**Html面试总结**

说明：面试或笔试（具体是面试公司情况而定）。

特别说明：本篇总结为个人心得总结，仅完美适用于作者本人。如有异议，请自行消化或删除文档。

作者：陶福荣

备注：格式如果错乱，请使用正版offic2013

版权所有，非作者本人同意，不得上传于网络

# 1、前端需要注意哪些SEO

## 1）什么是SEO

SEO是由英文Search Engine Optimization缩写而来，中文意思是“搜索引擎的优化”。SEO具体是指通过：网站结构调整、网站内容建设、网站代码优化、以及站外优化（网站站外推广、网站品牌建设等），使网站满足搜索引擎的收录排名需求，提高网站在搜索引擎中关键字的排名，从而吸引精准用户进入网站，获得免费流量，产生直接销售或品牌推广。

## 2）前端需要注意那些SEO

1）合理的title、description、keywords:搜索引擎对于三项的权重逐个减小；

title要强调重点,重要关键词出现不要超过两次而且要靠前，不同的页面要使用不同的title;

description把页面内容高度概括，长度合适，不可过分堆砌关键词，不同的页面要使用不同的description;

keywords列举出重要关键词即可;(description\keywords属于<meta>的属性);

2）重要的内容一定不要使用js输出：爬虫不会执行js获取内容；

3）如果图片不是单纯的装饰图片那必须添加alt;

4）重要内容的HTML放在前面：搜索引擎抓取HTML的顺序是从上而下，而且有的搜索引擎对搜索长度是由限制的；

为了避免重要内容被忽略，一定要往前放；

5）语义化HTML代码，符合W3C标准规范，可以让搜索引擎更加容易理解网页；【w3c标准：结构标准语言、表现标准语言、行为标准】

6）提高网站速度：网站加载速度是搜索引擎排序的重要指标；

7）尽量不要使用iframe标签：搜索引擎抓取不到；

# 2、<img/>的title和alt有什么区别

title可以在鼠标悬停时显示，且可以被搜索引擎搜索到。alt是用来替代图片不显示时显示的文字。

# HTTP的几种请求方法用途

1. GET方法

发送一个请求来取得服务器上的某一资源

2）POST方法

向URL指定的资源提交数据或附加新的数据

3）PUT方法

跟POST方法很像，也是想服务器提交数据。但是，它们之间有不同。PUT指定了资源在服务器上的位置，而POST没有

1. HEAD方法

只请求页面的首部

1. DELETE方法

删除服务器上的某资源

1. OPTIONS方法

它用于获取当前URL所支持的方法。如果请求成功，会有一个Allow的头包含类似“GET,POST”这样的信息

1. TRACE方法

TRACE方法被用于激发一个远程的，应用层的请求消息回路

1. CONNECT方法

把请求连接转换到透明的TCP/IP通道

# 4.从浏览器地址栏输入url到显示页面的步骤

面试题：浏览器输入一个地址www.baidu.com，到页面最终渲染完成，发生了什么？  
1）DNS解析

将域名解析为ip地址  
2）TCP连接  
 TCP三次握手：  
 第一次握手：由浏览器发起，服务器接受，我要发送请求了，你准备接受一下  
 第二次握手：由服务器发起，浏览器接受，我准备接受了，你放马过来  
 第三次握手：由浏览器发起，服务器接受，我马上要发了，你做好准备

3）发送请求  
 请求报文  
4）接受响应  
 响应报文  
5）渲染页面  
6）断开连接  
 TCP四次挥手：  
 第一次挥手：由服务器发起，浏览器接受，我的数据发送完毕了，你准备关闭接受吧  
 第二次挥手：由浏览器发起，服务器接受，我接受完了，我准备关闭了，你也准备关闭吧  
 第三次挥手：由浏览器发起，服务器接受，我准备断开连接了，你也准备吧  
 第四次挥手：由服务器发起，浏览器接受，我也准备断开连接了

# 5.如何进行网站性能优化

## 1）减少HTTP请求次数

频繁的请求服务器，会造成页面响应缓慢，数据加载缓慢的问题，若果并发量大的时候，很容易造成系统瘫痪。

在一次查询中，首次请求后台时，后台一次性将所需要的信息全部返回，前台进行缓存，后续所有的操作均解析前台缓存数据，从中取出页面操作所需要的信息然后再进行展现，有效减少请求后台服务器的次数。相对于程序和网络延迟来说，网络传输所耗费的时间，要比解析数据要慢的多，即使返回的数据量大，也不会比网络延迟的影响更大。

