Chemistry Passport Program

Charlotte Dye, Humaira Orchee, Sehr Sethi
May 2015

Introduction:

The Passport to Chemistry Adventure program is designed to help parents be involved in their child's learning of science. The original Chemistry Passport Program (with physical kits) provided kits at local libraries. Parent/child teams checked these out from the local libraries and got a sticker on their passport for completing the kit.To help the program expand to allow more participants and to overcome geographical restrictions, we converted the Chemistry Passport Program to a computer-based application. We converted the physical Bark Beetle Infestation Kit to a virtual kit. In addition, we designed our Chemistry Passport Program to allow other developers to easily add other virtual kits.

To draw the estimation grid, we constructed a grid with circles of two different colors to represent infested and non-infested trees. Only one-third of the grid is visible to the kids and they count a tree only if they see its center in the visible part of the grid. We also accounted for possible color-blindness of the user by making the color of the trees dependent on the kind of color-blindness (including no color-blindness) of the user. Upon successful completion of the kit, the kids earn a sticker in their passport and is also able to play the reward game for the kit.

How to Add Kits:

<u>Part 1: Adding your Code</u>
When adding your code:

1. Create a new package for your kit.
2. Create a class in that package that extends the Kit class.
3. Implement the methods inherited from the Kit class.
4. Add any other classes and code necessary for your kit to its package.
5. After you are done adding your kit, create a Runnable JAR file. Refer to the section titled “Exporting Runnable JAR instructions via Eclipse” for instructions.

Details on requirements for the class that implements Kit

1. Include one constructor that simply takes in an instance of ChemistryPassportGUI. This constructor is used when the user wants to play the reward game. There is no need to create all the other pieces of the kit because the user has already completed the kit and does not want to use it right now.
2. Include one constructor that takes in an instance of ChemistryPassportGUI, an instance of Integer (the kit progress), and another instance of Integer (the kit index). This constructor is the one that will normally be called when the user launches the kit.
3. Implement the startKit() method. This is the method that is called when the user clicks on the [kitName]button on the KitSelectionPage (the page that has the header "Choose Your Kit"). This method is in charge of starting and running your kit.
4. Implement the getButtonName() method. This should return the name for your kit that you want to be displayed on buttons and in the passport page specific to your kit. *********IMPORTANT NOTE**********: This must exactly match what you have entered in config.properties as the value for kit_[number]_Name, or your code will not work!
5. Implement the createRewardGame() method. This method should start the reward game for your kit and return a JComponent containing your reward game.
6. Implement the getRewardName() method. This should return the name of the reward game. The name of the reward game is necessary because the JComponent for the reward game is referred to by this name in the CardLayout.

Part 2: Modifying the Properties File

Find the config.properties file. It is located in resources/config/config.properties.
In that file, make the following changes:

1. Find numKits. Increment the number after it by 1 (for this example, we are assuming that there is only one kit initially. Thus, this line should now read numKits=2.)
2. Locate the package and class name of the main class of your new kit. The main class refers to the class that extends Kit. It should not have the 'public static void main(String[] args)" method.
3. Add the following line: kit_[number]=[kit_package_name].[kit_class_name]. For example, if you are adding a second kit in the package second_kit, with a main class named SecondKit, you would add the following line: kit_2=second_kit.SecondKit (In this case the class SecondKit extends Kit)
4. Below that, add a line specifying what you would like the name of your kit to be. This is the name that will be displayed on the buttons and in the passport. IMPORTANT NOTE: This should exactly match the return value of the getButtonName() method that your main class (i.e., the class that extends Kit) implements from the Kit abstract class.

This line should look like the following: kit_[number]_Name=[kit name].
In our example, we would add the following: kit_2_Name=Second Kit.

5. Add a line specifying the completion criteria. This should be an integer representing the maximum progress that can be obtained in your kit (when users have progress equal to the completion criteria, they can earn the reward and see their sticker in the passport kit page). This line should look like the following: kit_[number]_Completion_Criteria=[value]. In our example, we would add the following: kit_2_Completion_Criteria=10.
6. Save the changes to this file!

To summarize, in our example you would make the following changes.
numKits = 2 (change this line)
Add the following three lines:
kit_2=second_kit.SecondKit
kit_2_Name=Second Kit
kit_2_Completion_Criteria=10

Issue : Once a new kit is added and a new runnable JAR file is made available to the user, users should be asked to navigate to their root directory (for instance, on Windows, it is usually the C drive) and delete the folder called "Chemistry Passport". Unfortunately, this means that the user's progress in all the previous kits will be lost. However, not doing so will break the application.

