**Web application - 3 tier Architecture**

1. **Front-end:**

* Anything that a user faces is a part of Frontend.
* Frontend development has everything to do from design to dynamism of a web application.

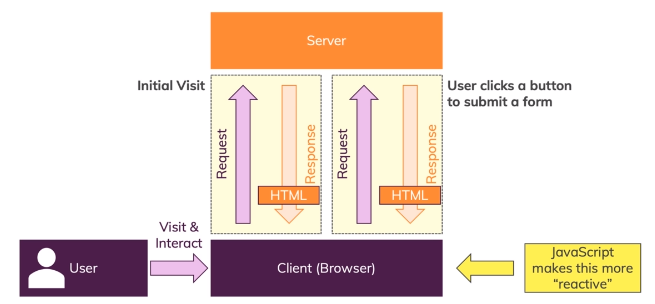
1. **Back-end:**

* Backend is like the brain.
* It has everything to do with the logical.
* It also takes care of data storage and management by connecting to the database.
* It can combine various services to produce the desired results.

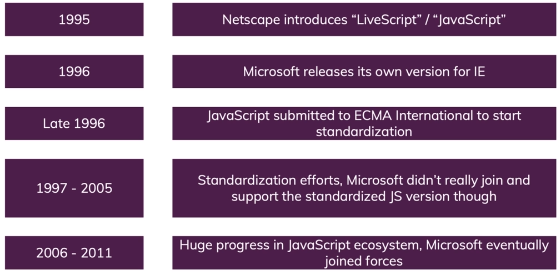
1. **Database:**

* Used to store data.

**How do Web Pages Work?**

****

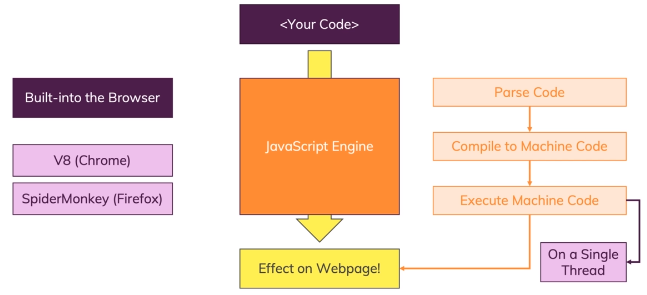
**Brief Overview of the JavaScript History**

****

**JavaScript Introduction**

* JavaScript is a client side scripting language (interpreted programming language).
* JavaScript is a dynamic, weakly typed language which compiled at run time.
* Open source and cross-platform.
* Case sensitive.
* Most commonly used as a part of web pages.
* JS was created to make web pages more Dynamic (Change content on a page directly from inside the browser).
* JavaScript was initially known as **LiveScript.**
* Supported by all major browsers and enabled by default.
* JavaScript was initially created as a browser-only language, but now it is used in many other environments as well.

**How is JavaScript Executing?**

****

**What JavaScript can do?**

* Add new HTML to the page, change the existing content, modify styles.

(**DOM Manipulation**).

* React to user actions, Execute on mouse clicks, pointer movements, key presses. (Events)
* Send requests over the network to remote servers, read and write files (Ajax).
* Get and set cookies, ask questions to the visitor, show messages.
* Remember the data on the client-side (“local storage”).

**Advantages of JavaScript**

* Executed on the client side.
* Instance response to the visitors.
* Rich interfaces.
* Speed.
* Less server interaction.

**Disadvantages of JavaScript**

* Code Always Visible.
* **Stop Render** : JavaScript single error can stop rendering with the entire site. However browsers are extremely tolerant of JavaScript errors.
* Less Security.

**Where to write JavaScript**

1. **Internal JS**

Internal Javascript code is code that's placed anywhere within the web page between the HTML tags.

<script>

alert("Hello Javatpoint");

