目录

[3.XHTML和HTML的区别 2](#_Toc80197302)

[1） 标签必须闭合/ 2）标签名必须小写 3）标签元素必须正确嵌套 4）必须有根元素 2](#_Toc80197303)

[5)属性值必加引号； 2](#_Toc80197304)

[5.使用 data- 属性的好处是什么？ 2](#_Toc80197305)

[8.为什么通常推荐将 CSS 的link标签放置在head标签之间，而将script标签放置在 body闭合标签之前？ 2](#_Toc80197306)

[9.什么是渐进式渲染 (progressive rendering)？ 2](#_Toc80197307)

[10.HMTL5新标签 3](#_Toc80197308)

[11.<meta>标签 4](#_Toc80197309)

[12.条件注释 5](#_Toc80197310)

[13.src和href的区别 5](#_Toc80197311)

[5、很多网站不常用table  iframe这两个元素，知道原因吗？ 5](#_Toc80197312)

[6、jpg和png格式的图片有什么区别？ 5](#_Toc80197313)

[7、请用html知识解决seo(搜索引擎优化)优化问题 6](#_Toc80197314)

[9 简述一下你对HTML语义化的理解？ 6](#_Toc80197315)

[8、常用浏览器内核都是什么？ 6](#_Toc80197316)

[10、a标签在新窗口打开链接怎么加属性？ 6](#_Toc80197317)

[12、form标签上定义请求类型的是哪个属性？定义请求地址的是哪个属性？ 6](#_Toc80197318)

[13页面导入样式时，使用link和@import有什么区别？ 6](#_Toc80197319)

[14对浏览器内核的理解？ 7](#_Toc80197320)

[10 HTML5的文件离线储存怎么使用，工作原理是什么？ 7](#_Toc80197321)

[12 iframe框架有那些优缺点？ 7](#_Toc80197322)

[15 如何实现浏览器内多个标签页之间的通信? 7](#_Toc80197323)

[16 页面可见性（Page Visibility）API 可以有哪些用途？ 7](#_Toc80197324)

[18 实现不使用 border 画出1px高的线，在不同浏览器的Quirks mode和CSS Compat模式下都能保持同一效果 8](#_Toc80197325)

[20 title与h1的区别、b与strong的区别、i与em的区别？ 8](#_Toc80197326)

[21 元素的alt和title有什么异同？ 8](#_Toc80197327)

[5、解释一下utf-8和GBK和ISISO8859-2 字符集 8](#_Toc80197328)

[12、关于置换元素和不可替换元素 8](#_Toc80197329)

[为什么替换元素可设置宽高 9](#_Toc80197330)

[13、HTML元素的显示优先级 9](#_Toc80197331)

[19、关于比较Ajax与Flash的优缺点 9](#_Toc80197332)

[21、常见的浏览器端存储的方式 9](#_Toc80197333)

[method 属性 10](#_Toc80197334)

# 3.XHTML和HTML的区别

XHTML更严格. 文档顶部doctype声明不同

1. 标签必须闭合/ 2）标签名必须小写 3）标签元素必须正确嵌套 4）必须有根元素

5)属性值必加引号；

# 5.使用 data- 属性的好处是什么？

data-\*是为前端开发者提供自定义的属性，这些属性集可以通过对象的dataset属性获取，不支持该属性的浏览器可以通过 getAttribute方法获取。

PS：需要注意的是，data-之后的以连字符分割的多个单词组成的属性，获取的时候使用驼峰风格

例子：<div id="myID" data-my-attr = “test”></div>，获取属性值

使用dataset，getElementById("myID").dataset.myAttr

使用getAttribute方法，getElementById("myID").getAttribute('data-my-attr')

# 8.为什么通常推荐将 CSS 的link标签放置在head标签之间，而将script标签放置在 body闭合标签之前？

<link>放置在<head></head>是因为浏览器要先渲染页面呈现给用户，在渲染时需要构建dom树(html标签内容)和render渲染树(css样式)，才能完整呈现，所以放在头部优先加载。

而JS脚本文件比较大，且一般是后期JS引擎运行，渲染引擎会将控制权交给JS引擎而停止渲染，如果JS文件较大，会导致长时间白屏，影响用户体验，所以才会有JS<script>放在</body>之前。或者上一个问题中提到的优化加载的方式。

# 9.什么是渐进式渲染 (progressive rendering)？

第一次只加载一个页面及其所需文件,页面第一次呈现速度稍快。但是每一次加载的速度基本相同的. 可用缓存进行优化 用户在网站中,停留时时间较长,点击较多时, 就会体现出劣势.

