Day:

Date:

Introduction to JavaScript

⟹

Date: 1-29-2024 Lecture 54 Day: Monday

Introduction to Java Script and Node JS -

Q No1 Introduction to Javasript ?

JavaScript is program language that we use front Devalpement and Backent Devalpement in web site.

What is Javasript ?

JavaScript is a programming language that is commonly used in web development. It is a client side language. which means that it is executed by the user's web browser rather than the web server. This allow Javasript to interact with the user and create dynamic, interaction web pages. Javascript is often used in combination with Html and CSS to create web page that are more interactive and engaging. It can be used to create all sorts of effects such as drop-down menus image slider and pop-up window.

Date: Day:

- JavaScript is a lightweight, OOP Language.
- It is a scripting language for web page.
- It is used to add interactivity and dynamic effect to web page.
- '.js' is the extension.
- Nowadays used in server-side develop
- Js frontend frameworks: React, Angular, Vue.
- Js Backend frameworks: Express, Node.

Javascript Can Change Html Content

One of many Javascript Html method is getElementById().

The below exemple "find" an Html element (with id="demo") and change the element content (inner HTML) to "Hello FrK";

```
document.getElementById("demo")
.innerHtml = "Hello FrK"
```

Note Javascript accept both double and single.

Date: Day:

- Javasrript can change Html Attribute value
Javascript can change the value of the (src) source attribute of an <img> tag.

- JavaScript can chang CSS style in Html.
we can chang ~~CSS to~~ Html style by useing to take the element ID.
document.getElementBy ID('demo').
style.fontSize="35px";

Note => they add Inline CSS in Html

=> Javasrript can chang the Displey property

=> Did you Know?

① Java script and Java are complety different language, both in consept and Design.

2 JavaScript was invented by Brendan Eich in 1995 and Became as ECMA standard in 1997.

③ ECMA-262 is the official name of the standard. ECMA is the ~~name~~ officiail nama of the Language.

Date: Day:

=> Where we use script tag in Html

The <script> tag in Html is used to embed or reference JavaScript code within Html document. Here are some common use cases for the <script> tag.

1 Inline JavaScript: You can include JavaScript code directly within the Html file using <Script> tag. This is done inside the <head> or <body> section of the Html document. For example:

```
<head>
<script>
    function showmassage() {
        alert ("Hellow") }
<script>
</head>
```

Note-> You can also use Inline Javascript in body.

2 External JavaScript:
You can link to an external Javescript file using the 'Src' attribute of the <script> tag

Date: Day:

This is often preferred for better code organization and reusebility for example:

```
<head>
    <script src = " script.js "></script>
</head>
```

Note = You can also add body but in last.

=> Why we use script (src in Head and also in body?

The use of <script> tag within the <head> or <body> of an Html document as well as the placement of the (src) attribute, depends on the requriment and best practice for organizing and loading your web page content. Here are some reasion for using these tag and thire placement:

1 Script in <head>

- Order of Execution: Placing the <script> tag in the <Head> section allows the java scrpt code to be loaded and executed before the rest of

Date: Day:

the Html content is parsed. This can be important in Scenar-ios where the script needs to run before the document is fully loaded.

- Dependency: If your javascript code depends on certain element or fuction defined in the <head> section. It's necessory to include the sript there.

However, one downside is that blocking script in the <head> can dalay the rendering of the page, which may result in a slower perceived page load time for the user.

2 Script in body (at the End)

- Page Loading performance: Placing the <script> tag just before the closing </body> tag in a common preatice for improving page loading performance. Thise allow the Html content to be parsed and displayed

Date: Day:

before the Javascript code is executed. User will see the page content more quickly, and script won't block the rendering of the page.

- DOM Content Loaded: Placing script at the end &body> is also a good prectice when you want to ensure that the DOM (Document Object Model) has been fully loaded before execting JavaScript. This is often done to avoid issues with trying to manipulate element that haven't been rendered yet.

Q.No2 -> What is Node JS:

Ans=> Node.js is runtime enviroment for execting JavaScript code on the server side. It allow developers to use JavaScript to write server side script, enabling the development of scalable and high-performance web application. Node.js is built on the V8 JavaScript runtime the same runtime engine the power the Google Chrome browser.

Date: _____ Day: _____

1 JavaScript on the Server:
Node.js enable the user of Javescript for server-side programing Traditionally Javascript was Primarily used for client-side scripting in web browser, but Node.js extent it use to server-side develoment.

2 NPM (Node Package manger)
NPM is the package manger for Node.js providing a vast ecosystem of open-source libraries and tool that development can use in their project.
- It simplifies the process of managing project dependencies versioning and installing thrid party Package.

CamScanner

Date: 01/30/2024 Lecture 55 Day: Tuesday

① JS-Variables ② Data types
③ Primitive Data type ④ Object
⑤ The value of null

Q No'1 what is variable in Javascript?

Ans=> Variable are Container for storing Data.

- Automatically
- using "Var"
- using "Let"
- using "Const"

In first example x, y, z are undeclared variable.
They are automatically declared when first used.

- Automatically

```
x = 5;
y = 6;
z = x+y;
```

Note= It is considered good programming precitice to always declare varibles before use.

<u>Var</u> vs <u>let</u> vs <u>const</u>

Date: Day:

In javascript there are three way to declare variable var, let and const. Each of these keyword has its own rules and uses and it is important to understand the differences between them in order to write effective and maintainable code.

Var = The var keyword is used to declare variable in javascript. It was introduced in the early day of the language and was the only way to declare variable for a long time. However, the 'var' keyword has some limitation and has been largely replaced by the 'let' and const keyword in modern javascript.

One of the main issue with 'var' is that it is function-scoped, rather than block-scoped. This means that variable declared with 'var' are accessible with the entire function in which they are declared

Date: Day:

than just Blocks of code in which they appear. This can lead to unexpected behavior and can make it difficult to reason about the scope of variable in your code.

syntax = for example:

```
var a = 15;
var b = 12;
var z = a+b;
```

Let = The 'let' keyword was introduced in ECMA6 (also known is ES6) and is used to declare variables that can be reassigned later. "Let" variable are block-scoped, which means that they are only accessible with in the block of code in which they are declared. This makes them more predictable and eaiar to reason about than "var" variables.

for example:

```
if (x > 20) {
    let y = 25;
    Console.log (y);
}
Consoleslog (y); // Reference error
```

y is not defined

Date: Day:

In this example the 'y' variable is declared with the 'let' keyword and is only accessible within the block of the if statement. If you try to access it outside of the block, you will get a "Reference Error" because 'y' is not define in that scope.

Const: The const keyword was also introduced in ES6 and is used to declare variable that cannot be reassignd latter. "const" variable are also block-scoped and behave similarly to "let" variable is that ~~respect~~ respect, the main difference is that const variable must be initialized with a value when they are declared and cannot change and reassigned later.

for example;

```
const PI = 3.12;
PI = 3.1221;
// Type Error: Assignment to constant variable.
```

Date: Day:

In this example the 'PI' varaible is declared with the "const" keyword and is assigned the value of 3.42 If you try to reassign a new value to 'PI' you will get the "Type Error" because PI is constant varaible and cannot be changed.

What is varaible naming Rule?
Javascript is a dynamicly-typed language which means that you don't have to specify the data type of a varaible when you declare it. The data type of a variabe is determine by the the value that is assigned to it

Varaible Naming Rules.

1. variable name can only ~~cat~~ contain letter, digits, underscores and dollor sign
2. Variable name cannot start with a digit.
3. Variable name are case-sensitive. It is also a good practice to use descriptive and meaningful name for your varaibles

Date: Day:

as this makes your code easier to read and understand.

Q.No2 What is Data types ?

Ans→ Data types describe the different types or kind of data that we're gonna be working with string in variable in javascript.

JavaScript has 8 Datatype
① String ② Number ③ Bigint
④ Boolean ⑤ Undefined ⑥ Null
⑦ Symbal ⑧ object.

① String : A string (or a text string) is a series of characters like "Furqan Khan" String are written with quotes. you can use single or double quotes.

```
let Name = "Furqan"
Cansolo.log (Name)
```

You can use quotes inside a string as long as they don't match the quotes surrounding the string.

Date: Day:

② Javascript Number:- All javascript number are stord as decimal number (floating point) Number can be written, or without decimals.

```
let x1 = 15; // without decimals.
let x2 = 15.00; // with decimals.
```

Exponential Notation

Extra large or extra small number can be written with scientific (exponential notation).

```
let y = 123e5 // 12300000
let y = 123e-5 // 0.00123
```

③ Javascript BigInt:

All javascript number are stored in a 64 bit ~~flto~~ floating-point format.

Javascript BigInt is a new datatype (ES2020) that can be used to stored integer that are too big to be represented by a normal javascript Number.

```
let a = BigInt("123456789");
```

Date: _____ Day: _____

④ Javascript Boolean: Boolean can only have two value: true or false.

```
let a = 5;
let b = 5;
let c = 10;
(x == y)  True
(x == z)  False.
```

⑤ Undefined: Represents a variable that has been declared but not assigned a value.

```
let a;
```

⑥ Null: Represent the intentional absence of any object value.

```
let KK = null;
```

Null types of is Object

⑦ Symbol (ES 6 and later)

Represant a unique identifier.

```
let sym = Symbol("unique Smble");
```

⑧ Object: Javascript object are written with curly braces { }. Object property are written an name: value pairs, separated by commas.

```
let person = { Name: "Khan" };
```

Date: Day:

What is type of operater?

you can use the javascript typeof operater to find the type of a javascript variable.

The ~~typeof~~ typeof operater return the type of a variable or an expression.

typeof "Furqan"; // String
typeof ~~a~~ True; // bool

What is ~~Prim~~ Primitive Data type in object.

primitive Datatype:

① Number ② String ③ Bool
④ Null ⑤ undefined ⑥ Symbol

Object Datatype:

① object ② Array ③ Data
④ Function ⑤ Regular Expression RejExp.

How many types of bracket

① [ ] Square bracket
② { } Curly bracket
③ < > angled bracket
④ ( ) Parentheses

Date: _____ Lecture 56 Day: _____

Operatore, Different types of operater, Comments, If and else if, else, condition.

Qno1 What is operater?

Ans-> Operater in javascript are symbol that perform operation on operends. Operends can be variable, value or expression. JavaScript has several types of operater and they can be categorized into different group based on their functionality.

Types of Operater

1. Arithmethic Operater
2. Assignment Operater
3. Comparison operater
4. String Operater
5. Logical Operater
6. Bitwise Operater
7. Ternary Operater
8. Type Operater

1. JavaScript Arithmetic Operater

Arithmethic operater used to perform arithmethic on number like add - Multi - sub - Divison

Date: _____ Day: _____

Operater	Description
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation
/	Division
%	Modulus (Division Reminder)
++	Increment
--	Decrement

② Java Script Assignment Operater

Assignment Operater assign value to javascript variabeaes. The Addition Assignment Operater (+=) add a value to a variable.

```
let x = 10;
x += 5;   // 15
```

Operater	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
**=	x **= y	x = x ** y

Date: _____ Day: _____

③ JavaScript Comparison Operator

In javascript, comparison operator are used to compare value and return a boolean result. They are often used in conditional ~~operator~~ statement, loops and other control flow structure. Here we discuss.

Operator	Description
==	equal to
===	equal value and Equal type
!=	not equal
!==	not Equal or not equal type
>	greater than
<	less than
>=	greater than or equal
<=	less then or equal
?	ternary operater.

```
// Equal Example
let a = 5;
let b = "5";
console.log (a == b);
// Result = True
```

Date: Day:

(4) String Operators in Javascript.

