TL; DR

- Projectiles carry `MoveData` from a caster to a target.
- How Projectiles do this can be customized by writing your own scripts or Blueprints.

Contents

0.1 What Is a Projectile?

A Projectile is a Actor Blueprint. It contains:

- A `UProjectileMove` that marries the physical Projectile with `MoveData`. This has required `UActorComponent`s:
  - `UProjectileDirection` to set the Projectile's direction (either initially or over time)
  - `UProjectileSpeed` to set the Projectile's speed (either initially or over time)
  - `UProjectileDamage` to control how the damage is doled out (e.g., single use vs multiple use)
  - `UExpriationComponent` to control what happens when the Projectile's use comes to an end
- A `UProjectileMovementComponent` that controls how it moves. This should not normally be altered since its settings are more flexibly set by the aforementioned components.
- The Projectile's visuals, such as a static mesh or particle effects.
- The Projectile's collider, which includes shape, collision settings (like what the Projectile interacts with), and physics settings (like if the Projectile bounces).

Of course, there might be other things there too for custom purposes. An additional note is that the `MoveData` is what decides the spawn location (via `UActorSpawnSchemes`).

While many of these are self-evident, a couple require explanation.

0.2 `UProjectileDirection`

By default, the Projectile's direction is set immediately when spawned. This optionally includes the Projectile's rotation.

Usually, the direction is retrieved from `UCombatTargetingComponent`. This could be from the player; for example, mouse position gets a vector direction or mouseover gets an Actor to inflict a status on. `TODO: While not implemented yet,` for non-players, this includes getting vectors or Actor targets via AI.

By default, a UProjectileDirection does not tick, but it might be a good idea to make a derived class that does. Examples are seeking Projectiles or a Projectile that changes behavior (e.g., explodes) after a certain time or distance.

0.3 UProjectileSpeed

This is incredibly similar to UProjectileDirection. You may want to tick it, for example, to have a Projectile that speeds up as it gets closer to its target.

0.4 UExpirationComponent

UExpirationComponents can be used on a variety of Actors, not just Projectiles. They delete an Actor after a set time or distance traveled. This could be used on not only Projectiles, but also limited-time events or objects that only appear in certain areas (and taking them outside that area deletes them).

Notably, they have one Outlet: GetExpirationDistanceOutlet. This may be useful for talents that make Projectiles last longer (for example, Junkrat's frag grenade launcher from Overwatch).