NOTE The developer adding new kits should not have to change the existing source code., unless they want to address the issue mentioned above.

Exporting Runnable JAR instructions via Eclipse

To generate a jar file, do the following:

- Right click on the Project.
- Select "Export.."
- Java → "Runnable JAR file"
- Click "Next"
- Click on the drop-down menu under "Launch configuration"
- Select the class "ChemistryPassport" from the drop-down menu. This is the class that has the "public static void main(String[] args)" method.
    - If you do not see this class, you have to actually run the class from eclipse once for it to show up under "Launch Configuration"
- Under "Export destination", click on "Browse"
- Navigate to the location where you want to save your Runnable JAR file

- In the “Save As:” box, type the name of your Runnable JAR file (e.g. ChemistryPassport.jar)
- Click “Save”
- Under “Library handling”, select “Package required libraries into generated jar”
- Click “Finish”
- Double-click on your Runnable JAR file to start it.

Developer Roadmap:

As written, the ChemistryPassport class extends JApplet and has an init method, and could be used as an applet. Currently, running it as an applet is not practical. This is what you would need to do in order to convert it to a web-based applet:

1. Currently, the user’s information is stored in a file locally on each user’s computer. For this reason, the code as written will not work on a website. You will need to have a server so that you can have a database on that server that can store information for all of the users. To see an example of what this database file looks like, do the following once you have run the application on your computer at least once.
    1. In the command line, navigate to your root directory
    2. Navigate to the Chemistry Passport folder
    3. Enter the following to confirm that the hidden file is there: ls -a
    4. You should see .userInfo.csv
    5. From the command line, use whatever method you prefer to view this hidden file. For example, you could enter: more .userInfo.csv
2. Export the JAR. See the section titled “Exporting JAR instructions via Eclipse” below.
3. You will want to put this JAR file in a separate folder. This folder should be located in the server you will be running the applet from.
4. Navigate to the folder mentioned (using the command line) to create a self-signed applet. Details are provided below.
5. Sign the applet. For long-term use, you would want to have a certificate. However, there is a temporary work-around that can be used, which is self-signing applets. To learn more about how to create self signed applets, please visit the following website: self-signing applets (navigate to “3.Creating Self-Signed Applets”).
6. To be able to do this, you will also need to add the site where you have your applet to the Java control panel exception site list. Refer to the section “How to add site to exception list” for instructions.
7. In the same location as your JAR, create an HTML file. See the file appletsite.html (included with the ChemistryPassportProject on GitHub under the folder “Documents”) to view an example of the HTML file. You will need to have your HTML file in the

same location as your JAR file. Note that “archive=”ChemistryPassport.jar”” should be replaced with the name of your JAR file. (Right-click on the HTML file to view it in a text editor)

Limitations of self-signed applets:

The signer certificate expires within six months. Furthermore, to display the applet on a website, users will have to add the link of the page with the applet to their Java exception list on their web browser.

Exporting JAR instructions via Eclipse

To generate a jar file, do the following:

- Select File → “Export”
- Select Java → “JAR file” and click “next”
- In the “resources to export", first, make sure that all boxes are unchecked.
- Then, open the projects/folders until the \search" package is showing. Select the box next to the search package. This will check both “search" as well as the “src" folder.
- Make sure that both "Export generated class files and resources" and "Export Java source files and resources" are checked
- Click the “\Browse..." button and select a filename and location to save the file.
- Click “Finish". This will create the jar file in the specified location.

How to add site to exception list:

