JAVASCRİPT

1) javascript i projeye dahil etmek:

- Sadece o sayfada kullanacaksam javascripti o html sayfasına girip <script> yazmalıyım.

- birkaç sayfada kullanacaksam .js uzantılı bir sayfa açıp htmlde body içinde en alta şu şekilde çağırmalıyım;

<script src=”sayfaismi.js”> </script>

2) Yazılar 3e ayrılır:

1.yazılımcının görmesini istiyorsam: CONSOLE

console.log(“loglama”);

Bilgilendirmeler için kullanılır. Sayfada incele kısmındaki console’da gözükür.

console.warn(“warn”);

Console da sarı uyarı işareti

console.error(“error”);

Console da kırmızı uyarı işareti

console.info(“info”);

Console da info yani bilgilendirme

2.ekran bildirimi: ALERT

alert(“bilgilendirme”);

Yukarıdan gelen bilgilendirme bildirimi

3.ekranda görmek istiyorsam: DOCUMENT

document.writeln(“ekranda yazı”);

Ekrana yazı yazmak için.

3) Window

alert(“bilgilendirme 1”);

window.alert(“bilgilendirme 2”);

ilk 1. Alert sonra tamama basınca 2. Alert bildirim gelir.

Window nesnesi her şeyin içinde olan yegane yapı. Window, browser'ın sayfaların görüntülendiği kısmıdır.

4) Variable değişkenler

var:

var değişken result değişkenin ismidir.

Var result =”js öğreniyorum”;

Console.log(result);

Bu kod consolea js öğreniyorum yazdırır.

Var result = 4423;

Console.log(result);

Bu kod consolea 4423 yazdırır. Sayılar “” içinde yazılmak zorunda değildir.

Var result = 44.23;

Console.log(result);

Float yani noktalı sayılarda

Var result = true;

Console.log(result);

Bool yani true false

Var değişkeniyle yazılabilir.

Typeof yapısı

Var result = 44.23;

Console.log(typeof result+” - ”+result);

Bu kodda typeof result yazılan değişkenin türünü gösterir. 4423 ve 44.23 number, js öğreniyorum string, true ise boolen çıkar.

naming convention (adlandırma kuralı)

sayıyla başlanmaz. \_ $ bunlarla başlanabilir. Ortasına sonuna sayı konabilir ama sayıyla başlanmaz.

Whoisting

Var result2=56;

Var result2;

Şeklinde de yazılabilir js te.

<script>

x = 5;   // x'e 5 değerini atadık

console.log(x) // x değerini konsola yazdırdık

var x;  // x değişkenini tanımladık

</script>

<script>

var x; // x değişken tanımlandı

x = 5; // 5 değeri atandı

console.log(x); //x değeri ekrana yazıldı

</script>

İkiside olur.

5) OPERATORS

+, - , \*, \*\*, /, % , !

! :değil

&&: ve

|| : veya

X++ :işlemden sonra 1 arttır

X-- : :işlemden sonra 1 azalt

++X : önce 1 arttır sonra işlemi yap

--X : önce 1 azalt sonra işlemi yap

= : atama

== : eşit (türüne bakma)

=== : eşit (türüne bak)

6) Kullanıcıdan veri alma

var kullaniciGiris=prompt("merhaba");

ekranın üstünde bildirim olarak yazı yazabileceğimiz kutucuk gelir.

Prompt datadan gelen veriler stringtir. Yazı olarak gözükür.

Cast

var kullaniciGiris=Number(prompt("Sayı Giriniz."));

artık sayı olarak kullanabiliriz.

Sayılar için number()

Yazılar için string()

7) Undefined (Tanımsız)

Var result;

Console.log(result);

Eğer değişkene değer vermezsek consoleda undefined yazar.

7) NaN

Var result=”asd”/4;

Console.log(result);

Eğer stringi sayıyla işleme sokarsak consoleda NaN yazar. Not a number demektir. Bu bir sayı değil ki işlem yapabileyim der.

8) isNaN

Var result=4;

Console.log(isNaN(result));

Console da false yazarsa result sayıdır

8) İnfinity

Console.log(4/0);

Bir sayıyı 0 a bölersen infinity yazar.

