https://shorturl.at/kAHNV

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Population of JSON data in table**

**Promises**

**Async await**

**Handling HTML elements**

**Events**

**Callbacks**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

===============================================

Populate JSON data in HTML table

===============================================

let products = [

{ "p\_id": 111, "p\_name": "P\_one", "p\_cost": 10000 },

{ "p\_id": 222, "p\_name": "P\_two", "p\_cost": 20000 },

{ "p\_id": 333, "p\_name": "P\_three", "p\_cost": 30000 },

{ "p\_id": 444, "p\_name": "P\_four", "p\_cost": 40000 },

{ "p\_id": 555, "p\_name": "P\_five", "p\_cost": 50000 }

]

products.forEach((element, index)=>{})

document.write(`

<table border = 1px

cellpadding = 10px

cellspacing = 10px

align = "center">

<thead>

<tr style = "background-color:lightgray">

<th>Sr no</th>

<th>P\_id</th>

<th>P\_name</th>

<th>P\_cost</th>

</tr>

</thead>

<tbody>

`)

products.forEach((element, index) => {

document.write(`

<tr>

<td>${index + 1}</td>

<td>${element.p\_id}</td>

<td>${element.p\_name}</td>

<td>${element.p\_cost}</td>

</tr>

`)

})

document.write(`

</tbody>

</table>

`)

let x = ''

x = x + `<table border = 1px

cellpadding = 10px

cellspacing = 10px

align = "center">

<thead>

<tr style = "background-color:lightgray">

<th>Sr no</th>

<th>P\_id</th>

<th>P\_name</th>

<th>P\_cost</th>

</tr>

</thead>

<tbody>

`

for(let i = 0; i<products.length;i++){

x = x + `

<tr>

<td>${i+1} </td>

<td>${products[i].p\_id} </td>

<td>${products[i].p\_name} </td>

<td>${products[i].p\_cost} </td>

</tr>`

}

x = x + `</tbody>

</table>

`

console.log(x)

document.getElementById('mytbl').innerHTML = x

Assignment

Populate JSON in HTML table

URLs

1. https://restcountries.com/v2/all

any 4 keys

2. https://www.w3schools.com/angular/customers.php

all

for hint

http://jsonviewer.stack.hu/

===============================================

Promises

===============================================

- Promises are used to handle asynchronous operations in Javascript.

- Promises are special Javascript objects.

- Promises are having two states

- Success (resolve)

- Failure (reject)

- Promises can be created using 'Promise' class.

- 'Promise' is the predefined class in Javascript.

- Promises can be consumed using 'then()'.

//Eg01

//Create a promise

let myPromise = new Promise((resolve, reject)=>{

resolve('Tomorrow i will be at home')

})

//consume promise

myPromise.then((posRes)=>{

console.log(posRes)

},(errRes)=>{

console.log(errRes)

})

//Eg02

let myPromise = new Promise((resolve, reject)=>{

setTimeout(()=>{

resolve('Success....!')

},5000)

})

myPromise.then((posRes)=>{

console.log(posRes)

},(errRes)=>{

console.log(errRes)

})

//Eg03

let myPromise = new Promise((resolve, reject)=>{

reject('Failure')

resolve('Success')

})

myPromise.then((posRes)=>{

console.log(posRes)

},(errRes)=>{

console.log(errRes)

})

//Eg04

let myPromise = new Promise((resolve, reject)=>{

setTimeout(()=>{

resolve('Success')

},4000)

setTimeout(()=>{

reject('Failure')

},5000)

})

myPromise.then((posRes)=>{

console.log(posRes)

},(errRes)=>{

console.log(errRes)

})

async and await

- These keywords are released in ES9.

- Above keywords are used to increase code readability.

- These keywords increase application performance.

//Eg05

let myPromise = new Promise((resolve, reject)=>{

resolve('Hello')

})

async function my\_fun(){

let res = await myPromise

console.log(res)

}

my\_fun()

//Eg06

function add(num) {

return new Promise((resolve, reject) => {

resolve(num + 5)

})

}

function sub(num) {

return new Promise((resolve, reject) => {

resolve(num - 3)

})

}

async function my\_fun() {

let res1 = await add(7)

let res2 = await sub(res1)

console.log(res2)

}

my\_fun()

===========================================

Create HTML element

document.createElement(‘<html\_element>’)

eg

const myh1 = document.createElement(‘h1’)

Insert Element in document

document.body.append(‘<html\_element>’)

eg

document.body.append(myh1)

Adding Contents

<html\_element>.innerHTML = ‘<contents>’

Eg

myh1.innerHTML = ‘Good Morning’

Setting Attributes

<html\_element>.setAttribute(‘<attribute>’,’<value>’)

Eg

myh1.setAttribute('style','color:red')

\*\*\*Task\*\*\*

\*\*\*index.html\*\*\*

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="style.css">

</head>

<body>

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

</body>

</html>

\*\*\*style.css\*\*\*

.mainDiv{

height: 200px;

width: 400px;

border: 5px double green;

border-radius: 50px 50px 0px 0px;

background: linear-gradient(to top, blue,red, yellow);

display: inline-block;

margin: 5px;

