```
1 module com.example.tradingcardgame_9_20_23 {
2     requires javafx.controls;
3     requires javafx.fxml;
4
5
6     opens com.example.tradingcardgame_9_20_23 to javafx.
  fxml;
7     exports com.example.tradingcardgame_9_20_23;
8 }
```

```java
package com.example.tradingcardgame_9_20_23;

public class People {
    //get methods
    public String getUserName(){return this.userName;}
    public int getMoneyInAccount(){return this.moneyInAccount;}
    public DinosaursOwned getSlot(String Slot){
        switch(Slot){//Returns a slot based on the string inputted
            case "Slot1": return slot1;
            case "Slot2": return slot2;
            case "Slot3": return slot3;
            case "Slot4": return slot4;
            case "Slot5": return slot5;
            case "Slot6": return slot6;
            default: return null;
        }
    }

    //set methods
    public void setMoneyInAccount(int newBalance){this.moneyInAccount = newBalance;}
    public void changeMoneyInAccount(int changeInBalance){this.moneyInAccount += changeInBalance;}
    public boolean addDinoCard(DinosaurReference dinoToAdd, int quantityToAdd){
        //returning true means the addition was successful. Returning false means otherwise
        if(slot1 != null && slot1.getDinosaurReference() == dinoToAdd){slot1.changeQuantityOwned(quantityToAdd); return true;}
        if(slot2 != null && slot2.getDinosaurReference() == dinoToAdd){slot2.changeQuantityOwned(quantityToAdd); return true;}
        if(slot3 != null && slot3.getDinosaurReference() == dinoToAdd){slot3.changeQuantityOwned(quantityToAdd); return true;}
        if(slot4 != null && slot4.getDinosaurReference() == dinoToAdd){slot4.changeQuantityOwned(quantityToAdd); return true;}
        if(slot5 != null && slot5.getDinosaurReference() == dinoToAdd){slot5.changeQuantityOwned(quantityToAdd); return true;}
        if(slot6 != null && slot6.getDinosaurReference() == dinoToAdd){slot6.changeQuantityOwned(quantityToAdd);
```

```
29 return true;}
30         if(slot1 == null){slot1 = new DinosaursOwned(
   dinoToAdd, quantityToAdd); return true;}
31         if(slot2 == null){slot2 = new DinosaursOwned(
   dinoToAdd, quantityToAdd); return true;}
32         if(slot3 == null){slot3 = new DinosaursOwned(
   dinoToAdd, quantityToAdd); return true;}
33         if(slot4 == null){slot4 = new DinosaursOwned(
   dinoToAdd, quantityToAdd); return true;}
34         if(slot5 == null){slot5 = new DinosaursOwned(
   dinoToAdd, quantityToAdd); return true;}
35         if(slot6 == null){slot6 = new DinosaursOwned(
   dinoToAdd, quantityToAdd); return true;}
36         return false;
37     }
38 
39     public void deleteEmptySlots(){//Simply clears the
   slots which have 0 dinosaurs in them
40         if(slot1 != null && slot1.getQuantityOwned() == 0){
   slot1 = null;}
41         if(slot2 != null && slot2.getQuantityOwned() == 0){
   slot2 = null;}
42         if(slot3 != null && slot3.getQuantityOwned() == 0){
   slot3 = null;}
43         if(slot4 != null && slot4.getQuantityOwned() == 0){
   slot4 = null;}
44         if(slot5 != null && slot5.getQuantityOwned() == 0){
   slot5 = null;}
45         if(slot6 != null && slot6.getQuantityOwned() == 0){
   slot6 = null;}
46     }
47     public boolean performTrade(DinosaurReference
   cardGained, DinosaurReference cardLost, int quantityGained
   , int quantityLost){
48         this.addDinoCard(cardLost, -quantityLost);//Gives
   the cards away
49         this.deleteEmptySlots();
50         if(this.addDinoCard(cardGained, quantityGained)){
   return true;}//checks if any slots are open and places them
    in if there are
51         else{//Restores the previous value by undoing the
   first line if trade cannot be made
52             this.deleteEmptySlots();
53             this.addDinoCard(cardLost, quantityLost);
54             return false;
55         }
```

```
56      }
57
58      public DinosaursOwned getDinoWithSlot(String dinoName
    ){//Based on the dinosaur name, it returns the slot with
    the specified dino
59          if(slot1 != null && slot1.getDinosaurReference().
    getDinoType().equals(dinoName)){return slot1;}
60          if(slot2 != null && slot2.getDinosaurReference().
    getDinoType().equals(dinoName)){return slot2;}
61          if(slot3 != null && slot3.getDinosaurReference().
    getDinoType().equals(dinoName)){return slot3;}
62          if(slot4 != null && slot4.getDinosaurReference().
    getDinoType().equals(dinoName)){return slot4;}
63          if(slot5 != null && slot5.getDinosaurReference().
    getDinoType().equals(dinoName)){return slot5;}
64          if(slot6 != null && slot6.getDinosaurReference().
