Game Template - Ultimate Tap Games Maker

Introduction:

Introducing **Game Template - Ultimate Tap Games Maker**: an exceptional Unity asset designed to empower hyper casual game developers and revolutionize game creation. This all-in-one package offers a seamless plug-and-play experience, enabling users to effortlessly craft their own captivating tap-based games.

With a wide array of remarkable features and functionalities, this asset streamlines the game development process. Its meticulously crafted scripts ensure a well-structured foundation, allowing developers to focus on unleashing their creativity without worrying about complex coding.

One of the standout features of this asset is its collection of cool and unique sprites. These visually stunning graphics inject life and vibrancy into every game, captivating players and creating an immersive experience. The asset's versatile nature accommodates both vertical and horizontal moving games, giving developers the freedom to explore various gameplay mechanics and styles.

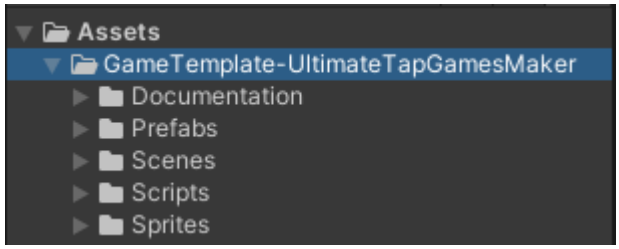
Additionally, the asset includes a high score feature that keeps players engaged and motivated to achieve their personal best. Moreover, the score-to-unlock stages feature adds an exciting element of progression, offering a sense of accomplishment and unlocking new challenges as players advance.

By providing an all-encompassing solution, the Ultimate Tap-Based Games Maker empowers game developers of all skill levels to bring their ideas to life. Whether you're a seasoned professional or a beginner taking their first steps into the world of game development, this asset is your ultimate companion, enabling you to create captivating hyper casual games with ease.

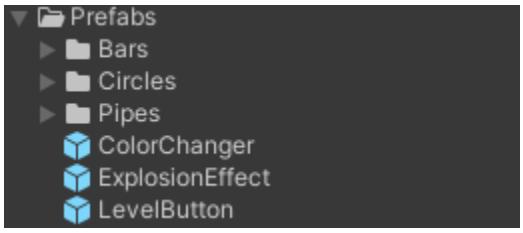
In summary, the Ultimate Tap-Based Games Maker is a game-changer in the world of hyper casual game development. Its exceptional sprites, well-structured scripts, and comprehensive functionality make it the go-to asset for any developer looking to create their own engaging tap-based games. Dive into the world of game creation and unlock your creative potential with this extraordinary asset.

Technical Instructions:

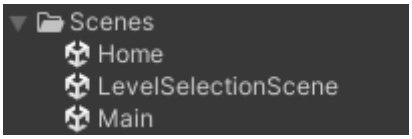
After importing the package, you will find a folder named “GameTemplate-UltimateTapGamesMaker” as shown in the above image. Inside that folder you can see some other folders as Documentation, Prefabs, Scenes, Scripts and Sprites.



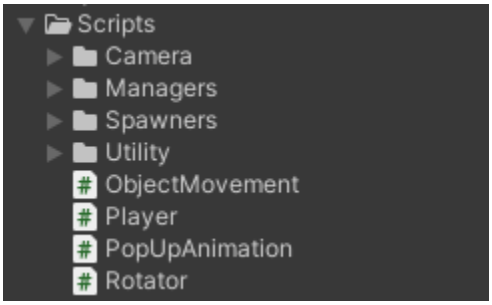
If you expand the Prefabs folder you will find all the prefabs that have been created and used in this package.



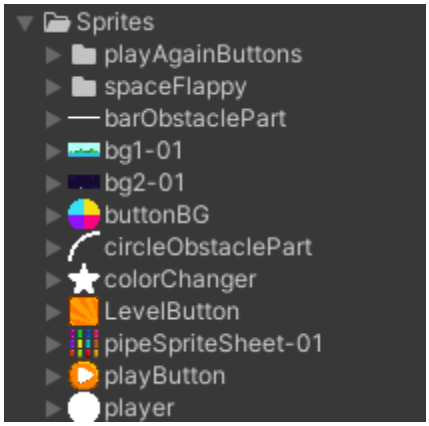
Inside the Scenes folder you can see three scenes as Home, LevelSelectionScene and Main scene. The Home scene is the introductory scene where you can have anything as your game's intro. One sample is provided in the Home scene already. The LevelSelectionScene is responsible for the Career Mode of your game. It shows which level you have unlocked and which levels are yet to be. At last, the Main scene is the scene where your actual game runs.



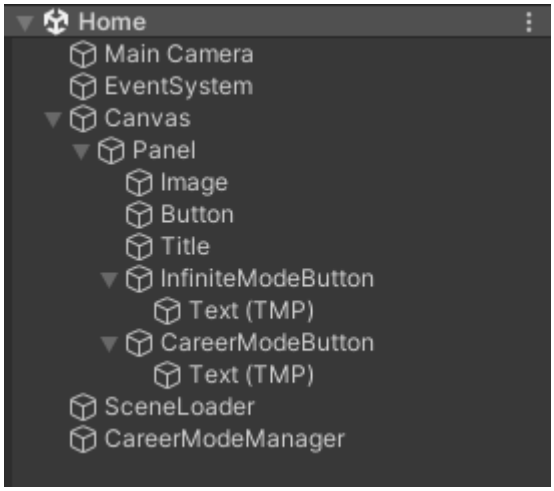
If you expand the Scripts folder, you will find folderized scripts used for specific tasks.



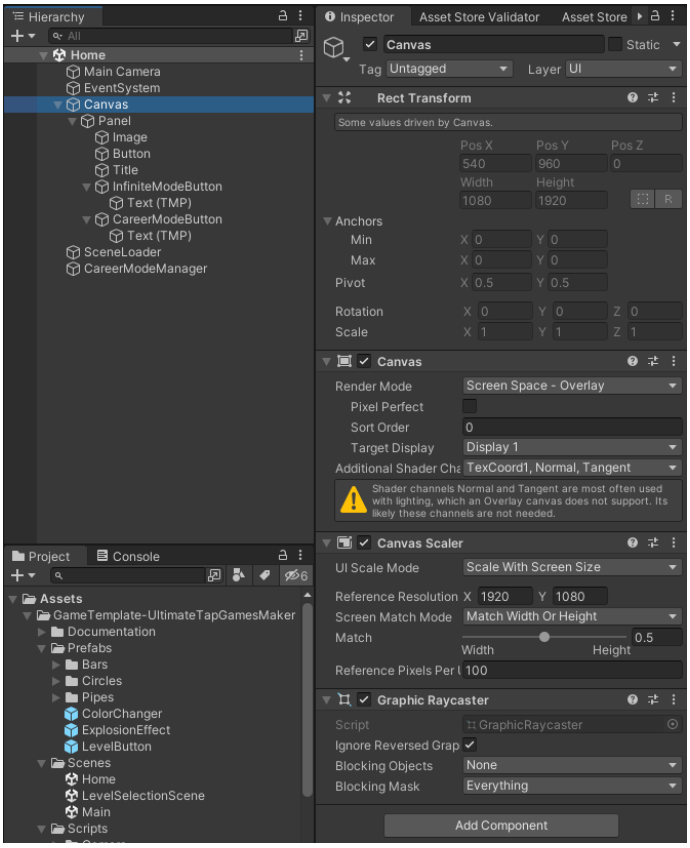
Inside the Sprites, folder you will find unique and high quality sprites created specially for this package. These sprites are created keeping in mind the games called "Color Switch" and "Flappy Bird". But the user of this package can simply change the sprite according to his own choice and have a totally different playable character.



