Functional Programming

for Object Oriented Programmers

Jeremy Bellows
https://twitter.com/JeremyBellows

What is in this presentation

- What is Functional Programming?
- How to think as a Functional Programmer?
- Some basic F# syntax
- Solving a simple problem in F#!

Before we answer what Functional Programming is….

Programming is...

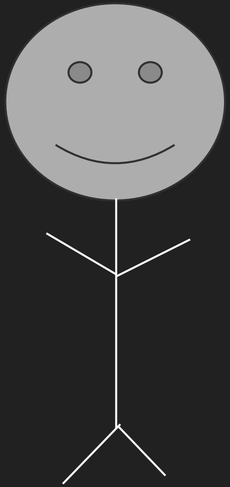
Part Logic
Part Grammar

What is Object Oriented Programming?

Objects

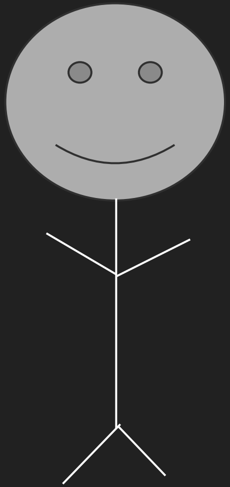
Objects are Nouns

Meet Jeremy…



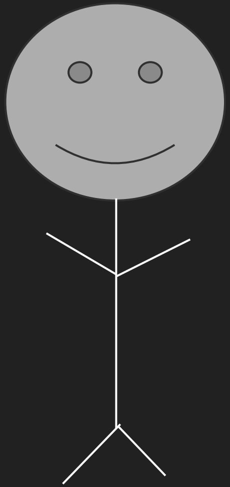
Meet Jeremy…

he is a person and has legs and can walk



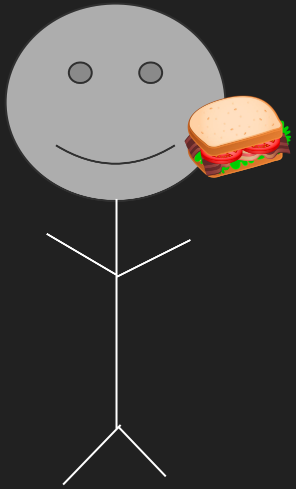
Meet Jeremy…

he is a person and has legs and can walk



he uses these legs to walk to the kitchen…

...and makes himself food



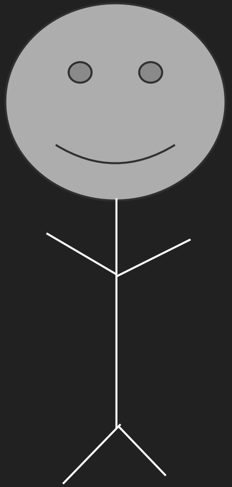
OOP Terms

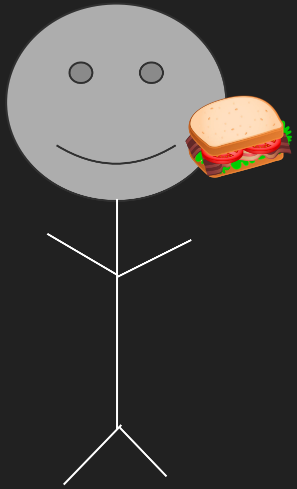
Meet Jeremy…
[noun]

he is a person and has legs and can walk
[noun] [noun]

he uses these legs to walk to the kitchen…
[noun] [noun]

...and makes himself food
[noun]





Meet Jeremy…

[object] he is a person and has legs and can walk

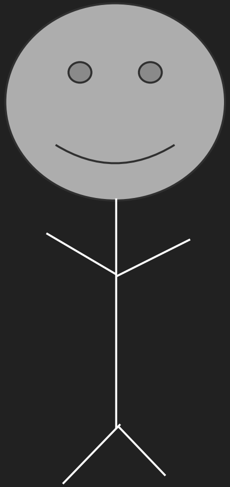
[interface] [objects]

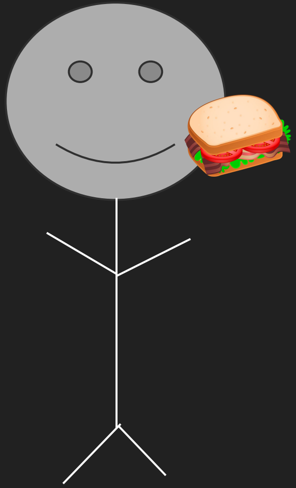
he uses these legs to walk to the kitchen…

