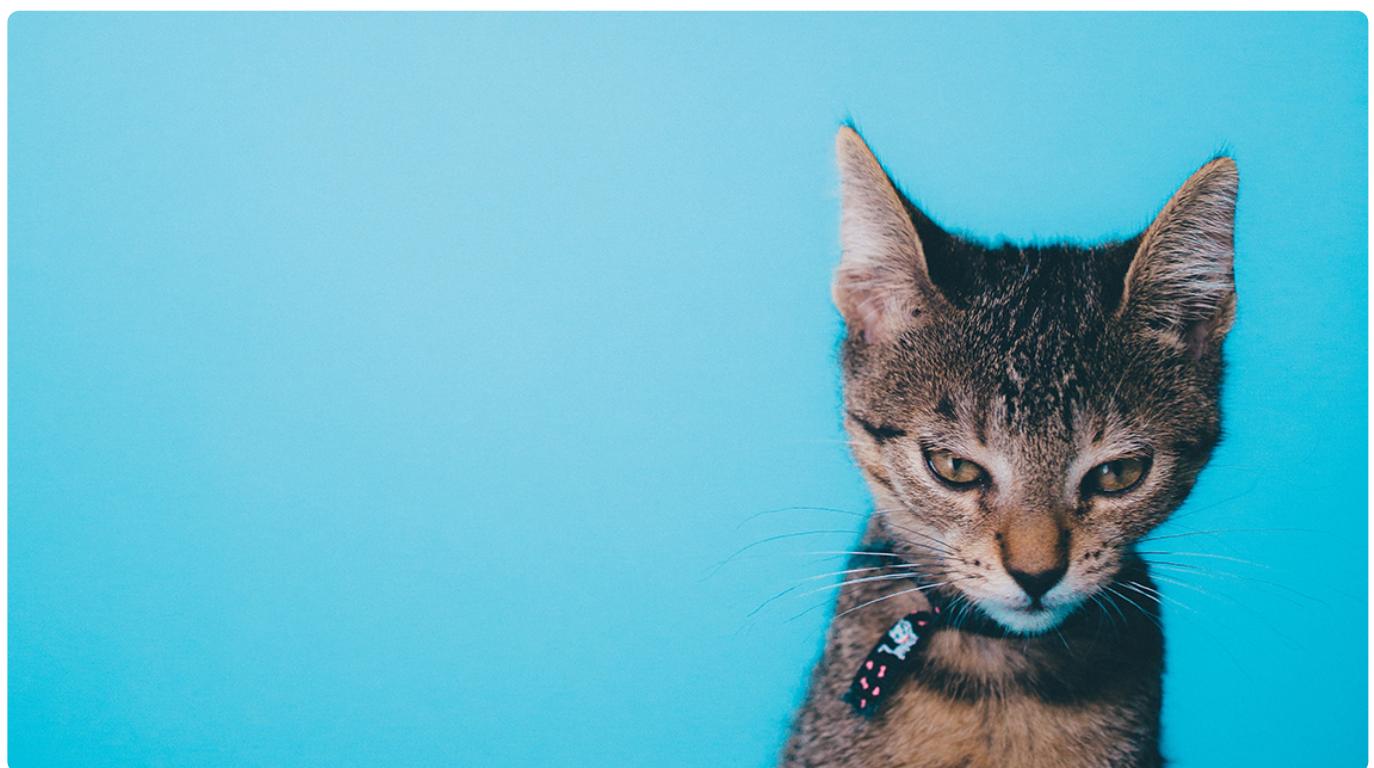


03 | HTML语义：div和span不是够用了吗？

2019-01-22 winter

重学前端

[进入课程 >](#)



讲述: **winter**

时长 16:17 大小 14.93M



你好，我是 winter。

今天这篇是我们正式开篇的第一篇文章，我想和你聊聊 HTML。

我猜屏幕那一边的你估计会说：“HTML 我很熟悉了，每天写，这不是初级程序员才学的内容么，这我还能不会吗？”

其实在我看来，HTML 并不简单，它是典型的“入门容易，精通困难”的一部分知识。深刻理解 HTML 是成为优秀的前端工程师重要的一步。

我们在上一篇文章中讲到了，HTML 的标签可以分为很多种，比如 head 里面的元信息类标签，又比如 img、video、audio 之类的替换型媒体标签。我今天要讲的标签是：语义类

