File - GdxGame.java

```java
package view;

import com.badlogic.gdx.Game;
import com.badlogic.gdx.graphics.OrthographicCamera;
import com.badlogic.gdx.graphics.g2d.SpriteBatch;
import com.badlogic.gdx.utils.viewport.ExtendViewport;
import model.DataManager;
import view.games.SpellingGameScreen;

public class GdxGame extends Game {

    public static final int WIDTH = 1920;
    public static final int HEIGHT = 1080;

    public OrthographicCamera camera;
    public ExtendViewport viewport;
    public SpriteBatch batch;
//    public BitmapFont font;

    public void create() {
        camera = new OrthographicCamera();
        camera.setToOrtho(false, WIDTH, HEIGHT);
        viewport = new ExtendViewport(WIDTH, HEIGHT, camera);

        batch = new SpriteBatch();
//        font = new BitmapFont();
        AssetManager.init();
        DataManager.populate();

        setScreen(new SpellingGameScreen(this, new view.start.StartScreen(this)));
    }

    public void render() {
        super.render(); //important!
    }

    public void dispose() {
        batch.dispose();
//        font.dispose();
    }
}

```

File - AssetManager.java

```
 1 package view;
 2
 3 import com.badlogic.gdx.Gdx;
 4 import com.badlogic.gdx.graphics.g2d.Animation;
 5 import com.badlogic.gdx.graphics.g2d.TextureAtlas;
 6 import com.badlogic.gdx.graphics.g2d.TextureRegion;
 7 import com.badlogic.gdx.scenes.scene2d.ui.ImageButton.
   ImageButtonStyle;
 8 import com.badlogic.gdx.scenes.scene2d.ui.Skin;
 9
10 public class AssetManager {
11     public static TextureAtlas atlas;
12     public static Skin buttonSkin;
13     public static ImageButtonStyle imageButtonStyle;
14
15     public static void init() {
16         atlas = new TextureAtlas(Gdx.files.internal("images/pack.
   atlas"));
17         buttonSkin = new Skin(Gdx.files.internal("skins/skin.json"))
   ;
18         imageButtonStyle = new ImageButtonStyle();
19         imageButtonStyle.up = AssetManager.buttonSkin.getDrawable("
   buttonUp");
20         imageButtonStyle.down = AssetManager.buttonSkin.getDrawable(
   "buttonDown");
21         imageButtonStyle.over = AssetManager.buttonSkin.getDrawable(
   "buttonOver");
22     }
23
24     public static TextureRegion getTextureRegion(String fileName) {
25         return new TextureRegion(AssetManager.atlas.findRegion(
   fileName));
26     }
27
28     public static Animation<TextureRegion> getAnimation(String
   fileName) {
29         return new Animation<TextureRegion>(0.33f, AssetManager.
   atlas.findRegions(fileName));
30     }
31 }
32
```

