## 摘要

今天要跟大家分享的主题是《[前端的多重分离解耦](https://speakerdeck.com/wenbin5243/qian-duan-de-duo-zhong-fen-chi-jie-ou)》，在进入正题之前，先看看经常引起争论的几个问题：

### 价值观问题

#### 完成比完美更重要（faceboock）

我们经常会想一开始就把东西做的很完美，但是现实没有那么美好。我们要在很短的工期内完成大部分的工作，不然就得加班赶工，但是后面往往有很长的空闲期可以来做优化。所以更好的办法是先完成再完美，一开始要构思好，后期才不会乱。

#### 一致比优秀更重要《高流量网站CSS开发技术》

在多人合作的时候，每个人的想法都不太一样，先保持一致，再考虑做的更优秀。我一直在思考如何让每个人写出来的结构基本都一样？如何在原型确认后就开始写结构，让视觉、前端、后端能并行工作。这就要求前端足够的分离，各个阶段都不冲突。

### 团队合作问题

#### 学习成本

技术都是会流动的，如何让别人理解我们写的东西？如何让新来的成员快速的进入工作？ 面对那么多的新名字，具体是干什么的？会不会牵一发动全身？都要实践了才知道。

#### 沟通成本

在需求变动的时候，团队如何更好的协调，降低沟通成本？好的办法是让每个人的工作独立开，相互没有影响。

#### 维护成本

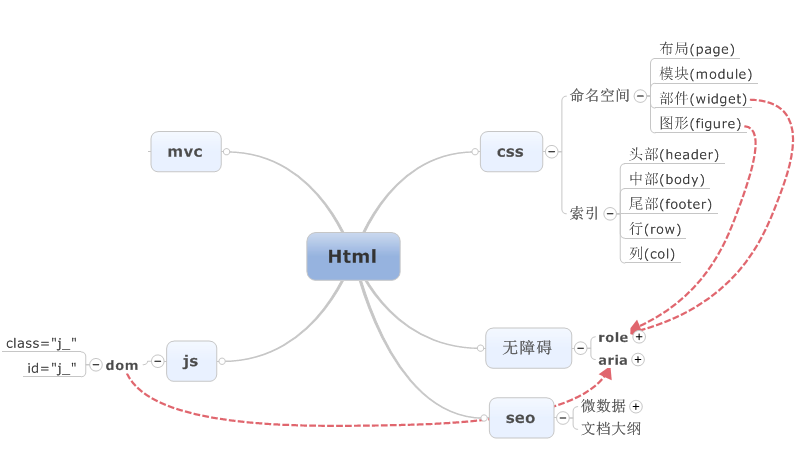
如何让自己做的东西，让后面的人好维护，而不是推翻重做？整理部件库是一个很好的方法。把公用的元素展示出来，方便后期直接调用，避免重复造轮子。

如何把这些成本降到最低，就是今天要讲的话题。

## 如何分离

我们都知道html，css，js要分离，可以减少代码的耦合，一般的做法是不使用表现型的标签、内联样式，事件的绑定不和html写在一起。但是这样往往分离的不够彻底。

