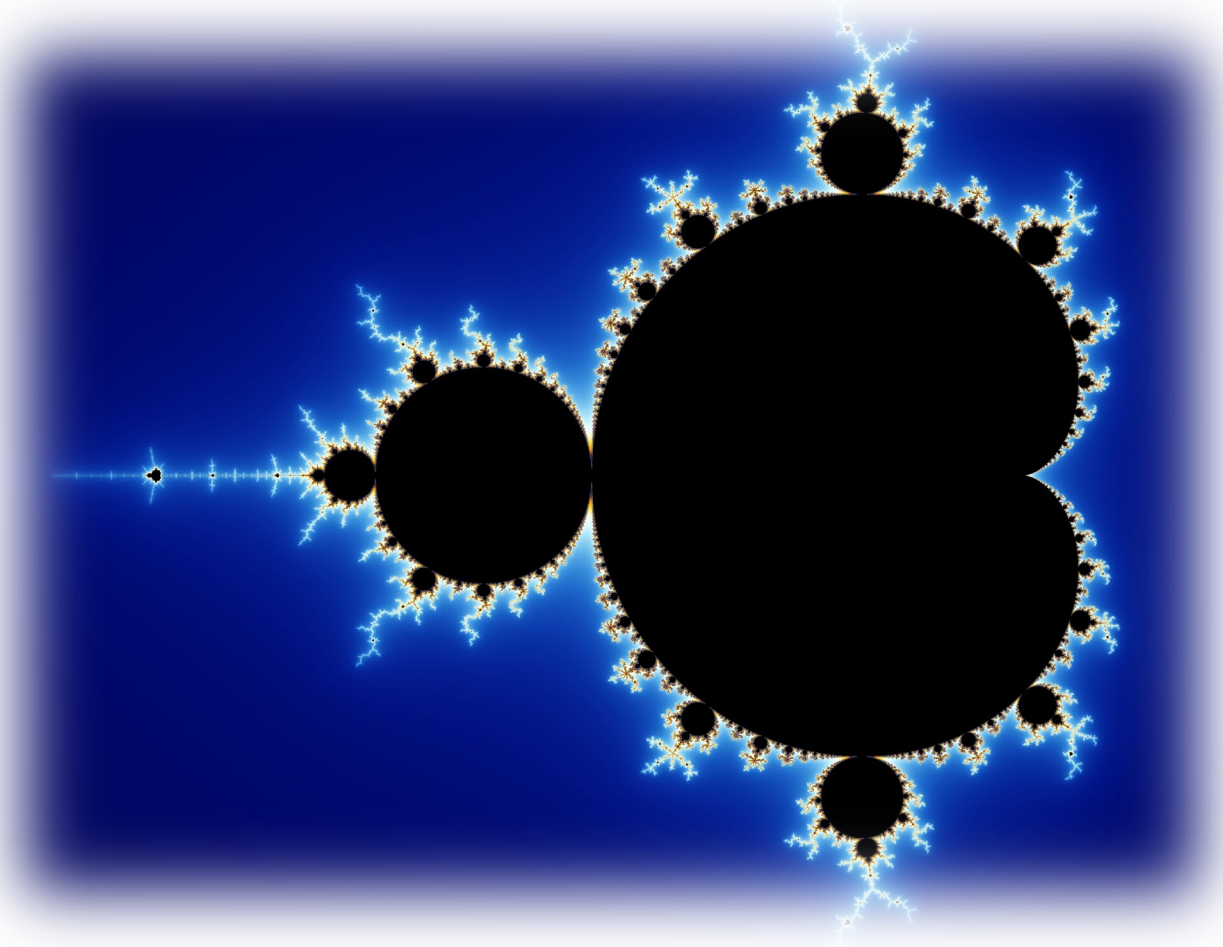


INTRO TO UNREAL & BLUEPRINTS

Ci541

ENGINES IN GENERAL

Great at 3D
- Overkill for 2D
- Cumbersome

Great at 2D
- Lacking tools for 3D
- Slow in 3D

Great at multiplatform
- Under optimized for each specific one

Free/Cheap
- Lack of support

Proprietary (e.g. Frostbite) & specific to needs
- Expensive to support & train staff

Product of compromises

SPECIFICALLY*

Unity 3D

- Easy to learn, simple deployment to mobile, good documentation.
- Primarily aimed at indie studios and mobile
- Many version updates
- C# a modern OOP language which is easy to use, runs on .NET , partially compiled, does not need header files.
- Little control over low level functionality, no source code (unless you pay), can be performance limited
- Low bar to entry results in medium relevance to finding a job programming in games

Unreal Engine

- Used by many major 3D games developers, access to source code, performance optimised. Suitable for large multi developer projects.
- Generally poor documentation
- C++ derived from C (circa 1970), its now modern however due to legacy issues with supporting C non OOP syntax can be quite complex. Requires separate header files
- Blueprints, visual coding allowing quite complex interactions
- High bar to entry results in high relevance to finding a job programming in games

*Personal opinion

OBJECT ORIENTED DESIGN

Encapsulation

- Objects define their own behaviour
- Interfaces connect Objects
- In C# and C++ (and Java) objects are called classes

Outside world

- Actions drive events
- Behaviour is response to environment

WHO'S IN CHARGE?

Engine drives design?

or

Design drives engine?

DEVELOPERS TOOLBOX

3D Assets

- Maya
- 3D Studio Max
- Blender (free)

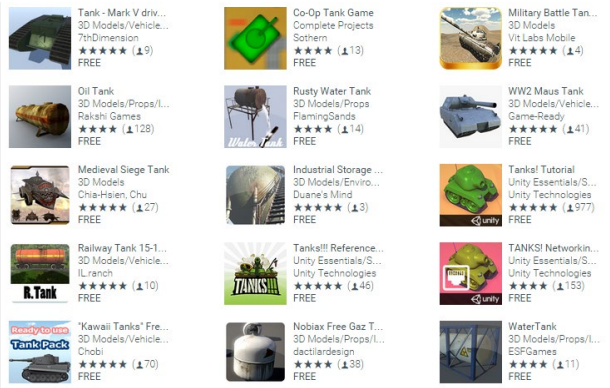
2D Assets

- Photoshop
- Gimp (free)

Coding

- Visual Studio (community free)
- XCode (free)
- GCC (free)
- Mono Develop (free)

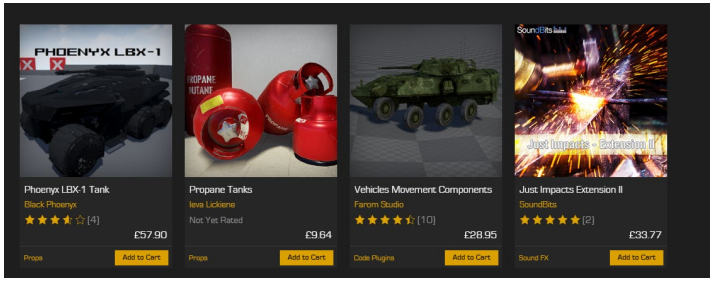
3RD PERSON ASSETS



Other peoples code, art, and methods can be a time saver

HOWEVER: When choosing consider both form and function

- Will it work in the way you want?
- What are its limitations
- Could you fix it if it goes wrong?



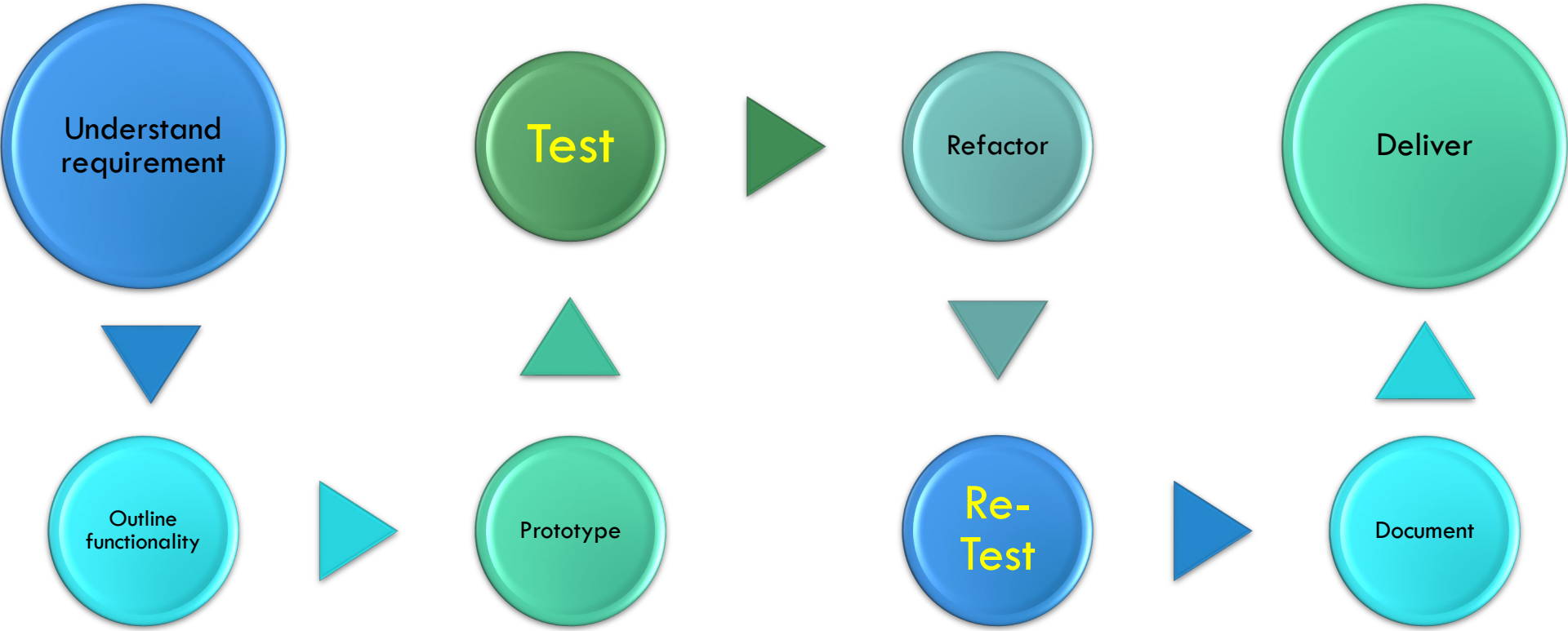
Rule of thumb

- If you don't know why/how/when it works don't use it
- Fixing other peoples stuff is very hard
- Only use if you feel confident that you could make it, so its just saving time
- Ensure it meets all the functionally you need
  - Texture budget, dependencies, modularity, construction method, documentation
  - If its free, its probably worth what you paid for it ☺

USE OF 3RD PARTY ASSETS & CODE IN CW

1. Must be credited
2. Has to have evidence its been understood
3. Should not be at the core of your submission
4. Must have suitable licence terms to be used

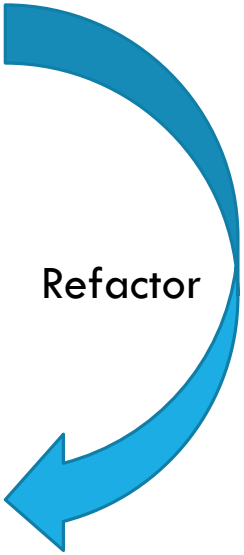
DEVELOPMENT SNAKE



PROTOTYPE VS DEVELOPMENT

Prototype = quick and possibly dirty first stab at solution

- Focus
  - Just get it working
  - Not about elegance or high performance
- Key goal
  - Expose what you do not yet understand
  - Share proposed solution with others
  - Cheap mistakes



Development = make is safe and future proof

- Focus
  - Make it obvious / safe for others to understand and use
- Key goal
  - Don't have it break your or other people code
  - High performance if needed
  - No mistakes

IMPLEMENTATION INDEPENDENT THINKING

Generic (system independent)

