

Dev - Basics 1

Minor Game Design &
Development



HOGESCHOOL ROTTERDAM



Dev Class Intro

- Unity, C#
- Focus on 2D, then 3D
- Github, Discord



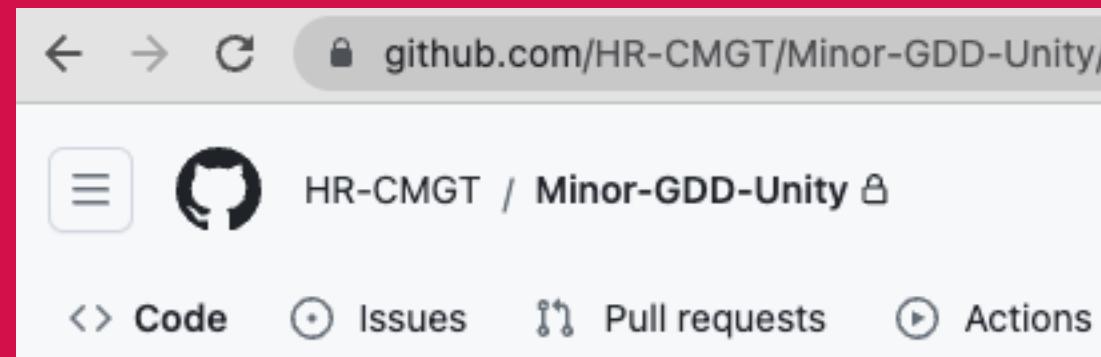
Part 1

- Github MGDD
- Github Documentation
- Unity Startproject
- Unity Package Manager
- Assignment (StartProject)
- Rigidbody2D - Component
- Rigidbody2D - Scripting
- Unity Documentation
- Unity Learn
- Unity's Script Lifecycle



Github MGDD

<https://github.com/HR-CMGT/Minor-GDD-Unity/>



Github Documentation

Resources

Tutorials

Project Files

Unity Tips & Best Practices

Code Snippets

Basics 1

[Presentation](#) - [Project Files](#) - [Resources](#) - [Tutorials](#) - [Assignment](#)

Presentation

This week's [presentation](#) can be found [here](#)

Resources

- Our own [tips, tricks and best practices](#) for working with Unity, with a bunch of gifs
- A list of [external tutorials](#) to help you with specific topics, from learning the basics to creating a certain effect.
- Get graphics, sounds, code and other free stuff from the [resources](#) page

