```java
package com.example.battlegame;

import javafx.scene.control.ListView;

import java.util.ArrayList;
import java.util.Random;

public class Shop {
    private Items[] itemList;

    private ArrayList<Items> currentList = new ArrayList<>();

    public Shop(Items[] itemList){
        this.itemList = itemList;
    }

    protected void randomizeShop(ListView listView){
        ArrayList<String> tempList = new ArrayList<>();
        currentList.clear();

        for (int i = 0; i < 10; i++) {
            Random random = new Random();
            int index = random.nextInt(itemList.length);
            tempList.add(itemList[index].getName() + "; Price: " + itemList[index].getShopPrice());
            currentList.add(itemList[index]);
        }

        listView.getItems().clear();
        listView.getItems().addAll(tempList);
    }

    public ArrayList<Items> getCurrentList() {
        return currentList;
    }
}

```

```java
package com.example.battlegame;

public class Items {
    private String name;

    private double damage;
    private double healing;
    private double speed;
    private double defense;

    private int shopPrice;

    public Items(String name, double damage, double healing, double speed, double defense, int shopPrice){
        this.name = name;

        this.damage = damage;
        this.healing = healing;
        this.speed = speed;
        this.defense = defense;
        this.shopPrice = shopPrice;
    }

    public String getName() {
        return name;
    }

    public double getDamage() {
        return damage;
    }

    public double getDefense() {
        return defense;
    }

    public double getHealing() {
        return healing;
    }

    public double getSpeed() {
```

```
40          return speed;
41      }
42
43      public int getShopPrice() {
44          return shopPrice;
45      }
46  }
47
```

```java
package com.example.battlegame;

import java.util.ArrayList;

public class Attack {
    private String attackName;

    private double attackDamage;

    private int cooldown = 0;

    private int lastused = 0;

    public Attack(String attackName, int attackDamage){
        this.attackName = attackName;
        this.attackDamage = attackDamage;
    }

    public Attack(String attackName, int attackDamage, int cooldown){
        this.attackName = attackName;
        this.attackDamage = attackDamage;
        this.cooldown = cooldown;
    }

    public String getAttackName() {
        return attackName;
    }

    public double getAttackDamage() {
        return attackDamage;
    }

    public int getCooldown() {
        return cooldown;
    }

    public int getLastused() {
        return lastused;
    }
```

```
40
41      public void setLastused(int lastused) {
42          this.lastused = lastused;
43      }
44  }
45
```

```java
package com.example.battlegame;

public class Player {
    private String name;

    private Classes fighterClass;

    private double[] attributes = {50,50,100,50};// Strength, Speed, Health, Defense

    private int playerlevel = 1;

    private int xp = 0;

    private int coins = 100;

    private Inventory inventory = new Inventory();

    public Player(String name, Classes fighterclass){
        this.name = name;
        this.fighterClass = fighterclass;
        for (int i = 0; i < attributes.length; i++) {
            attributes[i] += fighterclass.getAttributeChanges()[i];
        }
    }
    public Player(Classes fighterclass, Player player){
        this.name = "Computer";
        this.fighterClass = fighterclass;
        for (int i = 0; i < attributes.length; i++) {
            attributes[i] += fighterclass.getAttributeChanges()[i];
        }
        this.playerlevel = player.getPlayerlevel();
    }

    public Player(Classes fighterclass, Player player, String bossname){
        this.name = bossname;
        this.fighterClass = fighterclass;
```

```
37          for (int i = 0; i < attributes.length; i++) {
38              attributes[i] += fighterclass.
