GameDev.tv

Unity 2020.1

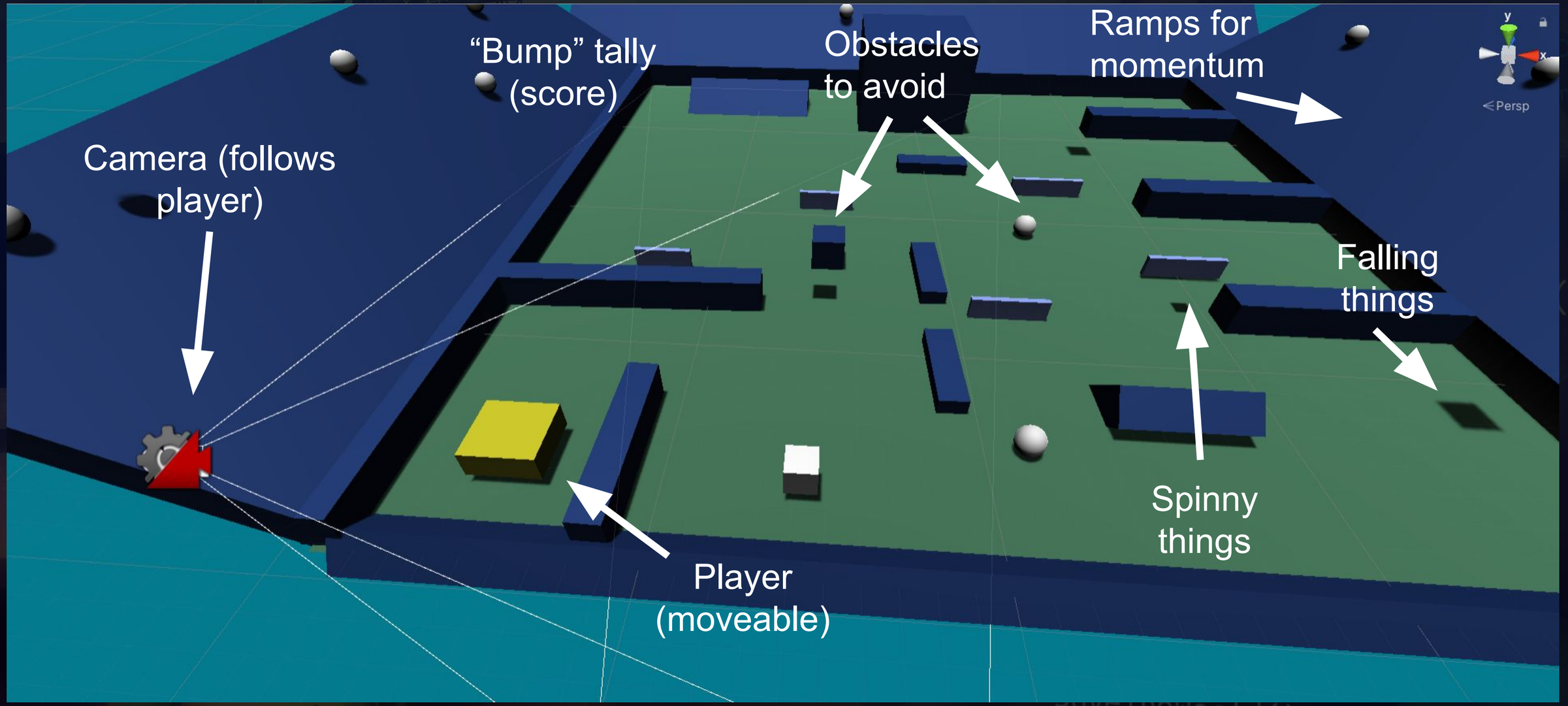
Section Intro - Obstacle Course

GameDev.tv

Unity 2020.1

Game Design - Obstacle Course

Core Gameplay Overview



Game Design

Player Experience:

