你好，我是winter。

在前面的课程中，我讲到了HTML的语义和元信息标签，今天这一课，我们来讲另一类HTML元素：链接。

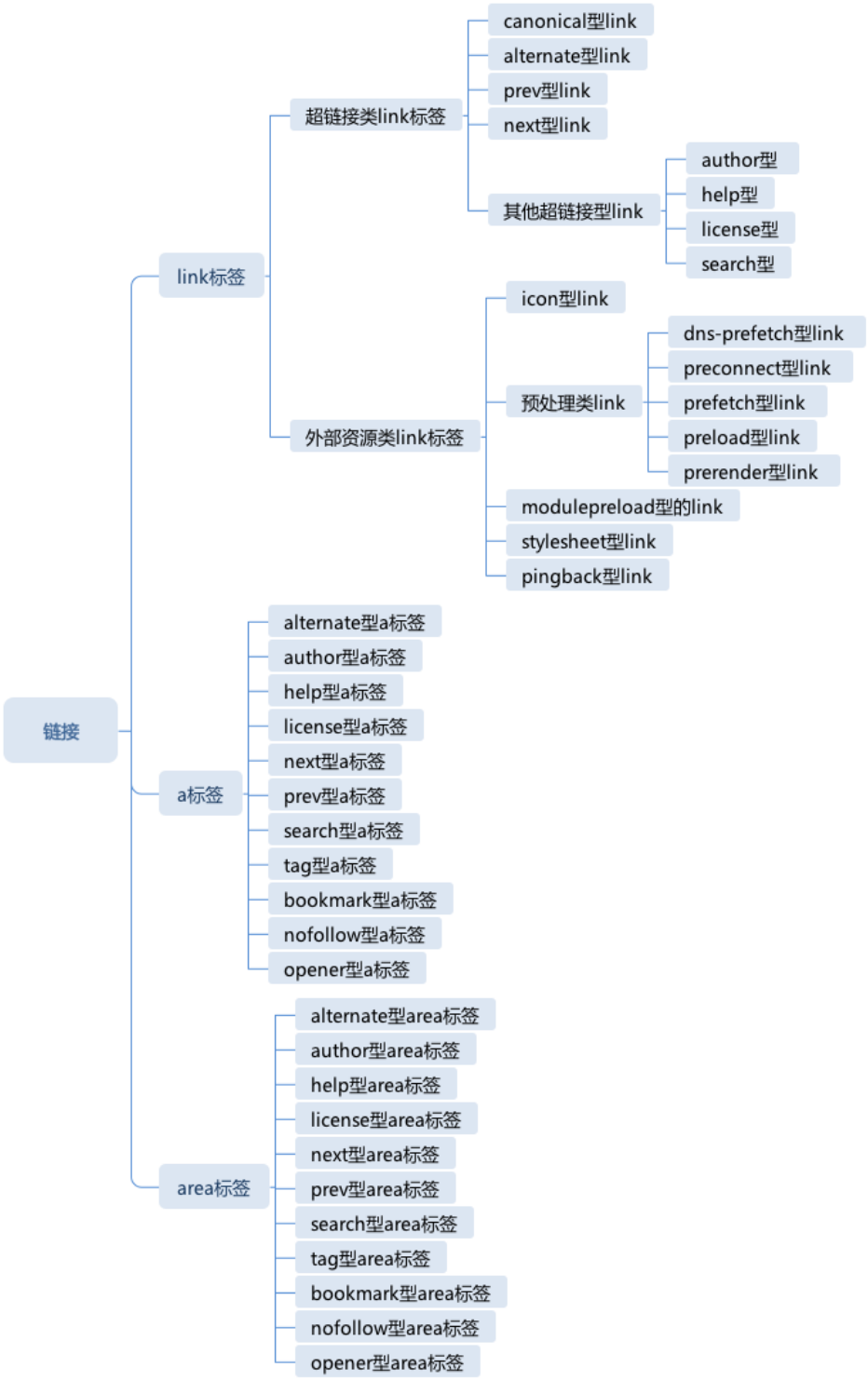
链接这种元素可以说是占据了整个互联网。也正是因为无处不在的超链接，才让我们的万维网如此繁荣。没有了超链接的HTML，最多可以称为富文本，没法称作超文本（hyper text）。

我想，作为互联网从业者，我们一定对链接都非常熟悉了。链接能够帮助我们从一个网页跳转到另一个网页。

不过，除了肉眼可见的这些链接，其实HTML里面还规定了一些不可见链接的类型，这节课，我就来给你介绍链接家族的全员，让你对它们有一个完整的认识。

链接是HTML中的一种机制，它是HTML文档和其它文档或者资源的连接关系，在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

链接的家族中有a标签、area标签和link标签。今天，我会逐一对它们进行介绍。



link 标签

提到链接，我们都知道a标签可以成为超链接，但是我们今天的内容，要从一个大家不太熟悉的标签开始，也就是link标签。

我们已经介绍过元信息类标签。实际上，我们并没有介绍完全，有些link标签也是元信息类标签的一种。

我们已经讲过，HTML标准并没有规定浏览器如何使用元信息，我们还讲到了元信息中有不少是被设计成“无需被浏览器识别，而是专门用于搜索引擎看的”。

link标签也是元信息的一种，在很多时候，它也是不会对浏览器产生任何效果的，这也是很多人会忽略link标签学习的原因。

link标签会生成一个链接，它可能生成超链接，也可能生成外部资源链接。

一些link标签会生成超链接，这些超链接又不会像a标签那样显示在网页中。这就是超链接型的link标签。

这意味着多数浏览器中，这些link标签不产生任何作用。但是，这些link标签能够被搜索引擎和一些浏览器插件识别，从而产生关键性作用。

比如，到页面RSS的link标签，能够被浏览器的RSS订阅插件识别，提示用户当前页面是可以RSS订阅的。

另外一些link标签则会把外部的资源链接到文档中，也就是说，会实际下载这些资源，并且做出一些处理，比如我们常见的用link标签引入样式表。

除了元信息的用法之外，多数外部资源型的link标签还能够被放在body中使用，从而起到把外部资源链接进文档的作用。

link标签的链接类型主要通过rel属性来区分，在本篇文章中，我们提到xx型link即表示属性rel为xx的link，其代码类似下面：

```
<link rel="xx" ...>
```

下面我们先来看看超链接型link标签。

超链接类link标签

超链接型link标签是一种被动型链接，在用户不操作的情况下，它们不会被主动下载。

link标签具有特定的rel属性，会成为特定类型的link标签。产生超链接的link标签包括：具有 rel=“canonical” 的link、具有 rel="alternate"的link、具有 rel=“prev” rel="next"的link等等。

canonical型link

这种link的代码写法是这样：

```
<link rel="canonical" href="...">
```

这个标签提示页面它的主URL，在网站中常常有多个URL指向同一页面的情况，搜索引擎访问这类页面时会去掉重复的页面，这个link会提示搜索引擎保留哪一个URL。

alternate型link

这种link的代码写法是这样：

```
<link rel="alternate" href="...">
```

这个标签提示页面它的变形形式，这个所谓的变形可能是当前页面内容的不同格式、不同语言或者为不同的设备设计的版本，这种link通常也是提供给搜索引擎来使用的。

alternate型的link的一个典型应用场景是，页面提供rss订阅时，可以用这样的link来引入：

```
<link rel="alternate" type="application/rss+xml" title="RSS" href="...">
```

除了搜索引擎外，很多浏览器插件都能识别这样的link。

prev型link和next型link

在互联网应用中，很多网页都属于一个序列，比如分页浏览的场景，或者图片展示的场景，每个网页是序列中的一个项。

这种时候，就适合使用prev和next型的link标签，来告诉搜索引擎或者浏览器它的前一项和后一项，这有助于页面的批量展示。

因为next型link告诉浏览器“这是很可能访问的下一个页面”，HTML标准还建议对next型link做预处理，在本课后面的内容，我们会讲到预处理类的link。

其它超链接类的link

其它超链接类link标签都表示一个跟当前文档相关联的信息，可以把这样的link标签视为一种带链接功能的meta标签。

- rel=“author” 链接到本页面的作者，一般是 mailto:协议
- rel=“help” 链接到本页面的帮助页
- rel=“license” 链接到本页面的版权信息页
- rel=“search” 链接到本页面的搜索页面（一般是站内提供搜索时使用）

到这里，我们已经讲完了所有的超链接类的link标签用法了。接下来我们讲讲外部资源类link标签。

外部资源类link标签

外部资源型link标签会被主动下载，并且根据rel类型做不同的处理。外部资源型的标签包括：具有icon型的link、预处理类link、modulepreload型的link、stylesheet、pingback。下面我们来一一介绍它们。

icon型link

这类链接表示页面的icon。多数浏览器会读取icon型link，并且把页面的icon展示出来。

icon型link是唯一一个外部资源类的元信息link，其它元信息类link都是超链接，这意味着，icon型link中的图标地址默认会被浏览器下载和使用。

如果没有指定这样的link，多数浏览器会使用域名根目录下的favicon.ico，即使它并不存在，所以从性能的角度考虑，建议一定要保证页面中有icon型的link。

只有icon型link有有效的sizes属性，HTML标准允许一个页面出现多个icon型link，并且用sizes指定它适合的icon尺寸。

预处理类link

我们都知道，导航到一个网站需要经过dns查询域名、建立连接、传输数据、加载进内存和渲染等一系列的步骤。

预处理类link标签就是允许我们控制浏览器，提前针对一些资源去做这些操作，以提高性能（当然如果你乱用的话，性能反而更差）。

下面我来列一下这些link类型：

- dns-prefetch型link 提前对一个域名做dns查询，这样的link里面的href实际上只有域名有意义。
- preconnect型link 提前对一个服务器建立tcp连接。
- prefetch型link 提前取href指定的url的内容。
- preload型link 提前加载href指定的url。
- prerender型link 提前渲染href指定的url。

modulepreload型的link

modulepreload型link的作用是预先加载一个JavaScript的模块。这可以保证JS模块不必等到执行时才加载。

这里的所谓加载，是指完成下载并放入内存，并不会执行对应的JavaScript。

```
<link rel="modulepreload" href="app.js">
<link rel="modulepreload" href="helpers.js">
<link rel="modulepreload" href="irc.js">
<link rel="modulepreload" href="fog-machine.js">
<script type="module" src="app.js">
```

这个例子来自HTML标准，我们假设app.js中有 import “irc” 和 import “fog-machine”, 而 irc.js中有 import “helpers”。这段代码使用moduleload型link来预加载了四个js模块。

尽管，单独使用script标签引用app.js也可以正常工作，但是我们通过加入对四个JS文件的link标签，使得四个JS文件有机会被并行地下载，这样提高了性能。

stylesheet型link

样式表大概是所有人最熟悉的link标签用法了。它的样子是下面这样的。

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

基本用法是从一个CSS文件创建一个样式表。这里type属性可以没有，如果有，必须是"text/css"才会生效。

rel前可以加上alternate，成为rel=“alternate stylesheet”，此时必须再指定title属性。

这样可以为页面创建一份变体样式，一些浏览器，如 Firefox 3.0，支持从浏览器菜单中切换这些样式，当然了，大部分浏览器不支持这个功能，所以仅仅从语义的角度了解一下这种用法即可。

pingback型link

这样的link表示本网页被引用时，应该使用的pingback地址，这个机制是一份独立的标准，遵守pingback协议的网站在引用本页面时，会向这个pingback url发送一个消息。

以上就是link标签的所有用法了。接下来我们来介绍一下最熟悉的 a 标签，当然了，也可能你学过了本节课以后，觉得自己其实也没那么熟悉。

a 标签

a标签是“anchor”的缩写，它是锚点的意思，所谓锚点，实际上也是一种比喻的用法，古代船舶用锚来固定自己的位置，避免停泊时被海浪冲走，所以anchor标签的意思也是标识文档中的特定位置。

a标签其实同时充当了链接和目标点的角色，当a标签有href属性时，它是链接，当它有name时，它是链接的目标。

具有href的a标签跟一些link一样，会产生超链接，也就是在用户不操作的情况下，它们不会被主动下载的被动型链接。

重点的内容是，a标签也可以有rel属性，我们来简单了解一下，首先是跟link相同的一些rel，包括下面的几种。

- alternate
- author
- help
- license
- next
- prev

- search

这些跟link语义完全一致，不同的是，a标签产生的链接是会实际显示在网页中的，而link标签仅仅是元信息。

除了这些之外，a标签独有的rel类型：

- tag 表示本网页所属的标签；
- bookmark 到上级章节的链接。

a标签还有一些辅助的rel类型，用于提示浏览器或者搜索引擎做一些处理：

- nofollow 此链接不会被搜索引擎索引；
- noopener 此链接打开的网页无法使用opener来获得当前页面的窗口；
- noreferrer 此链接打开的网页无法使用referrer来获得当前页面的url；
- opener 打开的网页可以使用window.opener来访问当前页面的window对象，这是a标签的默认行为。

a标签基本解决了在页面中插入文字型和整张图片超链接的需要，但是如果我们想要在图片的某个区域产生超链接，那么就要用到另一种标签了——area标签。

area 标签

area标签与a标签非常相似，不同的是，它不是文本型的链接，而是区域型的链接。

area标签支持的rel与a完全一样，这里就不多说了。

area是整个html规则中唯一支持非矩形热区的标签，它的shape属性支持三种类型。

- 圆形：circle或者circ，coords支持三个值，分别表示中心点的x,y坐标和圆形半径r。
- 矩形：rect或者rectangle，coords支持两个值，分别表示两个对角顶点x1，y1和x2，y2。
- 多边形：poly或者polygon，coords至少包括6个值，表示多边形的各个顶点。

因为area设计的时间较早，所以不支持含有各种曲线的路径，但是它也是唯一一个支持了非矩形触发区域的元素，所以，对于一些效果而言，area是必不可少的。

area必须跟img和map标签配合使用。使用示例如下（例子来自html标准）。

```
<p>
 Please select a shape:
 <img src="shapes.png" usemap="#shapes"
      alt="Four shapes are available: a red hollow box, a green circle, a blue triangle, and a yellow four-pointed star.">
 <map name="shapes">
  <area shape=rect coords="50,50,100,100"> <!-- the hole in the red box -->
  <area shape=rect coords="25,25,125,125" href="red.html" alt="Red box.">
  <area shape=circle coords="200,75,50" href="green.html" alt="Green circle.">
  <area shape=poly coords="325,25,262,125,388,125" href="blue.html" alt="Blue triangle.">
  <area shape=poly coords="450,25,435,60,400,75,435,90,450,125,465,90,500,75,465,60"
        href="yellow.html" alt="Yellow star.">
 </map>
</p>
```

这个例子展示了在一张图片上画热区并且产生链接，分别使用了矩形、圆形和多边形三种area。

结语

本节课我们介绍了几种链接类型。在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

我们逐次讲到了link标签、a标签和area标签，link标签一般用于看不见的链接，它可能产生超链接或者外部资源链接，a和area一般用于页面上显示的链接，它们只能产生超链接。