- Like riding any bike
- Loops (while, do, for, until)
- Conditions (if, else, switch)
- Variables (int, float, class)

Proprietary (tied to specific system)

- Like learning to pilot new type of vehicle
- Input, Output (console joystick, mouse, screen)
- AR/VR/Mobile

WORKSHOP 1
A FIRST LOOK AT UNREAL

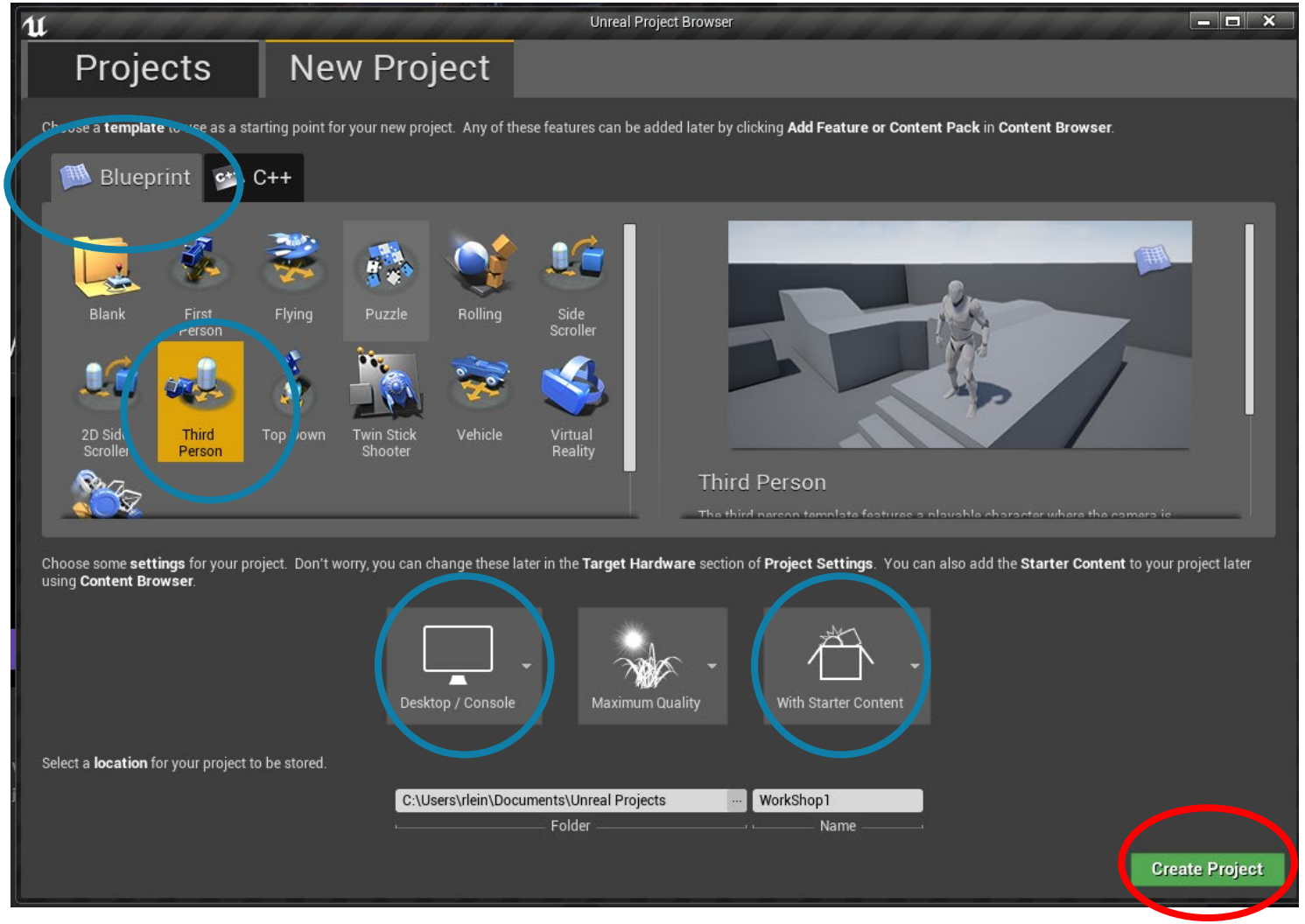
Live Demo, 3rd person Blueprint Game with Starter Content

Blueprints are a visual programming language and unlike code its very hard to learn it by reading

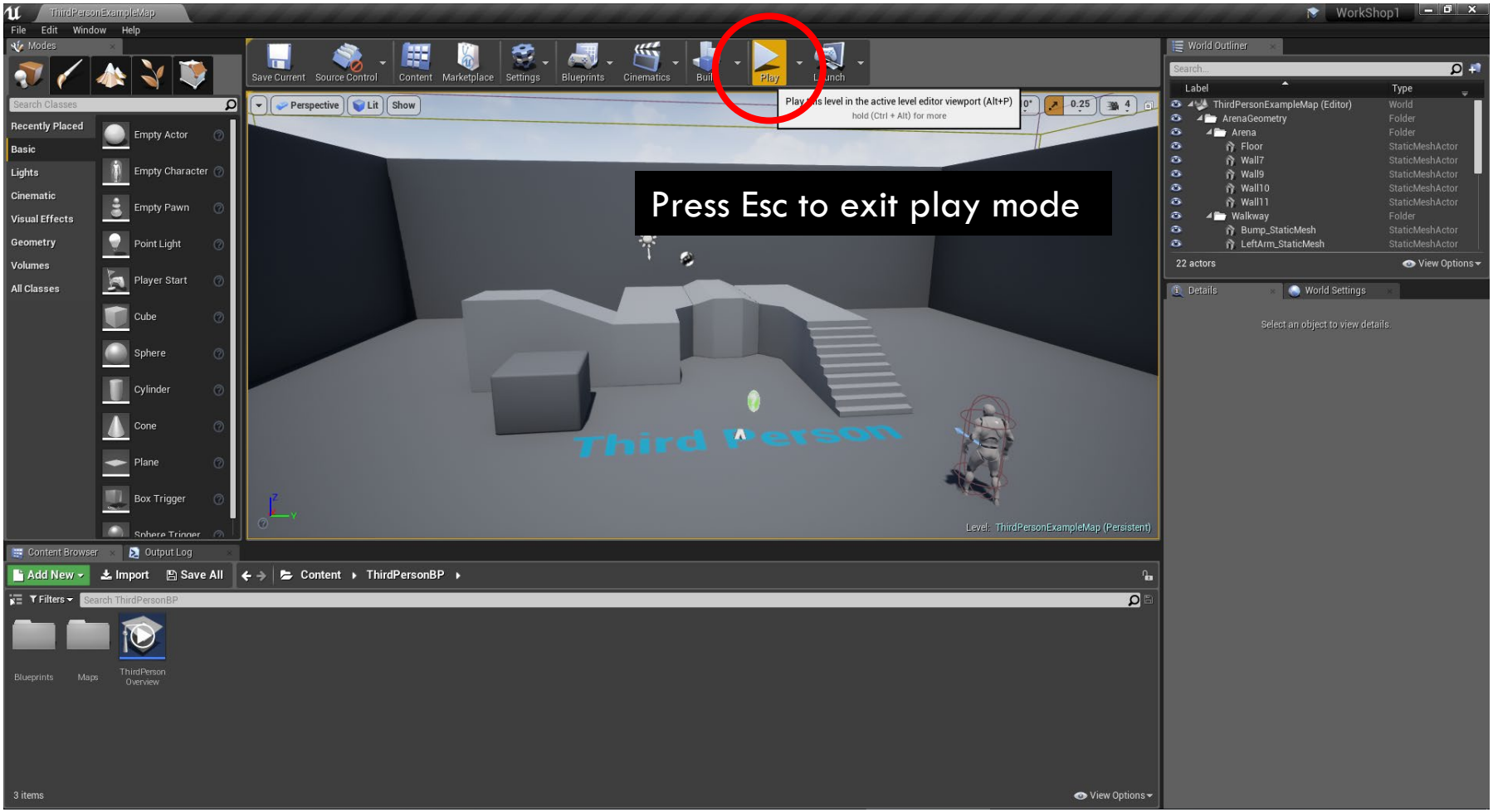
You need to be hands on

LAUNCH UNREAL

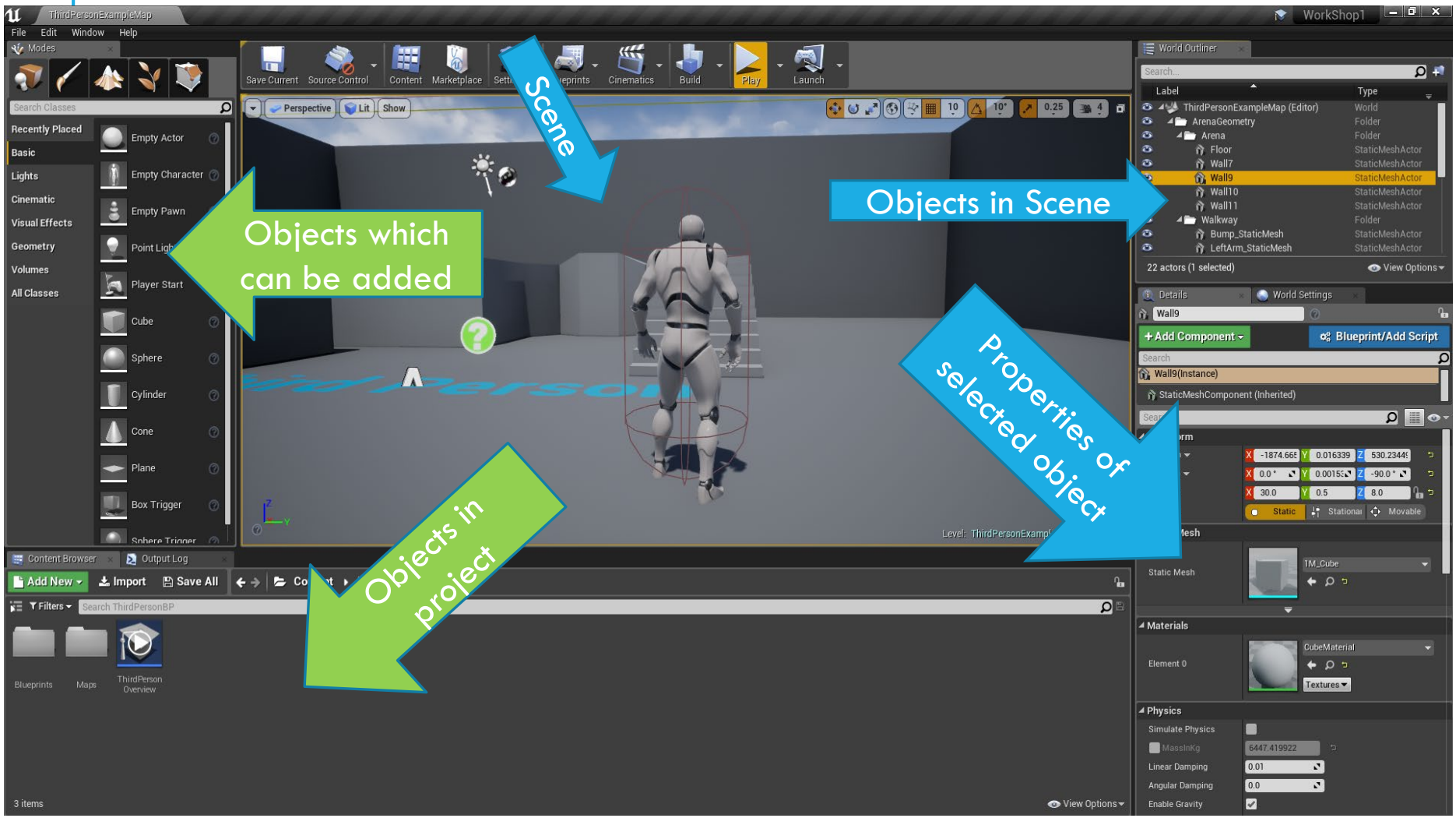
- Blueprint
- Third Person
- Desktop/Console
- With Starter Content
- Name it WorkShop1
- Create



