34 | HTML小实验：用代码分析HTML标准

2019-04-11 winter

重学前端 进入课程 >

讲述：winter
时长 07:51 大小 7.20M

你好，我是 winter。

前面的课程中，我们已经讲解了大部分的 HTML 标签。

然而，为了突出重点，我们还是会忽略一些标签类型。比如表单类标签和表格类标签，我认为只有少数前端工程师用过，比如我在整个手机淘宝的工作生涯中，一次表格类标签都没有用到，表单类则只用过 input，也只有几次。

那么，剩下的标签我们怎么样去了解它们呢？当然是查阅 HTML 标准。

由于阅读标准有一定门槛，需要了解一些机制，这节课，我为你设计了一个小实验，用 JavaScript 代码去抽取标准中我们需要的信息。

HTML 标准

我们采用 WHATWG 的 living standard 标准，我们先来看看标准是如何描述一个标签的，这里我们看到，有下面这些内容。

复制代码

```
Categories:
    Flow content.
    Phrasing content.
    Embedded content.
    If the element has a controls attribute: Interactive content.
    Palpable content.
Contexts in which this element can be used:
    Where embedded content is expected.
Content model:
    If the element has a src attribute: zero or more track elements, then transparent, l
    If the element does not have a src attribute: zero or more source elements, then ze
Tag omission in text/html:
    Neither tag is omissible.
Content attributes:
    Global attributes
    src — Address of the resource
    crossorigin — How the element handles crossorigin requests
    poster — Poster frame to show prior to video playback
    preload — Hints how much buffering the media resource will likely need
    autoplay — Hint that the media resource can be started automatically when the page i
    playsinline — Encourage the user agent to display video content within the element's
    loop — Whether to loop the media resource
    muted — Whether to mute the media resource by default
    controls — Show user agent controls
    width — Horizontal dimension
    height — Vertical dimension
DOM interface:
    [Exposed=Window, HTMLConstructor]
    interface HTMLVideoElement : HTMLMediaElement {
      [CEReactions] attribute unsigned long width;
      [CEReactions] attribute unsigned long height;
      readonly attribute unsigned long videoWidth;
      readonly attribute unsigned long videoHeight;
      [CEReactions] attribute USVString poster;
      [CEReactions] attribute boolean playsInline;
    };
```

我们看到，这里的描述分为 6 个部分，有下面这些内容。

Categories：标签所属的分类。

Contexts in which this element can be used：标签能够用在哪里。

Content model：标签的内容模型。

Tag omission in text/html：标签是否可以省略。

Content attributes：内容属性。

DOM interface：用 WebIDL 定义的元素类型接口。

这一节课，我们关注一下 Categories、Contexts in which this element can be used、Content model 这几个部分。我会带你从标准中抓取数据，做一个小工具，用来检查 X 标签是否能放入 Y 标签内。

代码角度分析 HTML 标准

HTML 标准描述用词非常的严谨，这给我们抓取数据带来了巨大的方便，首先，我们打开单页面版 HTML 标准 https://html.spec.whatwg.org/

在这个页面上，我们执行一下以下代码：

复制代码

```
Array.prototype.map.call(document.querySelectorAll(".element"), e=>e.innerText);
```

这样我们就得到了所有元素的定义了，现在有 107 个元素。

不过，比较尴尬的是，这些文本中并不包含元素名，我们只好从 id 属性中获取，最后代码类似这样：

复制代码

```
var elementDefinations = Array.prototype.map.call(document.querySelectorAll(".element")
  text:e.innerText,
  name:e.childNodes[0].childNodes[0].id.match(/the\-([\s\S]+)\-element:/)?RegExp.$1:nul
```

接下来我们用代码理解一下这些文本。首先我们来分析一下这些文本，它分成了 6 个部分，而且顺序非常固定，这样，我们可以用 JavaScript 的正则表达式匹配来拆分六个字段。

我们这个小实验的目标是计算元素之间的包含关系，因此，我们先关心一下 categories 和 contentModel 两个字段。

复制代码

```
for(let defination of elementDefinations) {

  console.log(defination.name + ":")
  let categories = defination.text.match(/Categories:\n([\s\S]+)\nContexts in which thi
  for(let category of categories) {
      console.log(category);
  }


/*
  let contentModel = defination.text.match(/Content model:\n([\s\S]+)\nTag omission in
  for(let line of contentModel)
    console.log(line);
*/
}
```

接下来我们来处理 category。

首先 category 的写法中，最基本的就是直接描述了 category 的句子，我们把这些不带任何条件的 category 先保存起来，然后打印出来其它的描述看看：

复制代码

```
for(let defination of elementDefinations) {

  //console.log(defination.name + ":")
  let categories = defination.text.match(/Categories:\n([\s\S]+)\nContexts in which thi
  defination.categories = [];
  for(let category of categories) {
    if(category.match(/^([^ ]+) content./))
      defination.categories.push(RegExp.$1);
    else
      console.log(category)
  }

