Neler93

2D Action Platformer Game System

FAQ

version 1.1

CONTENTS

01. Introduction..... 3
02. Project structure..... 3
03. Scene structure..... 4
04. Set up your scene..... 4
05. Setup characters..... 5
    05.a Player..... 5
    05.b Enemy..... 9
06. Interactible Objects..... 10
07. Inventory and Items..... 10
08. Input and Example..... 12
09. Final word..... 12

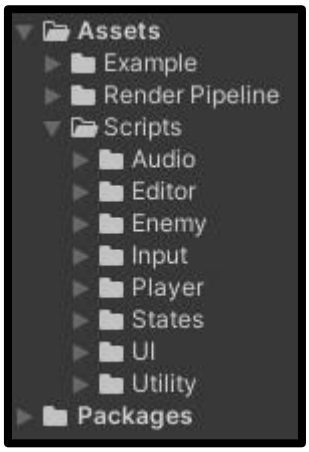
01. Introduction

This asset is a customizable framework for 2D action oriented platform games. It allows users with ease to create a fully functional action-oriented platform game and offers many customization options for a better suited experience.

02. Project structure

Project structure is very simple. Down below you can see that project structure is as follows:

- Example folder (contains example how to use asset with preset player, enemy, sounds etc.)
- Render Pipeline (contains custom 2D pipeline)
- Scripts (the backbone of this asset containing all solutions for the game system)



Folder structure

03. Scene structure

Down below you can see the most important parts of this asses regarding scene structure. Every prefab that is vital for usage with this asset has "[2DAPGS]" in its name. In picture bellow you can see Main Camera, Sound Manager, UI, Player and Enemy prefabs.

Scene Structure

04. Set up your scene

When creating your scene using this asset take in account that it is made for 2D environment specifically. To use this asset in your scene you need to place these specific prefabs that are inside Prefab folder of asset:

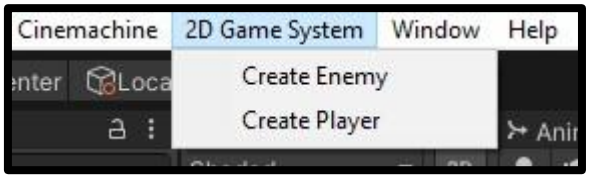
- "[2DAPGS] MainCamera" – prefab of the main camera using Cinemachine to follow player character

- “[2DAPGS] SoundManager” – prefab containing scripts with logic that asset uses to play sounds
- “[2DAPGS] UI” – prefab containing health bar for our player character, black image for fade in/out.

When you place all these prefabs in your scene you are good to go. You also need to have environment for your characters to move unto. Those environments used as ground must be set to “Ground” layer. Also do not forget to setup colliders for your ground. If you have surface in which you want to let your players to swim in you must set its tag and layer to “Water”.

05. Setup characters

To setup your player character you can select “Create Player” from “2D Game System” menu from top bar of Unity 3D or you can place in your scene “Player” prefab from Example/Prefabs folder.

