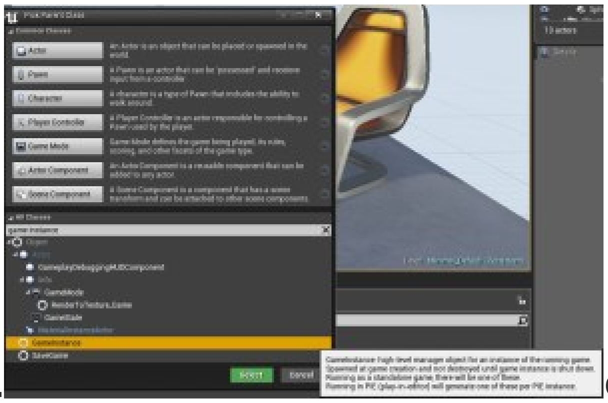
- **Object** Object is base class for rest of Unreal Engine classes. It can’t be added to world (can’t be spawned or placed in level) but it can have data and functions. If you don’t need to have something in the level Object is a class for you because you won’t have draw calls from Object. Unfortunately currently (UE4 4.7) you can’t use Objects in Blueprints. For such behavior you should use Actor Component which I will describe in later posts.
- **Actor** Actor can be spawned in Level. It can have components for example Static Mesh Component which will be graphics representation for Actor. If you want to have access to Actor you would need to spawn it in Level. You will get draw calls from Actor so if you have class which is data only don’t extend from Actor.
- **Pawn** Simply Pawn is just an Actor but it can be possessed by PlayerController or AIController. For example Pawn can be a dog controlled by the player or unit controlled by AI in RTS game.
- **Character** Character is a Pawn but with MovementComponent added. Thanks to that it can use navigation and move around. Character have SkeletalMeshComponent added as well so you can assign graphics representation by default.Let’s create new Character and name it GameplayCharacter.

USEFUL TIP: *On Mobile MovementComponent is expensive so if you are doing RTS game with lot of Characters you should extend from Pawn and implement your own movement.*

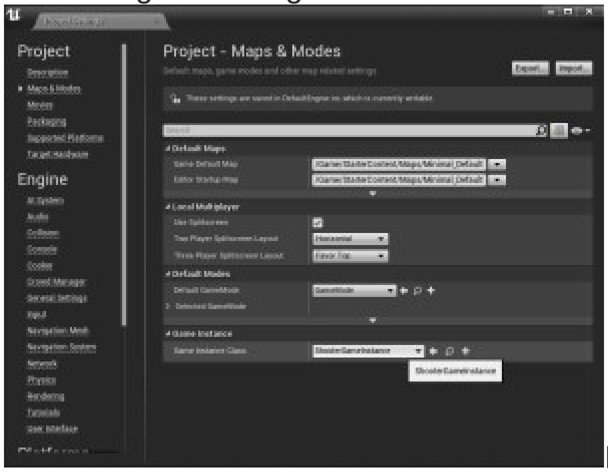
- **PlayerController** The most important part of PlayerController is that it can get input from Player (keyboard, mouse, touch, pad etc) it can posses Pawn and Character. You can have different PlayerControllers if you have different movement functionalities. For example if you are controlling a car with CarPlayerController you should have different PlayerController for controlling an air plain. The same thing comes for controlling Main Menu or controlling Gameplay. **Let’s create new PlayerController and name it GameplayPlayerController.**
- **AIController** Like PlayerController it can posses Pawn and Character but it doesn’t have access to Player input. Instead it have access to all AI tools – behavior trees, sensing components etc.

USEFUL TIP: Pawns vs Controllers *Basically Pawn should be graphics representation (mesh, animations etc) and Controller should be Pawn’s brain telling him where to move, what to do etc.*

- **Game Instance.** Game Instance is the best class for storing global data. This class will be spawned at game start and deleted when closing game. Other classes are spawned at level start and deleted after traveling to other level. You have access to GameInstance class from every blueprint! **Let’s create one. Name it**

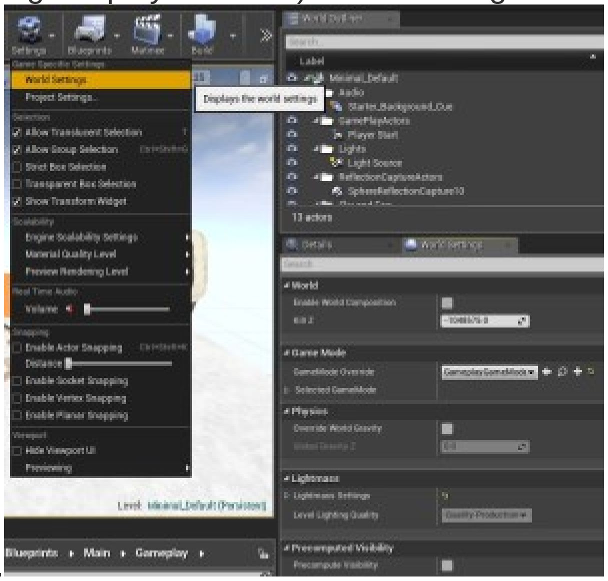


ShooterGameInstance. Game Instance need to be connected with your project. To do so go to Edit->Project Settings->Maps&Modes and change default game instance to ShooterGameInstance.