Macintosh

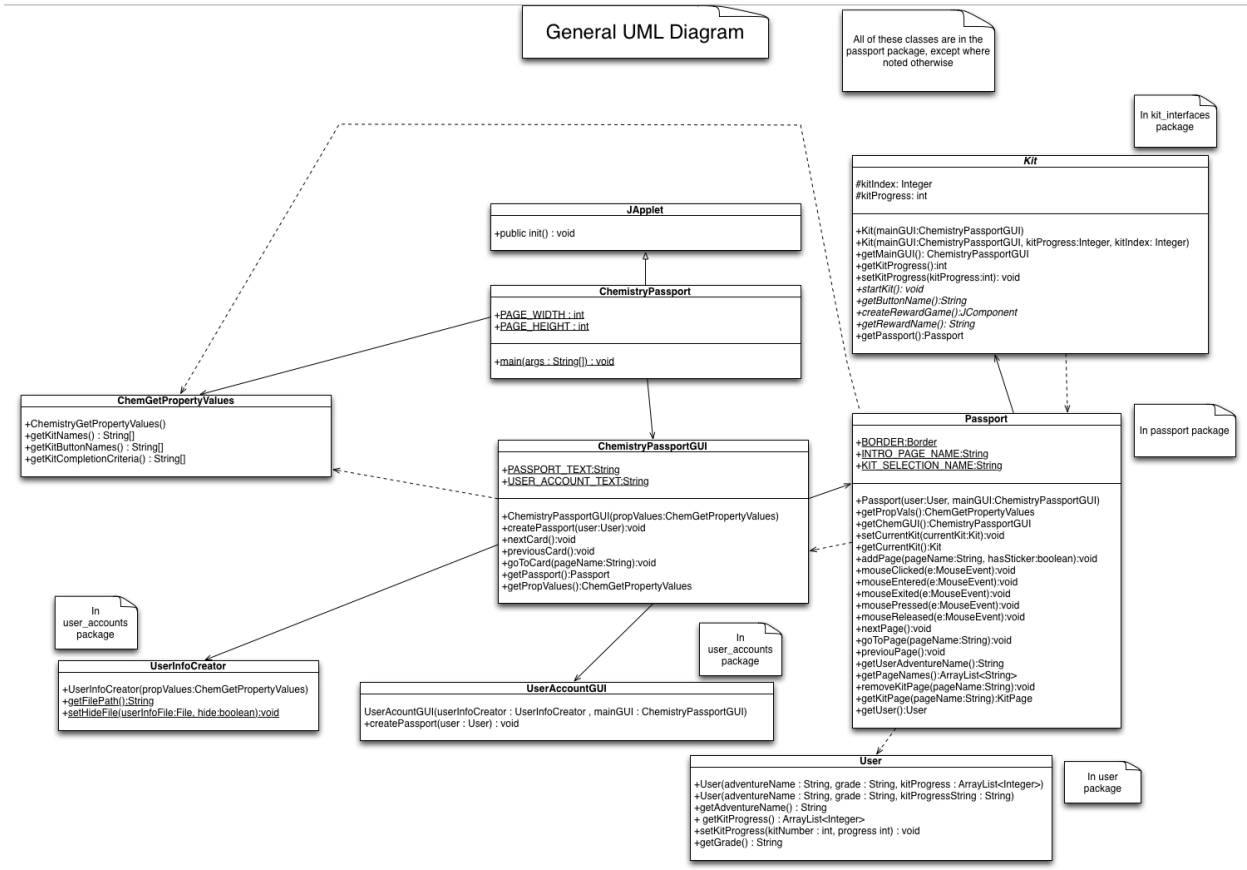
- Copy the URL of the website containing the applet
- Open “System Preferences”
- Click on “Java”
- The Java control panel will open in a separate window.
- Click on “Security”
- Make sure that security level for applications not on exception is set to “high” instead of “very high”
- Click on “Edit Site List”
- Click the “Add” button
- Paste the URL into the open slot
- Click “OK”
- Click “OK” on the Java control panel window