    getDinoType().equals(dinoName)){return slot6;}
65          return null;
66      }
67
68      //constructor
69      public People(String newName, int newAccountBalance){
70          this.slot1 = this.slot2 = this.slot3 = this.slot4
     = this.slot5 = this.slot6 = null;
71          this.moneyInAccount = newAccountBalance;
72          this.userName = newName;
73      }
74
75      //private attributes
76      private final String userName;
77      private DinosaursOwned slot1, slot2, slot3, slot4,
    slot5, slot6;
78      private int moneyInAccount;
79  }
80
```

```java
package com.example.tradingcardgame_9_20_23;

public class DinosaursOwned {
    //get methods
    public DinosaurReference getDinosaurReference() {return dinosaurReference;}
    public int getQuantityOwned(){return this.quantityOwned;}

    //set methods
    public void changeQuantityOwned(int changeInBalance){this.quantityOwned += changeInBalance;}

    //constructor
    public DinosaursOwned(DinosaurReference dinosaurReference, int quantityOwned){
        this.dinosaurReference = dinosaurReference;
        this.quantityOwned = quantityOwned;
    }

    //private attributes
    private final DinosaurReference dinosaurReference;
    private int quantityOwned;
}

```

```
 1 package com.example.tradingcardgame_9_20_23;
 2
 3 import javafx.fxml.FXML;
 4 import javafx.scene.control.*;
 5 import javafx.scene.image.Image;
 6 import javafx.scene.image.ImageView;
 7 import javafx.scene.layout.HBox;
 8
 9 import java.io.FileInputStream;
10 import java.io.FileNotFoundException;
11
12 //Fix the money labels
13
14 public class HelloController{
15     //FXML elements
16     @FXML public ListView<String> ViewingCardsListView,
   BuyingCardsListView, SellingCardsListView,
   MyTradingListView, TraderListView, ViewingPeopleListView,
   SellingPeopleListView, FirstPeopleListView,
   SecondPeopleListView, BuyingPeopleListView;
17     @FXML public ImageView ViewingImageView,
   BuyingImageView, SellingImageView, MyTradeImageView,
   TheirTradeImageView;
18     @FXML public Label ViewingLabel, DisplayNameLabel,
   BuyingMoneyLabel, BuyingInfoLabel, SellingMoneyLabel,
   SellingInfoLabel, TradingInfoLabel;
19     @FXML public HBox IntroHBox;
20     @FXML public TabPane MyTabPane;
21     @FXML public TextField NameTextBox;
22     @FXML public Button SubmitNameButton,
   ConfirmPurchaseButton, ConfirmSellingButton,
   ConfirmTradeButton;
23     @FXML public Slider BuyingSlider, SellingSlider,
   MyTradingSlider, MyAmountDesiredSlider;
24
25     @FXML public void handleConfirmTrade(){ //Performs the
   trade once the button is clicked
26         getTraderFromListView(FirstPeopleListView).
   performTrade(getDinoReference(TraderListView.
   getSelectionModel().getSelectedItem()), getDinoReference(
   MyTradingListView.getSelectionModel().getSelectedItem()), (
   int)MyAmountDesiredSlider.getValue(), (int)MyTradingSlider.
   getValue());
27         getTraderFromListView(SecondPeopleListView).
   performTrade(getDinoReference(MyTradingListView.
   getSelectionModel().getSelectedItem()), getDinoReference(
```

```
27 TraderListView.getSelectionModel().getSelectedItem()), (int
   )MyTradingSlider.getValue(), (int)MyAmountDesiredSlider.
   getValue());
28         resetAll();
29     }
30
31     @FXML public void handleTradeAmountChange(){
32         TradingInfoLabel.setText(getTraderFromListView(
   FirstPeopleListView).getUserName() + " is trading with " +
   getTraderFromListView(SecondPeopleListView).getUserName
   () + "\nThe trade is " + (int)MyTradingSlider.getValue() +
   " " + MyTradingListView.getSelectionModel().getSelectedItem
   () + "s for " + (int)MyAmountDesiredSlider.getValue() + " "
    + TraderListView.getSelectionModel().getSelectedItem() + "
   s");
33         if(Math.abs(getDinoReference(MyTradingListView.
   getSelectionModel().getSelectedItem()).calculateValue() *
   MyTradingSlider.getValue() - MyAmountDesiredSlider.getValue
   () * getDinoReference(TraderListView.getSelectionModel().
   getSelectedItem()).calculateValue()) <= 0.1 * (
   getDinoReference(MyTradingListView.getSelectionModel().
   getSelectedItem()).calculateValue() * MyTradingSlider.
   getValue() + MyAmountDesiredSlider.getValue() *
   getDinoReference(TraderListView.getSelectionModel().
   getSelectedItem()).calculateValue())){
34             String errorLabel = "";
35
36             //Check if the trade can be made by making the
   trade.
37             if(!getTraderFromListView(FirstPeopleListView).
   performTrade(getDinoReference(TraderListView.
   getSelectionModel().getSelectedItem()), getDinoReference(
   MyTradingListView.getSelectionModel().getSelectedItem()), (
   int)MyAmountDesiredSlider.getValue(), (int)MyTradingSlider.
   getValue())){errorLabel += getTraderFromListView(
   FirstPeopleListView).getUserName();}
38             else{getTraderFromListView(FirstPeopleListView
   ).performTrade(getDinoReference(TraderListView.
   getSelectionModel().getSelectedItem()), getDinoReference(
   MyTradingListView.getSelectionModel().getSelectedItem()), (
   int)-MyAmountDesiredSlider.getValue(), (int)-
   MyTradingSlider.getValue());}
39             if(!getTraderFromListView(SecondPeopleListView
   ).performTrade(getDinoReference(MyTradingListView.
   getSelectionModel().getSelectedItem()), getDinoReference(
   TraderListView.getSelectionModel().getSelectedItem()), (int
```

```
39 )MyTradingSlider.getValue(), (int)MyAmountDesiredSlider.
   getValue())){errorLabel += getTraderFromListView(
   SecondPeopleListView).getUserName();}
40             else{getTraderFromListView(SecondPeopleListView
   ).performTrade(getDinoReference(MyTradingListView.
   getSelectionModel().getSelectedItem()), getDinoReference(
   TraderListView.getSelectionModel().getSelectedItem()), (int
   )-MyTradingSlider.getValue(), (int)-MyAmountDesiredSlider.
