**Week 10 – GameMode & Respawning**

*The Gamemode is a class that manages the state of the game, it is generally responsible for the game starting and ending, spawning/respawning of characters and tracking win/loss conditions.*

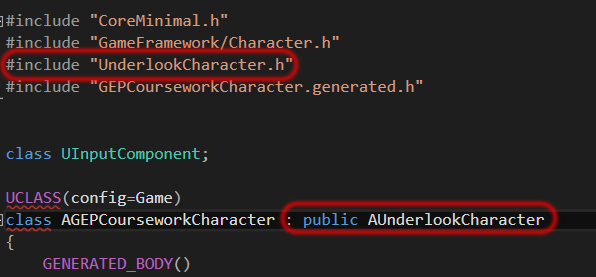
*Our base Gamemode class will have this functionality but we could also create derived Gamemodes with more specific rules, for example if we wanted a capture the flag Gamemode.*

*In this tutorial we will be creating a basic deathmatch Gamemode.*

**Section 1 – Setting up teams**

*Even with single player games it is a good idea to have some kind of team ID so we can distinguish friend from foe. NPC\_Base and our Character classes both derive from ACharacter, which is an engine file so is read only, we need to create a class between ACharacter and our classes that they both derive from to add common functionality. i.e ACharacter->****AUnderlookCharacter****->ANPCBase*

1. Create a new C++ class derived from Character
2. Name it something that makes sense, for me it was “UnderlookCharacter”
3. Open up your NPC\_Base and your Character header files and add an include to it above the generated include
4. Change the parent class definition to your new class for both of them
5. Compile and nothing should change and you should have no errors

**

1. Add a protected variables:

UPROPERTY(EditAnywhere, BlueprintReadOnly, Category = Gameplay)

int TeamID;

UPROPERTY(BlueprintReadWrite, Category = Gameplay)

bool CanRespawn = true; //default to true until you attempt the challenge.

1. And these private variables

bool WaitingToRespawn;

float RespawnTime;

1. Create these public functions:

UFUNCTION(BlueprintCallable)

int GetTeamID();

UFUNCTION(BlueprintCallable)

void SetTeamID(int newTeamID);

UFUNCTION(BlueprintCallable)

void StartRespawnTimer(float timeToRespawn);

1. Create definitions for the functions that get and set the TeamID.
2. The define the respawning functions but leave empty for now, you will have to figure out what to do as the challenge at the end.

**Section 2 – Spawner setup**

1. Create a new C++ class called Spawner
2. In the header add an integer for this spawners team

UPROPERTY(EditAnywhere, BlueprintReadOnly, Category = SpawnSettings)

int TeamID;

*This must be EditAnywhere so we can set place it in the scene for a specific team.*

1. Add a public function called SpawnActor

UFUNCTION(BlueprintCallable)

virtual AActor\* SpawnActor(TSubclassOf<AActor> BPToSpawn);

1. In the .cpp and the include

#include "Engine/World.h"

1. Add the definition for the SpawnActor function

AActor\* ASpawner::SpawnActor(TSubclassOf<AActor> BPToSpawn)

{

AActor\* spawnedActor = GetWorld()->SpawnActor<AActor>(BPToSpawn,GetActorTransform());

return spawnedActor;

}

1. Compile
2. Create a blueprint class derived from spawner called Spawner\_BP
3. Drag 3 of them into the scene, set one of the to team 1 and the other two to 2.
4. Rename them something helpful so you remember what team they are for.

**Section 3 – GameMode setup**

1. In visual studio find the gamemode that was generated when we first created out project, for me it is called GECCourseworkGameMode and open the header
2. Add the following public function declarations

//called from level blueprint when placed assets have been assigned. i.e spawners or pre-placed enemies.

UFUNCTION(BlueprintCallable)

void LevelSetupComplete();

UFUNCTION(BlueprintCallable)

void CharacterKilled(class AUnderlookCharacter\* killedCharacter, AActor\* killedBy);

UFUNCTION(BlueprintCallable)

void GameOver(int winningTeamID);

UFUNCTION(BlueprintCallable)

void ObjectiveComplete(int teamID, AUnderlookCharacter\* completedBy);

*These are what we can call from anywhere in the game, mostly they just send off events for blueprint to pick up.*

1. Add the following protected events

UFUNCTION(BlueprintImplementableEvent)

void OnCharacterKilled(AUnderlookCharacter\* killedCharacter, AActor\* killedBy);

UFUNCTION(BlueprintImplementableEvent)

void OnGameStart();

UFUNCTION(BlueprintImplementableEvent)

void OnGameEnd(int winningTeamID);

UFUNCTION(BlueprintImplementableEvent)

void OnObjectiveComplete(int teamID, AUnderlookCharacter\* completedBy);

