


一.src和href的区别

src(Source)属性会将资源嵌入到当前文档中元素所在位置，也就是会替换当前内容

href(Hypertext Reference超文本引用)属性通过指定Web资源的位置，来定义当前元素或者当前文档与目标资源之间的链接或关系，

需要说明的是，**src** 属性并不代表资源会阻塞页面，当资源为时，浏览器通常会异步加载图片，不会阻塞页面的渲染。这有助于提高页面加载性能。当资源为一个[文件](#)时，它会阻塞页面渲染，直到资源完全加载完毕。

二.link和@import引入css 区别

```
<link href="a.css" rel="stylesheet">
```

```
<style> @import url('a.css'); </style>
```

区别1: 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,要注意的是导入语句应写在样式表的开头，否则无法正确导入外部文件

区别2: link是XHTML标签，无兼容问题；@import是在CSS2.1提出的，低版本的浏览器不支持。

区别3: link支持使用Javascript控制DOM去改变样式；而@import不支持

我们应尽量使用 `<link>` 标签导入外部 CSS 文件

三.css的四种引入方式

1.内联样式

内联样式，也叫行内样式，指的是直接在 `HTML` 标签中的 `style` 属性中添加 `CSS`。示例：

```
<div style="display: none;background:red"></div>
```

2.嵌入样式

嵌入方式指的是在 HTML 头部中的 `<style>` 标签下书写 `CSS` 代码。示例：

```
<head>
  <style>
    .content {
      background: red;
    }
  </style>
</head>
```

3.链接样式

链接方式指的是使用 `HTML` 头部的 标签引入外部的 `CSS` 文件。示例：

```
<head>
  <link rel="stylesheet" type="text/css"
  href="style.css">
</head>
```

4.导入样式

导入方式指的是使用 `CSS` 规则引入外部 `CSS` 文件。示例：

```
<style>
  @import url(style.css);
</style>
```

四.html语义化的理解

HTML语义化：就是页面去掉样式或者加载失败的时候能够让页面呈现出清晰的结构。[HTML5](#)新增了很多语义化的标签，例如：header、footer、nav、menu、section、article等等，单单从字面上理解，就知道标签的含义。

好处：

- html 语义化让页面的内容结构化，结构更清晰，便于对浏览器、搜索引擎解析。
- 即使在没有样式 CSS 情况下也能以一种易读的文档格式显示
- 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。

五.文档声明的作用

原因是不同版本所支持的 *HTML* 元素类型是不同的，开发人员需要告诉浏览器以哪一种文档类型方式来解析当前的这个 *HTML* 文件。

以上代码是 HTML5 的文档类型声明，它告诉浏览器当前页面是使用 HTML5 规范编写的，HTML5 是最新的 HTML 版本，拥有更多的功能和优化，因此推荐在新的 Web 页面中使用它。

六.meta标签

HTML的meta标签是用于提供关于HTML文档的元数据（metadata）信息的标签。它位于HTML文档的头部（head）部分，这些元数据将服务于浏览器（如何布局或重载页面），搜索引擎和其它网络服务。meta标签有多种属性：

1. charset：指定文档的字符编码方式，例如 `<meta charset="UTF-8">` 表示使用UTF-8编码。
2. viewport：用于控制页面的视口（viewport）属性，用于在**移动设备**上显示网页时进行适配（**响应式**）。例如 `<meta name="viewport" content="width=device-width, initial-scale=1.0">` 表示视口宽度与设备宽度相同，并且初始缩放比例为1。
3. keywords：指定页面的关键词，用于搜索引擎优化（SEO）。例如 `<meta name="keywords" content="HTML, meta, 标签">` 表示页面的关键词是"HTML"、"meta"和"标签"。
4. description：指定页面的描述，也用于SEO。
5. http-equiv：可用于模拟一个 HTTP 响应头。