要跟html打交道的只有css和js吗？不止，这里还加入无障碍，seo，mvc。



这张图体现了大致的构思，有一些内容比较多，暂时隐藏，下面会详细的介绍。

### Css

Css都用class来命名，主要是方便公用。这里引入命名空间和索引。

#### 命名空间

page类用来设置页面布局，module类具有唯一性，widget是公用的块级元素，图形是公用的行内块或行内元素。

##### 布局（page）

以p为开头的只设置宽度和间距，和模块分离开。

##### 模块（module）

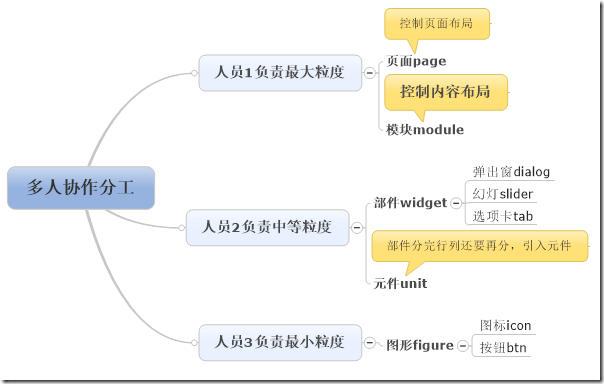
以m开头的在同一个页面内具有唯一性，可以用来控制公用的元素。模块的布局要和部件分离。

##### 部件（widget）

以w开头的说明是公用的元素，部件用“w\_功能”来命名，功能可以参考无障碍的role属性。部件的布局要和图形分离。如果要做细微的修改用模块来控制，不会污染到其他地方。

##### 图形（figure）

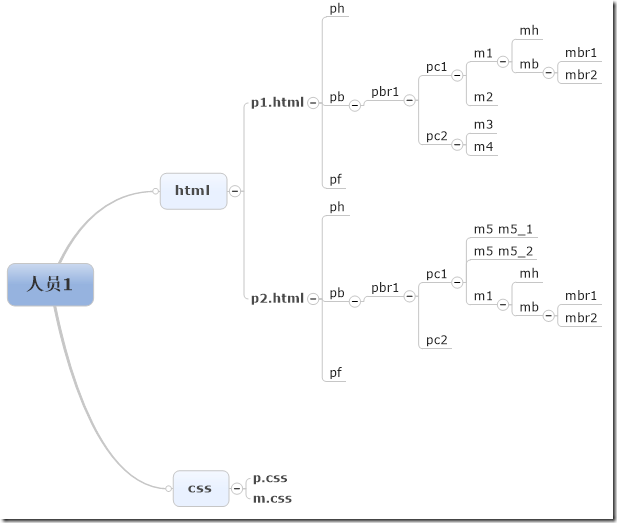
以f开头的是最小的粒度，一般可以用一个标签来设置样式，公用程度高，可以放在不同的部件或模块里面。



#### 索引

索引就是各个命名空间的细分，方便控制到每个dom节点。页面、模块、部件都可以分头中尾，头中尾里面再分行列，这样就有足够的选择器可以使用，而不是什么标签没用到就用什么标签。布局和一些细微的部分就交索引来控制，把公用部分组合起来。

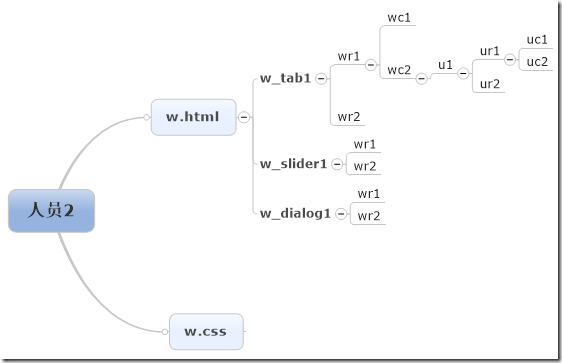
##### 索引（布局和模块）



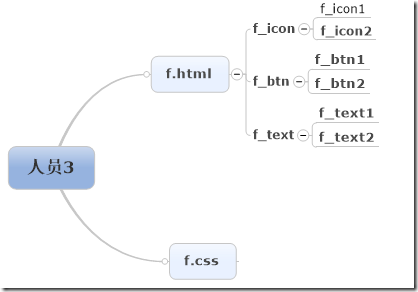
为什么这么划分？举个例子，我们去电影院看电影，电影票上面会写几号厅，几排几号，这样我们就能找到自己的位置。模块也是一样我们只要给他一个唯一的数值，不同的人分配不同的地址空间，这样就不会有冲突了。