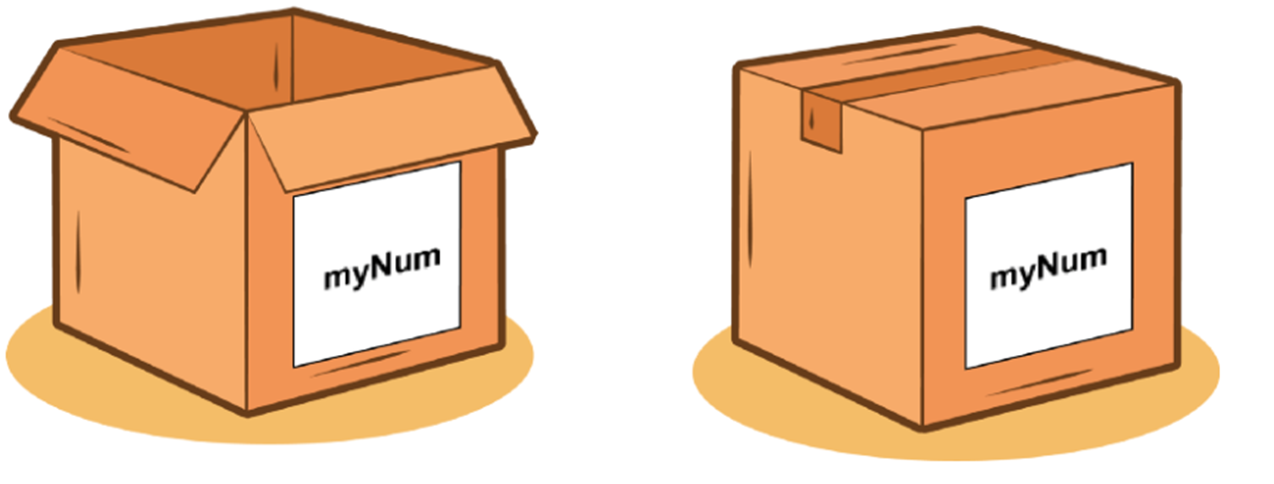
</script>

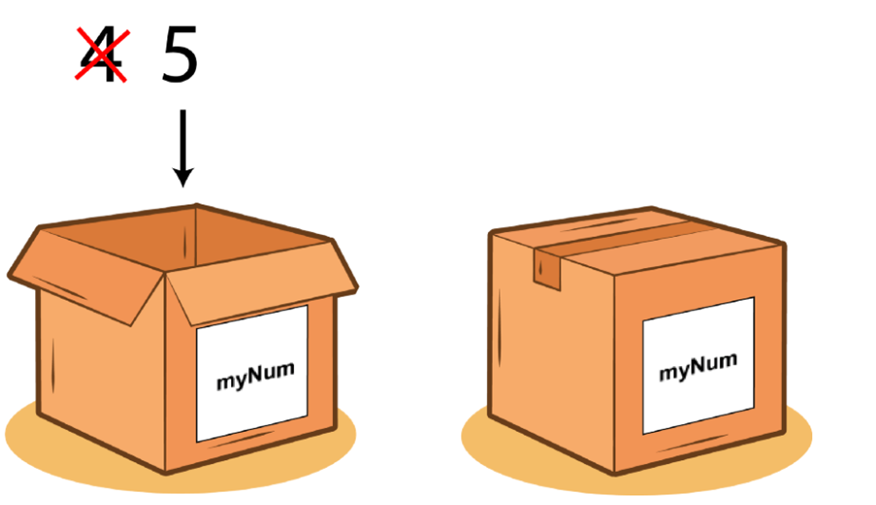
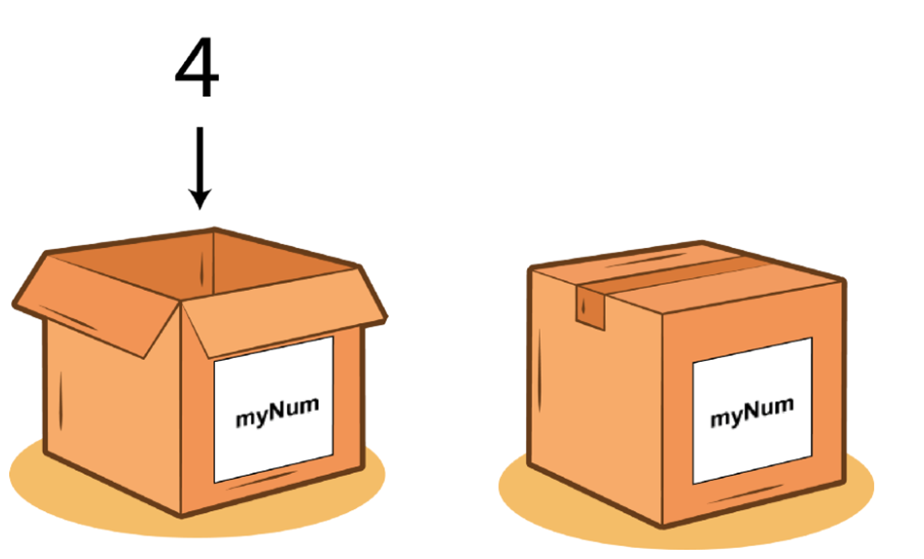
**2. External JS**

Javascript code placed in a file separate from the HTML code is called external Javascript. External Javascript code is written and used in the same way as internal Javascript. The file should have the ".js" extension.

**JavaScript Variables**

* Variables are containers for storing data (values) that hold information and allow us access them later.
* All JavaScript variables must be identified with unique names
* These unique names are called identifiers
* We will think of this as a box that has a label on it.





**Declaring JavaScript Variables and Rules**

* Declare a JavaScript variable with the **var** keyword.
* Variables can contain letters, digits, underscores, and dollar signs.
* Variables must begin with a letter.
* Variables are case sensitive.
* Can not use Reserved words as Variables.
* Use proper names and if it has more than one word use camel case.

A JavaScript variable is simply a name or container for storing data values.

There are two types of variables in JavaScript : **local variable** and **global variable**.

**Local Variable**: A JavaScript local variable is declared inside block or function. It is accessible within the function or block only.

**Global Variable**: A JavaScript global variable is declared outside the function or declared with window object. It can be accessed from any function.

**Data Types**

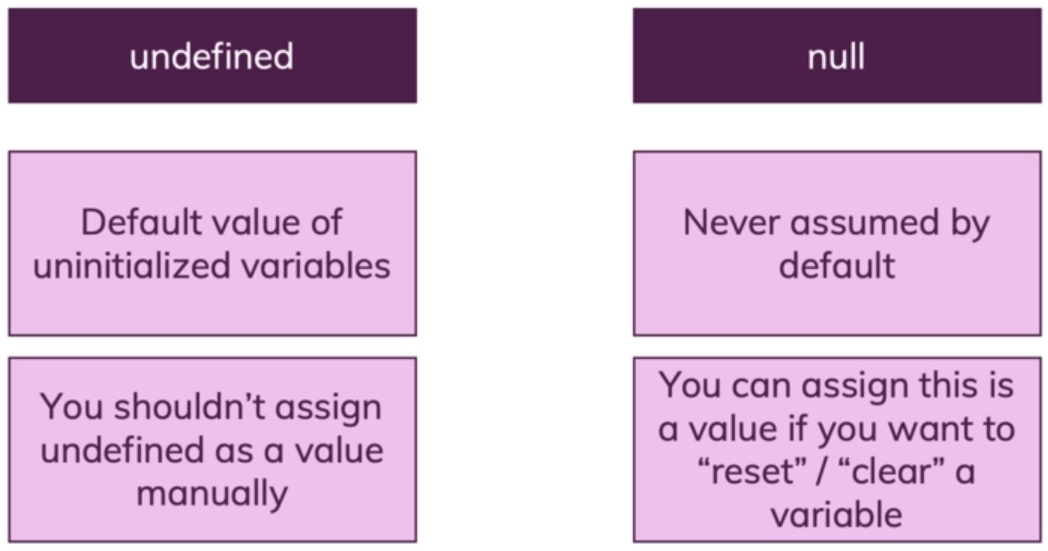
Data types plays an important role in programming languages.

