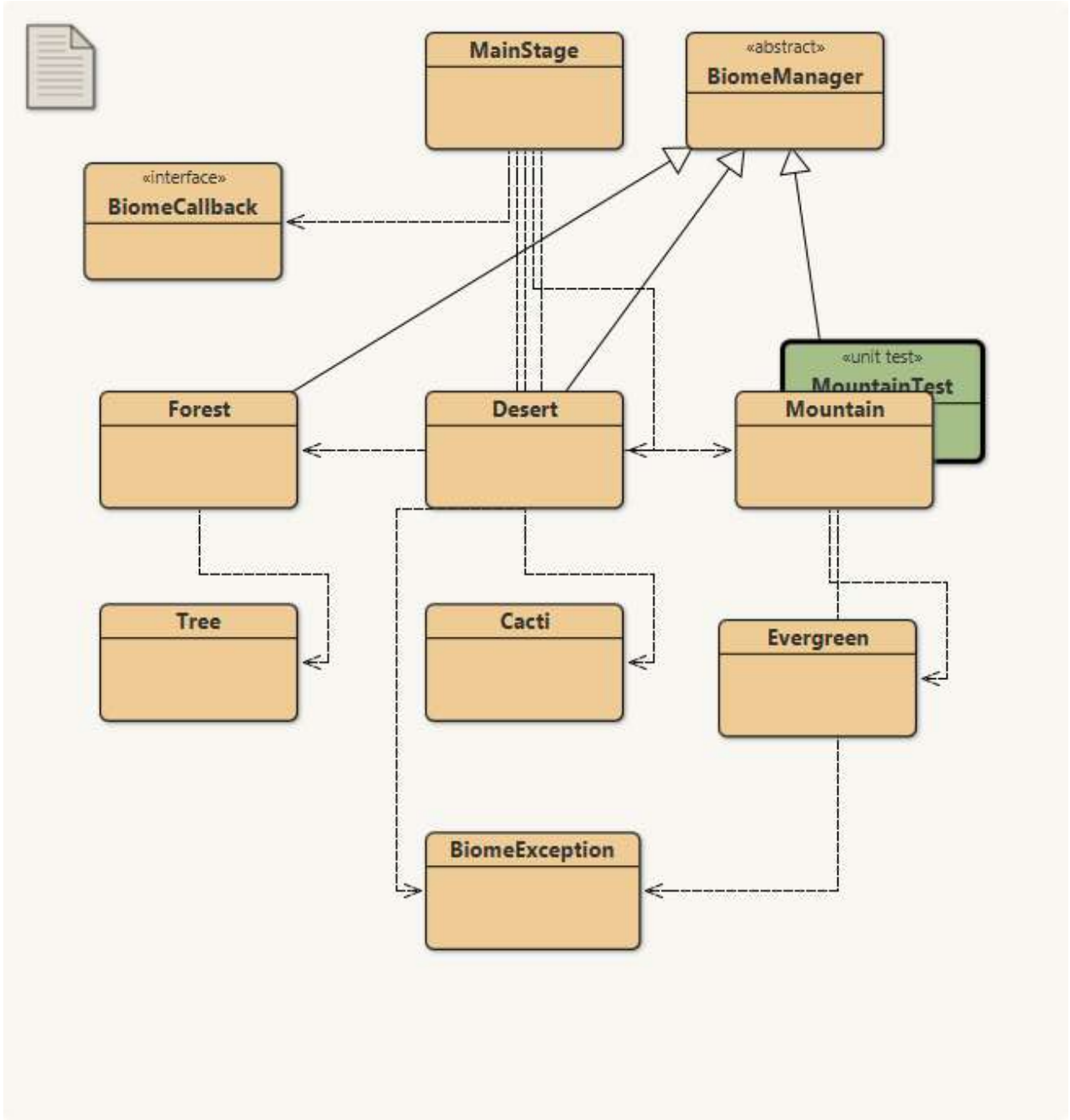
MainStage
«abstract»
BiomeManager
«interface»
BiomeCallback
«unit test»
MountainTest
Forest
Desert
Mountain
Tree
Cacti
Evergreen
BiomeException

```java
import javafx.scene.Group;
//
public abstract class BiomeManager
{
    public Group fullGroup = new Group();
    public abstract Group grow();
    //D3 my abstract method grow() is overriden in all the subclasses and specifici biomes

    //D4 overridden in subclasses using super
     public void setupBiome() {
        System.out.println("Setting up base biome");
        fullGroup.getChildren().clear();
    }


}