[objects] [object]

...and makes himself food

[object]





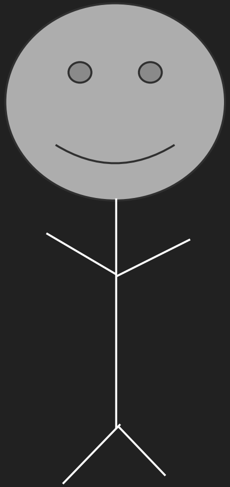
Functional Programming Terms

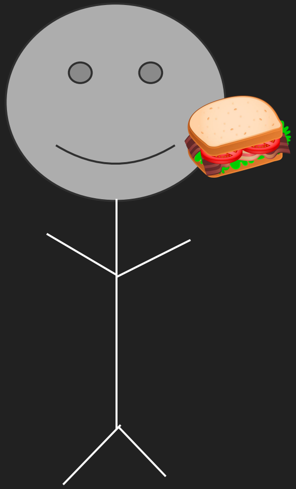
Meet Jeremy…

he is a person and has legs and can walk
[verb]

he uses these legs to walk to the kitchen…
[verb]

...and makes himself food
[verb]





To make a sandwich…

Start with bread...

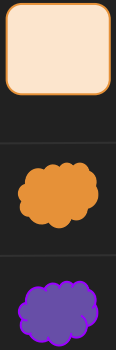
Start with bread...

Add peanut butter...

Start with bread...

Add peanut butter...

Add jelly...



Start with bread...

Add peanut butter...

Add jelly...

Add bread...

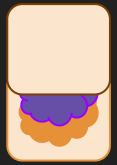
Start with bread...

Add peanut butter...

Add jelly...

Add bread...

Sandwich!!!



Function Signatures

Bread -> Peanut Butter -> Jelly -> Bread -> Sandwich

Object Oriented Programming

- Describes what something is
  - A sandwich is multiple ingredients between two pieces of bread

Functional Programming

- Describes how something works
  - Add bread, ingredients, bread, get sandwich!

Both Paradigms Need Nouns and Verbs

Types and FSharp

F# Types

The Nouns

- Abbreviations/Aliases
- Records
- Classes
- Interfaces
- Tuples
- Discriminated Unions
- Enum
- Struct

Immutable State

Program

Program

Function A

- Variable X

Program

Function A

Function B

- Variable Y

Program

Function A

Function B

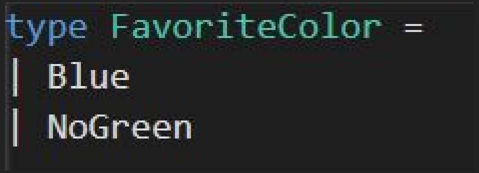
- Variable Y
- Variable X

Program Starts -> Function A -> Function B -> Something

Discriminated Unions

What is your favorite color?

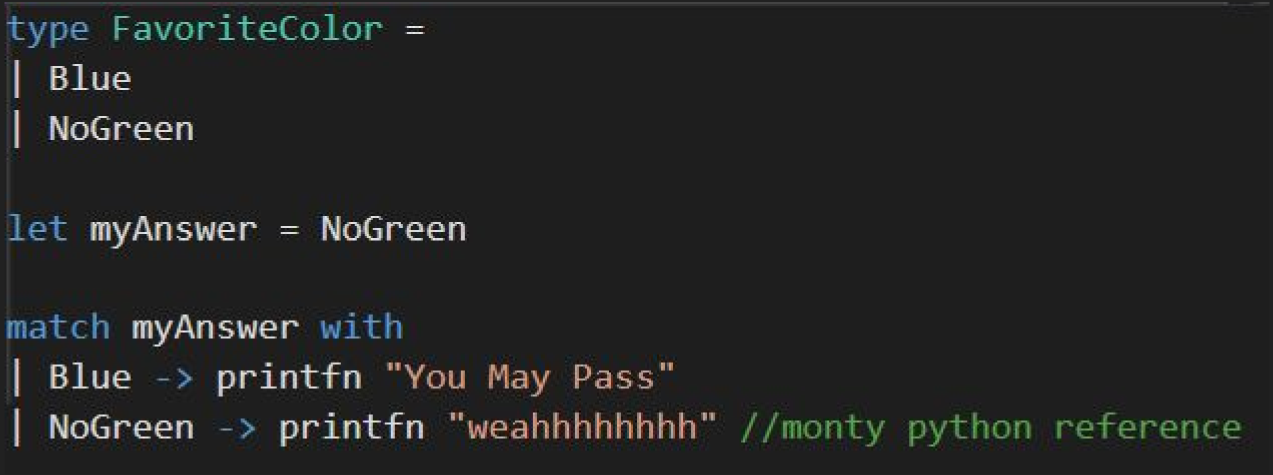
Discriminated Unions



```
type FavoriteColor =
| Blue
| NoGreen
```