最后，留给你一个思考问题，你的工作中，是使用过哪些类型的link标签的呢？

猜你喜欢

你好，我是winter。

在前面的课程中，我讲到了HTML的语义和元信息标签，今天这一课，我们来讲另一类HTML元素：链接。

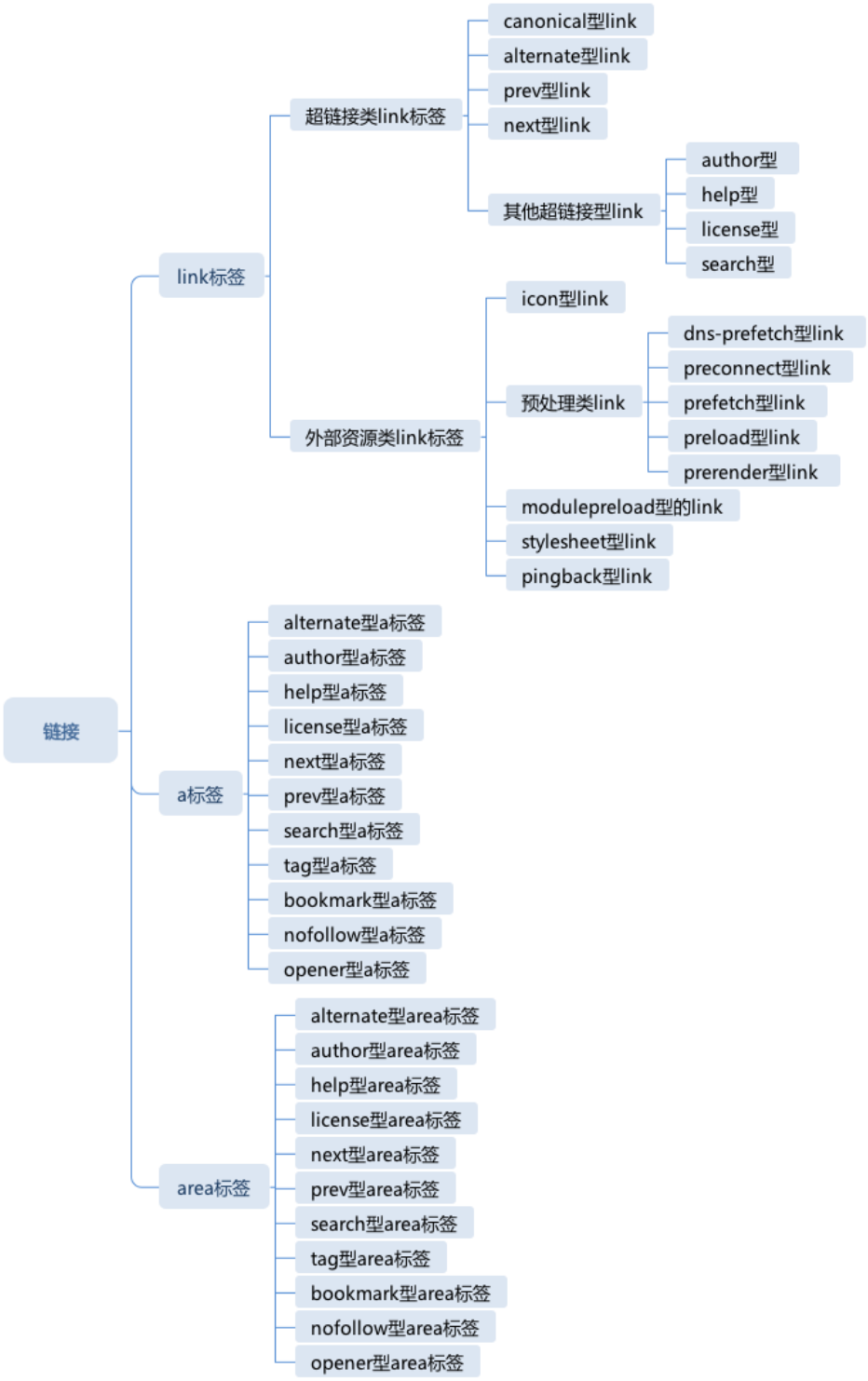
链接这种元素可以说是占据了整个互联网。也正是因为无处不在的超链接，才让我们的万维网如此繁荣。没有了超链接的HTML，最多可以称为富文本，没法称作超文本（hyper text）。

我想，作为互联网从业者，我们一定对链接都非常熟悉了。链接能够帮助我们从一个网页跳转到另一个网页。

不过，除了肉眼可见的这些链接，其实HTML里面还规定了一些不可见链接的类型，这节课，我就来给你介绍链接家族的全员，让你对它们有一个完整的认识。

链接是HTML中的一种机制，它是HTML文档和其它文档或者资源的连接关系，在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

链接的家族中有a标签、area标签和link标签。今天，我会逐一对它们进行介绍。



link 标签

提到链接，我们都知道a标签可以成为超链接，但是我们今天的内容，要从一个大家不太熟悉的标签开始，也就是link标签。

我们已经介绍过元信息类标签。实际上，我们并没有介绍完全，有些link标签也是元信息类标签的一种。

我们已经讲过，HTML标准并没有规定浏览器如何使用元信息，我们还讲到了元信息中有不少是被设计成“无需被浏览器识别，而是专门用于搜索引擎看的”。

link标签也是元信息的一种，在很多时候，它也是不会对浏览器产生任何效果的，这也是很多人会忽略link标签学习的原因。

link标签会生成一个链接，它可能生成超链接，也可能生成外部资源链接。

一些link标签会生成超链接，这些超链接又不会像a标签那样显示在网页中。这就是超链接型的link标签。

这意味着多数浏览器中，这些link标签不产生任何作用。但是，这些link标签能够被搜索引擎和一些浏览器插件识别，从而产生关键性作用。

比如，到页面RSS的link标签，能够被浏览器的RSS订阅插件识别，提示用户当前页面是可以RSS订阅的。

另外一些link标签则会把外部的资源链接到文档中，也就是说，会实际下载这些资源，并且做出一些处理，比如我们常见的用link标签引入样式表。

除了元信息的用法之外，多数外部资源型的link标签还能够被放在body中使用，从而起到把外部资源链接进文档的作用。

link标签的链接类型主要通过rel属性来区分，在本篇文章中，我们提到xx型link即表示属性rel为xx的link，其代码类似下面：

```
<link rel="xx" ...>
```

下面我们先来看看超链接型link标签。

超链接类link标签

超链接型link标签是一种被动型链接，在用户不操作的情况下，它们不会被主动下载。

link标签具有特定的rel属性，会成为特定类型的link标签。产生超链接的link标签包括：具有 rel=“canonical” 的link、具有 rel="alternate"的link、具有 rel=“prev” rel="next"的link等等。

canonical型link

这种link的代码写法是这样：

```
<link rel="canonical" href="...">
```

这个标签提示页面它的主URL，在网站中常常有多个URL指向同一页面的情况，搜索引擎访问这类页面时会去掉重复的页面，这个link会提示搜索引擎保留哪一个URL。

alternate型link

这种link的代码写法是这样：

```
<link rel="alternate" href="...">
```

这个标签提示页面它的变形形式，这个所谓的变形可能是当前页面内容的不同格式、不同语言或者为不同的设备设计的版本，这种link通常也是提供给搜索引擎来使用的。

alternate型的link的一个典型应用场景是，页面提供rss订阅时，可以用这样的link来引入：

```
<link rel="alternate" type="application/rss+xml" title="RSS" href="...">
```

除了搜索引擎外，很多浏览器插件都能识别这样的link。

prev型link和next型link

在互联网应用中，很多网页都属于一个序列，比如分页浏览的场景，或者图片展示的场景，每个网页是序列中的一个项。

这种时候，就适合使用prev和next型的link标签，来告诉搜索引擎或者浏览器它的前一项和后一项，这有助于页面的批量展示。

因为next型link告诉浏览器“这是很可能访问的下一个页面”，HTML标准还建议对next型link做预处理，在本课后面的内容，我们会讲到预处理类的link。

其它超链接类的link

其它超链接类link标签都表示一个跟当前文档相关联的信息，可以把这样的link标签视为一种带链接功能的meta标签。

- rel=“author” 链接到本页面的作者，一般是 mailto:协议
- rel=“help” 链接到本页面的帮助页
- rel=“license” 链接到本页面的版权信息页
- rel=“search” 链接到本页面的搜索页面（一般是站内提供搜索时使用）

到这里，我们已经讲完了所有的超链接类的link标签用法了。接下来我们讲讲外部资源类link标签。

外部资源类link标签

外部资源型link标签会被主动下载，并且根据rel类型做不同的处理。外部资源型的标签包括：具有icon型的link、预处理类link、modulepreload型的link、stylesheet、pingback。下面我们来一一介绍它们。

icon型link

这类链接表示页面的icon。多数浏览器会读取icon型link，并且把页面的icon展示出来。

icon型link是唯一一个外部资源类的元信息link，其它元信息类link都是超链接，这意味着，icon型link中的图标地址默认会被浏览器下载和使用。

如果没有指定这样的link，多数浏览器会使用域名根目录下的favicon.ico，即使它并不存在，所以从性能的角度考虑，建议一定要保证页面中有icon型的link。

只有icon型link有有效的sizes属性，HTML标准允许一个页面出现多个icon型link，并且用sizes指定它适合的icon尺寸。

预处理类link

我们都知道，导航到一个网站需要经过dns查询域名、建立连接、传输数据、加载进内存和渲染等一系列的步骤。

预处理类link标签就是允许我们控制浏览器，提前针对一些资源去做这些操作，以提高性能（当然如果你乱用的话，性能反而更差）。

下面我来列一下这些link类型：

- dns-prefetch型link 提前对一个域名做dns查询，这样的link里面的href实际上只有域名有意义。
- preconnect型link 提前对一个服务器建立tcp连接。
- prefetch型link 提前取href指定的url的内容。
- preload型link 提前加载href指定的url。
- prerender型link 提前渲染href指定的url。

modulepreload型的link

modulepreload型link的作用是预先加载一个JavaScript的模块。这可以保证JS模块不必等到执行时才加载。

这里的所谓加载，是指完成下载并放入内存，并不会执行对应的JavaScript。

```
<link rel="modulepreload" href="app.js">
<link rel="modulepreload" href="helpers.js">
<link rel="modulepreload" href="irc.js">
<link rel="modulepreload" href="fog-machine.js">
<script type="module" src="app.js">
```

这个例子来自HTML标准，我们假设app.js中有 import “irc” 和 import “fog-machine”, 而 irc.js中有 import “helpers”。这段代码使用moduleload型link来预加载了四个js模块。

尽管，单独使用script标签引用app.js也可以正常工作，但是我们通过加入对四个JS文件的link标签，使得四个JS文件有机会被并行地下载，这样提高了性能。

stylesheet型link

样式表大概是所有人最熟悉的link标签用法了。它的样子是下面这样的。

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

基本用法是从一个CSS文件创建一个样式表。这里type属性可以没有，如果有，必须是"text/css"才会生效。

rel前可以加上alternate，成为rel=“alternate stylesheet”，此时必须再指定title属性。

这样可以为页面创建一份变体样式，一些浏览器，如 Firefox 3.0，支持从浏览器菜单中切换这些样式，当然了，大部分浏览器不支持这个功能，所以仅仅从语义的角度了解一下这种用法即可。

pingback型link

这样的link表示本网页被引用时，应该使用的pingback地址，这个机制是一份独立的标准，遵守pingback协议的网站在引用本页面时，会向这个pingback url发送一个消息。

以上就是link标签的所有用法了。接下来我们来介绍一下最熟悉的 a 标签，当然了，也可能你学过了本节课以后，觉得自己其实也没那么熟悉。

a 标签

a标签是“anchor”的缩写，它是锚点的意思，所谓锚点，实际上也是一种比喻的用法，古代船舶用锚来固定自己的位置，避免停泊时被海浪冲走，所以anchor标签的意思也是标识文档中的特定位置。

a标签其实同时充当了链接和目标点的角色，当a标签有href属性时，它是链接，当它有name时，它是链接的目标。

具有href的a标签跟一些link一样，会产生超链接，也就是在用户不操作的情况下，它们不会被主动下载的被动型链接。

重点的内容是，a标签也可以有rel属性，我们来简单了解一下，首先是跟link相同的一些rel，包括下面的几种。

- alternate
- author
- help
- license
- next
- prev

- search

这些跟link语义完全一致，不同的是，a标签产生的链接是会实际显示在网页中的，而link标签仅仅是元信息。

除了这些之外，a标签独有的rel类型：

- tag 表示本网页所属的标签；
- bookmark 到上级章节的链接。

a标签还有一些辅助的rel类型，用于提示浏览器或者搜索引擎做一些处理：

- nofollow 此链接不会被搜索引擎索引；
- noopener 此链接打开的网页无法使用opener来获得当前页面的窗口；
- noreferrer 此链接打开的网页无法使用referrer来获得当前页面的url；
- opener 打开的网页可以使用window.opener来访问当前页面的window对象，这是a标签的默认行为。

a标签基本解决了在页面中插入文字型和整张图片超链接的需要，但是如果我们想要在图片的某个区域产生超链接，那么就要用到另一种标签了——area标签。

area 标签

area标签与a标签非常相似，不同的是，它不是文本型的链接，而是区域型的链接。

area标签支持的rel与a完全一样，这里就不多说了。

area是整个html规则中唯一支持非矩形热区的标签，它的shape属性支持三种类型。

- 圆形：circle或者circ，coords支持三个值，分别表示中心点的x,y坐标和圆形半径r。
- 矩形：rect或者rectangle，coords支持两个值，分别表示两个对角顶点x1，y1和x2，y2。
- 多边形：poly或者polygon，coords至少包括6个值，表示多边形的各个顶点。

因为area设计的时间较早，所以不支持含有各种曲线的路径，但是它也是唯一一个支持了非矩形触发区域的元素，所以，对于一些效果而言，area是必不可少的。

area必须跟img和map标签配合使用。使用示例如下（例子来自html标准）。

```
<p>
 Please select a shape:
 <img src="shapes.png" usemap="#shapes"
      alt="Four shapes are available: a red hollow box, a green circle, a blue triangle, and a yellow four-pointed star.">
 <map name="shapes">
  <area shape=rect coords="50,50,100,100"> <!-- the hole in the red box -->
  <area shape=rect coords="25,25,125,125" href="red.html" alt="Red box.">
  <area shape=circle coords="200,75,50" href="green.html" alt="Green circle.">
  <area shape=poly coords="325,25,262,125,388,125" href="blue.html" alt="Blue triangle.">
  <area shape=poly coords="450,25,435,60,400,75,435,90,450,125,465,90,500,75,465,60"
        href="yellow.html" alt="Yellow star.">
 </map>
</p>
```

这个例子展示了在一张图片上画热区并且产生链接，分别使用了矩形、圆形和多边形三种area。

结语

本节课我们介绍了几种链接类型。在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

我们逐次讲到了link标签、a标签和area标签，link标签一般用于看不见的链接，它可能产生超链接或者外部资源链接，a和area一般用于页面上显示的链接，它们只能产生超链接。

