```java
package com.example.template;

import java.util.ArrayList;

public class Player {
    private String name;
    private int amountOfTerritories = 0;
    private ArrayList<Territory> allTerritoriesControlled = new ArrayList<>();
    private ArrayList<Territory> attackTerritories = new ArrayList<>();
    private String color;
    private int credits = 100;

    private int playerID;

    private boolean lost = false;

    public Player(String name, int amountOfTerritories, Territory startingTerritory, String color, int playerID) {
        this.name = name;
        this.amountOfTerritories = amountOfTerritories;
        allTerritoriesControlled.add(startingTerritory);
        this.color = color;
        this.playerID = playerID;
    }

    public void playerLost() {
        lost = true;
    }

    public boolean getLost() {
        return lost;
    }

    public void removeFromCredits(int amountToRemove) {
        credits-=amountToRemove;
```

```
35      }
36
37      public void addToCredits(int amountToAdd) {
38          credits+=amountToAdd;
39      }
40
41      public int getCredits() {
42          return credits;
43      }
44
45      public int getPlayerID() {
46          return playerID;
47      }
48
49      public void addToAttackTerritories(Territory
    territory) {
50          attackTerritories.add(territory);
51      }
52
53      public ArrayList<Territory> getAttackTerritories
    () {
54          return attackTerritories;
55      }
56
57      public Player(String color) {
58          this.color = color;
59      }
60
61      public ArrayList<Territory>
    getAllTerritoriesControlled() {
62          return allTerritoriesControlled;
63      }
64
65      public String getName() {
66          return name;
67      }
68
69      public int getAmountOfTerritories() {
70          return amountOfTerritories;
71      }
72
```

```
73      public String getColor() {
74          return color;
75      }
76  }
77
```

```java
package com.example.template;

import java.util.ArrayList;

public class Territory {
    private String territoryType;
    private int territoryTypeID;
    private Player controllingPlayer;
    private Coordinates coordOfTerritory;
    private int amountOfSoldiers = 0;

    private int creditGain;
    private int baseAmountOfSoldiers;
    private int level = 1;
    private String description;
    private String imagePath;
    private int workerGain;
    private boolean soldierGainBool;
    private int soldierGain;

    private int territoryCost;

    private ArrayList<Integer> creditCostsForUpgrade
     = new ArrayList<>();

    public Territory(int creditGain, int
   baseAmountOfSoldiers, String description, String
   imagePath, int workerGain, int soldierGain, boolean
   soldierGainBool, int level, String nameWithCredits,
   int cost, int territoryTypeID, int Level2Cost, int
   Level3Cost, int Level4Cost) {
        this.creditGain = creditGain;
        this.amountOfSoldiers = baseAmountOfSoldiers;
        this.description = description;
        this.imagePath = imagePath;
        this.soldierGain = soldierGain;
        this.workerGain = workerGain;
        this.level = level;
        this.soldierGainBool = soldierGainBool;
        this.territoryCost = cost;
        this.territoryType = nameWithCredits;
```

```
36          this.territoryTypeID = territoryTypeID;
37          creditCostsForUpgrade.add(Level2Cost);
38          creditCostsForUpgrade.add(Level3Cost);
39          creditCostsForUpgrade.add(Level4Cost);
40      }
41
42      public Territory(int row, int column, int
    amountOfSoldiers, int territoryTypeID) {
43          coordOfTerritory = new Coordinates(row,column
    );
44          this.amountOfSoldiers = amountOfSoldiers;
45          this.territoryTypeID = territoryTypeID;
46      }
47
48      public Territory(int row, int column, Player
    controllingPlayer, int amountOfSoldiers,int
    territoryTypeID) {
49          coordOfTerritory = new Coordinates(row,column
    );
50          this.controllingPlayer = controllingPlayer;
51          this.amountOfSoldiers = amountOfSoldiers;
52          this.territoryTypeID = territoryTypeID;
53      }
54
55      public Territory(Territory territory) {
56          this.creditGain = territory.creditGain;
57          this.amountOfSoldiers = territory.
    amountOfSoldiers;
58          this.description = territory.description;
59          this.imagePath = territory.imagePath;
60          this.soldierGain = territory.soldierGain;
61          this.workerGain = territory.workerGain;
62          this.level = territory.level;
63          this.soldierGainBool = territory.
    soldierGainBool;
64          this.territoryCost = territory.territoryCost;
65          this.territoryType = territory.territoryType;
66          this.territoryTypeID = territory.
    territoryTypeID;
67          creditCostsForUpgrade.add(territory.
    creditCostsForUpgrade.get(0));
```

```
 68         creditCostsForUpgrade.add(territory.
    creditCostsForUpgrade.get(1));
 69         creditCostsForUpgrade.add(territory.