类似懒加载，一开始先加载首屏显示的内容，之后再随着时间或者滚动页面才进行后面的加载

懒加载和预加载是常用的web优化的手段。  
懒加载：(延迟加载)，延迟加载网络资源或符合某些条件时才加载资源。常见的就是图片延时加载。  
懒加载的意义：懒加载的主要目的是作为服务器前端的优化，减少请求数或延迟请求数。  
懒惰实现方式：  
1.第一种是纯粹的延迟加载，使用setTimeOut或setInterval进行加载延迟.  
2.第二种是条件加载，符合某些条件，或触发了某些事件才开始异步下载。  
3.第三种是可视区加载，即仅加载用户可以看到的区域，这个主要由监控滚动条来实现，一般会在距用户看到某图片前一定距离遍开始加载，这样能保证用户拉下时正好能看到图片。

预加载：提前加载图片，当用户需要查看时可直接从本地缓存中渲染。

两种技术的本质：两者的行为是相反的，一个是提前加载，一个是迟缓甚至不加载。

懒加载对服务器前端有一定的缓解压力作用，预加载则会增加服务器前端压力。预加载应用如广告弹窗等。

# 10.HMTL5新标签

**HTML5新增的元素**

增加了header，footer，nav,aside,section等语义化标签

在表单方面，为了增强表单，为input增加了color，emial,data ,range等类型

在存储方面，提供了sessionStorage，localStorage,和离线存储，通过这些存储方式方便数据在客户端的存储和获取

在多媒体方面规定了音频和视频元素audio和vedio

另外还有地理定位，canvas画布，拖放，多线程编程的web worker和websocket协议

1）新的特殊内容元素，比如 article、footer、header、nav、section

2）新的表单控件，比如 calendar、date、time、email、url、search

3）用于媒介回放的 video 和 audio 元素

4）用于绘画的 canvas 元素

5）对本地离线存储的更好的支持

6）浏览器异步的webWorker

HTML块元素： div  p  ul  li  table  h1 h2 h3 ... h6  form 等

常用的内联元素有：

<a>、<span>、<br>、<i>、<em>、<strong>、<label>、<q>、<var>、<cite>、<code>

常用的内联块状元素有：

<img>、<input>

1、独占一行

2、高度、宽度、行高以及顶和底边距可设置。

3、元素宽度在不设置的情况下，是它本身父容器的100%（和父元素的宽度一致）

行内元素

在html中，<span>、<a>、<label>、<input>、<strong> 和<em>就是典型的行内元素（inline）元素。display:inline

1、和其他元素都在一行上；

2、元素的高度、宽度、行高及顶部和底部边距不可设置；宽度就是包含文字或图片的宽度。

内联块状元素

同时具备内联元素、块状元素的特点，<img>、<input>

1、和其他元素都在一行上；

2、高度、宽度、行高以及顶和底边距都可设置

哪个元素可以自闭合（空元素）：input img br hr meta link。。。

1.一般标签属性

id : id标识

style : 行内样式

href : a超链接地址,link外链css路径

src : 图片路径,script外链js路径

alt : 图片替换文字

2. 表单控件属性

name : 控件名字

type : 控件类型

value : 控件值

outline : 轮廓

# 11.<meta>标签

提供了 HTML 文档的元数据。META元素通常用于指定网页的描述，关键词，文件的最后修改时间，作者及其他元数据。

元数据可以被使用浏览器（如何显示内容或重新加载页面），搜索引擎（关键词），或其他 Web 服务调用。

注意点：

<meta> 标签通常位于 <head> 区域内。

如果没有提供 name 属性，那么名称/值对中的名称会采用 http-equiv 属性的值。

# 12.条件注释

常用注释 <!--xxx -->

条件注释

1. <!--[if !IE]><!--> 除IE外都可识别 <!--<![endif]-->

2. <!--[if IE]> 所有的IE可识别 <![endif]-->

# 13.src和href的区别

src

source的简写，【引用一个资源】，用来代替本身的内容。如img、script、iframe。如果不写src，本身没有内容，script会不存在脚本代码，img会显示x，iframe显示空白页。例如<script type="text/javascript" src="b.js">alert(1)</script>，b.js:alert(2); 标准的浏览器里会显示2，因为内容被替代；另一方面，当浏览器解析到这一句的时候暂停其他资源的下载处理，直至将该资源加载，编译，执行完，这也是为什么要把js放置在底部而不是头部。