   getAttributeChanges()[i];
39          }
40          this.playerlevel = player.getPlayerlevel();
41      }
42
43      public String getName() {
44          return name;
45      }
46
47      public Classes getFighterClass() {
48          return fighterClass;
49      }
50
51      public double[] getAttributes() {
52          return attributes;
53      }
54
55      public int getPlayerlevel() {
56          return playerlevel;
57      }
58
59      protected void changeAttributes(int index, double
    change){
60          attributes[index] += change;//Strength, Speed
   , Health, Defense
61      }
62      protected void setAttributes(int index, double
   amount){
63          attributes[index] = amount;
64      }
65
66      public int getCoins() {
67          return coins;
68      }
69
70      public int getXp() {
71          return xp;
72      }
73
```

```
74      public Inventory getInventory() {
75          return inventory;
76      }
77
78      public void changeCoins(int coins) {
79          this.coins += coins;
80      }
81
82      public void changePlayerlevel(int playerlevel) {
83          this.playerlevel += playerlevel;
84      }
85
86      public void changeXp(int xp) {
87          this.xp += xp;
88      }
89  }
90
```

```java
package com.example.battlegame;

import javafx.scene.image.Image;

import java.lang.reflect.Array;
import java.net.URL;
import java.util.ArrayList;

public class Classes {
    private String className;

    private double[] attributeChanges;//Strength, Speed, Health, Defense

    private ArrayList<Attack> attacks = new ArrayList<>();

    private Image image;

    public Classes(String className){
        this.className = className;
        if(className.equals("knight")){
            image = new Image(getClass().getResource("knightlogo.jpg").toString());

            this.attributeChanges = new double[]{20,-5,0,10};

            attacks.add(new Attack("Piercing Stab", 15));
            attacks.add(new Attack("Slice", 5));
            attacks.add(new Attack("Strength++", 0));
            attacks.add(new Attack("Lance Dash", 20));
            attacks.add(new Attack("Sweep & Slice", 10));
            attacks.add(new Attack("Flee", 0));

        } else if(className.equals("mage")){
            image = new Image(getClass().getResource("magelogo.jpeg").toString());
```

```
34
35              this.attributeChanges = new double[]{-10,
   15,70,0};
36
37              attacks.add(new Attack("Fire Rain", 30));
38              attacks.add(new Attack("Shadow Slice", 25
   ));
39              attacks.add(new Attack("Health++", 0));
40              attacks.add(new Attack("Sun Spear", 35));
41              attacks.add(new Attack("Water Whip", 10
   ));
42              attacks.add(new Attack("Flee", 0));
43
44          } else if(className.equals("archer")){
45              image = new Image(getClass().getResource(
   "archerlogo.jpeg").toString());
46
47              this.attributeChanges = new double[]{-10,
   25,0,10};
48
49              attacks.add(new Attack("Piercing Arrows"
   , 10));
50              attacks.add(new Attack("Exploding Arrows"
   , 25));
51              attacks.add(new Attack("Speed++", 0));
52              attacks.add(new Attack("Loudest Arrows",
   10));
53              attacks.add(new Attack("Multishot", 20));
54              attacks.add(new Attack("Flee", 0));
55
56          } else if(className.equals("bard")){
57              image = new Image(getClass().getResource(
   "bardlogo.png").toString());
58
59              this.attributeChanges = new double[]{20,-
   5,-40,10};
60
61              attacks.add(new Attack("Bagpipe Shriek",
   5));
62              attacks.add(new Attack("Flute Slice", 10
   ));
```

```
63              attacks.add(new Attack("Health Song", 0
   ));
64              attacks.add(new Attack("Xylophone
   Confusion", 5));
65              attacks.add(new Attack("Speed Song", 15
   ));
66              attacks.add(new Attack("Flee", 0));
67
68          } else if(className.equals("shooter")){
69              image = new Image(getClass().getResource
   ("shooterlogo.jpeg").toString());
70
71              this.attributeChanges = new double[]{-10
   ,-5,-30,10};
72
73              attacks.add(new Attack("Piercing Bullets
   ", 20));
74              attacks.add(new Attack("Bayonet Slice",
   10));
75              attacks.add(new Attack("Speed++", 0));
76              attacks.add(new Attack("Snipe", 50, 5));
77              attacks.add(new Attack("MultiShot", 30
   ));
78              attacks.add(new Attack("Flee", 0));
79
80          } else if(className.equals("boss1")){
