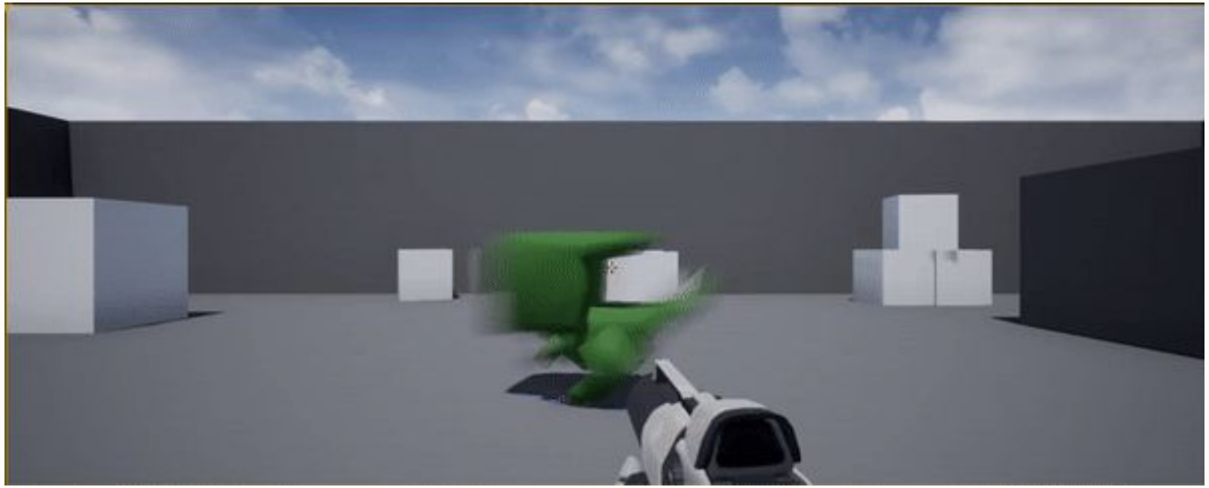
Intro to UNREAL ENGINE

FPS

acmForge

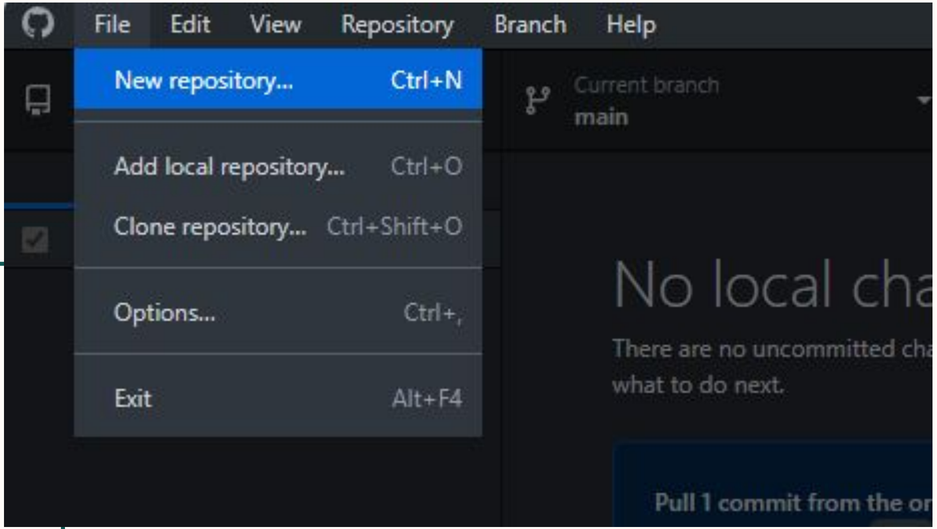
UNREAL ENGINE

Setting up a Unreal Repo



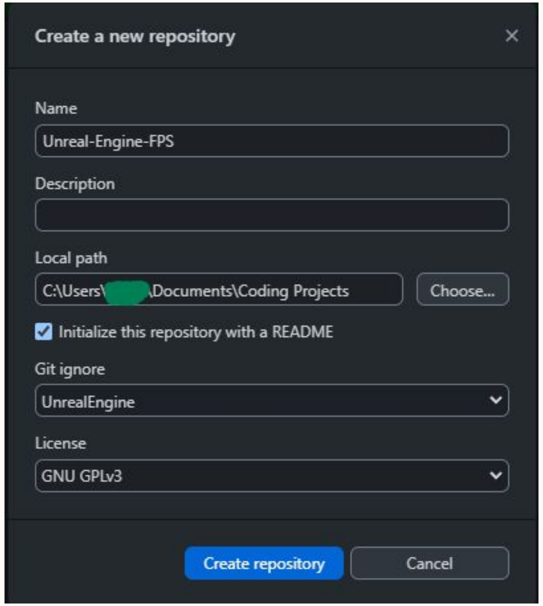
Create a New Github Repo

Open **GitHub Desktop** and select **File > New repository**



No local cha

There are no uncommitted cha
what to do next.



Name: Unreal-Engine-FPS

Description:

Local path: C:\Users\[redacted]\Documents\Coding Projects

☑ Initialize this repository with a README

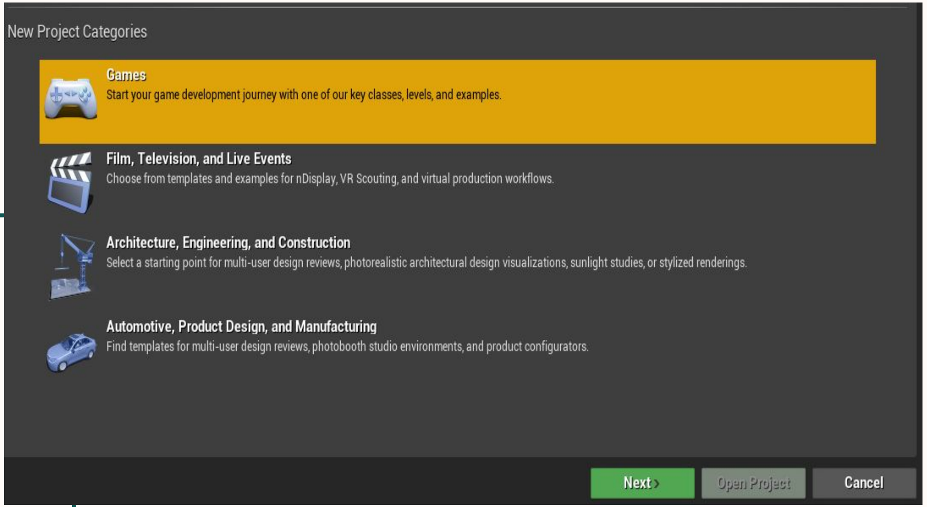
Git ignore: UnrealEngine

License: GNU GPLv3