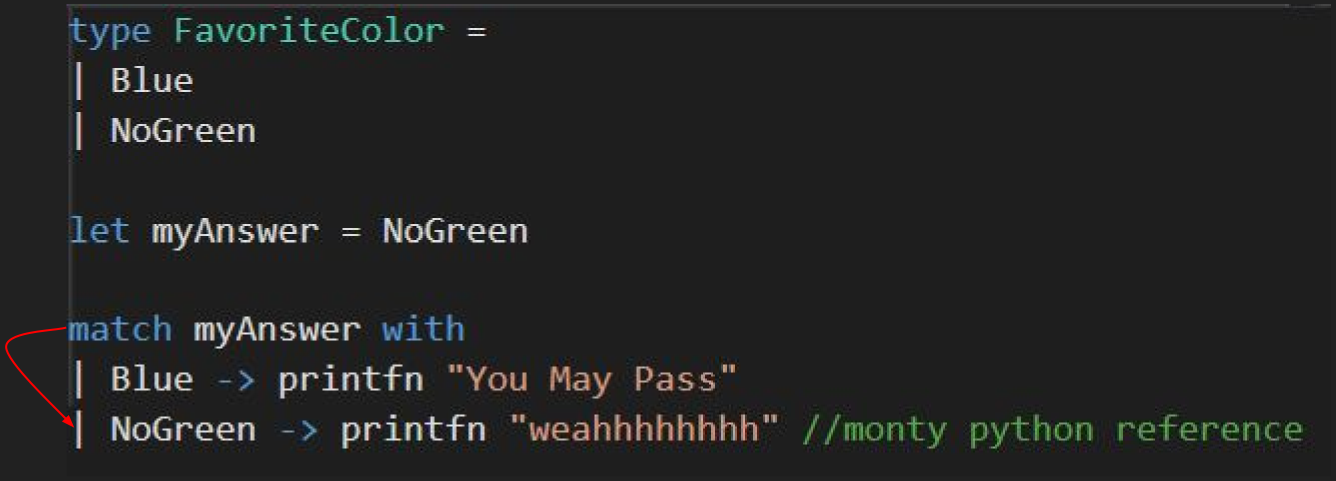
Pattern Matching

Discriminated Unions

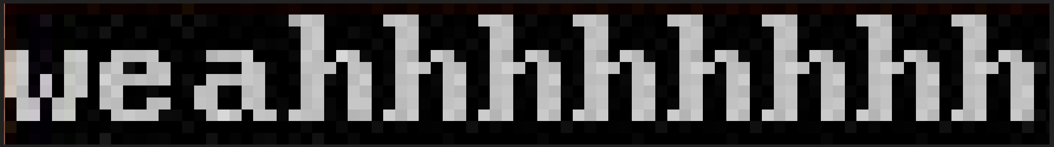


```
type FavoriteColor =
| Blue
| NoGreen

let myAnswer = NoGreen

match myAnswer with
| Blue -> printfn "You May Pass"
| NoGreen -> printfn "weahhhhhhhh" //monty python reference
```

Discriminated Unions



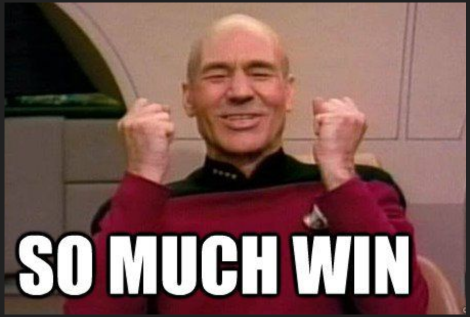
```
type FavoriteColor =
| Blue
| NoGreen

let myAnswer = NoGreen

match myAnswer with
| Blue -> printfn "You May Pass"
| NoGreen -> printfn "weahhhhhhhh" //monty python reference
```



```
weahhhhhhhh
```

The Option Type

Free from Nulls!