To be able to operate on variables, it is important to know something about the type.

Without knowing data types, a program cannot handle safely.

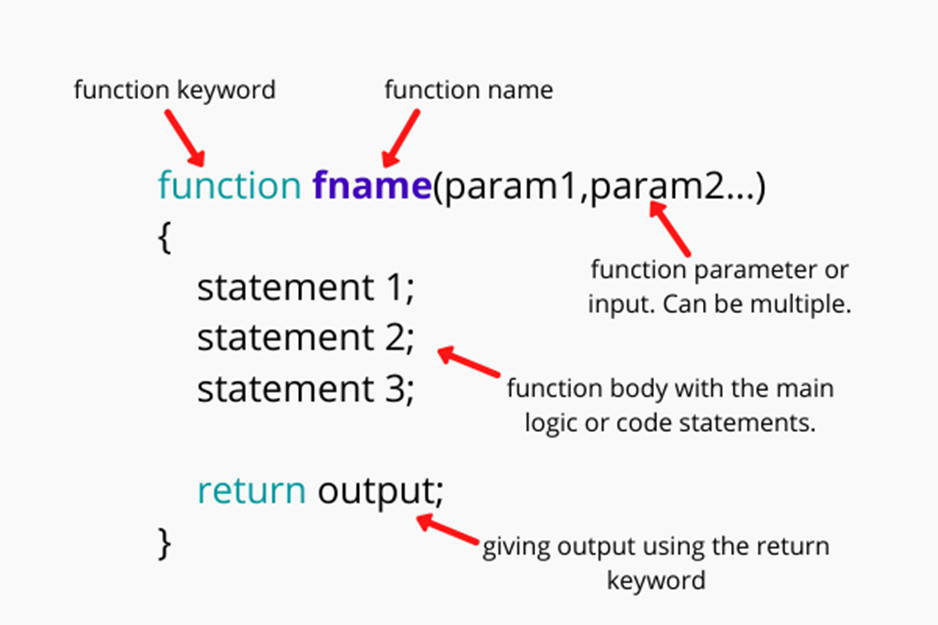
**List of Data Types:**

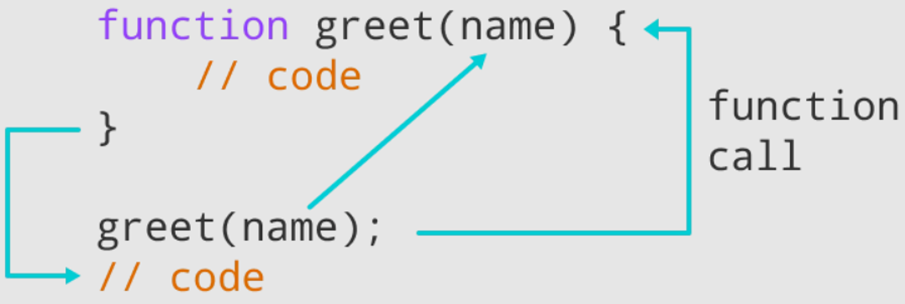
1. **String:** Sequence of characters which is denoted by single or double quotes.
2. **Number**: Represent a numeric value with decimal or without decimal.
3. **Array**: Array is a special variable used to store multiple values in a single variable.
4. **Object:** Object is a variable containing many variables.
5. **Boolean:** Represents value in two states: true or false.
6. **Null**: null is "nothing". Something that doesn't exist.
7. **Undefined:** JS can’t find any value for this. variable can be emptied, by setting the value to undefined.

****

**JavaScript Function**

Functions are one of the fundamental building blocks in JavaScript. A function is a JavaScript procedure, a set of statements that performs a task or calculates a value.





**How to create a Function?**

JavaScript functions are defined with the **function** keyword and followed by the functionName.

**There are 3 different types of functions,**

Functions declared with the following syntax are not executed immediately. They are "saved for later use" and will be executed later, when they are invoked (called upon).

These types of functions work if you call them BEFORE or AFTER where they have been defined. If you call a deceleration function before where it has been defined - Hoisting - works properly.

1. **Function declaration // Named function**

Function declarations load before any code is executed.

***function functionName()***

***{***

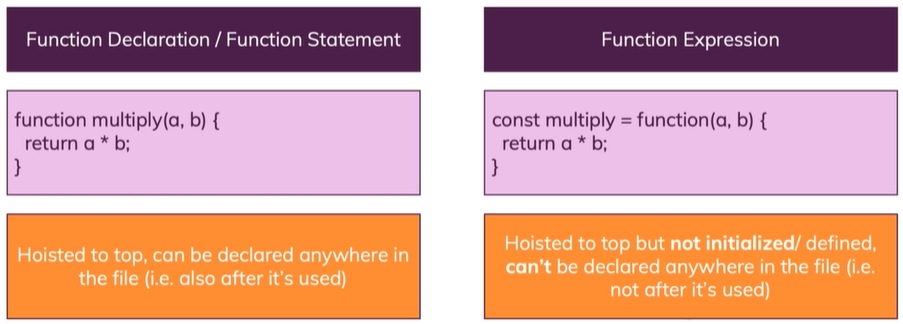
***}***

1. **Function expression // anonymous** function

Function expressions load only when the interpreter reaches that line of code.

var a = function() { };

Also this function is called as **anonymous** function (a function without a name).