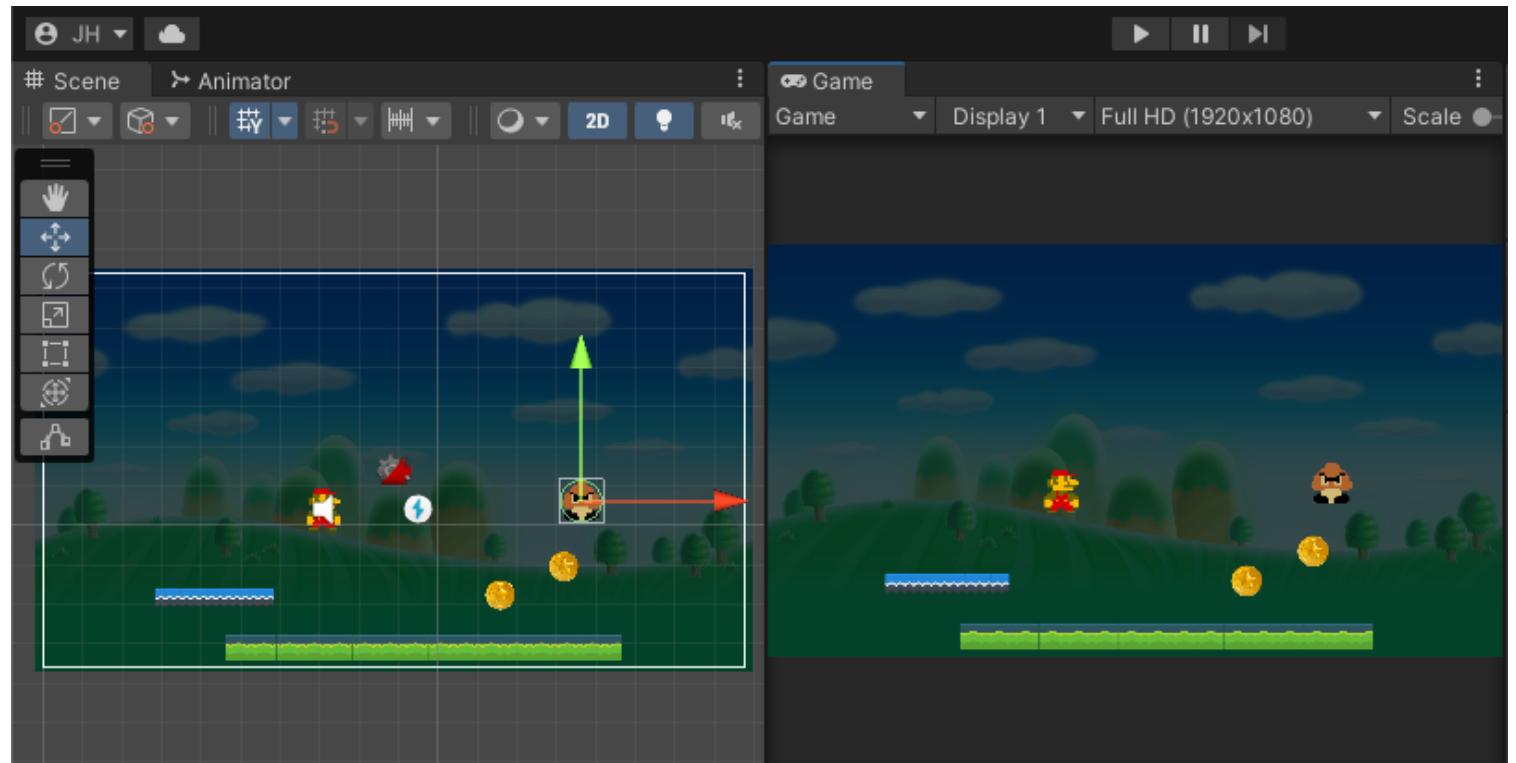
Assignment

1. Download and open Unity
2. Create a new 2D project



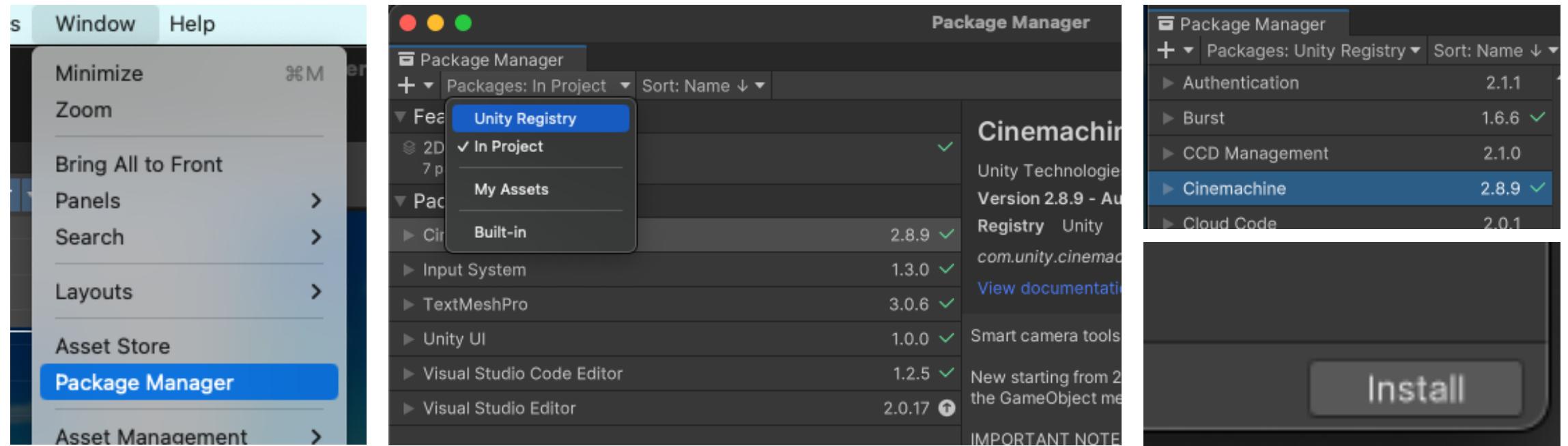
Unity Startproject

UnityPackage
[[Github link](#)]



Unity Package Manager

Open Package Manager, select Unity Registry, install Cinemachine & Input System



Assignment (StartProject)

Camera Follow

Configure the Player

Programming the Player

Jump

Move

ResetPlayer

OnCollisionEnter2D

CollectCoin

Jump Sprite

Add Coins

Play Around



Rigidbody2D - Component

Note: Requires a Collider2D to work!