Careful? Clever? Nah

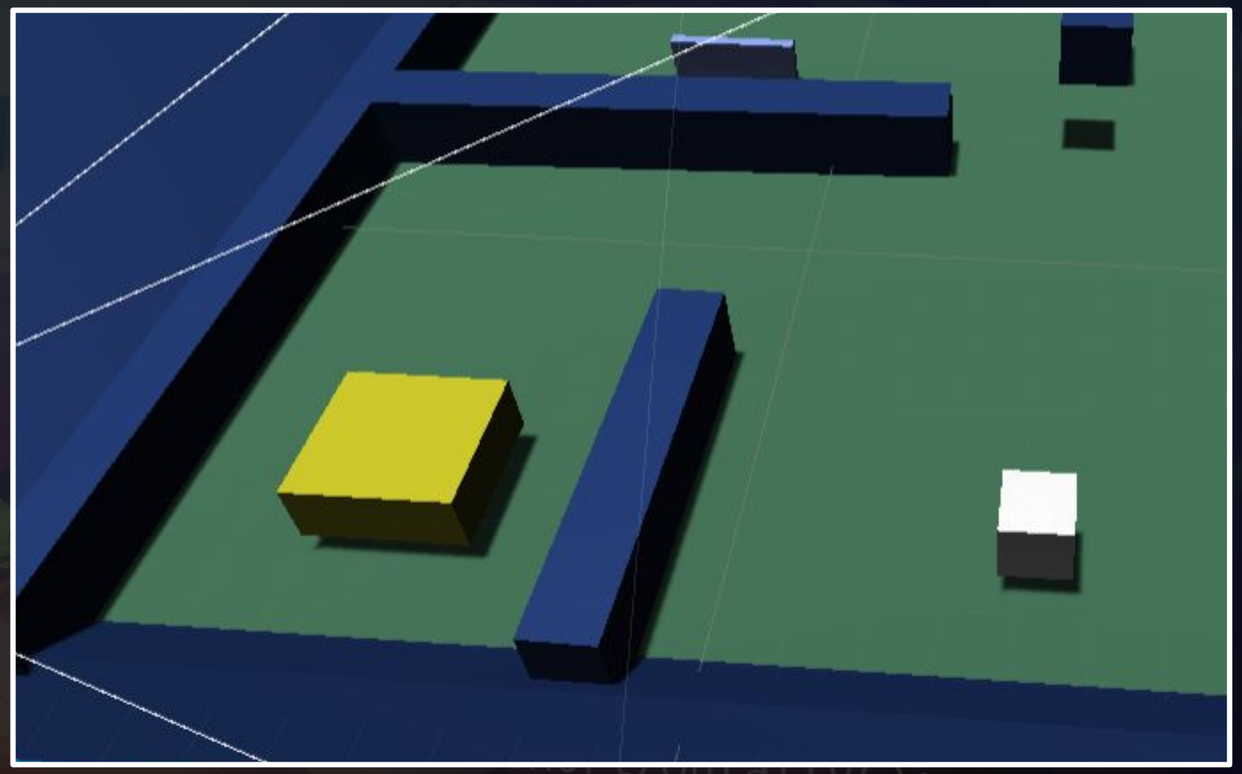
Nimble / agile

Core Mechanic:

Move & dodge obstacles

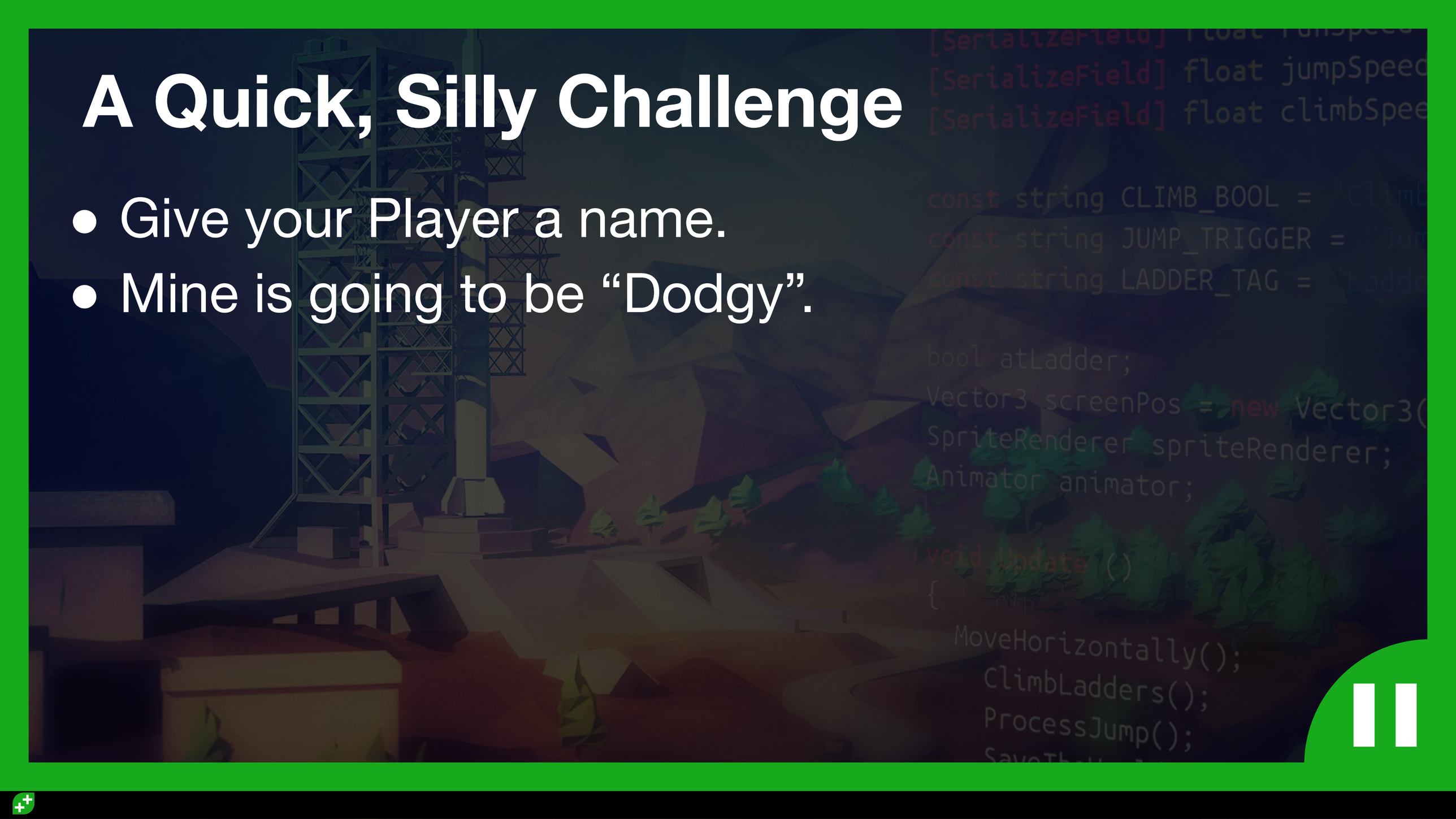
Game Loop:

Get from A to B



A Quick, Silly Challenge

- Give your Player a name.
- Mine is going to be “Dodgy”.