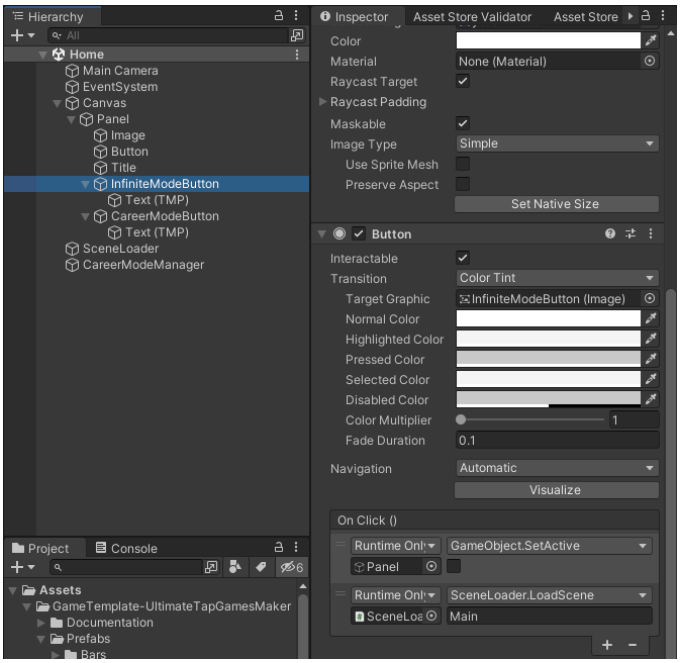
Time to get into the scenes now. Double click on the Home scene to open it. When you open it, you will find the hierarchy similar to something that has been shown below. You will have a Main Camera, a Canvas, a SceneLoader gameObject and a CareerModeManager gameObject.



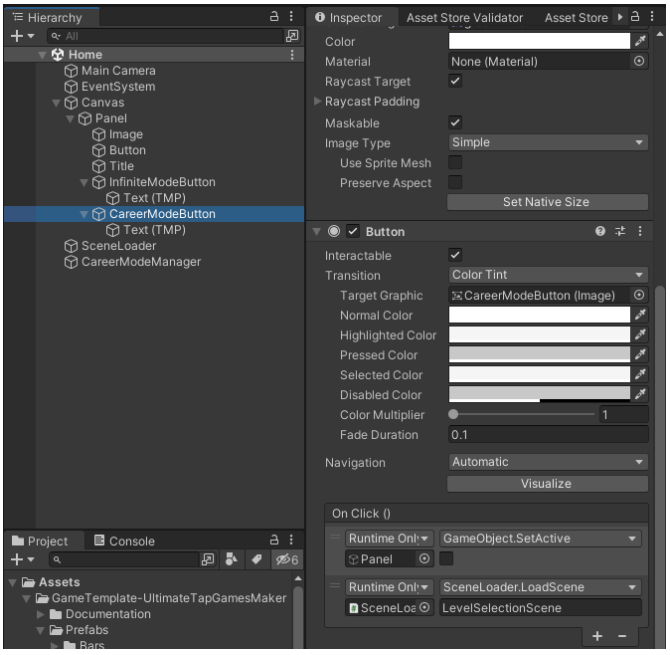
Select the Canvas and do following settings to have scalable game according to the different screen sizes.



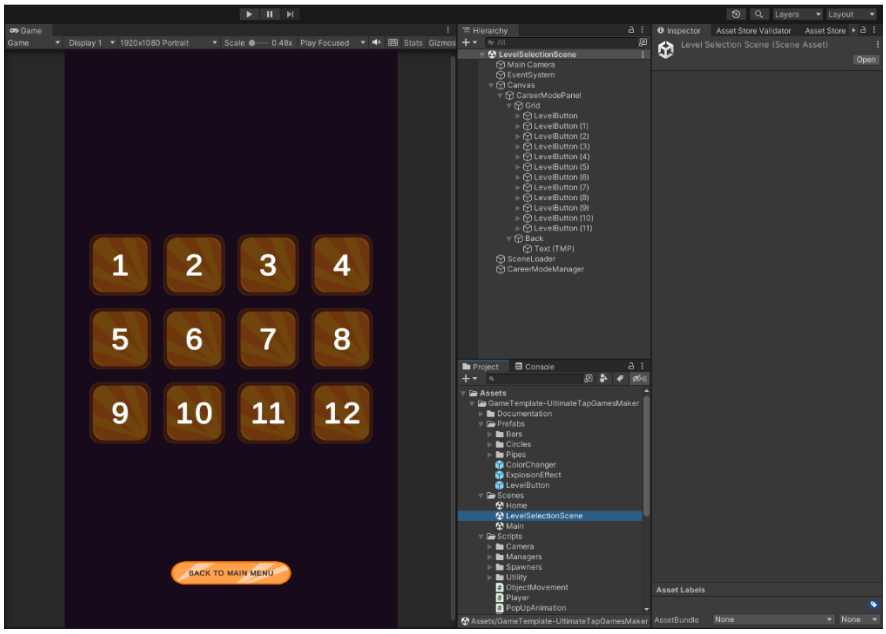
Expand the Panel section and choose the InfiniteModeButton and configure the button according to the following image. This button starts your game as an infinite runner game where player will try to beat his own previous personal best high score.



