



1. src 和 href 的区别

src 和 href 都是**用来引用外部的资源**, 它们的区别如下:

- **src:** 表示对资源的引用, 它指向的内容会嵌入到当前标签所在的位置。src 会将其指向的资源下载并应用到文档内, 如请求 js 脚本。当浏览器解析到该元素时, 会暂停其他资源的下载和处理, 直到将该资源加载、编译、执行完毕, 所以一般 js 脚本会放在页面底部。
- **href:** 表示超文本引用, 它指向一些网络资源, 建立和当前元素或本文档的链接关系。当浏览器识别到它指向的文件时, 就会并行下载资源, 不会停止对当前文档的处理。常用在 a、link 等标签上。

2. 对 HTML 语义化的理解

语义化是指 根据内容的结构化 (内容语义化), 选择合适的标签 (代码语义化)。通俗来讲就是用正确的标签做正确的事情。

语义化的优点如下:

- 对机器友好, 带有语义的文字表现力丰富, 更适合搜索引擎的爬虫爬取有效信息, 有利于 SEO。除此之外, 语义类还支持读屏软件, 根据文章可以自动生成目录;
- 对开发者友好, 使用语义类标签增强了可读性, 结构更加清晰, 开发者能清晰的看出网页的结构, 便于团队的开发与维护。

常见的语义化标签：

```
<header></header>  头部

<nav></nav>    导航栏

<section></section>  区块（有语义化的div）

<main></main>    主要区域

<article></article>  主要内容

<aside></aside>    侧边栏

<footer></footer>  底部
```

3. DOCTYPE(文档类型) 的作用

DOCTYPE 是 HTML5 中一种标准通用标记语言的文档类型声明，它的目的是告诉浏览器（解析器）应该以什么样（html 或 xhtml）的文档类型定义来解析文档，不同的渲染模式会影响浏览器对 CSS 代码甚至 JavaScript 脚本的解析。它必须声明在 HTML 文档的第一行。

浏览器渲染页面的两种模式（可通过 `document.compatMode` 获取，比如，语雀官网的文档类型是 **CSS1Compat**）：

- **CSS1Compat：标准模式（Strick mode）**，默认模式，浏览器使用 W3C 的标准解析渲染页面。在标准模式中，浏览器以其支持的最高标准呈现页面。
- **BackCompat：怪异模式（混杂模式）（Quick mode）**，浏览器使用自己的怪异模式解析渲染页面。在怪异模式中，页面以一种比较宽松的向后兼容的方式显示。

4. script 标签中 defer 和 async 的区别

如果没有 defer 或 async 属性，浏览器会立即加载并执行相应的脚本。它不会等待后续加载的文档元素，读取到就会开始加载和执行，这样就阻塞了后续文档的加载。

下图可以直观的看出三者之间的区别：



其中蓝色代表 js 脚本网络加载时间，红色代表 js 脚本执行时间，绿色代表 html 解析。

defer 和 async 属性都是异步去加载外部的 JS 脚本文件，它们都不会阻塞页面的解析，其区别如下：

- **执行顺序**: 多个带 `async` 属性的标签，不能保证加载的顺序；多个带 `defer` 属性的标签，按照加载顺序执行；

- **脚本是否并行执行**：`async` 属性，表示后续文档的加载和执行与 `js` 脚本的加载和执行是并行进行的，即异步执行；`defer` 属性，加载后续文档的过程和 `js` 脚本的加载(此时仅加载不执行)是并行进行的(异步)，`js` 脚本需要等到文档所有元素解析完成之后才执行，`DOMContentLoaded` 事件触发执行之前。

5. 常用的 meta 标签有哪些

`meta` 标签由 `name` 和 `content` 属性定义，**用来描述网页文档的属性**，比如网页的作者，网页描述，关键词等，除了 HTTP 标准固定了一些 `name` 作为大家使用的共识，开发者还可以自定义 `name`。

常用的 meta 标签：

- (1) `charset`，用来描述 HTML 文档的编码类型：

```
<meta charset="UTF-8" >
```

- (2) `keywords`，页面关键词：

```
<meta name="keywords" content="关键词" />
```

- (3) `description`，页面描述：

```
<meta name="description" content="页面描述内容" />
```

- (4) `refresh`，页面重定向和刷新：

```
<meta http-equiv="refresh" content="0;url=" />
```

- (5) `viewport`，适配移动端，可以控制视口的大小和比例：

```
<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">
```