具体缓存的策略，可以是一个数组对象，一个列表对象，或一个HashMap对象等，也可以是JSON数据，或XML内容。总之，只要有利于缓存的数据结构，都可以用于数据的缓存。

从网络方面而言，减少数据的交互就意味着减少了网络带宽的压力，能够提供更大并发量的服务，减少网络传输的时间。

由于减少了请求次数，服务器的资源开销得到有效减少，后台服务器不用频繁的创建线程来处理外部的请求，由于操作系统创建和销毁线程的操作是非常消耗系统CPU等资源，因此减少请求次数也就大大减少了系统开销。

## 2）减少DNS访问

### （1）基础知识

DNS(Domain Name System)： 负责将域名URL转化为服务器主机IP。

DNS查找流程：首先查看浏览器缓存是否存在，不存在则访问本机DNS缓存，再不存在则访问本地DNS服务器。所以DNS也是开销，通常浏览器查找一个给定URL的IP地址要花费20-120ms，在DNS查找完成前，浏览器不能从host那里下载任何东西。

TTL(Time To Live)：表示查找返回的DNS记录包含的一个存活时间，过期则这个DNS记录将被抛弃。

### （2）影响DNS缓存的因素

服务器可以设置TTL值表示DNS记录的存活时间。本机DNS缓存将根据这个TTL值判断DNS记录什么时候被抛弃，这个TTL值一般都不会设置很大，主要是考虑到快速故障转移的问题。

浏览器DNS缓存也有自己的过期时间，这个时间是独立于本机DNS缓存的，相对也比较短，例如chrome只有1分钟左右。

浏览器DNS记录的数量也有限制，如果短时间内访问了大量不同域名的网站，则较早的DNS记录将被抛弃，必须重新查找。不过即使浏览器丢弃了DNS记录，操作系统的DNS缓存也有很大机率保留着该记录，这样可以避免通过网络查询而带来的延迟。

### （3）最佳实践

当客户端的DNS缓存为空时，DNS查找的数量与Web页面中唯一主机名的数量相等。所以减少唯一主机名的数量就可以减少DNS查找的数量。

然而减少唯一主机名的数量会潜在地减少页面中并行下载的数量，避免DNS查找降低了响应时间，但减少并行下载可能会增加响应时间。当页面的组件量比较多的时候，可以考虑将组件分别放到至少2-4个主机名，已获得最大收益。

## 3）减少对DOM的操作

### （1）基本原理

对DOM操作的代价是高昂的，这在网页应用中的通常是一个性能瓶颈。

在《高性能JavaScript》中这么比喻：“把DOM看成一个岛屿，把JavaScript(ECMAScript)看成另一个岛屿，两者之间以一座收费桥连接”。所以每次访问DOM都会教一个过桥费，而访问的次数越多，交的费用也就越多。所以一般建议尽量减少过桥次数。

### （2）解决办法

修改和访问DOM元素会造成页面的Repaint和Reflow，循环对DOM操作更是罪恶的行为。所以请合理的使用JavaScript变量储存内容，考虑大量DOM元素中循环的性能开销，在循环结束时一次性写入。

减少对DOM元素的查询和修改，查询时可将其赋值给局部变量。

## 4）减少请求文件的大小

缩减cookie的大小；

压缩合并css和js文件；

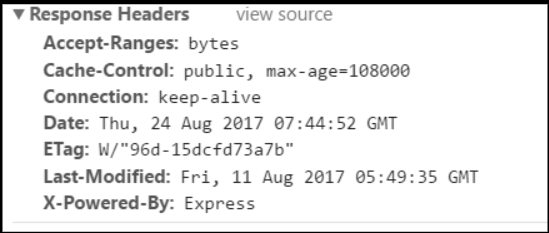
删除不必要的脚本；

压缩图片大小；

## 5）解决HTTP缓存

如果不懂HTTP缓存的请访问这个网站<http://www.cnblogs.com/chenqf/p/6386163.html>；

直接通过服务器配置，对访问静态资源进行缓存 ，如采用node的express框架的app.use(express.static(path.join(\_\_dirname, 'public'),{maxAge:1000\*60\*60\*30}));