In JavaScript, there is no specific "string operator" per se but there are various operation and operator that are commonly used with string.

(i) Concatenation Operator ('+'):

The '+' operator is used for concatenating strings joing two or more string together.

```
let str1 = "Hellow";
let str2 = "world";
let result = str1 + str2;
console.log (result); // Hellow word
```

(ii) String Template Literals.

Template literals, introduced in ECMA 2015 (ES6), allow embedding expression within string literal They ar enclosed by backticks (`) and use '${}' to include variable or expressions.

```
let name = "Furqan";
let greeting = `Hello, ${name}!`;
console.log (greeting);
```

Date: _____ Day: _____

iii String Methods.

JavaScript provides various method that operators on string. Some common ones include:

- Concat() Method:

```
let a = "Hellow";
let b = " world";
let result = a.concat(b);
console.log (result).
```

- Slice () method;

Dividing in to part:

```
let fullstr = "Hellow world";
let slicestr= fullstr.slice(0,5);
Console (slicestr);//R: Hellow
```

- Substring () Method ;

```
let originalstr = "Hellow world";
let substring = original str.substing
(0,5);
console.log (substring);
// Result : Hellow.
```

- indexof () method;

```
let o.str = "Hellow world";
let index= o.str .indexof("word")
Console.log ( index) R: 6
```

Date: Day:

- replase () method:

```
let o.str = "Hellow world";
let replacestr = o.str.replace("world,
"Universe");
console.log (replace str)
// R: Hellow Universe.
```

(5) Logical Operators

Javascript includes logical operator that allow you to perform logical operation on boolean value. These operation are often used in conditional statement to make decision based on the truth or falsity of condition. Here are the main logical operation in Javascript.

1 Logical And (&&):

The Logical AND operator return true if both are true otherwise, it return false

```
let a = true;
let b = false;
let resut = a && b;
console.log (resut);
// Flase.
```

Date: _____ Day: _____

ii Logical OR (||)

The logical OR operator return true if at least one of the operation is true. otherwise false

```
let a = true;
let b = false;
let result = a || b;
console.log = (resut) // True
```

iii logical NOT (!);

The logical NOT operator negates the truth value of its operend if the operand is true. It return false and if the operend is false it return true.

```
let a = True;
let resut = !a;
consol.log (resut) false
```

Combining Logical opertor.

logical operater can be combine to create more complex condition

```
let x = 5;
let y = 10;
let result = (x<10 && y>5) || x==
5;
console.log (result) // true
```

Date: 31-1-2024 Day: Wednesday

(6) JavaScript Type Operator:

Two Type of Types Operator.

i. typeof Operator: The typeof operator in JavaScript is used to get the data type of its operand. It return a string indicating the type of the variable.

```
let x = 5;
let y = "Hello";
console.log (typeof x); // Number
console.log (typeof y); // string
```

ii. Instance Operator: The 'instance of' operator in javascript whether an object has in its prototype chain the prototype property of a constructor.

```
function Car (make, model) {
    this.make = make;
    this.model = model;
}
let myCar = new Car ('Toyota', 'Camry');
console.log (myCar instanceof Car);
output // True.
```

Date: Day:

7 Ternary Operator:

The ternary operator is a shorthand way to write an if-else statement in javascript. It takes the form of condition ? value1 : value2, where condition is a boolean expression and value1 and value2 are expression of any type. If condition is true the ternary operator return value1; if condition is false it return value2:

```
let x = 10;
let y = 20;
let max;
max = (x>y) ? x : y;
console.log (max); out = 20
```

The ternary operator can be useful and concise way to write simple if-else statement, but it can because difficult to read and understand when used for more complex statment or nested inside other expression. In these cases it may be better to use a regular if-else statment instead.

Date: Day:

Q.No2 => If else condition in javascript?

The 'If' statement in javascript is used to execute a block of code if a certain condition is met. The 'else' clause is used to execute a block of code if the condition is not met.

The condition is a boolean expression that evaluates to either true or false. If the condition is true, the code in the 'if' block is executed. If the condition is false the code in the else block is executed.

```
let x = 10;
let y = 5;
if (x > y) {
    console.log("x is greater");
} else {
    console.log("x is less");
}
```

If else ladder: The if else ladder is a control structure in javascript that allow you to execute a different block of code depending on

Date: Day:

multiple conditions. It is called a ladder because it consists of multipal if and else statment arranged in a ladder-like fashion.

```
let x = 10;
if (x > 15){
  console.log('x is greater');
} else if (x > 10){
  console.log("x is equal");
} else{
  console.log("x is less");
}
```

multipal else if you can put.

QNo3 what is comment?
single line comment is //
multiple line comment is */ */

Date: Lecture 57 Day:

For loop, for in loop, for off loop
While loop, do while loop.

Q No1 What is Loop?
Ans:- Loops are handy if you want to run the same code over and over again each time with a different value.

Different Kind of loops

For = loop throught a block of code a number of times.
For/in = loop throught the property of an object.
For/off = loops throught the value of an iterable object.
While = loops throught a block of code while a specified condition is true.
do/while = also loops throught a block of code while a specified condition is true.

For Loop = The for statment create a loop with 3 optional expression.

for (expresion 1; expresion 2; expression 3) { code }

Date: Day:

The basic for loop has the following syntax:

```
for (initialization; condition; itera
    { // Cod to be executed }
```

Here is example:

```
for (let i=0; i<5; i++) {
    console.log(i)
}
```

② For-in loop: The for in loop is used for iterating over the enumerable properties of an object. It commonly used with object, and it iterates over the key (property name) of an object.

Syntax:

```
for (variable in object) {
    // code }
```

The variable is assigned the name of each property in the object as the loop iterates over them. Here an example of for-in loop that iterates over the property of an object.

Date: Day:

```
let person = { Name : "Furqan",
               age : 21,
               Job : "developer" };
for (let key in person) {
    console.log (key + ":" + person[key]);
}
```

Keep in mind that for...in may not be suitable for iterating over arrays, as it might also iterate over inherited properties. It's generally recommented to use for...of for arrays.

3 For...of loop: The for-of loop is used for iterating over the value of an iterable object, such as arrays, string, or other iterable object.

The variable is assigned the value of each element in the object as the loop iterates over them.

```
let num = [1, 2, 3, 4, 5];
for (let number of Num) {
    console.log (number);
}
```

Date: Day:

4 While loop; While loops are a control flow structure in programming that allow you to repeat a block of code while a certain condition is true. In JavaScript the syntax:

```
while (condition){
    // code of executed }
```

The condition is checked at the begining of each iteration and if it is true, the code block is execu
If it is false the loop exits.

```
let i = 1;
while (i < 10){
    console.log (i);
    i++; }  1234 --- 10
```

It is important to include a way to updata the condition within the loop, otherwise it will ~~the loop~~ become an infini loop and never end and run foreever. In the example about the i++ statment increment the value of i by 1 at the end of each iteration which eventuall

Date: Day:

causes the condition to be false and the loop to exit.

While loop can be useful when you don't know exactly how many times you need to execute a block of code. for example you might use a while loop to keep prompting a user for input until they.

do - while loop : loop ensure that the block of code is executed at least once, regardless of whether the condition is initially true or false.

```
let i = 0;
do {
    console.log(i);
    i++
} while <i<5;
```

In this example the code inside the do block will be executed at least one time print the value.

Date: Lecture 58 Day:

Function in Javascript and Arrow function.

Q No1 What is function in javascript?

Ans→ In javascript, a function is a block of reusebee code designed to perform a specific task or a set of task. function are funda-mantal building block in java script allowing you to organize your code into modular and reusebee component.

Syntax -

```
function functionName (parameter
{ // code to be executed
  // optionally, return a value
}
```

Let's break down the component.

- FunctionName: This is the name you give to your function. It should be a valid identifier.
- Parameters: These are variable that act as placeholder for value that make up the function will recive when it is called.

Date: Day:

- Code to be execuled: This is the block of statement that makes up the function functionality.
- return statment (optional): The return statament specities the value that the function will return to the caller. A function can also return nothing (in ~~wich~~ which case it return (undefined);

example:

```
function data (Name, age) {
    console.log ("your Name is" +Name+
    "and your age is" +age+)}

data ("Furqan", 21)
```

Default Parameters: If a function is called with missing argument (less then declared), the missing value are set to undefined.
Sometime this is acceptable, but sometime it is better to assign a default value to the parameter.

```
function data (Name, age=18){
    return Name, age }
show = data ("Furqan")
Consolelog (show)
```

Date: _____ Day: _____

Q\No2 What is arrow function in javascript?

Ans=> Arrow function [() =>] is concise way of writing java script function in shorter way. Arrow function were introduct in the ES6 version. They make our code more structure and readable. Arrow function are anonymous function. function without a name but they are often assigned to any variable.

```
Const Square = (x) => {
    return x * x;
};
```

Arrow function are often used when you want to create small, one line of code function that does't require a separate function keyword. function can be defined inside other function which is known as "nesting". This is useful for creating smaller reusable block of code that can be called from within earger function.

Date: 2/2/2024 Lecture 59 Day: Friday

Quiz 9

Create a faulty calculator using javascript.

This faulty calculator does following

① It take two number as input from the user

② It perform wrong operation as follows:

+ --> -

* --> +

- --> /

/ --> **

It perform wrong operation 10% of the time.

hand- using math.random to genrate random number.

What is Js Math?

The javascript Math object allow you to perform mathematical task or numbers.

The Math object

unlike other object, the Math object has no constructor.

⇒ The math object is static.

⇒ All method and property can be used with creating Math object first

Date: Day:

Math Property (constants)
The syntax for any math property is math.property.

Javascript provides 8 mathematical constants that can be accessed as math property.

What is math.random()?
Math.random() return a random number between 0 (inclusive) and 1 (exclusive).

Math.random() always return a number lower than 1.

Javascript Random Integers.
Math.random() used with math.floor() can be used to return random integer

Note:- There is no such thing as Javascript integers.
We are talking about number with no decimal here.

// Return a random from 0 to 9.
math.floor (math.random() * 10);

Date: Lecture 60 Day:

Javascript: String =

Some string Topic we discuss on lecture No 56 like, concat, "replace", "indexof", "slice", "substring", "String Templat Literal"

Here we discuss some more Topic on string:

What is Templete strings?
Templates were introduct with ES6.
Templates are string enclosed in backticks `: HI Physyco "your" ';

```
Let a = " Hellow";
Let b = " world";
console.log(` $[a] and $[b] `);
```

What is string method.
Javascript are primitive and immutable
All string method Produces a new String without altering the original String.

① String length = let a = text.lenght
② String charAt() = let char = text.charAt(0);
This is charAt() method return the character at a specified index (Position) in a String.

Date: _____ Day: _____

Extracting string Characters

There are 4 method for extracting String characters:

- The at (Position) method
- The charAt (position) method
- The charCodeAt (Position) Method.
- using property access [ ] like in arrays.

The 4th use:

```
let a = "Furqan";
let char = a[0]; // F
```

Extracting String Parts

There are 3 method for extracting a part of a string.

