Для создания возможности взаимодействовать с различными объектами установки нужно создать класс, от которого будут наследоваться блупринты интерактивных объектов.

Первым делом необходимо создать общий интерфейс интерактивных объектов. Для этого создаем класс IInteractable (подробное описание создания классов указано в Муравьёв\_КП). В этом классе будут объявлены чисто виртуальные методы, необходимые для реализации поставленных задач.

Так как все методы являются чисто виртуальными, а их реализация будет в отдельном классе, описан будет только заголовочный файл.

В нем создаем объявление следующих методов:

* virtual void Interact(ADBaseCharacter\* Character) – метод для взаимодействия с объектом.
* virtual void SetOpacity(ADBaseCharacter\* Character) – метод для настройки прозрачности интерактивного объекта.
* virtual void SetHighlightObject(ADBaseCharacter\* Character) – метод для установки выделения объекта.
* virtual void RemoveHighlightObject(ADBaseCharacter\* Character) – метод для снятия выделения с объекта.
* virtual FName GetActionEventName() const – константный метод для определения описания интерактивного объекта.

Для объявления чисто виртуальных методов в Unreal Engine используется конструкция PURE\_VIRTUAL(param1, param2, ).

Весь код класса Interactive представлен в приложении 1.

Далее нужно создать класс, реализующий интерфейс IInteractable. Название класса – ComponentOfMechanism.

В заголовочном файле объявляем следующие публичные методы и свойства:

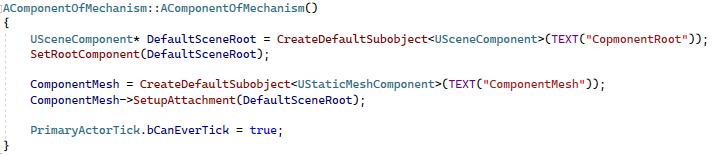
* AComponentOfMechanism() – конструктор по умолчанию.
* virtual void Interact(ADBaseCharacter\* Character) override – метод, реализующий действие объекта при взаимодействии персонажа с ним.
* virtual void SetOpacity(ADBaseCharacter\* Character) override – метод, регулирующий прозрачность объекта.
* virtual void SetHighlightObject(ADBaseCharacter\* Character) override – метод, реализующий выделение объекта при пересечении с лучом.
* virtual void RemoveHighlightObject(ADBaseCharacter\* Character) override – метод, снимающий выделение объекта по окончанию пересечения с лучом.
* void PlusTemp(), void SubTemp() – методы, меняющие материал объекта
* FName GetDesciption() – метод, получающий описание объекта взаимодействия.
* UPlayerHUDWidget\* PlayerHUDWidget – ссылка на виджет интерфейса (см. Муравьёв\_КП).
* virtual void Tick(float DeltaTime) override – метод базового класса.
* bool isOpascity = false – переменная, отвечающая за прозрачность.

Далее объявляются залащенные свойства и методы:

* UPROPERTY(VisibleAnywhere, BlueprintReadOnly, Category = "Interactive | Copmonent") UStaticMeshComponent\* ComponentMesh – ссылка на объект.
* UPROPERTY(EditDefaultsOnly, BlueprintReadOnly, Category = "Widgets") TSubclassOf<class UPlayerHUDWidget> PlayerHUDWidgetClass – ссылка на класс пользовательского интерфейса.
* UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Widgets") FName Description – описание объекта.
* UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Widgets") UMaterial\* BasicMaterial – базовый материал объекта.
* UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Widgets") UMaterial\* InteractMaterial – материал взаимодействия с персонажем.
* UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Widgets") UMaterial\* IntersectionMaterial – материал пересечения с лучом.
* UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Widgets") UMaterial\* HotMaterial – материал для горячей воды.
* UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Widgets") UMaterial\* ColdMaterial – материал для холодной воды.
* virtual void BeginPlay() override – метод базового класса.

