





## 摘 要

一方面,自从 2005 年互联网进入 Web 2.0 的时代以来,各种类似桌面软件的 Web 应用大量涌现,网站的前端开发由此发生了翻天覆地的变化。网页不再只是承载单一的文字和图片,用户也要求更好的交互体验,这些都是依靠前端开发的技术实现的。但在前端开发的工作中普遍存在代码臃肿、维护成本大的问题。

另一方面,伴随着移动互联网发展到来的是多样化的手持设备如智能手机、平板电脑和其他消费电子产品。用户对网站项目在不同设备的使用体验要求也越来越高,这使得前端工程师的工作更加困难。

面对前端开发工作中遇到的问题,本人尝试地提出一个轻巧灵活的解决方案的设想。本人研读了 W3C 相关的 Web 标准,在此基础上了解了浏览器对 HTML 和 CSS 的解析过程。在对 HTML 和 CSS 渲染过程有了全局的了解后,又学习了关于 Web 性能优化的一些方法并分析研究了现有 CSS 框架 OOCSS,然后在 OOCSS 框架的设计思路结合自己对前端开发的理解进行扩展得出一套新的 CSS 设计模式,最后结合团队实际的需求,并使用 LESS 设计了一个统一的响应式的 CSS 框架: XSS(Extensible Style Sheets)。

XSS 的架构分为服务层与应用层。在 XSS 中,本人还提出了结构与表现分离、组件与皮肤分离、组件与容器分离三大 CSS 设计原则。总的来说, XSS 轻巧易用,应用此框架,在规范了 CSS 开发、促进团队成员间合作、提高开发效率的同时,也能容易地满足越来越高的用户体验要求。

**关键词:** 移动互联网 响应式设计 CSS 框架

## ABSTRACT

On one hand, since the Internet has entered the age of Web 2.0 in 2005, a plenty of Web applications have appeared, which result in the great changes of the Web front-end development. Web pages no longer only contain single words and images, and users also require better interaction experience on them. These are all rely on the development of the front-end technology implementation. But, the heavy code and cost of maintainance make development of the front-end difficult.

On the other hand, along with the arrival of mobile Internet development is a variety of handheld devices such as smartphones, tablets and other consumer electronic products. The use of the website project in different devices experience is required, which in result, the work of front-end engineers become more and more difficult.

To solve the front-end development problems in the work, I try to put forward a light flexible solution. I learned the web standard on W3C, parsing process of HTML and CSS behind browser. After learning the HTML and CSS rendering process, I learned about how to optimize the web performance. Then I analysis the CSS framework called OOCSS. I expand the OOCSS with my own idea, which result of a new CSS design pattern. At last, in combination with the practical needs of teams, I used LESS to design a new responsive CSS framework, which called XSS (Extensible Style Sheets).

XSS consists of the service layer and application layer. In XSS, I recommend developer should separated the structure from the presentation, separated the component from the skin, separated the component from the container. In all, XSS is light and easy enough to use. XSS improves the standard of CSS development, the efficiency of development, and meet the strong requirements of UE.

**Key Words:** Mobile Internet      Responsive Web Design      CSS Framework

# 目 录

摘 要 .....	I
ABSTRACT .....	II
第 1 章 绪论 .....	1
1.1 课题设计的背景 .....	1
1.2 Web 前端相关的研究现状 .....	2
1.2.1 网页设计的研究现状 .....	2
1.2.2 Web 可访问性的研究现状 .....	3
1.2.3 Web 可用性的研究现状 .....	4
1.2.4 Web 性能的研究现状 .....	4
1.2.5 CSS 框架的研究现状 .....	5
1.3 课题设计的目的与内容 .....	6
1.4 本论文的写作思路 .....	6
第 2 章 课题研究关键技术 .....	8
2.1 CSS 概述 .....	8
2.1.1 CSS 基本语法 .....	8
2.1.2 CSS 选择器 .....	9
2.1.3 CSS 盒子模型 .....	10
2.2 HTML 概述 .....	10
2.2.1 HTML 基本语法 .....	11
2.2.2 HTML 标签语义化 .....	11
2.3 LESS 概述 .....	13
2.3.1 LESS “变量”概述 .....	13
2.3.2 LESS “混合”概述 .....	14
2.3.3 LESS “嵌套”概述 .....	14
2.3.4 LESS “函数与运算”概述 .....	15
2.4 Media Queries 概述 .....	16
2.4.1 媒体类型 .....	16
2.4.2 媒体查询 .....	17
第 3 章 00CSS 框架的研究与评价 .....	20

3.1	00CSS 框架的概述	20
3.2	00CSS 框架的设计原则	20
3.2.1	结构与皮肤分离	20
3.2.2	容器与内容分离	20
3.3	00CSS 框架的基本文件	20
3.4	00CSS 框架的对象模型	21
3.5	00CSS 框架的局限性	22
3.5.1	移动设备网页体验方面	22
3.5.2	样式污染风险方面	23
3.5.3	对象扩展机制方面	24
第 4 章	XSS 框架的设计	26
4.1	XSS 框架的基本介绍	26
4.2	XSS 框架的设计目标	26
4.3	XSS 框架的设计原则	27
4.3.1	结构与表现分离	27
4.3.2	组件与皮肤分离	28
4.3.3	组件与容器分离	28
4.4	XSS 框架的核心文件	29
4.5	XSS 框架的优势	30
4.5.1	服务层与应用层分离	30
4.5.2	增强了对对象扩展机制	30
4.5.3	兼容 HTML5	31
4.5.4	维护成本低	31
结 论		32
参 考 文 献		33
致 谢		34

# 第 1 章 绪 论

## 1.1 课题设计的背景

自 1989 年 Berners-Lee 提出 Web 的概念以来,网页技术发生了翻天覆地的变化,其用途由最初的纯学术交流,延伸至如今的搜索、邮箱、门户网站、电子商务网站、论坛、SNS、Wiki 等,涉及人们的工作、生活、学习和娱乐的方方面面。互联网世界有着成千上百的网页,负责承载和展示信息。近年来,移动互联网发展迅猛,开始重构人们的生活与思维,前端开发机遇与挑战并存。

在用户类型层面上:Web 是一个开放的平台,面向所有的用户群。Web 用户可能是残障人士,他们可能有视觉障碍,可能有听觉障碍,可能有肢体障碍,可能有认知和神经障碍;Web 用户还可能非残障人士,他们也许是使用移动手机,也许是使用 Web-TV 的用户,也许是在嘈杂环境下使用网站的用户,也许是第二语言访问的用户。

在网络环境层面上:目前,农村地区信息基础设施建设相对落后,宽带网络在农村入户率较低,网络接入条件落后于城镇,城乡之间“数字鸿沟”较大<sup>[1]</sup>。

在终端设备层面上:近几年内,移动设备快速崛起,移动互联网慢慢进入人们的生活,预计未来 5 年内移动设备的使用会超过桌面计算机。所以需要网站不仅要在桌面计算机大尺寸屏幕上可以为用户提供友好的 UI 和用户体验,同时在小尺寸屏幕上也应该可以提供一致的用户体验,使得用户可以在桌面大屏幕上和移动小屏幕上平滑的切换使用,同时没有任何的不适应感觉。

在前端技术层面上:虽然 Web 标准自 W3C 在 1994 年成立以来一直被致力推广,但被重视并普遍采用的时间至今却不超过 6 年,而且整个大环境对 Web 标准的理解还停留在概念层面,对 Web 前端开发规范和最佳实践方案还处于摸索阶段。不同的公司,不同的团队,不同的工程师,在不同的项目里,对 Web 前端开发最佳实践方案有着不同的定义。

在团队合作层面上:随着用户对使用体验的要求不断增加,对网页的表现力的要求也越来越高,从而导致实现代码越来越复杂,这无疑给团队合作带来了麻烦。页面越复杂,对团队合作的要求就越高。如果合