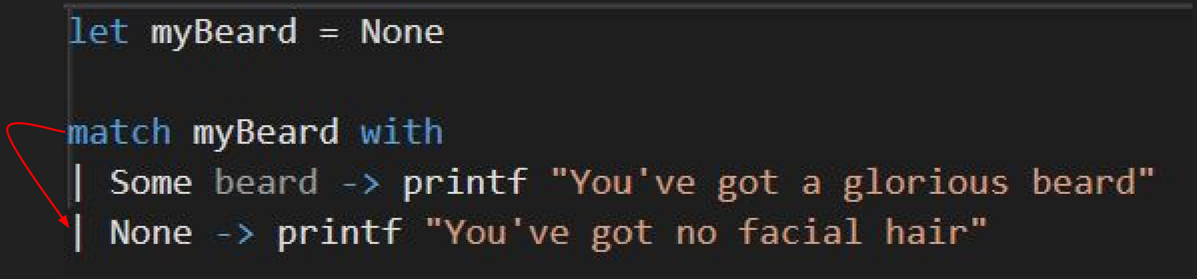
Some

- I’ve got a lot of something

None

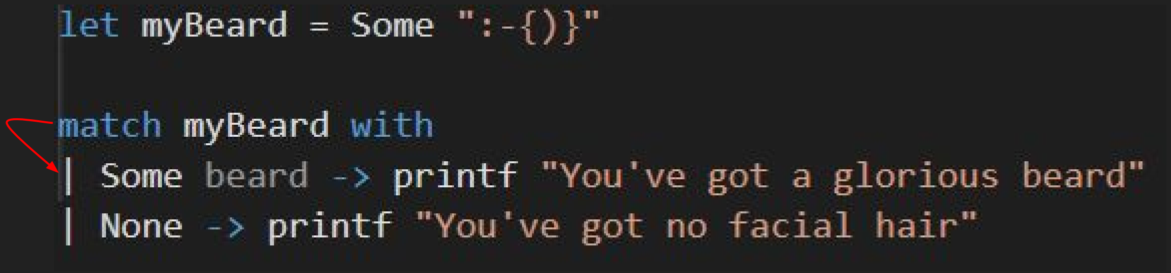
- I’ve got a lot of nothing

Option type

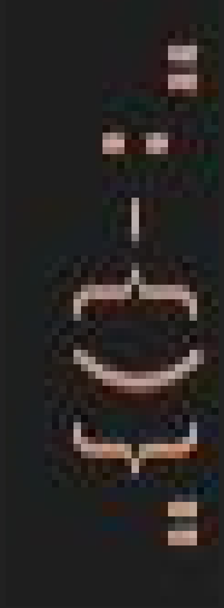


```
let myBeard = None

match myBeard with
| Some beard -> printf "You've got a glorious beard"
| None -> printf "You've got no facial hair"
```

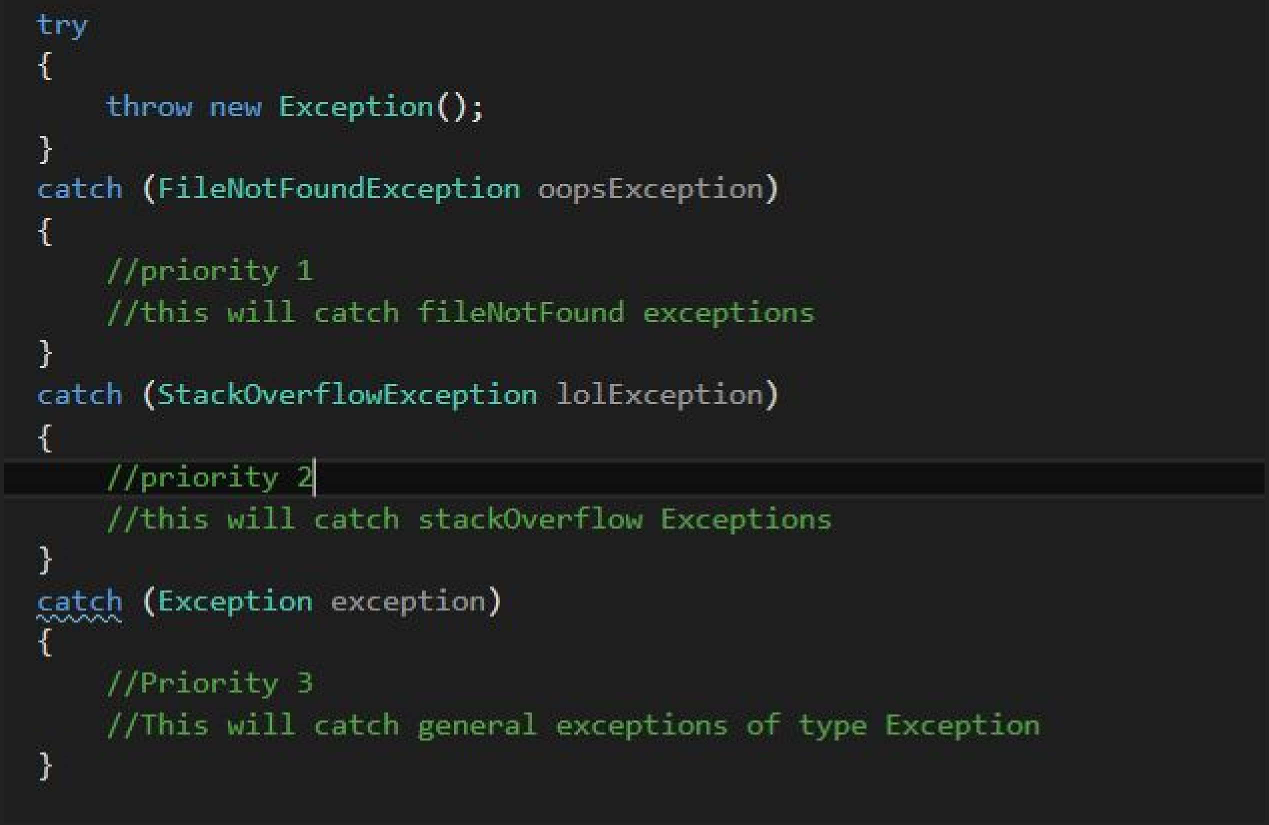
Option type



```
let myBeard = Some ":-{)}"

match myBeard with
| Some beard -> printf "You've got a glorious beard"
| None -> printf "You've got no facial hair"
```

