Subjects. Низко-уровневые Сущности.

Subjects - основополагающие легковесные сущности, управляемые Аппаратом. Главным образом, они UE-независимы и состоят из трейтов и флажков.

Хэндлеры сущностей

Сущности не используются напрямую и их внутреннее устройство спрятано от пользователя плагина. Вместо этого предлагается специальный концепт *Subject Handle*. В терминах Unreal-а хэндлеры похожи на слабые указатели (weak pointer). Когда вы деспавните сущность, то все указывающие на неё хэндлеры автоматически становятся невалидными. Реализация этого подхода использует generation-based технику подсчёта ссылок.

Уровень сущностных объектов

У сущности может быть дополнительная высоко-уровневая надстройка, которая называется Subjective или сущностный объект. Сущностные объекты - это UE-управляемые UObject-ы, которые могут содержать детали. Сущностные объекты с деталями, вообще говоря, предоставляют большую гибкость и имеют дополнительные особенности по сравнению с простыми сущностями и трейтами. Эта функциональность достигается за счёт потенциального уменьшения производительности и отсутствия поддержки кэша. Пожалуйста, обратите внимание, что Сущностные объекты - это опциальная фича Аппарата, и что вы можете реализовать логику своего проекта полностью на сущностях, если хотите.

Сущности без надстройки сущностного объекта над ними называются костяными (*barebone*) сущностями. Оба типа - сущности с объектом типа сущностный (Subjective-based Subjects) и костяные сущности (barebone Subjects) - можно называть просто сущностями - *Subjects*. Это потому что каждый сущностный объект имеет в себе встроенную сущность и использует её как естественную свою часть. Выражаясь простым языком: Subjectives are Subjects. Это что-то наподобие наследственной связи, поэтому каждый сущностный объект может иметь трейты и флаги как часть своей программной структуры.

Спавн (Spawning)

Spawning (или спавн) - это процесс создания сущности как часть механизма.

Организация в C++

Чтобы заспавнить новую сущность внутри Механизма, вам потребуется вызвать один из

методов SpawnSubject. Самый простой вариант это сделать:

```
FSubjectHandle Subject = Machine::SpawnSubject();
```

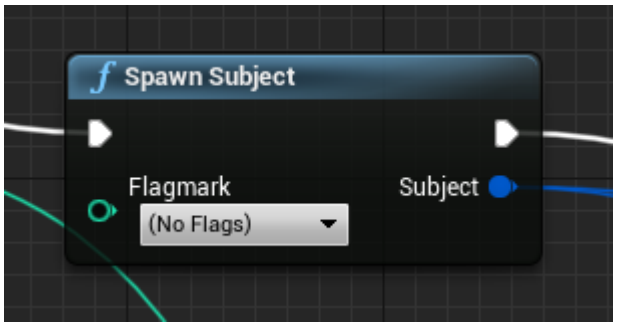
Если хотите, то можете спавнить сущности, указывая трейты, которые будут в неё установлены, и для этого вам потребуется специальный шаблонный метод с тем же названием:

```
FBurning Burning{10, 15.5f};
FSword Sword{2};
FSubjectHandle BurningSword = Machine::SpawnSubject(Burning, Sword);
```

Данный отрывок кода эффективно выделит память под слот для сущности в нужном чанке и проинициализирует его в соответствии с переданными трейтами.

Организация в Blueprint-ах

Спавн сущностей из блупринтов выполняется через выделенную Spawn Subject ноду:



Вы также можете определить начальный набор флажков через соответствующий аргумент.

Деспавн

Процесс *despawn*-а является прямой противоположностью процессу спавна и он означает удаление сущности. Удаление уже удалённого (или невалидного) хэндлера - легальная операция, которая ничего не делает и не вызывает ошибок.

Организация C++

Чтобы удалить сущность в своём C++ коде, используйте Despawn метод, предоставляемый хэндлером. Это можно делать так:

```
void PickPowerup(FSubjectHandle Player, FSubjectHandle Powerup)
```

```
{
 // Добавление силы/скорости/жизней игроку.
   ...
 // Удаление предмета из мира:
   Powerup.Despawn();
}
```