Kieran Abelen

Mell1Games

Unity Methodology 2

Contents

[Introduction 2](#_Toc55141155)

[UI Development 3](#_Toc55141156)

[References 4](#_Toc55141157)

Unity

# Introduction

## Brief

This second methodology will document my learning when it comes to deploying and production a game to be posted on a website. The game is in a complete state but needs much more polish and testing to be ready for deployment.

Like before many things will be changed in the introduction as I learn new information and end up adding it to the overall methodology. Each heading under the instruction will try an summarize my goals as they evolve with each section.

Static levels should be the initial priority, but the following methodology should delve into procedural generation of levels.

# UI Development

## Brief

This section will entail Completion of the UI elements and menus. There will be three headings, one will be the wireframes that where the result of me using unity to make the menus. The second will be the testing of these UI elements, and the final third will be the look and color of all the screens.

## Wireframe

These diagrams all use a star as the anchor point for the image, so all positioning is based of the relation to that centred anchor.

The screen size is based of 800 by 600, so the menus all take about half the screen in total with exception of map selection. The screen will auto resize based on the platform either scaling up or down my initial screen size.

### Main Menu



### Select Level



### Popup Menu



## UI Testing

I only need to test the ability to move to each page at this point since I have not hosted the game yet.

### UI Functionality Test Case

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Case | Goal | Steps | Expected Result | Result |
| 1 | Main transitions | Button Start | Level started | Completed |
|  |  | Button Select Level | Transition and levels populated | Completed |
|  |  |  |  |  |
| 2 | Select Level transitions | Button Level 1 | Level started | Completed |
|  |  | Button Back | Transition to Main Menu | Completed |
|  |  |  |  |  |
| 3 | Popup transitions | Button Next | Transition to next level | Not Complete |
|  |  | Button Main Menu | Transition to Main Menu | Not Complete |
|  |  | Button Select Level | Transition and levels populated | Not Complete |
|  |  |  |  |  |
| 4 | Game transitions | Game completed | Transition to Popup | Completed |
|  |  | Pause game | Transition to Popup (no next level button) | Not Complete |

### Popup transitions Fix

The main issue is the game screen does hide itself once started; this has led to hiding the underlying menu with a white screen. I need the ability to turn off the game screen and, in the process, resetting it to a before launched screen.

#### Solution

It turns out if you have screen being turn off outside the main game loop it wont work, so the solution is placing it in levelupdate fucntion. By turning off the mapClick game onject here it will unload the game.

if (level.tileSelected.SequenceEqual(level.tileCorrect))

{

// Debug.Log("Won");

levelActive\_state = false;

UIMenuPopup.gameObject.SetActive(true);

GameTilemap\_clickable.gameObject.SetActive(false);

UIButtonGametoMain.gameObject.SetActive(false);

}

### Game transitions (pause game) Fix

This one is a little trickier since the real issue it with the game states. We have an active game state which will let the update have access to the LevelUpdate function but what we need a way to exit like when completing a level.

#### Solution

I need another state for the game itself and not the level (loaded or not). The pause game will just provide exit from the LevelUpdate function while it is running. It will be a hard exit so closing the game Tilemap but it needs to directly tied to a state so it can be changed with a button.

if (level.tileSelected.SequenceEqual(level.tileCorrect))

{

// Debug.Log("Won");

levelActive\_state = false;

UIMenuPopup.gameObject.SetActive(true);

GameTilemap\_clickable.gameObject.SetActive(false);

UIButtonGametoMain.gameObject.SetActive(false);

}

else if (gameActive\_state == false)

{

levelActive\_state = false;

UIMenuMain.gameObject.SetActive(true);

GameTilemap\_clickable.gameObject.SetActive(false);

UIButtonGametoMain.gameObject.SetActive(false);

}

Then a button can just flick the state as if it completed the level.

public void StateGame\_pause()

{

gameActive\_state = false;

}

# New Levels

## Brief

The first thing that needs to take place to add new level is the json files need to be more readable. They currently look like this, which means they are upside-down. Since the ones should be on top like how they are rendered.

1,2,2,2,

1,2,2,2,

0,0,0,0,

0,0,1,1

The next step will be making at least 10 levels that can be in published version. These need to develop inside separate json files and be correctly format. The 3 Tilemaps should be added to subheads for each map number.

Each one should be play tested to see if it can be completed. Any updates should also be listed under the table for playtesting

# Tutorial

# Publish the Game

# References

**There are no sources in the current document.**