    creditCostsForUpgrade.get(2));
 70     }
 71
 72     public void upgradeLevel() {
 73         level++;
 74         amountOfSoldiers = amountOfSoldiers + (level
    *5);
 75         creditGain = creditGain + (level*15);
 76         if (soldierGainBool) {
 77             amountOfSoldiers+=20;
 78         }
 79     }
 80
 81     public void setCreditGain(int creditGain) {
 82         this.creditGain = creditGain;
 83     }
 84
 85     public ArrayList<Integer>
    getCreditCostsForUpgrade() {
 86         return creditCostsForUpgrade;
 87     }
 88
 89     public int getLevel() {
 90         return level;
 91     }
 92
 93     public String getDescription() {
 94         return description;
 95     }
 96
 97     public int getBaseAmountOfSoldiers() {
 98         return baseAmountOfSoldiers;
 99     }
100
101     public int getCreditGain() {
102         return creditGain;
103     }
104
```

```
105     public int getSoldierGain() {
106         return soldierGain;
107     }
108 
109     public boolean getSoldierGainBool() {
110         return soldierGainBool;
111     }
112 
113     public int getTerritoryCost() {
114         return territoryCost;
115     }
116 
117     public int getWorkerGain() {
118         return workerGain;
119     }
120 
121     public String getImagePath() {
122         return imagePath;
123     }
124 
125     public String getTerritoryType() {
126         return territoryType;
127     }
128 
129     public void setTerritoryTypeID(int
    territoryTypeID) {
130         this.territoryTypeID = territoryTypeID;
131     }
132 
133     public void setTerritoryType(String
    territoryType) {
134         this.territoryType = territoryType;
135     }
136 
137     public void setCoordOfTerritory(Coordinates
    coordOfTerritory) {
138         this.coordOfTerritory = coordOfTerritory;
139     }
140 
141     public void setAmountOfSoldiers(int
    amountOfSoldiers) {
```

```
142         this.amountOfSoldiers = amountOfSoldiers;
143     }
144 
145     public int getTerritoryTypeID() {
146         return territoryTypeID;
147     }
148 
149     public int getAmountOfSoldiers() {
150         return amountOfSoldiers;
151     }
152 
153     public Coordinates getCoordOfTerritory() {
154         return coordOfTerritory;
155     }
156 
157     public void setControllingPlayer(Player
    controllingPlayer) {
158         this.controllingPlayer = controllingPlayer;
159     }
160 
161     public Player getControllingPlayer() {
162         return controllingPlayer;
163     }
164 }
165 
```

```java
package com.example.template;

public class Coordinates {
    private int row;
    private int column;
//    private boolean isABorderCord = false;

    public Coordinates(int row, int column) {
        this.row = row;
        this.column = column;
    }

    public int getRow() {
        return row;
    }

    public int getColumn() {
        return column;
    }
}

```

```
 1 package com.example.template;
 2
 3 import javafx.event.ActionEvent;
 4 import javafx.event.EventHandler;
 5 import javafx.fxml.FXML;
 6 import javafx.scene.control.*;
 7 import javafx.scene.image.Image;
 8 import javafx.scene.image.ImageView;
 9 import javafx.scene.layout.AnchorPane;
10 import javafx.scene.layout.GridPane;
11 import javafx.scene.shape.Rectangle;
12
13 import javax.swing.JOptionPane;
14 import java.io.FileInputStream;
15 import java.io.FileNotFoundException;
16 import java.io.FileReader;
17 import java.io.PrintWriter;
18 import java.text.DecimalFormat;
19 import java.util.ArrayList;
20 import java.util.Scanner;
21
22 public class HelloController {
23     @FXML
24     public Button skipTurnButton;
25     @FXML
26     public Button addToPlayersButton;
27     @FXML
28     public Button removeFromPlayersButton;
29     @FXML
30     public Label amountOfPlayersLabel;
31     @FXML
32     public Button confirmPlayerCountButton;
33     @FXML
34     public Label SecondaryDisplayLabel;
35     private ArrayList<Territory> allBaseTerritories
    = new ArrayList<>();
36     public HelloController() {
37         allBaseTerritories.add(new Territory(40,10,"
   Helps Heal Soldiers (Generates Soldiers)","src/main/
   resources/images/hospital.png",0,10,true,1,"Hospital
    - 100 Credits",100,2,50,100,150));
```

```
38          allBaseTerritories.add(new Territory(100,15,"
   Generates a good amount of credits","src/main/
   resources/images/powerPlant.png",0,0,false,1,"Power
   Plant - 50 Credits",50,3,75,150,300));
39          allBaseTerritories.add(new Territory(75,20,"
   Many benefits including generating credits and
   transport workers and soldiers.","src/main/resources/
   images/airport.png",10,0,false,1,"Airport - 100
   Credits",100,4,100,200,400));
40          allBaseTerritories.add(new Territory(0,25,"
   Fight against raiders","src/main/resources/images/
   defenseTower1.png",0,5,true,1,"Defense Tower - 25
   Credits",25,5,25,50,100));
41          allBaseTerritories.add(new Territory(0,30,"
   Holds Soldiers and trains new ones","src/main/
   resources/images/militarybarracks.png",0,10,true,1,"
   Military Barracks - 50 Credits",50,6,50,100,150));
42          allBaseTerritories.add(new Territory(0,50,"
   contribues to condinated attacks around the map with
   a far range.","src/main/resources/images/
   militarybase1.png",0,0,true,1,"Military Base - 100
   Credits",100,7,150,300,600));
43          allBaseTerritories.add(new Territory(150,5,"
   Provides massive amounts of credit and many workers",
   "src/main/resources/images/city.png",20,0,false,1,"
   City - 100 Credits",100,8,100,150,300));
44          allBaseTerritories.add(new Territory(100,0,"
   Generates good amounts of credits but starts with
   little amounts of soldiers","src/main/resources/
   images/livingquarters.png",15,0,false,1,"Living
   Quarters - 50 Credits",50,9,50,100,200));
45      }
46      @FXML
47      public GridPane mainGridPane;
48      @FXML
49      public Button startButton;
50      @FXML
51      public Label attacksLeftLabel;
52      @FXML
53      public Label MainDisplayLabel;
54      @FXML
```

```
55      public Button turnDisplayButton;
56      @FXML
57      public Label currentTurnLabel;
58      @FXML
59      public Label creditsLabel;
60      @FXML
61      public CheckBox possibleOptionsCheckBox;
62      @FXML
63      public Rectangle blurRectangle;
64      @FXML
65      public GridPane secondaryGridPane;
66      @FXML
67      public Rectangle menuRectangle;
68      @FXML
69      public Label operationBoxLabel3;
70      @FXML
71      public Label operationBoxLabel2;
72      @FXML
73      public GridPane AttackGridPane;
74      @FXML
75      public Label operationBoxLabel1;
76      @FXML
77      public Label operationBoxHeaderLabel;
78      @FXML
79      public Button operationBoxCancelButton;
80      @FXML
81      public Button operationBoxAcceptButton;
82      @FXML
83      public AnchorPane operationBOX;
84      @FXML
85      public AnchorPane editTerritoryBOX; public
    Rectangle editTerritoryRectangle; public Button
    editTerritoryConfirmButton; public Button
    editTerritoryCancelButton; public Label
    editTerritoryActionLabel; public ListView
    editTerritoryConfigureListVIew; public ListView
    editTerritoryUpgradeListView; public Label
    editTerritoryHeaderLabel;
86      //    @FXML
87  //    public ImageView tempImageView;
88      Button[][] buttonArray = new Button[10][15];
```

```
 89     Button[][] attackButtonArray = new Button[3][3];
 90     Button[][] displayButtonArray = new Button[10][
    15];
 91     ImageView[][] imageViewArray = new ImageView[10
    ][15];
 92     Territory[][] territoryArray = new Territory[10
    ][15];
 93     private int numPlayers = 2;
 94     private int turn;
 95     private ArrayList<Player> allPlayers = new
    ArrayList<>();
 96     private boolean editingMode = false;
 97     private int attacksLeft = 5;
 98     private int editsLeft = 5;
 99     private int row;
100     private int column;
101
102     private boolean upgradingTerritory = false;
103
104     private Territory territoryChangingTo;
105     @FXML
106     public void startSetup() {
107         startButton.setVisible(false);
108         startButton.setDisable(true);
109         playerConfigUI(true,false);
110     }
111
112     public void playerConfigUI(boolean visible,
    boolean disable) {
113         removeFromPlayersButton.setVisible(visible);
114         removeFromPlayersButton.setDisable(disable);
115         addToPlayersButton.setDisable(disable);
116         addToPlayersButton.setVisible(visible);
117         confirmPlayerCountButton.setVisible(visible
    );
118         confirmPlayerCountButton.setDisable(disable
    );
119         amountOfPlayersLabel.setDisable(disable);
120         amountOfPlayersLabel.setVisible(visible);
121         if (!visible) {
122             removeFromPlayersButton.toBack();
```

```
123             addToPlayersButton.toBack();
124             confirmPlayerCountButton.toBack();
125             amountOfPlayersLabel.toBack();
126         }
127         else {
128             removeFromPlayersButton.toFront();
129             addToPlayersButton.toFront();
130             confirmPlayerCountButton.toFront();
131             amountOfPlayersLabel.toFront();
132         }
133     }
134     @FXML
135     public void addToPlayers() {
136         if (numPlayers<4) {
137             numPlayers++;
138             amountOfPlayersLabel.setText(numPlayers+
    " Players");
139         }
140     }
141 
142 
143     public void removeFromPlayers() {
144         if (numPlayers>2) {
145             numPlayers--;
146             amountOfPlayersLabel.setText(numPlayers+
    " Players");
147         }
148     }
149 
150     public void start() {
151         mainGridPane.setVisible(true);
152         playerConfigUI(false,true);
153         for(int row =0;row<buttonArray.length;row
    ++) {
154             for(int column=0;column<buttonArray[0].