为什么模块不用标题来命名呢？举个例子，1号厅放映的是《霍比特人2》，但是我们不会叫他为霍比特人2号厅，因为他可能放映其他的影片。模块也一样，标题可能会变，而且同样的标题可能展现形式会不同，所以模块作为唯一的需要跟内容分离。

##### 索引（公用块级）

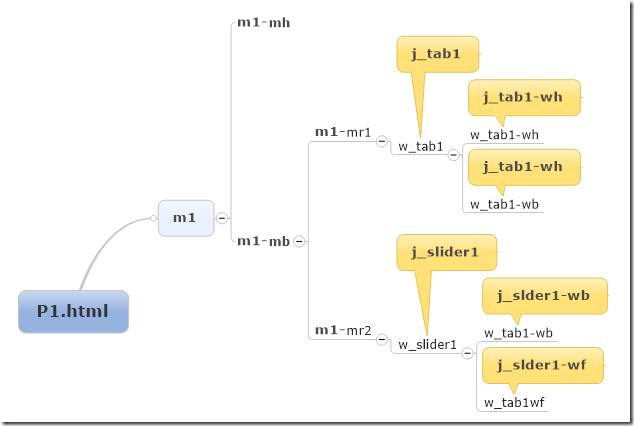


##### 索引（公用行内）



公用的部分需要放在一起展示，方便后期的查找，颗粒分离的尽可能的细，重用度会更高。

### Js



Js用的比较多的是dom操作，因为ie6-8不支持直接获取class，所以用id会比较快，模块具有唯一性，可以用模块来定id，加上j前缀。公用部分的class也加上j前缀，跟样式的class区分开。这样可以比较方便的看出html哪里有dom操作，修改也不会跟样式有冲突。

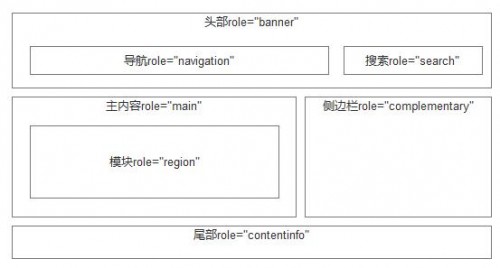
### 无障碍

#### Role属性

Role属性可以让屏幕阅读器读出来，帮助视障用户更好的理解网站，下表是nvda和争渡读屏对各浏览器的支持情况，c代表chrome，i代表ie，f代表firefox。样式的公用元素可以用这些名字来命名，加上命名空间的前缀。

