Трейты (Traits)

Трейты - низкоуровневые блоки данных (компоненты) в составе сущностей. В отличие от деталей, трейты являются простыми UStruct данными. Плагином они управляются вручную кэш-эффективным способом и главным образом направлены на runtime-производительность.

Изменение трейтов

Трейт - это то, что в других языках программирования называется value-type. Способ работы с ним достаточно родной для UE. Он заключается в том, что для изменения данных трейта, сперва надо его прочитать, изменить, затем записать обратно, применяя изменения к сущности. Эта процедура необходима для атомарного выполнения и безопасности. Не пугайтесь этой рутины копирования, потому что такая процедура - ещё один шанс для программы подвергнуться значительным улучшениям производительности вследствие локальности данных.

Заметки о Garbage Collection

Как было сказано ранее Аппарат использует свою низко-уровневую модель для эффективного хранения трейтов. Игра стоит свечей. Иными словами, вы должны сами гарантировать безопасность обращения к другим UObject-ам (Actor-ам, их компонентам и т.д.) в своих трейтах, потому что такие указатели должны быть подсчитаны Garbage Collector-ом, который отсутствует.

К счастью, вы можете легко достичь этой безопасности тем, что будете ссылаться на те же объекты, что и трейты, в каких-нибудь других ассетах или объектах, или (второй вариант) будете использовать глобальный GC-управляемый инстанс UObject-а, - в обоих случаях GC не удалит объекты, на которые ссылаются трейты. Эта глобальная инстанция также может быть добавлена в корень (added to root), чтобы не стать собранной коллектором. Вы можете также добавить объекты, на которые ссылаетесь в своих трейтах, к корню, чтобы достичь того же результата.

Ссылки на остальные сущности (Subjects) через хэндлеры (Subject Handles) - прекрасное решение, поскольку управляются самим плагином. Просто запомните, что эти хэндлеры - как слабые ссылки (weak references). Они не поддерживают ссылаемую сущность, просто становятся невалидными, когда сущность была удалена.

Создание трейтов

Все Unreal структуры типа UStruct должны быть доступными в Аппарате автоматически. Если вы не можете найти свою структуру в списке структур, дважды кликните на неё в Content-браузере, чтобы она загрузилась. В целом вы должны сделать это только один раз, потому что она будет загружена автоматически, если где-либо есть на неё ссылки.

Организация в C++

Вы, главным образом, обращайтесь к официальному Unreal Engine мануалу по созданию UStruct-ов.

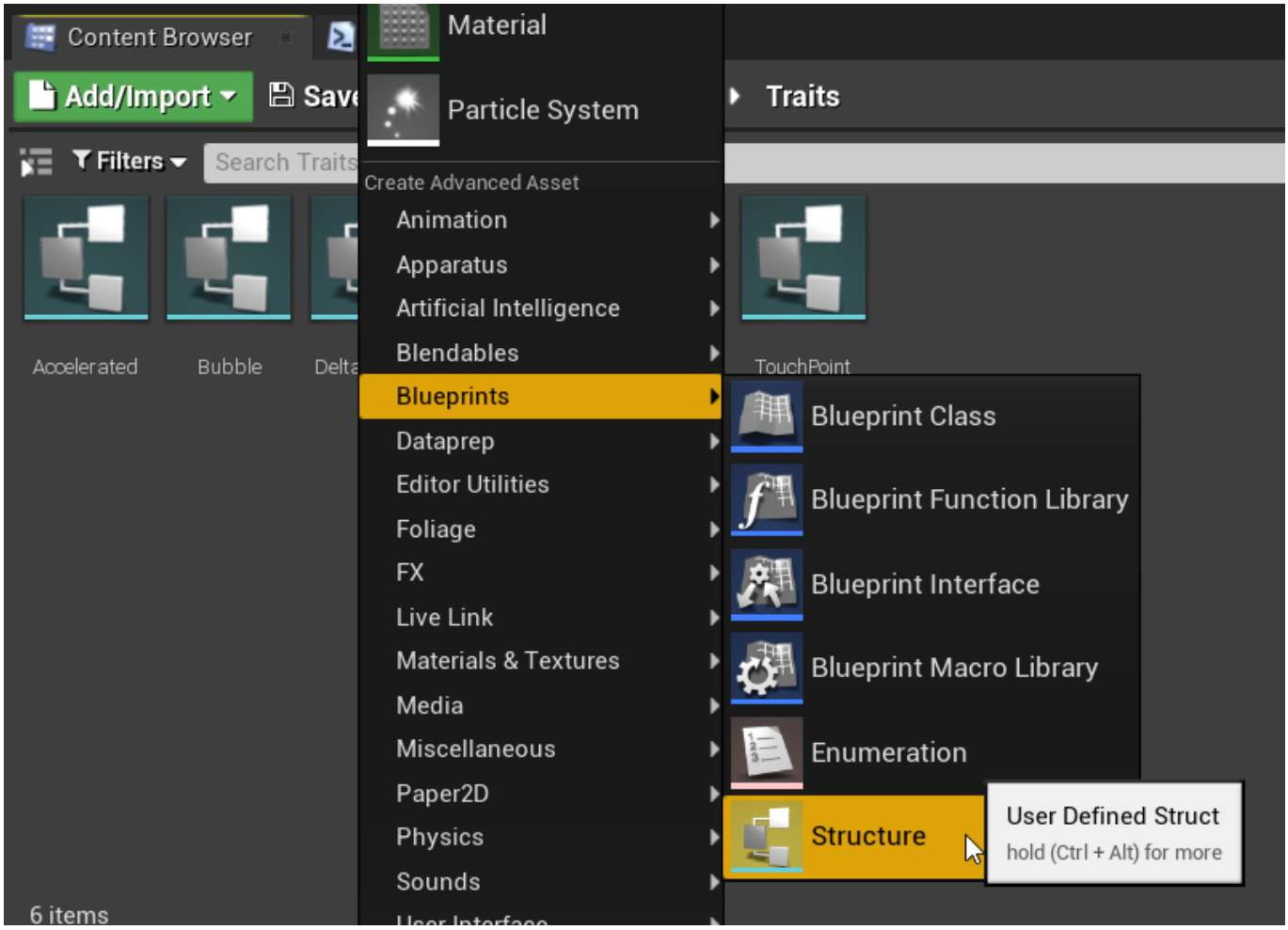
По существу, вы создаёте хеадер (`.h`) файл и (опциально) файл-ресурс (`.cpp`). Пример трейта, объявленного с помощью только заголовочного файла:

```
#pragma once

#include "CoreMinimal.h"
#include "Moving.generated.h"

/**
 * Состояние перемещения с определённой скоростью.
 */
USTRUCT(BlueprintType)
struct MY_API FMoving
{
    GENERATED_BODY()

  public:

    UPROPERTY(BlueprintReadWrite, EditAnywhere)
    float VelocityX = 0;

    UPROPERTY(BlueprintReadWrite, EditAnywhere)
    float VelocityY = 0;
};
```

Вы можете опустить спецификации UPROPERTY, но с указанными ключевыми словами у вас появляется возможность использовать `FMoving` как в C++ коде, так и в блупринтах.

Организация в Blueprint

Поскольку каждый USTRUCT в Unreal Engine является трейтом, то создать новый можно просто через Редактор. Нажмите правой кнопкой мыши на пустое место в Content Browser и выберете **Blueprints → Structure**.



Это создаст новый файл-ассет внутри выбранной папки. Вам, скорее всего, потребуется дать ему соответствующее имя после этого.

На этом всё. Созданную структуру можно использовать в своих фильтрах Filters.