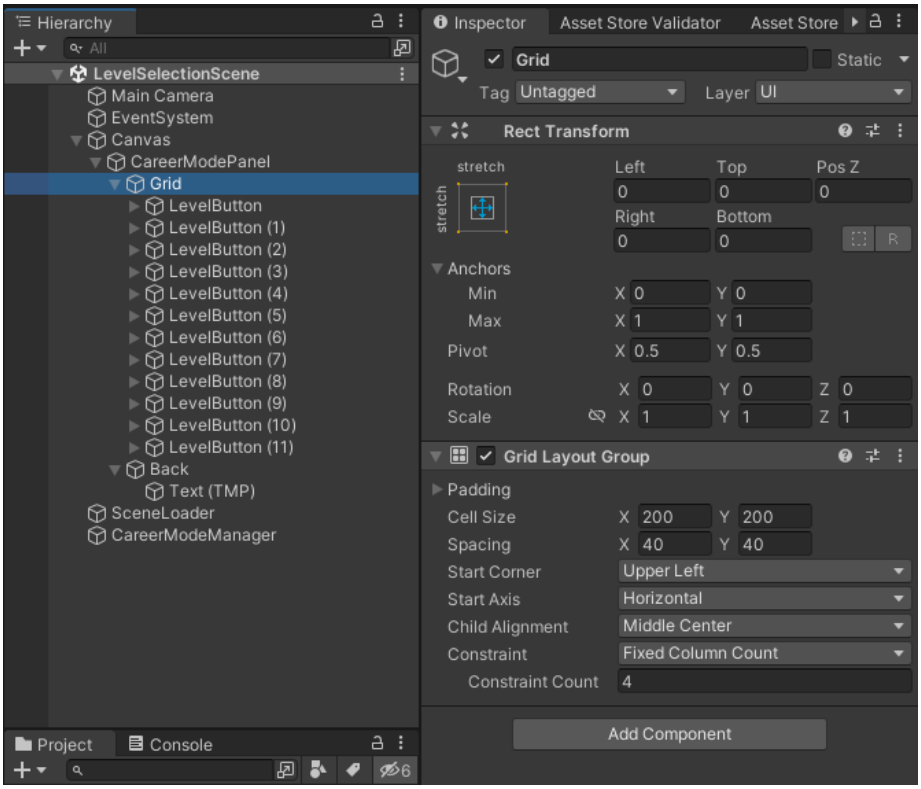
Choosing the CareerModeButton and configure it according to the image below will provide the player to access the Career Mode of your game. Where player will have locked stages that will require certain parameters to be unlocked. In this package, every stage gets unlocked depending on score. First stage is unlocked by default. If you score 5 score, the second stage gets unlocked. Third one gets unlocked if your score 10, then 15 then 20 and so on…I think you get the pattern? (Multiples of 5)



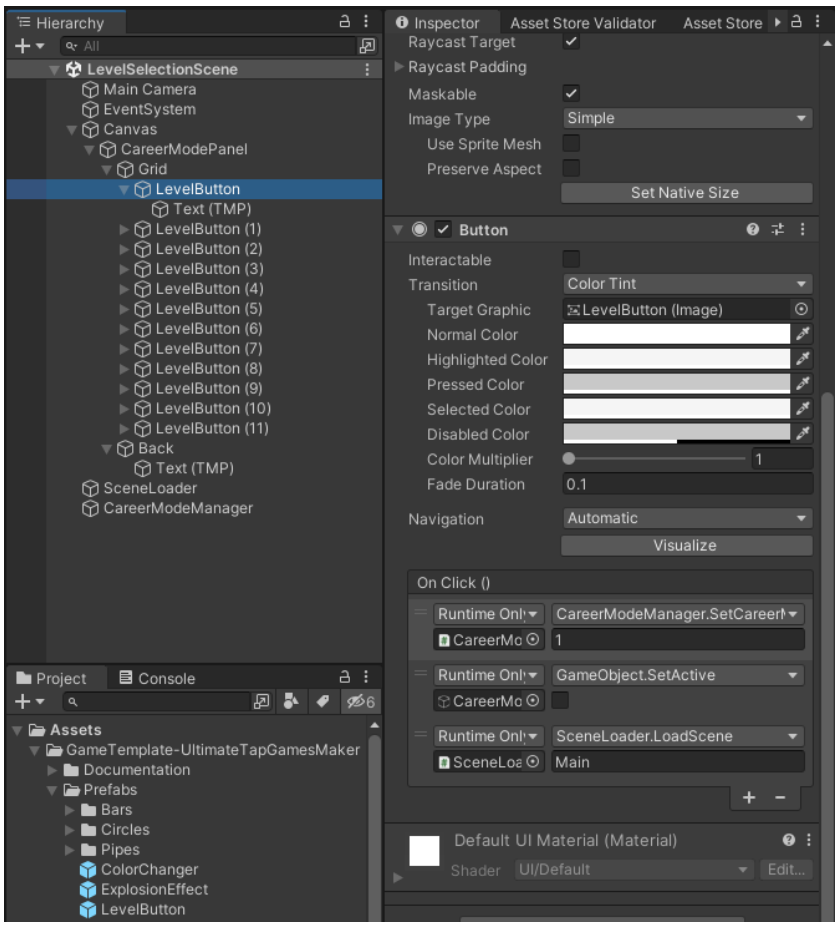
Now open the LevelSelectionScene. This is a very simple scene and the function of this scene is described already. After opening the scene, you will find a Canvas, expanding the canvas you will find a gameObejct called Grid. This Grid holds all the buttons that give you access to different stages. Currently I have 12 stages in my Grid. Also, there is a Back button, pressing which you will be directed to the “Home” scene.



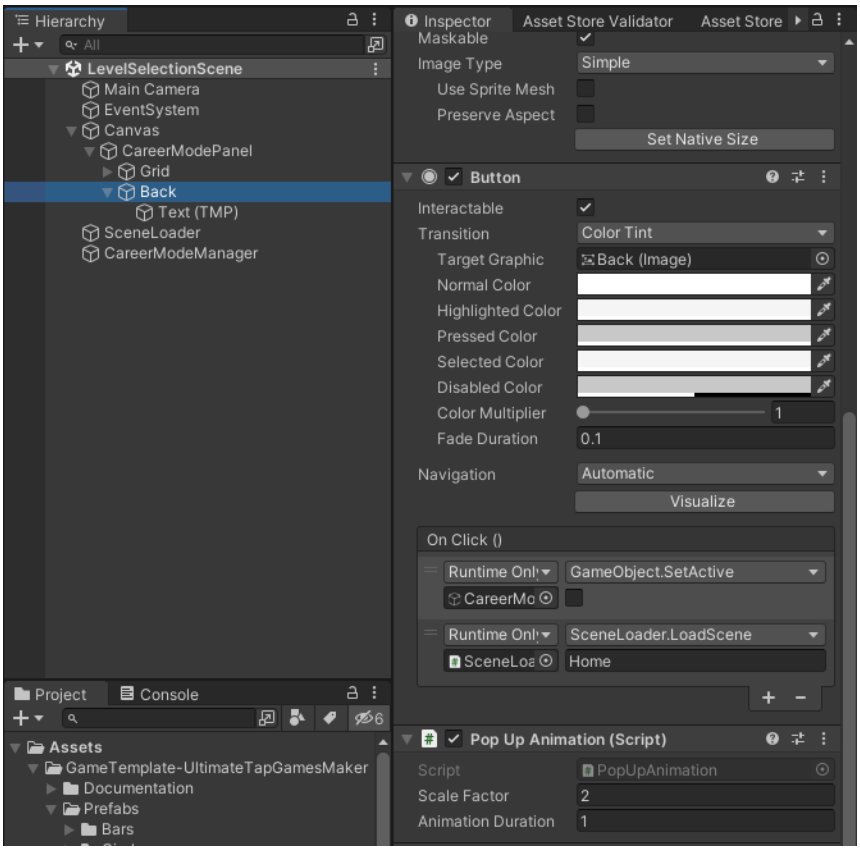
The Grid gameObject has a Grid Layout attached to it. The settings of the Grid Layout are shown below. The user of this package can modify the Grid layout as his/her choice or look of the game.