作不默契，很可能需要不停的打补丁，最后代码变得不可维护。

总之，从整个大环境来看，前端开发的工作变得越来越难。而实际工作中本人还面临着各种各样的其它问题。微信团队全球总人数不到 300 人，却搭建了一个承载着超过 3 亿用户量的平台，需求变更速度更是超乎想象。微信采用敏捷开发的方式，用最快的迭代速度不断追求卓越，大多数互联网产品，最多只能做到一天几个变更，微信却可以做到每天 20 个变更。微信前端团队还处在一个摸索的阶段，面对如此快速迭代的开发环境，开发者迫切需要一个精简、强悍的 CSS 框架，既能够方便团队及个人的开发，又能使开发出来的网站满足用户的体验要求。

## 1.2 Web 前端相关的研究现状

### 1.2.1 网页设计的研究现状

随着 3G 的普及，越来越多的人使用手机上网。移动设备已超过桌面设备，成为访问互联网的最常见终端。于是，让同样的网页在不同大小的设备上依然有良好体验成为前端开发工程师的挑战。

手机的屏幕比较小，宽度通常在 600 像素以下；PC 的屏幕宽度，一般都在 1000 像素以上（目前主流宽度是  $1366 \times 768$ ），有的还达到了 2000 像素。同样的内容，要在大小迥异的屏幕上，都呈现出满意的效果，并不是一件容易的事。

很多网站的解决方法，是为不同的设备提供不同的网页，比如专门提供一个 mobile 版本，或者 iPhone / iPad 版本。这样做固然保证了效果，但是比较麻烦，同时要维护好几个版本，而且如果一个网站有多个入口，会大大增加架构设计的复杂度。

2010 年，Ethan Marcotte 提出了自适应网页设计（Responsive Web Design）的概念<sup>[2]</sup>。自适应网页设计指可以自动识别屏幕宽度、并做出相应调整的网页设计。

目前，一些概念已经得到了实践，比如流体布局、弹性布局、帮助页面重新格式化的 Media Queries 等。但是响应式 Web 设计不仅仅是关于屏幕分辨率自适应以及自动缩放图片等等，它更像是一种对于设计的全新思维模式。



### 1.2.2 Web 可访问性的研究现状

在这个数字化的时代，上网成了人们日常生活的一部分。互联网用户除了健康人群外，还包括了残障人群。

视障用户包括色盲用户、完全失明用户。如果图片不带有相关文字描述，则视障用户在理解图片方面会存在问题。看不见图片的盲人用户就无法知道图片表达的是什么。色盲用户在识别设计元素方面也会存在问题，因为色盲用户所能识别的色彩不足以辨别所有的设计元素。

听障用户在听觉上存在问题。用声音传达的信息无法被听障用户所理解，简单解决方法是提供另外途径的信息传达方式，而不仅仅是声音，例如用文字描述、用图片。

肢体障碍用户经常无法使用鼠标，除非创建网站的导航和输入方式的需求中就考虑残障人士的需求，否则残障人士可能完全无法使用你的网站。

如果网站比较复杂，要想找到所想要的信息经常不太容易。如果网站设计的过于复杂、导航不一致、存在让人分心的重复性动画，情况会更加糟糕。这些设计元素会导致认知和神经有障碍的用户的使用问题，甚至会让这些用户完全无法使用网站。

确实，如果存在某方面残障，使用互联网是件困难的事情。网页内容无障碍依靠 Web 可访问性实践，对 Web 可访问性的研究越来越受到 Web 设计领域重视。然而，Web 可访问性访问不仅帮助到残障人士，良好理解和遵循 Web 可访问性设计，可以让所有用户都受益。更具体地说，可访问性意味着残疾人能感知、理解、浏览网站内容，并与之交互。可访问性也有益于其他人，包括由于衰老能力发生变化的老年人等。

处理好 Web 可访问性，网页搜索问题也可以改善。拥有良好可访问性的网页对搜索引擎更加透明，文档中结构化的信息使得网页容易被搜索引擎找到并评估，从而建立更加精确的索引，自然也可以得到更好的搜索结果。

在标准方面，自 2008 年推出 Web Content Accessibility Guidelines(WCAG)以来，W3C 对其不断优化调整。WCAG 介绍了如何使 Web 内容对于残疾人也具有可访问性<sup>[3]</sup>。本文档的主要目的是促进可访问性的发展，并且遵循这些指南可以使 Web 内容对所有人更加有用，无论使

用什么终端(例如:桌面浏览器、语音浏览器、移动电话、车载个人电脑等)或者在条件限制的情况下使用(例如:嘈杂的环境、过暗或过亮的房间、或者是免提情况等)。

### 1.2.3 Web 可用性的研究现状

1998 年, Jakob Nielsen, Donald Norman 和 Bruce Tognazzini 创立了 Nielsen Norman Group (NN/g), 这家公司专门提供以数据为基础的用户体验研究、培训和咨询。NN/g 根据 Web 在近些年关于眼动追踪技术在网站设计中应用, 结合了大量研究成果, 在 2011 年推出了用眼动追踪来提升网站可用性的实践指南。

眼动<sup>[4]</sup>实际上包括注视与眼跳两种最基本的运动。人们在看世界的时候自我感觉视线是连续的, 但从眼动记录当中可以明显看到, 事实上眼睛的活动是跳跃式的, 某些时候是短暂的停顿, 称之为“注视”, 某些时候是快速的移动, 称之为“眼跳”。在眼动结果图中会通过圆圈与线段来表示, 这样既可以看到注视又可以看到眼跳的称之为眼动轨迹图。

眼动追踪并不是科学研究中的一个崭新的方法, 认知心理学领域的科学家从 19 世纪起就开始研究眼动追踪, 以其发现眼睛的工作原理。到了 21 世纪, 眼动追踪最终发展成为一门实践技术, 不仅用于学术研究, 还应用在网页设计的研究中。通过眼动追踪研究准确记录人们使用网页时所注视的屏幕的位置, 应用这些数据, 就能够制定出提高网页设计可用性的指南<sup>[5]</sup>, 在这些指南的指导下设计网页将让用户使用更加方便。

### 1.2.4 Web 性能的研究现状

一次 Web 应用程序请求, 就是从浏览器发出一些参数到服务器, 然后服务器上的程序对请求进行处理, 再生成浏览器可以识别的内容, 包括 HTML、CSS、Javascript、图片、Flash 等, 最后由浏览器将这些内容展现给访问者。雅虎实验室的研究表明, 对于大多数网站来说, 只有不到 10%到 20%的响应时间是消耗在从 Web 服务器上下载 HTML 文档到浏览器中的。其余的 80%到 90%的时间都是消耗在页面组件的加载与呈现上<sup>[6]</sup>。

内容再丰富的网站, 如果慢到无法访问也是毫无意义的。SEO 做的

再好的网站，如果搜索蜘蛛抓取不到也是没用的。UE 设计的再人性化的网站，如果用户连看都看不到也是空谈。

在经过大量研究，反复实践与测试之后，雅虎实验室公布了 Web 性能优化的最佳实践方式，一直以来，这些方式指导着国内外前端开发工程师对 Web 性能的优化，更被誉为“黄金法则”。

### 1.2.5 CSS 框架的研究现状

近年来的 Web 开发中，“框架”是一个相当热门的话题。比如 JavaScript 框架 YUI、jQuery 和 Prototype 都引起广泛的关注，Web 应用框架 Rails and Dojo 更是引人注目。

框架就是一套包含工具、函数库、约定，以及尝试从常用任务中抽象出可以复用的通用模块，目标是使设计师和开发人员把重点放在任务项目所特有的方面，避免重复开发。随着开发过程的需要，CSS 框架也渐渐被重视起来，开发者都认识到：从具体的表现中抽出抽象的模块来重复使用，是减少用户下载、方便团队及个人开发最重要的手段。简单地说，CSS 框架就是预先准备好的库，旨在帮助开发者使用 CSS 实现更简单，更符合标准的样式的 web 页面。