|  |  |  |
| --- | --- | --- |
| Role属性 | Nvda | 争渡读屏公益版 |
| alert |  | 警告(i,f) |
| alertdialog | 对话框(c,i,f) | 对话框(i,f) |
| application | 应用程序(f) | 窗格(i)主窗口(f) |
| article |  |  |
| banner | 横幅区书签(c,i,f) |  |
| button | 按钮(c,i,f) | 按下按钮(i,f) |
| checkbox | 复选框未选中(c,i,f) | 复选框未选中(i,f) |
| columnheader | 列标题(c,i,f) | 列标头(i,f) |
| combobox | 组合框(c,i),组合框已折叠(f) | 组合框(i,f) |
| complementary | 其他内容区书签(c,f) |  |
| contentinfo | 内容信息区书签(c,i,f) |  |
| definition |  |  |
| dialog | 对话框(c,i,f) | 对话框(i,f) |
| directory | 列表(c,i,f) | 列表(i)列表共0项(f) |
| document |  |  |
| form |  |  |
| grid | 表格(c,i)表格有0行和0列编辑框(f) | 表格(i,f) |
| gridcell | 单元格(c)第1项共1项(f) | 单元格(i,f) |
| group |  | 分组(i,f) |
| heading | 标题(c,i,f) | 可编辑文本(i) |
| img | 图形(c,i,f) | 图形(i,f) |
| link | 链接(c,i,f) | 链接(i,f) |
| list | 列表(c,i)列表有1个项目(f) | 列表(i)列表只读共0项(f) |
| listbox | 列表(c,i,f) | 列表(i)列表共0项(f) |
| listitem | 列表项目(c) | 列表项目(i)只读(f) |
| log |  |  |
| main | 主要内容区书签(c,i,f) |  |
| marquee |  | 动画(i,f) |
| math |  | 数学公式(f) |
| menu | 菜单(c,i,f) | 菜单(i) |
| menubar | 菜单栏(c,i,f) | 菜单栏(i,f) |
| menuitem | 第1项共3项(f) |  |
| menuitemcheckbox | 可单击(f) |  |
| menuitemradio | 可单击(f) |  |
| navigation | 导航区书签(c,i,f) |  |
| note |  |  |
| option | 未选择第1项共1项(f) | 列表项目(i) |
| presentation |  |  |
| progressbar | 进度栏(c,f) | 进度栏(i)进度栏只读(f) |
| radio | 单选按钮未选中(c,i)单选按钮未选中第1项共1项(f) | 单选按钮未选中(i,f) |
| radiogroup |  | 分组(i,f) |
| region |  | 窗格(i)分组(f) |
| row | 行(c,i)行未选择第1项共1项(f) | 行(f) |
| rowgroup |  |  |
| rowheader | 行标题(c,f) | 行标头(i,f) |
| scrollbar | 没听清(c,f) | 滚动条(f) |
| search | 搜索区书签(c,i,f) |  |
| separator | 分隔符(c,i,f) | 分隔符(i,f) |
| slider | 滑块(c,i,f) | 滑块(i,f) |
| spinbutton | 旋钮(c,i,f) | 微调组合框(i,f) |
| status |  | 状态栏(i,f) |
| tab | 选项卡(c,i)选项卡已选择第1项共1项(f) | 选项卡(i,f) |
| tablist | 选项卡列表(c,i,f) | 选项卡列表(i,f) |
| tabpanel |  | 窗格(i)属性页(f) |
| textbox | 编辑框(c,i,f) | 可编辑文本(i) |
| timer |  |  |
| toolbar | 工具栏(c,i,f) | 工具栏(i,f) |
| tooltip |  |  |
| tree | 树视图(c,i,f) | 树视图(i,f) |
| treegrid | 表格(c,i)树式图(f) | 树视图(f) |
| treeitem | 树视图项目(c)未选择第1项共1项第一级(f) | 树视图项目(i) |