1. Slice (start, end)
2. substring (start, end)
3. substr (start, lenght)

JavaScript String slice()

Slice() extract a part of a string and return the extracted part in a new string. The method take 2 parameter: Start position and End position (end not include).

```
let slice = text.slice[9, 13];
```

Date: Day:

Converting to Upper and Lower Case -

A string is converted to upper case with toUpperCase();

A string is converted to lower case with toLowerCase();

```
let text1 = text.toUpperCase();
```

JavaScript String trim();

The trim() method remove whitespace from both side:

```
let text = "  Hellow world  ";
let T1 = text.trim();
```

There are two other attribut of trim. ① trimStart() ② trimEnd.

JavaScript String Padding:

~~add~~ ECMA 2017 added two new string method to JavaScript: padStart() and padEnd() to support padding at the besining and at the end of a string.

String PadStart();

The padstart() method pad a string from the start. It pads a string with another string (multipl time) it reaches a given lenght.

```
let text = "5";
text = text.padStart(4,'0');
```

Date: Day:

It same like CSS Pseude Element : after : before.

Javescript String repeat();

=> The repeat() method return a string with a number of copies of a string.

=> The repeat() method return a new string.

=> The repeat() method does not change the origianl string.

```
let text = " Hellow " ;
let result = text.repeat(2);
```

the Internet Explorer not support.

Date: 3/2/2024 Lecture 62 Day: Saturday

Create a business name generator by combining list of adjectives and Shop name and another word.

Adjective:
1. Crazy
2. Amazing
3. Fire

Shop Name:
1. Engine
2. Foods
3. Garmants

Another word:
1. Bros
2. Limited
3. Hub

using string and function don't use Array.

Date: 04/12-2024 Lecture 63 Day: Su

Arrays, method o Arraws,
Different between function and me
What is object?

Q.No1 What is Arrays?
Ans→ An array is special variable which can hold more than one va

```
const arr = ["FK", "DK", "HK"]
```

Why we use Arrays?
If you have a list of item ca list of car name, for ~~exe~~ example, storing the cars in single variable could looks this like this:

```
let car1 = "Honda"
let car2 = "BMW"
let car3 = "volvo"
```

However, what if you want to loop through the car and find a specific one? And what if you had not 3 cars, but 300?

The solution is an Arrays?
An arrays can hold many valu under a single name, and you access the values by referring to an index number.

Date: Day:

Creating an Arrays:

using an arrays literal is the easiest way to create a javascript Array:

Syntax

```
const array-Name = [item1, item2.....]
```

It is common practice to declare array with the const keyword.

-> Note: Spaces and line break are not important. A declarction can span multiple line.

-> Note: you can also create an array and than provide the element.

Exemple.

```
const carts = [];
car1 [0] = " Saab";
car2 [2] = " valvo";
so on.
```

Using the JavaScript Keyword New

```
Const car = New array (1, 2, 3, 4)
```

There is no need to use New Array()
For simplicity, readability and execution speed use the array liteled keyword.

Date: Day:

Accessing Array Element:

you access an array element by referring to the index number.

```
const Num = [1, 0, 3, 4]
let Num = Num[2]; // Output = 3
```

Changeing an Array Element:

This statment change the value of the Array:

```
const Num = [1, 0, 3, 4];
Num[0] = 5;
```

Converting an Array to a string

The javascript method toString() converts an array to a string of (comma separated) array value.

```
const Num = ["Pa", "Kis", "tan"]
Num.toString();
Output = Pa,Kis,tan.
```

Date: Day:

Basic Arrays Method

① Array lenght ② Array tostring()
③ Array at() ④ Array join()
⑤ Array POP() ⑥ Array push()
⑦ Array shift() ⑧ Array unshift()
⑨ Array delete ⑩ Array concat()
⑪ Array copy within ⑫ Array flat()
⑬ Array Splice ⑭ Array to spliced()
⑮ Array Slice

Popping and Pushing:
when you work with array, it is easy to remove element and add new element.
Poping item. Delete last item in array and Pushing ~~Delet~~ Add an array.

```
const Num=[1, 2, 3, 5, 8]
Num.Pop();
```

Delete the last element = 8

```
Num.Push(1);
```

Add an array. Last

Array shift()
The shift() method remove the first array element and "shifs" all all other to a lower index.

```
Const Data = ["Furqan", "Khan", "obid"]
Data.shift();
```

They remove the first element

Date: Day:

Array unshift();

The unshift() method adds a new element to an array (at the begining) and "unshifts" older element;

```
Const Data = ["Furavan", "waqas", "Hanish"]
Data.unshifts("FK");
```

Array delete() -

warring? using delete leves undefined hold in the way.

use pop() and push();

```
Delete Data [0];
```

JavaScript Array Concat();

The Concat() method create a new array by merging (concatination) existing Array;

```
Const MyChildren = MyGirls.concat(MyBoy)
```

Note: The concat() method does not existing arrays. It always return a new arrays.

The Concat() method can take any number of array arguments.

Date: _____ Day: _____

Array find and Search method.

① Array indexof() ② Array last Indexof()
③ Array includes() ④ Array find()
⑤ Array findIndex() ⑥ Array find Last()

One other array method
Splicing and Slicing Array
The Splice() method adds new item to an array.
The slice() method slices out a pieces of an array.

Array Splice()
The Splice method can be used to add new item to an array.

```
const Data = ["PK" "KA" "FK"]
Data.splice (1, 0, "DK" "MK");
```

⇒ The first parameter (1) defined the position, where new element should be added (spliced in).
⇒ The second parameter (0) define How many element should be removed.
⇒ The rest of the parameter (DK, MK) define the new element to be added.
⇒ The splice() method return an deleted item

Date: 2/5/2024 Day: Monday

Array Iteration Methods:
Array iteration method operate on every arrays item.

① Array forEach() ② Array map()
③ Array flatmap() ④ Array filter()
⑤ Array reduce() ⑥ Array reduce Right()
⑦ Array every() ⑧ Array from()
⑨ Array Key() ⑩ Array entries()
⑪ Array with() ⑫ Array Spread()

Loops with array:
One of the most important data Structure in javascript is the array, which is a collection of element. When working with array it is often necessory to iterate throught each element in the array, which is where loops come in. In this blog post we will explore how to use loops with array in javascript. Javascript provides several way to iterate throught an array including the for loop for Each method and for of loop.

Date: Day:

For loops: This is the most basic way to iterate throught an array. The for loop uses a counter variable that is incremented on each iteration. For example, the following code will print out each element in the array.

```
var Arr = [1, 2, 3, 4, 5]
for (var i = 0; i < Arr.lenght; i++) {
    console.log(arr[i]);
}
```

For each methods: The method is a more concise way to iterate throught an array. The forEach method takes a callback function as its argument, which is called on each element in the array:

```
Const Arr = [1, 2, 3, 4, 5];
Arr.foreach (function (element) {
    console.log (element);
});
```

The foreach () method call a function (a callback function) once for each array element.

Date: Day:

```
const number = [45, 20, 30, 21]
let text = " ";
number.forEach(myfunction);
function myfunction(value, index, array){
    text += value;
```

Note that the function take 3 argument.

- The item value
- The item index
- The array itself

The examples aboute uses the value parameter. The example can be rewritten to:

for...of loop = This is a more recent addition to javascript, and it is the most concsise way to iterte throught an array. It allows you to iterate throught the element of an array without having to access the index and it works with any iterable object, not just array. for example the following code will print each element in the array.

Date: Day:

```
const Arr = [1, 2, 3, 4, 5];
for (var element of Arr){
    console.log(element);
}
```

It is important to note that when you are iterating throught an array using a for loop and you plan to change the array during iterating you should use a for loop with a separate counter variable.

Map, filter and Reduce:

One of the most important data structure in javascript in the array which is a collection of element when working with arrays, it is often necessory to manipulate the element in various way. where the built-in method map, filter and reduce come in.

Map => The map method is used to create a new array with the result of a callback function called on each element in the original array The callback function

Date: Day:

take one argument, the current element, and return the new value for that element. for example, the following code will return an array of each element multiplied by 2.

```
const Arr = [ 1, 2, 3, 4, 5];
var multi_Arr = Arr.map (Function(elem
{ return element * 2;
});
console.log (multi_Arr) // [2, 4, 6, 8, 1
```

Filter - The filter method is used to filter an array based on a certain condition. The filter method take a callback function as its argument which is called on each element in the array. If the function return ture, the element is included in the new filtered array for-example the following code will return an array of all even number in the original array.

```
const myAr = [ 1, 2, 3, 4, 5];
var even_Num = MyAr.Filter (Function(
element) {
    return element % 2 == 0; });
console.log (even Number);
```

Date: Day:

Reduce = The reduce method is used to reduce an array to a single value. The reduce method take a callback function as its argument, which is called on each element in the array. The callback function take two argument, the accumulator and the current element, and return the new value for the accumalter. for example The following code will return the sum of all element in the array

```
const myArray = [1, 2, 3, 4];
var sum = myArray.reduce(function(acc, cur)
{
    return acc + cur;
}, 0);
console.log(sum) = // 10
```

It important to note that these method work on the original array and doesn't change it, it return a new array.

Combining these three method can be powerfull, for example you can filter an array than map the filtered array and

Date: _____ Day: _____

Finally reduce the mapped array

In Conclusion, understanding and utilizing the map, filter and reduces method in Javascript can greatly improve you ability to manipulate array and perform complex operation on them. These method are not only more concise but also more efficient that traditional loops make your code more reudable.

What is callback function:
In Javascript, a callback function in the context of an array typically refers to a function that is passed as an argument to an array method and is executed once for each method like 'foreach, map, filter, reduce' to perform operation an array element.

```
const Array = [1, 2, 3, 4, 5]
~~my~~ Array.foreach (function(element){
    Console.log(element * 2);
});
```

Date: Day:

In this example:

- The 'forEach' method over each element in the 'Array'.
- The anoymous function (function (element) { console.log (element *2); }) is a callback function that is exected for each element.
- In this case, the callback function simply double each element and logs the result to the console.

Callback function provide a flexible way to define custom behavior for array method allowing development in function programing and asynchronous javascript.

Q.No 2=> Different Between function and method.

Ans=> A function in JavaScript is a block of code designed to perform a specific task.

- function are standalone and can be define and called independanly.
- function can be declared using the function keyword and they can accept parameter and return value.

Date: _____ Day: _____

```
function add (a, b){
    return a+b;
}
add (2, 3);
console.log (add);
```

Method: A method is a function that is associted with an object. It is essentially a property of an object that point to a function.

- when a function is a property of an object, it is referred to as a method. method are called on objecti and typically perform action related to the object.

```
const myArray [1, 2, 3, 4];
myArray. push (5);
```

In summary, the distinction between a function and a method lies in the association with an object. If a function and is a property of an object (like an array or any other object) it is commonly reffered to as a method.

Date: Day:

Q.No3 -> What is object?

Ans -> Javascript is designed on a simple object-based paradigm. An object is a collection of properties, and a property is an association between a name (or key) and a value. A Property value can be a function, in which case the property is know as a method.

Object in Javascript just as in many other programing language, can be compared to object in real life. In javascript, an object is a standalone entity, with property and type. Compare it with a cup for example:- A cup is an object with property. A cup have a color, a design, weight, a material it is made of etc. The same way javascript object can have property which define their characteristics.

```
Const obj = {"Name" : "Furqan"
             "age" : 21
             "Color" : "Black"
            }
Console.log(Obj)
```

Date: _____ Lecture 64 and Lecture 65 Day: _____

In this lectur ~~we~~ they ~~do~~ give the solution of Quiz 10 in lecture 62.

x——x——x——x——x——x

Lecture 65

Quiz NO 11 -> calculate the factorial =

write a program to calculate factorial of a number using reduce and using for loop.