More detailed tutorial on ACM Blog

Create New Project in Unreal

Open **Unreal Engine**, select **Games** and **First Person**



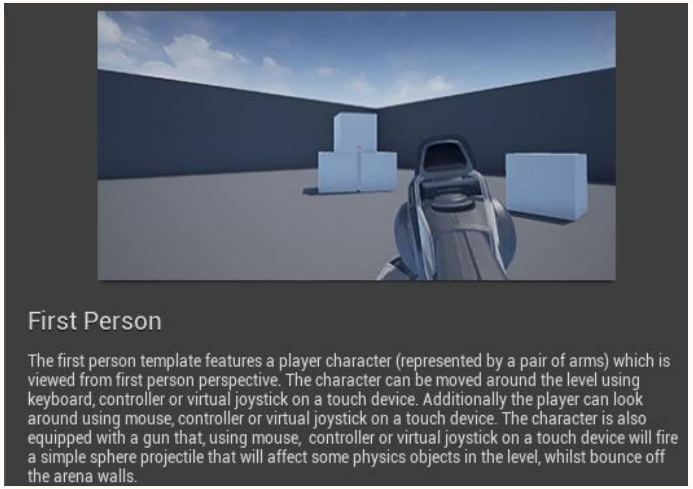
New Project Categories

Games
Start your game development journey with one of our key classes, levels, and examples.

Film, Television, and Live Events
Choose from templates and examples for nDisplay, VR Scouting, and virtual production workflows.

Architecture, Engineering, and Construction
Select a starting point for multi-user design reviews, photorealistic architectural design visualizations, sunlight studies, or stylized renderings.

