New Riders

CSS禅意花园

the Zen of CSS design

[美] Dave Shea
Molly E. Holzschlag 著
陈黎夫 山崴颐 译

Web视觉艺术设计的王者之书

人民邮电出版社
POSTS & TELECOM PRESS

图书在版编目（CIP）数据

CSS禅意花园 /（美）莫里（Molly,D.S.）著；陈黎夫等译. —北京：人民邮电出版社，2007.7
ISBN 978-7-115-16035-5
Ⅰ. C... Ⅱ. ① 莫... ② 陈... Ⅲ. 主页制作—软件工具，CSS Ⅳ. TP393.092
中国版本图书馆CIP数据核字（2007）第044763号

版 权 声 明

Dave Shea and Molly E. Holzschlag:The Zen of CSS Design:Visual Enlightenment for the Web
ISBN:0201303474
Copyright © 2006 by Dave Shea,Molly E. Holzschlag.
Authorized translation from the English language edition published by New Riders.
All rights reserved.

本书中文简体字版由美国New Riders出版公司授权人民邮电出版社出版。未经出版者书面许可，对本书任何部分不得以任何方式复制或抄袭。

版权所有，侵权必究。

CSS 禅意花园

◆ 著　[美] Dave Shea　Molly E.Holzschlag
译　陈黎夫　山崴颋
责任编辑　李　际

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
邮编　100061　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京精彩雅恒印刷有限公司印刷
新华书店总店北京发行所经销

◆ 开本：880 × 1230　1/20
印张：14.5
字数：263 千字　2007 年 6 月第 1 版
印数：1– 6 000 册　2007 年 6 月北京第 1 次印刷

著作权合同登记号　图字：01-2007-0992 号
ISBN 978-7-115-16035-5/TP

定价：49.00 元

读者服务热线：(010)67132705　印装质量热线：(010)67129223

内容提要

本书作者是世界著名的网站设计师，书中的范例来自网站设计领域最著名的网站——CSS Zen Garden（CSS禅意花园）。全书分为两个主要部分。第1章为第一部分，讨论网站“CSS禅意花园”及其最基本的主题，包含正确的标记结构和灵活性规划等。第二部分包括6章，占据了本书的大部分篇幅。每章剖析“CSS禅意花园”收录的6件设计作品，这些作品围绕一个主要的设计概念展开，如文字的使用等。通过探索36件设计作品面临的挑战和解决的问题，读者将洞悉主要的Web设计原则以及它们运用的CSS布局技巧，理解CSS设计的精髓，恰当地处理图形和字体来创建界面优美、性能优良且具有强大生命力的网站。

本书原版书自出版以来持续畅销，受到众多网站设计师的推崇。本书适合网站设计人员和网站设计爱好者阅读，更是专业网站设计师必读的经典著作。

译者序

“CSS禅意花园”网站（www.csszengarden.com）的超常创意就已经足够让我们唏嘘不已——亘古不变的HTML文本，仅靠着数百个不同的CSS样式表，就成就了数百个风格迥异而又精彩绝伦的网页设计作品！而一本计算机技术著作的书名若是敢包含“禅（Zen）”，那么作者又将会对该技术有怎样的理解，对自己有着怎样的自信？

而此时此刻，这样一本讲述“CSS禅意花园”网站的、名为《CSS禅意花园》的书就这样静静躺在你手中！毫不夸张地说，它丝毫没有辜负“禅”这个美名的褒扬，加上“CSS禅意花园”在Web标准领域内无可争议的地位，二者相辅相成，交相辉映，让本书当之无愧地站在了Web标准设计领域中旁人难以企及的巅峰。

这本书是教我们学习CSS的么？不尽然。这本书是教我们学习Web标准的么？不尽然。这本书是教我们学习网页设计的么？不尽然。那这本书是做什么的呢？为什么要读呢？

实际上，这并不是一本用来入门的教科书，它已经跳出了某种技术或是某个主题的限制，涉及的范围异常广泛：从Web技术的发展到用户体验，从简单的模仿到富有灵性的创意，从心理学、人类学和社会学到深层次的艺术理念……无所不包却又大都点到为止，留给我们充分的思考空间——薄薄的300页篇幅、屈指可数的7个章节，寥寥的36个实例无一不体现了作者的精巧与良苦用心。所谓滴水藏海，阅读本书，我们将体会到的是一种开阔、超然的对技术、对美的总体把握能力——这或许也正是“禅”的真谛吧！全书始终围绕在符合Web标准的网页设计的方方面面，但却从没有任何枯燥的说教，抑或是冗长的代码，而是在更高的位置、从一个全新的角度娓娓道来，如涓涓溪流般潜移默化地让你体会到Web标准的好处。正如老子在《道德经》中所说的那样，大音希声，大象无形。

100个人眼中有100个哈姆雷特，本书也是如此。对于CSS、Web标准以及网页设计技术，无论你是初出茅庐的新手，还是渴望更进一步的提高者，抑或是浸淫多年的专家，都能在书中领悟到独一无二的意境，学到适合自己的知识。当然，你最好已经有了丰富的Web设计、开发经验，否则只被本书中那一个个漂亮精致的设计所吸引，而却对其背后隐藏的那片世界望洋兴叹，岂不遗憾？

若你只是想要一本CSS语法的快速参考手册，或是一本从头到尾循序渐进讲述Web标准的优势以及实现方法的“传统”技术图书，那么本书显然不会是一个好的选择——若你正在书店中，那么方圆一米之内就有无数种更“好”的替代品。阅读本书，你需要小资一般的情调——暖暖阳光的午后、安静舒适的房间、一盏弥漫的香

茗……仔细领悟其中蕴含的深意，时而沉浸其中，时而掩卷沉思，时而打开电脑记录下最新的灵感，你的心灵将要充盈着沉静、安闲，直至领悟到技术与艺术结合起来的那份超然的意境。

正是因为本书的原版如此经典，我们在翻译时备感诚惶诚恐。原作语言带有强烈的西方文化特色，很多俗语、典故更是让翻译的过程充满了艰辛，加之时间有限，以致一度让我们萌生了退却的念头，深恐一不小心糟蹋了如此的传世之作。虽在翻译的过程中竭力以求信、达、雅，但限于自身水平，译本很多地方仍旧有所不足，还望各位读者不吝指正。您可以通过我的博客（http://dflying.cnblogs.com）或E-mail（DflyingChen@hotmail.com）与我们联系。本书的勘误也将发布到我的博客网站上。

在CSS领域，很多术语的翻译颇有争议，也非常值得玩味。这里我们列出翻译中遇到的一些常见的、争议较大的CSS术语对照。选择的标准主要考虑约定俗成的翻译习惯以及更明确的表意。我们还在http://dflying.cnblogs.com/archive/2007/02/04/639837.html中一一列出了选择这些译法的详细原因以及各种译法之间的比较，若有更好的建议，也请至上述地址指出，我们会在再版时予以修订。

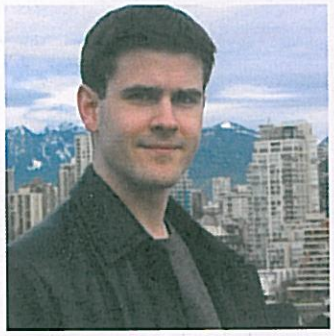
- box model：盒模型
- padding：内边距
- margin：外边距
- hack：（不译）
- liquid layout：流式布局
- ancestor element：祖先元素
- descendant element：后代元素
- filter：滤镜
- selector：选择器
- accessibility：可访问性

在翻译过程中，我们得到了众多家人、朋友和同事的关心与鼓励。感谢Aditya Bansod先生，不厌其烦地为我们解释了很多原著中艰深晦涩的语句；感谢段大为先生为我们解释原著中的日语；感谢人民邮电出版社的谢工女士、李际女士和付飞先生，ajaxcn.org的李锟先生给了我们这宝贵的翻译机会；感谢Cure、xilihwala、sheva.wen、kevin、hax、Cat Chen、琳琳的小狗、THIN、small tiger和liucsoft等博客园网友为本书的术语翻译提出的宝贵意见。没有你们的支持与帮助，我们不可能在如此短的时间之内做到这些！

陈黎夫

2007年3月1日

作者简介



Dave Shea是一位图像设计师，“CSS禅意花园”网站的创始人和耕耘者，该网站获得了很多奖项，包括SouthWest Interactive会议授予的“Best of Show”。Dave Shea曾在全球的行业大会上发表演讲，其作品被世界各地的书籍和杂志广泛收录。他还是Web标准项目（Web Standards Project，WaSP）的成员，该项目是一个由Web开发人员和设计师组成的小组，致力于推广基于跨平台和标准技术的Web设计。Dave还是Web设计机构Bright Creative的拥有者兼总监，其在线出版物mezzoblue.com中所有的Web内容几乎均为Dave所作。

Molly E.Holzschlag是一位作家、教师和Web设计师，编写了30多本有关Web设计和开发的图书。她享有“最伟大的数字作家之一”的美誉，同时被认为是网上最有影响力的25位女性之一。Molly无疑是Web设计和开发界最有趣、最活跃的人物之一。

作为WaSP指导委员会的成员，Molly同其他Web开发人员和设计师一道，致力于推广W3C推荐规范。她还在亚利桑那大学、菲尼克斯大学和比马社区学院教授“网站管理员（Webmaster）”课程。在Web Techniques Magazine关闭之前的最后三年中，Molly在该杂志撰写的专栏Integrated Design深受欢迎；她还在WebReview.com担任过一年的执行编辑。若想更详细地了解Molly，请访问她的网站http://molly.com。

前言

Note

可以到www.csszengarden.com浏览“CSS禅意花园”网站和本书中列出的所有设计作品。

若想查看某段HTML标记，可使用浏览器的“查看源文件”功能，该功能通常位于“查看” 菜单中。若想查看设计中使用的CSS，首先选择该作品，然后找点击页面中的“View This Design's CSS”链接。需要注意的是，不同作品中该链接在页面中的位置也各不相同。

欢迎阅读这本关于一个将要塑造Web未来的项目的图书。

本书的诞生缘于2003年5月8日创办的一个网站。该网站的初衷是改变人们对Web设计的看法，但最终却出乎意料地改变了全球技术发展的趋势。该网站名为“CSS禅意花园”，是来自世界各地的众多天才协力合作的结晶。

“CSS禅意花园”在发布宣言和展示作品两方面双管齐下，其创建目的在于展示基于CSS设计可实现的视觉效果。网站以样式表和图像的方式向设计师征集原创作品，但规则是所有作品都必须使用相同的HTML文件。每件作品的HTML都完全相同，无一例外。

不可思议？甚至有点疯狂？也许兼而有之，但毫无疑问，只要使用得当，CSS与任何设计工具相比都毫不逊色。本书展示了网站收到的36件最杰出的作品，深入剖析了创作人员的设计历程及他们使用的工具和编码方法。

整本书就谈论一个网站？是真的吗？如果这是一个普通的网站，当然有理由这样怀疑。但“CSS禅意花园”的内容是如此丰富多彩而又如此实用，并且都由世界各地各怀绝技的设计师创作，因此对本书来说最严峻的挑战则是如何将其揉合到一本书中。

本书介绍的内容

本书是介绍和描述该网站的吗？从某种意义上说，是的。本书是有关Web设计最佳实践和技巧的教程吗？显然如此。它是一个令人赏心悦目的设计画廊，设计网站时能从中获取灵感吗？当然没问题。

“CSS禅意花园”涉及了众多有关如何完成现代Web设计的主题，如果专门探讨设计的某个方面，比如布局，那么连展示该网站的冰山一角都谈不上。编写本书时，我们就清楚地认识到了这一点，因此必须涵盖方方面面：文字的使用、色彩理论、布局设计技巧，应有尽有。

从本书中能够学到CSS知识吗？当然，但不是入门水平的基础知识。要

侧重于视觉设计技巧，必须跳过入门课程，直接进入中级。市面上的另外一些不错的技术书可帮助读者快速掌握有关CSS句法和语法的基础知识。我们建议读者购买其中一本进行快速入门，然后在本书中学习如何以极具视觉震撼力的方式应用CSS技术。

本书将重点放在解决视觉设计问题上，而不是详细描述CSS规范的方方面面。在恰当的时候也会讨论浏览器的支持情况，以保证实用性。

本书针对的读者

本书适合所有对设计和开发网站感兴趣的读者阅读。

设计师

本书采用视觉化的方式讨论CSS设计理论和实践，以使用基于标准的最新技术完成实际作品和解决实际问题为目标。

程序员

本书的重点是视觉艺术，但并不要求读者具有图形设计基础。作者从始至终都在尽最大努力以最佳的方式定义设计术语，确保即使读者不具备任何设计知识，也能理解并运用本书中介绍的概念和技巧。

CSS新手

尽可能少用行话和技术术语，即使读者对CSS没有足够清晰的认识，也能理解大部分章节。建议CSS新手从推荐读物（参见本书末尾的“结语”部分）中选择一本阅读，学习基本概念和语法，然后通过阅读本书来拓展这些概念并在作品中实际应用。

书中探讨了一些基本的Web设计原则，适用于大多数但不是所有情况。当然可以使用Adobe Dreamweaver [1]等自动设计工具来创建基于CSS的网页，但对那些使用HTML编辑器手工编写代码的设计人员来说，本书介绍的大部分CSS技巧将显得更加有用。如果仅依赖拖放来完成Web设计，将会发现本

1 原文为“Macromedia Dreamweaver”，但Macromedia已与Adobe公司合并，产品名称也有所改变。——译者注

书在很多方面技术性太强。但愿这不会让读者丧失拓展知识的勇气。

CSS高级用户

即使是对CSS了如指掌的读者也能从阅读本书的过程中受益良多。本书侧重于视觉艺术，辅以实际的CSS示例，这些实例均出自当今最具创意的一些设计师之手；书中涉及的主题丰富多彩，即使是那些经验最为丰富的专业人员也能从中学到一些全新的技巧。

如何阅读本书

网站通常都有入口，被称为主页。在这里，用户可以通过点击某个标题跳转到网站的任何地方，浏览其中包含的内容；也可以使用搜索功能查找所需的内容；还可以随意单击，了解能获悉哪些更深入的内容。

本书采用的也是这种模型，阅读的方式也没有对错之分。既可以从头开始逐页阅读，也可以直接跳到感兴趣的章节。读者可以以任何非线性方式阅读本书，以任何顺序研究书中的每个作品。

本书分为两个主要部分：

- 第一部分为第1章，讨论网站“CSS禅意花园”的创建及其最基本的主题，包含正确的标记结构和灵活性设计等。
- 第二部分包括6章，占据了本书的大部分篇幅。每章剖析“CSS禅意花园”收录的6件设计作品，这些作品围绕一个主要的设计概念展开，如文字的使用。通过探索36件设计作品面临的挑战和解决的问题，读者将洞悉主要的Web设计原则和CSS布局技巧的运用。

下面概述这7章的内容。

第1章　追本溯源

概述“CSS禅意花园”的来历及CSS的发展历程，讨论正确的标记结构和设计。

第2章　设计

阐述基本设计元素及其在Web设计中的应用，包括色彩理论、比例、位置、文字和图片之间的关系以及如何协调地使用Adobe Photoshop和CSS。

第3章　布局

详细阐述如何创建复杂的CSS布局，包括分栏、浮动内容和定位方案。

第4章　图像

使用图像改善布局以及如何生成图像。包括图像替换、图形文件格式及寻找图像素材。

第5章　文字排印

Web上字体的局限性及其避免方法等有关文字的方方面面，包括字号、字体的选择和基于图像的文字等。

第6章　特效

掌握基本知识后该做什么？根据用户使用的浏览器采用不同样式的高级CSS效果、创建灵活布局的技巧，以及使用巧妙的代码克服技术的局限性。

第7章　重构

介绍6位设计师进行重构的方法。给出了这6件设计作品创建的过程，详细阐述了每一步。综合运用本书其他5章介绍的设计原则，就能在这里获得最佳的效果。

可以看到，我们竭尽所能，将尽可能多的优秀内容揉合到本书中，这几乎是市面上最全面的有关基于标准的现代Web视觉设计的参考资料。

编写本书是一种享受，但愿你阅读时也是如此！

Dave Shea，Molly Holzschlag

2004年8月

Dave Shea感谢

首先也是最重要的，感谢我的妻子April容忍我在那么多的夜晚中独自在星巴克咖啡店中写作。若是没有了她的支持、理解和耐心，我不可能完成本书。

（这也提醒了我——感谢这家本地的星巴克咖啡店的所有员工给我带来那么美味的那铁咖啡，且对总是独自呆在角落中敲击着PowerBook的我没有丝毫厌烦。）

感谢我的父母Gord和Linda以及我的兄弟Matt和Jeremy，能善意地装作理解我的所作所为。

感谢DreamFire工作室（www.dreamfirestudios.com），虽然访客越来越多，但从开始到现在，它一直为禅意花园网站提供着服务器。

围绕在禅意花园的设计师社区给了我很多支持和鼓励，这些都是我在两年前着手本书是从未想到过得。这里有太多太多的人需要感谢，而书中却遗憾地不能全部列出，但你们却知道曾经给我的帮助。虽然很多人在下面的列表中，也有还有很多人不在，但你们的那些让Web更加丰富多彩的作品却无时无刻地带给我无穷的勇气和灵感。

最好、最多、最真诚的感谢，献给我的合作者Molly，感谢她能把她的想法和经验与我分享，感谢她给我的鼓励。很多年前她的一本书将我带入了Web设计的殿堂，能够与她共同完成本书是我的荣幸。

Molly E. Holzschlag感谢

若是某个领域中的学生很快就能成为我们的老师，那么能处在这个领域中绝对会是段不平常的经历。我们现在做的正是这样的事情——立即改变人们对Web的想法。与Dave合著该书时我也再次感受到了从学生到老师的那个奇迹般的过程——我的作品启发了一个人，这个人很快又启发了新的一批Web设计师。Dave以及很多禅意花园设计师都让我更好地理解了CSS，进而更有效地教授CSS，最终以一种全新的方式对Web又充满了热情。因此，最

主要的感谢必须致以Dave，以及所有本书的贡献者——很多人都是我的朋友或老师，都曾在过去的一段时间中帮助我认识到了Web项目中的最重要之处。

从个人角度，我要感谢深爱的Michael以及我的家人和朋友，你们忍受了我善变的性格并自始至终陪在我身边。还要感谢David Fugat，这位异常优秀的文稿代理人，也是我的好朋友，从不吝惜他的鼓励和指导。

更要感谢我的网站的读者和访客：对我而言，你们长久以来的支持和鼓励是任何其他东西都无法替代的！

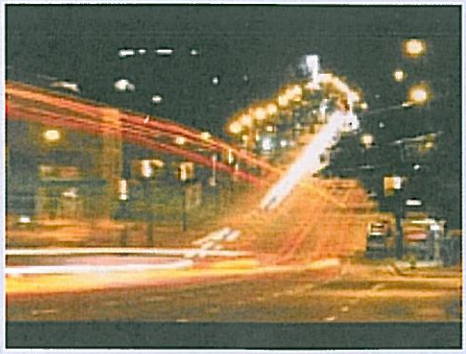
Dave和Molly共同的感谢

献给所有参与本书编辑出版工作的朋友一个最真诚的感谢！Cheryl England不但是个绝佳的编辑，也让编辑的过程充满了乐趣！感谢Shaun Inman帮助本书在技术上精益求精，感谢Elissa Rabellino细心的编辑工作。同样感谢产品团队的成员（Charlene Will、Kim Scott和Hilal Sala）帮助我们保证了本书的质量。最后，一份特殊的感谢献给Nancy Ruenzel和Cliff Colby，是你们坚信本书的取材，并随后帮助我们付诸于实践。

特别感谢

感谢所有认为CSS禅意花园值得参与的朋友，是你们造就了禅意花园的今天。若是没有你们的努力，世上决计不会出现本书。

（人名列表省略）



Night Drive[DC1]，70页

White Lily[DC4]，90页

Coastal Breeze[DC4]，140页

目录

第1章　追本溯源　1

第2章　设计　39

2.1　Atlantis（亚特兰蒂斯）　40
2.2　Zunflower　46
2.3　Springtime（春天）　52
2.4　Viridity（翠绿活力）　58
2.5　Ballade（叙事曲）　64
2.6　Night Drive　70

第3章　正文布局　77

3.1　Backyard（庭院）　78
3.2　Entomology（昆虫学）　84
3.3　White Lily（白百合）　90
3.4　prêt-à-porter　96
3.5　CS(S) Monk　102
3.6　Not So Minimal　108

第4章　图像　115

4.1　Japanese Garden（日式花园）　116
4.2　Revolution!（革命！）　122
4.3　Deco　128
4.4　No Frontiers!　134

4.5 Coastal Breeze 140
4.6 What Lies Beneath 146

第5章 文字排印（Typography） 153

5.1 Oceans Apart（天水之间） 154
5.2 si6 160
5.3 Release One（释放） 166
5.4 Dead or Alive（生与死） 172
5.5 Blood Lust（血色欲望） 178
5.6 Golden Mean（黄金平衡） 184

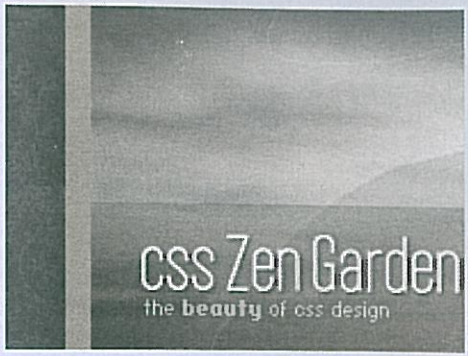
第6章 特效 191

6.1 This is Cereal 192
6.2 Gemination（双子座） 198
6.3 Bonsai Sky（盆景天空） 204
6.4 Tulipe（郁金香） 210
6.5 door to my garden（花园之门） 216
6.6 Elastic Lawn（绿草如茵） 222

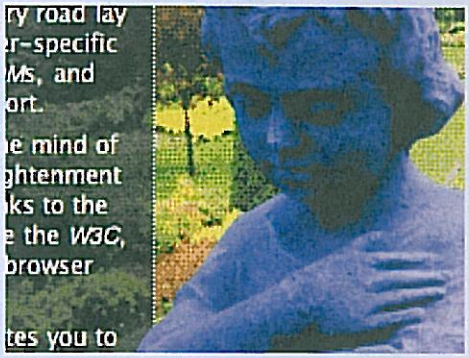
第7章 重构 229

7.1 Hedges（树篱） 230
7.2 Radio Zen 236
7.3 South of the Border（南部边境） 242
7.4 Corporate Zen Works 248
7.5 Open Window（敞开窗口） 254
7.6 mnemonic（记忆） 260

结语 266

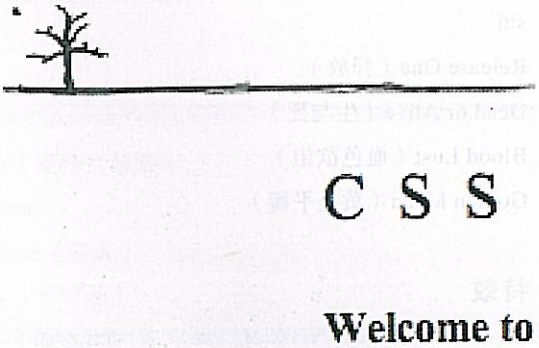


Oceans Apart[DC1]，154页


Gemination[DC2]，198页

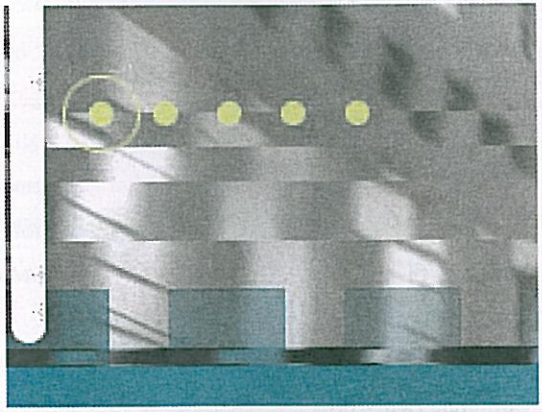
South of the Border[DC5]，242页

CSS Zen Garden（CSS禅意花园），2页



Wind （风），6页

Stormweather（暴风雪），7页



arch（拱门），7页

第1章

追本溯源

Salmon Cream Cheese，7页

A Scripted design，36页

长久以来，Web设计就在快速变化着。Web设计师需要逐字逐句地查看源文件，反复试验并挑战自己的极限。这是一个有趣的过程，也通常是我们自发来做的工作

随着时间的流逝，Web标准及其最佳实践开始逐渐浮出水面，这个过程直接指向了我们的工作所基于的最重要的软件——Web浏览器。在本书中，你将学到掌握当代Web设计的理念、过程和技术所必需的一切知识。

Zen Garden

The Beauty of CSS Design

A demonstration of what can be accomplished visually through CSS-based design. Select any style sheet from the list to load it into this page.

Download the sample html file and css file

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support.

Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators.

The css Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

So What is This About?

There is clearly a need for CSS to be taken seriously by graphic artists. The Zen Garden aims to entire, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The code remains the same, the only thing that has changed is the external *.css file*. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. To date, most examples of neat tricks and hacks have been demonstrated by structurists and coders. Designers have yet to make their mark. This needs to change.

Participation

Graphic artists only please. You are modifying this page, so strong CSS skills are necessary, but the example files are commented well enough that even CSS novices can use them as starting points. Please see the CSS Resource Guide for advanced tutorials and tips on working with CSS.

You may modify the style sheet in any way you wish, but not the *HTML*. This may seem daunting at first if you've never worked this way before, but follow the listed links to learn more, and use the sample files as a guide.

Download the sample html file and css file to work on a copy locally. Once you have completed your masterpiece (and please, don't submit half (finished work) upload your .css file to a web server under your control. Send us a link to the file and if we choose to use it, we will spider the associated images. Final submissions will be placed on our server.

Benefits

Why participate? For recognition, inspiration, and a resource we can all refer to when making the case for CSS-based design. This is sorely needed, even today. More and more major sites are taking the leap, but not enough have. One day this gallery will be a historical curiosity; that day is not today.

Requirements

We would like to see as much CSS1 as possible. CSS2 should be limited to widely-supported elements only. The css Zen Garden is about functional, practical CSS and not the latest bleeding-edge tricks viewable by 2% of the browsing public. The only real requirement we have is that your CSS validates.

Unfortunately, designing this way highlights the flaws in the various implementations of CSS. Different browsers display differently, even completely valid CSS at times, and this becomes maddening when a fix for one leads to breakage in another. View the Resources page for information on some of the fixes available. Full browser compliance is still sometimes a pipe dream, and we do not expect you to swear up with pixel-perfect code across every platform. But do test in as many as you can. If your design doesn't work in at least IE5+/Win and Mozilla (run by over 90% of the population), chances are we won't accept it.

We ask that you submit original artwork. Please respect copyright laws. Please keep objectionable material to a minimum; tasteful nudity is acceptable, outright pornography will be rejected.

This is a learning exercise as well as a demonstration. You retain full copyright on your graphics (with limited exceptions, see submission guidelines), but we ask you release your CSS under a Creative Commons license identical to the one on this site so that others may learn from your work.

Bandwidth graciously donated by DreamFire Studios. Coming soon: Zen Garden, the book.

1.1 追本溯源

20世纪90年代末的互联网是一个疯狂且危险的新领域。任何事情都可能发生，新东西不停地出现。没有人能确定互联网当年应该是怎样的。就像孩童拿到了一件新玩具一样，“挑战极限”成了那时人们的口头禅。

1.1.1 起源

人们还在不停地挑战极限。但同时，HTML也以一种用来标记研究论文的简单语言的身份呱呱坠地。浏览器厂商和图形设计师都对其低下的布局能力颇有微词。于是，类似 <font>标签和被强行用作布局的<table>标签等hack [1]应运而生，用来克服HTML的这些布局上的弱点。但这些hack却破坏了HTML语言的本意——以结构化的方式表示文档。

一些基本的HTML元素被滥用于与其本意毫不相关的页面布局中。同时一些具有明确功能的元素却因为不够“美观”而被人们抛弃。为了迁就网站的视觉效果，正确的文档结构往往被忽视。

当然，那时主流的浏览器生产厂商并没有帮上什么忙。在事情向好的方向发展之前，情况似乎还将变得更加糟糕。当时的两大浏览器霸主——微软和网景都在不停地开发出自己的私有扩展，再基于这些扩展进一步提供新的扩展，这就是著名的浏览器之战。虽然竞争有利于创新，但这个过程也同样让人感到窒息：Web开发者不得不为同一个站点维护多个版本的HTML代码，因为为某种浏览器编写的HTML代码极少能够在另一种浏览器中正常显示。

---

[1] 指一些偏重于技巧类的、非常规的用来实现某种布局要求的手段和方法。例如用<table>标签进行布局就属于一种hack。——译者注

总之，当时的情况是一团糟。尽管存在着这样那样的问题，但.com兴起时吸引的大量的资金仍旧让Web保持了不断的增长——有了足够的钱，即使要为同一个网站编写两套不同的HTML也没什么大不了的。但在千年之交的时候，无论是网站编写者还是投资者，都终于无法忍受了。

1. 曙光初现

在新世纪之初，资金枯竭，Web上的预算减少。随着状况急转直下，那些尚未失业的Web开发者很快认识到，为各种非标准浏览器编写代码所带来的额外工作已不能承受。是时候去改变一些东西了。

一些来自“草根”的Web设计师组成了一个叫做Web标准项目（Web Standards Project，WaSP；www.webstandards.org）（图1-1-1）的组织，并劝说主流浏览器生产厂商将它们的那些“狂放不羁”的浏览器标准统一起来，共同遵从W3C（World Wide Web Consortium）所“建议”的规范。这样，浏览器生产厂商基于W3C规范进行标准化的过程就为Web开发注入了一丝“一致性”的意识。人们有理由相信，在未来的一段时间之内，Web设计师所创建的复杂HTML页面完全可能以统一的样式呈现在各种浏览器和操作系统之上。

W3C规范大都专注于解决较高层的编码及架构方面的问题，但普通的Web设计师却很少需要关心这些。即使这样， W3C还是从20世纪90年中期发布标准化HTML版本开始，陆续提供了一系列的规范。这一系列规范约束了发送给浏览器的HTML代码，旨在以一种智能、易用并便于实现的方式解决浏览器之间的兼容性问题。

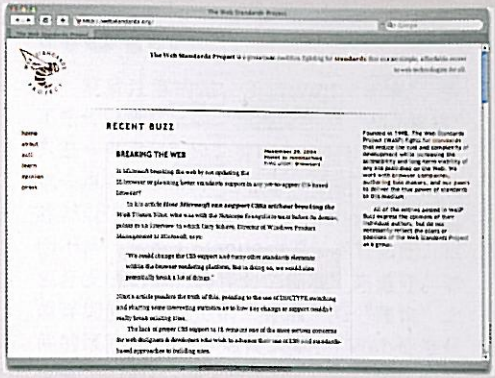
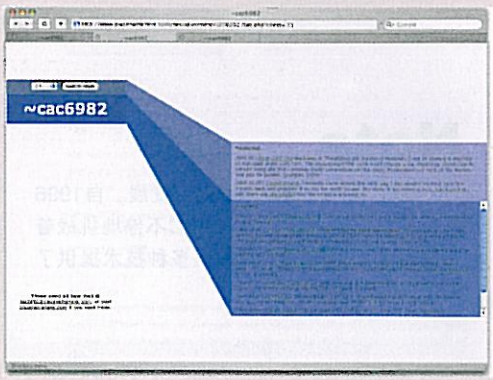


图1-1-1　Web标准项目

Note

W3C致力于推动互联网的发展。自1996年发布第一份规范开始，W3C不停地挑战着现有的互联网规则，并为很多种技术提供了规范的解决方案。



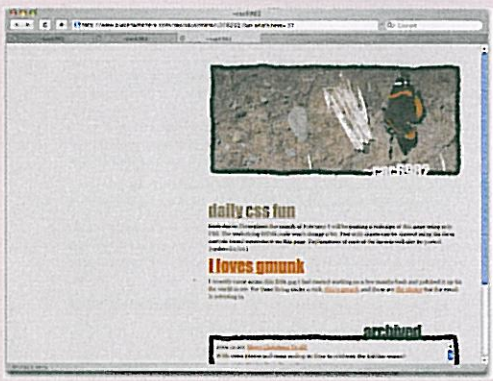


图1-1-2　几个Daily CSS Fun的示例CSS效果

2. 为什么是这些标准

W3C 工作组成员由很多不同背景的专业人士组成，这些专业人士聚集在一起，讨论解决一些超前的、普通Web设计师永远都不用关心的问题。他们的目标就是提供一系列经过深思熟虑的建议，这些建议为软件开发者和浏览器指明了发展的方向。若网站编写者和设计师遵从了这些建议，就可以保证其站点能够在最多数的用户代理（user agent）中正常显示并正常使用。

当然，这里的“用户代理”指的是浏览器。但普通的桌面浏览器仅仅是众多“用户代理”中的冰山一角而已。随着移动设备（例如移动电话）的迅速发展，越来越多的用户不再被束缚在PC机前。而且，也不是所有的用户都有能力用浏览器访问网站——因为有些用户不能使用浏览器。例如，一些有特殊可访问性（accessibility）需求的视力不便的用户可能需要借助屏幕阅读器软件、盲文设备或屏幕放大软件等来帮助访问网站。

因为W3C创建这些规范的目的是改变20世纪90年代Web领域中的各种混乱的不兼容现象，所以很难为所有用户代理制定出统一的网站规范。考虑到这些，W3C发布了一系列的规范，包括HTML 4.01、XHTML、层叠样式表（Cascading Style Sheets，CSS）和DOM等。

这些规范所包含的技术都是Web标准项目组织里的设计师和开发者所希望看到的。遵从这些规范，我们就可以在保证网站支持大多数最广泛的用户代理的同时，也能够对其布局进行精确的控制（使用CSS），但这确实是一种与以往完全不同的网站创建方法。要让设计师们相信学会CSS是他们最好的出路并不是件容易的事。新千年的头几年一直都在阐释如何使用CSS。

就在这样的互联网大环境中，“CSS禅意花园”迎来了它的春天。当时，研究CSS的人们主要是程序设计师，这些程序设计师在如何从技术角度使用CSS方面非常精通，但他们的作品却都略显简单乏味，从艺术角度看，实在不敢恭维。而图形设计师却无法掌握使用CSS的精髓，因为当时并没有什么出色的CSS设计作品可供参考。

3. 播下种子

在2002年的晚些时候，禅意花园的创立者Dave Shea开始考虑这个问题。对于那些有能力学会并用CSS做出漂亮效果的设计师来说，这类由程序员创

建的平常之作又怎能引起他们的兴趣呢？Shea同时具有程序设计和视觉艺术两方面的背景，这也让他坚信CSS有成为一门设计语言的潜质。Shea同样意识到，那些真正应该使用CSS的人们反而并没有在使用。于是他产生了这样的想法：如果有一个网站，集合了大量的的优秀CSS设计该多好啊。简单地搜集现有的资源用处不大——因为还没有什么现成的，唯一的办法就是鼓励优秀的设计师去创造全新的作品。就这样，Shea在内心播下了禅意花园的种子。

• 先驱者

Web标准项目的创始人之一Jeffrey Zeldman在2002年发表了一篇文章，文中恳求那些已经了解了基于Web标准设计的好处的人们暂时放弃“说教”他人，而是去“使用”Web标准。“使用”成了宣扬Web标准最有说服力的武器，“展示而不是说教”也成为这批Web标准拥护者的信条。

2002年，一个叫Chris Casciano的Web开发者发起了一个名为Daily CSS Fun的项目（图1-1-2）。Casciano计划在为期一个月的时间内，每天发布一个新版本的CSS文件，让同一个HTML页面呈现出完全不同的样子。Casciano从未声称过他是图形设计师，但这个项目的结果却异常引人注目——人们都纷纷叹服于CSS也能实现如此特别的布局效果。

由于Netscape Navigator 4在当时还非常流行，且它对CSS的支持并不好，所以Casciano的这个项目也仅仅在理论上让人赞叹而已，离实际应用还有相当的距离。随着Netscape Navigator 4市场占有率的不断下降，直到2003年IE开始一统天下，CSS布局大行其道的日子似乎已经指日可待。CSS的全面应用重新被提到了议事日程之上。

• 早期的想法

禅意花园的构思终于成熟，2003年5月，Shea发布了他的个人站点，www.mezzoblue.com，很快又转到了现在的域名之上：www.csszengarden.com。

[1]原文给出的链接已经失效。——译者注

Note

需要注意的是，所谓Web“标准”，并不是传统意义上的“标准”——“Web标准”只是一种建议而已，并不是强制执行的规定。这一点在W3C的文档中也有体现：文档的名称都以“建议（Recommendation）”命名，而不是“标准（Standard）”。尽管如此，很多开发者仍旧将其作为“标准”来遵守。若想详细的了解这里面“标准”与“建议”之间的区别，请参考这篇文章：“Web标准是什么”（http://www.webstandards.org/2004/10/26/what-is-a-web-standard/[1]）

Note

可以在www.zeldman.com/daily/0802c.html#Evangeline这个页面浏览Zeldman的原文。

Note

可以在www.placenamehere.com/neuralustmirror/200202这个页面浏览Daily CSS Fun项目。

在2002年的晚些时候，Shea就已开始尝试为禅意花园编写原型，这个原型能够在四套样式表之间切换，分别代表了四种古代的元素——木、水、火和风。但这个想法并没有充分体现，主题中也没有用到足够的图片。如图1-1-3~图1-1-6所示，第一版的禅意花园草案还远远称不上吸引人。

这些设计似乎显得有些令人厌倦，因为它们都缺少了图片修饰。背景图片可以通过CSS的background-image属性声明，而作为内容的前景图片则不能。若是使用了<img>标签，则页面就会丧失了灵活性：我们将不得不被限制于一些确定了的图片中，无论CSS样式如何变化，都不能改变这些已经用<img>标签定义了的图片。但是，很多时候只使用背景图片是不够的，因为设计时所用的图片往往用来修饰文本，也就是说，这些图片需要放置于文本的两侧，而不是作为背景和文字混在一起。显然，只使用背景图片还不能完全满足这样的设计要求。

Douglas Bowman在其文章“使用背景图片来替换文本”中给出了一种权宜之计（这种解决方案目前已经是声名狼藉）：首先在页面中将前景图片

图1-1-3　禅意花园原型之一，大地

图1-1-4　禅意花园原型之一，火

图1-1-5　禅意花园原型之一，风

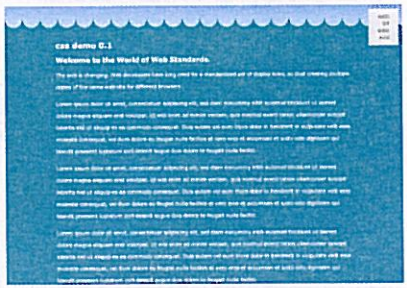


图1-1-6　禅意花园原型之一，水

和某些文本元素关联起来，然后用图片来代替原来的文本。使用CSS加上一些少量的额外标记，即可很容易地实现这个hack——与<img>标签及其alt属性的表现非常类似。

2003年4月，禅意花园的计划正式开始了。设计者发布了一个单独的HTML文件作为所有设计的基础，随着这个HTML文件发布的还有5个初始设计（图1-1-7~图1-1-11）。

图1-1-7　1号设计，安静

图1-1-8　2号设计，鲑鱼奶油奶酪

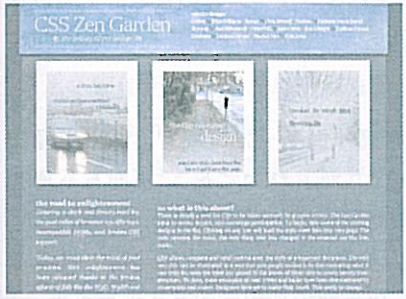


图1-1-9　3号设计，暴风雪

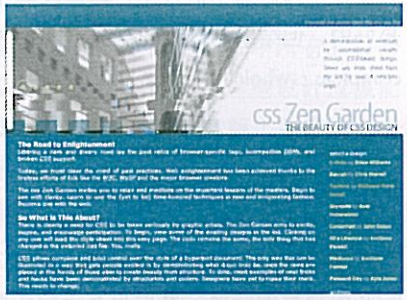


图1-1-10　4号设计，拱门

图1-1-11　5号设计，血色欲望

Note

如果你曾读过Bowman的原文（http://www.stopdesign.com/articles/replace_text/），则会发现这个解决方案存在着一些非常致命的可访问性（accessibility）问题。禅意花园的很多设计也使用了同样的技巧，该技巧将在第4章中详细介绍。

