Common Issues in OOP

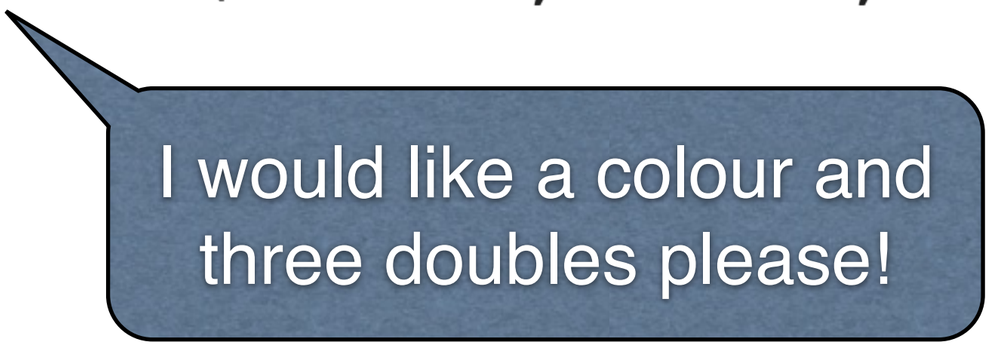
Charlotte Pierce

SWINBURNE UNIVERSITY OF TECHNOLOGY

We call methods a lot in object-oriented programming

Parameters define the **expectations** of a method

```
public static void DrawCircle(Color clr, double x, double y, double radius)
```



When we call a method we need to **fulfil** those expectations

```
SplashKit.DrawCircle(Color.Black, X, Y, 10);
```

Please draw a circle for me, here is a colour and three doubles

```
public static void DrawCircle(Color clr, double x, double y, double radius)
```

Ok got it!

When we call a method we need to **fulfil** those expectations

```
SplashKit.DrawCircle(Color.Black, true, Y, Radius + 1);
```

readonly struct System.Boolean

Represents a Boolean (true or false) value.

Argument 2: cannot convert from 'bool' to 'double'

Show potential fixes

Please draw a circle for me, here is a colour, a boolean, and two doubles

```
public static void DrawCircle(Color clr, double x, double y, double radius)
```

I have no idea how to do that…

The method already knows what kind of data it is expecting

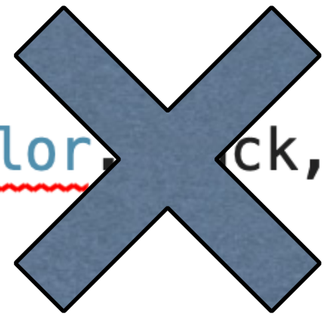
```
SplashKit.DrawCircle(Color.Black, X, Y, 10);
```

```
SplashKit.DrawCircle(Color Color.Black, double X, double Y, double 10);
```

The method already knows what kind of data it is expecting

```
SplashKit.DrawCircle(Color.Black, X, Y, 10);
```

```
SplashKit.DrawCircle(Color Color.  ck, double X, double Y, double 10);
```



The names of the parameters don’t matter!

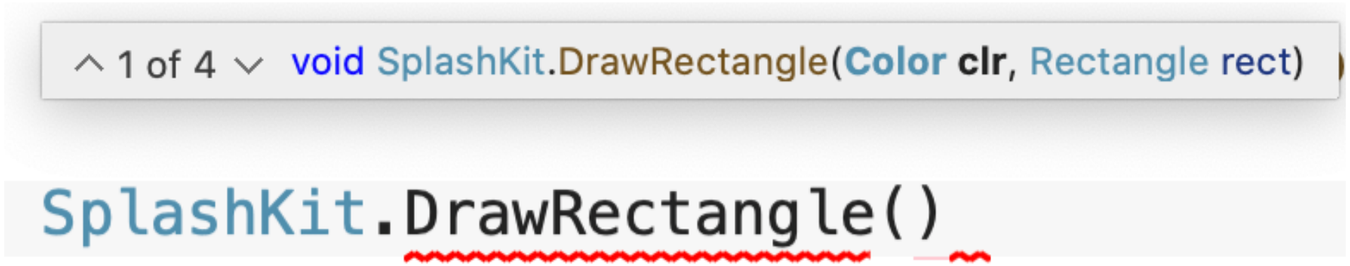
```
public static void DrawCircle(Color , double , double , double )
```

