Menu

Scripting Reference

VelocityByA

Calculates the launch velocity by the coefficient ***a*** of the quadratic function $f(x) = ax^2 + bx + c$ which determines the trajectory of the projectile motion.

```
public static Vector3 VelocityByA(Vector3 start, Vector3 end, float a)
```

`start`: The starting point of the projectile motion.

`end`: The target point you want the projectile motion to hit or pass through.

`a`: The ***a*** coefficient of the quadratic function $f(x) = ax^2 + bx + c$. It determines the shape and speed of the trajectory, for example, -0.2f makes the trajectory curvier and slower while -0.01f makes it straighter and faster. Should always be negative.

VelocityByAngle

Calculates the launch velocity by a given launch angle in degrees.

```
public static Vector3 VelocityByAngle(Vector3 start, Vector3 end, float
elevationAngle)
```

`start`: The starting point of the projectile motion.

`end`: The target point you want the projectile motion to hit or pass through.

`elevationAngle`: The launch angle in degrees. 0 means launch horizontally. Should be from -90f (exclusive) to 90f (exclusive) and greater than the elevation angle formed by `start` to `end`.

VelocityByTime

Calculates the launch velocity by the time in seconds the projectile flies from `start` to `end`. The projectile object will be exactly at the end point `time` seconds after launch.

```
public static Vector3 VelocityByTime(Vector3 start, Vector3 end, float time)
```

`start`: The starting point of the projectile motion.

`end`: The target point you want the projectile motion to hit or pass through.

`time`: The time in seconds you want the projectile to fly from `start` to `end`.

VelocityByHeight

Calculates the launch velocity by a given max height of the projectile motion.

```
public static Vector3 VelocityByHeight(Vector3 start, Vector3 end, float
heightFromEnd)
```

`start`: The starting point of the projectile motion.

`end`: The target point you want the projectile motion to hit or pass through.

`heightFromEnd`: The height measured from the `end` point (for example, 1f means the max height of the trajectory is 1 meter above the end point). The algorithm automatically clamps the value if it is lower than the `y` value of `start` or `end`.

AnglesBySpeed

Calculates the two angle results by a given launch speed. Returns `false` if out of reach.

```
public static bool AnglesBySpeed(Vector3 start, Vector3 end, float speed, out
float lowAngle, out float highAngle)
```

`start`: The starting point of the projectile motion.

`end`: The target point you want the projectile motion to hit or pass through.

`speed`: The launch speed of the projectile object.

`lowAngle`: The lower angle that satisfies the conditions, or 0 if the method returns false.

`highAngle`: The higher angle that satisfies the conditions, or 0 if the method returns false.

VelocitiesBySpeed

Calculates the two velocity results by a given launch speed. Returns `false` if out of reach.

```
public static bool VelocitiesBySpeed(Vector3 start, Vector3 end, float speed,
out Vector3 lowAngleForce, out Vector3 highAngleForce)
```

`start`: The starting point of the projectile motion.

`end`: The target point you want the projectile motion to hit or pass through.

`speed`: The launch speed of the projectile object.

`lowAngleForce`: The lower-angle force that satisfies the conditions, or (0, 0, 0) if the method returns false.

`highAngleForce` The higher-angle force that satisfies the conditions, or (0, 0, 0) if the method returns false.

> **Note**
>
> If `AnglesBySpeed` or `VelocitiesBySpeed` returns `true`, then there are always two effective and different `out` results. This is mathematically correct. One special case is that when the `start` and the `end` form exactly the maximum range that the `speed` can reach, the two `out` results will be the same. No matter whether the return value is true or false, any value originally supplied in `out ...` will be overwritten.