Note

因为禅意花园的HTML文件一旦发布，就不能再修改（其他人会开始为这个HTML编写CSS样式文件并提交上来），所以Shea花费了一段时间来检查文档的标记和内容是否合适，且还要保证文档中有足够的class和id元素，以便于日后用不同CSS进行修饰。但即使如此细心，在HTML正式发布之后还是出现了一些事先难以预料的问题——这些将在本章稍后部分介绍。

ABCDEFGHIJKLM
NOPQRSTUVWXYZ
abcdefghijklm
nopqrstuvwxyz

图1-1-12　Calligraphic 421字体

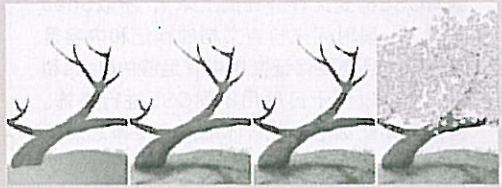


图1-1-13　树的绘制过程图解

4. 默认的设计——安静

虽然前5个初始设计都对禅意花园的成功起到了至关重要的作用，但从长远来看，只有默认的第一个设计始终保持着较高的知名度。精致简洁的“安静”始终被设定为禅意花园的默认界面。虽然也曾考虑将2号设计“鲑鱼奶油奶酪”作为默认界面，因为该设计第一眼给人的冲击力较强，但谢天谢地，最后还是选择了“安静”。

除了正文部分之外，默认设计中的所有图片和文字都是在Adobe Photoshop中手工绘制的。起初Shea希望能够用书法手工写出标题的这段文本，不过在尝试了一个小时的毛笔和墨汁之后，Shea还是决定使用另一种让人更加满意的方法——Bitstream的Calligraphic 421字体（图1-1-12）。

因为汉字笔画复杂、样式美观，所以很多设计师非常欣赏它所体现出的细节。一种典型的西方设计方式就是一知半解地将若干汉字拼凑起来，以体现出某种“神秘”的意味，但Shea却不想这样。Shea花费了一段时间研究将要放在页面左上角卷轴中的汉字，希望能够让这些汉字表示出一定的含义。虽然初衷是好的，但结果仍旧让人遗憾：卷轴中的汉字在英语中的含义大概为beginning、complete、whole和skill[1]。

还记得小学时候的这类小作品么？将墨汁倒在一张白纸上，然后用吸管将墨汁吹成类似树的形状，最后撕下一些卫生纸片作为“花”。页面右下角的那棵树的设计过程将会让你回忆起这段美好的时光。从屏幕截图中可以看到，这棵树经历了若干次修正（图1-1-13）。整个过程从树的轮廓开始，只包含一些很简单的颜色，然后再基于该轮廓，以不断添加图层的方式进行改善。

5. 禅意花园的影响

几天之内，陆续有用户开始提交他们的设计：在站点发布的第一周里，每天就有成千上万个访问者到来。人们还志愿将禅意花园翻译成数十种语言。很明显，禅意花园的出现满足了人们压抑已久的对CSS设计方面的需求，这在当时没有任何其他网站能够做到。

自发布以来，禅意花园已经出现在全球范围内的数十种杂志、图书中，巩固了它在Web设计历史上独一无二的地位。感谢信更是从世界的各个角落

[1] 在图7中我们可以看到，这几个汉字为“完”、“全”、“技”等。隐约可以感到Shea似乎要表达出类似“最高水平”的意思。——译者注

纷至沓来，设计师和开发者用禅意花园向他们的雇主、客户以及同事证明了使用CSS进行设计的优势。提交作品的设计师也由此声名远扬，禅意花园甚至推进了他们的职业发展。

虽然创建和维护是由一个人完成的，但禅意花园在很大程度上是志愿者协作的结果——才华横溢的设计师们贡献给自己的作品，帮助它获得成功。每个人都能在禅意花园中找到不同的价值。虽然创建禅意花园的本意是演示CSS在设计中的作用，但它同样在很多当初没有预料到的领域中给人们带来了帮助。

- 若是某位设计师正在被一个布局问题所困扰，那么现在他有了寻求帮助的地方——禅意花园中有数百个不同的设计，他所遇到的问题极有可能已经被别人解决了。禅意花园允许并鼓励人们学习它的CSS，并将其使用在其他站点上。
- 类似地，禅意花园中也不断出现新的页面布局技术及CSS效果。我们都有机会在数百个作品中找到一些新奇的东西。
- 禅意花园还是一个绝佳的浏览器兼容性测试网站。其中所有的设计都使用有效的、基于标准的代码编写，所以理论上这些设计应该能够在所有浏览器中以一致的方式呈现出来。（当然实际上并非如此完美。由于某些浏览器的缺陷或对某些元素缺乏支持，大多数设计师都不得不考虑浏览器之间显而易见的呈现差别，并使用多种CSS过滤器和 hack来解决。）
- 当你正尝试说服你的雇主或客户使用Web标准设计页面时，禅意花园就是一个极其令人信服的证明。
- 当你灵感枯竭的时候，禅意花园中的设计或许能够给你一些启发。感觉江郎才尽了？来禅意花园逛逛吧，说不定灵感就在不经意之间出现了呢！
- 禅意花园为经验丰富的专家和积极努力的设计师提供了一个良好的展示自我的平台。雇主和潜在的客户很愿意主动联系它们中意作品的设计师，提供工作或者合作机会。这些优秀的作品也经常出现在国际性的图书和杂志中。
- 禅意花园已成为教师和讲师在教授当代Web设计实践时的一个有力工具。

Note

只有对HTML元素非常熟悉，才能知道何时应该选用何种标签。这里我们推荐Dan Cederholm的*Web Standards Solutions: The Markup and Style Handbook*一书（2004年由Friends of ED出版社出版），作为实际开发中关于结构化标记的参考。

1.1.2 打好CSS布局的基础

在开始使用CSS布局之前，首先要保证你的Web页面是以标记结构化的方式编写的。既然正在阅读本书，那么你十有八九已经听过将结构（structure）和表现（presentation）分开的说法。若你对此尚不熟悉，也不用担心——基于CSS进行设计的最主要的优势就是，只要HTML文档结构良好，那么今后将很容易通过添加一层CSS作为修饰，让文档样式变得非常漂亮。

长期以来，格式化Web页面时基本HTML元素的选择都取决于它们的显示的样式。若需要一个带有缩进的段落，你可能会选择用<blockquote>标签将这段文字包围起来，让段落两端显示出一些空白。但顾名思义，blockquote元素表示其中包含的内容是一段引用自其他来源的文字。如果我们仅仅是为了让某一段文字以缩进形式显示而使用blockquote元素，而这段文字却并非引用自其他来源的话，那么就意味着我们错误地使用了HTML标签——HTML标签是用来表示文档结构的。对于文档的显示样式（例如某一元素的缩进），应该使用CSS来实现。

结构化HTML文档主要体现在对HTML标签正确、合理地使用中。选择HTML标签时要根据其语义，而不是其样式进行。

既然现在CSS设计已经是可用的了，那么就应该把所有的格式化和设计相关的实现放在CSS文件中，让HTML只用来表示文档结构。作为一门用来设计表现样式的语言，CSS建立于HTML的文档结构之上并为其应用可视化样式，而HTML则只用来表示文档的结构，不再用于页面样式的设计中。

那么正确、合理的文档结构能够带来什么好处呢？这一节中所提到的“结构”到底又是什么东西呢？

1. 带有语义的标记

为表现样式编码和为实现正确的语义编码，二者在基本原理上的差别比较微妙。这里给出了一段示例代码来帮助我们更好地区别。下面这一段HTML就是为了实现某种表现样式而编写的：

```
<br><br>
<b><font size="2">Our Family</font></b>
<br><br>
<font size="1">Pictured are Matt and Jeremy.As
```

```
usual,Matt is msking a funny face. We don't have many hotos
where he isn't.</font>
```

下面这一段HTML是为了达到正确的语义而编写的：

```
<div id="family">
 <h3>Our Family</h3>
  <p>Pictured are Matt and Jeremy. As usual, Matt is
making a funny face. We don't have many photos where he
isn't.</p>
 </div>
```

在第一个代码片断中，所有的HTML标签都是用来表现样式的。
标签用来换行，<b>标签用来给文本加粗，<font>标签则指定了文本显示的大小。

而在第二个代码片断中，却一点都没有对页面显示样式的说明——h3和p元素仅仅用来描述页面中某一段内容的功能。p元素中包含了一个自然段的文字，而h3元素则表示其内容是页面中的第三级标题。

关键之处就在于，第二个代码片段并没有考虑这些元素将要以何种样式呈现出来，因为元素的呈现样式将由CSS控制。之所以选择这些标签，是因为它们恰如其分地描述了其中的内容。这也正是HTML中语义的概念：选择何种HTML标签取决于标签本身的语义，而不是其默认显示出的样式。

针对语义编写HTML所带来的巨大优势将在本章稍后部分介绍。现在我们只要记住，使用CSS进行设计时的主要任务之一，就是从编写合法的、带有良好结构并拥有语义的HTML文档开始。完成了这一步之后，我们才能在这个坚实牢固的基础之上应用各种CSS样式。

2. 创建良好的标记

每个页面都在结构上有自己不同的需求，所以对于创建HTML来说，很难找到一个完美的一揽子解决方案。但我们却能够总结出一些通用的指导方针，将这些要点烂熟于胸，一定会对编写HTML大有裨益。

• 选择DOCTYPE

在HTML的最开始部分声明DOCTYPE（Document Type（文档类型）的简写）可以让浏览器或其他用户代理知道你要使用的HTML语言的类型。无论你打算选择何种类型的HTML语言，DOCTYPE都是一个健壮的文档所必需的，也是你在编写HTML时要做的第一件事情。

Note

“<p>标签”和“p元素”之间到底有什么区别？大多数时候它们表示的是同一个东西，但仍有一些细微的区别。“标签”指简单的HTML分隔符，例如<p>、 和</body>等。而“元素”则是由一对开始结束“标签”构成的，用来包含某一些内容——这里有一个值得注意的例外，即
本身既是开始标签又是结束标签，其中不包含任何内容。

Note

若想了解更多关于将文档结构和表现分来的内容，请参考Roger Johansson在456 Berea Street网站给出的白皮书——“Developing with Web Standards”（www.456bereastreet.com/dwws）。

Note

若想了解更多关于DOCTYPE的相关内容，请参考Jeffrey Zeldman在A List Apart网站中的文章——“Fix Your Site with the Right DOCTYPE”（www.alistapart.com/articles/doctype）。

一般地，“元素”用来表示某一段明确的结构，而“标签”则仅仅是用来创建文档结构的语法而已。

示例：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

• 指定语言和字符集

与声明DOCTYPE同样重要的是将HTML文档指定为某一种人类的语言。如果你曾偶然闯入某个看似乱码的网页，而实际上其文字却是日语、希腊语、斯瓦希里语或者其他什么语言，你就会知道页面编写者忘记设置页面编码所造成的后果了。因为Google和其他搜索引擎会根据搜索请求的语言来过滤搜索结果，所以如果你不想放弃搜索引擎带来的流量的话，那么正确地设置页面的编码将显得极为重要。本章稍后将详细介绍字符编码的相关内容。

示例：

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" >
```

设置文档的XML语言。这里设置为英语的ISO代码en。

```
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
```

设置文档的字符集。这里为UTF-8。

• 指定标题

每个HTML页面都需要一个用来描述页面内容的<title>元素。在访客将页面保存至收藏夹时，<title>元素中的内容将作为链接的描述同样被保存在访客的浏览器中。搜索引擎也对<title>元素中包含的关键字格外感兴趣，因此好的<title>可以明显地提高页面在搜索引擎中的排名。网站中的每个页面都应该尽可能地有一个惟的标题，标题应该极具描述性地概括该页面的内容，而不是泛泛地介绍整个站点。

示例：

```
<title>css Zen Garden: The Beauty in CSS Design</title>
```

• 选用恰当的元素

我们在前面已经分析了这个问题，但还是有必要在这里重复一遍：根据

文档内容的结构而选择HTML元素，而不是根据HTML元素的样式。例如，用p元素包含文字段落，而不是为了换行；用blockquote包含被引用的文字，而不是为了得到缩进等。

当然，并不是所有的HTML元素都可以在XHTML 1.0 Strict标准中使用的。事实上，虽然XHTML 1.0 Strict标准（以及XHTML 1.1标准）中可使用的HTML元素较少，但这些元素却有着更加明确的意义。若你在创建文档时找不到合适的、可以恰当描述其结构的标签，那么可以考虑使用通用的div或span元素。当然，这也是一个警告……接下来将会介绍。

示例：

```
<h3>The Road to Enlightenment</h3>
```

而不要使用这样的HTML：

```
<code style="font-size: 1.5em;">The Road to Enlightenment</code>
```

• 避免过度使用div和span

设计HTML时一个常见的错误就是过度使用div和span。少量、必要的合理使用可以明显地增强文档的结构性。但若是文档中使用了太多的div和span，那么你就应该考虑一下是不是还有更加合适的HTML元素供选择了。例如，若是h3更能表示内容的含义，那么就不应该使用div，span也不应该替代label的作用。

但这并不意味着div和span应避免使用。文档中合适位置的少量div元素可以使文档更易于应用样式，且在逻辑段落上也显得更加清晰。我们可以将div看作一个可以被重用的容器：将内容深深隐藏于太多的容器中并不是一个好主意，但是几个合理位置中的div却可以让文档显得井井有条。

示例：

```
<div id="pageHeader">
  <h1>css Zen Garden</h1>
   <h2>The Beauty of <acronym title="Cascading Style Sheets"> CSS</acronym> Design</h2>
</div>
```

Note

在W3Schools的“XHTML 1.0 Reference”（http://w3schools.com/xhtml/xhtml_reference.asp）中可以找到XHTML 1.0中允许使用的所有HTML元素。

• 尽可能少地使用标签

前面两个提示所隐藏的含义就是，HTML标签使用得越少越好——事实上也确实如此。若想创建出良好的文档结构，那么尽可能少地使用标签则应该是一个自始至终都要考虑的问题。仅仅使用必要的标签，将其他不必要的都删掉。这样做的好处不仅在于能够让文件保持小巧（也就意味着用户下载速度的提高），也可以让浏览器能够很快完成文档的解析，即得到更快地呈现速度。

示例：

```
<p>The Zen Garden aims to excite, inspire, and encourage participation.</p>
```

而不要这样：

```
<div>
  <p><span class="text">The Zen Garden aims to excite, inspire, and encourage participation.</span></p>
</div>
```

• 适当地使用class和id

为元素添加起标示作用的属性可以让你在稍后容易地将该元素与CSS或JavaScript关联起来。class是一类可重用的属性，能够被重复应用到到页面中的任何元素上；而id则是独一无二的，同一个id在每个页面上只能出现一次。

一个元素可以应用多个class，一个class也可以应用在多个页面元素上。我们也可以为同一个元素同时应用class和id，但要记住：在同一个页面上不能为多个元素指定相同的id。class和id可以选用字母或数字（a~z、A~Z、0~9）命名[1]，但必须以字母开头。虽然有些浏览器对class和id名称的大小写不敏感，但为了今后便于调试，还是小心为妙。

三个合法的标识符：

```
<body class="homepage">
<p id="introduction">
<div id="section5" class="top corner solid">
```

两个不合法的标识符：

[1] 下划线“_”也可以使用。——译者注

```
<span class="15pxhigh">
<div id="footer"><div id="footer">
```

3. 时间的检验

若曾编写过在未来相当长的一段时间内都要保持不变的文档，你就会发现在编写的过程中必须处处小心，让文档尽可能地完美。

以出版行业为例，为了保证语言流畅简洁和内容的正确性，这本书曾被很多人修改编辑过。原因很明显：与网页在浏览器缓存中存在的时间相比，书籍在书架中停留的时间要长得多。所有这些预先的检查工作都是为了确保书籍在出版时的正确性（但愿出版后好运相伴）。

但是在同样的问题上，Web却没有此类的限制。每次用户来访，Web站点都从服务器将内容发送给访客。这样就意味着，即使在网站发布之后，你仍旧可以修改页面的内容或重新设计页面的样式，每个再次访问的用户都能够立即看到这些变化，而并不需要像图书这样，只有重印或再版的时候才有机会修改。自然，网站的不断改进会让人非常高兴，但这也会让那些签了固定合同的设计师感到些许郁闷。

虽然禅意花园也是一个网站，但在编写HTML的过程中却遇到了一些独特的挑战，让这个过程更像是写书而不是网站。若是在用户已经提交了设计之后再修改页面的HTML，那么就有可能破坏掉从前的设计，这也与禅意花园的最初目标相悖。即使修改的仅仅是页面中的文本，也会对文档以及文档中某块内容的大小造成影响，并有可能无法预知地改变某些作品的布局。这样看来，一旦发布之后再对禅意花园的HTML进行任何结构或内容的修改都可能是有害且不可接受的，所以在编写HTML时就需要尽可能小心准确。

4. 禅意花园的HTML源代码

我们应该把禅意花园的HTML结构看作像书一样持久不变。虽然有数不清的可改善之处，但禅意花园的HTML结构仍然没有过大的修改。文档的内容可能会略加改进，但最基本的HTML元素以及文档结构则不会，就像被刻在石头上一样[1]。

为了便于本书后面章节参考，下面几页中完整地列出了禅意花园的HTML

[1] 禅意花园现在的HTML代码已经与接下来列出的有所不同，但在文档结构上仍没有任何改变。——译者注

源代码。你也可以访问禅意花园站点（www.csszengarden.com），并在浏览器中查看源文件来得到这份代码[1]。关于查看源文件的步骤，在大多数浏览器中都差不多：从工具条中选择View > View Source或其他最类似的选项即可。或者也可以通过禅意花园页面中的下载源代码链接得到其HTML文件。本书列出的代码删除了原版本中的注释，而改为在边栏对某些段落或条目给出了一些注解。若想看到原版本中的注释，请直接访问禅意花园站点。

指定用来标识文档类型的DOCTYPE是每个页面都必须的。请参考本章前面的“选择DOCTYPE”小节。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" >
<head>
```

在全球化的环境中（例如Web），为文档指定字符编码也极其重要。请参考本章前面的“指定语言和字符集”小节。若想了解更多关于字符编码的内容，请参考Joel Spolsky在Joel on Software网站上的文章：“The Absolute Minimum Every Software Developer Absolutely, Positively Must Know About Unicode and Character Sets (No Excuses!)”（www.joelonsoftware.com/articles/Unicode.html）。

```
    <meta http-equiv="content-type" content="text/html;
    charset=iso-8859-1" />
    <meta name="author" content="Dave Shea" />
    <meta name="keywords" content="design, css, cascading, style, sheets,
    xhtml, graphic design, w3c, web standards, visual, display" />
    <meta name="description" content="A demonstration of what can be
    accomplished visually through CSS-based design." />
    <meta name="robots" content="all" />

    <title>css Zen Garden: The Beauty in CSS Design</title>
```

这个空的script元素用来避免Windows下IE中可能出现的文档样式短暂失效问题。该问题在Blue Robot网站中的“Flash of Unstyled”一文（www.bluerobot.com/web/css/fouc.asp）中有详细介绍。

```
    <script type="text/javascript"></script>
```

这种方式用来让Netscape Navigator 4浏览器不要加载并应用外部样式文件，而是直接显示出简单的、未经CSS修饰过的HTML文档[2]。就目前来讲，这种方法已经不是那么必要了[3]。若想了解更多相关内容，请参考Eric Meyer on CSS网站中的“Tricking Browsers and Hiding Styles”一文（www.ericmeyeroncss.com/bonus/trick-hide.html）。

```
    <style type="text/css" media="all">
        @import "sample.css";
    </style>
</head>
```

[1] 指在IE加载页面时的一个缺陷：首先显示出没有应用任何样式的文档，随后页面闪烁一次并重新显示应用了样式的文档。——译者注

[2] 因为Netscape Navigator 4对CSS的支持非常不好，加载了CSS不但不能起到修饰页面样式的作用，甚至反倒会让页面难以辨认。——译者注

[3] 因为目前Netscape Navigator 4浏览器已经近乎绝迹。——译者注

```
<body id="css-zen-garden">
```

应用到body元素上的id叫做“CSS签名（CSS Signature）”。借助CSS签名的帮助，用户自定义的样式表可以覆盖页面中的部分甚至所有CSS样式。请参考“[css-d] CSS signatures”（http://archivist.incutio.com/viewlist/css-discuss/13291）。

```
<div id="container">
    <div id="intro">
        <div id="pageHeader">
            <h1><span>css Zen Garden</span></h1>
            <h2><span>The Beauty of <acronym title="Cascading Style
            Sheets">CSS</acronym> Design</span></h2>
        </div>
```

将页面中的元素按照逻辑上的相关程度分开放置于不同的div元素中，可以让后期应用样式变得更加简单灵活。

```
        <div id="quickSummary">
            <p class="p1"><span>A demonstration of what can be
            accomplished visually through <acronym title="Cascading
            Style Sheets">CSS</acronym>-based design. Select any style
            sheet from the list to load it into this page.</span></p>

            <p class="p2"><span>Download the sample <a href="zengarden-
            sample.html" title="This page's source HTML code, not to be
            modified.">html file</a> and <a href="zengarden-sample.css"
            title="This page's sample CSS, the file you may
            modify.">css file</a></span></p>
        </div>
```

让事情简单点，再简单点！虽然那些准备为禅意花园提交设计的朋友都应该知道如何下载这些文件（在链接上右键单击，选择Save As[1]），但这个方法却不是每个想要学习的人都了解的。仍不时有电子邮件询问如何下载这些文件，这也从一个侧面证明了你的目标群体中总是会有一些无法预料到的成员。

```
        <div id="preamble">
            <h3><span>The Road to Enlightenment</span></h3>
            <p class="p1"><span>Littering a dark and dreary road lay the
            past relics of browser-specific tags, incompatible <acronym
            title="Document Object Model">DOM</acronym>s, and broken
            <acronym title="Cascading Style Sheets">CSS</acronym>
            support.</span></p>
            <p class="p2"><span>Today, we must clear the mind of past
            practices. Web enlightenment has been achieved thanks to the
            tireless efforts of folk like the <acronym title="World Wide Web
            Consortium">W3C</acronym >, <acronym title="Web Standards
            Project">WaSP</acronym> and the major browser creators.</span></p>
```

事后诸葛地考虑，若是#intro div在这里结束，并将#preamble[2]置于#supportingText中，页面的结构无疑将变得更有意义。#preamble的内容在逻辑上更适合文章正文，而不是文章的介绍部分。从样式或布局的角度考虑，这样的改变也会增强其灵活性。

从语法和标记上都能看出来，W3C是一个技术上的简称（abbreviation），而不是取首字母的缩写词（acronym）。事实也正是如此，且HTML中专门为声明简称定义了abbr元素，虽然IE并不支持这个元素，但我们也可以使用IE支持的acronym元素来代替。

[1] 中文版浏览器中该选项一般是“另存为”。——译者注

[2] preamble的英文含义为“导言”。——译者注

```
            <p class="p3"><span>The css Zen Garden invites you to relax and
            meditate on the important lessons of the masters. Begin to see with
            clarity. Learn to use the (yet to be) time-honored techniques in
            new and invigorating fashion. Become one with the web. </span></p>
        </div>
    </div>

    <div id="supportingText">
        <div id="explanation">
            <h3><span>So What is This About?</span></h3>
            <p class="p1"><span>There is clearly a need for <acronym
            title="Cascading Style Sheets">CSS</acronym> to be taken seriously
            by graphic artists. The Zen Garden aims to excite, inspire, and
            encourage participation. To begin, view some of the existing
            designs in the list. Clicking on any one will load the style sheet
            into this very page. The code remains the same, the only thing th-
            at has changed is the external .css file. Yes, really.</span></p>

            <p class="p2"><span><acronym title="Cascading Style
            Sheets">CSS</acronym> allows complete and total control over
            the style of a hypertext document. The only way this can be
            illus-trated in a way that gets people excited is by
            demonstrating wh-at it can truly be, once the reins are placed
            in the hands of those able to create beauty from structure. To
            date, most examp-les of neat tricks and hacks have been
            demonstrated by structurists and coders. Designers have yet to
            make their mark. This needs to change.</span></p>
        </div>

        <div id="participation">
            <h3><span>Participation</span></h3>
            <p class="p1"><span>Graphic artists only please. You are modifying
            this page, so strong <acronym title="Cascading Style Sheets"> CSS<
            /acronym> skills are necessary, but the example files are
            commented well enough that even <acronym title="Cascading Style
            Sheets">CSS</acronym> novices can use them as starting points.
            Please see the <a href="http://www.mezzoblue.com/zengarden
            /resources/" title="A listing of CSS-related resources"> <acronym
            title="Cascading Style Sheets">CSS</acronym> Resource Guide</a>
            for advanced tutorials and tips on working with <acronym title="
            Cascading Style Sheets">CSS</acronym>.</span></p>
            <p class="p2"><span>You may modify the style sheet in any way you
            wish, but not the <acronym title="HyperText Markup
            Language">HTML</acronym>. This may seem daunting at first if you
            &#8217;ve never worked this way before, but follow the listed link s
            to learn more, and use the sample files as a guide.</span></p>
            <p class="p3"><span>Download the sample <a href="zengarden-
            sample.html" title="This page's source HTML code, not to be
            modified.">html file</a> and <a href="zengarden-sample.css"
```

我们并不知道“结构至上者(structurist)”描述的是哪一类人。在当初编写页面内容时似乎有一种感觉，那就我们是应该把那些编写代码的人和那些注重于HTML结构的人区别开来。现在看来，其原因已经无从知晓了。

’表示什么？它是一个起到替代作用的字符编码，表示印刷中用到的朝向左边的单引号(’)，而不是我们常用的普通单引号(')。Web页面能够显示很多种印刷符号和习惯用法，可在任何可能的地方使用。请参考A List Apart上的“The Trouble With EM ‘n EN (and Other Shady Characters)”一文(http://www.alistapart.com/articles/emen)。

```
    title="This page's sample CSS, the file you may modify.">css
    file</a> to work on a copy locally. Once you have completed your
    masterpiece (and please, don&#8217;t submit half-finished work)
    upload your .css file to a web server under your control. <a
    href="http://www.mezzoblue.com/zengarden/submit/" title="Use the
    contact form to send us your CSS file">Send us a link</a> to the
    file and if we choose to use it, we will spider the associated
    images. Final submissions will be placed on our server.</span></p>
</div>

<div id="benefits">
    <h3><span>Benefits</span></h3>
    <p class="p1"><span>Why participate? For recognition,
    inspiration, and a resource we can all refer to when making
    the case for <acronym title="Cascading Style Sheets">
    CSS</acronym>-based design. This is sorely needed, even
    today. More and more major sites are taking the leap, but
    not enough have. One day this gallery will be a historical
    curiosity; that day is not today.</span></p>
</div>

<div id="requirements">
    <h3><span>Requirements</span></h3>
    <p class="p1"><span>We would like to see as much <acronym
    title="Cascading Style Sheets, version 1">CSS1</acronym> as
    possible. <acronym title="Cascading Style Sheets, version
    2">CSS2</acronym> should be limited to widely-supported elements
    only. The css Zen Garden is about functional, practical <acronym
    title="Cascading Style Sheets">CSS</acronym> and not the latest
    bleeding-edge tricks viewable by 2% of the browsing public. The
    only real requirement we have is that your <acronym title
    ="Cascading Style Sheets">CSS</acronym> validates.</span></p>
    <p class="p2"><span>Unfortunately, designing this way highlights
    the flaws in the various implementations of <acronym
    title="Cascading Style Sheets">CSS</acronym>. Different browsers
    display differently, even completely valid <acronym title
    ="Cascading Style Sheets">CSS</acronym> at times, and this becomes
    maddening when a fix for one leads to breakage in another. View
    the <a href="http://www.mezzoblue.com/zengarden/resources/" title
    ="A listing of CSS-related resources">Resources</a> page for
    information on some of the fixes available. Full browser compl-
    iance is still sometimes a pipe dream, and we do not expect you to
    come up with pixel-perfect code across every platform. But do test
    in as many as you can. If your design doesn&#8217;t work in at
    least IE5+/Win and Mozilla (run by over 90% of the population),
    chances are we won&#8217;t accept it.</span></p>
    <p class="p3"><span>We ask that you submit original artwork.
    Please respect copyright laws. Please keep objectionable
    material to a minimum; tasteful nudity is acceptable,
```

由于提交量过多，对提交作品的要求有所改变。在提交作品之前，请参考mezzoblue.com上最新版本的“Submission Guidelines”（www.mezzoblue.com/zengarden/submit/guidelines）。

目前已经有了一些方法，在使用某些高级CSS效果的同时，也略微减少了专为低版本浏览器设计的样式，这些将在第7章中介绍。在一些设计中，该方法已经被付诸实践，所以这个对浏览器支持的需求或许可以改为“使用自己的最佳判断”。

在将某个作品添加到禅意花园之前，通常都要在几个不同的浏览器中检查一下，设计者自然也应该对此项检查负责。但若你还是发现某个作品的样式不正常，那么也不必太过计较——毕竟作品的征集是全世界范围内的，设计者可能接触不到您所用的浏览器或操作系统。

```
outright pornography will be rejected.</span></p>
<p class="p4"><span>This is a learning exercise as well as a
demonstration. You retain full copyright on your graphics (with
limited exceptions, see <a href="http://www.mezzoblue.com/
zengarden/submit/guidelines/">submission guidelines</a>), but we
ask you release your <acronym title="Cascading Style
Sheets">CSS</acronym> under a Creative Commons license identical
to the <a href="http://creativecommons.org/licenses/sa/1.0/"
title="View the Zen Garden's license information.">one on this
site</a> so that others may learn from your work.</span></p>
<p class="p5"><span>Bandwidth graciously donated by <a
href="http://www.dreamfirestudios.com/">DreamFire
Studios</a>. Get the <a href="/book/"> Book </a>!
</span></p>
</div>

<div id="footer">
<a href="http://validator.w3.org/check/referer" title="Check
the validity of this site&#8217;s XHTML">xhtml</a>  
<a href="http://jigsaw.w3.org/css-validator/check/referer"
title="Check the validity of this site&#8217;s CSS">css</a>
 
<a href="http://creativecommons.org/licenses/by-nc-sa/1.0/"
title="View details of the license of this site, courtesy
of Creative Commons.">cc</a>  
<a href="http://bobby.watchfire.com/bobby/bobbyServlet?URL
=http%3A%2F%2F www.mezzoblue.com%2Fzengarden%2F&amp;output
=Submit&amp;gl=sec508&amp;test=" title="Check the accessi-bility
of this site according to U.S. Section 508">508</a>  
<a href="http://bobby.watchfire.com/bobby/bobbyServlet?
URL=http%3A%2F%2F www.mezzoblue.com%2Fzengarden%2F&amp;
output=Submit&amp;gl=wcag1-aaa&amp;test=" title="Check the
accessibility of this site according to WAI Content
Accessibility Guidelines 1">aaa</a>
</div>
</div>

<div id="linkList">
<div id="linkList2">
<div id="lselect">
<h3 class="select"><span>Select a Design:</span></h3>
<ul>
<li><a href="/" title="AccessKey:a" accesskey ="a">Sample
#1</a> by <a href="http://www .mezzoblue .com/" class=
"c"> Dave Shea</a>  </li>
```

#footer中的这些链接是个列表……难道不应该用ul来组织吗？这是个反问句——这些链接确实应该包含在某个列表元素中。

在禅意花园最初版本的HTML代码中，包含链接的元素是span，而不是现在的li。但在使用某些自动化的工具或服务（例如HiSoftware的Cynthia Says, www.cynthiasays.com）检查页面的可访问性（accessibility）时会导致检测失败。每个链接结尾部分的那个额外的 [1]就是为了尝试解决这个问题而添加的，但事实证明这样做只是徒劳。好在我们很快悬崖勒马，用列表元素代替了原本的span，但这些 却在不经意间流传了下来[2]。

[1] 在页面中显示为空格。——译者注

[2] 在最新版本的禅意花园HTML代码中，该 已经被移除。——译者注

```
        <li><a href="/" title="AccessKey: b" accesskey
        ="b">Sample #2</a> by <a href="http://www.mezzob-
        lue.com/" class="c">Dave Shea</a>  </li>
        <li><a href="/" title="AccessKey: c" accesskey
        ="c">Sample #3</a> by <a href="http://www.mezzo-
        blue.com/" class="c">Dave Shea</a>  </li>
        <li><a href="/" title="AccessKey: d" accesskey
        ="d">Sample #4</a> by <a href="http://www.mezzo-
        blue.com/" class="c">Dave Shea</a>   </li>
        <li><a href="/" title="AccessKey: e" accesskey
        ="e">Sample #5</a> by <a href="http: //www .mezzob-
        lue.com/" class="c">Dave Shea</a>   </li>
        <li><a href="/" title="AccessKey: f" accesskey
        ="f">Sample #6</a> by <a href="http: //www.mezzob-
        lue.com/" class="c">Dave Shea</a>  </li>
        <li><a href="/" title="AccessKey: g" accesskey
        ="g">Sample #7</a> by <a href="http: //www.mezzo-
        blue.com/" class="c">Dave Shea</a>  </li>
        <li><a href="/" title="AccessKey: h" accesskey
        ="h">Sample #8</a> by <a href="http:// www.mezzo-
        blue.com/" class="c">Dave Shea</a>  </li>
      </ul>
    </div>

    <div id="larchives">
      <h3 class="archives"><span>Archives:</span></h3>
      <ul>
        <li><a href="/" title="View next set of designs.
        AccessKey: n" accesskey="n"> <span class="access-
        key">n</span>ext designs &raquo;</a> </li>
        <li><a href="/" title="View previous set of
        designs.AccessKey: p" accesskey="p">&laquo; <span
        class="accesskey">p</span>revious designs</a></li>
        <li><a href="http://www.mezzoblue.com/zengarden
        /alldesigns/" title="View every submission to the
        Zen Garden. AccessKey: w" accesskey="w">Vie<span
        class="accesskey">w</span> All Designs</a></li>
      </ul>
    </div>

    <div id="lresources">
      <h3 class="resources"><span>Resources:</span></h3>
      <ul>
        <li><a href="http://www.csszengarden.com/001/001 .css"
        title="View the source CSS file for the currently-
        viewed design, AccessKey: v" accesskey= "v"><span
        class="accesskey">V</span>iew This Design&#8217;s
```

快捷键（Accesskey）是一项起辅助作用的技术，用来让浏览者更加容易地在页面中导航，但快捷键也并非是完美无瑕的。请参考WATS.ca上的《Using Accesskeys–Is it worth it?》这篇文章（www.wats.ca/articles/accesskeys/19）。

用这种方式在作品之间导航似乎非常可笑，现在我们已经有了超过100件的作品，将这个链接指向一个外部的列表或许会更加有效(www.mezzoblue.com/zengarden/alldesigns)[1]。但很多人都没注意到这个链接，他们以为禅意花园只有8个作品，且从来没有看到过其他作品。不要犯同样的错误哦！

虽然“察看源代码”的方法一直存在于各个浏览器中，但还是有不少的初学者发邮件询问如何察看某个作品的CSS文件。所以很快我们又添加了这个链接。

[1]在最新版本的禅意花园HTML代码中，该链接已经指向了该站点。——译者注

```
                    <acronym title="Cascading Style Sheets">CSS</acronym>
                    </a></li>
                    <li><a href="http://www.mezzoblue.com/zengarden
                    /resources/" title="Links to great sites with
                    information on using CSS. AccessKey: r"
                    accesskey="r"><acronym title="Cascading Style
                    Sheets">CSS</acronym> <span class="accesskey">
                    R</span>esources</a></li>
                    <li><a href="http://www.mezzoblue.com/zengarden
                    /faq/" title="A list of Frequently Asked
                    Questions about the Zen Garden. AccessKey: q"
                    accesskey="q"><acronym title="Frequently Asked
                    Questions">FA<span class="accesskey">Q</span>
                    </acronym></a></li>
                    <li><a href="http://www.mezzoblue.com/zengarden
                    /submit/" title="Send in your own CSS file.
                    AccessKey: s" accesskey="s"><span class="
                    accesskey"> S</span>ubmit a Design</a></li>
                    <li><a href="http://www.mezzoblue.com /zengarden
                    /transl ations/" title="View translated version
                    sof this page. AccessKey: t" accesskey="t"><span
                    class="accesskey">T</span> ranslations</a></li>
                </ul>
            </div>
        </div>
    </div>
</div>
<div id="extraDiv1"><span></span></div>
<div id="extraDiv2"><span></span></div>
<div id="extraDiv3"><span></span></div>
<div id="extraDiv4"><span></span></div>
<div id="extraDiv5"><span></span></div>
<div id="extraDiv6"><span></span></div>

</body>
</html>
```

这些完全空白的div和span可以用来在页面上添加原本不在其中的图片，稍后可以看到一些实例。我们可以将其看作一类可重用的img元素。

5. HTML结构的图示

设计师Andy Clarke为禅意花园HTML结构创建了一个图形化的示意图（图1-1-14）。这个示意图清晰地显示了各个元素之间的嵌套关系，如果想学习禅意花园的HTML结构，或是想为禅意花园设计新的样式，那么这个图示无疑是个很好的开始。请参考Clarke的说明文档“3D CSS Zen Garden”（www.stuffandnonsense.co.uk/archives/3d_css_zen_garden.ht ml）。

6. 提供足够的灵活性

从一开始，我们编写这套HTML的一个主要目标就是为各个设计的变化提供足够的灵活性支持。为禅意花园的HTML提供尽可能多的额外信息是非常必要的，这样才能让页面中的每个元素都能依照需要被修改和调整。如果没有这些额外的信息，选择页面中的某个特定元素将变得很难，甚至不可能实现，虽然使用后代选择器（descendant selector）可以完成某一类元素的选择，例如：

```
#quickSummary p {
    color: red;
}
```

但按照上述CSS的定义，#quickSummary中所有的p元素都将变为红色。如果我们只想修改#quickSummary中的第二个段落，那又将如何实现呢？除了为#quickSummary中的所有段落都添加独一无二的class之外，我们只能选用更加高级的CSS2选择器：

```
#quickSummary > p + p {
    color: red;
}
#quickSummary > p + p + p {
    color: inherit;
}
```

这段代码似乎已经有些难以理解了。第一条规则将子选择器（child selector）和邻接选择器（adjacent selector）组合了起来，表示的含义为："若某个段落元素[1]紧跟着另外一个段落元素出现，且该段落元素还定义于#quickSummary中，则为其应用该样式"。这样就意味着除了第一段之外，#quickSummary中所有的其他段落都将应用该样式。接下来又用第二条规则覆盖了第一条中的定义，为紧跟着两个或两个以上段落后出现的段落恢复了其默认样式，当然这些段落也必须同样在#quickSummary中，即除了前两段之外，#quickSummary中所有的其他段落都将应用该样式。

天啊！只是要选择一个元素而已，居然弄出了这么冗长的代码。谢天谢地，在CSS3中我们有了一个简单的方法：

```
#quickSummary p:nth-child(2) {
    color: red;
}
```

