Intro To Javascript

Coder Academy

Day 1

- History of Javascript - From Browser to server
- What is node?
- Javascript vs Ruby
- JS in depth
  - Syntax Parser
  - Execution Context
  - Lexical Environment
  - Outer Environment
  - Hoisting
  - Scope Chain
  - this, null and undefined

History of Javascript

- JavaScript was created in 1995 by Brendan Eich while he was an engineer at Netscape
- It was originally going to be called LiveScript, but renamed based on popularity of Sun MicroSystem's Java
- Netscape submitted JavaScript to Ecma International, hence the name ECMAScript, 6th major editions was published in 2015 - JS version ES6
- ES6 vs ES5
- JS is not limited to browser anymore - frontend, backend, desktop apps, mobile apps, IOT
- JS has no concept of input or output - runs as a scripting language in the host environment and it is up to host environment to provide ways to communicate with outside world
- Host environment includes - browser, server side environment such Node.js
- Eg : prompt in browser = process.stdin.read() in server!

https://github.com/NandiniNayak/javascript-LessonPlan/blob/master/lesson1-terminal-gets-chomp.js

NCSA Mosaic
Graphical browser

Netscape Navigator
1994 - Mocha/LiveScript

Microsoft Internet explorer reused components of LiveScript/javascript and called it JScript

Netscape navigator collaborated with Java Developed script version of java in browser
1995

To overcome browser incompatibility, Netscape submitted javascript for browser standard ECMA- hence called ECMA script

What is Javascript?

- JavaScript is a multi-paradigm, dynamic language, with types and operators standard built in objects and methods.

- JavaScript supports both object oriented programming and functional programming.

- In Js functions are objects, hence allowing functions to be passed around like any other objects, hence making functions first class citizens.

What is Node?

- Node is a server side javascript
- It has two sections:

  1. Acts as a backend similar to irb : Js code can be run in the terminal

     - node can also use the global objects in terminal, in the browser global object is window, in terminal if you jump into node and type global, you can access various parameters related to your computer
     - >process.env.USER // prints out your computer name

  2. Has inbuilt function to start server

```
var http = require("http");

http.createServer(function(request, response){
  response.end('Hello World \n');
}).listen(8000);

console.log(`server running at http://127.0.0.1:8000/ or localhost:8000:
${process.env.USER} `);
```

Js primitive types

Number

String

Boolean

Symbol

Object - Function, Array

Null - Lack of existence, set by the developer

Undefined - Lack of existence set by the browser

(https://github.com/NandiniNayak/javascript-LessonPlan/blob/master/lesson2-undefined.js)

Javascript Style Guide

- https://github.com/airbnb/javascript
- https://www.w3schools.com/jS/js_conventions.asp
- https://www.w3schools.com/jS/js_best_practices.asp

- **Thumb Rule**:
  - Use **c**amel**C**ase while naming variables, objects functions and instances
  - Use **P**ascal**C**ase while naming classes.
  - Do not use trailing or leading underscores

Type Coercion

while using operators: == operator coerces one of the operators if not same datatype

Hence use === which is strict equality and !== strict inequality

```
> Number(true)
<- 1
> Number(false)
<- 0
> Number(undefined)
<- NaN
> Number(null)
<- 0
> String(1)
<- "1"
> String(true)
<- "true"
> String(undefined)
<- "undefined"
> "3" == 3
<- true
> "3" === 3
<- false
>
```

```
> false == 0
<- true
> true == 1
<- true
> '1' == 1
<- true
> null == 0
<- false
> '1' === 1
<- false
> null < 1
<- true
> "" == 0
<- true
> " " == 0
<- true
> "" === 0
<- false
> null === 0
<- false
>
```

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Operator_Precedence

Js - Objects

- Js Objects can be defined like a hash in ruby -> called **object literal** which is a key value pair

  let user = {name: “nandini” , age: 10};

- Object can be declared using **object constructor**

  let user = new Object();