   getValue());}
41
42             //Shows an error label if at least one of the
   traders cannot make the trade due to a lack of slots
43             if(!errorLabel.isEmpty()){TradingInfoLabel.
   setText(TradingInfoLabel.getText() + "\nWhoops! " +
   errorLabel + " cannot make that trade due to a lack of
   slots!");}
44             if(MyTradingSlider.getValue()==0 ||
   MyAmountDesiredSlider.getValue()==0){TradingInfoLabel.
   setText("This is not really a trade. You're literally
   trading 0 cards for at least one of them.");}
45             ConfirmTradeButton.setDisable(!errorLabel.
   isEmpty()||MyTradingSlider.getValue()==0 ||
   MyAmountDesiredSlider.getValue()==0); //Disable the trade
   if it doesn't work
46         }else{TradingInfoLabel.setText(TradingInfoLabel.
   getText() + "\nThe Trade Doesn't Seem Fair");}
47     }
48
49     @FXML public void handleTradingSelectionChange(){ //
   Updates the area whenever a new user or dino is selected
   for each list view
50         if(MyTradingListView.getSelectionModel().
   getSelectedItem() != null && !MyTradingListView.
   getSelectionModel().getSelectedItem().equals("No Cards"
   ) && TraderListView.getSelectionModel().getSelectedItem
   () != null && !TraderListView.getSelectionModel().
   getSelectedItem().equals("No Cards")) {
51             TradingInfoLabel.setText("");
52             if (getTraderFromListView(FirstPeopleListView
   ) == getTraderFromListView(SecondPeopleListView)) {
   TradingInfoLabel.setText("Why Are You Trading With Yourself
   ?");}
53             else if (MyTradingListView.getSelectionModel().
   getSelectedItem().equals(TraderListView.getSelectionModel
   ().getSelectedItem())) {TradingInfoLabel.setText("Why Are
   You Trading The Same Exact Card?");}
```

```
54              else {
55                  MyTradingSlider.setDisable(false);
56                  MyAmountDesiredSlider.setDisable(false);
57                  MyTradingSlider.setMax(
   getTraderFromListView(FirstPeopleListView).getDinoWithSlot(
   MyTradingListView.getSelectionModel().getSelectedItem()).
   getQuantityOwned());
58                  MyAmountDesiredSlider.setMax(
   getTraderFromListView(SecondPeopleListView).getDinoWithSlot
   (TraderListView.getSelectionModel().getSelectedItem()).
   getQuantityOwned());
59                  handleTradeAmountChange();
60              }
61              setImage(MyTradeImageView, MyTradingListView.
   getSelectionModel().getSelectedItem());
62              setImage(TheirTradeImageView, TraderListView.
   getSelectionModel().getSelectedItem());
63          }
64      }
65
66      @FXML public void handleConfirmSelling(){//Handles the
   selling of the dino and resets
67          getTraderFromListView(SellingPeopleListView).
   changeMoneyInAccount((int)(SellingSlider.getValue() *
   getDinoReference(SellingCardsListView.getSelectionModel().
   getSelectedItem()).calculateValue()));
68          getTraderFromListView(SellingPeopleListView).
   addDinoCard(getDinoReference(SellingCardsListView.
   getSelectionModel().getSelectedItem()), -(int)(
   SellingSlider.getValue()));
69          getTraderFromListView(SellingPeopleListView).
   deleteEmptySlots();
70          resetAll();
71      }
72
73      @FXML public void handleSellSliderClick(){
   SellingInfoLabel.setText("You Are Selling " + (int)
   SellingSlider.getValue() + " "+ SellingCardsListView.
   getSelectionModel().getSelectedItem() + "'s for $" +
   getDinoReference(SellingCardsListView.getSelectionModel().
   getSelectedItem()).calculateValue() + " each. That's a
   total of $" + (int)SellingSlider.getValue() *
   getDinoReference(SellingCardsListView.getSelectionModel().
   getSelectedItem()).calculateValue());} //Changes the info
   label whenever the value of the slider changes
74
```

```
75      @FXML public void handleSellClick(){//Initializes the
    area whenever a dino in the selling area is clicked
76          if(SellingCardsListView.getSelectionModel().
    getSelectedItem() != null && !SellingCardsListView.
    getSelectionModel().getSelectedItem().equals("No Cards")){
77              SellingSlider.setDisable(false);
78              SellingSlider.setMax(getTraderFromListView(
    SellingPeopleListView).getDinoWithSlot(
    SellingCardsListView.getSelectionModel().getSelectedItem
    ()).getQuantityOwned());
79              setImage(SellingImageView,
    SellingCardsListView.getSelectionModel().getSelectedItem
    ());
80              ConfirmSellingButton.setDisable(false);
81              SellingInfoLabel.setText("You Are Selling "
     + (int)SellingSlider.getValue() + " "+
    SellingCardsListView.getSelectionModel().getSelectedItem
    () + "'s for $" + getDinoReference(SellingCardsListView.
    getSelectionModel().getSelectedItem()).calculateValue() +
    " each. That's a total of $" + (int)SellingSlider.getValue
    () * getDinoReference(SellingCardsListView.
    getSelectionModel().getSelectedItem()).calculateValue());
82          }
83      }
84
85      @FXML public void handlePurchase(){//Handles buying
    the certain dino
86          getTraderFromListView(BuyingPeopleListView).
    changeMoneyInAccount((int)(-1*BuyingSlider.getValue() *
    getDinoReference(BuyingCardsListView.getSelectionModel().
    getSelectedItem()).calculateValue()));
87          getTraderFromListView(BuyingPeopleListView).
    addDinoCard(getDinoReference(BuyingCardsListView.
    getSelectionModel().getSelectedItem()), (int) BuyingSlider
    .getValue());
88          getTraderFromListView(BuyingPeopleListView).