    length;column++) {
155                 territoryArray[row][column] = new
    Territory(row,column, new Player("-fx-background-
    color: LightBlue;"),5,0);
156                 buttonArray[row][column] = new
    Button("");
```

```
157                 buttonArray[row][column].
    setPrefHeight(100);
158                 buttonArray[row][column].
    setPrefWidth(100);
159                 displayButtonArray[row][column] =
    new Button("");
160                 displayButtonArray[row][column].
    setPrefHeight(100);
161                 displayButtonArray[row][column].
    setPrefWidth(100);
162                 imageViewArray[row][column] = new
    ImageView();
163                 imageViewArray[row][column].
    setFitHeight(60);
164                 imageViewArray[row][column].
    setFitWidth(60);
165 //                Font font = new Font(0);
166 //                buttonArray[row][column].setFont(
    font);
167                 buttonArray[row][column].setStyle("-
    fx-border-color: LightBlue; -fx-background-color:
    LightBlue; -fx-border-radius: 0px; -fx-background-
    radius: 0px; ");
168                 displayButtonArray[row][column].
    setStyle("-fx-border-color: transparent; -fx-
    background-color: transparent; -fx-border-radius:
    0px; ");
169                 mainGridPane.add(buttonArray[row][
    column],column,row);
170                 secondaryGridPane.add(
    displayButtonArray[row][column],column,row);
171             }
172         }
173         setUpPlayers();
174         turn = (int) (Math.random()*numPlayers);
175         playTurn();
176         //printWorldBorders();
177         EventHandler z = new EventHandler<
    ActionEvent>() {
178             @Override
179             public void handle(ActionEvent event) {
```

```
180                 row = GridPane.getRowIndex((Button)
    event.getSource());
181                 column = GridPane.getColumnIndex((
    Button) event.getSource());
182                 SecondaryDisplayLabel.setText("");
183                 if (editingMode) {
184                     if (validTerritory(row, column,
    true)) {
185                         setUpEditUI(true,false);
186                         generateConfigureOptions();
187                         generateUpgradeOptions();
188                     }
189                     else {
190                         SecondaryDisplayLabel.
    setText("You cant edit that territory!");
191                     }
192                 }
193                 else {
194                     if (validTerritory(row, column,
    false)) {
195                         setUpAttackUI();
196 //                          territoryArray[row][column
    ].setControllingPlayer(allPlayers.get(turn%
    numPlayers));
197 //                          allPlayers.get(turn%
    numPlayers).getAllTerritoriesControlled().add(
    territoryArray[row][column]);
198 //                          updateMap();
199 //                          generateAreasToAttack();
200                     }
201                     else {
202                         SecondaryDisplayLabel.
    setText("you cant attack that!!!");
203                     }
204                 }
205             }
206         };
207         for(int row =0;row<buttonArray.length;row
    ++) {
208             for(int column=0;column<buttonArray[0].
    length;column++) {
```

```
209                 buttonArray[row][column].setOnAction
    (z);
210             }
211         }
212     }
213
214     public void generateConfigureOptions() {
215         editTerritoryConfigureListVIew.getItems().
    clear();
216         if (territoryArray[row][column].
    getTerritoryTypeID()==1 && territoryArray[row][
    column].getControllingPlayer().getColor().equals(
    allPlayers.get(turn%numPlayers).getColor())) {
217             editTerritoryConfigureListVIew.getItems
    ().add("THIS TERRITORY CANNOT BE CHANGED");
218         }
219         else {
220             editTerritoryConfigureListVIew.getItems
    ().add("Hospital - 100 Credits");
221             editTerritoryConfigureListVIew.getItems
    ().add("Power Plant - 50 Credits");
222             editTerritoryConfigureListVIew.getItems
    ().add("Airport - 100 Credits");
223             editTerritoryConfigureListVIew.getItems
    ().add("Defense Tower - 25 Credits");
224             editTerritoryConfigureListVIew.getItems
    ().add("Military Barracks - 50 Credits");
225             editTerritoryConfigureListVIew.getItems
    ().add("Military Base - 100 Credits");
226             editTerritoryConfigureListVIew.getItems
    ().add("City - 100 Credits");
227             editTerritoryConfigureListVIew.getItems
    ().add("Living Quarters - 50 Credits");
228         }
229         for(int i = 0;i<
    editTerritoryConfigureListVIew.getItems().size();i
    ++) {
230             if(editTerritoryConfigureListVIew.
    getItems().get(i).equals(territoryArray[row][column
    ].getTerritoryType())) {
231                 editTerritoryConfigureListVIew.
