33-HTML替换型元素：为什么link一个CSS要用href，而引入js要用src呢？

你好，我是winter。我们今天来讲讲替换型元素。

我们都知道一个常识，一个网页，它是由多个文件构成的，我们在之前的课程中，已经学过了一种引入文件的方案：链接。

这节课我们要讲的替换型元素，就是另一种引入文件的方式了。替换型元素是把文件的内容引入，替换掉自身位置的一类标签。

我们首先来看一种比较熟悉的标签：script标签。

script

我们之所以选择先讲解script标签，是因为script标签是为数不多的既可以作为替换型标签，又可以不作为替换型标签的元素。

我们先来看看script标签的两种用法：

```
<script type="text/javascript">
console.log("Hello world!");
</script>

<script type="text/javascript" src="my.js"></script>
```

这个例子中，我们展示了两种script标签的写法，一种是直接把脚本代码写在script标签之间，另一种是把代码放到独立的js文件中，用src属性引入。

这两种写法是等效的。我想这种等效性可以帮助你理解替换型元素的“替换”是怎么一回事。

这里我们就可以回答标题中的问题了：凡是替换型元素，都是使用src属性来引用文件的，而我们之前的课程中已经讲过，链接型元素是使用href标签的。

虽然我不知道当初是怎么设计的，但是style标签并非替换型元素，不能使用src属性，这样，我们用link标签引入CSS文件，当然就是用href标签啦。

接下来我们再看看别的替换型元素，先来了解一下img标签。

img

毫无疑问我们最熟悉的替换型标签就是img标签了，几乎每个前端都会日常使用img标签。

img标签的作用是引入一张图片。这个标签是没有办法像script标签那样作为非替换型标签来使用的，它必须有src属性才有意义。

如果一定不想要引入独立文件，可以使用data uri，我们来看个实际的例子：

```
 <img src='data:image/svg+xml;charset=utf8,<svg version="1.1" xmlns="http://www.w3.org/2000/svg"><rect widt
```

这个例子中我们使用了data uri作为图片的src，这样，并没有产生独立的文件，客观上做到了和内联相同的结果，这是一个常用的技巧。

img标签可以使用width和height指定宽度和高度。也可以只指定其中之一。我们看个例子：

```
 <img src='data:image/svg+xml;charset=utf8,<svg width="600" height="400" version="1.1"
xmlns="http://www.w3.org/2000/svg"><ellipse cx="300" cy="150" rx="200" ry="80"
style="fill:rgb(200,100,50);
stroke:rgb(0,0,100);stroke-width:2"/></svg>' width="100"/>
```

这个例子中，为了方便你理解，我们把图片换成了椭圆，我们可以看到，当我们指定了宽度后，图片被**等比例缩放了**。这个特性非常重要，适用于那种我们既要限制图片尺寸，又要保持图片比例的场景。

如果从性能的角度考虑，建议你同时给出图片的宽高，因为替换型元素加载完文件后，如果尺寸发生变换，会触发重排版（这个概念我们在浏览器原理部分已经讲过，可以复习一下）。

此处要重点提到一个属性，alt属性，这个属性很难被普通用户感知，对于视障用户非常重要，可以毫不夸张地讲，给img加上alt属性，已经做完了可访问性的一半。

img标签还有一组重要的属性，那就是srcset和sizes，它们是src属性的升级版（所以我们前面讲img标签必须有src属性，这是不严谨的说法）。

这两个属性的作用是在不同的屏幕大小和特性下，使用不同的图片源。下面一个例子也来自MDN，它展示了srcset和sizes的用法

```
<img srcset="elva-fairy-320w.jpg 320w,
             elva-fairy-480w.jpg 480w,
             elva-fairy-800w.jpg 800w"
     sizes="(max-width: 320px) 280px,
            (max-width: 480px) 440px,
            800px"
     src="elva-fairy-800w.jpg" alt="Elva dressed as a fairy">
```

srcset提供了根据屏幕条件选取图片的能力，但是其实更好的做法，是使用picture元素。

picture

picture元素可以根据屏幕的条件为其中的img元素提供不同的源，它的基本用法如下：

```
<picture>
  <source srcset="image-wide.png" media="(min-width: 600px)">
  <img src="image-narrow.png">
</picture>
```

picture元素的设计跟audio和video保持了一致（稍后我会为你讲解这两个元素），它跟img搭配srcset和sizes不同，它使用source元素来指定图片源，并且支持多个。

这里的media属性是media query，跟CSS的@media规则一致。

video

在HTML5早期的设计中，video标签跟img标签类似，也是使用src属性来引入源文件的，不过，我想应该是考虑到了各家浏览器支持的视频格式不同，现在的video标签跟picture元素一样，也是提倡使用source的。