81              image = new Image(getClass().getResource
   ("boss1.png").toString());
82
83              this.attributeChanges = new double[]{10,
   10,10,10};
84
85              attacks.add(new Attack("The Stomper", 40
   ));
86              attacks.add(new Attack("Crazy Slice", 20
   ));
87              attacks.add(new Attack("Jump Scare", 10
   ));
88              attacks.add(new Attack("Weaken", 0));
89              attacks.add(new Attack("Weaken", 0));
90
```

```
 91          } else if(className.equals("boss2")){
 92              image = new Image(getClass().getResource
    ("boss2.png").toString());
 93
 94              this.attributeChanges = new double[]{20,
    20,20,20};
 95
 96              attacks.add(new Attack("Fire Spin", 20
    ));
 97              attacks.add(new Attack("Incinerator", 40
    ));
 98              attacks.add(new Attack("Missile Attack"
    , 30));
 99              attacks.add(new Attack("Slow", 0));
100              attacks.add(new Attack("Slow", 0));
101
102          }
103      }
104
105      public ArrayList<Attack> getAttacks() {
106          return attacks;
107      }
108
109      public double[] getAttributeChanges() {
110          return attributeChanges;
111      }
112
113      public String getClassName() {
114          return className;
115      }
116
117      public Image getImage() {
118          return image;
119      }
120 }
121
```

```java
package com.example.battlegame;

import java.util.ArrayList;

public class Inventory {
    private ArrayList<OwnedItems> itemsOwned = new ArrayList<>();

    public void addItem(OwnedItems ownedItems) {
        itemsOwned.add(ownedItems);
    }

    public void setItem(OwnedItems newItem, OwnedItems oldItem) {
        itemsOwned.remove(oldItem);
        itemsOwned.add(newItem);
    }

    public void removeItem(OwnedItems ownedItems) {
        itemsOwned.remove(ownedItems);
    }

    public ArrayList<OwnedItems> getItemsOwned() {
        return itemsOwned;
    }
}

```

```java
package com.example.battlegame;

public class OwnedItems {
    private Items item;

    private int numItems;

    public OwnedItems(Items item){
        this.item = item;

        numItems = 1;
    }

    public Items getItem() {
        return item;
    }

    public int getNumItems() {
        return numItems;
    }

    public void changeNumItems(int numItems) {
        this.numItems += numItems;
    }
}

```

```java
package com.example.battlegame;

import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
import javafx.scene.control.*;
import javafx.scene.image.ImageView;

import java.net.URL;
import java.util.ArrayList;
import java.util.Random;
import java.util.ResourceBundle;

public class HelloController implements Initializable {

    public Button refreshShopButton;
    public Button startBattleButton;
    public Button createCharacterButton2;
    public Button compAttackButton;
    @FXML
    private ListView<String> classesListView;

    @FXML
    private Label classSetup, nameSetup, weaponSetup;
    @FXML
    private ImageView playerSetupPic;

    @FXML
    private TextField nameField;

    @FXML
    private SplitMenuButton weaponMenu;

    @FXML
    private Button createCharacterButton, hardModeButton;

    @FXML
    private ListView<String> playerStats,
```

```
39 inventoryView1, opponentsView, opponentsStats;
40
41     @FXML
42     private Label playerStatsLabel, inventoryLabel1,
   opponentsLabel, opponentStatsLabel, goToBattleLabel;
43
44     @FXML
45     private ImageView playerPic, compPic;
46
47     @FXML
48     private ListView<String> playerInventory,
   playerAttacks;
49
50     @FXML
51     private Label battleResultLabel, compHealthLabel
   , compLabel, playerHealthLabel, playerLabel,
   playerInventoryLabel, playerAttackLabel;
52
53     @FXML
54     private ProgressBar compHealthBar,
   playerHealthBar;
55
56     @FXML
57     private Label coinsLabel, inventoryLabel,
   shopLabel, statsLabel;
58
59     @FXML
60     private ListView<String> inventoryView, shopView
   , statsView;
61
62     private Classes knight = new Classes("knight");
63     private Classes mage = new Classes("mage");
64     private Classes archer = new Classes("archer");
65     private Classes bard = new Classes("bard");
66     private Classes shooter = new Classes("shooter");
67     private Classes boss1 = new Classes("boss1");
68     private Classes boss2 = new Classes("boss2");
69
70     private Items leatherArmor = new Items("
   leatherArmor", 0,0,-1,10,20);
71     private Items ironArmor = new Items("ironArmor",
```

```
71 0,0,-1,20,40);
72     private Items goldArmor = new Items("goldArmor"