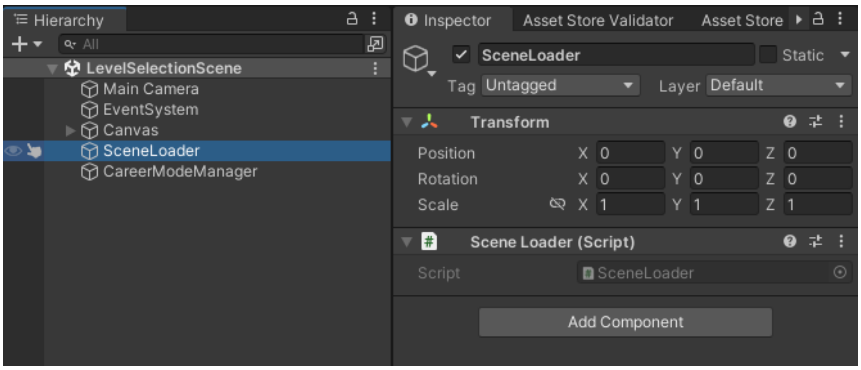
All the LevelButton has three functions to perform. First, set the CareerMode ON by passing an integer value as “1”. Second, set the visibility of the CareerModePanel of the current hierarchy to false. Last, from the SceneLoader choose the LoadScene method and pass the “Main” scene name as the parameter to start the game.



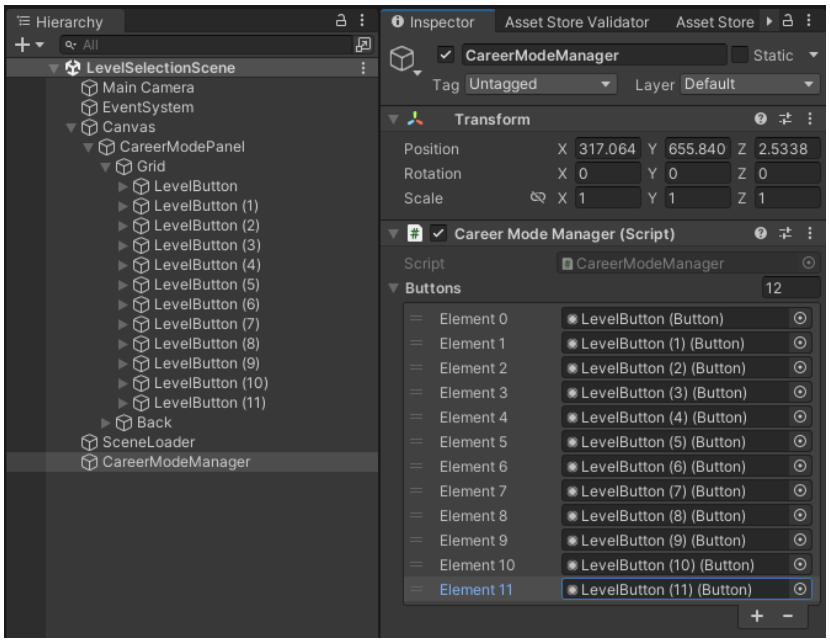
Next you will need to configure the Back button functionality just as shown in the following image. This button has two tasks. First, it turns off the CareerModePanel visibility. Second, it loads the “Home” scene.



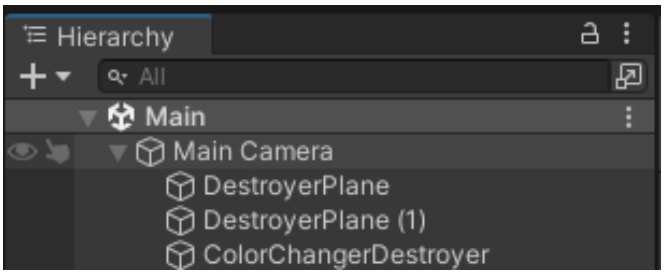
The SceneLoader gameObejct has a SceneLoader script attached to it. This script has a LoadScene Method which takes a string as the scene name and loads that particular scene(if available, to make a scene available you need to add the scenes to your active scene list from the project settings.)



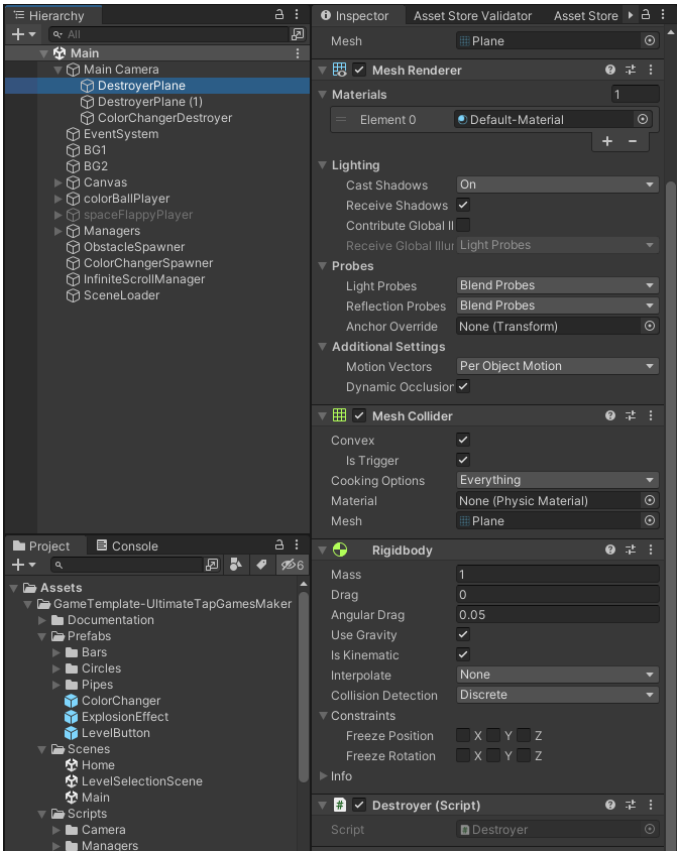
CareerModeManager will have a CareerModeManager script attached to it. This script handles the locked and unlocked state of the buttons/stages. So, you will need to have all the references of the buttons present in the Grid. The references are shown in the following image. Simply drag and drop the child of the Grid, and you are good to go.



