Artemis Projectile

This is a Unity component that controls a projectile.

Features

- The projectile will get the relative thickness of any object it hits and will only go through it if it's penetration value is greater then the thickness of the object.
- The projectile will reflect off of any object if the angle hit is equal to or smaller than the set ricochet angle.

Usage

- Download the Unity package and import it into your project
- Create a new C# class that inherits from ProjectileController
- Use the overridable methods to implement your custom logic

ArtemisProjectile.ProjectileController

Properties

```
// The speed of the projectile in m/s.
public float Speed { get; protected set; }

// The value by which Physics.gravity is multiplied.
public float GravityMultiplier { get; protected set; }

// The layer mask that is applied to the projectile's collisions.
public LayerMask LayerMask { get; protected set; }

// The update loop that the projectile is running in
public UpdateLoop UpdateLoop { get; }

// The velocity vector of the projectile
public Vector3 Velocity { get; }
```

Penetration

```
// Whether the projectile should be able to penetrate objects.
public bool PenetrationEnabled { get; protected set; }

// The maximum thickness the projectile can penetrate in mm. (inclusive)
public float Penetration { get; protected set; }
```

Ricochet

```
// Whether the projectile should bounce off of surfaces.
public bool RicochetEnabled { get; protected set; }

// The maximum angle at which a ricochet can occur. (inclusive)
public float RicochetAngle { get; protected set; }
```

Debug

```
// Enable debbuging tools.
public bool DebugEnabled { get; protected set; }

// Debug lines will keep rendering after the proectile is destroyed.
public bool IgnoreDestroy { get; protected set; }

// The color path lines will be drawn
public Color PathColor { get; protected set; }

// The color the path through an object is drawn
public Color PenetrationColor { get; protected set; }

// The color normals will be drawn
public Color NormalColor { get; protected set; }
```

Overridable Methods

```
// Summary:
//     Called when the projectile sucssesfully penetrates an object
//
// Parameters:
//   entry:
//     The RaycastHit of when the projectile entered the object
//
//   velocity:
//     The velocity at which the projectile struck the object
//
//   thickness:
//     The reletive thickness of the object
protected virtual void OnPenetrationEnter(RaycastHit entry, Vector3 velocity,
float thickness) { }

// Summary:
//     Called when the projectile exits a penetrated object
//
// Parameters:
//   exit:
//     The RacastHit of when the projectile exited the object
//
```

```
//   velocity:
//     The velocity at which the projectile exited the object
protected virtual void OnPenetrationExit(RaycastHit exit, Vector3 velocity) { }

// Summary:
//     Called when the Projectile fails to penetrate an object. if penetration is
disabled,
//     this will always be called upon collision.
//
// Parameters:
//   hit:
//     The RaycastHit of the collision with the object
//
//   velocity:
//     The velocity at which the projectile struck the object
//   thickness:
//     The reletive thickness of the object
protected virtual void OnPenetrationFailed(RaycastHit hit, Vector3 velocity, float
thickness) { }

// Summary:
//     Called when the projectile ricochets off of an object
//
// Parameters:
//   hit:
//     The RaycastHit of the contact with the surface
//
//   inAngle:
//     The angle that the projectile hit the surface.
//
//   entryVelocity:
//     The velocity the projectile hit the surface.
//
//   exitVelocity:
//     The velocity of the projectile after reflection.
protected virtual void OnRicochet(RaycastHit hit, float inAngle, Vector3
entryVelocity, Vector3 exitVelocity) { }
```

Technologies

- C#
- F#