在HTML body部分中的JavaScripts会在页面加载的时候被执行。放在body部分的脚本通常被用来生成页面的内容。这样不会出现大面积空白  
在HTML head部分中的JavaScripts会在被调用的时候才执行。当你把脚本放在head部分中时，可以保证脚本在任何调用之前被加载。

* 在head中时，所代表的functions只加载而不执行，执行是在某一事件触发后才开始。
* 在body中时，直接加载并执行

href

Hypertext Reference的缩写，表示“超链接”， 建立一个“通道”，让当前标签能够链接到目标地址上，并不是为了把目标拉进页面本身内来替换本身的内容，例如a标签的href不会被当做一个资源替换自身，link标签本身并不包含实际的元素意义来做内容，需要rel明确的表示【被连接的文档是做什么的】。例如：<link href="head.css" rel="stylesheet"/>，浏览器解析到这一句的时候会识别该文档为css文件，会下载并不会停止当前文档处理，这也是为什么建议使用link方式来加载css而不是使用@import。

# 5、很多网站不常用table  iframe这两个元素，知道原因吗？

答：浏览器页面渲染的时候是从上至下的，而table 和 iframe 这两种元素会改变这样渲染规则，等待自己元素内的内容加载完才整体渲染。用户不友好

# 6、jpg和png格式的图片有什么区别？

答： jpg有损压缩，png无损，相同图片jpg体积小。一些官网的banner图一般很大，适合jpg类型。但png分8位的和24位的，8位的体积会小很多，但在某些浏览器下8位的png图片会有锯齿。

# 7、请用html知识解决seo(搜索引擎优化)优化问题

答：1.设置 TDK

TDK是SEO术语，是三个单词的缩写。 即：title/description/keywords。

1. html语义化标签，要简洁合理，这样css和js加载不全的时候，使html文档尽量清晰的展示，而不会特别乱

# 9 简述一下你对HTML语义化的理解？

丢失样式时能让页面呈现清晰结构。

有利于SEO和搜索引擎建立良好沟通，有助于爬虫抓取更多的信息，爬虫依赖于标签来确定上下文和各个关键字的权重。

方便其它设备解析。 便于团队开发和维护，语义化根据可读性。

关于HTML语义化，你知道的都有哪些标签？header、article、address

# 8、常用浏览器内核都是什么？

答： 内核：IE的是 Trident

火狐的是 Gecko

chrome和safari 用的是 Webkit

360和搜狗这些分极速模式和兼容模式，极速模式用的Webkit的内核，兼容模式用的Trident内核。

# 10、a标签在新窗口打开链接怎么加属性？

答： <a target="\_blank">链接</a>

# 12、form标签上定义请求类型的是哪个属性？定义请求地址的是哪个属性？

答：form表单定义请求类型的是  method 属性  ， 定义请求地址的是  action属性

# 13页面导入样式时，使用link和@import有什么区别？

都是外部引用CSS方式，区别：

link是xhtml标签，除了加载css外，还可以定义RSS等其他事务；@import属于CSS范畴，只能加载CSS

link引用CSS时候，页面载入时同时加载；@import需要在页面完全加载以后加载，而且@import被引用的CSS会等到引用它的CSS文件被加载完才加载

link是xhtml标签，无兼容问题；@import是在css2.1提出来的，低版本的浏览器不支持

link支持使用javascript控制去改变样式，而@import不支持

link方式的样式的权重高于@import的权重

import在html使用时候需要<style type="text/css">标签

# 14对浏览器内核的理解？

主要分成两部分：渲染引擎和JS引擎。

渲染引擎：负责取得网页的内容（HTML、XML、图像等等）、整理讯息（例如加入CSS等），以及计算网页的显示方式，然后会输出至显示器或打印机。浏览器的内核的不同对于网页的语法解释会有不同，所以渲染的效果也不相同。  
JS引擎：解析和执行javascript来实现网页的动态效果。最开始渲染引擎和JS引擎并没有区分的很明确，后来JS引擎越来越独立，内核就倾向于只指渲染引擎。