在资源数据返回的响应报文中，会返回相应的数据和缓存规则（缓存有效期max-age=108000），然后浏览器会根据返回的缓存规则执行相应的缓存机制。当然，如果你留意的话，会发现有些缓存提示是from memory cache，有些又是from disk cache，说实话，其中原理本人也不甚清楚，据本人实测及猜测，数据进行缓存时，会优先将数据缓存到磁盘上，然后浏览器再通过判断文件类型，将图片/css的文件缓存在内存中。当重新打开页面后，由于内存被清空了，浏览器会先从磁盘上读取，再缓存在内存上，故再次刷新后，图片/css的文件直接从内存中读取，而其它文件的直接从磁盘上读取。

至于状态码200和304，出现304的原因的浏览器先访问服务器，服务器通过标识判断该文件没有修改，直接返回个304，然后浏览器通过状态码直接读取缓存（也就是对比缓存），200则是在数据未失效时，采用强制缓存，浏览器直接使用缓存中的资源。

## 6）重绘和重排

在理解重绘和重排之前，我们先回忆下网页的生成过程：输入网址 -> 加载html/css -> 将html转化成DOM树 -> 将css转化成CSSOM -> 结合两者生成一棵渲染树 -> 生成布局及将布局绘制在屏幕上，也就是人们常说的渲染。在这过程中，主要的耗时在于渲染这一步，主要存在以下几点原因：

（1）浏览器在代码中发现一个标签引用了一张图片，向服务器发出请求。此时浏览器不会等到图片下载完，而是继续渲染后面的代码，当图片返回后，因影响到后面的排布，浏览器需要重新渲染；

（2）服务器返回图片文件，由于图片占用了一定面积，影响了后面段落的排布，因此浏览器需要回过头来重新渲染这部分代码；

（3）浏览器发现了一个包含一行Javascript代码的标签，会立即运行该脚本并阻塞文档解析直到脚本执行完。（web的模式是同步的，开发者希望解析到一个script标签时立即解析执行脚本，并阻塞文档的解析直到脚本执行完。如果脚本是外引的，则网络必须先请求到这个资源——这个过程也是同步的，会阻塞文档的解析直到资源被请求到。因此在HTML5中，指明了一个新属性defer，以使其不阻塞文档解析，并在文档解析完成后再执行，也增加了标记脚本为异步的选项，以使脚本的解析执行使用另一个线程。）

（4）等到执行javascript脚本后，因为必要的一些dom操作，导致DOM树发生变化，如某些节点添加删除隐藏等，会导致浏览器重新渲染这部分代码。

（5）过多的使用table布局。table及其内部元可能需要多次计算才能确定好其在渲染树中节点的属性，通常要花3倍于同等元素的时间。

重绘就是一个元素外观发生，如改变visibility、outline、背景色等属性变化，导致浏览器重新绘制受影响的部分到屏幕上的一种行为。而重排就是由于DOM树结构/DOM节点元素属性发生变化导致浏览器因渲染树部分失效而重新构造渲染树的一种行为。重绘不一定需要重新布局，也就不一定会引起重排，但重排一定会伴随重绘。

**以下几点会导致浏览器发生重排：**

1、添加或者删除可见的DOM元素

2、元素位置改变

3、元素尺寸改变

4、元素内容改变（例如：一个文本被另一个不同尺寸的图片替代）

5、页面渲染初始化（这个无法避免）

6、浏览器窗口尺寸改变

如今浏览器大部分已经针对重排做了优化，通过队列优化修改并批量执行优化重排过程，一次性完成，但某些DOM操作会导致浏览器强制刷新队伍并要求计划任务立即执行，如获取元素的布局信息操作。

故，在开发中应当尽量减少重排次数和缩小重排的影响范围。

1. 将多次改变样式属性的操作合并成一次操作，或者先集合操作获取属性，再集中操作修改属性

2. 将需要多次重排的元素，position属性设为absolute或fixed，这样此元素就脱离了文档流，它的变化不会影响到其他元素。例如有动画效果的元素就最好设置为绝对定位。

3. 在内存中多次操作节点，完成后再添加到文档中去。如先生成HTML字符串，再整体赋值。

4. 由于display属性为none的元素不在渲染树中，对隐藏的元素操作不会引发其他元素的重排。如果要对一个元素进行复杂的操作时，可以先隐藏它，操作完成后再显示。