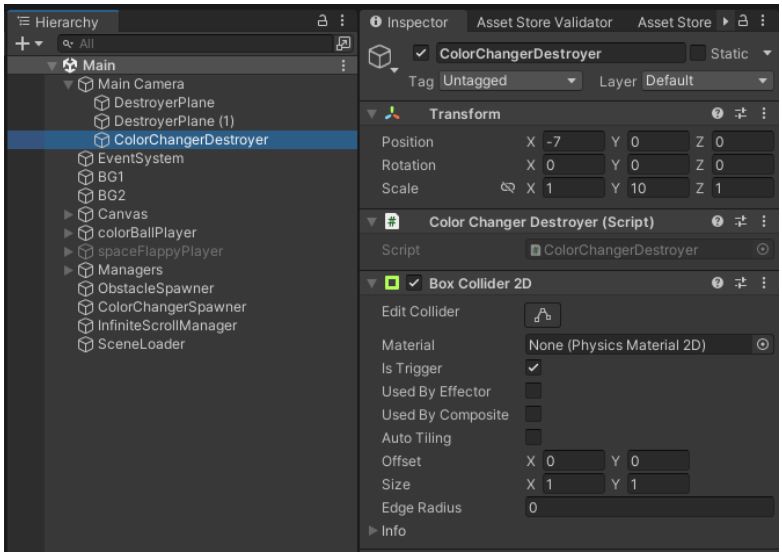
Now, its time to open the Main scene. In the Main scene, there is a Main Camera. The Main Camera has three children gameObjects shown in the following image. The DestroyerPlane and DestroyerPlane(1) are used to destroy anything that triggers with them. ColorChangerDestroyer destroys the ColorChanger gameObject(discussed later) when triggered with it.



DestoyerPlane and DestroyerPlane(1) has the following components shown in the next image. They have also a script called “Destroyer”

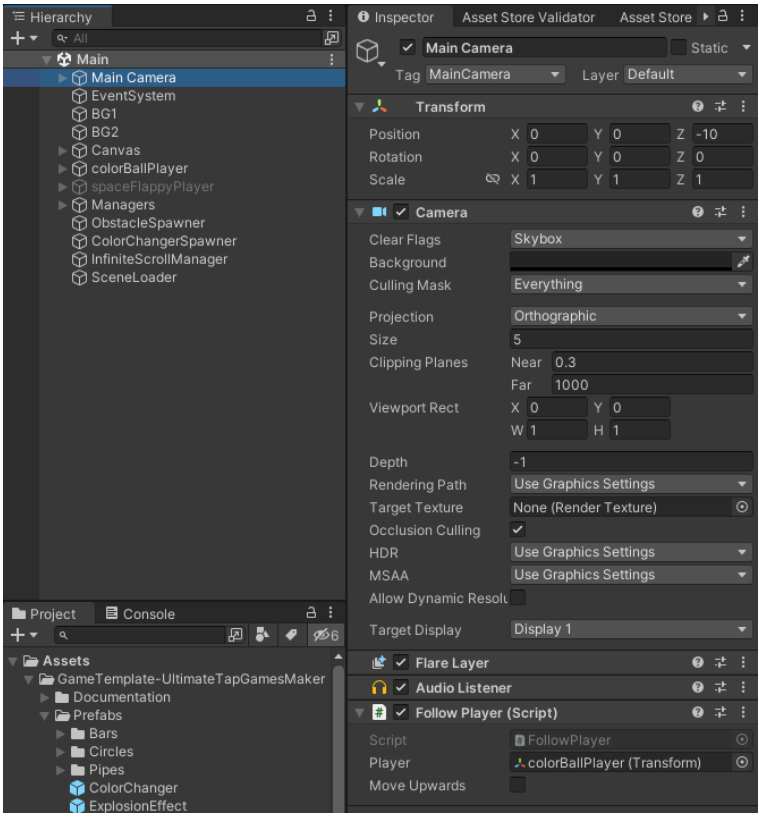


The ColorChangerDestroyer has the following components shown in the next image. It has a script attached to it called “ColorChangerDestroyer”.



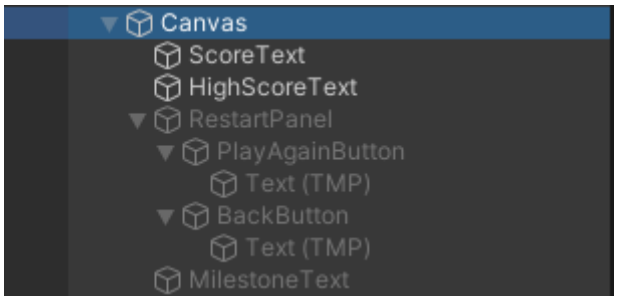
These Destroyer scripts has a OnTriggerEnter() Method which handles the destruction of obstacles, player and color changer when triggred.

Next choose the Main Camera. It has a script attached to it named “FollowPlayer”. It handles the camera movement. When the distance between the player and the camera passes a certain distance then the camera changes its position. This script has a Boolean named “moveUpwards” also shown in the editor. If this Boolean is true then the camera will move upwards or the camera will move horizontally. The script will have a player reference as well. Currently we are having “colorBallPlayer” as our player so we dragged and dropped the colorBallPlayer from the hierarchy.

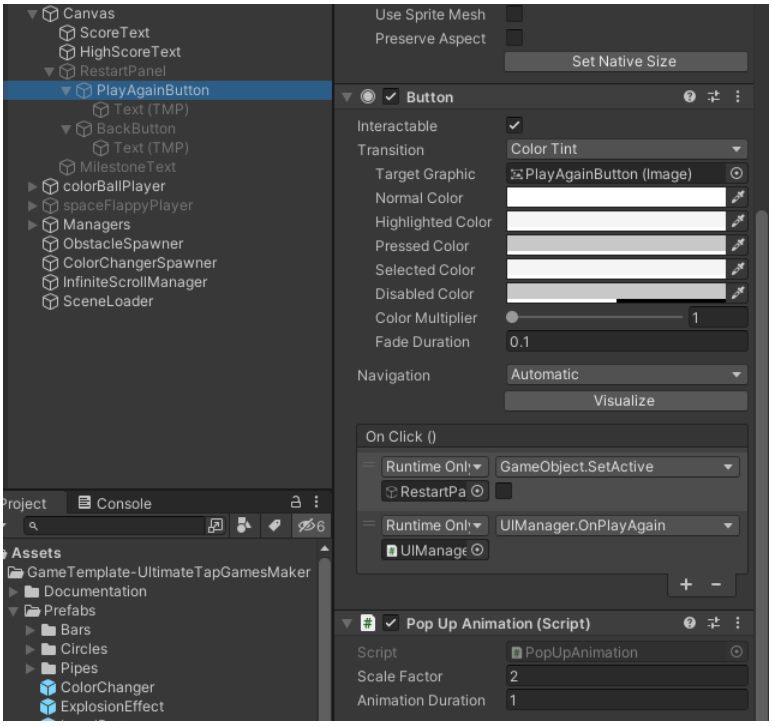


This scene has two gameObjects called BG1 and BG2. These are two gameObjects with Sprite Renderer component. This two gameObjects keep changing their position to create an infinite scrolling background effect (handled by InfiniteScrollManager shown in later part).