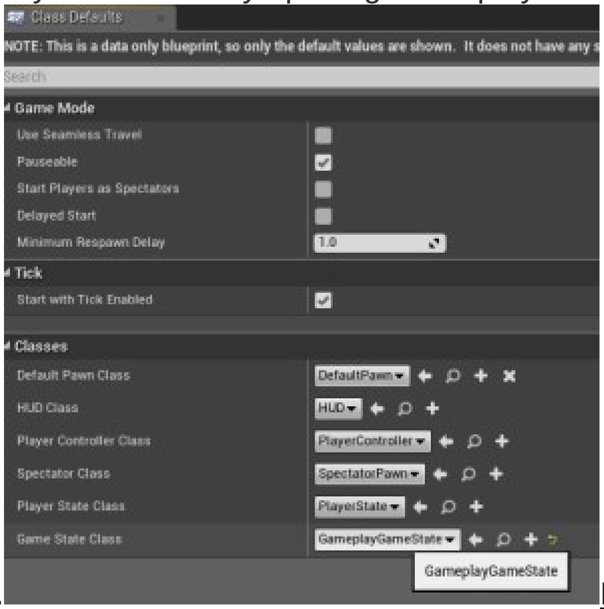


From now when you start the game ShooterGameInstance will be spawned and you will have access to it from all other classes.

- **Game Mode** The most important part of Game Mode is that only server have access to it. In single player game player is server so it isn’t necessary to use server->client functionality but still you should know about it. It have events when someone connected to your Level or someone was disconnected. Basically GameMode stores Level Default’s information:
  – Every level in Unreal Engine need to have some Game Mode connected,
  – Game Mode store data for Default Pawn, Default Player Controller, Default HUD Class and Default Game State,
  – They will be spawned when Level will be opened,Mostly Game Mode is used for determining which player win the Level or what are the level completion circumstances. Simply Game Mode is about Level rules and managing them.**Let’s create Game Mode class and name it: GameplayGameMode**. Later on we will have

more GameModes (for menu or different gameplay activities) After creating Game 

Mode it need to be assigned to a Level.

- **Game State** Simply Game State is about tracking progress of the Level (Game) and everyone have access to Game State. For example Player want to know which turn it's in turn based game or what's the current goal in shooter game.**Let's create GameMode class and name it: GameplayGameState**. Assign created GameplayGameState to GameplayGameMode by opening GameplayGameMode and choosing GameplayGameState. 
Class Defaults
NOTE: This is a data only blueprint, so only the default values are shown. It does not have any s

Game Mode	
Use Seamless Travel	☐
Pauseable	✓
Start Players as Spectators	☐
Delayed Start	☐
Minimum Respawn Delay	1.0
Tick	
Start with Tick Enabled	✓
Classes	
Default Pawn Class	DefaultPawn
HUD Class	HUD
Player Controller Class	PlayerController
Spectator Class	SpectatorPawn
Player State Class	PlayerState
Game State Class	GameplayGameState

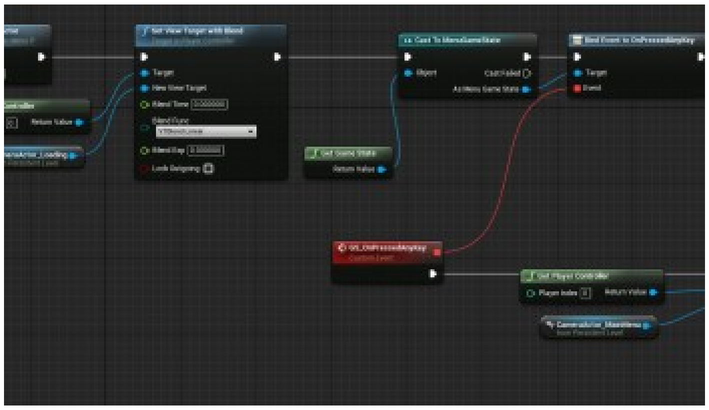
<u>Please read Rama's post about Game Mode vs Game State in UE4 AnswerHub.</u>
- **HUD** In Unreal Engine 3 this was the main class for creating 2d UI. In Unreal Engine 4 we have UMG which is much more better. Still I'm using this class to store references for my UMG Widgets and it's useful for it because you have acces to HUD from all of the classes.**Let's create HUD class and name it: GameplayHUD**. As GameState HUD need to be assigned to GameMode. Please do so.Please connect created earlier GameplayPlayerController and GameplayCharacter to GameMode as well.

