Maintenance Guide: Jungli Game

Environment.

IDE: Unity 3D, minimum version 2017.3.0.

Scripting language: C#

Structure of the program.

Program is divided to scenes to make development easier. Scenes and scripts related to them are stored inside assets folder. Scenes are following:

Menu:

Inquires for player's name. Pressing *Go* -button runs the *toStoryScene* script. This script saves given name to *PlayerLogic* and moves program to next scene.

JangliStory:

Shows the story of the game to player. Story is stored in *Player Logic* script, and player's name gets filled to it. Pressing button on the screen runs *toJangliGame* script. Script moves program to actual game.

Level1:

Level 1 consists of swinging vines and floating tiles. Player must jump and grab the vines to proceed in the game. Vines are implemented as *swinging vines* inside *VineController.* Parameters of vines can be modified trough *VineController's Vines* script. (for example make them longer, closer to each other etc.)

PlayerL1 script inside Player -object conceal logic of character's movement and collisions in this level. If new objects were to be added to the level, interaction with the player can be implemented here or in new scripts defining new object's behaviour.

Music can be added or changed trough *Audio Source.*

Level2:

Level 2 consists of water, crocodiles and bubbles rising from the bottom of the river. Crocodiles are controlled by *CrocodileController* and bubbles via *BubbleController.* Controllers contain scripts for spawning objects to the scene. Again, by modifying controller's parameters developer can, for example, make objects spawn more frequently.

Bubble and *Crocodile* scripts control behaviour of these objects colliding the player inside this level.

PlayerL2 script contains character's operations needed for moving it. Whenever player rises to the surface, script invokes *AirMeter's* Breath() method.

OxygenSlider represents the air meter on the scene. It is controlled by *AirMeter* script. Amount of oxygen can be altered trough this script.

Music can be added or changed trough *Audio Source.*

Level3:

Third scene is an uphill level, where player must dodge rolling stones.

StoneController script spawns *RollingStone* objects to the level.

PlayerL3 script controls character's movement. *PlatformController* spawns new platforms as the player moves. *StoneController* spawns new stones to the scene. Again, all relevant parameters can be altered trough these scripts.

Music can be added or changed trough *Audio Source.*

Level4:

Fourth scene is a level full of cannibals (5). Player needs to dodge cannibals to save his wife and beat the game. Cannibal's movement behaviour is defined in *CannibalMovement* script. Player's behaviour and inputs are defined in *PlayerL4* script, which is attached to Player object. Parameters can be modified trough Player object.

EndScreen:

Shows player's final score, and contains a button, which directs back to *Menu* screen.

GameOver:

Whenever player runs out of lives, program is moved to this screen. Scene contains a button, which directs back to menu.

Modifying program.

Adding components is done by creating an in game object to chosen scene, and adding a script which defines new object's behaviour. At least following aspects should be considered:

- Collision with ground.
- Collision with player.
- Collision with other in-game objects.

If new levels (or scenes) were to be implemented, they should be written in their own folders for clarity. Code contains spritesheets for the player character. Player's score and health are stored to PlayerLogic script (PlayerLogic.PlayerLives, PlayerLogic.PlayerScore). These parameters need to be used to keep track of scores and health. Program's behaviour when lives run out need to be defined in the new scene.

Movement between levels is done via SceneLoader.MoveToScene() command. Modifications need to be done to previous level's script, and next scene needs to be defined as well. Whenever player's lives run to zero, SceneLoader.LoadScene("GameOver") should be called. If new level is a final level, next scene should be "EndScreen".

If existing scenes were to be edited, main functionality lies in *PlayerLX* and *ObjectController* scripts. In-game object's locations and spawning frequencies can be altered trough *Controller* scripts. Each Object has their own script defining their movement, collisions etc. Player scripts contain logic to move character, and other level-specific funtionality.