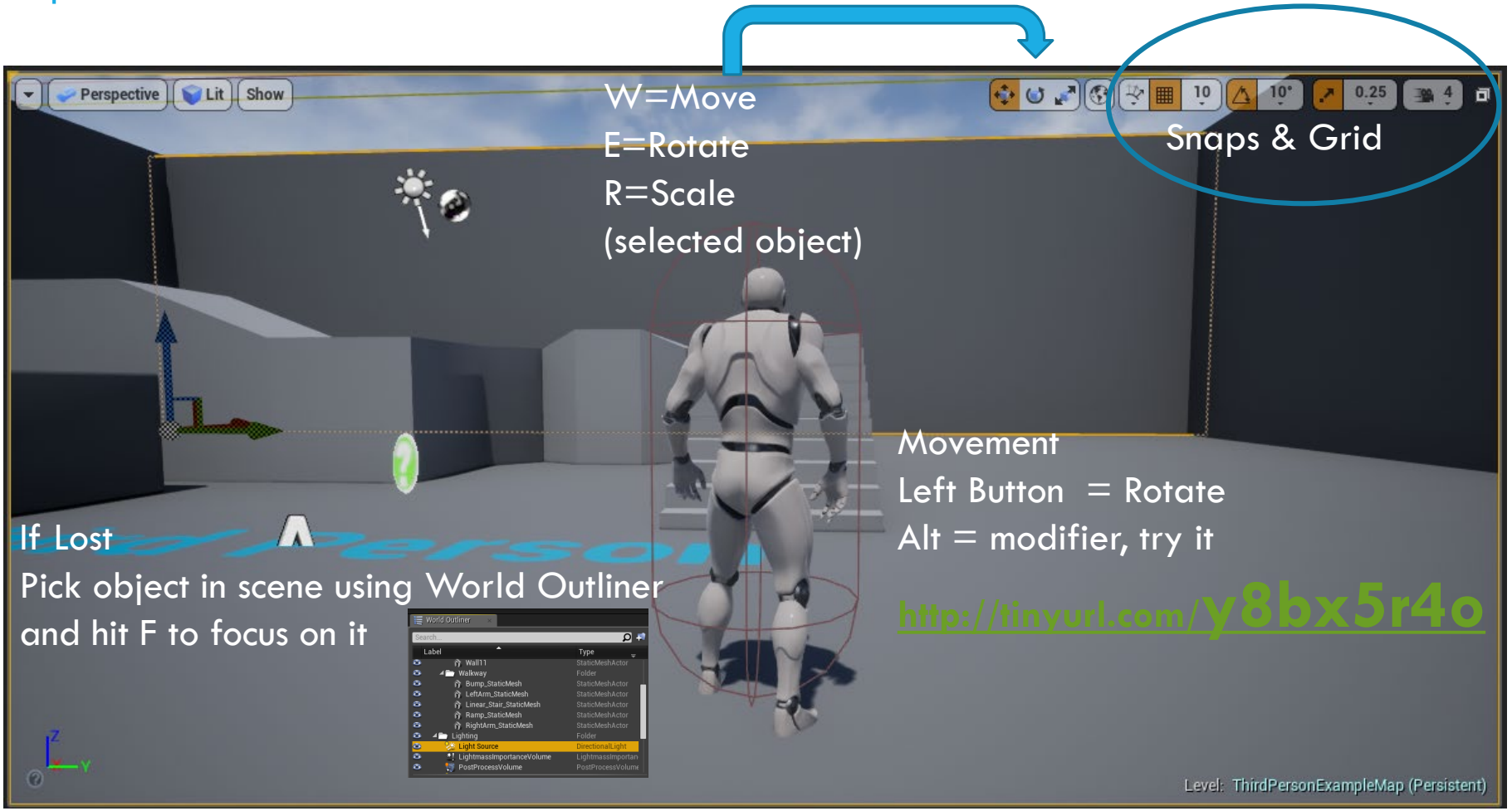
TRY IT OUT


ThirdPersonExampleMap
WorkShop1
File Edit Window Help
Modes
Save Current
Source Control
Content
Marketplace
Settings
Blueprints
Cinematics
Play
Launch
Perspective
Lit
Show
Press Esc to exit play mode
Third Person
Level: ThirdPersonExampleMap (Persistent)
World Outliner
Label
Type
ThirdPersonExampleMap (Editor)
ArenaGeometry
Arena
Floor
Wall7
Wall9
Wall10
Wall11
Walkway
Bump_StaticMesh
LeftArm_StaticMesh
22 actors
View Options
Details
World Settings
Select an object to view details.
Recently Placed
Basic
Lights
Cinematic
Visual Effects
Geometry
Volumes
All Classes
Empty Actor
Empty Character
Empty Pawn
Point Light
Player Start
Cube
Sphere
Cylinder
Cone
Plane
Box Trigger
Content Browser
Output Log
Add New
Import
Save All
Content
ThirdPersonBP
Filters
Search ThirdPersonBP
Blueprints
Maps
ThirdPerson Overview
3 items
View Options

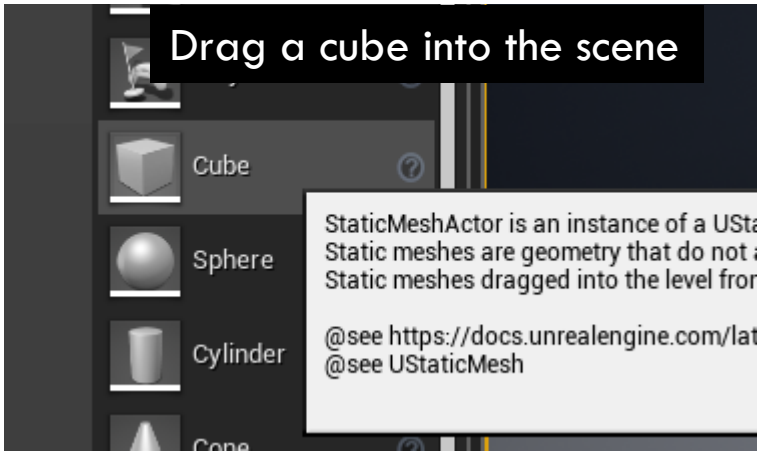
THE EDITOR

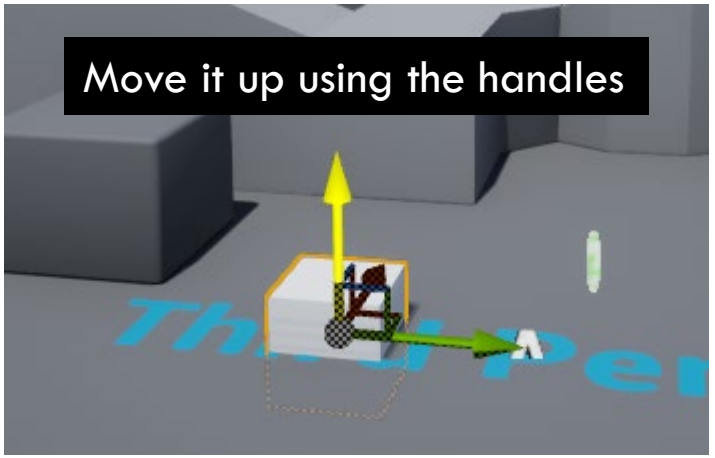

Scene
Objects which can be added
Objects in Scene
Properties of selected object
Objects in project

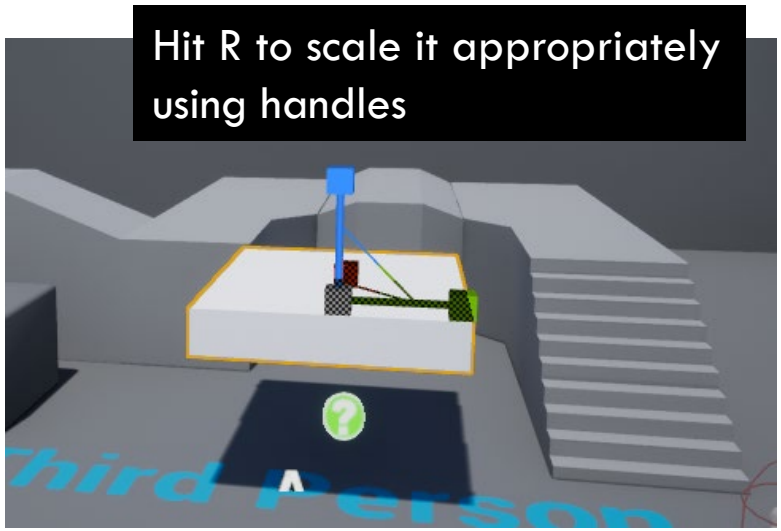
NAVIGATION


Perspective
Lit
Show
W=Move
E=Rotate
R=Scale
(selected object)
Snaps & Grid
Movement
Left Button = Rotate
Alt = modifier, try it
http://tinyurl.com/y8bx5r4o
If Lost
Pick object in scene using World Outliner
and hit F to focus on it
Level: ThirdPersonExampleMap (Persistent)