Body Types:

Dynamic / Kinematic / Static

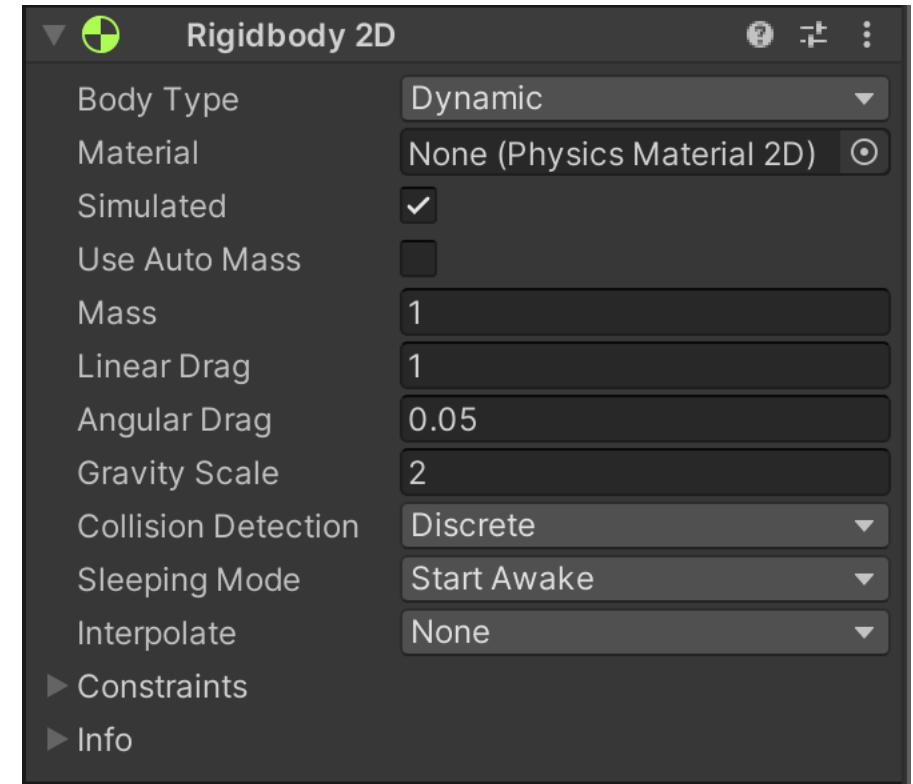
Constraints:

Freeze Position X/Y, Freeze Rotation Z

Material:

PhysicsMaterial2D, friction and bounciness

Mass, Gravity Scale and Linear Drag ('air-friction') settings for "weight"



Rigidbody2D - Scripting

Frequently used properties and functions of Rigidbody2D

Tip: use AddForce [[script ref](#)] for moving a dynamic Rigidbody2D, and use a big

```
rb = GetComponent<Rigidbody2D>();

rb.velocity = Vector2.right; // set the velocity directly, ignore mass
rb.AddForce(Vector2.right); // push the object, include mass
rb.isKinematic = true; // "freeze" the object
rb.MovePosition(new Vector2(0, 0)); // set the world position directly
```

[Unity Script Reference - Rigidbody2D](#)



Unity Documentation

1. Unity Manual

“Tutorials” on how Unity’s systems work

The screenshot shows the Unity Manual interface. At the top, there's a navigation bar with the Unity logo, "Manual", "Scripting API", a search bar, and a language selector set to "English". Below the bar, the version is listed as "Version: 2022.3". The main content area has a sidebar on the left with a "Unity Manual" section containing links to various topics like User Manual, LTS, and Physics 2D Reference. The main content area displays the "Introduction to Rigidbody 2D" page, which includes a breadcrumb trail ("Unity User Manual 2022.3 (LTS) / 2D game development / Physics 2D Reference / Rigidbody 2D / Introduction to Rigidbody 2D"), back and forward navigation buttons, and a "SWITCH TO SCRIPTING" button. The page content explains what a Rigidbody 2D component does and how it differs from its 3D counterpart.