5. 在需要经常获取那些引起浏览器重排的属性值时，要缓存到变量。

# HTTP状态码及其含义

## 1XX：信息状态码

100 Continue 继续，一般在发送post请求时，已发送了http header之后服务端将返回此信息，表示确认，之后发送具体参数信息

## 2）2XX：成功状态码

1. 正常返回信息
2. 201 Created 请求成功并且服务器创建了新的资源

202 Accepted 服务器已接受请求，但尚未处理

## 3）3XX：重定向

301 Moved Permanently 请求的网页已永久移动到新位置。

302 Found 临时性重定向。

303 See Other 临时性重定向，且总是使用 GET 请求新的 URI。

304 Not Modified 自从上次请求后，请求的网页未修改过。

## 4）4XX：客户端错误

400 Bad Request 服务器无法理解请求的格式，客户端不应当尝试再次使用相同的内容发起请求。

401 Unauthorized 请求未授权。

403 Forbidden 禁止访问。

404 Not Found 找不到如何与 URI 相匹配的资源。

## 5）5XX: 服务器错误

500 Internal Server Error 最常见的服务器端错误。

503 Service Unavailable 服务器端暂时无法处理请求（可能是过载或维护）。

## 6）请求方式

（1）head：类似于get请求，只不过返回的响应中没有具体的内容，用户获取报头；

（2）options：允许客户端查看服务器的性能，比如说服务器支持的请求方式等等。

# 7.语义化的理解

## 基础知识

1）<hgroup></hgroup>

代表网页或者section的标体  
2）<header></header>

代表网页或section的页眉  
3）<nav></nav>

代表页面的导航链接区域  
4）<section></section>

代表文档中的 节 或 段，段可以是指一篇文章里按照主题的分段；节可以是指一个页面里的分组。

**section使用注意：**  
 section不是一般意义上的容器元素，如果想作为样式展示和脚本的便利，可以用div。  
 article、nav、aside可以理解为特殊的section，  
 所以如果可以用article、nav、aside就不要用section，没实际意义的就用div   
5）<footer></footer>

代表网页或者section的页脚  
6）<article></article>

article元素最容易跟section和div容易混淆，其实article代表一个在文档，页面或者网站中自成一体的内容

**article使用注意：**  
 独立文章：用article  
 单独的模块：用section  
 没有语义的：用div  
7）<aside></aside>

aside元素被包含在article元素中作为主要内容的附属信息部分，其中的内容可以是与当前文章有关的相关资料、标签、名次解释等。

**aside使用总结：**  
 aside在article内表示主要内容的附属信息；  
 在article之外则可做侧边栏  
 如果是广告，其他日志链接或者其他分类导航也可以用

## 语义化的作用

1）用正确的标签做正确的事情！

2）html语义化就是让页面的内容结构化，便于对浏览器、搜索引擎解析；

3）在没有样式CSS情况下也以一种文档格式显示，并且是容易阅读的。

4）搜索引擎的爬虫依赖于标记来确定上下文和各个关键字的权重，利于SEO。

5）使阅读源代码的人对网站更容易将网站分块，便于阅读维护理解

# 8.介绍一下你对浏览器内核的理解

## 1）浏览器内核种类

Trident (IE，360，搜狗浏览器【360和搜狗为双核trident+webkit】)

　　　Gecko(Netscape6及以上版本，firefox)

　　　Webkit(Chorme，Safari)

　　　Blink(Chorme[2013年开始使用]，Opera)

Presto内核：Opera

## 2）内核的组成

1.html,css文档解析模块 : 负责页面文本的解析  
2.dom/css模块 : 负责dom/css在内存中的相关处理布局和渲染模块 : 负责页面的布局和效果的绘3.制布局和渲染模块 : 负责页面的布局和效果的绘制  
4.定时器模块 : 负责定时器的管理  
5.网络请求模块 : 负责服务器请求(常规/Ajax)  
6.事件响应模块 : 负责事件的管理