ADD A NEW PLATFORM

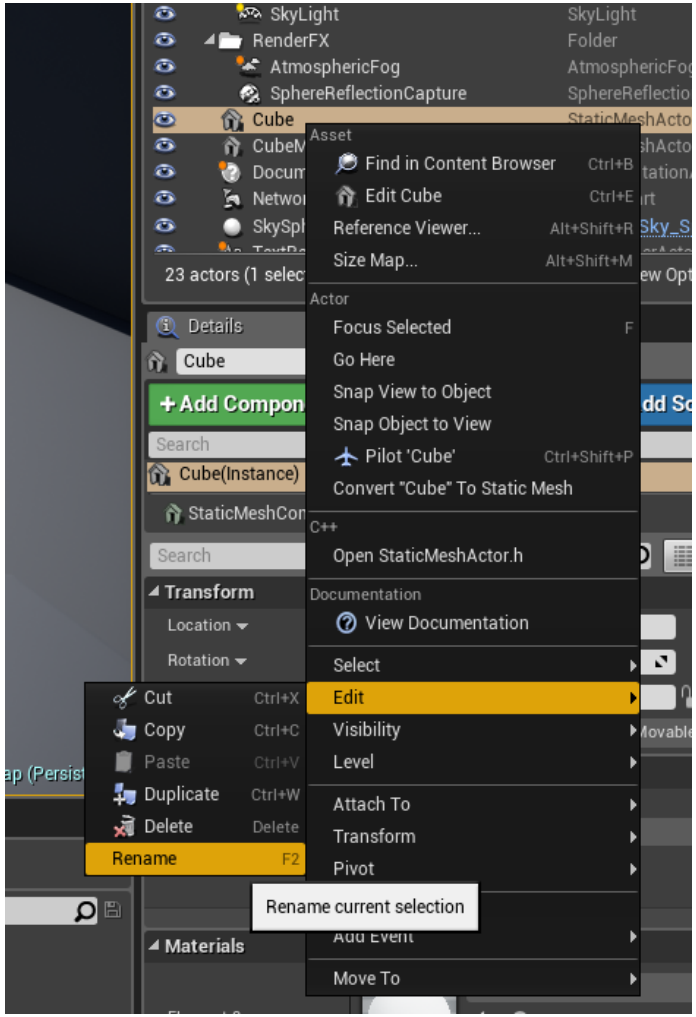






TEST IT!
Can you jump on platform
Space to jump

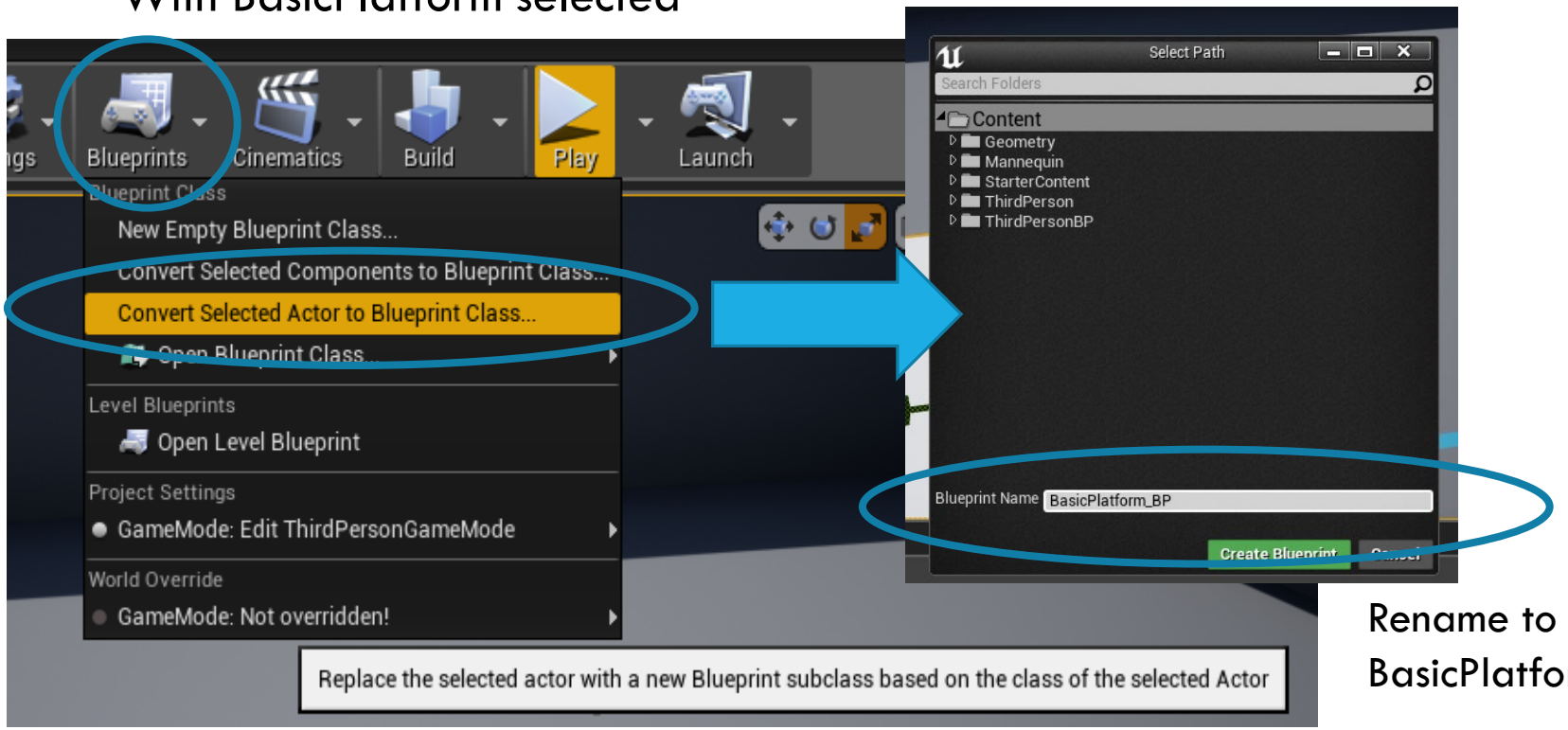
HOUSE KEEPING



Rename Cube as “BasicPlatform”

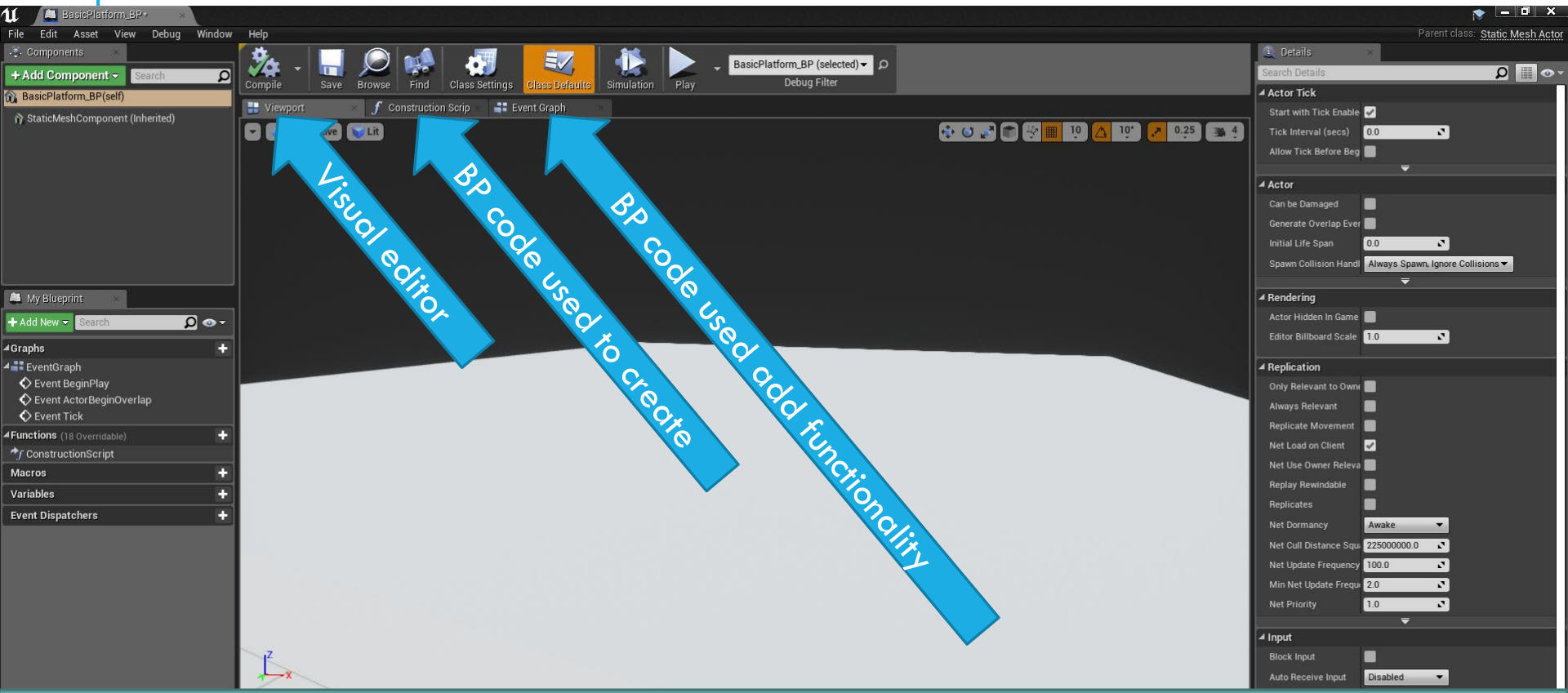
MAKING A BLUEPRINT (RE-USEABLE COMPONENT)

With BasicPlatform selected



Rename to BasicPlatform_BP

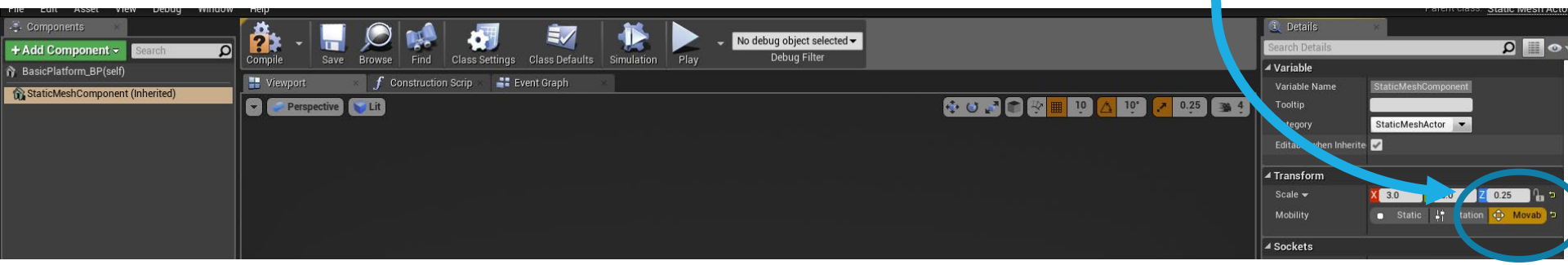
THIS WILL OPEN THE BP EDITOR



When using 2 screens put this window on 2nd screen so you can see the scene & BP editors

