Содержание

Введение

Глава 1. Основы

1.1 Переменные

1.1.1 Основные типы

В Unreal Engine C++ могут использоваться как и стандартные типы языка C++, так и инициализированные типы языка Blueprints. В данной таблице содержаться основные типы, которые используются при разработке на языке чертежей:

Название	Синтаксис C++	Include
Integer	int32	-
Float	float	-
Vector	FVector	#include "Math/Vector.h"
Rotator	FRotator	#include "Math/Rotator.h"
Text	FText	#include "Internationalization/Text.h"
Actor	AActor	#include "GameFramework/Actor.h"
Static Mesh	UStaticMesh	#include "Engine/StaticMesh.h"
Skeletal Mesh	USkeletalMesh	#include "Engine/SkeletalMesh.h"
Texture2D	UTexture2D	#include "Engine/StaticMesh.h"
Material	UMaterial	#include "Engine/Texture2D.h"

1.1.2 Модификаторы

Чтобы дать переменной в C++ особые свойства по типу - видимость в Unreal Engine Blueprints и так далее используется специальная конструкция - **UPROPERTY(<Params>)**. Вот основные параметры которые могут подаваться на вход:

- BlueprintReadWrite - полная видимость в Blueprints
- EditAnywhere - аналогия EditOnSpawn
- Category="Name" - добавляет переменную в категорию Name

Синтаксис:

```
{
        UPROPERTY(BlueprintReadWrite, Category="Params")
        int32 SIZE_X = 0;
};
```

1.1.3 Массивы

Для создания массивов в Unreal Engine CPP используется - **TArray<TYPE>**, где TYPE - тип элементов массива.

Синтаксис:

```
{
        TArray<int32> indices;
        TArray<APuzzle*> puzzles;
};
```

Основные операции с массивом:

- Arr.Num() - длина массива
- Arr[Index] - элемент массива
- Arr.Add(Element) - добавить элемент массива
- Arr.RemoveAt(Index) - удаляет элемент по индексу

1.2 Функции

Глава 2. Structs

2.1 Создание структуры

Для создания структуры используется пустой экземпляр класса CPP **None**. Из сгенерированных файлов необходимо удалить файл **<Name>.cpp** и оставить только **<Name>.h**. В Header файла создается следующая структура:

```
#pragma once

#include "<Name>.generated.h"

USTRUCT(BlueprintType)
struct F<Name> : public FTableRowBase
{
        GENERATED_BODY()

        FORCEINLINE F<Name>();
};

FORCEINLINE F<Name>::F<Name>(){
}
```

Глава 3. AActor

3.1 Spawn Actor

Для спауна эктора используется метод - **SpawnActor()**

Синтаксис:

```
{
    FVector location = FVector();
    FRotator rotation = FRotator();
    FActorSpawnParameters params;
    AActor *actor = (AActor*) GetWorld()->SpawnActor(
      AActor::StaticClass(),\&location,\&rotation,params);

};
```