# WebStorage

**前言**

HTML5 的 web Storage 存储方式有两种：localStorage 和 sessionStorage。

这两种方式都是通过键值对保存数据，存取方便，不影响网站性能。他们的用法相同，存储时间不同。  
localStorage 的数据保存在本地硬件上，可以永久保存，可以手动调用api清除数据。sessionStorage 保存在 session 对象中，会在浏览器关闭时被清除。

web Storage 的大小在浏览器上是有限制的，不同浏览器大小会有区别，在主流浏览器中，大小约为 5M，用来存储普通数据其实已经足够。

**用法**

以 localStorage 为例，sessionStorage 用法一样：

**setItem**

保存数据：localStorage.setItem(key,value);

示例：

localStorage.setItem('name','Hello World');

当 key 相同时会覆盖之前的 value，用于修改数据。如果 value 为对象，需转为 json 字符串，否则你读取出来的将会是 [object Object]

**getItem**

读取数据：localStorage.getItem(key);

示例：

localStorage.getItem('name'); // Hello World

**removeItem**

删除单个数据：localStorage.removeItem(key);

示例：

localStorage.removeItem('name');

localStorage.getItem('name'); // null

删除 key 为 name 的数据后，loaclStorage 里已经获取不到该数据，则返回 null;

**clear**

删除所有数据：localStorage.clear();

示例：

localStorage.clear();

此时会把 localStorage 中的所有数据都删除。

**key**

得到某个索引的key：localStorage.key(index);  
示例：

localStorage.setItem('name1','Hello World');

localStorage.setItem('name2','Hello Linxin');

localStorage.key(1); // name2

获取到索引为 1 的 key，即 name2。

**构造函数**

在实际项目中，可能需要多次对 localStorage 进行操作，我们可以通过一个构造函数来更好的操作。

示例：

var localEvent = function (item) {

this.get = function () {

return localStorage.getItem(item);

}

this.set = function (val) {

localStorage.setItem(item, val);

}

this.remove = function () {

localStorage.removeItem(item);

}

this.clear = function () {

localStorage.clear();

}

}

// 使用new字符把构造函数实例化出多个对象

var local1 = new localEvent('name1');

var local2 = new localEvent('name2');

local1.set('Hello World');

local2.set('Hello Linxin');

local1.get(); // Hello World

local2.get(); // Hello Linxin

这里只是简单的演示，像我们平时在项目中可能要把对象存储起来，就需要在代码里做些处理。

**监听 storage 事件**

可以通过监听 window 对象的 storage 事件并指定其事件处理函数，当页面中对 localStorage 或 sessionStorage 进行修改时，则会触发对应的处理函数。

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

console.log('key='+e.key+',oldValue='+e.oldValue+',newValue='+e.newValue);

})

触发事件的时间对象（e 参数值）有几个属性：

* key : 键值。
* oldValue : 被修改前的值。
* newValue : 被修改后的值。
* url : 页面url。
* storageArea : 被修改的 storage 对象。

注意：在谷歌浏览器中，需要在不同标签页中修改 storage 才会触发该事件，即 网页A 监听该事件，在 网页B 中修改 localStorage，则 网页A 会触发事件函数。但是在 IE 中，在同个网页修改 localStorage 都会触发该事件。

**调试**

谷歌浏览器自带调试工具(chrome devtools)非常好用，可以用来调试 localStorage 和 sessionStorage。打开浏览器按f12调出调试工具，可以看到 Application ，点击打开可以看到左边栏有 Storage，包括了 localStorage、sessionStorage、IndexedDB等，选中我们要调试的网站域名，可以看到右边有对应的 key 和 value，可以通过右键进行编辑或删除等。

**兼容**

IE8 以上就兼容，但是比较特别，需要在服务器上打开的才支持，直接双击打开文件的 file:// 是不兼容的。

到了 IE11 才支持 file:// 下打开的，其他浏览器的支持程度都很高，包括在手机上的兼容。具体兼容可查看：<http://caniuse.com/#search=localstorage>

| **特性** | **Cookie** | **localStorage** | **sessionStorage** |
| --- | --- | --- | --- |
| 数据的生命期 | 一般由服务器生成，可设置失效时间。如果在浏览器端生成Cookie，默认是关闭浏览器后失效 | 除非被清除，否则永久保存 | 仅在当前会话下有效，关闭页面或浏览器后被清除 |
| 存放数据大小 | 4K左右 | 一般为5MB | |
| 与服务器端通信 | 每次都会携带在HTTP头中，如果使用cookie保存过多数据会带来性能问题 | 仅在客户端（即浏览器）中保存，不参与和服务器的通信 | |
| 易用性 | 需要程序员自己封装，源生的Cookie接口不友好 | 源生接口可以接受，亦可再次封装来对Object和Array有更好的支持 | |