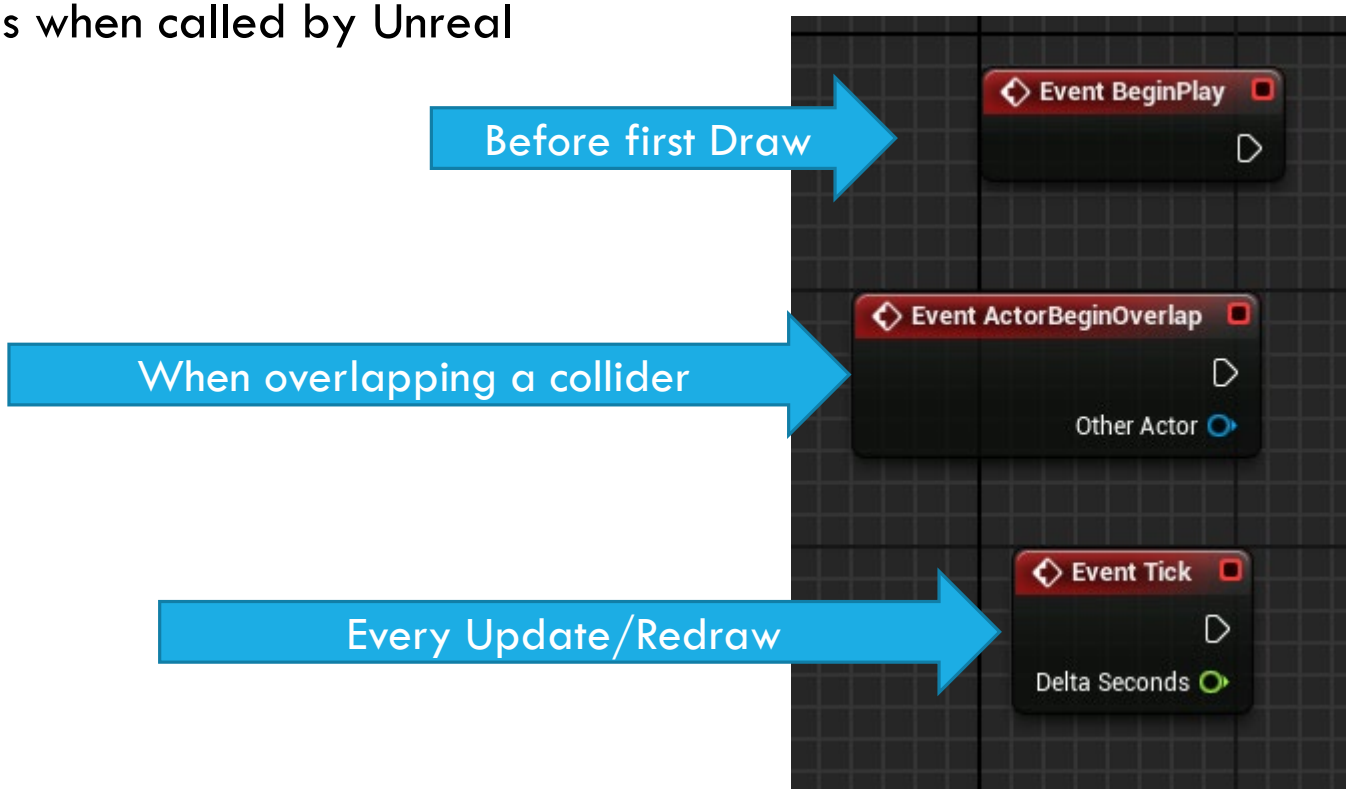
MAKE SURE OBJECT IS MOVABLE

With Static mesh selected change type to moveable



INSIDE THE EventGraph

You code runs when called by Unreal



PROGRAMMING WITH BLUEPRINTS

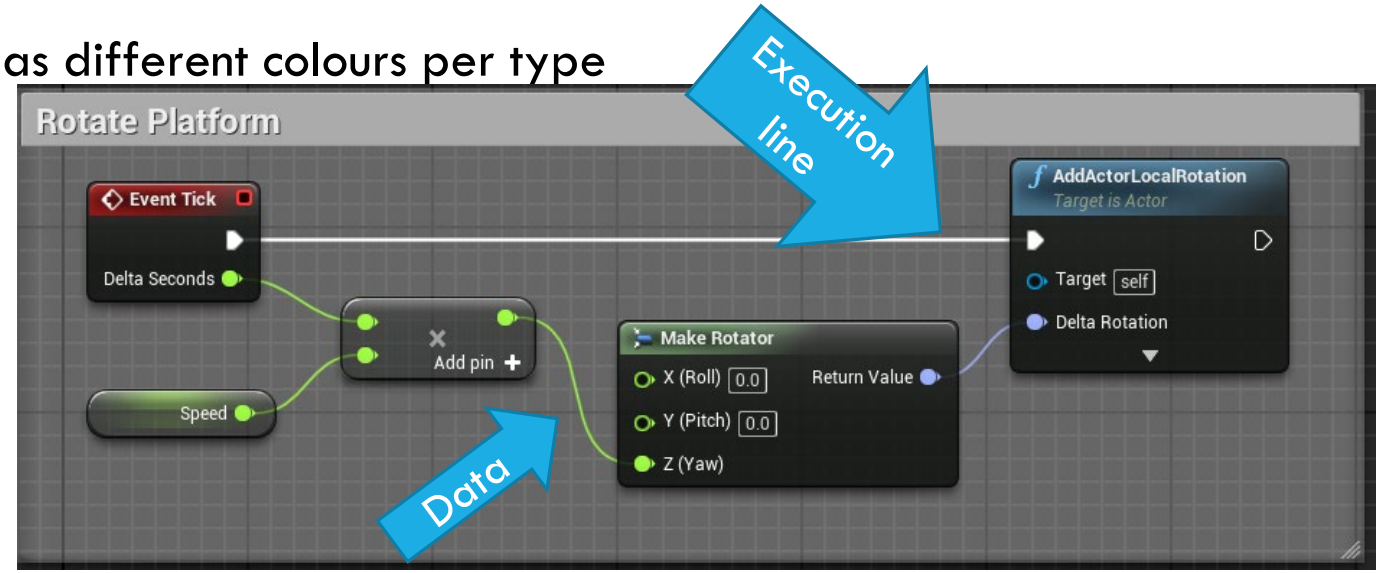
Blueprints are a visual programming language

The editor is context sensitive

So it needs context, as the functions work on data

If the function you are using operates on an object, drag the data out to see everything you can do with it

The data line has different colours per type

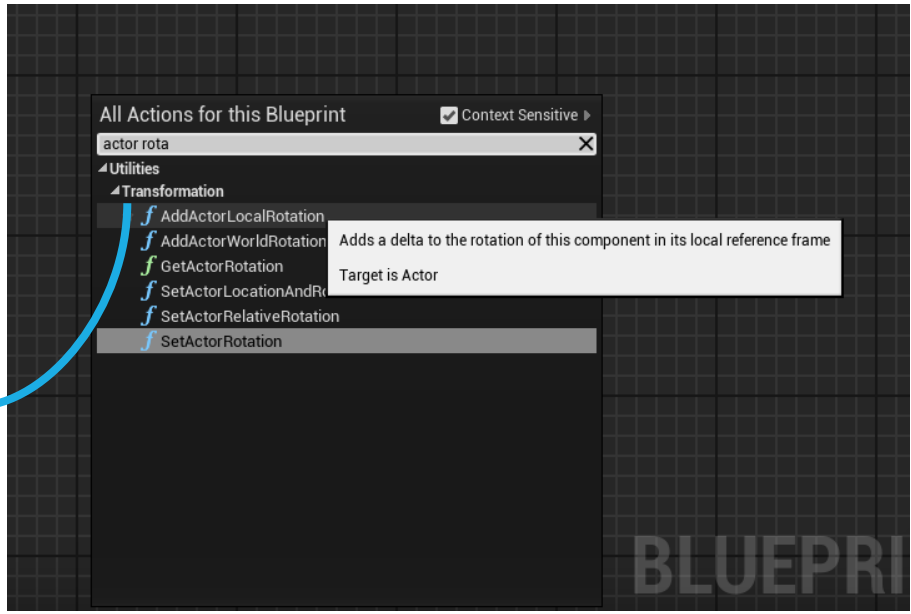


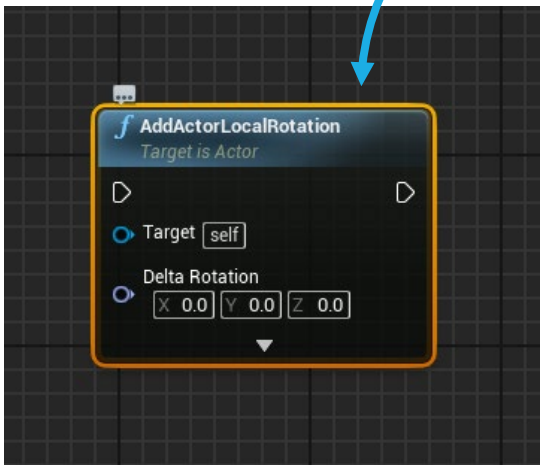
CONTEXT SENSITIVE HELP

In unreal GameObjects are called Actors

Lets assume we want to rotate the platform (Actor).

Type Actor Rotate to see options we will use





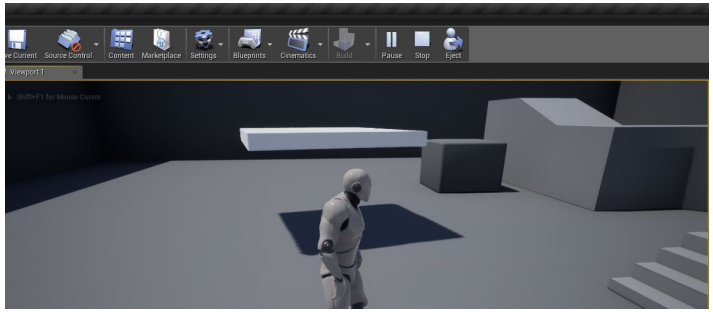
CONNECTING THE EXECUTION LINE

Drag from EventTick to the exec pin on the Rotate Function

Make the Z Rotation 1

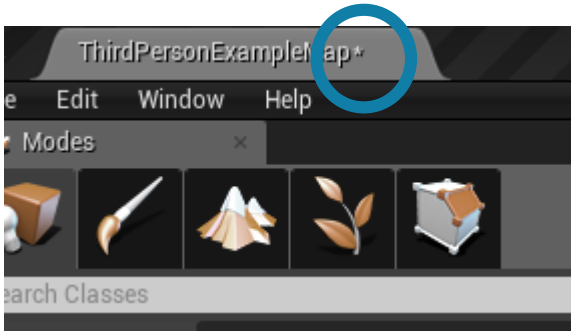
Press play in Scene Editor

Platform should rotate

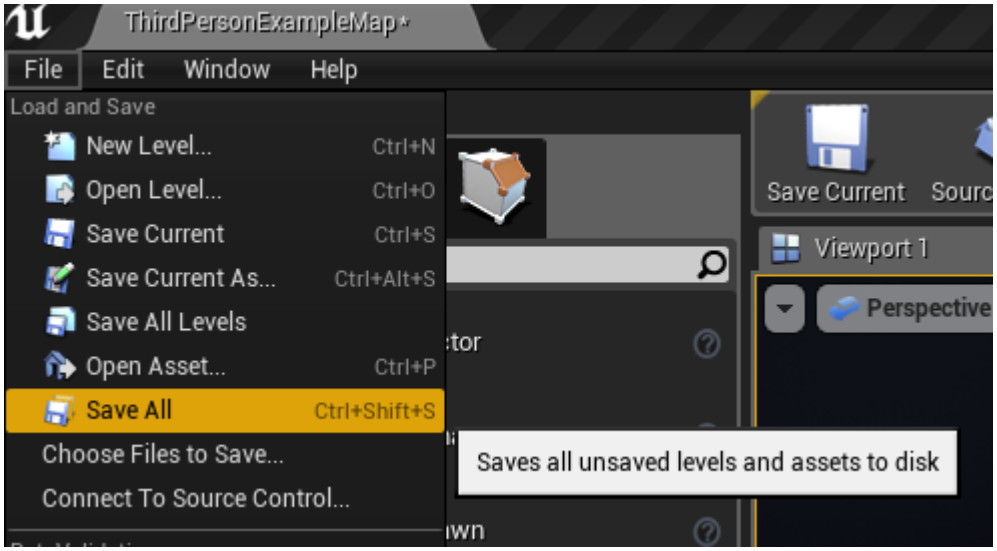


SAVE AS YOU GO

* means unsaved

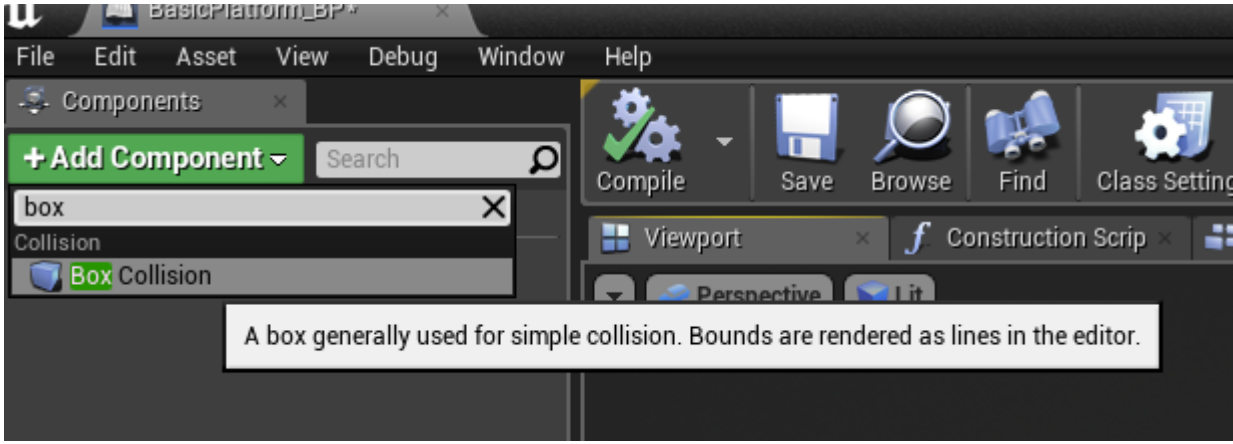


Frequently SaveAll

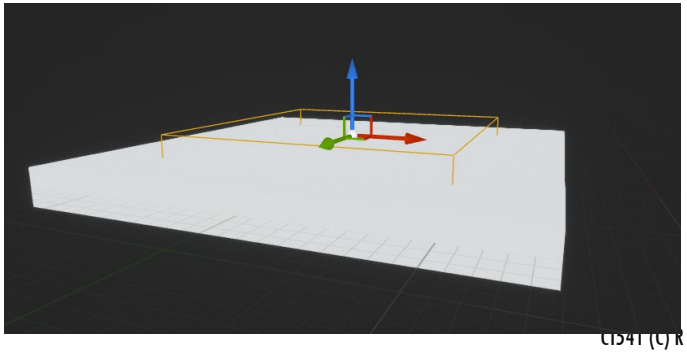


ADDING INTERACTION

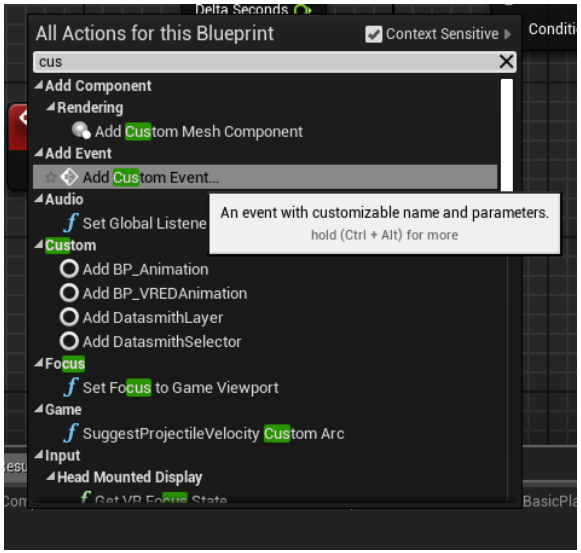
Add BoxCollision component via the Viewport