目前业界主流的 CSS 框架有 normalize、G5、Bootstrap、Golden-Grid-System、Semantic GRID SYSTEM 等。图 1-1 显示了 Google Trends 上关于 CSS Framework 的数据曲线，可以看到 CSS 框架越来越受到大家的重视。

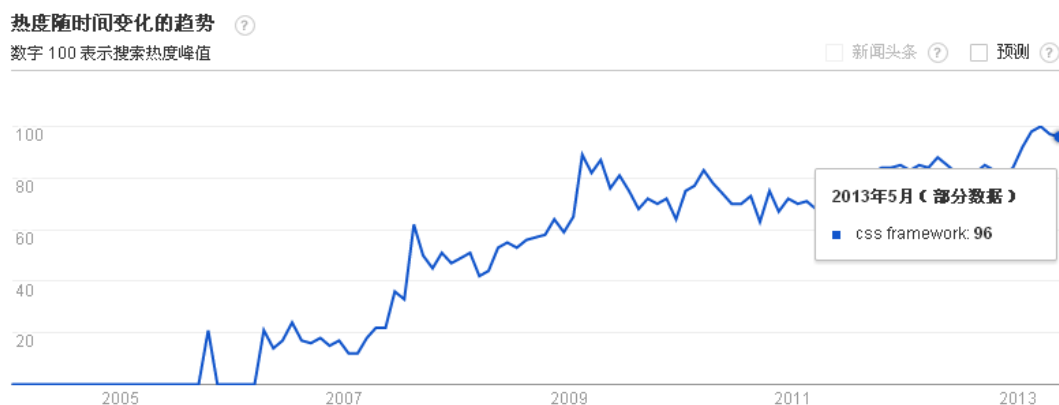


图 1-1 Google Trends 上的 CSS Framework 数据曲线图

### 1.3 课题设计的目的与内容

CSS 语言与 HTML 语言一样，都是一种弱类型的描述语言，CSS 代码被下载到客户端后通过本地浏览器解析，并把 CSS 样式的效果呈现出来。开发 CSS 样式的时候，代码排错和维护一直是件令人烦恼的事情，根本原因就在于 CSS 语言本身的不严谨性，而浏览器对于 CSS 又有排错能力，只要发现 CSS 存在语法错误，就会忽略不计，而不影响浏览器的正常工作。这种包容性无形中又增加了开发者的维护难度。另外，一个网站或项目中会存在大量的 CSS 重复代码，如何发挥 CSS 自身的特性，避免这种代码的重复，也是开发者应该思考的问题。

另一方面，对于 Web 的内容，设备厂商、用户以及 Web 内容的作者都有着不同的需求和期望。Web 程序和设备厂商自然是希望能给用户提供一个特殊的功能组合体，但是往往很少 Web 内容的作者会单独为他们的产品提供内容。用户则更加希望能够在有相似功能的不同设备访问相同的内容，即使那些设备功能有所差别，用户还是希望能够访问到适合的版本。而 Web 程序的开发者也不想为不同的版本开发多份代码，他们希望的是“一次设计，普遍适用”，尽可能地减少 Web 程序的建设与维护成本。

本人尝试提出一个统一的响应式 CSS 基础框架的设想并进行研究与设计。课题的研究内容涉及 Web 标准、网页响应式设计、网页可访问性、网页可用性、面向对象软件思想、LESS 动态样式语言和浏览器工作原理等。

### 1.4 本论文的写作思路

第 1 章 绪论：阐述课题设计的背景、Web 前端开发相关的研究现状、课题设计的目的与内容以及本论文的写作思路。

第 2 章 课题设计关键技术介绍，包括 CSS（层叠样式表）、HTML（超文本标记语言）、Media Queries（媒体查询）以及 LESS（动态样式表语言）。

第 3 章 分析总结 00CSS 框架，说明现有解决方案仍存在不足之处。

第 4 章 构想一个统一的响应式 CSS 基础框架，在国内外权威研究基础

上，并通过与微信 UI 团队负责人多次交流，研究并实现了框架的设计。  
结论 对本课题的研究做出总结。

## 第 2 章 课题关键技术介绍

### 2.1 CSS 概述

层叠样式表（Cascading Style Sheets）简写成 CSS，是一种用来描述 HTML、XML（包括各种 XML 语言如 SVG、XHTML）文档表现的样式表语言。CSS 在屏幕、纸、音频或者其它媒体类型中描述了结构化元素怎样被呈现。它是开放 Web 的核心语言之一，有一个标准化的 W3C 规范。CSS 现在共有 4 个版本，CSS1 现在已经废弃，CSS2.1 作为推荐标准。CSS3 现在分成很多较小的模块，正处于标准化的进程中。CSS4 模块的第一份早期草案正在编写和审核中。

#### 2.1.1 CSS 基本语法

CSS 规则由两个主要的部分构成：选择器以及一条或多条声明，如下行代码所示：

```
selector {declaration1; declaration2; ... declarationN }
```

每条声明由一个属性和一个值组成。属性是希望被设置的样式属性，每个属性有一个值，属性和值间用冒号分开。

```
selector {property: value}
```

例如：下面这行代码的作用是将 h1 元素内的文字颜色定义为红色，同时将字体大小设置为 14 像素。

```
h1 {color:red; font-size:14px;}
```

图 2-1 展示了上面这段代码的语法结构：

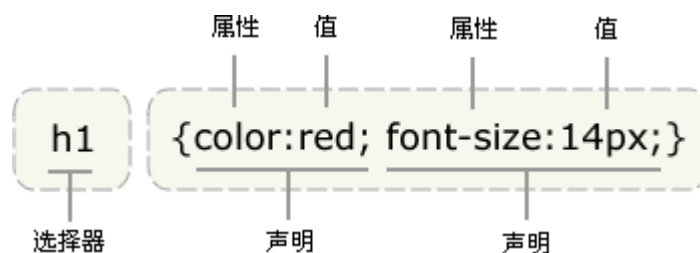


图 2-1 CSS 基本语法示例图

## 2.1.2 CSS 选择器

选择器是 CSS 中很重要的概念,所有 HTML 语言中的标记样式都是通过不同的 CSS 选择器进行控制的。用户只需通过选择器对不同的 HTML 标签进行选择,并赋予各种样式声明,就可以实现各种效果,本质上就是一种“内容”与“表现形式”的对应关系。<sup>[7]</sup>

在 CSS 中,有几种不同类型的选择器,基本选择器有标记选择器、class 选择器和 ID 选择器;通过对基本选择器的重新组合,还可以产生更多种类的选择器,实现更强、更方便的选择功能,复合选择器由两个或两个以上的基本选择器通过不同的连接方式构成,可分为“交集”选择器、“并集”选择器和后代选择器。CSS2 中又引入属性选择器、伪类选择器和伪元素选择器,CSS3 中更引入了大量新的选择器。

当写 CSS 的时候必须注意有些选择器在级联上会高于其它选择器,也就是说,写在最后面的选择器不一定会覆盖前面写在同一个元素的样式。

把特殊性分为 4 个等级<sup>[8]</sup>,每个等级代表一类选择器,每个等级的值为其所代表的选择器的个数乘以这一等级的权值,最后把所有等级的值相加得出选择器的特殊值。(见图 2-2)

4 个等级的定义如下:

第 1 等级:代表内联样式,如: `style=""`,权值为 1000。

第 2 等级:代表 ID 选择器,如: `#content`,权值为 100。

第 3 等级:代表类,伪类和属性选择器,如 `.content`,权值为 10。

第 4 等级:代表类型选择器和伪元素选择器,如 `div p`,权值为 1。

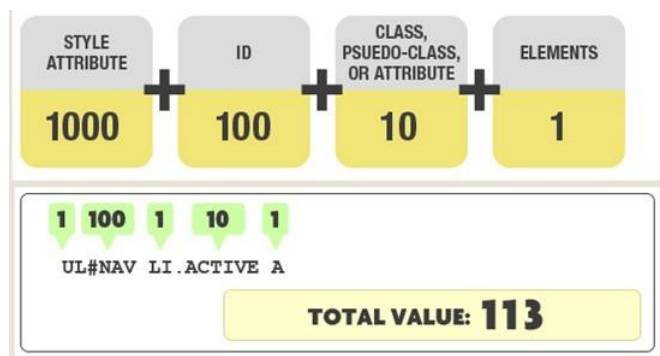


图 2-2 CSS 选择器权重计算示例图

### 2.1.3 CSS 盒子模型

本质上，CSS 中的每个元素都被一个盒子所包围（见图 2-3）。这个盒子规定了元素框的处理，其中由中心向外扩展包括盒子的尺寸（宽和高：width 和 height），内边距（padding）、边框（border）和外边距（margin）。<sup>[9]</sup>

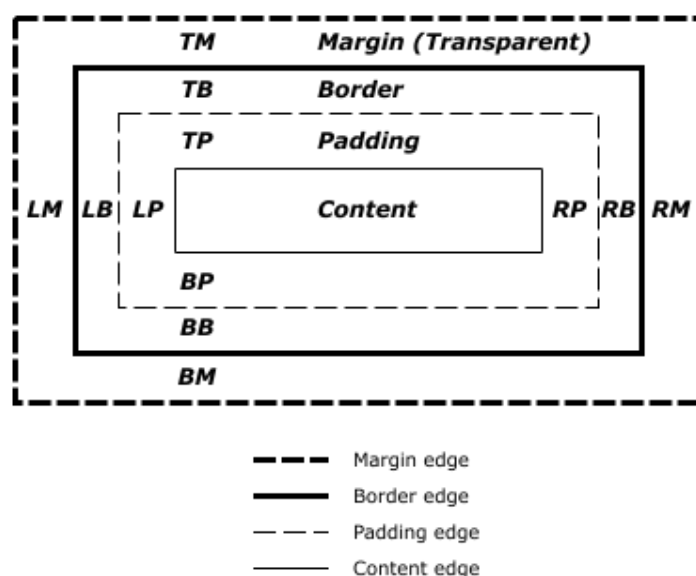


图 2-3 CSS 盒子模型示例图

如图 2-3 所示，盒子模型中由 width 和 height 规定的区域为内容所能使用的区域；接着，直接包围内容的是内边距（padding），内边距区域内可以显示盒子元素的背景颜色或者背景图片；与内边距接壤的边缘边框（border）；边框以外是外边距（margin），外边距默认是透明的，即盒子的背景颜色或者背景图片在外边距不可见，但是其父元素的内容可透出来，是可视的。

## 2.2 HTML 概述

超文本标记语言（HyperText Markup Language）简称 HTML，是网页的核心语言，它描述了 Web 文档结构和语义。开发者可以用<img>、



<title>、<p>、<div>等 HTML 元素对网页进行标记。不同于 CSS 的是，HTML 规范是由 W3C 和 WHATWG 一起制定并推行的，随着移动互联网的发展，HTML5 应运而生。但 HTML5 仍处于开发阶段，目前推行的是 W3C 于 1999 年 12 月 24 日发布的 HTML 4.01。

### 2.2.1 HTML 基本语法

HTML 文档是由 HTML 元素定义的。HTML 元素指的是从开始标签(start tag)到结束标签(end tag)的所有代码。HTML 元素的基本语法包括：1、HTML 元素以开始标签起始；2、HTML 元素以结束标签终止；3、元素的内容是开始标签与结束标签之间的内容；4、某些 HTML 元素具有空内容(empty content)；5、空元素在开始标签中进行关闭（以开始标签的结束而结束）；6、大多数 HTML 元素可拥有属性。

下面代码中的<p>元素定义了 HTML 文档中的一个段落。这个元素拥有一个开始标签<p>，以及一个结束标签</p>。元素内容是：“This is my first paragraph.”

```
<p>This is my first paragraph.</p>
```

### 2.2.2 HTML 标签语义化

标签语义化是指用合理 HTML 标记以及其特有的属性去格式化文档内容。语义化的(X)HTML 文档有助于提升网站的可访问性。对于搜索引擎或者爬虫软件来说，则有助于它们建立索引，并可能给予一个较高的权值，图 2-4 中列举了 W3C 定义的部分语义化 HTML 标签。

标签	原单词	说明	语义化(Y/N)
h1-h6	head	定义 HTML 标题	Y
p	paragraph	定义段落	Y
ul	unordered list	定义无序列表	Y
ol	ordered list	定义有序列表	Y
li	list item	定义列表的项目	Y
dl	definition list	定义定义列表	Y
dt	definition term	定义定义列表中的项目	Y
dd	definition description	定义定义列表中项目的描述	Y
table	table	定义表格	Y
thead	table head	定义表格中的表头内容	Y
tbody	table body	定义表格中的主体内容	Y
th	table head cell	定义表格中的表头单元格	Y
tr	table row	定义表格中的行	Y
td	table data cell	定义表格中的单元	Y
a	anchor	定义锚	Y
img	image	定义图像	Y
div	division	定义文档中的节	N
span	span	定义文档中的节	N

图 2-4 HTML 基本标签列表

以上只列出了常用的一些标签，基本所有的 HTML 标签都是一个单词或者词组的缩写，这样本意是更便于开发者对语义化的理解。在所有的 HTML 标签中，除了<div>和<span>这二个无语义的标签，其他标签都有它存在的意义，只有知道有哪些标签，以及对各个标签的本意有详细的了解才能知道如何去使用它。

如果选用的标签几乎全是不带语义的，那么在去样式后网页中几乎看不到任何结构信息，可读性非常差；如果选用的都是语义适合的标签，去样式后网页依然具有非常好的可读性；各个浏览器有自己的默认样式，默认的风格给予了各个标签不同的显示，标签使用的正确与否能体现网站的可用性，这也是检验一个网站可用性的最简单的方法之一。

另一方面，使用语义标签可以确保各种设备以一种有意义的方式来渲染网页。理想情况下，观看设备的任务是符合设备本身的条件来渲染网页。例如：一部手机可以选择使一段标记了标题的文字以粗体显示，而掌上电脑可能会以更大的字体来显示，但本质上都是一种标题的强调性显示，所以无论哪种方式，一旦你对文本标记为标题，就可以确信读取设备将根据其自身的条件来合适地显示页面。

## 2.3 LESS 概述

LESS 是一种动态的样式语言。LESS 扩展了 CSS 的动态行为，比如说，设置变量(Variables)、混合书写模式(mixins)、运算(operations)和函数(functions)等，而且 LESS 使用了现有的 CSS 语法。LESS 的基本语法如下：

### 2.3.1 LESS “变量” 概述

变量允许单独定义一系列通用的样式，然后在需要的时候去调用。所以在做全局样式调整的时候只需要修改少量代码即可。

```
// LESS                                     /* 生成的 CSS */

@color: #4D926F;                             #header {
                                              color: #4D926F;
#header {                                     }
    color: @color;                             h2 {
                                              color: #4D926F;
}                                              }
h2 {                                          }
    color: @color;
}
```

上面的代码中，在 LESS 部分，定义了@color 变量，然后在表示标题的#header 和 h2 这两个声明中使用，如果体验需要更改网页中标题的颜色，可以简单地改变@color 的值即可。