```

```
231 getItems().remove(i);
232                 i--;
233             }
234         }
235     }
236 
237     public void generateUpgradeOptions() {
238         editTerritoryUpgradeListView.getItems().
    clear();
239         if(territoryArray[row][column].
    getTerritoryTypeID()==0) {
240             editTerritoryUpgradeListView.getItems().
    add("THIS TERRITORY DOES NOT HAVE UPGRADE OPTIONS");
241         }
242         else {
243             for (int i = territoryArray[row][column
    ].getLevel();i<4;i++) {
244                 editTerritoryUpgradeListView.
    getItems().add(getNameWithoutCredits(territoryArray[
    row][column].getCreditCostsForUpgrade().get(i-1)+"
    Credits - "+territoryArray[row][column].
    getTerritoryType())+" Upgrade "+(i+1));
245             }
246         }
247     }
248 
249     public void setUpEditUI(boolean visible, boolean
     disable) {
250         boolean printOnLeft = getPrintSide();
251         editTerritoryBOX.setVisible(visible);
252         editTerritoryBOX.setDisable(disable);
253         blurRectangle.setDisable(disable);
254         blurRectangle.setVisible(visible);
255         if (printOnLeft) {
256             editTerritoryBOX.setLayoutX(25);
257         }
258         else {
259             editTerritoryBOX.setLayoutX(25+750);
260         }
261         if (visible) {
262             blurRectangle.toFront();
```

```
263             editTerritoryBOX.toFront();
264         }
265         else {
266             blurRectangle.toBack();
267             editTerritoryBOX.toBack();
268         }
269     }
270 
271     private boolean firstTime = true;
272     private Boolean[][] activatedButtons = new
    Boolean[3][3];
273     public void setUpAttackUI() {
274         openAttackPlan();
275         if (firstTime) {
276             firstTime = false;
277             setUpAttackGridPane();
278         }
279         for(int row =0;row<3;row++) {
280             for(int column=0;column<3;column++) {
281                 activatedButtons[row][column] =
    false;
282                 attackButtonArray[row][column].
    setDisable(false);
283             }
284         }
285         displayAttackGridPane();
286         tallyUpTotals();
287     }
288 
289     public void displayAttackGridPane() {
290         ArrayList<Boolean> directionsCanGo =
    findDirections(territoryArray[row][column]);
291         if (directionsCanGo.get(0)) {
292             attackButtonArray[0][1].setStyle(
    buttonArray[row-1][column].getStyle());
293         }
294         else {
295             attackButtonArray[0][1].setStyle("-fx-
    border-color: Black; -fx-background-color: Black;");
296             attackButtonArray[0][1].setDisable(true
    );
```

```
297         }
298         if (directionsCanGo.get(1)) {
299             attackButtonArray[0][2].setStyle(
    buttonArray[row-1][column+1].getStyle());
300         }
301         else {
302             attackButtonArray[0][2].setStyle("-fx-
    border-color: Black; -fx-background-color: Black;");
303             attackButtonArray[0][2].setDisable(true
    );
304         }
305         if (directionsCanGo.get(2)) {
306             attackButtonArray[1][2].setStyle(
    buttonArray[row][column+1].getStyle());
307         }
308         else {
309             attackButtonArray[1][2].setStyle("-fx-
    border-color: Black; -fx-background-color: Black;");
310             attackButtonArray[1][2].setDisable(true
    );
311         }
312         if (directionsCanGo.get(3)) {
313             attackButtonArray[2][2].setStyle(
    buttonArray[row+1][column+1].getStyle());
314         }
315         else {
316             attackButtonArray[2][2].setStyle("-fx-
    border-color: Black; -fx-background-color: Black;");
317             attackButtonArray[2][2].setDisable(true
    );
318         }
319         if (directionsCanGo.get(4)) {
320             attackButtonArray[2][1].setStyle(
    buttonArray[row+1][column].getStyle());
321         }
322         else {
323             attackButtonArray[2][1].setStyle("-fx-
    border-color: Black; -fx-background-color: Black;");
324             attackButtonArray[2][1].setDisable(true
    );
325         }
```

```
326         if (directionsCanGo.get(5)) {
327             attackButtonArray[2][0].setStyle(
    buttonArray[row+1][column-1].getStyle());
328         }
329         else {
330             attackButtonArray[2][0].setStyle("-fx-
    border-color: Black; -fx-background-color: Black;");
331             attackButtonArray[2][0].setDisable(true
    );
332         }
333         if (directionsCanGo.get(6)) {
334             attackButtonArray[1][0].setStyle(
    buttonArray[row][column-1].getStyle());
335         }
336         else {
337             attackButtonArray[1][0].setStyle("-fx-
    border-color: Black; -fx-background-color: Black;");
338             attackButtonArray[1][0].setDisable(true
    );
339         }
340         if (directionsCanGo.get(7)) {
341             attackButtonArray[0][0].setStyle(
    buttonArray[row-1][column-1].getStyle());
342         }
343         else {
344             attackButtonArray[0][0].setStyle("-fx-
    border-color: Black; -fx-background-color: Black;");
345             attackButtonArray[0][0].setDisable(true
    );
346         }
347         attackButtonArray[1][1].setStyle("-fx--fx-
    border-color: Navy Blue; -fx-background-color: Navy
    Blue;");
348     }
349
350     public void setUpAttackGridPane() {
351         for(int row =0;row<attackButtonArray.length;
    row++) {
352             for(int column=0;column<
    attackButtonArray[0].length;column++) {
353                 attackButtonArray[row][column] = new
```

```
353  Button("");
354                 attackButtonArray[row][column].
    setPrefHeight(200);
355                 attackButtonArray[row][column].
    setPrefWidth(200);
356                 attackButtonArray[row][column].
    setStyle("-fx-background-radius: 0px; -fx-border-
    color: LightBlue; -fx-background-color: LightBlue;"
    );
357                 AttackGridPane.add(attackButtonArray
    [row][column],column,row);
358             }
359         }
360         EventHandler x = new EventHandler<
    ActionEvent>() {
361             @Override
362             public void handle(ActionEvent event) {
363                 int row1 = GridPane.getRowIndex((
    Button) event.getSource());
364                 int column1 = GridPane.
    getColumnIndex((Button) event.getSource());
365                 Coordinates coords =
    findCoordinatesOnBoard(row1, column1);
366                 if (buttonClickedIsValid(row1,
    column1,coords)) {
367                     if (activatedButtons[row1][
    column1]) {
368                         attackButtonArray[row1][
    column1].setStyle(allPlayers.get(turn%numPlayers).
    getColor() +" -fx-border-color: Black; -fx-border-
    width: 0px;");
369                         activatedButtons[row1][
    column1] = false;
370                     }
371                     else {
372                         activatedButtons[row1][
    column1] = true;
373                         attackButtonArray[row1][
    column1].setStyle(allPlayers.get(turn%numPlayers).