其中，`content` 参数有以下几种：

- `width viewport`：宽度(数值/device-width)
- `height viewport`：高度(数值/device-height)
- `initial-scale`：初始缩放比例
- `maximum-scale`：最大缩放比例
- `minimum-scale`：最小缩放比例
- `user-scalable`：是否允许用户缩放(yes/no)

- (6) 搜索引擎索引方式：

```
<meta name="robots" content="index, follow" />
```

其中, `content` 参数有以下几种:

- `all`: 文件将被检索, 且页面上的链接可以被查询;
- `none`: 文件将不被检索, 且页面上的链接不可以被查询;
- `index`: 文件将被检索;
- `follow`: 页面上的链接可以被查询;
- `noindex`: 文件将不被检索;
- `nofollow`: 页面上的链接不可以被查询。

6. HTML5 有哪些更新

1. 语义化标签

- `header`: 定义文档的页眉 (头部) ;
- `nav`: 定义导航链接的部分;
- `footer`: 定义文档或节的页脚 (底部) ;
- `article`: 定义文章内容;
- `section`: 定义文档中的节 (section、区段) ;
- `aside`: 定义其所处内容之外的内容 (侧边) ;

2. 媒体标签

(1) audio: 音频

```
<audio src='' controls autoplay loop='true'></audio>
```

属性:

- `controls` 控制面板
- `autoplay` 自动播放
- `loop='true'` 循环播放

(2) video 视频

```
<video src='' poster='imgs/aa.jpg' controls></video>
```

属性:

- `poster`: 指定视频还没有完全下载完毕, 或者用户还没有点击播放前显示的封面。默认显示当前视频文件的第一帧画面, 当然通过 `poster` 也可以自己指定。
- `controls` 控制面板
- `width`

- height

(3) source 标签

因为浏览器对视频格式支持程度不一样，为了能够兼容不同的浏览器，可以通过 source 来指定视频源。

```
<video>
  <source src='aa.flv' type='video/flv'></source>
  <source src='aa.mp4' type='video/mp4'></source>
</video>
```

3. 表单

表单类型：

- email：能够验证当前输入的邮箱地址是否合法
- url：验证 URL
- number：只能输入数字，其他输入不了，而且自带上下增大减小箭头，max 属性可以设置为最大值，min 可以设置为最小值，value 为默认值。
- search：输入框后面会给提供一个小叉，可以删除输入的内容，更加人性化。
- range：可以提供给一个范围，其中可以设置 max 和 min 以及 value，其中 value 属性可以设置为默认值
- color：提供了一个颜色拾取器
- time：时分秒
- date：日期选择年月日
- datetime：时间和日期(目前只有 Safari 支持)
- datetime-local：日期时间控件
- week：周控件
- month：月控件

表单属性：

- placeholder：提示信息
- autofocus：自动获取焦点
- autocomplete="on" 或者 autocomplete="off" 使用这个属性需要有两个前提：
 - 表单必须提交过
 - 必须有 name 属性。
- required：要求输入框不能为空，必须有值才能够提交。
- pattern=" " 里面写入想要的正则模式，例如手机号 patte="^(+86)?\d{10}\$"
- multiple：可以选择多个文件或者多个邮箱
- form=" form 表单的 ID"

表单事件：

- oninput 每当 input 里的输入框内容发生变化都会触发此事件。
- oninvalid 当验证不通过时触发此事件。

4. 进度条、度量器

- progress 标签：用来表示任务的进度（IE、Safari 不支持），max 用来表示任务的进度，value 表示已完成多少
- meter 属性：用来显示剩余容量或剩余库存（IE、Safari 不支持）
 - high/low：规定被视作高/低的范围
 - max/min：规定最大/小值
 - value：规定当前度量值

设置规则：min < low < high < max

5.DOM 查询操作

- document.querySelector()
- document.querySelectorAll()

它们选择的对象可以是标签，可以是类(需要加点)，可以是 ID(需要加#)

6. Web 存储

HTML5 提供了两种在客户端存储数据的新方法：

- localStorage - 没有时间限制的数据存储
- sessionStorage - 针对一个 session 的数据存储

7. 其他

- 拖放：拖放是一种常见的特性，即抓取对象以后拖到另一个位置。设置元素可拖放：

```
<img draggable="true" />
```

- 画布（canvas）： canvas 元素使用 JavaScript 在网页上绘制图像。画布是一个矩形区域，可以控制其每一像素。canvas 拥有多种绘制路径、矩形、圆形、字符以及添加图像的方法。

```
<canvas id="myCanvas" width="200" height="100"></canvas>
```

- SVG：SVG 指可伸缩矢量图形，用于定义用于网络的基于矢量的图形，使用 XML 格式定义图形，图像在放大或改变尺寸的情况下其图形质量不会有损失，它是万维网联盟的标准
- 地理定位：Geolocation（地理定位）用于定位用户的位置。’