1. In the .cpp have each function definition call its respective event.

For example:

void AGEPCourseworkGameMode::LevelSetupComplete()

{

OnGameStart();

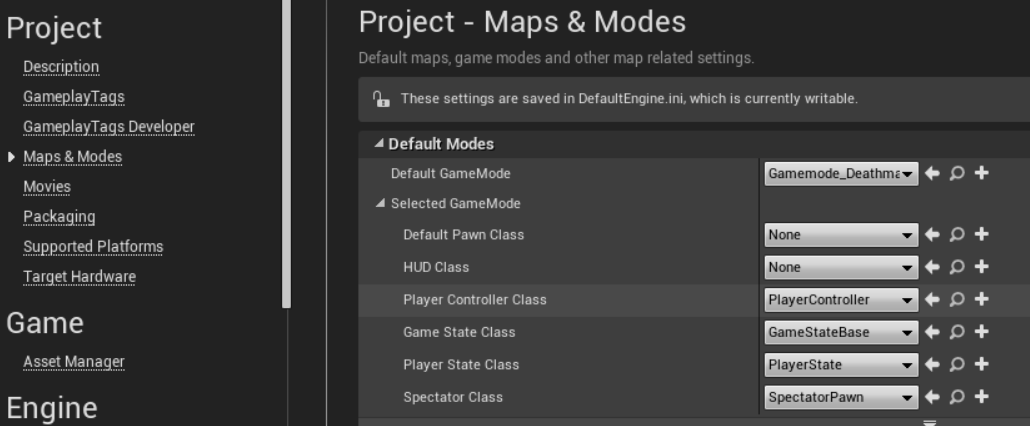
}

1. Compile
2. Back in editor create a new blueprint derived from your gamemode class
3. Call it “GameMode\_Deathmatch”

**Section 4 – Editor Setup**

*We will need to make a few changes to our project, i.e changing the default game mode and pawn.*

1. In editor go to Edit->Project Settings..->Maps & Modes
2. Set the Default GameMode to Gamemode\_Deathmatch and default pawn to None



1. In your NPC blueprint make sure you break the link so that your behavior tree does not run.

*As we set up them to patrol points that we manually assigned, when we spawn them in they will not be able to run that behavior unless they search for the patrol points first.*

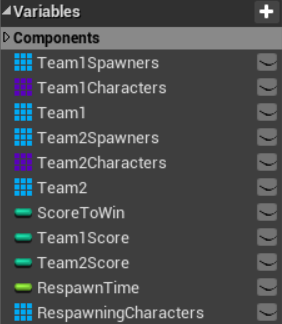
1. Delete the NPC and any characters in the scene.

*We will now be spawning them in instead.*

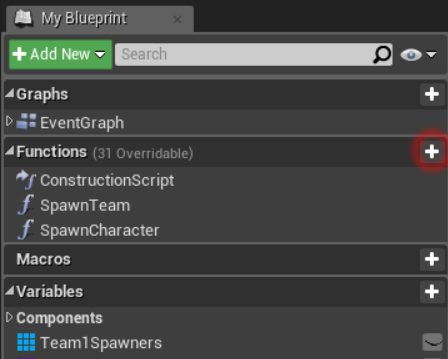
**Section 5 – Deathmatch**

1. Open your Gamemode\_Deathmatch blueprint editor
2. Add the following variables:

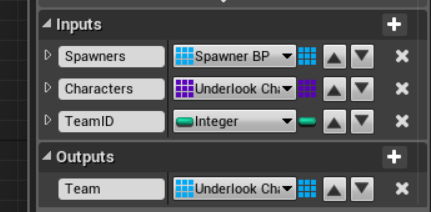
* *Team1Spawners is type Spawner\_BP (object)*
* *Team1Characters is type UnderlookCharacter(****class****) – or whatever you called it.*
* *Team1 is type UnderlookCharacter(object)*
* *RespawningCharacters is type UnderlookCharacter(object)*



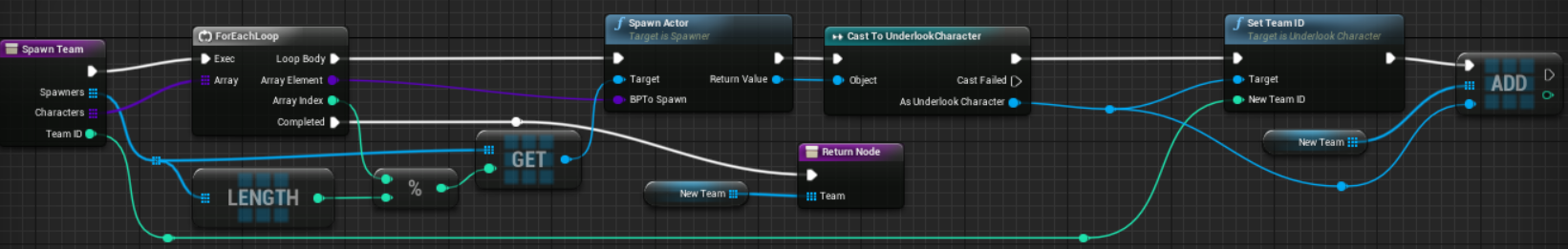
1. Create a new function called SpawnTeam by clicking the + symbol shown below.