8) Math

Console.log(Math.PI); pi değerini gösterir

Console.log(Math.E); e değerini gösterir

Console.log(Math.min(4,1,99,55,999,-100)); en küçük değeri gösterir

Console.log(Math.max(4,1,99,55,999,-100)); en büyük değeri gösterir

Console.log(Math.sqrt(16)); yazan sayının karakökünü alır

Console.log(Math.abs(-25)); mutlak değerini gösterir

Console.log(Math.pow(2,5)); üslü sayılardır. 2 üzeri 5

Console.log(Math.floor(3.9)); aşağı yuvarlar. 3e

Console.log(Math.ceil(3.1)); yukarı yuvarlar. 4e

Console.log(Math.round(6.4)); 4 ve altındakileri aşaği yuvarlar. 6ya

Console.log(Math.round(6.5)); 5 ve üstündekileri yukarı yuvarlar. 7ye

Console.log(Math.sin(45)); sin hesaplar.

Console.log(Math.random()); 0 ila 1 arasında random sayı atar.

9) Number

Var sayi=15000;

Console.log(sayi);

Var sayi2=1.5E+4;

Console.log(sayi2);

İkisi de 15000 yazar

2lik sayı sistemi 0-1;

Var binary=0b00011;

Console.log(binary);

8lik sayı sistemi 0-7;

Var octal=0o00011;

Console.log(octal);

10luk sayı sistemi 0-9;

Var decimal=23213;

Console.log(decimal);

16lık sayı sistemi 0-F;

Var hexadecimal=0xffa;

Console.log(hexadecimal);

9) Cast

Console.log(Number(“10”)+10);

Number() yapısı ile stringi numbera dönüştürür. 20 yazar

Console.log(parseInt(“10”)+10);

Verilen stringi int e çevirir. 20 yazar

Console.log(String(“30”)+10);

Numberı stringe çevirir. 3010 yazar.

Var number=10;

Console.log(number.toString()+10);

Numberı stringe çevirir. 3010 yazar.

4) Variable değişkenler

var

Var data1=44;

Console.log(data1);

Var data1=23;

Console.log(data1);

Console da 44 de yazar 23 de. Global değişkenlerde kullanıyoruz. Global değişken nedir? Bir projede her yerde kullanacaksanız bu global değişkendir.

Let

Ancak sadece o scope içinde kullanacaksanız let kullanıcaz.

Var data1=11;

Console.log(data1);

Var data1=22;

Console.log(data1);

let data2=33;

Console.log(data2);

let data2=33;

Console.log(data2);

Error verir çünkü let yanına aynı değişkene farklı sayılar verdik.

let data2=33;

Console.log(data2);

let data3=33;

Console.log(data3);

Error vermez alt alta 33 yazar çnkü farklı değişkenlere veri girdik.

let data3=33;

data3=44;

Console.log(data3);

44 yazar. Sonradan verilen değer yazar.

Const

const data4=55;

Console.log(data4);

const data4=55;

Console.log(data4);

Error verir.

const data5=55;

data5=66;

Console.log(data5);

Error verir. Letteki gibi bir daha değişmez.

10) String yapılar

Let str=”js öğreniyorum”;

Console.log(str);

Console.log(str.length); kaç harfli olduğunu söyler

Console.log(str.trim()); başındaki ve sonundaki boşlukları alır.

Console.log(str.toUpperCase()); her şeyi büyük karakterli yapar.

Console.log(str.charAt(0)); bu değişkendeki 0. İndexteki karakteri gösterir. Burada j gösterir.

Console.log(str.indexOf(“js”)); arama yapar. İlk indexten başlar ve kaçıncı indexte başladığını söyler. Burada 0 yazar.

Console.log(str.lastIndexOf(“js”)); aramayı sondan yapar. Son indexten başlar. Burada 11 yazar.

Console.log(str.contact(“js”)); sona js yazar. js öğreniyorum js olur.

Console.log(str.replace(str,”değiştir”)); string içini değiştirir. js öğreniyorum yerine değiştir yazar.

Console.log(str.substring(1)); 0. İndexi göstermez. 1.indexten itibaren gösterir. 1. İndexten itibaren göster demektir.

Console.log(str.substring(0,4)); 0. Ve 4. İndex arasını göster. Js ö yazar.

