Web Storage

Web storage sayesinde tarayıcı üzerinde veriler saklanabilmektedir.

Storage üzerinde işlemler yapılırken sessionStorage ve localStorage objeleri kullanılmaktadır. Bu objeler aşağıdaki özellik ve metotları içermektedir.

setItem(key{s}, value{s})

getItem(key{s})

removeItem(key{s})

clear() : Storage’daki tüm verileri siler.

length : Storage’daki eleman sayısını verir.

key(index{n}) : Bu metot sayesinde storage’daki tüm keylere ve bu vasıtayla tüm verilere ulaşabilirsiniz.

Example

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

let key = sessionStorage.key(i);

console.log(sessionStorage.getItem(key));

}

* getItem() metodu ile okunmak istenen veri yoksa geriye null değer döner.
* removeItem() metodu ile silmek istediğiniz veri storage’da mevcut değilse herhangi bir hata meydana gelmez.
* Aynı key değerine sahip farklı bir eleman eklemeniz durumunda değer değişikliği yapılır.
* Storage’daki veriler string türündedir. Dizi ve objeler storage’a string formatında eklenebilmektedir. Bunun için JSON.stringfy() metodu kullanılmaktadır. (Okuma yapılırken JSON.parse() metodu ile string veriler yeniden obje ve dizilere dönüştürülebilmektedir.)
* Session storage oturum bazında local storage ise tıpkı çerezler gibi kalıcı olarak veri saklama imkanı sunmaktadır.
* Storage kelimesi Türkçe’de **depolama** anlamına gelmektedir.
* Büyük boyutlu veriler storage’a eklenirken kota aşılabilir. Bu durumda hata meydana gelir. Hataların önüne geçmek için veri ekleme satırı try-catch blokları içerisine alınabilir.

Example

try{

localStorage.setItem("key", "value");

}

catch(error){

console.log("Exceeded Storage Quota!");

}

* W3C tarafından önerilen storage limiti 5MB’dır. Bazı tarayıcılar daha fazlasına da izin vermektedir.
* Web storage browser desteğini kontrol etmek için aşağıdaki yapıyı kullanabilirsiniz.

Example

if(typeof(Storage) !== undefined){

}

else{

}

(Session Storage)

* Session storage tarayıcı sekmesine özeldir. Bu, bir sekmenin verilerine başka bir sekmeden ulaşamazsınız demektir. Örnek vermek gerekirse iki farklı sekmede aynı web sayfası dahi açık olsa her iki sekmenin verileri farklıdır.
* Tarayıcı sekmesi kapatıldığında ilgili sekmenin session storage verileri kaybolur.

(Local Storage)

* Local storage’da saklanan veriler her request’te çerezler gibi sunucuya gönderilmez. Bu sayede request’lerin boyutu küçülür ve bandwith kullanımı azalır.
* Local storage’daki veriler siz ellemediğiniz sürece silinmezler. Tabii kullanıcıların storage’daki verileri elle silmesi de mümkün bir seçenektir. Bu hem local storage hem de session storage için geçerlidir.
* Ziyaretçi aynı web sitesini farklı sekme ya da pencerelerde açmış olabilir. Local storage üzerinde veri ekleme, silme veya güncelleme işlemleri yapıldığında diğer sekme ve pencerelerde **storage** adında bir event meydana gelir.(storage eventi değişikliğin yapıldığı sayfada meydana gelmez.) (storage eventi sadece **değişiklik** olmuşsa tetiklenir. Örneğin getItem() ile okuma yapıldığında tetiklenmez. removeItem() ile olmayan bir veriyi silmek istediğinizde tetiklenmez. Varolan bir key-value çiftini tekrar eklediğinizde tetiklenmez. Bu konuda tek istisna clear() metodudur, storage boş bile olsa eventin tetiklenmesine neden olur.)
* storage eventi fonksiyonun parametresine bir takım bilgiler göndermektedir. Bu bilgiler aşağıda listelenmiştir.

key : (clear() kullanılmışsa bu alanın değeri null olur.)

newValue : (clear() veya removeItem() kullanılmışsa bu alanın değeri null olur.)

oldValue : setItem() veya removeItem() kullanıldığında ilgili key’e ait önceki değer aktarılır.

url : storage eventini fırlatan web sayfasının adresini içerir.

storageArea : Local storage objesini döndürür.