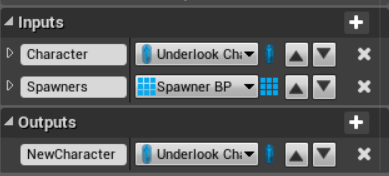
1. On the right panel create the following inputs and output

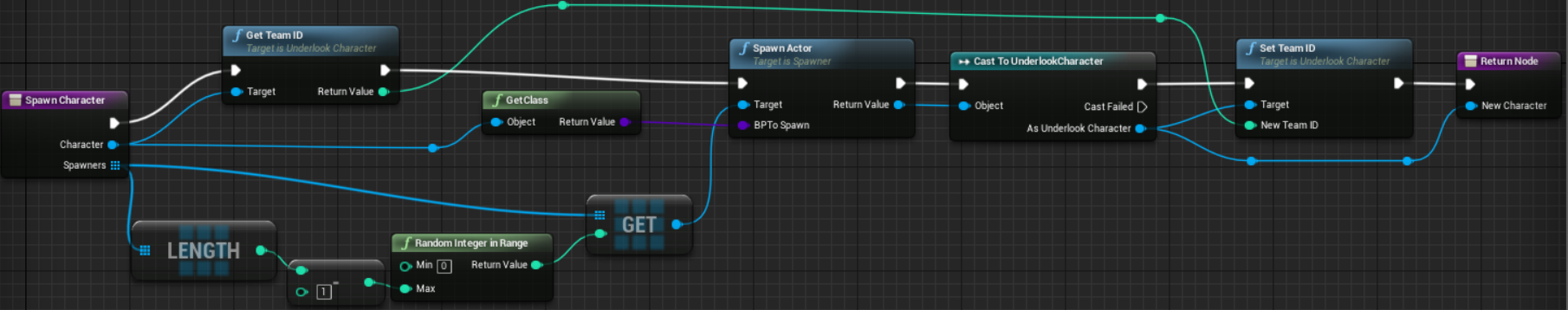


1. Add the following blueprint script *(hold ctrl and scroll wheel to zoom in on MS word)*

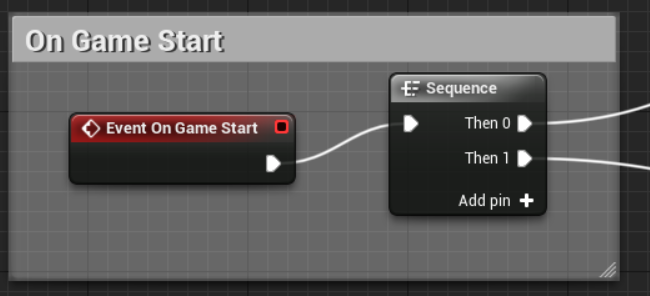


1. Create a SpawnCharacter function as follows:

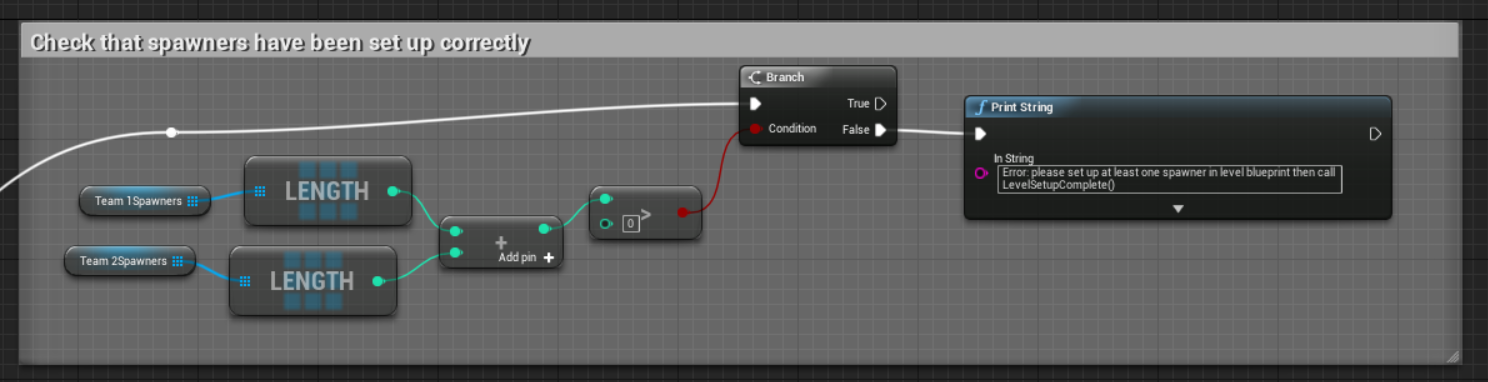




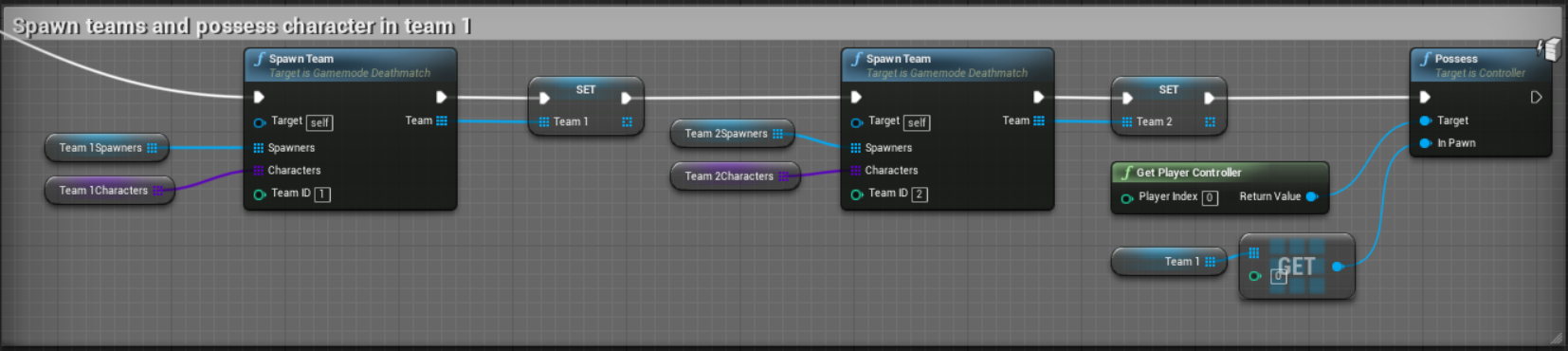
1. Back in the event graph create an OnGameStart node (this is one of the events we created earlier)
2. Create a sequence node with two pins



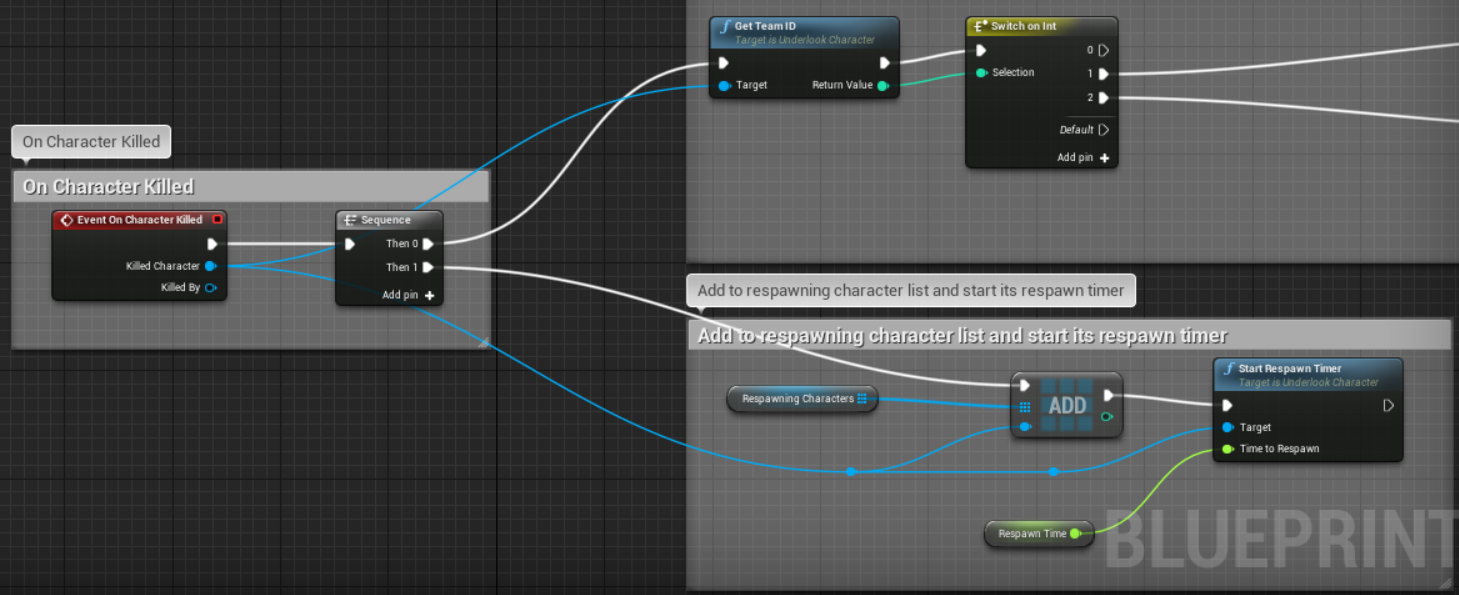
1. From the top pin create the following:



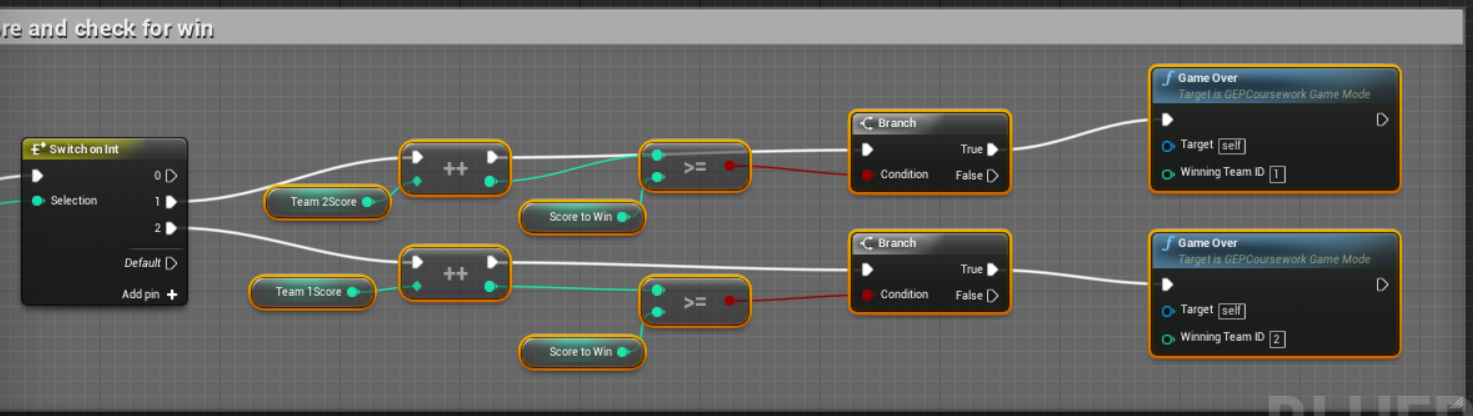
1. From the second pin:



1. Create an OnCharacterKilled node and replicate the following two images



Continued from the switch on int..

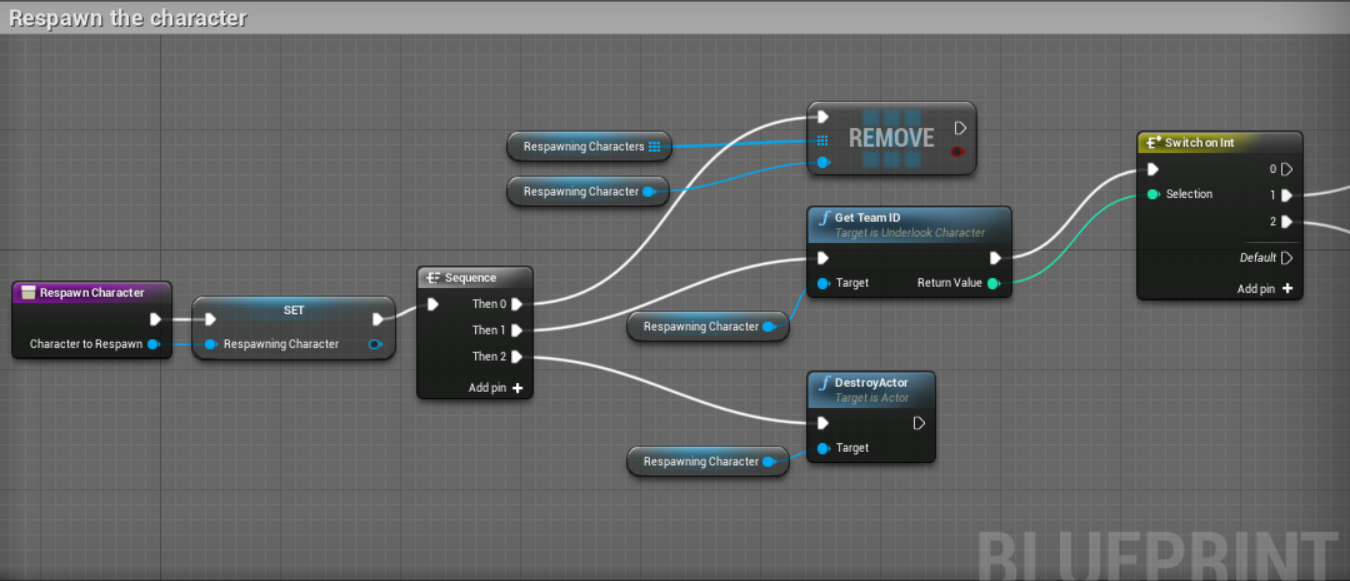


1. Create a new function called RespawnCharacter, it has an input called CharacterToRespawn of type UnderlookCharacter
2. Add a Local Variable called “RespawningCharacter” of type UnderlookCharacter