File - SpellingGameScreen.java

```java
package view.games;

import com.badlogic.gdx.Gdx;
import com.badlogic.gdx.Screen;
import com.badlogic.gdx.graphics.GL30;
import com.badlogic.gdx.scenes.scene2d.*;
import com.badlogic.gdx.scenes.scene2d.ui.Container;
import com.badlogic.gdx.scenes.scene2d.ui.Image;
import com.badlogic.gdx.scenes.scene2d.ui.ImageButton;
import com.badlogic.gdx.scenes.scene2d.ui.Table;
import com.badlogic.gdx.scenes.scene2d.utils.ChangeListener;
import com.badlogic.gdx.scenes.scene2d.utils.DragAndDrop;
import com.badlogic.gdx.scenes.scene2d.utils.TextureRegionDrawable;
import viewmodel.SpellingGameStateMachine;
import view.GdxGame;
import view.actors.Letter;
import view.start.StartScreen;

import java.util.ArrayList;

public class SpellingGameScreen implements Screen {
    private final int separatorHeight = 40;
    private final int letterSize = 110;
    private final int imageSize = 400;
    private final int spaceSize = 140;
    private Stage stage;
    private GdxGame game;
    private Screen parentScreen;
    private DragAndDrop dragAndDrop;
    // Actors added to the screen are drawn in the order they were added. Actors drawn later are drawn on top of everything before.
    // These groups are used to add actors to the screen in the right order. All actors added to groups are drawn when the group is drawn.
    // These are added in this order in setStage. All actors are added to these groups and not the screen directly.
    private Group backgroundGroup;
    private Group actorsGroup;
    private Group animationsGroup;

    private ImageButton backButton;
    private Table letterTable;
    private Table pictureTable;
    private Table spaceTable;
    private Container<Image> pictureContainer;
    private ArrayList<Container<Letter>> letterContainers;
    private SpellingGameStateMachine spellingGameStateMachine;
```

```
44
45      public SpellingGameScreen(final GdxGame gdxGame, Screen
    parentScreen) {
46          this.game = gdxGame;
47          stage = new Stage(gdxGame.viewport, gdxGame.batch);
48          Gdx.input.setInputProcessor(stage);
49          dragAndDrop = new DragAndDrop();
50          dragAndDrop.setDragTime(0);
51
52          this.parentScreen = parentScreen;
53
54
55          setStage();
56      }
57
58      private void setStage() {
59          stage.addActor(backgroundGroup = new Group());
60          stage.addActor(actorsGroup = new Group());
61          stage.addActor(animationsGroup = new Group());
62
63          Image backgroundImage = new Image(view.AssetManager.
    getTextureRegion("background"));
64          backgroundImage.setSize(GdxGame.WIDTH, GdxGame.HEIGHT);
65          backgroundGroup.addActor(backgroundImage);
66
67          Table mainTable = new Table();
68  //          mainTable.setDebug(true);
69          mainTable.setBounds(0, 0, GdxGame.WIDTH, GdxGame.HEIGHT);
70          actorsGroup.addActor(mainTable);
71
72          backButton = new ImageButton(view.AssetManager.
    imageButtonStyle);
73          backButton.addListener(new ChangeListener() {
74              @Override
75              public void changed(ChangeEvent event, Actor actor) {
76                  SpellingGameScreen.this.game.setScreen(new
    StartScreen(game));
77              }
78          });
79          letterTable = new Table();
80          pictureTable = new Table();
81          spaceTable = new Table();
82
83  //          mainTable.add(backButton).size(80).left();
84          mainTable.row();
85          mainTable.add().height(separatorHeight);
86          mainTable.row();
```

```
 87          mainTable.add(letterTable).height(letterSize * 2);
 88          mainTable.row();
 89          mainTable.add().height(separatorHeight);
 90          mainTable.row();
 91          mainTable.add(pictureTable).height(imageSize);
 92          mainTable.row();
 93          mainTable.add().height(separatorHeight);
 94          mainTable.row();
 95          mainTable.add(spaceTable).height(spaceSize);
 96          mainTable.row();
 97          mainTable.add().height(GdxGame.HEIGHT - ((separatorHeight *
     3) + (letterSize * 2) + imageSize + spaceSize));
 98
 99          pictureTable.add(pictureContainer = new Container<Image>().
    size(imageSize));
100          letterContainers = new ArrayList<Container<Letter>>();
101
102          spellingGameStateMachine = new SpellingGameStateMachine(
    this);
103      }
104
105      public void setDisplayLanguage(SpellingGameStateMachine.
    Language language) {
106          setAlphabet(language);
107      }
108
109      private void setAlphabet(SpellingGameStateMachine.Language
    language) {
110          letterTable.clearChildren();
111          switch (language) {
112              case ENGLISH:
113                  String[] alphabet = {
114                          "a", "b", "c", "d", "e", "f", "g", "h", "i"
    , "j", "k", "l", "m",
115                          "n", "o", "p", "q", "r", "s", "t", "u", "v"
    , "w", "x", "y", "z"};
116                  int letterIndex = 0;
117                  for (int i = 0; i < 2; i++) {
118                      for (int j = 0; j < 13; j++) {
119                          if (letterIndex < alphabet.length) {
120                              final Letter letter = new Letter(
    alphabet[letterIndex++]);
121                              letterTable.add(new Container<Letter>(
    letter).size(letterSize)).size(letterSize);
122
123                              // Creates a copy when letter is
    dragged from the alphabet. The copy does not create a copy when
```

```
123 moved.
124                                 dragAndDrop.addSource(new DragAndDrop.
    Source(letter) {
125                                     public DragAndDrop.Payload
    dragStart(InputEvent event, float x, float y, int pointer) {
126                                         DragAndDrop.Payload payload =
    new DragAndDrop.Payload();
127                                         Letter letterCopy = new Letter(
    getActor());
128                                         letterCopy.setSize(spaceSize,
    spaceSize);
129                                         payload.setObject(letterCopy);
130                                         payload.setDragActor(letterCopy
    );
131                                         dragAndDrop.addSource(new
    DragAndDrop.Source(letterCopy) {
132                                             public DragAndDrop.Payload
    dragStart(InputEvent event, float x, float y, int pointer) {
133                                                 DragAndDrop.Payload
    payload = new DragAndDrop.Payload();
134                                                 payload.setObject(
    getActor());
135                                                 payload.setDragActor(
    getActor());
136                                                 return payload;
137                                             }
138                                         });
139                                         return payload;
140                                     }
141                                 });
142                             }
143                         }
144                         letterTable.row();
145                     }
146                     return;
147                 case HMONG:
148
149                     return;
150             }
151         }
152
153         public void setPictureAndWordLength(String pictureId, int
    wordLength) {
154             pictureContainer.setActor(new Image(view.AssetManager.
    getTextureRegion(pictureId)));
155             setSpaces(wordLength);
156         }
```

```
157
158     private void setSpaces(int spaces) {
159         spaceTable.clearChildren();
160         letterContainers.clear();
161         for (int i = 0; i < spaces; i++) {
162             Container<Letter> letterContainer = new Container<
    Letter>();
163             letterContainer.setTouchable(Touchable.enabled);
164             letterContainer.setBackground(new TextureRegionDrawable
    (view.AssetManager.getTextureRegion("underline")));
165             spaceTable.add(letterContainer.size(spaceSize,
    spaceSize)).size(spaceSize + 10, spaceSize + 50);
166             letterContainers.add(letterContainer);
167
168             dragAndDrop.addTarget(new DragAndDrop.Target(
    letterContainer) {
169                 public boolean drag(DragAndDrop.Source source,
    DragAndDrop.Payload payload, float x, float y, int pointer) {
170                     return true;
171                 }
172
173                 public void drop(DragAndDrop.Source source,
    DragAndDrop.Payload payload, float x, float y, int pointer) {
174                     Container<Letter> container = (Container)
    getActor();
175                     container.setActor((Letter) payload.getObject()
    );
176                     SpellingGameScreen.this.
    spellingGameStateMachine.doEvent(SpellingGameStateMachine.Event.
    DROPPED_LETTER);
177                 }
178             });
179         }
180     }
181
182     public ArrayList<Container<Letter>> getLetterContainers() {
183         return letterContainers;
184     }
185
186     @Override
187     public void render(float delta) {
188         Gdx.gl.glClearColor(0.3f, 0, 0, 1);
189         Gdx.gl.glClear(GL30.GL_COLOR_BUFFER_BIT);
190 //          stage.getBatch().setColor(1,1,1,1);
191         stage.act(delta);
192         stage.draw();
193     }
```

```
194
195     @Override
196     public void resize(int width, int height) {
197         stage.getViewport().update(width, height, false);
198     }
199
200     @Override
201     public void show() {
202     }
203
204     @Override
205     public void hide() {
206     }
207
208     @Override
209     public void pause() {
210     }
211
212     @Override
213     public void resume() {
214     }
215
216     @Override
217     public void dispose() {
218         stage.dispose();
219     }
220
221
222 }
223
```