}

.ttl{

float:left;

width: 100%;

text-align: center;

font-family: cursive;

background: linear-gradient(to bottom, rgba(0,0,0,0),orange);

color: navajowhite;

border-radius: 50px 50px 0px 0px;

}

.rout{

width: 100%;

text-align: center;

margin-top: 120px;

font-family:'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Grande', 'Lucida Sans', Arial, sans-serif;

color:white;

}

\*\*\*script.js\*\*\*

function myBus(dest,rout) {

const mainDiv = document.createElement('div')

const myttl = document.createElement('h3')

const myrt = document.createElement('p')

/\*mainDiv.append(myttl)

mainDiv.append(myrt)\*/

mainDiv.append(myttl, myrt)

document.body.append(mainDiv)

myttl.innerHTML = dest

myrt.innerHTML = rout

mainDiv.setAttribute('class', 'mainDiv')

myttl.setAttribute('class', 'ttl')

myrt.setAttribute('class', 'rout')

}

myBus('Mumbai',112)

myBus('Kolkata',113)

myBus('Delhi',114)

myBus('Nwyork',1012)

myBus('Romania',2012)

myBus('Hyderabad',301)

myBus('Chennai','c32')

===========================================

Events

===========================================

Keyboard events

Mouse Events

Keyboard events

keydown

keyup

Mouse events

addEventListener

- mouseenter

- mouseleave

- mousedown

- mouseup

- mousemove

Keyboard events

Mouse Events

\*\*\*script.js\*\*\*

document.onkeydown = (e) => {

console.log(e)

console.log("Key down ", e.key)

if (e.key == 'l')

document.write('Pressed l')

}

document.onkeyup = (e) => {

console.log("Key up", e.key)

}

//create oval

const ov = document.createElement('div')

//insert element

document.body.append(ov)

//set attribute

ov.setAttribute('class', 'ovl')

//create square

const sq = document.createElement('div')

//insert element

document.body.append(sq)

//set attribute

sq.setAttribute('class', 'square')

ov.addEventListener('mouseenter', () => {

console.log('Mouse Enter')

sq.style.backgroundColor = 'gray'

})

ov.addEventListener('mouseleave', () => {

console.log('Mouse Leave')

sq.style.backgroundColor = 'transparent'

})

ov.addEventListener('mousedown', (e) => {

e.preventDefault()

console.log('Mouse down \n Co-ordinates are x = ', e.clientX, ' y = ', e.clientY)

sq.style.backgroundColor = 'yellow'

})

ov.addEventListener('mouseup', (e) => {

e.preventDefault()

console.log('Mouse up \n Co-ordinates are x = ', e.clientX, ' y = ', e.clientY)

sq.style.backgroundColor = 'transparent'

})

ov.addEventListener('mousemove', (e) => {

console.log('Mouse move \n Co-ordinates are x = ', e.clientX, ' y = ', e.clientY)

})

\*\*\*style.css\*\*\*

.square{

height : 200px;

width: 200px;

border : 2px double black;

margin : 10px;

}

.ovl{

height : 200px;

width: 200px;

border : 2px double blue;

border-radius: 50%;

margin : 10px;

}

===========================================

Callbacks

===========================================

- passing one function to another function as argument is called as callback

//Eg01

function fun\_one(arg) {

console.log(arg())

}

fun\_one(()=>{

return `Hello...!`

})

//Eg02

function fun\_one(arg1, arg2, arg3) {

console.log(arg1(), arg2(), arg3())

}

fun\_one(()=>{

return 'hi'

},()=>{

return `Hello`

},()=>{

return 'buy'

})

//Eg03

function add(num, callback) {

return callback((num + 5), false)

}

add(7, (addRes, err) => {

if (!err) {

console.log(addRes)

}

})

//Eg04

function add(num, callback) {

return callback((num + 5), false)

}

function sub(num, callback) {

return callback((num - 3), false)

}

function mul(num, callback) {

return callback((num \* 4), false)

}

function div(num, callback) {

return callback((num / 2), false)

}

add(5, (addRes, err) => {

if (!err) {

sub(addRes, (subRes, err) => {

if (!err) {

mul(subRes, (mulRes, err) => {

if (!err) {

div(mulRes,(divRes,err)=>{

if(!err){

console.log(divRes)

}

}) //div

}

}) //mul

}

}) //sub

}

}) //add

//call back hell

//Eg05

function add(num) {

return new Promise((resolve, reject) => {

resolve(num + 5)

})

}

function sub(num) {

return new Promise((resolve, reject) => {

resolve(num - 3)

})

}

function mul(num) {

return new Promise((resolve, reject) => {

resolve(num \* 4)

})

}

function div(num) {

return new Promise((resolve, reject) => {

resolve(num / 2)

})

}

async function myFun() {

let addRes = await add(5)

let subRes = await sub(addRes)

let mulRes = await mul(subRes)

let divRes = await div(mulRes)

console.log(divRes)

}

myFun()

================================================================

================================================================