   , 0,0,-1,30,60);
73     private Items diamondArmor = new Items("
   diamondArmor", 0,0,-1,40,80);
74     private Items legendaryArmor = new Items("
   legendaryArmor", 0,0,-1,50,100);
75
76     private Items basicHealthPotion = new Items("
   basicHealthPotion", 0,10,0,0,20);
77     private Items goodHealthPotion = new Items("
   goodHealthPotion", 0,50,0,0,100);
78     private Items basicSpeedPotion = new Items("
   basicSpeedPotion", 0,0,10,0,20);
79     private Items goodSpeedPotion = new Items("
   goodSpeedPotion", 0,0,50,0,100);
80
81     private Items ironSword = new Items("ironSword"
   , 20,0,0,0,40);
82     private Items goldSword = new Items("goldSword"
   , 30,0,0,0,60);
83     private Items diamondSword = new Items("
   diamondSword", 40,0,0,0,80);
84     private Items legendarySword = new Items("
   legendarySword", 50,0,0,0,100);
85
86     private Items ironBow = new Items("ironBow", 20,
   0,0,0,40);
87     private Items goldBow = new Items("goldBow", 30,
   0,0,0,60);
88     private Items diamondBow = new Items("diamondBow
   ", 40,0,0,0,80);
89     private Items legendaryBow = new Items("
   legendaryBow", 50,0,0,0,100);
90
91     private Items ironAxe = new Items("ironAxe", 20,
   0,0,0,40);
92     private Items goldAxe = new Items("goldAxe", 30,
   0,0,0,60);
93     private Items diamondAxe = new Items("diamondAxe
   ", 40,0,0,0,80);
```

```
 94     private Items legendaryAxe = new Items("
    legendaryAxe", 50,0,0,0,100);
 95
 96     private Items[] startingWeapons = {ironSword,
    ironBow, ironAxe};
 97
 98     private Items[] allItems = {leatherArmor,
    ironArmor, goldArmor, diamondArmor, legendaryArmor,
    basicHealthPotion, goodHealthPotion,
    basicSpeedPotion, goodSpeedPotion, ironSword,
    goldSword, diamondSword, legendarySword, ironBow,
    goldBow, diamondBow,legendaryBow, ironAxe, goldAxe,
    diamondAxe, legendaryAxe};
 99
100     private Items[] hardItems = {leatherArmor,
    ironArmor, goldArmor, diamondArmor, legendaryArmor,
    basicSpeedPotion, goodSpeedPotion, ironSword,
    goldSword, diamondSword, legendarySword, ironBow,
    goldBow, diamondBow,legendaryBow, ironAxe, goldAxe,
    diamondAxe, legendaryAxe};
101
102     private Shop shop;
103
104     private Player p1;
105     private ArrayList<Player> compPlayers = new
    ArrayList<>();
106
107     private Classes[] fighterclasses = {knight, mage
    , archer, bard, shooter};
108
109     private Player battleplayer1;
110     private Player battleplayer2;
111
112     private int turn = 0;
113
114     private Items selectedWeapon;
115
116     private OwnedItems selectedStartingItem;
117
118     private int extramoney = 0;
119
```

```
120     EventHandler<ActionEvent> event1 = new
    EventHandler<ActionEvent>() {
121         public void handle(ActionEvent e)
122         {
123             String selectedItemName = ((MenuItem)e.
    getSource()).getText();
124 
125             Items selectedItem = null;
126 
127             for (Items items:startingWeapons) {
128                 if(selectedItemName == items.getName
    ()){
129                     selectedItem = items;
130                 }
131             }
132 
133             selectedStartingItem = new OwnedItems(
    selectedItem);
134         }
135     };
136     private boolean hardmode;
137 
138 
139     @Override
140     public void initialize(URL url, ResourceBundle
    resourceBundle) {
141         classesListView.getItems().clear();
142         for (Classes fighterClass: fighterclasses) {
143             classesListView.getItems().add(
    fighterClass.getClassName());
144         }
145 
146         weaponMenu.getItems().clear();
147         for (Items items: startingWeapons) {
148             weaponMenu.getItems().add(new MenuItem(
    items.getName()));
149         }
150 
151         for (MenuItem items: weaponMenu.getItems
    ()) {
152             items.setOnAction(event1);
```

```
153         }
154
155         goToBattleLabel.setVisible(false);
156     }
157
158     @FXML
159     protected void createCharacter() {
160         String name = nameField.getText();
161         int index = classesListView.
    getSelectionModel().getSelectedIndex();
162         Classes fighterclass = fighterclasses[index
    ];
163         p1 = new Player(name, fighterclass);
164         p1.getInventory().addItem(
    selectedStartingItem);
165
166         updateCompPlayers();
167         updateCompPlayers();
168         updateCompPlayers();
169         displayOpponents();
170
171         printStats(playerStats, p1);
172         displayItems();
173
174         playerSetupPic.setImage(fighterclass.
