



```
1  /**
2      * The main entry point for the application.
3      *
4      * @param args the command line arguments
5      */
6  public static void main(String[] args) {
7      launch(args);
8  }
```



```
1  /**
2      * Initializes the primary stage of the application.
3      *
4      * @param primaryStage the primary stage
5      */
6  @Override
7  public void start(Stage primaryStage) {
8      this.primaryStage = primaryStage;
9      initializePrimaryStage();
10 }
```

```
1  /**
2      * Initializes the primary stage with the title and initial screen.
3      */
4  private void initializePrimaryStage() {
5      primaryStage.setTitle("Puzzle game");
6      setupInitialScreen();
7      primaryStage.show();
8      adjustMainMenuSize();
9      centerStage(primaryStage);
10 }
```

```
1  /**
2      * Centers the given stage on the screen.
3      *
4      * @param stage the stage to center
5      */
6  private void centerStage(Stage stage) {
7      Screen screen = Screen.getPrimary();
8      Rectangle2D bounds = screen.getVisualBounds();
9      double centerX = bounds.getMinX() + (bounds.getWidth() - stage.getWidth()) / 2;
10     double centerY = bounds.getMinY() + (bounds.getHeight() - stage.getHeight()) / 2;
11     stage.setX(centerX);
12     stage.setY(centerY);
13 }
```



```
1  /**
2      * Navigates to the main menu screen.
3      */
4  private void goToMainMenu() {
5      setupMainMenu();
6  }
```



```
1  /**
2      * Sets up the main menu UI components.
3      */
4  private void setupMainMenu() {
5      Text mainMenuLabel = createText("Main Menu", 24, Color.VIOLET, TextAlignment.CENTER);
6      Button game1Button = createButton("Game 1", 24, 160, 20, 0, this::goToGame1);
7      Button game2Button = createButton("Game 2", 24, 160, 20, 0, this::goToGame2);
8      Button returnButton = createButton("Return", 24, 160, 20, 0, this::initializePrimaryStage);
9      Button exitButton = createButton("Exit", 24, 160, 20, 0, Platform::exit);
10
11      VBox menuLayout = new VBox(20);
12      menuLayout.setAlignment(Pos.CENTER);
13      menuLayout.getChildren().addAll(mainMenuLabel, game1Button, game2Button, returnButton, exitButton);
14
15      root.getChildren().clear();
16      root.getChildren().add(menuLayout);
17  }
```

```
1  /**
2   * Sets up the initial screen with a welcome message and play button.
3   */
4   private void setupInitialScreen() {
5       Text text = createText("Welcome to the puzzle game!", 40, Color.BLUE, TextAlignment.CENTER);
6       Button playButton = createButton("Play", 25, 160, 80, 200, this::goToMainMenu);
7
8       root = createRootPane(text, playButton, "file:./assets/Background1.jpg");
9
10      Scene scene = createScene(root, primaryStage.getWidth(), primaryStage.getHeight());
11      primaryStage.setScene(scene);
12  }
```

```
1  /**
2   * Adjusts the size of the main menu to fit the screen.
3   */
4   private void adjustMainMenuSize() {
5       Screen screen = Screen.getPrimary();
6       Rectangle2D bounds = screen.getVisualBounds();
7       primaryStage.setWidth(bounds.getWidth());
8       primaryStage.setHeight(bounds.getHeight() - 27);
9   }
```

```


1  /**
2   * Creates a Text component with the specified content, font size, color, and alignment.
3   *
4   * @param content    the text content
5   * @param fontSize   the font size
6   * @param color      the text color
7   * @param alignment  the text alignment
8   * @return the created Text component
9   */
10 private Text createText(String content, double fontSize, Color color, TextAlignment alignment) {
11     Text text = new Text(content);
12     text.setStyle("-fx-font-size: " + fontSize + "");
13     text.setFill(color);
14     text.setFont(new Font(40));
15     text.setTextAlignment(alignment);
16     return text;
17 }