总结：

- (1) 新增语义化标签：nav、header、footer、aside、section、article

- (2) 音频、视频标签：audio、video
- (3) 数据存储：localStorage、sessionStorage
- (4) canvas（画布）、Geolocation（地理定位）、websocket（通信协议）
- (5) input 标签新增属性：placeholder、autocomplete、autofocus、required
- (6) history API：go、forward、back、pushstate

移除的元素有：

- 纯表现的元素：basefont, big, center, font, s, strike, tt, u;
- 对可用性产生负面影响的元素：frame, frameset, noframes;

7. img 的 srcset 属性的作用？

响应式页面中经常用到根据屏幕密度设置不同的图片。这时就用到了 img 标签的 srcset 属性。srcset 属性用于设置不同屏幕密度下，img 会自动加载不同的图片。用法如下：

```

```

使用上面的代码，就能实现在屏幕密度为 1x 的情况下加载 image-128.png, 屏幕密度为 2x 时加载 image-256.png。

按照上面的实现，不同的屏幕密度都要设置图片地址，目前的屏幕密度有 1x,2x,3x,4x 四种，如果每一个图片都设置 4 张图片，加载就会很慢。所以就有了新的 srcset 标准。代码如下：

```

```

其中 srcset 指定图片的地址和对应的图片质量。sizes 用来设置图片的尺寸零界点。对于 srcset 中的 w 单位，可以理解成图片质量。如果可视区域小于这个质量的值，就可以使用。浏览器会自动选择一个最小的可用图片。

sizes 语法如下：

```
sizes="[media query] [length], [media query] [length] ... "
```

sizes 就是指默认显示 128px, 如果视区宽度大于 360px, 则显示 340px。

8. 行内元素有哪些？块级元素有哪些？空(void)元素有那些？

- 行内元素有：a b span img input select strong;
- 块级元素有：div ul ol li dl dt dd h1 h2 h3 h4 h5 h6 p;

空元素，即没有内容的 HTML 元素。空元素是在开始标签中关闭的，也就是空元素没有闭合标签：

- 常见的有：
、<hr>、、<input>、<link>、<meta>;
- 鲜见的有：<area>、<base>、<col>、<colgroup>、<command>、<embed>、<keygen>、<param>、<source>、<track>、<wbr>。

9. 说一下 web worker

在 HTML 页面中，如果在执行脚本时，页面的状态是不可响应的，直到脚本执行完成后，页面才变成可响应。web worker 是运行在后台的 js，独立于其他脚本，不会影响页面的性能。并且通过 postMessage 将结果回传到主线程。这样在进行复杂操作的时候，就不会阻塞主线程了。

如何创建 web worker：

1. 检测浏览器对于 web worker 的支持性
2. 创建 web worker 文件 (js, 回传函数等)
3. 创建 web worker 对象

10. HTML5 的离线储存怎么使用，它的工作原理是什么--了解即可

离线存储指的是：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

使用方法：

- (1) 创建一个和 html 同名的 manifest 文件，然后在页面头部加入 manifest 属性：

```
<html lang="en" manifest="index.manifest">
```

- (2) 在 cache.manifest 文件中编写需要离线存储的资源：

```
CACHE MANIFEST
#v0.11
CACHE:
js/app.js
css/style.css
NETWORK:
resource/logo.png
FALLBACK:
/ /offline.html
```

- **CACHE:** 表示需要离线存储的资源列表，由于包含 manifest 文件的页面将被自动离线存储，所以不需要把页面自身也列出来。

- **NETWORK**: 表示在它下面列出来的资源只有在在线的情况下才能访问，他们不会被离线存储，所以在离线情况下无法使用这些资源。不过，如果在 CACHE 和 NETWORK 中有一个相同的资源，那么这个资源还是会被离线存储，也就是说 CACHE 的优先级更高。
- **FALLBACK**: 表示如果访问第一个资源失败，那么就使用第二个资源来替换他，比如上面这个文件表示的就是如果访问根目录下任何一个资源失败了，那么就去访问 offline.html。

