Date: 28-5-2020

Morning Session: 9 am – 11 PM

By ~ Sundeep Charan Ramkumar Today

Topics: **Introduction to JavaScript**

1. Introduction
2. JS datatypes
3. JS operators

THE 90'S PERIOD

- Back in the period of 1995, NetScape Communications was a predominant identity on web, and gave a tough competition to Mosaic browser.
- They felt that static websites won't be the future of web and were missing a key component. Interaction. Marc Andreesen (founder of Netscape Comms) was the reason for this vision.
- Web required a small script language that could manipulate the elements on a certain basis.
- The idea was to have a simple scripting language, which was easy enough for designers to understand. Then came the dawn of MOCHA.

- However, it lacked the expected feature set and performance. Brenden Eich (Father of JavaScript) was brought to play to develop a language like Scheme, but for browsers.
- During early and mid 90's Java was ruling the software industry. Hence, there was a lot of pressure put upon the team to come up with a working prototype of Mocha.
- Sun Microsystems (Now Oracle) were about to close the deal with NetScape Comms to make Java as the official language for the web.
- Turns out, Java was only reserved to be a corporate friendly lanugage. It wasn't easily accessible and learnable across scripters, designers.
- Many candidates were also involved in the making of browser friendly scripting language.

IMPLEMENTATION OF JAVASCRIPT

- On December 1995, Netscape launched the working prototype of Mocha, which soon renamed as LiveScript.
- But Netscape wasn't the only browser available at that time. Microsoft's Internet explorer was also in the making and they wanted a separate implementation of JavaScript, hence the dawn of JScript.
- NetScape also came with an implementation called SpiderMonkey, which is still the browser engine for Firefox. JScript and Spidermonkey were the monopolies for several years and paved the way o how the web environment should behave in interaction wise.
- Many candidates were also involved in the making of browser friendly scripting language.

- On November 1996, Netscape launched the working prototype of Mocha, which soon renamed as LiveScript.
- But Netscape wasn't the only browser available at that time. Microsoft's Internet explorer was also in the making and they wanted a separate implementation of JavaScript, hence the dawn of JScript.
- NetScape also came with an implementation called SpiderMonkey, which is still the browser engine for Firefox. JScript and Spidermonkey were the monopolies for several years and paved the way o how the web environment should behave in interaction wise.
- The major marketing feature that was pushed amongst scripters was its Java like syntax.

```java
public class AddTwoNumbers {
    public static void main(String[] args) {
        int num1 = 5, num2 = 15, sum;
        sum = num1 + num2;
        System.out.println("Sum of these numbers: "+sum);
    }
}
```

A simple Java program

```javascript
function addTwoNumbers (a, b) {
  return a + b
}

addTwoNumbers(1, 4)
```

A simple JavaScript program

RELEASES OF JAVASCRIPT

- Netscape Navigator (Grandfather of Firefox) 2.0 had the very first JavaScript. The main drawback was that most of the features weren't working. Even if they were available in the design, they weren't implemented.
- In 1996, Netscape Navigator had released its v3, and it received a huge reception amongst designers, especially scripters. The error information was more prominent and clear
- Even there were errors present and rooms for improvement were clearly needed, the implementation served perfect during that time.

DAWN OF ECMASCRIPT

- The perfect way to remove errors from an implementation is to bring up some standards that define the rulesets to make sure the script is working at its optimal conditions.
- ECMA is an industry association which was formed in 1961 which was concerned with standardisation of information, and they started the standardisation for JavaScript at 1996.
- Due to trademark reasons, JavaScript was not accepted by the team and had to go another renaming scheme called ECMAScript, although we call it JavaScript as its the commercial name.
- The first version of ECMAScript came live during 1997 along with Netscape 4. It removed all of the errors concerning the prototypes and object creations.

- The improvements didn't stop there as JavaScript still missed many critical features such as JSON conversion, Regular expressions. ECMAScript version 2 was the way to go for it at 1998.
- Following ES2, were only minor updates till the launch of ES4. The ability to connect websites was a reality after AJAX (Async JavaScript and XML) was born. Netscape 4 and Internet Explorer 5 were accepting it.
- ECMAScript6 during 2015 was the most respected and awaited upgrade for JavaScript. It literally supercharged the implementation with tons of new features. However, many browsers (still now) are struggling to keep it running.
- Transpilers like Babel and other tools were into play to make sure they turn back to older versions of the code. The current version is ES10 with some minor changes and squashing bugs in the implementation if any.

