 /\*\*

63 \* @return the UUID

64 \*/

65 public String getUUID() {

66 return uuid;

67 }

68

69 /\*\*

70 \* @return the creation date

71 \*/

72 public Date getCreationDate() {

73 return creationDate;

74 }

75 }

76

|  |
| --- |
| FancyShaderServer\src\fancyshader\server\FancyShaderSessionSingleton.java |

1 /\*

2 \* Copyright (C) 2014 Reuben Steenekamp

3 \*

4 \* This program is free software: you can redistribute it and/or modify

5 \* it under the terms of the GNU General Public License as published by

6 \* the Free Software Foundation, either version 3 of the License, or

7 \* (at your option) any later version.

8 \*

9 \* This program is distributed in the hope that it will be useful,

10 \* but WITHOUT ANY WARRANTY; without even the implied warranty of

11 \* MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

12 \* GNU General Public License for more details.

13 \*

14 \* You should have received a copy of the GNU General Public License

15 \* along with this program. If not, see <http://www.gnu.org/licenses/>.

16 \*/

17 package fancyshader.server;

18

19 import fancyshader.service.FancyShaderSessionExpiredException;

20 import java.util.HashMap;

21 import java.util.Iterator;

22 import java.util.Map;

23

24 /\*\*

25 \* Singleton flyweight of user sessions.

26 \* @author Reuben Steenekamp

27 \*/

28 public class FancyShaderSessionSingleton {

29 public static final FancyShaderSessionSingleton instance = new FancyShaderSessionSingleton();

30

31 private Map<String, FancyShaderSession> sessions;

32

33 private FancyShaderSessionSingleton() {

34 sessions = new HashMap<>();

35 }

36

37 /\*\*

38 \* @param uuid the UUID of the session to get from the flyweight

39 \* @return the corresponding session

40 \* @throws FancyShaderSessionExpiredException when the session has expired or never existed

41 \*/

42 public FancyShaderSession getSession(String uuid) throws FancyShaderSessionExpiredException {

43 reviseSessions();

44 if(sessions.containsKey(uuid))

45 return sessions.get(uuid);

46 else

47 throw new FancyShaderSessionExpiredException();

48 }

49

50 /\*\*

51 \* Add a session to the flyweight

52 \* @param session the session to add to the flyweight

53 \*/

54 public void putSession(FancyShaderSession session) {

55 sessions.put(session.getUUID(), session);

56 }

57

58 private void reviseSessions() {

59 Iterator<String> it = sessions.keySet().iterator();

60 while(it.hasNext()) {

61 String uuid = it.next();

62 if(sessions.get(uuid).hasExpired()) invalidateSession(uuid);

63 }

64 }

65

66 /\*\*

67 \* Remove the session corresponding to to given UUID from the flyweight, rendering it expired.

68 \* @param sessionUUID

69 \*/

70 public void invalidateSession(String sessionUUID) {

71 sessions.remove(sessionUUID);

72 }

73

74 }

75

|  |
| --- |
| FancyShaderServer\fancyshaderdb.sql |

1 create table shader\_demo (

2 id int not null generated always as identity,

3 title varchar(50) not null unique,

4 creator int not null,

5 date\_created date not null,

6 code varchar(20000) not null,

7 availability int not null

8 );

9 alter table shader\_demo add constraint shader\_demo\_pk primary key (id);

10

11 create table account (

12 id int not null generated always as identity,

13 auth\_detail int not null unique,

14 username varchar(20) not null unique,

15 date\_created date not null

16 );

17 alter table account add constraint account\_pk primary key (id);

18 alter table shader\_demo add constraint shader\_demo2account\_fk foreign key (creator) references account (id) on delete cascade;

19

20 create table auth\_detail (

21 id int not null generated always as identity,

22 hashcode char(20) not null,

23 salt char(10) not null

24 );

25 alter table auth\_detail add constraint auth\_detail\_pk primary key (id);

26 alter table account add constraint account2auth\_detail\_fk foreign key (auth\_detail) references auth\_detail (id)

27 on delete cascade;

28

29 create table visit (

30 account int not null,

31 shader\_demo int not null

32 );

33 alter table visit add constraint visit\_pk primary key (account, shader\_demo);

34 alter table visit add constraint visit2account\_fk foreign key (account) references account (id) on delete cascade;

35 alter table visit add constraint visit2shader\_demo\_fk foreign key (shader\_demo) references shader\_demo (id) on delete cascade;

36

37 create table delightful (

38 account int not null,

39 shader\_demo int not null

40 );

41 alter table delightful add constraint delightful\_pk primary key (account, shader\_demo);

42 alter table delightful add constraint delightful2account\_fk foreign key (account) references account (id) on delete cascade;

43 alter table delightful add constraint delightful2shader\_demo\_fk foreign key (shader\_demo) references shader\_demo (id) on delete cascade;

