今天学习的内容

1:复习昨天重点知识

HTML5新特性

(1)新的语义标签

(2)增强型表单

(3)视频和音频

(4)Canvas绘图

(5)SVG绘图

(6)地理定位

navigator.geolocation{}

(7)拖放API

拖动源对象 dragstart/drag{0,0}/dragend

拖动目标对象 dragenter/dragover{阻止默认行为}/dragleave/drop

(8)WebWorker

(9)WebStorage

(10)WebSocket

2:作业

3:今天的目标

HTML5新特性--WebWorker

程序:Progam 指可以CPU执行的代码，存储在外存中.

进程:Process/Task 指程序被OS调入内存， 分配执行空间，

随时供CPU调用执行.

线程:Thread :线程进程内执行代码基本单位

进程和线程:

(1)进程是操作系分配内存基本单位

(2)线程是CPU执行代码的基本单位

(3)线程必须处理某个进程内部

(4)一个进程内必须至少有一个线程也可以有多个

(5)一个操作系统可能同时存在几千个线程，它们是"并发执行的"--宏观看同时执行，微观上看依次执行.

Chrome浏览器中的线程模型

一个Chrome进程内，至少有6个线程，可以"并发"向WEB服务器发起HTTP请求，以获得所需的资源

---资源请求线程

还有**一**个线程负责将所有**内容绘制**到浏览器页面中--UI主线程--不允许多线程同时绘图，防止内容布局错乱.

<button>按钮1</button>  
<script src="3.js">很耗时的JS任务</script>  
<button>按钮2</button>

现象:JS执行过程中，按钮1可见，按钮2不可见

原因:浏览器中执行代码只有一个线程--UI主线程

解析办法:创建新的线程，由它来执行耗时JS任务;

UI主线程继续执行后续的HTML渲染.

<button>按钮1</button>  
<script>

var w = new Worker("3.js");

</script>

<button>按钮2</button>

3:Worker线程缺陷

浏览器不允许Worker线程操作任何的DOM、BOM对象

原因:浏览器只允许UI主线程操作DOM/BOM，若多个线程同时都可以DOM/BOM，页面混乱.

所以:类以JQUERY脚本不同使用Worker来加载执行.

Worker线程可以给UI主线程发数据消息

UI主线程

var w = new Worker("6.js");

w.onmessage = function(e){

e.data;//接收数据

}

Worker线程

postMessage(strinMsg);

UI主线程发数据消息Worker

UI主线程

var w6 = new Worker("6.js");

w6.postMessage(stringMsg);

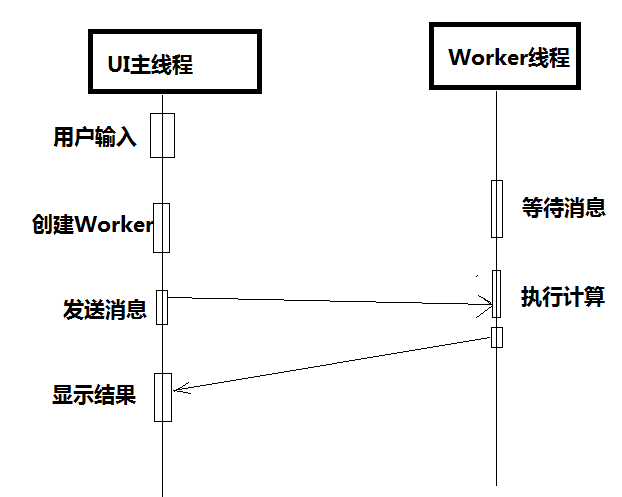
Worker线程

onmessage = function(e){e.data}

练习:在HTML中有一个INPUT,按钮"开始计算累加和",

点击按钮，创建一个Worker线程计算出用户输入数字

的累加和，下方一个div是计算结果.



|  |
| --- |
| 项目中Worker的使用场景  (1)只要js中有DOM&BOM就不能用Worker  (2)Worker适合于执行耗时的JS任务!如复杂计算,加密和解密，大数据统计，路径规则.... |

4:HTML5新特性---WebStorage

在浏览器中存储当前用户专有的数据，访问历史，内容定制，样式 定制...