### 2.3.2 LESS “混合”概述

混合可以将一个定义好的 class A 轻松的引入到另一个 class B 中，从而简单实现 class B 继承 class A 中的所有属性。还可以带参数地调用，就像使用函数一样。

```
// LESS                                /* 生成的 CSS */

.rounded-corners (@radius: 5px) {      #header {
    border-radius: @radius;              border-radius: 5px;
    -webkit-border-radius: @radius;      -webkit-border-radius: 5px;
    -moz-border-radius: @radius;          -moz-border-radius: 5px;
}                                          }

#header {                                #footer {
    .rounded-corners;                    border-radius: 10px;
                                          -webkit-border-radius: 10px;
                                          -moz-border-radius: 10px;
}                                          }
#footer {                                }
    .rounded-corners(10px);
}
```

以上代码 LESS 部分定义了一个圆角的函数 `.rounded-corners`，并混合在 `#header`、`#footer` 这两个容器中，这样一来，`#header` 和 `#footer` 就具备了 `.rounded-corners` 所有特性。

### 2.3.3 LESS “嵌套”概述

可以在一个选择器中嵌套另一个选择器来实现继承，这样很大程度减少了代码量，并且代码看起来更加的清晰，如以下示例代码：

```
// LESS                                     /* 生成的 CSS */

#header {                                  #header h1 {
  h1 {                                    font-size: 26px;
    font-size: 26px;                    font-weight: bold;
    font-weight: bold;                  }
  }                                     #header p {
  p { font-size: 12px;                  font-size: 12px;
    a { text-decoration: none;          }
      &:hover { border-width: #header p a {
1px }                                text-decoration: none;
    }                                  }
  }                                     #header p a:hover {
}                                       border-width: 1px;
                                     }
                                     }
```

### 2.3.4 LESS “函数与运算”概述

运算提供了加、减、乘、除操作；可以做属性值和颜色的运算，这样就可以实现属性值之间的复杂关系，例如：

```
// LESS                                     /* 生成的 CSS */

@the-border: 1px;                            #header {
@base-color: #111;                            color: #333;
@red: #842210;                                border-left: 1px;
                                                border-right: 2px;
#header {                                    }
    color: @base-color * 3;                    #footer {
    border-left: @the-border;                    color: #114411;
    border-right: @the-border * 2;                border-color: #7d2717;
}                                                }
#footer {
    color: @base-color + #003300;
    border-color: desaturate(@red,
10%);
}
```

## 2.4 Media Queries 概述

### 2.4.1 媒体类型

媒体类型(Media Type)在 CSS2 中是一个常见的属性,也是一个非常实用的属性,可以通过媒体类型对不同的设备指定不同的样式,在 CSS2 中我们常碰到的就是 all(全部),screen(屏幕),print(页面打印或打印预览模式),其实媒体类型不止这三种,W3C 定义完整的媒体类型如图 2-5 所示。

类型	解释
all	所有设备
braille	盲文
embossed	盲文打印
handheld	手持设备
print	文档打印或打印预览模式
projection	项目演示，比如幻灯
screen	彩色电脑屏幕
speech	演讲
tty	固定字母间距的网格的媒体，比如电传打字机
tv	电视

图 2-5 媒体类型列表

## 2.4.2 媒体查询

媒体查询（Media Queries）是 CSS3 中引入的新概念，借助可以查询设备的屏幕尺寸颜色等信息，就可以根据不同的设备来写 CSS，让网页在不同设备上有更好的用户体验。

Media Queries 的基本语法如下：

```
// Media Queries 基本语法
```

```
@media screen and (min-width:1024px) and (max-width:1280px){  
body{padding:1em;}  
}
```

screen 是上述媒体类型里的一种；and 是关键字，其他关键字还包括 not(排除某种设备)和 only(限定某种设备)；(min-width:1024px) 和 (max-width:1280px) 中 max-width 是媒体固有宽度的特性。

媒体特性共 13 种（如图 2-6 所示），可以说是一个类似 CSS 属性的集合。但和 CSS 属性不同的是，媒体特性只接受单个的逻辑表达式作为其值，或者没有值。并且其中的大部分接受 min/max 的前缀，用来表示“大于等于/小于等于”的逻辑，以此避免使用“<”和“>”这些字

符。

Media features	Value	Applies to	Accepts min/max
<b>width</b>	length	visual and tactile media types	yes
<b>height</b>	length	visual and tactile media types	yes
<b>device-width</b>	length	visual and tactile media types	yes
<b>device-height</b>	length	visual and tactile media types	yes
<b>orientation</b>	portrait   landscape	bitmap media types	no
<b>aspect-ratio</b>	ratio	bitmap media types	yes
<b>device-aspect-ratio</b>	ratio	bitmap media types	yes
<b>color</b>	integer	visual media types	yes
<b>color-index</b>	integer	visual media types	yes
<b>monochrome</b>	integer	visual media types	yes
<b>resolution</b>	resolution	bitmap media types	yes
<b>scan</b>	progressive   interlace	"tv" media types	no
<b>grid</b>	integer	visual and tactile media types	no

图 2-6 媒体特性列表

页面中引入媒体类型方法也有多种：包括 link 方式、xml 方式、@import 方式和@media 方式。

### 1. 通过 link 方式插入

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

### 2. xml 方式插入

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

### 3. @import 方式插入

@import 引入有两种方式，一种是在样式文件中通过@import 调用另一个样式文件；另一种方法是在<head></head>中的<style></style>中引入，如：



// 样式文件中调用

```
@import url("css/reset.css") screen;
```

```
@import url("css/print.css") print;
```

// <style>中引入

```
<head>
```

```
<style type="text/css">
```

```
@import url("css/style.css") all;
```

```
</style>
```

```
</head>
```

#### 4. @media 方式插入

```
@media screen{
```

```
  选择器{
```

```
    属性：属性值；
```

```
  }
```

```
}
```

## 第 3 章 00CSS 框架的研究与评价

### 3.1 00CSS 框架的概述

“面向对象的编程”的概念在软件开发领域普遍存在，它已经成为任何现代编程语言的一种基本形式，数据的抽象化、模块化和继承等特点在编写代码中得到了大规模的应用<sup>[10]</sup>。面向对象的 CSS 是一种容易重用的一种 CSS 规则，也是 OOP 的概念，从而降低了页面的加载时间，提高了网面的性能。00CSS 是 Nicole Sullivan 在 2009 年提出来的一套 CSS 框架。

### 3.2 00CSS 框架的设计原则

#### 3.2.1 结构与皮肤分离

几乎页面上的每个元素都有不同的表现，比如背景、边框、字体等效果，这些视觉效果在不同的环境中重复地被使用。把这些视觉效果被抽象成一个皮肤模块，它们就变得可复用，可以运用到任何的元素上，并且具有相同的效果。

#### 3.2.2 容器与内容分离

把容器和内容独立出来，这样的好处是，同一个内容可以放在不同容器中，不同容器也可以容纳不同的内容，有效地增加了网页设计的灵活性。

### 3.3 00CSS 框架的基本文件

00CSS 包含 4 个基本文件和一些扩展样式，图 3-1 显示了 00CSS 框架的核心文件在 Chrome 的加载情况。










Elements	Resources	Network	Sources	Timeline	Profiles	Audits	Console				
Name	Path	Met...	Status	Type	Initiator	Size	Time	Timeline	433 ms	649 ms	866 ms
			Text			Conten	Latenc				
	exercise4.html	GET	304	text/...	Other	188 B	407 ms				
	/velocity		Not M...			3.9 KB	403 ms				
	libraries.css	GET	304	text/...	exercise4.h...	164 B	196 ms				
	/css		Not M...		Parser	1.1 KB	193 ms				
	template.css	GET	304	text/...	exercise4.h...	165 B	215 ms				
	/css		Not M...		Parser	815 B	215 ms				
	content.css	GET	304	text/...	exercise4.h...	164 B	389 ms				
	/css		Not M...		Parser	3.9 KB	388 ms				
	mod.css	GET	304	text/...	exercise4.h...	165 B	396 ms				
	/css		Not M...		Parser	1.5 KB	396 ms				
	mod_skins_ex4.css	GET	304	text/...	exercise4.h...	166 B	406 ms				
	/css		Not M...		Parser	7.1 KB	405 ms				
	ugly.css	GET	304	text/...	exercise4.h...	165 B	418 ms				
	/velocity		Not M...		Parser	553 B	416 ms				
	header.jpg	GET	304	imag...	exercise4.h...	166 B	428 ms				
	/img		Not M...		Parser	61.6 KB	427 ms				
	grids.css	GET	304	text/...	exercise4.h...	164 B	370 ms				
	/css		Not M...		Parser	913 B	358 ms				