6! = 6*5*4*2*1

Definition of a factorial :

The factorial of a number is the multiplication of all the number between one and the number it self.

using = Reduce one example and also using for loop

Done

Date: 6/2/2024 Lecture 66 Day: Tuesday

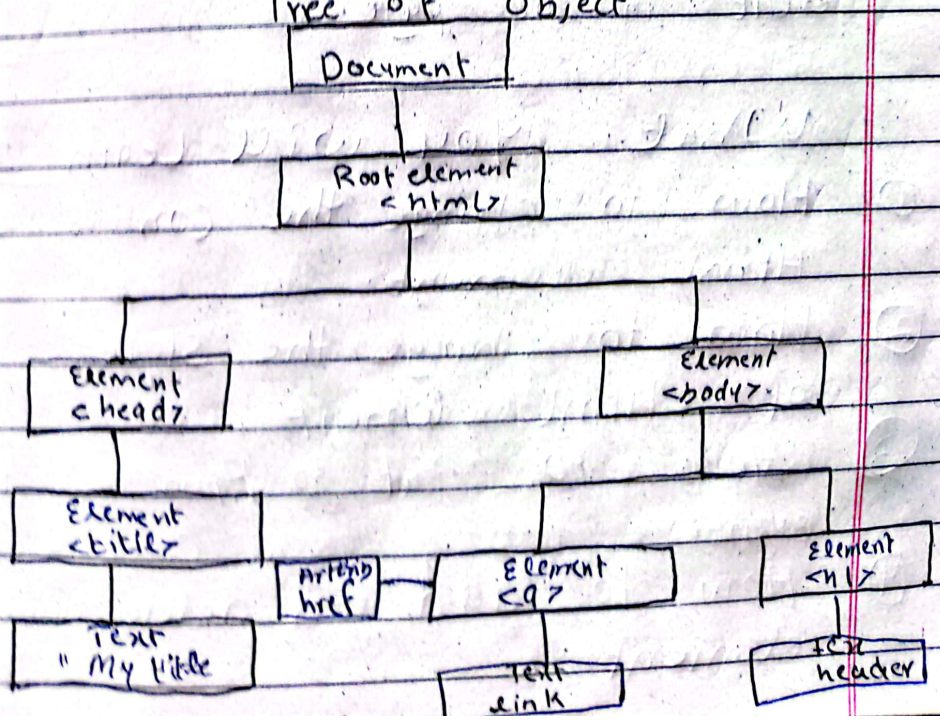
Dom , Bom , window Keyword
Javascript Core ,

Q.No What is DOM in HTML.
Ans: With the Html Dom, javascript can access and change all the elements of an Html Document.

The Html Dom (Document object model)
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.

Tree of Object



Date: Day:

With the object model, javascript get all the power it need to creat dynamic Html.

- JavaScript can change all the Html elements in the page.
- Javascript can change all the Html Attribut in the page.
- Javascript can chang all the CSS style in the page.
- Java script can removing existing Html element and altributes.
- Java Script can add new Html element and attributes.
- Java script can reat to all existing Html event in the page.
- Java Script can creat new Html event in the page.

What you will learn

1. How to change the content of Html element.
2. How to change the style(CSS) of Html element
3. How to reat to Html Dom event.
4. How to add and delete Html element

Date: Day:

What is the DOM?

- The DOM is a W3C (World wide web Consortium) standard.
- The DOM define a standard for accessing document.

Note: The W3C Document Object modal (DOM) is a platform and language neutral interface that allow program and scripts to dynamically access and update the content, structure and style of a document.

The W3C DOM standerd is seperated 3 differend part:

- Core DOM = Standard modall of all document types.
- XML DOM =
- Html DOM =

DOM The DOM in javascript is a powerful and essential concept for web developer. It serves as bridge between the static structure of Html and the dynamic behavior of javascript, enabling developer to manipulate and interact with the content of web-page in real time.

Date: Day:

Key point about the DOM

① Document Structure: The Dom reperesent a document as a hierarchial structure of object. Each Html or xml element, attribute and piece of text in the document is represented by a corresponding object in the DOM.

② Dynamic Interaction: Javascript can be used to manipulate the Dom dynamically. This means you can create, modify, or delete element and attribute in Html document using Javascript.

③ Browser Object: The Dom is often accessed through the "Document" object which is a part of the browser window object.

④ Event Handling: The DOM enables the handling of event such as user action (click, keyspace) Javascript can be used to attach event listiner

Date: Day:

```
<body>
  <h1 id="Heading">Hellow Dom</h1>
  <button onclick="changeText()">chang text
  </button>
  <script>
    function changeText() {
      docume.getElementByID("Heading)
      .textContent = " Hellow Sir";
  </script>
```

Q.N.2 => What is Bom ?

Ans => Bow stands for Browser object model. It is a programing interface provided by web browser that allow javascript code to interact with the browser window. while the DOM deal with the document structure (Html, xml) and its element, the Bom deals with the browser window itself and various browser related object.

Key Component of the BOM:

1. Window Object: The window object represant the browser window or tab. It is the

Date: Day:

top-level object in the BOM hierarchy and provides access to various property and method related to the browser window, such as dimension, location, history and, more

② Navigator object: The navigator object provide information about the browser enviroment, including the browser name, version, platform, and whether it support certain feature.

③ Screen Object:- The screen object proived information about the user's screen, such as dimension, color depth, and pixel density. It allows web page to adjust their layout or behavior based on the user screen property.

④ History Object: The ~~location~~ history object represent the browser history stack, allowing javascript code to navigate backward and forward through the user browser history.

Date: Day:

(5) Location Object:- The location object represent the URL of the current page and allowing javascript code to access and manipulate various part of the URL, such as the hostname pathname, query parameter and more.

(6) Document Object:- while technically part of the DOM, the document object is also closely related to the BOM, as it represents the Html document within the browser window. It Provides method and properties for interacting with the document structure.

(7) Timers:- The BOM provides function such as setTimeout() and setInterval() for scheduling code to execution at specified intervals or after a delay.

(8) Dialog Box:- Browser offer method like alert() confirm() and prompt() for displaying dialog box to the user from javascript code

Date: 7/2/2024 Day: wednesd

Paritical DOM

we can change the title using JS

= document.title = "Furqan"

if you want to access the Html Body in consoles

Console.log(documen.body)

if your on web browser you can select any container.
there is a $ doller sign

write = $0.backgroundColor = "green".

window = The window keyword is method to access document or body.

Date: Lecture 67 Day:

Note: Children, Parent and sibling - DOM Navigation.

Q No:- What is DOM Node?

Ans:- According to the W3C Html DOM standard, everything in the Html document is a node.

- The entire document is a document node
- Every Html element is an element node.
- The text inside Html element are text node.
- Every Html attribute is a attribute node
- All comment are comment node.

In Html DOM all nodes in the node tree can be accessed by javascript.
New node can be created and all node can be modified or deleted.

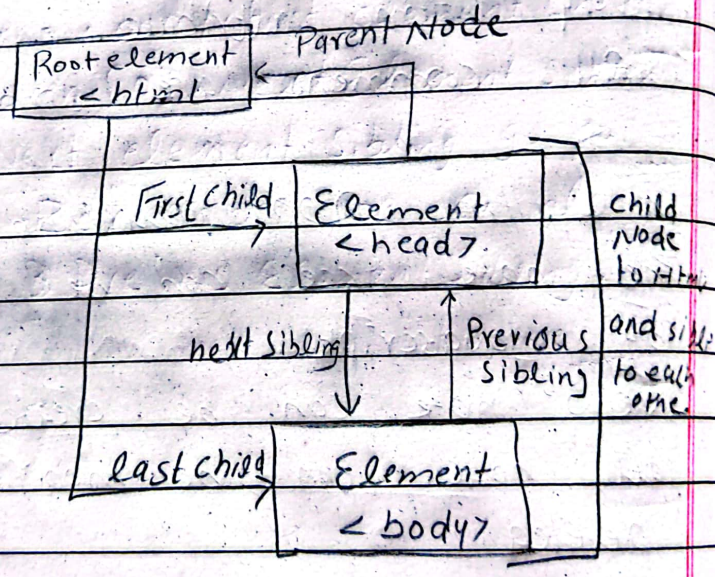
Node Relationships

The nodes in the node tree have a hierarchical relationship to each other.

Date: _____ Day: _____

The terms parent, child and Sibling are used to describe the relationship.

- In a node tree, the top node is called the root (or root node).
- Every node has exactly one perent except the root (which has no perent)
- A node can have a number of Children.
- Siblings (brother or sisters) are node with the same parent.



Navigating Between Nodes.

You can use the following node property to navigate between node with javascript.

Date: Day:

① Parent Node ② Child Node [node number]
③ first child ④ last child
⑤ next sibling ⑥ previous sibling

DOM Root Note -
There are two special property that allow access to the full document.

① Document . body = The body of all element
② document . document element = The full document

Node	Type	Example
Element Node	1	<h1 class "heg">KK</h1>
Attribute Node	2	Class = "head"
Text Node	3	Harry is
Comment Node	8	<!-- This com -->
Document Node	9	the parent of Html
Documen type Node	10	<! Doctype html>

for example - In body we have container, the inside containes is 5 boxes.
· How should we access this Box parent Node and The Box.

document . body -
used to target the Body in html.

Date: _____ Day: _____

document.body.Children:
to terget the container in body.

doument.body.first Element Child
direcly terget the contanie.

doucment.body.first Element Child
.children.
To access ~~the~~ all the boxes in container.

doucment.body.first Element Child
.children[0].Next Element Sibling
the Next element Sibling go to Next Box and we ca also use Previous Element Subling.

Date: 10/2/2024 Lecture 68 Day: Saturday

Select by ID, class, query selector, query selector all, By Name.

Q101:- How to use select by id?

Ans:- The getElementbyID() method is a part of the javascript document object and it allows developers to access a specific element on a web page by its unique ID. This method return the first element that matches the specified ID, or null element is found.

For example:- Consider the following Html Code:-

```
<div id="myDiv">This is div</div>
```

The following javascript code can be used to access the element with the ID "MyDiv" and change the text.

```
let myDiv = document.getElementBy ID
("MyDiv");
myDiv.innerHtml = "Hi Physco";
```

The getElementBy ID() method is

Date: Day:

a Convenient way to access a Specific element on a webpage, as it saves ~~det~~ develper form having to traverse the entire DOM tree to find the element. This method is especially useful when working with large and complex web page, as it allow developer to quick and easily access the element they need.

Its important to note that the Id of an element must be unique within the web page as the getElementByID() method only return the first element that match the specified ID. Also the getElementByID() method is case-sensitive, mening that "myDiv" and "myDiv" are considered to be different ID.

Date: Day:

Ques: How to use getElementByClassName

The getElementByClassName() method is a part of the Javascript document object and it allow developer to access multiple element on a web page by their class name. This method return a live Html collection of element with the given class name, or an empty Html collection if no such element are found.

For example:

```
<div class="mycass">Hellow</div>
<div class="mycass">Hellow</div>
<div class="mycass">Hellow</div>
```

The following javascript code can be used to access the elements with the class name "mycass" and change their text:

```
let element = document.getElementsByClassName
("mycass");
for (let i=0; i< element.length; i++){
    element[i].innerHtml = "Hi";
```

Date: Day:

In conclusion, the getElementBy class Name() method is a powerful tool for accessing multiple element on a web page by their class name. Understanding How to use this method is essential for creating dynamic and interactive web page, as it allow developer to quickly and easily access multiple element that share the same class name. However its important to keep in mind that the method return a live Html collection and in case-sensitive

Q-3-> ~~Wat~~ what is getElementByname

The getElementsByName() method is a part of the javascript document object and it allows developer to access multiple element on a webpage by their name attribute. This method return a live Html collection of element with the give attribute, or an empty Html collection if no such element are found.