11) Function normal fonksiyon

Projede birden fazla kullanacağımız kodları içine yazarız.

a) Returnsüz ve parametresiz böyledir:

Function returnsuzParametresiz(){

Console.log(“deneme”);

}

returnsuzParametresiz ()

deneme yazar.

deneme fonksiyonunu çağırıp ayağa kaldırdık.

b) Returnsüz ve parametreli böyledir:

Function returnsuzParametreli(adi){

Console.log(“deneme”+adi);

}

returnsuzParametreli (“HamitM”)

deneme HamitM yazar.

c) Returnlü ve parametresiz böyledir:

Function returnluParametresiz(){

return= “deneme”;

}

let result= returnluParametresiz ()

Console.log(result);

deneme yazar.

a) Returnlü ve parametreli böyledir:

Function returnluParametreli(soyadi){

return= “deneme ”+soyadi;

}

let result= returnluParametreli (Mizrak)

Console.log(result);

deneme Mizrak yazar.

12) anonymos Function fonksiyon

Anonimdir. İsimsizdir.

let value1 function( ) { }

13) arrow Function fonksiyon

let value2= ( ) => { }

14) Immedia Function fonksiyon

(function(){

Console.log(“deneme4”);

})();

15) conditional karar merkezi

a) if else

Const condi= ()=> {

Let number=4;

İf(number>2){

Console.log(“2den büyüktür.”);

}

Else{

Console.log(“küçüktür”);

}

}

Condi()

Bu aynı zamanda şöyle de gösterilir;

Const condi= ()=> {

Let number=4;

Let result=(number>2) ? “2den büyüktür.” : “küçüktür”;

Console.log(result);

}

Condi()

Bu aynı zamanda şöyle de gösterilir;

Const condi= ()=> {

Let number=4;

Console.log((number>2) ? “2den büyüktür.” : “küçüktür”;);

}

Condi()

b) if, else if, else

Const condi= ()=> {

Let number=4;

İf(number==1){

Console.log(“1”);

}

Else İf(number==2){

Console.log(“2”);

}

Else İf(number==3){

Console.log(“3”);

}

Else İf(number==4){

Console.log(“4”);

}

Else{

Console.log(“1 2 3 4 değildir.”)

}

}

Condi()

c)switch case

Const condi= ()=> {

Let number=4;

Switch(number){

Case 1:

Console.log(“1”);

Break;

Case 2:

Console.log(“2”);

Break;

Case 3:

Console.log(“3”);

Break;

Case 4:

Console.log(“4”);

Break;

Default:

Console.log(“1 2 3 4 değildir.”);

Break;

}

Condi()

Break: döngü kırmak

Return: metot kırmak

Continue: 1 kereye mahsus es geç sonra döngüye devam et

16)LOOP

A) for döngüsü

For(let i=0; i<=10;i++){

}

B)while

Let k=0;

While(k<=10){

k++;

}

17)try catch: hataları algılar

Let tryCatchTuto=()==>{

Try{

Alertx(“pop-up”)

}

Cacth(error){

Console.log(“hata mesajı:”+error.message);

Console.error(error);

}

Console.log(“son satır”);

}

tryCatchTuto()

Bu kodun consoleda çıktısı:

Hata mesajı: alertx is not defined

Console.error(error); kırmızı renkte hatanın kaçıncı satırda olduğunu yazar.

Son satır

Olur. Try catch hatanın nerede olduğunu söyler.

Let tryCatchTuto=()==>{

Try{

Alertx(“pop-up”)

}

Cacth(error){

Console.log(“hata mesajı:”+error.message);

Console.error(error);

}

Finally{

Console.log(“port.close”)

}

Console.log(“son satır”);

}

tryCatchTuto()

bu kodun en sonunda port.close yazar.

18)debug

Bi şeyler yazıyoruz. Yazdığımız yerde istediğimiz sonuç çıkmıyor. Hata ayıklama yani debug yapmalı.

Const debugData=()==>{

Let sum=0;

For(let i=1; i<10; i++){

Sum=sum+i;

}

Console.log(“toplam:” +sum);

}

debugData()

satır sayısı yanındaki boşluğa kırmızı nokta işaretle. Vscode da yanda böcek işaretli run and debuga bas. Run and debug de. Chrome seç. Çıkan sayfada url kısmına web sitenin urlsini yapıştır. Sol üstteki launch Chrome a tıkla. F5e bas. Sağ üstten kareye bas stop de.