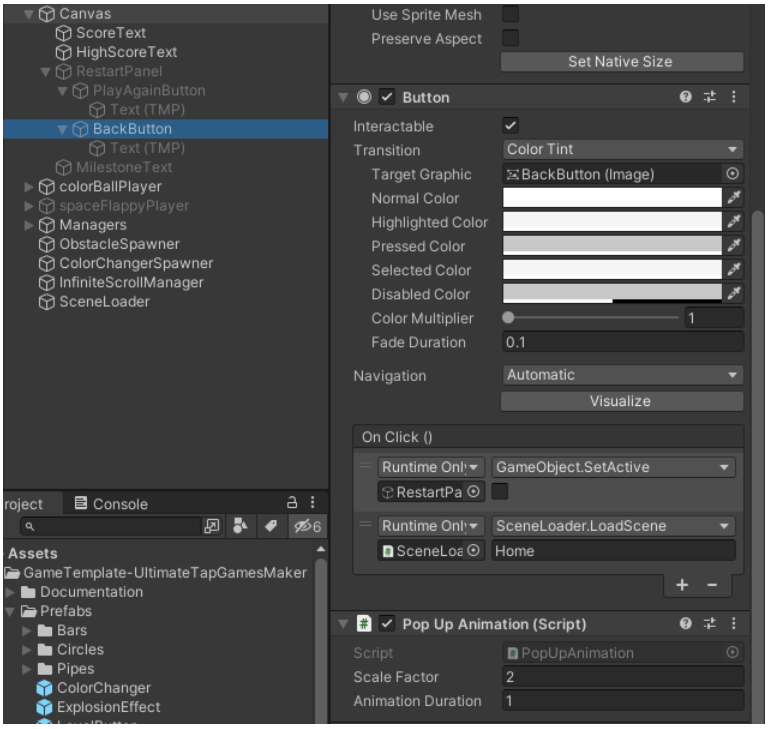
Open the Canvas. It has a ScoreText, HighScoreText and a RestartPanel. Expanding the RestartPanel you will have a PlayAgainButton, BackButton and a MilestoneText. ScoreText represents the current score your progressing game. HighScoreText represents the highest score of a user in the Infinite game mode. RestartPanel turns on when the player is dead or destroyed. The PlayAgainButton and BackButton also gets visible then. Clicking the PlayAgainButton will restart the current mode game and clicking the BackButton will lead the user to the “Home” scene. The MilestoneText represents the score a player needs to reach to unlock the next available stage in the career mode.



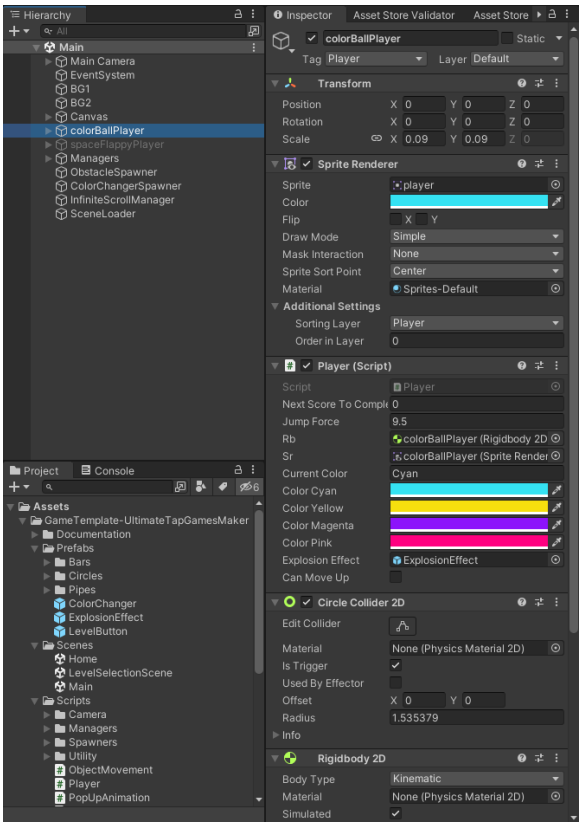
Here is the PlayAgainButton settings look like in the following image. It also has a script named PopUpAnimation. Using the DoTween package from asset store, the buttons are given a popup animation. You can handle the popup animation by simply tweaking the ScaleFactor value and AnimationDuration value.

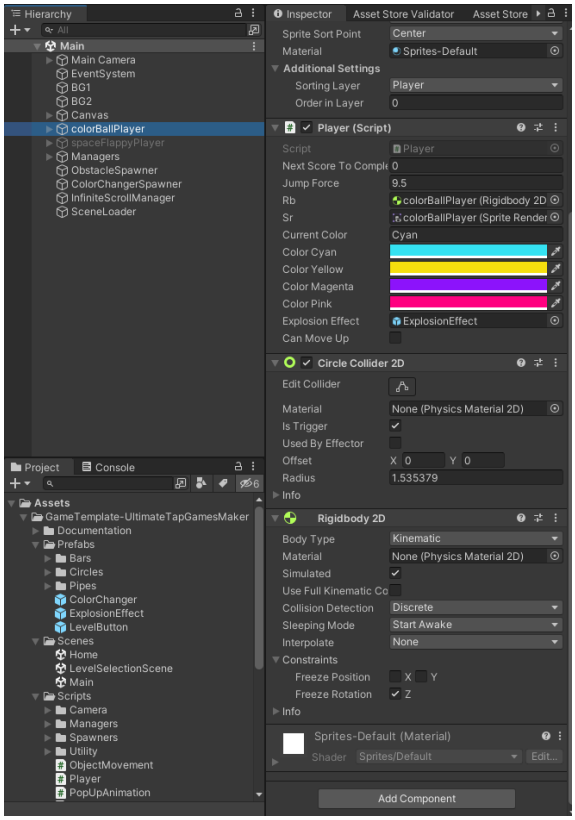


The BackButton has also similar kind of settings as shown below.