[1] 这里指HTML的p标签。——译者注

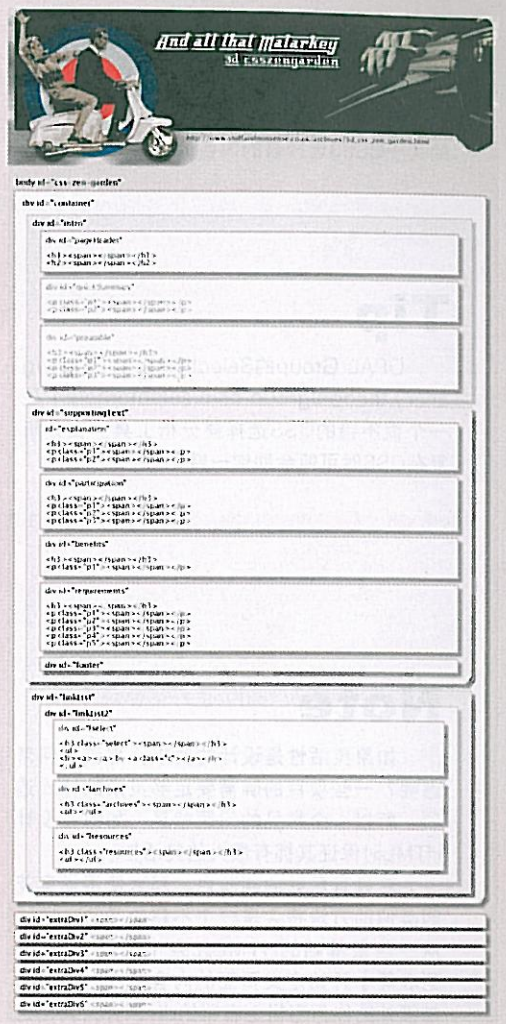


图1-1-14　禅意花园HTML结构的图形化示意图，由Andy Clarke编写

Tip

在W3C的"5 Selectors"（www.w3.org/TR/CSS21/selector.html）中可以了解到更多关于CSS2选择器的内容。

Tip

在W3C的“Selectors”（www.w3.org/TR/css3-selectors）中可以了解到更多的关于CSS3选择器的内容。

Tip

OPAL Group的SelectORacle（http://gallery.theopalgroup.com/selectoracle）是一个很不错的CSS选择器分析工具。在分析复杂CSS时可能会助你一臂之力。

Note

如果灵活性是设计的要求，那该如何考虑呢？一些项目的确需要足够灵活的样式选项，所以一个常见的问题就是，如何在设计HTML时保证其拥有最大的灵活性。

若具有足够的判断力，那么使用禅意花园类似的方法将会是一个不错的开始。建议从一些最基础的HTML元素（段落、标题、列表等）开始定义页面的内容。在必要时添加额外的作为逻辑上容器的div，并为其指定唯一的id。如果需要的话，适当且最小限度地为其中的元素指定class。禅意花园中用到的这些span基本上不是必要的，除非有极特别的需求，否则不要这样使用。

但是也别高兴得太早了，有件事情你必须知道：这两种方法IE都不能支持[1]！除非你有什么实在特殊的理由，以至于可以放弃绝大多数浏览者，否则你可以先把这两种方法忘掉了。IE的使用者比例可能将在几年之后有所变化，但在本书出版时它的市场占有率仍在90%左右。

上面这一长段讨论其实只是为了选择到一个简单的元素。从这段讨论中我们也可以明显地看出，为了保证灵活性，给元素添加额外class是非常必要的。由于类似的原因，我们也添加一些额外的span标签。在页面的段落中声明内部元素，也就意味着在设计时我们可以为这些元素应用某些额外的样式。

• 图像替换

这些额外的span标签还有个用途，那就是图像替换。这是为设计禅意花园的HTML时CSS设计领域中流行的一种全新的方法所准备的。让我们举例说明这个方法：若你想用图片替换页面上的某一个标题（例如一个h3标签）中的文字，那么就必须用两个标签（一个h3和一个span）将文字包围起来。

然后我们可以用display: none之类的规则将span中的文字隐藏起来，并为h3应用背景图像，这样一个带有丰富图形效果的标题就做好了。因为禅意花园中所有的元素中都包含了这样的一个span，所以这种形式的扩展也就非常易于应用。

但为什么宁可要这么麻烦，也不使用同样能实现该功能的img元素呢？同样也是出于灵活性方面的考虑。若是将图片的地址直接写入到HTML中，就不可能只通过替换样式文件让页面来个“大变脸”了。但这个图像替换的方法也存在着一些问题，将在本书稍后部分介绍。

• Lighten Up（“减肥”）

这些决定及实现方式都是基于当时大家认同的最佳设计方案。与最近出现的一些技术相比，这种实现方式的代码量偏多，现在回顾起来，不难发现很多可改进的地方。

禅意花园发布不久，一种新的、不需要额外span的图像替换方法横空出世，因此HTML文档中的大部分span自然也受到了冷落。98%的提交作品中都没有使用这些额外的span，所以基本上可以认为这些span是多余的。除了被某些设计师用于实现一些极为特殊的视觉效果之外，文档结尾部分的那些

[1] 最新发布的IE7已经支持了子选择器和邻接选择器。—译者注

空的div和span也没有太大的用处。

对于其他网站来说，文档结尾部分的那些空的div和span并没有什么意义。但在禅意花园中它们却有存在的必要，即为了实现最大的灵活性。因为并不是每一个网站都需要如此多样的设计，所以在为这些编写HTML时也不必考虑这些。

若你决定以禅意花园的HTML为基础开始编写页面（我们祝愿你的老板会同意你这样做），请记住基于标准的设计中很重要的一个目标就是避免不必要的元素。偶尔使用div元素将一些逻辑上相关的元素包围起来并没什么问题，但若是把页面中的每一个段落或标题都用span元素包围起来，那么就有些过度了。关于如何正确、合理地组织文档结构，将在本章稍后讨论。

1.1.3 学到的教训

禅意花园的主要目的是为了演示基于标准进行设计所能实现的效果。通过浏览这些种类繁多的作品，访客也能体会到CSS所提供的强大的灵活性。就其自身来讲，CSS是一种用来增强某种基本标记语言（例如HTML、XHTML或XML）的设计性语言。CSS1在1996年被提出，很快W3C又在1998年提出了CSS2标准。各种浏览器也开始逐步支持这两种标准，到了1998年，Web设计师普遍采用链接外部CSS文件的方式来部分地控制页面的样式。当时几乎所有人都这样做，直到4、5年之后的今天。

为什么设定字体时我们都会选用CSS？原因很简单，能在所有浏览器中有一致表现的，也几乎只有字体这一种CSS设定了。虽然效果并非总是如此理想，但使用CSS定义字体还有一种非常独特的优势，这种优势让很多设计师最终转向了CSS。

这个优势在于，某个单一的CSS文件可能被站点中所有的文档所引用。若在这个CSS文件中声明了字体，那么所有的页面都会应用这一设定。若你想对字体或字号进行修改，那么只要更新一下该CSS文件中的定义即可，站点中所有的页面都会立即被更新——大功告成，就这么简单。

相对于从前的<font>标签，这种方法有了明显的进步：那个声名狼藉的<font>标签是文档级别的、用于设定文字样式的元素，使用时需要将其应用于站点中的每一个实例上。最终几百个页面中的每一页上都可能会有7、8个<font>标签，若是这时想修改，想必过程一定非常耗时且痛苦吧。

Note

W3C的HTML验证器在W3C的"Validate Your Markup"页面中(http://validator.w3.org)。

Note

只有使用了外部的CSS文件，才会享受到节省带宽带来的好处。我们可以将CSS嵌入到HTML文档中，但这样的话浏览器将不能像对待外部CSS文件一样对其进行缓存，你也会由此失掉一些缓存带来的好处。

这个样式方面的分离只不过是CSS所带来的诸多优势中的冰山一角。一个简单的重新设计要求就让实现的过程有了数量级上的简化。从以前的方案迁移过来也同样简单：无须在众多文件中替换成千上万个<font>标签，只要将HTML内容拷贝过来，然后声明相应的CSS即可。

但<font>标签还是被使用了很长时间。一些较老的浏览器对比较高级的CSS样式的支持并不好，这些浏览器过了几年之后才逐渐消失。为了照顾这些浏览器，在这段时间之内使用CSS还是有一些不切实际。

到了2000年之后，浏览器对CSS的最小支持普遍有所提高，已经足够实现完全的CSS布局了。虽然当时浏览器对CSS的支持还不甚完美（现在也依旧如此），但这已经足够让CSS更进一步发挥，而不只是用来控制页面的字体。在经过了几年通力合作之后，全球范围内的Web设计师给出了一些足够可靠的使用CSS进行页面布局的方法，包括使用浮动（floating）或绝对定位模型进行布局，以及对非标准浏览器的一些临时补救措施等。

这就是我们现在所面对的情况。直到2005年，仍有一些浏览器在解析1998年就已经出台了的CSS2标准时还存在问题。但即便如此，主流浏览器对CSS的支持在大体上已经足够完善。很多网站都转向使用CSS且基于标准进行设计，设计者也自然享受到了基于标准所带来的好处。

接下来让我们简要总结一下这些好处。既然您已经在阅读本书，那么很可能您已经认识这些好处。如果还想更深入地了解一些细节，那么请参考本书结语中的推荐阅读列表。

基于Web标准开发的好处

基于Web标准的设计要同时使用CSS和结构化HTML。二者相互依存，很难将其带来的优势分割开来。我们只有用HTML结构化地描述文档的结构，同时使用CSS展现文档的样式，才会从Web标准中真正受益。

禅意花园的各种不同设计证明，只要为其应用不同的CSS，HTML页面的布局样式可以被随心所欲地修改。CSS正是一种用来展现设计的方法，我们只需将其与原始的HTML文档结构关联起来即可。虽然CSS依赖于HTML结构并建立于其上，但却能够相对独立地为其提供丰富的表现样式。

但非常重要的一点是，HTML文档必须有一个良好组织且合法的结构。一个非法的HTML文档可能会在浏览器中呈现的样式是不可预料的，且可能会导致很多本可以避免的麻烦。若的CSS布局出了问题，那么首先要检查的

就是HTML文档的合法性。

• 更小的文件、更快的下载速度

将样式相关的标记从HTML移至CSS之后，通常文件的大小也会降低。很有可能出现的情况就是，一张混杂着内容和表现的30K大小的HTML文档，完全可以用一个10K的HTML加上一个10K或不到10K的CSS文件代替。

除了将在最初降低文件大小之外，CSS文档还能够被缓存于浏览器中。考虑如果要添加另外一些页面，每一个都有30K，那么用户要浏览网站中的每个页面都必须从服务器下载30K的内容。每一次浏览都是如此，无论页面中实际有多少内容。若是访问了这样站点中的10个页面，则用户需要下载300K的内容，而其中大部分都是不必要的。

但若是将表现样式移至单独的CSS文件，那么不仅最终的页面大小将降低，每个页面共用的CSS文件也将不必被重复下载。这样用户访问某张页面时只要下载10K的页面内容即可，而另外10K的CSS文件可以直接从浏览器的缓存中得到。前面例子中的300K的总下载量在这里将减至110K。

无论对服务器还是对最终用户来讲，这种文件大小的降低有非常显著的效果。当站点的流量非常大时，这些省下来的字节将极大地降低你的带宽需求。运动主题的ESPN网站从2003年早些时候开始切换至CSS，这个数字可能并不准确。在对ESPN.com的开发者进行的一次采访中（采访内容最初发布在Netscape公司的DevEdge网站上。该网站目前已经停止运营，现在请参考这个地址：www.mikeindustries.com/blo-g/archive/2003 /06/espn-interview），该开发者估计ESPN网站每日可以由此节省大概2TB的流量！这就意味着ESPN在无需任何额外的硬件投资情况下就可以支持更多的并发访问人数，且响应速度也将有所提高。的确是一本万利！

• 增强对移动设备的支持

实际上，一个正确、合理组织的页面甚至并不需要CSS来辅助呈现。在某些场景中这一点非常重要：当你为一个足够大的1024×768的屏幕设计网页时，将有大量的空间可以随意发挥；但对于一个小巧的、240×320的掌上电脑来说，情况却大为不同。

即使没有了CSS的修饰，原始的文档内容也十分易读：超链接仍然可以点击，表单元素也可以正常使用。无需任何额外工作，结构化的HTML文档

Note

W3C的“7 Media Types”页面（www.w3.org/TR/CSS21/media.html）中介绍了一些CSS媒介类型的概要信息。

Note

W3C的“内容可访问性指导建议（WCAG）”可以从这个地址访问：www.w3.org/TR/WAI-WEBCONTENT。

就自动拥有在移动设备上正常显示的能力，这意味着可以免费得到网站的移动设备版本。

但CSS仍旧为多种显示设备提供了支持——你可以为某种显示设备指定不同的、专有的CSS样式。计算机屏幕、移动设备（例如移动电话）以及打印机都有它们自身的媒体类型（media type），且对更多的媒体类型的支持也正在研发中。

专门用于打印机设备的CSS宣告了那个小小的“打印该页面”链接的死亡。只要专门为打印机编写一个CSS文件，网站中的任何页面都可以直接打印出来。这个CSS文件能够有选择地格式化一些内容。隐藏另外一些条目，并专门根据打印机设备的特性进行适当的优化，以期得到最佳的打印效果。由于打印机与显示器相比所受到的限制更多，所以专门为打印机设备设计的CSS将变得非常有用。

理论上总会有这么一天，所有的设备在显示网页时都将依赖于自己的媒介类型。但现状却是，除了屏幕之外，很少有人使用其他类型的显示设备。对于移动电话来说，CSS的控制范围一般只局限于文本大小和颜色上；而用语言进行“浏览”的浏览器，即“听觉”设备类型还尚未出现。

这是个先有鸡还是先有蛋的问题，因为除非已经拥有了足够多的内容，否则该设备将不会得到更多的CSS支持。但同时又没有人为一些甚至并不存在的设备编写内容。当然，这种状态也在缓慢改进中，或许某一天将出现一个专为移动户电话和PDA编写的移动版本的禅意花园。

• 更好的可访问性

正确、合理地组织的HTML本身就具有良好的可访问性。如果用有效的标记编写页面，那么该页面的90%内容都将符合W3C的“内容可访问性指导建议（content-accessibility guidelines）”。

和那种只等同于文本的内容，还有多个版本的、维护起来极其痛苦困难的网站说再见吧，让我们来拥抱这种“一次编写，到处适用”的页面编写方式！在大多数情况下，即使是那些视力不佳而需要用屏幕阅读器（屏幕阅读器将朗读出网站的内容）的用户也同样可以访问你的站点，而这并不需要什么额外的工作。

当然，这其中也有一些需要注意的地方。本章稍后部分将介绍一些可能

由CSS引入的可访问性问题。

• 精确的控制

作为一门专注于文档表现样式的语言，CSS提供了很多极为丰富的、全新的功能，而这些功能是HTML所无法企及的。例如对行间距的精确控制（line-height属性）、对元素之间的距离的精确控制（margin和padding属性）和指定元素在页面中的绝对/相对位置（absolute和relative属性）等。

如果你曾受限于基于表格的布局所强加的网格，那么使用CSS进行定位则可以将表格一脚踢开，让你随心所欲地将元素置于页面中的任何位置上。如果你曾期待能对元素的背景有更加精确的控制，而不是只能选择某种颜色或是某个单一、重复的背景图片，那么CSS所提供的background属性将给你带来一些小小的启示。

1.1.4 学到的教训

没有任何网站是完美的，包括禅意花园在内。虽然现在亡羊补牢已经太晚了，但过去几年中相关讨论所揭示出的一些额外的知识却仍旧值得分享。

1. 可访问性检验

编写完成之后，我们使用Bobby[1]对禅意花园的HTML进行了一次快速的可访问性检验，并确认它已经满足了大多数可访问性要求。当然，在正式发布之前还必须修复由Bobby发现的一些小问题。禅意花园页脚部分添加的“AAA[2]”链接也表明了该页面在设计时曾考虑过这个问题。

真的考虑过么？似乎Bobby的检查并不是十分严格的。若熟悉W3C的“内容可访问性指导建议”（请参考本章前面部分的“更好的可访问性”小节），将会发现有些条款Bobby并不能检查。例如条款2.1，“在颜色不可用时，用户应该仍然能够得到那些由颜色传达的信息”，类似Bobby的软件无论如何也不能检查到这一点，特别是当信息和颜色被嵌入到图片中时。

---

[1] Bobby（http://webxact.watchfire.com/）是一个用来检查Web页面是否满足可访问性的软件。——译者注

[2] 按照W3C对页面可访问性的分级，AAA是最高一级，达到该级别的页面即表示其通过了W3C最严格的可访问性检验。请参考：http://www.w3.org/TR/WAI-WEBCONTENT/#wc-priority-3。——译者注

Note

Bobby（http://bobby.watchfire.com/）是一个在线的页面可访问性检查工具。例如Cynthia Says（www.cynthiasays.com）和WebAIM's Wave（http://wave.webaim.org/）等一些其他工具也提供了同样的功能。

实际上，在Bobby的网站上就有这样一段声明：

“可访问性在根本上还是由人决定的。带有各种不同程度残疾的人是否都能够有效地获取页面中的信息，才是可访问性问题的关键所在。Bobby仅仅是在提高页面的可访问性工作中的一步而已，而并不能保证其百分之百满足可访问性要求。”

——常见问题

http://bobby.watchfire.com/bobby/html/en/faq.jsp#faq_onetool

这样，禅意花园的HTML源代码在理论上通过了所有有关标记的可访问性检查，但可访问性的要求还不止这些。有些条款甚至还对CSS有所限制，但一段时间之后，我们可以很明显地发现，一些作品并没有考虑到这些问题。考虑一下Fahrner图像替换（将在第4章中详细介绍）对可访问性的影响，我们很容易就能看出，任何一个自动化的可访问性检查软件都没法发现这个由CSS带来的问题。

从中学到的教训就是，在创建具有良好可访问性页面时，使用类似Bobby的自动化工具可以帮我们迈出良好的第一步。但“内容可访问性指导建议”还包含很多同等重要的条款，对于这些条款的检查却并不能由此类工具代劳。

2. 文本的可缩放性

一个比较独特的、由CSS导致的可访问性问题就是文本的可缩放性。在编写网页时，我们经常会倾向于选择较小的字号，以便能够一次在页面中显示更多的内容。总之，我们都清楚自己在设计时有多么的“寸土必争”，所以也很有理由相信网页中文本的字号仍将越变越小。

问题在于，很多人又恰好看不清低于某个字号的文字。例如某些人无法辨认12像素高的文字，在某些极端的情况下，甚至24像素也还嫌小。当然，在现在的低分辨率[1]显示器的条件下，没有人会在开始设计时就选择24像素作为主要文字大小[2]。

幸运的是，当前所有浏览器都提供了内建的对文本进行缩放的机制。虽然不是所有的用户都知道如何使用，但对于那些知道的用户，这个功能却是无价

[1] 这里的“分辨率”指显示设备的单位长度/面积中像素点的数目，而并不是指屏幕整体“分辨率”（例如102×768等）。——译者注

[2] 在目前“Web 2.0”的浪潮中，很多网站也会反其道而行之，在正文中使用极大的字号，以方便用户把握其主要内容。但这样大字号的设计的主要目的并不是用来增强可访问性，而是考虑到了用户心理学。——译者注

之宝。但问题也随之而来：IE只能对用em或%为单位指定的文本进行缩放，若字号大小采用px设定，那么IE将对此无计可施。

对于这种情况，浏览器中内建的文本缩放机制将如鸡肋一般食之无味，字号的问题也仍然无法解决。但若是设计师不用px单位，那么就万事大吉了么？或许是这样的吧，但在本书第5章中也介绍了一些完全不同的、使用em或%单位造成的问题。

即使文本大小在任何浏览器中都可以正常缩放，我们还是要考虑很多问题：作品自身必须允许缩放（图1-1-15）。听起来似乎很简单，但若是将固定大小的元素（例如一幅20像素高度的背景图片）和可变大小的元素（例如一段文本，用户将其原始大小14px改变为了24px）放在了一起，你将很快发现在进行文本缩放时，这个依赖于固定大小的设计将出乎意料地显示得一团糟。大多数禅意花园的设计都通过了文本缩放测试，其中的文本都可以被调整至很大的字号。

从中学到的教训就是，无论你将文本设置为多大，有些用户仍然想要设置得再大一点。所以我们要特别注意并清楚地认识到当用户对文本进行缩放时可能遇到的潜在问题，且我们需要在合理范围之内对该问题有所准备。

3. XHTML和MIME类型

禅意花园是基于XHTML 1.1标准发布的。既然禅意花园的HTML代码通过了XHTML 1.1验证，有什么理由不选用XHTML 1.1来代替老版本的XHTML 1.0 Strict呢？

虽然如此，但XHTML却远远不止“完全将标签闭合”这一点要求。即使XHTML通过了验证且整篇文档都完全合法，若是不为其指定正确的MIME类型，从技术角度上说它仍不是一个真正的XHTML文档。

MIME是一种用来描述信息的标准，在这里介绍MIME的唯一原因就是XHTML的正确MIME类型为application/xhtml+xml。当然，也可以使用更加常见的text/html（HTML的默认值），但只有XHTML需要向下兼容的时候才应该这样做。千万不要把XHTML 1.1文档的MIME类型指定为text/html，XHTML 1.1文档始终应该拥有XML类的MIME类型。

为XML文档制定正确的MIME类型，就意味着已经以编程的方式改变了服务器发送的HTTP头部信息（通过.htaccess或httpd.conf文件）。如果您对上面这句话一头雾水，那也不是您的错。对于专注于设计的人来说，这些都是非常底层的技术问题，通常由服务器管理员来处理。

图1-1-15　即使图片以及区域的大小是不可变的，文本仍旧可被缩放

Note

A List Apart网站中介绍了更多的XHTML及其DOCTYPE，可在www.alistapart.com/articles/doctype访问。

Note

这两篇文章详细介绍了MIME类型相关的内容：其一在Web标准组织网站（www.webstandards.org/learn/askw3c/sep2003.html），其二在XML.com（www.xml.com/pub/a/2003/03/19/dive-into-xml.html）中。

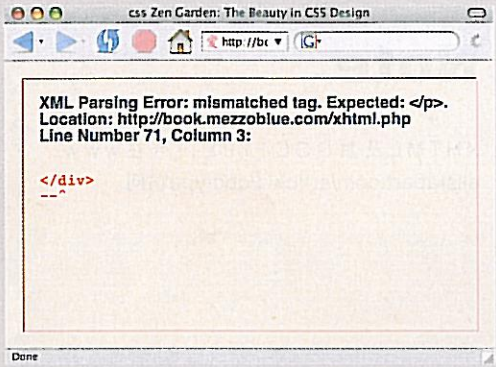


图1-1-16　浏览结构不正确的XML文档的结果

Tip

关于字符编码的更多建议请参考“WaSP Asks the W3C”（www.webstandards.org/learn/askw3c/dec2002.html）这篇文章。

但即使已经为文档指定了正确的MIME类型，问题也远远没有结束。只有最新的浏览器才支持application/xhtml+xml，所以我们需要一个转换的方法，用来有选择性地发送合适的MIME类型，以便为那些较老版本的浏览器指定一个更加兼容的MIME类型。

此外，若指定了application/xhtml+xml MIME类型，将触发浏览器的XML解析过程。XML有一套自己的判断错误的方法，例如若文档中缺失了一个</p>标签，浏览器将停止解析并返回一条错误信息。发生错误的文档也不会被部分呈现在浏览器中，也就是说，在浏览器中将无法看到任何一段页面内容（图1-1-16）。XML文档必须有着良好的结构——必须这样，不允许有任何错误发生。

XHTML真的那么值得使用么？毕竟XHTML的整体目标就是将HTML重构为一种XML的应用。所以若使用了XHTML却不按照XML格式编写，就显得有些奇怪了。XHTML所带来的优势却基本上并不存在，但理论上只有这样做才能称其为XHTML，application/xhtml+xml的MIME类型仍是必不可少的。即使不想不使用application/xhtml+xml MIME类型也没有关系——很多世界上一流的Web设计师也同样毫不顾忌地将XHTML的 MIME类型指定为text/html。不要因此而不安，只要避免使用XHTML 1.1并选择XHTML 1.0即可。同样，坚持使用HTML 4.01也可以帮助您完全避免这个问题——HTML 4.01使用更加宽容的text/html作为MIME类型。当然，在之后编写文档标记的过程中，仍旧可以遵循XHTML的规范。

从中学到的教训就是，XHTML看上去似乎很简单，但是一旦你开始深入到其中，将会发现前面等待的却是一大团纠缠不清的问题。

4. 其他语言

在计算机行业中曾经有那么一段时间，若你想用非英语的语言写点什么（假设你并不在英语国家），则必须要靠其他软件的帮助才行。

谢天谢地，当前的软件在设计时都考虑了相当多的国际化问题。万维网中就能找到相当好的证据——非英语的网站数目正在一刻不停地快速增长。

为了能够让更多人了解，禅意花园鼓励志愿者将其翻译成各国语言（www.mezzoblue.com/zengarden/translations）。当各种各样的翻译版本出炉之后，问题也随之浮出水面：处理其他语言需要对字符编码有初步的了解，而翻译却更加类似一门艺术，而不是科学。

字符编码

目前的操作系统都能够良好地处理非英语字符。但按照默认设定，最基本的Windows XP或Mac OS X安装并不包含对世界上所有语言的支持。但安转CD中都附带了语言包，加载了这些语言包之后，计算机即可正确显示多种非英语语言的字符。

能够看到文字不过是完成了一半的任务：只有在字符编码匹配时，将内容直接从源文档拷贝并粘贴至目标HTML文档才行得通。例如，使用shift_jis编码（日语）的文本将无法在utf-8编码（Unicode）的文档中正确显示。所以我们必须要保证当前编写的内容要与HTML文档的字符编码相匹配。

字符编码可以在服务器端设定，并作为网站中所有页面的默认配置。即使已经有了这些，仍需要在HTML文档中用meta标签再次指明字符编码：

```
<meta http-equiv="content-type" content="text/html;
➥charset=iso-8859-1" />
```

禅意花园使用的字符编码是iso-8859-1，该字符编码中包含了大多数主要的欧洲语言。各种各样的翻译版本都使用了不同的编码，但事后才发现最好的选择却是统一的UTF-8。

UTF-8是一种Unicode，Unicode则是一种流行的国际性编码方案。使用UTF-8所带来的好处就是，不同字符集的多种语言，例如法语、日语、阿拉伯语和希腊语等，将能够共存于同一张页面中。

目前唯一阻碍UTF-8在Web中广泛应用的就是，一些古老的、没有考虑到国际化需求的服务器端软件以及页面开发工具并不支持UTF-8。目前的各种浏览器在处理UTF-8时都没有任何问题，所以用户这一端并不是问题所在——问题正是出现在服务器端。但在处理亚洲语言时，由于其需要极其复杂的字符集，UTF-8也将生成不必要的庞大文件。因此，若文档的主要语言是中文，那么指定一个更加明确的字符集可能会更好些。

从中学到的教训就是，考虑到UTF-8在处理多语言时的强大之处，若是相关服务器端软件和开发工具能够支持的话，我们应该尽可能地选择使用UTF-8编码。

翻译中的差异

因为禅意花园的翻译是自愿行为，所以翻译的质量也自然良莠不齐。以

这些翻译版本所用语言作为母语的读者现在还时常提出建议和修改意见：显而易见的语法以及拼写错误很容易纠正，但对于词句给出的替代翻译将会变得难以抉择，毕竟仁者见仁，智者见智。

即使选择专业的翻译服务，不同的人类的语言之间也并不是可以直接互换的。同样的一个短语，却可以以不同的方式理解成不同的含义，因此如何选择就取决于翻译者的经验以及该语言所代表的文化背景。例如，短语“That will never fly”指的是一个馊主意，还是某架有缺陷的飞机？这种模棱两可的二义性可以借助上下文来澄清，但前提条件是翻译者正确地理解了上下文的含义。

与计算机语言相比，人类的语言显得非常“不精确”，所以关于翻译的问题始终存在。解决这个问题最值得信赖的方法就是遵从大多数使用这种语言的人的意见。但区域性的方言所带来的一些变化却对此雪上加霜（例如对于“颜色”这个单词，是使用美国式的color还是英国式的colour呢？），甚至大多数人的意见也很难统一。特别是若你不讲这门语言的话，那更是无从下手了！

从中学到的教训就是，翻译并不是个轻松的任务，想做到完美更是几乎不可能。所以结果就是，禅意花园愉快地接受了这个现实，并声明翻译上的差错是无法避免的，我们愿意忍受这些小小的不便。

5. 版权保护和抄袭者

除了起到演示的作用之外，禅意花园的价值还在于灵感的展现以及知识的分享。这是一把双刃剑，因为分享就意味着迟早有一天，人们希望得到的要超过你愿给出的范围。

从一开始，禅意花园的提交作品都遵循这样的协议，即作品的CSS应用创作共用许可协议[1]，允许其他人在某个限制范围之内使用。但考虑到图像对于作品而言独一无二的重要性，其著作权仍完全保留给设计师。如果你想要基于某件作品进行创作，那么可以自由拷贝并修改其CSS，但若要使用作品中的GIF、JPE或PNG文件，或是要使用整个的设计，则首先要征得原设计师的同意。

理论上，这样的版权规定既可以让别人参考、重用作品的部分CSS代码，以其达到学习的目的，也能够为作品的设计师保留足够的权力。禅意花园的本意是为了分享知识，所以这样的规定看起来也似乎非常完美。但实际上，却很难实现这样的理想。

---

[1] 关于创作共用许可协议以及其发布者——知识共享组织，可在其网站http://cn.creativecommons.org/中了解更多内容。——译者注

在禅意花园发布不久，第一个涉嫌抄袭的站点出现了。该站点中的一些内容原封不动地抄袭了禅意花园中某些作品的设计和图像等。因为作品图像的版权规设计者所有，所以这样做显然违反了禅意花园的版权声明。

有些站点同时使用并修改了禅意花园部分的图像和CSS，或是重用了一些设计上的元素，给人一种“似曾相识”的感觉。在一些情况下，我们也很难辨认某个网站是否抄袭了禅意花园的创意。还有一种情况是，某些设计中CSS起到了非常主要的作用（无论是否使用了图像），然而模仿者原封不动地照搬却让原作品的设计师感到极其无奈。

对于第二种情况，我们有什么办法么？大多数情况下，我们会给抄袭者发送一封电子邮件，简单说明问题——因为大多数人都会尊重原作者的权利和意愿。但若是抄袭者并不是那么肯合作，我们也无计可施。毕竟创作共用许可协议的条款比较宽松，除非抄袭者直接使用了原设计的图像，否则我们并没有追索权。从技术角度上考虑，虽然会让很多设计师感到不快，但这类的重用却是完全合法的。

创作共用许可协议在侵权行为的定义上有一个漫长的灰色地带。对原有作品进行了多少的修改之后才能算作新作品？若最终作品与原作品完全不同的话，允许人们以原作品为基础并开始自己的创作么？这些问题甚至在法庭上都会引起很大争论，所以目前也并没有一个完美的答案。

对于禅意花园来说，最理想的解决方案就是既让CSS公开，让别人有足够的参考资源，同时却不鼓励未经原作者允许就全盘或部分照搬其代码，甚至用于商业作品中。这里需要良好把握的就是尺度问题——过于自由会让设计师在提交作品时顾虑重重，而过于严格的保护也将影响到禅意花园建立的初衷。

除非彻底关闭禅意花园，否则将永远没有办法保证作品不被与设计师初衷相反的方式“重用”。是否有版权协议保护，是发布创造性作品时自然要考虑的问题——这类作品非常容易被剽窃。这样的事情一直在发生——看看pirated-sites.com就知道了。

借助于每个浏览器中都内建的“查看源文件”功能，抄袭Web页面变得非常容易。但同时借助于引用记录以及搜索引擎，发现这类抄袭也较为容易。

在撰写本书时，彻底解决这个问题仍是遥遥无期。目前我们仍决定允许受限制地使用作品的CSS，并用创作共用许可协议保护，但却禁止将其用于商业用途。这并不是个完美的解决方案，但至少我们为此已经做出了努力。

Note

禅意花园的CSS文件应用了By-Attribution Non-Commercial Share-Alike许可。Creative Commons网站中列出了该许可的全文：http://creativecommons.org/licenses/by-nc-sa/2.0。

图1-1-17　一个带有JavaScript脚本的禅意花园设计作品，其特色在于页首移动的云彩

从中学到的教训就是，无论应用了什么样的版权协议，还是会有一些人会无视这些条款，为所欲为地进行抄袭。所以解决方法就是选择某种有效的途径警告这些人，让其知道他们可以做什么，不可以做什么。

6. 将禅意花园做大、做好

很多访问者都提出了建议，希望禅意花园能够做大些、做好些、做得与众不同、做得更有影响力、甚至是添加一些维生素C……只要是你能想到的建议，我们都收到过。下面就列举了一些建议：

• CMS版本的禅意花园

为一张页面给出不同的CSS设计，这个想法非常好。但为什么不将一张页面扩展为一个完整的网站呢？这个网站可以通过内容管理系统（Content Management System，CMS）维护。这样也可以更好地反映出真实世界中网站的状况。

• 电子商务版本的禅意花园

类似于CMS禅意花园的创意，为什么不为一个提供了真实且不断改变的商品的电子商务网站应用不同的CSS设计呢？CSS可以用于修饰购物车、商品列表、推荐商品等页面。

• JavaScript版本的禅意花园

将文档的结构和表现样式分开是个很不错的目标。但为什么不更进一步，添加一个控制页面行为的JavaScript层呢？这一层增强禅意花园的表现能力，且同样也能与前两层彻底分开。

这个想法的创始者之一，Rares Portan提交了一个非常不错的示例作品（图1-1-17，www.csszengarden.com/javascript/）。随之也出现一些对这种增强做法优劣之处的讨论（www.mezzoblue.com/archives/ 2004/05/04/javascript_b）。

• XML/XSLT/RSS/随意版本的禅意花园

首先用最基本的标记语言编写内容，然后用CSS修饰其样式。那为什么不充分展现出CSS和基于XML的语言之间配合的潜力，而却只选用XHTML呢？

• 这些禅意花园在哪里？

毫无疑问，这些主意都非常吊人胃口。其中很多都能给Web设计社区带来

额外的宝贵资源。大多数情况下，这些想法的创始者都会收到正面的反馈与建议，人们都很乐于看到将这些想法付诸实现的示例站点。

但事实上，禅意花园只是专注于几个目标，也不能满足所有人的所有需求。从这一点考虑，前面那些建议就都意味着需要修改创建禅意花园的本意，这对那些已经参与进来的贡献者来讲是不公平的。

但创建一些新的网站来实现这些创意却是不错的想法。若能发明一种更好的捕鼠器，那就去做吧！一两个这样的站点随之而生，但却并没有维持太长的时间。关于为什么会出现这样的结果，每个人都有自己的想法，我们也不例外。

首先，为禅意花园设计作品非常容易。其难点在于设计技巧以及CSS编程知识，但作品却并不需要花费长达数周的时间来完成。为整个站点编写CSS将困难很多，特别是对于那些内容不停变化的站点来说。当设计师由此得到实际利益时，前面花费的时间自然物有所值。但若是不能，那么耗费大量时间也就自然成为制约这类网站发展的门槛。

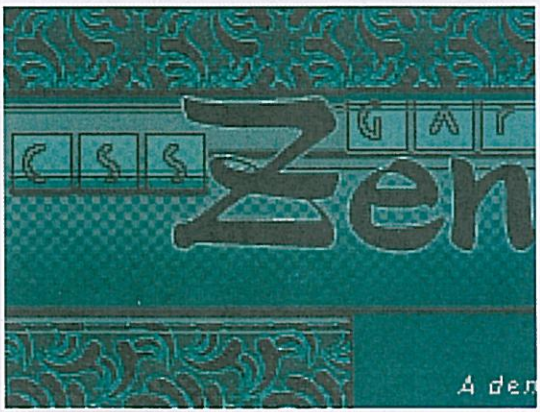
其次，禅意花园最主要的目标已经实现。禅意花园的创办目的就是为了改变人们对基于CSS进行设计的理解，这个任务已经完成。既然CSS样式可以很容易地应用到一张页面上，那么也不难对其进行扩展，并将其应用到多页面网站中。

若你还是无法想象如何将CSS应用到多于一个的页面中，那么很多真实世界中基于标准设计的网站或许能够更好地证明这一点。目前，很多企业站点都转向了基于标准的设计，例如Wired News（www.wire d.com）、ESPN.com、Chevrolet（www.chevrolet.com）、AOL.com等。

从中学到的就是，人们非常需要一个编写各种各样网站时都能够遵从的、“放之四海皆真理”的标准。但这样的标准是没有的，相反，真实世界中的实例网站却是最好的学习对象，它们在阐明标准方面要远远好于禅意花园之类的演示站点。

Atlantis（亚特兰蒂斯），40页

Zunflower，46页

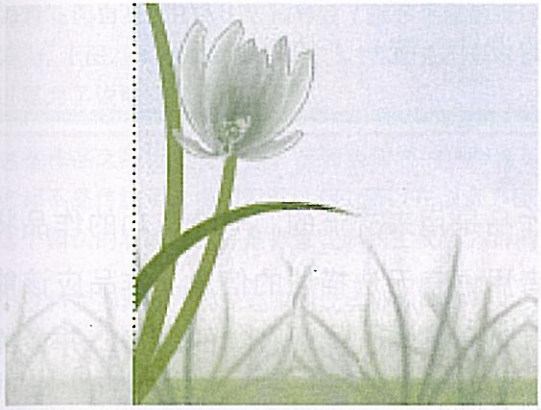


Viridity（翠绿活力），58页

Ballade（叙事曲），64页

第2章

设计

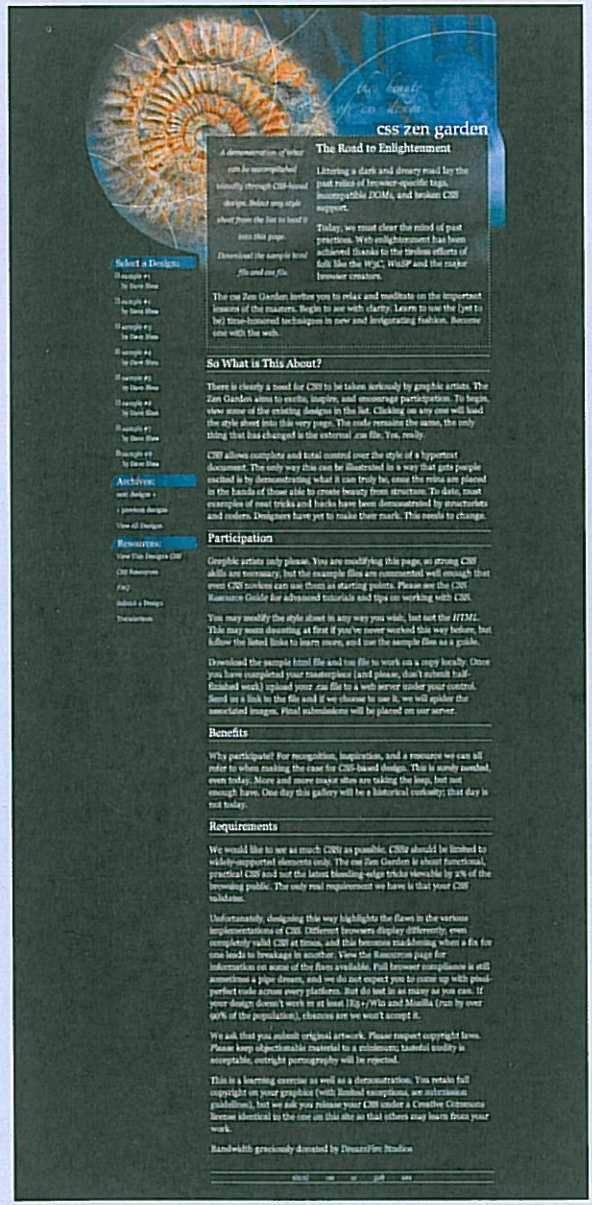


Springtime（春天），52页

Night Drive，70页

Web设计的流程经历了戏剧性的变化，但优秀设计的原则却从未改变过。无论是选用调色板，还是把握欣赏者浏览作品的顺序，或者合理有效地使用图形，亦或是管理页面相关的元素，传统设计中的关键之处在Web设计中同样重要。

这一章中将要介绍的作品能够阐明设计的关键之处，同时也会澄清当代Web设计师心中的一个疑虑：应该如何把握设计的流程。是首先在Adobe Photoshop中拟出设计的草图，然后再转向到CSS中，还是一开始就直奔主题编写CSS代码？相信在阅读完这一章之后，您一定会有所收获。