图 3-1 00CSS 框架核心文件

这 4 个基本文件包括 `libraries.css`、`template.css`、`grids.css` 和 `content.css` 等。其中：

1. `libraries.css` 用来定义 Reset 和文字的基本设置。
2. `template.css` 用来定义头部、底部以及中部的布局，00CSS 最多支持三栏布局，左右定宽之后中间会自适应。
3. `grids.css` 用来定义小范围内的浮动布局，所有的浮动条目都用百分比来定宽，并且最后一个条目 `.lastunit` 不设定宽度，这样前面的条目过宽或者过窄的话也不会把最后一条挤下来。
4. `content.css` 是对内容和定义的样式。它对所有标题和 P 都加上 `padding:10px`。

此外还有扩展文件。在 00CSS 的官方示例中，除了载入上述文件以外，还载入了两个特定的样式文件：`mod.css` 和 `mod_skins.css`，这两个样式根据特定项目可以自行编写的组件 CSS。

### 3.4 00CSS 框架的对象模型

00CSS 使用 Java 和 PHP 语言中对象的概念来描述面向对象的理论，

尽管它们的复杂度不一样。OOCSS 中的对象由 4 部分组成。

第一部分是一个或者多个表示 DOM 节点的 HTML 元素；

第二部分是作为包裹节点类名开始的所有 CSS 声明；

第三部分是用于展示背景图片和拼合图片的组件；

第四部分是这个对象相关的 Javascript 交互，媒体或者与之相关的其它方法。

下面代码表示 OOCSS 所描述的对象原型：

```
<div class="mod">
  <div class="inner">
    <div class="hd"> Mod Head</div>
    <div class="bd"> Mod Body</div>
    <div class="ft"> Mod Foot</div>
  </div>
</div>
```

这个模块化的对象包含四部分节点属性，这些节点属性不能脱离模块独立存在，其中 inner 和 body (bd) 是必须的区域，head (hd) 和 foot (ft) 可选的区域。

## 3.5 OOCSS 框架的局限性

### 3.5.1 移动设备网页体验方面

从 OOCSS 的 grids.css 中可以看出所有容器都是用百分比定义宽度的。这种流体式布局使得网页整体宽度会适配整个屏幕，而页面的元素也将进行等比缩放。图 3-2 给出 OOCSS 官方 Sample 在 PC、mobile 中的显示情况：

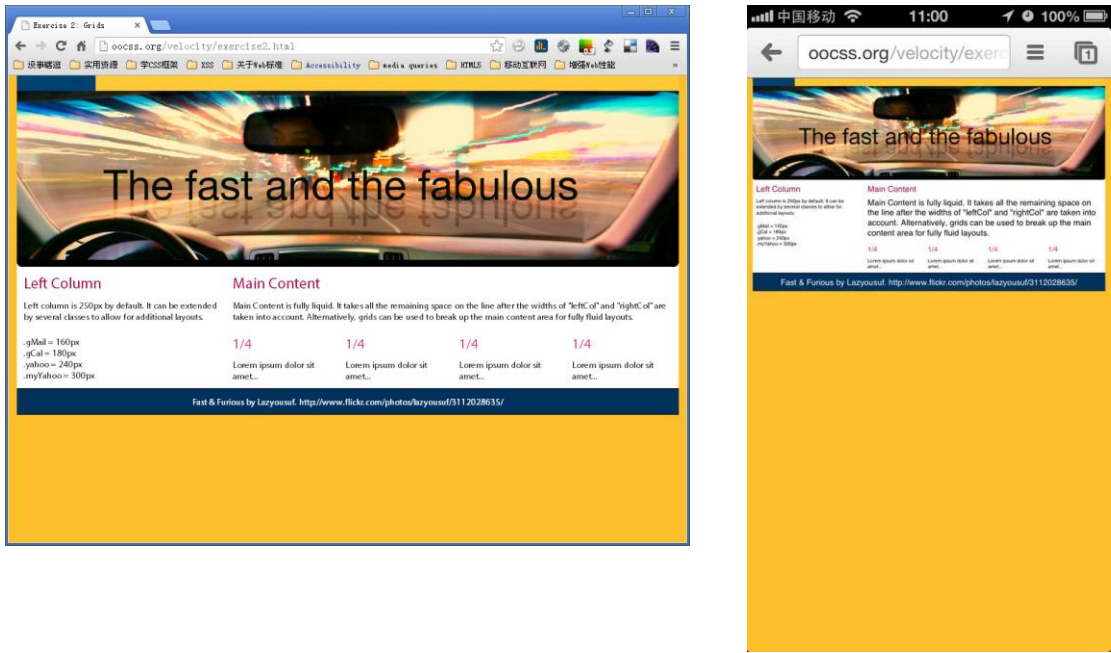


图 3-2 00CSS 框架官方示例网页在 PC（左）和 iPhone（右）上的显示情况

可以看到，在手机设备屏幕等较小的屏幕中，整个网页只是被缩小了，用户不能得到良好的体验。

### 3.5.2 样式污染风险方面

00CSS 框架不能避免样式污染的风险。图 3-3 中的容器，用 00CSS 实现起来，核心代码如下：

```
// HTML 部分
<div class="mod">
  <h2>...</h2>
  <p class="content">...</p>
</div>

// CSS 部分
.mod{...}
.mod .content{...}
```

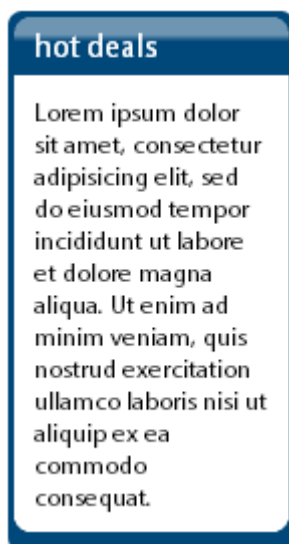


图 3-3 00CSS 框架 Demo 示例图

这样的代码看似没问题，但假如这个容器里面还包含了一个小容器，而这个小容器里面恰好就有一个`.content`，这个时候小容器的`.content`也会匹配到上述的代码，但实际上不希望它匹配到大容器的规则，这种情况就往往造成了样式污染。

### 3.5.3 对象扩展机制方面

为了让基础对象可重用，在不同的地方将其放入不同的容器中，从而通过后代选择器的使用，让基础对象达到可重用的作用。虽然这样表面上达到了重用性，但对于 00CSS 来说并不是最理想的方法。在 00CSS 中，对象扩展的机制是将基础对象进行扩展，根据上下文语境，通过更多的类名，让基础对象在不同位置实现不同的效果，从而实现了基础对象的重用性。简单说就是通过对基础对象扩展类名，从而达到基础对象的可重用性。

比如一个 160px 的左列，而非默认值，可以再列上增加额外的 `class`。

```
// HTML 部分
```

```
<div class="leftCol gMail">
```

```
...
```

```
</div>
```

```
// CSS 部分
```

```
.leftCol{float:left;width:250px; }
```

```
.gMail{width:160px;}
```

