11th Sep 2019 | Wednesday

Javascript Scope Chaining-

|  |
| --- |
| if (true){  let fromif=1;  //Values of variable is 1.  {  //Since we re-declare variable from '{' till the declaration line the  //variable is undefine or not accessible before initialization.  console.log(fromif);  let fromif=2;    //after redeclaration value of varible till the '}' is 2.  }  //Values of variable is 1.    }  //This is known as scope chaining. |

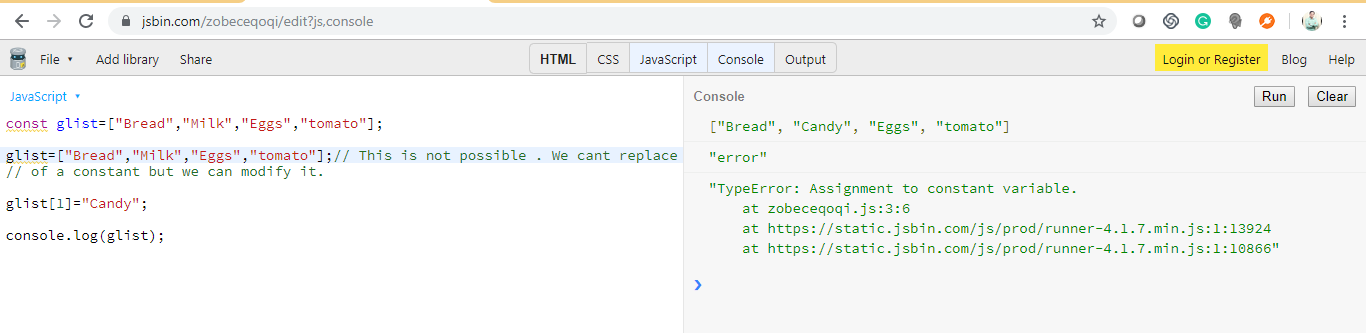
12th Sep 2019 | Thursday



Array in Javascript-

|  |
| --- |
| let glist=["Bread","Milk","Milk","Milk","Eggs","tomato"];  //console.log(glist.constructor===Array); //true  console.log(Array.isArray(glist)); //true    console.log(glist[3]); // "tomato"  console.log(glist[5]); // undefined  console.log(glist[-1]); // undefined  console.log(glist.indexOf("Eggs")); //4  console.log(glist.indexOf("Candy")); //-1  console.log(glist.indexOf("Milk")); // 1 By Default checks first occurance of Milk  console.log(glist.indexOf("Milk", 2)); //2 Second Occurance of Milk |

We should declare an array as a constant and most of the cases it should work.



|  |
| --- |
| const glist=["Bread","Milk","Eggs","tomato"];  glist=["Bread","Milk","Eggs","tomato"];// This is not possible . We cant replace value  // of a constant but we can modify it.  glist[1]="Candy";  console.log(glist); |