    getColor()+" -fx-border-color: #ff00ab; -fx-border-
    width: 5px;");
```

```
374                     }
375                     tallyUpTotals();
376                 }
377             }
378         };
379         for(int row =0;row<attackButtonArray.length;
    row++) {
380             for(int column=0;column<
    attackButtonArray[0].length;column++) {
381                 attackButtonArray[row][column].
    setOnAction(x);
382             }
383         }
384     }
385     private double winProbability = .50;
386     public void tallyUpTotals() {
387         double yourTotal = 0;
388         int enemyTotal = territoryArray[row][column
    ].getAmountOfSoldiers();
389         for(int row =0;row<3;row++) {
390             for(int column=0;column<3;column++) {
391                 if (activatedButtons[row][column]) {
392                     Coordinates coords =
    findCoordinatesOnBoard(row,column);
393                     yourTotal = yourTotal +
    territoryArray[coords.getRow()][coords.getColumn()].
    getAmountOfSoldiers();
394                 }
395             }
396         }
397         winProbability = (yourTotal/(enemyTotal+1))/
    2;
398         if (winProbability>.99 || enemyTotal==0) {
399             winProbability = 1;
400         }
401         if (winProbability<0 || yourTotal==0) {
402             winProbability = 0;
403         }
404         DecimalFormat df = new DecimalFormat("#.##"
    );
405         operationBoxLabel1.setText("Your Soldiers: "
```

```
405 +df.format(yourTotal));
406         operationBoxLabel2.setText("Enemy Soldiers
    : "+enemyTotal);
407         operationBoxLabel3.setText("Win Percentage
    : "+df.format(winProbability*100)+"%");
408     }
409 
410     public boolean buttonClickedIsValid(int row2,
    int column2,Coordinates coords) {
411         return territoryArray[coords.getRow()][
    coords.getColumn()].getControllingPlayer().
    getPlayerID() == allPlayers.get(turn%numPlayers).
    getPlayerID();
412     }
413 
414     public Coordinates findCoordinatesOnBoard(int
    row3, int column3) {
415         if (row3==0 && column3==0) {
416             return new Coordinates(row-1,column-1);
417         }
418         if (row3==0 && column3==1) {
419             return new Coordinates(row-1,column);
420         }
421         if (row3==0 && column3==2) {
422             return new Coordinates(row-1,column+1);
423         }
424         if (row3==1 && column3==0) {
425             return new Coordinates(row,column-1);
426         }
427         if (row3==1 && column3==1) {
428             return new Coordinates(row,column);
429         }
430         if (row3==1 && column3==2) {
431             return new Coordinates(row,column+1);
432         }
433         if (row3==2 && column3==0) {
434             return new Coordinates(row+1,column-1);
435         }
436         if (row3==2 && column3==1) {
437             return new Coordinates(row+1,column);
438         }
```

```
439         if (row3==2 && column3==2) {
440             return new Coordinates(row+1,column+1);
441         }
442         return new Coordinates(0,0);
443     }
444
445     public void openAttackPlan() {
446         setUpSecondary();
447         boolean printOnLeft = getPrintSide();
448         attackBoxUI(true,false,printOnLeft);
449     }
450
451     public boolean getPrintSide() {
452         return column>6;
453     }
454
455     public void attackBoxUI(boolean visisble,
    boolean disable,boolean leftSide) {
456         blurRectangle.setVisible(visisble);
457         blurRectangle.setDisable(disable);
458         operationBOX.setVisible(visisble);
459         operationBOX.setDisable(disable);
460         if (visisble) {
461             blurRectangle.toFront();
462             operationBOX.toFront();
463         }
464         else {
465             blurRectangle.toBack();
466             operationBOX.toBack();
467             secondaryGridPane.toBack();
468         }
469         if (leftSide) {
470             operationBOX.setLayoutX(25);
471         }
472         else {
473             operationBOX.setLayoutX(775);
474         }
475
476     }
477
478     public void setUpSecondary() {
```

```
479         secondaryGridPane.toFront();
480         for(int row =0;row<displayButtonArray.length
    ;row++) {
481             for(int column=0;column<
    displayButtonArray[0].length;column++) {
482                 displayButtonArray[row][column].
    setStyle("-fx-border-color: transparent; -fx-
    background-color: transparent;");
483             }
484         }
485         displayButtonArray[row][column].setStyle(
    buttonArray[row][column].getStyle());
486     }
487
488     public boolean validTerritory(int rowClicked,
    int columnClicked, boolean editing) {
489         if (editing) {
490             return territoryArray[rowClicked][
    columnClicked].getControllingPlayer().getColor().
    equals(allPlayers.get(turn%numPlayers).getColor());
491         }
492         else {
493             for (int i =0;i<allPlayers.get(turn%
    numPlayers).getAttackTerritories().size();i++) {
494                 if (allPlayers.get(turn%numPlayers).
    getAttackTerritories().get(i).getCoordOfTerritory().
    getRow()==rowClicked && allPlayers.get(turn%
    numPlayers).getAttackTerritories().get(i).
    getCoordOfTerritory().getColumn()==columnClicked) {
495                     return true;
496                 }
497             }
498         }
499         return false;
500     }
501
502     public void playTurn() {
503         if (allPlayers.get(turn%numPlayers).getLost
    ()) {
504             turn++;
505             playTurn();
```

```
506         }
507         else {
508             updateMap();
509             setUpAttackingMode();
510             generateAreasToAttack();
511             addCreditsToAllPlayers();
512             SecondaryDisplayLabel.setText("");
513         }
514     }
515 
516     public void addCreditsToAllPlayers() {
517         for (int i = 0;i<allPlayers.size();i++) {
518             if (!allPlayers.get(i).getLost()) {
519                 int creditTotalForPlayer = 0;
520                 for (int row = 0; row <
    territoryArray.length; row++) {
521                     for (int column = 0; column <
    territoryArray[0].length; column++) {
522                         if (territoryArray[row][
    column].getControllingPlayer().getColor().equals(
    allPlayers.get(i).getColor())) {
523                             creditTotalForPlayer +=
    territoryArray[row][column].getCreditGain();
524                         }
525                     }
526                 }
527                 allPlayers.get(i).addToCredits(
    creditTotalForPlayer);
528             }
529         }
530     }
531 
532     public void setUpEditingMode() {
533         editsLeft = 5;
534         MainDisplayLabel.setText("Editing Mode.
    Click on your territories to upgrade them.");
535         currentTurnLabel.setText("Current Turn:");
536         turnDisplayButton.setDisable(false);
537         turnDisplayButton.setVisible(true);
538         editingMode = true;
539         turnDisplayButton.setStyle("-fx-background-
```

```
539 radius: 0px;");
540         attacksLeftLabel.setText("Edits Left: "+
    editsLeft);
541         turnDisplayButton.setStyle(allPlayers.get(
    turn%numPlayers).getColor());
542         skipTurnButton.setDisable(false);
543         skipTurnButton.setVisible(true);
544         possibleOptionsCheckBox.setVisible(false);
545         possibleOptionsCheckBox.setDisable(true);
546         possibleOptionsCheckBox.setSelected(false);
547     }
548 
549     @FXML
550     public void onClickEndEditingMode() {
551         if (editingMode) {
552             turn++;
553             playTurn();
554             skipTurnButton.setDisable(true);
555             skipTurnButton.setVisible(false);
556         }
557     }
558     public void generateAreasToAttack() {
559         allPlayers.get(turn%numPlayers).