For example - consider the following code

Date: Day:

<input type="tex" name="Na" value="" />
repeat 2 more.

The following javascript code can be used to access the element with the name attribute "Na" and change their value

```
let element = document.getElementsByName("Na")
for (let i=0; i< element.length; i++ );{
    element[i].value = " The New";
```

Q.No 4:) what is getElementByTagName?

Ans:) The getElementByTagName() method is a part of the javascript document object and it allow developer to access multiple element on a web page by their Html tag name. This method return a live Html Calelection.

```
html
<P> This is pug </P>
javascript

let element = document.getElemenByTagName("p
; element.innerHtml = "Hellow Sir";
```

Date: _____ Day: _____

Q.No5 => What is quary Slector(); The
This method return the first element within that document that matches the specifed CSS Selector (s) If no matches are found, it return 'null'.

=> quary Selector is used to select one single element form the DOM Tree

var element = document.querySelector("kk")

Q.No 6 => What is quary Selector All();
Ans => quary Selector All() = method return all the element was the document which match the specified CSS selector It return all the element that match with the selector in the form of static NodeList object which is a collection of nodes. To access each element, we usually use a loop. Each element can be accessed via an index. The index start with 0. The property length can to get the number of element that match the specified selector.

```
var x = document.quary Sellector All("kk")
for (var i=0; i< x.lenghth, i++) {
    x[i].style.color = "#60000
}
```

Lecture 70

Date: 11/2/2024 Lecture 69 . Day: Sunday

In this Lecture we Study the Solution of Lecture No 65.

x——x——x——x——x——x

Lecture 70 Quiz 12

Given 5 box, Assign a random color and a random backround to each box using DOM concepte'

Quiz Done

Date: 13/2/2024 Lecture 71 Day: Tuesday

inner Html, outer Html Attribute method in Dom, Insertion method Event.

Q.No1 what is console.log and Console.dir in Concept of DOM.

Ans:- The main difference between these two method is that they Console.log() method display to "to string" repersantation of any object passed to it. Whereas the console.dir method display an interactive list of the properties of the specifed Javascript object.

Q.No2 what is tag name and Node Name.

Tag Name:- The Tagname refere to the name of an Html element.

- forexample:- in the Html <div id="div" div" is that Tag Nam.
- It represents type or Kind of element

document.quary selector(".box").Tag Name

Node Name:-

① The node name repersent the name of a node in the DOM Tree.

②

Date: Day:

② Node is the DOM tree can be element attribute, text node etc

③ For element node, the node name is the same as the tag Name.

In summary the tag name refers specifically to the name of the Html element, while the node name is more general and refers to the name of a node in the DOM tree. which can include element, attribute, text node.

Q No3-) what is innerHTML and outer HTML.

Inner HTML -) InnerHTML is a property of DOM element that represant the Html markup contained within that element, including its descendants.

- It allow you to target Html content inside the element.
- when you set " innerHtml, it replace the existing Html content
- Getting innerHtml return the Html content as a string.

Date: Day:

- ~~Int~~ It is important to note that setting 'innerHtml' can cause the re-parsing and re-rendering of the entire subtree of the element

For example =

Html =

<div = ~~[illegible]~~ class = box>

Hi my Name is Furqan</d

Js:

<script>

document. quarySelector(.box)

.innerHtml = " Name is waqas"

we can also change the text

<u>outerHTML</u> - outerHTML is also a property of DOM element itself ~~along~~ that repersant the Html markup of the element and its contents.

- unlike innerHtml, outerHtml include the element itself along with it Html content.

- when you set outerHtml, it replace that entire element along with its content

Date: Day:

In summary the innerHtml ~~and~~ is used to get or set the Html content inside an element (div) while outerHtml is used to get or set the Html content of the element along with the element itself.

Q.No 47 What is hidden property?

Ans:- The hidden property in the Dom is a boolean attribute that is used to hide an element from view. When an element has its 'hidden' attribute set to 'ture' its is typically not displayed by browser, effectively hiding it form the user.

Here how you can the 'hidden' property

<div id = "myElement" hidden> Hello </div

In this example, the element with the ID 'myElement' has the hidden attribut set. As a result it will not be dispeay in the browser.

// To hide an element
document.getElementById('myElement').hidden = true;
// To show as element write false.

Date: _____ Day: _____

Setting the 'hidden' property to 'true' hides the element, and setting it to 'false' show the element.

Its important to note that using the 'hidden' property does not remove the element from the DOM. It simply hides it from view.

QNo 5 ⇒ What is Attribute method?

Ans ⇒ Some commen attribute method in Javascript include:

① Get Attribute(); This method retrieves the value of the specified attribute on the element. It take the name of the attribute as an argument.

document.querySelector(".box").hasattribute("style") output = false

The output show Boolen value if there is any attribute like (style):

② Set Attribute(); This method set the value of the specified

Date: Day:

attribute on the element. It takes the name of the attribute and the value to set as arguments.

document.querySelector(".box").setAttribute("style", "display: inline")

③ hasAttribute(); This method checks whether the element has the specified attrbut.

document.querySelector(".box").hasAttribute("style");

④ removeAttribute(); This method remove the specified attribute from the element.

document.querySelector(".box").RemoveAribute("style");

In summry These method provide a way to manipulate attribute of Html element dynamically using JavaScript, allowing you to change element ~~provide~~ property based on user interaction, application state or other dynamic conditions.

Date: 14/2/2024 Day: Wed

Q.No6-> What is designMode on in javascript.

Ans-> In javascript the 'design mode' property is a feature of the DOM that allows you to make the document's content editable, essentially turning the entire document into an editor.

```
document.designMode = "on";
```

Once you's set 'designMode' to on user can click on element within the document and start typing to edit the content directly. They can also use keyboard shortcuts for common editing action like copying, pasting, and formatting text.

Q.No 7-> What is insertion method?

Ans-> Insertion method refer to javascript method that allow you to insert new content into the document dynamically. These method provide way to add element, text node or Html markup into the DOM.

Date: Day:

Some common insertion method in Javascript:

① Append Child() => This method inserts a node as the last child of another node, effectively adding it to the end of the specified parent node list of children.

=> The <u>append()</u> method is used to insert one or more nodes or DOMString object as the last child of the parent element. It adds the specified content as the last child of the target element.

```
Let NewDiv= document. Create Element ("div")
NewDiv. InnerHtml = "Hellow Sir";
NewDiv. Set Attribute ("class", "div");
document. query Selector (".Container") .append(New)
```

2 ~~Qno2=>~~ ~~The~~ ~~P~~ Prepend(): The prepend() method is similar to append() ; but it insert the specified content as the first child of the Parent element:

```
document. query Selector (".container"). Prepend(div)
```

Date: 15/2/2024 Day: Thurs

③ before (); The before method insert the specified contend before the target element in the same parent node.

document.querySelector(".contanir").before(d

④ after() = The after() Method insert the specified content after the target element in the same parent Note,

(".contanier").after(div);

⑤ Replace (): The "replacewith() method replace the target element with the specified content.

(".container").Replacewith(div);

These method provid flexibility for dynamically modifying the structure and content of the DOM based on user action, application state, or other dynamic requirements. They are commonly used in web development to create interactive and responsive user interface.

Date: Day:

Q108:> what is insertAdjasentHtml();
and there types:-

Ans:> The InsertAdjasent HTML(), InsertAdjasentElement(); and insertAdjancet Text(): These method are all part of the DOM API. And they allow the you to insert new content adjacent to a specified element in various way. These method provide flexibility in dynamically modifying the structure and content of the DOM.

① insert Adjacent HTML(); This method is used to insert HTML markup as adjacent relative to the specified element.

Here an explanation of each position.

① Before begin;

- This postion refers to just before the specified element in the document structure.
- They insert Before the terget Element.

Let New = document.querySelector("container");
new. insertAdjance HTML ("beforebegin", '<P>
Hellow i am furqan </P>');

Date: Day:

② afterbegin: This position refer to just inside the specified element before its first child.

- Its the inside first child of the container.

New. InsertAdjacenceHTML("Afterbegin", <P> Hi my name is furqan </P>;

③ Afterend(); This position refers to just after the specified element in the document structure.

- It outside of the container.

("Afterend", "<P> Hi </P>");

④ beforeend; This position is last child of the target container.

- The position refore to just the specified element, after its last child.

("beforeend", ' Hi ')

Draw a grap; Html

Before begin — here we Add Beforebegan.

```
<div class="continer"> Hellow
sir
div class="box"> box1 </div>
</div>
Aterfer End
```

afterbegin

beforeend

after

Date: Day:

(2) InsertAdjacentElement();. This method insert an element as adjacent content relative to the specified element.
- It take two argument the position where the element should be inserted ('beforebegin' afterbegin' or after end beforeend').

(3) Insert Adjacent Text(). This method is used to insert text as adjacent content relative to the specified element.

Node Remove =
To remove a node, there is a methe node.remove,

document.querySelector(".box").remove();

QNo9= What is class List and class Name;
Ans=> Class Name => The class Name property of a DOM element repersents the value of the 'class' attribute for that element as a single string.
- you can ~~you~~ use it to get or set the class name of an element.

document.querySelector(".contarnir").ClassName'

Date: Day:

How many class we use in the container.

Class List:
The classList property of DOM element refure a list of class and a collection of the class Attribute of the element.

document.querySelector(".container") ~~ele~~
.classList.

Here we discuss classList Attribute

① Add: Add one or more class name to the element.
document.querySelector(".container").
classlist.Add("black").

② Remove: If you want to remove class.
document.querySelector(".container")
.classlist.Remove("Red").

③ Toggle => Toggles the presence of a class, If it exite it remove and if not exit they add the class. The second optional Parement can used to force adding and remove.

Date: Day:

document.querySelector(".Container")
.classList.toggle("Red")

(9) Contains:- Checks if the element has the specified class. It return "true" if the class exists and "false" otherwise.

document.querySelector("•Container")
•classList.contains("Block")
output = True.

Both ClassName and ClassList can be used with element having either single or multiple class. However 'classlist' privide a more convenient and powerful way to work with muliple classes especilly when you to add, remove and toggle an contains or check of the presence of specific classes.

Date: Lecture 72 Day:

Solution of lecture 70

Here we discuss How to generate random a and B.

1 t 10
a B

formaula = a + r (b - a)

r = Random number 0 and 1

① If r value = 0 then you get a and r value = 1 than you get b.

③ If r value between 1 - 0 you will get Random number that alig a and B.

a = 0
r = math. random
b = 255

Math.ceil(a + Math.random() * 255);

0 + Math.random() * 255

they generat the random number between 0 to 255.