但是这样的代码却是很脆弱的，因为很多时候扩展类与基类是有重写关系的，所以它们在 CSS 文件中的位置关系必须受到约束。假如把 `.gMail` 和 `.leftCol` 位置颠倒了，显示出来的就不是预想中的效果了。事实上开发者在使用框架的时候只需关心基类是什么，如何对它进行扩展而已。

## 第 4 章 XSS 框架的设计

### 4.1 XSS 框架的基本介绍

为了能更简单的开发、调试和部署，适应需求增长和变化，项目易维护和代码高性能，本人设计了一个响应式的、轻量级 CSS 框架，取名 XSS (Extensible Style Sheets)。

这个框架是 00CSS 的扩展，很多方面 00CSS 已经做得够好了，但是如上文所言，00CSS 在实际应用中仍存在不足之处，所以有必要对 00CSS 进行延伸设计，XSS 提出结构与表现分离、组件与皮肤分离、组件与容器分离三大设计原则，旨在建立一套可预见的、可重用的、可维护的和可扩展的 CSS 设计模式。

特别地，XSS 还具备响应式布局的特性。不管是手机屏幕还是 PC 屏幕还是 Pad 屏幕，不管分辨率是 1280 x 800、1024 x 768、800 x 480 还是 640 x 960，同样的内容，在这些不同的屏幕上都能呈现出满意的效果。

XSS 不单是 CSS 文件组合，此框架是基于动态样式表语言 LESS 进行开发的，基于 LESS 的开发，可以对组件进行更好的抽象，更好的封装与扩展，也可以增大代码的可复用性，减少开发过程中的代码冗余。对于有复杂交互逻辑的组件，如果使用 LESS 进行开发会变得更加高效，同时也降低了开发者间的沟通成本。

### 4.2 XSS 框架的设计目标

在使用 XSS 的项目中，CSS 是可预见、可重用、可维护和可扩展且是响应式的。

#### 1. 可预见

可预见性的 CSS 就是说定义的规则的最终表现正如开发者原先所设想的，当添加或更新一条规则，不应该影响网站上不想要受影响的部分。对于一个小型网站，因为很少需要修改，所以并不是很重要。但是对于一个有着几十或几百个页面的大型网站，可预见性的 CSS 是非常必要的。



## 2. 可复用

CSS 规范是足够抽象的和耦合的，这样可以根据现有代码部分很快创建出新的组件，而不需要重新编写已经处理过的样式和问题。

## 3. 可维护

当网站需要添加、更新或重新安排一些新的组件和特性，这样做不会重构现有的 CSS。给页面添加 X 组件不会破坏已经存在的组件 Y。

## 4. 可扩展

随着网站的规模和复杂程度的增长，往往需要更多的开发人员来维护。可扩展的 CSS 意味着可以轻松地由一个人或一个大型的技术团队管理这个网站。也意味着该网站的 CSS 架构容易掌握不需要很陡的学习曲线。

## 5. 响应式

移动设备已超过桌面设备，成为访问互联网的最常见终端。可响应的 CSS 意味着同一个网站，在不同的终端，不管是 PC 还是 Mobile 等，都能自动调整布局，让用户获得良好体验。

# 4.3 XSS 框架的设计原则

XSS 框架的设计原则包括：结构与表现分离、组件与皮肤分离和组件与容器分离。

## 4.3.1 结构与表现分离

结构与表现分离是语义化 HTML 的开始，结构指的是 HTML 标签，HTML 本身是没有任何样式的，但是每个浏览器都会有默认样式，尽管这样做的目的也是为了更好的表达 HTML 的语义，但是开发者不应该依靠这些默认的样式去描述网页的内容。比如图 4-1 的 Demo 中的标题，开发者可以使用表达标题含义的标签，如 h1、h2、h3、h4、h5、h6。但这并不是因为这些标签默认的加粗效果而用，仅仅是因为它们的本义就是标题而已，所以如果有需要的话还必须在 CSS 里面显式声明该标签是加粗显示的。



图 4-1 XSS 框架 Demo 示例图

### 4.3.2 组件与皮肤分离

网页设计过程中经常看到这样的场景：网页上的某个元素的视觉特点在不同的地方有所差异，或者某个元素的视觉特点被重复用在其它元素上。当这些视觉特点抽象成类作为基础的模块时，它们就成为可重复使用，可用于任何元素具有相同的基本结果。例如，图 4-1 的 Demo 中，本人抽象出三套不同的皮肤基类，然后可以在更多的元素上应用这些皮肤而避免重复定义出现代码冗余的情况。示例代码如下：

// 第一张卡片	// 第二张卡片	// 第三张卡片
<code>&lt;div class="card skin1"&gt;</code>	<code>&lt;div class=" card skin2"&gt;</code>	<code>&lt;div class="card skin2"&gt;</code>
<code>&lt;h2&gt;...&lt;/h2&gt;</code>	<code>&lt;h2&gt;...&lt;/h2&gt;</code>	<code>&lt;h2&gt;...&lt;/h2&gt;</code>
<code>&lt;/div&gt;</code>	<code>&lt;/div&gt;</code>	<code>&lt;/div&gt;</code>

### 4.3.3 组件与容器分离

简单地说，就是尽量不要依赖后代选择器达到改变组件的样式。如果从软件工程的角度来讲，也就是“开-闭”原则（Open-Closed principle, OCP）：一个软件实体应当对扩展开放，对修改关闭。也就是说在设计一个模块的时候，应当使这个模块可以在不被修改的前提下被扩展。

例如，图 4-1 的 Demo 可以抽象成如下模型：

// 第一张卡片	// 第二张卡片	// 第三张卡片
<code>&lt;div class="card1"&gt;</code>	<code>&lt;div class=" card2"&gt;</code>	<code>&lt;div class=" card3"&gt;</code>
<code>  &lt;h2&gt;...&lt;/h2&gt;</code>	<code>  &lt;h2&gt;...&lt;/h2&gt;</code>	<code>  &lt;h2&gt;...&lt;/h2&gt;</code>
<code>&lt;/div&gt;</code>	<code>&lt;/div&gt;</code>	<code>&lt;/div&gt;</code>

假如其中的标题 h2 就是一个组件，可以看到 h2 共有两种不同的颜色。所以开发者就可以把 h2 抽象出来作为一个基类，然后使用不同的扩展类例如 `.card1_title`、`.card2_title` 对它进行扩展。但是不推荐通过 `.card1 h2`，`.card2 h2` 这样的方式对 h2 进行修改。

首先，这种方式下当 h2 出现在没有 `.card1` 或 `.card2` 或 `.card3` 的容器中时，它的效果是不可预见的。另外，它也是不可重用的，假如在一个新的 `.card4` 的容器中需要一个像 `.card1` 里面的 h2 一样的组件，这时候就必须多写一个关系 `.card4` 的声明，这将造成代码冗余。另外，它也比较难维护。一旦这个 h2 需要重新设计，那么开发者不得不修改其他几个 CSS 样式，这样一来也会影响开发效率，如果处理不当的话还会造成样式错乱。

在实际应用中，这三个原则只作为前端团队开发的规范指南，不要求开发者一开始就严格按照 XSS 设计原则进行 CSS 开发。理由是，无论用什么语言在编写程序，都要做好重构的准备。所谓重构，就是在不改变代码外在行为的前提下，对代码做出修改，以改进程序的内部结构。本质上说，重构就是在代码写好之后改进它的设计<sup>[11]</sup>。按照目前对软件开发的理解，本应该是先设计而后编码：首先得有一个良好的设计，然后才能开始编码。但是在实际开发中，随着时间流逝，我们不断修改以前的代码，于是根据原先设计所得的系统，整体结构逐渐衰弱。代码质量慢慢沉沦，编码工作从严谨的工程堕落成胡砍乱劈的随性行为。正确的做法是先做，然后优化，不断地进行代码重构，最终才能得出一个真正结构严谨的程序。所以只要保证在进行 CSS 开发的时候能为接下来的代码重构工作留下扩展空间即可。