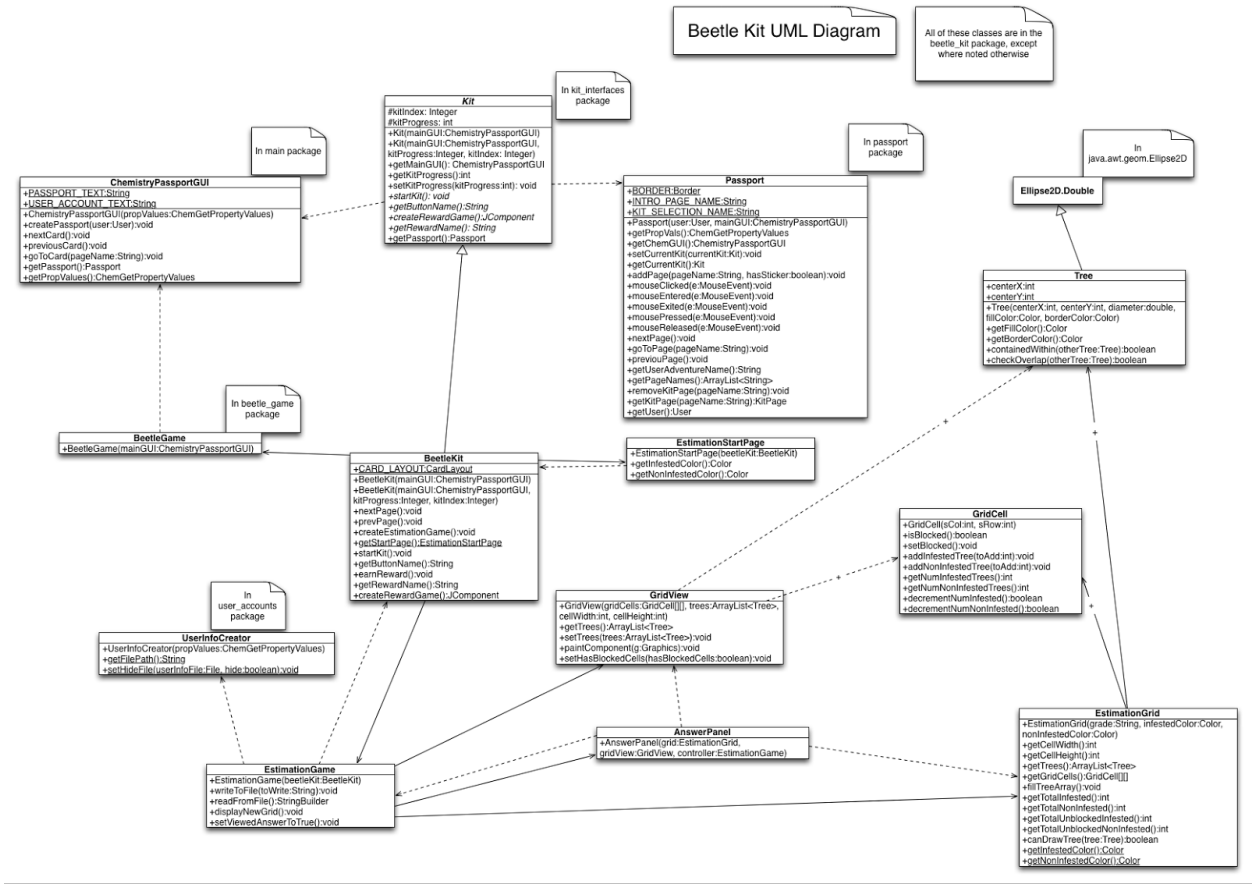
Window

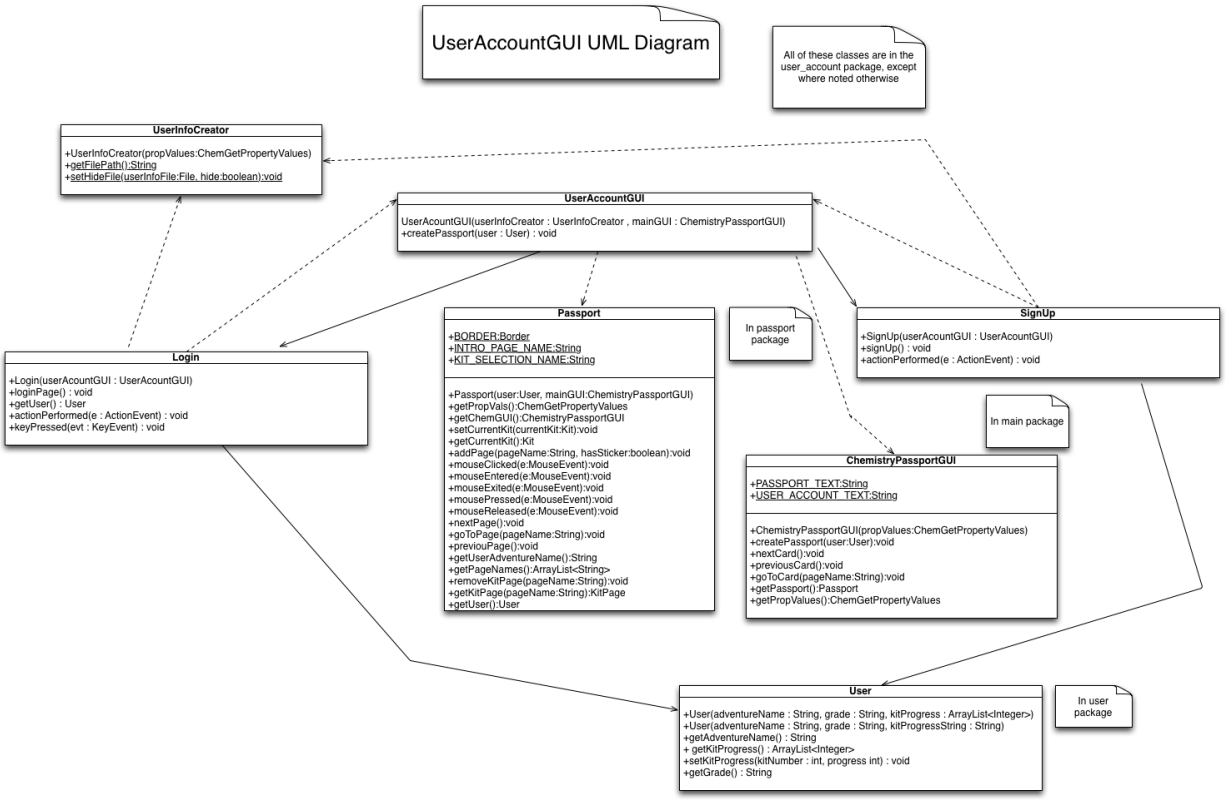
- Copy the URL of the website containing the applet
- Search for “Configure Java” from the start menu and open it
- Click on “Java”
- The Java control panel will open in a separate window.
- Click on “Security”
- Make sure that security level for applications not on exception is set to “high” instead of “very high”
- Click on “Edit Site List”
- Click the “Add” button
- Paste the URL into the open slot
- Click “OK”
- Click “OK” on the Java control panel window