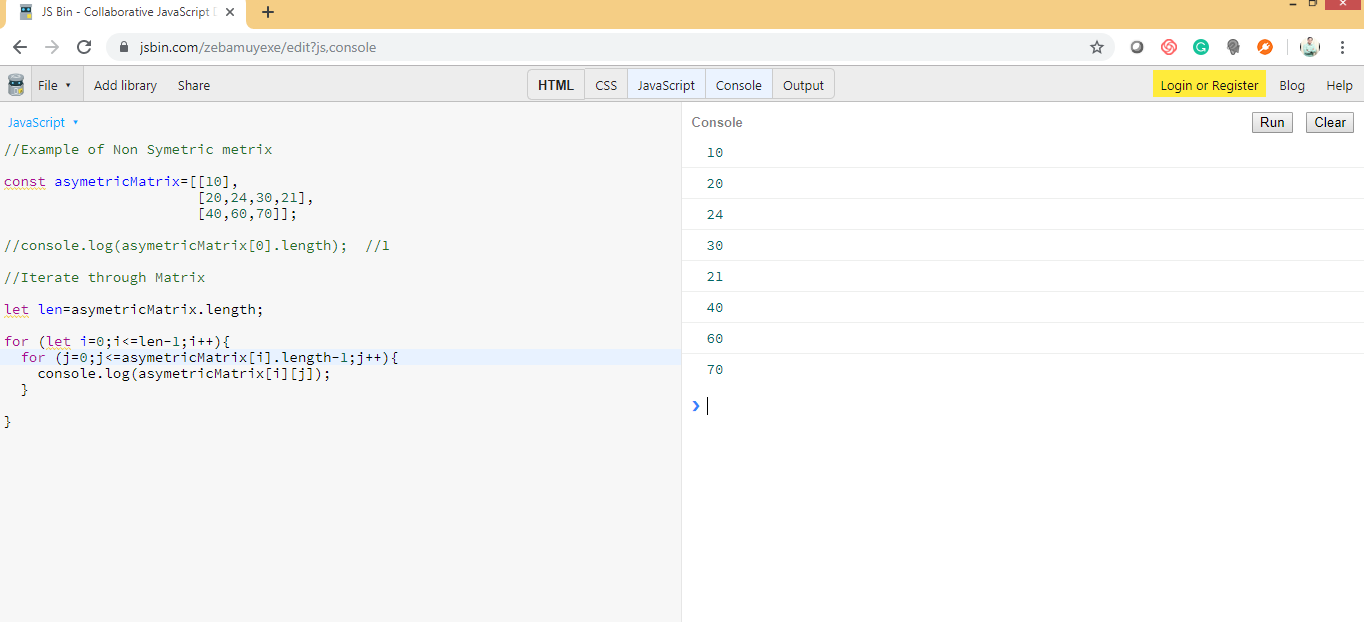
16th Sep 2019 | Monday

Mutating Array-

|  |
| --- |
| const glist=["Bread","Milk","Eggs","Tomato"];  glist.push("Candy"); // ["Bread", "Milk", "Eggs", "Tomato", "Candy"]  glist.pop(); // ["Bread", "Milk", "Eggs", "Tomato"]  glist.unshift("Candy"); //["Candy", "Bread", "Milk", "Eggs", "Tomato"]  glist.shift(); //["Bread", "Milk", "Eggs", "Tomato"]  //If you want to add/remove element from middle of an array then use splice()  // The first argument specifies the inndex where you want to add, second argument specifies  // if how many indexes you want to remove and third aregument is the value you want to add.  //glist.splice(1,1,"Spinach");  // If you dont want to remove any item just provide second argument as 0.  //glist.splice(1,0,"Spinach"); //["Bread", "Spinach", "Milk", "Eggs", "Tomato"]  //glist.splice(1,3,"Spinach"); // ["Bread", "Spinach"]  //In the next example, we are removing 3 indexes (by specifying number of  //idexes to remove in second Argument) from the array and adding threee items to it.  // We have added items (to be inserted) as arguments to the splice method.  //glist.splice(1,3,"Spinach","Paneer", "Cheese", "Onions"); //["Bread", "Spinach", "Paneer", "Cheese", "Onions"]  console.log(glist); |

