Enemy Contact Damage To Player

Layer Masks and Bitwise Operators

Enemy Contact Damage To Player

Layer Masks

Unity uses ‘Layers’ to help control how different game objects interact with each other.

In the Unity editor, 32 layers can be defined. The first 5 layers are defined by Unity, the remaining 27 layers can be defined by the user.

The 32 layers can be represented by a binary digit in a 32 bit binary number

i.e. 0000 0000 0000 0000 0000 0000 0000 0000

The first layer (0:Default) can be represented by

0000 0000 0000 0000 0000 0000 0000 0001

To represent multiple layers, we just put a 1 in the binary number in the position representing the layer. So to represent all 5 layers defined by Unity we would write:-

0000 0000 0000 0000 0000 0000 0001 1111

By specifying the layers in this way (known as a layer mask) we can use binary logical operations to check if a layer is in a layer mask (or if multiple layers are in a layer mask)

Enemy Contact Damage To Player

Bit Shifting Notation

Binary Number	Bitshift Notation	Decimal Number
00000001	1<<0	1
00000010	1<<1	2
00000100	1<<2	4
00001000	1<<3	8
00010000	1<<4	16
00100000	1<<5	32
01000000	1<<6	64
10000000	1<<7	128

Enemy Contact Damage To Player

Unity Layers & Bit Shifting Notation

Layer	Layer Name	Bitshift Notation	Binary Value	Decimal Value
Layer 0	Default	1<<0	00000000001	1
Layer 1	TransparentFX	1<<1	00000000010	2
Layer 2	Ignore Raycast	1<<2	00000000100	4
Layer 3	Water	1<<3	00000001000	8
Layer 4	UI	1<<4	00000010000	16
Layer 5		1<<5	00000100000	32
Layer 6		1<<6	00001000000	64
Layer 7	Ignore Ammo	1<<7	00010000000	128
Layer 8	Camera 1	1<<8	00100000000	256
Layer 9	Camera 2	1<<9	01000000000	512
Layer 10	Player	1<<10	10000000000	1024

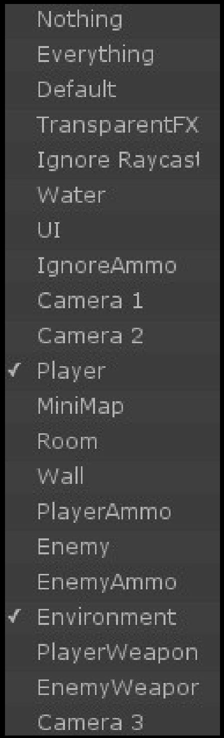
Enemy Contact Damage To Player

Defining Layer Masks In The Unity Editor

In Unity you can use the “LayerMask” type if you want to define a layer mask in the editor

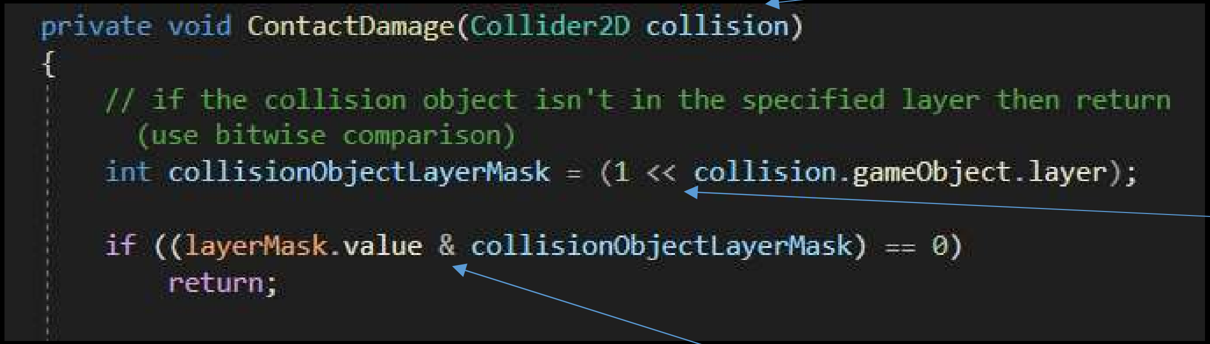
In the inspector in the editor you can then select multiple layers to add to the mask