```


```

1  /**
2   * Creates a Button component with the specified text, font size, width, height, translateY, and action.
3   *
4   * @param text        the button text
5   * @param fontSize    the font size
6   * @param width       the button width
7   * @param height      the button height
8   * @param translateY  the translation on the y-axis
9   * @param action      the action to be performed when the button is clicked
10  * @return the created Button component
11  */
12 private Button createButton(String text, int fontSize, int width, int height, int translateY, Runnable action) {
13     Button button = new Button(text);
14     button.setStyle("-fx-font-size: " + fontSize + "");
15     button.setPrefSize(width, height);
16     button.setTranslateY(translateY);
17     button.setOnAction(e -> {
18         updateContent("Loading...");
19         simulateDelay(action);
20     });
21     return button;
22 }

```



```
1  /**
2   * Creates a StackPane as the root pane with the specified Text, Button, and image path.
3   *
4   * @param text    the Text component
5   * @param button  the Button component
6   * @param imagePath the path to the background image
7   * @return the created StackPane
8   */
9  private StackPane createRootPane(Text text, Button button, String imagePath) {
10     root = new StackPane();
11     root.getChildren().addAll(text, button);
12     root.setBackground(createBackground(imagePath));
13     return root;
14 }
```



```
1  /**
2   * Creates a Scene with the specified root pane, width, and height.
3   *
4   * @param root    the root pane
5   * @param width   the scene width
6   * @param height  the scene height
7   * @return the created Scene
8   */
9  private Scene createScene(StackPane root, double width, double height) {
10     return new Scene(root, width, height);
11 }
```



```
1  /**
2   * Updates the content of the root pane with the specified text.
3   *
4   * @param content the new content
5   */
6  private void updateContent(String content) {
7      root.getChildren().clear();
8      Text text = new Text(content);
9      text.setStyle("-fx-font-size: 24;");
10     text.setFont(new Font(40));
11     root.getChildren().add(text);
12 }
```



```
1  /**
2   * Simulates a delay of 1 second and then runs the specified action on the JavaFX application thread.
3   *
4   * @param action the action to be performed after the delay
5   */
6  private void simulateDelay(Runnable action) {
7      new Thread(() -> {
8          try {
9              Thread.sleep(1000);
10             } catch (InterruptedException e) {
11                 e.printStackTrace();
12             }
13             Platform.runLater(action);
14         }).start();
15 }
```

```

1  /**
2   * Creates a Background with the specified image path.
3   *
4   * @param imagePath the path to the background image
5   * @return the created Background
6   */
7  private Background createBackground(String imagePath) {
8      Image image = new Image(imagePath);
9
10     BackgroundImage backgroundImage = new BackgroundImage(
11         image,
12         BackgroundRepeat.NO_REPEAT,
13         BackgroundRepeat.NO_REPEAT,
14         BackgroundPosition.CENTER,
15         new BackgroundSize(BackgroundSize.AUTO, BackgroundSize.AUTO, false, false, true, false)
16     );
17
18     return new Background(backgroundImage);
19 }

```

```

1  /**
2   * Navigates to the Game 1 screen.
3   */
4  private void goToGame1() {
5      ContinuousJumpingGame game1 = new ContinuousJumpingGame();
6      Scene game1Scene = new Scene(game1.getRoot(), primaryStage.getWidth(), primaryStage.getHeight());
7      primaryStage.setScene(game1Scene);
8
9      game1Scene.getWindow().setOnCloseRequest(event -> {
10         if (event.getEventType() == WindowEvent.WINDOW_CLOSE_REQUEST) {
11             goToMainMenu();
12         }
13     });
14 }

```




```
1  /**
2   * Navigates to the Game 2 screen.
3   */
4  private void goToGame2() {
5      SudokuGame game2 = new SudokuGame();
6      Scene game2Scene = new Scene(game2.getRoot(), primaryStage.getWidth(), primaryStage.getHeight());
7      primaryStage.setScene(game2Scene);
8
9      game2Scene.getWindow().setOnCloseRequest(event -> {
10         if (event.getEventType() == WindowEvent.WINDOW_CLOSE_REQUEST) {
11             goToMainMenu();
12         }
13     });
14 }
```