Kevin Davis，设计师
www.csszengarden.com/028

2.1 Atlantis(亚特兰蒂斯)

简洁的设计，统一，象征手法

视觉作品是用来交流的。一个成功的作品将传递给欣赏者用语言无法描述的信息。作品应该能够唤起某种气氛和感情，体现独特的风格，并最终引起欣赏者的共鸣。

如同室内设计师将桌椅、柜台、油画组合成一间带有忧郁色彩的咖啡屋一样，在创建亚特兰蒂斯时，Kevin Davis也希望能够将文字和图像以类似的方式组合起来。Davis从相片和调色板入手，将各种元素不断组合直至满意。然后再将设计好的作品付诸于CSS实现。

2.1.1 简洁的设计

除了充满细节的页首部分之外，亚特兰蒂斯给人的第一印象就是其相对的简洁性。通过将相片元素和令人不可思议的简单布局结合起来，Davis用一种少见的方式将复杂和简洁统一了起来。

禅意花园自身的HTML结构导致了很多平庸的设计都显得方方正正，过于“简洁”（图2-1-1A~图2-1-1C）。但这并不是这些作品的错，列出它们只不过是为了说明问题而已。

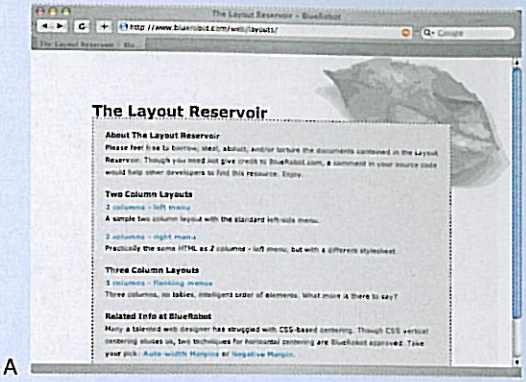
这些作品之所以显得平庸，是因为作者的一个常见的误解：这些作品的设计者都不是传统观念中图形设计师，所以在创意方面自然略显欠缺。当然，纠正这个错误的观点也正好是禅意花园的主要创办目的之一。

但亚特兰蒂斯的出现无疑是对此类疑虑的一个辛辣嘲讽，它用事实证明了简洁并不一定就等同于糟糕：当您一次又一次看到同一件东西的无数种样子之后，将很快感到厌倦。一餐中同时品尝马铃薯泥、烤马铃薯和薯片，再好的东西也将变得没什么新意。火鸡配一些马铃薯泥（或是豆腐）才是一顿简单却美味的晚餐。

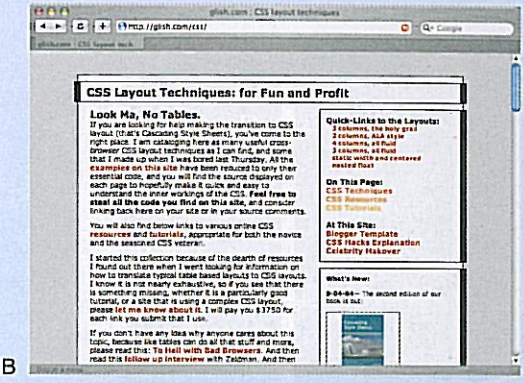
也正是在这种情况下，这样一个低调、简洁的设计才会显得如此成功。毕竟内容决定了设计，若是内容本身就十分复杂，且足够吸引浏览者注意的话，那么也没有必要为其添加过多的修饰。

若想设计一个极其精彩且简洁的作品，从另一个角度看……即使对于专家来说也不是一件容易的事。当设计元素很简单时，浏览者也将用更挑剔的目光检阅页面的各个角落。这时每个元素都会显得格外重要，所以在这种情况下，只有对细节的良好把握才能让作品从平庸中脱颖而出。

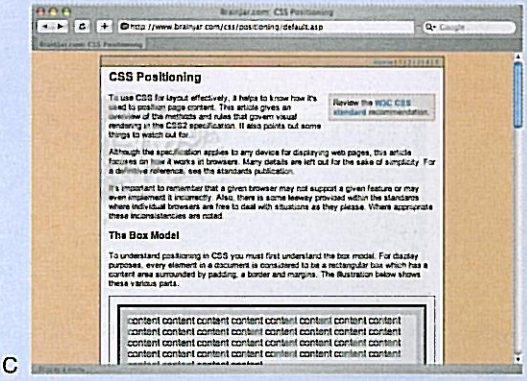
若想学习如何恰到好处地使用设计元素，那么你定会从研究亚特兰蒂斯强大、简洁的布局中受益非凡。若我们移除页首的图像，那么其简洁性不言自明（图2-1-2）：中间为主要的内容区域，左边围绕着一个略小的菜单。



A



B



C

图2-1-1A~图2-1-1C　一些早期的简单设计

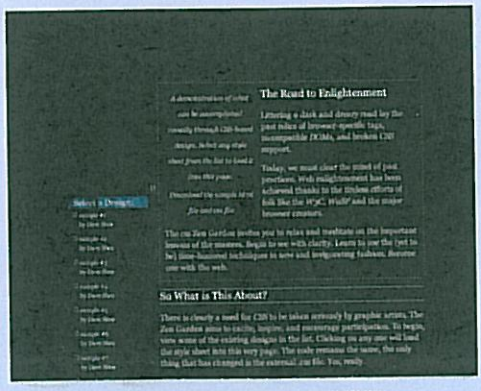


图2-1-2　除去页首图像的亚特兰蒂斯

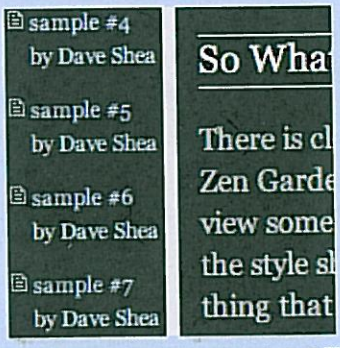


图2-1-3　正文部分的文本大小几乎是菜单部分的两倍

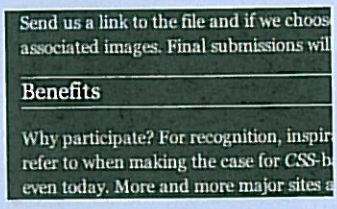


图2-1-4　白色的实线将标题与正文分开

1. 排版样式

在亚特兰蒂斯中，正文部分的文字按照比例来讲要远大于菜单部分的文字，用来更好地区分这两个区域。Davis通过合理地使用字号来暗示浏览者一左边的内容并没有右边的内容那么重要（图2-1-3）。同样，更大的标题字体也将标题区域与正文区域区别开来。这些视觉上的差异为浏览者提供了额外的信息，即表明每一段文字之间重要性的相对高低。

排版是一种起交流作用的表现方式，它拥有内建的层次结构、模式和韵律。通过运用不同的文本大小、文字间距和颜色来调整版面样式，作品将为读者带来很多文字之外的丰富信息。

2. 图标

左边的菜单中，Davis在每个作品名的左边都插入了一个小图标。因为这些图标的大小和文字大小接近，所以给人的感觉是它们半隐藏于文字中，仅仅起到标号的作用，而不会喧宾夺主地吸引或让浏览者转移视线。这个简单的细节为作品链接之间提供了上下文关系，并与菜单的其他部分区别开，也为这个设计简洁的菜单添加了一些令人玩味的独到之处。

图标是一类简化了的符号，用来表示某件物品或者某类概念，或者用来以抽象的图形化概念替代一段较长的文字描述。物品通常都会有较直观的视觉抽象方法，但类似过程和操作之类的概念却很难描绘。一个好的图标所代表的含义无需加以解释，但前提是图标是对物体本质的抽象。设计图标的关键在于，要将图标尽可能地以某种强烈、一致的方式与其要表示的概念关联起来。这样，当浏览者对该图标产生重复印象之后，就能用直觉体会出图标的含义。

3. 分界线

Davis用两种样式的分界线将作品中的不同区域分开。页面最上边的一段与页首部分重叠的文字用虚线包围，白色的实线更进一步将标题和页脚与正文分开（图2-1-4）。这些简单的分界线既为作品添加了额外的上下文信息，又不至于让浏览者感到迷惑。

适当地使用线条能够在页面中定义区域的边界，让页面显得有韵律，甚至给页面添加额外的维度。但也不要过度使用——未闭合的方框以及过多的平行线将给浏览者类似网格结构的暗示，而实际上您的本意却并非如此。浏

览者也可能会尝试寻找本不存在的规律与模式，进而迷失于作品中，所以在使用线条时必须足够谨慎、明智，只用在最需要的地方。

4. 外边距（Margin）

为了让作品始终处于浏览器窗口的中心区域，Davis尽力控制了页面中各元素间的距离。页面的两边都采用了可变宽度的空白区域，以期让内容居中显示。有时候全屏的设计让人愉悦，但另外一些时候定宽的布局似乎更为可取。

对于这两种情况，适当的左右外边距将会让作品的正文部分不至于显得太局促。默认情况下，大多数浏览器会在页面的边缘添加一个极小的外边距。但不幸的是，即使这样，文本仍将从左到右近乎写满整个屏幕。

这是个布局理念方面的问题——文本需要一些呼吸的空间，缺乏足够的外边距将让文本与浏览器的边缘重合在一起。这种情况下默认的外边距设定将无法满足我们的要求，因此设计师自然有责任做出适当的调整。

在这种简洁的布局中，外边距是仅有的几种被使用的设计元素之一，合理地应用外边距能让页面显得完全不同。足够大的外边距更是可以将内容置于浏览器窗口的中心位置，让浏览者一眼就能看到。

Note

大多数设计师不会受到浏览器默认的外边距设置的影响，因为他们总是简单地将默认设置覆盖掉。但对于非专业的业余爱好者来说，这一点的确值得注意。但若是浏览器在这方面做得再好一些，例如能够像标准打印机一样在内容周围自动添加足够的外边距，那么也就不会有这种混乱情况出现了。

2.1.2 统一和象征手法

在用设计元素营造气氛方面，亚特兰蒂斯如Davis所愿地堪称经典之作。据古希腊哲学家柏拉图所述，亚特兰蒂斯是远古时期沉入到海底的一片大陆。页首锈迹斑斑的螺旋贝壳以及古典的石雕支柱都会让欣赏者陷入到对这个传说的深深回忆中，深蓝和黑色也营造出了深海的气氛。Davis在动笔之前就为作品选择好了图像和色彩，这样就在设计的过程中坚守住了最先的设计灵感。

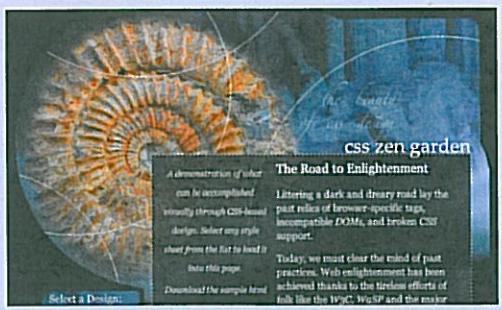


图2-1-5　橙黄色系的贝壳提供了恰到好处的对比度

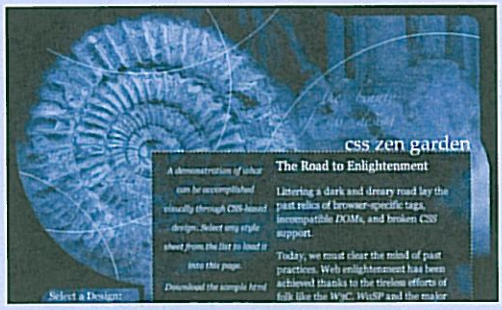


图2-1-6　蓝色的页首图像，这样可能会显得与主题更加统一，但整个作品却因此缺少了对比度

1. 一致性

页首贝壳的颜色经过了一定的调整，以便与旁边的抽象图案融合起来。贝壳的边缘也用Adobe Photoshop进行了更大程度的柔化处理，与整个页首图片融为一体。但Davis并没有将贝壳的颜色调整为与主题一致的蓝色系，而是留下了足够多的、锈迹斑斑的黄色，增强整个作品的对比度（图2-1-5）。若是整个作品都使用蓝色系，那么将给人以过多的沉重感（图2-1-6）。

设计中用到的元素可能各不相同，但作品整体达到统一的意境却是最重要的。除了使用单一的色系之外，还有一些更加巧妙的方式来实现一致性：例如将最重要的、起支配地位的元素用艺术线条或照片表示，或在悉心调整匹配区域的大小后再用线条环绕等。

在亚特兰蒂斯的页首，我们可以看到一些白色的实线圆圈，它们都以类似的方式融合在贝壳和旁边的抽象图案中。若是没有上下文的话，这类简单的白色圆圈并没有什么实在的意义。但在圆弧形以及蓝色背景的衬托下，浏览者将很容易将其认同为水中荡漾的波纹。这些圆圈的大小也和贝壳相近，二者交相辉映，显得非常协调和统一。

2. 借喻

借喻（用简单的设计元素表示复杂的图像或概念）是一种强大的设计方法，可以为作品添加一些微妙的气氛并增加作品的深度。若不是我们在前面已经指出了的话，您可能很难发现页首的那些白色线条。但现在您一眼就能看出来，并能把它们想象成水中荡漾的波纹。

亚特兰蒂斯中使用象征的地方不仅仅是这些简单的线条——页首部分的图片中还包含了很多。从前的亚特兰蒂斯——一个强盛富有的帝国——现在，已经带着那些象征着文明与力量的石柱和美轮美奂的雕像消失在了蓝色的背景中。现今的亚特兰蒂斯——锈迹斑驳、静静地躺在海底。一个消失了的帝国，荣耀已经远去，页首左右部分的空间关系恰如其分地体现了这些。这种看似简单的设计却在不经意之间委婉道出了一段沉重的历史，并为整个作品添加了相当的深度，让欣赏者不得不陷入其中，仔细体会。

但对于那些从CSS角度审视作品的人来说，Davis在页首使用的贝壳还有另一个意义：这体现了他对Eric Meyer的CSS经典示例作品Complexspiral（可在www.meyerweb.com/eric/css/edge/complexs piral/demo.html浏览，图2-1-7）的敬意。

2.1.3 作品亮点

这个作品在与读者交流，引起共鸣方面做得非常好：贯穿网站的一条清楚且统一的主线为提高用户体验打下了良好的基础。但也并不是所有的站点都需要如此的精心设计，对于那些内容和功能占主要地位的网站来说，设计师一般应该尽可能地简化图片等设计元素，以简洁实用为首要目标。

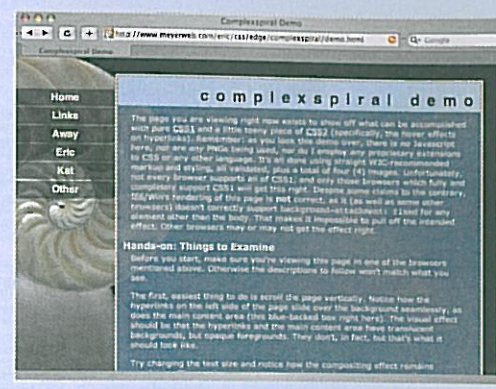
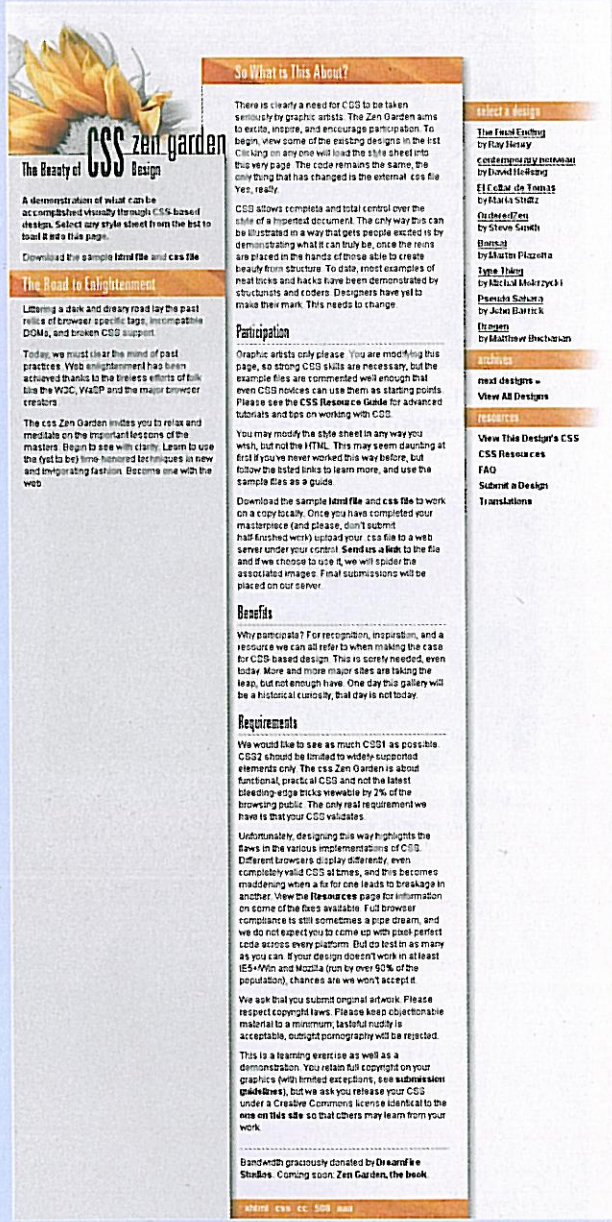


图2-1-7　Eric Meyer的示例作品Complexspiral，亚特兰蒂斯中同样在页首使用了类似的贝壳



Radu Darvas，设计师
www.csszengarden.com/026

2.2 Zunflower

游离于光、影、形状和空间中

某一天，Radu Darvas的双眼突然被阳台上的向日葵所吸引住，随后他开始为这些向日葵拍照，Zunflower的设计灵感也随之迸发。Adobe Photoshop中对这些相片的处理更是让他理清了思路，最终完成了Zunflower的设计。

Zunflower的精妙之处在于，它并不是以“框”的形式设计的。深入观察这件作品，可以看到每一栏的位置都被仔细调整过，在视觉上形成重叠的效果。这种效果表现了光和影之间、形状和空间之间的相互影响，最终成就了作品明亮积极的主色调。

2.2.1 寻找光源

当你读到这里的时候，无论正坐在什么地方，总会至少有一个光源让你看清书上的文字。把你的视线从书本中暂时移开——光源在什么地方？它是如何照亮你周围的物体的？影子又在哪里呢？

视觉艺术家通常使用光和影为作品添加真实感，或者让作品中的某个部分凸现出来，以体现层次感。

回顾一下Web设计过去的十年，我们将发现很多对光的使用。例如让图像实现梯度效果（图2-2-1），为按钮添加立体的斜边（图2-2-2），给照片创建阴影（图2-2-3）等，光影效果在Web中的用途非常广泛。

不幸的是，一些没有正式学习过视觉设计的Web设计师经常错误地使用光线，在种种限制中能够确保完成任务也成为不小的挑战。但最终，趋势依然影响了技术——即使光线的效果并不真实，光源的位置也显得矛盾（图2-2-4）。

2.2.2 保留阴影

Zunflower演示了正确使用光源投射阴影的方法。虽然强烈的梯度和斜角在当代的Web设计中已经基本上绝迹，但投射阴影仍一直被广泛应用。原因如下：

- 阴影可为页面添加丰富的视觉效果。
- 投射出的阴影能让浏览者感到页面以及页面中的元素存在着层次感。
- 在绘图软件中，阴影非常易于实现。

为了避免上面提到的那些使用阴影可能会导致的问题，例如相互矛盾的光源等，这里我们给出了如下的几条一般性的指导建议：

- 注意阴影颜色的深浅——阴影的亮度表示物体与阴影投射表面的距离。深色的阴影代表物体与背景较近，而浅色的阴影则表示物体与背景较远，即更靠近观察者。
- 阴影的长宽也会给用户造成不同的距离感。较为细小的阴影让人感觉物体与背景较近，而较宽大的阴影则让物体看起来离背景稍远。

图2-2-1　梯度填充，作品创建于1995年左右。在高压缩比率时图像会出现抖动和条带，显得不够专业

图2-2-2　带有立体文字的立体按钮。这类视觉效果常见于20世纪90年代中叶的Web页面中

图2-2-3　应用了经典的投射阴影效果的相片

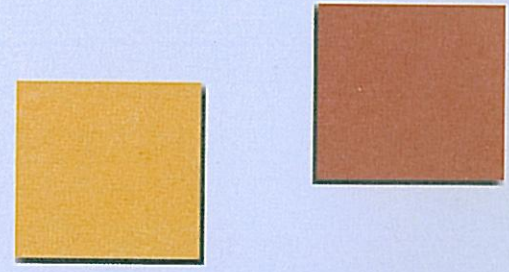


图2-2-4　错误地使用光源——注意两个物体的阴影投射方向并不一致

Tip

若使用Photoshop实现阴影特效，请确保选中了“Use Global Angle”单选框。这样即可让作品中各个阴影的效果保持一致。

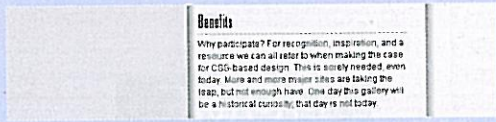


图2-2-5　正确使用光源可以实现恰到好处的阴影效果

图2-2-6　继续使用同一个光源，为作品带来强烈的真实感

□ 要确保光源的真实性。若页面中包含多个对象，则它们的阴影方向要保持统一，以体现出光源的一致性。

例如，头两栏的重叠让Zunflower居中一栏的阴影显得很自然，就像是光线从左边照射过来一样（图2-2-5）。

同样，右面一栏的淡出效果也为作品增加了更多的光影效果。这一栏的颜色逐渐变淡，最终在我们的视线中淡出（图2-2-6）。

2.2.3　走近形状

与色彩一样，形状也能影响我们的心理。设计师可以通过选择某类特定性质的形状来唤起欣赏者的某种特定的情绪。

形状对人心理的影响不仅仅限于计算机或印刷作品中。人类自从能够在地面画出符号、在岩洞的石壁上刻出描述故事的象形符号开始就已经在使用形状了。环顾四周，形状无处不在——这几乎是人类与生俱来的天性。无论是用来描述复杂的数学公式，还是为某个建筑添加修饰，甚至是代表象征意义的图腾，使用形状是我们的一种本能的自我表达方式。

在Zunflower中，页首的一丛花极其显眼——任何人都会被它的颜色和形状所表现出的内涵所深深吸引。

1. 主要形状

圆弧形经常与女性、阴柔关联起来，表示温暖、舒适、暧昧以及爱情等。进一步拓宽适用范围，圆弧形也能用来代表社区、全体、忍耐、运动以及安全等含义。很多以女性、社区、整体生活或浪漫传奇为主题的网站经常使用圆弧形来强调其主题。

三角形则通常会让人联想到男性、阳刚，用来表达诸如强壮、攻击以及运动感等含义。棱角的方向也会影响形状的含义——浏览者的目光通常会被吸引至处于支配地位的棱角所指向的方向。指向上方体现了向上的趋势以及攻击性，而指向下方则通常暗示出消极的意味，表示被动和无力。

当然，三角形（包括方形）的适用范围也可以被拓宽。方形暗示了力量和根基，这很有可能因为它们看上去足够的坚实稳重。方形的设计能给浏览者带来有序、逻辑、严密以及安全的感觉。

2. 组合各种形状带来最大限度的视觉冲击

可以看到，Zunflower图像本身是由三角形和圆弧形构成的。当然，这是大自然创造的结果，但Darvas又在Photoshop中对这一印象进行了强化，使花的颜色和形状更加鲜明。因此，这幅图片给人以强烈的感官刺激——阳刚和阴柔并存，并创造出一种非常独特且让人难以忘怀的意境（图2-2-7）。

在Zunflower中，作者还用显眼的方形平衡了由花带来的阴柔感，奠定了整个作品的基石（图2-2-8）。

形状之间合理的组合能够给欣赏者带来最大限度的视觉冲击。再为其添加适当的色彩和光影之后，作品将更加引人入胜。页首部分恰当保留的部分空间也进一步增强了作品给人带来的冲击感。

图2-2-7　近距离观察Zunflower是如何将三角形和圆弧形组合起来的

图2-2-8　方形用来让设计保持均衡，不至于过度偏向阴柔

2.2.4　把我的空间给我

正如光、影、形状一样，间隔空间也能够对人产生心理上的影响，从古到今的艺术家们一直都在采用这种方法拉近作品与欣赏者的距离。

在视觉设计中添加间隔空间是考虑到这样一个原理——适当的空间可以将浏览者的目光引导至页面中的关键位置。空间还能够对文本和图像的过密产生一定的缓冲，让眼睛有休息的余地。这将大大提高内容的可读性，并给浏览者足够的思考内容的时间。

1. 心理和社会意义

空间感在心理和社会方面都有着极为明显的含义。从心理学角度考虑，主要有如下几点：

- 间隔空间和经济形势息息相关。间隔空间的多少及其使用的方式同样也能够传达出很多信息。挤成一团的设计决不会显得美观，这时恰到好处的间隔空间就会让作品充满优雅的气息。
- 使用间隔空间并不意味着这个位置什么都没有（间隔空间反而能够帮助

图2-2-9　当背景被设置为黑色时，我们即可清楚地看到负空间是如何定义对象边界的。负空间自身也是由一系列的形状组成的

我们定义对象）。拥挤的设计意味着信息过多，会让浏览者绷紧了神经。但若是添加了足够且恰当的间隔空间，传达给信息的方式就会变得更为完善、优雅，让浏览者倍感轻松。

□ 间隔空间与特定地区的文化也有关系。正如带有不同文化背景的人对同样色彩和形状的感受各不相统一样，文化背景也决定了人们对间隔空间的不同看法。在某些国家中，交流中的身体接触表示关系的亲密。但在另一些文化背景中，例如美国，对身体接触的看法却大相径庭。

若我们在设计时曾考虑过上述几点，就会认识到间隔空间并不仅仅是页面缺少内容的副产品而已，而是作品中的一个极其重要的、需要全方面考虑的大问题。

2. 正空间（Positive Space）和负空间（Negative Space）

首先让我们解释一下正空间和负空间的区别，之后你会更容易地理解间隔空间与作品融为一体的方法。

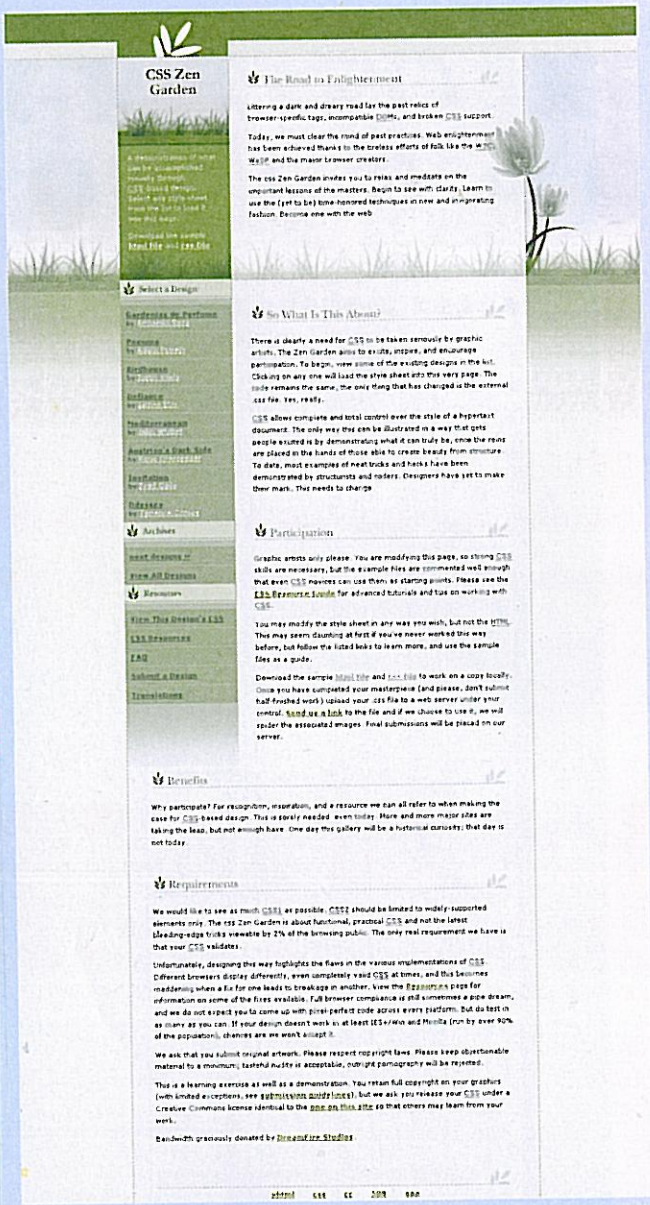
正空间是指组成作品的对象所占据的空间，负空间则是指非作品对象部分（即背景）占据的空间。尽管定义如此清晰，我们也不能将正空间和负空间割裂开来，二者互相依赖、相辅相成。负空间起到定义正空间边界的作用，作品中对象的边界也正是负空间的开始，这个边界本身就是一种形状（图2-2-9）。

2.2.5　添加专业的光泽

理解正确使用光源的方法将会帮助你在创建阴影及相关效果时达到协调统一。无论是在整体视觉效果中，还是在诸如网站标志或

图标等特定之处，形状在设计中所起到的角色都值得深入讨论。充分地理解认识空间及其使用方法和使用目的，将会帮助您把想要表达的信息更好地传达给浏览者。

学会并掌握对光线、空间和形状的控制将让您的作品更加完美，如Zunflower一样熠熠生辉。



Boér Attila，设计师
www.csszengarden.com/083

2.3 Springtime（春天）

用色彩激发情感

寒冷、单调的冬天过去之后，Boér Attila迎来了第一个晴朗明媚的春日，让他感受到了向上、充满活力和欢乐的色彩。从冬到春的颜色变化也让Attila产生了创作的灵感。

人们很早就发现，颜色可以引起人情绪的变化。这也正是将红色定义为暖色的原因——红色让人容易联想到火焰或辛辣的胡椒粉。随后，我们又自然地将这种联想应用到设计中的颜色选择上。例如，人们经常将象征着土地、大自然的棕色和褐色用于天然产品的包装上。

在春天中，冬日那单调乏味的颜色和春天万物萌发的颜色被混合在了一起，让作品的颜色选择既不显得过冷，也不显得过热。但整个设计却强烈地表达出了温馨与和谐的意味，给欣赏者带来积极愉悦的感官享受。

2.3.1 色彩的影响力

若是设计师希望向观众传达某种信息，那么理解色彩所包含的心理效果、文化影响、性别偏好等将与掌握其使用技巧同等重要。

1. 清楚地表达信息

页面的内容、布局及图片的使用无疑都会体现出该站点将要表达的意愿，但色彩的使用同样是一种非常重要的交流方式。优秀的设计师了解如何恰当地使用色彩来表达设计意图，并加深浏览者的印象。对于浏览者来说，色彩表达的含义与图像或文本一样令人信服，某些时候甚至还要强于它们。

若你要为一个重金属乐队设计站点，那么选择在白色的背景下使用黄色或粉色等明亮的颜色显然将与主题格格不入。重金属带有黑暗的意味，虽然作品中仍可以使用一些明亮的颜色，但对于该音乐流派，深沉、浓重的色彩基调则毫无疑问最具代表性。

相反，若你要为某个古典四重奏设计站点，那么深色主题也很难与这一流派音乐的轻柔舒缓特点相符合。选择与网站主题一致的色彩在设计时至关重要，这也是创建成功网站过程中的重要一环。

2. 色彩和人类心理

色彩对人类情绪有着非常大的影响。二者关系非常复杂，包括很多方面的因素。

对于无法感知光线的盲人来说，色彩无关紧要。对于那些色盲（一种先天性疾病，患者无法区分两种特定的颜色，最常见的是红色和绿色，即红绿色盲）或色弱的人来说，颜色的理解也和正常人有所差别。色盲人群的分布较广泛，在北美，大约每12个人中就有一例[1]。因此，考虑到这些潜在的浏览者，设计时就更要在选择色彩上多加小心。

某种颜色的色调同样可以影响它的含义。一般来讲，浅绿色能够唤起某种较愉悦的感觉，但深绿色却恰恰相反。类似地，添加了材质（texture）的颜色也会让人对其有不同的理解。表2-3-1列举了一些常见的颜色及其带来的一般心理联想。

[1] 在我国，男性红绿色盲的发病率为7%~8%,女性的发病率为0.5%。——译者注

Note

当然，这类挑战常理的色彩选择也许反而能达到某些意想不到的效果。尽管如此，在这样使用前也必须要格外小心地准备和验证，确保能够实现你的预期的目标。文中的观点仅仅用来说明通常情况下人类对色彩的反应。

图2-3-1　春天的颜色选择，包括一系列的绿色、一种淡蓝色和白色

表2-3-1 常见的颜色及其心理联想

颜色	心理联想
红色	力量、活力、爱、激情、进攻、危险
蓝色	信任、保守、安全、清洁、悲伤、有序
绿色	大自然、健康、嫉妒、复苏
橙色	愉快、幸福
黄色	乐观、希望、冷静、懦弱
紫色	神秘
褐色	可靠、舒适、忍耐、大地
灰色/银色	智慧、未来、谦虚、悲哀、腐朽、高雅
黑色	力量、性、完善、神秘、恐惧、忧愁、死亡
白色	纯洁、干净、精确、清白、中性、不毛、死亡

你可能会注意到，某种颜色可能会引起两种截然相反的心理联想，就像黑白两种相反的颜色一样。例如红色，既可能表示激情也同样能代表危险（或许激情和危险的差别并不是很大？）。此外，还有很多因素与这种看上去的矛盾相关，例如文化背景和性别差异都会让人对相同的颜色产生不同联想。

在春天中，对绿色的色调以及亮度的选择都非常恰如其分，让人联想到青草和树叶，带来温和、舒适的感觉。作品中偶见的一丝蓝色也进一步体现出了大自然的意味（图2-3-1）。

3. 色彩、文化和性别

Attila是匈牙利人。艺术家所生长的文化背景会影响到他（或她）对颜色的理解和选择吗？专家们对此都持肯定的态度。类似地，欣赏者的文化背景以及性别也会影响达他们对作品的理解。

文化背景和性别差异都会扩展并复杂化前一节中列出的颜色引发的心理联想。表2-3-2列出了一些由不同的文化背景和性别导致的对颜色的不同理解。从中我们可以很容易看出，色彩的使用，特别是对于在世界范围，例如互联网中的作品来说，必须慎之又慎。

表2-3-2 文化背景和性别差异对色彩理解的影响

颜色	影响
红色	在中国，红色是好运的象征。红色与白色同时使用将加强这种意味。相对于蓝色，很多女性更偏好于红色
蓝色	总的说来，蓝色在世界范围内都是一种被尊敬的、表示崇高的颜色
绿色	在美国，绿色让人联想到金钱，但在其他文化背景中却不尽相同。与男性相比，女性可以分辨出更多色调的绿色
橙色	在美国有这样一种包装设计上的惯例，即用橙色表示其中的物品较便宜。所以在用橙色设计商业网站之前，我们必须考虑到这个问题。与黄色相比，男性更喜欢橙色
黄色	在亚洲人眼里，黄色是神圣的。与橙色相比，女性更喜欢黄色，因为它让人联想到温暖和乐观
紫色	紫色会让欧洲人联想到悲哀，所以紫色的使用也一直备受争议。自然界中紫色相对来说比较少见，仅在某些花以及一些咸水鱼上出现
棕色	对于任何文化背景和性别的人来说，棕色都是一种中立的颜色
黑色	在大多数西方国家以及很多其他国家中，黑色都表示悲恸和死亡
白色	在大多数亚洲国家中，白色表示悲恸和死亡。但在西方社会中却表示贞操、纯洁

2.3.2 网站中使用的调色板

在为某个网站选择调色板时，应该对上述色彩相关的问题有所了解。正如要清楚作品要表达的含义一样，你还需要对浏览者的期望有足够的了解，以便于根据需要做出最恰当的选择。

研究你的听众

交流中的一个黄金法则就是了解你的听众。因为人与人之间文化背景和性别等差异都会影响到他们对同一色彩的理解，所以在使用色彩之前对作品的浏览者进行充分的研究就显得非常必要。对于某些颜色，大多数人对其的理解和认同感都惊人的相似。这也正是蓝色、棕色以及春天中用到的绿色等被广泛应用于各种作品中的原因。

色彩学专家J. L. Morton（前面介绍过他的网站）建议设计师们尽可能地

Note

若想了解更多关于色彩的研究论文、图书以及其他资源，请访问J. L. Morton教授的网站：Colorcom（www.color com.com）。还有一个色彩相关的站点也非常不错，Causes of Color（http://webexh ibits.org/causesofcolor）。

Note

在开始设计之前，最好先创建一些测试用的调色板。在Photoshop中创建调色板非常容易（图2-3-2）。新建一个文档，然后使用颜色选择器（Color Picker）将不同颜色以条带形式添加进去，最后看看它们的配合效果。之后你还可以创建一个测试HTML文档，并在CSS中尝试使用一下这些颜色。

CSS在为作品创建原型的阶段能够起到非常大的帮助：对于这些测试用的调色板，你可以为其相应地创建若干个CSS文件，然后将这些CSS依次应用到测试HTML文档上并观察其实际效果。这种方法甚至可以用于可用性研究中——把应用不同调色板的设计草图发送给你的用户，然后分析他们的反馈。

图2-3-2　在Photoshop中创建调色板

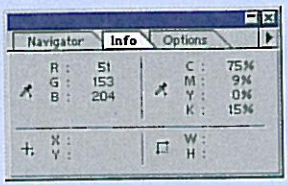


图2-3-3　用Photoshop得到颜色的RGB值

选择与预期浏览者的通常理解相接近的颜色。例如在为儿童设计网站时，一般应该选取表达欢乐的颜色。若要表现出高雅，那么黑色和银色的组合在美国将会非常受欢迎。若想吸引男性浏览者的关注，那么通常说来大量地使用蓝色总归没有错。而若是想引起女性浏览者的兴趣，那么红色和粉红色则要比蓝色好得多。要是想设计大自然主题相关的网站，那么绿色、蓝色和棕色也自然将是你的不二之选。

2.3.3　在CSS中使用颜色

幸运的是，CSS提供了很多描述颜色的方法。我们可以在下面几种方法中随意选择：

- 颜色名称。CSS目前支持17种颜色的名称，分别为black（黑色）、silver（银色）、gray（灰色）、white（白色）、maroon（栗色）、red（红色）、purple（紫色）、fuchsia（紫红色）、green（绿色）、lime（鲜绿色）、olive（橄榄色）、yellow（黄色）、navy（藏青色）、blue（蓝色）、teal（凫蓝）、aqua（浅蓝绿色）和orange（橙色）。例如：

```
color: orange;
```

- RGB值（绝对值表示）。我们可以使用红（Red，R）、绿（Green，G）、蓝（Blue，B）三种颜色的绝对值来描述某种颜色。颜色的RGB值可以在Photoshop或其他图像处理软件中得到（如图2-3-3所示）。例如下面这段代码就表示一种淡蓝色：

```
color: rgb(51,153,204);
```

- RGB值（百分比表示）。按照百分比混合红绿蓝即可得到想要的颜色。例如下面这段代码表示一种紫色：

```
color: rgb(93%, 51%, 93%);
```

- 十六进制值。使用六位的十六进制值表示一种颜色，Photoshop（图2-3-4）和很多在线参考文档中都使用了这种方法。这种表示方法混合使用了数字和字母，每两位相应地表示红绿蓝三种颜色中的一种。十六进制值颜色表示方法已经在Web中使用了很长时间，所以大多数读者都比较熟悉：

```
color: #0000FF;
```

- 十六进制简写值。只有分别表示红绿蓝的三对字母均两两相同时，才能使用这种简写形式。例如#FF6699就可以使用简写形式[1]，但#808080则不能。上面例子中的蓝色的简写形式如下：

```
color: #00F;
```

- 系统颜色。系统颜色是在CSS 2.1规范中引入的一系列关键词，允许设计师使用浏览者计算机上的系统颜色配置。例如：

```
color: WindowText;
```