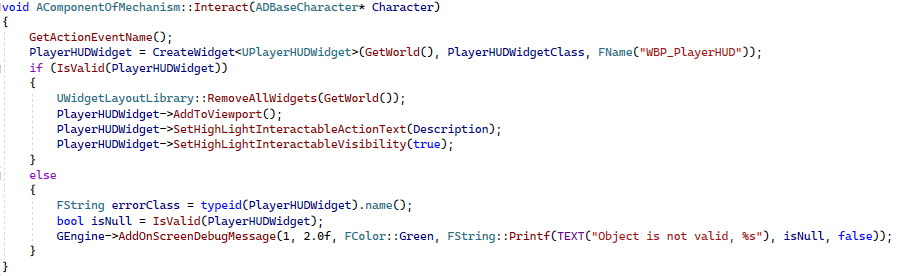
В исполняемом файле реализуем конструктор по умолчанию. В нем создается корень объекта, к корню прикрепляется основная модель объекта и выставляется возможность для актера срабатывать каждый тик (листинг …).

Листинг … - реализация конструктора по умолчанию



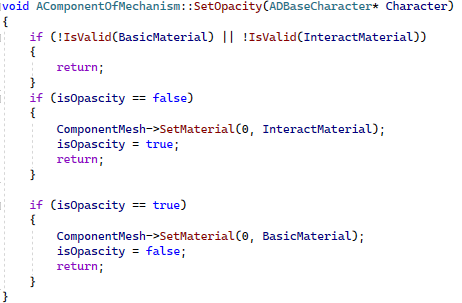
В реализации метода Interact заполняется ссылка на виджет интерфейса, и в случае взаимодействия он добавляется на экран пользователя (листинг ).

Листинг – реализация метода Interact



В реализации метода SetOpacity проверяется валидность материалов и в случае валидности выставляется значение нужного материала (листинг ).

Листинг – реализация метода SetOpacity



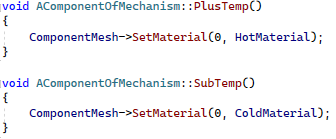
В методах SetHighlightObject и RemoveHighlightObject устанавливается выделение объекта и снимается соответственно. RemoveHighlightObject возвращает состояние объекта в соответствии с его состоянием (листинг ).

Листинг – методы SetHighlightObject и RemoveHighlightObject



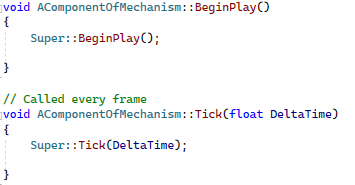
Методы PlusTemp и SubTemp устанавливают объекту материал для горячей и холодной воды соответственно (листинг ).

Листинг – методы PlusTemp и SubTemp



В реализации методов BeginPlay и Tick вызываются методы базового класса (листинг ).

Листинг – методы BeginPlay и Tick



После этого необходимо создать блупринты интерактивных объектов, наследуемые от этого класса. В настройках блупринта выставляем модель объекта и все необходимые для него материалы (рисунок ). Экземпляров блупринта на сцене может быть больше одного.

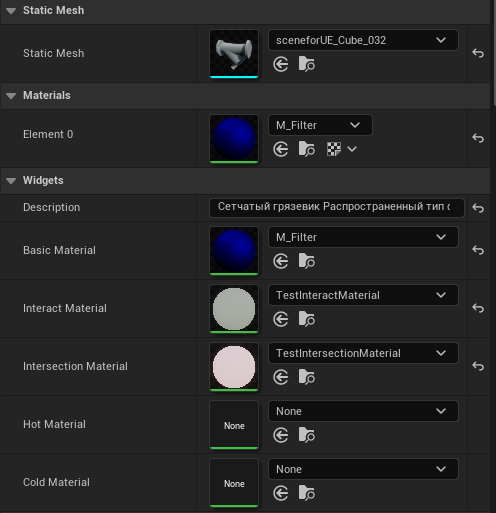


Рисунок – настройка блупринта интерактивного объекта

Для того, чтобы создать материал необходимо выбрать пункт *Material* (рисунок ). В нем нужно создать ноды VectorParameter и два ScalarParameter. Их можно переименовать для удобства взаимодействия и восприятия. Выставляем необходимые значения и подключаем их в соответствующие входы – BaseColor для VectorParameter, Metallic и Roughness для ScalarParameter (рисунок ).

