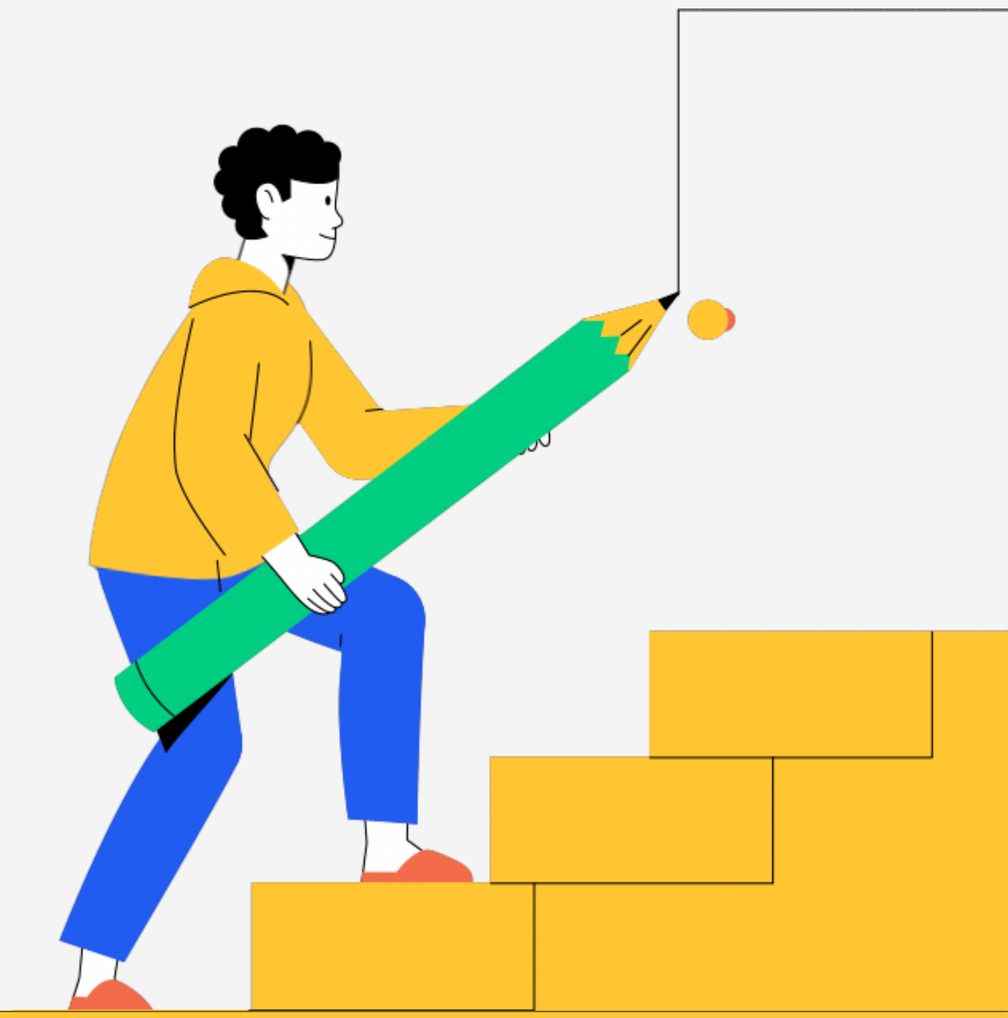
IT SNP

1 Month JS Workshop

JAVASCRIPT CONTINUED....



SAROJ DAHAL
Software Developer (Web + Mobile)

ITSNPORG
An organization working for the welfare of techies in Nepal.

ABOUT ME

Hi! **My name is Saroj Dahal**. I am full-stack web Developer & Mobile App Developer.

I started to work on the market in 2017 after high school (+2).

I worked in Infinite Tech Solutions as a Java Developer & a Web Developer.

After that, I had worked on outsourced projects.

Currently, I am working on my Startup company named KodeBliss.