2. Unity Script Reference

Overview of Scripting API

The screenshot shows the Unity Script Reference interface. At the top, there's a navigation bar with the Unity logo, "Manual", "Scripting API", a search bar, and a language selector set to "C#". Below the bar, the version is listed as "Version: 2022.3". The main content area displays the "Rigidbody2D" class page. On the left, there's a sidebar with a list of Unity classes. The main content area includes a "Description" section with a brief overview of the Rigidbody2D class, a "See Also" section with links to related classes, and a "Properties" section with a table showing the "angularDrag" property. A "Leave feedback" link is also present.



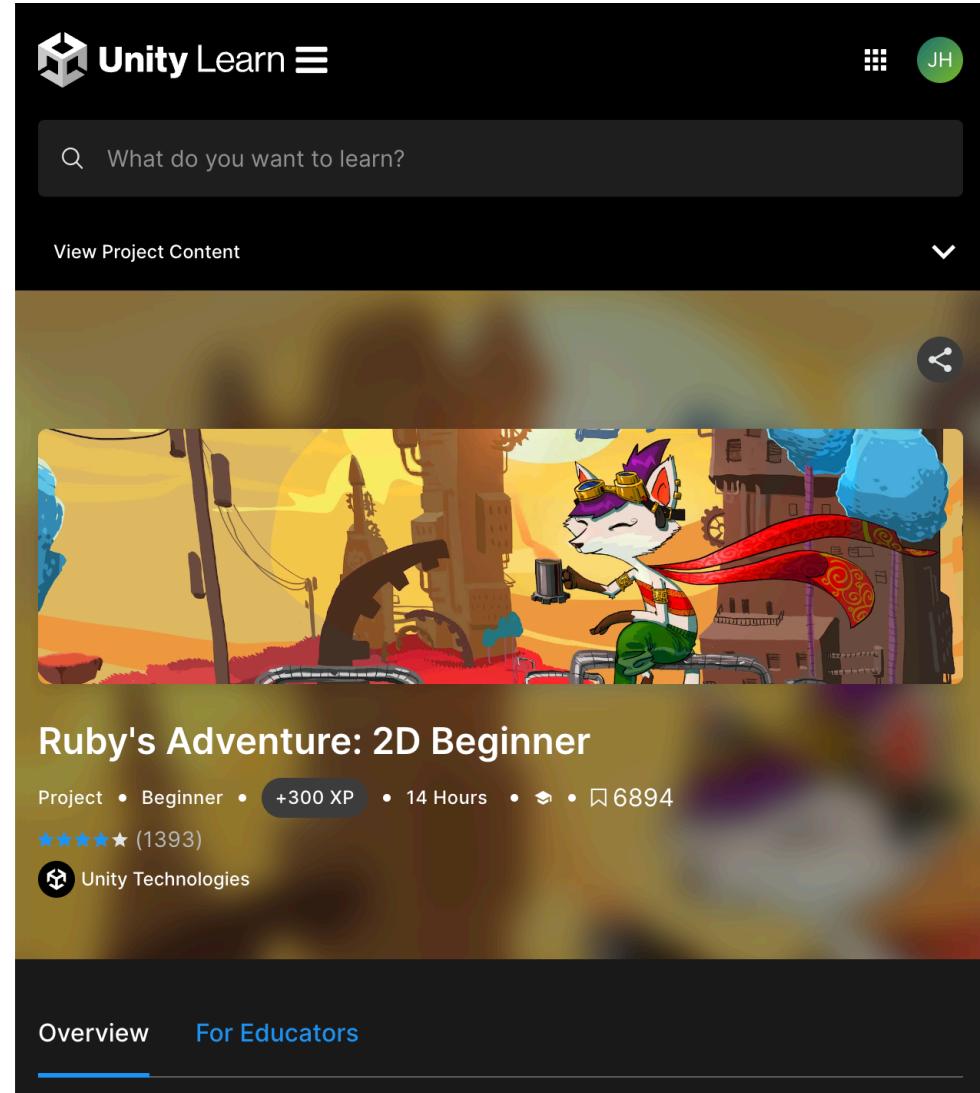
Unity Learn

Unity Learn:

- Unity's official tutorials
- All "difficulty" levels
- (New) systems explained
- Up-to-date