# 10 HTML5的文件离线储存怎么使用，工作原理是什么？

在线情况下，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，那么浏览器就会根据manifest文件的内容下载相应的资源，并进行离线存储。如果已经访问过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，就不会做任何操作，如果文件改变了，那么就会重新下载文件中的资源，并且进行离线存储。例如，

在页面头部加入manifest属性

<html manifest='cache.manifest'>

在cache.manifest文件中编写离线存储的资源

# 12 iframe框架有那些优缺点？

优点：原封不动的把嵌入的网页展现出来。如果有多个网页引用iframe，只需要修改iframe，就可以实现调用的每一个页面内容的更改，方便快捷。网页如果为了统一风格，头部和版本都是一样的，就可以写成一个页面，用iframe来嵌套，增加代码的可重用。加载缓慢的第三方内容如图标和广告可由iframe解决。

缺点：搜索引擎无法解读 出现各种滚动条 增加服务器的http请求

# 15 如何实现浏览器内多个标签页之间的通信?

WebSocket SharedWorker

也可以调用 localstorge、cookies 等本地存储方式。 localstorge 在另一个浏览上下文里被添加、修改或删除时，它都会触发一个事件，我们通过监听事件，控制它的值来进行页面信息通信。localstorage是浏览器多个标签共用的存储空间，所以可以用来实现多标签之间的通信(ps：session是会话级的存储空间，每个标签页都是单独的）。

注意：Safari 在无痕模式下设置 localstorge 值时会抛出QuotaExceededError 的异常

# 16 页面可见性（Page Visibility）API 可以有哪些用途？

通过visibility state的值得检测页面当前是否可见，以及打开网页的时间。

在页面被切换到其他后台进程时，自动暂停音乐或视频的播放。

17 如何在页面上实现一个圆形的可点击区域？

map+area或者svg

border-radius

纯js实现，一个点不在圆上的算法

# 18 实现不使用 border 画出1px高的线，在不同浏览器的Quirks mode和CSS Compat模式下都能保持同一效果

<div style="height:1px;overflow:hidden;background:red"></div>

# 20 title与h1的区别、b与strong的区别、i与em的区别？

title属性没有明确意义，只表示标题；h1表示层次明确的标题，对页面信息的抓取有很大的影响  
strong标明重点内容，语气加强含义；b是无意义的视觉表示  
em表示强调文本；i是斜体，无意义的视觉表示  
**视觉**样式标签：b i u s  
**语义**样式标签：strong em ins del code

# 21 元素的alt和title有什么异同？

alt作为图片的替代文字，使屏幕阅读器能获取到关于图片的信息 title是图片的解释文字。

# 5、解释一下utf-8和GBK和ISISO8859-2 字符集

charset 属性规定 HTML 文档的字符编码。

UTF-8是一种针对Unicode的可变长度字符编码，又称万国码。UTF-8用1到6个字节编码UNICODE字符。用在网页上可以同一页面显示中文简体繁体及其它语言（如英文，日文，韩文）。

GBK是汉字编码，是双字节码，可表示繁体字和简体字。

ISO8859-2 字符集，也称为 Latin-2，收集了 东欧 字符。

8、组成HTML网页必不可少的是，head标签中必不可少的是（此题略不严谨，HTML5中可以省略HTML组成网页的部分）

Html,head,body;title. title可有可无。但是mate中的chartset 一定要有，要不然会乱码。这是实际开发的经验。

# 12、关于置换元素和不可替换元素

置换元素：浏览器根据元素的标签和属性，来决定元素的具体显示内容。 例如：浏览器会根据<img>标签的src属性的值来读取图片信息并显示出来，而如果查看(x)html代码，则看不到图片的实际内容；<input>标签的type属性来决定是显示输入框，还是单选按钮等。 (x)html中 的<img>、<input>、<textarea>、<select>、<object>都是置换元素。这些元素往往没有实际的内容，即是一个空元素。置换元素在其显示中生成了框，这也就是有的内联元素能够设置宽高的原因。 （因此img是inline还可以设置宽高）

不可替换元素：(x)html 的大多数元素是不可替换元素，即其内容直接表现给用户端（如浏览器）。例如：<label>label中的内容</label> 标签<label>是一个非置换元素，文字label中的内容”将全被显示。