File - StartScreen.java

```
 1 package view.start;
 2
 3 import com.badlogic.gdx.Gdx;
 4 import com.badlogic.gdx.Screen;
 5 import com.badlogic.gdx.graphics.GL30;
 6 import com.badlogic.gdx.scenes.scene2d.Actor;
 7 import com.badlogic.gdx.scenes.scene2d.Stage;
 8 import com.badlogic.gdx.scenes.scene2d.ui.Image;
 9 import com.badlogic.gdx.scenes.scene2d.ui.ImageButton;
10 import com.badlogic.gdx.scenes.scene2d.ui.Table;
11 import com.badlogic.gdx.scenes.scene2d.utils.ChangeListener;
12 import view.GdxGame;
13 import view.games.SpellingGameScreen;
14
15 public class StartScreen implements Screen {
16
17     private Stage stage;
18     private GdxGame game;
19
20     public StartScreen(GdxGame gdxGame) {
21         this.game = gdxGame;
22         stage = new Stage(gdxGame.viewport, gdxGame.batch);
23         Gdx.input.setInputProcessor(stage);
24
25         setStage();
26     }
27
28     private void setStage() {
29         Table mainTable = new Table();
30 //          mainTable.setDebug(true);
31         mainTable.setBounds(0, 0, GdxGame.WIDTH, GdxGame.HEIGHT);
32         stage.addActor(mainTable);
33
34         Image titleImage = new Image(view.AssetManager.
   getTextureRegion("gem"));
35         ImageButton studentButton = new ImageButton(view.
   AssetManager.buttonSkin);
36         ImageButton teacherButton = new ImageButton(view.
   AssetManager.buttonSkin);
37         studentButton.addListener(new ChangeListener() {
38             @Override
39             public void changed(ChangeEvent event, Actor actor) {
40                 StartScreen.this.game.setScreen(new
   SpellingGameScreen(game, StartScreen.this));
41             }
42         });
43         teacherButton.addListener(new ChangeListener() {
```

