Genevieve Saab

Final Project Documentation

Project Description

My final project is a dress-up game. The user can input a name into the text area on the start page that will appear over the model, or a file can be read that inputs the name for the user. In the game, the player drags the clothing items onto the model to dress her up, and when the outfit is done there is a "Done!" button that provides a nice comment. The user can go back to the start page by clicking the 'Home" button and information about the game is provided in the "About" section on the start or with the "?" button on the game page.

Note: All references are contained in the code.

Program Name: Gms4kcFinalProject

Language: Java 8 SE

Tools: NetBeans IDE JavaFX Scene Builder

UI Development: Interface created with Scene Builder.

Architecture: The two models are GameModel.java and Person.java. The two views are Start.fxml and Game.fxml. The two controllers are StartController.java and GameController.java. Changes of the model and view are done through the controller using property change listeners and all data is stored in the models. More specific information is given in the requirements section.

Requirements

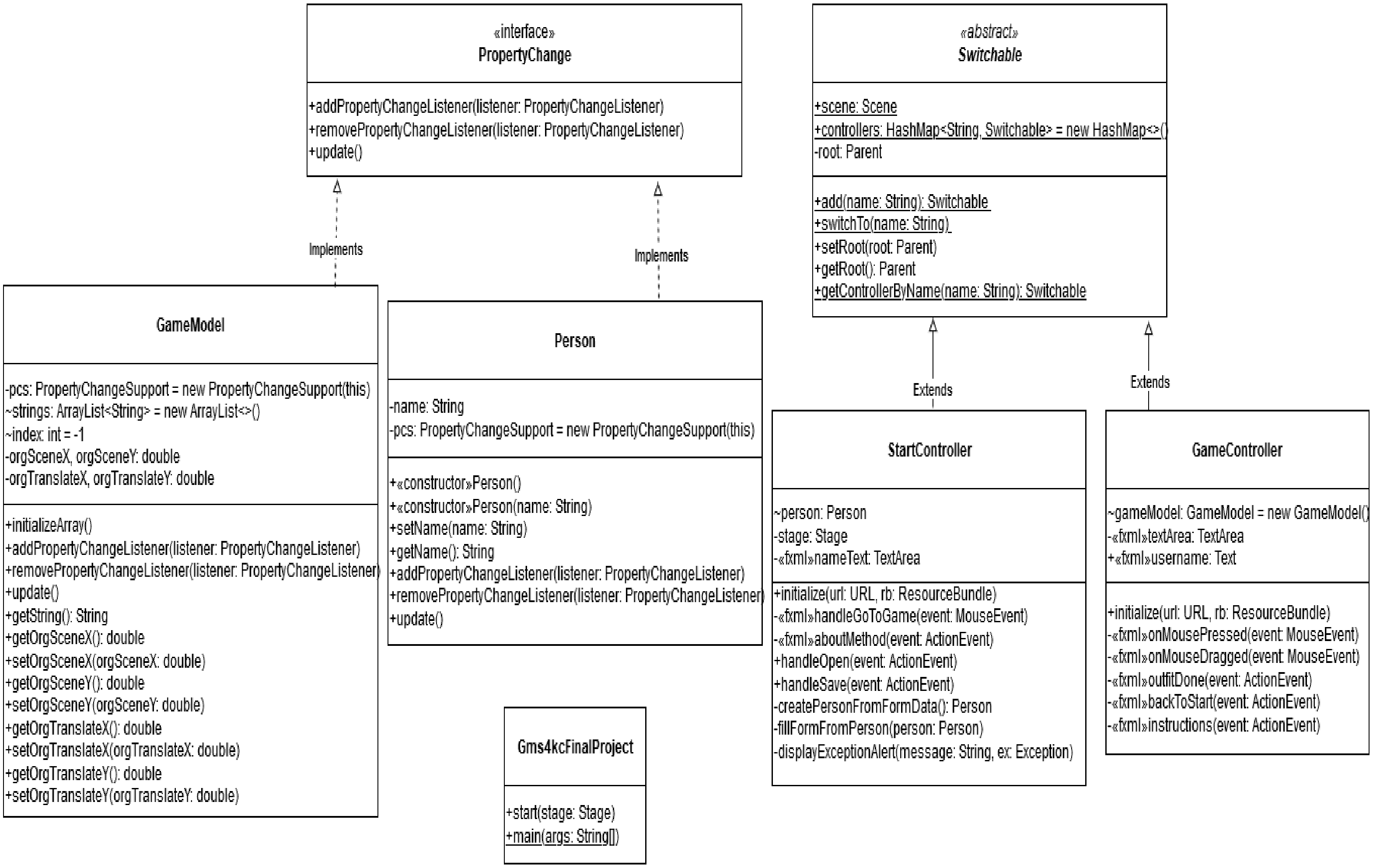
1. Object oriented elements
    a. Classes: Gms4kcFinalProject.java, StartController.java, GameController.java, Person.java, GameModel.java, and Switchable.java
    b. Subclasses: Both StartController,java and GameController,java are subclasses of Switchable.java
    c. Abstract class: Switchable.java is an abstract class
    d. Interface: PropertyChange.java is an interface
2. Code elements utilized
    a. Collection classes
        i. HashMap: Swtichable.java lines 9 and 22
        ii. ArrayList: GameModel.java lines 11 and 21
    b. Exception handling
        i. Try-catch 1: StartController.java lines 91-118