最后，留给你一个思考问题，你的工作中，是使用过哪些类型的link标签的呢？

猜你喜欢

你好，我是winter。

在前面的课程中，我讲到了HTML的语义和元信息标签，今天这一课，我们来讲另一类HTML元素：链接。

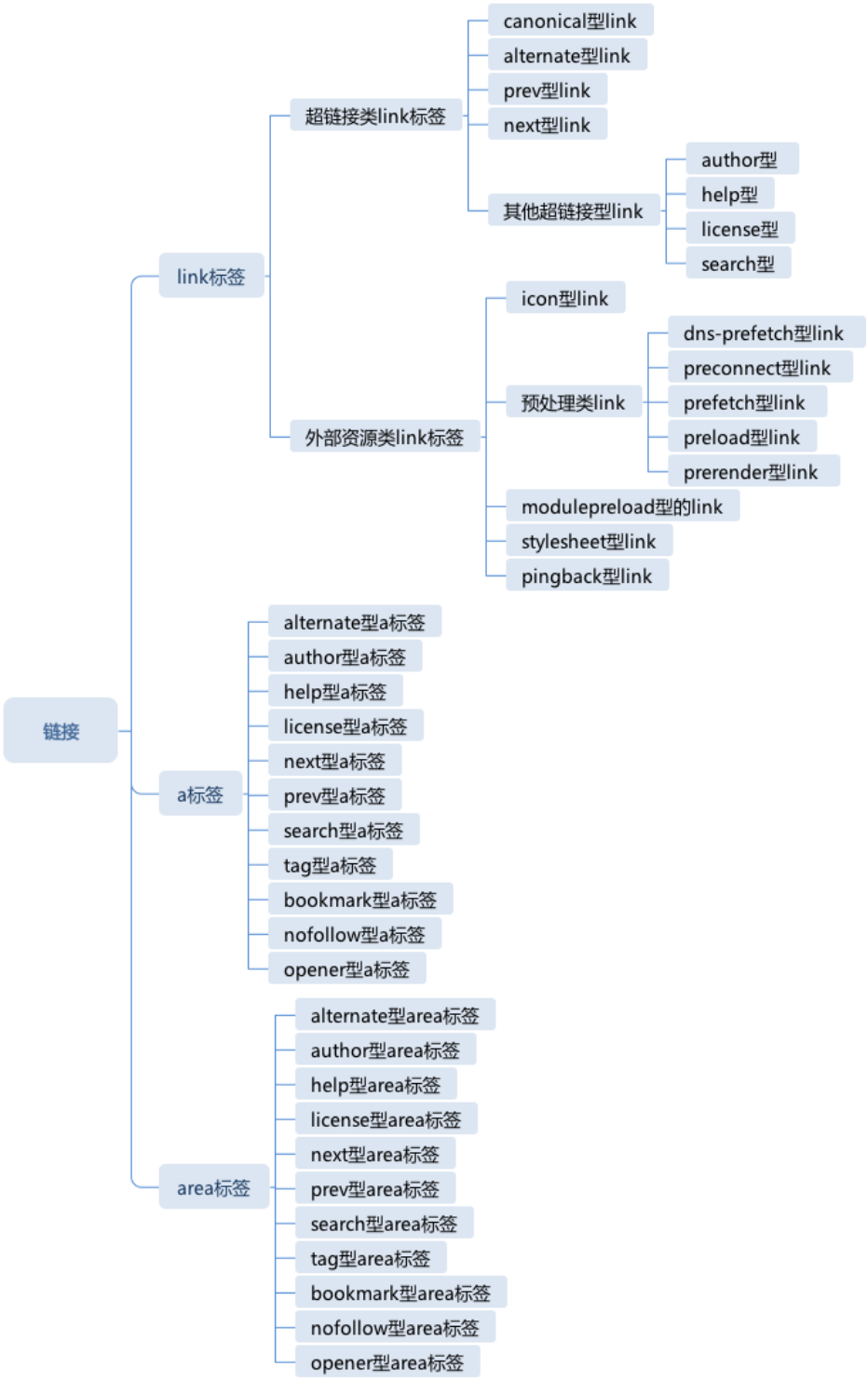
链接这种元素可以说是占据了整个互联网。也正是因为无处不在的超链接，才让我们的万维网如此繁荣。没有了超链接的HTML，最多可以称为富文本，没法称作超文本（hyper text）。

我想，作为互联网从业者，我们一定对链接都非常熟悉了。链接能够帮助我们从一个网页跳转到另一个网页。

不过，除了肉眼可见的这些链接，其实HTML里面还规定了一些不可见链接的类型，这节课，我就来给你介绍链接家族的全员，让你对它们有一个完整的认识。

链接是HTML中的一种机制，它是HTML文档和其它文档或者资源的连接关系，在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

链接的家族中有a标签、area标签和link标签。今天，我会逐一对它们进行介绍。



link 标签

提到链接，我们都知道a标签可以成为超链接，但是我们今天的内容，要从一个大家不太熟悉的标签开始，也就是link标签。

我们已经介绍过元信息类标签。实际上，我们并没有介绍完全，有些link标签也是元信息类标签的一种。

我们已经讲过，HTML标准并没有规定浏览器如何使用元信息，我们还讲到了元信息中有不少是被设计成“无需被浏览器识别，而是专门用于搜索引擎看的”。

link标签也是元信息的一种，在很多时候，它也是不会对浏览器产生任何效果的，这也是很多人会忽略link标签学习的原因。

link标签会生成一个链接，它可能生成超链接，也可能生成外部资源链接。

一些link标签会生成超链接，这些超链接又不会像a标签那样显示在网页中。这就是超链接型的link标签。

这意味着多数浏览器中，这些link标签不产生任何作用。但是，这些link标签能够被搜索引擎和一些浏览器插件识别，从而产生关键性作用。

比如，到页面RSS的link标签，能够被浏览器的RSS订阅插件识别，提示用户当前页面是可以RSS订阅的。

另外一些link标签则会把外部的资源链接到文档中，也就是说，会实际下载这些资源，并且做出一些处理，比如我们常见的用link标签引入样式表。

除了元信息的用法之外，多数外部资源型的link标签还能够被放在body中使用，从而起到把外部资源链接进文档的作用。

link标签的链接类型主要通过rel属性来区分，在本篇文章中，我们提到xx型link即表示属性rel为xx的link，其代码类似下面：

```
<link rel="xx" ...>
```

下面我们先来看看超链接型link标签。

超链接类link标签

超链接型link标签是一种被动型链接，在用户不操作的情况下，它们不会被主动下载。

link标签具有特定的rel属性，会成为特定类型的link标签。产生超链接的link标签包括：具有 rel=“canonical” 的link、具有 rel="alternate"的link、具有 rel=“prev” rel="next"的link等等。

canonical型link

这种link的代码写法是这样：

```
<link rel="canonical" href="...">
```

这个标签提示页面它的主URL，在网站中常常有多个URL指向同一页面的情况，搜索引擎访问这类页面时会去掉重复的页面，这个link会提示搜索引擎保留哪一个URL。

alternate型link

这种link的代码写法是这样：

```
<link rel="alternate" href="...">
```

这个标签提示页面它的变形形式，这个所谓的变形可能是当前页面内容的不同格式、不同语言或者为不同的设备设计的版本，这种link通常也是提供给搜索引擎来使用的。

alternate型的link的一个典型应用场景是，页面提供rss订阅时，可以用这样的link来引入：

```
<link rel="alternate" type="application/rss+xml" title="RSS" href="...">
```

除了搜索引擎外，很多浏览器插件都能识别这样的link。

prev型link和next型link

在互联网应用中，很多网页都属于一个序列，比如分页浏览的场景，或者图片展示的场景，每个网页是序列中的一个项。

这种时候，就适合使用prev和next型的link标签，来告诉搜索引擎或者浏览器它的前一项和后一项，这有助于页面的批量展示。

因为next型link告诉浏览器“这是很可能访问的下一个页面”，HTML标准还建议对next型link做预处理，在本课后面的内容，我们会讲到预处理类的link。

其它超链接类的link

其它超链接类link标签都表示一个跟当前文档相关联的信息，可以把这样的link标签视为一种带链接功能的meta标签。

- rel=“author” 链接到本页面的作者，一般是 mailto:协议
- rel=“help” 链接到本页面的帮助页
- rel=“license” 链接到本页面的版权信息页
- rel=“search” 链接到本页面的搜索页面（一般是站内提供搜索时使用）

到这里，我们已经讲完了所有的超链接类的link标签用法了。接下来我们讲讲外部资源类link标签。

外部资源类link标签

外部资源型link标签会被主动下载，并且根据rel类型做不同的处理。外部资源型的标签包括：具有icon型的link、预处理类link、modulepreload型的link、stylesheet、pingback。下面我们来一一介绍它们。

icon型link

这类链接表示页面的icon。多数浏览器会读取icon型link，并且把页面的icon展示出来。

icon型link是唯一一个外部资源类的元信息link，其它元信息类link都是超链接，这意味着，icon型link中的图标地址默认会被浏览器下载和使用。

如果没有指定这样的link，多数浏览器会使用域名根目录下的favicon.ico，即使它并不存在，所以从性能的角度考虑，建议一定要保证页面中有icon型的link。

只有icon型link有有效的sizes属性，HTML标准允许一个页面出现多个icon型link，并且用sizes指定它适合的icon尺寸。

预处理类link

我们都知道，导航到一个网站需要经过dns查询域名、建立连接、传输数据、加载进内存和渲染等一系列的步骤。

预处理类link标签就是允许我们控制浏览器，提前针对一些资源去做这些操作，以提高性能（当然如果你乱用的话，性能反而更差）。

下面我来列一下这些link类型：

- dns-prefetch型link 提前对一个域名做dns查询，这样的link里面的href实际上只有域名有意义。
- preconnect型link 提前对一个服务器建立tcp连接。
- prefetch型link 提前取href指定的url的内容。
- preload型link 提前加载href指定的url。
- prerender型link 提前渲染href指定的url。

modulepreload型的link

modulepreload型link的作用是预先加载一个JavaScript的模块。这可以保证JS模块不必等到执行时才加载。

这里的所谓加载，是指完成下载并放入内存，并不会执行对应的JavaScript。

```
<link rel="modulepreload" href="app.js">
<link rel="modulepreload" href="helpers.js">
<link rel="modulepreload" href="irc.js">
<link rel="modulepreload" href="fog-machine.js">
<script type="module" src="app.js">
```

这个例子来自HTML标准，我们假设app.js中有 import “irc” 和 import “fog-machine”, 而 irc.js中有 import “helpers”。这段代码使用moduleload型link来预加载了四个js模块。

尽管，单独使用script标签引用app.js也可以正常工作，但是我们通过加入对四个JS文件的link标签，使得四个JS文件有机会被并行地下载，这样提高了性能。

stylesheet型link

样式表大概是所有人最熟悉的link标签用法了。它的样子是下面这样的。

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

基本用法是从一个CSS文件创建一个样式表。这里type属性可以没有，如果有，必须是"text/css"才会生效。

rel前可以加上alternate，成为rel=“alternate stylesheet”，此时必须再指定title属性。

这样可以为页面创建一份变体样式，一些浏览器，如 Firefox 3.0，支持从浏览器菜单中切换这些样式，当然了，大部分浏览器不支持这个功能，所以仅仅从语义的角度了解一下这种用法即可。

pingback型link

这样的link表示本网页被引用时，应该使用的pingback地址，这个机制是一份独立的标准，遵守pingback协议的网站在引用本页面时，会向这个pingback url发送一个消息。

以上就是link标签的所有用法了。接下来我们来介绍一下最熟悉的 a 标签，当然了，也可能你学过了本节课以后，觉得自己其实也没那么熟悉。

a 标签

a标签是“anchor”的缩写，它是锚点的意思，所谓锚点，实际上也是一种比喻的用法，古代船舶用锚来固定自己的位置，避免停泊时被海浪冲走，所以anchor标签的意思也是标识文档中的特定位置。

a标签其实同时充当了链接和目标点的角色，当a标签有href属性时，它是链接，当它有name时，它是链接的目标。

具有href的a标签跟一些link一样，会产生超链接，也就是在用户不操作的情况下，它们不会被主动下载的被动型链接。

重点的内容是，a标签也可以有rel属性，我们来简单了解一下，首先是跟link相同的一些rel，包括下面的几种。

- alternate
- author
- help
- license
- next
- prev

- search

这些跟link语义完全一致，不同的是，a标签产生的链接是会实际显示在网页中的，而link标签仅仅是元信息。

除了这些之外，a标签独有的rel类型：

- tag 表示本网页所属的标签；
- bookmark 到上级章节的链接。

a标签还有一些辅助的rel类型，用于提示浏览器或者搜索引擎做一些处理：

- nofollow 此链接不会被搜索引擎索引；
- noopener 此链接打开的网页无法使用opener来获得当前页面的窗口；
- noreferrer 此链接打开的网页无法使用referrer来获得当前页面的url；
- opener 打开的网页可以使用window.opener来访问当前页面的window对象，这是a标签的默认行为。

a标签基本解决了在页面中插入文字型和整张图片超链接的需要，但是如果我们想要在图片的某个区域产生超链接，那么就要用到另一种标签了——area标签。

area 标签

area标签与a标签非常相似，不同的是，它不是文本型的链接，而是区域型的链接。

area标签支持的rel与a完全一样，这里就不多说了。

area是整个html规则中唯一支持非矩形热区的标签，它的shape属性支持三种类型。

- 圆形：circle或者circ，coords支持三个值，分别表示中心点的x,y坐标和圆形半径r。
- 矩形：rect或者rectangle，coords支持两个值，分别表示两个对角顶点x1，y1和x2，y2。
- 多边形：poly或者polygon，coords至少包括6个值，表示多边形的各个顶点。

因为area设计的时间较早，所以不支持含有各种曲线的路径，但是它也是唯一一个支持了非矩形触发区域的元素，所以，对于一些效果而言，area是必不可少的。

area必须跟img和map标签配合使用。使用示例如下（例子来自html标准）。

```
<p>
 Please select a shape:
 <img src="shapes.png" usemap="#shapes"
      alt="Four shapes are available: a red hollow box, a green circle, a blue triangle, and a yellow four-pointed star.">
 <map name="shapes">
  <area shape=rect coords="50,50,100,100"> <!-- the hole in the red box -->
  <area shape=rect coords="25,25,125,125" href="red.html" alt="Red box.">
  <area shape=circle coords="200,75,50" href="green.html" alt="Green circle.">
  <area shape=poly coords="325,25,262,125,388,125" href="blue.html" alt="Blue triangle.">
  <area shape=poly coords="450,25,435,60,400,75,435,90,450,125,465,90,500,75,465,60"
        href="yellow.html" alt="Yellow star.">
 </map>
</p>
```

这个例子展示了在一张图片上画热区并且产生链接，分别使用了矩形、圆形和多边形三种area。

结语

本节课我们介绍了几种链接类型。在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

我们逐次讲到了link标签、a标签和area标签，link标签一般用于看不见的链接，它可能产生超链接或者外部资源链接，a和area一般用于页面上显示的链接，它们只能产生超链接。