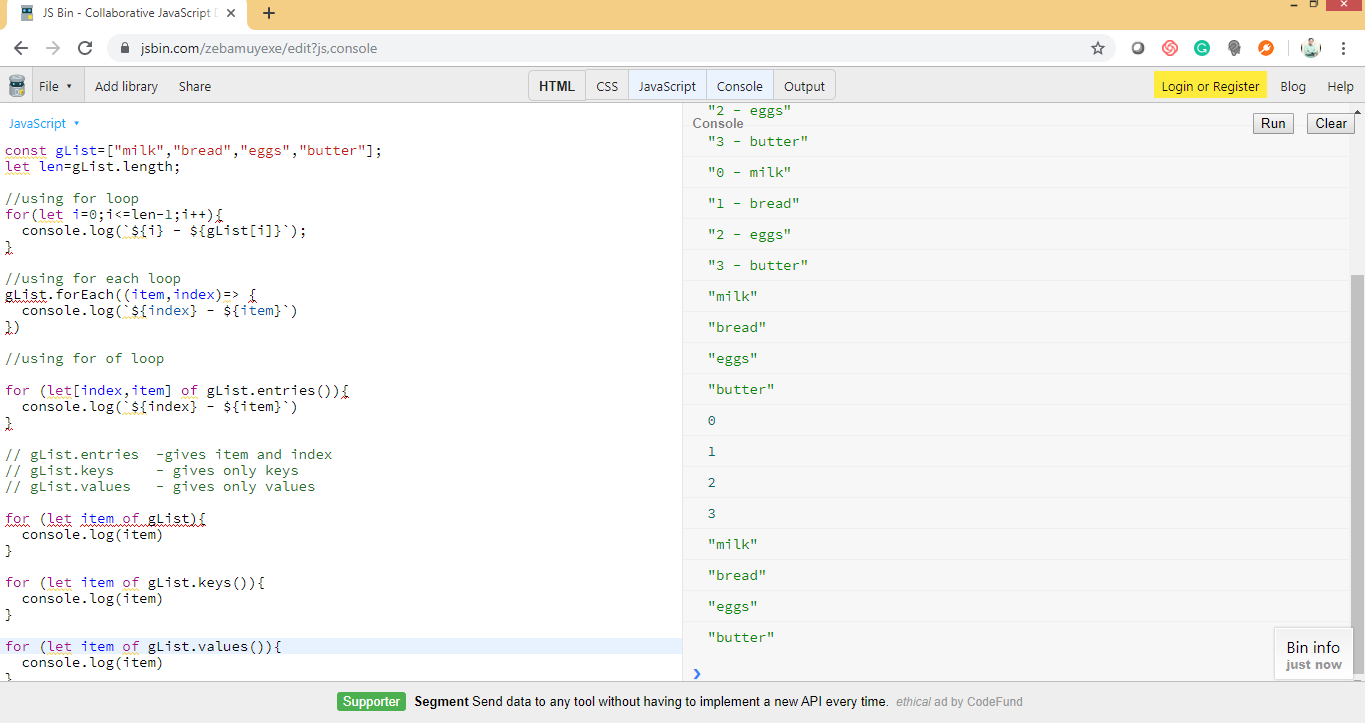
Multidimentional Array-

|  |
| --- |
| //10 15 17  //20 24 30  //40 60 70  //const row1=[10,15,17];  //const row2=[20,24,30];  //const row3=[40,60,70];  //const matrix=[row1,row2,row3];  //This is Symentric metrix  const matrix=[[10,15,17],  [20,24,30],  [40,60,70]];  //console.log(matrix);  //console.log(matrix[0].length);  //console.log(matrix[2][0]); //40  //console.log(matrix[1][1]); //24  //console.log(matrix[1][2]); //30  //console.log(matrix[2][2]); //70  //Example of Non Symetric metrix  const asymetricMatrix=[[10],  [20,24,30,21],  [40,60,70]];  console.log(asymetricMatrix[0].length); //1 |



|  |
| --- |
| //Example of Non Symetric metrix  const asymetricMatrix=[[10],  [20,24,30,21],  [40,60,70]];  //console.log(asymetricMatrix[0].length); //1  //Iterate through Matrix  let len=asymetricMatrix.length;  for (let i=0;i<=len-1;i++){  for (j=0;j<=asymetricMatrix[i].length-1;j++){  console.log(asymetricMatrix[i][j]);  }    } |

Iterating array using “for each” & “for of”



|  |
| --- |
| const gList=["milk","bread","eggs","butter"];  let len=gList.length;  //using for loop  for(let i=0;i<=len-1;i++){  console.log(`${i} - ${gList[i]}`);  }  //using for each loop  gList.forEach((item,index)=> {  console.log(`${index} - ${item}`)  })  //using for of loop  for (let[index,item] of gList.entries()){  console.log(`${index} - ${item}`)  }  // gList.entries -gives item and index  // gList.keys - gives only keys  // gList.values - gives only values  for (let item of gList){  console.log(item)  }  for (let item of gList.keys()){  console.log(item)  }  for (let item of gList.values()){  console.log(item)  } |

18th Sep 2019 | Wednesday

Iterating object using for in and for of

|  |
| --- |
| const profile={  name: "John",  single : true,  "got a job" : true,  kids: [ {name: "Peter",age:3},{name:"Sara", age: 1}]  };  //If we want to get only keys, we can use prop variable directly  /\*  for(let prop in profile)  {  console.log(prop)  }  OutPut-  "name"  "single"  "got a job"  "kids"  \*/  //If we want to get values  for (let prop in profile){  console.log(`Key: ${prop} - Value: ${profile[prop]}`);  }  //Output-  "Key: name - Value: John"  "Key: single - Value: true"  "Key: got a job - Value: true"  "Key: kids - Value: [object Object],[object Object]"  \*/  /\*  //Using for of loop  for (let prop of Object.keys(profile)){  console.log(`Key: ${prop} - Value: ${profile[prop]}`);  }  //Output-  "Key: name - Value: John"  "Key: single - Value: true"  "Key: got a job - Value: true"  "Key: kids - Value: [object Object],[object Object]"  \*/ |