在客户端存储数据可以使用技术

(1)Cookie技术: 浏览器兼容好，不能超过4KB,操作复杂

(2)Flash存储 依赖于Flash播放器

(3)H5 WebStorage 不能超过8MB，操作简单

(4)IndexDB 可存大量数据，还不是技术标准

|  |
| --- |
| Session:会话:浏览器从打开某个网站的一个页面开始，中间可能打开多个页面，直到关闭浏览器为止，整个过程称为"浏览器与web服务器的**一次会话**" |

WebStorage技术中，浏览器为用户提供了两个对象

(1)**sessionStorage 类数组对象，会话级数据存储;**

在浏览器进程所有分的内存中存储数据，可供此次会话中所有页面共同使用；浏览器一旦关闭了.

作用: 在同一个会话中所有页面共享数据:如登录用户名.

sessionStorage[key]=value //保存一个数据

sessionStorage.setItem(key,value); //保存一个数据

var v = sessionStorage[key] //读取一个数据

var v = sessionStorage.getItem(key);//读取一个数据

sessionStorage.removeItem(key); //删除一个数据

sessionStorage.clear(); //清除所有数据

sessionStorage.length; //数据数量

sessionStorage.key(i); //获取第i个key

练习:  
 创建index.html 右上角提示"请登录"超链接  
 创建login.html输入用户名，密码,提交按钮,用户点击  
 提交按钮提示 "登录成功" 3s后将自跳转回首页..  
 setTimeout+location.href  
 返回index.html 右上角提示"欢迎回来 xxx 退出登录"  
 创建logout.html,提示"您己退出登录",3s后自动跳转首页"  
 返回index.html,右上角提示"请登录" 超链接

**(2)localStorage:类数组对象，本地存储(跨会话存储)**  
在浏览器所有能管理外存(硬盘)中存储用户的浏览数据，  
可供此次会话以及后续的会话中的页面共同使用;即使  
浏览器关闭也不会消失--永久存在，  
作用:在当前客户端所对应的所有会话中共享数据.

localStorage[key]=value //保存一个数据

localStorage.setItem(key,value); //保存一个数据

var v = localStorage [key] //读取一个数据

var v = localStorage.getItem(key);//读取一个数据

localStorage.removeItem(key); //删除一个数据

localStorage.clear(); //清除所有数据

localStorage.length; //数据数量

localStorage.key(i); //获取第i个key

17:35--17:45

练习:创建index.html 有一个下拉菜单,"请选择您喜欢主题"

"蔚兰天空","芭比公主","黑暗主题", 各对应一个class名

.blue{background:#ddf,color:#33a}

.pink{background:#fdf,color:#a3a}

.dark{background:#333,color:#eee}

用户选中某个选项{select.onchange}为当前页面body使用指定的样式。

创建usercenter.html打开此页面即可应用index.html中的主题样式，

即使重启浏览器，直接访问返回index.html,也应该自动应用之前曾选的主题样式.

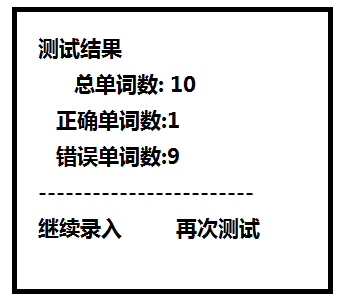
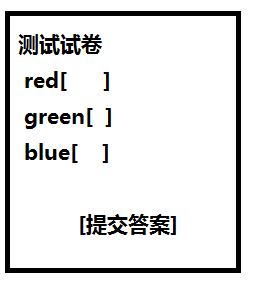
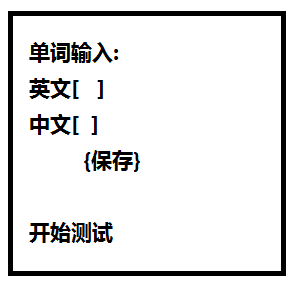
作业:单词测试系统

(1)用户可以在save.html中不停的录入新单词

(2)进入test.html开始测试，需要对之前录入过的所有单词测试

(3)提交答案后,在result.html中显示出测试成绩

提示:录入单词需要永就保存，而此次测试结果只需保存在当前会话中.



1`