GameDev.tv

Unity 2020.1

Start() & Update()


```
[SerializeField] float runSpeed =
[SerializeField] float jumpSpeed =
[SerializeField] float climbSpeed

const string CLIMB_BOOL =
const string JUMP_TRIGGER =
const string LADDER_TAG =

bool atLadder;
Vector3 screenPos = new Vector3();
SpriteRenderer spriteRenderer;
Animator animator;

void Update ()
{
    MoveHorizontally();
    ClimbLadders();
    ProcessJump();
    SaveTheWorld();
}
```

Get The Party Started

- Create a new Unity 3D project
- Add a ground plane
- Create your “player”
- Rename your player

GameDev.tv

Unity 2020.1

Introducing Variables

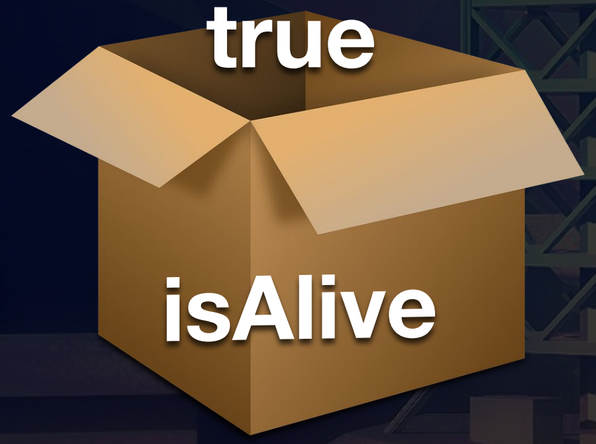
Variables Are Like Boxes

- Variables help us store, manipulate and refer to information
- Each variable has a NAME
- Each variable contains DATA
- Each variable is of a particular TYPE


```
int hitPoints = 20;
```

Variables Are Like Boxes

```
float speed = 3.8f;
```




```
bool isAlive = true;
```

```
string myName = “Rick”;
```

Create Two More Variables

- Create a variable for the y value and for the z value.
- Change the values so your player flies straight up in the air

GameDev.tv

Unity 2020.1

Using SerializeField

Serialize Your Other Variables

- Make all three of your variables serialized and therefore accessible in the inspector.
- While in play mode, make your player run away from the camera and then run back to the camera.

GameDev.tv

Unity 2020.1

C# Formatting & Input.GetAxis()

Add Vertical Axis

- Update one of our variables so we are moving our player forwards and backwards (along the ground plane, not flying in the air).

GameDev.tv

Unity 2020.1

Time.deltaTime

Using Time.deltaTime

- Using `Time.deltaTime` Unity can tell us how long each frame took to execute.
- When we multiply something by `Time.deltaTime` it makes our game “frame rate independent”.
- Ie. The game behaves the same on fast and slow computers

On Update (each frame) move 1 unit to the left

	Slow Computer	Fast Computer
Frames per second	10	100
Duration of frame	0.1s	0.01s
Distance per second	1 x 10 x 0.1 = 1	1 x 100 x 0.01 = 1

Multiply By A Speed Variable

- Create a new variable called moveSpeed. The value of moveSpeed does not need to update each frame.
- Make it available in the inspector.
- Multiply your xValue and zValue by moveSpeed.
- Tune your player movement (as best you can for now).

GameDev.tv

Unity 2020.1

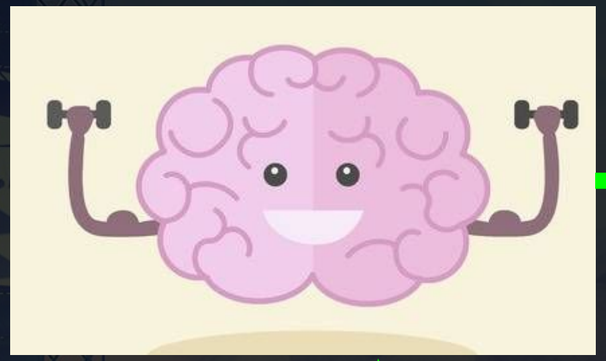
Cinemachine Follow Camera

Rick Davidson

What Is Cinemachine?

- **Cinemachine** is a powerful package that lets us:
  - manage multiple cameras in our scene
  - Easily create rules for our cameras

Cinemachine Brain



Main Camera

Virtual Cameras

Add A Follow Camera

- Install Cinemachine from Package Manager
- Add Cinemachine Brain component to camera
- Add Virtual Camera
- Rename Virtual Camera
- Point the Virtual Camera to follow the player
- Tune distance

GameDev.tv

Unity 2020.1

Basic Collisions

Make Some Walls

- Add some walls around the outside of your play area
- If you like, make them a different colour

GameDev.tv

Unity 2020.1

Introduction To Methods

What Are Methods?

