1. Что необходимо сделать, чтобы указать UE, что мы хотим видеть переменные в Editor’e, специальный тип int в UE для кроссплатформенности, параметры макроса, как работают

2. Как создать дочерний Blueprint-класс

3. Второй параметр макроса – за что отвечает

4. Какую функцию имеет каждый Actor (для его имени)

1. Убрали наши переменные из функций printTypes() и printStringTypes() и добавили их в качестве полей у нашего класса:

Изображение выглядит как текст

Автоматически созданное описание

2. Для того, чтобы указать UE, что мы хотим видеть переменные в Editor’е, необходим макрос UPROPERTY() у каждого поля. При этом у полей типа int мы заменим тип на int32 – специальный тип UE для обеспечения кроссплатформенности, а также добавим каждой property различные спецификаторы доступа – это параметры макроса UPROPERTY():

Изображение выглядит как текст

Автоматически созданное описание

Спецификаторов UPROPERTY намного больше, сейчас рассмотрим эти.

3. Зайдя в проект, мы можем увидеть, что в Details нашего актора появились почти все поля:

Изображение выглядит как текст, электроника

Автоматически созданное описание

4. Создадим дочерний Blueprint-класс, для этого кликаем ПКМ на нашего актора в ContentBrowser и выбираем Create Blueprint class based on ... и задаем ему свое имя (используют BP\_имя\_класса).

5. При добавлении блюпринта или нашего C++-класса на сцену, создается его инстанс (то есть объект).

Как мы видим, в редакторе блюпринта и в Details в UE у нас разные поля:

BP:



UE:

Изображение выглядит как текст, электроника

Автоматически созданное описание

Например, KillsNum можно редактировать только в дефолтных настройках архетипа, в инстансе – нет.

За это и отвечает спецификаторы.

6. Вторым параметром в UPROPERTY мы можем указать категорию, к которой относится данный property:

Изображение выглядит как текст

Автоматически созданное описание

7. Вызовем printTypes() в BeginPlay() и добавим в него еще один UE\_LOG:

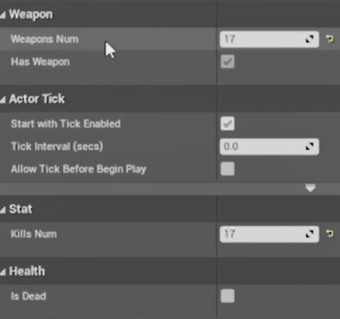
Изображение выглядит как текст

Автоматически созданное описание

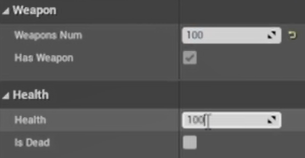
Каждый актор имеет функцию GetName(), которая возвращает имя текущего инстанса актора. Возвращает FString, поэтому надо использовать оператор \*.

8. Запустив редактор видим, что теперь наши параметры разбились по категориям:

BP:



UE:



При этом в логе видим всю инфу:

Изображение выглядит как текст

Автоматически созданное описание