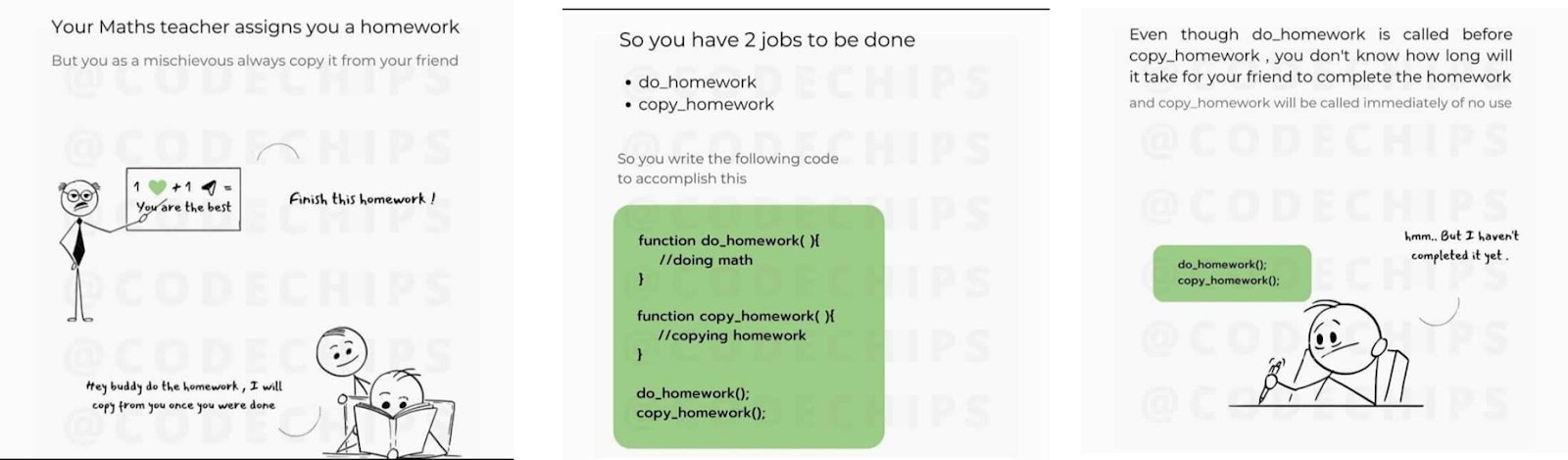
**3**. **Constructor function**

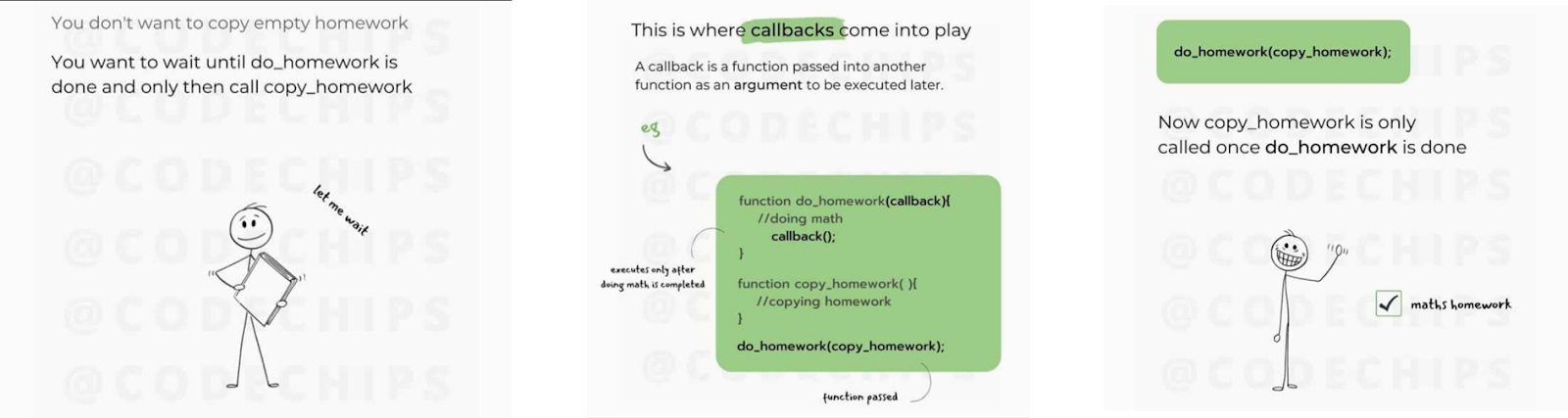
Constructors are like regular functions, but we use call them with the "new" keyword. There are two types of constructors:

1. Native (built-in) constructors like **Array, Number** and **Object**
2. Custom Constructor functions.

**Function Invocation methods**

1. Self Invocation
2. By Events
3. Invoking a function from another one function



****

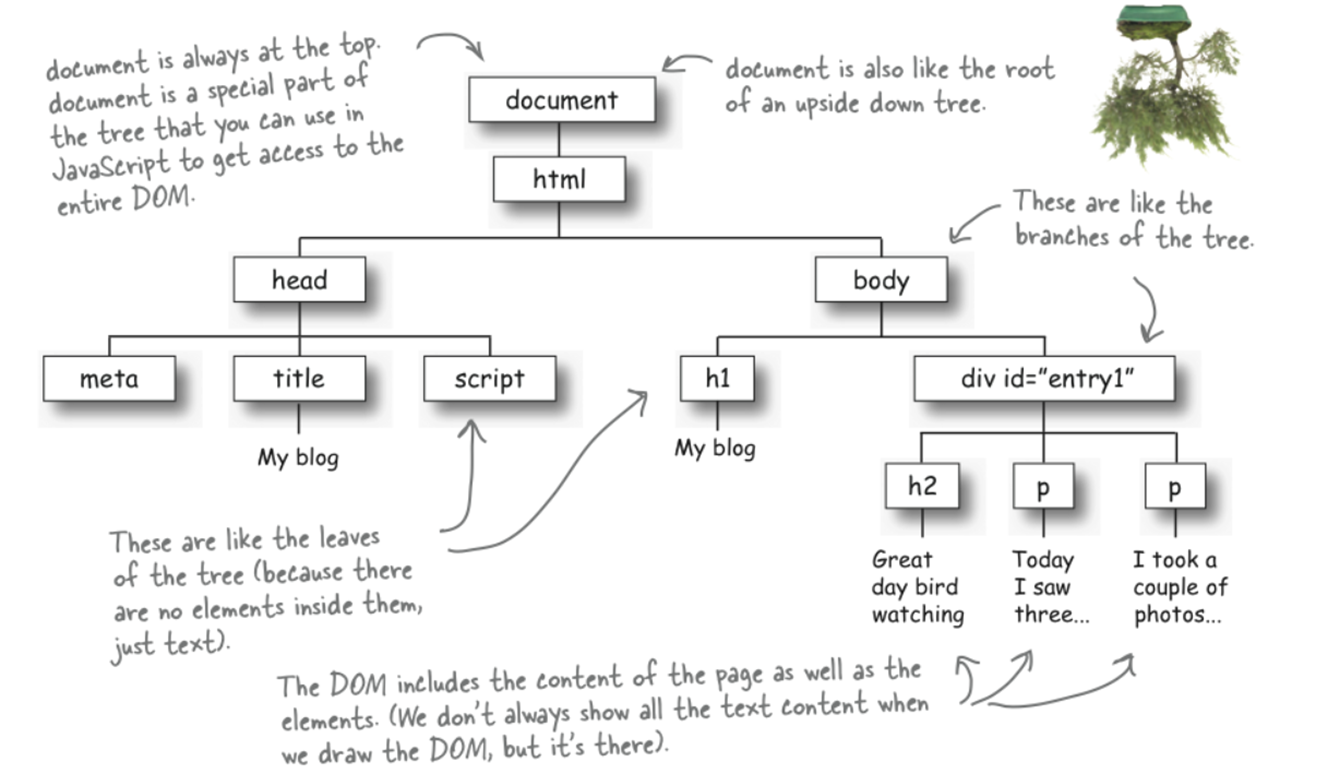
**Difference between function and methods:**

**Function**: A function is a piece of code which is called by Name.

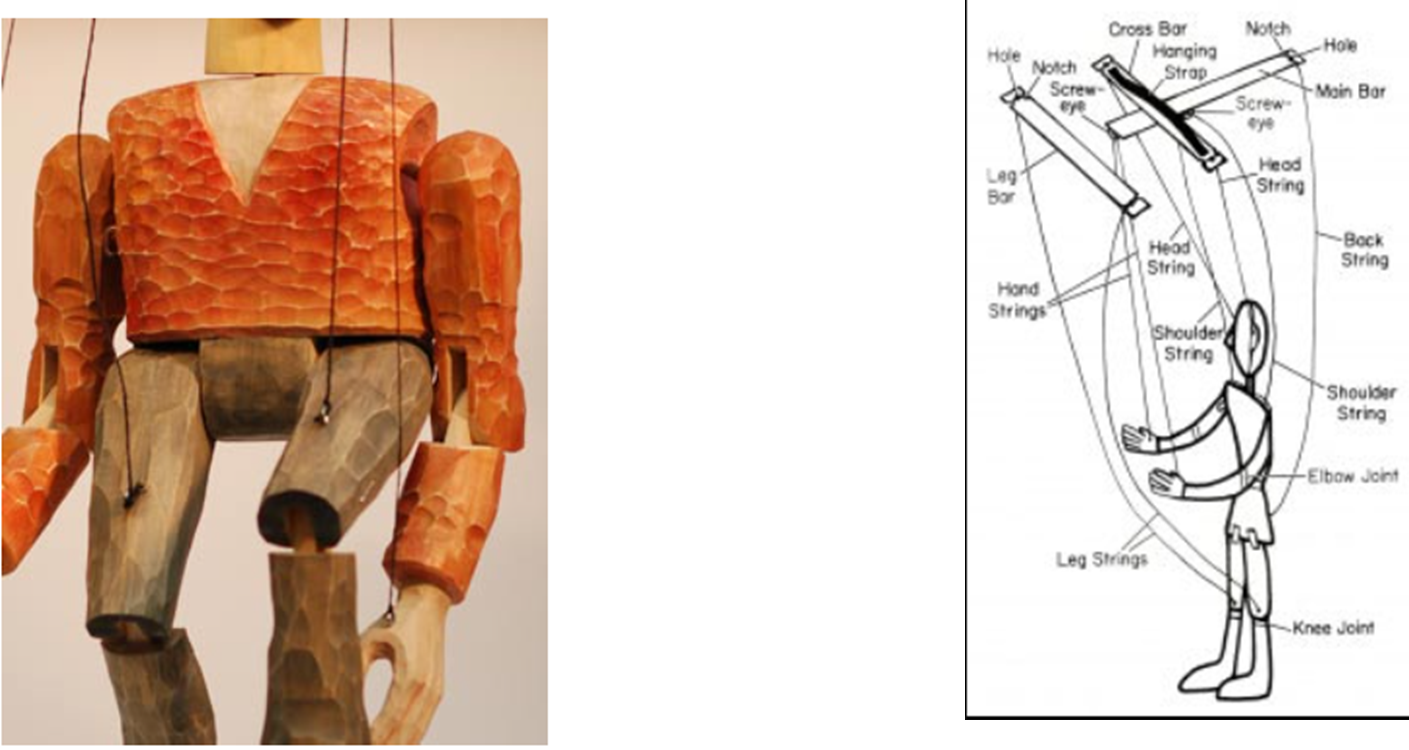
**Method**: A method is a piece of code which is called by Name that is associated with an Object.