Unreal Engine 4 is really big engine and you should try to use it as intended. Learn those classes and think where to put functionalities before implementing them! Last thing that I want to explain Today is how UE4 handles running the game because it’s connected to those classes.

1. When you run the game GameInstance class will be spawned,
2. Default Level will be loaded,
3. When Level is opening its checking for GameMode and spawning all of the default classes from GameMode,
4. DefaultPlayerController is asked to posses DefaultPawn so you don’t need to posses it manually,
5. It’s spawning all Actors that was placed in the Level.

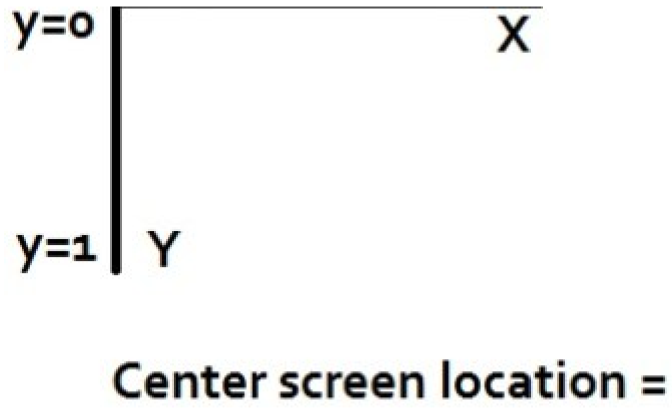
Of course its high level overview. When you are opening different level (or restarting the current one) all classes are destroyed except of GameInstance class! If you have questions about Unreal Engine 4 base classes please let me know, hopefully I will be able to answer!

Related



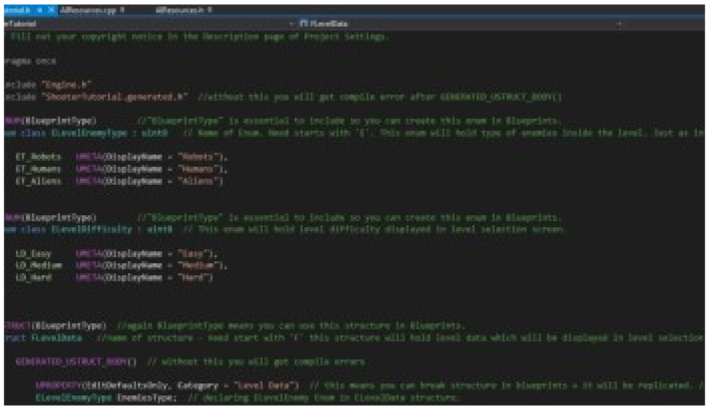
Menu - In-Game Loading Screen

In "Old Tutorial"



UE4 Tips: Plug and Play Radar

In "UE4 Tips"



```
 Fill out your copyright notice in the Description page of Project Settings.

ragma once

nclude "Engine.h"
nclude "ShooterTutorial.generated.h"  //without this you will get compile error after GENERATED_USTRUCT_BODY()

NUM(BlueprintType)        //"BlueprintType" is essential to include so you can create this enum in Blueprints.
um class ELevelEnemyType : uint8   // Name of Enum. Need starts with 'E'. This enum will hold type of enemies inside the level. Just as in

  ET_Robots   UMETA(DisplayName = "Robots"),
  ET_Humans   UMETA(DisplayName = "Humans"),
  ET_Aliens   UMETA(DisplayName = "Aliens")


NUM(BlueprintType)        //"BlueprintType" is essential to include so you can create this enum in Blueprints.
um class ELevelDifficulty : uint8  // This enum will hold level difficulty displayed in level selection screen.

  LD_Easy     UMETA(DisplayName = "Easy"),
  LD_Medium   UMETA(DisplayName = "Medium"),
  LD_Hard     UMETA(DisplayName = "Hard")



STRUCT(BlueprintType)  //again BlueprintType means you can use this structure in Blueprints.
ruct FLevelData   //name of structure - need start with 'F' this structure will hold level data which will be displayed in level selection

  GENERATED_USTRUCT_BODY()  // without this you will got compile errors

      UPROPERTY(EditDefaultsOnly, Category = "Level Data")  // this means you can break structure in blueprints + it will be replicated. /
      ELevelEnemyType EnemiesType;  // declaring ELevelEnemy Enum in ELevelData structure.
```

Menu - Level Selection - C++ Base Data

In "Old Tutorial"

CATEGORIES **OLD TUTORIAL** **TAGS** **CONFIGURING NEW PROJECT**, **GAMEINSTANCE**, **GAMEMODE**, **GAMESTATE**, **PLAYERCONTROLLER**, **PLAYERPAWN**, **WORD SETTINGS**

19 Replies to “Configuring new project. UE4 main classes explanation.”

1. Pingback: First person look around controls – mouse touch tilt | Shooter Game Tutorial
2. **levenlol**

   JUNE 29, 2015 AT 5:06 PM

   Hi, i just need to change, into gamemode, the Default Pawn Class to connect