(3) 在离线状态时，操作 `window.applicationCache` 进行离线缓存的操作。

如何更新缓存：

- (1) 更新 manifest 文件
- (2) 通过 javascript 操作
- (3) 清除浏览器缓存

注意事项：

- (1) 浏览器对缓存数据的容量限制可能不太一样（某些浏览器设置的限制是每个站点 5MB）。
- (2) 如果 manifest 文件，或者内部列举的某一个文件不能正常下载，整个更新过程都将失败，浏览器继续全部使用老的缓存。
- (3) 引用 manifest 的 html 必须与 manifest 文件同源，在同一个域下。
- (4) FALLBACK 中的资源必须和 manifest 文件同源。
- (5) 当一个资源被缓存后，该浏览器直接请求这个绝对路径也会访问缓存中的资源。
- (6) 站点中的其他页面即使没有设置 manifest 属性，请求的资源如果在缓存中也从缓存中访问。
- (7) 当 manifest 文件发生改变时，资源请求本身也会触发更新。

11. 浏览器是如何对 HTML5 的离线储存资源进行管理和加载？

- **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，如果是第一次访问页面，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。如果已经访问过页面并且资源已经进行离线存储了，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，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**，浏览器会直接使用离线存储的资源。

12. title 与 h1 的区别、b 与 strong 的区别、i 与 em 的区别？---了解即可

- strong 标签有语义，是起到**加重语气**的效果，而 b 标签是没有的，b 标签只是一个**简单加粗**标签。b 标签之间的字符都设为粗体，strong 标签加强字符的语气都是通过粗体来实现的，而**搜索引擎更侧重 strong 标签**。
- title 属性没有明确意义只表示是个，H1 则表示

层次明确的标题

，对页面信息的抓取有很大的影响

- **i** 内容展示为斜体，**em** 表示强调的文本

13. iframe 有那些优点和缺点？

iframe 元素会创建包含另外一个文档的内联框架（即行内框架）。

优点：

- 用来加载速度较慢的内容（如广告）
- 可以使脚本并行下载
- 可以实现跨子域通信

缺点：

- iframe 会阻塞主页面的 onload 事件
- 无法被一些搜索引擎所识别
- 会产生很多页面，不容易管理

14. label 的作用是什么？如何使用？ ---了解即可

label 标签来定义表单控件的关系：当用户选择 label 标签时，浏览器会自动将焦点转到和 label 标签相关的表单控件上。

- 使用方法 1：

```
<label for="mobile">Number:</label>
<input type="text" id="mobile"/>
```

- 使用方法 2：

```
<label>Date:<input type="text"/></label>
```

15. Canvas 和 SVG 的区别

(1) SVG：

SVG 可缩放矢量图形（Scalable Vector Graphics）是基于可扩展标记语言 XML 描述的 2D 图形的语言，SVG 基于 XML 就意味着 SVG DOM 中的每个元素都是可用的，可以为某个元素附加 JavaScript 事件处理器。在 SVG 中，每个被绘制的图形均被视为对象。如果 SVG 对象的属性发生变化，那么浏览器能够自动重现图形。

其特点如下：

- 不依赖分辨率
- 支持事件处理器
- 最适合带有大型渲染区域的应用程序（比如谷歌地图）
- 复杂度高会减慢渲染速度（任何过度使用 DOM 的应用都不快）
- 不适合游戏应用

(2) Canvas:

Canvas 是画布，通过 JavaScript 来绘制 2D 图形，是逐像素进行渲染的。其位置发生改变，就会重新进行绘制。

其特点如下：

- 依赖分辨率
- 不支持事件处理器
- 弱的文本渲染能力
- 能够以 .png 或 .jpg 格式保存结果图像
- 最适合图像密集型的游戏，其中的许多对象会被频繁重绘

注：矢量图，也称为面向对象的图像或绘图图像，在数学上定义为一系列由线连接的点。矢量文件中的图形元素称为对象。每个对象都是一个自成一体的实体，它具有颜色、形状、轮廓、大小和屏幕位置等属性。

16. head 标签有什么作用，其中什么标签必不可少？

作用

- 用于定义文档的头部，它是所有头部元素的容器。
- 引用资源：脚本、样式表等。
- 描述文档：元信息、文档标题、在 Web 中的位置以及和其他文档的关系等
- 绝大多数文档头部包含的数据都不会真正作为内容显示给用户

常见标签

- `base`
- `link`
- `meta`
- `script`
- `style`
- `title`(唯一必需)

17. 文档声明 (Doctype) 和 `<!Doctype html>` 有何作用? 严格模式与混杂模式如何区分? 它们有何意义?