**What is DOM ?**

* When a web page is loaded, the browser creates a Document Object Model of the page
* The HTML DOM model is constructed as a tree of Objects:



**HTML page as a puppet**, then the **DOM will be the strings** and the points at which they attach to the puppet



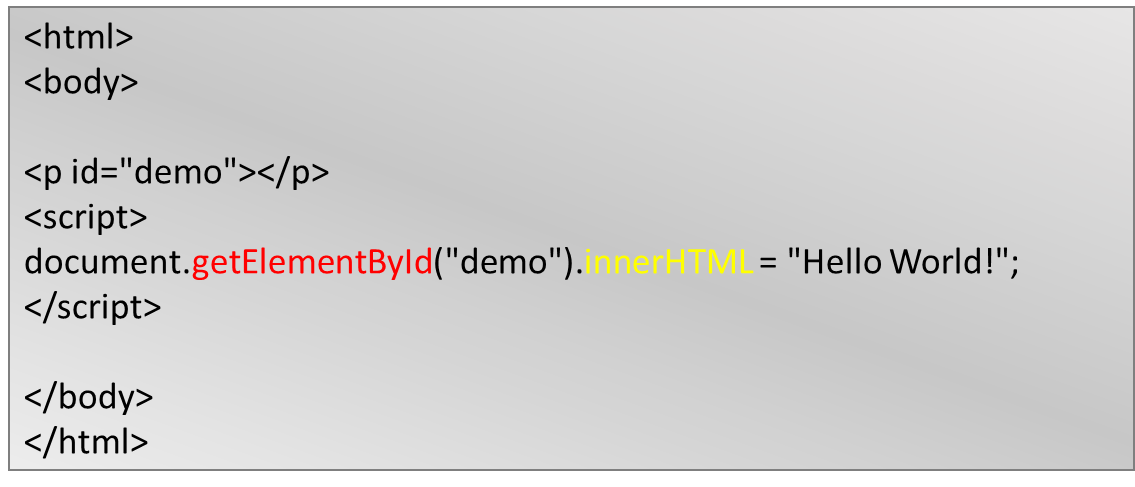
**What can we do using DOM?**

Using DOM, JavaScript gets all the power it needs to create/update HTML:

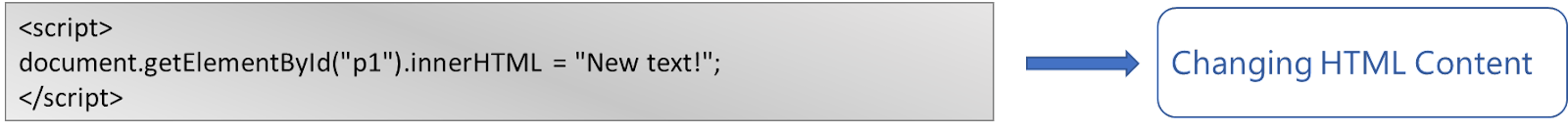
* JavaScript can change all the HTML elements in the page
* JavaScript can change all the HTML attributes in the page
* JavaScript can change all the CSS styles in the page
* JavaScript can remove existing HTML elements and attributes
* JavaScript can add new HTML elements and attributes
* JavaScript can react to all existing HTML events in the page

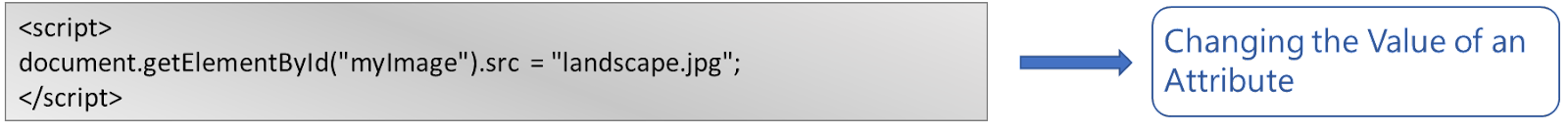
**DOM Methods**

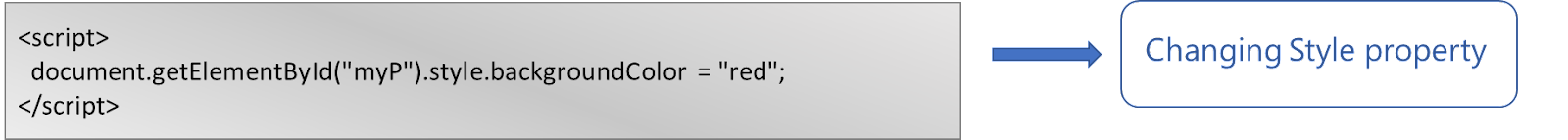
* HTML DOM methods are actions you can perform (on HTML Elements)
* HTML DOM properties are values (of HTML Elements) that you can set or change.



**getElementById** - To change/modify the content of an HTML element







**Conditional Statements**

Based on condition we can execute a particular task.