File - StartScreen.java

```
44              @Override
45              public void changed(ChangeEvent event, Actor actor) {
46
47              }
48          });
49
50          mainTable.row().height(300);
51          mainTable.add(titleImage).size(80);
52          mainTable.row().height(300);
53          mainTable.add(studentButton).size(80);
54          mainTable.add(teacherButton).size(80);
55  //          mainTable.row().height();
56      }
57
58      public void changeScreen(Screen screen) {
59          game.setScreen(screen);
60          this.dispose();
61      }
62
63      @Override
64      public void render(float delta) {
65          Gdx.gl.glClearColor(0.3f, 0, 0, 1);
66          Gdx.gl.glClear(GL30.GL_COLOR_BUFFER_BIT);
67  //          stage.getBatch().setColor(1,1,1,1);
68          stage.act(delta);
69          stage.draw();
70      }
71
72      @Override
73      public void resize(int width, int height) {
74          stage.getViewport().update(width, height, false);
75      }
76
77      @Override
78      public void show() {
79      }
80
81      @Override
82      public void hide() {
83      }
84
85      @Override
86      public void pause() {
87      }
88
89      @Override
90      public void resume() {
```

File - StartScreen.java

```
91      }
92
93      @Override
94      public void dispose() {
95          stage.dispose();
96      }
97  }
98
```