They random method give a float value 131.55

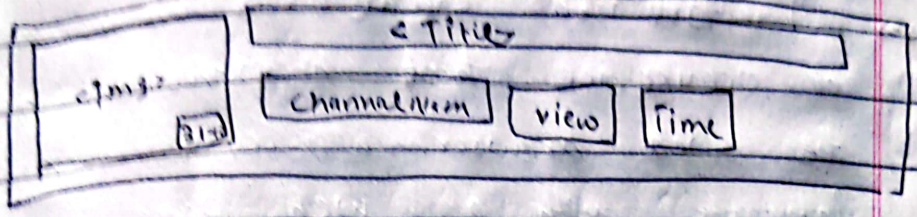
If we want to get int value math.ceil

=> Math.ceil(0 + Math.random() * 255)

Date: 2/11/2024 Lecture 73 Day: Monday

Quiz 13

I give you a card you can enter Data in Dom.



creat a function and invoke the Data.

Done

Date: 20/2/2024 Lecture 74 Day: Tuesd

browser event add event Listner
~~Time~~ Set Interval() Set Time out()
bubble probagation()

Q\No1 what is browser event?
Ans⇒ A browser event is an action or occurrence that happens within the web browser while a user interacts with a web page. These event can triggered by user action (Such as clicking a button, moving the mouse, typing on the keyboard) change in the browser's state (such as loading, resizing, or closing the window.)

Example of browser event includ-

① Click: Triggered when a user click on an element.
② Mouseover: Triggered when a mouse pointer move over an element
③ Keydown: Triggered when a Key on keyboard is pressed.
④ Load: Triggered when a web page finishes loading.
⑤ Resize: Triggered when the browser window resize.

Date: Day:

(6) Submit: Triggerd when a form is submitted.

(7) Scroll: Triggered when a user scroll the page.

(8) DomContentLoaded: Triggered when the initial HtmL document has been completely loaded and paresd.

(9) Error: Triggered when an error occure during the loading or executing of a resource such as an image or script.

(10) Focuse => Triggered when an element gains focuce when a user click on an input field.

These are just a few example of browser events, and there are many more available for developer to use when building web application. Understanding and effectively utilizing browser event is essential for creating responsive and interactive user interface.

Date: Day:

Q.No2 What is addevent Listener?

Ans=> "addevent.Listener" is a method in Javascript used to attach an event handler to an element which mens you can specify a function to be executed when a particular event occer on that element. This method provide a way to add multiple event listenner to an element without overwriting existing ones.

Here the basic syntax:-

element.addEvent Listerner(event, function, use Capture);

- element: The dom element to which you want to attach a event
- event: A string specifying the name of the event you want to listen for example ("click" dblclick "mouseover" "keydown")
- function- The function to be exectued when the specified event occur. This can be a reference to an exiting function or an anomous function.

Date: Day:

useCapture = (optional) : A boolean value indicating whether to use the capturing phase (True) or the bubbling phase (false) for handling the event. This parameter is optional and default to false (bubbling phase) if not specified.

example :-

```
<button id="myButton">click me</button>
<script>
document.getElementById("mybutton").
addeventListener("click", function(){
    alert("Button Click");
});
</script>
```

How many event we have

① click ② dblclick ③ mouseover
④ mousemove ⑤ mouse down.

explan Every Event :

① Mouse Event :

① click = fires when the mouse button is click
② dblclick = fires when the mouse button is double-click.
③ mouse down => fire when the mouse button is pressed down on an element

Date: _____ Day: _____

④ mouseup => fires when the mouse button is released.

⑤ mousemove => fires when the mouse pointer is moved.

⑥ mouseover => fires when the mouse pointer enters an element.

⑦ mouseout => fires when the mouse pointer leves an element.

② Keybored Event:

① Keydow: fire when a Keybored Key is pressed down.

② Keypress => fires when a Keybored Key is pressed (deprecated in favor of "Keydown" and "Keyup".

③ Keyup => fires when Kebord Key pres

③ Form Event =>

① submit => fire when a form is submitted.

② change => fires when the value of an input element change (on text input chackbox selection).

③ input: fire when the value of an input element changes (similar to change but fire immediately)

Date: Day:

(4) Window Event:-

(1) load => fire when the entire page (including all image and script) is load.

(2) unload => fires when the browser window is resized.

(3) scroll = fired when the user scroll the page.

(5) focus Event:-

(1) focus => fire when an element gain focus.

(2) blur = fire when an element loses focus.

(6) Miscellaeous Event:-

(1) DomContentLoaded => fire when the initial HTML document has been completely loading and paresd, without waiting for stylesheets, image, and subframe to finish loading.

(2) error => fire when an error occur during the loading or execution of an external resource (image or script).

Date: Day:

There are just some of the most commonly used event in Javascript. There are many more event aviable and you can also create custom event using the "CoustomEvent" constru Understanding and utilizing these event effectively allow you to create dynamic and interactive web application.

How to remove event?

To remove an event listener in javascript, you can use the ~~add~~ "removeEventListener()" method This method allow you to remove a previously added event listener from an element.

```
button.removeEventListener("click", handleClick);
```

What is event Object:

The event object in Javascript is an object that contain Information or Detitel about the event that has occured

Day:

Date:

such as user action or a browser action. When an event is triggered an event object is automatically created and passed as an argument to the event handler function.

The event object provider valuable about the event. Including.

① Type = The type of event that occurred (click, mouseover, Keypress)"

② Target => The element on which the event was Triggered.

③ current Target => The element that currently handly the event.

④ Event Coordinater => The coordinater of the event relative to the viewport or the target element.

⑤ modifiers => Information about modifier key (ctrl, shift, Alt) that were pressed during the event.

```
button.addEventListner("click", function(details){
consolelog(details).
});
```

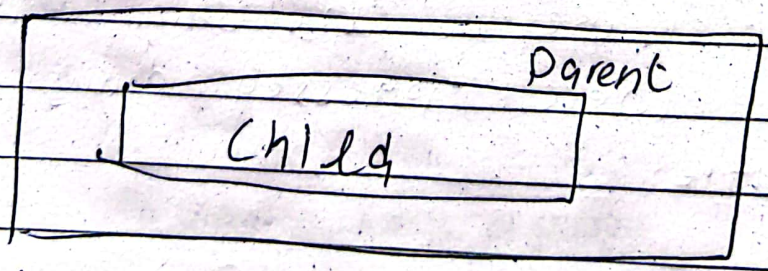
Date: Day:

Q03: What is bubble event:
If there are multiple event occur in the content Div also content Div parent. If we click only child but the parent is also occur. So we can use porpagation

for example:

```
<Script>
document.qurey selector ('.container)
.addevent Listner ("Click",(e)=>{
e.stop propagation()
alert ("Parent Div")})

document.qurey selector ("Child")
.addevent listner ("Click",(e)=>{
e.stop propagation()
alert ("Child Poragraph")
```



Now Click the child Only child occur.

Date: Day:

Ques3 -> what is setTimeout() and Set interval()?

Ans -> ① SetTimeout(): The setTimeout() Function is used to execute a function or a piece of code after a specified delay (in millseconds).

Syntax : SetTimeout (function, delay)

Parameters =

① function : The function to be executed after the delay.

② delay : The delay (in milliseconds) before executing the function.

③ Return = A unique identifier for the timeout, which can be used to cancel the timeout using "clearTimeout()"

```
SetTime out. ( function () {
Consel.log (" Hi print me after 2s")
  }, 2000);
```

② SetInterval(): The setInterval function is used to repeatedly execute a function or a piece of code at the specified interval in millisecond.

Date: _____ Day: _____

Syntax : setInterval (Function, interval)

parameter:

① Function: same as setTimeout

```
var comuter=0
var intervalID = SetInterval (Function()
{ console.log ("Interval conut "+ Comuter)
counter ++;}, 1000);
```

Date: 2/21/2024 Lecture 75 Day: Wed

JS Asynchronous, JS Callbacks
JS Promises, JS

Q1 What is Asynchronous in JS.

Ans: In javascript asynchronous programming refers to the ability to execute code out of order or concurrently, allowing tasks to be preformed in the background without blocking the execution of other code. Asynchronus operation are essential for handling tasks that may take some time to complete such as fetching data from a server reading files from disk or executing time consuming operation.

Javascript is inherently single-thre-aded, meaning it can only execute one task at a time, However, asynchronous programing enable javascript to handle multiple tasks simultaneously by leveraging non-blocking operation and event-driven architecture.

There are several mechanisms for asynchronous programming in javascript.

Date: Day:

① Callbacks => Callback function are function that are passed as argument to other function and are exected at a leter time or after a perticular event occur. They are commonly used to asycohronous operation to handle the result of the operation once its completed.

② Promises => Promises are object representing the eventual completion or failure of an asyncohronous ~~code~~ operation and its resulting value. They provide a cleaner and more readable way to handle asynchronous operation sequentially or in parallel.

③ Async/Await => is a modern syntax introduced in ECMAScript 2017 (ES8) that provides a more elegant way to write asynchronous code. It allow you to write asynochrouns code that looks synchronous, make it easier to understand and maintain. Async function return Promises implicity, and you can use the (await) keyword with in them to pause exection until a promises is resolved.

Date: 22/2/2024 Day: Thursday

Q.8 ⇒ Synchronus in JS ⇒ Synchronous means the code runs in a particular sequence of instruction given in the program. Each instruction waits for the previous instruction to complete to complete its execution.

Asynchronous in JS ⇒ Due to synchronous Programming, sometimes imp instruction get blocked due to some previous instruction, which causes a delay in the UI. Asynchronus code execution allow to execute next instruction immediately and doesit block the flow.

forexample=

Output	
One	console.log ('One');
Two	console.log ('Two');
Three	setTimeout (() => {
2s Hellow	console.log ("Hellow");
	}, 2000);
	Console.log ('Three');

Because the ansh nature in JS the can't wait 2s for three first they finshd code.

Date: 23/2/2024 Day: Friday

Qno2 What is Callback.

- A callback is a function passed as an argument to another function.
- This technique allows a function to call another function.
- A callback function can run after another function has finished.

A callback function, in programing is a function that is passed as an argument to another function and is expected to be called (or "call back") at a leter time. Callback function are commonaly used in asynchrouns program, event handling and function programming pardigm.

for example:

```
// using a callback function
function fetchData (callback) {
// Simulating as asynchronous operation
(fetching data from a server)

SetTimeout (function () {
Const data = 'Some fetched data';
callback (data);
}, 2000); }
// calling back the provided function
with the fetched data
```

Day:

Date:

```
// Callback function to handle the fetched data
function handleData (data) {
  console.log ('Data received:', data);
}
// Invoking the fetchData function with the handleData callback function.
fetchData ( handleData);
```

In this example:-

- fetchData : is a function that simulates fetching data asynchronously
- It take a callback function ('callback') as an argument.
- Inside 'fetchData' after a simulated delay.

This pattern is commonly used in scenarios where you want to execute code after as asynchronous operation complete, like fetching data from a server handling user input, or responsive to event in a web application

Date: Day:

Callback Hell => Callback hell also known as "Pyramid of doom" or is a situation that arises in asynchrouns javascript programming when multiple nested callback function are used, leading to code that becomes difficault to read, understand, and maintain. This often occure when dealing with several layers of asynchronous operation such as multiple database queries, file system operation, or network requests.

```
asyfunc1(function(result1)){
  asyfunc2(result1, function(result2)){
    asyfunc3(result2, function(result3)){
      //and so no
```

In this example, each callback function is nested within another callback function, makeing the code deeply indented and hard to follow.

Example 2:)

```
function getData(dataID, getNextDa
{ setTimeout(() => {
    Console.log("data", dataID);
    if (getNextData) {
      getNextData();
    },
} 2000);
```

Date:

Day:

```
getData (1, () => {
    getData(2, () => {
        getData(3, () => {
            getData(4, () => {
            });
        });
    });
});
```

This types of code is very Dificalty to read Developer thatway this callback hell and nested callback

Q No3 => What is Promises =>

Ans => • Promise is for "eventual" completion of task. It is an object in js. It is a solution to callback hell.

