**CSE 1242- COMPUTER PROGRAMMING II**

**TERM PROJECT**

**“PIPE PUZZLE”**

**Ahmet Abdullah Ömer Deligöz**

**Gültekin**

**150121025 150120035**

CSE1242: Computer Programming II, Spring 2022

Date Submitted: May 8, 2022

**Problem Definition**

* There are 16 tiles on the game board.
* Tiles have different types (e.g. Starter, End, Pipe, CurvedPipe...)
* Tiles have different properties (e.g. horizontal, vertical, free, none, 00 …)
* Tiles with static property and starter/end tiles cannot move.
* Tiles can move one tile distance, vertically and horizontally, if the desired tile is empty free.
* Tiles cannot move diagonally.
* If the puzzle is complete, the ball starts moving from the starter tile to the end tile.
* After the ball animation is over, a greeting message will appear, and the user can move on to the next level.
* The user cannot play the next level until a level is completed.
* If all levels are completed, a congratulatory message will appear.

**Implementation Details**

Uml Diagrams

|  |  |
| --- | --- |
| **Main** | |
| -  -  -  -  -  -  -  #  +  -  -  -  #  -  -  -  -  - | input: String  icon: Image  pane: AnchorPane  fold: File  allLevels: File[]  levels: File[]  orderOfLevels: String  starterIndex: int  saveLevel: File  scan: Scanner  pr: PrintWriter  music: Media  mediaPlayer: MediaPlayer  circlePink: Circle  circleTurquoise: Circle  circleWhite: Circle  circleOrange: Circle  musicLabel: Label |
| +  +  -  -  +  - | main(args: String[]): void  start(primaryStage: Stage): void  entranceAnimation(pane: AnchorPane): void  findLevels(): void  findOrderOfLevels(): void  makeTiles(levels: File): Image[]  StartStopMusic(runPause: Runnable, runPlay: Runnable) |
| + | getter/setter methods |

* This class has several data fields, and methods.
* **start** method creates a main menü pane and adds it to a scene and shows them in primary stage
* **findLevels** method find the number of levels in game according to the Levels folder.
* **findOrderOfLevels** method ordered the level files
* **MakeTiles** method reads the level file according to the level to be played and creates each tile.
* **entranceAnimation** method creates an animation in the main menu.

|  |  |
| --- | --- |
| **Controller** | |
| -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  #  -  -  -  -  - | game: GridPane  menu: AnchorPane  creditsPane: AnchorPane  cboxLevel: ComboBox<?>  btPlay: Button  btCredits: Button  btMenuQuit: Button  btBacktoMenu: Button  btRestartLevel: Button  ellipseBlue: Ellipse  ellipseGreen: Ellipse  ellipseRed: Ellipse  circle: Circle  ball: Circle  labelCounter: Label  musicLabel: Label  stage: Stage  scene: Scene  main: Main  selectedLevel: String  content: ClipboardContent  iwLast: ImageView  lastTouchedIW: ImageView  resetImages: Image[]  switchedImages: Image[]  fold: File  allLevels: File[]  levels: File[]  orderOfLevels: String[]  counter: short  levelFile: File  firstsRow: int  firstsColumn: int  lastsRow: int  lastsColumn: int  selectedColor: Color  path: Path  pathIndex: ArrayList<Integer>  directions: ArrayList<Integer>  read: Scanner  pr: PrintWriter |
| +  +  +  -  -  -  +  -  +  +  +  + | backToMenu(event: ActionEvent): void  playButton(event: ActionEvent): void  selectLevel(event: ActionEvent): void  isStatic(image: Image): Boolean  findLevels(): File[]  findOrderOfLevels(): void  checkSolution(): void  drawPath(): void  startAnimation(): void  credits(): void  Reset(e: ActionEvent): void  Quit(e: ActionEvent): void |
| + | getter/setter methods |