I would like a colour and three doubles

We can **overload** methods with different expectations

Methods can have the same name as long as they have different parameters

```
public static void DrawRectangle(Color clr, Rectangle rect)...
public static void DrawRectangle(Color clr, Rectangle rect, DrawingOptions opts)...
public static void DrawRectangle(Color clr, double x, double y, double width, double height)...
public static void DrawRectangle(Color clr, double x, double y, double width, double height, DrawingOptions opts)
```



Often overloaded methods will call each other

```
public Shape(Color shapeColor)
{
    ShapeColor = shapeColor;
    X = 0;
    Y = 0;
    Selected = false;
}

public Shape() : this(Color.Yellow) { }
```

Only overload when you **need** to

It’s important to understand the **scope** of a variable

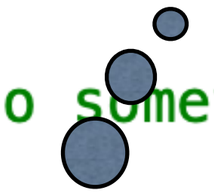
Scope tells us **where** a variable can be accessed

I know about a variable called ‘i’!

```
public void MethodOne()
{
    int i;

    for (i = 0; i < 10; i++)
    {
        // do something
    }
}
```

```
public void MethodTwo()
{
    // do something
}
```



What is an ‘i’?

Scope tells us **where** a variable can be accessed

I know about a variable called ‘i’!

```
public void MethodOne()
{
    int i;

    for (i = 0; i < 10; i++)
    {
        // do something
    }
}
```

I know about a different ‘i’! Neat!

```
public void MethodTwo()
{
    int i;

    for (i = 0; i < 10; i++)
    {
        // do something
    }
}
```

Scope tells us **where** a variable can be accessed

```
public void MethodOne()
{
    int i;

    for (i = 0; i < 10; i++)
    {
        // do something
    }

    int i;
}
```

Scope tells us **where** a variable can be accessed

```
public void MethodOne()
{
    int i;

    for (i = 0; i < 10; i++)
    {
        // do something
    }

    int i;
```