44

45 create table tag (

46 shader\_demo int not null,

47 tag\_name varchar(30) not null

48 );

49 alter table tag add constraint tag\_pk primary key (shader\_demo, tag\_name);

50 alter table tag add constraint tag2shader\_demo\_fk foreign key (shader\_demo) references shader\_demo(id)

51 on delete cascade;

## Generated Javadocs

Example Snippet (Source code available at FancyShaderLib\src\fancyshader\service\FancyShaderService.java):

Interface FancyShaderService

[**fancyshader.service**](#b88)

public interface **FancyShaderService**

Interface providing methods for Client-Server communication in FancyShader.

|  |  |  |
| --- | --- | --- |
| Method Summary | | **Page** |
| [ShaderDemo](#b76) | [**create**](#b112)(String sessionUUID, [ShaderDemo](#b76) source)  Create a new [ShaderDemo](#b76) demo, copying relevant fields from source. |  |
| void | [**delete**](#b114)(String uuid, [ShaderDemo](#b76) demo)  Delete an existing [ShaderDemo](#b76) demo. |  |
| void | [**delightful**](#b115)(String sessionUUID, [ShaderDemo](#b76) demo)  Toggle demo as delightful for the owner of the session identified by sessionUUID. |  |
| [Account](#b23) | [**getAccount**](#b109)(String sessionUUID)  Get the owner of the session identified by the sessionUUID |  |
| boolean | [**isDelightful**](#b116)(String sessionUUID, [ShaderDemo](#b76) demo)  Get whether a given demo is marked as delightful for the owner of the session identified by sessionUUID |  |
| boolean | [**isDemoTitleTaken**](#b119)(String title)  Test whether a [ShaderDemo](#b76) with a title title exists. |  |
| boolean | [**isUsernameTaken**](#b118)(String username)  Test whether an [Account](#b23) with a username username exists. |  |
| String | [**login**](#b107)(String username, String password)  Establish a session with the respective [Account](#b23) as the owner |  |
| void | [**logout**](#b108)(String sessionUUID)  Invalidates a session |  |
| void | [**registerAccount**](#b110)(String username, String password)  Create a new account with the username and password provided. |  |
| List<[ShaderDemo](#b76)> | [**search**](#b111)(String sessionUUID, String query)  Search all the [ShaderDemo](#b76) demos in the context of the session identified by sessionUUID with query as the search parameter. |  |
| void | [**update**](#b113)(String sessionUUID, [ShaderDemo](#b76) demo)  Update an existing [ShaderDemo](#b76) demo, copying relevant fields from source. |  |
| void | [**visit**](#b117)(String sessionUUID, [ShaderDemo](#b76) demo)  Marks demo as visited by the owner of the session identified by sessionUUID. |  |

Method Detail

login

String **login**(String username,  
             String password)  
      throws [FancyShaderLoginException](#b92),  
             [FancyShaderInternalException](#b90)

Establish a session with the respective [Account](#b23) as the owner

**Parameters:**

username - the account username

password - the account password

**Returns:**

the session UUID of the established session

**Throws:**

[FancyShaderLoginException](#b92)

[FancyShaderInternalException](#b90)

logout

void **logout**(String sessionUUID)

Invalidates a session

**Parameters:**

sessionUUID - session UUID of the session to invalidate

getAccount

[Account](#b23) **getAccount**(String sessionUUID)  
            throws [FancyShaderSessionExpiredException](#b122)

Get the owner of the session identified by the sessionUUID

**Parameters:**

sessionUUID - the session UUID of the session from which the owner is retrieved

**Returns:**

the account with which the session was created

**Throws:**

[FancyShaderSessionExpiredException](#b122)

registerAccount

void **registerAccount**(String username,  
                     String password)  
              throws [FancyShaderInternalException](#b90)

Create a new account with the username and password provided.

**Parameters:**

username - the username to use

password - the password to use

**Throws:**

[FancyShaderInternalException](#b90)

search

List<[ShaderDemo](#b76)> **search**(String sessionUUID,  
                        String query)  
                 throws [FancyShaderSessionExpiredException](#b122)

Search all the [ShaderDemo](#b76) demos in the context of the session identified by sessionUUID with query as the search parameter.

**Parameters:**

sessionUUID - the sessionUUID of session providing context for the search

query - the search parameter

**Returns:**

the result list of [ShaderDemo](#b76) demos

**Throws:**

[FancyShaderSessionExpiredException](#b122)

create

[ShaderDemo](#b76) **create**(String sessionUUID,  
                  [ShaderDemo](#b76) source)  
           throws [FancyShaderSessionExpiredException](#b122)

Create a new [ShaderDemo](#b76) demo, copying relevant fields from source. Demos with accessability of greater than fancyshader.entity.ShaderDemo.PRIVATE will only appear in the search if their creator is the owner of the session.

