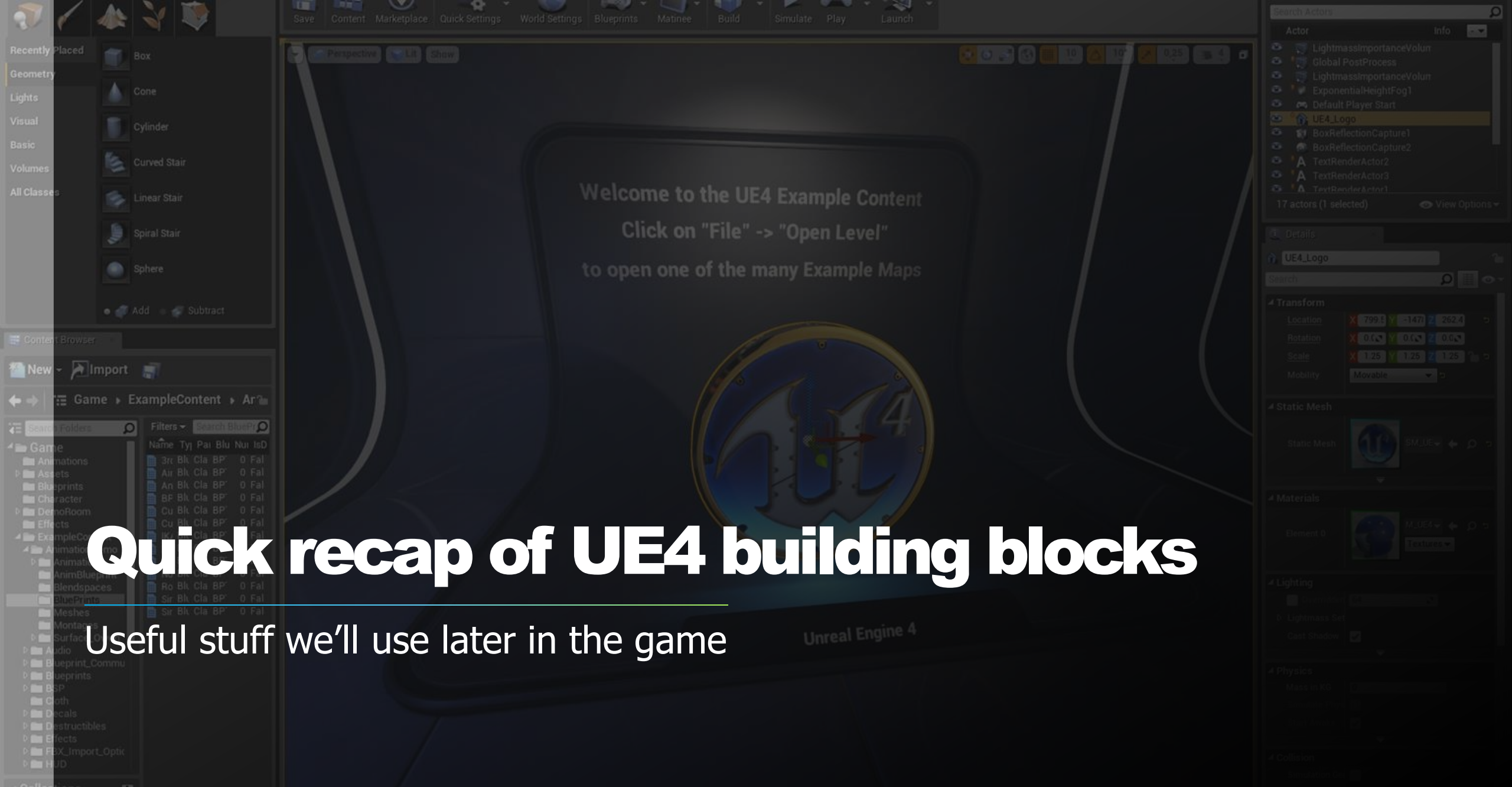


Michele Mischitelli

Our first game using UE4 and C++

... and a little bit of Blueprints ☺



Quick recap of UE4 building blocks

Useful stuff we'll use later in the game


```
#pragma once

#include "GameFramework/Actor.h"
#include "MyActor.generated.h"
```

```
UCLASS()
class AMyActor : public AActor
{
    GENERATED_BODY()

public:

    // Sets default values for this actor's properties
    AMyActor();

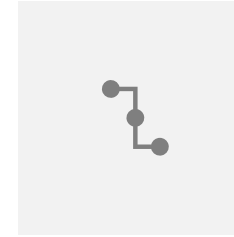
    // Called when the game starts or when spawned
    virtual void BeginPlay() override;

    // Called every frame
    virtual void Tick( float DeltaSeconds ) override;

};
```

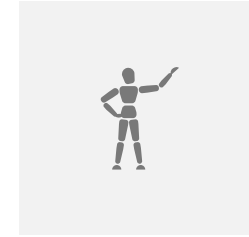
Unreal Engine 4

Blueprints, actors, delegates and subsystems



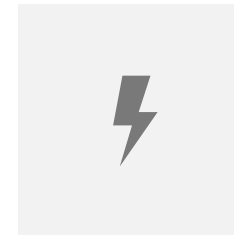
Blueprints

Used for fast prototyping, UI and less important logic



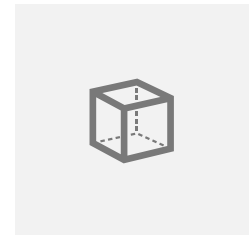
Actors

Core class that has the concept of position and components



Delegates

Dynamic binding of member functions, used for async ops



Subsystems

Automatically instantiated objects managed by UE4

Two ways of programming in Unreal



Blueprints

○ PRO

- Fast to learn (if unexperienced with C++)
- Rapid prototyping
- Mandatory for UI

○ CONS

- Slower execution
- Binary files (hard to work with in teams)
- Easy to make a mess → Hard to decode
- No support for merge/diff (although...)



C++

○ PRO

- Full access to UE4's source code
- UE4's assisted C++
- Fast execution
- Flexibility
- Source control support (merge, rebase...)