# 9.Html5新特性

**HTML5 现在已经不是SGML的子集，主要是关于图像，位置，存储，多任务等功能的增加**

1）绘画 canvas

2）用于媒介回放的 video 和 audio 元素

3）本地离线存储localStorage长期存储数据，浏览器关闭后数据不丢失

4)sessionStorage 的数据在浏览器关闭后自动删除

5)语意化更好的内容元素，比如article、footer、header、nav、section

6)表单控件，calendar、date、time、email、url、search

7)新的技术webworker, websocket, Geolocation

**移除的元素：**

1)纯表现的元素：basefont，big，center，font, s，strike，tt，u`

2)对可用性产生负面影响的元素：frame，frameset，noframes

**支持HTML5新标签：**

1)IE8/IE7/IE6支持通过document.createElement方法产生的标签

2)可以利用这一特性让这些浏览器支持HTML5新标签

3)浏览器支持新标签后，还需要添加标签默认的样式

4)当然也可以直接使用成熟的框架、比如html5shim

# ****10、Doctype作用? 严格模式与混杂模式如何区分？它们有何意义?****

1）<!Doctype>声明位于文档中的最前面，处于<html>标签之前。告知浏览器的解析器，用什么文档类型规范来解析这个文档。  
 2）严格模式（use strice）的排版和JS运作模式是以该浏览器支持的最高标准运行。  
 3）在混杂模式中，页面以宽松的向后兼容的方式显示。模拟老式浏览器的行为以防止站点无法工作。  
 4）DOCTYPE不存在或格式不正确会导致文档以混杂模式呈现。

# **11、HTML5的离线储存怎么使**用，工作原理能不能解释一下？

在用户没有与因特网连接时，可以正常访问站点或应用，在用户与因特网连接时，更新用户机器上的缓存文件

**原理：**

HTML5的离线存储是基于一个新建的.appcache文件的缓存机制(不是存储技术)，通过这个文件上的解析清单离线存储资源，这些资源就会像cookie一样被存储了下来。之后当网络在处于离线状态下时，浏览器会通过被离线存储的数据进行页面展示

**如何使用：**

页面头部像下面一样加入一个manifest的属性；

<!DOCTYPE HTML>

<html manifest = "cache.manifest">

...

</html>

在cache.manifest文件的编写离线存储的资源

CACHE MANIFEST

#v0.1

CACHE:

js/index.js

css/index.css

NETWORK:

images/logo.png

FALLBACK:

\*.html /404.html /\* / /404.html 或 /html/ /404.html 也可\*/

在离线状态时，操作window.applicationCache进行需求实现

以#号开头的是注释，一般会在第二行写个版本号，用来在缓存的文件更新时，更新manifest以实现浏览器重新下载新的文件，可以是版本号，时间戳或md5码等。

# 12、浏览器是怎么对HTML5的离线储存资源进行管理和加载的

在线的情况下，浏览器发现html头部有manifest属性，它会请求manifest文件，如果是第一次访问app，那么浏览器就会根据manifest文件的内容下载相应的资源并且进行离线存储。如果已经访问过app并且资源已经离线存储了，那么浏览器就会使用离线的资源加载页面，然后浏览器会对比新的manifest文件与旧的manifest文件，如果文件没有发生改变，就不做任何操作，如果文件改变了，那么就会重新下载文件中的资源并进行离线存储。

离线的情况下，浏览器就直接使用离线存储的资源。

# 13、请描述一下 cookies，sessionStorage 和 localStorage 的区别？

cookie是网站为了标示用户身份而储存在用户本地终端（Client Side）上的数据（通常经过加密）

cookie数据始终在同源的http请求中携带（即使不需要），记会在浏览器和服务器间来回传递

sessionStorage和localStorage不会自动把数据发给服务器，仅在本地保存

## 1）存储大小：

cookie数据大小不能超过4k；

sessionStorage和localStorage虽然也有存储大小的限制，但比cookie大得多，可以达到5M或更大；

## 2）有期时间：

localStorage 存储持久数据，浏览器关闭后数据不丢失除非主动删除数据

sessionStorage 数据在当前浏览器窗口关闭后自动删除

cookie 设置的cookie过期时间之前一直有效，即使窗口或浏览器关闭

## 3）拓展（两个页面相互通信）

### （1）storage事件

Storage 对象发生变化时（即创建/更新/删除数据项时，Storage.clear() 方法至多触发一次该事件）会触发；在同一个页面内发生的改变不会起作用，在相同域名下的其他页面发生的改变才会起作用。(修改的页面不会触发事件，与它共享的页面会触发事件)  
 key : 修改或删除的key值，如果调用clear(),为null  
 newValue : 新设置的值，如果调用clear(),为null  
 oldValue : 调用改变前的value值,如果调用clear(),为null  
 url : 触发该脚本变化的文档的url  
 storageArea : 当前的storage对象

### （2）localStorage事件

**方法：**

添加：setItem(key,value);

获取：getItem(key)

删除： removeItem(key)

清除： clear();