    getAttackTerritories().clear();
560         for (int i = 0; i<allPlayers.get(turn%
    numPlayers).getAllTerritoriesControlled().size();i
    ++) {
561             ArrayList<Boolean> directionsCanGo =
    findDirections(allPlayers.get(turn%numPlayers).
    getAllTerritoriesControlled().get(i));
562             convertDirectionsToCoordinates(
    directionsCanGo, allPlayers.get(turn%numPlayers).
    getAllTerritoriesControlled().get(i));
563         }
564         removeRedundantTerritories();
565     }
566 
567     public void removeRedundantTerritories() {
568         for (int i = 0; i<allPlayers.get(turn%
    numPlayers).getAttackTerritories().size();i++) {
569             if(allPlayers.get(turn%numPlayers).
```

```
569 getAttackTerritories().get(i).getControllingPlayer
    ().getColor().equals(allPlayers.get(turn%numPlayers
    ).getColor())) {
570                 allPlayers.get(turn%numPlayers).
    getAttackTerritories().remove(i);
571                 i--;
572             }
573         }
574     }
575
576     public void convertDirectionsToCoordinates(
    ArrayList<Boolean> directions, Territory
    territoryChecking) {
577         if (directions.get(0)) {
578             allPlayers.get(turn%numPlayers).
    getAttackTerritories().add(territoryArray[
    territoryChecking.getCoordOfTerritory().getRow()-1][
    territoryChecking.getCoordOfTerritory().getColumn
    ()]);
579         }
580         if (directions.get(1)) {
581             allPlayers.get(turn%numPlayers).
    getAttackTerritories().add(territoryArray[
    territoryChecking.getCoordOfTerritory().getRow()-1][
    territoryChecking.getCoordOfTerritory().getColumn()+
    1]);
582         }
583         if (directions.get(2)) {
584             allPlayers.get(turn%numPlayers).
    getAttackTerritories().add(territoryArray[
    territoryChecking.getCoordOfTerritory().getRow()][
    territoryChecking.getCoordOfTerritory().getColumn()+
    1]);
585         }
586         if (directions.get(3)) {
587             allPlayers.get(turn%numPlayers).
    getAttackTerritories().add(territoryArray[
    territoryChecking.getCoordOfTerritory().getRow()+1][
    territoryChecking.getCoordOfTerritory().getColumn()+
    1]);
588         }
```

```
589         if (directions.get(4)) {
590             allPlayers.get(turn%numPlayers).
    getAttackTerritories().add(territoryArray[
    territoryChecking.getCoordOfTerritory().getRow()+1][
    territoryChecking.getCoordOfTerritory().getColumn
    ()]);
591         }
592         if (directions.get(5)) {
593             allPlayers.get(turn%numPlayers).
    getAttackTerritories().add(territoryArray[
    territoryChecking.getCoordOfTerritory().getRow()+1][
    territoryChecking.getCoordOfTerritory().getColumn()-
    1]);
594         }
595         if (directions.get(6)) {
596             allPlayers.get(turn%numPlayers).
    getAttackTerritories().add(territoryArray[
    territoryChecking.getCoordOfTerritory().getRow()][
    territoryChecking.getCoordOfTerritory().getColumn()-
    1]);
597         }
598         if (directions.get(7)) {
599             allPlayers.get(turn%numPlayers).
    getAttackTerritories().add(territoryArray[
    territoryChecking.getCoordOfTerritory().getRow()-1][
    territoryChecking.getCoordOfTerritory().getColumn()-
    1]);
600         }
601     }
602
603     public ArrayList<Boolean> findDirections(
    Territory territoryChecking) {
604         ArrayList<Boolean> tempList = new ArrayList
    <>();
605         for (int i = 0 ;i<8;i++) {
606             tempList.add(false);
607         }
608         if(territoryChecking.getCoordOfTerritory().
    getColumn()+1<=(buttonArray[0].length-1)) {
609             tempList.set(2,true);
610         }
```

```
611         if(territoryChecking.getCoordOfTerritory().
    getColumn()>0) {
612             tempList.set(6,true);
613         }
614         if(territoryChecking.getCoordOfTerritory().
    getRow()+1<=(buttonArray.length-1)) {
615             tempList.set(4,true);
616         }
617         if(territoryChecking.getCoordOfTerritory().
    getRow()>0) {
618             tempList.set(0,true);
619         }
620         if (tempList.get(0) && tempList.get(2)) {
621             tempList.set(1,true);
622         }
623         if (tempList.get(2) && tempList.get(4)) {
624             tempList.set(3,true);
625         }
626         if (tempList.get(4) && tempList.get(6)) {
627             tempList.set(5,true);
628         }
629         if (tempList.get(0) && tempList.get(6)) {
630             tempList.set(7,true);
631         }
632         return tempList;
633     }
634 
635     public void setUpAttackingMode() {
636         editingMode = false;
637         attacksLeft = 5;
638         MainDisplayLabel.setText("Attack Mode. Click
     On The Surrounding Territories To Invade.");
639         currentTurnLabel.setText("Current Turn:");
640         turnDisplayButton.setDisable(false);
641         turnDisplayButton.setVisible(true);
642         turnDisplayButton.setStyle("-fx-background-
    radius: 0px;");
643         turnDisplayButton.setStyle(allPlayers.get(
    turn%numPlayers).getColor());
644         attacksLeftLabel.setText("Attacks Left: "+
    attacksLeft);
```

```
645         skipTurnButton.setDisable(true);
646         skipTurnButton.setVisible(false);
647         possibleOptionsCheckBox.setVisible(true);
648         possibleOptionsCheckBox.setDisable(false);
649     }
650 
651 
652     public void setUpPlayers() {
653         for (int i =0;i<numPlayers;i++) {
654             if (i==0) {
655                 allPlayers.add(new Player("temp",1,
    new Territory(0,0,10,1), "-fx-background-color: Blue
    ;", (int) (Math.random()*10000)));
656                 territoryArray[0][0].