标签。

语义类标签是什么，使用它有什么好处？

语义类标签也是大家工作中经常会用到的一类标签，它们的特点是视觉表现上互相都差不多，主要的区别在于它们表示了不同的语义，比如大家会经常见到的 section、nav、p，这些都是语义类的标签。

语义是我们说话表达的意思，多数的语义实际上都是由文字来承载的。语义类标签则是纯文字的补充，比如标题、自然段、章节、列表，这些内容都是纯文字无法表达的，我们需要依靠语义标签代为表达。

在讲语义之前，我们来说说为什么要用语义。

现在我们很多的前端工程师写起代码来，多数都不用复杂的语义标签，只靠 div 和 span 就能走天下了。

这样做行不行呢？毫无疑问答案是行。那这样做好不好呢？按照正确的套路，我应该说不好，但是在很多情况下，答案其实是好。

这是因为在现代互联网产品里，HTML 用于描述“软件界面”多过于“富文本”，而软件界面里的东西，实际上几乎没有语义的。比如说，我们做了一个购物车功能，我们一定要给每个购物车里的商品套上 ul 吗？比如说，加入购物车这个按钮，我们一定要用 Button 吗？

实际上我觉得没必要，因为这个场景里面，跟文本中的列表，以及表单中的 Button，其实已经相差很远了，所以，我支持在任何“软件界面”的场景中，直接使用 div 和 span。

不过，在很多工作场景里，语义类标签也有它们自己无可替代的优点。正确地使用语义标签可以带来很多好处。

语义类标签对开发者更为友好，使用语义类标签增强了可读性，即便是在没有 CSS 的时候，开发者也能够清晰地看出网页的结构，也更为便于团队的开发和维护。

除了对人类友好之外，语义类标签也十分适宜机器阅读。它的文字表现力丰富，更适合搜索引擎检索（SEO），也可以让搜索引擎爬虫更好地获取到更多有效信息，有效提升网页

的搜索量，并且语义类还可以支持读屏软件，根据文章可以自动生成目录等等。

不过，不恰当地使用语义标签，反而会造成负面影响。这里我们举一个常见的误区作为例子。我们都知道 ul 是无序列表，ol 是有序列表，所以很多接触过语义这个概念，半懂不懂的前端工程师，特别喜欢给所有并列关系的元素都套上 ul。

实际上，ul 是长成下面这种样子的（以下来自 HTML 标准）。

I have lived in the following countries:

Switzerland

Norway

United Kingdom

United States

ul 多数出现在行文中间，它的上文多数在提示：要列举某些项。但是，如果所有并列关系都用 ul，会造成大量冗余标签。

错误地使用语义标签，会给机器阅读造成混淆、增加嵌套，给 CSS 编写加重负担。

所以，对于语义标签，我的态度是：“用对”比“不用”好，“不用”比“用错”好。当然了，我觉得有理想的前端工程师还是应该去追求“用对”它们。

与 JavaScript 这样严格的编程语言相比，HTML 中语义标签的使用更接近我们平常说话用的自然语言。我们说话并没有唯一的标准措辞，语义标签的使用也是一样。下面，我挑选了几种（我认为）比较重要的语义标签使用场景，来为你介绍一下。

作为自然语言延伸的语义类标签

其实语义问题不仅仅属于理科，它还是个文科问题。

所以我们这里讲语义标签的第一个场景，也是最自然的使用场景，就是：作为自然语言和纯文本的补充，用来表达一定的结构或者消除歧义。

我们先来看看“表达一定的结构”这个场景。

在日语中，有一个语法现象叫做：ルビ，它的读音是 ruby（著名的 ruby 语言就是据此命名的），它中文的意思大约类似于注音或者意思的注解，它的形式可以看下图：



图中的例子选自动画片《某科学的超电磁炮》第二季第一话。图中把 teleport 放在空间移动上方的用法，就是日文中 ruby 的用法。“空间移动”是动画中白井黑子的技能，这里动画字幕上写的是“空间移动”，动画里的台词则用了英文发音“Teleport”，这里就形成了一个使用 ruby 的场景。

ruby 的这个形式，在中国的网友中间最近被玩出了新花样，比如表情包。



有时候微信聊天，不能用 ruby 这样的东西真的是好急啊，只好用括号代替，效果真是差了不少。

在 HTML5 中，就引入了这个表示 ruby 的标签，它由 ruby、rt、rp 三个标签来实现。

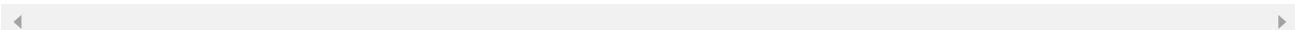
所以说，这些情况里存在的语义，其实原本就存在了，只是我们用纯文字是没法表达的，HTML 作为一种“超文本”语言，支持这些文字表达就是必要的了。