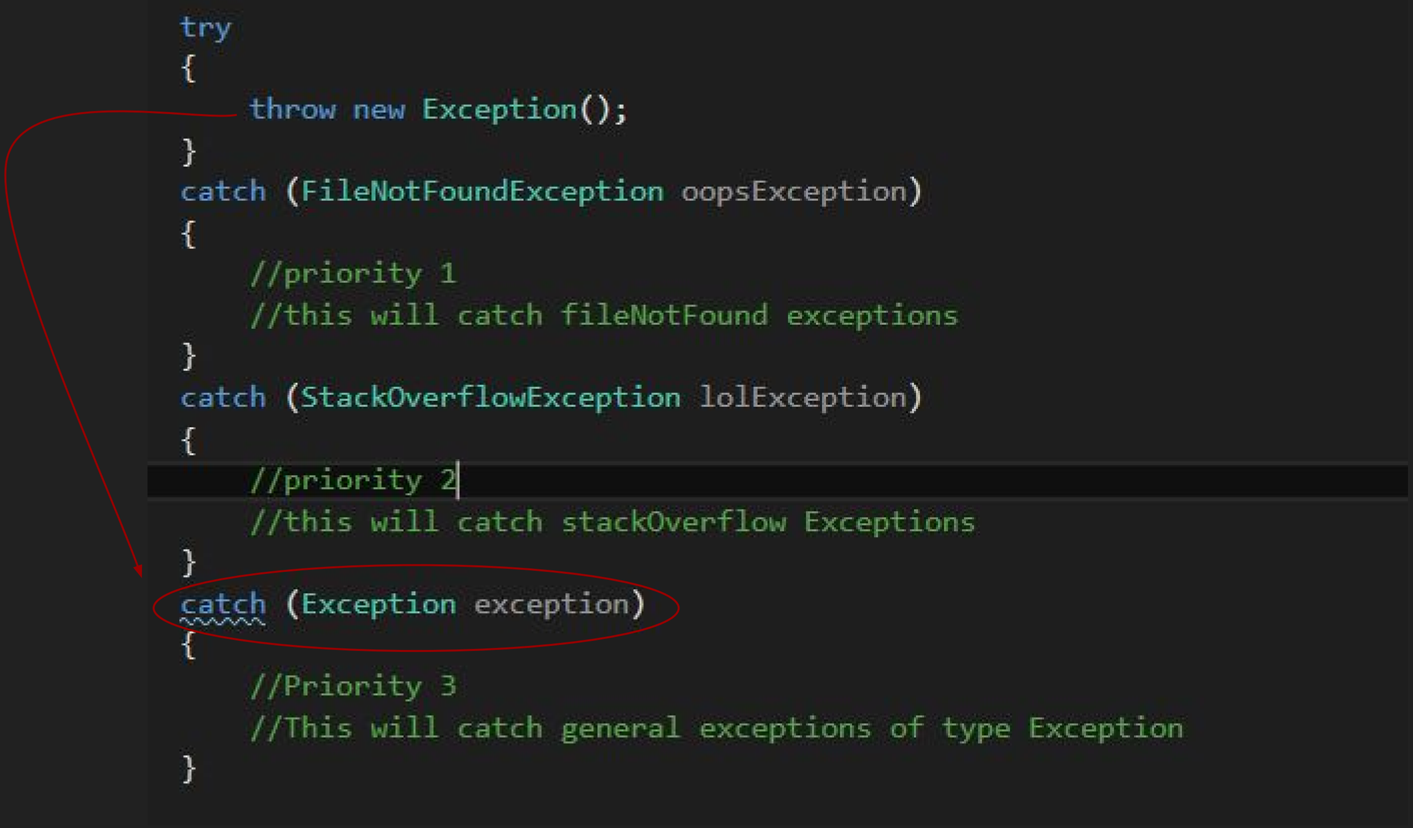


Try Catch C#



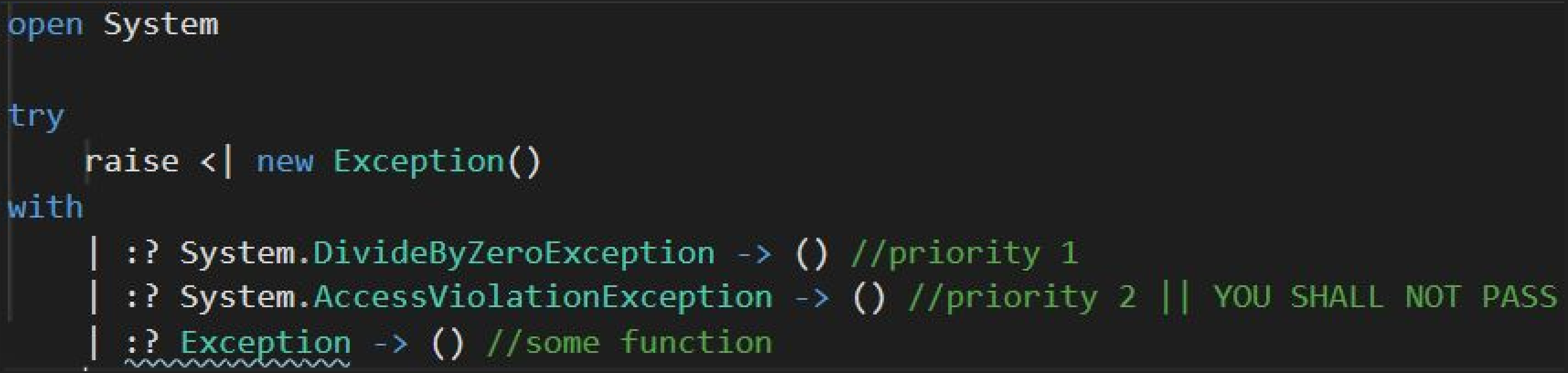
```
try
{
    throw new Exception();
}
catch (FileNotFoundException oopsException)
{
    //priority 1
    //this will catch fileNotFound exceptions
}
catch (StackOverflowException lolException)
{
    //priority 2
    //this will catch stackOverflow Exceptions
}
catch (Exception exception)
{
    //Priority 3
    //This will catch general exceptions of type Exception