Automotive, Product Design, and Manufacturing
Find templates for multi-user design reviews, photobooth studio environments, and product configurators.

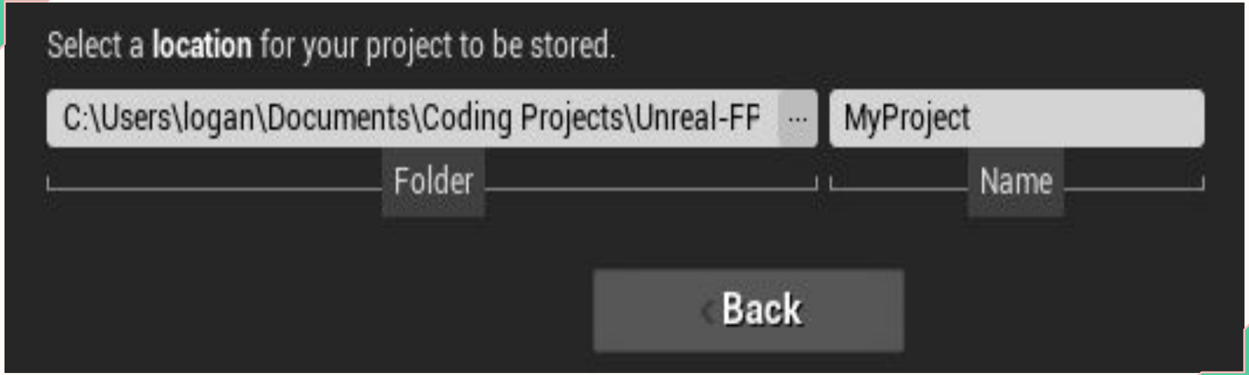


First Person

The first person template features a player character (represented by a pair of arms) which is viewed from first person perspective. The character can be moved around the level using keyboard, controller or virtual joystick on a touch device. Additionally the player can look around using mouse, controller or virtual joystick on a touch device. The character is also equipped with a gun that, using mouse, controller or virtual joystick on a touch device will fire a simple sphere projectile that will affect some physics objects in the level, whilst bounce off the arena walls.

IMPORTANT!

If you are initializing this project in the repo from before,


Select a **location** for your project to be stored.

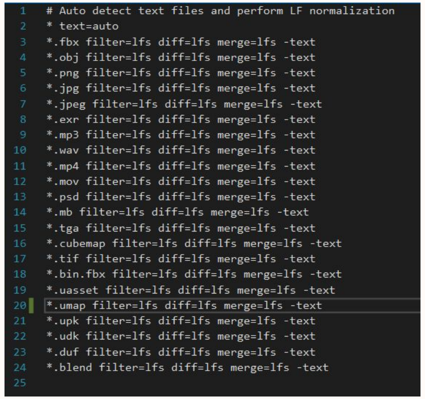
Folder: C:\Users\logan\Documents\Coding Projects\Unreal-FF

Name: MyProject

make sure to put your new game into
that folder!!!

Github Large File Storage (LFS)