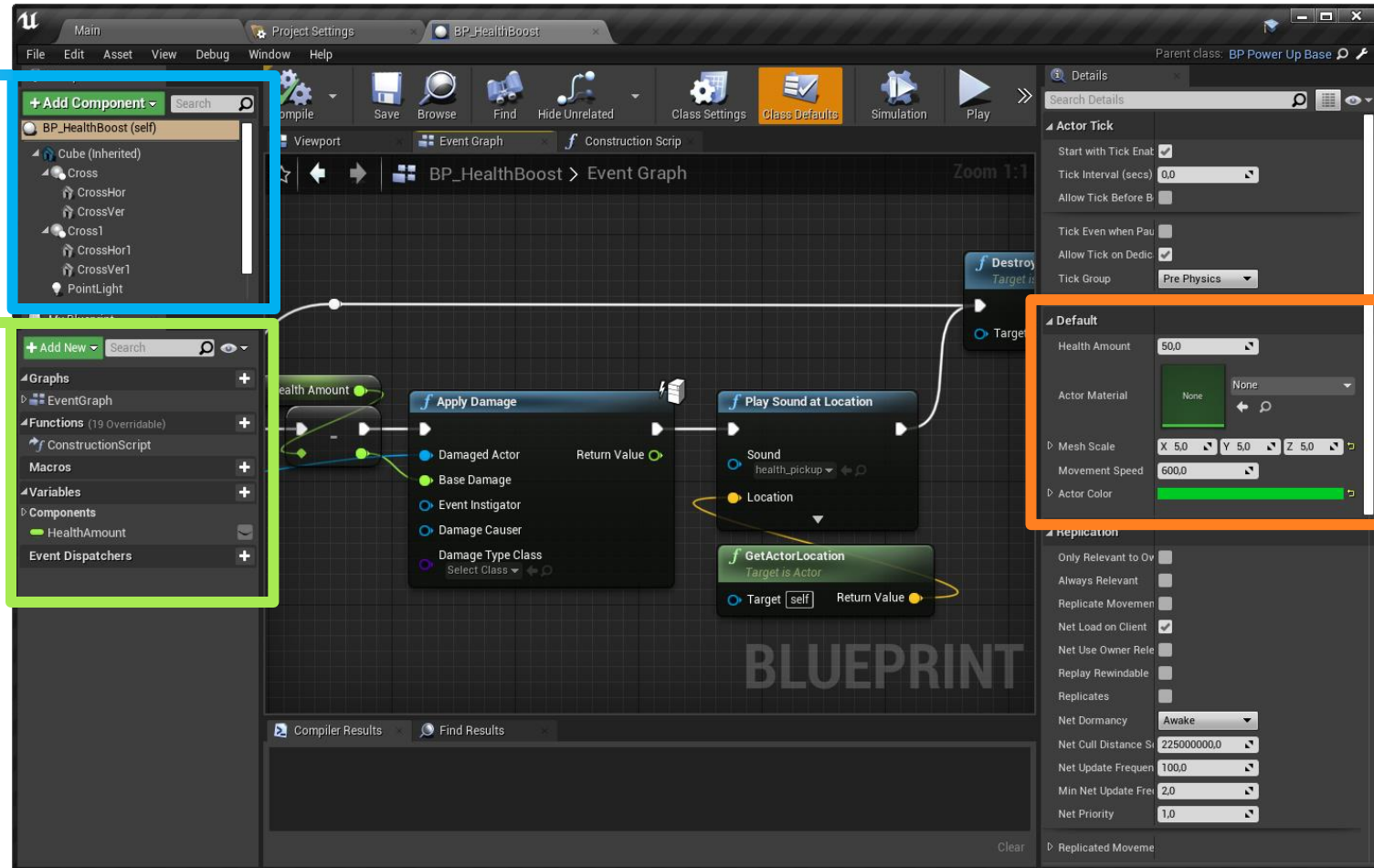
○ CONS

- Hard to learn

Hello, Blueprints

You can add more components here

Add events, functions, variables to this Actor



Here you can change properties and assign assets

Gameplay Classes

Unreal Objects: `UObject`

- Reflection of properties and methods