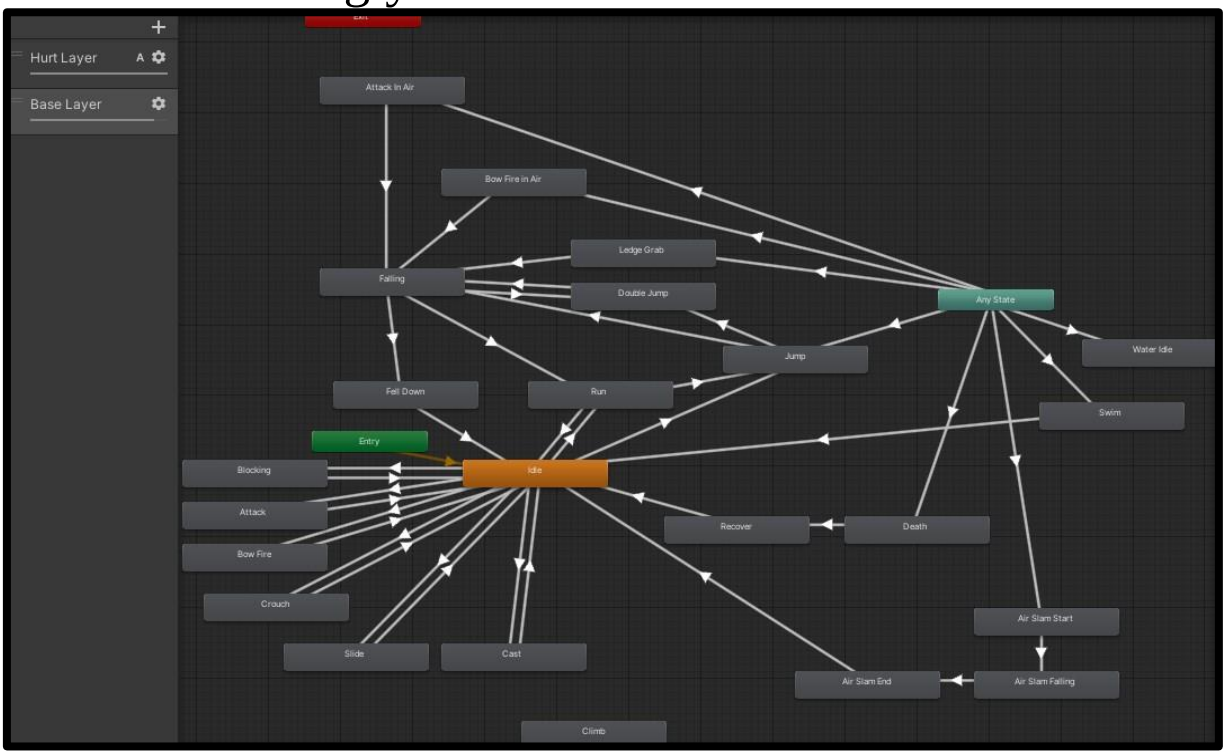


Top bar menu

05.a Player

If you have created player from top bar menu you should have a new game object in your scene that will be automatically selected, and editor camera will be centered on it.

You need to make animations for your player character. To do so it is easiest to just copy animator and animations from Example/Animations/Player folder and edit it to contain your animations. You can create your completely new animations to use them in copied animator but take notice that certain animations contain certain animation events so if you want to create completely new animations look at old animation for Example to set those events accordingly.