七.html5新特性

1. 语义化标签：

HTML5引入了诸如

、
、
、

等语义化标签，html 语义化标签让页面的内容结构化，结构更清晰，便于对浏览器、搜索引擎解析。

2. 增强型表单：

HTML5为表单元素带来了一系列增强特性，包括新的输入类型（如日

期、时间、邮箱等）、表单验证、自动完成等，提高了表单的功能和便利性。比如placeholder属性，标签（用户会在他们输入数据时看到域定义选项的下拉列表）等等。

3. 视频和音频：

HTML5提供了内置的多媒体支持，包括

和标签，使得在网页中嵌入音频和视频变得更加简单。

4. Canvas绘图：

HTML5的

5. SVG绘图：HTML5中的SVG（可缩放矢量图形）是一种用于在网页上绘制矢量图形的技术
6. 地理定位：HTML5的地理定位API允许网页获取用户的地理位置信息，为基于地理位置的Web应用提供了支持。
7. 拖放API：可以对拖放这个操作提供多样的支持
8. WebWorker：HTML5引入了Web Workers标准，允许在后台运行脚本，提高了网页的并行处理能力，改善了用户体验。
9. WebStorage：HTML5引入了Web Storage API，用于在客户端（浏览器）上存储和管理数据。
10. WebSocket：HTML5中的WebSocket是一种在Web浏览器和服务器之间进行双向通信的协议。它提供了一种实时、持久的连接，使得服务器可以主动向客户端推送数据，而不需要客户端不断地发送请求。

SVG和Canvas的区别：

1. 图形处理方式：

- SVG: SVG使用矢量图形描述, 它基于XML标记语言, 可以通过直接操作元素的属性来绘制和修改图形。
- Canvas: Canvas使用位图(bitmap)方式绘制, 它提供了一个画布, 可以在上面绘制像素。通过JavaScript中的绘图API, 我们可以在Canvas上绘制2D图形

2. 渲染方式:

- SVG: SVG图形是矢量图形, 它以真实的图形对象存在于DOM中, 并且可以通过CSS样式表进行样式控制。当SVG图形发生变化时, 浏览器会自动重新渲染图形。
- Canvas: Canvas是一个空白画布, 它只是一个像素网格, 在绘制过程中, 我们直接操作像素点。当内容发生变化时, 需要手动清除画布并重新绘制。

3. 功能和适用场景:

- SVG: SVG适用于需要保留图形的矢量性质, 并且需要对图形进行交互或动画效果的场景。它支持丰富的图形元素和属性, 可以实现复杂的图形和动画效果, 并与DOM和事件交互紧密结合。
- Canvas: Canvas适用于需要实时生成和操作像素的场景, 例如游戏绘制、数据可视化等。由于它是基于位图的, 对于复杂的图形和动画, 性能可能受到影响。

谈谈Webworker:

Web Worker的基本原理就是在当前javascript的主线程中, 使用Worker类加载一个javascript文件来开辟一个新的线程, 起到互不阻塞执行的效果, 并且提供主线程和新线程之间数据交换的接口: `postMessage`、`onmessage`, `onmessage`用于主线程接收处理结果, `postMessage`用于主线程发送需要处理的数据

```
// 创建Web Worker对象
var worker = new Worker('worker.js');

// 监听Web Worker的消息事件
worker.onmessage = function(event) {
    console.log('接收到Worker的消息: ' + event.data);
};

// 向Worker发送消息
worker.postMessage('开始执行任务');
```

谈谈WebStorage

Web Storage 提供了两种机制：sessionStorage 和 localStorage。

1.sessionStorage:

- 数据存储在一个会话期间（session）中，当用户关闭浏览器标签页或窗口时会被清除。
- 只能存储字符串类型的数据，可以通过JSON.stringify和JSON.parse来存储和读取对象。
- 可以通过JavaScript的sessionStorage对象进行访问。

2.localStorage

- 数据长期存储在客户端，即使关闭浏览器后再次打开也会保留。
- 同样只能存储字符串类型的数据，也可以使用JSON方法处理对象数据。
- 可以通过JavaScript的localStorage对象进行访问。