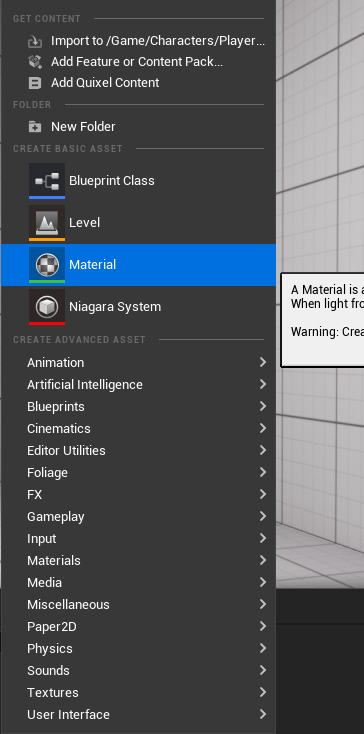


Рисунок – создание материала

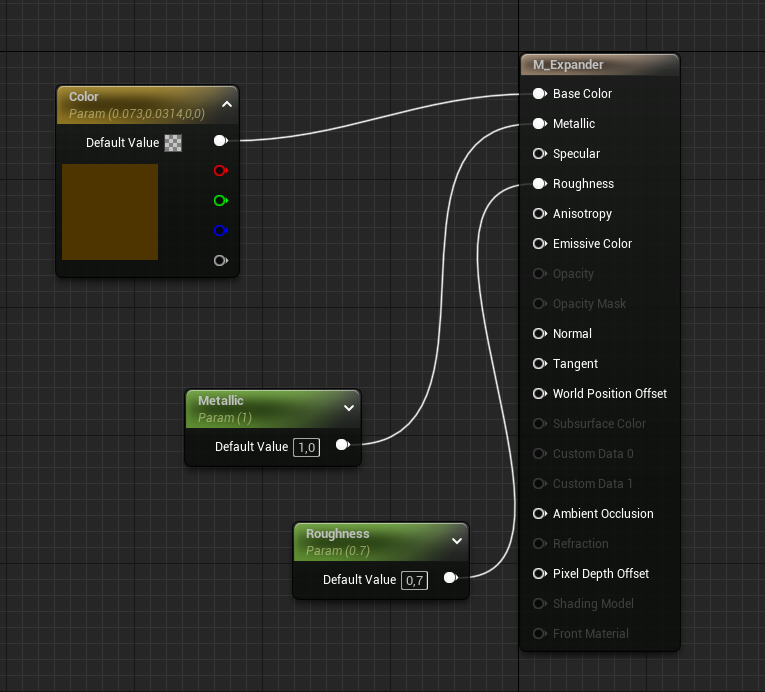


Рисунок – настройка материала

Для настройки блупринта сложносоставного объекта следует добавить к нему дополнительные StaticMeshComponent (рисунок ).

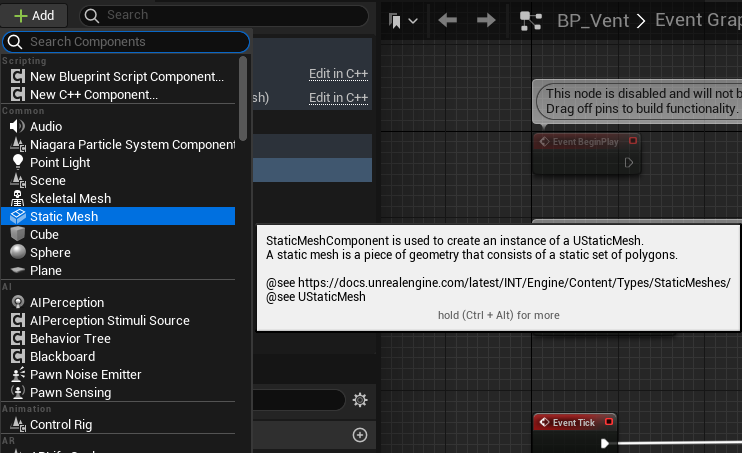


Рисунок – дополнительные объекты в блупринте

В Event Graph реализован скрипт для применения одного материала от базового объекта ко всем объектам блупринта. В нем берется значение материала основного объекта и присваивается ко всем остальным объектам блупринта (рисунок ).