- Serialization of properties
- Garbage collection
- Networking support for properties and methods

Actors: `AActor`

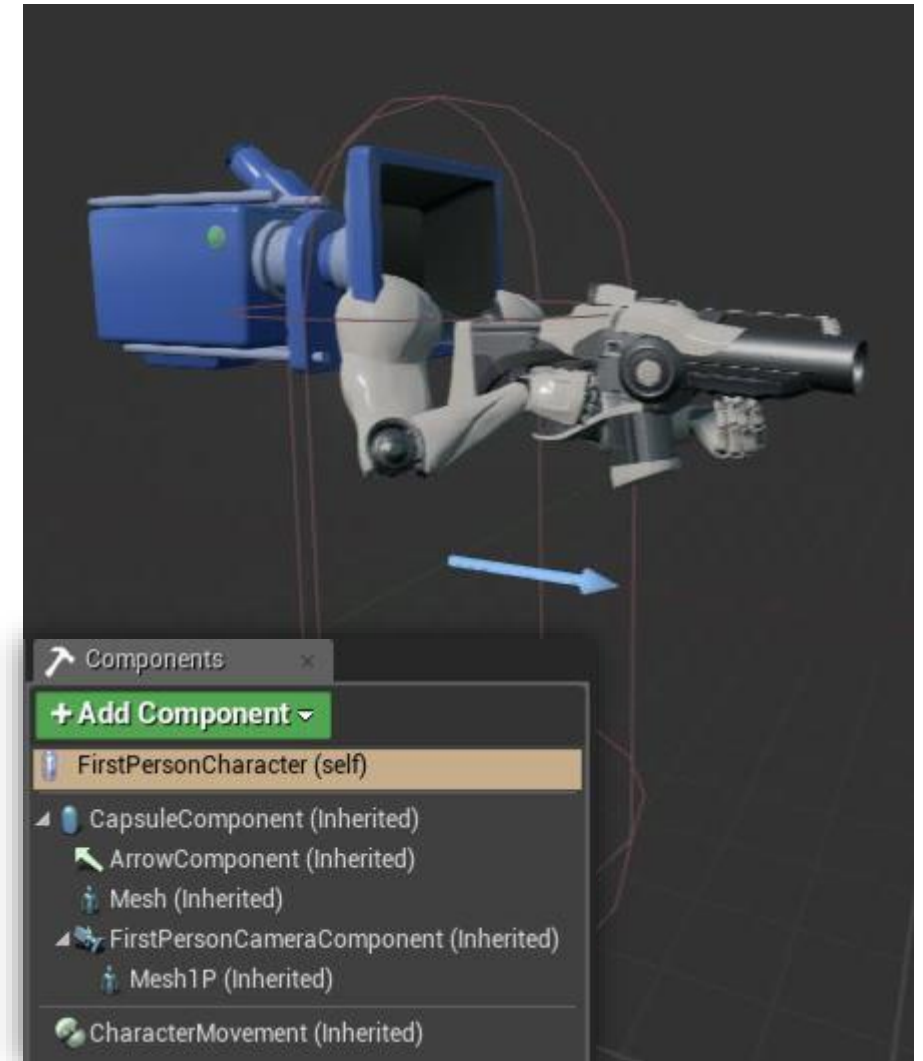
- Inherits from `UObject`, core to gameplay experience
- Objects that can be *placed*
- Composed of `UActorComponents`
- Network replication