}
```

Try Catch C#



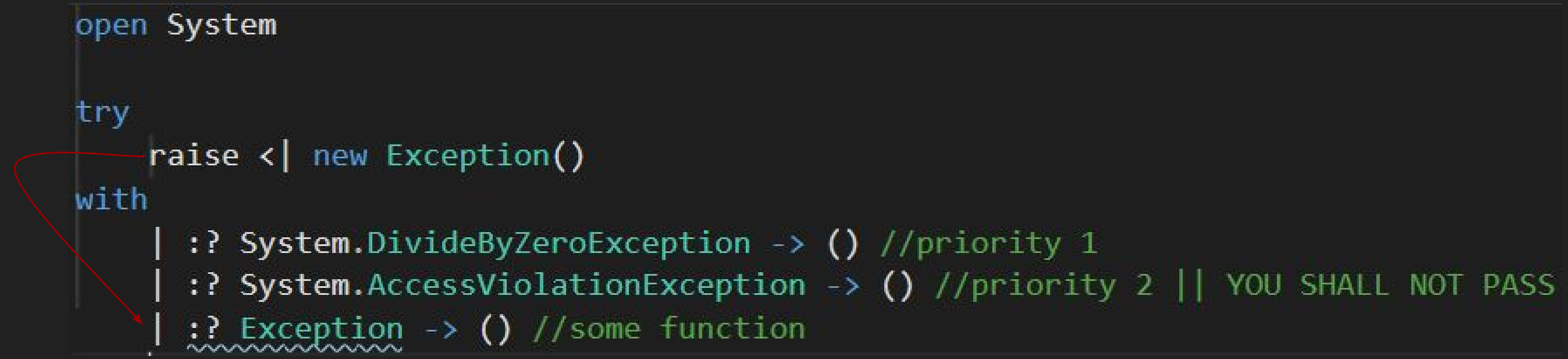
```
try
{
    throw new Exception();
}
catch (FileNotFoundException oopsException)
{
    //priority 1
    //this will catch fileNotFound exceptions
}
catch (StackOverflowException lolException)
{
    //priority 2
    //this will catch stackOverflow Exceptions
}
catch (Exception exception)
{
    //Priority 3
    //This will catch general exceptions of type Exception
}
```

