**1.Introduction**

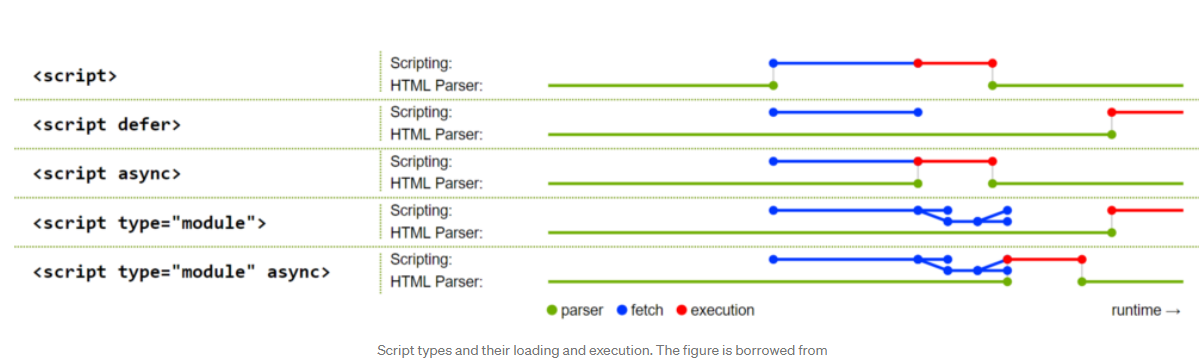
* 5 Primitive Data types (**Numbers, Strings, Booleans, null, undefined**). Strings -> Single and double quotes both ok rest are normal (string.length/ string[0] etc). Concat two string with + operator.
* The Variables that are declared but not initialized are set to undefined. Null is “explicitly nothing”.
* Alert (“”) -> opens a notification for user, prompt(“”) -> prompt for something from user, console.log(“”)-> prints on console only , clear()-> clears screen.
* <script src=””></script> is how to add the js file. If we put wrong filename/path it will show on console that the file wasn’t found.

**Importing Scripts correctly:**

* Ideally, we want to load the scripts as early possible but execute them after the HTML/CSS is parsed. Question is where do we put the <script> tag.
  + If we put the tag at the head of the page, we will download the script early, but also execute the script early as well before loading all the html element. So, this will cause error when the JavaScript try to work on the document since it’s not loaded fully.
  + If we put the tag in the bottom after body, it will load after the document is parsed and it will have some load time, which will give bad experience to users in case time is a lot

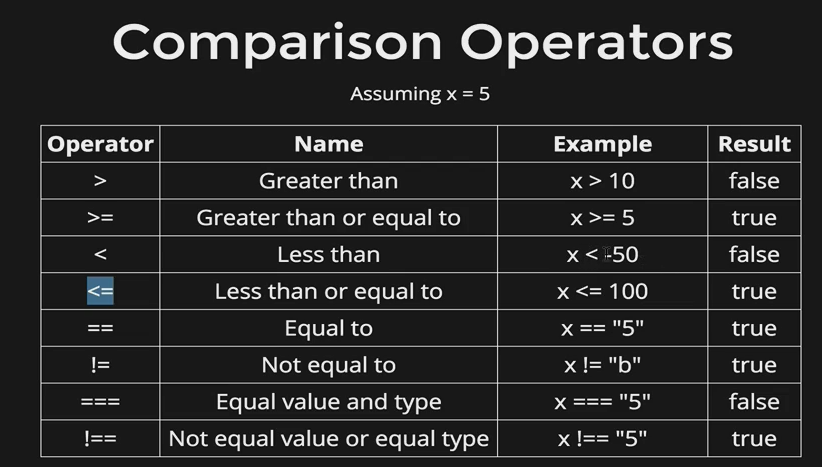
The solution is to get the best of the both worlds, put the script at the top but load it after parsing the document.

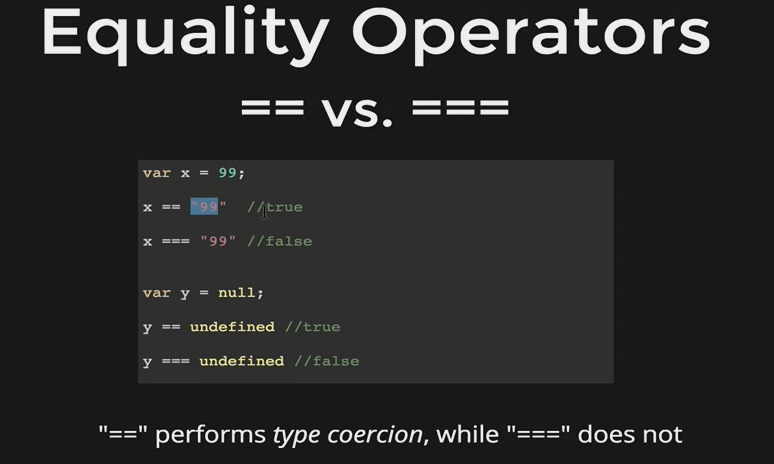
* + **Defer:** Defer keyword tells the browser that
    1. it should download the scripts right away
    2. it should not block parsing the html
    3. only execute the script after everything been parsed
  + **Async:** Sometimes we might have scripts that we want to load early, but also want to execute early because they don’t rely on the HTML document. In such case we can use Async keyword.
    1. browser starts loading the script as early as possible.
    2. The html parsing goes on in parallel since async is a non-blocking operation.
    3. As soon as script loading finish, it executes the script.
    4. The order of the execution in case multiple async isn’t guaranteed.