```

```

/*
  let contentModel = defination.text.match(/Content model:\n([\s\S]+)\nTag omission in 
  for(let line of contentModel)
    console.log(line);
*/
}
```

这里我们要处理的第一个逻辑是带 if 的情况。

然后我们来看看剩下的情况：

复制代码

```
None.
Sectioning root.
None.
Sectioning root.
None.
Form-associated element.
Listed and submittable form-associated element.
None.
Sectioning root.
None.
If the type attribute is not in the Hidden state: Listed, labelable, submittable, rese
If the type attribute is in the Hidden state: Listed, submittable, resettable, and aut
Listed, labelable, submittable, and autocapitalize-inheriting form-associated element.
Listed, labelable, submittable, resettable, and autocapitalize-inheriting form-associa
None.
Listed, labelable, submittable, resettable, and autocapitalize-inheriting form-associa
Listed, labelable, resettable, and autocapitalize-inheriting form-associated element.
Labelable element.
Sectioning root.
Listed and autocapitalize-inheriting form-associated element.
None.
Sectioning root.
None.
Sectioning root.
Script-supporting element.
```

这里出现了几个概念：

None

Sectioning root

Form-associated element

Labelable element

Script-supporting element

如果我们要真正完美地实现元素分类，就必须要在代码中加入正则表达式来解析这些规则，这里作为今天的课后问题，留给你自己完成。

接下来我们看看 Content Model，我们照例先处理掉最简单点的部分，就是带分类的内容模型：

复制代码

```

for(let defination of elementDefinations) {

  //console.log(defination.name + ":")
  let categories = defination.text.match(/Categories:\n([\s\S]+)\nContexts in which thi
  defination.contentModel = [];
  let contentModel = defination.text.match(/Content model:\n([\s\S]+)\nTag omission in
  for(let line of contentModel)
    if(line.match(/^([^ ]+) content./))
      defination.contentModel.push(RegExp.$1);
    else
      console.log(line)
}

```

好了，我们照例看看剩下了什么：

复制代码

```
A head element followed by a body element.
If the document is an iframe srcdoc document or if title information is available from
Otherwise: One or more elements of metadata content, of which exactly one is a title e
Text that is not inter-element whitespace.
Nothing.
Text that gives a conformant style sheet.
One or more h1, h2, h3, h4, h5, h6 elements, optionally intermixed with script-support
Nothing.
Zero or more li and script-supporting elements.
Either: Zero or more groups each consisting of one or more dt elements followed by one
```

Or: One or more div elements, optionally intermixed with script-supporting elements.

Either: one figcaption element followed by flow content.

Or: flow content followed by one figcaption element.

Or: flow content.

If the element is a child of a dl element: one or more dt elements followed by one or

If the element is not a child of a dl element: flow content.

Transparent, but there must be no interactive content or a element descendants.

See prose.

Text.

If the element has a datetime attribute: Phrasing content.

Otherwise: Text, but must match requirements described in prose below.

Nothing.

Transparent.

Zero or more source elements, followed by one img element, optionally intermixed with

Nothing.

Zero or more param elements, then, transparent.

Nothing.

If the element has a src attribute: zero or more track elements, then transparent, but

If the element does not have a src attribute: zero or more source elements, then zero

If the element has a src attribute: zero or more track elements, then transparent, but

If the element does not have a src attribute: zero or more source elements, then zero

Nothing.

Transparent.

Nothing.

In this order: optionally a caption element, followed by zero or more colgroup element

If the span attribute is present: Nothing.

If the span attribute is absent: Zero or more col and template elements.

Nothing.

Zero or more tr and script-supporting elements.

Zero or more td, th, and script-supporting elements.

Nothing.

Zero or more option, optgroup, and script-supporting elements.

Either: phrasing content.

Or: Zero or more option and script-supporting elements.

Zero or more option and script-supporting elements.

If the element has a label attribute and a value attribute: Nothing.

If the element has a label attribute but no value attribute: Text.

If the element has no label attribute and is not a child of a datalist element: Text t

If the element has no label attribute and is a child of a datalist element: Text.

Text.

Optionally a legend element, followed by flow content.

One summary element followed by flow content.

Either: phrasing content.

Or: one element of heading content.

If there is no src attribute, depends on the value of the type attribute, but must mat

If there is a src attribute, the element must be either empty or contain only script d

When scripting is disabled, in a head element: in any order, zero or more link element

When scripting is disabled, not in a head element: transparent, but there must be no n

Otherwise: text that conforms to the requirements given in the prose.

Nothing (for clarification, see example).