## 4.4 XSS 框架的核心文件

XSS 框架的核心文件包括 `fn.less`、`libraries.less` 和 `layout.less`

等，其中：

1. `fn.less` 文件编译后不产生任何样式，它只是作为一个函数库，里面的函数可以供其它 `less` 文件随意调用。

2. `libraries.less` 文件包含了对浏览器默认表现的重置样式、修复 HTML 元素可用性的样式以及所有项目通用的原子类样式。

3. `layout.less` 文件包含了页面整体布局以及较小范围的栅格化布局。和 `00CSS` 一样，`XSS` 最多支持三栏布局，左右定宽之后中间会自适应。

## 4.5 XSS 框架的优势

### 4.5.1 服务层与应用层分离

在 `XSS` 里，服务层指的是 `fn.less`，应用层由公共样式库 `libraries.less`、布局 `layout.less`、通用组件库 `mod.less` 和业务专用样式库 `demo.less` 组成。服务层定义了一些变量与函数，它们都是允许并推荐跨项目调用的，这样一来所有项目的代码就都只存在一个统一的服务型接口。应用层定义了一套统一的解决浏览器兼容问题的方案，以及一些通用的组件。把服务与应用分离使得代码具有更好的规范性、可维护性和更高的重用率，而且在项目开发过程中，也可以同时对服务层与应用层进行开发，加大开发效率。

### 4.5.2 增强了对对象扩展机制

在 `XSS` 中，利用 `CSS` 选择器权重的不同，有效地消除了 `00CSS` 框架中对对象扩展机制存在的问题。

```
// HTML 部分                                     // LESS 部分

<div class="leftCol gMail">                        .leftCol{
    ...                                              float:left;width:250px;
</div>                                              &.gMail{width:160px;}
                                                    }

                                                    // 由 LESS 生成的 CSS

                                                    .leftCol{float:left;width:250px; }
                                                    .leftCol.gMail{width:160px;}
```

上面示例代码中，XSS 在 `.leftCol` 的基础上，加上了 `.gMail`，这和 OOCSS 框架在处理上是不同的。XSS 的这种定义方式提高了 CSS 选择器权重，不至于因为两个类位置有变化而破坏样式。

### 4.5.3 兼容 HTML5

XSS 对浏览器默认样式的重置和普通的 `reset` 不同，除了一些必要的默认样式重置外，XSS 更多的是对 HTML5 或者其它一些在各主流浏览器里出现不兼容问题的修复，并且通过一些小技巧增加了 HTML 元素的可用性。

### 4.5.4 维护成本低

除了 XSS 提供的三个核心文件外，开发者需自行新建一个专门管理可复用组件的 LESS 文件（如 `mod.less` 文件）和一个表示某个需求或者某个项目的专用 LESS 文件（如 `demo.less` 文件），并且在 `demo.less` 中把 `fn`、`libraries`、`layout` 以及 `mod` 这几个文件通过 `@import` 方式引用进来。接着就可以依据 XSS 的三大原则在 `mod.less` 中定义页面中的组件了，把特定需求或项目的样式放在 `demo.less` 维护。

## 结 论

针对前端开发中存在的代码规范、代码维护等问题，以及网页在多终端的体验需求，本人通过学习 CSS、HTML 等 Web 前端开发相关技术及网页设计的相关知识，在 OOCSS（面向对象 CSS）框架的基础上，使用 LESS 语言，设计了一个统一的响应式 CSS 框架：XSS。

XSS 框架分离了服务性与应用性的代码，从架构上来说，它分为服务层与应用层。

本质上来说，服务层是一个通用的变量与函数库，这些变量与函数都是允许跨项目调用的，所有项目的代码都只存在一个统一的服务型接口，不仅加强了代码的重复利用，也方便了开发者的代码管理工作，特别是服务层经过编译后不会产生任何实际代码，几乎不会占用额外的空间。

应用层又分为公用样式库、组件库、布局类库和业务专用样式库。公用样式库提供了一套统一的解决浏览器兼容问题的方案，同时也对浏览器的一些默认样式进行重置并修改以增强 HTML 元素的可用性。组件库与布局类库也分别提供了统一的组件模型和布局方案。

良好的设计是为了更灵活的应用。XSS 轻巧灵活的设计使得应用 XSS 开发的网站更容易适应多终端、屏幕分辨率碎片化与越来越高的用户体验要求的环境。

但是，XSS 仍存在不足之处。由于 CSS 是一种弱类型的描述型语言，XSS 不免存在代码松散的缺点；另外，虽然 XSS 是一个通用的框架，但在不同项目中还是会存在一些微小却能致命的冲突，所以推荐开发者在对应用层有所侧重的修改后再使用。

真正强大的 CSS 框架应该是在实际开发中不断去提炼和总结，渐进式完善的。目前 XSS 也只是提出三大设计准则、代码的分层观，并设计出核心库，在今后的工作中，本人还将继续完善此框架，并与开发同事定义好接口，Web 可访问性和可用性方面的技巧也会越来越多地应用到 XSS 中。

XSS 属开源项目，本人将源代码放在 Github 上托管，写好框架的帮助文档后会邀请一些同行一起维护，希望能为前端开发在中国的发展出一份力。

## 参考文献

- 1 中国工业和信息化部电信研究院. 移动移动互联网白皮书 (2013). 2013 年 2 月
- 2 Ethan Marcotte. Responsive Web Design. A List Apart. 2010 年 5 月
- 3 World Wide Web Consortium. Introduction to Web Accessibility. WAI. 2010 年 5 月
- 4 闫国利 白学军. 眼动研究心理学导论. 科学出版社, 2012 年 1 月
- 5 Jakob Nielsen Kara Pernice 冉令华. 用眼动追踪提升网站可用性. 电子工业出版社, 2011 年 5 月
- 6 Steve Souders. O'Reilly. High Performance Web Sites. O'Reilly Media, 2007 年 9 月
- 7 温谦. CSS 彻底研究. 人民邮电出版社, 2008 年 2 月
- 8 林小志. CSS 那些事儿: 掌握网页样式与 CSS 布局核心技术. 电子工业出版社, 2009 年 10 月
- 9 World Wide Web Consortium. CSS 2.1 Spec. 2011 年 7 月
- 10 Brett D. McLaughlin Gary Pollice Dave West. Head First Object-Oriented Analysis and Design. O'Reilly Media, 2006 年 12 月
- 11 Martin Fowler Kent Beck John Brant William Opdyke Don Roberts. Refactoring. Addison-Wesley Professional, 1999 年 7 月
- 12 孙亮. 无懈可击的 Web 设计 II. 人民邮电出版社, 2010 年 8 月
- 13 傅捷 祝军 李宏. 网站重构. 电子工业出版社, 2011 年 3 月
- 14 Smashing Magazine. 众妙之门: 网站重新设计之道. 人民邮电出版社, 2013 年 4 月
- 15 Dave Shea Molly E. Holzschlag. CSS 禅意花园. 人民邮电出版社, 2007 年 6 月
- 16 Michael Bowers. Pro CSS and HTML Design Patterns. Apress , 2007 年 4 月

## 致 谢

在这个过程中，本人把大学四年吸收的知识做了一次系统的沉淀，这是一个艰辛的过程，这也是一个不断获得惊喜的过程。

本课题设计的完成要感谢学院教研室的詹秀菊老师和微信 UI 团队负责人翁乐腾老师的鼎力指导以及学院其他老师、UI 团队其他同事的支持，还有学院 09 级田杰和何春梅的友情帮助以及其他同学的提醒，感谢学生创新实践基地给予了一个安静舒适的学习环境，感谢基地里的每位同学给予的创作灵感，最后，还要感谢自己的坚持。