**4. Crosshair Icon for each Weapon**

1. Добавили для начало папку с чем-то в проект?

2. Какой элемент добавили на виджет? Как настроили ему позицию, размер?

3. Теперь сделаем так, чтобы при смене оружия выставлялась соответствующая картинка прицела – какую создали структуру, проперти и функцию и в каком файле?

3. Создали в WeaponComponent функцию с той же целью – какой функционал?

4. Добавили функцию в виджет, какую и какой функционал?

5. Суммируем – три пункта, что сделали?

6. Забиндили функцию для какого параметра и как определили поведение?

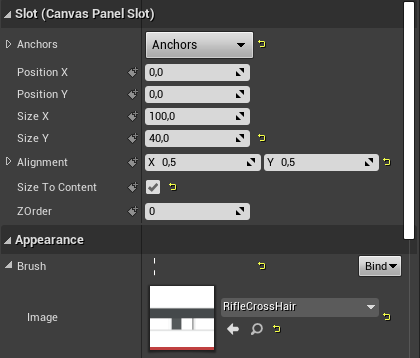
1. В папку нашего проекта добавили картинки для прицела и иконки еще какой-то (в данном уроке речь только про прицел). И перенесли драгндропом папку к нам в эдитор. При этом UE автоматически создаст ассеты.

2. На наш виджет добавим элемент Image.

Выберем SizeToContent – используются исходные размеры картинки.

Для фиксации в центре вьюпорта есть поле Anchors, в котором мы выбираем соответствующий элемент – то есть начало координат для нашей картинки будет в центре. Этого нам недостаточно, так как отчет идет от левого верхнего угла картинки.

Для исправления этого есть поле Alignment – изменяется от 0 до 1. Если мы поставим 0.5, 0.5, то точка начала отчета сдвинется в центр картинки.



2. Теперь сделаем так, чтобы при смене оружия выставлялась соответствующая картинка прицела.

Для начала в STUCoreTypes.h создадим новую структуру данных. В данной структуре будут хранится прицел оружие и его иконка. Тип данных для текстур в UE – Utexture2D:

Изображение выглядит как текст

Автоматически созданное описание

В BaseWeapon создаем объект данной структуры в секции protected, а так же геттер:

Изображение выглядит как текст

Автоматически созданное описание

Для каждого вида оружия в BP нам надо будет проинициализировать данную структуру.

3. В WeaponComponent надо так же создать функцию обертку, которая возвращает UIData текущего оружия.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Таким образом, для использования данной функции сначала проверить, возвращает ли она true, и лишь потом обратиться к данным.

4. И добавим функцию геттер в класс виджета, она будет иметь такую же сигнатуру, но к данной функции мы добавляем BlueprintCallable, чтобы получить доступ к ней в блюпринтах (и не забываем добавить ЗФ STUCoreTypes.h с нашими структурами):

Изображение выглядит как текст

Автоматически созданное описание

Функция будет иметь примерно такой же вид, как GetHealthPercents + не забываем подключить ЗФ с WeaponComponent:

Изображение выглядит как текст

Автоматически созданное описание

По сути в виджете мы делаем функцию обертку над функцией WeaponComponent, чтобы не делать данное преобразование в блюпринте.

5. Суммируем:

а) объявили структуру для UI-данных оружия в STUCoreTypes, создали для нее проперти и геттер для нее в базовом классе оружия;

б) в WeaponComponent мы получаем UIData для текущего оружия;

в) а в виджете мы получаем доступ к компоненту оружия и вызываем его функцию.

6. В панели Details нашего элемента Image биндим новую функцию Get\_CrosshairImage. Просто забиндить изображение нельзя, потому что оно является частью структуры:

Изображение выглядит как текст, внутренний

Автоматически созданное описание

У функции GetWeaponUIData, как у ноды, которая возвращает данные по ссылке, имеет дополнительный пин справа.

Картинку устанавливаем с помощью MakeSlateBrush. Slate brush – это набор данных, который определяет то, как мы будем отображать что-то в UE.

Так как мы хотим размер как у оригинального изображения, получить их мы можем с помощью GetSizeX и GetSizeY.

Остается только засетапить картинки для каждого из типов оружия.

Сделать небо темным можно в элементе SkySphere, исправив там значение Directional Light Actor на Clear и Sun Height = -1,0.