    deleteEmptySlots();
89          resetAll();
90      }
91
92      @FXML public void handleBuyingSliderChange(){
    BuyingInfoLabel.setText("You Are Buying " + (int)
    BuyingSlider.getValue() + " " + BuyingCardsListView.
    getSelectionModel().getSelectedItem() + "'s for a total of
     $" + (int)BuyingSlider.getValue() * getDinoReference(
    BuyingCardsListView.getSelectionModel().getSelectedItem
```

```
92 ()).calculateValue());}//Changes the info label whenever
   the slider value changes
93
94     @FXML public void handleShopClicked(){//Updates the
   buying area whenever a new dino or a new dino or person is
    selected
95         if(BuyingCardsListView.getSelectionModel().
   getSelectedItem() != null && !BuyingCardsListView.
   getSelectionModel().getSelectedItem().equals("No Cards"
   )) {
96             setImage(BuyingImageView, BuyingCardsListView.
   getSelectionModel().getSelectedItem());
97             BuyingSlider.setMax((int) Math.floor((double)
   getTraderFromListView(BuyingPeopleListView).
   getMoneyInAccount() / getDinoReference(BuyingCardsListView
   .getSelectionModel().getSelectedItem()).calculateValue
   ()));
98             BuyingSlider.setDisable(BuyingSlider.getMax
   () == 0);
99             ConfirmPurchaseButton.setDisable(false);
100            BuyingInfoLabel.setText("You Are Buying " + (
   int)BuyingSlider.getValue() + " " + BuyingCardsListView.
   getSelectionModel().getSelectedItem() + "'s for a total of
    $" + (int)BuyingSlider.getValue() * getDinoReference(
   BuyingCardsListView.getSelectionModel().getSelectedItem
   ()).calculateValue());
101            if (BuyingSlider.getMax() == 0) {
102                BuyingInfoLabel.setText("You Don't Have
   Enough Money For That Dino");
103                ConfirmPurchaseButton.setDisable(true);
104            }
105            //Tests if there's any available slots
106            if (!getTraderFromListView(
   BuyingPeopleListView).addDinoCard(getDinoReference(
   BuyingCardsListView.getSelectionModel().getSelectedItem
   ()), 0)) {
107                BuyingInfoLabel.setText(BuyingInfoLabel.
   getText() + "\nYou don't have any available slots for that
    card.");
108                ConfirmPurchaseButton.setDisable(true);
109            }
110            getTraderFromListView(BuyingPeopleListView).
   deleteEmptySlots();
111        }
112        BuyingMoneyLabel.setText("Money In " +
   getTraderFromListView(BuyingPeopleListView).getUserName
```

```
112 () + " Account: " + getTraderFromListView(
    BuyingPeopleListView).getMoneyInAccount());
113     }
114 
115     @FXML public void handleViewingListClick(){//When the
    user wants to view a certain dinosaur owned by a certain
    person, the corresponding image is put up and the info is
    shown
116         if(ViewingCardsListView.getSelectionModel().
    getSelectedItem() != null && ViewingPeopleListView.
    getSelectionModel().getSelectedItem() != null && !
    ViewingCardsListView.getSelectionModel().getSelectedItem
    ().equals("No Cards")){
117             setImage(ViewingImageView,
    ViewingCardsListView.getSelectionModel().getSelectedItem
    ());
118             DinosaurReference dinosaurReference =
    getTraderFromListView(ViewingPeopleListView).
    getDinoWithSlot(ViewingCardsListView.getSelectionModel().
    getSelectedItem()).getDinosaurReference();
119             ViewingLabel.setText("My " + dinosaurReference
    .getDinoType() + ":\nRarity: " + dinosaurReference.
    getRarity() + "\nFamily: " + dinosaurReference.getFamily
    () + "\nAttack: " + dinosaurReference.getAttack() + "\n
    Health: " + dinosaurReference.getHealth() + "\nValue: "+
    dinosaurReference.calculateValue() + "\nQuantity Owned: "
     + getTraderFromListView(ViewingPeopleListView).
    getDinoWithSlot(ViewingCardsListView.getSelectionModel().
    getSelectedItem()).getQuantityOwned());
120         }
121     }
122 
123     @FXML public void resetAll(){
124         if(BuyingCardsListView != null) {
125             //Resets all the list views and updates them
126             ViewingCardsListView.getItems().clear();
127             SellingCardsListView.getItems().clear();
128             MyTradingListView.getItems().clear();
129             TraderListView.getItems().clear();
130             ViewingCardsListView.getSelectionModel().
    clearSelection();
131             BuyingCardsListView.getSelectionModel().
    clearSelection();
132             SellingCardsListView.getSelectionModel().
    clearSelection();
133             MyTradingListView.getSelectionModel().
```

```
133 clearSelection();