还有一种情况是，HTML 的有些标签实际上就是必要的，甚至必要的程度可以达到：如果没有这个标签，文字会产生歧义的程度。

这里我们可以介绍一下 em 标签。

 复制代码

1 今天我吃了一个苹果。

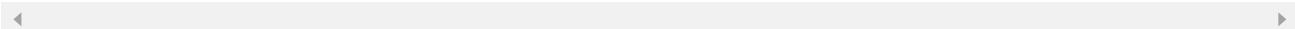


我们看看这句话，看上去它很清楚，但是实际上，这句话放到不同上下文中，可能表达完全不同的意思。

 复制代码

1 昨天我吃了一个香蕉。

2 今天我吃了一个苹果。

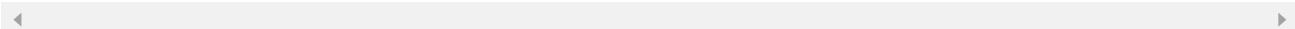


再比如：

 复制代码

1 昨天我吃了两个苹果。

2 今天我吃了一个苹果。



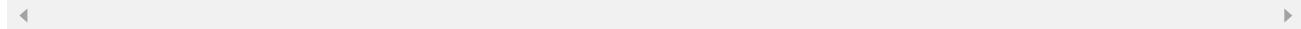
试着读一读，这两段里面的“今天我吃了一个苹果”，你是不是发现读音不自觉地发生了变化？

实际上，不仅仅是读音，这里的意思也发生了变化。前一段中，表示我今天吃的是苹果，而不是别的什么东西，后一段中，则表示我今天只吃了一个苹果，没有多吃。

当没有上下文时，如何消除歧义呢？这就要用到我们的 em 标签了。em 表示重音：

 复制代码

- 1 今天我吃了一个 `` 苹果 ``。
- 2 今天我吃了 `` 一个 `` 苹果。



通过 `em` 标签，我们可以消除这样的歧义。

一些文章常常会拿 `em` 和 `strong` 做对比，实际上，我们只要理解了 `em` 的真正意思，它和 `strong` 可谓天差地别，并没有任何混淆的可能。

作为标题摘要的语义类标签

介绍完自然语言的语义场景后，我想介绍的另一个语义重要使用场景，就是文章的结构。中国古代小说就形成了“章 - 回”的概念，西方的戏剧也有幕的区分，所以人类的自然语言作品也是如出一辙。

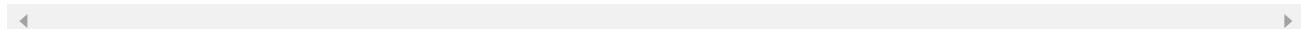
HTML 也应该支持这样的需求。HTML 语义标签中，有不少是用于支持这样的结构的标签。

语义化的 HTML 能够支持自动生成目录结构，HTML 标准中还专门规定了生成目录结构的算法，即使我们并不打算深入实践语义，也应该尽量在大的层面上保证这些元素的语义化使用。

首先我们需要形成一个概念，一篇文档会有一个树形的目录结构，它由各个级别的标题组成。这个树形结构可能不会跟 HTML 元素的嵌套关系一致。

 复制代码

- 1 例如：
- 2
- 3 `<h1>HTML 语义 </h1>`
- 4 `<p>balah balah balah balah</p>`
- 5 `<h2> 弱语义 </h2>`
- 6 `<p>balah balah</p>`
- 7 `<h2> 结构性元素 </h2>`
- 8 `<p>balah balah</p>`
- 9



这段 HTML 几乎是平铺的元素，但是它的标题结构是：

HTML 语义

弱语义

结构性元素

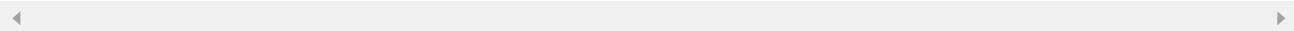
.....