Drag it just above platform



CUSTOM EVENTS

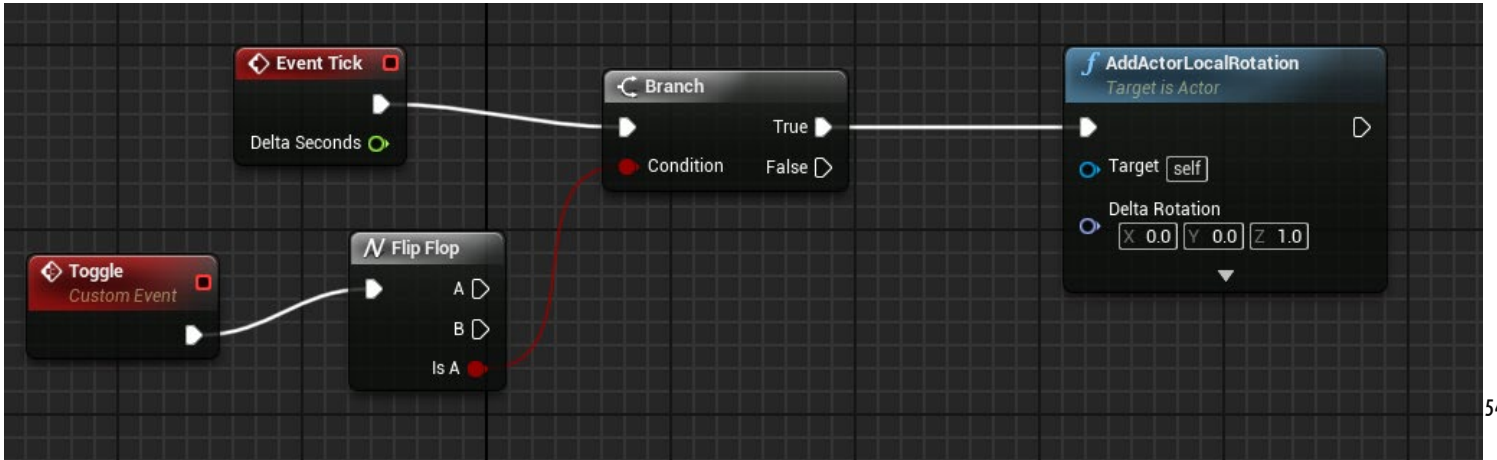


Code which you can call to do your bidding

Add a custom Event rename it Toggle

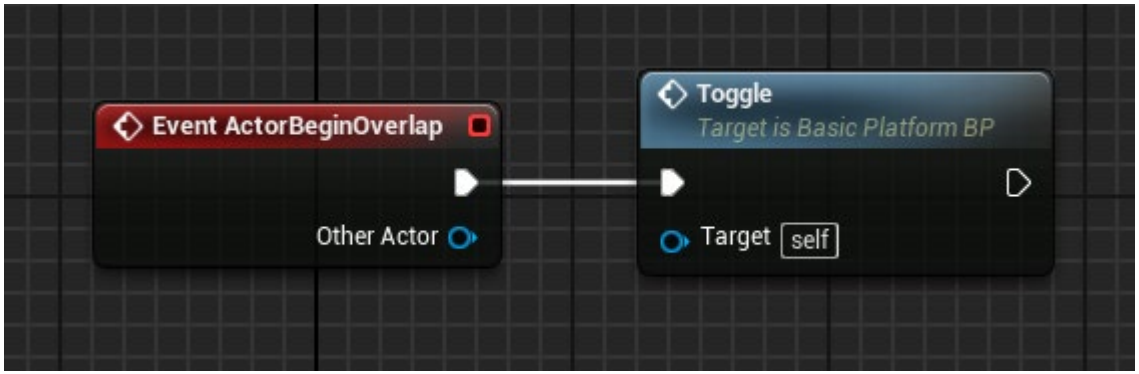
Wire up this EventGraph, just right click and try search for the names of the boxes shown below, make sure all the lines connect

FlipFlop will change from 0-1-0 etc every time its called, it starts and stops the rotation



DETECTING THE PLAYER

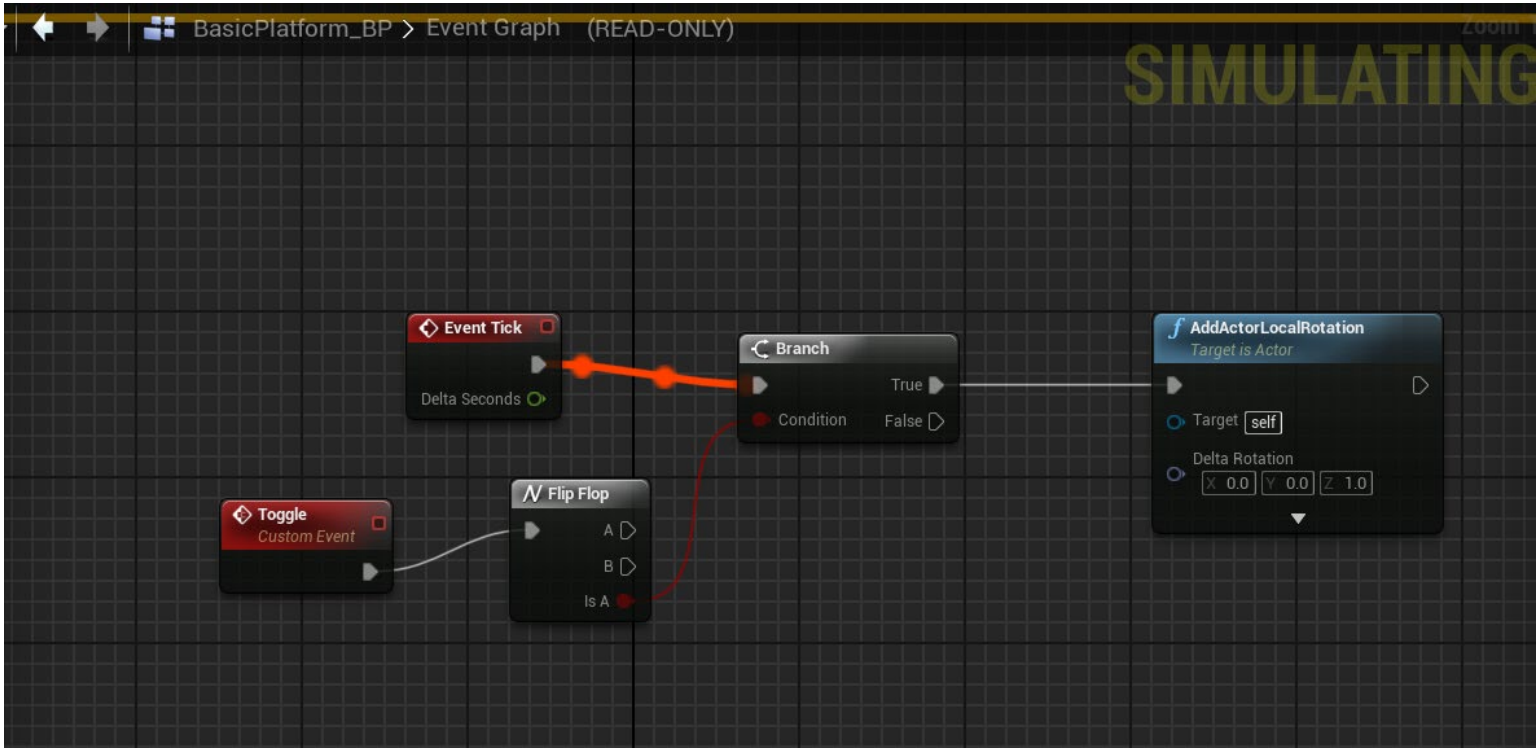
Once we have added the collider we can use its ActorOverlap event to trigger and stop the rotation by calling our Toggle Code



Test your code, the platform should rotate when stepped on and stop when stepped on again

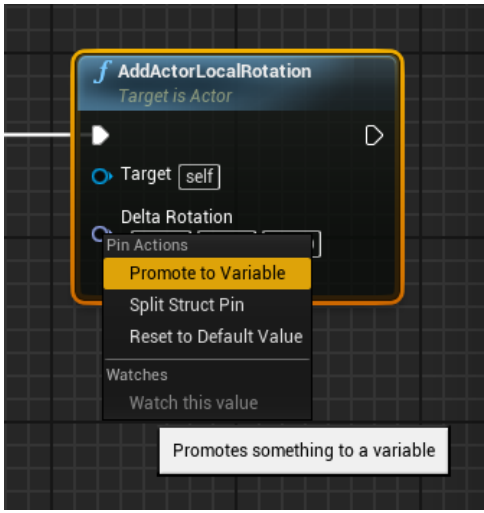
RUN THE CODE AND VIEW THE BP

The BP will animate as code executes, great for testing



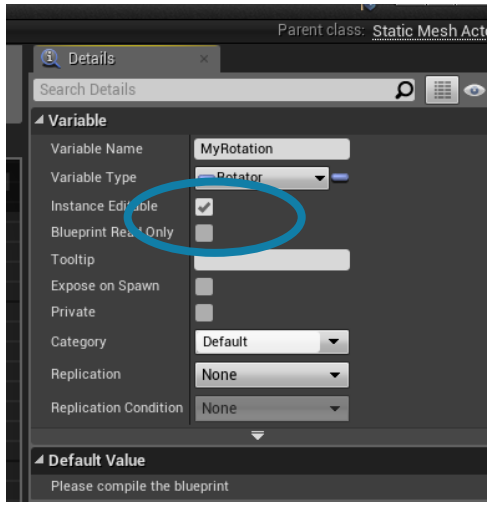
ADD MORE ROTATING PLATFORMS

Make a level by scaling the platforms which the player can only traverse by starting and stopping them



Note how it would be nice to allow the platforms to rotate in different directions/speeds