Try Catch F#



```
open System

try
    raise <| new Exception()
with
    | :? System.DivideByZeroException -> () //priority 1
    | :? System.AccessViolationException -> () //priority 2 || YOU SHALL NOT PASS
    | :? Exception -> () //some function
```

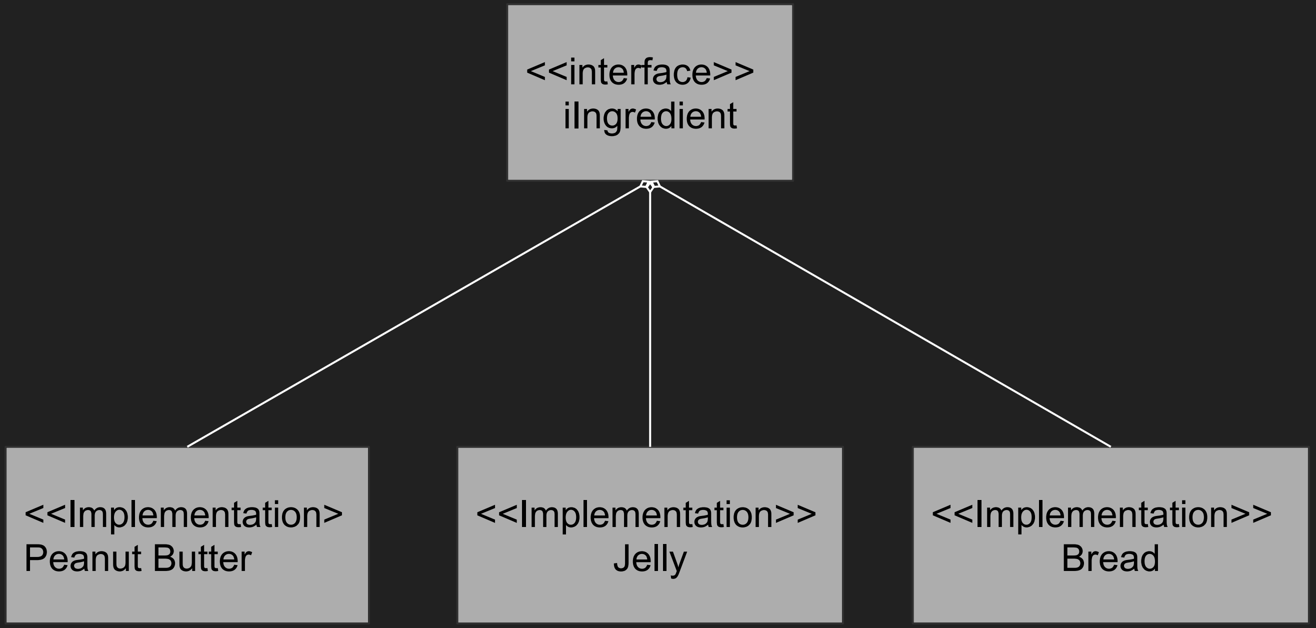
Try Catch F#



```
open System

try
    raise <| new Exception()
with
    | :? System.DivideByZeroException -> () //priority 1
    | :? System.AccessViolationException -> () //priority 2 || YOU SHALL NOT PASS
    | :? Exception -> () //some function
```

