MOBILE APPLICATION DEVELOPMENT

Introduction to JavaScript

Zaheer ul Hussain Sani

JavaScript

- Client Side Language (Vanilla JS, jQuery, React, D3.js, Vue.js etc)
- Server Side Language (Node.js, Express, MongoDB etc)
- Easy to Learn
- ES6 (ECMAScript 6) introduced and standardized in 2015

JS Syntax

- Very similar to other programming languages
- JavaScript uses the **var** keyword to declare variables.
- Universal principle for identifiers
- **Strings** are text, written within double or single quotes
- Output on console using **`console.log`**

```
var x, y, z;       // How to declare variables
x = 5; y = 6;      // How to assign values
z = x + y;         // How to compute values
```

```
"John Doe"

'John Doe'
```

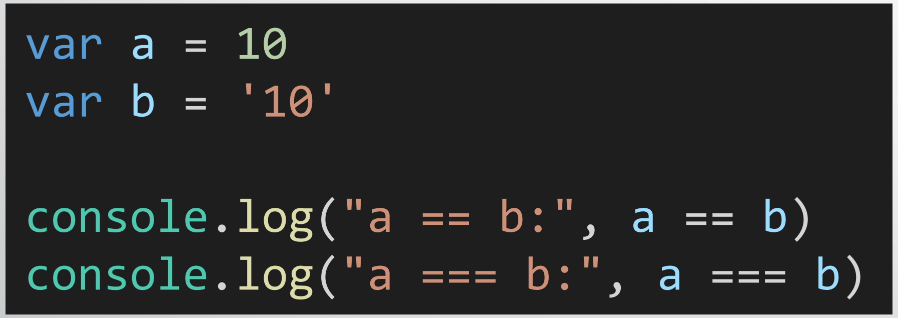
JS Variables

- Can create uninitialized variables or variable declaration (undefined)
- Change type of same variable anytime
- Recreate same variable again

```
var a; //undefined
var b = a + 10; // NaN
var x = 10
var x = 100
console.log(x) // 100
var x = "I am one hundred"
console.log(x) // I am one hundred
```

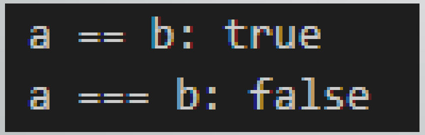
JS Operators

- Similar to other programming languages except === and !==

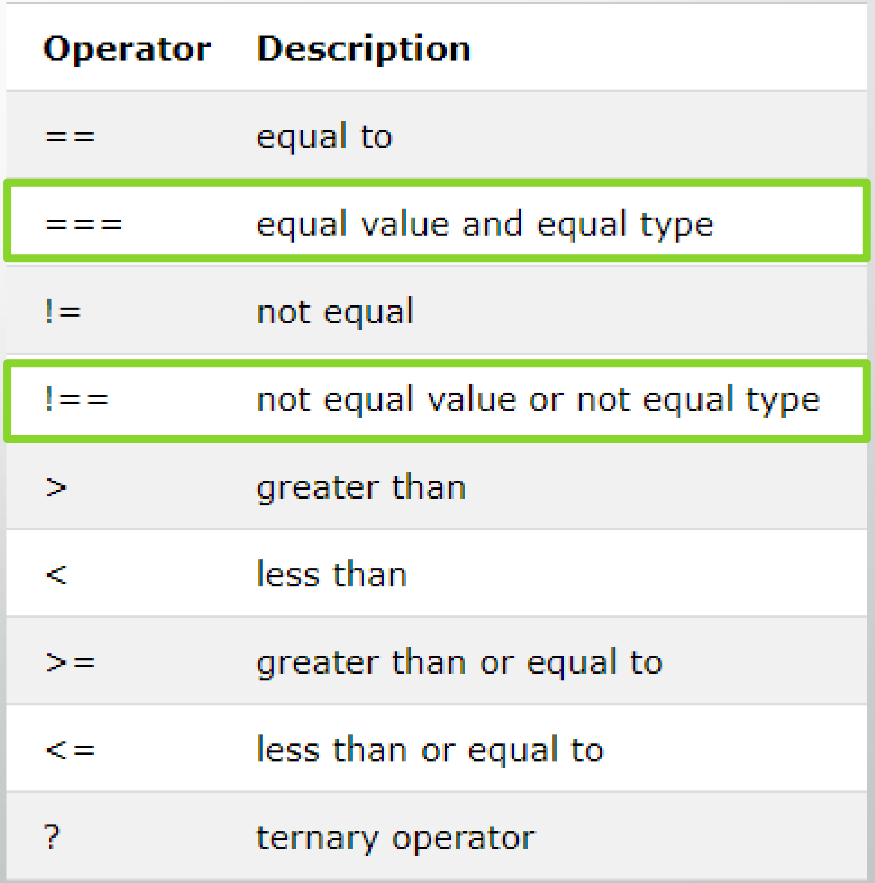


```
var a = 10
var b = '10'

console.log("a == b:", a == b)
console.log("a === b:", a === b)
```



```
a == b: true
a === b: false
```



Operator	Description
==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
?	ternary operator

JS DataTypes

- Number, String, Object etc
- Decimal

```
var x = 15.56
```

- Exponential

```
var x = 10e5
```

- Object

```
var person = { firstName:"John", lastName:"Doe", age:50 };
```

JS `typeof` Operator

- You can use the JavaScript `typeof` operator to find the type of a JavaScript variable.
- The `typeof` operator returns the type of a variable or an expression:

```
typeof ""                  // Returns "string"
typeof "John"              // Returns "string"
typeof "John Doe"          // Returns "string"
```

```
typeof 0                   // Returns "number"
typeof 314                 // Returns "number"
typeof 3.14                // Returns "number"
typeof (3)                 // Returns "number"
typeof (3 + 4)             // Returns "number"
```

JS undefined vs null

- In JavaScript null is "nothing". It is supposed to be something that doesn't exist.
    - Unfortunately, in JavaScript, the data type of null is an object.
    - You can empty an object by setting it to null
- undefined and null are equal in value but different in type:

```
typeof undefined           // undefined
typeof null                // object

null === undefined         // false
null == undefined          // true
```

Primitive Data

- A primitive data value is a single simple data value with no additional properties and methods.
- The `typeof` operator can return one of these primitive types:
  - `string`
  - `number`
  - `boolean`
  - `Undefined`

```
typeof "John"                 // Returns "string"
typeof 3.14                   // Returns "number"
typeof true                   // Returns "boolean"
typeof false                  // Returns "boolean"
typeof x                      // Returns "undefined" (if x has no value)
```

Complex Data

- The `typeof` operator can return one of two complex types:
  - `function`
  - `object`
- The `typeof` operator returns "object" for objects, arrays, and null.
- The `typeof` operator does not return "object" for functions.

```
typeof {name:'John', age:34} // Returns "object"
typeof [1,2,3,4]             // Returns "object" (arrays are objects)
typeof null                  // Returns "object"
typeof function myFunc(){}   // Returns "function"
```

ES6 Variables

	`var`	`let`	`const`
Scope	Function	Block	Block
Can change value after creation?	Yes	Yes	No