2.3.4 色彩丰富的结论

无论是在设计个人网站还是专业网站，都应该事先确定网站将要传达的主旨，并充分地了解浏览者群体的喜好。这些内容都将在选择颜色时助您一臂之力，也会让您更好地表达出作品的设计意图。

春天就是一个非常好的例子，它完全实现了作品在设计上的本意：表达出由冬到春的过渡，唤起浏览者平静且乐观情绪。作品还同时保证了来自全球的、各种文化背景、不同性别的欣赏者都能体会到同样的感觉，这一点已经毫无疑问地被禅意花园的浏览者所证实。

[1] 简写形式为#F69。——译者注

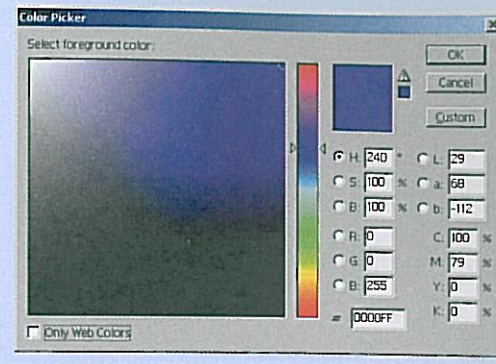


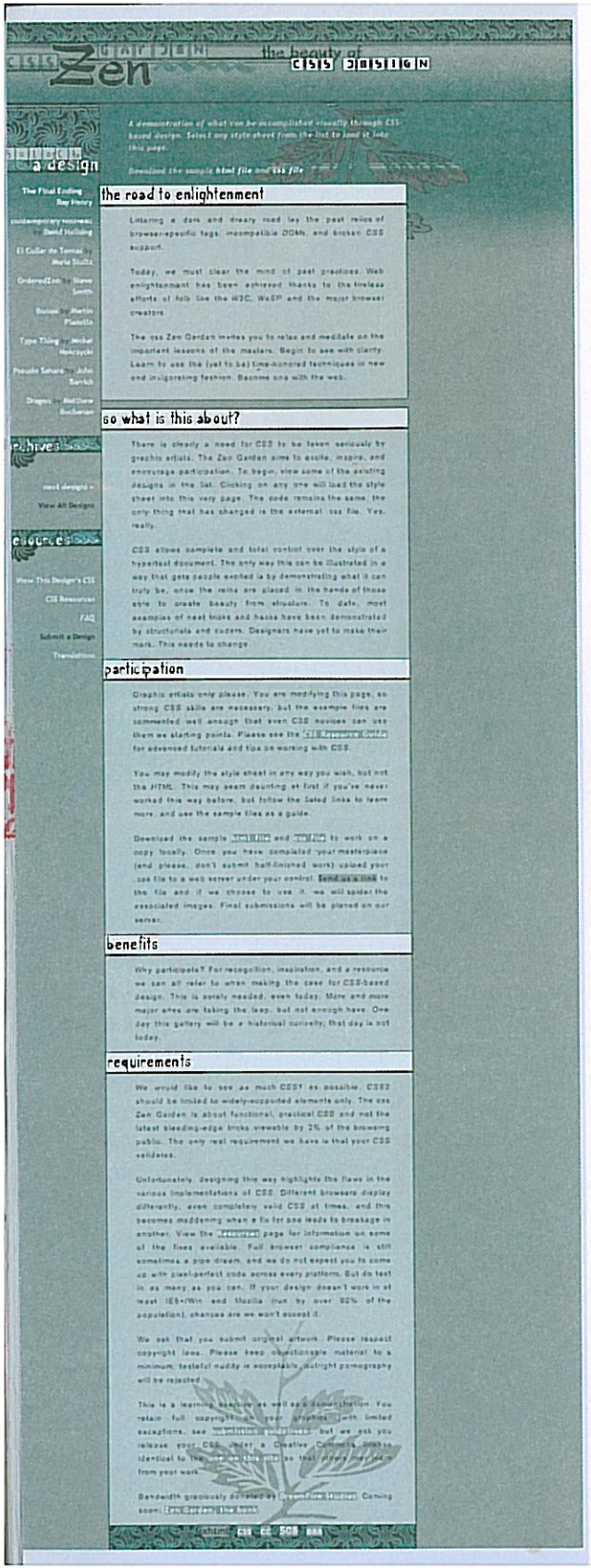
图2-3-4 使用Photoshop的颜色选择器得到十六进制的颜色值

Note

若想了解更多有关系统颜色的信息，请访问www.w3.org/TR/CSS21/ui.html#system-colors。注意，这些颜色将不会被CSS 3.0支持。

Note

很多人仍然坚持使用“Web安全（Web Safe）”的调色板。但由于目前主流的计算机都能够显示数百万种颜色，所以已经没有必要再受限于这个规定了，除非你确认作品是为某个特定的、使用古老计算机的人群所专门设计的。熟悉“Web安全”调色板的读者将很容易地发现，春天中选用的一些颜色也并不在Web安全所规定的256种颜色当中。



Laura MacArthur，设计师

www.csszengarden.com/022

2.4 Viridity（翠绿活力）

在图案、材质和对比度中找到平衡

翠绿活力的色彩基调是单调的绿色，但若着眼细节，将会发现作品让我们联想到了森林中长满苔藓的大地，或天鹅绒一般柔软的织物。Laura MacArthur也正是希望能用这个作品捕获住这份平衡、宁静的感觉。

通过使用重复的图案以及模拟的材质效果，MacArthur用丰富的细节平衡了作品其他方面的简单。深沉的单一色系让作品不至于变得太过拥挤，且虽然作品中使用了繁复的图案，但大片的绿色空白区域也给眼睛留下了足够的休息空间。

2.4.1 图案

翠绿活力中使用的图像主要是很多视觉上相似的图案。深入分析可以发现，这些图案在整个作品中都保持一致，主要分为两种类型：第一种由网点组成，而第二种则由类似植物形态的弯曲花纹组成（图2-4-1、图2-4-2）。两种图案均用Adobe Photoshop的滤镜将剪贴图案元素拼接而成。当然，这些图案在细微之处均有所变化，丰富细节的同时也不至于让作品的视觉效果过于一致，导致单调乏味。

图案是指重复填充到指定区域内的某一种（也可能多于一种）小块视觉元素。在各个视觉作品，例如时装、室内装潢乃至自然界（例如桔皮上的气孔、树叶上的脉络等）中，都经常能够找到图案的身影。

漂亮的图案是Web设计师最好的朋友。对于一片内容很少或者根本没有内容的区域，我们必须想些办法：某些作品选择使用单一的填充颜色，另一些作品则干脆直接留出一片空白。虽然这些方法都能在某种程度上解决问题，但在某些特殊情况下，我们还需要对这片区域添加更多的细节修饰。

当代Web设计中的一些限制引起了人们对图案的关注和使用。一个简单的小图片可以很容易地被重复填充到页面中的某个区域中，很小的图片也意味着在作品中添加这种细节修饰并不需要花费太多的网络带宽。例如，哪怕是一个只有500字节大小的图片，也能很容易地完全填充到某个900像素宽的大区域中。

CSS中内建的background-repeat属性就是用来设定重复显示选项的。background-repeat属性的使用方法非常简单，其可选值包括repeat、no-repeat、repeat-x以及repeat-y，分别代表让图像重复显示、只显示一次（非重复显示）、只在x方向重复显示以及只在y方向重复显示。

```
h3 {
 background-repeat: no-repeat;
}
body {
 background: #669999 url(z_bgrnd.gif) repeat-x 0px 1px;
}
```

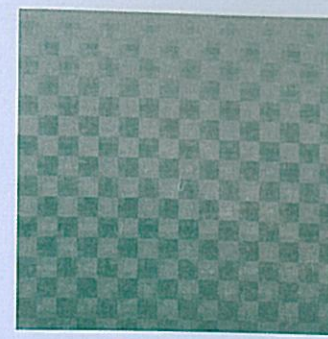


图2-4-1　近距离观察网点类型的图案

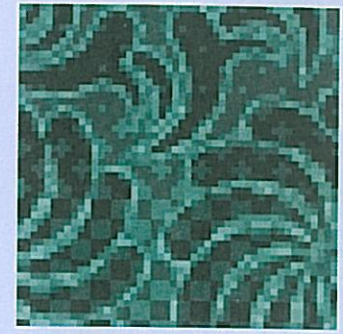


图2-4-2　近距离观察弯曲花纹类型的图案

图2-4-3　用雕刻手法为石像模拟出衣物的材质感

在前面这段示例代码中可以发现，我们可以独立指定background-repeat属性。该属性并不需要依赖于其他任何的背景（background）相关属性，也并不一定要写在冗长的background规则中。

2.4.2　材质

在浏览Web页面时，我们并不能用触觉感受到什么信息，通常来讲，整个计算机行业中也都是如此。虽然简单的力反馈功能在游戏手柄上较为常见，但除此之外触觉却并没有什么更进一步的用武之地。

在本世纪初的几年里，罗技（Logitech）公司实验性地发明了一种名为“iFeel”的鼠标产品，这种鼠标能够给手指一定的反馈，让操作系统中的元素以及网页带有一定的“材质”感。例如，将指针移到超链接上将会感受到类似在橡胶表面上移动鼠标的感觉。但最近一段时间这种技术似乎销声匿迹了，罗技公司也不再生产这种鼠标。虽然看上去不错，但却没有人知道这种技术到底有什么用，所以它也一直未能流行起来。

因此，在传统观念中Web并不能很好地体现材质：唯一的触觉反馈可能就是在点击按钮时。但材质却并不一定必须要通过触觉才能感受，视觉也能够体现材质感。

以人物雕像为例，设计师一般都会在石头上仔细雕琢出人物的衣物。我们自然可以想象到，若触摸这些石制的衣物，将只能感觉到光滑、坚硬的外表（图2-4-3）。但我们也都理解它要代表的是真实的衣物，因为二者在视觉上有很多的类似之处。

图案是体现材质感所必不可少的。通常情况下，无论是否是刻意而为，图案都能够模拟出材质的效果。很多材质的元素都自然地由图案的重复而体现出来。

虽然翠绿活力中的图案并不是由照片得来的，但这些图案却毫无疑问地模拟出了真实世界中的材质感。网点类型的图案让人联想到凸凹不平的表面，上面的小方块也让人感觉这像是一块带有方格纹的棉布。弯曲花纹类型的图案则给人以怀旧风格的地毯或华丽的布料的感觉。若以织物的角度体会，翠绿活力将很容易让人联想到一个诱人、舒适的长沙发。

2.4.3 对比度

选择像翠绿活力一样单调的色系在设计中是个非常冒险的行为，除非你经过了仔细考虑。但丰富的色彩和细节也可能导致潜在的对比度问题，这一点在设计时同样不容忽视。

对比度差别可以用来区分不同的元素。若对比度过低，则元素将失去彼此的界限并混合在一起，文字也变得难以辨认。若对比度过高，则将给浏览者带来压迫感，同样不会觉得舒适。

1. 高对比度

对比度过高将让作品显得过于拥挤，有过度设计之嫌。虽然翠绿活力中用到的颜色范围较窄，但材质和文本的合理使用却依然起到了提高对比度的作用。若颜色种类过多，作品看上去将充满了噪音，显得不够和谐（图2-4-4）。

几个世纪以来，各种出版物均是在白纸上印刷出黑色的文字。虽然彩色印刷的历史也不算短，但白底黑字仍旧是最好的对比度组合，读者也都对这样的组合非常习惯。一般来讲，白底黑字原则在计算机屏幕上同样是达到最佳对比度的不二之选。

但还是有些例外情况。真正的书本靠反射光线阅读，但计算机屏幕却是自身发光的，这种差别也给屏幕显示带来了一些特有的问题。例如，黑暗房间中的一个明亮的LCD屏幕将显得极为刺眼。在这种环境下，阅读任何纯白背景上的文字无疑都是对视力的巨大挑战。

考虑到对比度之后，使用颜色将变得愈加复杂。由于同时色对比（simultaneous contrast）[1]的原因，某些颜色的色块放在一起会格外显眼。若将两种互补色，例如橙色和蓝色或红色和绿色相邻放置，那么将显示出明显的边界以及强烈的对比效果（图2-4-5），进而引发令人不愉快的视觉上的紧张感。

但在某些需要表现浓烈色彩的情况中，我们也可以充分利用同时色对比所带来的效果。例如，黄色背景下的橙黄色将显得更接近橙色，但若是在橙色或红色背景下，则看起来会更接近黄色（图2-4-6）。在更加暗淡的颜色背景衬托下，偏暗的颜色也会变得明亮起来。例如，若是将某种比较暗淡的红色置于灰

[1] 同时色对比是指两种不同的色光同时作用于视网膜的相邻区域时，颜色视觉因受到影响所发生的变化。眼睛为了更加清楚地分辨出颜色差异，会尝试夸大颜色的色调、亮度和对比度等。——译者注

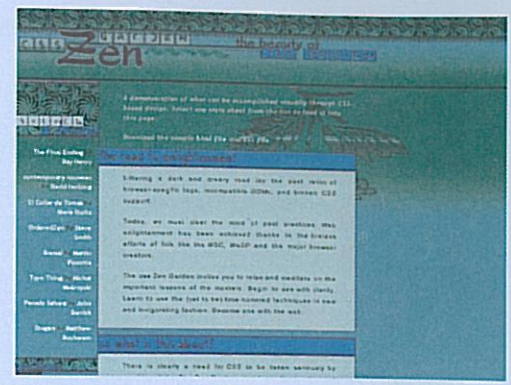


图2-4-4　若是在翠绿活力中使用更强的颜色对比度，将与丰富的图案产生冲突

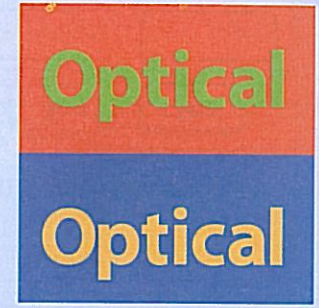


图2-4-5　同时色对比导致的颜色边缘强化

图2-4-6　同样的橙色，在黄色背景下显得偏橙，在红色背景下显得偏黄

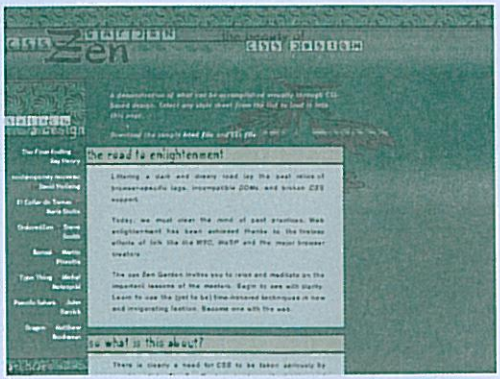


图2-4-7　对比度不足的翠绿活力

图2-4-8　使用较淡的按钮边框颜色会让其显得更细

色背景中，则该红色看起来会鲜艳许多。

2. 低对比度

若作品的对比度过低，则往往会给人带来压抑的感觉。在表现变化和差别时，我们需要适当的对比度。翠绿活力中选择了从明到暗的多种绿色，但若是这些绿色都趋于一致，那么整个作品将缺少了让人感觉眼前一亮和平静舒缓的区别，变得毫无生气，最终导致打破整个视觉上的平衡（图2-4-7）。

但低对比度也有其独到之处。在设计中，有些时候即使是一个像素也会显得太大，但因为像素已经是屏幕显示的最小单位，所以我们也没有任何办法画出一条比单一像素还要细的直线。这时若是降低这条线与其背景之间的对比度，则能够让其看上去变得更细一些（图2-4-8）。

用类似的方式降低设计元素之间的对比度同样也可以实现一种柔和的效果。例如，对于某段不是那么重要的文本，我们就可以为其应用一些较浅的颜色。对于某些非活动的图标或链接，我们同样可以通过降低其对比度来实现“变灰（grayed out）”效果，让用户了解该选项当前不可用。

但对于低对比度的使用仍旧需要格外小心。双眼视力都是5.0的读者在阅读浅灰色背景上的深灰色文字时可能没什么问题。但对于那些视力不那么好的读者来说，若想看清楚这段文字，将不得不凑近屏幕或者增大文本的字号。但无论在何种情况下，黑色的文字总是能够带来更好的可读性。

色彩的对比度同样会影响到内容的可读性，不同的色调对应的可读性也不尽相同。例如，纯黄色看起来就比蓝色显得淡一些。所以即使读者的双眼视力都是5.0，白色背景上的黄色文字依然基本上无法阅读。

3. 色盲

若想更深入地研究色彩对比度问题，那么自然会考虑到那些患有色盲的读者（图2-4-9A和图2-4-9B）。颜色方面的缺陷有很多种，每种也都有它的变化。最常见的可分为如下两大类：

- 红色弱/绿色弱——患有红色弱或绿色弱的人仍旧能够看到大多数颜色，但对不同颜色的分辨能力会有所减弱。红色弱/绿色弱也被叫做三色感知异常（anomalous trichromacy）。

□ 红色盲/绿色盲——这类的颜色感知障碍较为少见，但却更加致命。患者通常无法正常分辨出红色和绿色，他们的世界中只有黄色、棕色和蓝色。红色盲/绿色盲也被叫做dichromacy。

以上各种类型的色弱/色盲患者均部分甚至完全不能感受红色和绿色（也可能会有蓝色，但非常少见）。一般地，每12个人中就有一个存在着颜色感知障碍，且男性的发病率要远高于女性。

但若是某个患有颜色感知障碍的人在欣赏某个作品，且该作品中恰好包含了一些他无法感知的颜色的话，会出现什么问题呢？若是设计师想要依赖颜色传递某些重要的信息，那么在这些患有颜色感知障碍的浏览者眼中，其精心挑选出的调色板可能将变得毫无用处。例如，若某个患有色盲的浏览者无法看出红色和深棕色或黑色之间的区别，那么我们就不能给出诸如“点击红色按钮”的提示。虽然对正常人来讲区分颜色不过是小菜一碟，但在这类特殊的浏览者看来，很多时候某种颜色将和其背景完全混在了一起，区分二者成为了根本不可能完成的任务。

没有什么方法可以完全模拟出色盲患者看到的世界。将作品转化为灰度图像似乎是个不错的办法，但这也不是作品在大多数色盲患者眼中的真实样子。很多图像处理软件中都提供了各种各样的滤镜，用来模拟不同类型色盲患者眼中图像的样子，但这种模拟也仅仅是近似而已。颜色感知障碍存在着个体差异，每个人的症状都会有所不同。

既然针对颜色感知障碍浏览者的测试不可行，那么又该如何是好呢？“Web Content Accessibility Guidelines”中的第2条指导意见（www.w3.org/TR/WCAG10/#gl-color）给出了解答：对于任何依赖于颜色传递的信息，我们都应该提供如下方式中的至少一种来告知用户：蓝色且带有下划线的超链接、包含图标或文本的绿色按钮、带有实线边框的红色警告信息等。

2.4.4 统一

为了让作品的视觉效果趋于统一，在设计时就需要考虑到作品的各个方面。若是翠绿活力没有考虑到材质与色彩、文本与图案之间的相互影响，那么作品将变得过分张扬且拥挤不堪。但由于Laura MacArthur注意到了这些问题并仔细平衡了各种设计元素，作品的整体效果最终够远远地超过了其每一个组成部分。

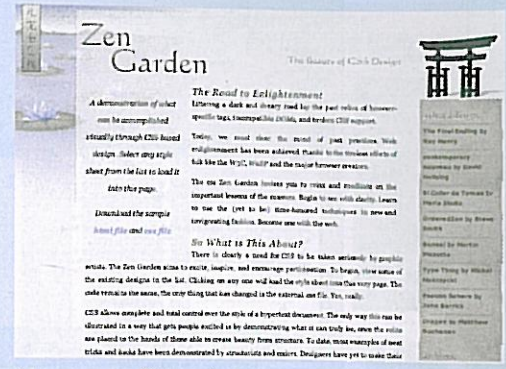
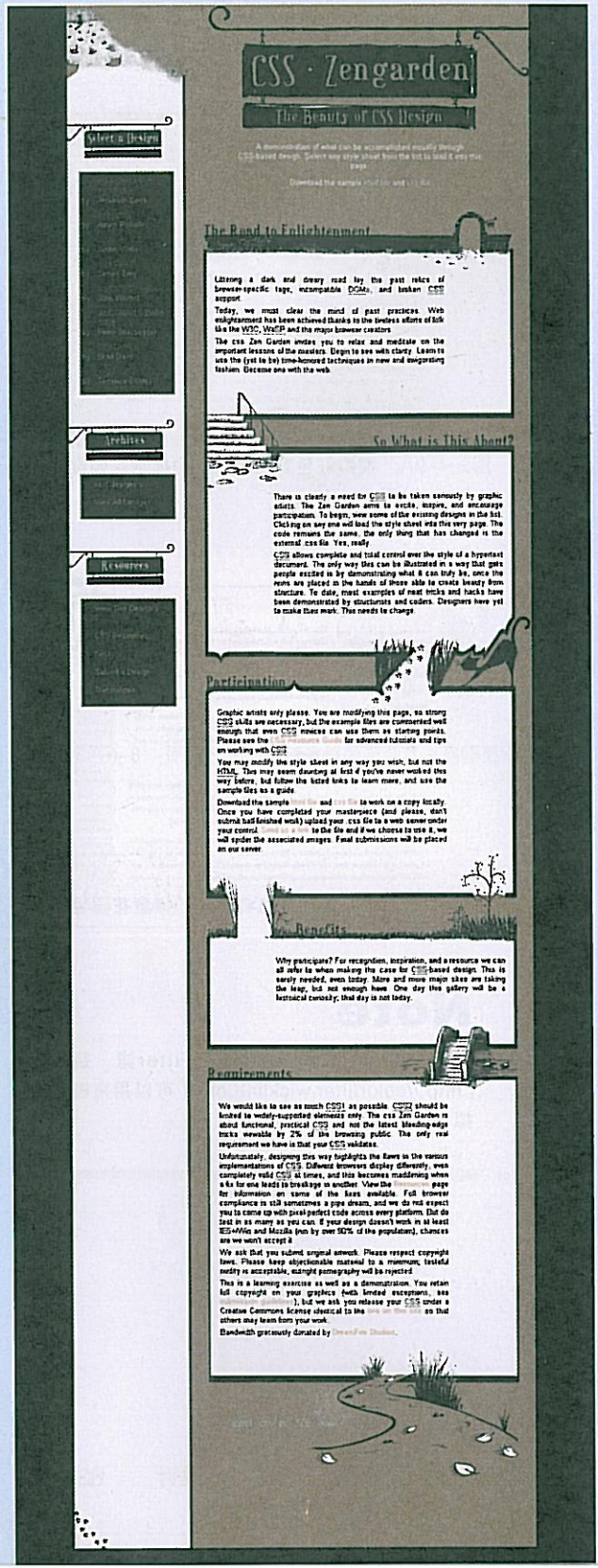


图2-4-9A 模拟红色盲患者眼中的禅意花园设计

图2-4-9B 模拟绿色盲患者眼中的禅意花园设计

Note

Colorblind Web Page Filter滤镜（http://colorfilter.wickline.org）可以用来模拟色盲患者眼中的图像。



Charlotte Lambert，设计师
www.csszengarden.com/068

2.5 Ballade（叙事曲）

用想象创造视觉流向并引导视线

禅意花园的本意就是让人们沉思，Charlotte Lambert的视觉作品叙事曲也正是为了表达这种感受而作。Lambert在她的花园中来了一次精神上的散步。想象在花园中自由飞驰，一路上她遇到了很多具有象征性的东西：一段楼梯、一座小桥、一只闲逛的猫。

从想象回到现实，Lambert创建了几幅草图来帮助自己完善关于作品的想法。她在这些草图上手工绘制出了旅途中的所见，然后用中国的墨水和白色的颜料在纸上，甚至是啤酒包装箱上开始了创作。之后，Lambert在Photoshop中将它们组织起来，最终完成了将用于作品中的图像。

2.5.1 引导视线

一个良好的作品应该能够引导读者的视线，让读者以合理的逻辑顺序逐一阅读内容中的重点，但同时又不会让读者觉察出来。这一点至关重要，既表达了设计师希望传达的意图，也不会让读者觉得被控制。

为了引导读者的视线，我们必须引出视觉流向的概念。视觉流向是指通过使用图片和内容，让读者的视线从一个重要地点移动至下一个的过程。在叙事曲中，我们就能看到这样一条清晰且相当优雅的视觉流向。

网站配置导航条、作品存档、资源链接等起辅助导航作用的内容都被统一放置在了页面的左边，与主要内容分开（图2-5-1）。大多数人在浏览网页时都习惯于在页面的左边或上边寻找导航条，所以将导航条布置在这些位置上将让浏览者易于发现，进而迅速转向页面的主要内容。

在Lambert的作品中，我们不停地被从一段中带入到另一段，就像真的走在花园小路上一样。我们从门廊（图2-5-2）开始，经过了第一段之后走下台阶（图2-5-3）来到了第二段。花园中的每一段都有着独一无二的样子，并为内容提供了足够的空间。标题部分描述了每一段的概要，让该段的内容与视觉流向融为一体。

在引导读者的视线时，还有一个重要之处，即为读者的眼睛提供一些休息的空间。达到这个目的的最常见作法就是在页面中添加足够、适当的空白（图2-5-4）。颜色可以用来将读者引导至页面中的关键部分，而形状和线条则可以为读者的视线移动指明方向。

在叙事曲中，设计师用块状的白色来表示主要内容区域。这在提供足够的、易于阅读的对比度的同时，也给内容区域中保留了足够的空白让眼睛休息。作品的颜色选择极为简单，只使用了棕色、黑色和白色，似乎为我们留下了这样的疑问：这是在冬天的花园中么？还是秋天的？带着这些问题，读者将会饶有兴致地欣赏整个作品并尝试去寻找更多的线索。

最终，树叶（图2-5-5）、小溪、拱桥和脚印（图2-5-6）也为作品增添了一些非同寻常的元素，将我们的视线引至远方，并留给我们无尽的美好遐想。

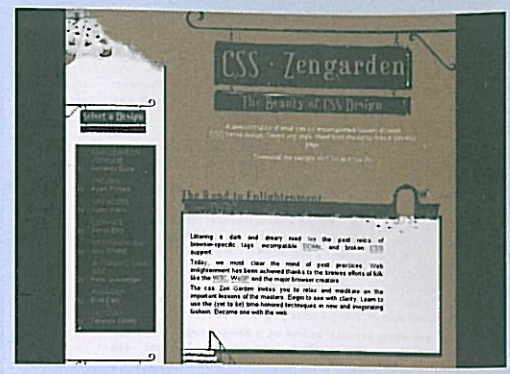


图2-5-1　在叙事曲中，导航和辅助信息与内容区域完全分开。采用这种方式，作品可以很容易地将读者的视线吸引到其内容上

图2-5-2　叙事曲中的门廊。这个小插图让整个作品显得生动活泼

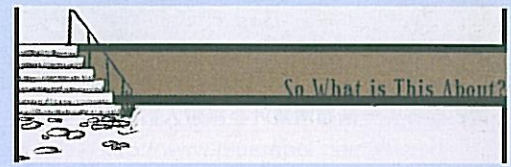
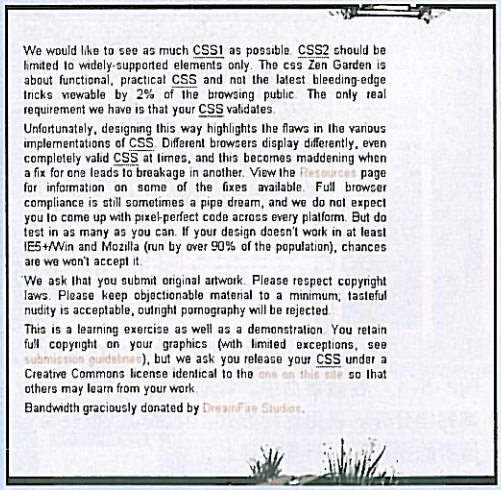


图2-5-3　从门口到台阶，以充满技巧性的暗喻方式引导了读者的视线



We would like to see as much CSS1 as possible. CSS2 should be limited to widely-supported elements only. The css Zen Garden is about functional, practical CSS and not the latest bleeding-edge tricks viewable by 2% of the browsing public. The only real requirement we have is that your CSS validates.

Unfortunately, designing this way highlights the flaws in the various implementations of CSS. Different browsers display differently, even completely valid CSS at times, and this becomes maddening when a fix for one leads to breakage in another. View the Resources page for information on some of the fixes available. Full browser compliance is still sometimes a pipe dream, and we do not expect you to come up with pixel-perfect code across every platform. But do test in as many as you can. If your design doesn't work in at least IE5+/Win and Mozilla (run by over 90% of the population), chances are we won't accept it.

We ask that you submit original artwork. Please respect copyright laws. Please keep objectionable material to a minimum; tasteful nudity is acceptable, outright pornography will be rejected.

This is a learning exercise as well as a demonstration. You retain full copyright on your graphics (with limited exceptions, see submission guidelines), but we ask you release your CSS under a Creative Commons license identical to the one on this site so that others may learn from your work.

Bandwidth graciously donated by DreamFire Studios.

图2-5-4　每一块内容中都提供了大量的白色空余空间，既让眼睛有休息的余地，也增加了内容的可读性

图2-5-5　一条布满落叶小路带人们走出了花园

2.5.2　CSS的思考

最近几年中，一个有趣的现象就是出现了一大批创新的、不愿按照守旧派的方式使用表格布局的设计师。作为table的替代品，他们认为CSS应该承担起布局并呈现样式的重任。虽然所有禅意花园的作品都力求说服人们使用CSS进行布局，但仍有那么一些守旧的设计师固执己见，坚持使用复杂的table布局。在叙事曲中，设计师完全能够直接用CSS实现预期的设计效果。但若是选择table布局，情况又将如何呢？

1. 守旧派的悲哀

让我们来看看守旧派眼中应该如何实现这个作品。你将很容易发现在该作品中使用表格布局将是一次噩梦般的尝试。若是在Macromedia Fireworks中用图片分割工具进行布局，哪怕仅仅是从作品中取出两个主要的小节，你也将得到一个复杂的表格加上其中大量的分割图片：

```
<table bgcolor="#ffffff" border="0" cellpadding="0"
cellspacing="0"
➥width="597">
    <tr>
        <td rowspan="3"><img name="ballade_r1_c1"
        ➥src="images/ballade_r1_c1.gif" width="30"
        height="887"
        ➥border="0" alt=""></td>
        <td><imgname="ballade_r1_c2" src="images/
        ballade_r1_c2.gif"
        ➥width="537" height="273" border="0" alt=""></td>
        <td rowspan="3"><img name="ballade_r1_c3"
        src="images/ballade_
        ➥r1_c3.gif" width="30" height="887" border="0"
         alt=""></td>
        <td><img src="images/spacer.gif" width="1"
        height="273"
        ➥border="0" alt=""></td>
    </tr>
    <tr>
        <td><img name="ballade_r2_c2" src="images/
```

```
        ballade_r2_c2.gif"
        ➥width="537" height="429" border="0" alt=""></td>
        <td><img src="images/spacer.gif" width="1"
        height="429"
        ➥border="0" alt=""></td>
    </tr>
    <tr>
        <td><img name="ballade_r3_c2" src="images/
        ballade_r3_c2.gif"
        ➥width="537" height="185" border="0" alt=""></td>
        <td><img src="images/spacer.gif" width="1"
        height="185"
        ➥border="0" alt=""></td>
    </tr>
</table>
```

查看上述代码，很容易发现许多明显的复杂之处，包括：

- 表格布局添加了大量的不必要标记。
- 使用间隔图像[1]，而不是CSS中的内外边距设定，不必要地增加了页面的文件大小。
- 图片是用img标签添加到页面中的，这些额外的img标签增加了页面的文件大小，导致页面加载需要更长时间。
- 既然选择了表格布局，那么网站中的各个页面也必须统一使用table。每个页面都包含着类似的table和img标记，因此每次页面加载时浏览器都必须重新请求这些内容并进行重绘[2]。但若是使用CSS，这些布局信息即可缓存于浏览器中。例如在禅意花园的作品中，应用图片的代码只需要加载一次。

自然，这些都是表格布局中遇到的常见问题。绝大多数设计师也都了解过度使用表格导致的另外一些问题，包括可访问性——很多为盲人准备的屏幕阅读设备都无法正确理解采用表格布局的页面。再来考虑一下叙事曲中最重要的设计

[1] 间隔图像是在table布局中常用到的一个技巧，通常是一个1×1大小的透明GIF图像，用在撑大表格等辅助布局中。——译者注

[2] 事实上，浏览器都会自动缓存<img>标签引用的图片。反而有些浏览器（例如IE6）无法缓存定义于CSS中的图片。——译者注

图2-5-6　一只小猫守望在作品的左上角，让叙事曲充满了令人愉悦、引人入胜的细节，欣赏者也总是能够在不经意之间找到小小的惊喜

Note

虽然从总体说来，CSS布局的呈现速度以及友好性要强于过度复杂的表格布局，但它也存在着一个常见的问题——FOUC（无样式内容瞬间，Flash of Unstyled Content）。在IE浏览器中，若以import而不是link的形式引入CSS，则将发生FOUC现象。但只要在HTML文档的head部分存在着一个script或link元素，即可避免FOUC。禅意花园选择了用script元素来避免这个问题，让页面加载更为平滑。若想了解更多FOUC的相关内容，请参考：http://www.bluerobot.com/web/css/fouc.asp

Note

若图片实际上起到了标题的作用，那么将其包围在标题中也就显得恰如其分。禅意花园是用“图像替换”技术来实现的，关于“图像替换”，将在第4章中详细介绍。

Note

Mozilla和Mozilla Firefox的Web Developer工具可以在此免费下载：http://chrispederick.com/work/firefox/webdeveloper。

意图——引导读者视线，我们就会发现使用表格将导致的另外一个问题：因为table布局页面的加载过程通常都不会那么平稳自然，所以这种笨拙守旧的布局方式将很有可能扰乱作品的视觉流向。

2. 创新派的设计流程

但若是使用了CSS，那么我们很容易就能想象出让作品平滑加载且对用户友好的方法。设计师不必再陷入复杂的表格、不规则的图像切片或是间隔图像中，而是可以相对独立地为页面元素设置样式。

但这并不是说在基于CSS的设计中就不能使用嵌入到HTML中的img标签。若图像属于“内容”，而不是“样式”的一部分，例如“公司介绍”页面中的一张CEO照片或是软件网站中某个产品的屏幕截图，则显然应该使用img标签。使用CSS的关键之处在于将页面的表现样式从页面结构中分离开来——页面的表现样式属于设计部分，而并不属于内容。

在下面的代码中，可以看到叙事曲的主要内容段部分是如何将图像以背景的形式应用到段首标题之上的：

```
#preamble h3 {
    background: url(road.jpg) top left no-repeat;
    height: 106px;
    }
#explanation h3 {
    background: url(about.jpg) top left no-repeat;
    height: 168px;
    }
#participation h3 {
    background: url(particip.jpg) top left no-repeat;
    height: 154px;
    width: 565px;
    }
#benefits h3 {
    background: url(benef.jpg) top left no-repeat;
    height: 171px;
    }
#requirements h3 {
    background: url(req.jpg) top left no-repeat;
```

```
height: 125px;
}
```

在Mozilla和Mozilla Firefox的Web Developer扩展工具中，我们可以让其自动标出页面中的块级（block level）元素，并查看其id与class（图2-5-7）。图中可以看到，这个工具能够帮助我们容易地找到页面元素的精确位置。

通过将内容置于HTML页面中并使用背景图像创建样式，作品将能够得到很多极为明显的改善：

- 通过改进标记方法，降低了页面文件大小。
- 由于CSS将被浏览器缓存，所以若是将大多数（甚至所有）的样式标记置于CSS中的话，浏览器将不必每次都从服务器端重复取得页面的样式，也自然大大加快了页面的加载速度。
- 网站的可访问性得到了立竿见影的增强，因为HTML中再没有任何表示样式相关的代码。
- 网站更加易于管理。比如说春天来了，我们想给叙事曲添加一些绿色和金色的元素，让作品不再是现在这样冷冰冰的感觉。按照现在的设计方式，我们只要修改CSS以及其使用到的图片即可，具有相当的灵活性。修改这种CSS布局的网站要远远简单于表格布局的网站，即降低了管理费用。

2.5.3 寻找跨越的桥梁

寻找一种有效的Web设计方法并不简单。设计师必须全方面地考虑许多问题。在叙事曲中，我们学到了如何让作品充分引导读者的视线，如何用色彩表达某种特定的感觉，以及为什么CSS布局要远强大于表格布局等很多知识。CSS让设计师挣脱了传统技术的桎梏，来到了一个全新的世界。

对于那些正从table布局转向CSS布局的设计师来说，最大的挑战就是如何在这种新技术下找到灵感。虽然CSS让我们从前人设计的限制中解放了出来，但Photoshop、HTML标记和样式仍旧是设计中不可或缺的环节。挑战实际上在于如何找到新旧技术间跨越的桥梁。Charlotte Lambert的叙事曲给出了足够的启发，让作为设计师的我们找到了这座桥梁，展开想象的翅膀，使用最新的技术从传统的Web设计方式中飞跃至最新的CSS设计理念中。

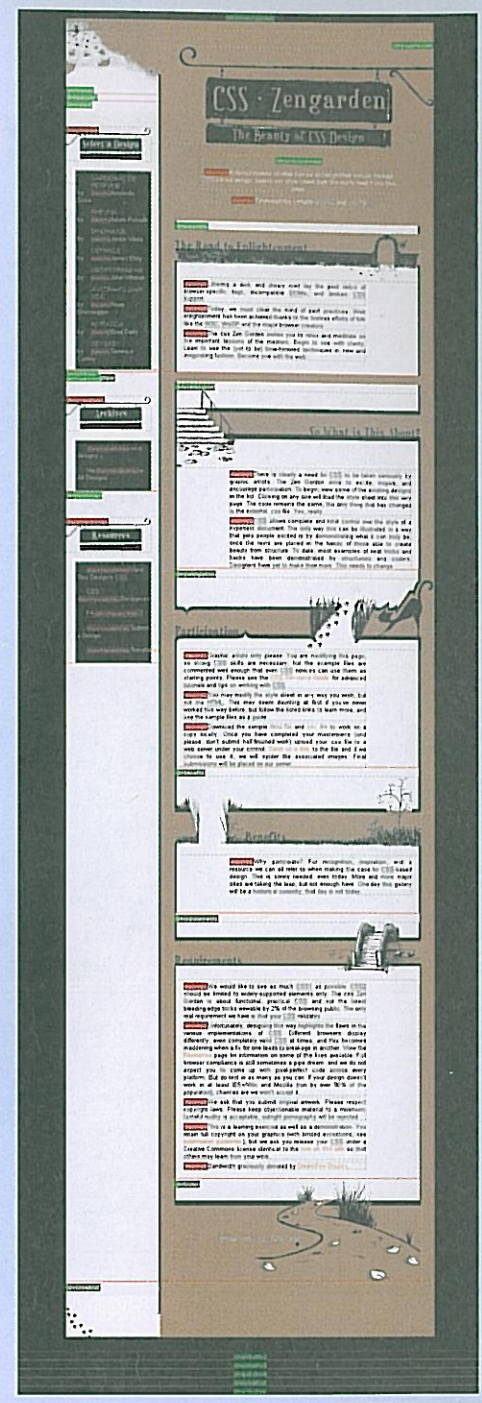
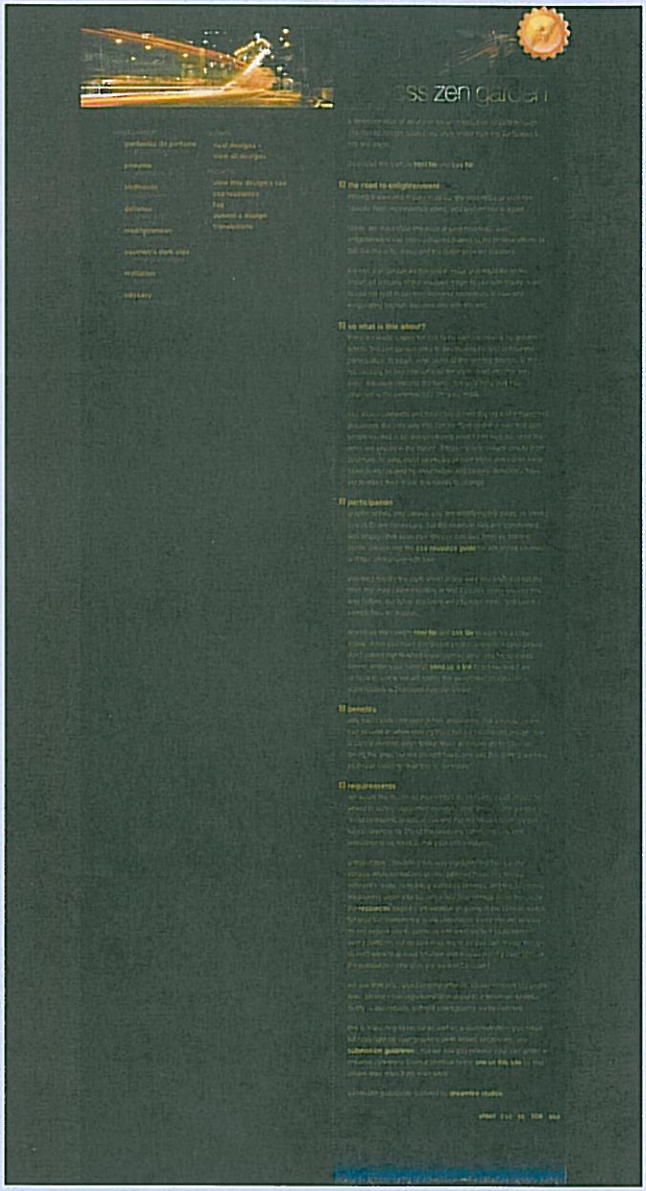