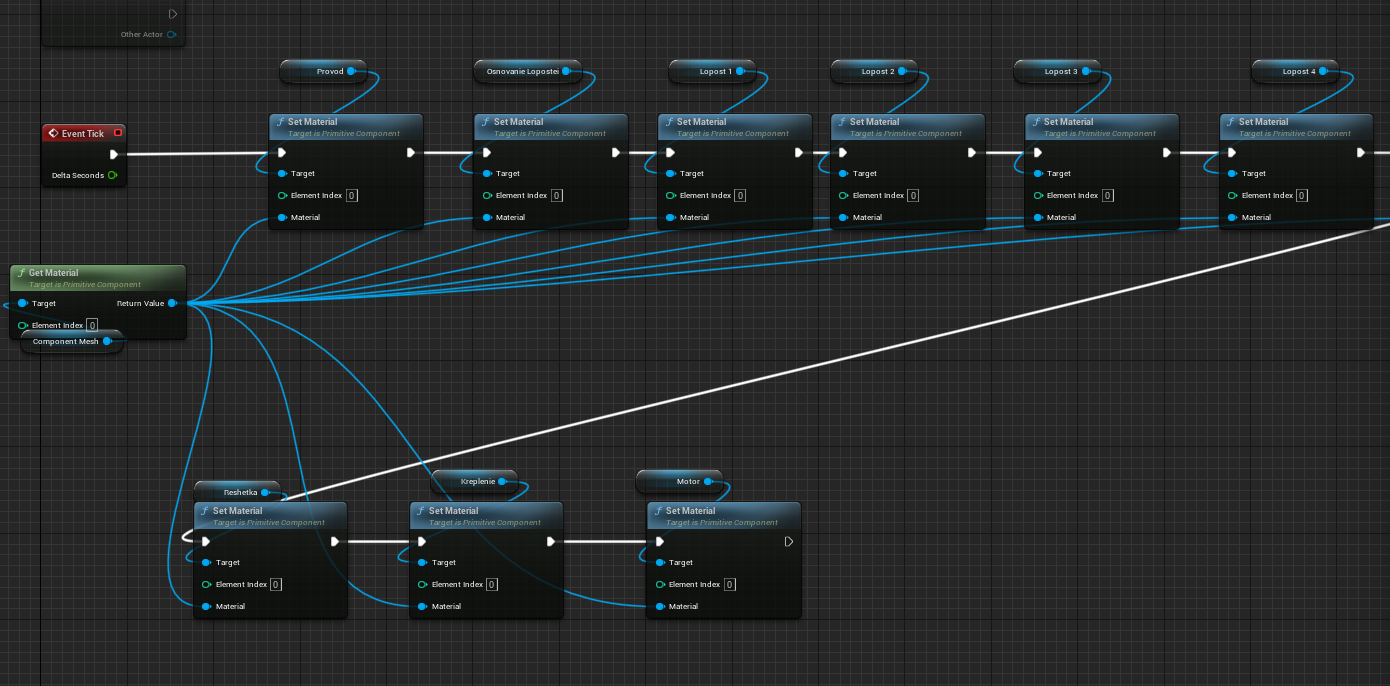


Рисунок – блупринт применения одного материала ко всем объектам

Для возможности взаимодействия с объектом необходимо выделить объект, для этого в классе DBaseCharacter был реализован метод TraceLineOfSight (листинг ). В нем реализованы выпуск луча из персонажа по направлению его взгляда и передача ссылки на объект столкновения, если такой имеется, а так же записывается описание объекта.

Листинг – метод TraceLineOfSight