Besides Tech, I am into Business thing, Practicing Astrology, Exploring Spirituality and Loving the Astronomy.

isarojdahal

isarojdahal

ITSNPORG

Topics we will be learning.

JavaScript Syllabus :
https://docs.google.com/document/d/1uvNTZgxcOu44RUn9oxEi3enfuy1nNLWX25hAfJxEsJc

What We're Learning Today

Here are the topics we will cover this evening:

- Brief Intro to ECMAScript
- History of JavaScript.
- Understanding JavaScript Engine
- Exploring some ES6 Features.

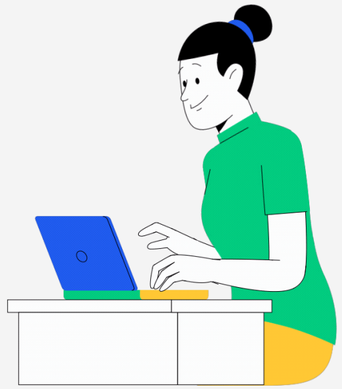
ECMA Script

ECMA Stands for European Computer Manufacturers Association. Commonly known as ECMA International.

ECMA International is an organization that creates standards for Technology. (Operating Since 1961)

ECMA Script is the standard, guidelines, or rules that a language should follow for uniformity of code.

ECMA Script is a Standard for scripting Language such as JavaScript, JScript.



History of ECMA Script 

At the beginning age of Browser, Netscape and Internet Explorer were Popular browsers. They had different Scripting languages for their browser.

Windows came with JScript whereas Netscape came with Mocha. The problem was the website made using JScript was unable to run on Netscape. Similarly, the website made using Mocha was unable to run on InternetExplorer.

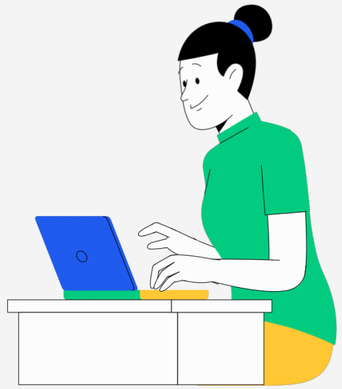
To Solve this, ECMA international came with the standard called ECMA Script, which standardized Script that runs on the Web.

Later on , All Scripting Languages such as JavaScript, Action Script, JScript began to follow ECMA standard.

JavaScript was initially Named Mocha and then LiveScript. Later it was named JavaScript and launched in 1995.

ECMA Script on Detail.

https://en.wikipedia.org/wiki/ECMAScript



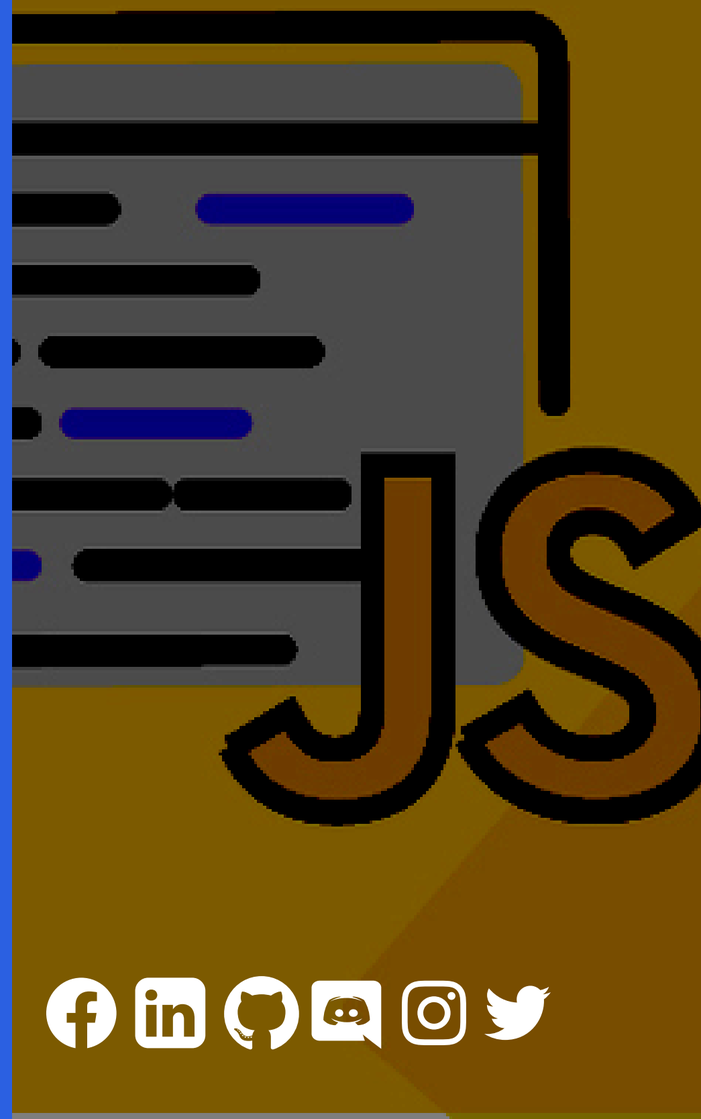
JavaScript .engines.

