TSubclassOf

Using the TSubclassOf template class to provide type safety.

TSubclassOf is a template class that provides UClass type safety. For instance, let's imagine that you are creating a projectile class that allows the designer to specify the damage type. You could just create a UPROPERTY of type UClass and hope the designer always assigns a class derived from UDamageType or you could use the TSubclassOf template to enforce the choice. The sample code below illustrates the difference:

```
/** type of damage */
UPROPERTY(EditDefaultsOnly, Category=Damage)
UClass* DamageType;

```

Copy full snippet

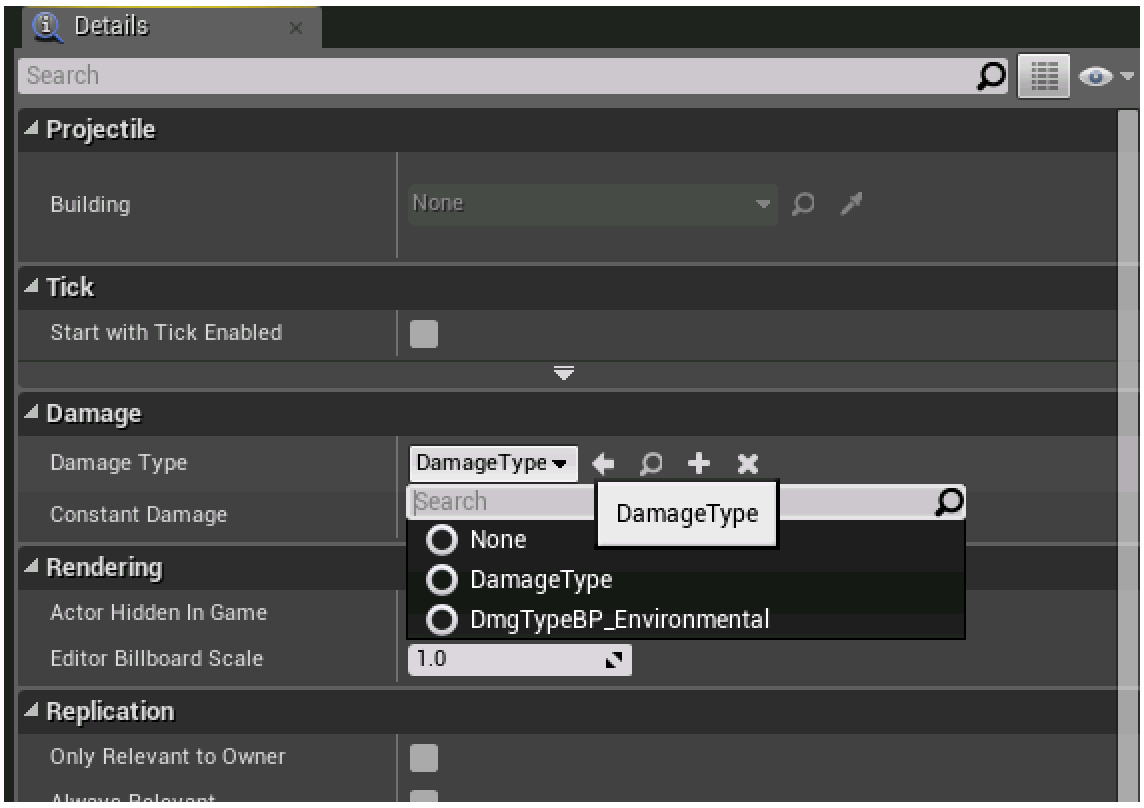
Vs.

```
/** type of damage */
UPROPERTY(EditDefaultsOnly, Category=Damage)
TSubclassOf<UDamageType> DamageType;

```

Copy full snippet

In the second declaration, the template class tells the editor's property windows to list only classes derived from UDamageType as choices for the property. In the first declaration any UClass can be chosen. The image below illustrates this.



Example from StrategyGame's projectile Blueprint

In addition to this UPROPERTY safety, you get type safety at the C++ level too. If you try to assign incompatible TSubclassOf types to each other you'll get a compilation error. In the case you are trying to assign a generic UClass, it will perform a runtime check to verify that it can do the assignment. If the runtime check fails, the resulting value is nullptr.

```
UClass* ClassA = UDamageType::StaticClass();

TSubclassOf<UDamageType> ClassB;

ClassB = ClassA; // Performs a runtime check

TSubclassOf<UDamageType_Lava> ClassC;

ClassB = ClassC; // Performs a compile time check
```

Copy full snippet