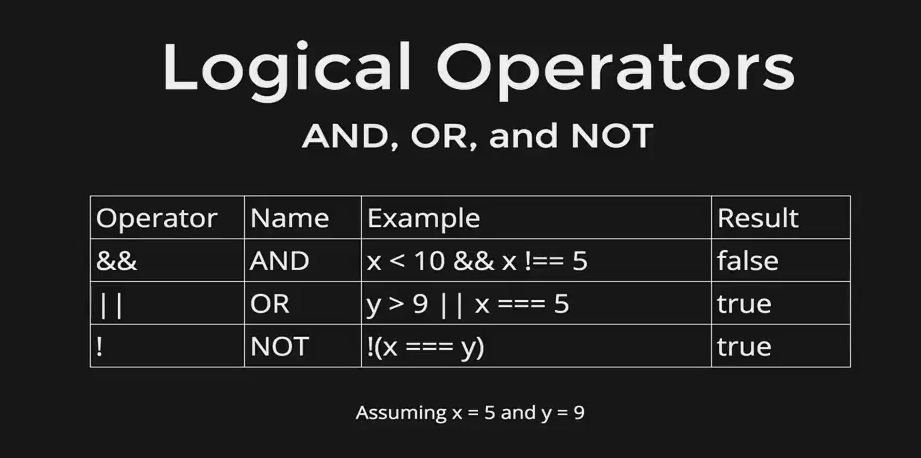
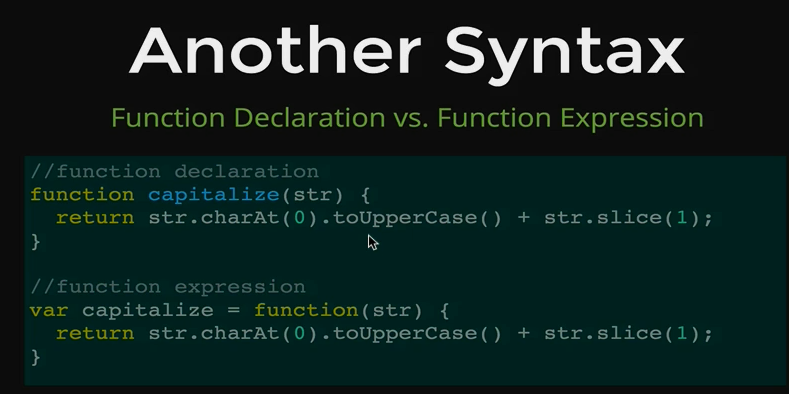
**2.Basics**

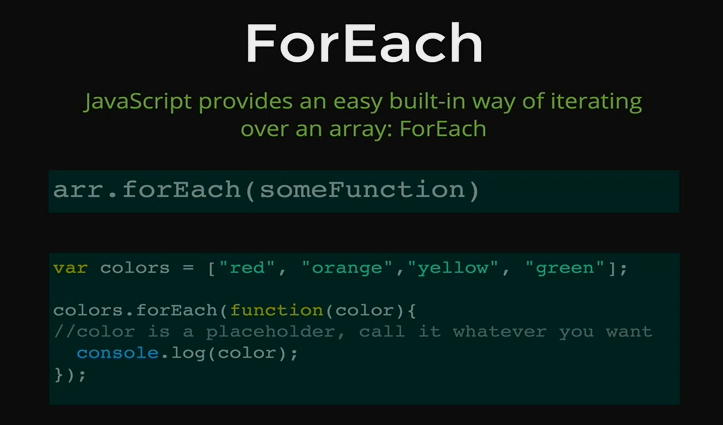
* **Operators**

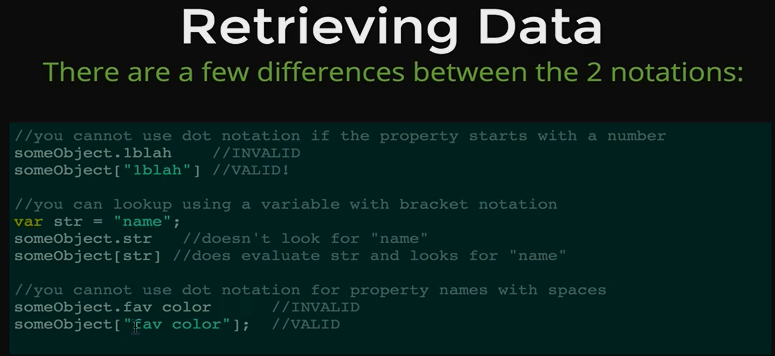






* **Logical Operators**
* **Loops:** Do while, While, for loop -> nothing new here.
* **Functions** -> you know already mostly. 
* We can pass functions as argument inside a function. We don’t need parenthesis when we use function as argument. For example -> setInterval(somefunction, 200);
* **Array** example -> var friends= [“Charlie”,” Liz”];
* **Array Method -> push/pop** -> like stack, **shift/unshift** -> add first item to an array, removes first item of an array, **indexof() ->** returns the index of the element, **slice(x,y)** -> to copy part of the array, x is starting index and y is the ending index. Rest of the methods we can look up at documentation
* **Array Iteration** ->



* **Object** are stored in a key-value pair. For example -> var person = {name:” Cindy”, age: 32, city: “Dhaka”}; To access a data -> person[“name”] or person.name
* **Creating** -> var person = new object (); person. Age=21; and so on /

var person = {name:” Cindy”, age: 32, city: “Dhaka”};

var person= {}; person.name=” Cindy”;

* **Nested Objects** -> example of an object of post