Scope tells us **where** a variable can be accessed

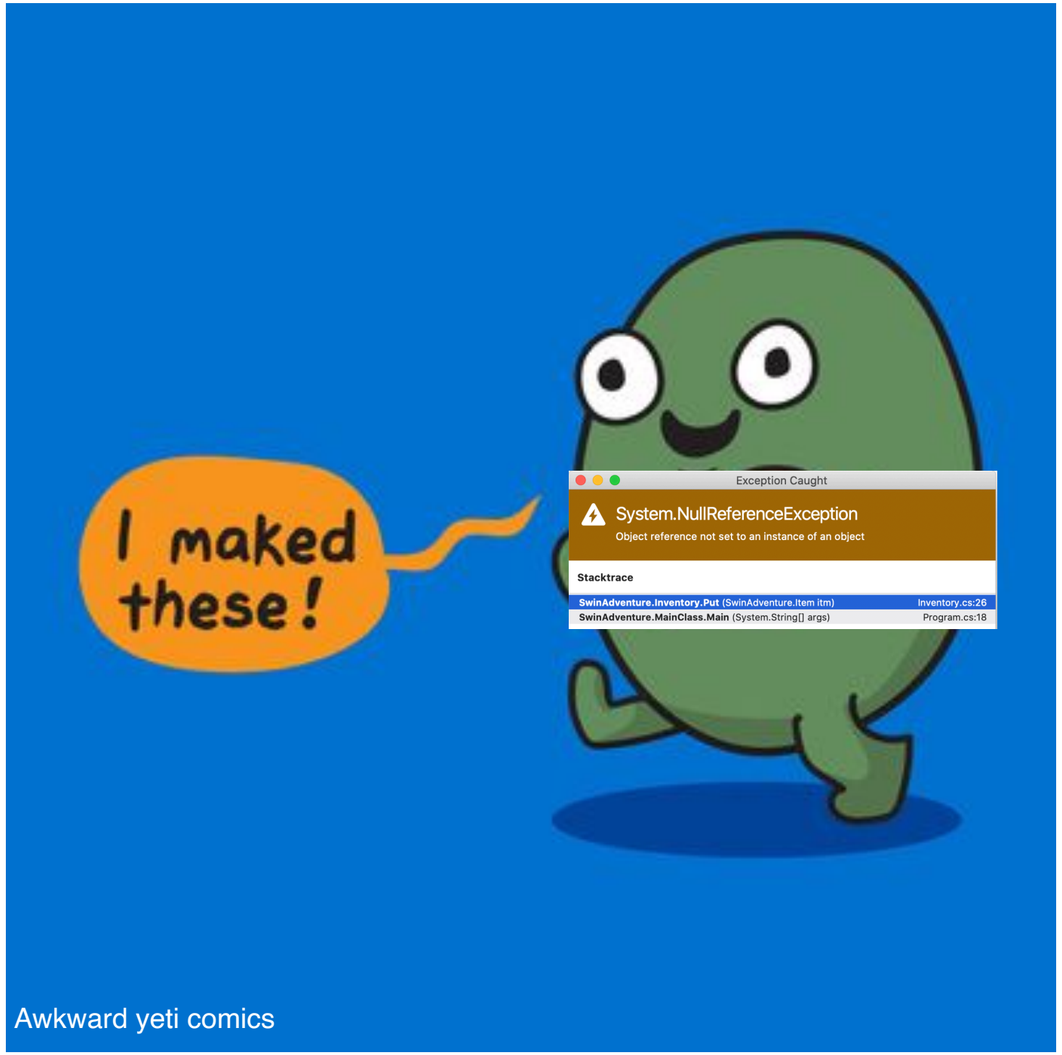
```
public void MethodOne(int X)
{
    // do something
}

public void MethodTwo(int X)
{
    // do something
}
```