    setControllingPlayer(allPlayers.get(allPlayers.size
    ()-1));
657                 territoryArray[0][0].
    setTerritoryTypeID(1);
658                 territoryArray[0][0].setCreditGain(
    100);
659                 territoryArray[0][0].
    setAmountOfSoldiers(10);
660             }
661             else if (i==1) {
662                 allPlayers.add(new Player("temp",1,
    new Territory(9,14,10,1), "-fx-background-color: Red
    ;", (int) (Math.random()*10000)));
663                 territoryArray[9][14].
    setControllingPlayer(allPlayers.get(allPlayers.size
    ()-1));
664                 territoryArray[9][14].
    setTerritoryTypeID(1);
665                 territoryArray[9][14].setCreditGain(
    100);
666                 territoryArray[9][14].
    setAmountOfSoldiers(10);
667             }
668             else if (i==2) {
669                 allPlayers.add(new Player("temp",1,
    new Territory(9,0,10,1), "-fx-background-color:
    Yellow;", (int) (Math.random()*10000)));
```

```
670                 territoryArray[9][0].
    setControllingPlayer(allPlayers.get(allPlayers.size
    ()-1));
671                 territoryArray[9][0].
    setTerritoryTypeID(1);
672                 territoryArray[9][0].setCreditGain(
    100);
673                 territoryArray[9][0].
    setAmountOfSoldiers(10);
674             }
675             else {
676                 allPlayers.add(new Player("temp",1,
    new Territory(0,14,10,1), "-fx-background-color:
    Green;", (int) (Math.random()*10000)));
677                 territoryArray[0][14].
    setControllingPlayer(allPlayers.get(allPlayers.size
    ()-1));
678                 territoryArray[0][14].
    setTerritoryTypeID(1);
679                 territoryArray[0][14].setCreditGain(
    100);
680                 territoryArray[0][14].
    setAmountOfSoldiers(10);
681             }
682         }
683         updateMap();
684     }
685
686     public void updateMap() {
687         setUpImageViews();
688         updateCredits();
689         for(int row =0;row<territoryArray.length;row
    ++) {
690             for(int column=0;column<territoryArray[0
    ].length;column++) {
691                 buttonArray[row][column].setStyle(
    territoryArray[row][column].getControllingPlayer().
    getColor());
692                 buttonArray[row][column].setGraphic(
    imageViewArray[row][column]);
693             }
```

```
694         }
695     }
696 
697     public void updateCredits() {
698         creditsLabel.setText("Credits: "+allPlayers.
    get(turn%numPlayers).getCredits());
699     }
700 
701     public void setUpImageViews() {
702         for(int row =0;row<imageViewArray.length;row
    ++) {
703             for(int column=0;column<imageViewArray[0
    ].length;column++) {
704                 try {
705                     if (territoryArray[row][column].
    getTerritoryTypeID()==0) {
706                         //L Bozo
707                     }
708                     else if (territoryArray[row][
    column].getTerritoryTypeID()==1) {
709                         FileInputStream input = new
    FileInputStream("src/main/resources/images/homeBase.
    png");
710                         imageViewArray[row][column].
    setImage(new Image(input));
711                         buttonArray[row][column].
    setGraphic(imageViewArray[row][column]);
712                     }
713                     else {
714                         FileInputStream input = new
    FileInputStream(territoryArray[row][column].
    getImagePath());
715                         imageViewArray[row][column].
    setImage(new Image(input));
716                         buttonArray[row][column].
    setGraphic(imageViewArray[row][column]);
717                     }
718                 }catch (FileNotFoundException e) {
719                     e.printStackTrace();
720                 }
721             }
```

```
722         }
723     }
724 
725     @FXML
726     public void onClickShowAttackOptions() {
727         if (!editingMode) {
728             if(possibleOptionsCheckBox.isSelected
    ()) {
729                 for (int i =0;i<allPlayers.get(turn%
    numPlayers).getAttackTerritories().size();i++) {
730                     buttonArray[allPlayers.get(turn%
    numPlayers).getAttackTerritories().get(i).
    getCoordOfTerritory().getRow()][allPlayers.get(turn%
    numPlayers).getAttackTerritories().get(i).
    getCoordOfTerritory().getColumn()].setStyle("-fx-
    border-color: Cyan; -fx-background-color: Cyan;");
731                 }
732             }
733             else {
734                 for (int i =0;i<allPlayers.get(turn%
    numPlayers).getAttackTerritories().size();i++) {
735                     buttonArray[allPlayers.get(turn%
    numPlayers).getAttackTerritories().get(i).
    getCoordOfTerritory().getRow()][allPlayers.get(turn%
    numPlayers).getAttackTerritories().get(i).
    getCoordOfTerritory().getColumn()].setStyle("-fx-
    border-color: LightBlue; -fx-background-color:
    LightBlue;");
736                 }
737             }
738         }
739     }
740 
741     @FXML
742     public void cancelInvasion() {
743         attackBoxUI(false,true,true);
744     }
745 
746     @FXML
747     public void attemptInvasion() {
748         double randInt = Math.random();
```

```
749         if (randInt<winProbability) {
750             territoryArray[row][column].
    setControllingPlayer(allPlayers.get(turn%numPlayers
    ));
751             allPlayers.get(turn%numPlayers).
    getAllTerritoriesControlled().add(territoryArray[row
    ][column]);
752             boolean someoneWon = checkIfLoses();
753             if (!someoneWon) {
754                 updateMap();
755                 generateAreasToAttack();
756                 attackBoxUI(false,true,true);
757                 SecondaryDisplayLabel.setText("You
    have invaded the territory.");
758             }
759         }
760         else {
761             SecondaryDisplayLabel.setText("The
    invasion failed.");
762             attackBoxUI(false,true,true);
763         }
764         attacksLeft--;
765         if(attacksLeft==0) {
766             setUpEditingMode();
767         }
768         else {
769             attacksLeftLabel.setText("Attacks Left
    : "+attacksLeft);
770         }
771     }
772
773     public boolean checkIfLoses() {
774         boolean won = false;
775         for (int i = 0;i<allPlayers.size();i++) {
776             boolean playerLost = true;
777             for(int row =0;row<territoryArray.length
    ;row++) {
778                 for(int column=0;column<
    territoryArray[0].length;column++) {
779                     if (territoryArray[row][column].
    getControllingPlayer().getColor().equals(allPlayers.