下面例子是一个古典的video用法：

```
<video controls="controls" src="movie.ogg">
</video>
```

这个例子中的代码用src来指定视频的源文件。但是因为一些历史原因，浏览器对视频的编码格式兼容问题分成了几个派系，这样，对于一些兼容性要求高的网站，我们使用单一的视频格式是不合适的。

现在的video标签可以使用source标签来指定接入多个视频源。

```
<video controls="controls" >
  <source src="movie.webm" type="video/webm" >
  <source src="movie.ogg" type="video/ogg" >
  <source src="movie.mp4" type="video/mp4">
  You browser does not support video.
</video>
```

从这个例子中，我们可以看到，source标签除了支持media之外，还可以使用type来区分源文件的使用场景。

video标签的内容默认会被当做不支持video的浏览器显示的内容吗，因此，如果要支持更古老的浏览器，还可以在其中加入object或者embed标签，这里就不详细展开了。

video中还支持一种标签：track。

track是一种播放时序相关的标签，它最常见的用途就是字幕。track标签中，必须使用 srclang 来指定语言，此外，track具有kind属性，共有五种。

- subtitles：就是字幕了，不一定是翻译，也可能是补充性说明
- captions：报幕内容，可能包含演职员表等元信息，适合听障人士或者没有打开声音的人了解音频内容
- descriptions：视频描述信息，适合视障人士或者没有视频播放功能的终端打开视频时了解视频内尔用
- chapters：用于浏览器视频内容。
- metadata：给代码提供的元信息，对普通用户不可见。

一个完整的video标签可能会包含多种track和多个source，这些共同构成了一个视频播放所需的全部信息。

audio

接下来我们来讲讲audio，跟picture和video两种标签一样，audio也可以使用source元素来指定源文件。我们看一下例子：

```
<audio controls>
  <source src="song.mp3" type="audio/mpeg">
  <source src="song.ogg" type="audio/ogg">
  <p>You browser does not support audio.</p>
</audio>
```

但比起video，audio元素的历史问题并不严重，所以使用src也是没有问题的。

iframe

最后我们来讲一下iframe，这个标签能够嵌入一个完整的网页。

不过，在移动端，iframe受到了相当多的限制，它无法指定大小，里面的内容会被完全平铺到父级页面上。

同时很多网页也会通过http协议头禁止自己被放入iframe中。

iframe标签也是各种安全问题的重灾区。opener、window.name、甚至css的opacity都是黑客可以利用的漏洞。

因此，在2019年，当下这个时间点，任何情况下我都不推荐在实际开发中用以前的iframe。

当然，不推荐使用是一回事，因为没人能保证不遇到历史代码，我们还是应该了解一下iframe的基本用法：

```
<iframe src="http://time.geekbang.org"></iframe>
```

这个例子展示了古典的iframe用法。

在新标准中，为iframe加入了sandbox模式和srcdoc属性，这样，给iframe带来了一定的新场景。我们来看看例子：

```
<iframe sandbox srcdoc="<p>Yeah, you can see it <a href="/gallery?mode=cover&amp;amp;page=1">in my gallery<
```

这个例子中，使用srcdoc属性创建了一个新的文档，嵌入在iframe中展示，并且使用了sandbox来隔离。

这样，这个iframe就不涉及任何跨域问题了。

总结

这节课，我们又认识了一组HTML元素：替换型元素。它们的特点是，引入一个外部资源来进入页面，替换掉自身的位置。

我们通过对script、img、picture、audio、video、iframe几个标签的讲解，了解了不同的资源引入方式：

- src属性；
- srcset属性；
- source标签；
- srcdoc属性。

这中间，我们也介绍了一些小技巧，比如src属性的好朋友——data uri，这在实际开发中非常有用。

最后，留给大家一个小问题，请查资料总结一下，在多数现代浏览器兼容的范围内，src属性支持哪些协议的uri（如http和我们提到的data）。

精选留言：

- favorlm 2019-04-09 07:18:25
  早起第一件事，学习 [4赞]

- Geeker1992 2019-04-10 08:50:51
  老师，style 既然也可以这么用
  <style>css 规则</style>，
  为什么没有 <style src=“”></style> ？
  [1赞]

- umaru 2019-04-10 08:48:01
  Style元素不能使用css属性，这句话没看懂 [1赞]

- 赵健 2019-04-09 23:28:40
  老师好，想请问下，业务场景中需要嵌入公司其他行业线的页面，这种不使用iframe该咋办？ [1赞]

- 阿成 2019-04-09 10:10:36
  常见的有：http://,https://,file://,data... [1赞]

- AbyssKR 2019-04-11 19:44:18
  style 标签不在替换中元素中，所以不使用 src 属性。<style src=""> W3C 的回答是从未讨论过，而且看到貌似以前还出现过 link 引入 JavaScript 的想法。