- Methods (also called Functions) execute blocks of code that makes our game do things.
- To achieve this we must:
  1. DECLARE and define our method
  2. CALL our method when we want it to execute

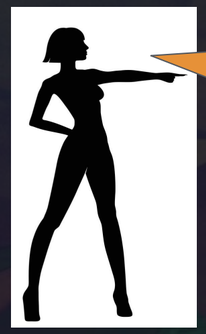
Real World Example

DECLARE Method

CleanYourRoom

- Pick up clothes
- Throw out garbage
- Kill the mutated science project that threatens to destroy life as we know it

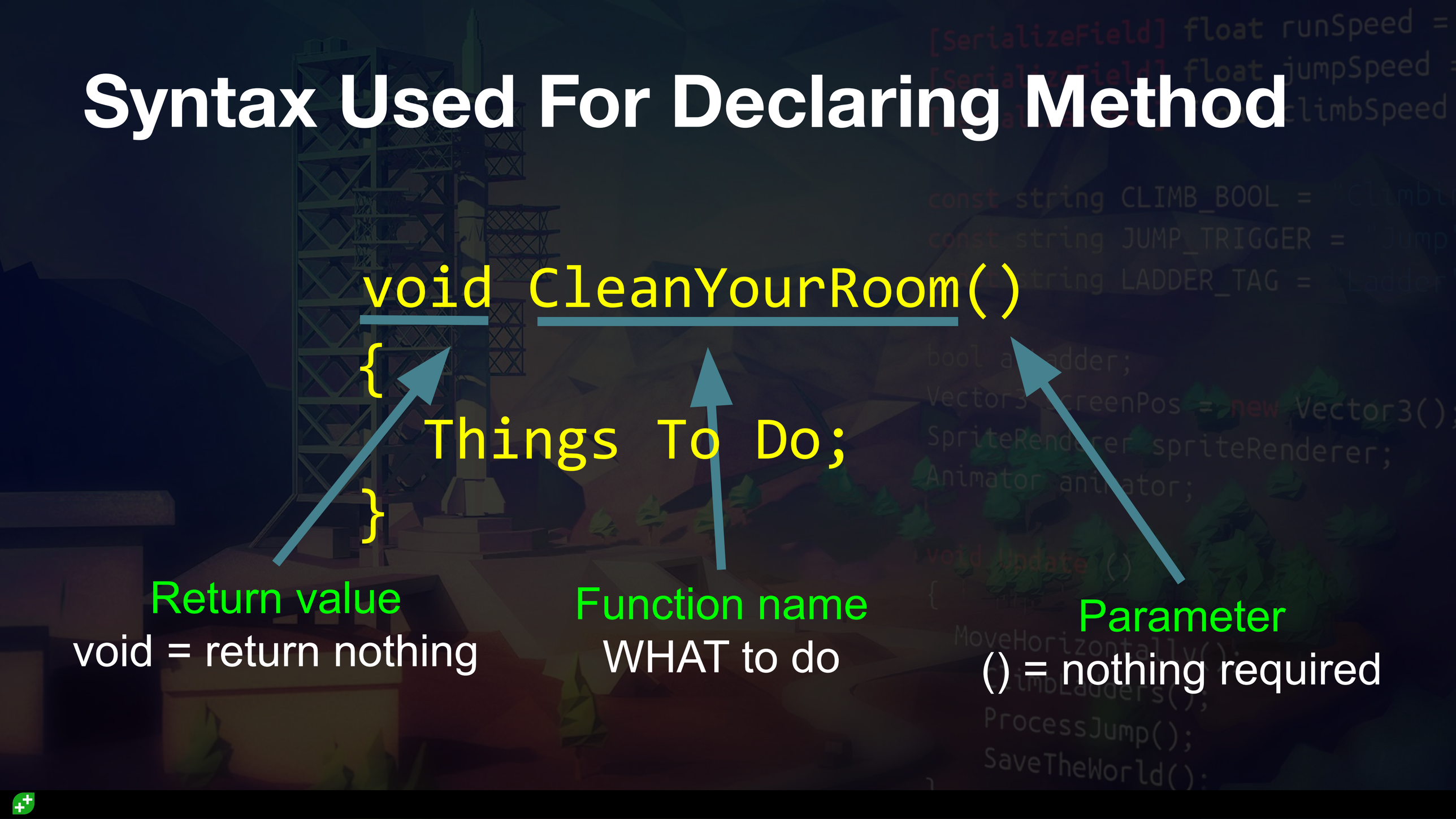
CALL Method



Hey, go and *CleanYourRoom*

“Go and do the steps we’ve defined and agreed to”

Syntax Used For Declaring Method



More Flavour Can Be Added

Two more common things when calling methods:

1. We can ask for some information to be RETURNED
2. We can specify some PARAMETERS are needed when calling the method

Using our real world example again...

More Flavour Can Be Added

DECLARE Method

CleanYourRoom(deadline)

- Do it before (deadline)
- Pick up clothes
- Throw out garbage
- Kill the mutated science project that threatens to destroy life as we know it
- Is the room dirty?

