4. ACharacter and APlayerController classes

1. Что добавляется персонажу по сравнению с пауном?

2. Какие два класса создали (логика)?

3. Баг с подключением хедеров, как решаем?

4. Далее необходимо добавить наши новые два класса классами по умолчанию, как это сделали?

5. Далее необходимо создать BP-версии наших классов, зачем?

6. Открыли BP нашего персонажа, из каких компонент он состоит (4 штуки, про каждую подробно, какая является корневой)

7. Выбрали нужный ассет и настроили его (для какого компонента и как настроили?)

8. Что будет с камерой персонажа, если мы сейчас запустим игру? Как фиксим это в коде?

9. Настроили положение камеры в UE

1. Чарактер расширяет класс пауна, так как в него добавлен дополнительный функционал для работы с движением, анимацией и физическим взаимодействием.

2. Создали классы STUBaseCharacter и STUPlayerController в папке Player.

3. Иногда случается так, что UE плохо подключает вложенные в другие папки (как в нашем случае – в папку Player) файлы, поэтому можно это поведение исправить в C# файле /название\_проекта/.Build.cs, добавим туда строчку:

Изображение выглядит как текст

Автоматически созданное описание

4. Далее создали конструктор для нашего STUGameModeBase и сделали в нем дефолтным пауном и контроллером сгенерированные нами классы. Почему дефолтным пауном сделали чарактера? Потому что чарактер наследуются от пауна.

5. Далее перешли в едитор и в нем создали BP-варианты классов для нашего мода, чарактера и контроллера (последних двух положили в новую папочку Player – еще одну).

И далее мы в нашем гейм моде выбираем как дефолтные именно эти BP-версии наших классов, чтобы производить все **визуальные** настройки в блюпринтах, а не в кодовых файлах (так, банально, удобнее).

6. Открываем блюпринт нашего персонажа – он состоит из следующих компонент:

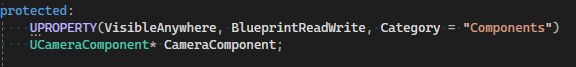
* CapsuleComponent – отвечает за коллизию в мире, то есть за физическое взаимодействие – это корневой компонент
* ArrowComponent – это компонент создан для дебага, он не отображается в игре, он лишь показывает направление вектора движения персонажа
* Mesh (Skeletal Mesh) – отвечает за визуальное представление персонажа. В Skeletal Mesh помимо обычных полигонов есть еще иерархия костей, с помощью которых данный полигоны могут быть анимированы.
* Character Movement – отвечает за движение нашего персонажа, компонент чисто логический, у него нет внешнего представления.

Изображение выглядит как текст

Автоматически созданное описание

7. Так как мы импортировали из проекта UE ассеты, можем здесь сразу выбрать Меш для нашего персонажа. Выбрали нужный ассет, опустили персонажа так, чтобы он совпадал с капсулой, а также повернули персонажа на 90 градусов, чтобы его forward вектор совпадал с направлением arrow компонента.

8. Если сейчас запустить – камера будет в середине персонажа. Для изменения положения камеры перейдем в код. Добавляем компонент камеры нашему персонажу:



Далее создаем данный компонент в конструкторе персонажа и привязываем к корневому компоненту:

Изображение выглядит как текст

Автоматически созданное описание

Напоминаю, что у нашего персонажа корневым элементом является компонент коллизии. Запускаем едитор.

10. В едиторе снова открыв в блюпринт едиторе нашего персонажа, мы можем увидеть, что добавилась камера. Настраиваем ее на удобном расстоянии.