网页开发模块化

说起模块化，也许我们首先想到的是编程中的模块设计，以功能块为单位进行程序设计，最后通过模块的选择和组合构成最终产品。把这种思想运用到页面构建中，也已经不是什么新鲜事。相信很大一部分页面构建工程师都经历了这样几个阶段：

第一阶段：是在一个 css 文件中把多个页面按自己的习惯顺序从上往下编写样式，基本不考虑有无公用样式，以完成设计呈现为首要目的；

第二阶段：是提取不同页面中的通用样式，如公用颜色、图标、按钮等，实现一些基本元素的复用；

第三阶段：是提取公用功能模块，如导航、版权信息等，实现部分公用模块的复用。

刚才描述的第三阶段的方法已经包含了模块化思想，不少团队也都有一套成熟的模块化开发方案。而我第一次听说模块化构建方式，是三年前在一家韩国互联网企业工作时，某些产品中要求使用一种称为 UIO 方式，模块化通用的功能模块或组件，以达到最大程度的模块独立性与复用性，当时团队中很多同事和我一样，认为这种工作方式约束了编码的自由性，过多的结构约束反而降低了工作效率，加之产品之间也存在不统一，最后并没有运用到整个团队。

那么，如果我们运用模块化构建的方式，优势在哪呢？也许在开始尝试之处，需要一个适应的过程，可能会使团队成员出现之前类似我当时的想法，但当大家都适应并熟练这种工作方式之后，必定能极大地提高页面构建的效率。

假设有这样一个场景，团队接到一个页面非常多、工作量非常大的紧急项目，

第一个团队这么做：组长给每人分配几个页面，大家分头做完各自的页面，统一交付，对于不同页面之间结构呈现相似的模块，细心点的团队可能会约定让某个人写好，再复制给每个需要用到的人，不太在意的，则让每个人把各自页面上的所有内容都写一遍，已完成任务为重。

第二个团队事先根据所有的页面划分公用或重复模块，再按模块唯一性分配给每个人，有人负责搭建框架，有人制作模块，最后合并框架和模块，再按开发的工作计划，顺序交付页面。对比的结果是，由于第二个团队是多人共同制作一个页面，他们能以最快的速度产出开发需要的第一页面，而且越到后期越能发现页面中可重用的模块越多，最后整个工作时间

也许能比第一个团队缩减一半。模块的复用不单是对本团队的工作时间有很大影响，同样，对于下游的开发者来说，意味着他们也不需要为相同的模块重套代码或重新开发。此外，代码的冗余量、以及产品升级时两种工作方式的代码扩展性也体现出很大的差距。再者，如果你的团队将要运用 BIGPIPE 或者 LESS 的开发方式，CSS 的模块化是最好的配合手段，或者说是必须的。

当决定使用模块化构建的工作方式时，遵循某些原则对模块化的顺利推进有很大的帮助。曾经有一篇关于面向对象 CSS 的文章中指出，

面向对象的 CSS 有两个主要原则：separate the structure from the skin，separate the container from the content。

第一个原则 体现在模块化思想可以理解为，模块的设计制作和布局框架本身相分离，意味着你的模块不能只为某个布局而编写样式，像微博这类存在换肤功能的产品更是如此，如果模块在不同的皮肤样式下需要另写很多样式甚至是修改结构的时候，这个模块的制作就是失败的；

第二个原则 说的布局与内容的分离，布局中某个位置不必只能放置某种内容，反过来可以理解为模块的灵活性和复用性。

其次遵守团队协作开发规范原则。

这个规范可以包含文件目录结构、文件和样式命名规范、图片 sprite 规范、模块划分和调用规范等，例如我们对文件目录深度的规定、公用样式使用规定、模块的样式名唯一性规定、模块文件名和样式名必须一致的规定等等，确保所有人产出的模块是统一、规范的。

按结构呈现形式划分模块的原则。

这一点和模块化编程有较大的区别，通常在编程开发时是以模块的功能来划分的，而在页面构建上，有时候不同功能的模块呈现的样式是一样的，为达到模块样式最大程度的复用，就不能按功能来划分模块，简单来说，哪些模块外观结构一样，我们就可以把它们归为一个模块，以微博右侧模块举例，“可能感兴趣的人”和“推荐应用”模块的外观是一样，都是左侧一个图片、右侧文字和功能按钮，那它们就是同一个样式模块。