FUTURE OF WEB

- Right now, there are communities which are trying to run native Assembly language on to the web.
- Even if the implementation right now isn't even at beta stages, it opens up a massive optimisation strategy.
- Assembly language is something that runs close to the processor, thereby compiling faster and operating web applications at a faster rate.
- Frameworks like WebAssembly, Asm.js are being responsible in development of Assembly language compilation onto the web.

How To create a js file?

Create a file with .js Extension

How to link js file?

index.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>

<script src="app.js"></script>

</body>
</html>

```

Comments in js?

One-line comments start with two forward slash characters `//`.

Multiline comments start with a forward slash and an asterisk `/*` and end with an asterisk and a forward slash `*/`.

How to Print Something is JS?

Output data: there are two ways 1) developer based output, 2) user based output
User based outputs are: 1) document.write() 2) alert ()
Developer based output : Developer tools allow us to see errors, run commands, examine variables, and much more.

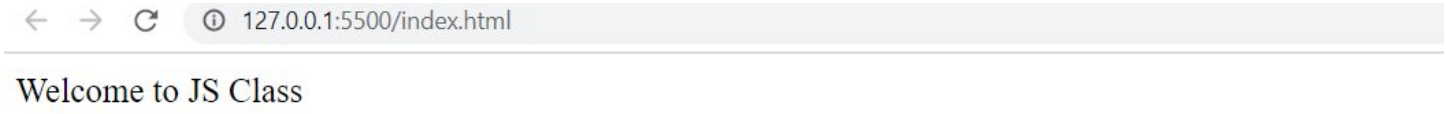
Console.log()

app.js

```js
// output data:

// Developer based output: Console.log()

console.log("attaninu");

```

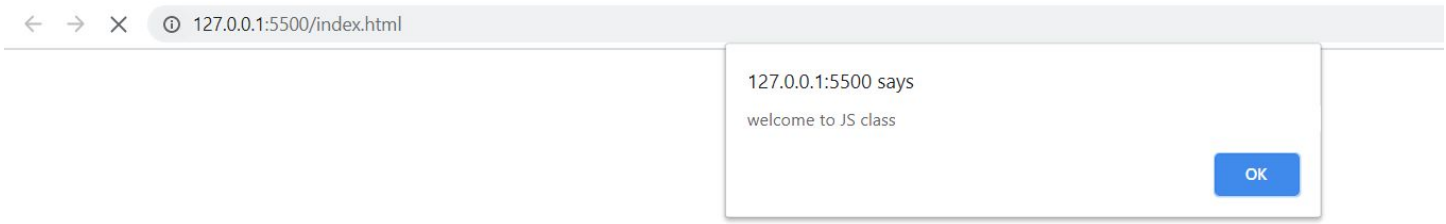
Document.write():

```
document.write("Welcome to JS Class");
```



Alert(): It shows a message and waits for the user to press “OK”. The mini-window with the message is called a modal window. The word “modal” means that the visitor can’t interact with the rest of the page, press other buttons, etc, until they have dealt with the window. In this case – until they press “OK”.

```
alert("welcome to JS class");
```



Variable: A variable is a “named storage” for data. We can use variables to store goodies, visitors, and other data. to create a variable in JavaScript, use the **var** keyword.
There are two different stages of creating variable

1) Declaration (to create)
2) Initialization (to store)

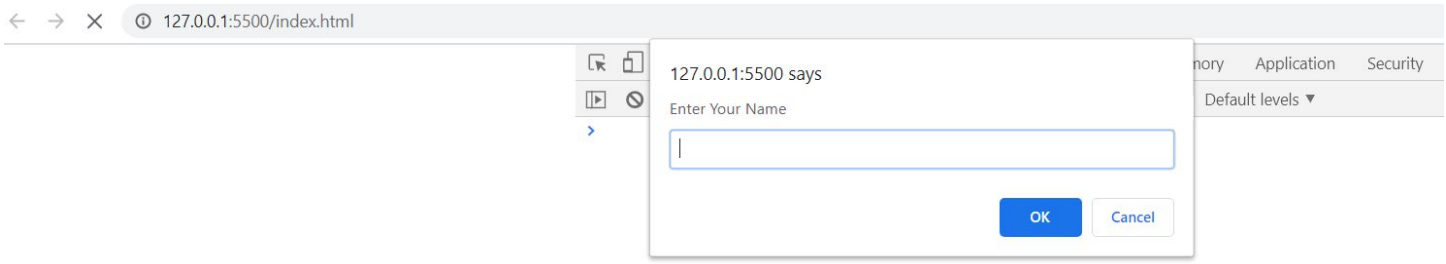
```
var fristName;
var fristName = " yodraj dendukuri"
document.write(fristName);