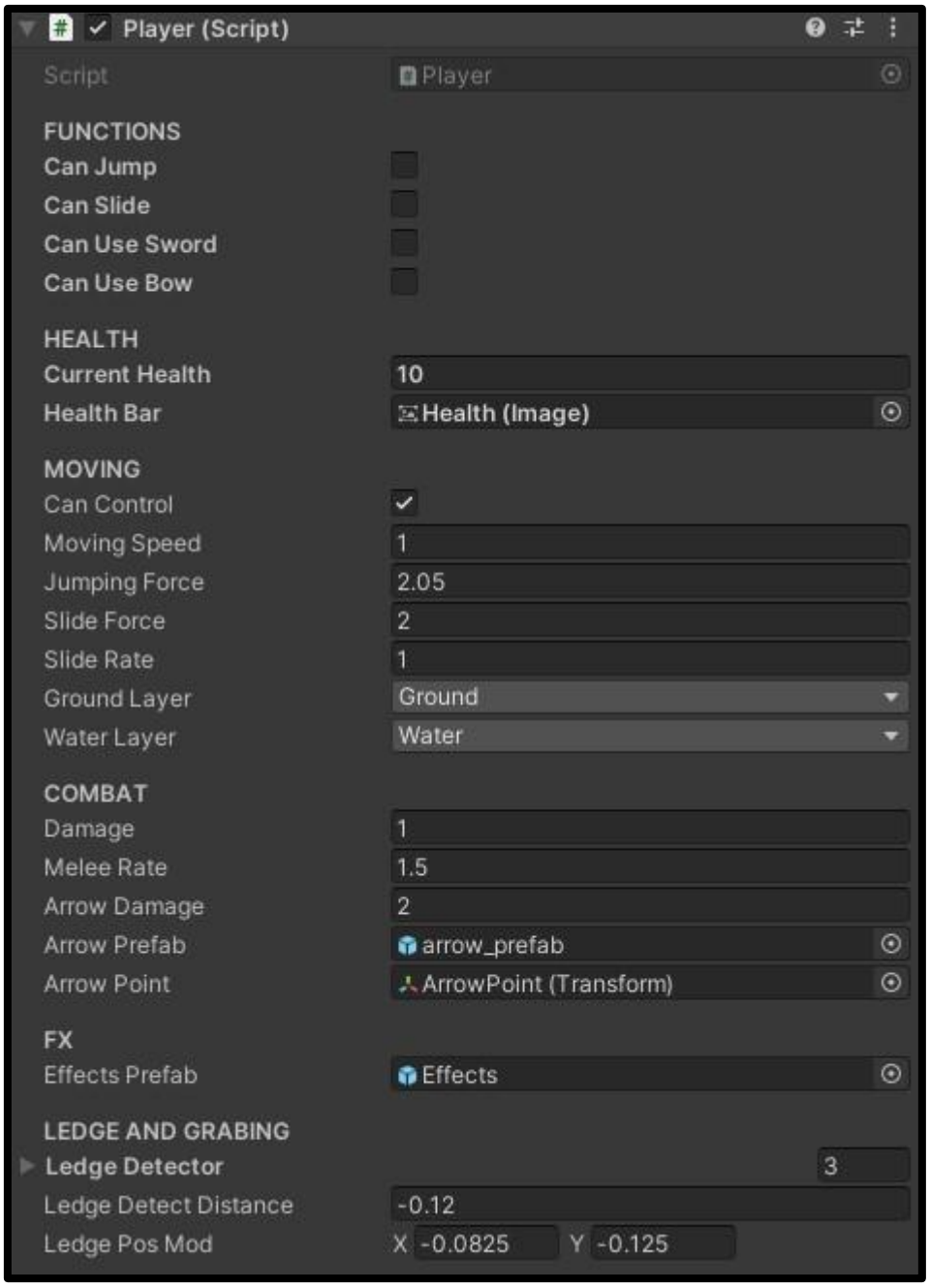


Animator and animations

If you look into animator, you will notice that there is Base Layer and above it Hurt Layer. Hurt layer is used to flash character (player or enemy) when it is hurt and that is why Hurt Layer is above Base Layer.

You can have look into all components that are present on newly created player game object and fiddle with

them around, but most important one is Player.cs. You can select what functions player can use in above part of the scripts, down below you can assign how much health does player have, and it will represent players maximum health at the start of the scene. You also must assign Health bar image in Health Bar field that will represent players health inside UI. After that you can select if player is controllable and his moving speed, jumping force, sliding force and rate, and Ground and Water layer. Also, you can choose how much damage will he deal and speed of melee attack and ranged attack. Down below you have ledge detecting variables to modify to your needs.



Take notice that you can call many functions from Player.cs. You can call these functions:

- EnableSlide()
- DisableSlide()
- EnableJump()
- DisableJump()

- EnableSword()
- DisableSword()
- EnableBow()
- DisableBow()
- SetPlayerEnableDisable(0 for disable or 1 for enable)
- HealthSet(amount)
- HealthAdd(amount)
- HealthRemove(amount)

By calling some of those function you can easily make objects like potions and other health regens with on collision call that will set/add/remove certain amount of players health and so on.

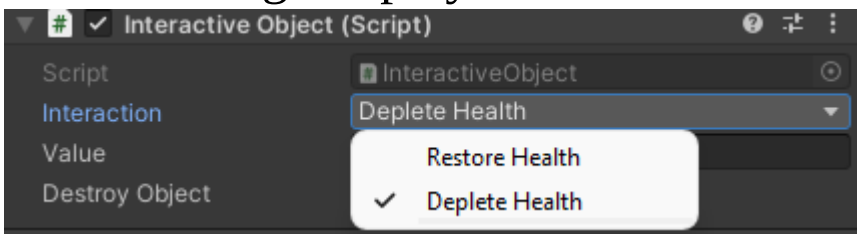
05.b Enemy

When you want to create enemy, you can also select “Create Enemy” from top bar menu “2D Game System” or you can copy and modify enemy prefabs from Example. If you create enemy from top bar menu, take notice that it would be best advised to look at Example folder so that you setup your enemy properly (setting its world UI etc.)

Enemy does not contain all the same components as Player but most important one is Enemy.cs. Variables are very much like those in Player.cs with difference that in Enemy.cs you must set sounds for various states that Enemy will use when playing his animations.