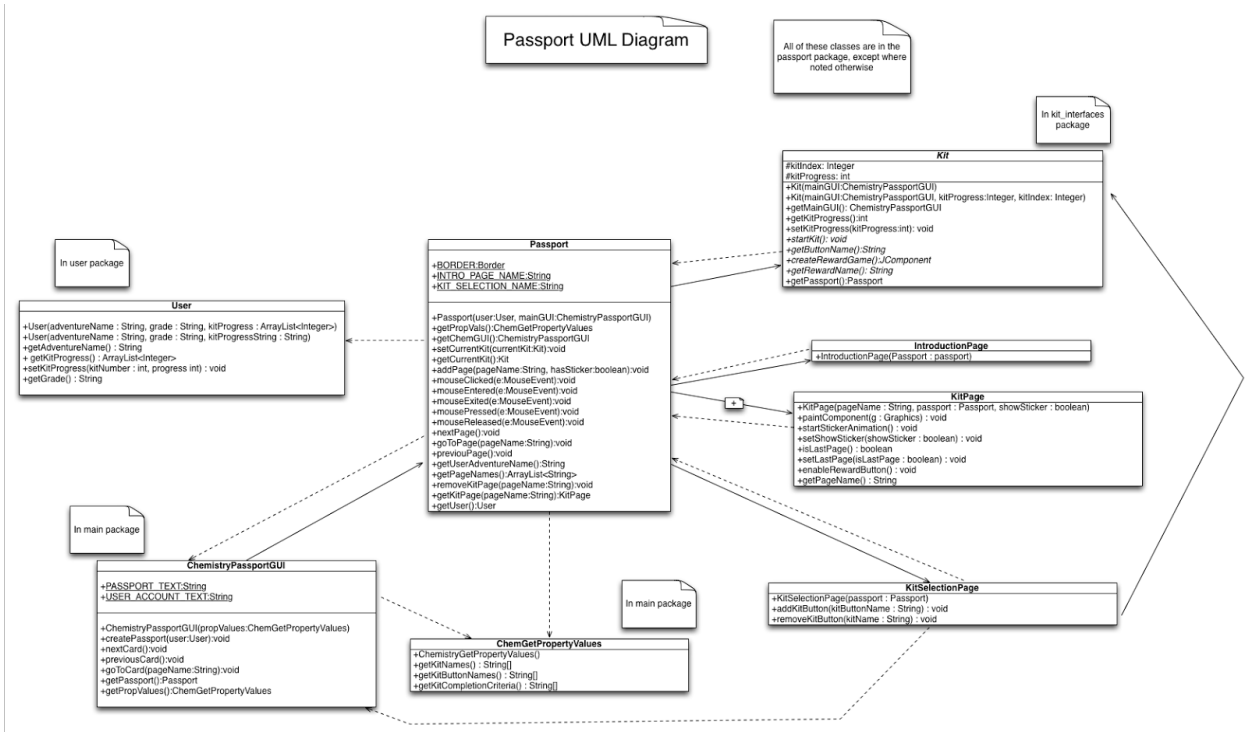
To get a better idea about how the source code is structured, refer to the UML diagrams below. The pdf versions of the UML diagrams are included with the ChemistryPassportProject on GitHub under Documents/UML/

In addition, javadoc for each of the classes is provided with the ChemistryPassportProject on GitHub under the folder “doc”. (You can navigate starting from index.)