• Promises in Javascript are object that represent the eventual completion or failure of an asynchrouns operation. Then allow you to handly asyschrouns operation in more organized and manageable way compared to using callback directly.

```
let promI = new promise((resolve, reject) => { - - })
```

Function with Two handler

Date: _____ Day: _____

A Promises can be one of Three states:

① Pending: Initial State, neither fulfilled neither Reject.

② Fulfilled = (Resolved) The asynchorous operation has complete successfull and the promises has a value.

③ Reject => The asynchorous operation has faild, and the promises has a reason (typically an error).

The promises object support two properties: state and Result.

① While a promise object is "pending" (working) the result is undefined.

② when a Promise object is Fulfill the result is value.

③ when a promise object is Reject the result is an error object

Date: Day:

.than() and .catch()

① .than(); The .than() method is used to handle the fulfillment of a Promise. It takes one or two callback function as argument.

- The first

② .catch() The .catch() method is used to handle the rejection of a promise.

For example =>

```
let prom1 = New Promise((resolve, reject) => {
let math = Math.Random();
if (math < 0.5) {
  setTimeout(() => {
    consale.log("I am Accept")
    resolve();
  }, 2000);
} else {
  setTimeout(() => {
    console.log("I am Reject")
    reject("sorry")
  }, 2000);
}});
prom1.than(() => {
Console.log("this is solve")});
Prom1.catch((err) => {
  console.log(err)})
```

Date: _____ Day: _____

What is Promise Chain

A promise chain refer to a series of asynchronous operation that are executed sequentially by chaining multiple .than() method. This chaining allow you to perform a sequence of asynchonous task in a more organized and readable manner, avoiding the callback hell problem that can occur with deeply nested callbacks.

example of chain

```
function Async() {
  return new Promise((resolve, reject)=>{
    setimmout(() => {
      Console.log("this is first")
        resolev("first done")
    }, 2000);
// copey this code Two time more
Promise().then((res) => {
  return promise2(res)
}).than((res) => {
  return promise3(res)
});
```

Date: 2/24/2024 Lecture 76 Day: Saturday

Async function, Await, fetch API

Async function and the 'await' keyword are feature introduced in ECMA Script 2017 (ES8) to simplify asynchronous code in Jave Script.

Q.No1 What is Async function?

An async function is function that operater asynchrouls via the event loop, always return a promise. Within as async function, you can use the "await" keyword to pause the execution of the function until a Promise is resolved or rejected. This makes it easier to write and maintain asynchro nous code in a more synhronouse looking style.

```
async function fetchData() {
Const response = await fetch("http://
dummyjson.com/products/1")
Const data = await responce.json();
return data;
}
```

Date: Day:

In this examples 'fetchData" is as async function. Inside it, the 'await' keyword is used to pause the execution until the 'fetch' API call return a Promise with the HTTP response. Then, another 'await' keyword is used to pause the execution until the response is converted to JSON format using the .json() method finally the function return the parsed JSON data.

Note: async function always return a promise.

Q.No2) What is await Keyword.

The await keyword is used inside as async function to pause or wait the execution of the function until a Promise is settled (either resolved or reject) It can only be used inside asyncfunction. Here where we use await keyword Inside as Async function:

1. To use 'await' you need to declare a function as 'async'.
2. Inside the async function, you can prefix any Promise-based operation with the await keyword to pause execution until that promise settle.

Day:

Date:

for example

```
asyne function fetchData() {
const Response = await fetch('URL')
const Data = await response.json();
return data;
}

fetchData()
.than ( Data => {
        Console.log ('Data', data)
    })
.catch ( error => {
        Conso le.log ('Error', error);
    });
```

In this example:

(1) fetchData is an asyne function that fetches data from an API

(2) The 'await' Keyword is used before calling .fetch() and response.json() causing the execution of the Data or "fetchData" function to pause until the respective Promises are settled.

(3) This make the asynchrous code appear more synchronous and easir to read.

Date: Day:

Q\\03 ⇒ What is fetch API?

Ans ⇒ The fetch API is a modern javascript interface for making network request (HTTP request) in the browser environment. It provide a more powerful and flexible alternative to the older XMLHTTP Request.

The fetch API is feature that allow you to make HTTP requests (such as GET POST, PUT or Delete) to a web server. It's bulit into modren browser so you don't need additional libraries or packages to use it.

The fetch API offers a simple and straightforword way to fetch resources (such as JSON data HTTP page, image, etc) from a server or another location on the web. It uses promises, allowing you to write asynchronous code in more readable and mageable manner.

Date: Day:

Key feature of the fetch API,

① Promise-based: The fetch API returns Promises, making it easier to handle asyncronous operation.

② Build-in error handling => fetch API automatically reject the promise if the request fail (due to network issue, server error) allowing you to handly error using standard promise error handling machanisums.

```
fetch ("url")
  .than (responce => {
  });
  .catch (err => {
  })
```

Note=> The fetch API Provide an Interface for fetching (sanding/receiving resourese.

① I use Request and Response

- The fetch() method is used to fetch a resource (data)

```
let Promice=fetch(url.[option])
```

Lecture 78

Date: 25/2/2024 Lecture 77 Day: Sunday

In this lecture we will the soulation of Quiz 13
How to make Dynamic card.

Quiz 14 Lecture 78

Create a hacking simulator which has green backround and it show these massage with random delay of 1 to 7 second

1. Initializing Hacking...
2. Reading your files...
3. Password file Detected...
4. Sending all password and
5. personal files to server...
6. Cleaning up

Note The three dots must blink so that it look like a real terminal.

Done

Date: 26/2/2024 Lecture 2789 Day: Monday

try, catch, and finally, Error handling:

Q No 1 => What is try catch and finally.

Ans => ① try => The code within this block is executed first, Ist the part where you anticapate that error might occur.

. The code in the try block is executed first, and if they throws error, Than exeuted the code in catch Block.

② Catch => if an error occur within the try Block, control is transferred to the catch block. Here you can handle the error, log it. or perform any necessory action to recover from it. The 'catch' block take an error object as a parameter which contain information about the error.

③ finally => This block is optional It executed regardless of whether an error occurred in the try block or not. It commonly used for cleanup operation, such [illegible] file, releasing the state

Date: Day:

For example:

```
let a = prompt("Enter first number")
let B = prompt("Enter second number")
let sum = parseInt(a) + parseInt(B)

try {
    console.log("sum is", sum + x)
} catch (error) {
    console.log("x not define")
}
finally {
    console.log('Please retry')
}
```

What is parseInt: if you want to convert string into integer.

Why we use finally:
the finally block help us in function in function we use Return. The down code canot Run. but the finall is always Run.

Why we use try and catch.
1) Error handling 2) Preventing program crashes 3) Grace full failer. 4) Debugging and logging.

Day:

Date:

Q2- What is throw error

Ans- In javascript a throw error refer to an error that is deliberately generated and thrown by the program to indicate that something unexpected or erroneous has occurred during it execution. These error are typically created using the 'throw' statment and they can be of various types such as 'Error', 'SyntaxError' 'TypeError' 'ReferenceError etc.

Throw error are essentile for error handling in javascript as they allow developer to gracefully handle unexepeted situation provide meaningful feedback to users or log detailed error information for debugging purposes.

```
let a = 5
let b = 10
if (a < 10){
throw SyntaxError("a is less than
5")}
console.log(a+b).
```

Date: Lecture 80 Day:

Class, Object, Constructor, this Keyword, Super, Instance of Extend Keword. Setter, Getter Method overriding.

Q1: What is OOps?

Ans:- OOP in javascript allow developers to model real-word entites as object and organize code into reuseble and maintainable components. Here an explanation of key OOP Concepts in Javascript.

What is class.

A class is a blueprint for creating object with shared property and method. In Javascript, classes were introduce in ECMAscript 2015. They provide a more intuitive syntax for creating object and define their behavior compared to using prototypes directly.

Classes provide a way to model real-world entities and concept in code, allowing for code organization, reuse and abstraction.

Date: Day:

What is Object?

- A object is a instance of a class. It represents a specific occurrence or relization of the class.
- object have properties that store data associated with them and method that define their behavior.
- Each object created from a class has its own set of property value, independent of other object created from the same class.

for example

```
Class Person {
        Constructor ( name, place) {
        this.name = name
        this.place = place.
        }
        big () {
console.log (" My Name is ${this.name} and
        my village is ${this.villag}")
        }
}
let person1 = new person ("Furqan"
                "Ampiri Kall ")
console.log (person1)
data1.big()
```

Date: 27/2/2024 Day: Tuesday

What is prototype in OOP:
In Javascript each object has a hidden internal property called [[Prototype]] (also referred to as '__Proto__' in some context) that point to another object known as its prototype. When you try to access a property or method on an object. Javascript looks for that property or method first in the object itself. If it's not found, Javascript continues to search for it in the prototype of the object, forming a chain of prototype knows is as the Prototype chain.

```
Let obj1 = {
    'Name' : "Furqan"
}
Let x = {
    'name2' : " Waqas"
}
obj1.__Proto__ = x;
console.log (obj1.name2)
```

Date: Day:

What is Constructor?

The constructor is a special method inside a class that gets called when an object is instantiated. It is used to initialize object properties or perform any setup that need to be done when creating an instance of the class.

For example,

```
class modi {
    constructor (Name, ID)
        This.Name = Name
        This.ID = ID
    }
```

What is extend Keyword?

In Javascript the 'extend' keyword is used to create a subclass (a derived class) that inherits property and method from a superclass (a base class). This concept is known as inheritance and it allows for code reuse and creating of hierarchical relationship between classes.

Date: Day:

Here how the extends keyword work.

① Base class (SuperClass) you define a base class the 'class' keyword This class server as the blueprint for creating object and can have property and method.

② Derived class (Subclass)
you define a subclass useing the 'class' keyword along with the extent keyword, following by the name of the superclass. The subclass inherits properties and method from the superclass.

③ Constructor in Subclass: If the subclass has its own constructor it must call 'super()' within its constructor to call the consturce of the superclass. This ensure that the superclass's constructor is executed before the subclass constructor.

Day:

Date:

Accessing Superclass Method =>

Inside the subclass you can access method of the superclass using the super keyword.

for example =>

```
class perent {
constructor (name, id) {
    this.name = name
    this.ID = ID
}
Data() {

    console.log ("my name is {this..
    and The ID is {This.ID}")
}
class Child Extent Perent {
    Constructor ('Name, Id, Place
    {   super (Name, Id)
        this.Place = place }

Data () {
    super. Data()
    console.log ("The place is
        {This.place}")
}
let F1 = new child ('Furqan', 15/12/2019
    2070", Ampiri Kalla")
F1. run();
```

Date: _____ Day: _____

What is super() Keyword.

In Javascript the super() Keyword is used to call method on the parent class (superclass) from within the child class. (Subclass). It's most commonly used within subclass constructor to call the constructor of the superclass and pass argument to it. This ensures proper initialization of property inherited from the superclass.

Method overding => Method overriding is a concept in OOP where a subclass provides a specific implemention of a method that is already define in its superclass. This allow the subclass to customize or extend the behavior of the inherited method without modifing the superclass implemeting directly.

Day:

Date:

What is static Method.

A static method in javascript is a method that belongs to the class itself rather than to instance of the class. Static method are often used for utility function or for operation that don't require access to instance-specific data.