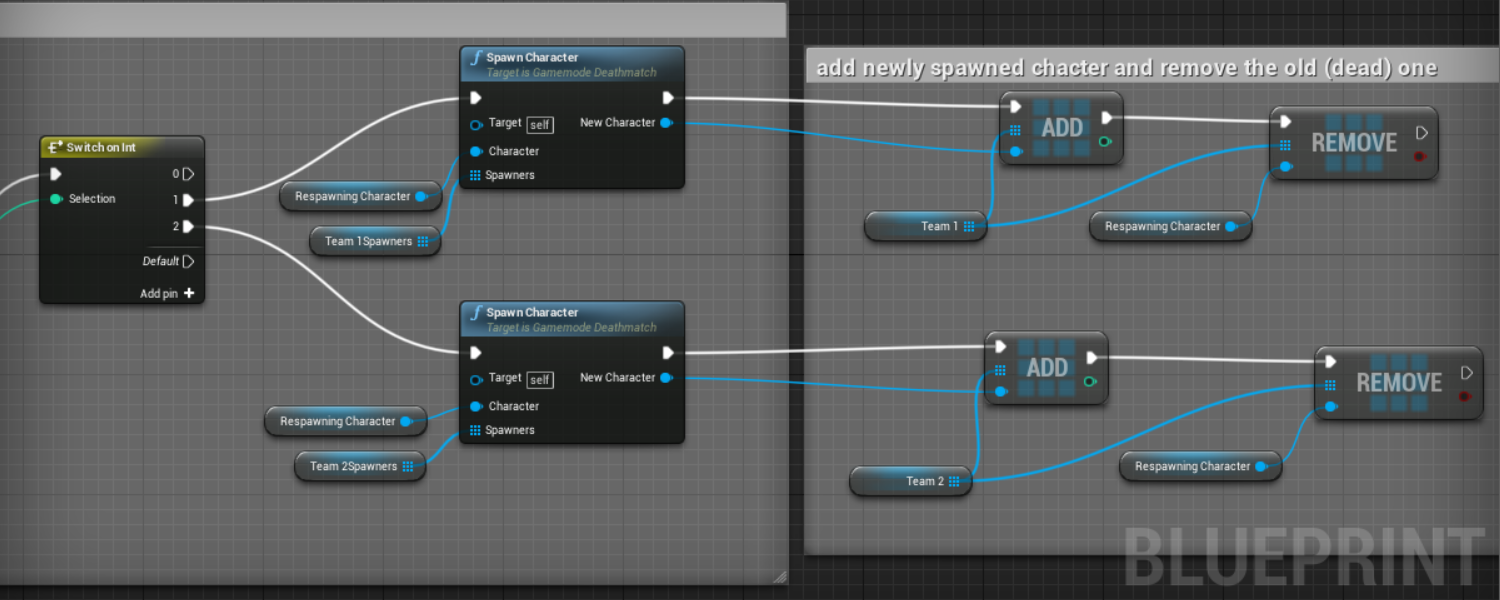
*This is to prevent ‘spaghetti’ code .*



1. Add the following

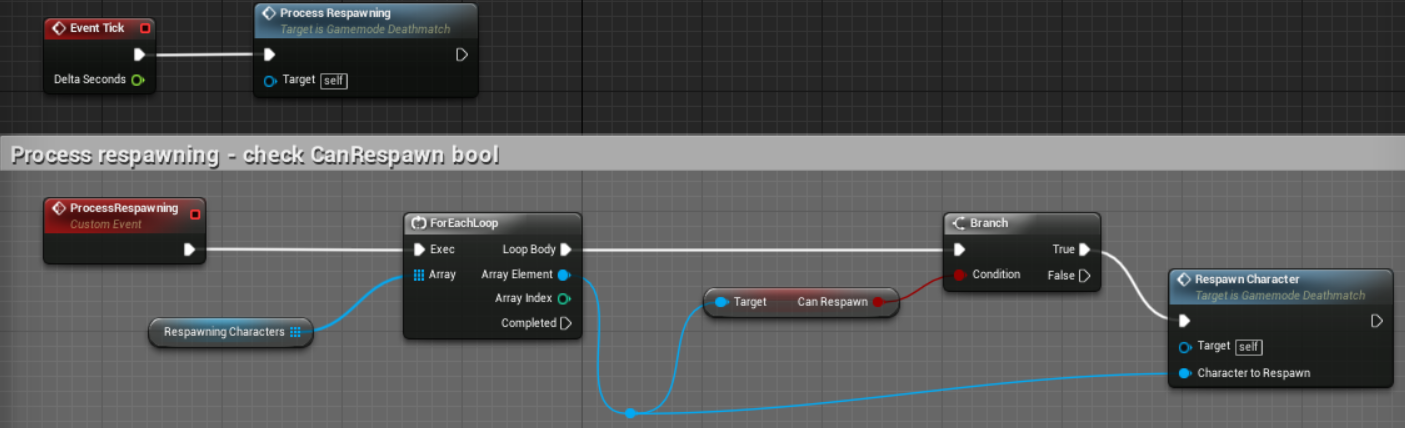


Continued..

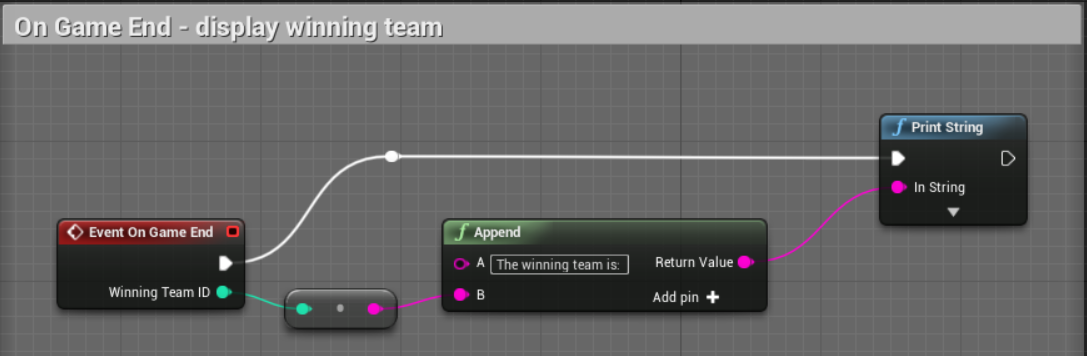


1. Create a new custom event called “ProcessRespawning”
2. Add (or use existing) Tick node and call that event.

*This will iterate through all the currently respawning characters and check if their respawn timers have expired and they can be respawned*