****文档声明的作用：****文档声明是为了告诉浏览器，当前HTML文档使用什么版本的HTML来写的，这样浏览器才能按照声明的版本来正确的解析。

`<!Doctype html>`的作用：`<!doctype html>`的作用就是让浏览器进入标准模式，使用最新的 HTML5 标准来解析渲染页面；如果不写，浏览器就会进入混杂模式，我们需要避免此类情况发生。

严格模式与混杂模式的区分：

- **严格模式：**又称为标准模式，指浏览器按照W3C标准解析代码；
- **混杂模式：**又称怪异模式、兼容模式，是指浏览器用自己的方式解析代码。混杂模式通常模拟老式浏览器的行为，以防止老站点无法工作；

区分：网页中的DTD，直接影响到使用的是严格模式还是混杂模式，可以说DTD的使用与这两种方式的区分息息相关。

- 如果文档包含严格的DOCTYPE，那么它一般以严格模式呈现（**严格 DTD ——严格模式**）；
- 包含过渡 DTD 和 URI 的 DOCTYPE，也以严格模式呈现，但有过渡 DTD 而没有 URI（统一资源标识符，就是声明最后的地址）会导致页面以混杂模式呈现（**有 URI 的过渡 DTD ——严格模式；没有 URI 的过渡 DTD ——混杂模式**）；
- DOCTYPE 不存在或形式不正确会导致文档以混杂模式呈现（**DTD 不存在或者格式不正确——混杂模式**）；
- HTML5 没有 DTD，因此也就没有严格模式与混杂模式的区别，HTML5 有相对宽松的 法，实现时，已经尽可能大的实现了向后兼容(**HTML5 没有严格和混杂之分**)。

总之，**严格模式让各个浏览器统一执行一套规范兼容模式保证了旧网站的正常运行。**

18. 浏览器乱码的原因是什么？如何解决？---了解即可

产生乱码的原因：

- 网页源代码是gbk的编码，而内容中的中文字是utf-8编码的，这样浏览器打开即会出现html乱码，反之也会出现乱码；
- html网页编码是gbk，而程序从数据库中调出呈现是utf-8编码的内容也会造成编码乱码；
- 浏览器不能自动检测网页编码，造成网页乱码。

解决办法：

- 使用软件编辑 HTML 网页内容；
- 如果网页设置编码是gbk，而数据库储存数据编码格式是UTF-8，此时需要程序查询数据库数据显示数据前进行程序转码；
- 如果浏览器浏览时候出现网页乱码，在浏览器中找到转换编码的菜单进行转换。

19. 渐进增强和优雅降级之间的区别

(1) 渐进增强 (progressive enhancement)：主要是针对低版本的浏览器进行页面重构，保证基本的功能情况下，再针对高级浏览器进行效果、交互等方面的改进和追加功能，以达到更好的用户体验。

(2) 优雅降级 graceful degradation：一开始就构建完整的功能，然后再针对低版本的浏览器进行兼容。

两者区别：

- 优雅降级是从复杂的现状开始的，并试图减少用户体验的供给；而渐进增强是从一个非常基础的，能够起作用的版本开始的，并在此基础上不断扩充，以适应未来环境的需要；
- 降级（功能衰竭）意味着往回看，而渐进增强则意味着往前看，同时保证其根基处于安全地带。

“优雅降级”观点认为应该针对那些最高级、最完善的浏览器来设计网站。而将那些被认为“过时”或有功能缺失的浏览器下的测试工作安排在开发周期的最后阶段，并把测试对象限定为主流浏览器（如 IE、Mozilla 等）的前一个版本。在这种设计范例下，旧版的浏览器被认为仅能提供“简陋却无妨 (poor, but passable)” 的浏览体验。可以做一些小的调整来适应某个特定的浏览器。但由于它们并非我们所关注的焦点，因此除了修复较大的错误之外，其它的差异将被直接忽略。

“渐进增强”观点则认为应关注于内容本身。内容是建立网站的诱因，有的网站展示它，有的则收集它，有的寻求，有的操作，还有的网站甚至会包含以上的种种，但相同点是它们全都涉及到内容。这使得“渐进增强”成为一种更为合理的设计范例。这也是它立即被 Yahoo 所采纳并用以构建其“分级式浏览器支持 (Graded Browser Support)”策略的原因所在。

20. 说一下 HTML5 drag API---了解即可

- dragstart: 事件主体是被拖放元素, 在开始拖放被拖放元素时触发。
- drag: 事件主体是被拖放元素, 在正在拖放被拖放元素时触发。
- dragenter: 事件主体是目标元素, 在被拖放元素进入某元素时触发。
- dragover: 事件主体是目标元素, 在被拖放在某元素内移动时触发。
- dragleave: 事件主体是目标元素, 在被拖放元素移出目标元素是触发。
- drop: 事件主体是目标元素, 在目标元素完全接受被拖放元素时触发。
- dragend: 事件主体是被拖放元素, 在整个拖放操作结束时触发。