**Parameters:**

sessionUUID - the session UUID of the session providing the context for the creation of the [ShaderDemo](#b76)

source - the [ShaderDemo](#b76) from which the fields are copied

**Returns:**

the [ShaderDemo](#b76) created

**Throws:**

[FancyShaderSessionExpiredException](#b122)

update

void **update**(String sessionUUID,  
            [ShaderDemo](#b76) demo)  
     throws [FancyShaderSessionExpiredException](#b122)

Update an existing [ShaderDemo](#b76) demo, copying relevant fields from source. Will do nothing if the owner of the session is not the creator of the demo.

**Parameters:**

sessionUUID - the session UUID of the session providing the context for the updating of the [ShaderDemo](#b76)

demo - the [ShaderDemo](#b76) from which the fields are copied and the id of the demo to update is obtained.

**Throws:**

[FancyShaderSessionExpiredException](#b122)

delete

void **delete**(String uuid,  
            [ShaderDemo](#b76) demo)  
     throws [FancyShaderSessionExpiredException](#b122)

Delete an existing [ShaderDemo](#b76) demo. Will do nothing if the owner of the session is not the creator of the demo.

**Parameters:**

uuid - the session UUID of the session providing the context for the deletion of the [ShaderDemo](#b76)

demo - the [ShaderDemo](#b76) to be deleted

**Throws:**

[FancyShaderSessionExpiredException](#b122)

delightful

void **delightful**(String sessionUUID,  
                [ShaderDemo](#b76) demo)  
         throws [FancyShaderSessionExpiredException](#b122)

Toggle demo as delightful for the owner of the session identified by sessionUUID. If state is true then demo will be marked as delightful, else demo will be unmarked as delightful.

**Parameters:**

sessionUUID - the session UUID of the session providing the context for the operation

demo - the demo to mark or unmark

**Throws:**

[FancyShaderSessionExpiredException](#b122)

isDelightful

boolean **isDelightful**(String sessionUUID,  
                     [ShaderDemo](#b76) demo)  
              throws [FancyShaderSessionExpiredException](#b122)

Get whether a given demo is marked as delightful for the owner of the session identified by sessionUUID

**Parameters:**

sessionUUID - the session UUID of the session providing the context for the operation

demo - the demo that is checked

**Returns:**

whether demo is marked as delightful

**Throws:**

[FancyShaderSessionExpiredException](#b122)

visit

void **visit**(String sessionUUID,  
           [ShaderDemo](#b76) demo)  
    throws [FancyShaderSessionExpiredException](#b122)

Marks demo as visited by the owner of the session identified by sessionUUID. The operation is idempotent.

**Parameters:**

sessionUUID - the session UUID of the session providing the context for the operation

demo - the demo to mark

**Throws:**

[FancyShaderSessionExpiredException](#b122)

isUsernameTaken

boolean **isUsernameTaken**(String username)  
                 throws [FancyShaderInternalException](#b90)

Test whether an [Account](#b23) with a username username exists.

**Parameters:**

username - the username to test

**Returns:**

whether such an [Account](#b23) exists

**Throws:**

[FancyShaderInternalException](#b90)

isDemoTitleTaken

boolean **isDemoTitleTaken**(String title)  
                  throws [FancyShaderInternalException](#b90)

Test whether a [ShaderDemo](#b76) with a title title exists.

**Parameters:**

title - the title to test

**Returns:**

whether such a [ShaderDemo](#b76) exists.

**Throws:**

[FancyShaderInternalException](#b90)

# Technical Documentation

Content

## External code

FancyShader uses the following libraries:

* EclipseLink
* JOGAMP
* Apache Derby

Code and diagrams were generated using the following tools:

* NetBeans
* PlantUML
* SQL Power Architect

All source code of FancyShader is written by Reuben Steenekamp or generated with 3rd party tools.

## Explanation of Critical Algorithms

### Rendering of Shader

#### Significance

Demos are the very reason that this program exists. This programs sole purpose is to view and edit shaders.

#### Process

* When the shader code is set in the **DemoViewer** 
  + The code is sent to the **DemoRunner**
  + **DemoRunner** compiles the code and attaches it to a shader program
  + If the compilation fails, then the error shader is attached instead
* The **DemoAnimator** uses **com.jogamp.opengl.util.Animator** which repeatedly calls **DemoRunner#**display(GLAutoDrawable drawable)
  + A check is performed to see if the shaders are compiled
    - If they are not then the method returns
  + The screen is cleared
  + The orientation and perspective of the camera is reset
  + The camera is shifted backwards
  + The program to be used is set to the shader program
  + The time variable of the shader program is set
  + The resolution variable of the shader program is set
  + A 1x1 quadralateral is draw at the origin
  + The program to be used is reset to default.