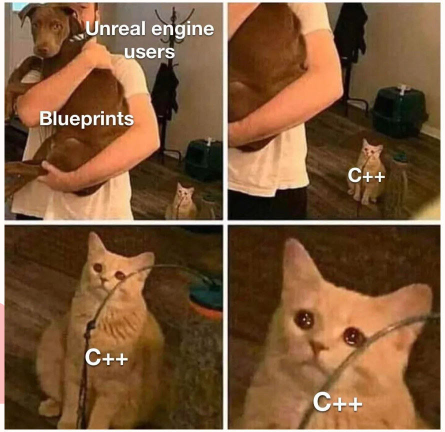
- Sometimes files are too large to push to github. For this reason, you can have your repo use git LFS
- Setting up Git and Git LFS: https://www.youtube.com/watch?v=_ewoEQFEURg
    - This video is prescribed for Unity. However, it can be used for Unreal.
    - Note: there is no defined way to set up git lfs with Github Desktop. If it is needed, Github Desktop may prompt you to initialize it.
- .gitignore tells the repo to ignore the updates to the files listed and not commit them
- .gitattributes tells the repo which files need to be stored in git lfs



```
# Auto detect text files and perform LF normalization
* text=auto
*.fbx filter=lfs diff=lfs merge=lfs -text
*.obj filter=lfs diff=lfs merge=lfs -text
*.png filter=lfs diff=lfs merge=lfs -text
*.jpg filter=lfs diff=lfs merge=lfs -text
*.jpeg filter=lfs diff=lfs merge=lfs -text
*.exr filter=lfs diff=lfs merge=lfs -text
*.mp3 filter=lfs diff=lfs merge=lfs -text
*.wav filter=lfs diff=lfs merge=lfs -text
*.mp4 filter=lfs diff=lfs merge=lfs -text
*.mov filter=lfs diff=lfs merge=lfs -text
*.psd filter=lfs diff=lfs merge=lfs -text
*.mb filter=lfs diff=lfs merge=lfs -text
*.tga filter=lfs diff=lfs merge=lfs -text
*.cubemap filter=lfs diff=lfs merge=lfs -text
*.tif filter=lfs diff=lfs merge=lfs -text
*.bin.fbx filter=lfs diff=lfs merge=lfs -text
*.uasset filter=lfs diff=lfs merge=lfs -text
*.umap filter=lfs diff=lfs merge=lfs -text
*.upk filter=lfs diff=lfs merge=lfs -text
*.udk filter=lfs diff=lfs merge=lfs -text
*.duf filter=lfs diff=lfs merge=lfs -text
*.blend filter=lfs diff=lfs merge=lfs -text
```

Programming in Unreal
(Object Oriented Architecture)

C++ - that thing you learned in CPSC 121;
For the workshop we will be using this



Blueprint Visual Scripting

- gameplay scripting system using a node-based interface to create gameplay elements from within Unreal Editor.
- It's like connecting a bunch of blocks together.

Using C++ and Blueprints together:
https://www.youtube.com/watch?v=VMZftEVDuCE

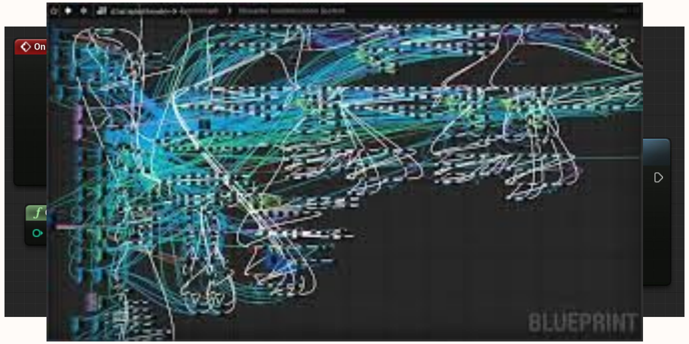
BLUEPRINTS vs C++

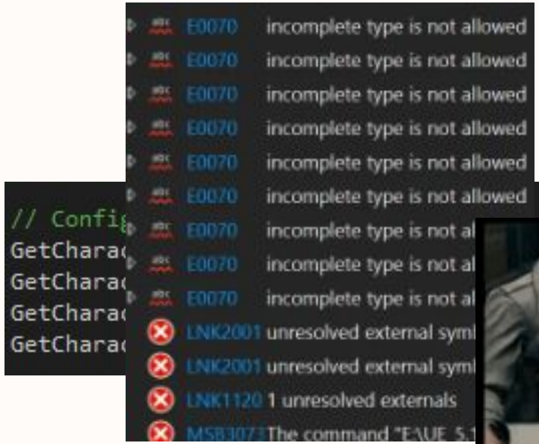
😀
- Designer friendly
- Visual

😐
- Can get messy

😎
- Powerful
- Practical

😩
- Difficult to find help





```
// Config
GetCharac
GetCharac
GetCharac
GetCharac
```

```
E0070   incomplete type is not allowed
E0070   incomplete type is not allowed
E0070   incomplete type is not allowed
E0070   incomplete type is not allowed
E0070   incomplete type is not allowed
E0070   incomplete type is not allowed
E0070   incomplete type is not al
E0070   incomplete type is not al
E0070   incomplete type is not al
LNK2001 unresolved external sym
LNK2001 unresolved external sym
LNK1120 1 unresolved externals
MSB3073 The command "E:\UE_5.
```