134             TraderListView.getSelectionModel().
    clearSelection();
135             setValueOfListView(ViewingCardsListView,
    getTraderFromListView(ViewingPeopleListView));
136             setValueOfListView(SellingCardsListView,
    getTraderFromListView(SellingPeopleListView));
137             setValueOfListView(MyTradingListView,
    getTraderFromListView(FirstPeopleListView));
138             setValueOfListView(TraderListView,
    getTraderFromListView(SecondPeopleListView));
139
140             //Hides all the image views
141             ViewingImageView.setVisible(false);
142             BuyingImageView.setVisible(false);
143             SellingImageView.setVisible(false);
144             MyTradeImageView.setVisible(false);
145             TheirTradeImageView.setVisible(false);
146
147             //Resets and disables the buttons and labels
148             ViewingLabel.setText("");
149             BuyingInfoLabel.setText("");
150             SellingInfoLabel.setText("");
151             TradingInfoLabel.setText("");
152             BuyingMoneyLabel.setText("Money In " +
    getTraderFromListView(BuyingPeopleListView).getUserName
    () + " Account: " + getTraderFromListView(
    BuyingPeopleListView).getMoneyInAccount());
153             SellingMoneyLabel.setText("Money In " +
    getTraderFromListView(SellingPeopleListView).getUserName
    () + " Account: " + getTraderFromListView(
    SellingPeopleListView).getMoneyInAccount());
154             ConfirmPurchaseButton.setDisable(true);
155             ConfirmSellingButton.setDisable(true);
156             ConfirmTradeButton.setDisable(true);
157             BuyingSlider.setDisable(true);
158             SellingSlider.setDisable(true);
159             MyTradingSlider.setDisable(true);
160             MyAmountDesiredSlider.setDisable(true);
161         }
162     }
163
164     //Initialize program
165     @FXML public void handleInitialize(){
166         IntroHBox.setVisible(false);
167         IntroHBox.setDisable(true);
```

```
168         DisplayNameLabel.setVisible(true);
169         MyTabPane.setVisible(true);
170         MyTabPane.setDisable(false);
171 
172         //Initializes all the list views
173         ViewingPeopleListView.getItems().addAll("Me", "
    Trader 1", "Trader 2", "Trader 3");
174         SecondPeopleListView.getItems().addAll("Me", "
    Trader 1", "Trader 2", "Trader 3");
175         FirstPeopleListView.getItems().addAll("Me", "
    Trader 1", "Trader 2", "Trader 3");
176         SellingPeopleListView.getItems().addAll("Me", "
    Trader 1", "Trader 2", "Trader 3");
177         BuyingPeopleListView.getItems().addAll("Me", "
    Trader 1", "Trader 2", "Trader 3");
178         ViewingPeopleListView.getSelectionModel().
    selectFirst();
179         SellingPeopleListView.getSelectionModel().
    selectFirst();
180         SecondPeopleListView.getSelectionModel().
    selectFirst();
181         FirstPeopleListView.getSelectionModel().
    selectFirst();
182         BuyingPeopleListView.getSelectionModel().
    selectFirst();
183         BuyingCardsListView.getItems().addAll("Tyrannosaur
    ", "Indominus", "Stegosaur", "Triceratops", "Sarcosuchus"
    , "Velociraptor", "Alanqa", "Carnotaurus", "Dilophosaurus"
    );
184 
185         //Initializes all the cards
186         Tyrannosaur = new DinosaurReference("Tyrannosaur"
    , "Legendary", "Carnivore",1603, 612);
187         Indominus = new DinosaurReference("Indominus", "
    Legendary", "Carnivore", 5430, 2074);
188         Stegosaur = new DinosaurReference("Stegosaur", "
    Super Rare", "Herbivore", 982, 251);
189         Triceratops = new DinosaurReference("Triceratops"
    , "Common", "Herbivore", 274, 70);
190         Sarcosuchus = new DinosaurReference("Sarcosuchus"
    , "Legendary", "Amphibian", 1500, 573);
191         Velociraptor = new DinosaurReference("Velociraptor
    ", "Super Rare", "Carnivore", 800, 306);
192         Alanqa = new DinosaurReference("Alanqa", "Common"
    , "Pterosaur", 217, 83);
193         Carnotaurus = new DinosaurReference("Carnotaurus"
```

```
193 , "Rare", "Carnivore", 419, 160);
194         Dilophosaurus = new DinosaurReference("
    Dilophosaurus", "Rare", "Carnivore", 396, 151);
195 
196         //Initializes the people
197         final int startingBalance = 1000;
198         if(NameTextBox.getText().isEmpty()){Me = new
    People("Anonymous", startingBalance);}
199         else{Me = new People(NameTextBox.getText(),
    startingBalance);}
200         Trader1 = new People("Trader 1", startingBalance);
201         Trader2 = new People("Trader 2", startingBalance);
202         Trader3 = new People("Trader 3", startingBalance);
203         DisplayNameLabel.setText("Hi, " + Me.getUserName
    ());
204 
205         resetAll();
206     }
207 
208     public People getTraderFromListView(ListView<String>
    listView){//Gets the value of the selected list view and
    returns the person selected
209         switch (listView.getSelectionModel().