JS Engine is software that resides in a web browser that is responsible for converting/parsing JavaScript source code to machine code.

Examples of JS Engines :
V8 engine (Chrome), SpiderMonkey (FireFox), JavaScript Core (Safari), Chakra (Internet Explorer)

ES6 Features

Template Literals
Arrow Functions
Array & Object Destructuring & Spread Operators.
Modules
and soon...



These ES6 Features are very crucial for learning any JS related Frameworks and Technologies.

ITSNPORG

Template Literals

Also known as String literals.

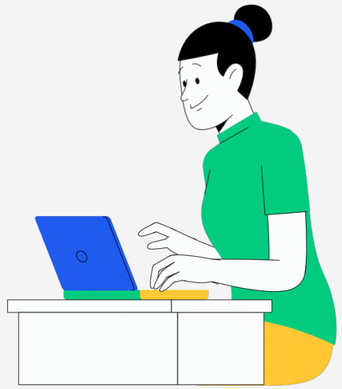
Template Literals are the literals that allow us to embed expressions, usually in the string.

Backticks (`) are used in order to perform template interpolation.

Useful things that can be achieved are :

1. Multiline Strings.
2. Direct variable usage.
3. Easily hookup an expression.

and makes far more easy to read and understand the code.



Arrow Functions

Also known as Anonymous function or lambda function expression.

Arrow Functions are exactly as normal functions; their working are same, but they only differ on syntax.

Arrow functions turns out to be very handy when we have to write small piece of code snippets. And, are mostly used in Higher Order Functions (HOF).

What We're Learning Today

Here are the topics we will cover this evening:

- Continue ES6 Features...
  - Destructuring Objects & Arrays.
  - Spread Operators
  - Modules

Destructuring Assignment

Destructuring assignment is a syntax that allows you to assign object properties or array items as variables.

This can greatly reduce the lines of code necessary to manipulate data in these structures. There are two types of destructuring: Object destructuring and Array destructuring.

Spread Operators

IT SNP

Spread Operator is a new operator introduced in ES6 Features; which is commonly used for expanding the values, taking the shallow copies of an object or of the array, or also can be used to take varied no.of arguments in the function.

1. Expanding the values of Arrays and objects into new ones.
2. Taking shallow copies of arrays or objects.
3. To take multiple params in functions.
4. and many more..

ES6 Modules

JS Modules is one of the best features of JavaScript, which helps us to make code reusable and helps to maintain the code.

The main purpose of Modules is to import & export different pieces of code from one file to another. Hence, making the code more easier to understand.

What We're Learning Today

Here are the topics we will cover this evening:

- Continue ES6 Modules...
- for-in and for-of loops
- Higher-Order Functions (map, filter, sort, reduce)

for..in and for..of Loops

SNP

Both forof and forin loops statements iterate over lists; but the value iterated are different. for...in returns a list of keys of the object iterated; whereas for...of returns a list of values

Higher Order Functions

In simple words, A function that accepts other functions as arguments is called a higher-order function.

