Chapter 1

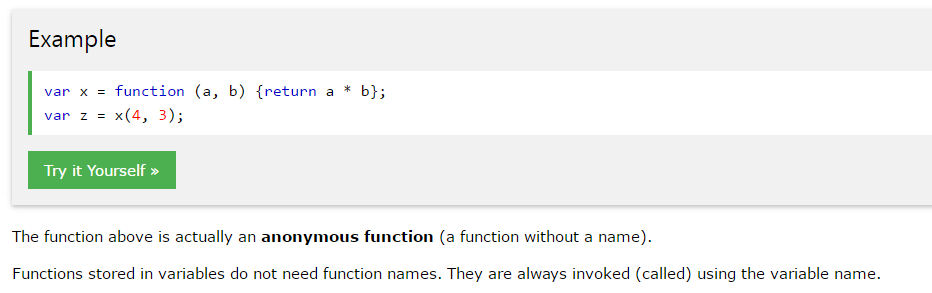
Javascript functions are first class objects

* + - It means that functions are **objects**, with a type and a behaviour. They can be dynamically built, passed around as any other **object**, and the fact that they can be called is part of their interface. It means that function actually inherits from **Object**

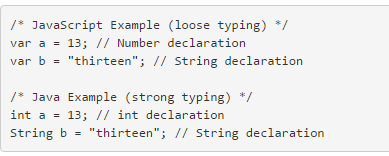
Javascript is the first Lambda

* + - In the context of JavaScript it usually refers to an anonymous function. That is a function that doesn't get named, but is usually used as a value passed to another function in order pass a behavior as a value.

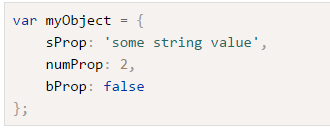
Anonymous function

* + - 

Loosely typed

* + - Loose typing means that variables are declared without a type.
    - 

Object literal

* + - a comma-separated list of name-value pairs wrapped in curly braces. Object literals encapsulate data
    - 

Prototypal inheritance

* + - all functions are also objects, which means that they can have properties. And as it so happens, they all have a property called `prototype`, which is also an object. Any time you create a function, it will *automatically* have a property called **prototype**, which will be initialized to an empty object.

Top-level variables

Namespace

* + - **Namespaces** are named program regions used to limit the scope of variables inside the program.

Chapter 2

Numbers

* + - Has single number type it is 64 bit floating point like java’s double.
    - NaN is a number value result from operation that cannot produce a normal result. It is not equal to anything including himself… use isNaN(number)

Strings

* + - Strings are inmutable

Chapter 4

Methods

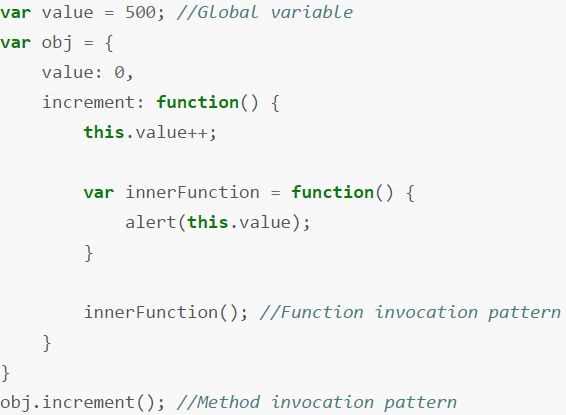
* + - JavaScript methods are the actions that can be performed on objects.
    - A method is a piece of code that is called by name that is associated with an object
    - 

Closure

* + - an inner function that has access to the outer (enclosing) function's variables

Function Invocation

* + - this is set to the global object.



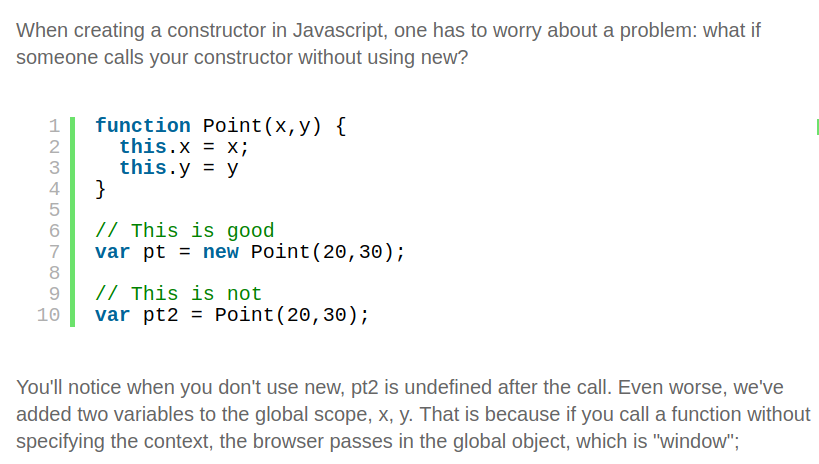
The result of obj.increment() will be *500* and not 1.

Class

Data structure or type in which you can define variables and methods

Constructor Invocation Pattern

When we dont use the new keyword this happens

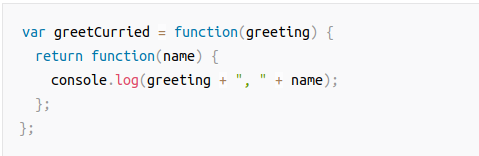


**Cascading**

refers to repeatedly calling one method after another on an object, in one continuous line of code.

Currying

What this means is that you can pass all of the arguments a function is expecting and get the result, or pass a subset of those arguments and get a function back that’s waiting for the rest of the arguments. It really is that simple.







Memoization