    getSelectedItem()){
210             case "Me" : return Me;
211             case "Trader 1": return Trader1;
212             case "Trader 2": return Trader2;
213             case "Trader 3": return Trader3;
214             default: return null;
215         }
216     }
217 
218     public DinosaurReference getDinoReference(String
    dinoReferenced){//Gets the dino reference from the dino
    name
219         switch (dinoReferenced){
220             case "Tyrannosaur": return Tyrannosaur;
221             case "Indominus": return Indominus;
222             case "Stegosaur": return Stegosaur;
223             case "Triceratops": return Triceratops;
224             case "Alanqa": return Alanqa;
225             case "Sarcosuchus": return Sarcosuchus;
226             case "Dilophosaurus": return Dilophosaurus;
227             case "Carnotaurus": return Carnotaurus;
228             case "Velociraptor": return Velociraptor;
229             default: return null;
```

```
230         }
231     }
232 
233     public void setValueOfListView(ListView<String>
    listViewToModify, People dataFromPerson){//Sets the values
     of the dino list views based on the inventory of the
    selected user
234         if(dataFromPerson.getSlot("Slot1") != null){
    listViewToModify.getItems().add(dataFromPerson.getSlot("
    Slot1").getDinosaurReference().getDinoType());}
235         if(dataFromPerson.getSlot("Slot2") != null){
    listViewToModify.getItems().add(dataFromPerson.getSlot("
    Slot2").getDinosaurReference().getDinoType());}
236         if(dataFromPerson.getSlot("Slot3") != null){
    listViewToModify.getItems().add(dataFromPerson.getSlot("
    Slot3").getDinosaurReference().getDinoType());}
237         if(dataFromPerson.getSlot("Slot4") != null){
    listViewToModify.getItems().add(dataFromPerson.getSlot("
    Slot4").getDinosaurReference().getDinoType());}
238         if(dataFromPerson.getSlot("Slot5") != null){
    listViewToModify.getItems().add(dataFromPerson.getSlot("
    Slot5").getDinosaurReference().getDinoType());}
239         if(dataFromPerson.getSlot("Slot6") != null){
    listViewToModify.getItems().add(dataFromPerson.getSlot("
    Slot6").getDinosaurReference().getDinoType());}
240         if(listViewToModify.getItems().toArray().length
     == 0){listViewToModify.getItems().add("No Cards");}
241     }
242 
243     private void setImage(ImageView imageView, String
    value){//Just sets the image of the selected image view
    based on the dinosaur name selected
244         imageView.setVisible(true);
245         try{imageView.setImage(new Image(new
    FileInputStream("src/main/resources/DinoImages/" + value
     + ".png")));}
246         catch(FileNotFoundException e){e.printStackTrace
    ();}
247     }
248 
249     //Private Variables
250     private DinosaurReference Tyrannosaur, Indominus,
    Stegosaur, Alanqa, Triceratops, Sarcosuchus, Dilophosaurus
    , Carnotaurus, Velociraptor;
251     private People Me, Trader1, Trader2, Trader3;
252 }
```

```java
package com.example.tradingcardgame_9_20_23;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.stage.Stage;

import java.io.IOException;

public class HelloApplication extends Application {
    @Override
    public void start(Stage stage) throws IOException {
        FXMLLoader fxmlLoader = new FXMLLoader(
HelloApplication.class.getResource("hello-view.fxml"));
        Scene scene = new Scene(fxmlLoader.load());
        stage.setTitle("Dylan's Dino Trading Bonanza!");
        stage.setScene(scene);
        stage.show();
    }

    public static void main(String[] args) {
        launch();
    }
}
```

```java
package com.example.tradingcardgame_9_20_23;

public class DinosaurReference {
    //get methods
    public String getDinoType(){return this.dinoType;}
    public String getRarity(){return this.rarity;}
    public int getAttack(){return this.attack;}
    public int getHealth(){return this.health;}
    public String getFamily(){return this.family;}
    public int calculateValue(){//Calculates the value based on the attack, health, and rarity of the card
        int tempValue;
        switch (this.rarity){
            case "Legendary": tempValue = 20; break;
            case "Super Rare": tempValue = 10; break;
            case "Rare": tempValue = 5; break;
            case "Common": tempValue = 1; break;
            default: tempValue = 0;
        }
        tempValue *= 25;
        tempValue += this.attack/10 + this.health/20;
        return tempValue;
    }

    //Constructor. Initialize attributes
    public DinosaurReference(String dinoType, String rarity, String family, int attack, int health){
        this.dinoType = dinoType;
        this.rarity = rarity;
        this.family = family;
        this.attack = attack;
        this.health = health;
    }

    //private attributes
    private final int attack, health;
    private final String dinoType, rarity, family;
}

```

```
.button{
    -fx-border-color: black;
    -fx-border-width: 2;
    -fx-border-style: solid;
    -fx-border-radius: 3;
    -fx-font-family: "serif";
    -fx-font-size: 15;
    -fx-background-color: white;
    -fx-cursor: "hand";
}

.tab{
    -fx-border-radius: 3;
    -fx-font-family: "serif";
    -fx-font-size: 15;
    -fx-background-color: white;
    -fx-cursor: "hand";
}

.label{
    -fx-font-family: "serif";
    -fx-font-size: 15;
}

#TitleLabel{
    -fx-font-size: 25;
}

.describingLabel{
    -fx-font-size: 12;
}
```

```
 1 <?xml version="1.0" encoding="UTF-8"?>
 2
 3 <?import javafx.scene.control.Button?>
 4 <?import javafx.scene.control.Label?>
 5 <?import javafx.scene.control.ListView?>
 6 <?import javafx.scene.control.Slider?>
 7 <?import javafx.scene.control.Tab?>
 8 <?import javafx.scene.control.TabPane?>
 9 <?import javafx.scene.control.TextField?>
10 <?import javafx.scene.image.ImageView?>
11 <?import javafx.scene.layout.AnchorPane?>
12 <?import javafx.scene.layout.HBox?>
13 <?import javafx.scene.layout.VBox?>
14 <?import javafx.scene.text.Font?>
15
16 <AnchorPane maxHeight="-Infinity" maxWidth="-Infinity"
   minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.
   0" prefWidth="600.0" stylesheets="@style.css" xmlns="http
   ://javafx.com/javafx/20.0.1" xmlns:fx="http://javafx.com/
   fxml/1" fx:controller="com.example.tradingcardgame_9_20_23.