图2-5-7　使用Mozilla和Mozilla Firefox的Web Developer扩展工具查看叙事曲的页面。这里标出了每个块级元素的位置，演示了设计师布局和使用背景图片的方法



Dave Shea，设计师
www.csszengarden.com/064

2.6 Night Drive

将网站设计变为代码，并解决转换中遇到的问题

Dave Shea驾车在夜幕刚刚降临的城市中行进，耳边传来John Coltrane的轻柔的萨克斯曲调，黄色的路灯闪过车窗，外面是深蓝色的天空。很快，他在这纯美的夏夜中找到了Night Drive的创作灵感。

深颜色确定了作品模糊、温暖的基调。文本和图像所使用的深橙色以及零星点缀的亮橙色平衡了作品的色调，让作品不至于显得过于沉闷。散布在作品各处的淡蓝色小块同样起到了很好的补充作用，表现朴素安静的同时也避免显得过于单调。

作品中的照片是根据其色彩和主题选择的——行驶在刚刚日落的街道中，你会看到昏黄模糊的灯光、玻璃一样的水面和朦胧的蓝色地平线。

2.6.1 将设计图转化为代码

最初在Adobe Photoshop中设计Night Drive时，Shea几乎没有考虑过如何用CSS实现的问题。设计图首先完成，这样，任何有关实现的问题就只能在编写CSS的过程中解决了。

应该如何将设计图转化为具有交互性的、真正的网站呢？首先，设计图应该分割成片存放，然后再用CSS将其组装起来。图片分割的过程没有什么硬性的规定，当然也没有什么放之四海皆准的法则，只要把握住最重要的一点——灵活性即可。事先多考虑几种方法，将会让你在作决定时游刃有余。

2.6.2 选择

首先考虑Night Drive的页首标题部分——页首由一系列方框组成，右上方还覆盖着一个闪闪发光的、带有立体效果的黄色徽章（图2-6-1）。这里我们将遇到两个挑战：首先是如何布局这些照片和线条，其次就是如何将背景透明的徽章图片置于最上方。仔细斟酌之后，Shea找到了如下两种方案：

- 将整个页首部分设计图保存成单独的一张JPG图片。但图片只能轻微压缩，以保证图像的质量。当然，这种方法带来的图像文件势必较大。
- 将页首部分的图像分割为若干个小图片，然后再将图片添加至#pageHeader中的各种元素，或是结尾部分的#extraDiv元素（使用绝对定位将这些元素放置在页面顶部）上，最后把这些小图片重新组装起来（图2-6-2）。

最终Shea选择了第二种方案，但也采用了一些第一种方案中的想法：Shea没有直接将这一大块图案简单分成几个小块，而是有选择性地将两幅照片图像存为了JPG文件（图2-6-3）。标题文字“css Zen Garden”以GIF格式保存，在这些图片后面、作为布局背景的衬底也被存为一个高度压缩的GIF图像。从设计图到代码的过程竟如此崎岖——我们仅仅是想把页首部分正确显示出来而已，最终结果却是混杂使用了两种布局方案，图像也根据需要分别被保存为GIF和JPG两种格式。

保存完这些图片文件之后，接下来的一个问题就是如何将它们再次组装起来：这些图片都分别应该应用到哪个HTML元素之上呢？在明智地选择使用background-image属性和图像替换技术之后，答案显而易见：#pageHeader中没有足够多的元素，我们还需要使用定义在HTML文档末尾的#extraDiv，并将其

图2-6-1　Night Drive的页首标题部分

图2-6-2　标题被分割为若干独立的切片

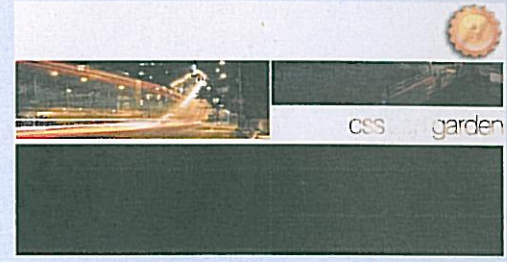


图2-6-3　照片和文字与背景图片分开保存

放置于页面的顶部。最后，Shea将最大的那张背景图片与h1元素绑定起来，并将标题文字“css Zen Garden”应用到了h2之上。

下面代码中的高亮部分演示了为元素应用背景图像的方法：

```
h1 {
  width: 770px;
  height: 166px;
  margin: 0;
  background: #000 url(bg2.gif) top left no-repeat;
}
h2 {
  width: 244px;
  height: 42px;
  background: #000 url(h1.gif) top left no-repeat;
  position: absolute;
  top: 120px;
  margin: 0;
  margin-left: 495px;
}

h1 span, h2 span {
  display: none;
}
```

上面代码中最下面的一个样式隐藏了span中的文本，因为其内容将被图像所替代。但目前已经不建议再使用这种方法了。

Note

关于图像替换技术的详细使用方法以及使用中的各种问题，请参考这篇文章：“Using Background-Image to Replace Text”（www.stopdesign.com/articles/replace_text）。

2.6.3 绝对定位技巧

从布局角度说，将本来在HTML结尾部分的照片图像定位到页首的确是个不小的挑战，这部分内容则为实现该效果打好了基础。绝对定位在处理居中时并不那么方便。Shea使用了一种非常有用的技巧来处理这个问题：首先将父元素绝对定位于页面的最左边，并将其宽度指定为100%，在水平方向占满整个浏览器窗口。然后将其子元素——这里就是那些span——设置为居中。

考虑到很多浏览器的缺陷，这部分代码如下：

```
#extraDiv1 {
 position: absolute;
 top: 41px;
```

```
 left: 0;
 text-align: center;
 width: 100%;
}
#extraDiv1 span {
 background: transparent url(granville.jpg) top left no-repeat;
 display: block;
 margin-left: auto;
 margin-right: auto;
 height: 123px;
 width: 770px;
}
```

若是希望子元素绝对居中的话，那么这样就算大功告成了。但若并不想要其绝对居中，需要加上些偏移量，又该如何处理呢？为子元素添加一定的内边距（padding）能够增加元素的总宽度，这样即可让其偏离原本的绝对中心位置。但这个偏移量的值在指定到内边距上时要加倍计算——因为在给元素添加内边距的同时也增加了元素的总宽度。还有，内边距要添加在与期望偏移方向相反的一侧，例如若要让在正中央图像向左移动，那么应该为图像添加右侧的内边距值（指定padding-right属性），反之亦然。

2.6.4 垂直分割线的花招

完成页首部分的布局之后，Shea在把它连接到页面其他内容时又遇到了麻烦。Shea想要用两条平行线将页面的左右两栏分开，但将背景图像垂直重复应用到左栏或右栏的方法却都行不通——左边的一栏（#linkList）的高度不够撑满页面，而右面一栏实际上却是由三个独立的div元素（#quickSummary、#preamble和#supportingText）组成的。

因为右栏中没有一个作为容器的最外层元素，所以Shea将这幅两条平行线的背景图片分别应用到了组成右栏的全部三个div上。Shea假定这三个div元素有着同样的宽度，且水平对齐。

```
#quickSummary, #preamble, #supportingText {
 background: transparent url(bg4.gif) top left repeat-y;
 margin: 0 0 0 400px;
 width: 342px;
}
```

这三个div元素的确水平对齐，但还是出现了一个意料之外的问题：这些

Note

内边距值需要二倍于期望的偏移量。例如指定了100px的padding-right，在中心坐标系中将只会让元素向左移动50px。

由于元素居中放置，且添加内边距值也同样增加了元素的宽度，所以元素将同时向左右两个方向移动相等的距离。

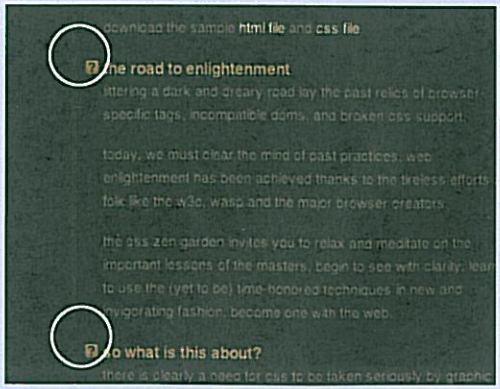


图2-6-4　div之间导致问题的缝隙

Note

外边距重叠定义与CSS 2.1规范中：两个或更多个块级元素（内容必须不为空，且没有内边距或边框的定义，也没有应用清除浮动样式）的相邻（外部相连或内部嵌套）外边距将重叠成一个单一的外边距。（www.w3.org/TR/CSS21/box.html#collapsing-margins）

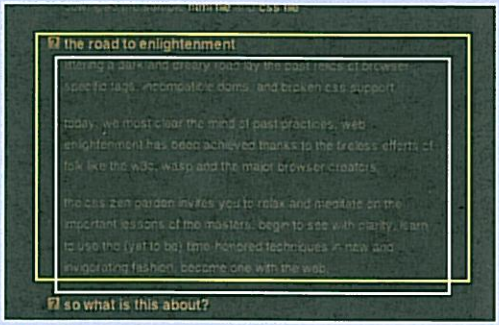


图2-6-5　段落元素“戳开”了包含它的div元素的底部

div两两之间都有一条水平缝隙（图2-6-4）。我们并没有为其指定任何的外边距（margin）值，那么这些缝隙又是从何而来的呢？

2.6.5　外边距重叠

CSS盒模型（box model）中有一个较为晦涩的概念——外边距重叠（collapsing margin）。外边距重叠是指，若两个元素上下毗邻且都定义了不为0的外边距值，同时又没有任何内边距、边框等设定，那么这两个元素之间的距离将小于二者外边距的和[1]。因此，两个都带有20px外边距的元素垂直相接并不会显示出40px的缝隙——两个外边距会重叠在一起，最终元素之间的距离只有20px。

但这和Shea遇到的那个缝隙问题又有什么关系呢？原因在于，重叠效果将同时作用于父元素和子元素之上——由于这三个div都没有设定外边距，所以每个div的底部外边距将与其最后一个子元素（即段落元素p）的底部外边距重叠。因为总共的外边距值并不为零，所以其最后一个子元素的10px外边距值仍然要被计算进去，看上去就像是子元素“戳开”了包含它的div父元素（图2-6-5）。

搞清楚产生原因之后，问题也就非常容易解决：为这三个div元素添加1px的垂直内边距，这样将不再满足形成外边距重叠的条件，div之间的缝隙自然也就消失了：

```
#quickSummary, #preamble, #supportingText {
 background: transparent url(bg4.gif) top left repeat-y;
 margin: 0 0 0 400px;
 padding: 1px 0;
 width: 342px;
}
```

2.6.6　图片使用技巧

现在页首和正文部分都已经完成，该考虑如何将那个黄色的徽章定位在页面右上角了。在将图片覆盖在其他图片上时，选用带有透明区域（这里是徽章投下的阴影）的图片所带来的好处就是无须考虑被覆盖图片上是什么内容。Shea自然可以选择在Photoshop中将原稿的各图层合并，然后将徽章以及其周围的阴影、被覆盖图片内容一起保存为GIF或JPG格式，这样即可保留住徽章漂亮的阴影（图2-6-6）。当然，Shea也可以抛弃掉

[1] 对于都为正数的两个外边距定义来说，两个元素之间的距离为二者外边距距离中较大的一个，即外边距要尽可能地重合。——译者注

那部分周围被覆盖的图片，只是将徽章本身保存为透明的GIF格式，但这样却无法留住徽章周围的阴影（图2-6-7）。

但因为对作品来说，这个徽章并不是“必不可少”的，所以Shea并没有使用上述两种方法中任何一种，而是将其存为了PNG格式，这样也免去了很多不必要的麻烦。因为无论放置于什么地方，带有256级透明度的PNG图像都能完美地显示出阴影效果。图2-6-8和图2-6-9就演示了该PNG图片在两种不同背景上的样子。

PNG格式将在本书稍后部分详细介绍，目前为止需要指出的就是，Windows平台下的IE浏览器并不能显示其透明度信息[1]，因此，我们就要想办法在IE中隐藏该徽章。最简单的实现方法是使用IE同样不支持的子孙选择器（child selector）[2]，这样即可有选择性地只将该徽章呈现在能够正常显示其透明度信息的浏览器中。

```
html>body #extraDiv4 span {
 background: transparent url(seal.png) top right no-
 repeat;
}
```

但显然，对于那些更为正式的Web项目来说，即使是那些非“必不可少”的图像也必须同样显示于IE中，以保证站点的严肃和统一。虽然GIF格式中简单的“开/关”透明度设置[3]会带来不少限制，但如果仔细规划的话，仍旧有办法实现所需的功能。禅意花园鼓励这种实验性质的超前尝试，但你的客户或许并不能如此“通情达理”。

2.6.7 迈出小小的一步

在规定期限内，仅仅从设计图开始，直至编写出完全基于CSS布局的Web页面的确是个不小的挑战。成功的关键在于将其合理地分割成一些较小的任务——例如，对于Night Drive的页首部分布局，我们就将其分为了两个步骤进行。

事先多考虑几种可能，将使您在问题到来时更加沉着冷静，并做出最正确的决定。对于任何问题，例如改变页面中元素的位置、调整元素的外边距或是为元素添加某种特效等，这一条都是颠扑不破的真理。

[1] 最新发布的IE7.0已经能够正常显示出PNG图像的透明度信息。——译者注

[2] 最新发布的IE7.0已经支持了子选择器。——译者注

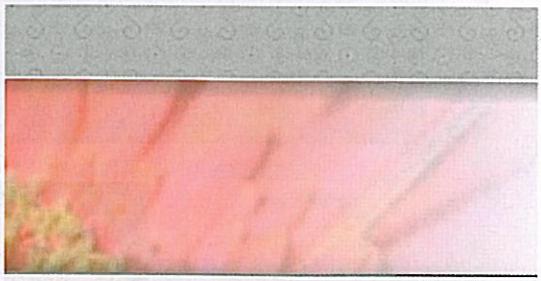
[3] GIF格式只支持设定某个像素是否完全透明，但并不能够设置其具体的透明度，例如30%透明等。——译者注

图2-6-6 徽章与其周围背景一起保存，没有透明度设定

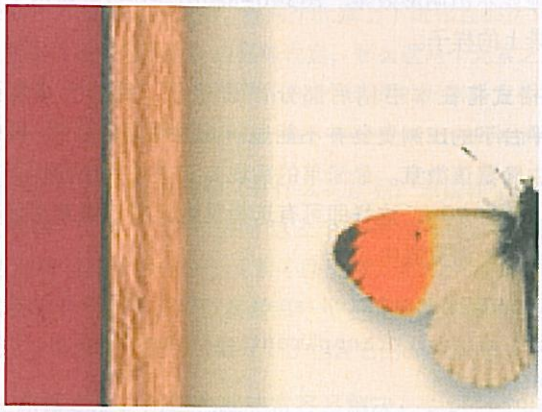
图2-6-7 去掉背景只保留徽章本身，受限于GIF对透明度的支持，阴影部分的效果并不理想

图2-6-8 透明PNG格式的徽章覆盖在白色背景上

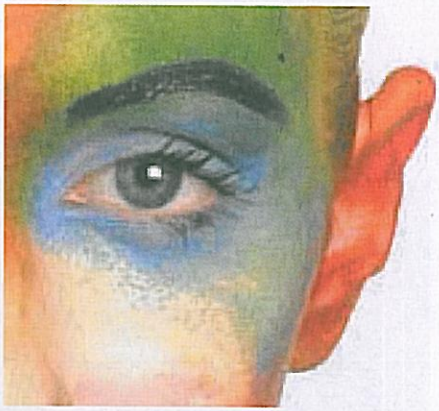
图2-6-9 透明PNG格式的徽章覆盖在黑色背景上



Backyard（庭院），78页



Entomology（昆虫学），84页



prêt-à-porter，96页

CS(S)Monk，102页

第3章

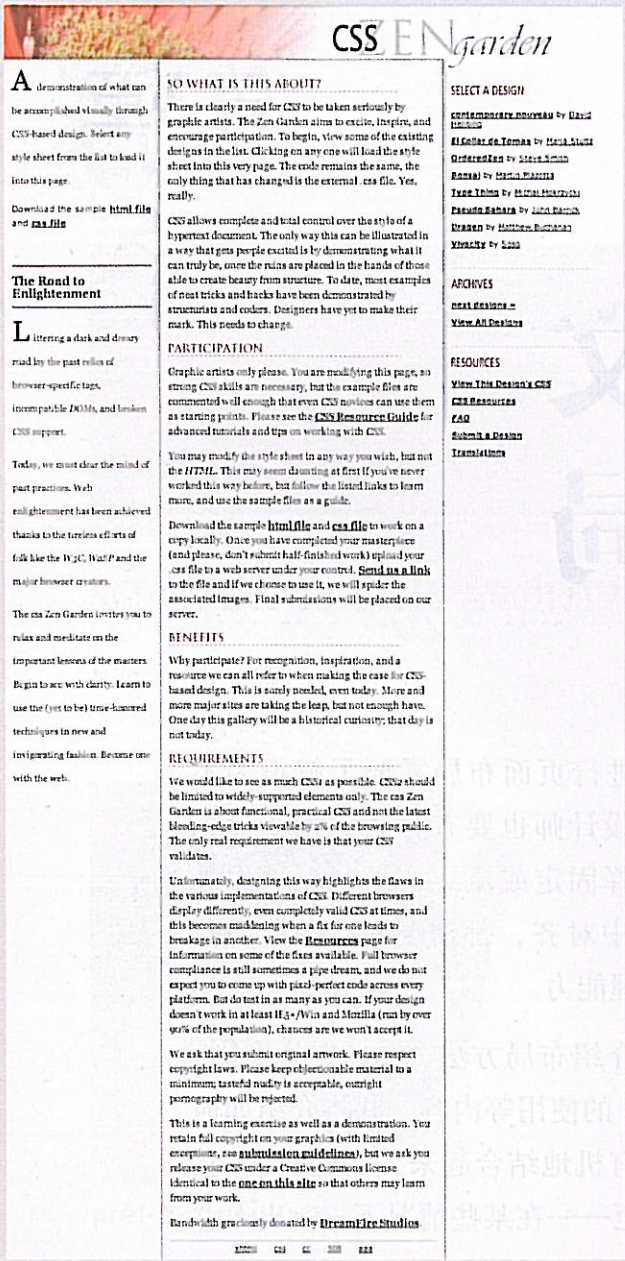
正文布局

White Lily（白百合），90页

Not So Minimal，108页

用CSS进行页面布局需要了解很多不同的技术，设计师也要有相当的创造性思维。无论选择固定或流式布局，无论作品靠左还是居中对齐，都需要对CSS布局有着良好的把握能力。

本章将介绍布局方法、CSS定位准则、浮动（float）的使用等内容，并将介绍如何将这些概念有机地结合起来，创建出实用、引人注目甚至——在某些情况下——出人意料的效果。



CSS ZEN garden

A demonstration of what can be accomplished visually through CSS-based design. Select any style sheet from the list to load it into this page.

Download the sample **html file** and **css file**

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support.

Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators.

The css Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

SO WHAT IS THIS ABOUT?

There is clearly a need for CSS to be taken seriously by graphic artists. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The code remains the same, the only thing that has changed is the external .css file. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the rules are placed in the hands of those able to create beauty from structure. To date, most examples of neat tricks and hacks have been demonstrated by structurists and coders. Designers have yet to make their mark. This needs to change.

PARTICIPATION

Graphic artists only please. You are modifying this page, so strong CSS skills are necessary, but the example files are commented well enough that even CSS novices can use them as starting points. Please see the **CSS Resource Guide** for advanced tutorials and tips on working with CSS.

You may modify the style sheet in any way you wish, but not the *HTML*. This may seem daunting at first if you've never worked this way before, but follow the listed links to learn more, and use the sample files as a guide.

Download the sample **html file** and **css file** to work on a copy locally. Once you have completed your masterpiece (and please, don't submit half-finished work) upload your .css file to a web server under your control. **Send us a link** to the file and if we choose to use it, we will spider the associated images. Final submissions will be placed on our server.

BENEFITS

Why participate? For recognition, inspiration, and a resource we can all refer to when making the case for CSS-based design. This is sorely needed, even today. More and more major sites are taking the leap, but not enough have. One day this gallery will be a historical curiosity; that day is not today.

REQUIREMENTS

We would like to see as much CSS1 as possible. CSS2 should be limited to widely-supported elements only. The css Zen Garden is about functional, practical CSS and not the latest bleeding-edge tricks viewable by 2% of the browsing public. The only real requirement we have is that your CSS validates.

Unfortunately, designing this way highlights the flaws in the various implementations of CSS. Different browsers display differently, even completely valid CSS at times, and this becomes maddening when a fix for one leads to breakage in another. View the **Resources** page for information on some of the fixes available. Full browser compliance is still sometimes a pipe dream, and we do not expect you to come up with pixel-perfect code across every platform. But do test in as many as you can. If your design doesn't work in at least IE5+/Win and Mozilla (run by over 90% of the population), chances are we won't accept it.

We ask that you submit original artwork. Please respect copyright laws. Please keep objectionable material to a minimum; tasteful nudity is acceptable, outright pornography will be rejected.

This is a learning exercise as well as a demonstration. You retain full copyright on your graphics (with limited exceptions, see **submission guidelines**), but we ask you release your CSS under a Creative Commons license identical to the **one on this site** so that others may learn from your work.

Bandwidth graciously donated by **DreamFire Studios**.

SELECT A DESIGN

contemporary nouveau by David Hellmann
El Collar de Tomas by Hania Stuck
OrderedZen by Steve Smith
Bonsai by Martin Kaczmarz
Type Thing by Michal Hlavinka
Pseudo Sahara by John Ramich
Dragon by Matthew Buchanan
Vivacity by Loco

ARCHIVES

next designs »
View All Designs

RESOURCES

View This Design's CSS
CSS Resources
FAQ
Submit a Design
Translations

Ray Henry，设计师

www.csszengarden.com/029

3.1 Backyard（庭院）

理解绝对定位和浮动

良好的开端是成功的一半，但有时作品的开端却并不是那么的容易。当然，灵感可能在任何时间、任何地点迸发。坐在公园的咖啡馆中一边品尝着美味的lattés，一边和朋友们谈天说地、消磨时光——不经意之间，很多足以改变世界的想法就这样悄然到来。Ray Henry设计Backyard的灵感来自两个花园——禅意花园和他自己的花园。Backyard页首的那幅栩栩如生的相片正是来自于Henry自家花园中的花朵。

Henry在用CSS完成作品时遇到的最基础的问题就是，如何实现三栏布局。Henry也想知道他所熟悉的表格布局效果能否同样用CSS实现。能将链接栏放在最右边吗？如果能的话，应该怎么做呢？需要什么特殊的技巧么？最好的方法就是一个接一个地慢慢尝试，Henry也正是在这样的尝试中完成了该作品。

3.1.1 布局基础

在用CSS实现分栏布局之前，需要了解一些预备知识。CSS并没有提供诸如“left-column: 200px”之类的直观分栏属性，因此我们必须要用一些技巧来突破这些限制。

实现分栏布局主要有两种方法，每种都各有利弊。第一种方法是使用四种CSS定位选项（absolute、static、relative和fixed）中的绝对定位（absolute positioning），它可以将文档中的某个元素从其原本位置上移除，并重新定位在期望的任何地点之上。第二种则是使用CSS中的浮动（float）概念。浮动并不是一种用作定位的选项，而是为元素分配一定的空间，然后让文档中的其他部分将自动环绕在其周围。

绝对定位或浮动都能够用来实现分栏效果。二者可以独立使用，也可以结合在一起，相辅相成。

1. 绝对定位

与浮动相比，绝对定位显得更易于理解。若在文档中依次声明了一系列元素（简称为常规文档流，normal document flow），就像下面这段HTML这样，那么这些元素将会按照声明次序显示在页面中（图3-1-1）——没有什么特别的。

```
<h3>The Road to Enlightenment</h3>
<p>Littering a dark and dreary road lay the past relics
   of browser-
   ➥specific tags, incompatible DOMs, and broken CSS
   support</p>
```

但若是用绝对定位将某一元素（这里是h3元素）移动至指定位置，那么该元素原来占据的空间将消失（图3-1-2）。绝对定位的元素也不会对其旁边的元素带来任何影响，就像文档中根本不存在这个元素一样（虽然实际上在页面中仍能看到）。实现绝对定位的代码如下：

```
h3 {position: absolute; top: 100px; }
```

绝对定位的优势就在于，我们可以丝毫不差地精确控制任何元素的位置——这里面没有什么需要猜测或者运气的成分。由于应用了绝对定位的元素被不留痕迹地从常规文档流中完全移除，所以它也不会为其他元素带来任何

Note

CSS并不能直接实现分栏布局，一般来讲这种说法是正确的。但CSS 2引入的CSS Table概念实际上却能够实现这个效果。很不幸的是，Windows平台上的IE并不支持这个标准，所以这个概念也无法用于实际开发中。噢，真遗憾！不过若你对CSS Table感兴趣的话，可以参考www.w3.org/TR/CSS21/tables.html。

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible *DOM*s, and broken *CSS* support.

图3-1-1　常规文档流中的元素

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible *DOM*s, and broken *CSS* support.

The Road to Enlightenment

图3-1-2　绝对定位h3之后，其原本的位置被下面的段落占据

图3-1-3　正常的文字

图3-1-4　插入图片之后，上下两段文字间出现了间隙

图3-1-5　环绕在图片周围的文字填充了图片右边的空白

的影响。当然，这也可能会导致一些其他问题，将在本节稍后部分介绍。

2. 浮动

与绝对定位相比，浮动更加具有技巧性。这里我们给出了几幅示意图：图3-1-3是一篇较长的文章。若是在文章中添加了一幅图片，那么该图片将占据一定的空间，并将其之后的文字推向页面下方（图3-1-4）。为了避免插入图片造成的两部分文字的不连续，我们可能会希望下面的文字填充到图片右边的空白中（图3-1-5），充分利用页面空间。

这正是CSS浮动的本意和要点所在：若将插入到文章中的图片设置为浮动，那么图片下方的文字将自动环绕在其周围，页面上也就自然不会出现那块丑陋的空白。

浮动的语法看似简单，但却并不那么容易完全掌握。对于上面的这个场景，CSS代码将类似如下所示：

```
#photo {float: left; margin: 10px 10px 10px 0;}
```

这段代码非常简单易懂，其效果也不难预测。

采用浮动实现布局的优势在于，浮动元素仍保留在文档流中，其周围环绕的元素也都清楚它的位置。但由于不同的浏览器对浮动的支持良莠不齐，所以浮动布局也并不是完美之选。因此，只有有了相当的实践经验之后，我们才能够在设计时游刃有余，合理地选择并使用上述两种布局方法。

3.1.2　布局方法

在本节开始部分Backyard的截图中可以看到，整个作品分为三栏，并带有页脚。出于对搜索引擎优化以及禁用浏览器CSS选项的用户的考虑，源代码中主要内容区域 1 声明在左右两栏之前—正文部分在源代码中的位置越靠前，就越会被首先下载并显示于浏览器中，那些次要的导航条和链接也不会占据下载刚开始时的那段宝贵时间。

```
<div id="wrapper">
  <div id="contentArea">
    <p>(headers and text here)</p>
```

[1] 即中间一栏。——译者注

```
    </div>
    <div id="leftPanel">
        <p>(links and various text items here)</p>
    </div>
    <div id="rightPanel">
        <p>(links and various text items here)</p>
    </div>
    <div id="footer">
        <p>(Copyright statement, links, etc.)</p>
    </div>
</div>
```

几乎不加任何思索，我们就能想出一个快速但并不那么优雅的布局方法——使用绝对定位：首先分别为三栏指定固定的宽度，然后从页面的左上角开始为这三栏依次指定固定的位置即可。若是不考虑页脚的话，现在就算大功告成了（图3-1-6）。

```
#contentArea {width: 450px; position: absolute; top: 0;
 left: 150px;}
#leftPanel {width: 150px; position: absolute; top: 0;
 left: 0;}
#rightPanel {width: 150px; position: absolute; top: 0;
 left: 600px;}
#footer {display: none;}
```

这三栏都是靠左的，但若是要让它们居中显示，就像图3-1-7那样，又该怎么做呢？同样很简单：绝对定位模型中有个极为有用的特性，那就是若某个绝对定位元素的容器也被定位过，那么为该元素指定的top和left值将不会基于文档的根元素html（也就是浏览器窗口的左上角）计算，而是会基于其容器的左上角计算这个偏移量。

有些糊涂了？没关系，你可以这样理解：被定位过的容器将扮演其中所有元素绝对定位起始点的角色。所以，若是我们将#wrapper定位，并让其居中显示的话，其包含的各栏将仍显示于其中。下面的CSS代码就演示了其具体的实现方法（由于Windows平台上IE5浏览器的缺陷导致其不能将外边距为auto的元素居中显示，所以我们需要使用两个text-align属性辅助IE5下的定位）：

```
body {
  text-align: center;
}
#wrapper {
```

[1] IE7已经修复了这个问题。——译者注

Tip

不知为何，Windows平台下的IE浏览器会神秘地将某些浮动元素的水平外边距值加倍[1]。若元素的外边距值足够大（例如超过10px）时，效果将变得极为明显。这是个令人头痛不已的问题，甚至会给你带来强烈的挫折感。一个权宜之计就是为该浮动元素应用display: inline属性。若你想了解更多有关这个缺陷的分析，请浏览Position is Everything网站（www.positioniseverything.net/explorer/doubled-margin.html）。

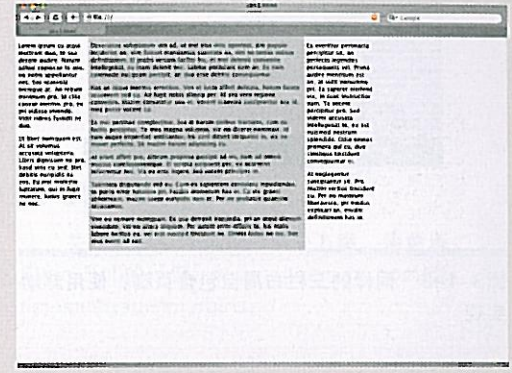


图3-1-6　简单的三栏布局

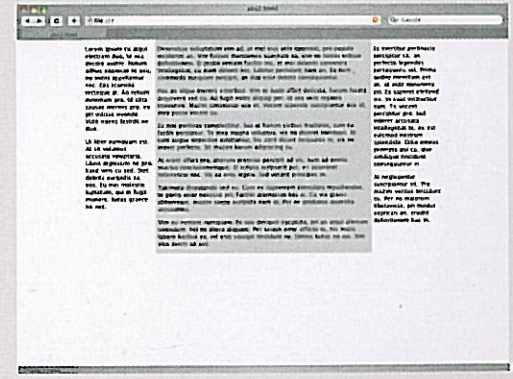


图3-1-7　同样的三栏布局，但居中显示

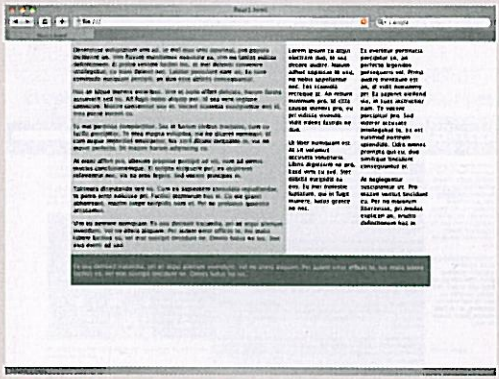


图3-1-8　同样的三栏布局但包含页脚，使用浮动实现

```
  width: 750px;
  margin: 0 auto;
  text-align: left;
  position: relative;
}
#contentArea {
  width: 450px;
  position: absolute;
  top: 0;
  left: 150px;
}
#leftPanel {
  width: 150px;
  position: absolute;
  top: 0;
  left: 0;
}
#rightPanel {
 width: 150px;
 position: absolute;
 top: 0;
 left: 600px;
}
```

但页脚又该如何处理呢？由于绝对定位的三栏并不会影响到页面中其他的元素，所以页脚将忽略它们的存在，就像这三栏根本不在页面中一样。因此，在这里页脚将显示在页面第一个元素的位置上。

若仍要采用绝对定位的话，那么必须预先知道这三栏中每一栏的高度，然后再根据其中最高的一栏定位页脚。若是三栏中任意一栏中的文本长度无法确定的话，除了使用JavaScript，我们也只能放弃绝对定位的念头，转而投入到浮动布局的怀抱中。

图3-1-8所示的页面布局就是使用浮动实现的。页脚自动清除了左中右三栏的浮动设置，并定位于三栏之下。

```
body {
  text-align: center;
}
#wrapper {
  width: 750px;
  margin: 0 auto;
```

```
    text-align: left;
    position: relative;
  }
  #contentArea {
    width: 450px;
    float: left;
  }
  #leftPanel {
    width: 150px;
    float: left;
  }
  #rightPanel {
    width: 150px;
    float: left;
  }
  #footer {
    clear: both;
  }
```

图3-1-8中可以发现一个明显的问题，那就是左栏和中栏交换了位置。因为浮动布局依然遵循常规文档流，所以与绝对定位相比，浮动定位时HTML源文件中元素声明的位置就显得格外重要。当然，解决这个问题的最简单方法就是在源文件中交换左栏和中栏的声明次序。对于这个问题还有些极为晦涩的解决方法——使用负外边距值等较另类的方式。所以一般情况下，人们十有八九都会选择交换源文件中左中两栏的声明次序。

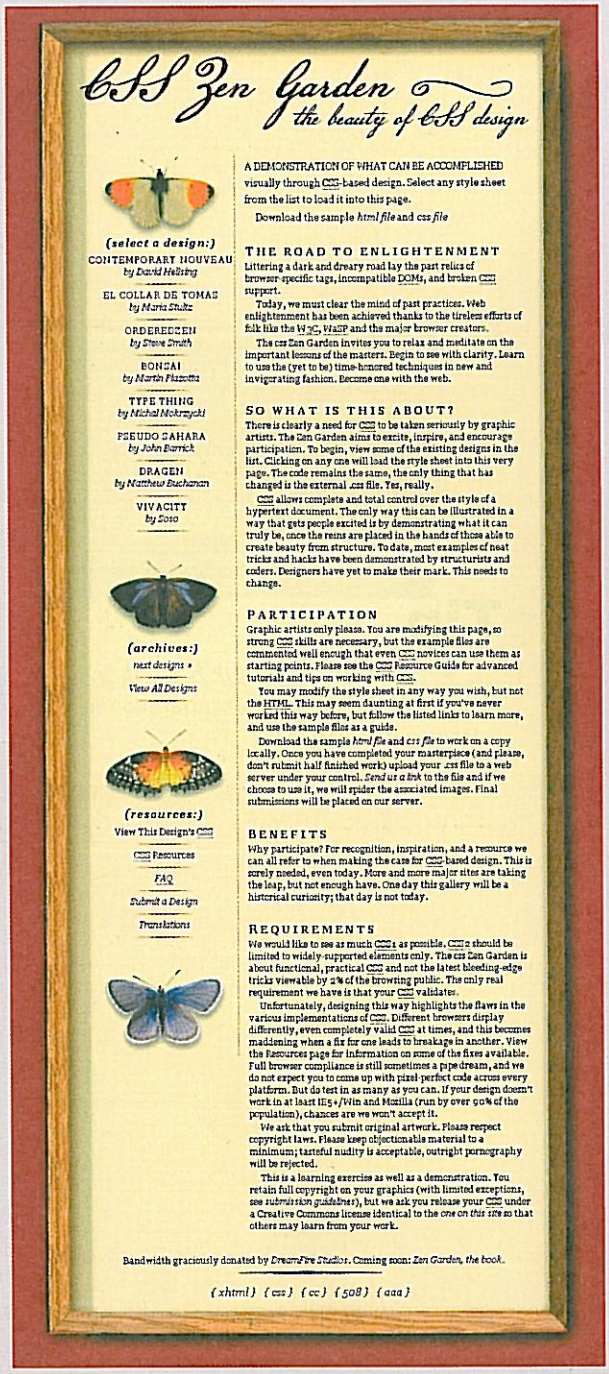
3.1.3 绝对定位的补充

在尝试过许多不同CSS布局方法之后，Ray Henry还是决定用绝对定位完成他的三栏作品Backyard。我们有理由相信，在许多情况下，包括Backyard在内，绝对定位是最简单且最有效的一种布局方法。

但绝对定位也不是万能的，使用时也难免会遇到一些问题。对于那些需要考虑页脚，或是页面中其他元素需要了解其周围元素的页面布局来说，浮动则将成为不二之选。无论绝对定位还是浮动都谈不上完美，但它们却组成了CSS中的最具可操作性的布局工具。合理恰当地选择某种方法，或者组合使用这两种方法，将能够随心所欲地控制页面的布局——唯一可能让作品受到限制的就是创造力。

Tip

关于使用负外边距的解决方案，请参考Ryan Brill的示例：www.alistapart.com/articles/negativemargins。



CSS Zen Garden
the beauty of CSS design

A DEMONSTRATION OF WHAT CAN BE ACCOMPLISHED visually through CSS-based design. Select any style sheet from the list to load it into this page.

Download the sample *html file* and *css file*

(select a design:)
CONTEMPORARY NOUVEAU
by David Hellzing
EL COLLAR DE TOMAS
by Maria Shultz
ORDEREDZEN
by Steve Smith
BONSAI
by Martin Fizzotta
TYPE THING
by Michal Mokrzycki
PSEUDO SAHARA
by John Barrick
DRAGEN
by Matthew Buchanan
VIVACITY
by Zoto

(archives:)
next designs »
View All Designs

(resources:)
View This Design's CSS
CSS Resources
FAQ
Submit a Design
Translations

THE ROAD TO ENLIGHTENMENT
Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support.

Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators.

The *csz* Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

SO WHAT IS THIS ABOUT?
There is clearly a need for CSS to be taken seriously by graphic artists. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The code remains the same, the only thing that has changed is the external .*css* file. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. To date, most examples of neat tricks and hacks have been demonstrated by structurists and coders. Designers have yet to make their mark. This needs to change.

PARTICIPATION
Graphic artists only please. You are modifying this page, so strong CSS skills are necessary, but the example files are commented well enough that even CSS novices can use them as starting points. Please see the CSS Resource Guide for advanced tutorials and tips on working with CSS.

You may modify the style sheet in any way you wish, but not the HTML. This may seem daunting at first if you've never worked this way before, but follow the listed links to learn more, and use the sample files as a guide.

Download the sample *html file* and *css file* to work on a copy locally. Once you have completed your masterpiece (and please, don't submit half finished work) upload your .*css* file to a web server under your control. *Send us a link* to the file and if we choose to use it, we will spider the associated images. Final submissions will be placed on our server.