CALL Method

Hey, go and *CleanYourRoom* (*you've got 1 hour*). *isRoomDirty?*

*Parameter / Argument

*Return data

Syntax Used For Declaring Method

```
void CleanYourRoom()
{
    Things To Do;
}
```

Return value
void = return nothing

Function name
WHAT to do

Parameter
() = nothing required

Syntax Used For Declaring Method

```
bool CleanYourRoom(int time)
{
   Things To Do;
    Deadline = time;
    return isRoomDirty;
}
```

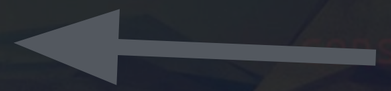
Return value
(Type = bool)

Return keyword
(Type = bool)

Parameter
Needs type
and name

Syntax Used For Calling Method

```
CleanYourRoom();
```



Remember semi colon

Executes statements defined in the code block...

```
{
  ...
}
```

Defining & Calling

DECLARE Method

```
void CleanYourRoom()
{
    Pick up clothes;
    Clean garbage;
    Kill mutated experiment;
}
```

CALL Method

```
CleanYourRoom();
```

But What About Start() & Update()?!

- We've been defining `Start()` & `Update()` but not calling them
- Where are they called?
- Unity's internal logic is taking care of calling them for us at the right time
- These are referred to as "callbacks"

GameDev.tv

Unity 2020.1

Practicing With Methods

Call Our Method

- Call our method so that our instructions print out one time to the console.

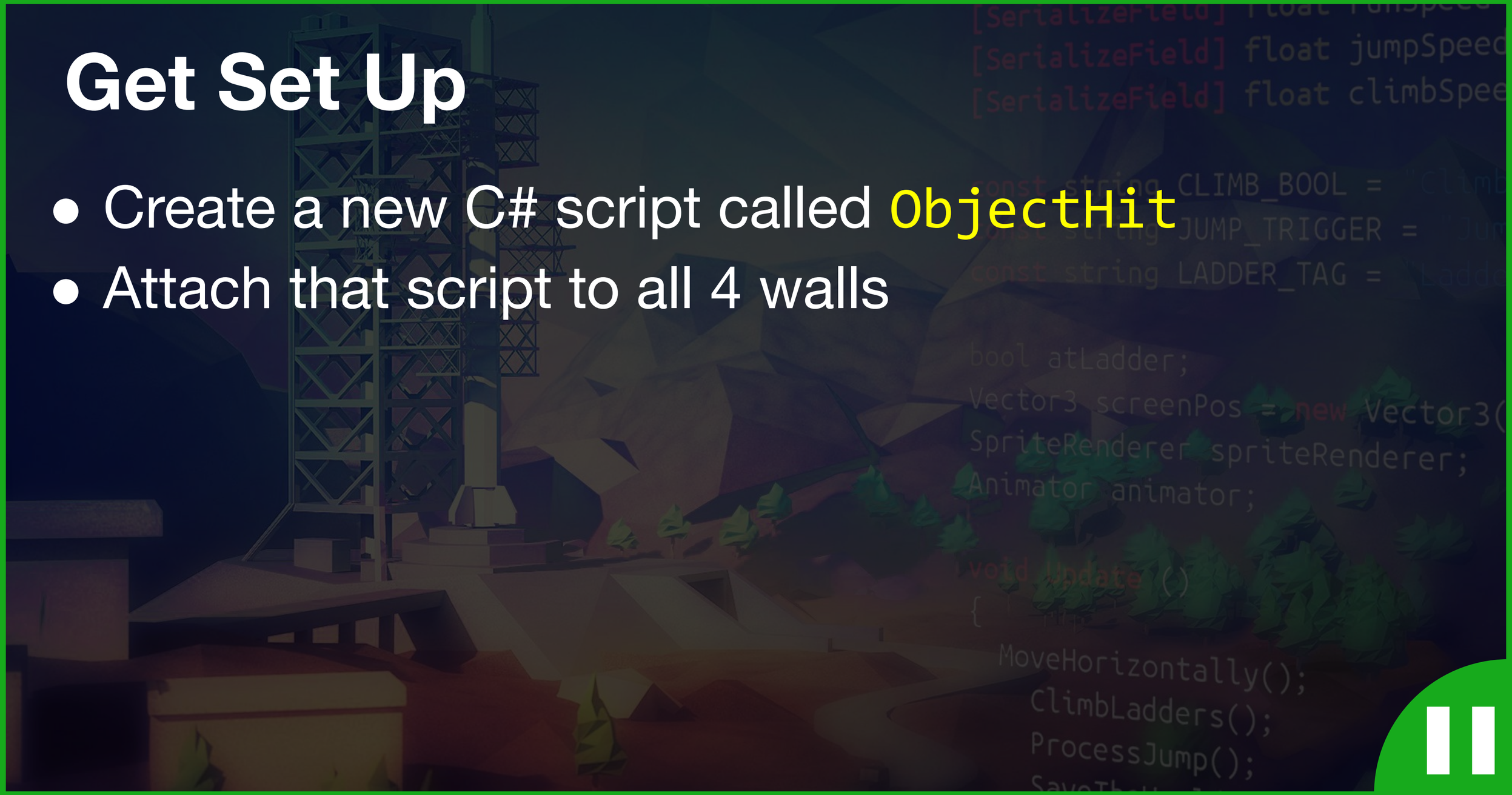
GameDev.tv

Unity 2020.1

Using OnCollisionEnter()

Get Set Up

- Create a new C# script called `ObjectHit`
- Attach that script to all 4 walls



GameDev.tv

Unity 2020.1

Using GetComponent<>()

GameDev.tv

Unity 2020.1

Incrementing A Score

Prepare For Score Feedback

- Create a new c# script called `Scorer`
- Create a new `OnCollisionEnter()` method
- When we hit something, print to the console, “You’ve bumped into a thing this many times: ”

GameDev.tv

Unity 2020.1

Using Time.time

Problem & Solution

Problem to solve:

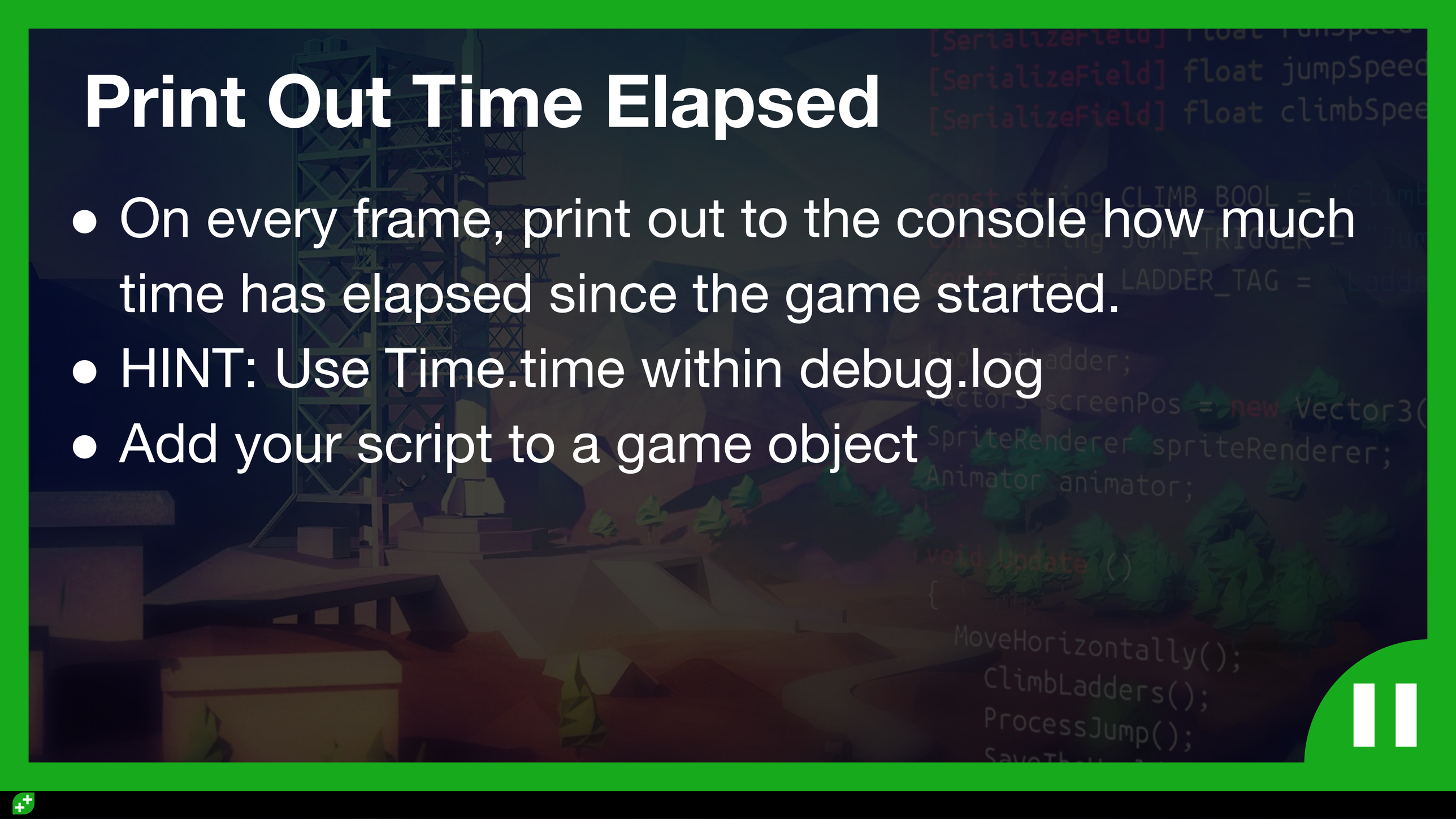
- Make an object fall after 3 seconds has passed

Solution:

1. A timer - `Time.time`
2. A mechanism to “do a thing if 3 seconds has elapsed” - if statement
3. A way to start the object falling after 3 seconds - disable / enable gravity

Print Out Time Elapsed

- On every frame, print out to the console how much time has elapsed since the game started.
- HINT: Use Time.time within debug.log
- Add your script to a game object