ii. Try-catch 2: StartController.java lines 135-155
iii. Try-catch 3: Switchable.java lines 30-46

3. Clearly defined model
   a. GameModel.java contains all the data for the game and is the main model
   b. Person.java is also a model that contains the data for the Person object that is created to store the name the user entered
4. Multiple scenes and one dynamic scene
   a. The two scenes are Start.fxml and Game.fxml
   b. Game.fxml is dynamic because the “Done!” button changes the string in the comment section and all the clothes are draggable.
      i. Code for “Done!” button in GameModel.java lines 58-62 and GameController.java lines 67-76. A property change listener is used to update the comment string.
      ii. Code for draggable items in GameController.java lines 47-64. The data variables used are stored in GameModel.java
5. About information
   a. On the start page the user can click “About” on the file menu and click on “Game Info” to receive an alert that gives information about the creator and basic instructions.
      i. Code: StartController.java lines 72-84
   b. On the game page, the user can click the “?” button for instructions on how to play the game, also through an alert box.
      i. Code: GameController.java lines 84-96
6. Save and load data
   a. Data can be loaded in the text area on the start page where a name is entered, and the name in this text area can also be saved.
      i. Load data code: StartController.java lines 86-120
      ii. Save data code: StartController.java lines 122-158

UML

PNG file for UML included in zip



```
«interface»
PropertyChange
+addPropertyChangeListener(listener: PropertyChangeListener)
+removePropertyChangeListener(listener: PropertyChangeListener)
+update()
```

```
«abstract»
Switchable
+scene: Scene
+controllers: HashMap<String, Switchable> = new HashMap<>()
-root: Parent
+add(name: String): Switchable
+switchTo(name: String)
+setRoot(root: Parent)
+getRoot(): Parent
+getControllerByName(name: String): Switchable
```

```
GameModel
-pcs: PropertyChangeSupport = new PropertyChangeSupport(this)
~strings: ArrayList<String> = new ArrayList<>()
~index: int = -1
-orgSceneX, orgSceneY: double
-orgTranslateX, orgTranslateY: double
+initializeArray()
+addPropertyChangeListener(listener: PropertyChangeListener)
+removePropertyChangeListener(listener: PropertyChangeListener)
+update()
+getString(): String
+getOrgSceneX(): double
+setOrgSceneX(orgSceneX: double)
+getOrgSceneY(): double
+setOrgSceneY(orgSceneY: double)
+getOrgTranslateX(): double
+setOrgTranslateX(orgTranslateX: double)
+getOrgTranslateY(): double
+setOrgTranslateY(orgTranslateY: double)
```

```
Person
-name: String
-pcs: PropertyChangeSupport = new PropertyChangeSupport(this)
+«constructor»Person()
+«constructor»Person(name: String)
+setName(name: String)
+getName(): String
+addPropertyChangeListener(listener: PropertyChangeListener)
+removePropertyChangeListener(listener: PropertyChangeListener)
+update()
```

```
StartController
~person: Person
-stage: Stage
-«fxml»nameText: TextArea
+initialize(url: URL, rb: ResourceBundle)
-«fxml»handleGoToGame(event: MouseEvent)
-«fxml»aboutMethod(event: ActionEvent)
+handleOpen(event: ActionEvent)
+handleSave(event: ActionEvent)
-createPersonFromFormData(): Person
-fillFormFromPerson(person: Person)
-displayExceptionAlert(message: String, ex: Exception)
```

```
GameController
~gameModel: GameModel = new GameModel()
-«fxml»textArea: TextArea
+«fxml»username: Text
+initialize(url: URL, rb: ResourceBundle)
-«fxml»onMousePressed(event: MouseEvent)
-«fxml»onMouseDragged(event: MouseEvent)
-«fxml»outfitDone(event: ActionEvent)
-«fxml»backToStart(event: ActionEvent)
-«fxml»instructions(event: ActionEvent)
```

```
Gms4kcFinalProject
+start(stage: Stage)
+main(args: String[])
```