BENEFITS
Why participate? For recognition, inspiration, and a resource we can all refer to when making the case for CSS-based design. This is sorely needed, even today. More and more major sites are taking the leap, but not enough have. One day this gallery will be a historical curiosity; that day is not today.

REQUIREMENTS
We would like to see as much CSS1 as possible. CSS2 should be limited to widely-supported elements only. The *csz* Zen Garden is about functional, practical CSS and not the latest bleeding-edge tricks viewable by 2% of the browsing public. The only real requirement we have is that your CSS validates.

Unfortunately, designing this way highlights the flaws in the various implementations of CSS. Different browsers display differently, even completely valid CSS at times, and this becomes maddening when a fix for one leads to breakage in another. View the Resources page for information on some of the fixes available. Full browser compliance is still sometimes a pipe dream, and we do not expect you to come up with pixel-perfect code across every platform. But do test in as many as you can. If your design doesn't work in at least IE5+/Win and Mozilla (run by over 90% of the population), chances are we won't accept it.

We ask that you submit original artwork. Please respect copyright laws. Please keep objectionable material to a minimum; tasteful nudity is acceptable, outright pornography will be rejected.

This is a learning exercise as well as a demonstration. You retain full copyright on your graphics (with limited exceptions, see *submission guidelines*), but we ask you release your CSS under a Creative Commons license identical to the *one on this site* so that others may learn from your work.

Bandwidth graciously donated by *DreamFire Studios*. Coming soon: *Zen Garden, the book*.

{xhtml} {css} {cc} {508} {aaa}

Jon Hicks，设计师

www.csszengarden.com/030

3.2 Entomology（昆虫学）

当代Web设计中的居中布局

兴趣是创意的源泉。Entomology的想法正是来自于一个音乐视频——Yeah Yeah Yeahs' single "Pin."。Jon Hicks虽然愿意基于禅意花园进行设计，但却不想让作品太过千篇一律。视频"Pin"中的很多用大头针钉起来的蝴蝶的照片却恰好和Hicks的想法出奇的一致。

在草草勾画出设计方案之后，Hicks直接开始了CSS实现，而并没有首先在Adobe Photoshop创建原型。由于这只不过是个个人的小项目，加之Hicks平日的工作也很繁忙，所以他并没有按照标准的设计流程进行。Hicks甚至还声称要戴上太阳帽，拿着用来捕捉蝴蝶的网去找到第一手的感觉！这样做的结果就是，该作品与Hicks的惯用作品风格截然不同。

3.2.1 固定布局和流式（liquid）布局

按照惯例，Web页面的布局可以分为两类：固定布局和流式布局。固定布局是指页面有着固定的宽度，居左、居右（这种情况比较少见）对齐，或者居中。但页面内容的宽度却不随着浏览器窗口的宽度变化而变化。Entomology就是一个固定布局的例子（图3-2-1）。

流式布局又叫做不固定（fluid）布局或动态（dynamic）布局。这种布局的页面宽度用百分比指定，而不是某个固定的宽度。所以当浏览器窗口大小变化时，页面宽度也随之变化，始终保持填满浏览器（图3-2-2）。

我们同样也可以将固定布局和流式布局结合起来使用。例如对于某个三栏页面来说，左右两栏可能有着固定的宽度，但中间一栏的宽度却用百分比定义（或者不定义其宽度）。这样，作品将始终填满浏览器窗口，在改变浏览器窗口大小，或是在不同分辨率下浏览该页面时，中间一栏的宽度将相应地变大或变小。

就像许多其他流行的东西一样，人们对固定布局和流式布局的偏好和倾向也在不停地变化。在早期的Web设计中，固定布局曾风靡一时。但一段时间之后，人们开始认识到流式布局的好处。几年中流式布局又变得如日中天。

在过去的两年中，固定布局似乎再一次占据了上风——特别是在那些推崇标准的设计师眼中。此时此刻，几乎所有的业内领先者都在他们的网站或作品中使用了固定布局。这是个很有意思的倾向，之所以人们再次开始钟情于固定布局，是因为它们主要考虑了如下几个明显问题。

首先，也是最重要的一点就是，由于作品区域大小固定，所以设计师在控制固定布局页面时将更加游刃有余。即使对于图文混排的情况，一行中文字的长度同样易于控制。固定布局的再次流行在很大程度上正是因为这些原因。

当然，流式布局也有它的优势所在。填满浏览器窗口意味着最大限度地利用了空间，且与那些居左对齐的固定宽度作品相比，流式布局设计避免了另一半页面的空白，显得更为平衡。流式布局的劣势在于我们无法精确控制

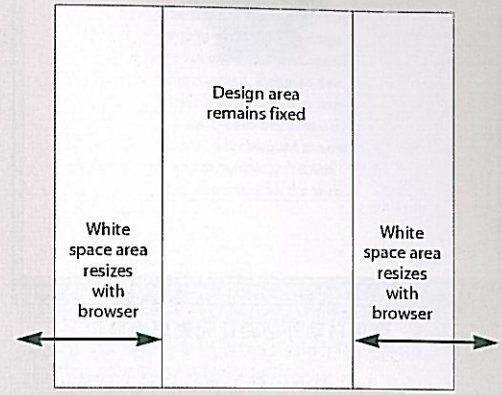


图3-2-1　固定布局页面示意图

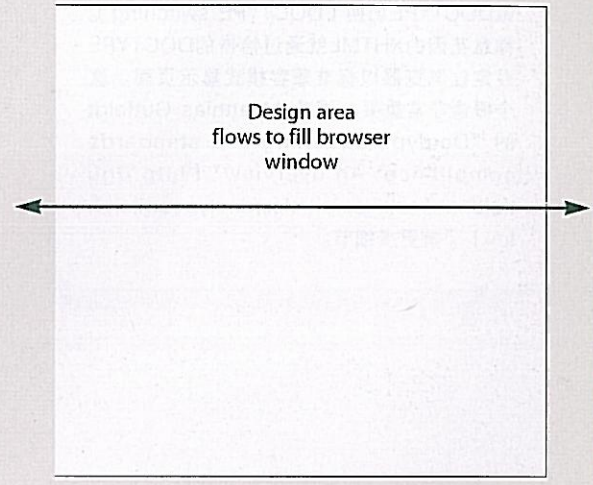


图3-2-2　流式布局页面示意图

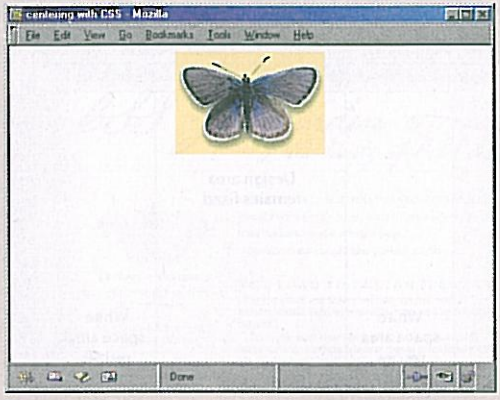


图3-2-3　使用自动外边距让元素居中显示

Note

标准兼容模式是一种浏览器技术，又叫做DOCTYPE切换（DOCTYPE switching）。禅意花园的XHTML就通过恰当的DOCTYPE设定让浏览器以标准兼容模式显示页面。这个概念非常重要，请参考Matthias Gutfeldt的“Doctype switching and standards compliance: An overview”（http://gutfeldt.ch/matthias/articles/doctypeswitch.html）了解更多细节。

行长、页面布局以及元素的位置。即使对于那些资深设计师来说，这些问题也都值得仔细斟酌，且现今也没有什么完美的解决方案。

对于固定布局作品导致的视觉失衡问题，设计师一般通过让作品水平居中显示来解决。这样当改变浏览器窗口大小时，空白部分将始终平均分配在页面的左右两侧，不至于让页面的右侧显得太过单调。

3.2.2　让作品居中显示

用CSS让元素居中显示并不是件很简单的事情——同样的合法CSS居中设置在不同浏览器中的表现行为却各有千秋。在处理Entomology时Hicks也自然遇到了这个问题，但幸运的是，他最终找到了一种临时补救措施。在介绍他的做法之前，让我们先来看一下CSS中常见的几种让元素水平居中显示的方法。

1. 使用自动外边距实现居中

CSS中首选的让元素水平居中的方法就是使用margin属性——将元素的margin-left和margin-right属性设置为auto即可。在实际使用中，我们可以为这些需要居中的元素创建一个起容器作用的div。需要特别注意的一点就是，必须为该容器指定宽度：

```
[ div#container {
    margin-left: auto;
    margin-right: auto;
    width: 168px;
{
```

图3-2-3演示了这种方法的实际效果，图中的蝴蝶图片来自Hicks的作品。将该图片放置于一个div中，并使用自动外边距的方法让其居中显示。

在大多数主流浏览器中，这种方法都非常有效，即使是Windows平台上的IE6，在其标准兼容模式（compliance mode）下也能够正常显示。但不幸的是，在更低版本的IE中，这种设置却并不能实现居中效果。所以若想在实际项目中使用这种方法，那么就要确保用户的IE浏览器版本必须不低于6.0。

表3-2-1列出了常见浏览器对自动外边距居中的支持情况。

表3-2-1 浏览器的自动外边距支持

浏览器	版N本	支持
Internet Explorer [1]	7.0，标准兼容模式	是
Internet Explorer	7.0，向前兼容模式（quirks mode）[2]	否
Internet Explorer	6.0，标准兼容模式	是
Internet Explorer	6.0，向前兼容模式	否
Internet Explorer	5.5，Windows版本	否
Internet Explorer	5.0，Windows版本	否
Internet Explorer	5.2，苹果版本	是
Mozilla	所有版本	是
Mozilla Firefox	所有当前版本	是
Netscape	4.x	否
Netscape	6.x及以上	是
Opera	6.0和7.0，Windows和苹果版本	是
Safari	1.2	是

尽管在支持上不尽如人意，但大多数设计师都建议尽可能地使用这种方法。该方法也被认为是在各种用CSS实现元素水平居中方法中最正确、最合理的一种。

2. 使用text-align实现居中

另一种实现元素居中的方法是使用text-align属性，将该属性值设置为center并应用到body元素上即可。这种做法是彻头彻尾的hack，但它却能兼容大多数浏览器，所以在某些情况下也自然必不可少。

之所以说它是hack，是因为这种方法并没有将文本属性应用到文本上，而是应用到了作为容器的元素上。这也就给我们带来了额外的工作。

在创建好布局必须的div之后，我们要按照如下代码为body应用text-align属性：

```
body {
    text-align: center;
}
```

之后会出现什么问题么？body的所有子孙元素都会被居中显示（图3-2-4）。

[1] 原书并没有包含IE 7.0的相关条目。——译者注

[2] 或译作“怪异模式”。——译者注

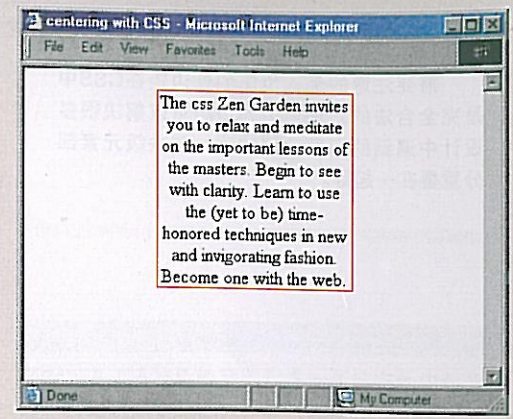


图3-2-4 在IE中使用text-align实现居中。可以看到div中的文本也继承了居中的样式

Note

需要注意的是，负值的外边距在CSS中是完全合法的。使用负外边距可以解决很多设计中遇到的问题，例如让一些块级元素部分重叠在一起等。

因此，我们就需要用再写一条规则，让其中的文本回到默认的居左对齐：

```
p {
   text-align: left;
}
```

可以想象这条附加的规则将带来一些不便。另外，真正完全遵循标准的浏览器并不会改变容器的位置，而只会让其中的文字居中显示。

3. 组合使用自动外边距和文本对齐

因为文本对齐居中方式有着良好的向下兼容性，且自动外边距方式也被大多数当代浏览器支持，所以很多设计师将二者组合起来使用，以期让居中效果得到最大限度的跨浏览器支持：

```
body {
text-align: center;
}
#container {
   margin-left: auto;
   margin-right: auto;
   border: 1px solid red;
   width: 168px;
   text-align: left
}
```

可是这始终是个hack，无论如何也算不上完美。我们还是需要为居中容器中的文本编写附加的规则，但至少在各个浏览器中看起来都不错。

4. 负外边距解决方案

考虑到Entomology的特殊挑战，Hicks决定用另外一种方案来实现页面居中——使用负外边距。

负外边距解决方案远不是仅仅为元素添加负外边距这么简单。这种方法需要同时使用绝对定位和负外边距两种技巧。

下面将介绍该方案的具体实现方法。首先创建一个包含居中元素的容器，然后将其绝对定位于相对页面左边缘50%的位置。这样，该容器的左外边距将从页面50%宽度的位置开始算起（图3-2-5）。

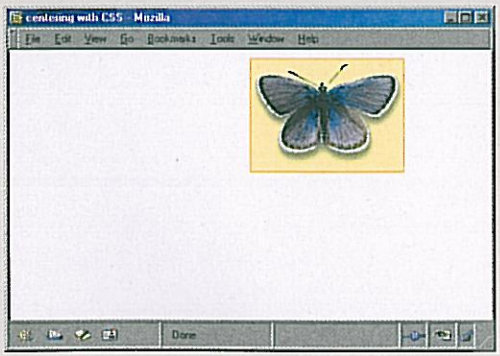


图3-2-5　将容器绝对定位于相对页面左边缘50%的位置

然后，将容器的左外边距值设置为负的容器宽度的一半。这样即可将该容器固定在页面水平方向的中点（图3-2-6）。

下面就是Entomology所用到的CSS代码：

```
#container {
   background: #ffc url(mid.jpg) repeat-y center;
   position: absolute;
   left: 50%;
   width: 760px;
   margin-left: -380px;
}
```

看，没有任何hack！虽然这并不是首选的解决方案，但也是个不错的方法，且适用性极广——甚至在Netscape Navigator 4.x中都没有任何问题，很令人吃惊，不是么？所以若想得到最广泛的浏览器支持，那么这种方法将会是最好的选择。

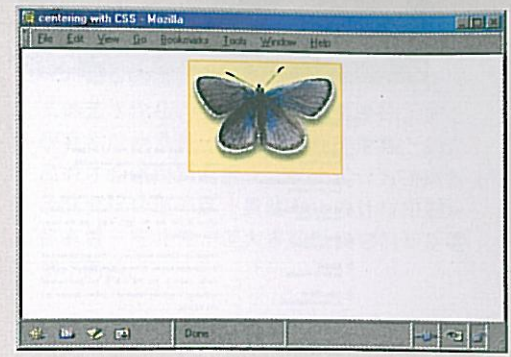


图3-2-6　添加负的容器宽度一半的左外边距值之后，容器将居中显示

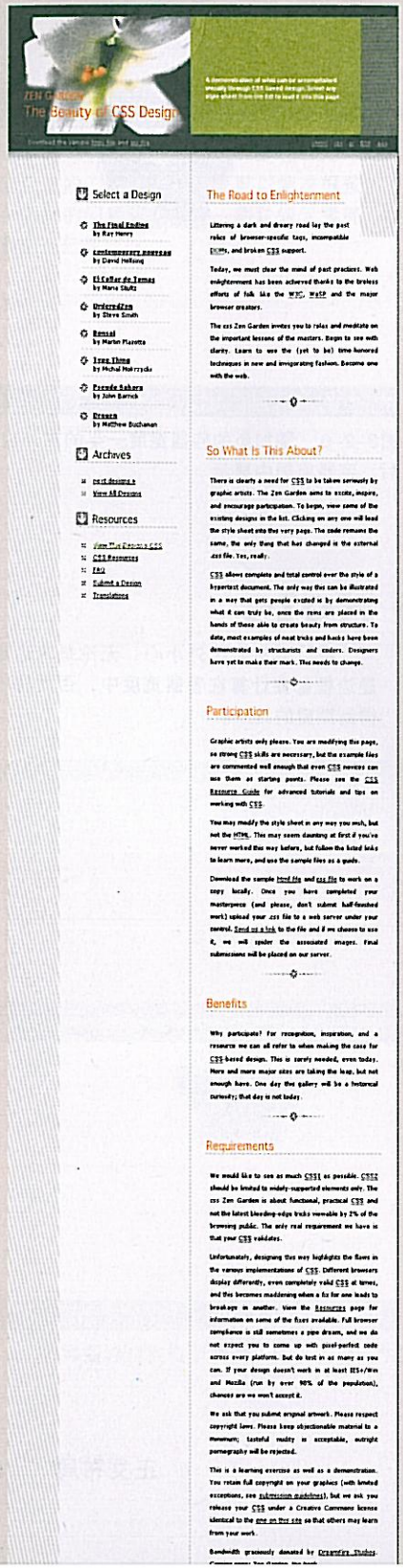
Note

在计算时要格外小心，无论是内边距还是边框都要计算在容器宽度中，否则将不会得到期望的结果。

3.2.3　跟上趋势的发展

虽然谁也无法确定今后趋势的发展方向，但这些固定布局的优秀作品将始终在Web设计中占有一席之地。固定布局是一类非常经典的Web页面布局方式，每个设计师都应该掌握它的实现方法。

可以根据目标用户的浏览器分布以及个人喜好来选择并使用本节列出的这几种居中方案。对于Entomology来说，用绝对定位和负外边距来实现内容居中确实是个“时髦”的选择，让作品能够在绝大多数浏览器中正常显示——甚至在那些相当“古老”的浏览器中也没有任何问题。



Jens Kristensen，设计师

www.csszengarden.com/036

3.3 White Lily（白百合）

创建有效布局的法则与步骤

尽管禅意花园没有对提交作品的设计方面有什么要求，Jens Kristensen还是发现了其中暗含的四个关键字，那就是：视觉效果（visual）、CSS、禅意（Zen）和花园（Garden）。这四个关键字让Kristensen牢牢把握住了设计的精髓，那就是作品应该给人以放松、平静的类似植物一般的平和感觉。

体会到暗含的精髓之后，Kristensen开始研究如何用恰当的设计布局来表现这个含义。从实现方式看，White Lily的布局极为实用，全世界的众多网站都能够从该作品中找到值得借鉴的地方。

3.3.1 Web页面的布局法则

我们经常简单地将“布局”这个单词理解为页面中包含多少栏内容。正在学习使用CSS的人们经常会这样问：“如何创建三栏布局？”或“如何创建带有页首和页脚的布局？”。这些问题自然无可厚非，并且也相当重要，但在开始考虑如何实现这些布局之前，我们最好还是先了解一下为何要选用这种布局，即布局背后的法则。

用来实现布局的任何一种技术，无论多么的巧妙甚至花哨，其根本都是为了体现某种布局法则。这种法则在Web诞生之前就已经存在了。虽然Web的出现也在不经意间改变了某些条款，但其仍旧是创建漂亮且实用的布局的最基本原理，重要性丝毫不减当年。

在任何一种以图画作为表意方式的艺术作品中，布局都将起到同样的作用——有效地组织内容。而实际作品中千变万化的布局效果，则是因为作品的传播媒介、内容以及欣赏者各有不同。对于Web设计来说，布局还有一个功能，那就是辅助实现页面的功能。一个成功的布局必须能够有效地将链接、菜单、表单等元素集成到作品中。最后，布局也是保证网站一致性的重要基石，风格统一的网站将更易于使用者理解，带来更好的可用性。

3.3.2 三思而后行

在完成White Lily设计的过程中，给Kristensen最大帮助的就是他在一开始就组织好了页面的内容。

事先组织内容主要出于如下几个目的。最明显的就是，能够对接下来要做的事情有清楚的认识，并自然分清不同内容之间的主次关系。这样也能对页面布局及其实现方式进行更深层次的思考，进而更好地把握住作品的形式以及功能。

Note

如果熟悉其他形式的以图画作为表意方式的艺术作品，例如印刷或包装设计，那么很容易就能发现一些只有在处理屏幕上的作品时才会遇到到的挑战。Web设计方面的新手只要记住这一点：虽然Web设计与印刷作品有着一些共性，但大多数时候它们却是截然不同的。

图3-3-1　White Lily的另一种样式

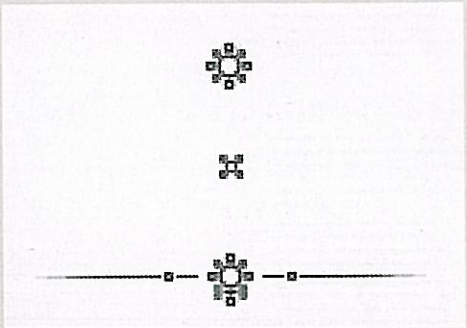


图3-3-2　White Lily中使用的三种象征着花的符号，增强了作品的“植物”意味

Kristensen首先将禅意花园的内容分为了三大类：

- 概要内容，例如模板、验证链接和禅意花园介绍等。
- 链接和导航部分内容。
- 核心内容

有了这些归类之后，Kristensen开始组合页面中的元素。目前他有两个非常清楚的目标：实现设计理念，并将内容清晰地分为三段。在确定最终样式之前，Kristensen还进行了若干次的修改（图3-3-1）。

最后，Kristensen将样式确定下来，也就是这个已提交的版本，并完成了White Lily的设计。需要特别注意的是，作品布局的选择取决于页面的内容以及视觉上的需求，并不能随意而为或是盲目跟随当前的流行趋势。

3.3.3　White Lily布局的实现方式

由于对设计步骤有着清晰的认识，Kristensen根据作品的主题和内容选择了最恰当的设计方式。

Kristensen发现百合花的照片最能体现出他所期待的植物的感觉。在这幅百合花照片中，他也确定了作品中将使用的颜色。Kristensen同样也在某些细节上尽可能恰到好处地进行润色，加深植物的意味（图3-3-2）。

在开始处理内容时，Kristensen的作品就已经相当光彩夺目。因为事先已经明确了目标，Kristensen很容易够判断出不同内容之间的重要性，然后通过适当的布局体现出来。

通过将概要信息放置于页首位置，Kristensen实现了两个很重要的目标：为作品打上烙印（这里是那幅百合花和“CSS Zen Garden”标题），以及表明该页面的创建动机。同样出现在这部分的还有示例HTML和CSS文件，以及验证链接等内容（图3-3-3）。

图3-3-3　这里实现了一个很重要的需求：将相关内容有效地放在一起

这样的设计也便于接下来处理最主要的两个关键之处：导航部分与核心部分的内容。Kristensen将导航链接置于页面的左栏中，这种布局非常不错，因为导航链接本身就应该在这里。右栏则用来显示核心内容。核心内容中每一节的样式都很轻柔，保持了作品的简洁性。另外，两栏布局也符合了设计的目标及需求，整个作品虽然简单，却仍然能够给人带来相当的美感。

3.3.4 布局的灵感和法则

一旦明确了设计目标，即可充分利用布局上的一些通用法则来帮助设计。这一节将介绍Web布局中的一些常见的注意事项。共分为两大类——用户界面（user interface，UI）设计和可用性，这些都是良好的Web设计中不可或缺的要素。

1. 用户界面设计

与印刷作品只能够被动地传递信息不同的是，Web页面还提供与用户交互的功能。由于计算机技术领域中交互性的界面无所不在，例如操作系统和软件应用程序等，所以用户界面设计的相关理论已经发展得较为成熟。

下面给出了一些用户界面设计中的基本法则。

- 了解将浏览页面的用户。
- 在页面和站点中给用户足够的导向。
- 使用被人们熟知的象征符号。一个很好的例子就是电子商务网站中的购物车符号。即使是刚学会上网的用户也能容易地了解其含义（图3-3-4）。
- 保证与功能相关的特性在页面中足够显眼。在White Lily中，功能相关的特性就是左栏中的各个链接，这些链接都极为简洁地表达了其功能含义（图3-3-5）。
- 保证设计元素的一致性。若把White Lily的布局应用到某个网站之上，那么该网站中的各页面间的导航链接应该保持一致，LOGO图片和标题的位置也要始终如一。若在站点的不同页面中标题或功能元素的位置不统一，用户将对此感到非常困惑——因此，我们要尽量避免这种情况的发生。

图3-3-4　在用户界面设计中，象征符号非常重要。例如这个来自amazon.com的购物车符号就是全球通用的，且非常易于识别，所以很多电子商务网站中都使用了类似的符号

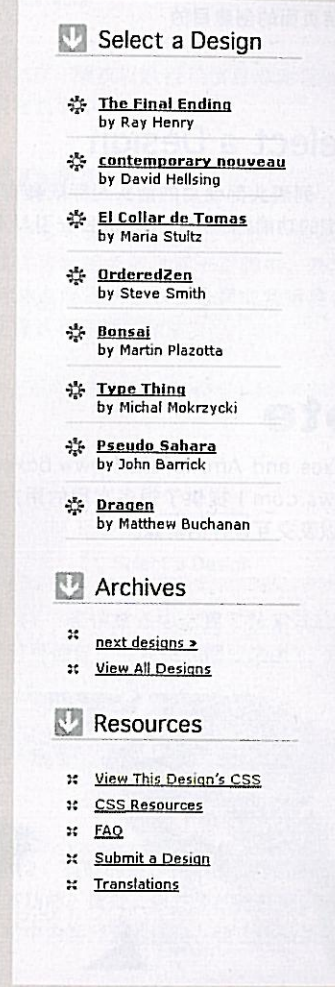


图3-3-5　White Lily中的导航链接

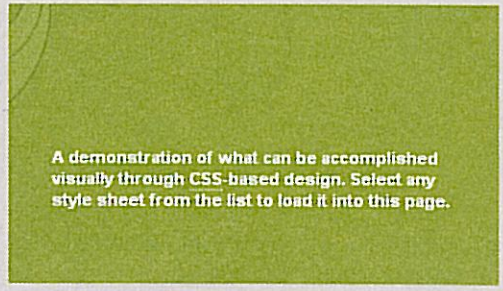


图3-3-6　将主题句布局在页面的独立位置有助于让用户了解页面的创建目的

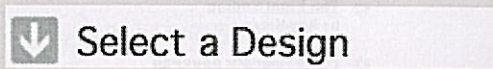


图3-3-7　列表头部使用的箭头图标既具有美感，也有着一定的功能性——将用户的目光引导至下方的内容

Note

Boxes and Arrows网站（www.boxesandarrows.com）提供了很多有用的用户界面设计以及交互设计的资源。

图3-3-8　包含页面中最重要信息的页首部分位于屏幕的最上部

- 了解页面中的关键元素。在一幅作品中，总是会有一些能够吸引用户最多关注的特殊元素。例如White Lily中的百合花就属于这类。意识到某图片对整个作品带来的重要影响之后，设计师即可通过调整其他的页面元素来相应地让作品达到视觉上的平衡。
- 清楚地表达页面的内容。人类的注意力持续时间较短，通常来讲最多9秒钟——若是在这个时间段之后还是不能了解页面的主要内容，那么用户就会彻底放弃。Kristensen通过在布局中将页面介绍分离并置于独立位置中，让用户一眼就可以把握到页面的主旨（图3-3-6）。

由于美学与页面功能有着紧密的联系，所以界面设计师对此也非常感兴趣。注意White Lily左栏链接区域中的箭头。每个大小都相同，给用户带来审美上的愉悦感的同时也给了用户充分的导向提示（图3-3-7）。

2. 可用性

虽然概念上与用户界面设计类似（在历史上二者的发源也相同），但可用性关注的领域却与前者不同。顾名思义，用户界面设计更加关注用户——其最主要的研究目的就是让开发者和工程师能够更好地满足用户的需求。

但可用性研究则更加关注最终的用户体验，即用户是如何与已经开发完成了的程序进行交互的。二者相辅相成，可用性研究影响了UI设计技术以及设计流程的发展，用户界面设计也给用户以测试修改的平台，帮助作品实现更好的整体可用性。

与UI设计一样，可用性研究也源于其他计算机技术领域并有着很长的历史。Web可用性研究是可用性研究中的一个子集，且有着它自己的法则与最佳实践。

下面列出了一些与布局相关的可用性问题。

- 重要信息应放在显眼的位置。当然，Web设计中“显眼”只是个相对概念，但通常来讲这一条也很易于实现：将用户最需要的信息尽可能地放在页面的最上部。在White Lily中，最重要的就是百合花的照片和页面标题，它们都放在了页面的顶部，让用户一眼就能发现（图3-3-8）。
- 永远在title元素中给出页面的简单介绍：

```
<title>css Zen Garden: The Beauty in CSS Design</title>
```

这段文字将显示在浏览器标题栏中，起到充分的导向作用（图3-3-9）。

- 尽可能地保证页面中的导航链接有着一致的表现。同样也要确保页面中包含回到站点首页的链接。作品中最好还要提供版权页面、隐私策略以及联系信息等内容。禅意花园中的所有作品都受到创作公用许可协议的保护，这些在文字上都有体现（图3-3-10）。
- 对于中型或大型站点来说，强烈建议提供搜索功能。当然，与通常的网站不同，禅意花园更加偏重于展示设计，但在真实世界中的网站中，用户非常乐于使用搜索功能。也就是说，搜索功能的位置要让用户用直觉就能够发现。通常来讲，搜索框一般应被放在与其逻辑相关的位置上：导航条附近就是个不错的选择。若为White Lily添加搜索功能，那么最佳的位置就是在导航链接之上（图3-3-11）。
- 用缩进和偏移将栏中内容分开。这一点在实现页面整体效果以及提高用户接受度中至关重要（图3-3-12）。

当然，可用性远不止这些。但在设计布局中合理地使用这些技术将大大充实、完善作品的功能与形式，最终带来更好的用户体验。

3.3.5 遵守设计流程

White Lily演示了遵守设计流程为我们在完成布局需求时带来的帮助。当然，不同的设计师或设计团队中都会有不同的设计流程，这些设计流程很有可能看起来截然不同。

但无论其他人的设计流程是怎样的，White Lily的结果却非常令人满意。通过首先确定作品的主题并分析其将要包含的内容，Jens Kristensen很容易地能够沿着一条清晰的道路走下去，直至达到最终目标：实现一个引人注目的、让用户满意并理解的布局。

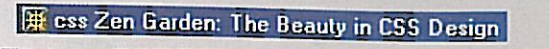


图3-3-9　虽然不是页面布局的一部分，但标题栏中的文字还是能够在很大程度上帮助用户了解站点的主题

This is a learning exercise as well as a demonstration. You retain full copyright on your graphics (with limited exceptions, see submission guidelines), but we ask you release your CSS under a Creative Commons license identical to the one on this site so that others may learn from your work.

图3-3-10　版权以及授权信息非常重要，要有效地体现在页面中

Note

很多常见布局中都将版权、隐私策略、授权许可等相关信息置于页脚中。若此类信息对站点所属的组织或公司非常重要，那么应选择这种通常的做法。

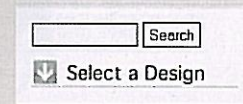


图3-3-11　若该站点中启用了搜索功能，那么该功能很有可能位于导航链接部分之上

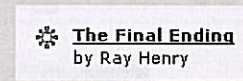


图3-3-12　有趣的是，栏中的文字缩进能够帮助用户理解其功能。此处，最主要的链接列表就使用了缩进，将栏中内容分开的同时，也让作品整体显得更加大气

Minz Meyer，设计师

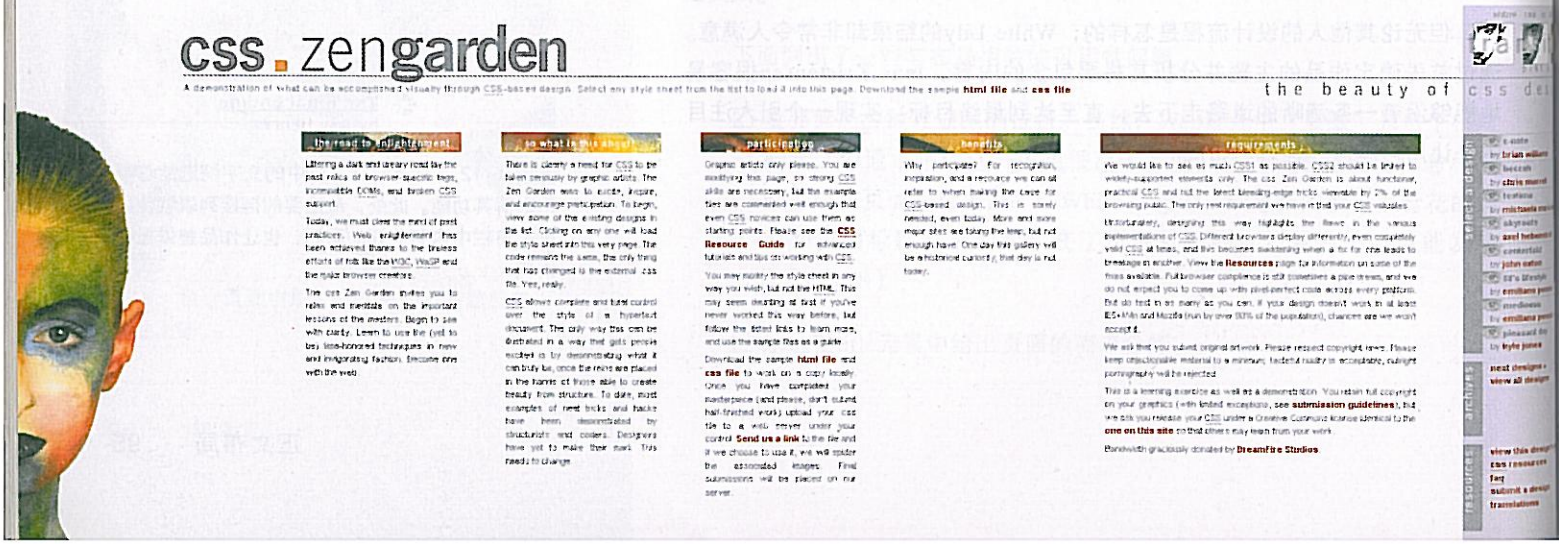
www.csszengarden.com/037

3.4 prêt–à–porter

在垂直的世界中使用水平布局

新设计师们最常问到的一个问题就是，“到哪去找到灵感呢？”很多人建议新手去一些地点进行观察。例如壮美、多彩、富有活力的大自然就常常成为灵感的发源地。

另外一个灵感的发源地就是人的性格和外貌。人的外形总是会在艺术中占有相当的地位。美术专业的学生都会在人体上花费很多时间——学习如何把握人体的轮廓、面孔的线条以及不同人之间截然不同的气质。Minz Meyer的prêt–à–porter就将大自然和人体完美地结合了起来，加上其一反常态的布局，更让整个作品显得独具匠心。



3.4.1 从印刷作品中找到灵感

已经从业一段时间的Web设计师们经常从前辈们口中听到这样的说法，“Web设计与印刷作品不同”。虽然这句话的正确性毋庸置疑，Web设计时不能生搬硬套印刷作品的理论，但我们却仍能够借鉴并从印刷作品中找到Web设计的灵感。prêt-à-porter正是在这一点上走到了我们的前面。

但这里prêt-à-porter的灵感并不是来自某件印刷作品，而是来自一件描述印刷作品的印刷作品（图3-4-1）。

在Meyer的一本介绍某印刷商店商品的小册子中，提到了一种用图像和文字创造视觉平衡的方法。在原设计的左边，垂直地排布了一系列的照片（图片来自该手册，仅起到说明作用），照片中的各个女性面孔都被涂抹上不同颜色的花纹。在右面部分，图像和文字混在一起，文字用德语书写，含义为“失去颜色”。通过色彩艳丽的图像以及清晰的文字，这份介绍手册起到了很好的宣传作用——人们自然可以确信该商店不会弄错印刷作品的颜色。

特别需要注意的是，原宣传手册是横向设计的。于是最让人着迷，同样也是最具有挑战性的一点就出现在了这里——如何在Web中实现这种水平延展的作品。

打破常规

虽然横向设计在印刷作品中广泛应用，效果也非常不错，但Web设计师们通常却都避免在Web设计中使用这种方法。这样做的原因大都基于可用性考虑。水平设计的作品通常都会在浏览器底部显示出滚动条，浏览者则需要水平地滚动页面以得到更多内容。由于大多数用户在获取信息时都倾向于尽可能少地进行操作，所以这样的设计往往会导致可用性问题。特别是对于那些肢体不便的残障人士来说，水平移动滚动条将显得更为困难[1]。这导致的结果就是，除了某些特殊设计，例如用于试验的作品，或者——很遗憾，设计师错误地估计了浏览者屏幕分辨率而设计出的作品，大多数Web设计师都会尽可能地避免横向设计。

[1] 设想用户使用的是带滚轮的三键鼠标：在垂直设计的页面中，只要滚动鼠标（一个操作）即可看到页面的其他内容；而对于横向设计的作品，则需要将指针移动到水平滚动条上，按住鼠标左键，然后进行水平拖拽（三个操作）。——译者注

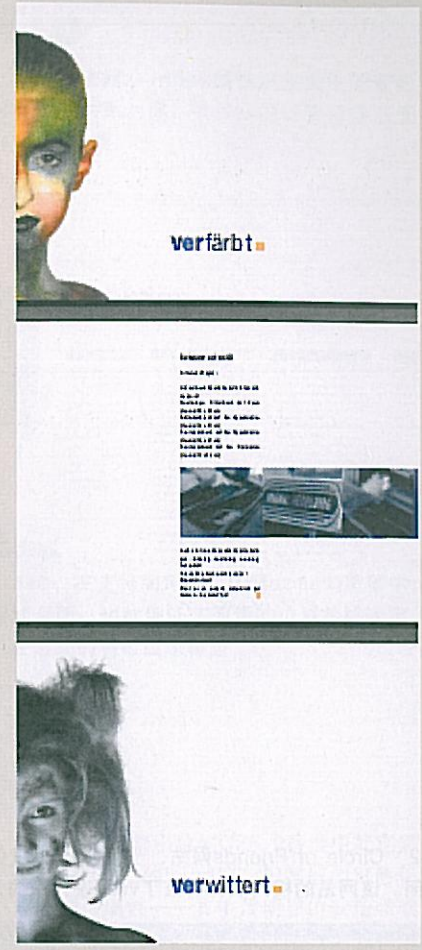


图3-4-1　启发prêt-à-porter设计灵感的若干页宣传手册。

较早打破该常规的网站是1998年发布的“Circle of Friends”，该社区网站致力于帮助那些脊髓受到损伤的群体，并向那些对此兴趣的人介绍相关信息。之所以该组织的倡导者Christopher Reeve（已故）决定让页面横向滚动，是因为很多脊髓受到损伤的人只能以这样的方式感知世界。在当时，这是个革命性的设计，创意的灵感显然来自人的体验（图3-4-2）。

怀着希望横向设计能够对Web产生足够影响的美好愿望，Meyer决定接受这个挑战，尝试创建出该印刷设计的Web版本，用CSS的力量证明这种设计在Web中同样可取。

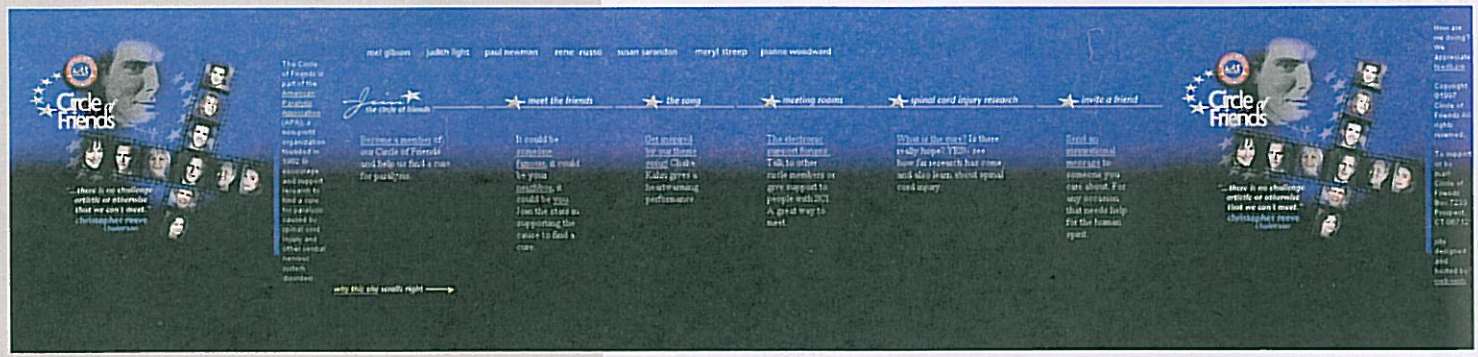
3.4.2 水平带来的挑战

在用CSS实现这个高度受到限制的作品时，最大的“拦路虎”就是不可避免的浏览器兼容性问题。为了让作品在各种操作系统上的各个浏览器中都可以正常使用且足够美观，我们需要使用一些技巧性的处理方法。