GameDev.tv

Unity 2020.1

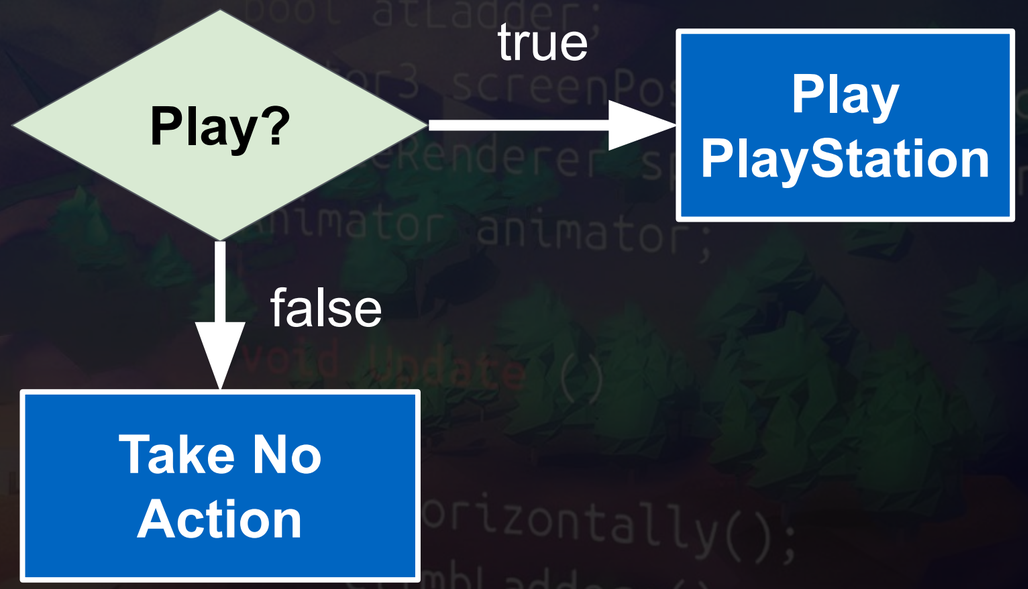
If Statements

Conditional `if` Statement

```
if (some expression evaluates to true)
{
  // execute this code block only
}
```

In Other Words...

```
if (in the mood for Pwning Noobs)
{
  // play some PlayStation
}
```



Use A Variable For Time To Wait

- Instead of “hard coding” 3 seconds, use a variable for the *time to wait* that can be easily changed in the Inspector.

GameDev.tv

Unity 2020.1

Caching A Reference

Simple Definition...

Caching is a technique of storing frequently used data/information in memory, so that it can easily be accessed when needed.

Make It Fall!

- Add 2 lines to our code that make our object become visible and fall after it has waited for the right amount of time.

GameDev.tv

Unity 2020.1

Using Tags

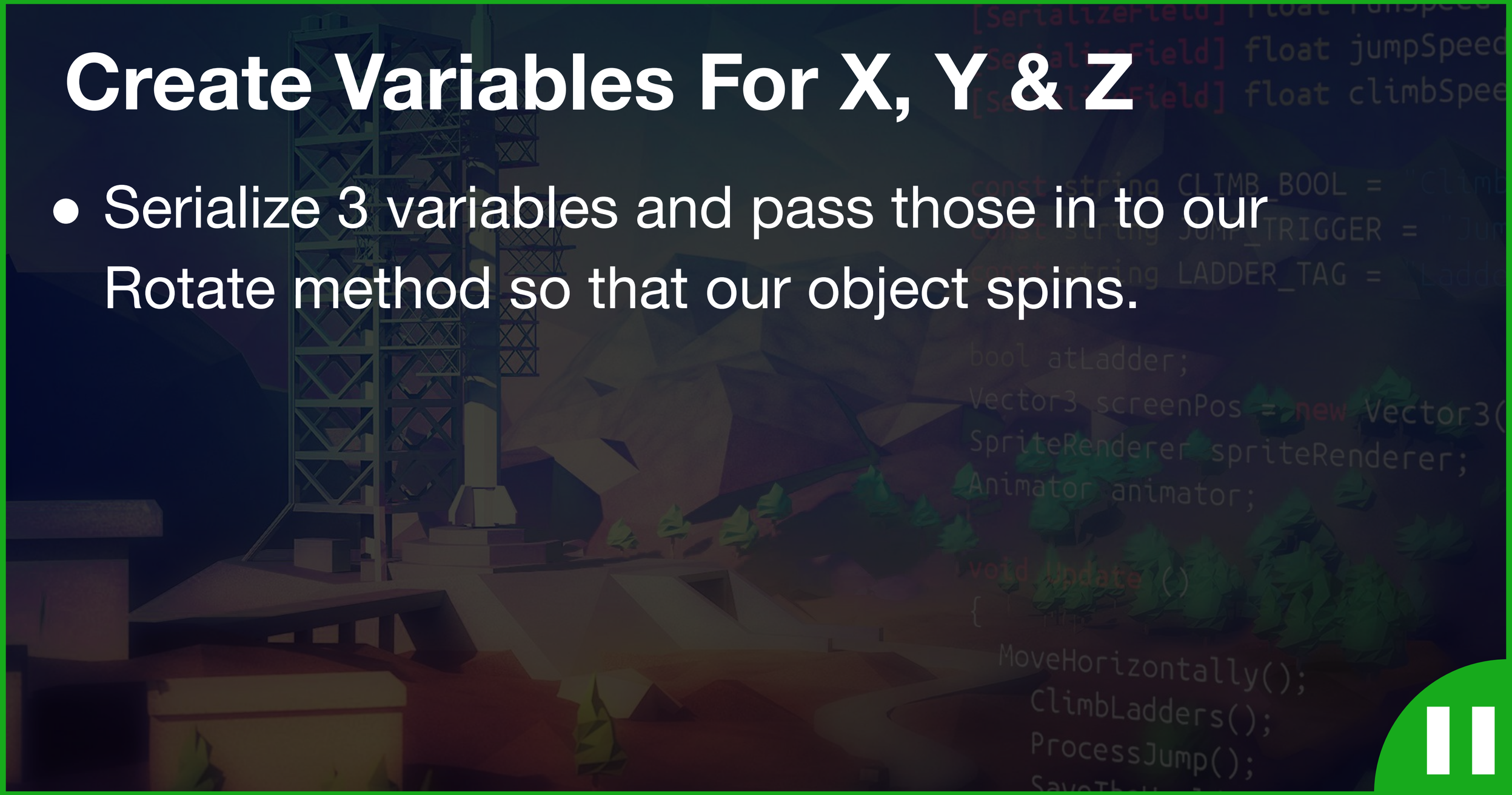
GameDev.tv

Unity 2020.1

Rotate An Object

Create Variables For X, Y & Z

- Serialize 3 variables and pass those in to our Rotate method so that our object spins.