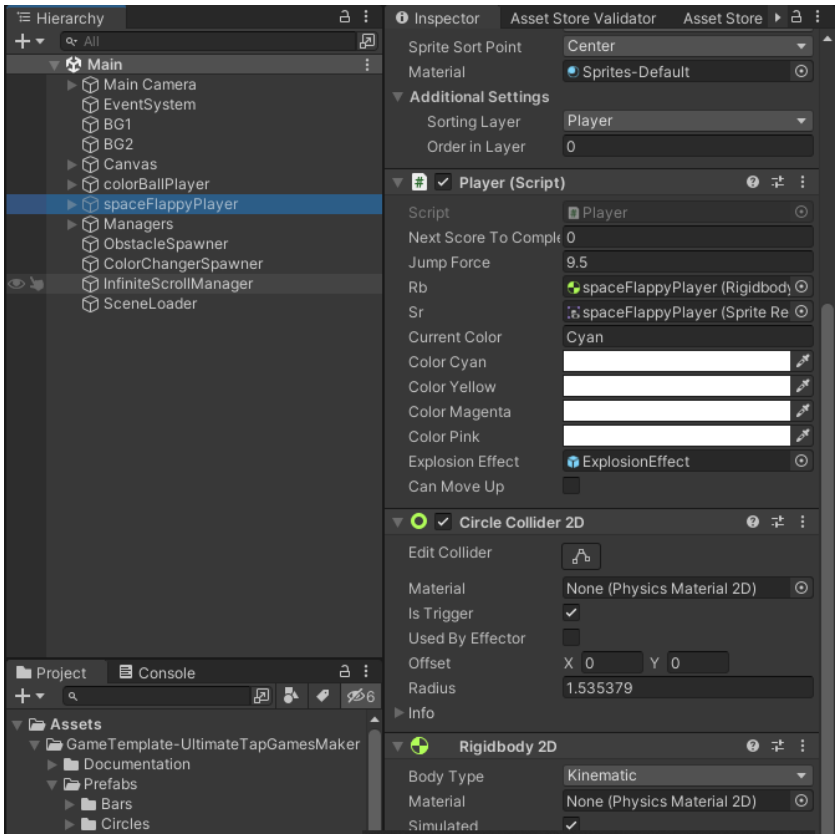


colorBallPlayer gameObject represents a ball player. It has a “Player” tag and a Sprite Renderer component. It has a Player script attached to it. Also, it has a Circle Collider 2D component and a RigidBody 2D component. The collider and rigidbody settings are shown below. Player script has a float value JumpForce, here used as (9.5f). It also has the reference for the rigidbody and sprite renderer of the player. Simply drag and drop the active player gameObject for the reference. Set the current color as Cyan as we have selected cyan as the color of the ball player inside the Sprite Renderer’s color. ColorCyan, ColorYellow, ColorMagenta and ColorPink are the colors that the player can itself into. You can change the colors from the editors as well by your choice. Explosion effect is a unity particle system created and saved as prefab in the Prefabs folder of the package. CanMoveUp is a Boolean which turns true when the game runs as a vertically moving game, turns false otherwise.

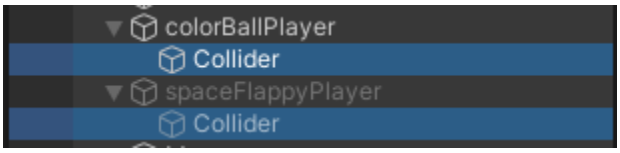




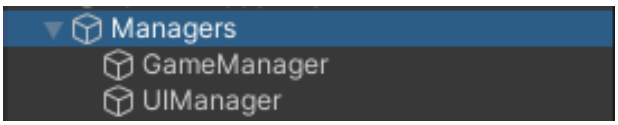
spaceFlappyPlayer is another version of our player. It is designed inspired from the Flappy Bird game. It also has the same Player settings as previous. Except we don’t wantour spaceFlappyPlayer to change its color, so we change all the available color to White as shown below.



Both colorBallPlayer and spaceFlappyPlayer has Colliders.

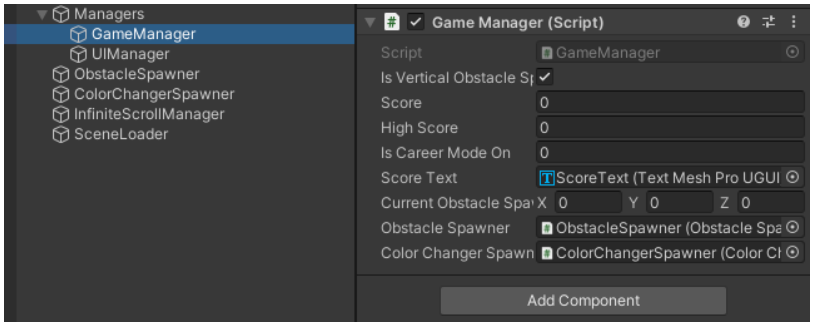


From the hierarchy you can see a “Managers” folder. Expanding that you can see a GameManager gameObject and a UIManager gameObject.

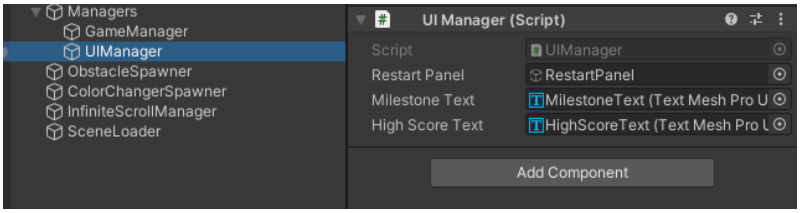


Select the GameManager and you will find the following settings shown in the next image. **This is the most important of this package. If you are a busy hypercasual game developer and want to create a tap based hypercasual game in the shortest time possible then pay attention to this part.**

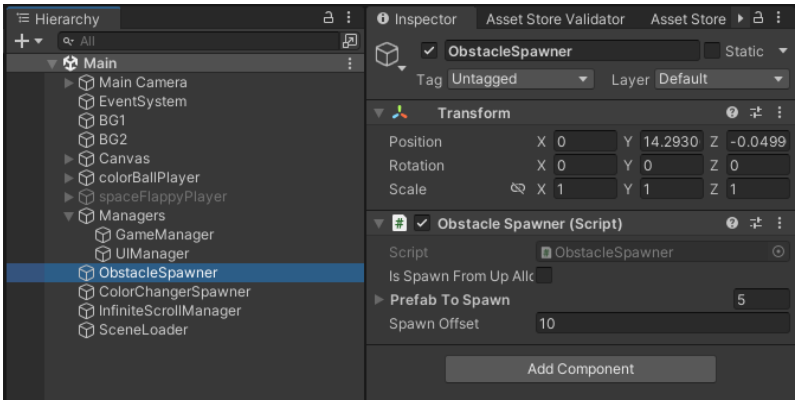
The GameManager script has a Boolean named IsVerticalObstacleSpawnAllowed. If the user of this package wants to create a Vertically moving game, then he/she needs to turn this ON. Or he/she should keep this turned OFF. Set the other settings as following. IsCareerModeOn = 0 represents that the game is currently ready to be played in the Infinite mode, IsCareerModeOn = 1 means the game is ready to be played in the career mode. This value is handled by Playerprefs in the GameManager script.



