CSS揭秘第一弹

序

有些bug涉及非常基础的层面，以至于各浏览器的布局行为完全不兼容，这迫使我们想出对策，反过来**利用浏览器自身的解析器bug来变相地纠正这些不一致地行为**！

当你把足够多的工作部件组合到一起之后，不管单个部件看起来有多简单，这个组合体也一定会产生有趣的结果。（关于这个话题的更多内容，请看《**乐高大电影**》）

你可以通过渐变图案来挖出凹角，让元素产生动画，扩大可点击区域，甚至创建饼图。

前言

CSS经历了一场巨变，正如JavaScript在2004年前后所经历的那场革命，它从一门极度简单、功能有限的样式语言，发展成为一项由**80多项W3C规范（含草案）**所定义的复杂技术，并建立起了独有的**开发者生态圈、专属的技术会议、专用的框架和工具链**。

- 开发者生态圈是什么?
- 专属的技术会议有哪些?
- 专用的框架?
- 工具链?

CSS已经如此壮大，以至于一个普通人已经无法把它完整地装进自己的头脑了。

在2009年之前，评价一个人的CSS专业程度并不是看他对这门语言的了解有多深。对于当时的CSS行业，**一个人能否称得上CSS高手，往往要看他能记住多少个浏览器bug和相应的对策**。

从2015年开始，现在的浏览器都是以web标准作为设计基准的，过去那些针对特定浏览器的脆弱hack早已风光不再。虽然不兼容的情况仍然无法避免，但是现在的浏览器几乎都已经实现自动更新了，把这些不兼容的情况记录在书中完全是浪费时间和空间。

如今的挑战是，在保证DRY，可维护、灵活性、轻量级并且尽可能符合标准的前提下，把我们手中的这些CSS特性转化为网页中的各种创意。这正是本书要呈现的内容。

- **DRY（Don't Repeat Yourself）：不应该重复你已经做过的事（所谓的干货）**
- **WET**
  - We Enjoy Typing：我们喜欢敲键盘
  - Write Everying Twice：同样的代码写两次
- 如何提升CSS代码的可维护性?
- 如何增强CSS代码的灵活性?
- 如何让CSS代码更轻量?

本书最核心的目的是教你如何用CSS解决难题

理解发现解决方案的过程比解决方案本身更有用

子主题

本书是怎样炼成的

这本书完全用HTML5写成，并用到了一些有O'Reilly的**HTMLBook标准**定义的data-属性

HTMLBook标准：http://oreillymedia.github.io/HTMLBook

本书的大量图片是由SVG生成的，或者是**由SCSS函数生成的SVG data URI**.

书中的公式是在**LaTeX**中写成，然后转换成**MathML**

书中的所有页码，章节号，攻略编号都是由纯粹的**CSS计数器**生成的。

编写本书用到的工具

- Git：用于版本控制
- SCSS：用于CSS预处理
- 整本书都是在Espresso（http://macrabbit.com/espresso）这款文本编辑器中写成的
- CodeKit用于把SCSS编译成CSS
- Dabblet（http://dabblet.com）用于存放在线演示
- 那些SVG格式的插图并不是手工写成的，而是在Adobe Illustrator中创建的
- 使用Adobe Photoshop来处理屏幕截图

网站

提供了及时有效的浏览器兼容性信息

Can I Use...: http://caniuse.com

WebPlatform.org：http://webplatform.org

Mozilla Developer Network：http://developer.mozilla.org

维基百科上的“浏览器排版引擎对比（CSS兼容性）”词条：http://en.wikipedia.org/wiki/Comparison_of_layout_engines_(Cascading_Style_Sheets)

注意事项

CSS代码的兼容性写法：应该把标准语法排在最后

提供回退机制：通常是一种很好的做法，可以确保你的网站不会在低版本浏览器中挂掉

当不能通过层叠样式提供完善的回退方案时，可以使用**Modernizr**（http://modernizr.com/）这样的工具来给根元素<html>添加一些辅助类。

比如textshadow或者no-textshadow等，可以针对支持或不支持某些特性的浏览器来分别编写样式了

```
h1 {color: grey};
.textshadow h1{
    color: transparent;
    text-shadow: 0, 0, .3rem grey;
}
```

尝试CSS新特性

可以使用@supports规则来实现回退，可以将其视作浏览器“原生”的Modernizr

```
h1 {color: grey};
@supports (text-shadow: 0, 0, .3rem grey){
    h1{
        color: transparent;
        text-shadow: 0, 0, .3rem grey;
    }
}
```

@supports使用范围比较窄

特性检测

检测某一元素上是否存在某一元素

```
function testProperty(property){
    var root = document.documentElement;
    if(property in root.style){
        root.classList.add(property.toLowerCase());
        return true;
    }
    root.classList.add('no-' + property.toLowerCase());
    return false;
}
```

检测某个具体的属性值是否支持

```
function testValue(id, value, property){
    var dummy = document.createElement('p');
    dummy.style[property] = value;
    if(dummy.style[property]){
        root.classList.add(id);
    }
    root.classList.add('no-' + id);
    return false;
}
```

浏览器可以解析某个CSS特性并不代表他已经实现（或正确实现）了这个特性