OR

Also, A function that returns other functions from the existing function is also called a higher-order function.

Some of the higher-order functions are forEach, filter, map, reduce, sort. In HOF, arrow functions are commonly used, as it makes shortcode and also makes code easier to read.

First class Functions

In JavaScript, functions are treated as objects.

1. They can be stored in a variable.
2. Can pass as a value.
3. Return them as a value.

What We're Learning Today

Here are the topics we will cover this evening:

- Continue Higher-Order Functions
  - Explore Higher-Order Functions such as map, filter, sort, reduce.
- Callback Functions

ITSNPORG

Higher Order Functions in JavaScript

In JavaScript, Array comes up with lots of Higher-Order functions such as forEach, map, filter, sort, reduce.. etc.

These functions are very handy when playing with Arrays in JS.

ITSNPORG

forEach

The forEach() method executes a provided function once for each array element.
returns undefined.

map

The map() method maps/transform the data into new ones. The map doesn't affect the current value of array; and returns the mapped data in array.

filter

The filter() method filters the data in the array as per our condition. Similar to map, it also returns value in array;
have to return true if condition is satisfied else false.

reduce

The reduce() method reduces our given array into a single value.

sort

The sort() method sorts the data in the array as per our given condition, and returns the sorted array.
have to return 1 if true else -1

IT SNP

ITSNPORG

What We're Learning Today

Here are the topics we will cover this evening:

- Continue Higher-Order Functions
  - sort().
  - Chaining of Higher Order Functions.
- Callback Functions.
- AJAX

Callback Functions

A callback function is a function that is passed as an argument to another function, to be “called back” at a later time.

Callbacks are just addon-condition on Higher-Order functions.

Callbacks are generally used when the function needs to perform events before the callback is executed, or when the function does not (or cannot) have meaningful return values to act on, as is the case for Asynchronous JavaScript (based on timers) or XMLHttpRequest requests.

AJAX

AJAX Stands for Asynchronous JavaScript And XML.

AJAX is not a programming language. It is one of the techniques of JavaScript, which enables us to send/receive data file asynchronously on the web.

One Main Advantage of AJAX is: We can update the content on the webpage and server without refreshing the whole webpage. Also, it helps in loading webpage faster.

Requesting Data

IT SNP

For AJAX:

- We first create an object of XMLHttpRequest.
- Check the status of the response. (readyState, status)
- Open the request. open()
- Send the request. send()

and handle our response from respective callback functions.

Note
In modern JavaScript, We can make use of fetch() to send/receive data.
In the old way, We use to make use of ActiveXObject to send/recieve data.

Codes.