**注意**：存储的时候只能存json数据

### （3）实现两个页面通信的代码

首先通过localStorage事件的setItem方法将数据存储到本地；

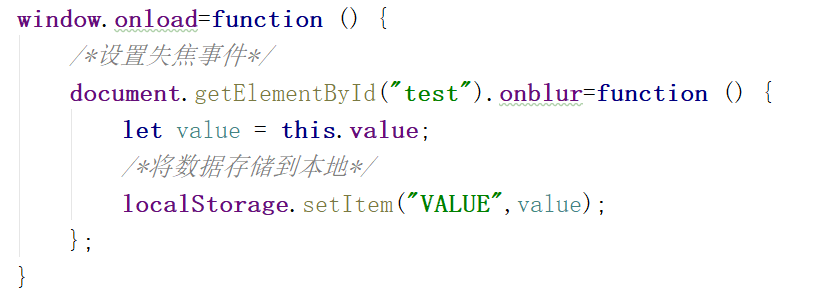


图1 将数据存储在本地

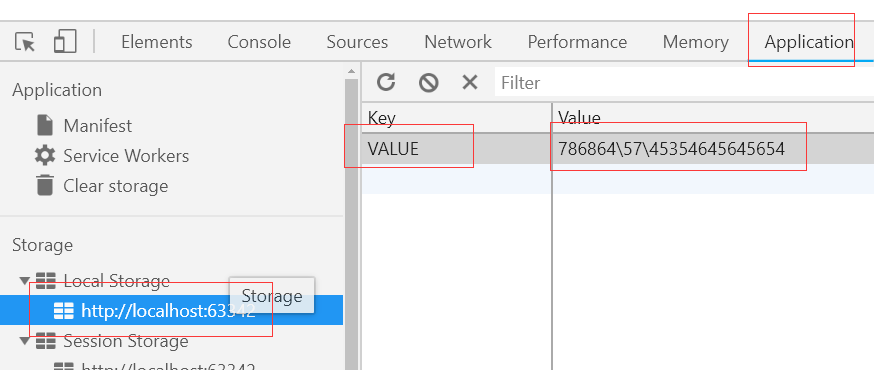


图2 浏览器界面

可以看到将数据存储在了本地。下面的例子我们可以看到，可以通过setitem和getitem来存储和获取本地数据，且不同页面都可以读取7到。

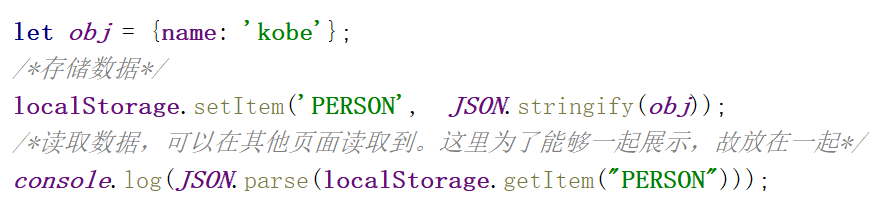


图3 存储及读取

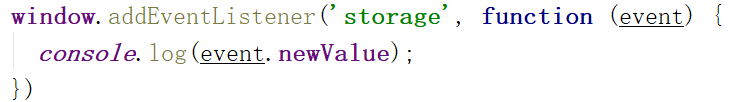


图4 storage监听改变存储值

通过图4和图1的代码我们可以在两个页面形成一个改变存储的页面和一个监听改变存储的页面。当在图1的text中输入文字改变值时，图4中的storage事件中就可以监听到新改变的值newValue，从而打印出来。

总得来说就是图3是单独的利用单一的localStorage方法进行存储和读取，而图1和图4是利用localStorage改变存储数据，然后再利用storage事件的newValue来监听改变的值；

### （4）cookie作用说明补充

保存用户登录状态。例如将用户id存储于一个cookie内，这样当用户下次访问该页面时就不需要重新登录了，现在很多论坛和社区都提供这样的功能。 cookie还可以设置过期时间，当超过时间期限后，cookie就会自动消失。因此，系统往往可以提示用户保持登录状态的时间：常见选项有一个月、三个 月、一年等。

跟踪用户行为。例如一个天气预报网站，能够根据用户选择的地区显示当地的天气情况。如果每次都需要选择所在地是烦琐的，当利用了 cookie后就会显得很人性化了，系统能够记住上一次访问的地区，当下次再打开该页面时，它就会自动显示上次用户所在地区的天气情况。因为一切都是在后台完成，所以这样的页面就像为某个用户所定制的一样，使用起来非常方便