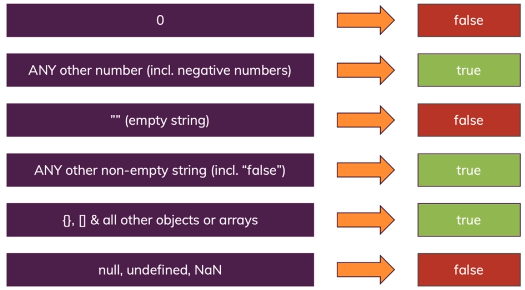
1. if .. else

i. **If** : If you have used multiple if statements then ***if*** the condition is ***true*** all will be executed.

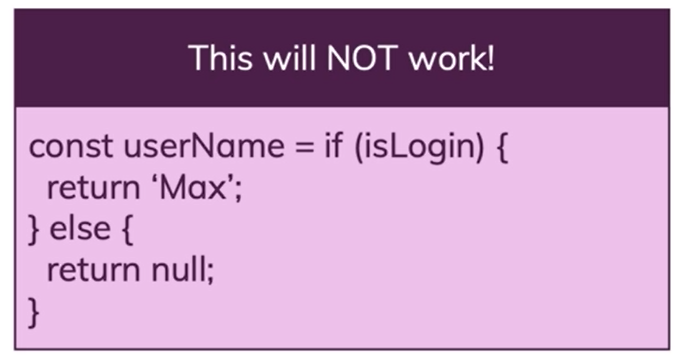
ii. **if else if** : If you have used ***if and else if*** combination only one will be executed where first comes the true value.

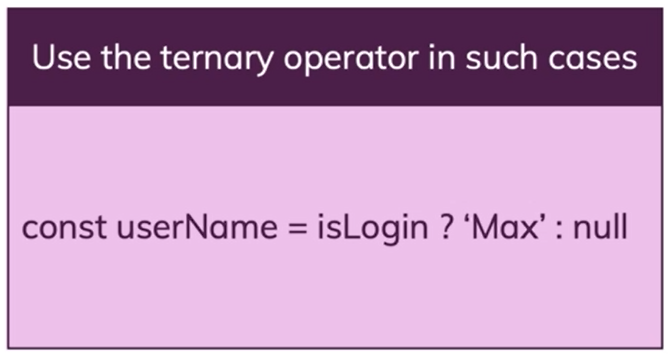
1. switch

**Truthy and Falsy Values**



**Conditional (ternary) operator**





**Syntax**

*condition ? exprIfTrue : exprIfFalse*

**condition**

An expression whose value is used as a condition.

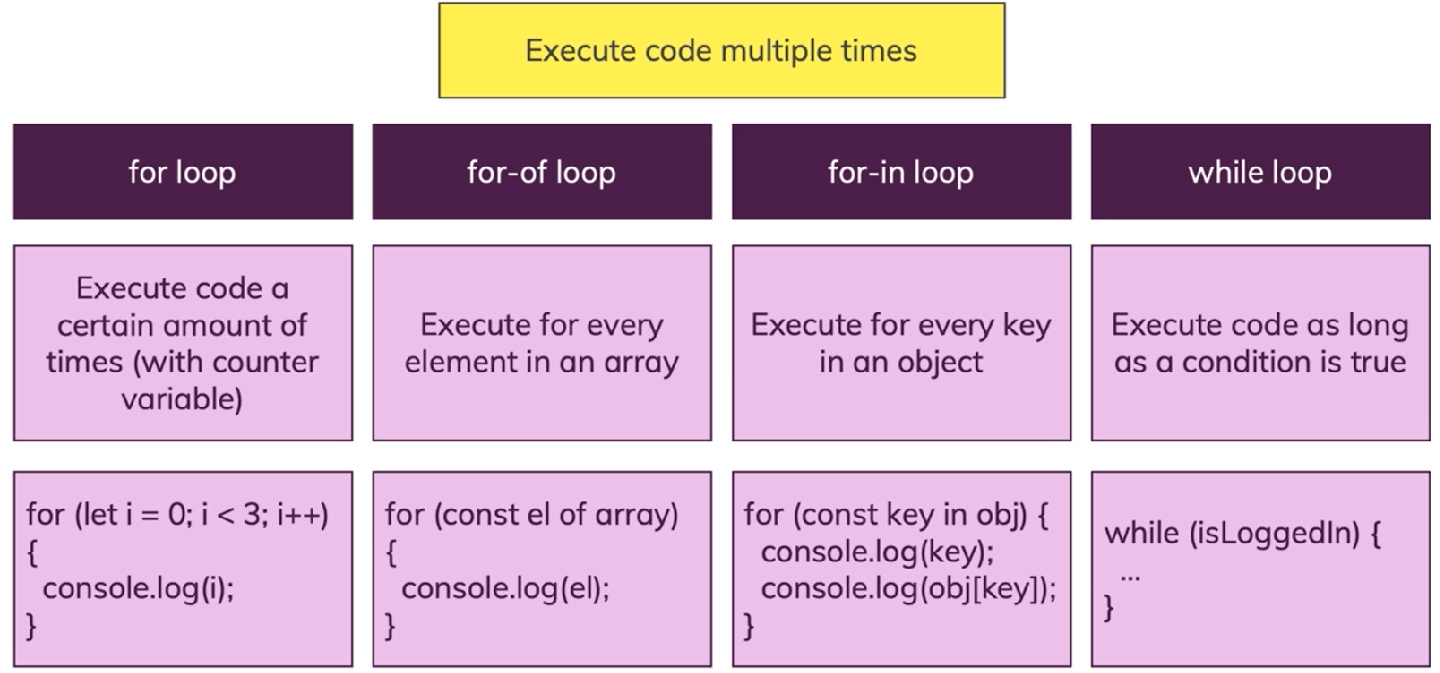
**exprIfTrue**

An expression which is evaluated if the condition evaluates to a truthy value (one which equals or can be converted to true).