06. Interactible Objects

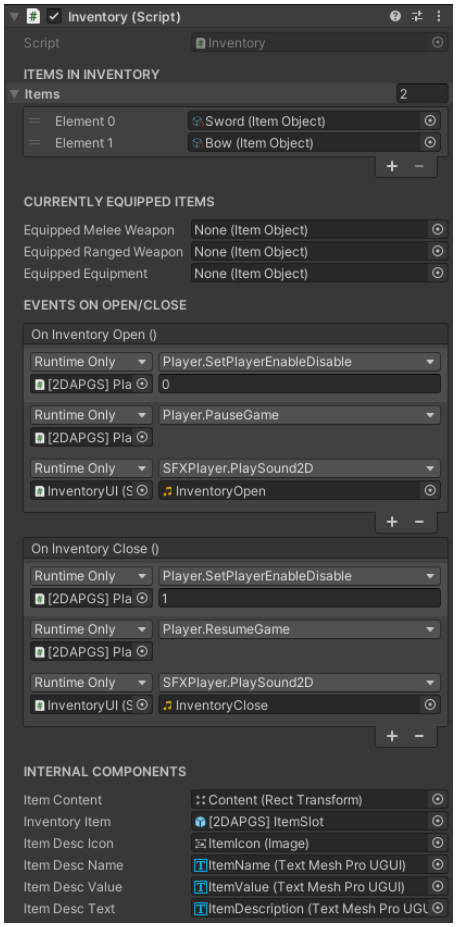
You can add interactable objects to your scene that can remove or add to health of player. Inside Demo scene you can find two kinds of interactable objects, trap, and potion. Trap sprite has animation and, on every Trigger Enter call, when players objects collide with it, it will inflict damage to player.



Also, as in Potion example in Demo scene, we can set it so that health is restored.

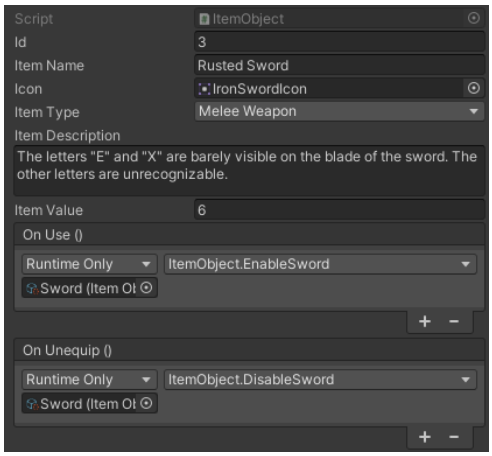
07. Inventory and Items

Inventory is big improvement of the asset. It allows for player to collect and store items and to equip them. This system allows us to control what player can do (melee/ranged attack, jump, slide etc.) when certain item is equipped. Inside Demo scene we can find few examples. While using Inventory System we do not need to use Interactive Object component, because we can set potion with restoration abilities to work with our inventory. Inventory is added to Scenes main Canvas, and it holds everything needed for the system to work. You can add it to your current project by dragging “InventoryUI” prefab from Prefabs folder into your



existing scene and adding it as a child object to your main canvas. Items List at the top of the component shows us what items our player currently has. If we want to start scene with some items already in our inventory, we can add them here. Equipped Melee Weapon/ Equipped Ranged Weapon/ Equipped Equipment shows us what has our player currently equipped from items in his inventory. Then we have On Inventory Open and On Inventory Close calls where we can set what will happen when Inventory is Opened/Closed. Lastly, we have 'Internal Components' part where everything for functional system is already set for us.

For system to work as intended we need Items. Items can be created by right clicking in your projects folder and going "Create/2D Game System/Create New Item". After then you need to set your item. ID of every item currently is not used by the asset, but in the future update it will be used for saving your game.



08. Input and Example

This asset uses Unity Input system. It is set to use inputs from keyboard and mouse, Xbox controllers and PlayStation controllers. If you investigate Example folder in Demo Scene you can see, how can you setup various on-screen notifications that is controlled by input. (In demo scene tutorial messages changes according to input it detects).

Inside Example you can see how you can fully use this asset and how easily you can make anything happen by using components and scripts that this asset provides.

09. Final word

I hope that you will have fun with this asset as much as I had making it. If you notice any bug, if you need support or have any question for me, be free to contact me at nenadradojcic@yahoo.com or to join my Discord server.

Have fun and all best to you!