Let’s talk Sandwiches

Object Oriented Programming

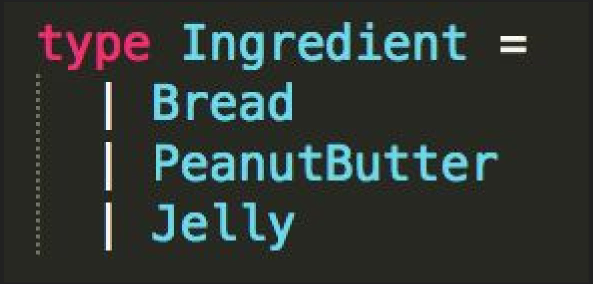


Object Oriented Programming

SandwhichMaker : Class

```
 -   Ingredients : List<iIngredients>
 +   void AddIngredient(iIngredient ingredient)
 +   Sandwhich FinishSandwhich()
```

Functional Programming

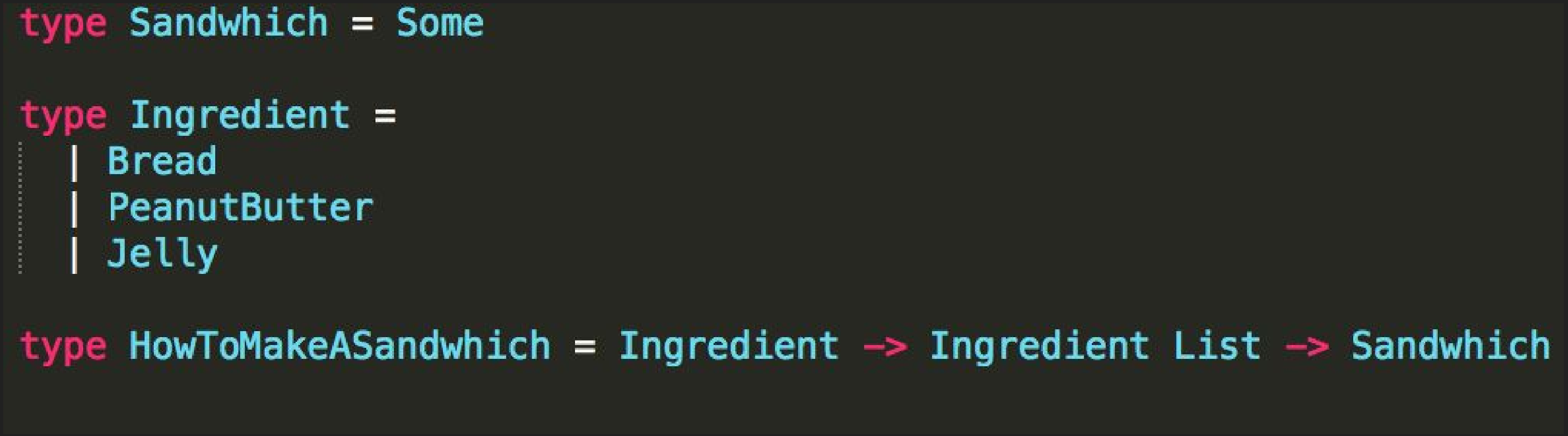


```
type Ingredient =
    | Bread
    | PeanutButter
    | Jelly
```

Ingredient List -> Sandwich

Bread -> Ingredient List -> Sandwich

Functional Programming



```
type Sandwhich = Some

type Ingredient =
    | Bread
    | PeanutButter
    | Jelly

type HowToMakeASandwhich = Ingredient -> Ingredient List -> Sandwhich
```

Let’s Solve a Problem!

Before we start...

There is no ‘void’ type

only the ‘unit’ type

I only need even numbers!

I need a program that

- Filters the list to only contain even numbers
- prints the even numbers

List:

[994 551 386 79 850 155 466 953 903 17 930 344 805 898 744]

Let’s Solve another Problem!

I want data for my excel sheet!

For some reason, Deli’s love using excel sheets! They want us to consume json sent by their undocumented api and convert the data into a csv file.

- Consumes json from a mocked endpoint
- Prints all fields of each item in a csv format

jsonData -> parser -> writeCsv -> unit

Things to Google Later

- FSharp
- FSharp types
- FSharp for Object Oriented Programmers
- FSharp Pattern Matching
- FSharp Lists
- FSharp Option
- FSharp null
- Where is the nearest sandwhich shop?

Further Reading

- http://fsharp.org/
- http://fsharpforfunandprofit.com/
- http://fsharpforfunandprofit.com/posts/why-use-fsharp-intro/

What we learned today

- Objects are nouns
- Functions are verbs
- Functional Programming is the art of thinking in verbs
- Function signatures are the composition of verbs
- Functional programming is another way to perceive the world!

Thank you!

Slides and Code can be found at

https://github.com/JeremyBellows/DallasTechFestFpForOOP