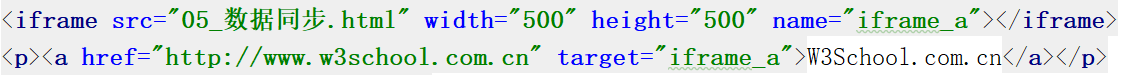
定制页面。如果网站提供了换肤或更换布局的功能，那么可以使用cookie来记录用户的选项，例如：背景色、分辨率等。当用户下次访问时，仍然可以保存上一次访问的界面风格。

# 14、 iframe有那些缺点？

## 1）什么是iframe？

标签：<iframe src="/example/html/demo\_iframe.html" width="200" height="200"></iframe>

实质就是在iframe内加一个src引入其他的地址或html文件，在iframe标签规定一个内联框架。可以在当前html文件中嵌入另一个html文件。



上面的代码是用来定义点击p标签点击跳转，设置target跳转目标定义name为iframe\_a的iframe标签。



Iframe标签示例

## 缺点

1.iframe会阻塞主页面的Onload事件

2.搜索引擎的检索程序无法解读这种页面，不利于SEO

3.iframe和主页面共享连接池，而浏览器对相同域的连接有限制，所以会影响页面的并行加载

4.使用iframe之前需要考虑这两个缺点。如果需要使用iframe，最好是通过javascript动态给iframe添加src属性值，这样可以绕开以上两个问题

# 15、WEB标准以及W3C标准是什么?

## 1）对于结构要求

标签规范可以提高搜索引擎对页面的抓取效率，对SEO很有帮助

1.标签字母要小写

2.标签要闭合

3.标签不允许随意嵌套

## 2）对于css和js来说

1.尽量使用外链css样式表和js脚本。是结构、表现和行为分为三块，符合规范。同时提高页面渲染速度，提高用户的体验。

2.样式尽量少用行间样式表，使结构与表现分离，标签的id和class等属性命名要做到见文知义，标签越少，加载越快，用户体验提高，代码维护简单，便于改版

3.不需要变动页面内容，便可提供打印版本而不需要复制内容，提高网站易用性。

# 16、xhtml和html有什么区别?

## 1）功能上的差别

主要是XHTML可兼容各大浏览器、手机以及PDA，并且浏览器也能快速正确地编译网页

## 2）书写习惯的差别

XHTML 元素：所有标签都必须小写、标签必须成双成对、标签顺序必须正确、所有属性都必须使用双引号、不允许使用target=“\_blank”;

# 17、 行内元素有哪些？块级元素有哪些？ 空(void)元素有那些？行内元素和块级元素有什么区别？

行内元素有：a span img input select strong

块级元素有：div ul ol li dl dt dd h1 h2 h3 h4…p

空元素：<br> <hr> <img> <input> <link> <meta>

行内元素不可以设置宽高，不独占一行

块级元素可以设置宽高，独占一行

# 18、HTML全局属性有哪些？

class:为元素设置类标识

data-\*: 为元素增加自定义属性

draggable: 设置元素是否可拖拽

id: 元素id，文档内唯一

lang: 元素内容的的语言

style: 行内css样式

title: 元素相关的建议信息

# 19、实现不使用 border 画出1px高的线，在不同浏览器的标准模式与怪异模式下都能保持一致的效果。



# 20.、http和https

https的SSL加密是在传输层实现的。

## 1）http和https的基本概念

http: 超文本传输协议，是互联网上应用最为广泛的一种网络协议，是一个客户端和服务器端请求和应答的标准（TCP），用于从WWW服务器传输超文本到本地浏览器的传输协议，它可以使浏览器更加高效，使网络传输减少。

https: 是以安全为目标的HTTP通道，简单讲是HTTP的安全版，即HTTP下加入SSL层，HTTPS的安全基础是SSL，因此加密的详细内容就需要SSL。

https协议的主要作用是：建立一个信息安全通道，来确保数组的传输，确保网站的真实性。

## 2）http和https的区别？

http传输的数据都是未加密的，也就是明文的，网景公司设置了SSL协议来对http协议传输的数据进行加密处理，简单来说https协议是由http和ssl协议构建的可进行加密传输和身份认证的网络协议，比http协议的安全性更高。 主要的区别如下：

* Https协议需要ca证书，费用较高。
* http是超文本传输协议，信息是明文传输，https则是具有安全性的ssl加密传输协议。
* 使用不同的链接方式，端口也不同，一般而言，http协议的端口为80，https的端口为443
* http的连接很简单，是无状态的；HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议，比http协议安全。