IT SNP

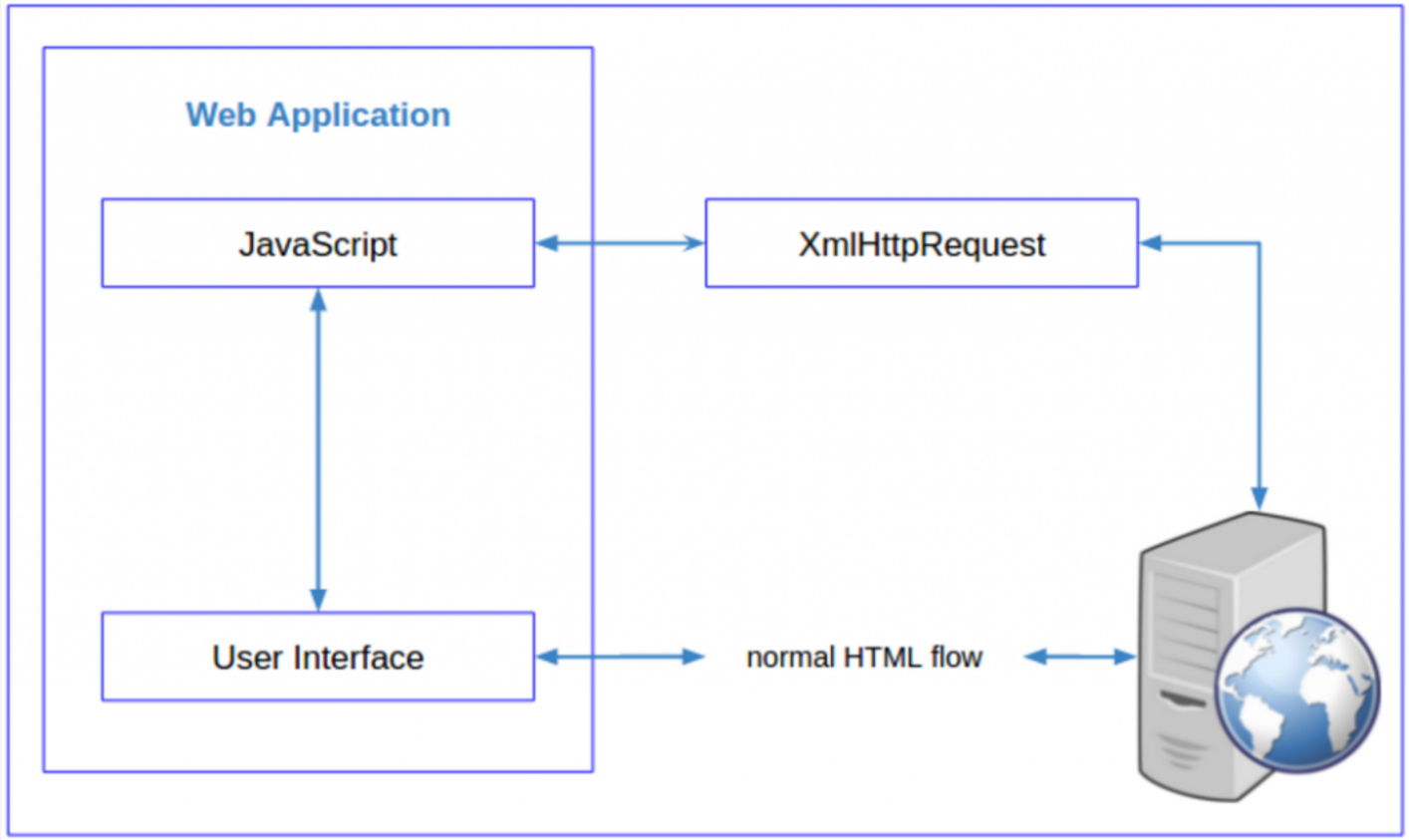
Ready State Code

0 : Request not initialized
1 : Server Connection established.
2 : Request recieved.
3 : Processing Request. (loading)
4 : Request Finished

Status Codes :

2xx : Success
3xx : Redirection
4xx : Client Error
5xx : Server Error

Request Flow



File Formats & JSON

CSV (Comma Separated Value) File.
JSON (JavaScript Object Notation)
XML (Extensible Markup Language).. and many more.

JSON stands for JavaScript Object Notation. JSON is one of the commonly used file formats for sending/receiving data. In JavaScript terms, It can be said as Nested Objects / Arrays of Objects but the only difference is it does not contain any methods.

What We're Learning Today

Here are the topics we will cover this evening:

- fetch() API
- Synchronous vs. Asynchronous Execution in JavaScript

Fetch() API

Fetch API is a promise-based method, used for requesting resources on the network. It is the modern way of interacting with the internet from the web browser.

Internally, it makes use of XMLHTTPRequest.

It is more advanced as compared to XMLHTTPRequest because the fetch() method comes with lots of features ; so it is very easier to work with.

Synchronous & Asynchronous Operation

Asynchronous or Synchronous operations are the operations that define the nature of control flow in the program. They are commonly known as threads; and their work process is called threading.

Synchronous operations in JavaScript entail having each step of an operation waits for the previous step to execute completely. This means no matter how long a previous process takes, the subsequent process won't kick off until the former is completed.