h1-h6 是最基本的标题，它们表示了文章中不同层级的标题。有些时候，我们会有副标题，为了避免副标题产生额外的一个层级，我们使用 hgroup 标签。

我们来看下有 / 无 hgroup 的对比：

 复制代码

```
1 <h1>JavaScript 对象 </h1>
2 <h2> 我们需要模拟类吗? </h2>
3 <p>balah balah</p>
4 .....
```



此段生成以下标题结构：

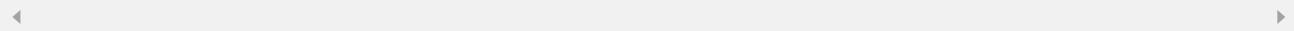
JavaScript 对象

我们需要模拟类吗?

...

 复制代码

```
1 <hgroup>
2 <h1>JavaScript 对象 </h1>
3 <h2> 我们需要模拟类吗? </h2>
4 </hgroup>
5 <p>balah balah</p>
6 .....
```



这一段生成以下标题结构：

JavaScript 对象——我们需要模拟类吗?

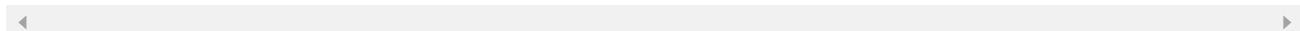
...

我们通过两个效果的对比就可以知道，在 hgroup 中的 h1-h6 被视为同一标题的不同组成部分。

从 HTML 5 开始，我们有了 section 标签，这个标签可不仅仅是一个“有语义的 div”，它会改变 h1-h6 的语义。section 的嵌套会使得其中的 h1-h6 下降一级，因此，在 HTML5 以后，我们只需要 section 和 h1 就足以形成文档的树形结构：

 复制代码

```
1 <section>
2   <h1>HTML 语义 </h1>
3   <p>balah balah balah balah</p>
4   <section>
5     <h1> 弱语义 </h1>
6     <p>balah balah</p>
7   </section>
8   <section>
9     <h1> 结构性元素 </h1>
10    <p>balah balah</p>
11  </section>
12 .....
13 </section>
```



这段代码同样会形成前面例子的标题结构：

HTML 语义

弱语义

结构性元素

.....

作为整体结构的语义类标签

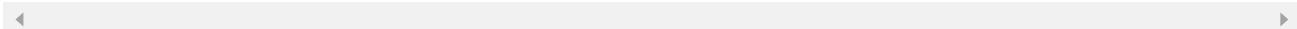
我们想介绍的最后一个场景是，随着越来越多的浏览器推出“阅读模式”，以及各种非浏览器终端的出现，语义化的 HTML 适合机器阅读的特性变得越来越重要。

应用了语义化结构的页面，可以明确地提示出页面信息的主次关系，它能让浏览器很好地支持“阅读视图功能”，还可以让搜索引擎的命中率提升，同时，它也对视障用户的读屏软件更友好。

我们正确使用整体结构类的语义标签，可以让页面对机器更友好。比如，这里一个典型的 body 类似这样：

复制代码

```
1 <body>
2   <header>
3     <nav>
4     .....
5     </nav>
6   </header>
7   <aside>
8     <nav>
9     .....
10    </nav>
11  </aside>
12  <section>.....</section>
13  <section>.....</section>
14  <section>.....</section>
15  <footer>
16    <address>.....</address>
17  </footer>
18 </body>
```



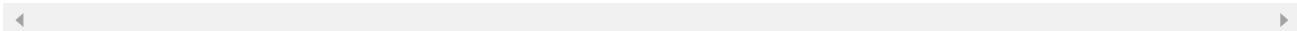
在 body 下面，有一个 header，header 里面是一个 nav，跟 header 同级的有一个 aside，aside 里面也有一个 nav。接下来是文章的整体，也就是一个一个的 section。section 里面可能还有嵌套，但是我们就不管了，最后是一个 footer，这个 footer 里面可能有 address 这样的内容。

除此之外，还有 article，article 是一种特别的结构，它表示具有一定独立性质的文章。所以，article 和 body 具有相似的结构，同时，一个 HTML 页面中，可能有多个 article 存在。