* This class has several data fields, some button actions, and methods.
* **backToMenu** method works every time the Back to Menu button is pressed and displays the main menu screen to the user.
* **playButton** method works every time the Play button is pressed and displays the game screen for the selected level.
* **selectLevel** method keeps the selectedLevel value according to the level selected by the user.
* **isStatic** method returns a boolean value depending on whether the tile to be moved is static or not. If that tile is static it cannot be moved.
* **findLevels** method find the number of levels in game according to the Levels folder.
* **findOrderOfLevels** method ordered the level files
* **checkSolution** method checks if the level is complete after each move, and calls drawPath method if completed.
* **drawPath** method adds creates animation for all tiles between start and end path and combine these animations.
* **startAnimation** method shows the finish animation and after the animation is over, it will shows the level completed message to the user.
* **credits** method works every time the Credits button is pressed and shows the credits stage to the user.
* **Reset** method works every time the Restart Level button is pressed and restarts the level being played.
* **Quit** method works every time the Quit button is pressed and the game is exited.

|  |  |
| --- | --- |
| **Pipe** | |
| -  -  -  - | id: int  type: String  property: String  image: Image |
| + | Pipe(id: int, type: String, property: String) |
| + | getter/setter methods |

|  |  |
| --- | --- |
| **PipeStatic** | |
| -  -  -  - | id: int  type: String  property: String  image: Image |
| + | PipeStatic(id: int, type: String, property: String) |
| + | getter/setter methods |

|  |  |
| --- | --- |
| **Starter** | |
| -  -  -  - | id: int  type: String  property: String  image: Image |
| + | Starter(id: int, type: String, property: String) |
| + | getter/setter methods |

|  |  |
| --- | --- |
| **End** | |
| -  -  -  - | id: int  type: String  property: String  image: Image |
| + | End(id: int, type: String, property: String) |
| + | getter/setter methods |

|  |  |
| --- | --- |
| **Empty** | |
| -  -  -  - | id: int  type: String  property: String  image: Image |
| + | Empty(id: int, type: String, property: String) |
| + | getter/setter methods |

**For Pipe/PipeStatic/Empty/Starter/End classes:**

* These classes contain constructors for different tiles
* Each tile should have an id, a type, and a property data field.
* Various tiles in the game are created and a picture is defined for each tile according to the information given in the level files.

**What have we completed?**

* Input file reading
* Creating the game board
* Static and non-static tiles
* Displaying buttons
* Counting number of moves
* Checking solution for any level
* Win animation
* Level lock system (the user will not be able to play the next level without completing a previous one.)

**What difficulties did we have?**

* Checking solution and win animation.

**Additional Functionalities**

* Main menu screen
* Credits screen
* Extra animations on main menu and credits screens
* Color of the ball can be changed.
* Quit button on main menu
* Restart level and back to menu button in game screen
* Music
* An error message that occurs when level is not selected
* A congratulatory message when the level is completed
* A congratulatory message when the game is completed

Test Cases

|  |  |
| --- | --- |
| metin, ekran, elektronik eşyalar, iPod içeren bir resim  Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, iPod, vektör grafikler içeren bir resim  Açıklama otomatik olarak oluşturuldu | User can change the color of the ball with the help of color picker |
| metin, ekran, dizüstü, ekran görüntüsü içeren bir resim  Açıklama otomatik olarak oluşturuldu | User can find out the developers name using the credits button |
| metin, ekran görüntüsü, ekran, iPod içeren bir resim  Açıklama otomatik olarak oluşturuldu | If the user has unlocked the level, s/he can choose a level s/he wants with the help of the combobox and play that level. |
|  | If the user presses the play button without selecting any level, an error message will appear. |
|  | The level number being played is written above the game window |
|  | The number of moves made by the user is written at the bottom of the game window |
|  | If the game is too confusing, the user can restart the level and try again with the restart level button. |
|  | When the user completes the level, a congratulatory message will appear and they can go to the next level with the next level button. |
|  | A congratulatory message appears when the user completes all levels |