    getImage());
175
176         setShop();
177
178         p1.changeCoins(extramoney);
179     }
180
181     private void setShop() {
182         if (hardmode){
183             shop = new Shop(hardItems);
184         } else {
185             shop = new Shop(allItems);
186         }
187     }
188
189     private void displayOpponents() {
```

```
190         opponentsView.getItems().clear();
191         for (int i = 1; i <= 3; i++) {
192             opponentsView.getItems().add(compPlayers
    .get(compPlayers.size()-i).getFighterClass().
    getClassName());
193         }
194     }
195 
196     @FXML
197     protected void updateCompPlayers(){
198         int bosschance = (int) (Math.random()*15);
199         if (bosschance == 0){
200             compPlayers.add(new Player(boss1, p1, "
    Stomp-Chomp"));
201         } else if (bosschance == 1) {
202             compPlayers.add(new Player(boss2, p1, "
    Fire Stalker"));
203         } else {
204             Random random = new Random();
205             int index = random.nextInt(
    fighterclasses.length);
206             compPlayers.add(new Player(
    fighterclasses[index], p1));
207         }
208     }
209 
210     @FXML
211     protected void printStats(ListView listview,
    Player player){
212         ArrayList<String> tempArray = new ArrayList
    <>();
213         tempArray.add("Name: " + player.getName());
214         tempArray.add("Strength: " + p1.
    getAttributes()[0]);
215         tempArray.add("Speed: " + p1.getAttributes
    ()[1]);
216         tempArray.add("Health: " + p1.getAttributes
    ()[2]);
217         tempArray.add("Defense: " + p1.getAttributes
    ()[3]);
218         tempArray.add("Class: " + player.
```

```
218 getFighterClass().getClassName());
219         tempArray.add("Attacks: ");
220         for (Attack attack: player.getFighterClass
    ().getAttacks()) {
221             tempArray.add(attack.getAttackName() +
    "," + attack.getAttackDamage() + "  ");
222         }
223         tempArray.add("Level: " + player.
    getPlayerlevel());
224         tempArray.add("Coins: " + player.getCoins
    ());
225         tempArray.add("XP: " + player.getXp());
226 
227         listview.getItems().clear();
228         listview.getItems().addAll(tempArray);
229         coinsLabel.setText("Coins: " + p1.getCoins
    ());
230     }
231 
232     @FXML
233     protected void startBattle(){
234         turn = 0;
235         battleplayer1 = p1;
236         int index = opponentsView.getSelectionModel
    ().getSelectedIndex();
237         battleplayer2 = compPlayers.get(compPlayers.
    size()-index-1);
238         updateCompPlayers();
239         displayOpponents();
240         compPlayers.remove(index);
241         goToBattleLabel.setVisible(true);
242 
243         ArrayList<String> tempList = new ArrayList
    <>();
244 
245         for (Attack attack: battleplayer1.
    getFighterClass().getAttacks()) {
246             tempList.add(attack.getAttackName() +
    "; Damage: " + attack.getAttackDamage());
247         }
248 
```

```
249         playerAttacks.getItems().clear();
250         playerAttacks.getItems().addAll(tempList);
251         setbattleVisibility(true);
252         compAttackButton.setVisible(true);
253         compHealthLabel.setText("COMP Health: " +
    Math.round(battleplayer2.getAttributes()[2]*100)/100
    );
254         playerHealthLabel.setText("Your Health: " +
    Math.round(battleplayer1.getAttributes()[2]*100)/100
    );
255
256         playerPic.setImage(battleplayer1.
    getFighterClass().getImage());
257         compPic.setImage(battleplayer2.
    getFighterClass().getImage());
258
259         battleResultLabel.setText("The Battle Has
    Started");
260     }
261
262     protected void attack(Player attacker, Player
    attacked, int attackindex){
263         if (attacker.getFighterClass().getAttacks().
    get(attackindex).getAttackName().equals("Flee")){
264             setbattleVisibility(false);
265             battleResultLabel.setText("You fled the
    battle, go fight someone else");
266         }
267         else if (attacker.getFighterClass().
    getAttacks().get(attackindex).getAttackName().equals
    ("Strength++")){
268             attacker.changeAttributes(0,5);
269             battleResultLabel.setText(attacker.
    getName() + " is stronger now");
270         }
271         else if (attacker.getFighterClass().