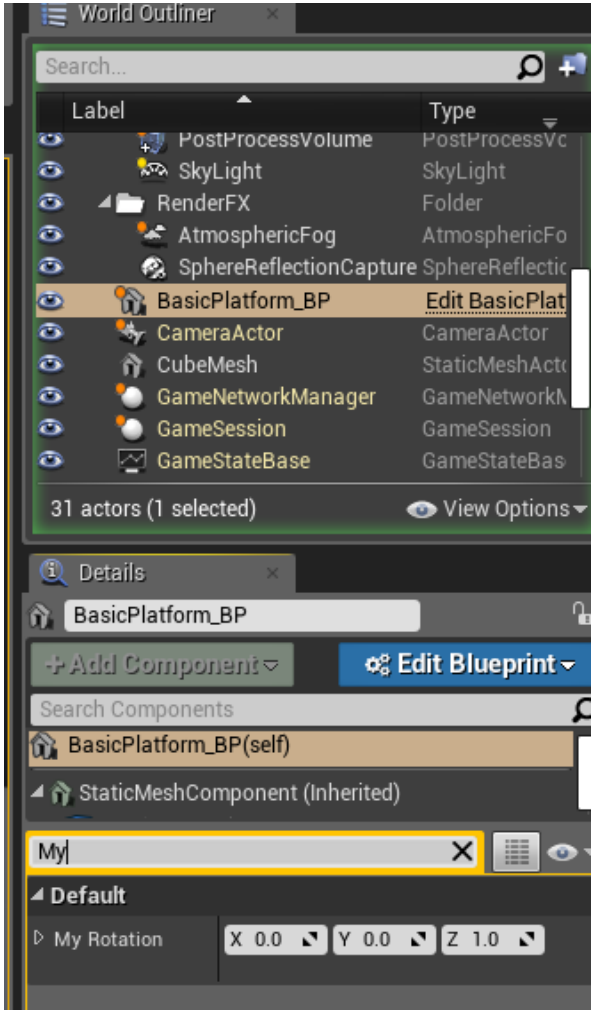
We can do this by making the rotation a variable, right click the Delta rotation pin and promote to variable rename it MyRoration & make it instance editable



THE VARIABLE WILL NOW SHOW UP

Check the details panel for the BasicPlatform

If you have clicked in the gameworld you may need to free the mouse with Shift-F1 to be able to select the Item in the WorldOutliner
Change the speed to 10, or even try a new axis of rotation



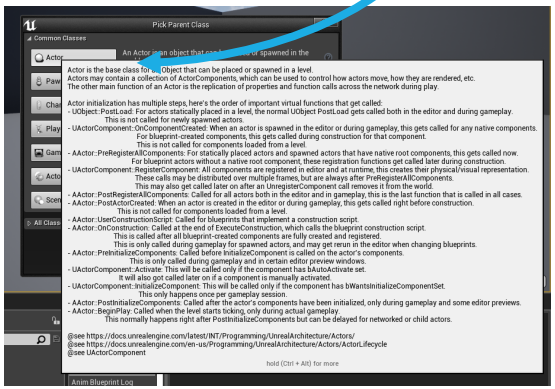
2 BP'S COMMUNICATING

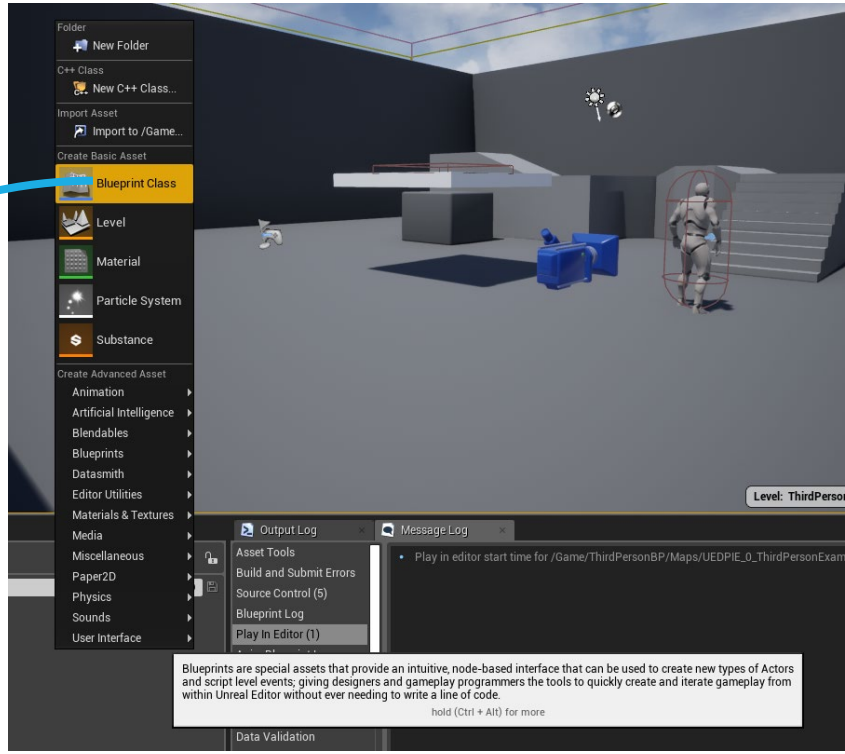
It would be better to separate the collider from the rotating object to act as a remote trigger

To do this we will build a trigger Object, which is just a sphere you walk into

We will do this from scratch

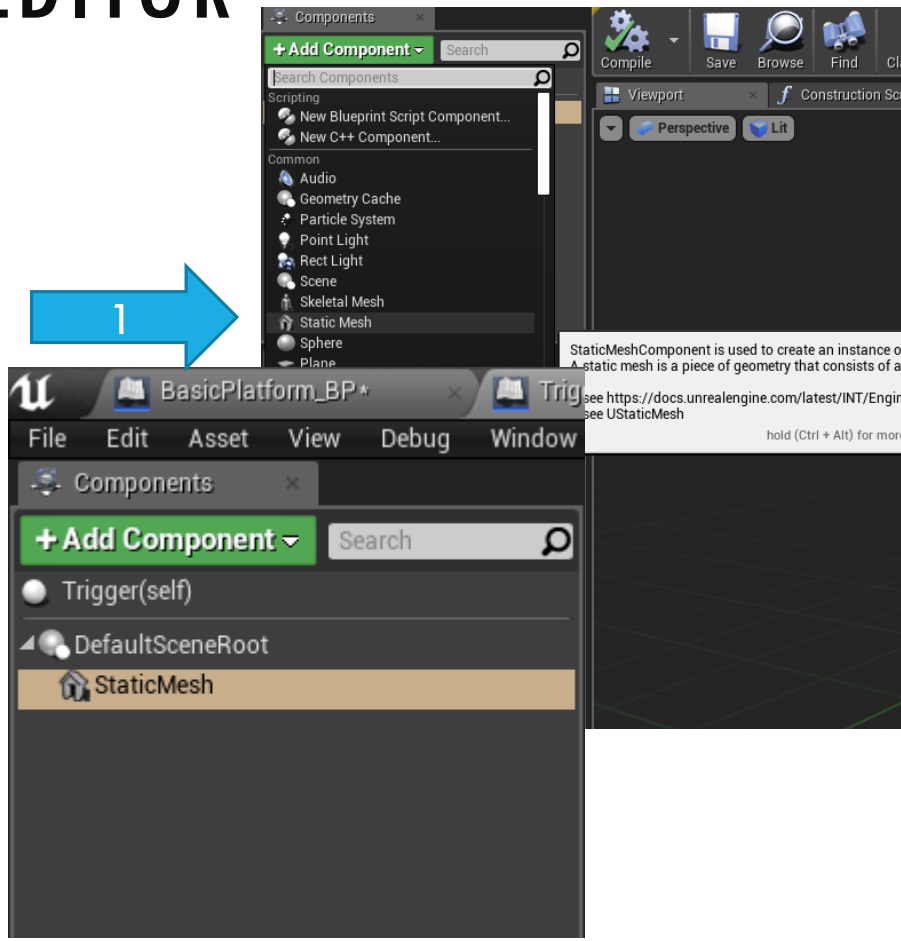
Right click inside the Content Window and make a new ActorBlueprint call it Trigger

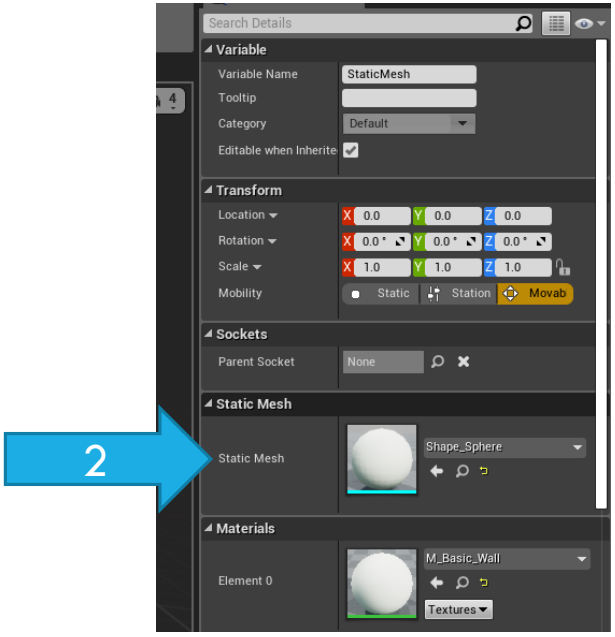




OPEN IT UP IN THE BP EDITOR

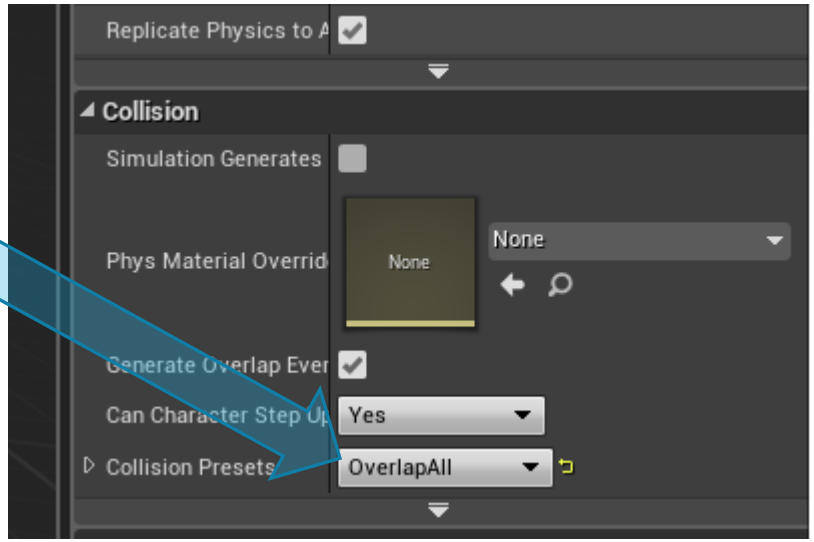
1. Lets add a Static mesh component
2. Make it into a sphere