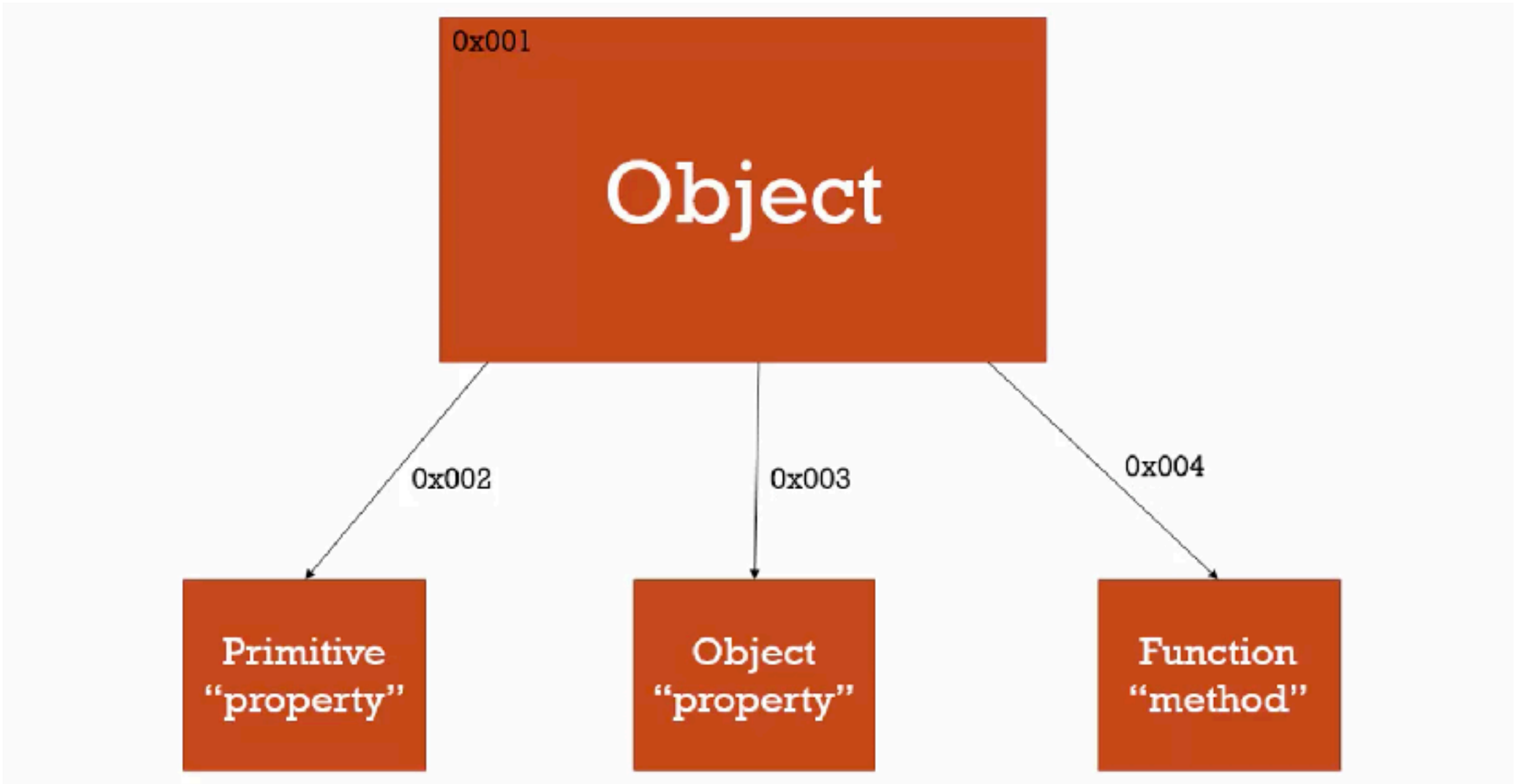
- However airbnb style guide recommends usage of object literal syntax over object constructor syntax.

- Square bracket and dot (.) notation
  - user.hasName = true;
  - User.has name = true; // throws an error, as symbols cannot have spaces hence cannot use dot notation
  - user[“has name”] = true; // works

- Object has several method constructors: click here

Js - Objects

Object can have its value to be a primitive property, another object(key value pair) or even a function



Object Allow NameSpacing

```
1  var greet = 'Hello!';
2  var greet = 'Hola!';
3
4  console.log(greet);
5
6  var english = {};
7  var spanish = {};
8
9  english.greet = 'Hello!';
10 spanish.greet = 'Hola!';
```

Execution context