提供的API有(以localStorage为例，sessionStorage同理)：

- 保存数据：localStorage.setItem(key,value);

- 读取数据：localStorage.getItem(key);
- 删除单个数据：localStorage.removeItem(key);
- 删除所有数据：localStorage.clear();
- 得到某个索引的key：localStorage.key(index);

谈谈WebSocket:

HTML5中的WebSocket是一种在Web浏览器和服务器之间进行双向通信的协议。它提供了一种实时、持久的连接，使得服务器可以主动向客户端推送数据，而不需要客户端不断地发送请求。

特点:

- 握手阶段采用HTTP协议，默认端口是80和443
- 建立在TCP协议基础之上，和http协议同属于应用层
- 可以发送文本，也可以发送二进制数据。
- 没有同源限制，客户端可以与任意服务器通信。
- 协议标识符是ws（如果加密，为wss），如ws://localhost:8023

八.行内元素和块级元素以及行内块元素:

块级元素，有自己的宽度和高度，也就是可自定义 width 和 height，并且独自占据一行高度，一般可以作为其他容器使用，可容纳块级元素和行内元素，可以设置margin和padding

常见的块级元素:


```
<div>      // 定义文档中的分区或节
<form>     // 创建 HTML 表单
<h1>       // 定义最大的标题
<h2>       // 定义副标题
<h3>       // 定义标题
<h4>       // 定义标题
<h5>       // 定义标题
<h6>       // 定义最小的标题
<hr>       // 创建一条水平线
<li>       // 标签定义列表项目
<ol>       // 定义有序列表
<ul>       // 定义无序列表
<p>        // 标签定义段落
<table>     // 标签定义 HTML 表格
<tbody>     // 标签表格主体（正文）
```

行内元素不可以设置宽（width）和高（height），行内元素的高度一般由元素内部的字体大小决定，宽度由内容的长度控制，但可以与其他行内元素位于同一行，行内元素内一般不可以包含块级元素。

```
<a>        // 标签可定义锚
<br>       // 换行
<span>     // 组合文档中的行内元素
<textarea> // 多行的文本输入控件
```

行内块级元素，它既具有块级元素的特点，也有行内元素的特点，**它可以自由设置元素宽度和高度，也可以在一行中放置多个行内块级元素**。如果把多个行内块元素放置在同一行里，之间会有空白缝隙，可以通过设置父级元素的 font-size 为 0 来消除间隙

```
<button>
<input>
<select>
<img>
```

html可以通过display属性来自己定义一个元素的类型

display: block, 定义元素为块级元素

display: inline, 定义元素为行内元素

display: inline-block, 定义元素为行内块级元素

九.各种图片的歌手以及应用场景:

格式	压缩方式	适应情况	缺点
JPEG	有损压缩, 高质量, 以24位存储, 可呈现1600万种颜色	照片, 色彩丰富的图片, Webp的兼容替代品	矢量图像、线条感较强和颜色对比感强的图像, 有明显的人为压缩模糊感。 不支持透明度处理
PNG	无损压缩, 8位支持256种颜色, 24位支持1600万种颜色	呈现小图如logo, 颜色简单且对比强烈的图片 , 需文字清晰图片	体积大, 画质中等
WebP	有损和无损压缩, 图像压缩算法好, 体积小但有肉眼识别无差别的图像质量	支持细节丰富、透明、动态图片, 最优方式	兼容性差 , 仅chrome内核、Edge、opera、Android支持
Base64	字符串形式 , 传输8bit字节代码	小图, 减少http请求	大图字符串过长, 不适应于大图
SVG	矢量图, 非位图 , 包含绘制图形指令	图标, 可通过css对图标叠加效果, 不丢失细节, 不出现锯齿	需要绘制, 设计复杂
BMP	无损, 没有压缩, 支持索引色和直接色	对图片质量要求高, 画质最好	体积太大 , 不利于网络传输
GIF	无损压缩不降低质量, 支持动画和透明	动态图片	最多256色, 画质差