19) Monad

Bir fonksiyonun çıktısı başka bir fonksiyonun girdisi ise buna monad denir.

const birinci=()==>{

let user=Number(prompt(“Lütfen bir sayı giriniz.”));

return user;

}

const ikinci=()==>{

let data=birinci();

console.log(Math.sqrt(data));

}

İkinci();

20) setTimeOut(), setInterval();

setTimeout(function(){

console.log(“setTimeOut çalıştı.”);

},2000);

2 saniye sonra konsolda yazar. Belirlemiş olunan zamandan sonra sistemde çalışması için kullanılır.

Belirli zaman sonra 1 kere çalışıp durur.

setInterval(function(){

console.log(“setInterval çalıştı.”);

},1000);

Settimeout yapısı çalışır biter ama setinterval yapısı çalışmaya devam eder.

Belirli bir zaman sonra çalışır ve sonsuza kadar kapanıp tekrar çalışır.

21) call back function

Fonksiyonu geri çağır anlamına gelir.

Çoklu işlemlerde kullanılır

const birinci=(data)==>{

console.log(Math,pow(2,data));

return user;

}

const ikinci=(callbackFunction)==>{

let user=Number(prompt(“Lütfen sayı giriniz.”));

let data= callbackFunction(user)

console.log(data);

}

İkinci(birinci)

22) promise

const prom=()==>{

let data=new Promise( (resolve, reject)==>{} ).

then(

()==>{console.log(“olumlu”)}

).

catch(

(err)==>{console.error(err);}

);

}

prom()

promise içinden gelen veri true ise thene düşer, eğer hata varsa mesela apiden veri gelmemesi gibi

23) diziler

Let arr=()🡺{

const numbers=[3,5,6,1,9];

return numbers;

}

let arrResult=()🡺{

let data=arr();

console.log(data);

console.log(data[0]);

console.log(data[data.length-1]); //son elemanı almak için.

}

arrResult();

konsolda

[3,5,6,1,9]

3

9

Yazar.

24) Iterative for loop, for in, for of, foreach

Let arr=()🡺{

const numbers=[3,5,6,1,9];

return numbers;

}

let arrResult=()🡺{

let data=arr();

console.log(data);

console.log(data[0]);

console.log(data[data.length-1]);

//iterative for loop

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

document.write(data[i]+” “) //ekrana diziyi yazdırır. Yani 3 5 6 1 9

}

Document.write(“<br/>”);

//for in

For(let temp in data){

document.write(temp+” 🡺 ”+data[temp]+”<br/>”); //0dan başlayarak numarasını ve hangi sayı olduğunu //yazar. İn yapısı index numarasını gösterir

}

}

arrResult();

//for of

For(let temp of data){

document.write(temp+” ”); //of yapısı ise elemanların kendisini gösterir. Yani 3 5 6 1 9 yazar.

}

//forEach()

Data.forEach(function(value,index,array){

document.write (index+” 🡺 ” +value+” ”);

// document.write (‘${index} 🡺 ${value} <br/> ’);

});

}

arrResult();

ekranda

3 5 6 1 9

0 🡺 3

1 🡺 5

2 🡺 6

3 🡺 1

4🡺 9

3 5 6 1 9

0 🡺 3

1 🡺 5

2 🡺 6

3 🡺 1

4🡺 9

Yazar

FIFO nedir?

İlk giren ilk çıkar (FIFO) yöntemi, mağazadan malların satışının veya çıkışının eldeki en eski stoktan yapıldığı bir tekniktir. Bozulabilir malların söz konusu olduğu bir senaryoda, en erken stok ilk olarak ele alındığından bu en uygun yöntemdir, dolayısıyla bozulabilirlik riskini azaltır.

LIFO nedir?

Son giren ilk çıkar (LIFO), son stok ürününün önce satılacağı varsayımına dayanan bir stok değerleme yöntemidir. Ancak bu tekniğin envanter hareketine aykırı ve mantıksız olduğu kanıtlanmıştır. Bir ekonomide enflasyonun olduğu bir senaryoda, satılmayan malların değeri düşerken, satılan malın maliyetinin değeri artacak ve bu da düşük kar ve gelir vergisi ile sonuçlanacaktır. Ancak bir ekonomide deflasyonun olduğu bir senaryoda satılmayan malların değeri artarken satılan malın maliyetinin değeri düşerek yüksek kâr elde edilir.