For example

```
class Mathhelper {
    static Square (number) {
        return number * number;
    }

    Static cube (number) {
        return number * number * number
    }
}

console.log (Mathhelper.square (5));25
console.log (Mathhelper.cube (4));
```

In this example;

① square () and cube() are static method define within the Mathhelper class.

② These method can be called directly on the class Mathhelper without to need to create an instance of Mathhelper.

③ The static method square() and cube() perform mathcut operation and return the result.

Date: Day:

What is instance of keyword?

The instance of operation in javascript is used to determine whether an object is an instance of a particular class or constructor function. It return 'true' if the object is an instance of the specified class or constructor function other wise it return false

object instance of constructor

- object => The object whose type is to be checked.

Constructor => The constructor function or class to test against.

eg: (animal instance of Animal):

Date: 2/28/2024 Lecture 81 Day: Wed Lecture 82

The Quiz 74 in Lecture 78 Lecture 81 is Ans of.

x——x——x——x——x

Lecture 82

IIFE, Destructuring, Spread Syntax, Local global and Block Scoper, Hoisting.

Q What is IIFE ?

IIFE stand for Immediately Invoked function Expression. It is JavaScript design pattern that allow you to execute a function immediately after its define. This pattern is often used to create a private scope for variable and avoid polluting the global namespace.

For example =>

```
(function () {
    var greeting = 'Hellow';
    var name = 'John';
    console.log (greeting + name);
})();
```

One of the main benefite of using an IIFE is to encapsulate variable within the function scope. Prevent them conflicting with variable in the local scope or varible in other function

Date: Day:

What is Destructuring and when we use it?

Destructuring in Javascript is a feature that allow you to extract value from arrays, object or nested object and assign them to variable in a more concise and intuitive way. It provide a syntax that mirrors the structure of the data you're working with.

For array

```
let arr = [1,2,]
Let [a,b] = arr
console.log(a,b) //output = 1,2
```

For object

```
const {x,y} = {1,2};
```

If we have multiple value so what can we do.

For example

```
Let arr= [1, 2, 3, 4, 5, 6]
Let [a, b, ...Rest] = arr
Console.log(a, b, Rest)
```

Destructuring can make you code cleaner, more readable and less verbose, especially when working with complex data structure.

Date: Day:

What is spread syntax ?

The spead syntax in Javascript allow an iterable (like an array or a string) to be expanded in place where zero or more arguement (for function call) or element (for array leterals) are expected. It provides a consice way to spread the element of an iterable or object into another array or object.

For example:

```
Function sum (a, b, c) {
    return a + b + c
}

const arr = [1, 8, 10]
console.log (sum(...Arr))
```

= output = 19

① It allow you add multiple array into New array.

Sperad syntax provide a convenient and concise way to works with array, object and function argument.

Date: Day:

What is local scope block scope and function scope.

In Javascript scope refer to the accessibility or visibility of varible and function with in code. These are several type of scope including local scope, block scope function scope.

1 Local Scope =>

- Local scope refers to the innermost scope, typically within a function.
- Varibles declared inside a function are only accessible within that function are are not visible outside of it.
- This ensure encapsulation and prevent conflict between variable with the same name in different function.

Example =>

```
function myfun() {
var x = 10 // variable x in Local scope
Console.log (x)
}

my fun()
Console.log(x); // Error x is
not define.
```

Date:

Day:

② Block Scope ⇒

- Block scope refer to the area with in if statment, loops or any other block enclosed by curly bracas { };
- variable declared using 'let' and 'Const' are block are scope meaning they are accessible only within the block in which they are define.
- unlike var, which is function-scoped Let and Const have a scope that is limited to the block in which they are declered.

For example ⇒

```
if ( true) {
    Let y = 20; //variable y block sc
    Console.log (y);
}
Console.log (y) // Error: y is not
    define.
```

③ Function Scope ⇒

- Function scope refer to variables define within a function using the var keyword

Date: Day:

- variable declared with var have a scope limited to the function in which they are declared.
- These variable are accessible any where within the function but not outside of it.

For example=>

```
function myfun() {
  if (true) {
    var z = 20; // z is function scope
  }
  Console.log (z) //
}

myfun()
Console.log (z) // Error z is not define
```

understanding scope in javascript is crucial for writting maintainable and bug-free code, as it help in managing variable life time and avoiding unintended variable across or modification.

Date: Day:

What is hosting in Javascript?

hosting is a behavior where variable and function declaration are moved to the top of their containing scope during the compilation phase before the code is executed. This means that regradless of where variable and function are declared within a scope, they are effectively "hosteing" to the top of the scope.

Note => Its important to node that only the declaration are hosted, not the initialization or assignement.

```
Console.log(x) // undefined
var x = 5
```

This code interpreted like this

```
var x;
console.log(x)
x = 5;
```

Hositing Let and Const= with Let and Con't hositing is different

```
Console.log(x) // Error if let or const
Let x = 5; // with var undefine
                print
```

function expressing and class expression are not hosted.

Date: 2/3/2024 Lecture 83 Day: Saturday

In this Lecture we discuss some Quiz or Exercise. The exercise on harry notes.

① Guss the number Done

② Snake Water Gun Done

③ Tell me a joke

Quiz 3: have some methed that i canit undersatend.

① fetch API

understanding Terms:

① AJAX = stand for = Asynchronous JS and XML

② JSON - is Javascript object Notation AJAJ

③ JSON() => method return a secound promise that resolves with the result of parsing the responce body text as Json

Date: 3/3/2024 Day: Sunday

What is Datae in time in javascript:

Make a digital clock

Handling dates and times is a common requirement in many JavaScript application, and javascript provide a built-in Date object to work with date in time.

Creating Data object

you can create a Date object in JavaScript in JavaScript using of following method.

① Current Date and Time

```
const currentDate = new Date();
```

② Specific Date an Time

To find specific Date an Time

```
const SpecificDate = new Data ('2024-02-28T
                    :00:00');
```

output => wed feb 28 2024 12:00:00

③ Specifying year, Month, Day, Houre, Min, Se

```
const specifDateTime = new Date(
2024, 1, 28, 12, 0, 0);
```

Date: Day:

Getting Components of a Date:

you can get various components of a date using method like 'getFullYear'

① getFullYear() ② getMonth()
③ getDate() ④ get Houres()
⑤ getMinutes() ⑥ get milliseconds()

```
const date = new Date()
const year = date.getFullYear()
const month = date.getMonth()
const day = date.getDate()
const hour = date.getHours()
const minutes = date.getMinutes()
const second = date.getMinutes()
```

Manipulating Dates

you can manipulating date using method like

① setFullYear() ② setMonth()
③ setDate() ④ setHours()
⑤ setMintes() ⑥ setSecound()

```
const date = New Date()
date.setFullYear(2025);
date.setMonth(4);
date.setDate(15);
```

Day:

Date:

Comparing Dates

you can compare date using comparision operators like < <= > >=

```
const date1 = Newdate ('2012-02-28')
const date2 = Newdate ('2022-02-28)
if (date1 < date2) {
    Console.log (" Hello I big)
}
else {
    Console.log ("I Small")
```

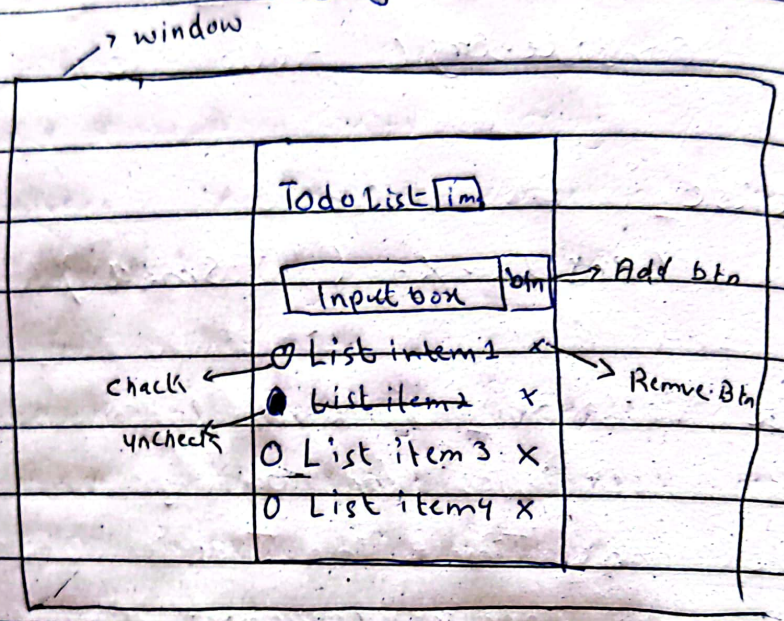
These are the fundamental concepts and techniques for working with date and Times in Javascript. Depending on your specfic requriment, you may need to use additional libraries or techniques for more advanced date time manipulation.

Date: 5/3/2024 Day: Tue

Question: To do list

create a todo list app capable of storing your TOdo in local storage. Add an option to create delect and access delete and access an the TOdo list.

Design



What is Local storage:

Local storage is a web storage mechanaism that allow web application to store date locally within the user's browser. It provides a way to persistently store key-value pair in web browser. Local storage is a part of the web API along with session storage

Date: 6/3/2024 Day: Wed

here How you can use local storage :

① Setting Items => you can store data in the local storage using the setItem() method for example =>

```
localStorage.SetItem("usse"; "Furqan")
```

② Getting Item => you can retrieve data from Local storage using the getItem() method. for example =>

```
let user = Local storage.getItem("user")
console.log(username);
```

③ Remove Item: you can remove item from local storage using the removeItem() method. for example:

```
local storage.RemoveItem("user")
```

④ Clearing Local Storge => you can clear all item from local storage using the clear() method for example:

```
localstorage.clear().
```

Date:

Day:

Local storage is commonly used in web application for various purposes such as:

User Preference: Storing user preference like theme setting, language perference etc.

User Authentication => Storing authentication tokens or user credentials for persistet login sessions.

- Caching => Storing frequently accesed data to reduce server request and improve performance.

Offline Data => Storing data locally for offline access in progressive web application (PWA).

Day:

Date:

Exerice 7 :>

create a javascript program capable of generationg a password which contain atlest one lowercase and one uppercase and one special characters.

create a password class to acheive the same.

for example :

var alarmTime = document.getElementsByID("",
.value;

Question= why we use .value and where we use it.

In javascript the .value property is commonly used to retrive the current value of an Html form element, such as <input> <textarea>, <select> etc.

When you use document.getElementByID ('element ID') to select an element from the (DOM) it return a

Date: 7/03/2024 Lecture 84 Day: Thursday

Porject : Spotify clone using Html, CSS and javascript.

Date 19/03/2024

In the previse two day I have Done the UI Design. To day I start the Javascirp.

What is audio and there element:

In Javascript, the Audio object represents an audio element in Html document. It provides a way to play audio file programmatically. you can create a new instance of the Audio object by providing the URL of the audio file you want to play.

```
// Create as Audio object
var audio = new Audio ('audio.mp3');
// Play the Audio
audio.play();
// Puse the Audio
audio.pause()
// set the valume (0.0 to 1.0)
```

Day:

Date:

// Get the duration of the audio in second.
var duration = audio.duration;

// Get or set the current playback position (in second)
var currentTime = audio.currentTime
audio.currentTime = 30;

The audio object has several attribute that you can use to control the playback and get information about the audio file.

Day:

Date:

END

Java script

CamScanner