Scope can be for a **method**,
a **class**, or a **namespace**

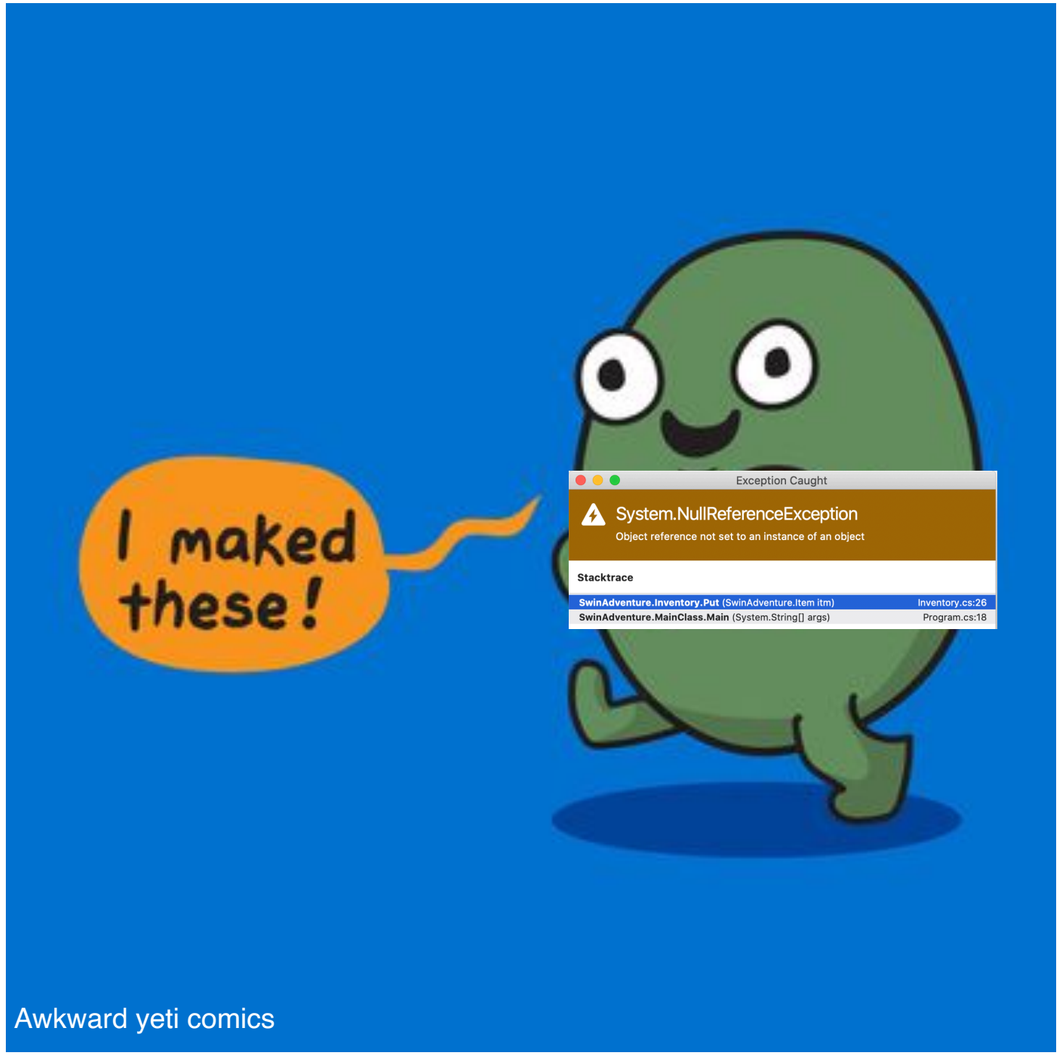
Let’s talk about error messages…

The computer is *trying* to help you…



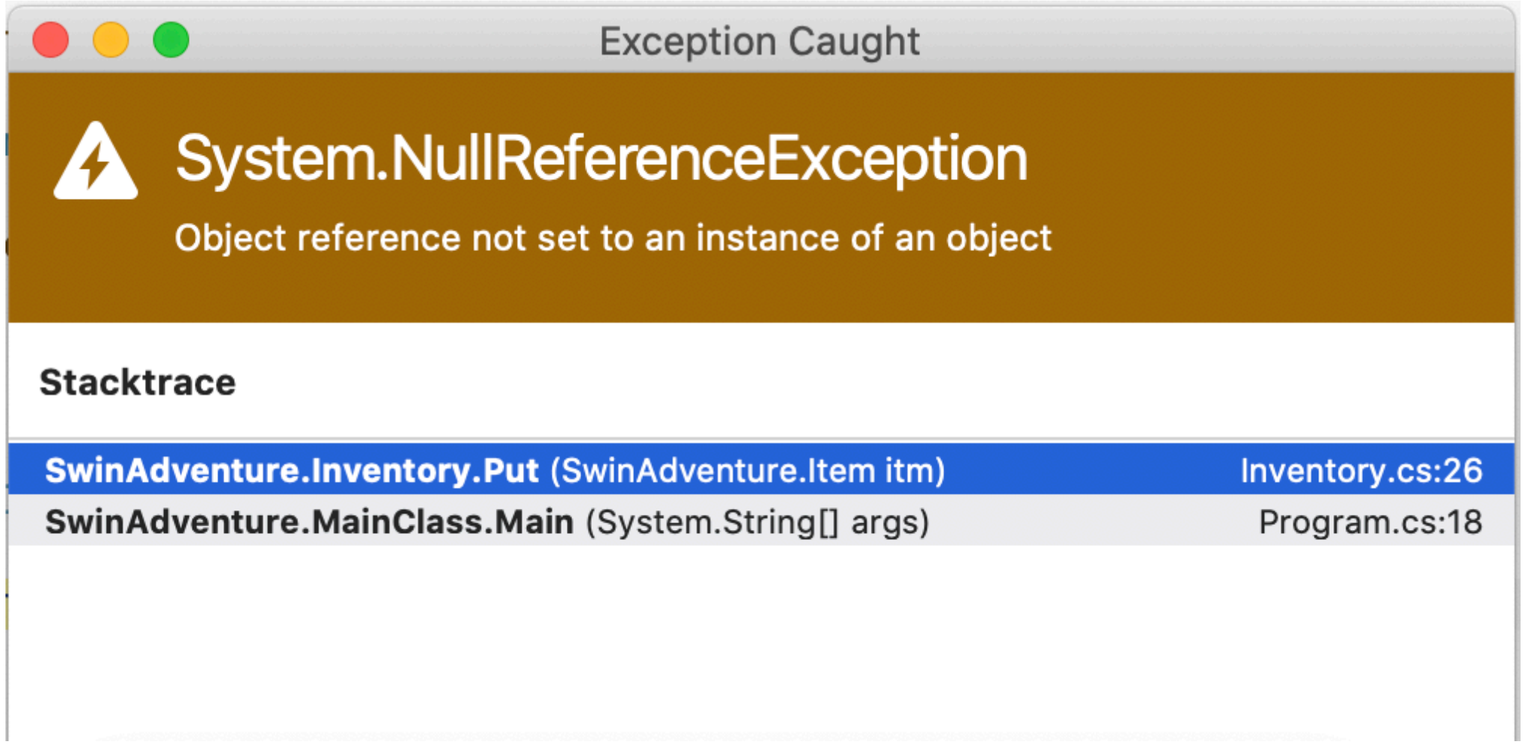
Awkward yeti comics

…but we need to learn how to listen

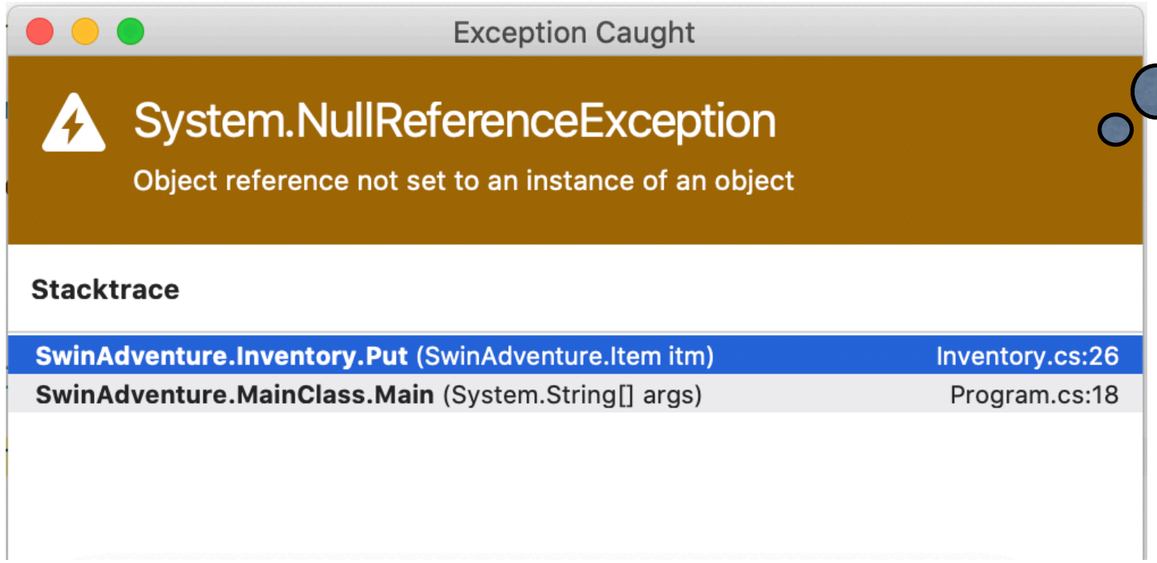


Awkward yeti comics

The computer will tell you where it **thinks** the error is happening



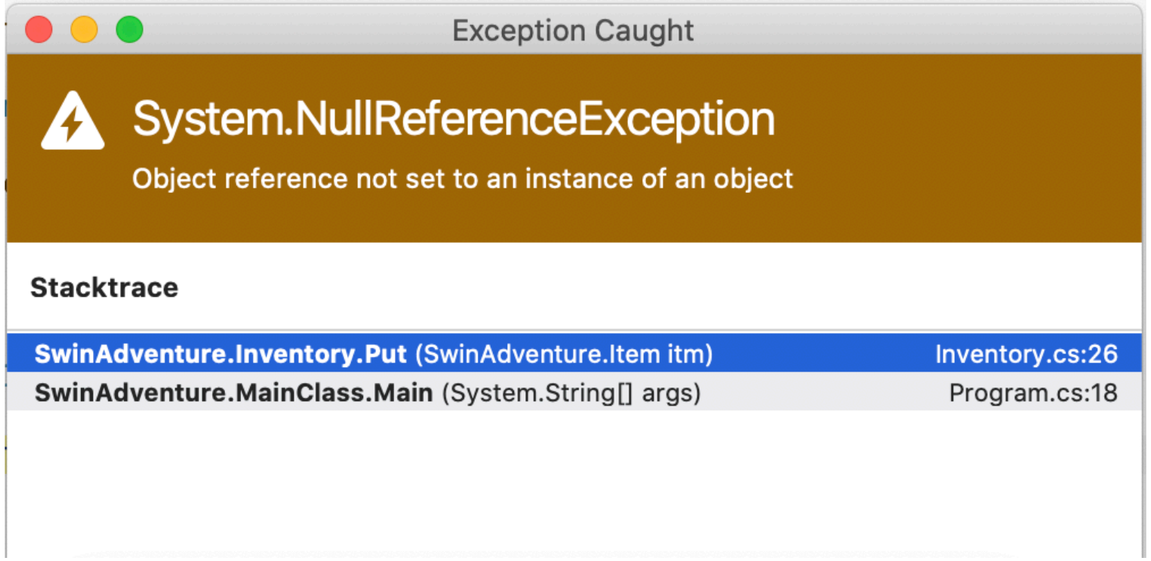
…but it's not always right



I think the problem is here…

```
24          public void Put(Item itm)
25          {
26              _items.Add(itm);
27          }
```

…but it’s not always right




```
6       public class Inventory
7       {
8           private List<Item> _items;
9
10          public Inventory()
11          {
12          }
13
```

It takes practice!

(and some debugging)

Documentation is your friend

Every good API will have documentation

Functions

Draw Circle

Overloaded

This function is overloaded. The following versions exist.

Draw Circle (*clr* : `Color` *c* : `Circle`)

Draw Circle (*clr* : `Color` *c* : `Circle` *opts* : `Drawing Options`)

Draw Circle (*clr* : `Color` *x* : `Double` *y* : `Double` *radius* : `Double`)

Draw Circle (*clr* : `Color` *x* : `Double` *y* : `Double` *radius* : `Double` *opts* : `Drawing Options`)

Every good API will have documentation

Draw Circle

Draw a circle onto the current window. The circle is centred on its x, y coordinates, and has the provided radius.

Parameters

Name	Type	Description
Clr	`Color`	The color of the circle
X	`Double`	The x location of the circle
Y	`Double`	The y location of the circle
Radius	`Double`	The radius of the circle