25) filter

Let arr=()🡺{

const numbers=[3,5,6,1,9];

return numbers;

Data.filter(function(value,index,array){

Return value%2==0;// Bu dizide 2ye bölünenleri gösterir.

}).forEach(function(value,index,array){

document.write (index+” 🡺 ” +value+” ”);

// document.write (‘${index} 🡺 ${value} <br/> ’);

//MAP

Data.map(function(value,index,array){

Return value += 10 //10 ekler. Map yapısı dizideki tüm indexlere gider ve işlem yapar.

}).forEach(function(value,index,array){

document.write (index+” 🡺 ” +value+” ”);

// document.write (‘${index} 🡺 ${value} <br/> ’);

}

arrResult();

data.push(99); //dizinin sonuna 99u ekler

data.unshift(11); //dizinin başına 11 ekler

data.pop();//sondan bir eleman çıkarır

data.shift()//baştan bir eleman çıkarır

data.short(); //küçükten büyüğe sıralar

data.reverse();//dizeyi ters çevirir

data.short().reverse();//büyükten küçüğe sıralar,

26)dizi objesi

let callbackFunctionComputer=()🡺{

const computerArray=[

{computerName:”callback computer 1”,price:100},

{computerName:”callback computer 2”,price:200},

{computerName:”callback computer 3”,price:300}

];

}

Dizi objesi içerisinde 5 tane random obje oluşturma.

let callbackFunctionComputer=()🡺{

const computerArray=[];

for(let index=0; index<5,index++5){

let computerObject={computerName:”callback computer”+( index +1),price:(index+1)\*100};

computerArray.push(computerObject);

}

console.log(computerArray);

}

//bu dizi içindeki computer Name bileşenlerini gösterme (map ile)

Const arrayInComputerName = () 🡺{

computerArray.map((temp)🡺<{

console.log(‘${temp.computerName}’);

})

}

//call back function object

Const arrayInComputerObject=(obj, callBackFnc)🡺{

ComputerArray.push(obj);

callBackFnc();

}

arrayInComputerObject({computerName:”computer 6”,price:600},arrayInComputerName)

}

callbackFunctionComputer();

27)PROMİSE

Dizi objesi içerisinde 5 tane random obje oluşturma.

let promiseFunctionComputer=()🡺{

const computerArray=[];

for(let index=0; index<5,index++){

let computerObject={computerName:” computer”+( index +1),price:(index+1)\*100};

computerArray.push(computerObject);

}

console.log(computerArray);

}

//bu dizi içindeki computer Name bileşenlerini gösterme (map ile)

Const arrayInComputerName = () 🡺{

computerArray.map((temp)🡺<{

console.log(‘${temp.computerName}’);

})

}

arrayInComputerName();

//object function

Const arrayInComputerObject=(obj)🡺{

const promiseReturn=new Promise(()🡺{

ComputerArray.push(obj);

})

Return promiseReturn;

}

arrayInComputerObject({computerName:”computer 6”,price:600})

.then((response)🡺{console.log(“response”)})

.catch((err)🡺{console.error();});

}

promiseFunctionComputer();

28)object

Let objectTutorials=()🡺{

const personObj={

“name”: “elif”,

surname:”kıvcı”, //böyle de yazılır

“number”:23,

“isLogin”:true,

“software”:[“html”,”css”,”js”] //dizi yazılabilir

fullName:function(){ //objede function kullanımı. arrow function kullanılmaz

return this.name+” “+this.surname; //objede bir datayı çağırmak istiyorsak this kullanılır.

}

};

Console.log(personObj);

//variable değişken

Console.log(personObj.name);

Console.log(personObj.surname);

//variable böyle de yazılır nested

Console.log(personObj[“name”]);

//array nasıl gösterilir

Console.log(personObj.software[0]);

}

Const objectToString=JSON.stringify(personObj); //JSON.stringify objeyi stringe çevirir

Console.log(objectToString);

//{“name:”elif”,surname:”Kıvcı”,”number”:… şeklinde yazar}

Const stringToObject=JSON.parse(stringToObject); //stringi objeye çevirme

Console.log(stringToObject.name); //elif yazar

// object variable

PersonObj.language=”engilish” //objemize sonradan bir özellik eklemek istiyorsak

objectTutorials(PersonObj.language);//engilish yazar

//function callback. Obje içindeki functionı çağırdık return kısmı döndü elif Kıvcı yazdırdı

console.log(personObj.fullName());

objectTutorials()