@稀土掘金技术社区

十.浏览器内核

浏览器的内核是指浏览器所使用的渲染引擎，它负责将网页的HTML、CSS和JavaScript代码解析并渲染成可视化的网页。浏览器的内核决定了网页的渲染效果、性能以及对各种Web标准的支持程度。

目前最为主流浏览器有五大款，分别是IE、Firefox、Google Chrome、Safari、Opera。

下面总结一下各常用浏览器所使用的内核。

- 1、IE浏览器内核：Trident内核，也是俗称的IE内核；
- 2、Chrome浏览器内核：统称为Chromium内核或Chrome内核，以前是Webkit内核，现在是Blink内核；
- 3、Firefox浏览器内核：Gecko内核，俗称Firefox内核；
- 4、Safari浏览器内核：Webkit内核；
- 5、Opera浏览器内核：最初是自己的Presto内核，后来是Webkit，现在是Blink内核；

十.script标签中defer和async的区别？

在HTML的 `<script>` 标签中，`defer` 和 `async` 是用来控制脚本加载和执行的属性。

区别如下：

`async`: 异步加载下载js脚本文件,在下载完之后立马执行js脚本文件.多个`async`脚本文件执行时执行顺序没有保障.执行js过程中,会阻塞html的解析和渲染

`defer`: 异步下载js脚本文件,在下载完之后等到html解析完毕才执行js脚本文件.多个`defer`脚本文件执行时的执行顺序有保障.

`async`和`defer`的共同作用：

- 在**加载和下载**js的时候不会阻塞html的解析和渲染

使用场景：如果想要使用DOM最好使用defer,因为它在html文档解析完毕后，DOMContentLoaded事件触发前才执行，而async的执行时间没有保障.它可能在html解析完全之前就执行了.不能获取到所有的DOM

tips: 什么是DOMContentLoaded事件

DOMContentLoaded 事件是在HTML文档解析完成并且所有的DOM元素被加载到文档树中之后触发的事件。它表示文档的初始HTML已经完全加载和解析，但外部资源（如图片、样式表、脚本等）可能仍在加载中。

DOMContentLoaded 事件的触发时机通常早于 **load** 事件，因为 **DOMContentLoaded** 只需等待HTML文档的解析完成，而 **load** 事件需要等待整个页面及其所有外部资源（如图片）完全加载。

十一.前端如何做好seo

搜索引擎优化（Search Engine Optimization）,简称SEO。是按照搜索引擎给出的优化建议，以增强网站核心价值为目标，从网站结构、内容建设方案、用户互动传播等角度进行合理规划，以改善网站在搜索引擎中的表现，吸引更多搜索引擎用户访问网站。SEO与搜索引擎，互相促进，互利互助。要想更好理解以上一段废话，首先需要理解关于搜索引擎的两个概念。

1. 避免过度堆砌关键词，以免被搜索引擎认为是垃圾信息。

2. 语义化HTML代码

- 语义化代码可以让搜索引擎更容易理解网页

3. 重要内容的HTML代码放在前面

- 搜索引擎抓取顺序是**从上到下**，保证搜索引擎优先抓取

4. 重要内容不使用JS输出

- 爬虫不会执行JS去获取内容

5. 少用iframe

- `<iframe>` 元素可以用于在一个网页中嵌入另一个网页或文档。当一个页面包含了 `<iframe>` 元素时，浏览器会加载和显示来自指定源的另一个文档。然而，搜索引擎爬虫（也称为蜘蛛或机器人）在抓取网页内容时，一般会忽略 `<iframe>` 中的内容

6. 非装饰性图片必须加alt

- alt是当图片无法显示的时候，对当前对象起描述作用（对SEO优化有一定帮助）

7. 提高网站速度

- 网站速度也是搜索引擎排序的一个重要指标

8. 适配移动设备

- 确保网页在不同屏幕尺寸和设备上都能够正常显示和操作，这对搜索引擎排名和用户体验都很重要。

9. 唯一的URL：

- 每个页面应该有唯一的URL，避免重复内容和重复标签。