Math for Games

CS 3540 – Game Programming

Roadmap

Vectors

Vector Operations

Elementary Trigonometry

Vectors – Mathematically

A list or array of numbers

The dimension of a vector describes the number of numbers contained in the vector

2-dimensional vector ➔ a = [1, 3]

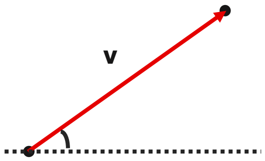
3-dimensional vector ➔ b = [3, 5, 4]

4-dimensional vector ➔ c = [0, 4, -1, 2]

3

Vectors – Geometrically

A **line** segment that has **magnitude** and **direction**



Magnitude

the length of the vector

Direction

which way the vector is pointing in space

4

Vector Operations: Scaling

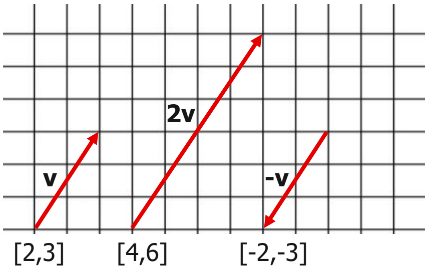
Can multiple a vector by a scalar

- The result is a vector parallel to the original vector
- With a different length
- And potentially opposite direction (if a negative scalar)

Scaling a vector

$\mathbf{v} = [x, y, z]$

$k\,\mathbf{v} = [kx, ky, kz]$



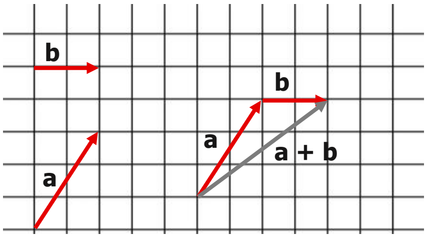
Vector Operations: Addition

To add two vectors, add the corresponding components

$\mathbf{a} = [x_1, y_1, z_1]$

$\mathbf{b} = [x_2, y_2, z_2]$

$\mathbf{a} + \mathbf{b} = [x_1 + x_2, y_1 + y_2, z_1 + z_2]$



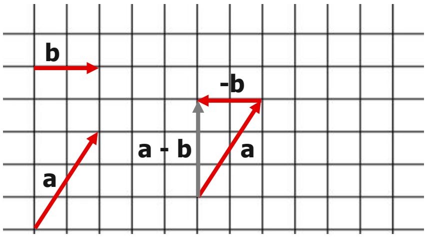
Vector Operations: Subtraction

Subtraction can be interpreted as adding the negative

$\mathbf{a} = [x_1, y_1, z_1]$

$\mathbf{b} = [x_2, y_2, z_2]$

$\mathbf{a} - \mathbf{b} = [x_1 - x_2, y_1 - y_2, z_1 - z_2]$



Vector3 Class in Unity

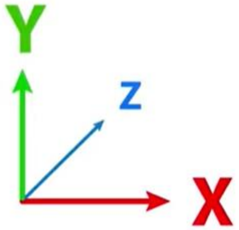
Representation of 3D vectors and points.

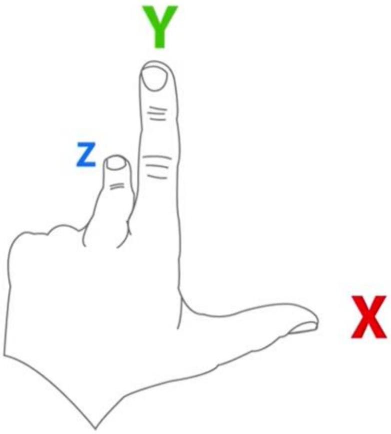
Used to pass 3D positions and directions around

Contains functions for performing common vector operations

Unity 3D Coordinates

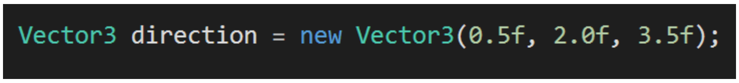
Unity uses a left-handed coordinate system