最后，留给你一个思考问题，你的工作中，是使用过哪些类型的link标签的呢？

猜你喜欢

你好，我是winter。

在前面的课程中，我讲到了HTML的语义和元信息标签，今天这一课，我们来讲另一类HTML元素：链接。

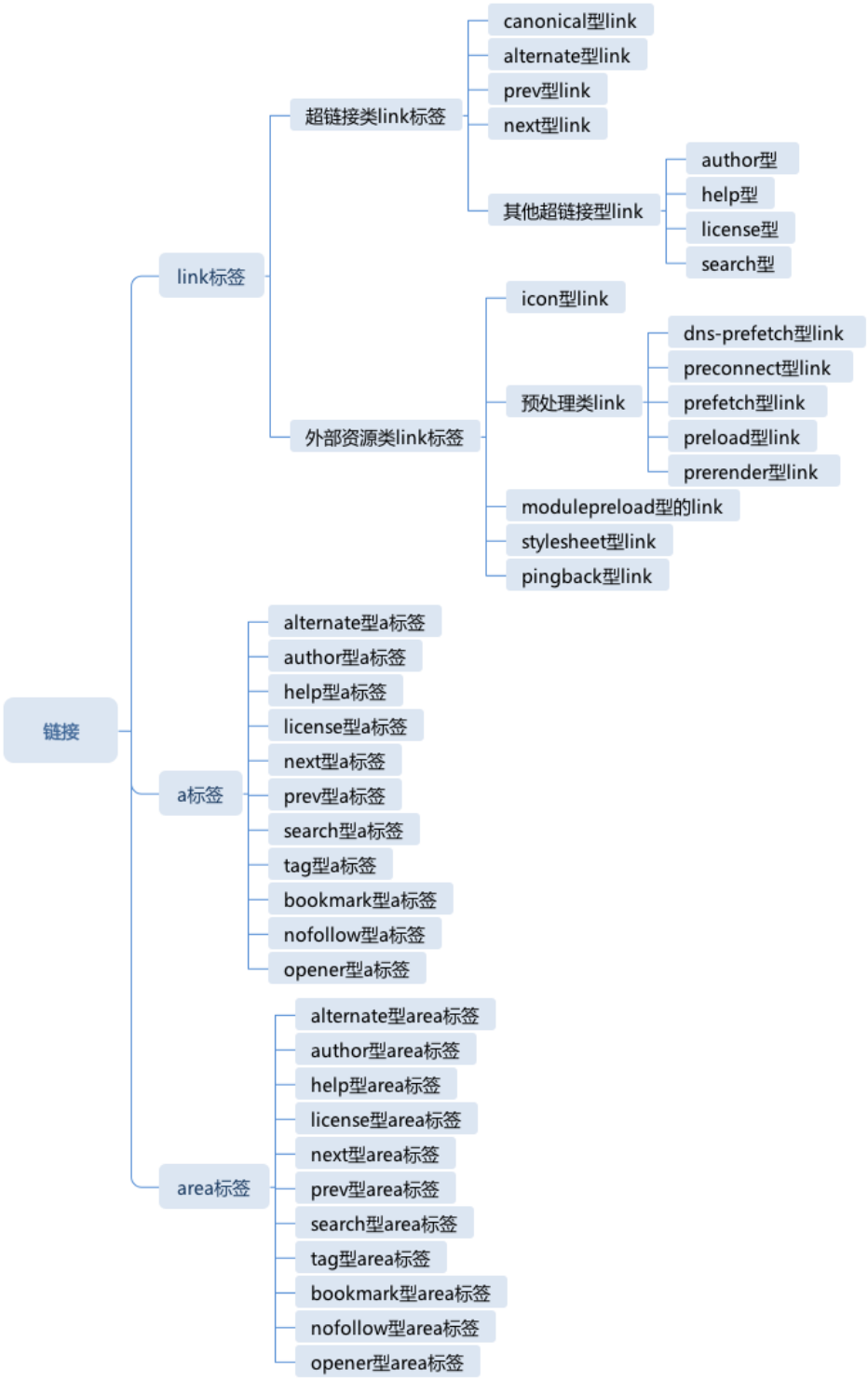
链接这种元素可以说是占据了整个互联网。也正是因为无处不在的超链接，才让我们的万维网如此繁荣。没有了超链接的HTML，最多可以称为富文本，没法称作超文本（hyper text）。

我想，作为互联网从业者，我们一定对链接都非常熟悉了。链接能够帮助我们从一个网页跳转到另一个网页。

不过，除了肉眼可见的这些链接，其实HTML里面还规定了一些不可见链接的类型，这节课，我就来给你介绍链接家族的全员，让你对它们有一个完整的认识。

链接是HTML中的一种机制，它是HTML文档和其它文档或者资源的连接关系，在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

链接的家族中有a标签、area标签和link标签。今天，我会逐一对它们进行介绍。



link 标签

提到链接，我们都知道a标签可以成为超链接，但是我们今天的内容，要从一个大家不太熟悉的标签开始，也就是link标签。

我们已经介绍过元信息类标签。实际上，我们并没有介绍完全，有些link标签也是元信息类标签的一种。

我们已经讲过，HTML标准并没有规定浏览器如何使用元信息，我们还讲到了元信息中有不少是被设计成“无需被浏览器识别，而是专门用于搜索引擎看的”。

link标签也是元信息的一种，在很多时候，它也是不会对浏览器产生任何效果的，这也是很多人会忽略link标签学习的原因。

link标签会生成一个链接，它可能生成超链接，也可能生成外部资源链接。

一些link标签会生成超链接，这些超链接又不会像a标签那样显示在网页中。这就是超链接型的link标签。

这意味着多数浏览器中，这些link标签不产生任何作用。但是，这些link标签能够被搜索引擎和一些浏览器插件识别，从而产生关键性作用。

比如，到页面RSS的link标签，能够被浏览器的RSS订阅插件识别，提示用户当前页面是可以RSS订阅的。

另外一些link标签则会把外部的资源链接到文档中，也就是说，会实际下载这些资源，并且做出一些处理，比如我们常见的用link标签引入样式表。

除了元信息的用法之外，多数外部资源型的link标签还能够被放在body中使用，从而起到把外部资源链接进文档的作用。

link标签的链接类型主要通过rel属性来区分，在本篇文章中，我们提到xx型link即表示属性rel为xx的link，其代码类似下面：

```
<link rel="xx" ...>
```

下面我们先来看看超链接型link标签。

超链接类link标签

超链接型link标签是一种被动型链接，在用户不操作的情况下，它们不会被主动下载。

link标签具有特定的rel属性，会成为特定类型的link标签。产生超链接的link标签包括：具有 rel=“canonical” 的link、具有 rel="alternate"的link、具有 rel=“prev” rel="next"的link等等。

canonical型link

这种link的代码写法是这样：

```
<link rel="canonical" href="...">
```

这个标签提示页面它的主URL，在网站中常常有多个URL指向同一页面的情况，搜索引擎访问这类页面时会去掉重复的页面，这个link会提示搜索引擎保留哪一个URL。

alternate型link

这种link的代码写法是这样：

```
<link rel="alternate" href="...">
```

这个标签提示页面它的变形形式，这个所谓的变形可能是当前页面内容的不同格式、不同语言或者为不同的设备设计的版本，这种link通常也是提供给搜索引擎来使用的。

alternate型的link的一个典型应用场景是，页面提供rss订阅时，可以用这样的link来引入：

```
<link rel="alternate" type="application/rss+xml" title="RSS" href="...">
```

除了搜索引擎外，很多浏览器插件都能识别这样的link。

prev型link和next型link

在互联网应用中，很多网页都属于一个序列，比如分页浏览的场景，或者图片展示的场景，每个网页是序列中的一个项。

这种时候，就适合使用prev和next型的link标签，来告诉搜索引擎或者浏览器它的前一项和后一项，这有助于页面的批量展示。

因为next型link告诉浏览器“这是很可能访问的下一个页面”，HTML标准还建议对next型link做预处理，在本课后面的内容，我们会讲到预处理类的link。

其它超链接类的link

其它超链接类link标签都表示一个跟当前文档相关联的信息，可以把这样的link标签视为一种带链接功能的meta标签。

- rel=“author” 链接到本页面的作者，一般是 mailto:协议
- rel=“help” 链接到本页面的帮助页
- rel=“license” 链接到本页面的版权信息页
- rel=“search” 链接到本页面的搜索页面（一般是站内提供搜索时使用）

到这里，我们已经讲完了所有的超链接类的link标签用法了。接下来我们讲讲外部资源类link标签。

外部资源类link标签

外部资源型link标签会被主动下载，并且根据rel类型做不同的处理。外部资源型的标签包括：具有icon型的link、预处理类link、modulepreload型的link、stylesheet、pingback。下面我们来一一介绍它们。

icon型link

这类链接表示页面的icon。多数浏览器会读取icon型link，并且把页面的icon展示出来。

icon型link是唯一一个外部资源类的元信息link，其它元信息类link都是超链接，这意味着，icon型link中的图标地址默认会被浏览器下载和使用。

如果没有指定这样的link，多数浏览器会使用域名根目录下的favicon.ico，即使它并不存在，所以从性能的角度考虑，建议一定要保证页面中有icon型的link。

只有icon型link有有效的sizes属性，HTML标准允许一个页面出现多个icon型link，并且用sizes指定它适合的icon尺寸。

预处理类link

我们都知道，导航到一个网站需要经过dns查询域名、建立连接、传输数据、加载进内存和渲染等一系列的步骤。

预处理类link标签就是允许我们控制浏览器，提前针对一些资源去做这些操作，以提高性能（当然如果你乱用的话，性能反而更差）。

下面我来列一下这些link类型：

- dns-prefetch型link 提前对一个域名做dns查询，这样的link里面的href实际上只有域名有意义。
- preconnect型link 提前对一个服务器建立tcp连接。
- prefetch型link 提前取href指定的url的内容。
- preload型link 提前加载href指定的url。
- prerender型link 提前渲染href指定的url。

modulepreload型的link

modulepreload型link的作用是预先加载一个JavaScript的模块。这可以保证JS模块不必等到执行时才加载。

这里的所谓加载，是指完成下载并放入内存，并不会执行对应的JavaScript。

```
<link rel="modulepreload" href="app.js">
<link rel="modulepreload" href="helpers.js">
<link rel="modulepreload" href="irc.js">
<link rel="modulepreload" href="fog-machine.js">
<script type="module" src="app.js">
```

这个例子来自HTML标准，我们假设app.js中有 import “irc” 和 import “fog-machine”, 而 irc.js中有 import “helpers”。这段代码使用moduleload型link来预加载了四个js模块。

尽管，单独使用script标签引用app.js也可以正常工作，但是我们通过加入对四个JS文件的link标签，使得四个JS文件有机会被并行地下载，这样提高了性能。

stylesheet型link

样式表大概是所有人最熟悉的link标签用法了。它的样子是下面这样的。

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

基本用法是从一个CSS文件创建一个样式表。这里type属性可以没有，如果有，必须是"text/css"才会生效。

rel前可以加上alternate，成为rel=“alternate stylesheet”，此时必须再指定title属性。

这样可以为页面创建一份变体样式，一些浏览器，如 Firefox 3.0，支持从浏览器菜单中切换这些样式，当然了，大部分浏览器不支持这个功能，所以仅仅从语义的角度了解一下这种用法即可。

pingback型link

这样的link表示本网页被引用时，应该使用的pingback地址，这个机制是一份独立的标准，遵守pingback协议的网站在引用本页面时，会向这个pingback url发送一个消息。

以上就是link标签的所有用法了。接下来我们来介绍一下最熟悉的 a 标签，当然了，也可能你学过了本节课以后，觉得自己其实也没那么熟悉。

a 标签

a标签是“anchor”的缩写，它是锚点的意思，所谓锚点，实际上也是一种比喻的用法，古代船舶用锚来固定自己的位置，避免停泊时被海浪冲走，所以anchor标签的意思也是标识文档中的特定位置。

a标签其实同时充当了链接和目标点的角色，当a标签有href属性时，它是链接，当它有name时，它是链接的目标。

具有href的a标签跟一些link一样，会产生超链接，也就是在用户不操作的情况下，它们不会被主动下载的被动型链接。

重点的内容是，a标签也可以有rel属性，我们来简单了解一下，首先是跟link相同的一些rel，包括下面的几种。

- alternate
- author
- help
- license
- next
- prev

- search

这些跟link语义完全一致，不同的是，a标签产生的链接是会实际显示在网页中的，而link标签仅仅是元信息。

除了这些之外，a标签独有的rel类型：

- tag 表示本网页所属的标签；
- bookmark 到上级章节的链接。

a标签还有一些辅助的rel类型，用于提示浏览器或者搜索引擎做一些处理：

- nofollow 此链接不会被搜索引擎索引；
- noopener 此链接打开的网页无法使用opener来获得当前页面的窗口；
- noreferrer 此链接打开的网页无法使用referrer来获得当前页面的url；
- opener 打开的网页可以使用window.opener来访问当前页面的window对象，这是a标签的默认行为。

a标签基本解决了在页面中插入文字型和整张图片超链接的需要，但是如果我们想要在图片的某个区域产生超链接，那么就要用到另一种标签了——area标签。

area 标签

area标签与a标签非常相似，不同的是，它不是文本型的链接，而是区域型的链接。

area标签支持的rel与a完全一样，这里就不多说了。

area是整个html规则中唯一支持非矩形热区的标签，它的shape属性支持三种类型。

- 圆形：circle或者circ，coords支持三个值，分别表示中心点的x,y坐标和圆形半径r。
- 矩形：rect或者rectangle，coords支持两个值，分别表示两个对角顶点x1，y1和x2，y2。
- 多边形：poly或者polygon，coords至少包括6个值，表示多边形的各个顶点。

因为area设计的时间较早，所以不支持含有各种曲线的路径，但是它也是唯一一个支持了非矩形触发区域的元素，所以，对于一些效果而言，area是必不可少的。

area必须跟img和map标签配合使用。使用示例如下（例子来自html标准）。

```
<p>
 Please select a shape:
 <img src="shapes.png" usemap="#shapes"
      alt="Four shapes are available: a red hollow box, a green circle, a blue triangle, and a yellow four-pointed star.">
 <map name="shapes">
  <area shape=rect coords="50,50,100,100"> <!-- the hole in the red box -->
  <area shape=rect coords="25,25,125,125" href="red.html" alt="Red box.">
  <area shape=circle coords="200,75,50" href="green.html" alt="Green circle.">
  <area shape=poly coords="325,25,262,125,388,125" href="blue.html" alt="Blue triangle.">
  <area shape=poly coords="450,25,435,60,400,75,435,90,450,125,465,90,500,75,465,60"
        href="yellow.html" alt="Yellow star.">
 </map>
</p>
```

这个例子展示了在一张图片上画热区并且产生链接，分别使用了矩形、圆形和多边形三种area。

结语

本节课我们介绍了几种链接类型。在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

我们逐次讲到了link标签、a标签和area标签，link标签一般用于看不见的链接，它可能产生超链接或者外部资源链接，a和area一般用于页面上显示的链接，它们只能产生超链接。

