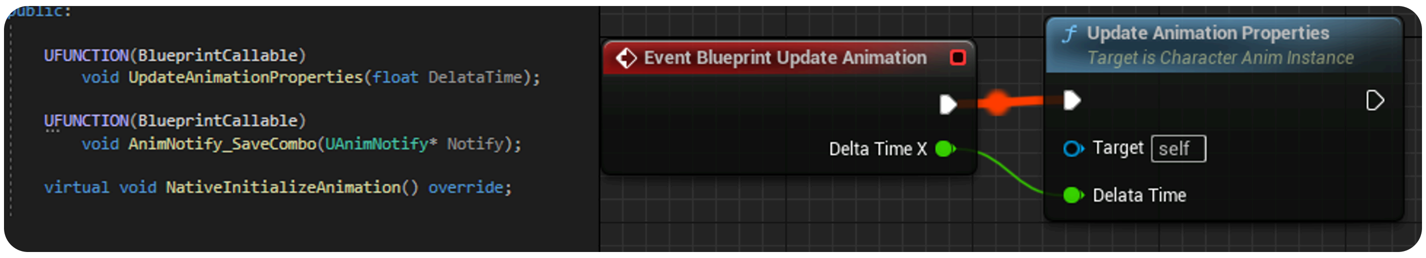
Unreal Engine Reflection System

Information for programmers developing Objects to be used with Unreal Engine.



```
public:

    UFUNCTION(BlueprintCallable)
        void UpdateAnimationProperties(float DelataTime);

    UFUNCTION(BlueprintCallable)
        void AnimNotify_SaveCombo(UAnimNotify* Notify);

    virtual void NativeInitializeAnimation() override;
```

Event Blueprint Update Animation
Delta Time X
Update Animation Properties
Target is Character Anim Instance
Target
self
Delata Time

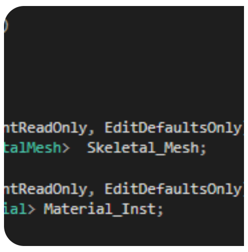
The **Unreal Engine Reflection System** encapsulates your classes with various macros that provide engine and editor functionality. When programming with **Unreal Engine**(**UE**), it is possible to have standard C++ classes, functions, and variables.

- The base class for objects in Unreal is UObject. Each class defines a template for a new Actor or Object.
- You can use the `UCLASS` macro to tag classes derived from `UObject` so that the UObject handling system is aware of them.
- TSubclassOf is a template class that provides `UClass` type safety. It is useful for assigning classes that derive from a specific type. For example, you may expose this variable to Blueprint where a designer can assign which weapon class is spawned for a Player Character.
- Classes can contain structs. Structs are data structures that help with the organization and manipulation of their related member properties. Structs can be defined on their own using the `USTRUCT()` macro.
- The Unreal Smart Pointer Library is a custom implementation of C++11 smart pointers designed to ease the burden of memory allocation and tracking. This implementation includes the industry standard Shared Pointers, Weak Pointers, **Unique Pointers**, and Shared References which act like non-nullable Shared Pointers.
- Interfaces provide functions and additional gameplay behavior you can implement in multiple or different classes. Your player character can interact with a variety of Actors in the world. Each of these interactions can cause a different reaction to an event.
- Metadata Specifiers control how classes, interfaces, structs, enums, functions, or properties interact with various aspects of the engine and editor. Each type of data structure or member has its own list of Metadata Specifiers.
- UFUNCTION, and UPROPERTY macros make UE aware of new classes, functions, and variables. These macros are garbage collected by the engine. When specifying macros, you can edit and display them within the Unreal Editor.

Section Directory

Objects

Explanations of the basic gameplay elements, Actors and Objects.



```
ntReadOnly, EditDefaultsOnly
talMesh>  Skeletal_Mesh;

ntReadOnly, EditDefaultsOnly
ial> Material_Inst;
```

Properties

Reference for creating and implementing properties for gameplay classes.

Structs

Reference to creating and implementing structs for gameplay classes.

TSubclassOf

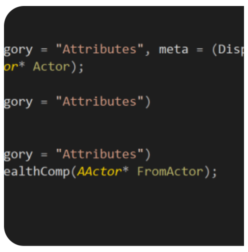
Using the TSubclassOf template class to provide type safety.

Unreal Interfaces

Create and implement Unreal Interfaces in C++ and Blueprints.

Metadata Specifiers

Metadata keywords used when declaring UClasses, UFunctions, UProperties, UEnums, and UInterfaces to specify how they behave with various aspects of Unreal Engine and the editor



```
gory = "Attributes", meta = (Disp
or* Actor);

gory = "Attributes")


gory = "Attributes")
ealthComp(AActor* FromActor);
```

UFunctions

Overview for creating and implementing functions for gameplay Classes

Unreal Smart Pointer Library

Custom implementation of shared pointers, including weak pointers and non-nullable shared references.