    getAttacks().get(attackindex).getAttackName().equals
    ("Speed++")){
272             attacker.changeAttributes(1,5);
273             battleResultLabel.setText(attacker.
    getName() + " is faster now");
```

```
274         }
275         else if (attacker.getFighterClass().
    getAttacks().get(attackindex).getAttackName().equals
    ("Health Song")){
276             if (hardmode){
277                 battleResultLabel.setText(attacker.
    getName() + " can't heal, its hardmode");
278             } else {
279                 attacker.changeAttributes(2, 10);
280                 battleResultLabel.setText(attacker.
    getName() + " healed");
281             }
282         }
283         else if (attacker.getFighterClass().
    getAttacks().get(attackindex).getAttackName().equals
    ("Health++")) {
284             if (hardmode){
285                 battleResultLabel.setText(attacker.
    getName() + " can't heal, its hardmode");
286             } else {
287                 attacker.changeAttributes(2, 5);
288                 battleResultLabel.setText(attacker.
    getName() + " healed");
289             }
290         }
291         else if (attacker.getFighterClass().
    getAttacks().get(attackindex).getAttackName().equals
    ("Weaken")){
292             attacker.changeAttributes(0,-5);
293             battleResultLabel.setText(attacker.
    getName() + " weakened " + attacked.getName());
294         }
295         else if (attacker.getFighterClass().
    getAttacks().get(attackindex).getAttackName().equals
    ("Slow")){
296             attacked.changeAttributes(1,-5);
297             battleResultLabel.setText(attacker.
    getName() + " slowed " + attacked.getName());
298         }
299         else {
300             int damagechance = (int) (Math.random
```

```
300 () * 25);
301             double damagemultiplier = 1;
302 
303             if (damagechance == 0) {
304                 damagemultiplier = 0;
305             }
306             if (damagechance == 1) {
307                 damagemultiplier = 1.5;
308             }
309 
310             int turnsSinceUsed = turn - attacker.
    getFighterClass().getAttacks().get(attackindex).
    getLastused();
311 
312             if (turnsSinceUsed < attacker.
    getFighterClass().getAttacks().get(attackindex).
    getCooldown()){
313                 battleResultLabel.setText(attacker.
    getName() + " missed because their cooldown wasn't
    over");
314             }
315 
316             double damage = attacker.getFighterClass
    ().getAttacks().get(attackindex).getAttackDamage
    () * damagemultiplier * attacker.getPlayerlevel();
317             double damageDealt = attacked.
    getAttributes()[3]/15 - damage * attacker.
    getAttributes()[0] / 50 * attacker.getAttributes()[1
    ] / 50 + selectedWeapon.getDamage() / 10;
318             damageDealt = Math.round(damageDealt*100
    )/100;
319             attacked.changeAttributes(2, (
    damageDealt));
320 
321             if (damagemultiplier == 0){
322                 battleResultLabel.setText(attacker.
    getName() + " missed");
323             } else if (damagemultiplier == 1.5) {
324                 battleResultLabel.setText(attacker.
    getName() + " did a critical hit - " + damageDealt
     + " damage");
```

```
325             } else {
326                 battleResultLabel.setText(attacker.
    getName() + " did " + damageDealt + " damage");
327             }
328 
329             if (attacked.getAttributes()[2] < 0){
330                 attacked.setAttributes(2,0);
331             }
332         }
333 
334         if (battleplayer1.getAttributes()[2] > 200){
335             battleplayer1.setAttributes(2,200);
336         }
337         if (battleplayer2.getAttributes()[2] > 200){
338             battleplayer2.setAttributes(2,200);
339         }
340 
341         compHealthLabel.setText("COMP Health: " +
    battleplayer2.getAttributes()[2]);
342         playerHealthLabel.setText("Your Health: " +
    battleplayer1.getAttributes()[2]);
343 
344         playerHealthBar.setProgress(battleplayer1.
    getAttributes()[2]/200);
345         compHealthBar.setProgress(battleplayer2.
    getAttributes()[2]/200);
346 
347         if (checkDead()){
348             setbattleVisibility(false);
349             compAttackButton.setVisible(false);
350             if (battleplayer2.getAttributes()[2]<=0
    ){
351                 if (battleplayer2.getFighterClass().