# 为什么替换元素可设置宽高

· 几乎所有的**替换元素**都是**行内元素**，例如<img>、<input>等等。

· 替换元素一般有内在尺寸，所以具有width和height，可以设定。例如你不指定img的width和height时，就按其内在尺寸显示，也就是图片被保存的时候的宽度和高度。对于表单元素，浏览器也有默认的样式，包括宽度和高度。

# 13、HTML元素的显示优先级

帧元素（frame）>HTML元素，表单元素总>非表单元素

# 19、关于比较Ajax与Flash的优缺点

Ajax的优势：1.可搜索性 2.开放性 3.费用 4.易用性 5.易于开发。

Flash的优势：1.多媒体处理 2.兼容性 3.矢量图形 4.客户端资源调度

Ajax的劣势：1.它可能破坏浏览器的后退功能 2.使用动态页面更新使得用户难于将某个特定的状态保存到收藏夹中，不过这些都有相关方法解决。

Flash的劣势：1.二进制格式 2.格式私有 3.flash 文件经常会很大，用户第一次使用的时候需要忍耐较长的等待时间 4.性能问题

# 21、常见的浏览器端存储的方式

：

Cookie：cookie会随着每次HTTP请求头信息一起发送，无形中增加了网络流量，另外，cookie能存储的数据容量有限，根据浏览器类型不同而不同，IE6大约只能存储2K。

Cookie可以实现记录访问者的信息、在页面之间传递信息、自动识别用户。

Flash ShareObject：这种方式能能解决上面提到的cookie存储的两个弊端，而且能够跨浏览器，应该说是目前最好的本地存储方案。不过，需要在页面中插入一个Flash，当浏览器没有安装Flash控件时就不能用了。所幸的是，没有安装Flash的用户极少。

缺点：需要安装Flash插件。

Google Gear：Google开发出的一种本地存储技术。 缺点：需要安装Gear组件。

userData：IE浏览器可以使用userData来存储数据，容量可达到640K，这种方案是很可靠的，不需要安装额外的插件。缺点：它仅在IE下有效。

sessionStorage：用这种方式存储的数据仅窗口级别有效，**同一个窗口**（或者Tab）页面刷新或者跳转，都能获取到本地存储的数据，当新开窗口或者页面时，原来的数据就失效了。

缺点：IE不支持、不能实现数据持久保存。

globalStorage：类似于IE的userData。 缺点：IE不支持。

localStorage：localStorage是Web Storage互联网存储规范中的一部分，现在在Firefox3.5、Safari 4和IE8中得到支持。 缺点：低版本浏览器不支持。

结论： Flash shareobject是不错的选择，如果你不想在页面上嵌入Flash，可以结合使用userData(IE6+)和globalStorage(Firefox2+)和localStorage(chrome3+)实现跨浏览器。

# method 属性

浏览器使用 method 属性设置的方法将表单中的数据传送给服务器处理。

POST 方法：首先，浏览器将与 action 属性中指定的表单处理服务器建立联系，一旦建立连接之后，浏览器就会按**分段传输**的方法将数据发送给服务器。

在服务器端，一旦 POST 样式的应用程序开始执行时，就应该从一个标志位置读取参数，而一旦读到参数，在应用程序能够使用这些表单值以前，必须对这些参数进行解码。用户特定的服务器会明确指定应用程序应该如何接受这些参数。

GET 方法，这时浏览器会与表单处理服务器建立连接，然后直接在一个传输步骤中**发送所有**的表单数据：**浏览器会将数据直接附在表单的 action URL 之后**。这两者之间用问号进行分隔。

POST 还是 GET？

希望获得最佳表单传输性能，采用 GET 方法发送只有少数简短字段的小表单。

一些服务器操作系统在处理可以立即传递给应用程序的命令行参数时，会限制其数目和长度，在这种情况下，对那些有许多字段或是很长的文本域的表单来说，就应该采用 POST 方法来发送。

如果你在编写服务器端的表单处理应用程序方面经验不足，应该选择 GET 方法。如果采用 POST 方法，就要在读取和解码方法做些额外的工作

如果想在表单之外调用服务器端的应用程序，而且包括向其传递参数的过程，就要采用 GET 方法，因为该方法允许把表单这样的参数包括进来作为 URL 的一部分。而另一方面，使用 POST 样式的应用程序却希望在 URL 后还能有一个来自浏览器额外的传输过程，其中传输的内容不能作为传统 <a> 标签的内容。