```

```java
public class BiomeException extends Exception
{
    //D10/11 more in Mountain and MainStage
    public BiomeException(String message){
        super(message);
    }
}
```

```java
import javafx.scene.Group;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;

import java.util.ArrayList;

public class Mountain extends BiomeManager{
    private Image mountain = new Image("mountain.jpg");
    private ImageView ivMount = new ImageView(mountain);
    private Group treeGroup = new Group();
    private Group fullGroup = new Group();
    private ArrayList<Evergreen> treeList = new ArrayList<>();
    private ArrayList<ImageView> ivList = new ArrayList<>();
    private TreeGenerator treeGen = new TreeGenerator();

    public void clear(){
        treeList.clear();
        ivList.clear();
        treeGroup.getChildren().clear();
    }
    @Override
    public void setupBiome() {
        //D4 Setup biome is being overrided using super
        super.setupBiome();
        System.out.println("Adding mountain and trees");
    }
    @Override
    public Group grow(){

        treeList.clear();
        ivList.clear();
        treeGroup.getChildren().clear();
        ivMount.setFitWidth(650);
        ivMount.setFitHeight(400);
        fullGroup.getChildren().clear();
        fullGroup.getChildren().addAll(ivMount, treeGroup);

        //D10 Try-catch exception to check if the image is accepted
        for (int i = 0; i < 10; i++){
            try
            {
                treeGen.growOneTree();
            }
            catch (BiomeException be)
            {
                be.printStackTrace();
            }
        }
        try {
        //D4 Setup biome is being overrided using super
```

```
        super.setupBiome();
        System.out.println("Growing Evergreems");
        for (int i = 0; i < 5; i++){
            treeGen.growOneTree();
            }
        }
        catch (BiomeException e){
            System.err.println("Tree growing failed: " + e.getMessage());
            //D11 Handle exception by printing out fail statement
        }

        return fullGroup;
    }
    public void growExtraTree() throws BiomeException {
        treeGen.growOneTree();
    }

    //D6 inner class of growOneTree inside of treeGenerator
    private class TreeGenerator{
        void growOneTree() throws BiomeException{
        //D2 Simpler operation like getting the image are done in the subclass
        Evergreen evergreen = new Evergreen();
        treeList.add(evergreen);
        System.out.println("Tree added");

        Image treeImg = evergreen.create(1);
        if (treeImg == null){//check if the image actually loaded
        throw new BiomeException("Failed to load tree");
        }

        ImageView ivTree = new ImageView(treeImg);
        ivTree.setX(Math.random() * 500);
        ivTree.setY(Math.random() * 300);
        ivTree.setFitWidth(100);
        ivTree.setFitHeight(170);
        ivTree.setPreserveRatio(true);

        ivList.add(ivTree);
        treeGroup.getChildren().add(ivTree);
    }
}
    public Group getGroup(){
        return fullGroup;
    }
}

```

```java
import javafx.scene.Group;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import java.util.ArrayList;
import javafx.scene.layout.Pane;
public class Cacti
{
    public Image create(int amt){
    Image cactus = new Image("cactus.jpg");
    return cactus;

}

}