一个典型的场景是多篇新闻展示在同一个新闻专题页面中，这种类似报纸的多文章结构适合用 article 来组织。

 复制代码

```
1 <body>
2     <header>.....</header>
3     <article>
4         <header>.....</header>
5         <section>.....</section>
6         <section>.....</section>
7         <section>.....</section>
8         <footer>.....</footer>
9     </article>
10    <article>
11        .....
12    </article>
13    <article>
14        .....
15    </article>
16    <footer>
17        <address></address>
18    </footer>
19 </body>
```



body 里面有自己的 header 和 footer，然后里面是竖篇的 article，每一个 article 里面都有自己的 header、section、footer。这是一个典型的多文章结构。

在这个结构里，我们看到了一些新标签，我也来逐个介绍一下。

header，如其名，通常出现在前部，表示导航或者介绍性的内容。

footer，通常出现在尾部，包含一些作者信息、相关链接、版权信息等。

header 和 footer 一般都是放在 article 或者 body 的直接子元素，但是标准中并没有明确规定，footer 也可以和 aside，nav，section 相关联（header 不存在关联问题）。

aside 表示跟文章主体不那么相关的部分，它可能包含导航、广告等工具性质的内容。

aside 很容易被理解为侧边栏，实际上二者是包含关系，侧边栏是 aside，aside 不一定是侧边栏。

aside 和 header 中都可能出现导航（nav 标签），二者的区别是，header 中的导航多数是到文章自己的目录，而 aside 中的导航多数是到关联页面或者是整站地图。

最后 footer 中包含 address，这是个非常容易被误用的标签。address 并非像 date 一样，表示一个给机器阅读的地址，而是表示“文章（作者）的联系方式”，address 明确地只关联到 article 和 body。

总结

本篇中我们介绍了一些基本原则和 HTML 文档的整体结构，从整体上了解了 HTML 语义。

至此，我们可以回答是否要语义化的问题：我们应该分开一些场景来看语义，把它用在合适的场景下，可以获得额外的效果。本篇文中，我们至少涉及了三个明确的场景：

自然语言表达能力的补充；

文章标题摘要；

适合机器阅读的整体结构。

下一篇中，我们会继续深入到更细致的结构中，进一步了解语义。你在工作中是否在使用语义化的标签开发？学习过本篇之后，答案有没有变化呢？你可以给我留言，我们一起讨论。

猜你喜欢

The image shows the front cover of a book titled 'Vue 开发实战' (Vue Development Practical Guide) by 唐金州 (Tang Jinzhou). The cover features a portrait of the author, a man with short dark hair wearing a blue sweatshirt. To the right of the portrait, vertical text reads '唐金州' (Tang Jinzhou), '一点资讯前端技术专家' (Front-end Technical Expert at iQIYI), and 'Ant Design Vue 作者' (Author of Ant Design Vue). The main title 'Vue 开发实战' is in large, bold, dark blue font, with the subtitle '从 0 开始搭建大型 Vue 项目' (Building a Large-scale Vue Project from Scratch) below it. A blue button at the bottom left says '戳此试读' (Read Preview) with a hand cursor icon.

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 02 | 列一份前端知识架构图

下一篇 04 | HTML语义：如何运用语义类标签来呈现Wiki网页？

精选留言 (171)

 写留言



莲

2019-01-22

 87

我写前端时间不长，写的都是小东西，确实觉得div和span够用

我认为html标签和自然语言的演化肯定会是一种逻辑：

汉语中「地」 「的」 「得」 的正确用法今天大部分中国人根本不会，都只用「的」字， ...

展开 ▼



Scorpio

2019-01-22

 46

老师也有看炮姐吗? (〃•Θ•〃)

展开 ▼

作者回复: 哈哈哈 不然怎么能从里面挑出ruby



王大可

2019-01-22

 45

咦？根据目录，难道不是该介绍模块一 javascript相关的吗

展开 ▼



人艰不拆

2019-01-23

 44

老夫写页面就是一把梭，div，div，div，div，什么都是div，display改一改什么都能写



完了...

2019-01-22

 44

完了...今天文章中的一半标签都不认识.....

展开 ▼



馒头小哥

2019-01-22

36

其实语义化更重要的是在于规范，渲染出来的网页是给大多数用户看的。还有一小部分用户比如程序员、机器人、视障用户。

如果一个页面只有 span 和 div，视障软件如果把这个网页读给用户？读 “div 开始 class="tile" 今天天气很好 div 结束” 还是读 “标题：今天天气很好” 那个方式更好呢...
展开 ▾



Derrickxy...