Example

window.addEventListener("storage", function(e){

  let key = e.key;

  let newValue = e.newValue;

  let oldValue = e.oldValue;

  let url = e.url;

  let storageArea = e.storageArea;

})

Geolocation

Geolocation API sayesinde ziyaretçilerin konum bilgileri(coğrafi kordinat değerleri) elde edilebilmektedir.(Ziyaretçinin onayı gereklidir)

Enlem ve boylam verileri elde ediliriken geolocation objesi kullanılmaktadır. Bu obje navigator objesi içerisinde yer almakta ve aşağıdaki yapı kullanılarak browser desteği kontrol edilebilmektedir.

Example

if(navigator.geolocation){

}

else{

}

Example

if(navigator.geolocation){

navigator.geolocation.getCurrentPosition(function(position){

*//Enlem*

let latitude = position.coords.latitude;

*//Boylam*

let longitude = position.coords.longitude;

*//Rakım. Metre cinsinden deniz seviyesinden yükseklik.(Varsa döndürülür)*

let altitude = position.coords.altitude;

*//Konum bilgisinin elde edilme zamanı.(Tarih türüne dönüştürülmelidir)*

let timestamp = position.timestamp;

*//metre/saniye cinsinden hız.(Varsa döndürülür)*

let speed = position.coords.speed;

console.log("Enlem : " + latitude);

console.log("Boylam : " + longitude);

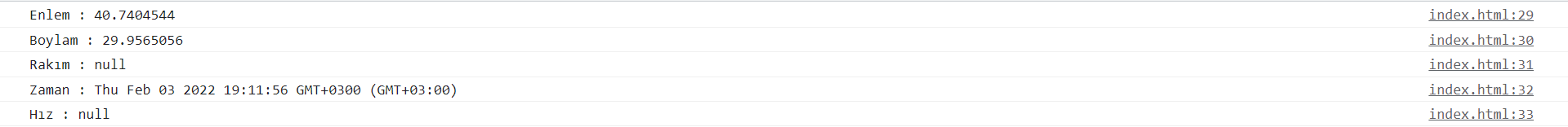
console.log("Rakım : " + altitude);

console.log("Zaman : " + new *Date*(timestamp));

console.log("Hız : " + speed);

});

}



getCurrentPosition() metodunun ikinci parametresini kullanarak hataları yönetebilirsiniz. Hata kodları ve anlamları aşağıdaki örnekte verilmiştir.

Example

function successCallback(position){

*//Enlem*

  let latitude = position.coords.latitude;

*//Boylam*

  let longitude = position.coords.longitude;

*//Rakım. Metre cinsinden deniz seviyesinden yükseklik.(Varsa döndürülür)*

  let altitude = position.coords.altitude;

*//Konum bilgisinin elde edilme zamanı.(Tarih türüne dönüştürülmelidir)*

  let timestamp = position.timestamp;

*//metre/saniye cinsinden hız.(Varsa döndürülür)*

  let speed = position.coords.speed;

  console.log("Enlem : " + latitude);

  console.log("Boylam : " + longitude);

  console.log("Rakım : " + altitude);

  console.log("Zaman : " + new *Date*(timestamp));

  console.log("Hız : " + speed);

}

function errorCallback(error){

let errorCode = error.code;

*//1 - PERMISSION\_DENIED*

if(errorCode == 1){

console.log("Kullanıcı konumunu paylaşmayı reddetmiştir.")

}

*//2 - POSITION\_UNAVAILABLE*

else if(errorCode == 2){

console.log("Kullanıcının konumu tespit edilememiştir.");

}

*//3 - TIMEOUT*

else if(errorCode == 3){

console.log("Zaman aşımı meydana gelmiştir.");

}

*//0 - UNKNOWN\_ERROR*

else{

console.log("Bilinmeyen bir hata meydana gelmiştir.");

}

}

if(navigator.geolocation){

  navigator.geolocation.getCurrentPosition(successCallback, errorCallback);

}

Konum verileri elde edilirken getCurrentPosition() yerine watchPosition() metodu kullanılabilir. Bu metodun farkı konum bilgilerini güncelleyebilmesidir. Ziyaretçinin pozisyonu değiştiğinde konum bilgileri güncellenir.

Önceki örneğimiz üzerinde ufak bir değişiklik yapacağız.

Example

let watchId = navigator.geolocation.watchPosition(successCallback, errorCallback);

watchPosition() metodu geriye benzersiz bir değer döndürmektedir. Bu değer clearWatch() metodu ile birlikte konum güncellemelerini durdurmak için kullanılır.

Example

navigator.geolocation.clearWatch(watchId);