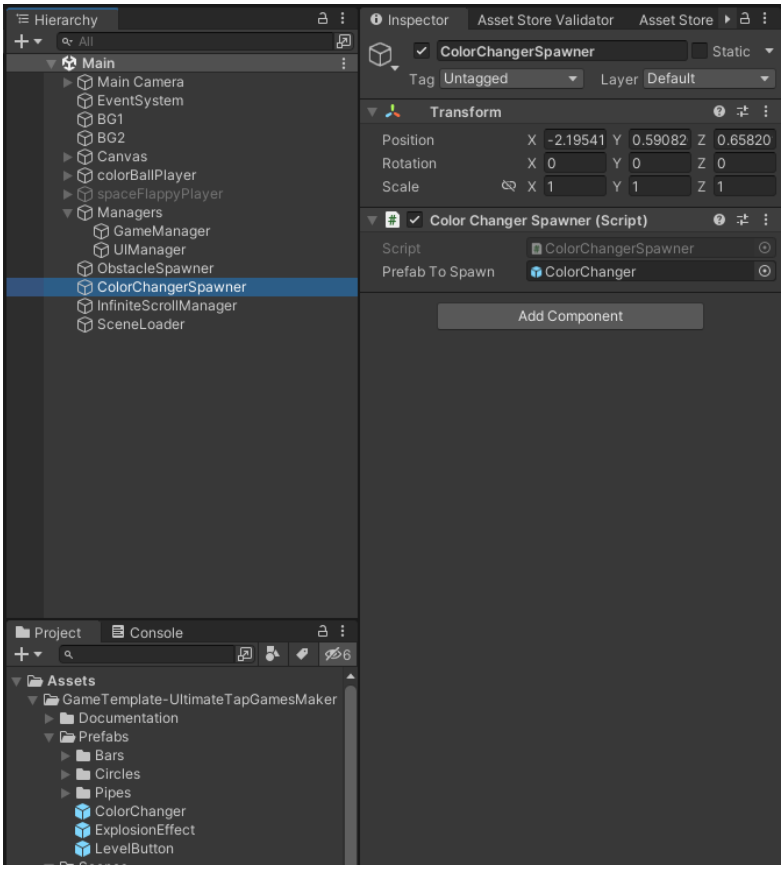
Here are the settings of the UIManager. It has reference of the RestartPanel, MilestoneText and HighScoreText.



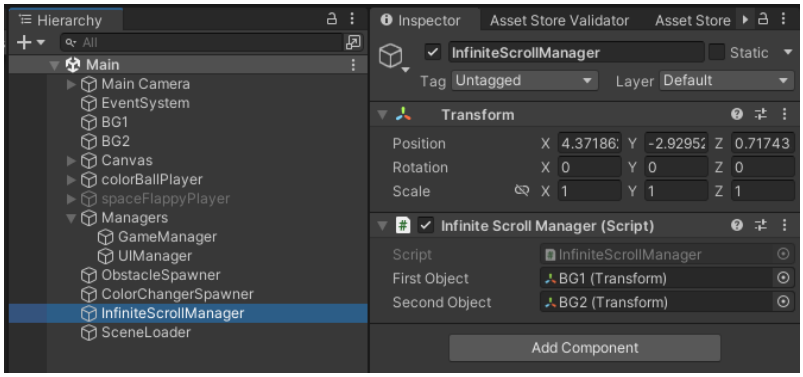
ObstaceSpawner settings are shown below. It has a script attached to it named ObstacleSpawner. It has a Boolean named IsSpawnFromUp and a list of obstacles. Spawn Offset is set to 10. This offset value represents the distance between two consecutive obstacles. The function of the Boolean is self - explanatory.



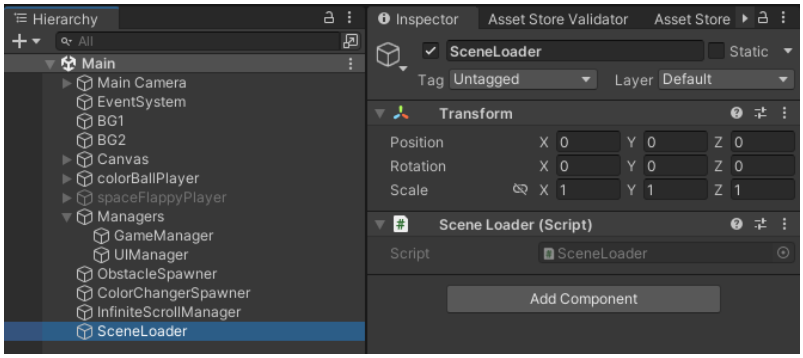
The ColorChangerSpawner just spawns a new ColorChanger whenever the current ColorChanger is destroyed.



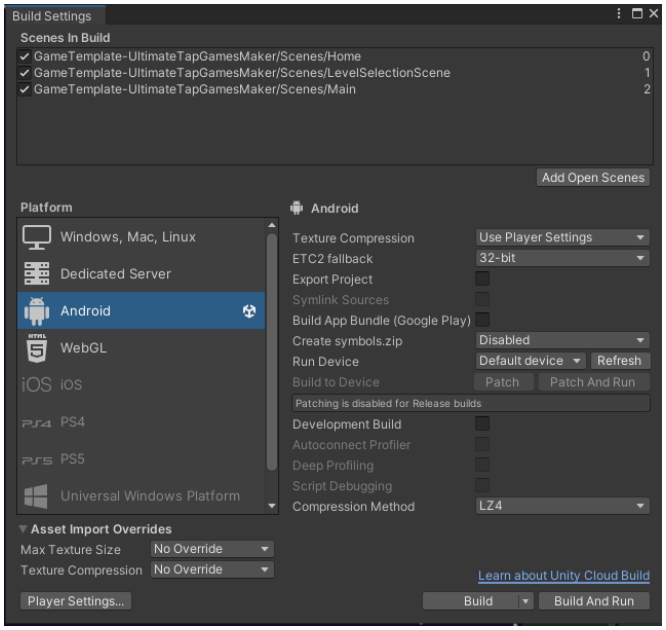
The InfiniteScrollManager is discussed earlier as well. The InfineScrollManager script takes references of the two background images (both should be same). We named the reference as firstObject and secondObject.



The SceneLoader is the same SceneLoader that we have been using in all the scenes.



Here are the Build settings. We have added all the Open Scenes in the list whenever we have opened a New Scene.



This is it. These are all that you need to know. If you are experienced in developing hyper casual games using unity then you might not even need to follow the instructions all along. But I have made this tutorial considering beginners.

Unity Version:

Unity version used in this package is **2021.3.26f1**.