### Hashing and Salting of User Account Passwords

#### Significance

Passwords must not be stored on a database as plaintext. If the database is comprised, then all username, password pairs are exposed. While the service comprimised may not in itself be all that critical, many users tend to use the same usernames and passwords for other services such as banks. Thus it is important to use a one-way function, called a hash function, to obtain the hashcode of the password and store it on the database. If the database is compromised, then all that is exposed are the hashcodes.

Another problem is if users use common insecure passwords. Hackers can create a dictionary of hashcodes from commonly used passwords. If a hashcode in the database matches a hashcode in the dictionary, then the hackers know the password. This is solved by appending a random string, called a salt, to the password before hashing it and storing the salt along with the the hashcode. Hackers can no longer use a dictionary as all the the passwords have a random string appended to them before being hashed and are thus no longer predicable.

#### Process

* When **FancyShaderServiceImpl**#register(String username, String password) is called
  + a random 10-character string called a salt is generated
  + the salt is appended to the password to form the salted password
  + the system’s implementation of the SHA-1 hashing algorithm is retrieved
  + the hashing algorithm is used to obtain the hash of the salted password
  + the hash and the salt are stored on the database.
* When **FancyShaderServiceImpl**#login(String username, String password) is called
  + the hash and salt corresponding to the username are retrieved from the database
  + the salt is appended to the password to form the salted password
  + the salted password is hashed and compared to the hash that was retrieved from the database
  + if they are identical
    - create a new session
    - otherwise throw a FancyShaderLoginException

## Advanced Techniques

The following has been used in

* Remote Method Invocation
* Object Relational Mapping through the use of the Java Persistence API
* Dependancy Injection in JavaFX controller classes
* Utililization of a custom component in JavaFX (**fancyshader.client.gui.viewer.DemoViewer**)
* Use of lambda expressions

# Testing

## Test plan and results

### Login

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test cases** | | | | | |
| **test no** | **username** | **password** | **correct username** | **correct password** | **reasoning** |
| 1 | reuben | password | yes | yes | test logging in with valid detail |
| 2 | Reuben | wrongpassword | yes | no | test logging in with valid username but invalid password |
| 3 | abc | 123 | no | n/a | test completely invalid input |
| 4 | 1 | 0 | no | n/a | test completely invalid input |
| 5 |  |  | no | n/a | test completely invalid input |

For each test case:

* Run the client
* Open **Account 🡪 Sign in**
* Enter username
* Enter password
* Press **Login**

|  |  |  |
| --- | --- | --- |
| **Test results** | | |
| **test no** | **result** | **expected** |
| 1 |  | yes |
| 2 |  | yes |
| 3 |  | yes |
| 4 |  | yes |
| 5 |  | yes |

### Register

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | | | | |
| **test no** | **username** | **password** | **username registered** | **reasoning** |
| 1 | guest | guest | no | test completely valid input |
| 2 | reuben | teehee | yes | test username availability checking |
| 3 | abc | 1234 | no | test username validity checking |
| 4 | abcd | 123 | no | test password validity checking |
| 5 | abc | 123 | no | test both username validity checking and password validity checking |

For each test case:

* Run the program
* Open **Account 🡪 Register**
* Enter username
* Enter password

|  |  |  |
| --- | --- | --- |
| **Test results** | | |
| **test no** | **result** | **expected** |
| 1 |  | yes |
| 2 |  | yes |
| 3 |  | yes |
| 4 |  | yes |
| 5 |  | yes |

### Search

|  |  |  |
| --- | --- | --- |
| **Test cases** | | |
| **test no** | **search** | **reasoning** |
| 1 |  | An empty search field should match all available entries |
| 2 | Infinite | Test searching from beginning of title, using matching case |
| 3 | infinite | Test searching from beginning of title, using lower case |
| 4 | gears | Test searching in the middle of title |
| 5 | (color) | Test searching end of title |
| 6 | demo | Test searching by tag |
| 7 | asdf | Test searching with no matches |
| 8 | Inf Gears | Test searching with partial matches (should not work) |
| 9 | '; DROP TABLE shaderdemo -- | SQL injection (should not work) |

For each test case:

* Run the program
* Sign in to user: reuben, password: password
* Enter search query

|  |  |  |
| --- | --- | --- |
| **test no** | **result** | **expected** |
| 1 |  | yes |
| 2 |  | yes |
| 3 |  | yes |
| 4 |  | yes |
| 5 |  | yes |
| 6 |  | yes |
| 7 |  | yes |
| 8 |  | yes |
| 9 |  | yes |

1. When the fields are empty, as it is initially in the dialog, it displays their placeholder text (“username” and “password” respectively) thus there is no ambiguity to the user, as the screenshot might suggest [↑](#footnote-ref-1)