General UML Diagram
All of these classes are in the passport package, except where noted otherwise
In kit_interfaces package
Kit
#kitIndex: Integer
#kitProgress: int
+Kit(mainGUI:ChemistryPassportGUI)
+Kit(mainGUI:ChemistryPassportGUI, kitProgress:Integer, kitIndex: Integer)
+getMainGUI(): ChemistryPassportGUI
+getKitProgress():int
+setKitProgress(kitProgress:int): void
+startKit(): void
+getButtonName():String
+createRewardGame():JComponent
+getRewardName(): String
+getPassport():Passport
JApplet
+public init() : void
ChemistryPassport
+PAGE_WIDTH : int
+PAGE_HEIGHT : int
+main(args : String[]) : void
ChemGetPropertyValues
+ChemistryGetPropertyValues()
+getKitNames() : String[]
+getKitButtonNames() : String[]
+getKitCompletionCriteria() : String[]
ChemistryPassportGUI
+PASSPORT_TEXT:String
+USER_ACCOUNT_TEXT:String
+ChemistryPassportGUI(propValues:ChemGetPropertyValues)
+createPassport(user:User):void
+nextCard():void
+previousCard():void
+goToCard(pageName:String):void
+getPassport():Passport
+getPropValues():ChemGetPropertyValues
Passport
+BORDER:Border
+INTRO_PAGE_NAME:String
+KIT_SELECTION_NAME:String
+Passport(user:User, mainGUI:ChemistryPassportGUI)
+getPropVals():ChemGetPropertyValues
+getChemGUI():ChemistryPassportGUI
+setCurrentKit(currentKit:Kit):void
+getCurrentKit():Kit
+addPage(pageName:String, hasSticker:boolean):void
+mouseClicked(e:MouseEvent):void
+mouseEntered(e:MouseEvent):void
+mouseExited(e:MouseEvent):void
+mousePressed(e:MouseEvent):void
+mouseReleased(e:MouseEvent):void
+nextPage():void
+goToPage(pageName:String):void
+previouPage():void
+getUserAdventureName():String
+getPageNames():ArrayList<String>
+removeKitPage(pageName:String):void
+getKitPage(pageName:String):KitPage
+getUser():User
In passport package
In user_accounts package
UserInfoCreator
+UserInfoCreator(propValues:ChemGetPropertyValues)
+getFilePath():String
+setHideFile(userInfoFile:File, hide:boolean):void
In user_accounts package
UserAccountGUI
UserAcountGUI(userInfoCreator : UserInfoCreator , mainGUI : ChemistryPassportGUI)
+createPassport(user : User) : void
User
+User(adventureName : String, grade : String, kitProgress : ArrayList<Integer>)
+User(adventureName : String, grade : String, kitProgressString : String)
+getAdventureName() : String
+ getKitProgress() : ArrayList<Integer>
+setKitProgress(kitNumber : int, progress int) : void
+getGrade() : String
In user package