Components: `UActorComponent`

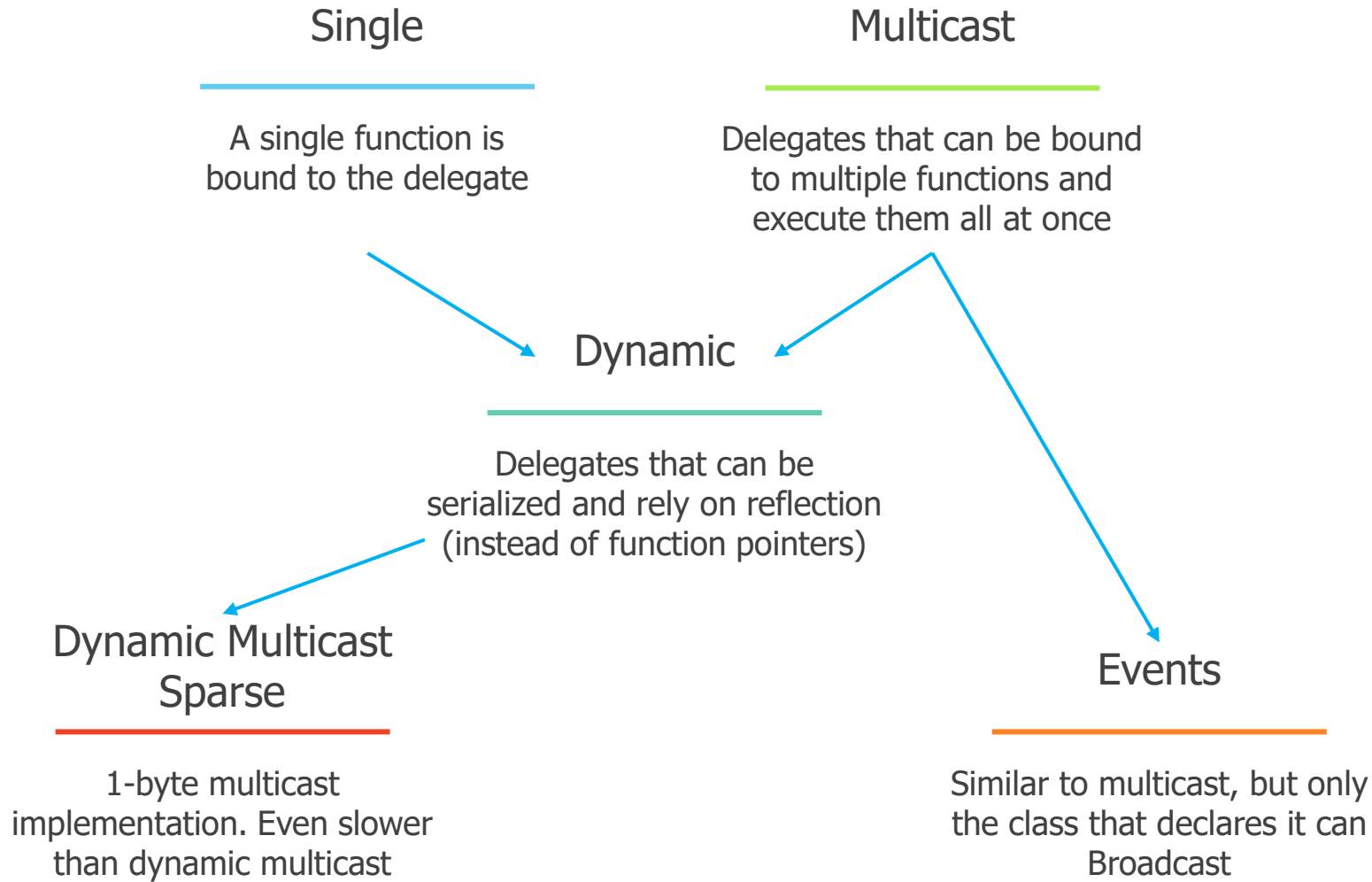
- Define their own behaviour
- Functionality that is shared across actors
- Actors are given high-level goals → components perform tasks that support those