```

```java
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.layout.BorderPane;
import javafx.scene.layout.GridPane;
import javafx.scene.control.Button;
import javafx.scene.layout.Pane;
import javafx.scene.control.RadioButton;
import javafx.scene.control.ToggleGroup;
import javafx.scene.layout.StackPane;
import javafx.scene.control.ToggleButton;
import javafx.scene.layout.VBox;
import javafx.scene.text.*;
import javafx.scene.control.Label;
import javafx.scene.layout.HBox;


//TYPES OF RELARTIONSHIPS:
//D7 Composition is seen in the types of trees/cacti and their respective
//biome, the Forest HAS A tree
//D8 Dependency is seen with the mainstage and the biomemanager which is requires to create
//the biomes

public class MainStage extends Application{
    private BiomeCallback callback;
    public void setBiomeCallback(BiomeCallback callback) {
        this.callback = callback;
    }

    @Override
  public void start(Stage primaryStage){
    //create objects
    //D2
    Forest myFor = new Forest();
    Desert myDes = new Desert();
    Mountain myMount = new Mountain();

    Text buttonsText = new Text("Biome Buttons: ");
    buttonsText.setFont(Font.font("Georgia", FontWeight.BOLD, 20));
    Label label = new Label("World Generator");
    label.setFont(Font.font("Georgia", FontWeight.BOLD, 40));
    Text intructions = new Text("Click the buttons to change the biome or click anywhere in the middle to change time of day!");
    intructions.setFont(Font.font("Georgia", FontWeight.BOLD, 20));

    //create panes with all the biomes
    Pane forestPane = new Pane(myFor.grow());
    Pane desertPane = new Pane(myDes.grow());
    Pane mountainPane = new Pane(myMount.grow());
```

```
49
50      // Set default visible pane
51      forestPane.setVisible(true);
52      desertPane.setVisible(false);
53      mountainPane.setVisible(false);
54
55      //Buttons
56      Button growButton = new Button("Grow");
57      growButton.setFont(Font.font("Cal Sans", FontWeight.BOLD, 30));
58      Button clearButton = new Button("Clear");
59      clearButton.setFont(Font.font("Cal Sans", FontWeight.BOLD, 30));
60
61      //Toggle buttons
62      ToggleGroup toggleGroup = new ToggleGroup();
63      ToggleButton forestBtn = new ToggleButton("Forest");
64      forestBtn.setFont(Font.font("Cal Sans", FontWeight.BOLD, 30));
65      ToggleButton desertBtn = new ToggleButton("Desert");
66      desertBtn.setFont(Font.font("Cal Sans", FontWeight.BOLD, 30));
67      ToggleButton mountainBtn = new ToggleButton("Mountain");
68      mountainBtn.setFont(Font.font("Cal Sans", FontWeight.BOLD, 30));
69
70      forestBtn.setToggleGroup(toggleGroup);
71      desertBtn.setToggleGroup(toggleGroup);
72      mountainBtn.setToggleGroup(toggleGroup);
73
74      //border and other layout
75      BorderPane root = new BorderPane();
76      HBox hbox = new HBox(100, buttonsText, label );
77      VBox controls = new VBox(30, forestBtn, desertBtn, mountainBtn);
78      VBox otherControls = new VBox(30, growButton,clearButton);
79      root.setTop(hbox);
80      root.setLeft(controls);
81      root.setRight(otherControls);
82      root.setBottom(intructions);
83
84      Pane backgroundPane = new Pane();
85      //create stackpane
86      StackPane biomeStack = new StackPane();
87      biomeStack.getChildren().add(0, backgroundPane);
88      biomeStack.getChildren().addAll(forestPane, desertPane, mountainPane);
89      //set bg pane color
90      backgroundPane.setStyle("-fx-background-color: lightblue;");//default  day
91      root.setCenter(biomeStack);
92
93      //button events\
94      //F1 buttons allow for user interactions
95      //F4 which chnages the proporties of the visual element
96      growButton.setOnAction(e -> {
97      //F2 Growing GUI components based on user
98      myFor.growOneTree();
```

```
    myDes.growOneCactus();
    //D10 try catch the growOnetree method
    try{
        myMount.growExtraTree();
    }
    catch (BiomeException be){
        //D11 handle the exception and print stack
        be.printStackTrace();
    }
    if (callback != null) {
        //D1 Interface is being calledf
        //D5a Callback is being made
        callback.onGrow("Biome elements grew!");
    }
    });

    clearButton.setOnAction(e -> {
    myFor.clear();
    myDes.clear();
    myMount.clear();
    });

    forestBtn.setOnAction(e -> {
    forestPane.setVisible(true);
    desertPane.setVisible(false);
    mountainPane.setVisible(false);
    });

    desertBtn.setOnAction(e -> {
    forestPane.setVisible(false);
    desertPane.setVisible(true);
    mountainPane.setVisible(false);
    });

    mountainBtn.setOnAction(e -> {
    forestPane.setVisible(false);
    desertPane.setVisible(false);
    mountainPane.setVisible(true);
    });
    //F3 mouse click for changing the day and night
    boolean[] isNightMode = {false};
    biomeStack.setOnMouseClicked(e -> {
        isNightMode[0] = !isNightMode[0];
    if (isNightMode[0]){
        backgroundPane.setStyle("-fx-background-color: darkblue;");
    }
    else{
        backgroundPane.setStyle("-fx-background-color: lightblue;");
    }
});
```

```
149
150
151     Scene scene = new Scene(root, 1000, 600);//create the scene
152     primaryStage.setScene(scene);//set the scene to the stage
153     primaryStage.show();
154   }
155   public static void main(String[] args) {
156     MainStage stage = new MainStage();
157     stage.setBiomeCallback(biomeName -> System.out.println("Callback for: " +
biomeName));
158
159     Application.launch(args);
160 }
161 }
162
163
164
165
```

```java

import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import javafx.scene.Group;
public class MountainTest
{
    //B3 Testing if grow actually grows something and returns the right type, and testing if clear works
    @Test
    public void testGrow(){
        Mountain mountain = new Mountain();
        Group result = mountain.grow();
        assertNotNull(result, "Biome grow() should return a Group object");
        assertFalse(result.getChildren().isEmpty(), "Biome should contain elements after growth");
    }

    @Test
    public void testClear(){
        Mountain mountain = new Mountain();
        mountain.grow();
        mountain.clear();
        assertTrue(mountain.getGroup().getChildren().isEmpty(),"Biome should be empty when i clear");
    }
}