    getClassName() == "boss1"){
352                     p1.changeCoins(40*turn);
353                     p1.changeXp(25*turn);
354                     battleResultLabel.setText("You
    won the boss battle!  GOOD JOB! You earned " + (40
     * turn) + " xp & " + (25 * turn) + "coins");
355                 } else if (battleplayer2.
    getFighterClass().getClassName() == "boss2"){
```

```
356                     p1.changeCoins(50*turn);
357                     p1.changeXp(30*turn);
358                     battleResultLabel.setText("You
    won the boss battle!  GOOD JOB! You earned " + (50
     * turn) + " xp & " + (30 * turn) + "coins");
359                 } else {
360                     p1.changeCoins(15*turn);
361                     p1.changeXp(25*turn);
362                     battleResultLabel.setText("You
    won the battle! You earned " + (25 * turn) + " xp
     & " + (15 * turn) + "coins");
363                 }
364                 while (p1.getXp() >= 300) {
365                     p1.changePlayerlevel(1);
366                     p1.changeXp(-300);
367                 }
368             }
369 
370             if (battleplayer1.getAttributes()[2]<=0
    ){
371                 battleResultLabel.setText("You died
    ! Go back to the Setup Tab!");
372                 extramoney = 200;
373             }
374         }
375         printStats(playerStats, p1);
376         coinsLabel.setText("Coins: " + p1.getCoins
    ());
377     }
378 
379     private void setbattleVisibility(boolean
    visibility) {
380         compHealthLabel.setVisible(visibility);
381         compHealthBar.setVisible(visibility);
382         compPic.setVisible(visibility);
383         compLabel.setVisible(visibility);
384         playerHealthLabel.setVisible(visibility);
385         playerHealthBar.setVisible(visibility);
386         playerPic.setVisible(visibility);
387         playerLabel.setVisible(visibility);
388         playerInventoryLabel.setVisible(visibility);
```

```
389         playerInventory.setVisible(visibility);
390         playerAttackLabel.setVisible(visibility);
391         playerAttacks.setVisible(visibility);
392     }
393 
394     private boolean checkDead() {
395         if (battleplayer1.getAttributes()[2]<=0){
396             return true;
397         } else if (battleplayer2.getAttributes()[2
    ]<=0){
398             return true;
399         }
400         return false;
401     }
402 
403     @FXML
404     public void showCompStats() {
405         int index = opponentsView.getSelectionModel
    ().getSelectedIndex();
406         Player selectedPlayer = compPlayers.get(
    compPlayers.size()-index-1);
407         printStats(opponentsStats, selectedPlayer);
408     }
409 
410     public void useItem() {
411         int index = inventoryView.getSelectionModel
    ().getSelectedIndex();
412 
413         if (p1.getInventory().getItemsOwned().get(
    index).getItem().getDamage() > 0){
414             selectedWeapon = p1.getInventory().
    getItemsOwned().get(index).getItem();
415         } else {
416             p1.changeAttributes(1, p1.getInventory
    ().getItemsOwned().get(index).getItem().getSpeed());
417             p1.changeAttributes(2, p1.getInventory
    ().getItemsOwned().get(index).getItem().getHealing
    ());
418             p1.changeAttributes(3, p1.getInventory
    ().getItemsOwned().get(index).getItem().getDefense
    ());
```

```
419             p1.getInventory().getItemsOwned().get(
    index).changeNumItems(-1);
420             if (p1.getInventory().getItemsOwned().
    get(index).getNumItems() == 0) {
421                 p1.getInventory().getItemsOwned().
    remove(index);
422             }
423         }
424 
425         displayItems();
426     }
427 
428     public void useItem1() {
429         int index = inventoryView1.getSelectionModel
    ().getSelectedIndex();
430 
431         if (p1.getInventory().getItemsOwned().get(
    index).getItem().getDamage() > 0){
432             selectedWeapon = p1.getInventory().
    getItemsOwned().get(index).getItem();
433         } else {
434             p1.changeAttributes(1, p1.getInventory
    ().getItemsOwned().get(index).getItem().getSpeed());
435             p1.changeAttributes(2, p1.getInventory
    ().getItemsOwned().get(index).getItem().getHealing
    ());
436             p1.changeAttributes(3, p1.getInventory
    ().getItemsOwned().get(index).getItem().getDefense
    ());
437             p1.getInventory().getItemsOwned().get(
    index).changeNumItems(-1);
438             if (p1.getInventory().getItemsOwned().
    get(index).getNumItems() == 0) {
439                 p1.getInventory().getItemsOwned().