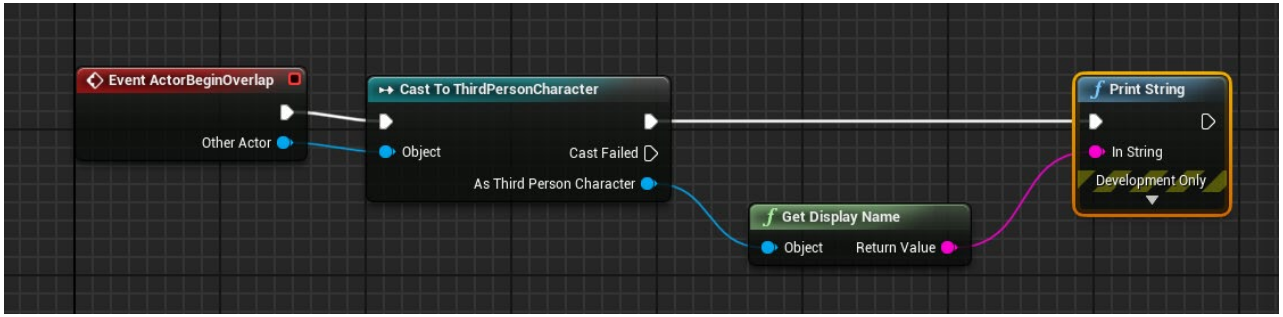
MAKE SURE WE CAN WALK THROUGH IT

Set its Collision to OverlapAll



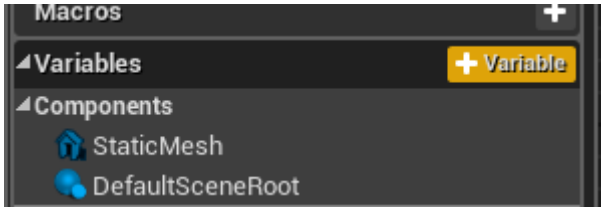
Add the code below to the EventGraph

Hint: Use the context sensitive help



Drag the item into the scene & test, you should get a message

LINKING THE OBJECTS

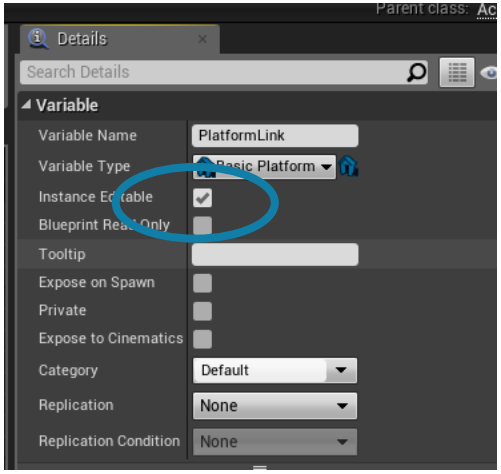


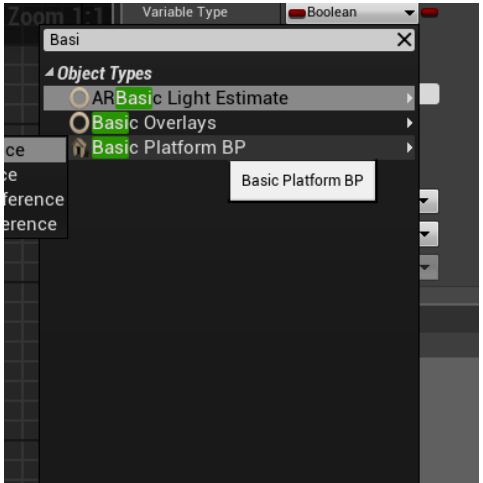
We will use an object reference

Make a new variable

Change its type to BasicPlatformBP
ie the platform BP you made before
Rename to PlatformLink

And make it InstanceEditable

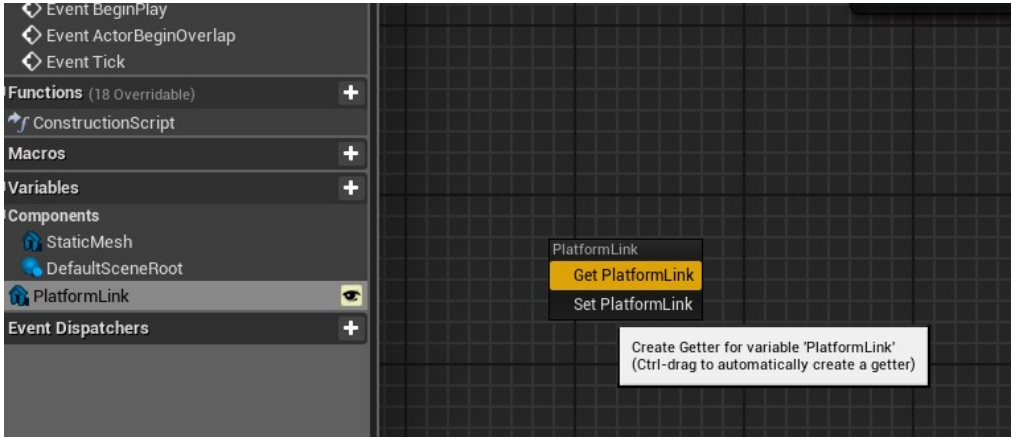


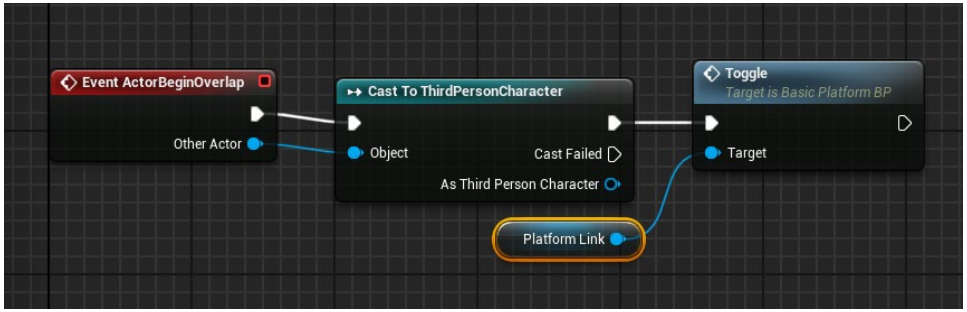


EDIT THE CODE

Get the Variable, to do this drag from the side into the BP and select Get

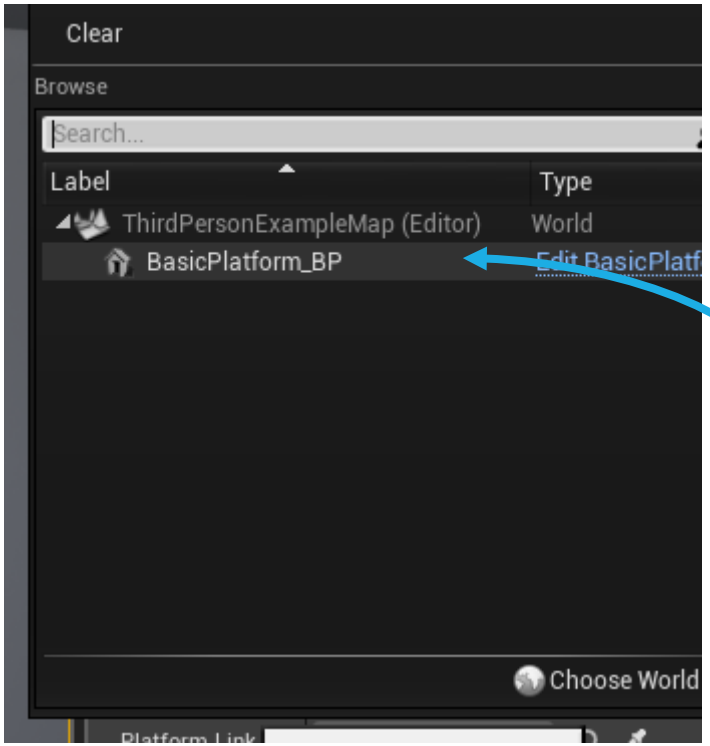
Amend the BP to show, note how you can call the event on the Platform by using its Object Reference

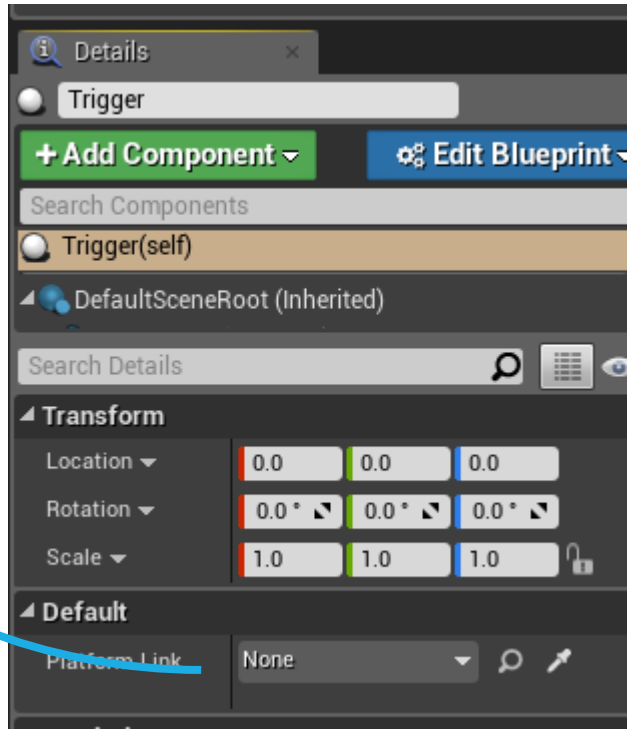




NOW WE JUST NEED TO LINK THEM

On the trigger link it to the platform you want to trigger & test



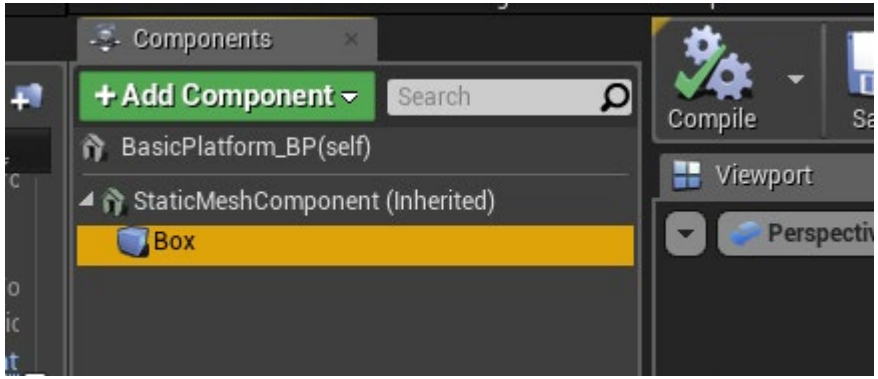


WE CAN NOW TIDY THE PLATFORM CODE

Lets remove the old collider and EventOverlap

Delete the Box

Delete the call to Toggle



MAKE A PLAYABLE LEVEL

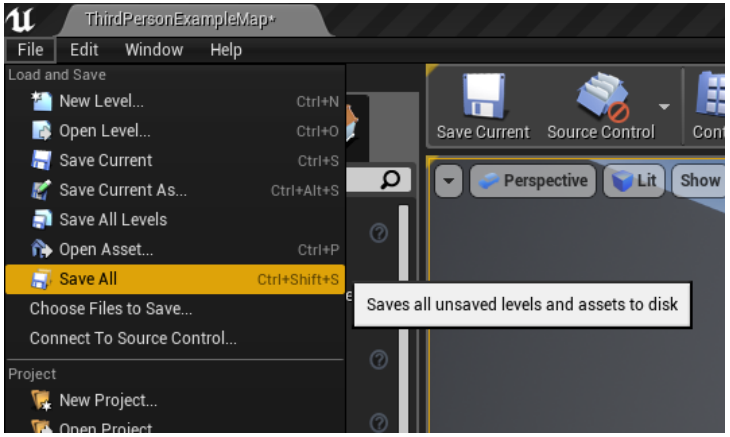
Use triggers & colliders to allow the player to get to s normally unreachable location

Impress your colleagues

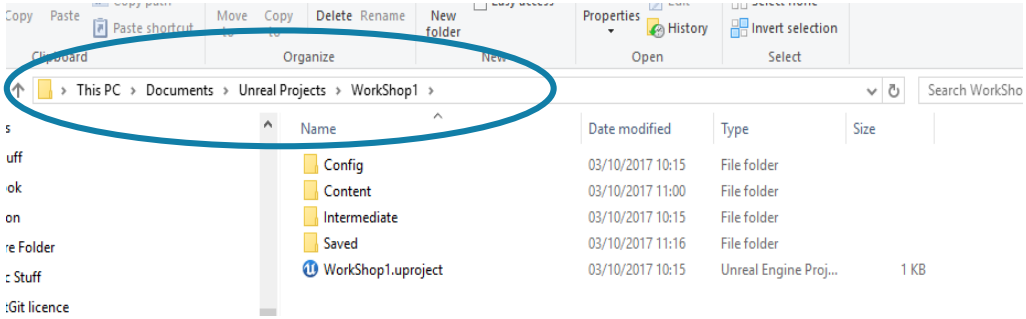
Help: Rotation BP https://blueprintue.com/blueprint/tbupb0lx/

Trigger BP: https://blueprintue.com/blueprint/asm6359x/

SAVE ALL



1. Save All
2. Find



3. Copy this whole folder to your Onedrive or memory stick for next week