Beetle Kit UML Diagram
All of these classes are in the beetle_kit package, except where noted otherwise
In kit_interfaces package
Kit
#kitIndex: Integer
#kitProgress: int
+Kit(mainGUI:ChemistryPassportGUI)
+Kit(mainGUI:ChemistryPassportGUI, kitProgress:Integer, kitIndex: Integer)
+getMainGUI(): ChemistryPassportGUI
+getKitProgress():int
+setKitProgress(kitProgress:int): void
+startKit(): void
+getButtonName():String
+createRewardGame():JComponent
+getRewardName(): String
+getPassport():Passport
In main package
ChemistryPassportGUI
+PASSPORT_TEXT:String
+USER_ACCOUNT_TEXT:String
+ChemistryPassportGUI(propValues:ChemGetPropertyValues)
+createPassport(user:User):void
+nextCard():void
+previousCard():void
+goToCard(pageName:String):void
+getPassport():Passport
+getPropValues():ChemGetPropertyValues
In passport package
Passport
+BORDER:Border
+INTRO_PAGE_NAME:String
+KIT_SELECTION_NAME:String
+Passport(user:User, mainGUI:ChemistryPassportGUI)
+getPropVals():ChemGetPropertyValues
+getChemGUI():ChemistryPassportGUI
+setCurrentKit(currentKit:Kit):void
+getCurrentKit():Kit
+addPage(pageName:String, hasSticker:boolean):void
+mouseClicked(e:MouseEvent):void
+mouseEntered(e:MouseEvent):void
+mouseExited(e:MouseEvent):void
+mousePressed(e:MouseEvent):void
+mouseReleased(e:MouseEvent):void
+nextPage():void
+goToPage(pageName:String):void
+previouPage():void
+getUserAdventureName():String
+getPageNames():ArrayList<String>
+removeKitPage(pageName:String):void
+getKitPage(pageName:String):KitPage
+getUser():User
In java.awt.geom.Ellipse2D
Ellipse2D.Double
Tree
+centerX:int
+centerY:int
+Tree(centerX:int, centerY:int, diameter:double, fillColor:Color, borderColor:Color)
+getFillColor():Color
+getBorderColor():Color
+containedWithin(otherTree:Tree):boolean
+checkOverlap(otherTree:Tree):boolean
In beetle_game package
BeetleGame
+BeetleGame(mainGUI:ChemistryPassportGUI)
BeetleKit
+CARD_LAYOUT:CardLayout
+BeetleKit(mainGUI:ChemistryPassportGUI)
+BeetleKit(mainGUI:ChemistryPassportGUI, kitProgress:Integer, kitIndex:Integer)
+nextPage():void
+prevPage():void
+createEstimationGame():void
+getStartPage():EstimationStartPage
+startKit():void
+getButtonName():String
+earnReward():void
+getRewardName():String
+createRewardGame():JComponent
EstimationStartPage
+EstimationStartPage(beetleKit:BeetleKit)
+getInfestedColor():Color
+getNonInfestedColor():Color
GridCell
+GridCell(sCol:int, sRow:int)
+isBlocked():boolean
+setBlocked():void
+addInfestedTree(toAdd:int):void
+addNonInfestedTree(toAdd:int):void
+getNumInfestedTrees():int
+getNumNonInfestedTrees():int
+decrementNumInfested():boolean
+decrementNumNonInfested():boolean
In user_accounts package
UserInfoCreator
+UserInfoCreator(propValues:ChemGetPropertyValues)
+getFilePath():String
+setHideFile(userInfoFile:File, hide:boolean):void
GridView
+GridView(gridCells:GridCell[][], trees:ArrayList<Tree>, cellWidth:int, cellHeight:int)
+getTrees():ArrayList<Tree>
+setTrees(trees:ArrayList<Tree>):void
+paintComponent(g:Graphics):void
+setHasBlockedCells(hasBlockedCells:boolean):void
AnswerPanel
+AnswerPanel(grid:EstimationGrid, gridView:GridView, controller:EstimationGame)
EstimationGame
+EstimationGame(beetleKit:BeetleKit)
+writeToFile(toWrite:String):void
+readFromFile():StringBuilder
+displayNewGrid():void
+setViewedAnswerToTrue():void
EstimationGrid
+EstimationGrid(grade:String, infestedColor:Color, nonInfestedColor:Color)
+getCellWidth():int
+getCellHeight():int
+getTrees():ArrayList<Tree>
+getGridCells():GridCell[][]
+fillTreeArray():void
+getTotalInfested():int
+getTotalNonInfested():int
+getTotalUnblockedInfested():int
+getTotalUnblockedNonInfested():int
+canDrawTree(tree:Tree):boolean
+getInfestedColor():Color
+getNonInfestedColor():Color