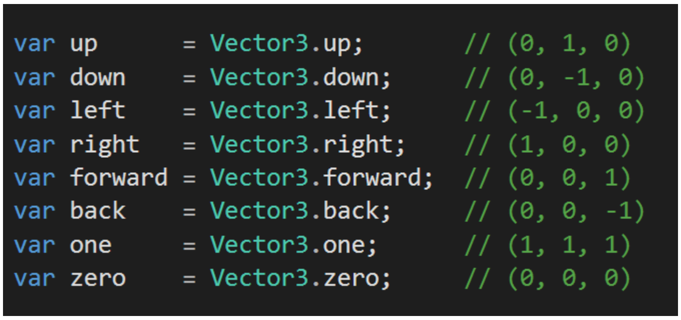
Vector3 Class in Unity

Defining a Vector3:



```
Vector3 direction = new Vector3(0.5f, 2.0f, 3.5f);
```

Built-in Vector3 definitions:



```
var up      = Vector3.up;       // (0, 1, 0)
var down    = Vector3.down;     // (0, -1, 0)
var left    = Vector3.left;     // (-1, 0, 0)
var right   = Vector3.right;    // (1, 0, 0)
var forward = Vector3.forward;  // (0, 0, 1)
var back    = Vector3.back;     // (0, 0, -1)
var one     = Vector3.one;      // (1, 1, 1)
var zero    = Vector3.zero;     // (0, 0, 0)
```

Frame Rate-Independent Behaviors

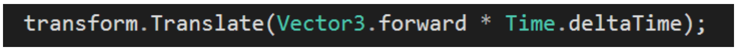
Time.deltaTime

Provides the time between the current and previous frame.

Time between each Update or FixedUpdate function call

Can be used to smooth out values used for movement and other incremental calculations

With Time.deltaTime

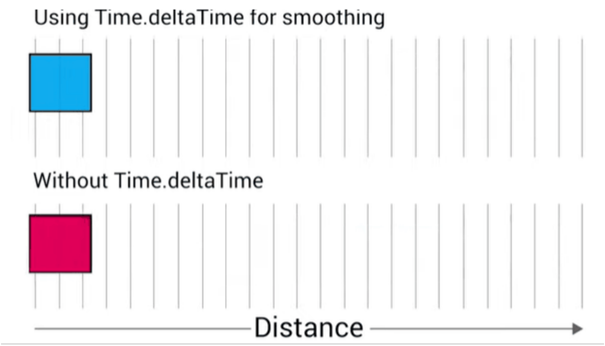


```
transform.Translate(Vector3.forward * Time.deltaTime);
```