File - Letter.java

```java
package view.actors;

import com.badlogic.gdx.scenes.scene2d.Actor;
import com.badlogic.gdx.scenes.scene2d.Touchable;
import com.badlogic.gdx.scenes.scene2d.ui.ImageButton;
import com.badlogic.gdx.scenes.scene2d.ui.Label;
import com.badlogic.gdx.scenes.scene2d.ui.Stack;
import view.AssetManager;

public class Letter extends Stack {
    private ImageButton imageButton;
    private Label nameLabel;

    public Letter(String name) {
        setName(name);
        imageButton = new ImageButton(AssetManager.imageButtonStyle);
        add(imageButton);
        nameLabel = new Label(name, AssetManager.buttonSkin);
        nameLabel.setTouchable(Touchable.disabled);
        nameLabel.setAlignment(1);
        nameLabel.setFontScale(3);
        addActor(nameLabel);
    }

    public Letter(Actor letter) {
        this(letter.getName());
    }
}

```

File - AnimatedActor.java

```java
package view.actors;

import com.badlogic.gdx.graphics.Color;
import com.badlogic.gdx.graphics.g2d.Animation;
import com.badlogic.gdx.graphics.g2d.Batch;
import com.badlogic.gdx.graphics.g2d.TextureRegion;
import com.badlogic.gdx.scenes.scene2d.Actor;

public class AnimatedActor extends Actor {
    private Animation<TextureRegion> animation;
    private float time = 0f;
    private boolean flip;

    public AnimatedActor(Animation<TextureRegion> animation, String name) {
//        setDebug(true);
        this.setName(name);
        this.animation = animation;
        setSize(animation.getKeyFrame(0).getRegionWidth(), animation.getKeyFrame(0).getRegionHeight());
        flip = false;
    }

    public void changeAnimation(Animation<TextureRegion> animation, String name) {
        this.setName(name);
        this.animation = animation;
        setSize(animation.getKeyFrame(0).getRegionWidth(), animation.getKeyFrame(0).getRegionHeight());
        time = 0f;
    }

    public boolean isFlip() {
        return flip;
    }

    public void setFlip(boolean flip) {
        this.flip = flip;
    }

    @Override
    public void act(float delta) {
        super.act(delta);
        time += delta;
    }

    @Override
```

File - AnimatedActor.java

```
44      public void draw(Batch batch, float parentAlpha) {
45          super.draw(batch, parentAlpha);
46          Color color = getColor();
47          batch.setColor(color.r, color.g, color.b, color.a *
    parentAlpha);
48          TextureRegion currentFrame = animation.getKeyFrame(time);
49          batch.draw(currentFrame, (flip ? getWidth() : 0) + getX(),
    getY(), getOriginX(), getOriginY(),
50                  getWidth(), getHeight(), (flip ? -1 : 1) * getScaleX
    (), getScaleY(), getRotation());
51  //          drawChildren(batch, parentAlpha);
52      }
53  }
54
```

File - StudentScreen.java

```java
package view.student;

public class StudentScreen {
}

```

File - TeacherScreen.java

```java
package view.teacher;

public class TeacherScreen {
}

```

File - Word.java

```java
package model;

import viewmodel.SpellingGameStateMachine;

public class Word {
    private String wordId;
    private String englishSpelling;
    private String hmongSpelling;

    public Word(String wordId, String englishSpelling, String hmongSpelling) {
        this.wordId = wordId;
        this.englishSpelling = englishSpelling;
        this.hmongSpelling = hmongSpelling;
    }

    public String getWordId() {
        return wordId;
    }

    public String getSpelling(SpellingGameStateMachine.Language language) {
        switch (language) {
            case ENGLISH:
                return englishSpelling;
            case HMONG:
                return hmongSpelling;
            default:
                return null;
        }
    }
}

```

File - History.java

```java
package model;

import java.util.ArrayList;

public class History {
    private String gamePlayed;
    private ArrayList<String> wordsSpelled;

    public History(String gamePlayed) {
        this.gamePlayed = gamePlayed;
        wordsSpelled = new ArrayList<String>();
    }

    public void addWord(String wordId) {
        wordsSpelled.add(wordId);
    }
}