   HelloController">
17    <children>
18       <Label id="TitleLabel" alignment="CENTER"
   contentDisplay="CENTER" layoutX="6.0" layoutY="14.0"
   prefHeight="18.0" prefWidth="590.0" text="Dylan's Dino
   Trading Bonanza!" underline="true">
19          <font>
20             <Font size="47.0" />
21          </font>
22       </Label>
23       <TabPane fx:id="MyTabPane" disable="true" layoutY="
   100.0" prefHeight="300.0" prefWidth="600.0" side="BOTTOM"
   tabClosingPolicy="UNAVAILABLE">
24         <tabs>
25           <Tab onSelectionChanged="#resetAll" text="My
   Collection">
26             <content>
27               <AnchorPane minHeight="0.0" minWidth="0.0"
   prefHeight="268.0" prefWidth="600.0">
28                  <children>
29                     <VBox alignment="CENTER" layoutX="
   14.0" layoutY="5.0" prefHeight="260.0" prefWidth="230.0"
   spacing="2.0">
30                        <children>
31                           <Label text="Select The
   Person And The Card" />
```

```
32                                 <ListView fx:id="
   ViewingPeopleListView" onMouseClicked="#resetAll"
   prefHeight="139.0" prefWidth="222.0" />
33                                 <ListView fx:id="
   ViewingCardsListView" onMouseClicked="#
   handleViewingListClick" prefHeight="218.0" prefWidth="222.0
   " />
34                              </children>
35                           </VBox>
36                           <ImageView fx:id="ViewingImageView"
    fitHeight="260.0" fitWidth="169.0" layoutX="252.0"
   pickOnBounds="true" preserveRatio="true" styleClass="images
   " y="5.0" />
37                           <Label fx:id="ViewingLabel"
   alignment="CENTER" contentDisplay="CENTER" layoutX="453.0"
   layoutY="5.0" prefHeight="260.0" prefWidth="136.0"
   textAlignment="CENTER" wrapText="true" />
38                        </children>
39                     </AnchorPane>
40                  </content>
41               </Tab>
42               <Tab onSelectionChanged="#resetAll" text="Buy
   Cards">
43                  <content>
44                     <AnchorPane minHeight="0.0" minWidth="0.0"
   prefHeight="180.0" prefWidth="200.0">
45                        <children>
46                           <Label fx:id="BuyingMoneyLabel"
   alignment="CENTER" prefHeight="17.0" prefWidth="600.0" text
   ="Label" textAlignment="CENTER" />
47                           <VBox layoutX="14.0" layoutY="39.0"
    prefHeight="220.0" prefWidth="230.0">
48                              <children>
49                                 <HBox alignment="CENTER"
   prefHeight="36.0" prefWidth="205.0">
50                                    <children>
51                                       <Label text="The Shop"
    />
52                                    </children>
53                                 </HBox>
54                                 <ListView fx:id="
   BuyingCardsListView" onMouseClicked="#handleShopClicked"
   prefHeight="218.0" prefWidth="220.0" />
55                              </children>
56                           </VBox>
57                           <ImageView fx:id="BuyingImageView"
```

```
57 fitHeight="218.0" fitWidth="130.0" layoutX="257.0" layoutY
   ="40.0" pickOnBounds="true" preserveRatio="true"
   styleClass="images" />
58                         <VBox alignment="CENTER" layoutX="
   390.0" layoutY="25.0" prefHeight="235.0" prefWidth="200.0"
    spacing="5.0">
59                            <children>
60                               <VBox alignment="CENTER"
   prefHeight="98.0" prefWidth="190.0">
61                                  <children>
62                                     <Label text="Select
   The Person Buying" />
63                                     <ListView fx:id="
   BuyingPeopleListView" onMouseClicked="#handleShopClicked"
   prefHeight="200.0" prefWidth="200.0" />
64                                  </children>
65                               </VBox>
66                               <Label text="How Much Do You
    Want?" />
67                               <Slider fx:id="BuyingSlider"
    majorTickUnit="2.0" max="10.0" minorTickCount="1"
   onMouseClicked="#handleBuyingSliderChange" onMouseReleased
   ="#handleBuyingSliderChange" prefHeight="0.0" prefWidth="
   100.0" showTickLabels="true" showTickMarks="true"
   snapToTicks="true" />
68                               <Label fx:id="
   BuyingInfoLabel" alignment="CENTER" contentDisplay="CENTER
   " prefHeight="123.0" prefWidth="149.0" styleClass="
   describingLabel" text="Label" textAlignment="CENTER"
   wrapText="true" />
69                               <Button fx:id="
   ConfirmPurchaseButton" mnemonicParsing="false"
   onMouseClicked="#handlePurchase" text="Confirm" />
70                            </children>
71                         </VBox>
72                      </children>
73                   </AnchorPane>
74             </content>
75          </Tab>
76             <Tab onSelectionChanged="#resetAll" text="Sell
    Cards">
77                <content>
78                   <AnchorPane minHeight="0.0" minWidth="0.0"
    prefHeight="180.0" prefWidth="200.0">
79                      <children>
80                         <Label fx:id="SellingMoneyLabel"
```

```
80 alignment="CENTER" prefHeight="17.0" prefWidth="600.0"
   text="Label" textAlignment="CENTER" />
81                         <VBox alignment="CENTER" layoutX="
   14.0" layoutY="39.0" prefHeight="220.0" prefWidth="230.0"
   spacing="2.0">
82                            <children>
83                               <Label text="Select The
   Person And The Card" />
84                               <ListView fx:id="
   SellingPeopleListView" onMouseClicked="#resetAll"
   prefHeight="200.0" prefWidth="200.0" />
85                               <ListView fx:id="
   SellingCardsListView" onMouseClicked="#handleSellClick"
   prefHeight="218.0" prefWidth="222.0" />
86                            </children>
87                         </VBox>
88                         <ImageView fx:id="SellingImageView
   " fitHeight="220.0" fitWidth="136.0" layoutX="254.0"
   layoutY="39.0" pickOnBounds="true" preserveRatio="true"
   styleClass="images" />
89                         <VBox alignment="CENTER" layoutX="
   397.0" layoutY="39.0" prefHeight="220.0" prefWidth="200.0"
   >
90                            <children>
91                               <Label alignment="CENTER"
   contentDisplay="CENTER" prefHeight="52.0" prefWidth="190.0