最后，留给你一个思考问题，你的工作中，是使用过哪些类型的link标签的呢？

猜你喜欢

你好，我是winter。

在前面的课程中，我讲到了HTML的语义和元信息标签，今天这一课，我们来讲另一类HTML元素：链接。

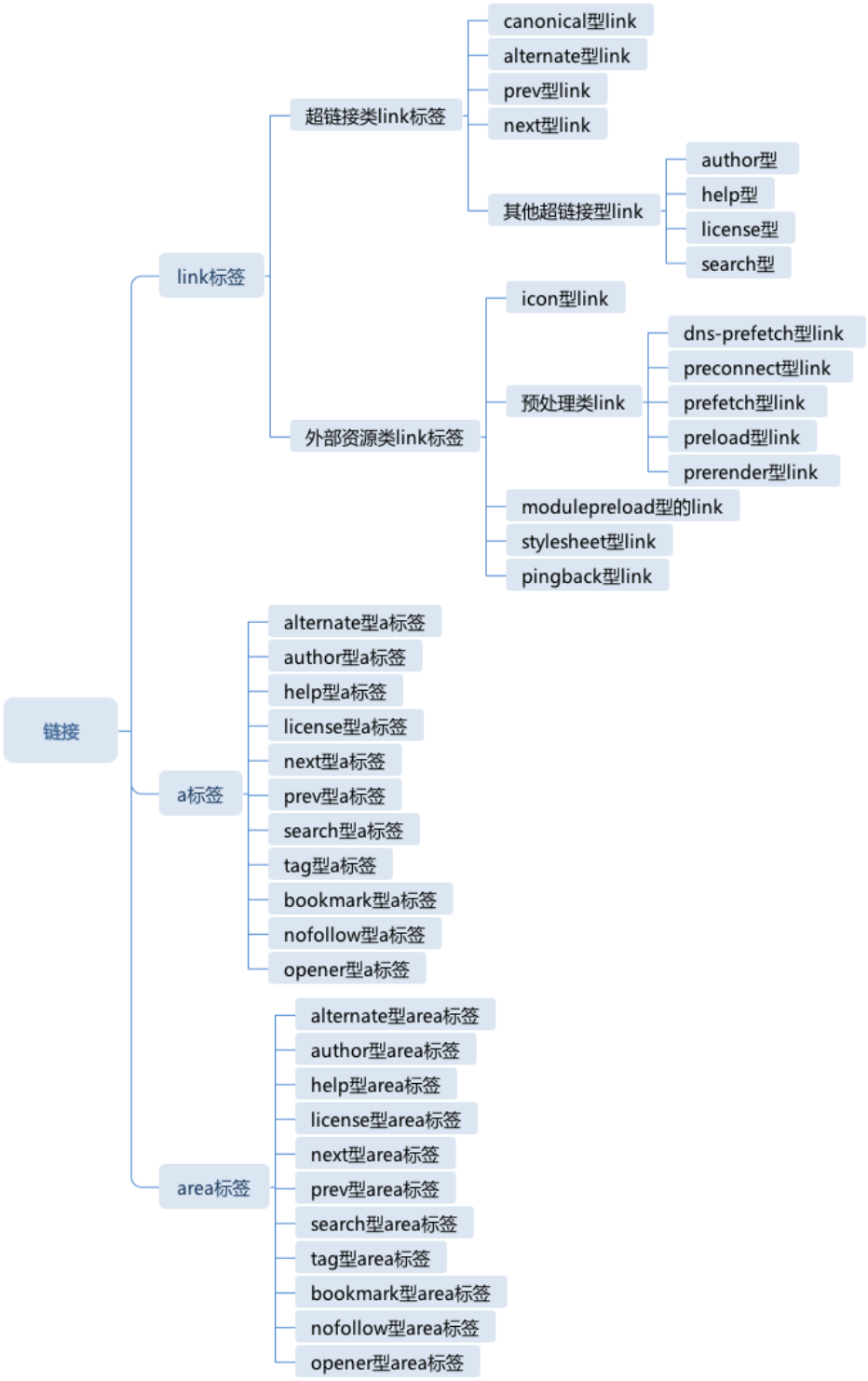
链接这种元素可以说是占据了整个互联网。也正是因为无处不在的超链接，才让我们的万维网如此繁荣。没有了超链接的HTML，最多可以称为富文本，没法称作超文本（hyper text）。

我想，作为互联网从业者，我们一定对链接都非常熟悉了。链接能够帮助我们从一个网页跳转到另一个网页。

不过，除了肉眼可见的这些链接，其实HTML里面还规定了一些不可见链接的类型，这节课，我就来给你介绍链接家族的全员，让你对它们有一个完整的认识。

链接是HTML中的一种机制，它是HTML文档和其它文档或者资源的连接关系，在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

链接的家族中有a标签、area标签和link标签。今天，我会逐一对它们进行介绍。



link 标签

提到链接，我们都知道a标签可以成为超链接，但是我们今天的内容，要从一个大家不太熟悉的标签开始，也就是link标签。

我们已经介绍过元信息类标签。实际上，我们并没有介绍完全，有些link标签也是元信息类标签的一种。

我们已经讲过，HTML标准并没有规定浏览器如何使用元信息，我们还讲到了元信息中有不少是被设计成“无需被浏览器识别，而是专门用于搜索引擎看的”。

link标签也是元信息的一种，在很多时候，它也是不会对浏览器产生任何效果的，这也是很多人会忽略link标签学习的原因。

link标签会生成一个链接，它可能生成超链接，也可能生成外部资源链接。

一些link标签会生成超链接，这些超链接又不会像a标签那样显示在网页中。这就是超链接型的link标签。

这意味着多数浏览器中，这些link标签不产生任何作用。但是，这些link标签能够被搜索引擎和一些浏览器插件识别，从而产生关键性作用。

比如，到页面RSS的link标签，能够被浏览器的RSS订阅插件识别，提示用户当前页面是可以RSS订阅的。

另外一些link标签则会把外部的资源链接到文档中，也就是说，会实际下载这些资源，并且做出一些处理，比如我们常见的用link标签引入样式表。

除了元信息的用法之外，多数外部资源型的link标签还能够被放在body中使用，从而起到把外部资源链接进文档的作用。

link标签的链接类型主要通过rel属性来区分，在本篇文章中，我们提到xx型link即表示属性rel为xx的link，其代码类似下面：

```
<link rel="xx" ...>
```

下面我们先来看看超链接型link标签。

超链接类link标签

超链接型link标签是一种被动型链接，在用户不操作的情况下，它们不会被主动下载。

link标签具有特定的rel属性，会成为特定类型的link标签。产生超链接的link标签包括：具有 rel=“canonical” 的link、具有 rel="alternate"的link、具有 rel=“prev” rel="next"的link等等。

canonical型link

这种link的代码写法是这样：

```
<link rel="canonical" href="...">
```

这个标签提示页面它的主URL，在网站中常常有多个URL指向同一页面的情况，搜索引擎访问这类页面时会去掉重复的页面，这个link会提示搜索引擎保留哪一个URL。

alternate型link

这种link的代码写法是这样：

```
<link rel="alternate" href="...">
```

这个标签提示页面它的变形形式，这个所谓的变形可能是当前页面内容的不同格式、不同语言或者为不同的设备设计的版本，这种link通常也是提供给搜索引擎来使用的。

alternate型的link的一个典型应用场景是，页面提供rss订阅时，可以用这样的link来引入：

```
<link rel="alternate" type="application/rss+xml" title="RSS" href="...">
```

除了搜索引擎外，很多浏览器插件都能识别这样的link。

prev型link和next型link

在互联网应用中，很多网页都属于一个序列，比如分页浏览的场景，或者图片展示的场景，每个网页是序列中的一个项。

这种时候，就适合使用prev和next型的link标签，来告诉搜索引擎或者浏览器它的前一项和后一项，这有助于页面的批量展示。

因为next型link告诉浏览器“这是很可能访问的下一个页面”，HTML标准还建议对next型link做预处理，在本课后面的内容，我们会讲到预处理类的link。

其它超链接类的link

其它超链接类link标签都表示一个跟当前文档相关联的信息，可以把这样的link标签视为一种带链接功能的meta标签。

- rel=“author” 链接到本页面的作者，一般是 mailto:协议
- rel=“help” 链接到本页面的帮助页
- rel=“license” 链接到本页面的版权信息页
- rel=“search” 链接到本页面的搜索页面（一般是站内提供搜索时使用）

到这里，我们已经讲完了所有的超链接类的link标签用法了。接下来我们讲讲外部资源类link标签。

外部资源类link标签

外部资源型link标签会被主动下载，并且根据rel类型做不同的处理。外部资源型的标签包括：具有icon型的link、预处理类link、modulepreload型的link、stylesheet、pingback。下面我们来一一介绍它们。

icon型link

这类链接表示页面的icon。多数浏览器会读取icon型link，并且把页面的icon展示出来。

icon型link是唯一一个外部资源类的元信息link，其它元信息类link都是超链接，这意味着，icon型link中的图标地址默认会被浏览器下载和使用。

如果没有指定这样的link，多数浏览器会使用域名根目录下的favicon.ico，即使它并不存在，所以从性能的角度考虑，建议一定要保证页面中有icon型的link。

只有icon型link有有效的sizes属性，HTML标准允许一个页面出现多个icon型link，并且用sizes指定它适合的icon尺寸。

预处理类link

我们都知道，导航到一个网站需要经过dns查询域名、建立连接、传输数据、加载进内存和渲染等一系列的步骤。

预处理类link标签就是允许我们控制浏览器，提前针对一些资源去做这些操作，以提高性能（当然如果你乱用的话，性能反而更差）。

下面我来列一下这些link类型：

- dns-prefetch型link 提前对一个域名做dns查询，这样的link里面的href实际上只有域名有意义。
- preconnect型link 提前对一个服务器建立tcp连接。
- prefetch型link 提前取href指定的url的内容。
- preload型link 提前加载href指定的url。
- prerender型link 提前渲染href指定的url。

modulepreload型的link

modulepreload型link的作用是预先加载一个JavaScript的模块。这可以保证JS模块不必等到执行时才加载。

这里的所谓加载，是指完成下载并放入内存，并不会执行对应的JavaScript。

```
<link rel="modulepreload" href="app.js">
<link rel="modulepreload" href="helpers.js">
<link rel="modulepreload" href="irc.js">
<link rel="modulepreload" href="fog-machine.js">
<script type="module" src="app.js">
```

这个例子来自HTML标准，我们假设app.js中有 import “irc” 和 import “fog-machine”, 而 irc.js中有 import “helpers”。这段代码使用moduleload型link来预加载了四个js模块。

尽管，单独使用script标签引用app.js也可以正常工作，但是我们通过加入对四个JS文件的link标签，使得四个JS文件有机会被并行地下载，这样提高了性能。

stylesheet型link

样式表大概是所有人最熟悉的link标签用法了。它的样子是下面这样的。

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

基本用法是从一个CSS文件创建一个样式表。这里type属性可以没有，如果有，必须是"text/css"才会生效。

rel前可以加上alternate，成为rel=“alternate stylesheet”，此时必须再指定title属性。

这样可以为页面创建一份变体样式，一些浏览器，如 Firefox 3.0，支持从浏览器菜单中切换这些样式，当然了，大部分浏览器不支持这个功能，所以仅仅从语义的角度了解一下这种用法即可。

pingback型link

这样的link表示本网页被引用时，应该使用的pingback地址，这个机制是一份独立的标准，遵守pingback协议的网站在引用本页面时，会向这个pingback url发送一个消息。

以上就是link标签的所有用法了。接下来我们来介绍一下最熟悉的 a 标签，当然了，也可能你学过了本节课以后，觉得自己其实也没那么熟悉。

a 标签

a标签是“anchor”的缩写，它是锚点的意思，所谓锚点，实际上也是一种比喻的用法，古代船舶用锚来固定自己的位置，避免停泊时被海浪冲走，所以anchor标签的意思也是标识文档中的特定位置。

a标签其实同时充当了链接和目标点的角色，当a标签有href属性时，它是链接，当它有name时，它是链接的目标。

具有href的a标签跟一些link一样，会产生超链接，也就是在用户不操作的情况下，它们不会被主动下载的被动型链接。

重点的内容是，a标签也可以有rel属性，我们来简单了解一下，首先是跟link相同的一些rel，包括下面的几种。

- alternate
- author
- help
- license
- next
- prev

- search

这些跟link语义完全一致，不同的是，a标签产生的链接是会实际显示在网页中的，而link标签仅仅是元信息。

除了这些之外，a标签独有的rel类型：

- tag 表示本网页所属的标签；
- bookmark 到上级章节的链接。

a标签还有一些辅助的rel类型，用于提示浏览器或者搜索引擎做一些处理：

- nofollow 此链接不会被搜索引擎索引；
- noopener 此链接打开的网页无法使用opener来获得当前页面的窗口；
- noreferrer 此链接打开的网页无法使用referrer来获得当前页面的url；
- opener 打开的网页可以使用window.opener来访问当前页面的window对象，这是a标签的默认行为。

a标签基本解决了在页面中插入文字型和整张图片超链接的需要，但是如果我们想要在图片的某个区域产生超链接，那么就要用到另一种标签了——area标签。

area 标签

area标签与a标签非常相似，不同的是，它不是文本型的链接，而是区域型的链接。

area标签支持的rel与a完全一样，这里就不多说了。

area是整个html规则中唯一支持非矩形热区的标签，它的shape属性支持三种类型。

- 圆形：circle或者circ，coords支持三个值，分别表示中心点的x,y坐标和圆形半径r。
- 矩形：rect或者rectangle，coords支持两个值，分别表示两个对角顶点x1，y1和x2，y2。
- 多边形：poly或者polygon，coords至少包括6个值，表示多边形的各个顶点。

因为area设计的时间较早，所以不支持含有各种曲线的路径，但是它也是唯一一个支持了非矩形触发区域的元素，所以，对于一些效果而言，area是必不可少的。

area必须跟img和map标签配合使用。使用示例如下（例子来自html标准）。

```
<p>
 Please select a shape:
 <img src="shapes.png" usemap="#shapes"
      alt="Four shapes are available: a red hollow box, a green circle, a blue triangle, and a yellow four-pointed star.">
 <map name="shapes">
  <area shape=rect coords="50,50,100,100"> <!-- the hole in the red box -->
  <area shape=rect coords="25,25,125,125" href="red.html" alt="Red box.">
  <area shape=circle coords="200,75,50" href="green.html" alt="Green circle.">
  <area shape=poly coords="325,25,262,125,388,125" href="blue.html" alt="Blue triangle.">
  <area shape=poly coords="450,25,435,60,400,75,435,90,450,125,465,90,500,75,465,60"
        href="yellow.html" alt="Yellow star.">
 </map>
</p>
```

这个例子展示了在一张图片上画热区并且产生链接，分别使用了矩形、圆形和多边形三种area。

结语

本节课我们介绍了几种链接类型。在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

我们逐次讲到了link标签、a标签和area标签，link标签一般用于看不见的链接，它可能产生超链接或者外部资源链接，a和area一般用于页面上显示的链接，它们只能产生超链接。