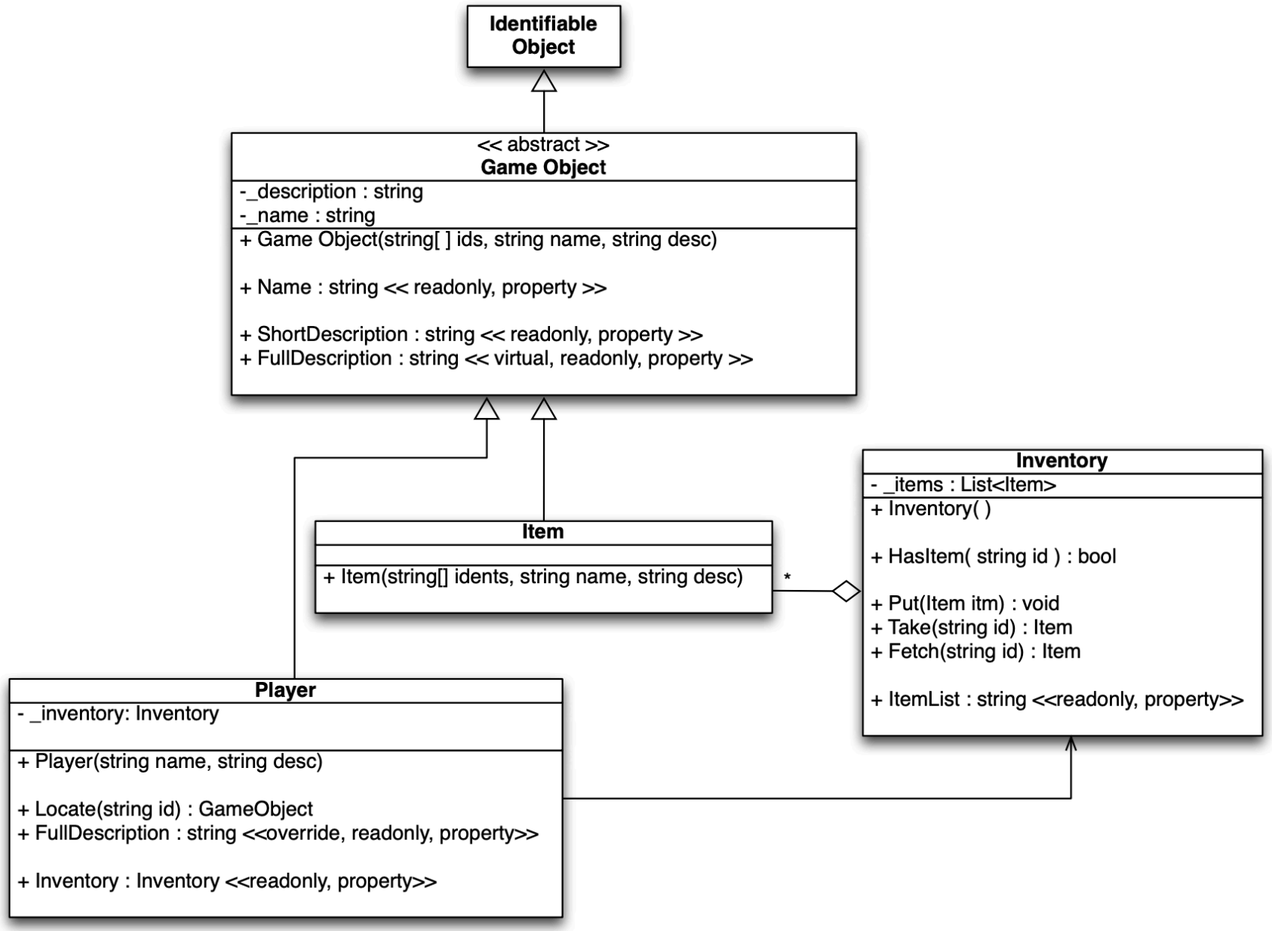
Signatures

C++ | **C#** | Pascal | Python

```
public static void SplashKit.DrawCircle(Color clr, double x, double y, double ra
```

Explore API documentation to find out what you can do!

When we give you a design, we have our reasons…



One last thing…

Google!

"Any developer who doesn't have a full tab bar of Google and Stack Overflow tabs isn't working"

“I google basic stuff all the time
Like, “****, what does String.format do again?”
googles "java string" and goes to the java docs”

Google!

“There are millions of developers out there, and it's very likely at least one of them has come across your problem before. Googling your issue will give you links to those discussions and solutions.”