The LayerMask variable will then be set to a value that represents all the binary digits representing the individual layers added together



Enemy Contact Damage To Player

Using Binary Operations To Test Layers



```
private void ContactDamage(Collider2D collision)
{
    // if the collision object isn't in the specified layer then return
      (use bitwise comparison)
    int collisionObjectLayerMask = (1 << collision.gameObject.layer);

    if ((layerMask.value & collisionObjectLayerMask) == 0)
        return;
```

1. In this example we have collided with an object (Collider2D collision).

2. We want to test if this collision object is on a layer that has been specified in the layerMask in the editor.

3. So first of all we use bit shifting to convert the integer layer value (i.e. 0 to 31) of the collision object to a bit shifted integer value. e.g. the Player layer 10 would be converted to a bit shifted binary value of 10000000000 or decimal 1024.

Layer	Layer Name	Bitshift Notation	Binary Value	Decimal Value
Layer 10	Player	1<<10	10000000000	1024

4. Next we use the binary Bitwise AND operator "&" to compare each bit of the layer mask with each bit of the collision object layer mask. Where there is a 1 in both bits – the result will be 1 (so we have a matching layer).

5. We return if there are 'no matching bits' (i.e. AND operation == 0) which means our collision object layer isn't one of the layerMask layers.

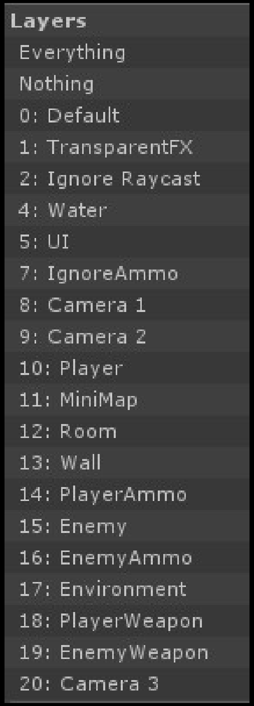
Enemy Contact Damage To Player

Example 1

So for the example we want to test whether the collision object is in either the Player layer or the Environment layer. In this example the collision object will be on the Default layer 0.

Player layer 10	000000010000000000
Environment layer 17	100000000000000000
	Add
Layer Mask	100000010000000000
	& (each binary digit in both arguments is “Anded” together)
Collision Object	000000000000000001 (Default Layer)
Bitwise And “&” result	000000000000000000 (0 for any binary digit where there isn’t a 1 in BOTH arguments)

First we create the layer mask for the Player and Environment layers



Layers
Everything
Nothing
0: Default
1: TransparentFX
2: Ignore Raycast
4: Water
5: UI
7: IgnoreAmmo
8: Camera 1
9: Camera 2
10: Player
11: MiniMap
12: Room
13: Wall
14: PlayerAmmo
15: Enemy
16: EnemyAmmo
17: Environment
18: PlayerWeapon
19: EnemyWeapon
20: Camera 3

The result = 0. The collision object didn’t match any of the layers specified in the layer mask

Enemy Contact Damage To Player

Example 2

In this example the collision object will be on the Player layer 10, and we'll test again to see whether the collision object is in either the Player layer or the Environment layer.

Player layer 10 000000010000000000

Environment layer 17 100000000000000000

Add

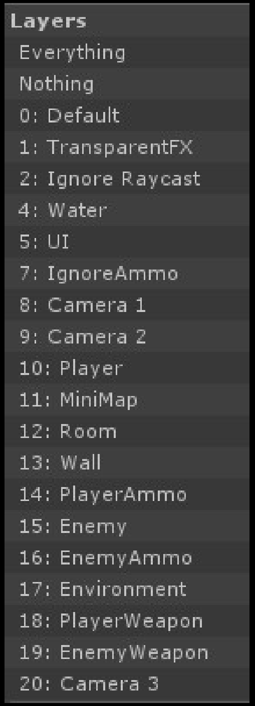
Layer Mask 100000010000000000

First we create the layer mask for the Player and Environment layers

& (each binary digit in both arguments is "Anded" together)

Collision Object 000000010000000000 (Player Layer)

Bitwise And "&" result 000000010000000000 (1 for any binary digit where there is a 1 in BOTH arguments)



This time the result != 0 . The collision object did match the Player layer in the layer mask.