2019-01-22

24

目前开发的是企业应用，也就是更关注于“软件界面”，没有使用语义化标签。使用div, span等就实现了需求。

企业应用不需要做针对性的SEO，也不需要像文章那样分层，划分章节。

语义化标签在企业应用中，除了增加可读性外，是否还有其他提升？

谢谢～

展开 ▾



雪松

2019-01-22

18

就拿写一个页面来说，直接div/span，让我更加专注页面布局样式内容等信息，如果去语义化，反而造成困扰，不过这个困扰是建立在自己不熟悉语义化的基础之上的。

所以我认为应该尽量追求语义化，这不仅是便于浏览器搜索引擎，也是竞争力的一种体现。

展开 ▾



Artyhacke...

2019-01-22

17

以前学习的时候刻意使用过语义化标签，但工作以后直接用react，组件也直接上ant-design，几乎就只需要div和span了。。



mfist

2019-01-22

14

1html语义化的优点，有利于人或机器更好的解析语义。

2日常业务开发中很多时候只用div span，站在应该走出自己舒适区的角度，应该正确的用语义化

3 我觉得在封装项目通用ui组件更需要考究语义化，这样就帮助使用者获取语义化的优点，屏蔽语义化的缺点

展开 ▼



北京知府

2019-01-22

12

我们是做业务系统的，团队中有一个同事（纯前端偏UI）特别喜欢使用语义化标签，但是我们在维护他的代码时总感觉他的代码乱糟糟的不好维护，很多样式就直接写标签来定义；而我们其他人（擅长做后台的）则喜欢用div，然后通过有业务含义的class来定义样式。

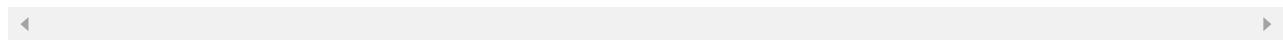
个人觉得做业务系统，特别是团队一起协作开发，还是尽量少用语义化标签，这样能减...

展开 ▼

作者回复: 1. 用语义化标签，跟用标签名选择器是两码事。

2. 给自己贴“做业务系统”的标签，潜台词是给自己找借口。

3. 不要跟没希望的团队一起工作。



Zp

2019-01-22

12

不是该先讲js?

展开 ▼



Geek_be7ad...

2019-01-24

10

例如「地」「的」「得」、「他」「她」「它」的区分，我觉得不应该因为分不清而不再区分，而是因为它有意义而去刻意地区分，而且也未必都不分，至少每年几百几千万的本科毕业生应该分得清。同理，虽然it行业入门是英雄不问出处，但是也大部分都有一定基础，所以有助于梳理页面结构的语义化概念还是应该大力推行的。

展开 ▼



Mowtwo

2019-01-23

10

虽然文章确实讲了很多有用的东西，但是对于hgroup的例子我觉得还是可以得到一些指正。

至少到目前为止，我尝试了一下，hgroup已经不再可以在网页上有人和作用了...而我搜索以后，也找到了一篇关于hgroup已经在HTML5.1标准下被取消的信息。所以文章中所提出的部分内容已经不再有效，希望重视。

展开 ▼

作者回复: 我本人比较倾向于WHATWG的living standard，所以保留了这个标签。

多数语义标签都不会产生实际效果，hgroup针对outline算法仍然有效。



sprintsy

2019-01-22

10

语义化标签适合的场景是不是很有限？

现实开发中除了文档、博客等类型的项目很少用到，我自己也很少用到语义化标签，因为想到其语义就觉得不适合现在的场景。



无恙

2019-01-22

8

语义开发确实能很好地提升源码的结构，方便阅读。但前提是团队人员都能正确使用语义化标签，否则就会出现div/span和语义标签各种嵌套，很难受



leslee

2019-01-22

8

感觉只有特别适用的场景才能适用语义化标签。工作中的需求太乱了。

展开 ▼



hhk

2019-01-23

6

我理解的语义化，就是对机器友好对人友好。在富文本这类场景时，我们应该尽量地语义化；写软件界面时，保持整体上的结构化即可



huayonbr...

2019-01-22

6

几乎不用语义化的开发

展开 ▼



Carson

2019-01-22

5

运气不错，在工作中一直刻意练习使用语义化标签。在往复阅读文档，修改标签中，能慢慢提升对标签的理解。

这篇文章再次感受到知识结构「完备性」的重要。

...

展开 ▼