```

```
779 get(i).getColor())) {
780                         if (territoryArray[row][
    column].getTerritoryTypeID()==1) {
781                             playerLost = false;
782                         }
783                     }
784                 }
785             }
786             if (playerLost) {
787                 allPlayers.get(i).playerLost();
788                 for(int row =0;row<territoryArray.
    length;row++) {
789                     for(int column=0;column<
    territoryArray[0].length;column++) {
790                         if (territoryArray[row][
    column].getControllingPlayer().getColor().equals(
    allPlayers.get(i).getColor())) {
791                             territoryArray[row][
    column].setControllingPlayer(new Player("-fx-
    background-color: LightBlue;"));
792                         }
793                     }
794                 }
795             }
796         }
797         int numPlayersLeft = 0;
798         for (int i =0;i<allPlayers.size();i++) {
799             if (!allPlayers.get(i).getLost()) {
800                 numPlayersLeft++;
801             }
802         }
803         if (numPlayersLeft==1) {
804             for (int i = 0;i<allPlayers.size();i
    ++) {
805                 if (!allPlayers.get(i).getLost()) {
806                     blurRectangle.setVisible(true);
807                     blurRectangle.setDisable(false);
808                     blurRectangle.toFront();
809                     won = true;
810                     if (i==0) {
811                         MainDisplayLabel.setText("
```

```
811 Blue Wins!!!");
812                     }
813                     if (i==1) {
814                         MainDisplayLabel.setText("
    Red Wins!!!");
815                     }
816                     if (i==2) {
817                         MainDisplayLabel.setText("
    Yellow Wins!!!");
818                     }
819                     if (i==3) {
820                         MainDisplayLabel.setText("
    Green Wins!!!");
821                     }
822                 }
823             }
824         }
825         return won;
826     }
827 
828     @FXML
829     public void upgradeTerritory() {
830         if (territoryChangingTo!=null) {
831             if (!upgradingTerritory) {
832                 if (territoryChangingTo.
    getTerritoryCost()<=allPlayers.get(turn%numPlayers).
    getCredits()) {
833                     territoryArray[row][column] =
    territoryChangingTo;
834                     territoryArray[row][column].
    setControllingPlayer(allPlayers.get(turn%numPlayers
    ));
835                     territoryArray[row][column].
    setCoordOfTerritory(new Coordinates(row,column));
836                     allPlayers.get(turn%numPlayers).
    removeFromCredits(territoryChangingTo.
    getTerritoryCost());
837                     updateMap();
838                     setUpEditUI(false,true);
839                     editsLeft--;
840                     attacksLeftLabel.setText("Edits
```

```
840 Left: "+editsLeft);
841                 }
842                 else {
843                     SecondaryDisplayLabel.setText("
    You Don't Have Enough Credits For That!!!");
844                 }
845             }
846             else {
847                 if (territoryArray[row][column].
    getLevel()+1==Integer.parseInt(
    editTerritoryUpgradeListView.getSelectionModel().
    getSelectedItem().toString().substring(
    editTerritoryUpgradeListView.getSelectionModel().
    getSelectedItem().toString().length()-1))) {
848                     if (territoryArray[row][column].
    getCreditCostsForUpgrade().get(territoryArray[row][
    column].getLevel()-1)<=allPlayers.get(turn%
    numPlayers).getCredits()) {
849                         allPlayers.get(turn%
    numPlayers).removeFromCredits(territoryArray[row][
    column].getCreditCostsForUpgrade().get(
    territoryArray[row][column].getLevel()-1));
850                         territoryArray[row][column].
    upgradeLevel();
851                         updateMap();
852                         setUpEditUI(false,true);
853                         editsLeft--;
854                         attacksLeftLabel.setText("
    Edits Left: "+editsLeft);
855                     }
856                     else {
857                         SecondaryDisplayLabel.
    setText("You Don't Have Enough Credits For That!!!"
    );
858                     }
859                 }
860                 else {
861                     SecondaryDisplayLabel.setText("
    You cant upgrade more than 1 level at a time!");
862                 }
863             }
```

```
864             if (editsLeft==0) {
865                 turn++;
866                 editsLeft = 5;
867                 playTurn();
868             }
869         }
870     }
871
872     @FXML
873     public void cancelUpgrade() {
874         setUpEditUI(false,true);
875     }
876
877     @FXML
878     public void selectTerritoryConfigure() {
879         if (editTerritoryConfigureListVIew.
    getSelectionModel().getSelectedItem()!=null) {
880             editTerritoryActionLabel.setText("Action
    : Change Into "+getNameWithoutCredits((String)
    editTerritoryConfigureListVIew.getSelectionModel().
    getSelectedItem()));
881             territoryChangingTo =
    returnTerritoryThatWasClicked();
882             editTerritoryConfirmButton.setText("
    CHANGE TERRITORY");
883             upgradingTerritory = false;
884         }
885     }
886
887     public Territory returnTerritoryThatWasClicked
    () {
888         for (int i = 0; i<allBaseTerritories.size();
    i++) {
889             if(allBaseTerritories.get(i).
    getTerritoryType().equals(
    editTerritoryConfigureListVIew.getSelectionModel().
    getSelectedItem())) {
890                 return new Territory(
    allBaseTerritories.get(i));
891             }
892         }
```

```
893             return new Territory(row,column,5,0);
894         }
895 
896         public String getNameWithoutCredits(String str
    ) {
897             String tempStr = "";
898             for (int i =0;i<str.length();i++) {
899                 if (str.charAt(i) == '-') {
900                     break;
901                 }
902                 else {
903                     tempStr = tempStr + str.charAt(i);
904                 }
905             }
906             tempStr = tempStr.substring(0,tempStr.length
    ()-1);
907             return tempStr;
908         }
909 
910         @FXML
911         public void selectTerritoryUpgrade() {
912             if (editTerritoryUpgradeListView.
    getSelectionModel().getSelectedItem()!=null) {
913                 editTerritoryActionLabel.setText("Action
    : Upgrade Into Level "+editTerritoryUpgradeListView.
    getSelectionModel().getSelectedItem().toString().
    substring(editTerritoryUpgradeListView.
    getSelectionModel().getSelectedItem().toString().
    length()-1));
914                 upgradingTerritory = true;
915                 editTerritoryConfirmButton.setText("
    UPGRADE TERRITORY");
916             }
917         }
918 }
```

```java
package com.example.template;

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(), 1200, 900);
        stage.setTitle("Infrastructure Risk");
        stage.setScene(scene);
        stage.show();
    }

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