最后，留给你一个思考问题，你的工作中，是使用过哪些类型的link标签的呢？

猜你喜欢

你好，我是winter。

在前面的课程中，我讲到了HTML的语义和元信息标签，今天这一课，我们来讲另一类HTML元素：链接。

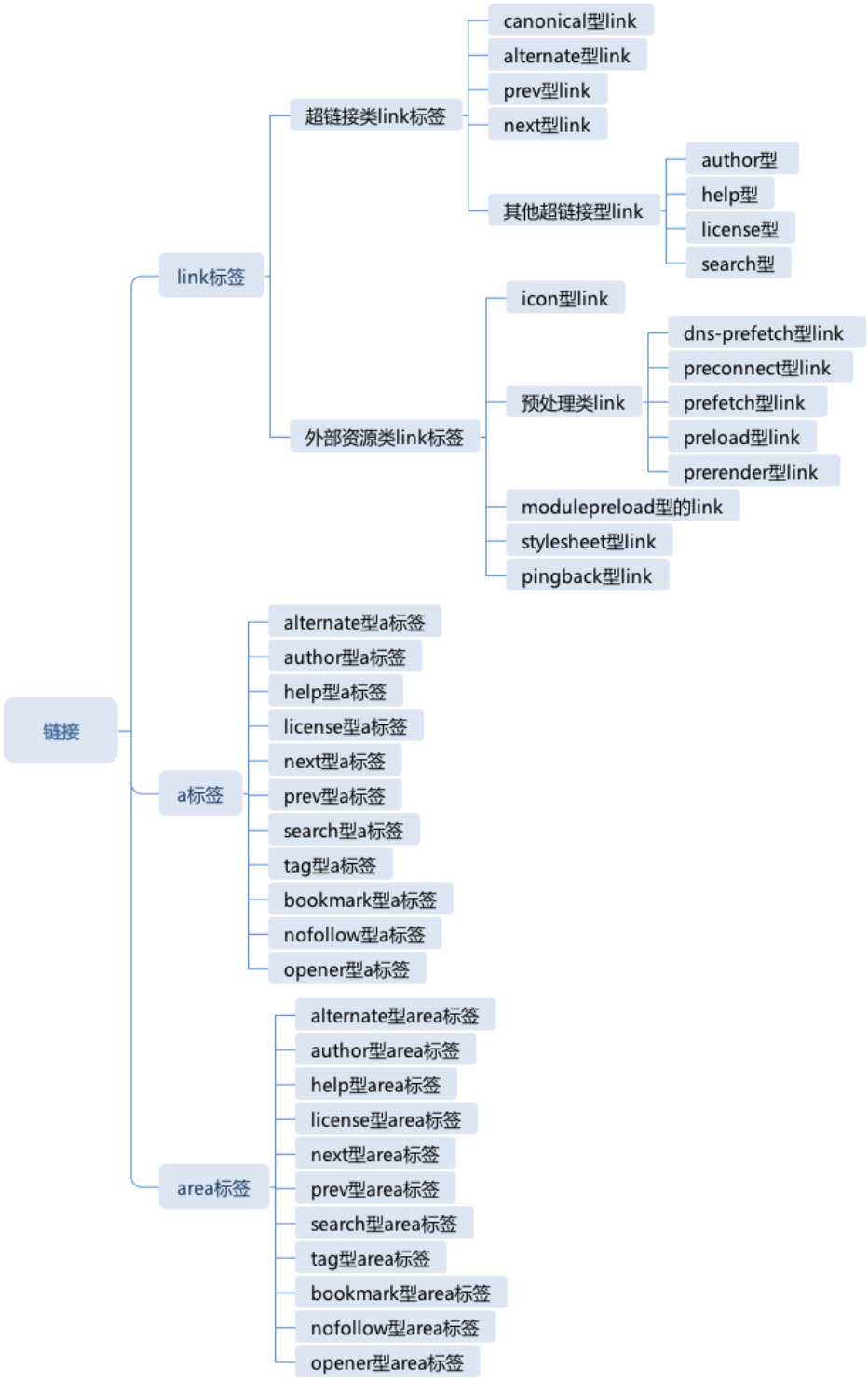
链接这种元素可以说是占据了整个互联网。也正是因为无处不在的超链接，才让我们的万维网如此繁荣。没有了超链接的HTML，最多可以称为富文本，没法称作超文本（hyper text）。

我想，作为互联网从业者，我们一定对链接都非常熟悉了。链接能够帮助我们从一个网页跳转到另一个网页。

不过，除了肉眼可见的这些链接，其实HTML里面还规定了一些不可见链接的类型，这节课，我就来给你介绍链接家族的全员，让你对它们有一个完整的认识。

链接是HTML中的一种机制，它是HTML文档和其它文档或者资源的连接关系，在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

链接的家族中有a标签、area标签和link标签。今天，我会逐一对它们进行介绍。



link 标签

提到链接，我们都知道a标签可以成为超链接，但是我们今天的内容，要从一个大家不太熟悉的标签开始，也就是link标签。

我们已经介绍过元信息类标签。实际上，我们并没有介绍完全，有些link标签也是元信息类标签的一种。

我们已经讲过，HTML标准并没有规定浏览器如何使用元信息，我们还讲到了元信息中有不少是被设计成“无需被浏览器识别，而是专门用于搜索引擎看的”。

link标签也是元信息的一种，在很多时候，它也是不会对浏览器产生任何效果的，这也是很多人会忽略link标签学习的原因。

link标签会生成一个链接，它可能生成超链接，也可能生成外部资源链接。

一些link标签会生成超链接，这些超链接又不会像a标签那样显示在网页中。这就是超链接型的link标签。

这意味着多数浏览器中，这些link标签不产生任何作用。但是，这些link标签能够被搜索引擎和一些浏览器插件识别，从而产生关键性作用。

比如，到页面RSS的link标签，能够被浏览器的RSS订阅插件识别，提示用户当前页面是可以RSS订阅的。

另外一些link标签则会把外部的资源链接到文档中，也就是说，会实际下载这些资源，并且做出一些处理，比如我们常见的用link标签引入样式表。

除了元信息的用法之外，多数外部资源型的link标签还能够被放在body中使用，从而起到把外部资源链接进文档的作用。

link标签的链接类型主要通过rel属性来区分，在本篇文章中，我们提到xx型link即表示属性rel为xx的link，其代码类似下面：

```
<link rel="xx" ...>
```

下面我们先来看看超链接型link标签。

超链接类link标签

超链接型link标签是一种被动型链接，在用户不操作的情况下，它们不会被主动下载。

link标签具有特定的rel属性，会成为特定类型的link标签。产生超链接的link标签包括：具有 rel=“canonical” 的link、具有 rel="alternate"的link、具有 rel=“prev” rel="next"的link等等。

canonical型link

这种link的代码写法是这样：

```
<link rel="canonical" href="...">
```

这个标签提示页面它的主URL，在网站中常常有多个URL指向同一页面的情况，搜索引擎访问这类页面时会去掉重复的页面，这个link会提示搜索引擎保留哪一个URL。

alternate型link

这种link的代码写法是这样：

```
<link rel="alternate" href="...">
```

这个标签提示页面它的变形形式，这个所谓的变形可能是当前页面内容的不同格式、不同语言或者为不同的设备设计的版本，这种link通常也是提供给搜索引擎来使用的。

alternate型的link的一个典型应用场景是，页面提供rss订阅时，可以用这样的link来引入：

```
<link rel="alternate" type="application/rss+xml" title="RSS" href="...">
```

除了搜索引擎外，很多浏览器插件都能识别这样的link。

prev型link和next型link

在互联网应用中，很多网页都属于一个序列，比如分页浏览的场景，或者图片展示的场景，每个网页是序列中的一个项。

这种时候，就适合使用prev和next型的link标签，来告诉搜索引擎或者浏览器它的前一项和后一项，这有助于页面的批量展示。

因为next型link告诉浏览器“这是很可能访问的下一个页面”，HTML标准还建议对next型link做预处理，在本课后面的内容，我们会讲到预处理类的link。

其它超链接类的link

其它超链接类link标签都表示一个跟当前文档相关联的信息，可以把这样的link标签视为一种带链接功能的meta标签。

- rel=“author” 链接到本页面的作者，一般是 mailto:协议
- rel=“help” 链接到本页面的帮助页
- rel=“license” 链接到本页面的版权信息页
- rel=“search” 链接到本页面的搜索页面（一般是站内提供搜索时使用）

到这里，我们已经讲完了所有的超链接类的link标签用法了。接下来我们讲讲外部资源类link标签。

外部资源类link标签

外部资源型link标签会被主动下载，并且根据rel类型做不同的处理。外部资源型的标签包括：具有icon型的link、预处理类link、modulepreload型的link、stylesheet、pingback。下面我们来一一介绍它们。

icon型link

这类链接表示页面的icon。多数浏览器会读取icon型link，并且把页面的icon展示出来。

icon型link是唯一一个外部资源类的元信息link，其它元信息类link都是超链接，这意味着，icon型link中的图标地址默认会被浏览器下载和使用。

如果没有指定这样的link，多数浏览器会使用域名根目录下的favicon.ico，即使它并不存在，所以从性能的角度考虑，建议一定要保证页面中有icon型的link。

只有icon型link有有效的sizes属性，HTML标准允许一个页面出现多个icon型link，并且用sizes指定它适合的icon尺寸。

预处理类link

我们都知道，导航到一个网站需要经过dns查询域名、建立连接、传输数据、加载进内存和渲染等一系列的步骤。

预处理类link标签就是允许我们控制浏览器，提前针对一些资源去做这些操作，以提高性能（当然如果你乱用的话，性能反而更差）。

下面我来列一下这些link类型：

- dns-prefetch型link 提前对一个域名做dns查询，这样的link里面的href实际上只有域名有意义。
- preconnect型link 提前对一个服务器建立tcp连接。
- prefetch型link 提前取href指定的url的内容。
- preload型link 提前加载href指定的url。
- prerender型link 提前渲染href指定的url。

modulepreload型的link

modulepreload型link的作用是预先加载一个JavaScript的模块。这可以保证JS模块不必等到执行时才加载。

这里的所谓加载，是指完成下载并放入内存，并不会执行对应的JavaScript。

```
<link rel="modulepreload" href="app.js">
<link rel="modulepreload" href="helpers.js">
<link rel="modulepreload" href="irc.js">
<link rel="modulepreload" href="fog-machine.js">
<script type="module" src="app.js">
```

这个例子来自HTML标准，我们假设app.js中有 import “irc” 和 import “fog-machine”, 而 irc.js中有 import “helpers”。这段代码使用moduleload型link来预加载了四个js模块。

尽管，单独使用script标签引用app.js也可以正常工作，但是我们通过加入对四个JS文件的link标签，使得四个JS文件有机会被并行地下载，这样提高了性能。

stylesheet型link

样式表大概是所有人最熟悉的link标签用法了。它的样子是下面这样的。

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

基本用法是从一个CSS文件创建一个样式表。这里type属性可以没有，如果有，必须是"text/css"才会生效。

rel前可以加上alternate，成为rel=“alternate stylesheet”，此时必须再指定title属性。

这样可以为页面创建一份变体样式，一些浏览器，如 Firefox 3.0，支持从浏览器菜单中切换这些样式，当然了，大部分浏览器不支持这个功能，所以仅仅从语义的角度了解一下这种用法即可。

pingback型link

这样的link表示本网页被引用时，应该使用的pingback地址，这个机制是一份独立的标准，遵守pingback协议的网站在引用本页面时，会向这个pingback url发送一个消息。

以上就是link标签的所有用法了。接下来我们来介绍一下最熟悉的 a 标签，当然了，也可能你学过了本节课以后，觉得自己其实也没那么熟悉。

a 标签

a标签是“anchor”的缩写，它是锚点的意思，所谓锚点，实际上也是一种比喻的用法，古代船舶用锚来固定自己的位置，避免停泊时被海浪冲走，所以anchor标签的意思也是标识文档中的特定位置。

a标签其实同时充当了链接和目标点的角色，当a标签有href属性时，它是链接，当它有name时，它是链接的目标。

具有href的a标签跟一些link一样，会产生超链接，也就是在用户不操作的情况下，它们不会被主动下载的被动型链接。

重点的内容是，a标签也可以有rel属性，我们来简单了解一下，首先是跟link相同的一些rel，包括下面的几种。

- alternate
- author
- help
- license
- next
- prev

- search

这些跟link语义完全一致，不同的是，a标签产生的链接是会实际显示在网页中的，而link标签仅仅是元信息。

除了这些之外，a标签独有的rel类型：

- tag 表示本网页所属的标签；
- bookmark 到上级章节的链接。

a标签还有一些辅助的rel类型，用于提示浏览器或者搜索引擎做一些处理：

- nofollow 此链接不会被搜索引擎索引；
- noopener 此链接打开的网页无法使用opener来获得当前页面的窗口；
- noreferrer 此链接打开的网页无法使用referrer来获得当前页面的url；
- opener 打开的网页可以使用window.opener来访问当前页面的window对象，这是a标签的默认行为。

a标签基本解决了在页面中插入文字型和整张图片超链接的需要，但是如果我们想要在图片的某个区域产生超链接，那么就要用到另一种标签了——area标签。

area 标签

area标签与a标签非常相似，不同的是，它不是文本型的链接，而是区域型的链接。

area标签支持的rel与a完全一样，这里就不多说了。

area是整个html规则中唯一支持非矩形热区的标签，它的shape属性支持三种类型。

- 圆形：circle或者circ，coords支持三个值，分别表示中心点的x,y坐标和圆形半径r。
- 矩形：rect或者rectangle，coords支持两个值，分别表示两个对角顶点x1，y1和x2，y2。
- 多边形：poly或者polygon，coords至少包括6个值，表示多边形的各个顶点。

因为area设计的时间较早，所以不支持含有各种曲线的路径，但是它也是唯一一个支持了非矩形触发区域的元素，所以，对于一些效果而言，area是必不可少的。

area必须跟img和map标签配合使用。使用示例如下（例子来自html标准）。

```
<p>
 Please select a shape:
 <img src="shapes.png" usemap="#shapes"
      alt="Four shapes are available: a red hollow box, a green circle, a blue triangle, and a yellow four-pointed star.">
 <map name="shapes">
  <area shape=rect coords="50,50,100,100"> <!-- the hole in the red box -->
  <area shape=rect coords="25,25,125,125" href="red.html" alt="Red box.">
  <area shape=circle coords="200,75,50" href="green.html" alt="Green circle.">
  <area shape=poly coords="325,25,262,125,388,125" href="blue.html" alt="Blue triangle.">
  <area shape=poly coords="450,25,435,60,400,75,435,90,450,125,465,90,500,75,465,60"
        href="yellow.html" alt="Yellow star.">
 </map>
</p>
```

这个例子展示了在一张图片上画热区并且产生链接，分别使用了矩形、圆形和多边形三种area。

结语

本节课我们介绍了几种链接类型。在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

我们逐次讲到了link标签、a标签和area标签，link标签一般用于看不见的链接，它可能产生超链接或者外部资源链接，a和area一般用于页面上显示的链接，它们只能产生超链接。