   " text="How Much Do You Want To Sell?" textAlignment="
   CENTER" wrapText="true">
92                                  <font>
93                                     <Font size="9.0" />
94                                  </font>
95                               </Label>
96                               <Slider fx:id="SellingSlider
   " majorTickUnit="2.0" max="10.0" minorTickCount="1"
   onMouseClicked="#handleSellSliderClick" onMouseReleased="#
   handleSellSliderClick" prefHeight="0.0" prefWidth="100.0"
   showTickLabels="true" showTickMarks="true" snapToTicks="
   true" />
97                               <Label fx:id="
   SellingInfoLabel" alignment="CENTER" contentDisplay="
   CENTER" prefHeight="123.0" prefWidth="149.0" styleClass="
   describingLabel" text="Label" textAlignment="CENTER"
   wrapText="true" />
98                               <Button fx:id="
   ConfirmSellingButton" mnemonicParsing="false"
   onMouseClicked="#handleConfirmSelling" text="Confirm" />
```

```
 99                            </children>
100                          </VBox>
101                       </children>
102                    </AnchorPane>
103                 </content>
104              </Tab>
105              <Tab onSelectionChanged="#resetAll" text="
    Trade Cards">
106                 <content>
107                    <AnchorPane minHeight="0.0" minWidth="0.0"
     prefHeight="180.0" prefWidth="200.0">
108                       <children>
109                          <VBox alignment="CENTER" layoutX="
    408.0" layoutY="14.0" prefHeight="254.0" prefWidth="186.0"
    >
110                             <children>
111                                <Label text="Amount I'm
    Willing To Trade" />
112                                <Slider fx:id="
    MyTradingSlider" majorTickUnit="2.0" max="10.0"
    minorTickCount="1" onMouseClicked="#
    handleTradeAmountChange" onMouseReleased="#
    handleTradeAmountChange" showTickLabels="true"
    showTickMarks="true" snapToTicks="true" />
113                                <Label text="Amount I Want"
     />
114                                <Slider fx:id="
    MyAmountDesiredSlider" majorTickUnit="2.0" max="10.0"
    minorTickCount="1" onMouseClicked="#
    handleTradeAmountChange" onMouseReleased="#
    handleTradeAmountChange" showTickLabels="true"
    showTickMarks="true" snapToTicks="true" />
115                                <Label fx:id="
    TradingInfoLabel" alignment="CENTER" prefHeight="103.0"
    prefWidth="245.0" styleClass="describingLabel" text="Label
    " textAlignment="CENTER" wrapText="true" />
116                                <Button fx:id="
    ConfirmTradeButton" mnemonicParsing="false" onMouseClicked
    ="#handleConfirmTrade" text="Confirm Trade" />
117                             </children>
118                          </VBox>
119                          <VBox alignment="CENTER" layoutX="
    300.0" layoutY="15.0" prefHeight="249.0" prefWidth="100.0"
     spacing="2.0">
120                             <children>
121                                <ImageView fx:id="
```

```
121 MyTradeImageView" fitHeight="112.0" fitWidth="76.0"
    pickOnBounds="true" preserveRatio="true" styleClass="
    images" />
122                                 <ImageView fx:id="
    TheirTradeImageView" fitHeight="112.0" fitWidth="76.0"
    pickOnBounds="true" preserveRatio="true" styleClass="
    images" />
123                              </children>
124                           </VBox>
125                           <VBox layoutX="13.0" layoutY="11.0
    " prefHeight="254.0" prefWidth="271.0">
126                              <children>
127                                 <VBox alignment="CENTER"
    prefHeight="163.0" prefWidth="262.0">
128                                    <children>
129                                       <Label text="Select
    The 1st Trader And The Cards" />
130                                       <HBox prefHeight="122.
    0" prefWidth="271.0" spacing="2.0">
131                                          <children>
132                                             <ListView fx:id
    ="FirstPeopleListView" onMouseClicked="#resetAll"
    prefHeight="89.0" />
133                                             <ListView fx:id
    ="MyTradingListView" onMouseClicked="#
    handleTradingSelectionChange" prefHeight="104.0" />
134                                          </children>
135                                       </HBox>
136                                    </children>
137                                 </VBox>
138                                 <VBox alignment="CENTER"
    prefHeight="176.0" prefWidth="271.0">
139                                    <children>
140                                       <Label text="Select
    the 2nd Person And the Cards" />
141                                       <HBox prefHeight="168.
    0" prefWidth="271.0" spacing="2.0">
142                                          <children>
143                                             <ListView fx:id
    ="SecondPeopleListView" onMouseClicked="#resetAll"
    prefHeight="200.0" />
144                                             <ListView fx:id
    ="TraderListView" onMouseClicked="#
    handleTradingSelectionChange" prefHeight="200.0" />
145                                          </children>
146                                       </HBox>
```

```
147                                     </children>
148                                  </VBox>
149                               </children>
150                            </VBox>
151                         </children>
152                      </AnchorPane>
153                   </content>
154                </Tab>
155             </tabs>
156          </TabPane>
157          <HBox fx:id="IntroHBox" alignment="CENTER" layoutX="
     3.0" layoutY="50.0" prefHeight="31.0" prefWidth="590.0"
     spacing="5.0">
158             <children>
159                <Label text="Hello! What's your name?" />
160                <TextField fx:id="NameTextBox" />
161                <Button fx:id="SubmitNameButton"
     mnemonicParsing="false" onMouseClicked="#handleInitialize"
      prefHeight="27.0" prefWidth="78.0" text="Submit" />
162             </children>
163          </HBox>
164          <Label fx:id="DisplayNameLabel" alignment="CENTER"
     layoutX="4.0" layoutY="78.0" prefHeight="17.0" prefWidth="
     590.0" text="Label" visible="false" />
165       </children>
166 </AnchorPane>
167
```