UserAccountGUI UML Diagram
All of these classes are in the user_account package, except where noted otherwise
UserInfoCreator
+UserInfoCreator(propValues:ChemGetPropertyValues)
+getFilePath():String
+setHideFile(userInfoFile:File, hide:boolean):void
UserAccountGUI
UserAcountGUI(userInfoCreator : UserInfoCreator , mainGUI : ChemistryPassportGUI)
+createPassport(user : User) : void
Passport
+BORDER:Border
+INTRO_PAGE_NAME:String
+KIT_SELECTION_NAME:String
+Passport(user:User, mainGUI:ChemistryPassportGUI)
+getPropVals():ChemGetPropertyValues
+getChemGUI():ChemistryPassportGUI
+setCurrentKit(currentKit:Kit):void
+getCurrentKit():Kit
+addPage(pageName:String, hasSticker:boolean):void
+mouseClicked(e:MouseEvent):void
+mouseEntered(e:MouseEvent):void
+mouseExited(e:MouseEvent):void
+mousePressed(e:MouseEvent):void
+mouseReleased(e:MouseEvent):void
+nextPage():void
+goToPage(pageName:String):void
+previouPage():void
+getUserAdventureName():String
+getPageNames():ArrayList<String>
+removeKitPage(pageName:String):void
+getKitPage(pageName:String):KitPage
+getUser():User
In passport package
SignUp
+SignUp(userAcountGUI : UserAcountGUI)
+signUp() : void
+actionPerformed(e : ActionEvent) : void
Login
+Login(userAcountGUI : UserAcountGUI)
+loginPage() : void
+getUser() : User
+actionPerformed(e : ActionEvent) : void
+keyPressed(evt : KeyEvent) : void
In main package
ChemistryPassportGUI
+PASSPORT_TEXT:String
+USER_ACCOUNT_TEXT:String
+ChemistryPassportGUI(propValues:ChemGetPropertyValues)
+createPassport(user:User):void
+nextCard():void
+previousCard():void
+goToCard(pageName:String):void
+getPassport():Passport
+getPropValues():ChemGetPropertyValues
User
+User(adventureName : String, grade : String, kitProgress : ArrayList<Integer>)
+User(adventureName : String, grade : String, kitProgressString : String)
+getAdventureName() : String
+ getKitProgress() : ArrayList<Integer>
+setKitProgress(kitNumber : int, progress int) : void
+getGrade() : String
In user package


Passport UML Diagram
All of these classes are in the passport package, except where noted otherwise
In kit_interfaces package
Kit
#kitIndex: Integer
#kitProgress: int
+Kit(mainGUI:ChemistryPassportGUI)
+Kit(mainGUI:ChemistryPassportGUI, kitProgress:Integer, kitIndex: Integer)
+getMainGUI(): ChemistryPassportGUI
+getKitProgress():int
+setKitProgress(kitProgress:int): void
+startKit(): void
+getButtonName():String
+createRewardGame():JComponent
+getRewardName(): String
+getPassport():Passport
Passport
+BORDER:Border
+INTRO_PAGE_NAME:String
+KIT_SELECTION_NAME:String
+Passport(user:User, mainGUI:ChemistryPassportGUI)
+getPropVals():ChemGetPropertyValues
+getChemGUI():ChemistryPassportGUI
+setCurrentKit(currentKit:Kit):void
+getCurrentKit():Kit
+addPage(pageName:String, hasSticker:boolean):void
+mouseClicked(e:MouseEvent):void
+mouseEntered(e:MouseEvent):void
+mouseExited(e:MouseEvent):void
+mousePressed(e:MouseEvent):void
+mouseReleased(e:MouseEvent):void
+nextPage():void
+goToPage(pageName:String):void
+previouPage():void
+getUserAdventureName():String
+getPageNames():ArrayList<String>
+removeKitPage(pageName:String):void
+getKitPage(pageName:String):KitPage
+getUser():User
In user package
User
+User(adventureName : String, grade : String, kitProgress : ArrayList<Integer>)
+User(adventureName : String, grade : String, kitProgressString : String)
+getAdventureName() : String
+ getKitProgress() : ArrayList<Integer>
+setKitProgress(kitNumber : int, progress int) : void
+getGrade() : String
IntroductionPage
+IntroductionPage(Passport : passport)
KitPage
+KitPage(pageName : String, passport : Passport, showSticker : boolean)
+paintComponent(g : Graphics) : void
+startStickerAnimation() : void
+setShowSticker(showSticker : boolean) : void
+isLastPage() : boolean
+setLastPage(isLastPage : boolean) : void
+enableRewardButton() : void
+getPageName() : String
In main package
ChemistryPassportGUI
+PASSPORT_TEXT:String
+USER_ACCOUNT_TEXT:String
+ChemistryPassportGUI(propValues:ChemGetPropertyValues)
+createPassport(user:User):void
+nextCard():void
+previousCard():void
+goToCard(pageName:String):void
+getPassport():Passport
+getPropValues():ChemGetPropertyValues
In main package
KitSelectionPage
+KitSelectionPage(passport : Passport)
+addKitButton(kitButtonName : String) : void
+removeKitButton(kitName : String) : void
ChemGetPropertyValues
+ChemistryGetPropertyValues()
+getKitNames() : String[]
+getKitButtonNames() : String[]
+getKitCompletionCriteria() : String[]