## 3）https协议的工作原理

客户端在使用HTTPS方式与Web服务器通信时有以下几个步骤，如图所示。

* 客户使用https url访问服务器，则要求web 服务器建立ssl链接。
* web服务器接收到客户端的请求之后，会将网站的证书（证书中包含了公钥），返回或者说传输给客户端。
* 客户端和web服务器端开始协商SSL链接的安全等级，也就是加密等级。
* 客户端浏览器通过双方协商一致的安全等级，建立会话密钥，然后通过网站的公钥来加密会话密钥，并传送给网站。
* web服务器通过自己的私钥解密出会话密钥。
* web服务器通过会话密钥加密与客户端之间的通信。

## 4）https协议的优点

* 使用HTTPS协议可认证用户和服务器，确保数据发送到正确的客户机和服务器；
* HTTPS协议是由SSL+HTTP协议构建的可进行加密传输、身份认证的网络协议，要比http协议安全，可防止数据在传输过程中不被窃取、改变，确保数据的完整性。
* HTTPS是现行架构下最安全的解决方案，虽然不是绝对安全，但它大幅增加了中间人攻击的成本。
* 谷歌曾在2014年8月份调整搜索引擎算法，并称“比起同等HTTP网站，采用HTTPS加密的网站在搜索结果中的排名将会更高”。

## 5）https协议的缺点

* https握手阶段比较费时，会使页面加载时间延长50%，增加10%~20%的耗电。
* https缓存不如http高效，会增加数据开销。
* SSL证书也需要钱，功能越强大的证书费用越高。
* SSL证书需要绑定IP，不能再同一个ip上绑定多个域名，ipv4资源支持不了这种消耗。

# 21、TCP和UDP的区别

不知道UDP，直接百度即可；尽量不要照搬套路，按照自己的理解简单描述（加分项）；

1）TCP是面向连接的，udp是无连接的即发送数据前不需要先建立链接。

2）TCP提供可靠的服务。也就是说，通过TCP连接传送的数据，无差错，不丢失，不重复，且按序到达;UDP尽最大努力交付，即不保证可靠交付。 并且因为tcp可靠，面向连接，不会丢失数据因此适合大数据量的交换。

3）TCP是面向字节流，UDP面向报文，并且网络出现拥塞不会使得发送速率降低（因此会出现丢包，对实时的应用比如IP电话和视频会议等）。

4）TCP只能是1对1的，UDP支持1对1,1对多。

5）TCP的首部较大为20字节，而UDP只有8字节。

6）TCP是面向连接的可靠性传输，而UDP是不可靠的。

# 22. HTML5 WebSocket

WebSocket是HTML5开始提供的一种在单个 TCP 连接上进行全双工通讯的协议。

在WebSocket API中，浏览器和服务器只需要做一个握手的动作【握手阶段与HTTP是相同的】，然后，浏览器和服务器之间就形成了一条快速通道。两者之间就直接可以数据互相传送。

浏览器通过 JavaScript 向服务器发出建立 WebSocket 连接的请求，连接建立以后，客户端和服务器端就可以通过 TCP 连接直接交换数据。

当你获取 WebSocket 连接后，你可以通过 send() 方法来向服务器发送数据，并通过 onmessage 事件来接收服务器返回的数据。

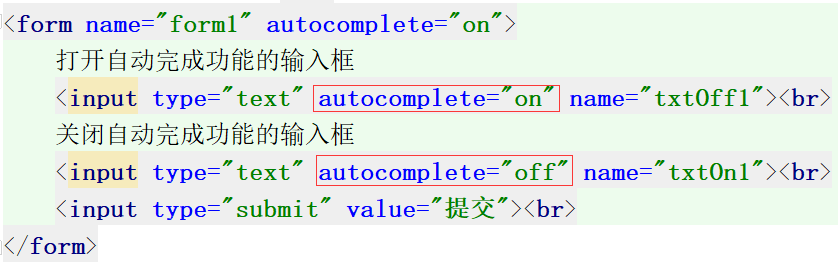
以下 API 用于创建 WebSocket 对象。

var Socket = new WebSocket(url, [protocol] );

具体请访问这个网站：<http://www.runoob.com/html/html5-websocket.html>

# 23、HTML5的form如何关闭自动完成功能

自动关闭功能就是用户输入提交之后，下次输入的使用搜索框不提示已经输入过的信息。



实质就是在对应的表单尚设置autocomplete=on或者autocomplete=off 来保证开启或者关闭；

## 附加：html5表单新特性