Structs: `UStruct`

- No need to inherit from a particular class
- Just mark it with `USTRUCT()`
- Not Garbage Collected
- PODs + reflection + networking + blueprint



Delegates in UE4



- **Safe to copy**
 - Prefer passing by ref
- **Declared using MACROs**
 - In global scope
 - Inside a namespace
 - Within a class declaration
- **Support for signatures that**
 - Return a value
 - Are const
 - Have up to 8 arguments
 - Have up to 4 additional payloads

Event delegate type

```
void Function()  
DECLARE_EVENT( OwingType, EventName )  
void Function( <Param1>, ... )  
DECLARE_EVENT_<Num>Params( OwingType, EventName, Param1Type, ... )  
void Function( <Param1>, ... )  
DECLARE_DERIVED_EVENT( DerivedType, ParentType::PureEventName, OverriddenEventName )
```

It's a multicast delegate

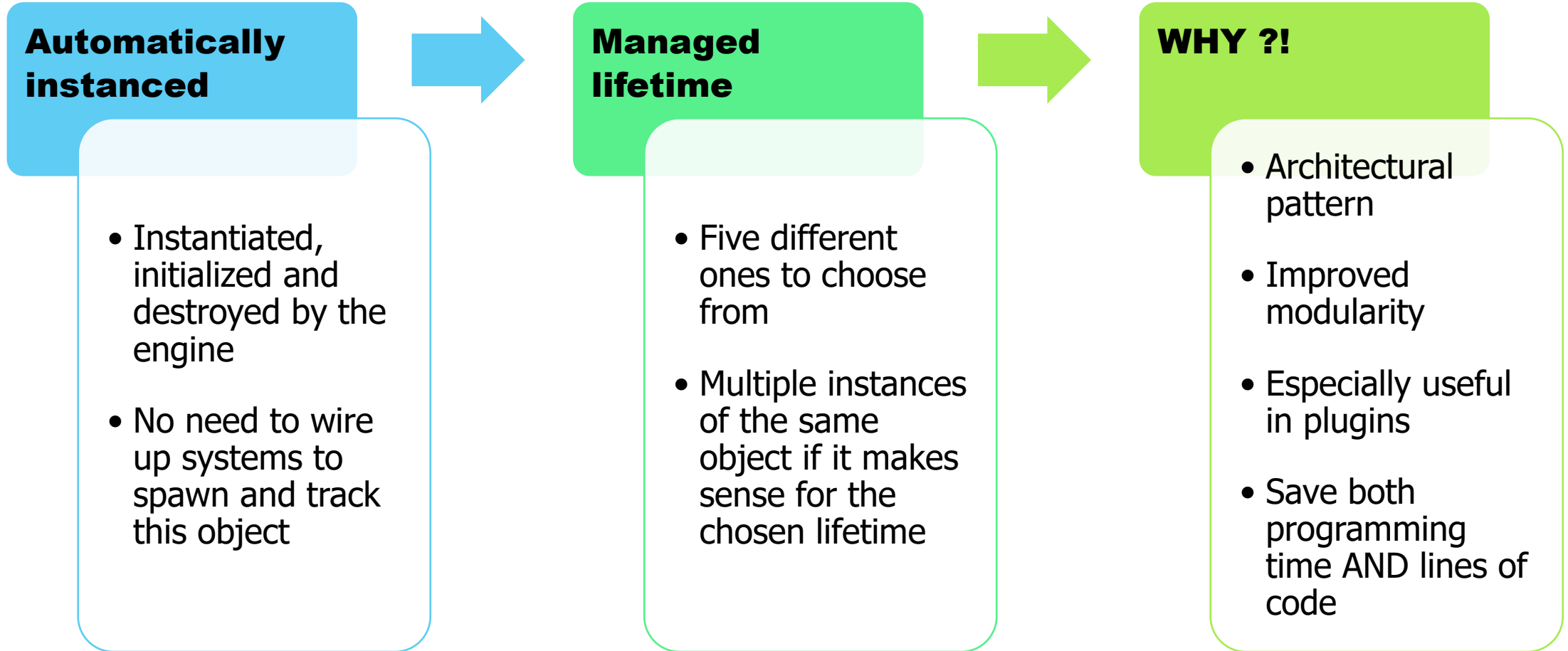
Any class can bind to events but only the one that declares it may invoke `Broadcast()`, `IsBound()` and `Clear()` functions

Event objects can be exposed in a public interface without worrying about who's going to call these functions