Asynchronous operations, on the other hand, deferred operations. Any process that takes a lot of time to process is usually run alongside other synchronous operations and completed in the future.


Synchronous
vs.
Asynchronous

JavaScript & Threadings

JavaScript is a single-threaded language because while running code on a single thread, it can be really easy to implement as we don’t have to deal with the complicated scenarios that arise in the multi-threaded environment like a deadlock.

Since JavaScript is a single-threaded language, it is synchronous in nature. But it can be made asynchronous, where one doesn't wait for the one call to complete instead it runs another task simultaneously.

Within JS we have a lexical environment, syntax parser, an execution context (memory heap and call stack) that is used to execute the JS code. But except these browsers also have Event Loops, Callback queue, and WebAPIs that is also used to run the JS code. Although these are not part of JS it also helps to execute the JS properly as we sometimes used the browser functions within the JS.

What We're Learning Today

Here are the topics we will cover this evening:

- Callback Hell
- Promises.
- Async-Await

ITSNPORG

Callback Hell

Multiple functions can be created independently and used as callback functions. These create multi-level functions. When this function tree created becomes too large, the code becomes incomprehensible sometimes and is not easily refactored. This is known as callback hell.

When working with large sets, this is not considered best practice. Because of this challenge, **Promises** were introduced to simplify deferred activities.

Promises

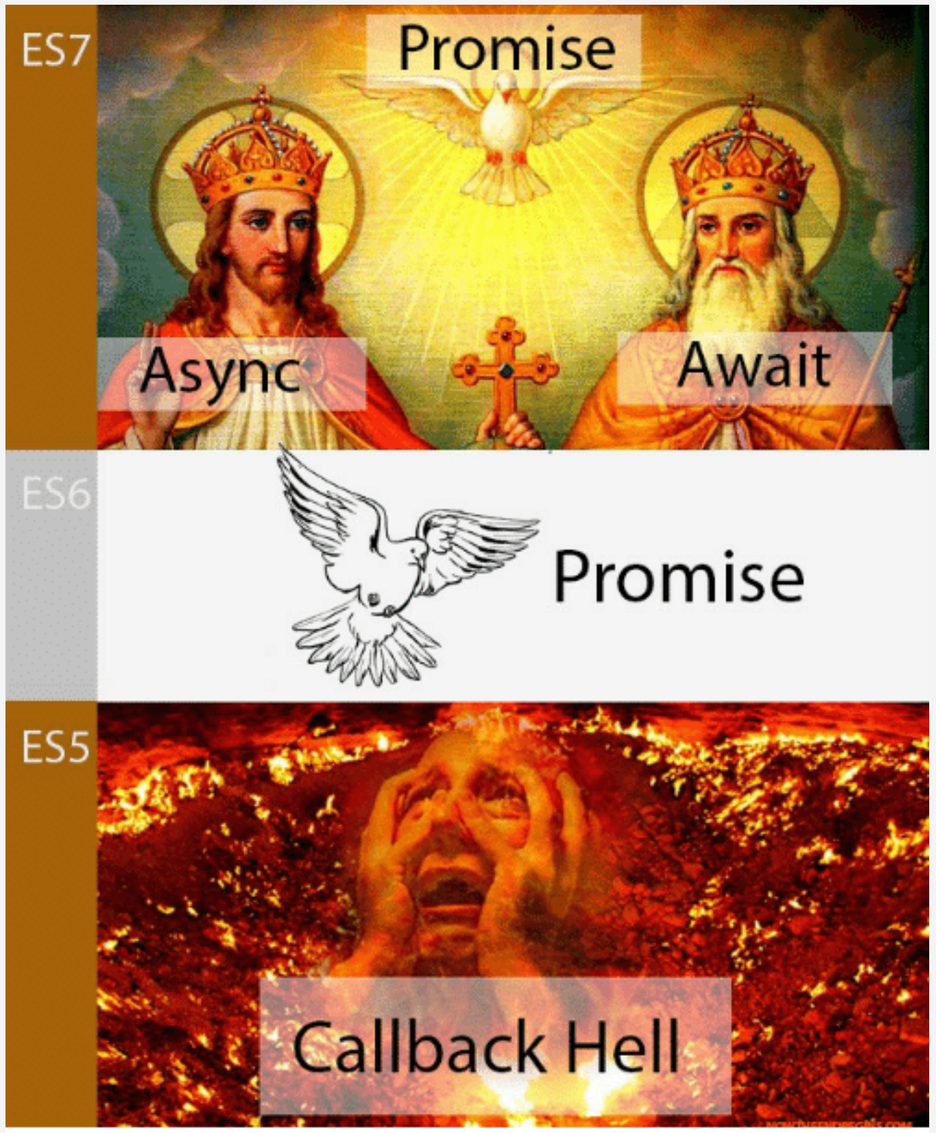
Promises are the JavaScript technique to perform asynchronous operations.