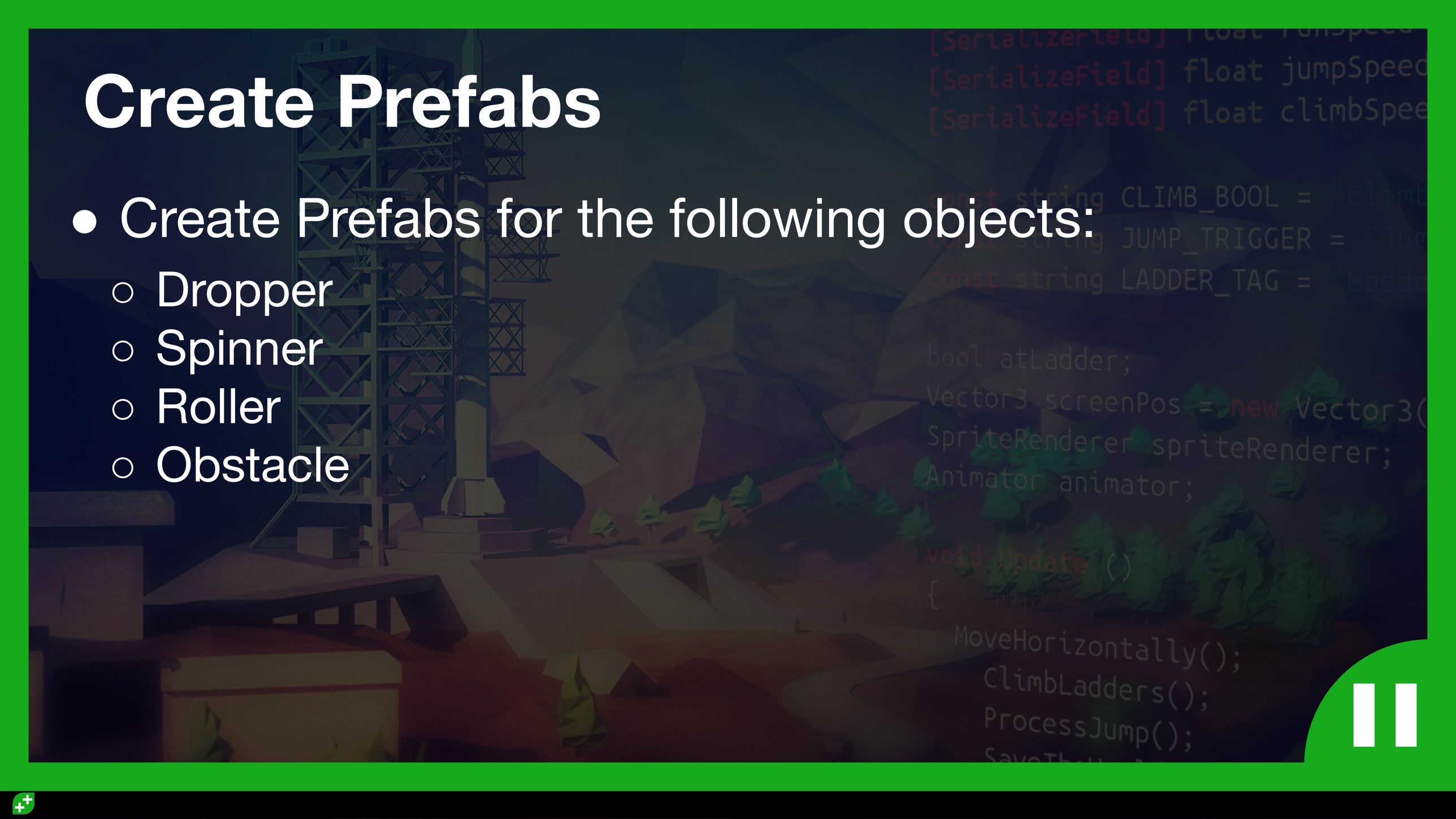
GameDev.tv

Unity 2020.1

Prepare Our Prefabs

Create Prefabs

- Create Prefabs for the following objects:
  - Dropper
  - Spinner
  - Roller
  - Obstacle



GameDev.tv

Unity 2020.1

Build An Obstacle Course

Rick Davidson

Build Your Obstacle Course!

- Create a fun layout that makes the player go from A to B.
- Use your droppers, rollers and spinners to create interesting moments for the player.

GameDev.tv

Unity 2020.1

Wrap Up - Obstacle Course


[SerializeField] float runSpeed =
[SerializeField] float jumpSpeed =
[SerializeField] float climbSpeed
const string CLIMB_BOOL = "Climbi
const string JUMP_TRIGGER = "Jump
const string LADDER_TAG = "Ladder
bool atLadder;
Vector3 screenPos = new Vector3();
SpriteRenderer spriteRenderer;
Animator animator;
void Update ()
{
MoveHorizontally();
ClimbLadders();
ProcessJump();
SaveTheWorld();
}