```

File - Student.java

```java
package model;

import java.util.ArrayList;

public class Student {
    private String name;
    private ArrayList<History> gameHistories;
    private History currentHistory;

    public Student(String name) {
        this.name = name;
        gameHistories = new ArrayList<History>();
    }

    public void startNewCurrentHistory(History history) {
        currentHistory = history;
        gameHistories.add(history);
    }

    public void addToCurrentHistory(String word) {
        currentHistory.addWord(word);
    }

    public String getName() {
        return name;
    }

    public ArrayList<History> getGameHistory() {
        return gameHistories;
    }
}

```

File - Teacher.java

```java
package model;

import java.util.ArrayList;

public class Teacher {
    private String name;
    private ArrayList<Student> students;

    public Teacher(String name) {
        this.name = name;
        students = new ArrayList<Student>();
    }

    public void addStudent(Student student) {
        students.add(student);
    }

    public String getName() {
        return name;
    }

    public ArrayList<Student> getStudents() {
        return students;
    }
}

```

File - DataManager.java

```
 1 package model;
 2
 3 import java.util.ArrayList;
 4 import java.util.Arrays;
 5 import java.util.HashMap;
 6
 7 public class DataManager {
 8     private static ArrayList<String> wordIdList;
 9     private static HashMap<String, Word> wordIdToWord;
10
11     public static void populate() {
12         wordIdList = new ArrayList<String>();
13         wordIdList.addAll(Arrays.asList("apple", "bird", "cherry", "
   gem", "money", "pear"));
14
15         wordIdToWord = new HashMap<String, Word>();
16         for (String wordId : wordIdList) {
17             wordIdToWord.put(wordId, new Word(wordId, wordId, wordId
   ));
18         }
19     }
20
21     public static ArrayList<String> getWordIdList() {
22         return wordIdList;
23     }
24
25     public static Word getWord(String wordId) {
26         return wordIdToWord.get(wordId);
27     }
28 }
29
```

File - SpellingGameStateMachine.java

```java
package viewmodel;

import com.badlogic.gdx.scenes.scene2d.ui.Container;
import model.DataManager;
import model.Word;
import view.actors.Letter;
import view.games.SpellingGameScreen;

import java.util.ArrayList;

public class SpellingGameStateMachine {
    private State state;
    private SpellingGameScreen spellingGameScreen;
    private Language language;
    private Word currentWord;

    public SpellingGameStateMachine(SpellingGameScreen spellingGameScreen) {
        state = State.COMPLETING_WORD;
        this.spellingGameScreen = spellingGameScreen;

        language = Language.ENGLISH;
        spellingGameScreen.setDisplayLanguage(language);

        currentWord = DataManager.getWord("bird");
        spellingGameScreen.setPictureAndWordLength(currentWord.getWordId(), currentWord.getSpelling(language).length());
    }

    public void doEvent(Event event) {
        switch (state) {
            case ANIMATION:
                return;
            case COMPLETING_WORD:
                switch (event) {
                    case DROPPED_LETTER:
                        if (wordIsCorrect()) {
                            changeToNextWord();
                        }
                }
                break;
        }
    }

    private boolean wordIsCorrect() {
        ArrayList<Container<Letter>> letterContainers = spellingGameScreen.getLetterContainers();
```

File - SpellingGameStateMachine.java

```
45          String currentString = "";
46          for (Container<Letter> letterContainer : letterContainers) {
47              if (letterContainer.getActor() == null) {
48                  currentString += " ";
49              } else {
50                  currentString += letterContainer.getActor().getName(
    );
51              }
52          }
53          return currentWord.getSpelling(language).equals(
    currentString);
54      }
55
56      private void changeToNextWord() {
57          currentWord = DataManager.getWord("gem");
58          spellingGameScreen.setPictureAndWordLength(currentWord.
    getWordId(), currentWord.getSpelling(language).length());
59      }
60
61      public enum State {
62          ANIMATION, COMPLETING_WORD, GAME_COMPLETE
63      }
64
65      public enum Event {
66          DROPPED_LETTER
67      }
68
69      public enum Language {
70          ENGLISH, HMONG
71      }
72  }
73
```