最后，留给你一个思考问题，你的工作中，是使用过哪些类型的link标签的呢？

猜你喜欢

你好，我是winter。

在前面的课程中，我讲到了HTML的语义和元信息标签，今天这一课，我们来讲另一类HTML元素：链接。

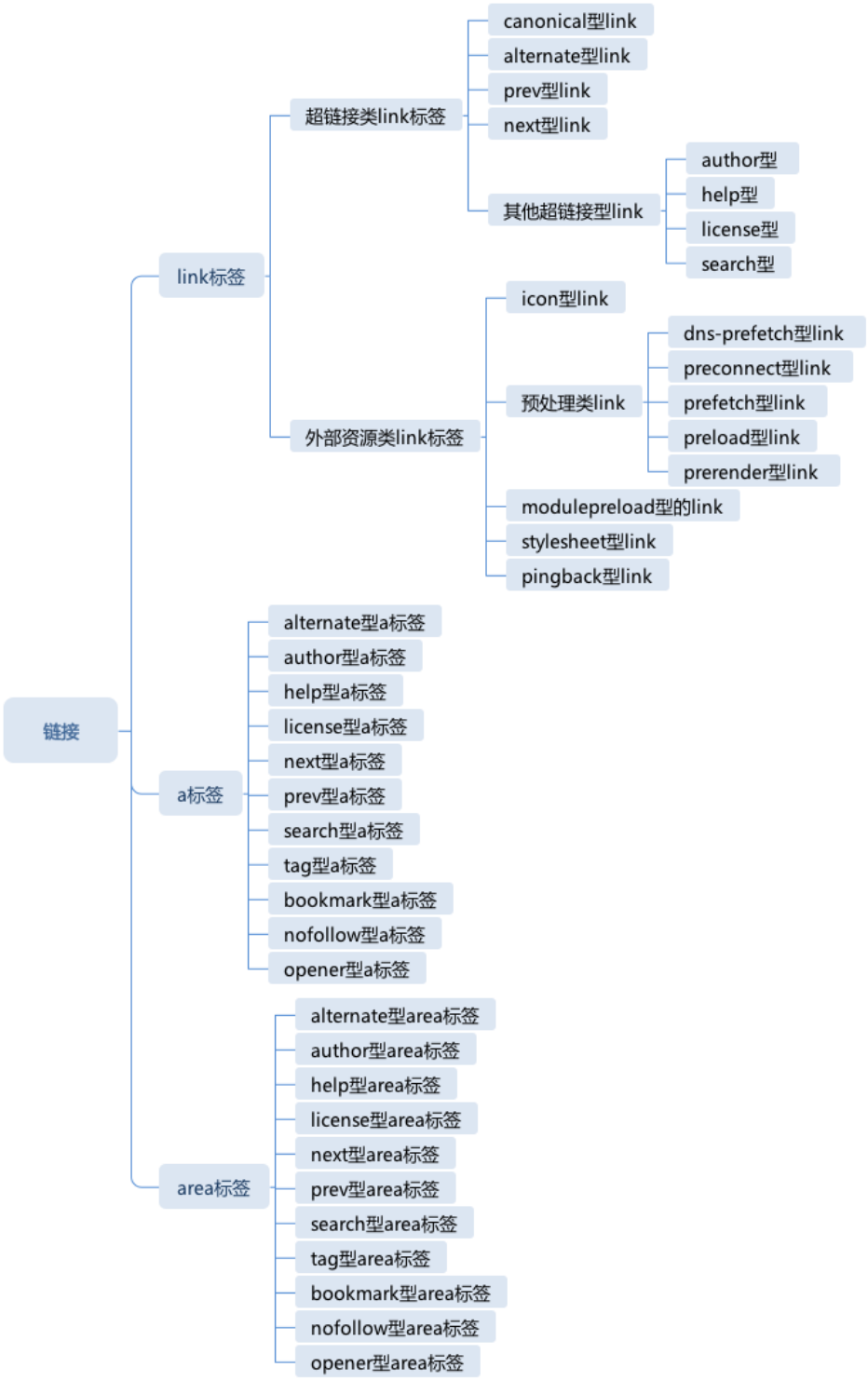
链接这种元素可以说是占据了整个互联网。也正是因为无处不在的超链接，才让我们的万维网如此繁荣。没有了超链接的HTML，最多可以称为富文本，没法称作超文本（hyper text）。

我想，作为互联网从业者，我们一定对链接都非常熟悉了。链接能够帮助我们从一个网页跳转到另一个网页。

不过，除了肉眼可见的这些链接，其实HTML里面还规定了一些不可见链接的类型，这节课，我就来给你介绍链接家族的全员，让你对它们有一个完整的认识。

链接是HTML中的一种机制，它是HTML文档和其它文档或者资源的连接关系，在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

链接的家族中有a标签、area标签和link标签。今天，我会逐一对它们进行介绍。



link 标签

提到链接，我们都知道a标签可以成为超链接，但是我们今天的内容，要从一个大家不太熟悉的标签开始，也就是link标签。

我们已经介绍过元信息类标签。实际上，我们并没有介绍完全，有些link标签也是元信息类标签的一种。

我们已经讲过，HTML标准并没有规定浏览器如何使用元信息，我们还讲到了元信息中有不少是被设计成“无需被浏览器识别，而是专门用于搜索引擎看的”。

link标签也是元信息的一种，在很多时候，它也是不会对浏览器产生任何效果的，这也是很多人会忽略link标签学习的原因。

link标签会生成一个链接，它可能生成超链接，也可能生成外部资源链接。

一些link标签会生成超链接，这些超链接又不会像a标签那样显示在网页中。这就是超链接型的link标签。

这意味着多数浏览器中，这些link标签不产生任何作用。但是，这些link标签能够被搜索引擎和一些浏览器插件识别，从而产生关键性作用。

比如，到页面RSS的link标签，能够被浏览器的RSS订阅插件识别，提示用户当前页面是可以RSS订阅的。

另外一些link标签则会把外部的资源链接到文档中，也就是说，会实际下载这些资源，并且做出一些处理，比如我们常见的用link标签引入样式表。

除了元信息的用法之外，多数外部资源型的link标签还能够被放在body中使用，从而起到把外部资源链接进文档的作用。

link标签的链接类型主要通过rel属性来区分，在本篇文章中，我们提到xx型link即表示属性rel为xx的link，其代码类似下面：

```
<link rel="xx" ...>
```

下面我们先来看看超链接型link标签。

超链接类link标签

超链接型link标签是一种被动型链接，在用户不操作的情况下，它们不会被主动下载。

link标签具有特定的rel属性，会成为特定类型的link标签。产生超链接的link标签包括：具有 rel=“canonical” 的link、具有 rel="alternate"的link、具有 rel=“prev” rel="next"的link等等。

canonical型link

这种link的代码写法是这样：

```
<link rel="canonical" href="...">
```

这个标签提示页面它的主URL，在网站中常常有多个URL指向同一页面的情况，搜索引擎访问这类页面时会去掉重复的页面，这个link会提示搜索引擎保留哪一个URL。

alternate型link

这种link的代码写法是这样：

```
<link rel="alternate" href="...">
```

这个标签提示页面它的变形形式，这个所谓的变形可能是当前页面内容的不同格式、不同语言或者为不同的设备设计的版本，这种link通常也是提供给搜索引擎来使用的。

alternate型的link的一个典型应用场景是，页面提供rss订阅时，可以用这样的link来引入：

```
<link rel="alternate" type="application/rss+xml" title="RSS" href="...">
```

除了搜索引擎外，很多浏览器插件都能识别这样的link。

prev型link和next型link

在互联网应用中，很多网页都属于一个序列，比如分页浏览的场景，或者图片展示的场景，每个网页是序列中的一个项。

这种时候，就适合使用prev和next型的link标签，来告诉搜索引擎或者浏览器它的前一项和后一项，这有助于页面的批量展示。

因为next型link告诉浏览器“这是很可能访问的下一个页面”，HTML标准还建议对next型link做预处理，在本课后面的内容，我们会讲到预处理类的link。

其它超链接类的link

其它超链接类link标签都表示一个跟当前文档相关联的信息，可以把这样的link标签视为一种带链接功能的meta标签。

- rel=“author” 链接到本页面的作者，一般是 mailto:协议
- rel=“help” 链接到本页面的帮助页
- rel=“license” 链接到本页面的版权信息页
- rel=“search” 链接到本页面的搜索页面（一般是站内提供搜索时使用）

到这里，我们已经讲完了所有的超链接类的link标签用法了。接下来我们讲讲外部资源类link标签。

外部资源类link标签

外部资源型link标签会被主动下载，并且根据rel类型做不同的处理。外部资源型的标签包括：具有icon型的link、预处理类link、modulepreload型的link、stylesheet、pingback。下面我们来一一介绍它们。

icon型link

这类链接表示页面的icon。多数浏览器会读取icon型link，并且把页面的icon展示出来。

icon型link是唯一一个外部资源类的元信息link，其它元信息类link都是超链接，这意味着，icon型link中的图标地址默认会被浏览器下载和使用。

如果没有指定这样的link，多数浏览器会使用域名根目录下的favicon.ico，即使它并不存在，所以从性能的角度考虑，建议一定要保证页面中有icon型的link。

只有icon型link有有效的sizes属性，HTML标准允许一个页面出现多个icon型link，并且用sizes指定它适合的icon尺寸。

预处理类link

我们都知道，导航到一个网站需要经过dns查询域名、建立连接、传输数据、加载进内存和渲染等一系列的步骤。

预处理类link标签就是允许我们控制浏览器，提前针对一些资源去做这些操作，以提高性能（当然如果你乱用的话，性能反而更差）。

下面我来列一下这些link类型：

- dns-prefetch型link 提前对一个域名做dns查询，这样的link里面的href实际上只有域名有意义。
- preconnect型link 提前对一个服务器建立tcp连接。
- prefetch型link 提前取href指定的url的内容。
- preload型link 提前加载href指定的url。
- prerender型link 提前渲染href指定的url。

modulepreload型的link

modulepreload型link的作用是预先加载一个JavaScript的模块。这可以保证JS模块不必等到执行时才加载。

这里的所谓加载，是指完成下载并放入内存，并不会执行对应的JavaScript。

```
<link rel="modulepreload" href="app.js">
<link rel="modulepreload" href="helpers.js">
<link rel="modulepreload" href="irc.js">
<link rel="modulepreload" href="fog-machine.js">
<script type="module" src="app.js">
```

这个例子来自HTML标准，我们假设app.js中有 import “irc” 和 import “fog-machine”, 而 irc.js中有 import “helpers”。这段代码使用moduleload型link来预加载了四个js模块。

尽管，单独使用script标签引用app.js也可以正常工作，但是我们通过加入对四个JS文件的link标签，使得四个JS文件有机会被并行地下载，这样提高了性能。

stylesheet型link

样式表大概是所有人最熟悉的link标签用法了。它的样子是下面这样的。

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

基本用法是从一个CSS文件创建一个样式表。这里type属性可以没有，如果有，必须是"text/css"才会生效。

rel前可以加上alternate，成为rel=“alternate stylesheet”，此时必须再指定title属性。

这样可以为页面创建一份变体样式，一些浏览器，如 Firefox 3.0，支持从浏览器菜单中切换这些样式，当然了，大部分浏览器不支持这个功能，所以仅仅从语义的角度了解一下这种用法即可。

pingback型link

这样的link表示本网页被引用时，应该使用的pingback地址，这个机制是一份独立的标准，遵守pingback协议的网站在引用本页面时，会向这个pingback url发送一个消息。

以上就是link标签的所有用法了。接下来我们来介绍一下最熟悉的 a 标签，当然了，也可能你学过了本节课以后，觉得自己其实也没那么熟悉。

a 标签

a标签是“anchor”的缩写，它是锚点的意思，所谓锚点，实际上也是一种比喻的用法，古代船舶用锚来固定自己的位置，避免停泊时被海浪冲走，所以anchor标签的意思也是标识文档中的特定位置。

a标签其实同时充当了链接和目标点的角色，当a标签有href属性时，它是链接，当它有name时，它是链接的目标。

具有href的a标签跟一些link一样，会产生超链接，也就是在用户不操作的情况下，它们不会被主动下载的被动型链接。

重点的内容是，a标签也可以有rel属性，我们来简单了解一下，首先是跟link相同的一些rel，包括下面的几种。

- alternate
- author
- help
- license
- next
- prev

- search

这些跟link语义完全一致，不同的是，a标签产生的链接是会实际显示在网页中的，而link标签仅仅是元信息。

除了这些之外，a标签独有的rel类型：

- tag 表示本网页所属的标签；
- bookmark 到上级章节的链接。

a标签还有一些辅助的rel类型，用于提示浏览器或者搜索引擎做一些处理：

- nofollow 此链接不会被搜索引擎索引；
- noopener 此链接打开的网页无法使用opener来获得当前页面的窗口；
- noreferrer 此链接打开的网页无法使用referrer来获得当前页面的url；
- opener 打开的网页可以使用window.opener来访问当前页面的window对象，这是a标签的默认行为。

a标签基本解决了在页面中插入文字型和整张图片超链接的需要，但是如果我们想要在图片的某个区域产生超链接，那么就要用到另一种标签了——area标签。

area 标签

area标签与a标签非常相似，不同的是，它不是文本型的链接，而是区域型的链接。

area标签支持的rel与a完全一样，这里就不多说了。

area是整个html规则中唯一支持非矩形热区的标签，它的shape属性支持三种类型。

- 圆形：circle或者circ，coords支持三个值，分别表示中心点的x,y坐标和圆形半径r。
- 矩形：rect或者rectangle，coords支持两个值，分别表示两个对角顶点x1，y1和x2，y2。
- 多边形：poly或者polygon，coords至少包括6个值，表示多边形的各个顶点。

因为area设计的时间较早，所以不支持含有各种曲线的路径，但是它也是唯一一个支持了非矩形触发区域的元素，所以，对于一些效果而言，area是必不可少的。

area必须跟img和map标签配合使用。使用示例如下（例子来自html标准）。

```
<p>
 Please select a shape:
 <img src="shapes.png" usemap="#shapes"
      alt="Four shapes are available: a red hollow box, a green circle, a blue triangle, and a yellow four-pointed star.">
 <map name="shapes">
  <area shape=rect coords="50,50,100,100"> <!-- the hole in the red box -->
  <area shape=rect coords="25,25,125,125" href="red.html" alt="Red box.">
  <area shape=circle coords="200,75,50" href="green.html" alt="Green circle.">
  <area shape=poly coords="325,25,262,125,388,125" href="blue.html" alt="Blue triangle.">
  <area shape=poly coords="450,25,435,60,400,75,435,90,450,125,465,90,500,75,465,60"
        href="yellow.html" alt="Yellow star.">
 </map>
</p>
```

这个例子展示了在一张图片上画热区并且产生链接，分别使用了矩形、圆形和多边形三种area。

结语

本节课我们介绍了几种链接类型。在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

我们逐次讲到了link标签、a标签和area标签，link标签一般用于看不见的链接，它可能产生超链接或者外部资源链接，a和area一般用于页面上显示的链接，它们只能产生超链接。

