



NeoSystems

# HTML5本地存储

武风龙

2011-3-29



HTML5本地存储 == Html5离线存储 ?

# 区别

**离线存储 (Offline Storage)** : 它实现的是文件的离线存储。

**本地存储 (Local Storage)** : 跟会话存储 (Session Storage) 一样同属于web的数据存储 (Web Storage)。

**Web SQL Database** : SQL操作的客户端数据库。

被称为“客户端存储”更合适，这样不至于跟Web Storage中的Local Storage概念冲突和混淆。

# 什么是HTML5 LOCALSTORAGE?

HTML5中介绍的Local Storage（有的地方把它叫做DOM Storage）是一种将web数据存储在用户本地的存储方式，这种形式的存储其实你早就接触过了，那就是cookie。那么为什么不直接用cookie好了，还要本地存储做什么？我们知道，html5带来更丰富的web表现形式，在web中实现大量富媒体应用的时候，对于数据存储的要求也相对提高，当然服务器的压力也相应变大，因此我们需要一种类似于cookie的存储方式，但它和cookie不大一样！

# LOCAL STORAGE和COOKIE的特点

- 我们需要更大的存储空间，本地存储默认有5MB，而cookie只能存4k的数据
- 和cookie一样都存储在客户端的
- 数据不会因为页面刷新或关闭等操作而改变
- 不会向服务器发出请求，cookie是会包含在每个HTTP请求中的

# 如何使用HTML5 LOCAL STORAGE?

Local Storage实现了Storage接口，该接口的原型如下：

```
interface Storage {  
  readonly attribute unsigned long length;  
  getter DOMString key(in unsigned long index);  
  getter any getItem(in DOMString key);  
  setter creator void setItem(in DOMString key, in any data);  
  deleter void removeItem(in DOMString key);  
  void clear();  
}
```

Local Storage是以键值对（key/value）的形式存储的，每个键值对称为一个项（item）；length属性返回项的数量；key(n)返回列表中第n个key的名字；getItem和setItem分别是取值和赋值的方法，如果要取值的key不存在返回null，如果赋值的key存在就覆盖，不存在就新建一个项；removeItem可以清除给定的key所对应的项，如果key不存在则什么都不做；clear会清除所有的项，如果列表本来就是空的就什么都不做

# HTML5本地存储

```
window.onload=function(){
    if(('localStorage' in window) && window['localStorage'] !== null){
        var name=localStorage.getItem('name');
        if(name===null){
            name=prompt("登录你的名字",'输入');
            if(name!==null){
                localStorage.setItem('name',name);

                document.getElementById('name').innerHTML=name;
            }
        }else{
            document.getElementById('name').innerHTML=name;
        }
    }else{
        alert('你的浏览器OUT了');
    }
}
function rmMem(){
    document.getElementById('name').innerHTML='你还没有登录! ';
    localStorage.removeItem('name');
    alert('成功退出登录! ');
}
```

# 多种表现形式

你可能还会看到这样的写法：

```
localStorage.setItem(1,'some values');
```

1这样不代表该项就是第一项，数字1会被转换为string，因为key的类型就是字符串。

```
localStorage.name='wanz';
```

```
localStorage['name']='wanz';
```

除了接口中那样通过getter跟setter的方式对本地存储的数据进行读写，我们还可以像对象属性访问一样读取与写入数据



# 本地存储事件监听

事件监听对于处理存储的异常还是必需的，先来看看 `StorageEvent` 接口的定义：

```
interface StorageEvent : Event {  
  readonly attribute DOMString key;  
  readonly attribute any oldValue;  
  readonly attribute any newValue;  
  readonly attribute DOMString url;  
  readonly attribute Storage storageArea;  
  void initStorageEvent(in DOMString typeArg, in boolean canBubbleArg, in boolean cancelableArg, in DOMString keyArg, in any oldValueArg, in any newValueArg, in DOMString urlArg, in Storage storageAreaArg);  
};
```

`initStorageEvent`可以不用管，它负责事件初始化；`key`返回的是被修改的`key`；`oldValue`和`newValue`分别返回`key`修改前和修改后的值；`url`返回修改该`key`的文档页面地址，有的浏览器可能会叫`uri`；`storageArea`返回受影响的`Storage`对象

# HTML5本地存储

一个简单的存储事件监听的方式可以像下面的代码这样：

```
1 addEventListener('storage', ev_handler, false);
```

不管是 `setItem()`, `removeItem()` 还是 `clear()` 操作，各个存储事件只有在存储内容真正发生变化时才会被触发