29) object constructor kalıtımlama bağlama

Let objectConstructor=()🡺{

Let personConstructor=function(name,surname){

this.name=name; //sağdaki yani sondaki name parametre olarak gelen namedir. This.namedeki name iste //kalıtımlanacak yapıdır.

this.surname=surname;

console.log(this); //this yukarıdaki objeyi işaret edecek console logda datamızı gösterecek

}

Let personResult=new personConstructor (“elif”,”kıvcı”)//new diyoki sisteme sen benim yerime git hafızada //yer aç

//consoleda name: elif surname: Kıvcı yazar

}

objectConstructor()

json nedir xml nedir aralarındaki fark nedir

JSON (JavaScript Object Notation), basit veri yapılarını temsil etmek için tasarlanmış bağımsız bir veri değişim formatıdır. Esas olarak iki sistem arasındaki veri alışverişi için kullanılır. Örneğin JSON kullanarak sunucu ile web uygulaması arasında veri aktarabilirsiniz.

JSON (JavaScript Object Notation) ve XML (eXtensible Markup Language), veri saklamak, iletmek ve almak için kullanılan iki farklı veri formatıdır. İşte bu iki format arasındaki temel farklar:

JSON vs. XML

* JSON çıkış noktası Javascript’tir. XML ile markup dillerinden çıkmıştır.
* JSON’da dizi desteği vardır. XML’de dizi diye ayrı bir kavram yoktur.
* JSON, XML’e göre daha küçük ve hızlıdır.
* JSON okunabilirlik açısından daha kolay ve anlaşılabilirdir.
* JSON, XML gibi geliştirilebilir bir yapıya sahip değildir.

30) call, apply, bind

Bir fonksiyon var bir obje var ikisini bağlayacağız

Let noParameterCallApplyBind =()🡺{

Let functionOtherObject=function(){

Document.writeln(‘parametresiz Function: ${this.adi}<br/>’)

};

Let objectData={

“adi”:”spora gidiyorum”

};

functionOtherObject.call(objectData);

functionOtherObject.apply(objectData);

let deneme= functionOtherObject.bind(objectData); //bind bir değişkene atanmalı

deneme();

}

noParameterCallApplyBind();

//ekranda 3 kere parametresiz function: spora gidiyorum yazar.

Document.writeln(‘<br/>’)

//parametreli nasıl olur

Let ParameterCallApplyBind =()🡺{

Let functionOtherObject=function(surname){

Document.writeln(‘parametresiz Function: ${this.adi} 🡺 ${this.surname}<br/>’)

};

Let objectData={

“adi”:”spora gidiyorum”

};

functionOtherObject.call(objectData,”kıvcı”);

functionOtherObject.apply(objectData[“kıvcı”]);

let deneme= functionOtherObject.bind(objectData,”kıvcı”);

deneme();

}

ParameterCallApplyBind();

//ekranda 3 kere parametresiz function: spora gidiyorum 🡺kıvcı yazar.

31) event

Htmlde butonumuza onclick=”eventFunction()” yazdık. Onclick tıkladığında demek. Tıklandığında eventFunction() fonksiyonu çalışacak.

Jste

Let eventFunction=()🡺{

alert(“deneme”)

}

Ondblclick yazarsak ise butona 2 kere basılınca çalışır

Onmouseleave dersek butonun üstüne gelip Mouse buton üstünden çıkınca çalışır

Onmouseover Mouse üzerine gelince çalışır

Onkeydown Mouse ile üstüne gelip tuşa tıklayınca çalışır

32) dom

Açılımı document object model. Yani sayfa obje modeli.

Document sayfanın içindeki html kodlarını görür. Document sayfa demektir. Js de document.writeln ile sayfaya bir şeyler yazdırabiliriz.

Data yazan butonun yanında … yazdık. Butona basınca … yerine güncel tarih gelsin istiyoruz.

<button class=”btn btn primary” onclick=”newDataDate()”>DATA</button>

<span class=”text-danger” id=”text\_data”>…</span>

Jste //getElementById ile ulaşmak erişmek istediğimiz değişiklik yapacağımız şeye ıd vererek ulaşırız. getElementByName de name ile ulaşırız.

let newData=()🡺{

document.getElementById(“text\_data”).innerHTML=”17/09/2023”; //innerHTML htmli değiştir demek. //Butona basınca 17/09/2023 yazar.