Use case: callbacks in purely abstract classes

`Broadcast()` is always safe to call

Subsystems intro



Subsystem lifetimes / types

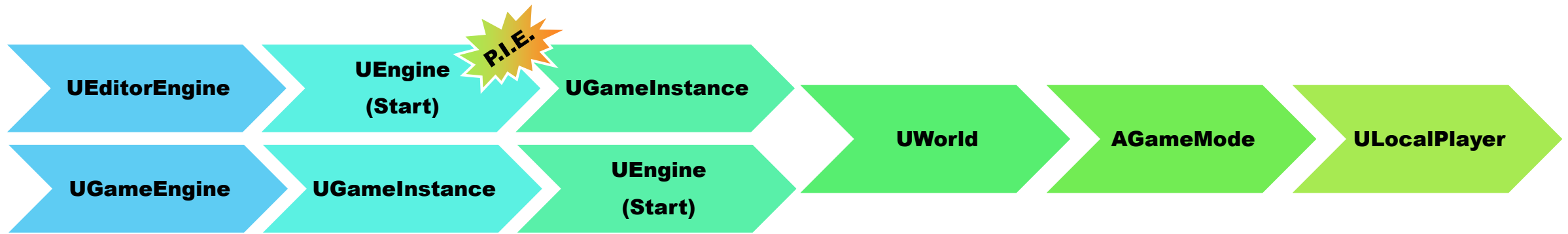
The base class you derive from determines also the lifetime of your subsystem

Game-centric Subsystems

- `UGameInstanceSubsystem`: lives before the world. Persists when changing levels (maps) in the game
- `ULocalPlayerSubsystem`: each player active on the current client is represented by an instance of `ULocalPlayer`
- `UWorldSubsystem`: a world can be a single persistent level with a list of streaming levels or composition of worlds

Advanced Subsystems

- `UEngineSubsystem`
- `UEditorSubsystem`






Let's start!

...the repo is available on GitHub 😊



Michele Mischitelli

Thank you

 [linkedin.com/in/michelemischitelli](https://www.linkedin.com/in/michelemischitelli)

 twitter.com/michelemischit1

 michelemischitelli@outlook.com

 [mmischitelli.github.io](https://github.com/mmischitelli)