Changes in values occur **per second** rather than per frame

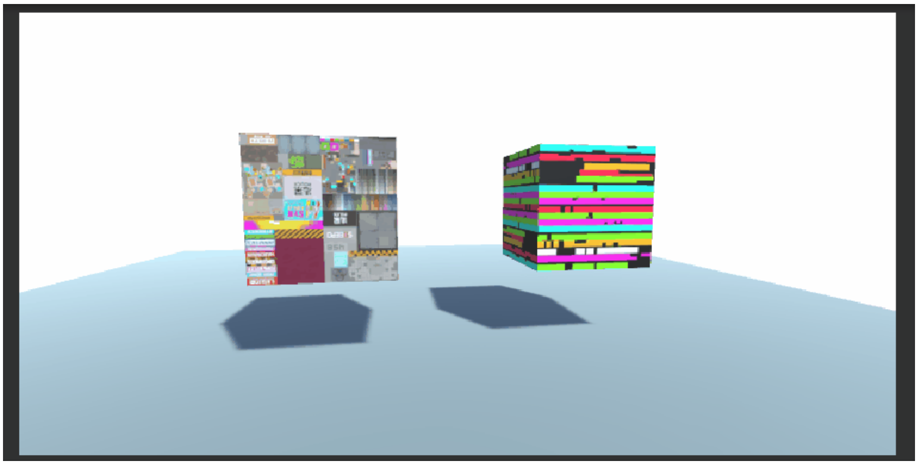
11

Frame Rate-Independent Behaviors



12

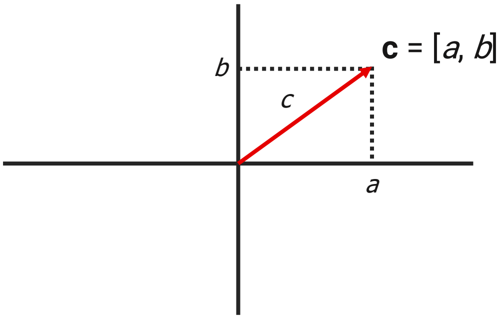
Frame Rate-Independent Behaviors



Vector Operations: Magnitude

The magnitude of the vector, ||**v**|| is the length of the vector, **v**

Distance between two points, the tail and head



$c = ||\mathbf{c}||$

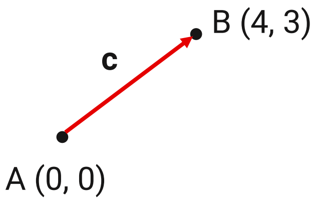
$c = \sqrt{a^2 + b^2}$

$\mathbf{v} = [x, y, z]$

$||\mathbf{v}|| = \sqrt{x^2 + y^2 + z^2}$

Vector Operations: Magnitude

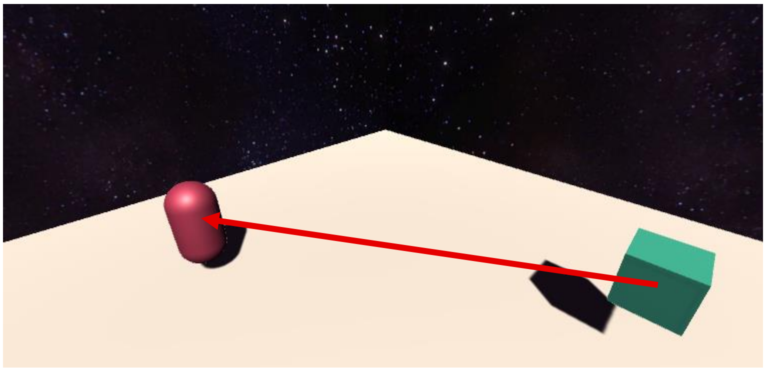
Distance between two points, or two objects in 2D



The magnitude of the vector from A to B gives you the distance between A and B:

$c = ||\mathbf{c}|| = 5$

Vector Operations: Magnitude



Player Position:

$[x_1, y_1, z_1]$

Enemy Position:

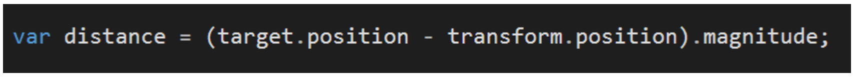
$[x_2, y_2, z_2]$

Distance from Enemy to Player:

$$D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}$$

Vector3 Magnitude Property

Returns the length of the vector on which it is called



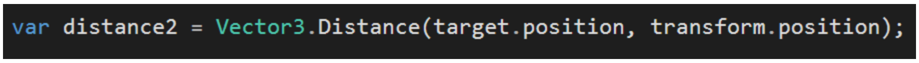
```
var distance = (target.position - transform.position).magnitude;
```

17

Vector3.Distance()

Returns the distance between **a** and **b**.

Identical to (**a** – **b**).magnitude



```
var distance2 = Vector3.Distance(target.position, transform.position);
```

18

Vector Operations: Direction

The direction of a vector is usually represented using unit vectors

A unit vector has a magnitude of one

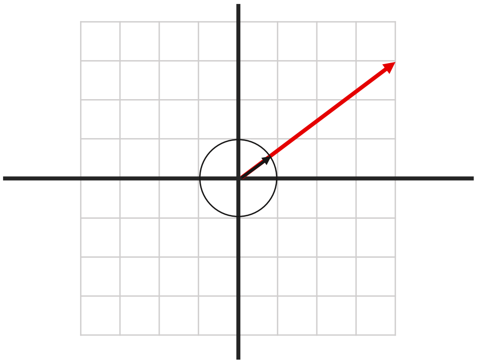
A unit vector is a normalized vector that points in the same direction as the original vector

To normalize a vector, divide the vector by its magnitude

$$\hat{\mathbf{v}} = \frac{\mathbf{v}}{||\mathbf{v}||}$$

19

Vector Operations: Direction



$\mathbf{v} = [4, 3]$

$$\hat{\mathbf{v}} = \frac{[4\ \ 3]}{5} = \left[\ \frac{4}{5}\ \ \frac{3}{5}\ \right]$$

