Javascript Fundamentals

Both FUN and drives you MENTAL

Javascript

- **JS** for short
- Javascript is to java as hamster is to ham
  - There is no correlation whatsoever
- Made in a little over a week by Brendan Eich
  - Twitter handle: @BrendanEich (he replies!)
- **Functional** language (does have some OO support)
  - The OO support is more object based and not object oriented though
- **Loosely typed** (but that does not mean it doesn't have data types)
- **Interpreted** not *compiled*

ECMASCRIPT

- Standardization of JS
- Just like how HTML has HTML5, JS has ES6 the latest standardized version

SUNBONS (AKA the data types in js)

- **Symbols**
- **Undefined**
- **Null**
- **Boolean**
- **Objects**
  - Functions are also a data type but they’re treated as objects in js
  - Confusing i know
- **Numbers**
- **Strings**

Scopes in js

- The scope of a variable describes where it exists
- 3 scopes:
  - Function
  - Global
  - Block
- Somewhat related to variable declaration
  - When you declare a variable using var because of hoisting it ends up having a global scope
  - When you declare a variable using let, it limits the scope of a variable to the block it is declared in
  - Thus, *make let not var*

Other types of variable declarations

- Using const when defining a variable prevents the value of the variable from being changed
  - Const is short for constant so obviously
- To declare an object you use the object literal syntax
  - Put stuff in {}
  - Note that if you put stuff in [] you’re making an array, but if you do {}, that’s an object

Truthy and Falsy

- In js, all values have a boolean equivalent, i.e, you can evaluate actual values instead of boolean expressions
  - Wild right?
  - Before your brain starts melting, this is actually pretty common in other languages, although, in other languages you can mostly substitute numeric values in place of boolean ones
- Falsy values: FUN0NE
  - Values that evaluate to false
  - False, undefined, null, 0 (zero, both positive and negative), NaN (is in fact a number), empty string
- Truthy values: literally everything else
  - Values that evaluate to true

Functions in JS

- Basic function
  - Y’all know what this is
- Callback function
  - Function within a function (functionception)
- Arrow function
  - Anonymous function
  - Lambda expression basically
- IIFE
  - Pronounced iffy
  - Stands for immediately invoked function expression

Encapsulation in JS

- Since JS is not OO, it has no access modifiers
- You can still achieve encapsulation using closure!
- What is closure?
  - It’s when you finally get over that function that’s been making you cry (just kidding)
  - Basically, when you have a function within a function, if you declare a variable in the outer function, only the inner function can access it
  - `outer(){ let private = 0; inner(){ does something with private}; };`

Inheritance in JS

- Before ES6, classes weren't a thing in js, but there was such a thing as prototypal inheritance
  - How prototypal inheritance works is based on property access
    - Whenever you try to access a property on an object, if it's not found, then the JS engine will search for it on the prototype and then on the prototype's prototype and so on
    - Only applies for access and not assignment
- ES5 had constructors
- By ES6, you are able to create classes, and method syntax
  - Use the extends keyword to implement inheritance in JS

Discussion

- Difference between null and undefined?
- == vs ===?
- What do you think is the difference if you use closure with a basic function as opposed to when you use it with an IIFE?