1. Finally add an OnGameEnd node and print out who won

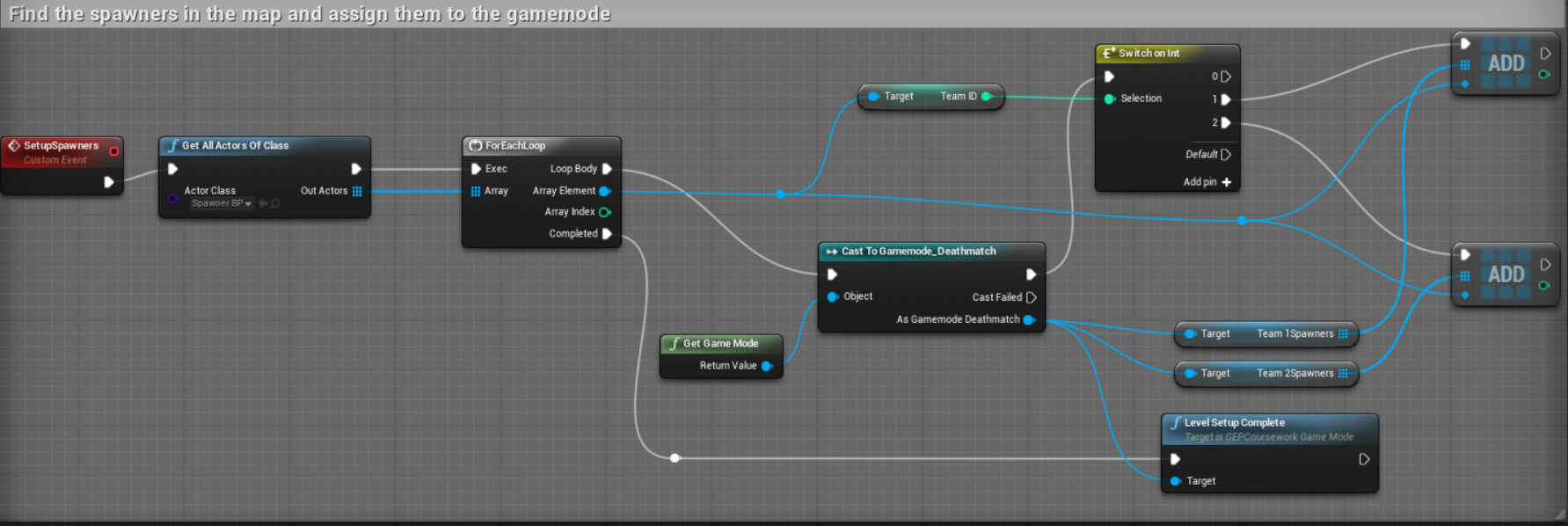


1. Make sure you set the RespawnTime and ScoreToWin variables, 2.0f and 5 is what I used.

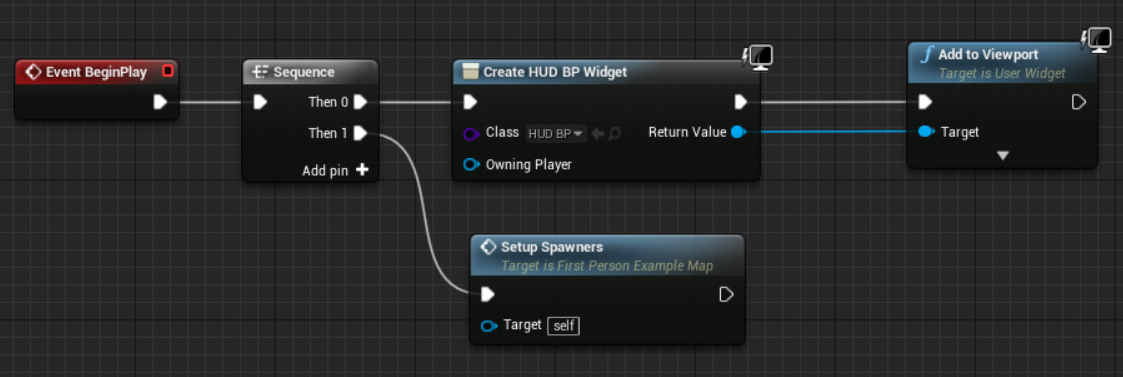
**Section 6 – Level blueprint setup**

*Because we ideally want the option of placing spawners in the level we are going to need the level blueprint to tell the gamemode what the spawners are and which team they spawn.*

1. Create a custom event called SetupSpawners



1. On BeginPlay call the event you just created.

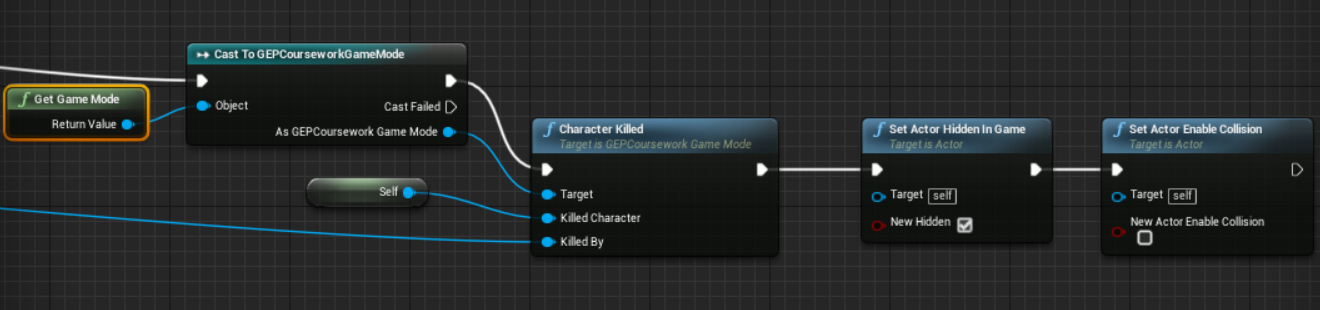


**Section 7 – Killing the NPC**

1. Wherever you kill your NPC and/or player you need to call our CharacterKilled function from our gamemode class (not the deathmatch one)

*We don’t want the NPC/Player to know about the specific gamemode as they could be part of any number of them.*

*We also have to set the actor hidden and disable collision, this is the downside of my attempt to try and keep the respawning simple, we keep them technically ‘alive’ so we can use them as a template to respawn a fresh copy, then destroy them.*



Challenges:

1. I left out how to implement the respawn timing in the UnderlookCharacter.cpp, its up to you to find a solution. As it stands they should respawn immediately.

*I recommend using the StartRespawnTimer function to set a bool then use Tick functions delta time to actually count. Working out the solution is going to involve you looking at my blueprint and understanding how it works!*

1. Create your own Gamemode for the assignment!
2. \*Optional\* Try and find a better way to handle timed respawning of multiple characters at the same time, like I said mine not perfect.

*You have all of Easter to complete this but I would recommend also making a start on your ability & weapon. Good luck!*