    remove(index);
440             }
441         }
442 
443         displayItems();
444     }
445 
```

```
446     public void useItem2() {
447         int index = playerInventory.
    getSelectionModel().getSelectedIndex();
448 
449         if (p1.getInventory().getItemsOwned().get(
    index).getItem().getDamage() > 0){
450             selectedWeapon = p1.getInventory().
    getItemsOwned().get(index).getItem();
451         } else {
452             p1.changeAttributes(1, p1.getInventory
    ().getItemsOwned().get(index).getItem().getSpeed());
453             p1.changeAttributes(2, p1.getInventory
    ().getItemsOwned().get(index).getItem().getHealing
    ());
454             p1.changeAttributes(3, p1.getInventory
    ().getItemsOwned().get(index).getItem().getDefense
    ());
455             p1.getInventory().getItemsOwned().get(
    index).changeNumItems(-1);
456             if (p1.getInventory().getItemsOwned().
    get(index).getNumItems() == 0) {
457                 p1.getInventory().getItemsOwned().
    remove(index);
458             }
459         }
460 
461         displayItems();
462     }
463 
464     public void displayItems() {
465         inventoryView.getItems().clear();
466         for (OwnedItems item: p1.getInventory().
    getItemsOwned()) {
467             inventoryView.getItems().add(item.
    getItem().getName() + "; Number: " + item.
    getNumItems());
468         }
469 
470         inventoryView1.getItems().clear();
471         for (OwnedItems item: p1.getInventory().
    getItemsOwned()) {
```

```
472             inventoryView1.getItems().add(item.
    getItem().getName() + "; Number: " + item.
    getNumItems());
473         }
474 
475         playerInventory.getItems().clear();
476         for (OwnedItems item: p1.getInventory().
    getItemsOwned()) {
477             playerInventory.getItems().add(item.
    getItem().getName() + "; Number: " + item.
    getNumItems());
478         }
479 
480         statsView.getItems().clear();
481         statsView.getItems().add("Strength: " + p1.
    getAttributes()[0]);
482         statsView.getItems().add("Speed: " + p1.
    getAttributes()[1]);
483         statsView.getItems().add("Health: " + p1.
    getAttributes()[2]);
484         statsView.getItems().add("Defense: " + p1.
    getAttributes()[3]);
485     }
486 
487     public void buyItem() {
488         int index = shopView.getSelectionModel().
    getSelectedIndex();
489         int price = shop.getCurrentList().get(index
    ).getShopPrice();
490 
491         if (p1.getCoins()>=price){
492             Items tempItem = shop.getCurrentList().
    get(index);
493 
494             boolean owned = false;
495 
496             for (OwnedItems items: p1.getInventory
    ().getItemsOwned()) {
497                 if (items.getItem() == tempItem){
498                     owned = true;
499                 }
```

```
500             }
501
502             if (owned){
503                 for (OwnedItems items: p1.
    getInventory().getItemsOwned()) {
504                     if (items.getItem() == tempItem
    ){
505                         items.changeNumItems(1);
506                     }
507                 }
508             } else {
509                 p1.getInventory().addItem(new
    OwnedItems(tempItem));
510             }
511             p1.changeCoins(price*-1);
512         } else {
513             System.out.println("Not enough Money");
514         }
515
516         displayItems();
517         coinsLabel.setText("Coins: " + p1.getCoins
    ());
518     }
519
520     public void attack() {
521         if (!checkDead()) {
522             attack(battleplayer1, battleplayer2,
    playerAttacks.getSelectionModel().getSelectedIndex
    ());
523         }
524         turn++;
525         setbattleVisibility(false);
526     }
527     public void compAttack() {
528         int attackindex = (int) (Math.random() * 5);
529         System.out.println(attackindex);
530         if (!checkDead()) {
531             attack(battleplayer2, battleplayer1,
    attackindex);
532         }
533         setbattleVisibility(true);
```

```
534     }
535
536
537     public void refreshShop() {
538         shop.randomizeShop(shopView);
539         printStats(playerStats, p1);
540         coinsLabel.setText("Coins: " + p1.getCoins
    ());
541     }
542
543     public void setHardMode() {
544         this.hardmode = true;
545     }
546 }
```

```java
package com.example.battlegame;

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(), 700, 500);
        stage.setTitle("Hello!");
        stage.setScene(scene);
        stage.show();
    }

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