模块稳固性原则。

我经常问新人一个问题，“你觉得怎样体现你写的代码质量高，比一般人好？”，大多数人会回答遵守语义化，减小不必要的嵌套，代码尽量精简。语义化和代码精简固然是评价质量的一个重要方面，但是我认为，代码是否考虑到数据遍历的合理性，是否考虑到 dom 节点的可操作性，是否考虑到因扩展造成的抗破坏行，更能体现一个页面构建工程师的水平。

模块自适应性原则。

指的是任何一个模块，都尽可能实现宽度和高度的自适应，非特殊情况不要设置模块的宽高，采取这种原则制作出的模块具有很好的即插即用功能，是高效完成页面拼合工作的重要前提。试想如果每个模块都定义了宽度，那么在不同的布局上你就必须重新定义每个模块的宽高或边距等属性来适应当前布局。

Margin-bottom 原则。

一般情况下，网页的布局都是从上到下的流式布局（多栏结构也可以看成各栏内的流式布局），所以，我们可以为每个模块统一预设 margin-bottom，达到统一间距的目的，避免出现有些模块设置上边距、有些模块设置下边距的情况发生。（左右间距通常是由布局框架的样式设置）

在制订好团队的合作规范、遵守的原则后，并不代表你就可以完全按你的思路启动工作，团队配合是多向的，除了团队内部，其他团队的支持也是不可或缺的，所以还需要以下两个前置条件：

设计必须严格遵循栅格化。

模块是独立的，但最终模块还是嵌套在布局中，因为我们的最终产出物是完整的静态页面，如何将分离的模块在最短的时间内，拼成一个符合设计师意图和产品要求的页面？栅格化是快捷的保障，在一个严格按照栅格化设计的布局框架中，工程师只需要设置好布局框架样式和分栏的内外间距，后续的工作只需要把该页面所使用的模块嵌套进来，再调用对应模块的样式，由于模块的自适应性，在所有模块准备充分的情况下，通常一个页面的拼合只需要几分钟的时间。

产品、设计与交互的规范统一。

通常在项目的某个阶段，产品和设计在模块上的统一是比较容易的，但如果在同一个项目的不同阶段，尤其是在不同项目之间或不同产品之间要达到规范统一，就不是一件简单的事情。当规范统一性出现问题时，导致模块化只停留在某个项目阶段，每次添加新功能、增加新内容都需要增加全新的模块样式，移植性和复用性大打折扣，无法发挥应有的效果。当然，产品是持续改变和创新的，我们不能要求一个产品永远按照某个规范来进行设计，但我们还是应该共同努力寻求阶段性共赢的解决方案。在微博，经过各方长时间的努力，特别是交互设计对产品功能组件的统一，构建的 WDL 规范库对我们的模块化提供了很大帮助。

根据实际情况来看，要达到所有满足的条件往往不是一帆风顺的，特别是第二个条件的达成。但是退一步来说，即使不能使模块化在每个项目、每个产品中长期稳定的发挥它的最大能量，至少可以在每一次项目任务中获得模块化给团队带来的效率提升。

如果经过大家的努力，在所有条件都满足，而且模块化工作方式能在团队顺利开展的情况下，我们依然可能会遇到各式各样的问题，一个无法避免的问题就是 —— 产品功能升级引起的模块变化，这时候是修改原有的模块还是另起一个新的模块？二是模块的划分程度，有些时候从模块的呈现和功能划分都比较模糊，有些时候对某些内容是否划为公用样式还是模块、还是页面独有内容都是见仁见智的；三是模块的分类，采取何种方式分类便于查找？类似这些问题还有很多，在不同的项目和形势下，需要具体问题具体分析，发挥团队的智慧，寻找最合理的应对方案。

虽然在实施过程中可能会遇到各种问题和团队配合之间的阻力，但是当你逐渐适应这种模块化团队构建的工作方式时，你会爱上它！而当你的团队高效地完成每个工作的时候，人们也会爱上你的团队！