最后，留给你一个思考问题，你的工作中，是使用过哪些类型的link标签的呢？

猜你喜欢

你好，我是winter。

在前面的课程中，我讲到了HTML的语义和元信息标签，今天这一课，我们来讲另一类HTML元素：链接。

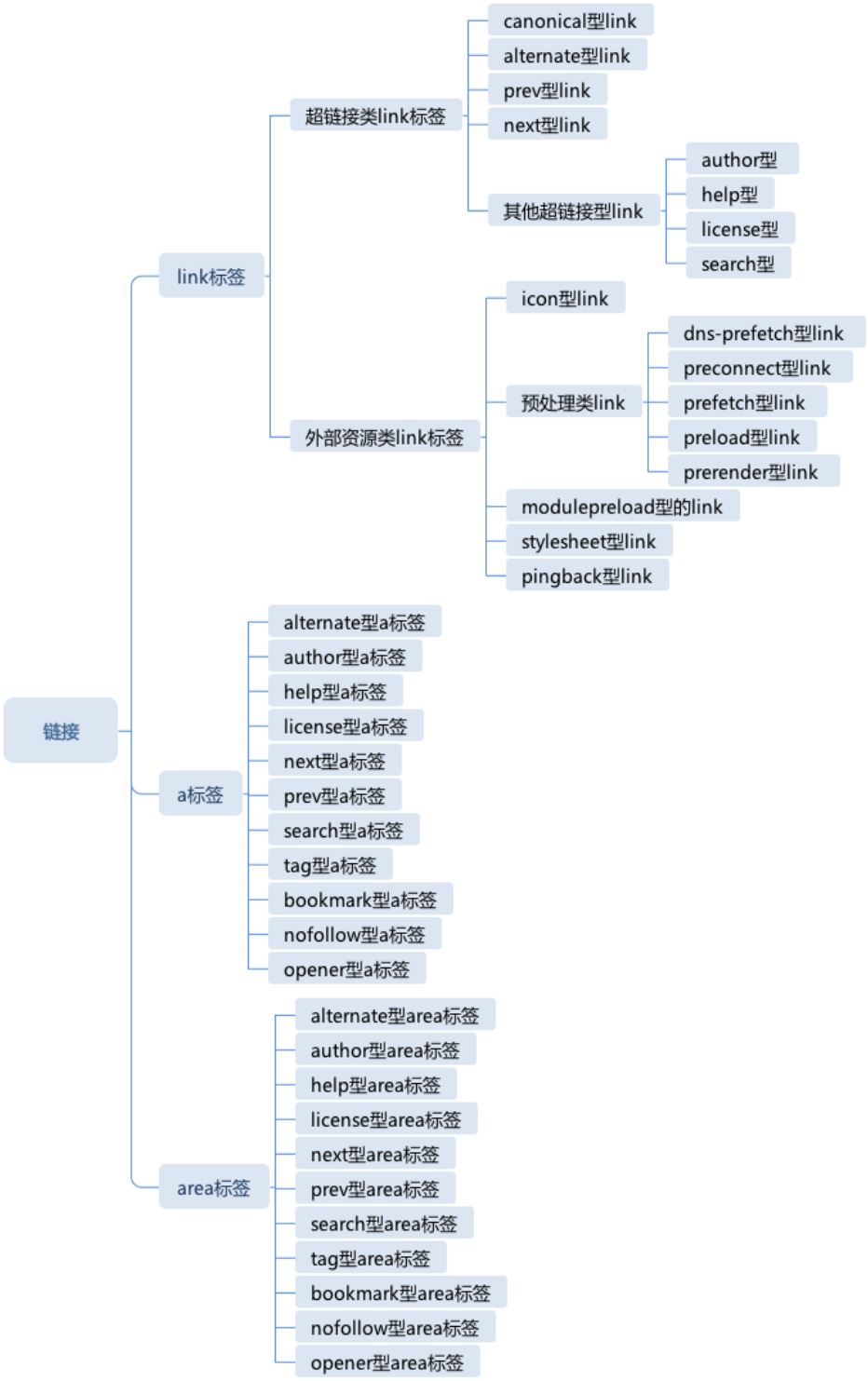
链接这种元素可以说是占据了整个互联网。也正是因为无处不在的超链接，才让我们的万维网如此繁荣。没有了超链接的HTML，最多可以称为富文本，没法称作超文本（hyper text）。

我想，作为互联网从业者，我们一定对链接都非常熟悉了。链接能够帮助我们从一个网页跳转到另一个网页。

不过，除了肉眼可见的这些链接，其实HTML里面还规定了一些不可见链接的类型，这节课，我就来给你介绍链接家族的全员，让你对它们有一个完整的认识。

链接是HTML中的一种机制，它是HTML文档和其它文档或者资源的连接关系，在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

链接的家族中有a标签、area标签和link标签。今天，我会逐一对它们进行介绍。



link 标签

提到链接，我们都知道a标签可以成为超链接，但是我们今天的内容，要从一个大家不太熟悉的标签开始，也就是link标签。

我们已经介绍过元信息类标签。实际上，我们并没有介绍完全，有些link标签也是元信息类标签的一种。

我们已经讲过，HTML标准并没有规定浏览器如何使用元信息，我们还讲到了元信息中有不少是被设计成“无需被浏览器识别，而是专门用于搜索引擎看的”。

link标签也是元信息的一种，在很多时候，它也是不会对浏览器产生任何效果的，这也是很多人会忽略link标签学习的原因。

link标签会生成一个链接，它可能生成超链接，也可能生成外部资源链接。

一些link标签会生成超链接，这些超链接又不会像a标签那样显示在网页中。这就是超链接型的link标签。

这意味着多数浏览器中，这些link标签不产生任何作用。但是，这些link标签能够被搜索引擎和一些浏览器插件识别，从而产生关键性作用。

比如，到页面RSS的link标签，能够被浏览器的RSS订阅插件识别，提示用户当前页面是可以RSS订阅的。

另外一些link标签则会把外部的资源链接到文档中，也就是说，会实际下载这些资源，并且做出一些处理，比如我们常见的用link标签引入样式表。

除了元信息的用法之外，多数外部资源型的link标签还能够被放在body中使用，从而起到把外部资源链接进文档的作用。

link标签的链接类型主要通过rel属性来区分，在本篇文章中，我们提到xx型link即表示属性rel为xx的link，其代码类似下面：

```
<link rel="xx" ...>
```

下面我们先来看看超链接型link标签。

超链接类link标签

超链接型link标签是一种被动型链接，在用户不操作的情况下，它们不会被主动下载。

link标签具有特定的rel属性，会成为特定类型的link标签。产生超链接的link标签包括：具有 rel=“canonical” 的link、具有 rel="alternate"的link、具有 rel=“prev” rel="next"的link等等。

canonical型link

这种link的代码写法是这样：

```
<link rel="canonical" href="...">
```

这个标签提示页面它的主URL，在网站中常常有多个URL指向同一页面的情况，搜索引擎访问这类页面时会去掉重复的页面，这个link会提示搜索引擎保留哪一个URL。

alternate型link

这种link的代码写法是这样：

```
<link rel="alternate" href="...">
```

这个标签提示页面它的变形形式，这个所谓的变形可能是当前页面内容的不同格式、不同语言或者为不同的设备设计的版本，这种link通常也是提供给搜索引擎来使用的。

alternate型的link的一个典型应用场景是，页面提供rss订阅时，可以用这样的link来引入：

```
<link rel="alternate" type="application/rss+xml" title="RSS" href="...">
```

除了搜索引擎外，很多浏览器插件都能识别这样的link。

prev型link和next型link

在互联网应用中，很多网页都属于一个序列，比如分页浏览的场景，或者图片展示的场景，每个网页是序列中的一个项。

这种时候，就适合使用prev和next型的link标签，来告诉搜索引擎或者浏览器它的前一项和后一项，这有助于页面的批量展示。

因为next型link告诉浏览器“这是很可能访问的下一个页面”，HTML标准还建议对next型link做预处理，在本课后面的内容，我们会讲到预处理类的link。

其它超链接类的link

其它超链接类link标签都表示一个跟当前文档相关联的信息，可以把这样的link标签视为一种带链接功能的meta标签。

- rel=“author” 链接到本页面的作者，一般是 mailto:协议
- rel=“help” 链接到本页面的帮助页
- rel=“license” 链接到本页面的版权信息页
- rel=“search” 链接到本页面的搜索页面（一般是站内提供搜索时使用）

到这里，我们已经讲完了所有的超链接类的link标签用法了。接下来我们讲讲外部资源类link标签。

外部资源类link标签

外部资源型link标签会被主动下载，并且根据rel类型做不同的处理。外部资源型的标签包括：具有icon型的link、预处理类link、modulepreload型的link、stylesheet、pingback。下面我们来一一介绍它们。

icon型link

这类链接表示页面的icon。多数浏览器会读取icon型link，并且把页面的icon展示出来。

icon型link是唯一一个外部资源类的元信息link，其它元信息类link都是超链接，这意味着，icon型link中的图标地址默认会被浏览器下载和使用。

如果没有指定这样的link，多数浏览器会使用域名根目录下的favicon.ico，即使它并不存在，所以从性能的角度考虑，建议一定要保证页面中有icon型的link。

只有icon型link有有效的sizes属性，HTML标准允许一个页面出现多个icon型link，并且用sizes指定它适合的icon尺寸。

预处理类link

我们都知道，导航到一个网站需要经过dns查询域名、建立连接、传输数据、加载进内存和渲染等一系列的步骤。

预处理类link标签就是允许我们控制浏览器，提前针对一些资源去做这些操作，以提高性能（当然如果你乱用的话，性能反而更差）。

下面我来列一下这些link类型：

- dns-prefetch型link 提前对一个域名做dns查询，这样的link里面的href实际上只有域名有意义。
- preconnect型link 提前对一个服务器建立tcp连接。
- prefetch型link 提前取href指定的url的内容。
- preload型link 提前加载href指定的url。
- prerender型link 提前渲染href指定的url。

modulepreload型的link

modulepreload型link的作用是预先加载一个JavaScript的模块。这可以保证JS模块不必等到执行时才加载。

这里的所谓加载，是指完成下载并放入内存，并不会执行对应的JavaScript。

```
<link rel="modulepreload" href="app.js">
<link rel="modulepreload" href="helpers.js">
<link rel="modulepreload" href="irc.js">
<link rel="modulepreload" href="fog-machine.js">
<script type="module" src="app.js">
```

这个例子来自HTML标准，我们假设app.js中有 import “irc” 和 import “fog-machine”, 而 irc.js中有 import “helpers”。这段代码使用moduleload型link来预加载了四个js模块。

尽管，单独使用script标签引用app.js也可以正常工作，但是我们通过加入对四个JS文件的link标签，使得四个JS文件有机会被并行地下载，这样提高了性能。

stylesheet型link

样式表大概是所有人最熟悉的link标签用法了。它的样子是下面这样的。

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

基本用法是从一个CSS文件创建一个样式表。这里type属性可以没有，如果有，必须是"text/css"才会生效。

rel前可以加上alternate，成为rel=“alternate stylesheet”，此时必须再指定title属性。

这样可以为页面创建一份变体样式，一些浏览器，如 Firefox 3.0，支持从浏览器菜单中切换这些样式，当然了，大部分浏览器不支持这个功能，所以仅仅从语义的角度了解一下这种用法即可。

pingback型link

这样的link表示本网页被引用时，应该使用的pingback地址，这个机制是一份独立的标准，遵守pingback协议的网站在引用本页面时，会向这个pingback url发送一个消息。

以上就是link标签的所有用法了。接下来我们来介绍一下最熟悉的 a 标签，当然了，也可能你学过了本节课以后，觉得自己其实也没那么熟悉。

a 标签

a标签是“anchor”的缩写，它是锚点的意思，所谓锚点，实际上也是一种比喻的用法，古代船舶用锚来固定自己的位置，避免停泊时被海浪冲走，所以anchor标签的意思也是标识文档中的特定位置。

a标签其实同时充当了链接和目标点的角色，当a标签有href属性时，它是链接，当它有name时，它是链接的目标。

具有href的a标签跟一些link一样，会产生超链接，也就是在用户不操作的情况下，它们不会被主动下载的被动型链接。

重点的内容是，a标签也可以有rel属性，我们来简单了解一下，首先是跟link相同的一些rel，包括下面的几种。

- alternate
- author
- help
- license
- next
- prev

- search

这些跟link语义完全一致，不同的是，a标签产生的链接是会实际显示在网页中的，而link标签仅仅是元信息。

除了这些之外，a标签独有的rel类型：

- tag 表示本网页所属的标签；
- bookmark 到上级章节的链接。

a标签还有一些辅助的rel类型，用于提示浏览器或者搜索引擎做一些处理：

- nofollow 此链接不会被搜索引擎索引；
- noopener 此链接打开的网页无法使用opener来获得当前页面的窗口；
- noreferrer 此链接打开的网页无法使用referrer来获得当前页面的url；
- opener 打开的网页可以使用window.opener来访问当前页面的window对象，这是a标签的默认行为。

a标签基本解决了在页面中插入文字型和整张图片超链接的需要，但是如果我们想要在图片的某个区域产生超链接，那么就要用到另一种标签了——area标签。

area 标签

area标签与a标签非常相似，不同的是，它不是文本型的链接，而是区域型的链接。

area标签支持的rel与a完全一样，这里就不多说了。

area是整个html规则中唯一支持非矩形热区的标签，它的shape属性支持三种类型。

- 圆形：circle或者circ，coords支持三个值，分别表示中心点的x,y坐标和圆形半径r。
- 矩形：rect或者rectangle，coords支持两个值，分别表示两个对角顶点x1，y1和x2，y2。
- 多边形：poly或者polygon，coords至少包括6个值，表示多边形的各个顶点。

因为area设计的时间较早，所以不支持含有各种曲线的路径，但是它也是唯一一个支持了非矩形触发区域的元素，所以，对于一些效果而言，area是必不可少的。

area必须跟img和map标签配合使用。使用示例如下（例子来自html标准）。

```
<p>
 Please select a shape:
 <img src="shapes.png" usemap="#shapes"
      alt="Four shapes are available: a red hollow box, a green circle, a blue triangle, and a yellow four-pointed star.">
 <map name="shapes">
  <area shape=rect coords="50,50,100,100"> <!-- the hole in the red box -->
  <area shape=rect coords="25,25,125,125" href="red.html" alt="Red box.">
  <area shape=circle coords="200,75,50" href="green.html" alt="Green circle.">
  <area shape=poly coords="325,25,262,125,388,125" href="blue.html" alt="Blue triangle.">
  <area shape=poly coords="450,25,435,60,400,75,435,90,450,125,465,90,500,75,465,60"
        href="yellow.html" alt="Yellow star.">
 </map>
</p>
```

这个例子展示了在一张图片上画热区并且产生链接，分别使用了矩形、圆形和多边形三种area。

结语

本节课我们介绍了几种链接类型。在HTML中，链接有两种类型。一种是超链接型标签，一种是外部资源链接。

我们逐次讲到了link标签、a标签和area标签，link标签一般用于看不见的链接，它可能产生超链接或者外部资源链接，a和area一般用于页面上显示的链接，它们只能产生超链接。

最后，留给你一个思考问题，你的工作中，是使用过哪些类型的link标签的呢？

猜你喜欢