**exprIfFalse**

An expression which is executed if the condition is falsy (that is, has a value which can be converted to false).

**Iteration Statements**

****

Execute a set of statements several times based on the given condition is true.

1. For
2. While
3. Do while

*do*

*{*

*wash\_hands();*

*}*

*while (hands\_are\_dirty());*

*do*

*{*

*eat();*

*}*

*while (still\_hungry());*

1. forEach
2. For in
3. For of

**String Inbuilt Methods**

Sequence of characters which is denoted by single or double quotes.

**To Find String Length**

String.length;

var str = “Hi”;

str.length; // 2

**charAt()**

Returns the character at the specified index position of the string.

**Syntax**:

string.charAt(index)

**indexOf()**:

Returns the index position of the **first** occurrence of a specified value in a string.

**Syntax**:

string.indexOf(specifiedSubString, startingIndexPosition);

**lastIndexOf():**

Returns the index position of the **last** occurrence of a specified value in a string.

**Syntax**:

string.lastIndexOf(specifiedSubString, EndIndexPosition);

**search():**

This method searches a **string** or a **regular expression** for a specified value, and returns the position of the match.

**Syntax**:

string.search(searchvalue)

**Extracting a Substring from a main string:**

string.slice(startIndexPosition, EndIndexPosition)

string.substring(startIndexPosition, EndIndexPosition)

string.substr(startIndexPosition, length)

\*Exclude EndIndexPosition

**replace()**

Searches in a string for a specified value and returns a new string where the specified values are replaced.

**Syntax**:

string.replace(searchString, newString)

*A regular expression is a sequence of characters that define a pattern.*

**How to write regular expression?**

***The Plus symbol ( + ):***

It tells the computer to repeat the preceding character (or set of characters) for atleast one or more times(upto infinite).

**Example** :

**The regular expression ab+c will give abc, abbc,**

**abbc, … and so on.**

***The curly braces {…}:***

It tells the computer to repeat the preceding character (or set of characters) for as many times as the value inside this bracket.

**Example** :

**{2, 10} means that the preceding character is to be repeated 2**

**times, {min,} means the preceding character is matches min or more times. {min,max} means that the preceding character is repeated at**

**least min & at most max times.**

***Wildcard – ( . )***

The dot symbol can take place of any other symbol, that is why it

is called the wildcard character.

**Example** :

**The Regular expression .\* will tell the computer that any character can be used any number of times.**

***Optional character – ( ? )***

This symbol tells the computer that the preceding character may

or may not be present in the string to be matched.

**Example** :

**We may write the format for document file as – “[doc]?”**

**The ‘?’ tells the computer that x may or may not be**

**present in the name of file format.**

***The caret ( ^ ) symbol:***

Setting position for match :tells the computer that the match must start at the beginning of the string or line.

**Example** :

**^\d{3} will match with patterns like "901" in "901-333-".**

***The dollar ( $ ) symbol:***

It tells the computer that the match must occur at the end of the string or before \n at the end of the line or string.

**Example** :

**-\d{3}$ will match with patterns like "-333" in "-901-333".**

***Character Classes***

A character class matches any one of a set of characters. It is used to match the most basic element of a language like a letter, a digit, space, a symbol etc.

/s : matches any whitespace characters such as space and tab

/S : matches any non-whitespace characters

/d : matches any digit character

/D : matches any non-digit characters

/w : matches any word character (basically alpha-numeric)

/W : matches any non-word character

/b : matches any word boundary (this would include spaces, dashes, commas, semi-colons, etc)

**startsWith()**

Checks whether a string begins with specified characters.

**Syntax**:

string.startsWith(searchString, startIndexPosition)

**endsWith()**

Checks whether a string ends with specified characters.

**Syntax**:

string.endsWith(searchString, Length)

**includes()**

Checks whether a string contains the specified string/characters

**Syntax**:

string.includes(searchString, startIndexPosition)

**match()**

Searches a string for a match against a regular expression, and returns the matches

**Syntax**:

string.match(regexp)

It will return an Array with matched values

**Regular expression for email**:

**^([a-zA-Z0-9\_\-\.\$]+)@([a-zA-Z0-9\_\-\.]+)\.([a-zA-Z]{2,12})$**

The above regular expression can be used for checking if a given set of characters is an email address or not.

**repeat()**

Returns a new string with a specified number of copies of an existing string

**Syntax**:

string.repeat(count)

**toString()**

Returns the value of a String object

**Syntax**:

string.toString()

**split()**

Splits a string into an array of substrings

**Syntax**:

string.split(separator, limit)

**toUpperCase()**

Converts a string to uppercase letters

**Syntax**:

string.toUpperCase()

**toLowerCase()**

Converts a string to lowercase letters

**Syntax**:

string.toLowerCase()

**trim()**

Removes whitespace from both ends of a string

**Syntax**:

string.trim()

**Number Inbuilt Methods**

**isFinite()**

Checks whether a value is a finite number

**Syntax**:

Number.isFinite(num)

**isInteger()**

Checks whether a value is an integer

**Syntax**:Number.isInteger(num)

**toFixed(x)**

Keeping a specified number of decimals

**Syntax**: num.toFixed(2)

**toString()**

Converts a number to a string

**Syntax**: num.toString();

**Converting String to Numbers**

1. **Number()** Returns a number which is converted from its argument.
2. **parseInt()** parses a string and returns a whole number. Spaces are allowed. Only the first number is returned.
3. **parseFloat()** parses a string and returns a number. Spaces are allowed. Only the first number is returned.