As in the real world; the concept of the promise is the same; Either they are fulfilled (resolved) or they are not fulfilled (rejected); and in between it goes through pending state.

Async Await

Just like promises, async/await are another way of doing asynchronous operations in JavaScript. Inside the hood, async/await make use of promises only. But it is quite easy to write and understand as compared to Promises.

Many programming languages (such as dart, python, kotlin), make use of the async-await syntax for implementing asynchronous operations.



Async JavaScript

ITSNPORG

Wrapping up.

Asynchronous JS

JavaScript, despite being a Single-threaded programming language; but through the use of different techniques such as callbacks, promises, async/await ; we can perform asynchronous operation in JavaScript.

What We're Learning Today

Here are the topics we will cover this evening:

- Exception Handling in JavaScript

Errors

No matter how slow or careful we program the application, we always encounter errors or bugs. The larger the LOC, the more complex and more the bugs.

The errors that we encounter can be syntax errors, semantic errors, linker errors, and run-time errors.

The errors which are expected & certain errors we can easily solve during compile time of the program. As for today's date, our Development environment is very powerful to indicate the errors in the development time.

In case of JavaScript, it is common to see errors in ajax or in asynchronous calls.

The problem.

When an unexpected error occurs in our runtime (during the program/application running time). Then it crashes our entire application. In the case of JavaScript, It will normally stop and generate an error message. So, this phenomenon does not deliver a good user experience as well as it won't make our application robust.

So, In order to handle those unexpected errors in our runtime; and to drive the program flow smoothly we make use of exception handling.

Exception Handling.

Exception handing mechanism is responsible to detect and report exceptional circumstances so that appropriate action can be taken.

Exceptional handling mechanism performs the following tasks
1. Find the problem (hit the exception)
2. Inform that error has occurred (throw the exception)
3. Receive the error information (catch the exception)
4. Take corrective actions (handle the exception)
5. Take final action after exception handling (finally)

Syntax:

```
try{

}

catch (exception e){

}

finally{

}
```

Notes:

The JavaScript statements try and catch come in pairs:

JavaScript will actually create an Error object with three properties: name, message, and stack.

We can also create our custom Exception using throw keyword.
In other languages, throw keyword has its different purpose; and in JS it has different purpose.

JavaScript InBuilt Exceptions (Errors)

IT SNP

Error Name	Description
EvalError	An error has occurred in the eval() function
RangeError	A number "out of range" has occurred
ReferenceError	An illegal reference has occurred
SyntaxError	A syntax error has occurred
TypeError	A type error has occurred
URIError	An error in encodeURI() has occurred

What We're Learning Today

Here are the topics we will cover this evening:

- Regular Expression.

The problem.

When we are dealing with extracting or validating or searching data in a string such as email, address or ip address or any data; In normal case, we always provide specific text to search for or validate for; but when it comes to similar kind of data then it will be tough for us to work with it;

So, to overcome this problem, we have regular expression; where we specify the format of data we want by writing some expression and we can find that data or validate that data or extract that data from the source.

Regular Expressions (regex)

Regular expressions are patterns used to match character combinations in strings. The patterns can be of email, phone numbers, name, password, address, and many more.