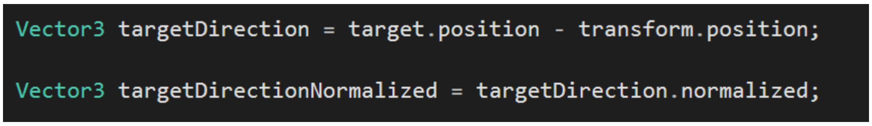
$\hat{\mathbf{v}} = [0.8\ \ 0.6]$

20

Vector3 Normalized

Returns the normalized Vector3

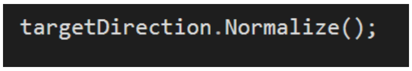
Does not modify the original Vector3



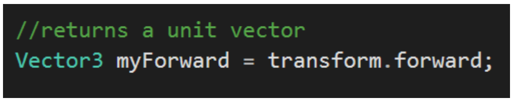
```
Vector3 targetDirection = target.position - transform.position;

Vector3 targetDirectionNormalized = targetDirection.normalized;
```

Normalize() method normalizes and modifies the vector



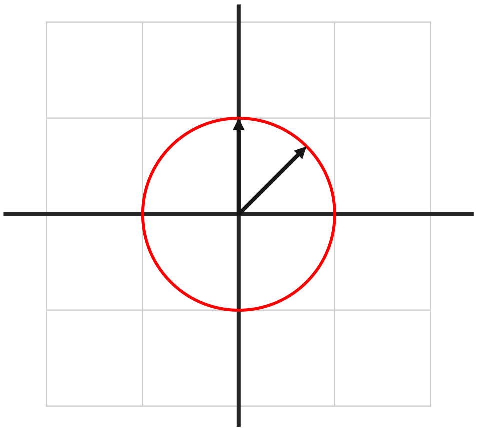
```
targetDirection.Normalize();
```



```
//returns a unit vector
Vector3 myForward = transform.forward;
```

21

Unit Circle

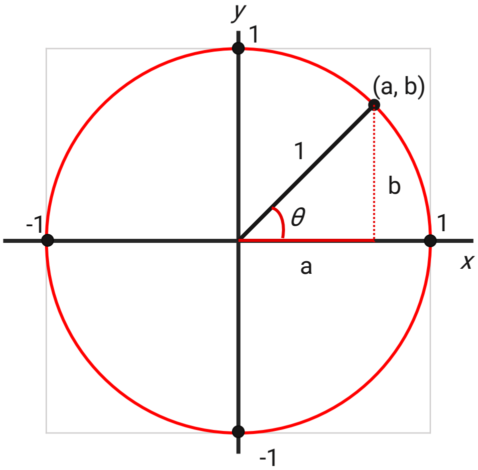


A unit circle has a radius of 1

If we draw a unit vector with the tail at the origin, the head of the vector will touch a unit circle centered at the origin

22

Unit Circle



A unit circle has a radius of 1

$$\cos\theta = \frac{\text{adjacent}}{\text{hypotenuse}} = a$$

$$\sin\theta = \frac{\text{opposite}}{\text{hypotenuse}} = b$$

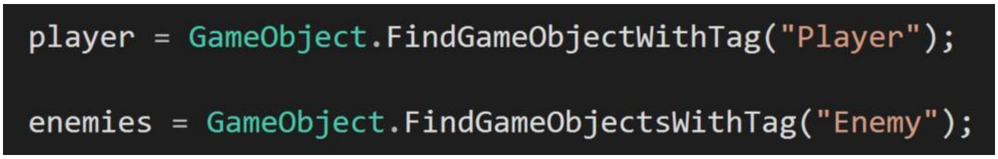
$$\tan\theta = \frac{\text{opposite}}{\text{adjacent}} = b/a$$

23

FindGameObjectWithTag()

Finds and returns an active GameObject with the specified tag

Returns null if no object has the specified tag



```
player = GameObject.FindGameObjectWithTag("Player");

enemies = GameObject.FindGameObjectsWithTag("Enemy");
```

24

Readings

https://docs.unity3d.com/ScriptReference/Vector3.html

https://docs.unity3d.com/ScriptReference/Time-deltaTime.html

https://docs.unity3d.com/ScriptReference/Vector3-magnitude.html

https://docs.unity3d.com/ScriptReference/Vector3.Normalize.html

https://docs.unity3d.com/ScriptReference/Vector3.Distance.html

https://docs.unity3d.com/ScriptReference/GameObject.FindWithTag.html