```

```java
import javafx.scene.shape.Circle;
import javafx.scene.Group;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import java.util.ArrayList;
import javafx.scene.layout.Pane;

public class Forest extends BiomeManager{
    private ArrayList<Tree> treeList = new ArrayList<>();
    private ArrayList<ImageView> ivList = new ArrayList<>();
    private Group treeGroup = new Group();
    private ImageView ivGrass;
    private Group fullGroup;
    public void clear(){
        treeList.clear();
        ivList.clear();
        treeGroup.getChildren().clear();
    }
    @Override
    public void setupBiome(){
        //D4 Setup biome is being overrided using super
        super.setupBiome();
        System.out.println("Adding grass and trees");
    }
    @Override
    public Group grow(){
        Image grass = new Image("grass.jpg");
        ivGrass = new ImageView(grass);

        //clear old trees
        treeList.clear();
        ivList.clear();
        treeGroup.getChildren().clear();
        //grow 10 trees
        for (int i = 0; i < 10; i++) {
            growOneTree();
        }

        //combine grass background with trees
        fullGroup = new Group(ivGrass, treeGroup);
        return fullGroup;
    }
    public void growOneTree(){
        Tree treeObj = new Tree();
        treeList.add(treeObj);
        System.out.println("Tree added");

        Image treeImg = treeObj.create(0);
```

```
        ImageView ivTree = new ImageView(treeImg);

        //Set position and size
        ivTree.setX(Math.random() * 500);
        ivTree.setY(Math.random() * 300);
        ivTree.setFitWidth(50);
        ivTree.setFitHeight(100);
        ivTree.setPreserveRatio(true);

        ivList.add(ivTree);
        treeGroup.getChildren().add(ivTree);
    }
    public Group getGroup() {
        return fullGroup;
    }
}

```

```java
import java.util.ArrayList;
import javafx.scene.shape.Circle;
import javafx.scene.Group;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
public class Tree
{
    Image tree= new Image("pine.png");;
    public Image create(int amt){
    if(amt == 1){
        Image tree = new Image("pine.png");
    }
    //if(amt == 0){
    //    Image tree = new Image("snowytree.png");
    //}
    return tree;
}
}

```

```java
import javafx.scene.Group;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;

import java.util.ArrayList;

public class Desert extends BiomeManager{
    private Image desert = new Image("desert.jpg");
    private ImageView ivDesert = new ImageView(desert);
    private Group cactiGroup = new Group();
    private Group fullGroup = new Group();
    private ArrayList<Cacti> cactiList = new ArrayList<>();
    private ArrayList<ImageView> ivList = new ArrayList<>();

    public void clear(){
        cactiList.clear();
        ivList.clear();
        cactiGroup.getChildren().clear();
    }
    @Override
    public void setupBiome(){
        //D4 Setup biome is being overrided using super
        super.setupBiome();
        System.out.println("Adding desert and cacti");
    }
    @Override
    public Group grow() {

        cactiList.clear();
        ivList.clear();
        cactiGroup.getChildren().clear();

        ivDesert.setFitWidth(650);
        ivDesert.setFitHeight(400);
        fullGroup.getChildren().clear();
        fullGroup.getChildren().addAll(ivDesert, cactiGroup);

        //grow 10
        for (int i = 0; i < 10; i++){
            growOneCactus();
        }
        return fullGroup;
    }

    //grow one
    public void growOneCactus(){
        //D2 Simpler operation like getting the image are done in the subclass
        Cacti cactus = new Cacti();
        cactiList.add(cactus);
        System.out.println("Cactus added");
```

```
        Image cactusImg = cactus.create(1);
        ImageView ivCactus = new ImageView(cactusImg);
        //position and size
        ivCactus.setX(Math.random() * 500);
        ivCactus.setY(Math.random() * 300);
        ivCactus.setFitWidth(70);
        ivCactus.setFitHeight(140);
        ivCactus.setPreserveRatio(true);

        ivList.add(ivCactus);
        cactiGroup.getChildren().add(ivCactus);
    }
    public Group getGroup() {
        return fullGroup;
    }
}

```

```java
@FunctionalInterface
//D5a FUnctional Interface using callbacks (More in MainStage)
public interface BiomeCallback {
    void onGrow(String biomeName);
}
```

```
1  import javafx.scene.Group;
2  import javafx.stage.Stage;
3  import javafx.scene.Scene;
4  import javafx.scene.image.Image;
5  import javafx.scene.image.ImageView;
6  import java.util.ArrayList;
7  import javafx.scene.layout.Pane;
8
9  public class Evergreen
10 {
11     public Image create(int amt){
12     Image evergreen = new Image("evergreen.png");
13     return evergreen;
14 }
15
16 }
17
```