Transparent

Transparent, but with no interactive content descendants except for a elements, img el

这有点复杂，我们还是把它做一些分类，首先我们过滤掉带 If 的情况、Text 和 Transparent。

复制代码

```
for(let defination of elementDefinations) {
  //console.log(defination.name + ":")
  let categories = defination.text.match(/Categories:\n([\s\S]+)\nContexts in which thi
  defination.contentModel = [];
  let contentModel = defination.text.match(/Content model:\n([\s\S]+)\nTag omission in
  for(let line of contentModel)
    if(line.match(/([^ ]+) content./))
      defination.contentModel.push(RegExp.$1);
    else if(line.match(/Nothing.|Transparent./));
    else if(line.match(/^Text[\s\S]*.$/));
    else
      console.log(line)
}
```

这时候我们再来执行看看：

复制代码

```
A head element followed by a body element.
One or more h1, h2, h3, h4, h5, h6 elements, optionally intermixed with script-supporti
Zero or more li and script-supporting elements.
Either: Zero or more groups each consisting of one or more dt elements followed by one
Or: One or more div elements, optionally intermixed with script-supporting elements.
If the element is a child of a dl element: one or more dt elements followed by one or m
See prose.
Otherwise: Text, but must match requirements described in prose below.
Zero or more source elements, followed by one img element, optionally intermixed with s
Zero or more param elements, then, transparent.
If the element has a src attribute: zero or more track elements, then transparent, but
If the element does not have a src attribute: zero or more source elements, then zero o
If the element has a src attribute: zero or more track elements, then transparent, but
If the element does not have a src attribute: zero or more source elements, then zero o
In this order: optionally a caption element, followed by zero or more colgroup elements
If the span attribute is absent: Zero or more col and template elements.
Zero or more tr and script-supporting elements.
Zero or more td, th, and script-supporting elements.
Zero or more option, optgroup, and script-supporting elements.
Or: Zero or more option and script-supporting elements.
Zero or more option and script-supporting elements.
```

```
If the element has a label attribute but no value attribute: Text.
If the element has no label attribute and is not a child of a datalist element: Text thi
If the element has no label attribute and is a child of a datalist element: Text.
When scripting is disabled, in a head element: in any order, zero or more link elements
When scripting is disabled, not in a head element: transparent, but there must be no no:
Otherwise: text that conforms to the requirements given in the prose.
```

这下剩余的就少多了，我们可以看到，基本上剩下的都是直接描述可用的元素了，如果你愿意，还可以用代码进一步解析，不过如果是我的话，会选择手工把它们写成 JSON 了，毕竟只有三十多行文本。

好了，有了 contentModel 和 category，我们要检查某一元素是否可以作为另一元素的子元素，就可以判断一下两边是否匹配啦，首先，我们要做个索引：

复制代码

```
var dictionary = Object.create(null);

for(let defination of elementDefinations) {
  dictionary[defination.name] = defination;
}

```

然后我们编写一下我们的 check 函数：

复制代码

```
function check(parent, child) {
  for(let category of child.categories)
    if(parent.contentModel.categories.conatains(category))
      return true;
  if(parent.contentModel.names.conatains(child.name))
      return true;
  return false;
}

```

总结

这一节课，我们完成了一个小实验：利用工具分析 Web 标准文本，来获得元素的信息。

通过这个实验，我希望能够传递一种思路，代码能够帮助我们从 Web 标准中挖掘出来很多想要的信息，编写代码的过程，也是更深入理解标准的契机。

我们前面的课程中把元素分成了几类来讲解，但是这些分类只能大概地覆盖所有的标签，我设置课程的目标也是讲解标签背后的知识，而非每一种标签的细节。具体每一种标签的属性和细节，可以留给大家自己去整理。

这一节课的产出，则是“绝对完整的标签列表”，也是我学习和阅读标准的小技巧，通过代码我们可以从不同的侧面分析标准的内容，挖掘需要注意的点，这是一种非常好的学习方法。

极客时间
重学前端
每天10分钟，重构你的前端知识体系
winter 程劭非
前手机淘宝前端负责人
新版升级：点击「请朋友读」，10位好友免费读，邀请订阅更有现金奖励。

上一篇 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？

下一篇 35 | CSS Flex排版：为什么垂直居中这么难？

精选留言 (7)

写留言

阿成
2019-04-14

3

这种“通过简单的文本分析，快速提炼出自己感兴趣的部分”的方法是非常值得借鉴的，我平时也会用这种方法去网页中做一些快速的统计和信息筛选。

不过，通过这样的文本分析去完成一个“检查一个元素是否能够放置在另一个元素内部”的小程序还是有点“把问题复杂化”的感觉（尽管这个过程中也可以锻炼一些能...

展开

一步
2019-04-29

1

老师 有个疑问： WHATWG 和 W3C 标准以哪个为准，这两个标准有什么区别？是不是相互不认可的

柠檬树
2019-06-02

没太看懂，好多语法基于这个页面https://html.spec.whatwg.org/

展开

---

away
2019-04-30

@一步 WHATWG 和 W3C 标准若有不同，一般以 WHATWG 为准

展开

嗨海海
2019-04-12

学不到，有因果关系，工作实际需要吗？

展开

被雨水过滤...
2019-04-11

学习了

展开

会飞的大猫

2019-04-11

Winter，刚看完文章，就在淘宝技术节视频看到了你持相机和大家自拍的图片

展开