var name = "yodraj dendukuri" //declaration and initialization in single line
document.write (name);
```

Data types: set of standards which will make sure that those standard data can be accepted.

1) String: A string must be surrounded by quotes. (“yodraj” or ‘yodraj’)
2) Number:The *number* type represents both **integer & floating point numbers**.(1, 1.254,-2, -1.5)
3) Boolean : The boolean type has only two values: **true** and **false**.
4) **Null : Empty (no memory specified)**
5) **Undefined : Empty (memory specified)**

Receiving input from the user:

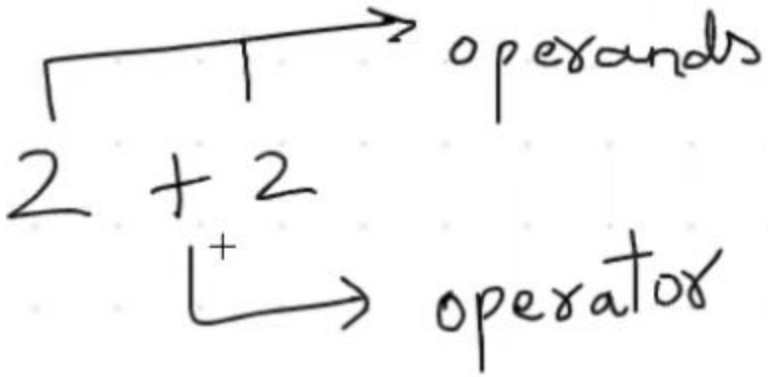
```
var name = prompt ("Enter Your Name");
console.log(name);
```



Operators:

1) Arithmetic
2) Logical
3) Comparison

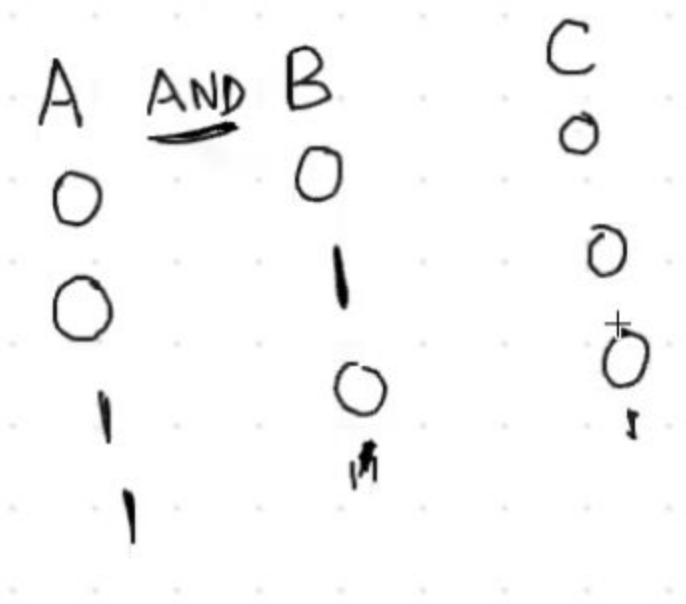
Operator vs Operand



1) **Arithmetic Operator :**

- **Addition +,**
- **Subtraction –,**
- **Multiplication *,**
- **Division /,**
- **Remainder %,**

2) **Logical: && (AND) | | (OR) (TRUE = 1 ; FALSE = 0)**



COMPARISON :

1) **Strictly Equal (===)**
2) **Normal Equal (==)**
3) **Less than or Equal (<=)**
4) **Greater than or Equal (=>)**
5) **Less than , Greater than (< , >)**
6) **Not Equal (!=)**
7) **Not Equal Strict (!==)**

BODMAS(**Bracket, Of, Division, Multiplication, Addition and Subtraction)**