##### Landmark role



#### Aria属性

Aria属性可以加强交互效果对屏幕阅读器的友好。Dom操作的时候可以适当添加这些属性。

|  |  |  |
| --- | --- | --- |
| aria属性 | nvda | 争渡读屏公益版 |
| aria-activedescendant=”" |  |  |
| aria-atomic=”true” |  |  |
| aria-atomic=”false” |  |  |
| aria-autocomplete=”both” |  |  |
| aria-autocomplete=”inline” |  |  |
| aria-autocomplete=”list” |  |  |
| aria-autocomplete=”none” |  |  |
| aria-busy=”false” |  |  |
| aria-busy=true”" | 忙碌中(i,f) |  |
| aria-checked=”false” |  |  |
| aria-checked=”mixed” | 混合选择(c) |  |
| aria-checked=”true” |  |  |
| aria-checked=”undefined” |  |  |
| aria-controls=”" |  |  |
| aria-describedby=”" |  |  |
| aria-disabled=”false” |  |  |
| aria-disabled=”true” | 不可用(c,i,f) | 不可用(i) |
| aria-dropeffect=”copy” | 放置目标(i,f) |  |
| aria-dropeffect=”execute” | 放置目标(i,f) |  |
| aria-dropeffect=”link” | 放置目标(i,f) |  |
| aria-dropeffect=”move” | 放置目标(i,f) |  |
| aria-dropeffect=”none” |  |  |
| aria-dropeffect=”popup” | 放置目标(i,f) |  |
| aria-expanded=”false” | 已折叠(i,f) |  |
| aria-expanded=”true” | 已展开(i,f) |  |
| aria-expanded=”undefined” |  |  |
| aria-flowto=”" |  |  |
| aria-grabbed=”false” | 可拖拽(i,f) |  |
| aria-grabbed=”true” | 拖拽(i,f) |  |
| aria-grabbed=”undefined” |  |  |
| aria-haspopup=”false” |  |  |
| aria-haspopup=”true” | 打开子菜单(c)子菜单(i,f) | 展开菜单(i) |
| aria-hidden=”false” |  |  |
| aria-invalid=”false” | 错误的记录(c,f) |  |
| aria-invalid=”grammar” | 错误的记录(c,f) |  |
| aria-invalid=”spelling” | 错误的记录(c,f) |  |
| aria-invalid=”true” | 错误的记录(c,f) |  |
| aria-label=”" |  |  |
| aria-labelledby=”" |  |  |
| aria-level=”" |  |  |
| aria-live=”assertive” |  |  |
| aria-live=”off” |  |  |
| aria-live=”polite” |  |  |
| aria-multiline=”false” |  |  |
| aria-multiline=”true” | 多行(i) |  |
| aria-multiselectable=”false” |  |  |
| aria-multiselectable=”true” |  |  |
| aria-orientation=”horizontal” |  |  |
| aria-orientation=”vertical” |  |  |
| aria-owns=”"（关联到指定的id） |  |  |
| aria-posinset=”" |  |  |
| aria-pressed=”false” |  |  |
| aria-pressed=”mixed” |  |  |
| aria-pressed=”true” |  |  |
| aria-pressed=”undefined” |  |  |
| aria-readonly=”false” |  |  |
| aria-readonly=”true” | 必须的(c,i,f) |  |
| aria-relevant=”additions” |  |  |
| aria-relevant=”all” |  |  |
| aria-relevant=”removals” |  |  |
| aria-relevant=”text” |  |  |
| aria-required=”false” |  |  |
| aria-required=”true” |  |  |
| aria-selected=”false” |  |  |
| aria-selected=”true” | 已选择(c,i) |  |
| aria-selected=”undefined” |  |  |
| aria-setsize=”" |  |  |
| aria-sort=”ascending” | 升序(i,f) |  |
| aria-sort=”descending” | 降序(i,f) |  |
| aria-sort=”none” |  |  |
| aria-sort=”other” | 排序(i,f) |  |
| aria-valuemax=”" |  |  |
| aria-valuemin=”" |  |  |
| aria-valuenow=”" |  |  |
| aria-valuetext=”" |  |  |

### Seo

#### 文档大纲



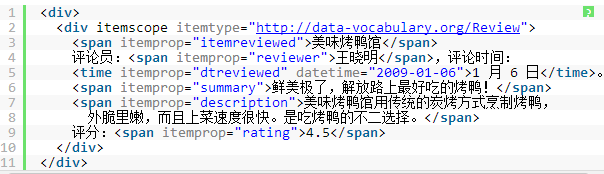
文档大纲是文档的h标题顺序，每个页面有一个唯一的标题h1，每个模块有一个h2标题。

#### 结构化数据

微数据、微元素、RDFa效果都差不多，主要是让谷歌的搜索结果更丰富，提升搜索体验。



微元素主要是用class来标记，为了跟css区分，不用这个。谷歌推荐使用微数据，主要谈谈这个。



微数据主要用itemscope，itemtype，itemprop等属性来标记，跟其他的分离开了，所有的元素如下表。

### Mvc

以前跟html比较有关的是视图层，用的比较多的是php的smarty模板引擎。现在js也有mvc，比如angularjs，下面的代码用到了ng-属性，跟其他的分离开了。