}

innerHTML=new Date().getFullYear(); yazarak da 2023 yazısına ulaşırız.

let newData=()🡺{

let now = document.getElementById(“text\_data”)

now.innerHTML= “<b>”+ new Date().getFullYear();+”</b>” //bolt yani kalın yaptık.

now.innerHTML= ‘ <b> ${new Date().getFullYear()} </b>‘ //şeklinde de yazılır.

now.innerText = ‘ <b> ${new Date().getFullYear()} </b>‘ //<b>2023</b> yazar. innerText html etiketlerini çalıştırmaz.

}

Css dom

html

<button class=”btn btn primary” onclick=”newDataDate()”>DATA</button>

<span id=”text\_data”>…</span>

js

let newData=()🡺{

let now=document.getElementById(“text\_data”);

now.innerHTML=new Date().getFullYear(); yazarak da 2023 yazısına ulaşırız. Queryselector : aynı mantık id alacaksak başına # class alacaksak başına .

//let now = document.queryselector(“#text\_data”); yazarak queryselector de kullanabiliriz.

//css kısmı

now.style.color=”orange”; //2023 kısmı turuncu yazar.

}

Confirm(“”) bu kod onaylama anlamına geliyor. Window.confirm(“”) şeklinde yazılıyor

Örn

Window.confirm(“bu sayfaya geçmek ister misiniz?”)

? window.location=[”https://www.google.com/](https://www.google.com/)” //evetse bu komut çalışır Google gider

:window.alert(“bu sayfada kalındı”) //hayırsa bu komut çalışır

<button class="btn btn-primary" onclick="sayfaDegis()">out site</button>

let sayfaDegis=() => {

        window.confirm("sayfayı değiştirmek istediğinize emin misiniz?")

            ? window.location="https://www.google.com/"

            : window.location="http://127.0.0.1:5500/index.html";

};

33) LİSTENER

//listener

let userListener=() => {

    let listenerParag=document.getElementById("parag\_list")

    listenerParag.addEventListener("click",function(e) {

        //alert(e.preventDefault+" "+e.target);

        document.getElementById("log\_admin").innerHTML="takip edildi."

        //$("#log\_admin").html("takip edildi"); //jquery

    });

};

userListener();

<p onclick="userListener()" id="parag\_list">Lorem50p>

    log information: <span id="log\_admin"></span>

bu kod paragrafa tıklayınca aşağıda takip edildi yazdırır. Bu sistemle paragrafa ne kadar okuma geldiğini yazdırabiliriz.

34)localStorage

Veriler bizim için önemlidir. Verileri saklarız. Database browser gibi yerlerde. localStorage sayfayı incele dediğimizde application bölümünde bulunur. LocalStorage verileri kaydetmemize saklamamıza yarar.

//localStorage

let localStorageData=()=>{

    const adi= localStorage.setItem("adi",prompt("adi:"));//prompt ile kullanıcıdan veri aldık.

    const soyadi= localStorage.setItem("soyadi","kıvcı");

    console.log(localStorage);

    //buraya kadarki kısım datalarımı kaydetmek istediğimde yazıyorum.

    const getAdi=localStorage.getItem("adi");

    alert(getAdi);

    //burada localstoragedaki verileri görmemiz için

    //veriyi kaydetmek istiyorsak setItem ile kaydediyoruz. geri çağırmak istiyorsak getItem ile çağırıyoruz

    localStorage.removeItem("adi"); //ilgili itemi kaldır demek

    localStorage.clear();//her şeyi siler

}

localStorageData()

JQUERY

<div id="clickme">

    Click here

  </div>

  <img id="book" src="elisaicon.png" alt="" width="100" height="123">

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

<script>

// With the element initially shown, we can hide it slowly:

    $( "#clickme" ).on( "click", function() {

      $( "#book" ).hide( "slow", function() {

        alert( "Animation complete." );

      });

    });

</script>

Jquery js kütüphanesidir. Hide yapısını işledik.

35) javascripti jsona json ı javascripte dönüştürme

Stringify jsi jsona parse jsonı jse dönüştürür.

Const kitapJSON=JSON.stringify(kitap); // js objesini jsona çevirme

Const kitapObje=JSON.parse(kitapJSON); //json ı js objesine çevirme