In JavaScript, validating expression can be achieved by using several JS String-related methods and methods from RegExp Object.

A regular expression can be a single character or a more complicated pattern.
Regular expressions can be used to perform all types of text search and text replace operations.

General Syntax:

/pattern/flag

The JS Way of doing...

The match() function of String :

the method searches a string for a match against a regular expression.
returns the matched data in an array.
returns null if no match is found.

The search() function of String :

the method searches a string for a value.
the method returns the index (position) of a search.
the method returns -1 if no match is found.
The search value can be a string or a regular expression.

The Terms...

Modifiers: Modifiers can be used to perform case-insensitive more global searches.

Brackets: Brackets are used to find a range of characters.

Quantifiers: Quantifiers are used in order to quantify or check the repetition of character sequences.

MetaCharacters: A metacharacter is simply an alphabetical character preceded by a backslash that acts to give the combination a special meaning.

Modifiers.

Sr.No.	Modifier & Description
1	**i** Perform case-insensitive matching.
2	**m** Specifies that if the string has newline or carriage return characters, the ^ and $ operators will now match against a newline boundary, instead of a string boundary
3	**g** Performs a global matchthat is, find all matches rather than stopping after the first match.

Brackets

Sr.No.	Expression & Description
1	**[...]** Any one character between the brackets.
2	**[^...]** Any one character not between the brackets.
3	**[0-9]** It matches any decimal digit from 0 through 9.
4	**[a-z]** It matches any character from lowercase **a** through lowercase **z**.
5	**[A-Z]** It matches any character from uppercase **A** through uppercase **Z**.
6	**[a-Z]** It matches any character from lowercase **a** through uppercase **Z**.

Quantifiers

Sr.No.	Expression & Description
1	**p+** It matches any string containing one or more p's.
2	**p*** It matches any string containing zero or more p's.
3	**p?** It matches any string containing at most one p.
4	**p{N}** It matches any string containing a sequence of **N** p's
5	**p{2,3}** It matches any string containing a sequence of two or three p's.
6	**p{2, }** It matches any string containing a sequence of at least two p's.
7	**p$** It matches any string with p at the end of it.
8	**^p** It matches any string with p at the beginning of it.

Meta Characters

Sr.No.	Character & Description
1	**.** a single character
2	**\s** a whitespace character (space, tab, newline)
3	**\S** non-whitespace character
4	**\d** a digit (0-9)
5	**\D** a non-digit
6	**\w** a word character (a-z, A-Z, 0-9, _)
7	**\W** a non-word character
8	**[\b]** a literal backspace (special case).

The JS Way of doing (2)...

The second way of doing regular expressing in JavaScript is by using RegExp Object.

The object contains two methods:

Method	Description
`exec()`	Executes a search for a match in a string. It returns an array of information or `null` on a mismatch.
`test()`	Tests for a match in a string. It returns `true` or `false`.

What We're Learning Today

Here are the topics we will cover this evening:

- Exploring some useful JS Libraries.

ITSNPORG

Need of External JS Libraries.

While creating a system, we usually don't do the buildup process from scratch. We usually take the help of external dependencies/modules or libraries; to achieve several features of the app.

So, JS libraries are a must for our project. They not only reduce our project time but also brings easiness to the entire development process.

As with other languages, JS also has a big community; and 99.99% of libraries are mostly opensource. So, there is always a library for our requirements.

Some of the Popular JS Libraries.

IT SNP

Jquery.js : To implement JS code in short-hands.
Parsley.js : To validate form datas.
Aos.js : To produce animate on scroll effect on the website.
Fullpage.js : To produce fullpage navigation on the website.
Moment.js : To update the date-time dynamically.
Passport.js : For authentication.
loadash.js : Utility library to work with numbers, arrays, objects, string efficiently and easily.
Chart.js : To implement chart and bardiagrams.
Leaflet.js : To implement maps.
Popper.js : To easily position popovers, dropdown, tooltips, and other contextual elements.

and many more..

We will be exploring...

IT SNP

=> Chart.js

=> Aos.js