Ruby's Adventure: 2D Beginner

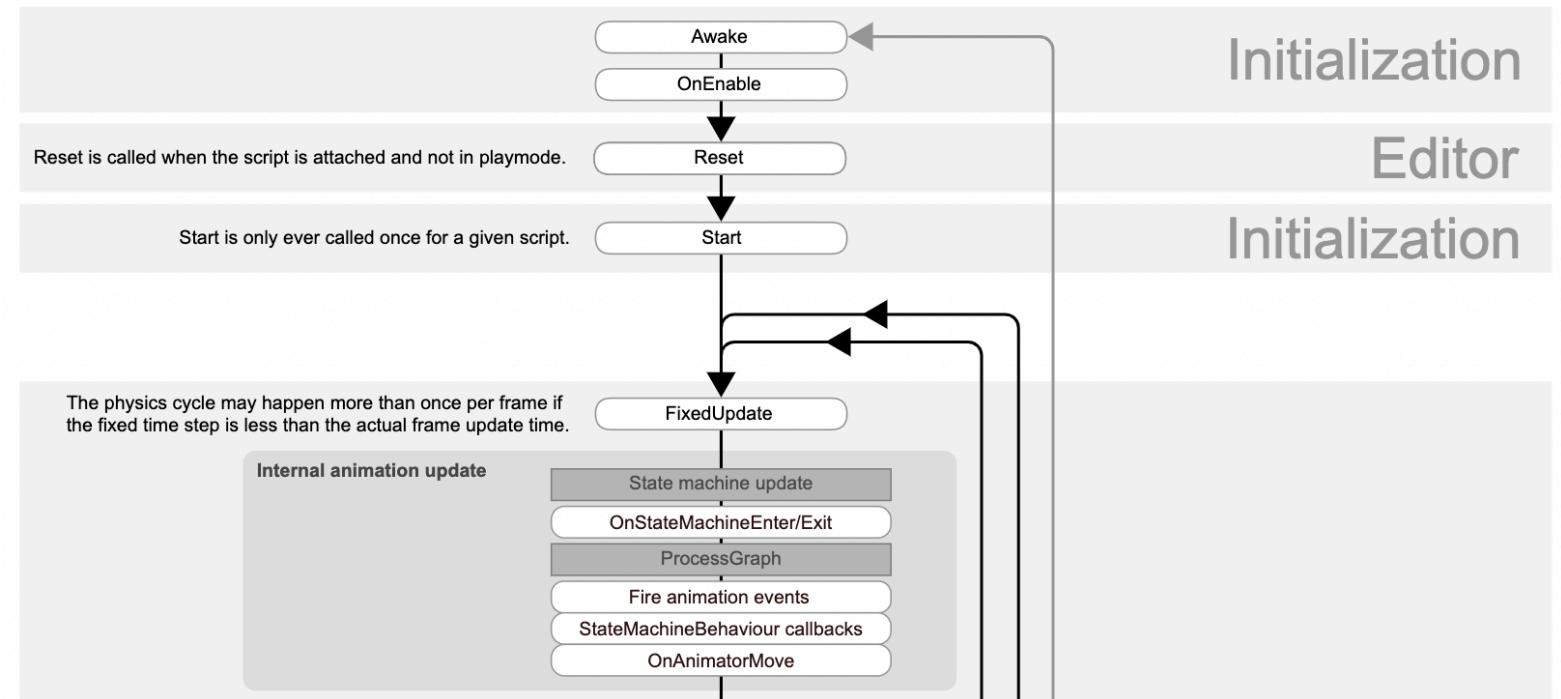
Recommended tutorial for 2D game dev [\[link\]](#) ->



Unity's Script Lifecycle

Most used, in order of execution:

- Awake
- OnEnable
- Start
- FixedUpdate
- OnCollision
(+Enter/Exit/Stay)
- Update
- OnDisable



From: [Unity Manual - Order of execution for event functions \[link\]](#)



Part 2

Creating and Destroying GameObjects

Unity's Component-based workflow

Using Prefabs

Referencing (Game)Objects

InputSystem



Creating and Destroying GameObjects

Asd



Unity's Component-based workflow

- Adding components
- Referencing by component



Using Prefabs

- Prefab editing
- Dragging prefabs into a scene



Referencing (Game)Objects

- How Unity handles dragging references
 - Difference between Project view and Hierarchy

(Next week more details)



InputSystem