Intro to C++ in Unreal Resources:

- Unreal Engine's Documentation:
  - https://docs.unrealengine.com/4.26/en-US/ProgrammingAndScripting/ProgrammingWithCPP/IntroductionToCPP/
- Free Youtube Course:
  - https://www.youtube.com/watch?v=zEcNn4gWas0&list=PL3gCaTLUSAUsHG2BzsAs-HIeP08DyWtHh
- NOTE:
  - Tutorials are useful only if you know how to use them.
    - Following along exactly with what a tutorial is teaching you will often result in not truly learning because you are simply mimicking what another person is doing.
    - To get something out of it, you should try to apply what you learn in the tutorial to your own project.

Common Unreal C++ items

UFUNCTION

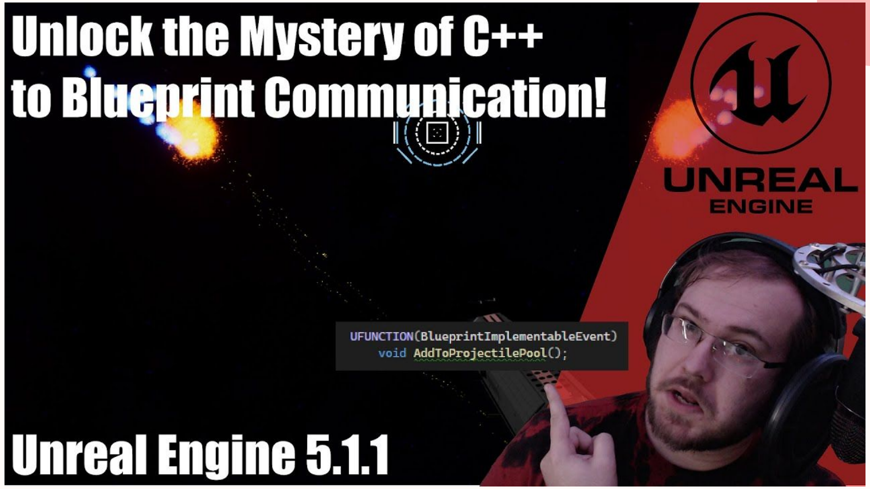
- Creates a callable function that can be accessed, and changed in the Unreal Editor

UPROPERTY

- Creates a variable that can be accessed, and changed in the Unreal Editor

Note:

- For those who have used Unity, this is much like when you make a variable public, or use the key word Serialize Field.
- You are able to edit the variable in the editor instead of the script itself.
- The macros UFUNCTION and UPROPERTY allow for blueprints to also use the functions and variables made in C++.



```
UFUNCTION(BlueprintImplementableEvent)
    void AddToProjectilePool();
```

Actor, Pawn, Character

Actor

- any object that can be placed into a level, such as a Camera, static mesh, or player start location

Pawn

- a subclass of Actor and serve as an in-game avatar or persona. Pawns can be controlled by a player or by the game's AI, as non-player characters (NPCs)

Character

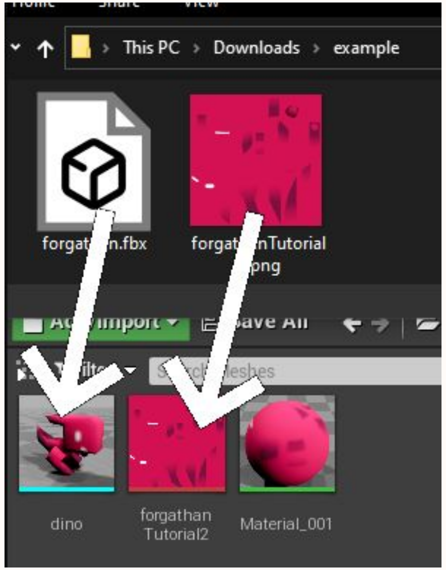
- subclass of a Pawn Actor that is intended to be used as a player character

For more information:
https://skwrites.in/important-unreal-game-engine-concepts/

Importing Models into Unreal

You can bring your dino from the Blender workshop into Unreal!

Drag'n Drop



Assets

Go to this google drive link for the **free assets!**

Thanks for coming!!