1. 考虑固定定位

由于该女性图片非常引人注目，所以Meyer希望让该图片始终固定在作品的同样位置上，并让文字部分进行水平滚动。这样的设计让作品变得对Web极为友好，而这在传统的印刷作品中是不可能实现的。

图3-4-2 Circle of Friends网站，于1998年上线但现在已经关闭。该网站的横向设计打破了Web设计中的常规



虽然CSS给设计师提供了很多种用来实现背景图像的方法，但我们却不能简单地将该女性图片作为背景应用到body元素上。因为这将导致用户滚动页面时，其他部分的文本覆盖到该图片之上，既削弱了图片表达的含义，也降低了文本的可读性。

一种解决该问题的办法就是使用固定（fixed）定位的div，并将图片作为背景应用于其上。无论是哪个元素，只要其应用了固定定位，那么它在页面中的位置都将始终不变。这样，该元素也不会随着页面其他元素的滚动而改变自身的位置。于是我们完全可以用下面的选择器来实现这个功能：

```
div#static-image {
    background-image: url(bg_face.jpg);
    background-repeat: no-repeat;
    background-position: left bottom;
    position: fixed;
    left: 0;
    bottom: 0;
    height: 594px;
    width: 205px;
    z-index: 2;
}
```

若浏览器完全兼容CSS标准，那么position为fixed的这个div将永远停留在同一位置上，且页面中的其他元素定位于其下方，被其遮盖（图3-4-3）。

2. 考虑浏览器缺陷

在解决跨浏览器问题时，我们经常先设计好页面在理论中的样子，然后考虑在出于一些原因不能支持某种效果的浏览器中将页面“降级”处理。为了实现这个目的，我们可以为不同浏览器创建不同的页面，然后用JavaScript分辨出浏览器类型。但有些人却质疑这种解决方案的有效性——很多人在浏览器设置中禁用了JavaScript。

对于prêt-à-porter的情况，考虑到IE浏览器中该人脸图像将随页面滚动到可见区域之外，我们需要为其创建一种替代的设计，让作品看起来同样美观且能够平滑呈现。在这种情况下，Meyer用一个看起来仍可接受的替代设计来解决IE中遇到的问题。

Tip

可以使用z-index属性设置固定元素的前后位置关系。高z-index的元素将覆盖低z-index元素。

图3-4-3　在支持固定定位的Mozilla浏览器中，即使滚动页面，也能保证该图像的位置始终固定，而文本在滚动时将被图像覆盖

Note

尽管1998年出台的CSS 2标准中就定义了position属性的fixed值，但某些浏览器仍不能支持该属性——其中最明显的就是微软公司的IE。所以，虽然在更加兼容标准的浏览器中这条规则能够正常工作，但在IE中该图像却会滚动到页面可见区域之外，也就破坏了作品的效果。

IE不仅不支持固定定位，还不支持子选择器（子选择器用来选择给定元素的某个特定的直接子元素）。子选择器的语法是一个右尖括号（>）。这样，下面的这个选择器：

```
p>strong
```

表示只有在段落元素（p）中直接出现的strong元素才能够应用接下来要声明的CSS样式。

由于IE不支持固定定位或子选择器，我们可以借此将IE和其他浏览器区分开来，让其他浏览器尝试应用理想情况下的CSS规则，而让IE使用另一种规则：

```
body#css-zen-garden>div#extraDiv2 {
    background-image: url(bg_face.jpg);
    background-repeat: no-repeat;
    background-position: left bottom;
    position: fixed;
    left: 0;
    bottom: 0;
    height: 594px;
    width: 205px;
    z-index: 2;
}
```

下面这段CSS只适用于无法理解上面这条规则的浏览器（主要是为IE设计）。因为第一段代码中的子选择器声明更加明确专一，所以该规则的优先级也更高，当然前提是浏览器能够正确理解该规则。

```
div#extraDiv2 {
    background-image: url(bg_face_ie.jpg);
    background-repeat: no-repeat;
    background-position: left bottom;
    position: absolute;
    left: 0;
    bottom: 0;
    height: 600px;
    width: 265px;
}
```

即使在IE 6.0中本应固定的图像滚动到了可见区域之外，作品仍足够吸引人，效果完全可以接受（图3-4-4）。

3. 考虑Opera

prêt-à-porter在设计中遇到的另外一个问题就是作品在Opera Software公司的Opera浏览器中的兼容性。即使浏览器开发者非常重视实现标准（Opera就是如此），但由于很多不同的原因，其对标准的实现并不总是完美的。

在低于7.5版本的Opera浏览器不会为prêt-à-porter显示水平滚动条。显然，在此类浏览器中该页面根本无法使用（图3-4-5）。

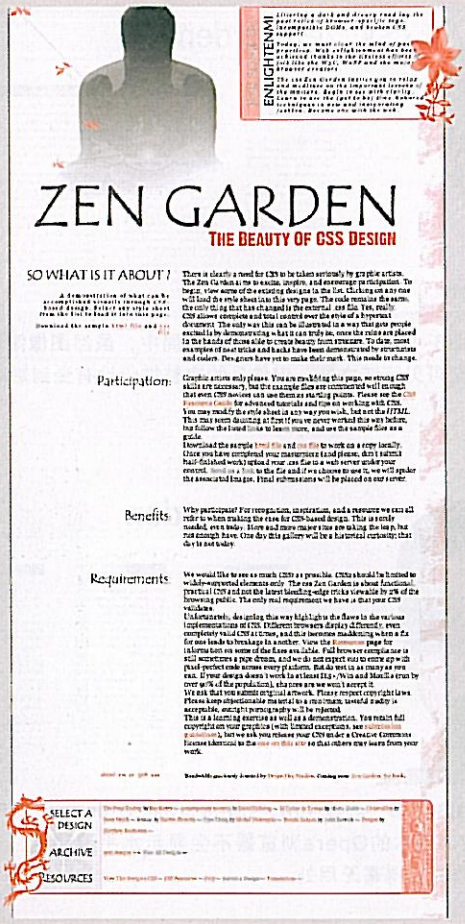
幸运的是，Opera在新版本中修复了这个问题，作品不但能够正常显示，而且显示得相当完美。固定定位的图像位置始终保持不变，与其他支持fixed属性的浏览器一样。

3.4.3 学到的东西

从找到灵感到将该灵感实现为完整的作品是一个过程，对于每个人来说，这个过程都如同最终的结果一样独一无二。这个由横向印刷作品带来的灵感也同时给Meyer带来了希望和挑战。通过使用一些巧妙的方法，Meyer最终完成了这个极其引人注目的漂亮作品。作品在教会我们如何打破常规地思考的同时，也演示了如何有效地将想法付诸实践。

图3-4-4　在IE 6.0的替代布局中，虽然图像滚动至可见区域之外，但作品的完整性仍没有受到影响

图3-4-5　在Opera 7.0中浏览prêt-à-porter。低于7.5版本的Opera浏览器不会显示水平滚动条，让作品变得毫无用处



Cedric Savarese，设计师

www.csszengarden.com/070

3.5 CS(S) Monk

含义丰富的定位，充分理解网格

是谁说作品一定要事先做好详细周密计划的？设计师Cedric Savarese正是通过在Adobe Photoshop中不断尝试，才最终完成了他的CS(S) Monk作品。这种从试验开始的方法非常灵活，当新的灵感出现时，设计师即可立即推翻或调整旧的设计。

Savarese让页首部分成为了作品的焦点，其他部分均环绕在其周围。在尝试过数百种不同字体之后，Savarese终于找到了最符合作品主题的一种。为了更加强调作品中“禅”的意味，Savarese还特地在页面中添加了花朵图案作为装饰。令人愉悦的红色和灰色基调也让作品魅力十足。

3.5.1 元素的位置以及其带来的含义

CS(S) Monk作品中最具有支配性的僧侣图像占据了页面的焦点位置——左上角。作品中的第一个元素的地位最为重要，它也被放在了最显眼的位置。

考虑一个固定大小的平面画布，例如一张纸，若在其水平和垂直方向都一分为二，那么将得到四个象限。西方文字的布局顺序是从左到右，从上至下。因此，浏览者一般都会无意中按照这个顺序来区分四个象限中内容的重要性。

你是否曾有过这样的疑问——为什么这么多的网站都把LOGO置于页面的左上角呢？原因就在于浏览者的目光一般都会首先移动到这里。将某个元素置于页面的左上部将毫无疑问地吸引住更多的关注，所以这里也自然成了LOGO的理想位置（图3-5-1）。

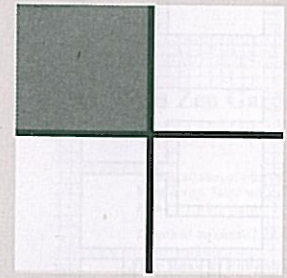


图3-5-1　左上象限

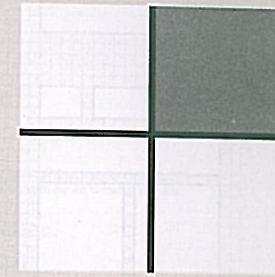


图3-5-2　右上象限

页面的右上部分同样非常重要（图3-5-2）——在任何页面中，无需滚动即可立即看到的区域都是最重要的部分。将某个元素置于页面的右上角意味着该元素的重要性略逊于左上角的元素，但仍远高于页面其他位置中的元素。

在大多数页面中，底部的两个象限并不是那么的重要（图3-5-3）。有些设计师在该处添加一条冗余的导航链接，还有些设计师只是简单地将版权条款置于此处，并没有任何别的修饰。在传统的印刷作品中，底部的元素将会让人觉得其上的元素位置有所升高、甚至有漂浮的感觉。印刷作品中很容易通过这种简单的布局营造出三维的空间效果，但在Web页面中，底部的元素却无法营造出同样的效果，甚至会完全被用户忽略。

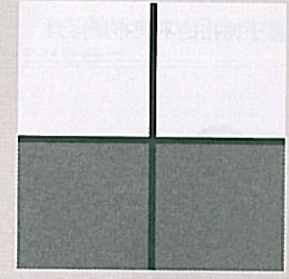


图3-5-3　左下和右下象限

页面底部的常见用途就是放置一些非必要的内容。对于那些没有足够层次性的链接，或是指向本页或网站其他页面的导航等，这里就是其最好的归宿。若是用户没看到这部分内容，也没什么大不了的——页面其他位置也有同样的实现。但若是用户有幸注意到了，那么这些内容也能够提供一些相当有价值的额外功能，帮助用户理解站点的结构。

3.5.2 网格

网格是一种重要的布局工具，但其呆板的样式往往给人过于严谨的感觉。因为在数学中，网格只不过是一系列相交线分割出的大小一致且极具逻辑性的方框而已。这些线条和方框能够在随后添加设计元素时起到辅助的作用。若在布局中严格依照网格，元素的比例和间距将变得更为协调统一，虽然这也可能让作品显得过于规整、缺乏创造性（图3-5-4）。

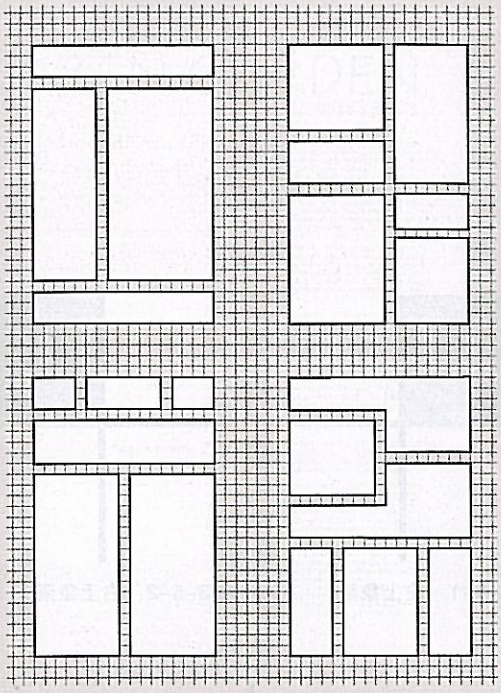


图3-5-4　基于网格的不同布局排列

经典的表格布局作品大都在无意中演示了过度结构化的网格所带来的限制——单元格中的元素别无选择，只能被束缚于其中。若想实现重叠效果，则需要相邻单元格的辅助来显示一部分效果。通常来讲，唯一能够打破网格限制的方法就是将图像切割成小片，然后再用网格中的网格——也就是复杂的嵌套表格——将其再次拼接起来。

而CSS却能够以非网格的形式对元素进行更为精确优雅的控制。对于一个基于网格结构布局的作品来说，难免会有一些元素需要摆脱网格的限制，以更加自由灵活的方式显示在页面中。CSS定位模型即可在这种情况下给元素以充分的自由，将元素布局在其容器之外，或是页面中的任何位置上。

3.5.3 创建网格

当然，超越网格结构的前提是创建网格。若以网格划分CS(S) Monk，则页首部分将被分成三块，主要内容区域将被分为两栏。

1. 页首

基于各个元素间相对重要性的高低，页首部分可分为三个不同的区域：僧侣图像、禅意花园的标题以及#preamble中的文字（该作品将这部分内容从常规文档流中提取出来，并置于页首）。

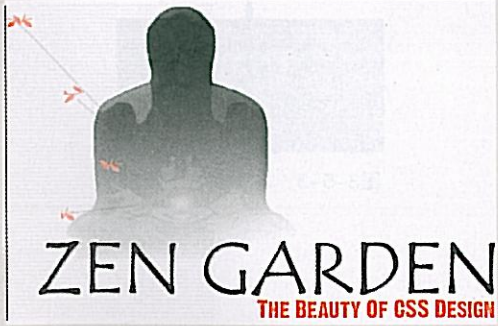


图3-5-5　僧侣图像和禅意花园标题被合并成一张图片

但若是深入研究一下CS(S) Monk页首部分的图片构成，就会发现上面提到的两个部分——僧侣图像和禅意花园标题被合并成了一张图片。虽然从视觉结构上来讲二者应该分属不同的部分，但为了方便CSS实现，仍将其看作一个整体。这是个Web设计中的惯例——为了方便随后的编码，原始图像的图层需要在保存前被合并（图3-5-5）。

2. 内容区域

虽然CS(S) Monk的主题内容为居中的单栏布局，但每一节的内容和标题都用不同外边距区分开来，看上去就像分为两栏一样。图3-5-6用简单的蓝色高亮展示了这两栏在页面中的分布，二者比例匀称，与页面配合得非常好（图3-5-6）。

左栏的宽度大概为右栏的2/3，即左右宽度比率为2:3。两栏之间的这种较为精确的比例可以自动划分：当指定好左栏的宽度之后，右栏的宽度就由页面中剩下的空间决定了。2:3也是个经典的和谐比率[1]，这样的分配使作品中的文字韵律十足，显得极为自然，让人愉悦。

若想了解更多设计中的比例知识，请参考Kimberly Elam的*Geometry of Design: Studies in Proportion and Composition*一书（由Princeton Architectural Press出版社2001年出版）。

3.5.4 跳出网格的限制

Savarese的CS(S) Monk作品大部分基于网格布局，但仍然恰到好处地让一些元素打破了网格的限制，为作品增色不少。

无论是绝对定位还是相对定位，CSS的定位模型都能让我们游刃有余地实现这样的想法。接下来对这两种定位方式的比较分析将帮助你做出最合理、恰当的选择。

1. 绝对定位

理解绝对定位的前提就是理解文档流（document flow）的概念。HTML是线性的——文档中可能首先出现一个h1元素，接下来是一个p元素，随后是个div元素。虽然元素允许嵌套，但若在不应用任何样式的情况下，我们很容易就能看出HTML解释器将从头到尾按顺序解析文档中的元素，并自上向下将元素依次显示在浏览器中（图3-5-7）。

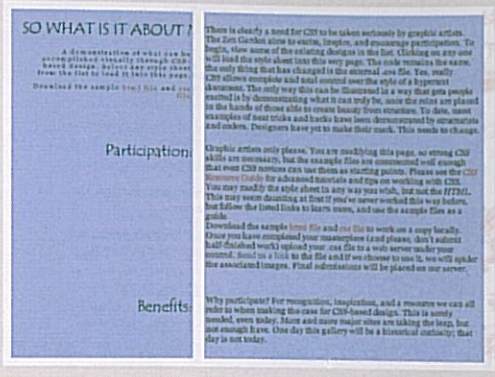


图3-5-6　正文部分隐含的两栏布局，符合网格布局

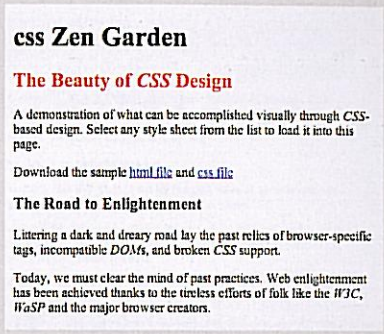


图3-5-7　未应用样式的HTML文档将自上向下线性地显示于浏览器中

[1] 2:3近似于黄金分割比率0.618。——译者注

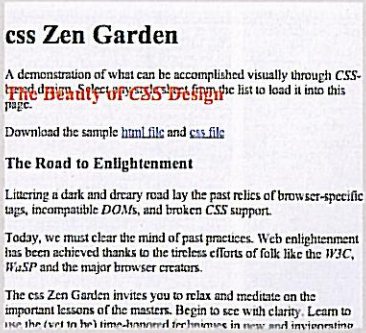


图3-5-8　其他元素忽略了绝对定位元素，就像根本不存在一样

而绝对定位则不仅可以将某元素置于页面中的任何位置之上，还能够将其从文档流中移除。绝对定位元素不会影响到文档中的其他元素，其他元素依然会以线性的方式进行布局，就像这个绝对定位元素根本不存在一样（图3-5-8）。绝对定位元素将被呈现于指定的位置，但考虑到页面中显示的其他元素，设计师一般都需要仔细斟酌，以处理元素之间可能出现的相互覆盖问题。

在唯一一种情况下，绝对定位元素也会影响到页面中的其他元素，即当某个元素被包含在该绝对定位元素中，且同样应用了CSS定位时。CSS定位模型将认为该元素的定位是基于其包含块（containing block）的——大多数情况下元素的包含块都是文档的根元素html。但若是在定位元素与文档根元素的层次关系中还存在着另一个定位元素，则子元素的定位将改为基于其父元素进行[1]。

若想了解更多包含块的知识，请参考CSS规范中的“10 Visual formatting model details”页面（www.w3.org/TR/REC-CSS2/vi-sudet.html#containing-block-details）。

2. 相对定位

相对定位则不会把元素从文档流中移除，而是基于其原位置进行一定的偏移。页面中其他的元素不会占据其偏移后留下的空白空间，就像该元素仍在原位一样（图3-5-9、图3-5-10和图3-5-11）。

若想为某元素应用一定的偏移，那么相对定位将是最好的选择：传统网格布局中的元素可以轻松地摆脱其束缚，移动到网格之外；在其他定位方案难以应用时，相对定位仍可广泛适用。例如，Savarese就用相对定位将那条龙的图案从其原本位置（正文开始处）移动到了页脚的左边。用绝对定位将元素置于页面底部附近要比将其置于顶部困难得多，这时使用相对定位将会更易于实现。

[1] 定位元素就将作为其子元素的包含块。——译者注

3.5.5 网格布局中的灵活性

除非仔细考虑并能良好把握作品的均衡性，Cedric Savarese使用的这种逐步求精的设计方式将让结果变得极为混乱。丰富的定位、比例知识将能够防止这类问题，或者用网格辅助设计也能达到较为和谐的目的。

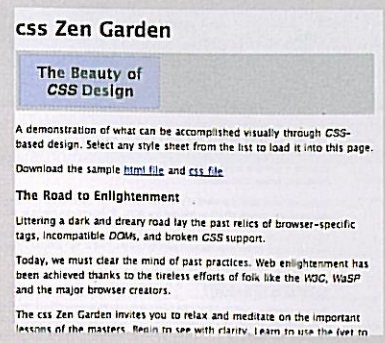


图3-5-9　常规文档流中的元素

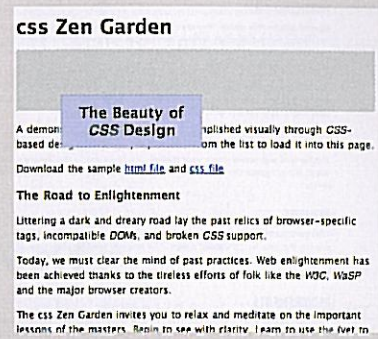


图3-5-10　相对定位元素仍将包含在文档流中，其原位置也会继续影响其他元素，但该相对定位元素的新位置将被文档流忽略

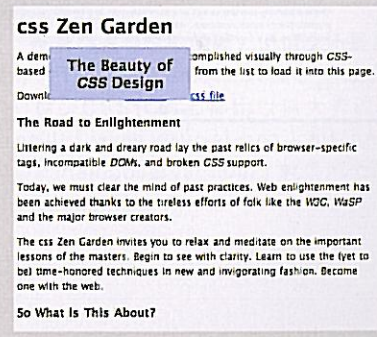
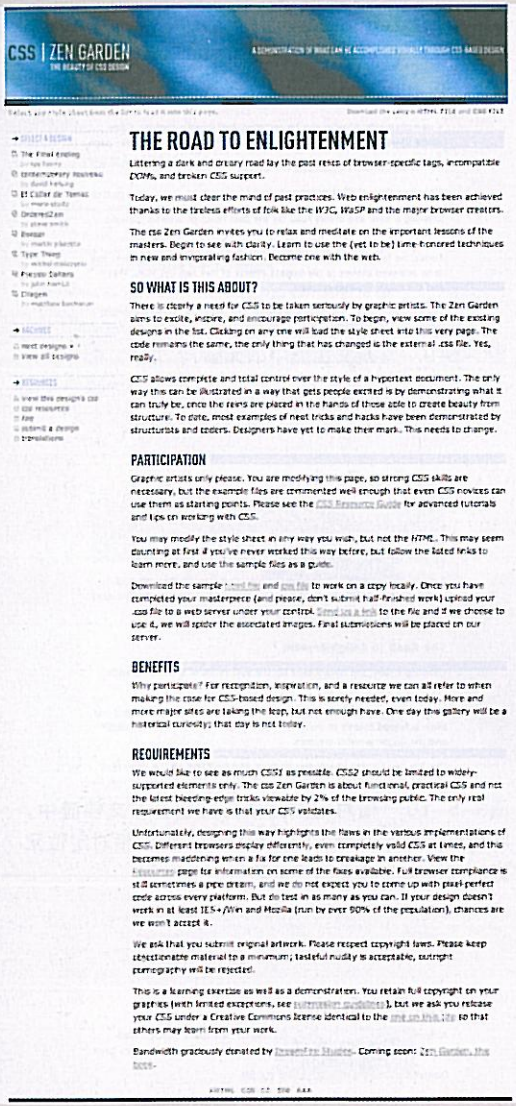


图3-5-11　相反，绝对定位元素将不会影响到文档流。该元素的原位置和新位置都将被文档流忽略



Dan Rubin，设计师

www.csszengarden.com/024

3.6 Not So Minimal

处理布局中常见的溢出问题

Not So Minimal的标题让我们产生了这样的疑惑：不想如什么一样的最小化？该作品有一段背后的历史：Dan Rubin向禅意花园提交的第一个作品叫做Grazay，可以在www.superfluousbanter.org/projects/zengarden/grazay/zengarden-grazay.html访问。

在Grazay（图3-6-1）中，Rubin有意地选择了较亮的色彩以及常见的字体（FF DIN Condensed和Verdana）。虽然禅意花园了解优秀的最小化设计作品的重要性，但网站建立的主要目的仍是为了反驳CSS只能用来实现最小化设计之类的言论，所以Grazay并不是禅意花园所希望看到的。因此，Rubin随后用一些图片对Grazay进行了润色，最终完成了现在这个作品——Not So Minimal。

3.6.1 内容溢出

CSS布局中最常见的一个难题就是如何应付多变的Web所带来的挑战。更明确一些，问题在于如何在不知道内容多少的情况下在CSS中预先定义容器的长宽。

内容长度不定似乎是Web的天性——页面中的内容可能极少，也可能极丰富。例如对于内容管理系统（content management system，CMS）之类的动态网站来说，页面内容既有可能一反常态地短小，也有可能让人难以置信地冗长。

除非CMS程序的开发者直接限制内容的长度——大多数情况下这似乎不现实——任何将要容纳可变长度内容的区域都要经过长短不一的文字的测试。与图像相比，文本内容的行为更加难以预测，因为图像的尺寸固定且能够预先了解，即不可缩放。但这也并不是硬性的规定——CMS中的某篇文章中可以插入多张图像，或者程序没有规定图像的高度，抑或其他的什么原因等都会导致页面中图像的大小不可预知。

目前，大多数浏览器均支持缩放页面的文本字号，很多视力不佳的人们也都依赖于这个功能。字号放大之后同样有可能导致类似的文本重叠问题。理论上，能够解决可变内容长度问题的主要内容区域也可以很好地处理由字号变大带来的问题，但类似导航链接之类的小块文字区域却往往难逃厄运。保证页面能够在150%甚至更高比率的文字大小设定中正常显示并不是杞人忧天——很多用户正是用这种字号浏览站点的。

无论如何，我们必然会遇到内容区域溢出的问题，或者是水平方向，或者是垂直方向。一些挑剔的用户往往对内容的重叠或某个包含大量内容区域的显示错位表现得极为苛刻。在实际设计中，最常见的两种内容溢出主要发生于两种情况中：绝对定位元素中内容过多，浮动元素尺寸超过其可容纳区域。

3.6.2 浮动导致的内容溢出

从技术角度讲，float（浮动）属性并不是用来布局的。CSS引入浮动的本意是让文本能够环绕在其中内嵌的其他元素周围，而绝不是为了页面布局。但即便如此，我们也不得不继续使用浮动，原因很简单——其他任何方法都没有“清除（clear）”功能，我们别无选择。

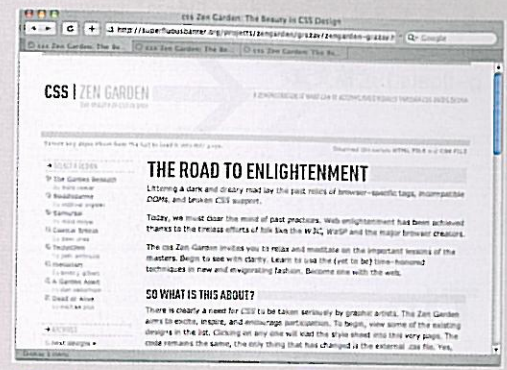


图3-6-1　Dan Rubin最初的作品Grazay演示了Not So Minimal的进化过程。

Note

此外，在同时使用基于标准设计和CMS时的另一个问题就是，很多CMS系统会生成一些不符合标准的过时标签，主要体现在其内容和表现样式混在一起，例如使用<font>标签等。且CMS系统大都支持用户开发自定义模块，以满足千变万化的需求，开发人员的良莠不齐也可能导致站点离Web标准的要求越来越远。

尽管存在着这些问题，但情况还是在慢慢好转，当前很多CMS系统在生成合法XHTML/CSS代码上都比以前大有改进。Drupal（www.drupal.org）和Mambo（www.mamboserver.com）就是两个流行的开源CMS系统。

Tip

Eric Meyer写过很多清除浮动的相关文章。不要错过“Containing Floats”（www.complexspiral.com/publications/containing-floats）这篇文章。

Tip

还有一种无需专门元素的清除浮动方法。该方法需要使用一些比较巧妙的hack来实现跨浏览器兼容，请参考“How to Clear Floats Without Structural Markup”www.positioniseverything.net/easyclearing.html）。

以最简单的方式描述，clear属性及clear的含义可以被理解为让某一元素不再跟随在另一元素的后面。大多数情况下，绝对定位导致的内容溢出都源于该方案缺少了这种“清除跟随”的功能。

但浮动布局也不是完美的，很多浏览器的缺陷更是让我们在使用浮动时举步维艰。

1. 用标签清除浮动

若要在布局中使用浮动，则HTML中至少要有两个元素：一个应用浮动，另一个则用来清除浮动。例如，页面中可能有两个div分别浮动在左右两侧，这时我们还必须要引入第三个元素来清除浮动。我们可以用现有元素顺便清除浮动，例如用紧随其后的p元素清除前面h3元素的浮动属性。

但有时我们却不得不额外引入一个“罪恶”的空元素来完成清除浮动的重任。若这种情况的确无法避免，我们推荐使用“全能”的空白div元素，类似如下代码：

```
<div class="clear"></div>
```

在该CSS Class中即可定义clear属性。虽然在HTML中添加的这个无意义标签似乎有些丑陋，但我们已经尽力了。

2. 精确控制浮动元素

若在一个装满了水的容器中再撒入一把沙子，会发生什么情况？自然，容器装不下这么多东西，一部分水将溢到容器外面。

在使用浮动时也会受到类似的限制，最常见的就是水平方向溢出。若两个50%屏幕宽度的浮动元素并排放置于页面上，且某个元素还有着100px的外边距，那么第二个就会被挤出去，最终排布在第一个的下面。考虑到这些，我们必须预先精确计算好浮动元素的宽度。

但对于width和height属性，Windows下的IE却并没有遵从CSS规范中的建议。在IE中最有可能发生的就是元素的实际宽度大于其CSS width属性所指定的宽度。这将导致两种可能的结果：一种是内容的布局受到影响，另一种是容器的宽度仍保持不变，但其中的部分内容却显示在容器之外。但不管怎样，即使内容越过了包含它的容器边界，甚至与其他元素重叠在一起，但浮动效果仍不会受到影响。

此外，IE还会在必要时自动扩张元素的宽度。哪怕是某个浮动元素的实际宽度只比其预期宽度多1个像素，仍会将其他的浮动元素挤出去（图3-6-2）。我们可以考虑用如下几种方法来解决这个问题。

• 增大区域的宽度

若能够适当增大浮动元素的容器宽度，那么通常这个问题也就不复存在了。虽然谈不上最理想的解决方案，但却是应该最先考虑的，因为这种方法的实现过程相对简单。

• 限制内容宽度并注意斜体

如果能够绝对保证浮动元素中的内容不会造成水平溢出，那么也就自然解决了这个问题。但这种方法通常却并不可行，因为若在最初设计时其中内容不多的话，我们很难在一开始就发现这个问题。

一个更加难缠的问题就是，即使内容的宽度完全合理，仍旧无法避免出现内容溢出。这个问题特别常见于Windows上的IE浏览器中。如果正被这个问题所困扰，且浮动元素中的某些文字是以斜体显示的，那么可以暂时取消斜体效果，看看是否解决了问题。IE中的斜体文字非常有可能导致水平溢出，这正是问题的元凶。虽然理论上任何的文本格式都不会影响到布局，但这里事情却的确发生了。

• 限制内容溢出

在下一节介绍垂直溢出时，我们将研究overflow属性，但在防止水平溢出时，overflow也能助上一臂之力。这种用overflow进行限制的方式并不是那么优雅，所以在使用时要倍加小心。

3.6.3 绝对定位导致的内容溢出

绝对定位中遇到最多的问题就是垂直方向的内容溢出。但也有一个例外——若绝对定位元素的预定宽度不足以容下其子元素，也会发生水平内容溢出（图3-6-3）

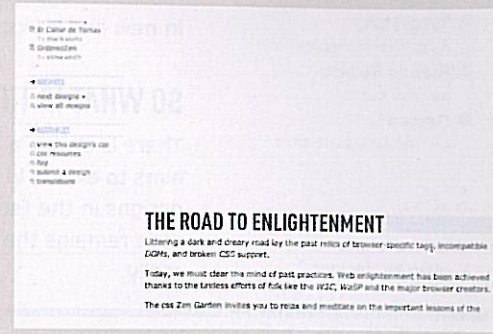


图3-6-2　若几个浮动元素的宽度总合大于其容器的宽度，则一个或几个元素将被挤到其他元素的下方

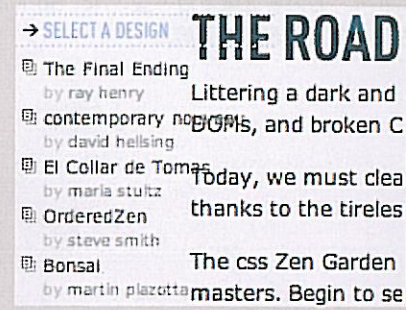


图3-6-3　对于100像素宽的容器来说，150像素宽的#linkList子元素显得太大，将导致水平溢出

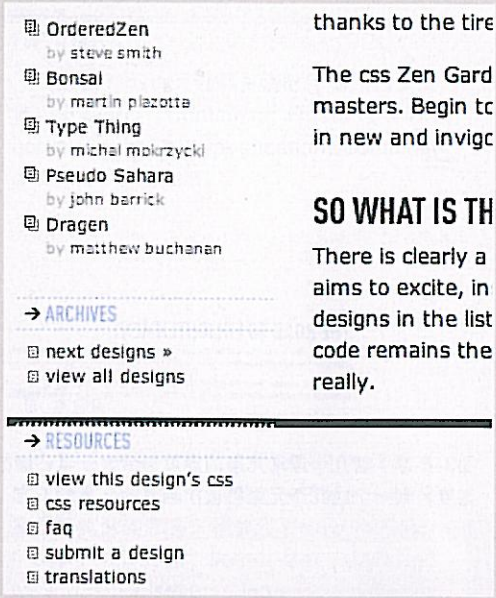


图3-6-4　若内容区域更短一些，Not So Minimal的左侧导航条将覆盖在页脚上

垂直溢出

内容垂直方向溢出的问题更加常见。原因在于，绝对定位元素不像浮动元素那样能够被“清除”。为元素指定位置，也就意味着将其从文档流中移除，该绝对定位元素自然无法再影响到其周围的元素了，它的宽度并不能限制其中子元素的宽度。

在处理页脚时我们经常会遇到这个问题——某个绝对定位的栏的高度可能会随内容的增多而变大，直至变成页面中最高的一栏，最终覆盖到页脚区域之上，甚至更往下（图3-6-4）。很不幸的是，对于这种情况，我们并没有什么足够安全的解决方案，只能给出一些权宜之计。

• 用浮动代替绝对布局

最有效的方案就是用浮动来代替绝对布局，并使用其“清除浮动”特性。虽然这种方法并不总是管用，但如果有可能的话，还是非常值得我们去尝试一下。

• 在设计中尝试避免该问题

另外，我们也可以尝试让绝对定位元素在垂直方向上能够无限伸展。若某元素可能覆盖页脚，那么干脆为页脚添加一定的外边距，给该元素让出足够的空间。这样若是这个绝对定位元素下面没有任何其他元素的话，覆盖自然无从谈起，问题也得到了解决。

• 限制溢出

使用CSS中内建的自动溢出控制功能要比强制控制内容长度好得多。overflow正是为此而设计的，但它却并不是你所期望的“银弹”。若你将某元素的高度固定为150px，且将其overflow属性设置为auto的话，那么若其中内容高度超过了150px，超过部分将被截去并显示出滚动条。

太多的滚动条很快会让用户对页面失去兴趣。为了避免这个麻烦，我们可以将overflow属性设置为hidden，但在移除滚动条的同时也会永久性地隐藏超过容器高度的那部分内容。大多数情况下都不应该这样做，因为隐藏的内容对于浏览者来说将变得毫无意义，但在某些少见的情况下，这种表现或许正是我们所期望的。

• 使用脚本

如果不介意的话，可以使用一小段JavaScript来辅助CSS布局。禅意花园的贡献者之一Shaun Inman给出了一段非常有用的脚本，用来“清除”绝对定位元素，其行为与清除浮动非常类似。这段脚本可以在他的“Absolute Clearance”（www.shauninman.com/plete /2004/07/absolute_clearance.php）一文中找到。

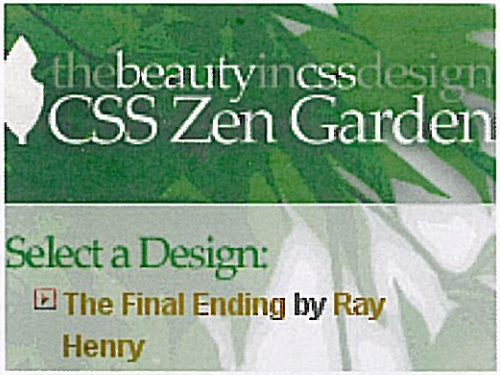
• 用em而不是px作为长度单位

这个技巧仅适用于由用户增大字号所导致的内容溢出中，而对于因内容长度可变造成的内容溢出，这种方法却无能为力。用px作为单位进行绝对定位将能够最精确地控制布局，但同时也丧失了布局的灵活性——一旦给定了元素的位置，那么它将始终固定在原地。但若是用em作为left、top等属性的单位，元素的位置将变得略加灵活一些——这个位置会根据用户选择不同的字号而相应地改变。若是重叠在一起的若干元素都是用em单位，那么任意一个位置的变化都将让其余元素的位置随之调整。虽然定位不是那么精确，但很多时候这种方法却相当好用。

3.6.4 设计时考虑溢出

由于右面的正文区域较长，所以Not So Minimal的左侧导航条并没有受到内容溢出的影响。但是随着禅意花园的不断壮大，各种不同的页面可能会让导航条越来越长，最终超过内容的高度。这时，类似图3-6-4所示的覆盖问题将不会再像现在这样鲜有出现，而是会变得越来越严重，设计师也不得不需要更加周全的考虑。

虽然设计方案不应该由技术实现主导，但对于大型网站来说，内容溢出确实是个棘手的问题。若能在设计时就从根本上避免了这个问题，那么上述几个远远称不上完美的权宜之计也能保证不会让我们在编码实现阶段感到捉襟见肘。



Japanese Garden（日式花园），116页

Revolution!（革命！），122页

No Frontiers!，134页

Coastal Breeze，140页

第4章

图像

Deco（装饰派），128页

What Lies Beneath，146页

设计师的主要工作之一就是创建各类图像，力求带给浏览者最大限度的视觉冲击。本章选择的各个作品都将图像运用得炉火纯青。各种不同的图像创作过程也会将作品引领至不同的方向。

本章将介绍很多图像相关的内容，包括获取灵感的方法、图像处理技术及技巧、一些资源丰富的照片库，以及根据情况需要为作品创建图像的方法等。

the beauty in design
CSS Zen Garden

Select a Design:
- Burnt Offering by Jenny Blair
- Salvage Yard by Justin Peters
- Switch On by Michael Pasani
- Mountain Resort by Jordi Romkema
- Grüener Entwurf by Hannah F. Liesong
- Perfume de Gardenias by Armando Sosa
- Pneuma by Adam Polselli
- Birdhouse by Justin Vitale

Archives:
- next designs »
- View All Designs

Resources:
- View This Design's CSS
- CSS Resources
- FAQ
- Submit a Design
- Translations

A demonstration of what can be accomplished visually through CSS-based design. Select any style sheet from the list to load it into this page.

Download the sample html file and css file

悟りへの道
The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support.

Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser makers.

The csz Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

何のためにあるの？
So What is This About?

There is clearly a need for CSS to be taken seriously by graphic artists. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The code remains the same, the only thing that has changed is the external .css file. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. To date, most examples of neat tricks and hacks have been demonstrated by structurists and coders. Designers have yet to make their mark. This needs to change.

参加
Participation

Graphic artists only please. You are modifying this page, so strong CSS skills are necessary, but the example files are commented well enough that even CSS novices can use them as starting points. Please see the CSS Resource Guide for advanced tutorials and tips on working with CSS.

You may modify the style sheet in any way you wish, but not the HTML. This may seem daunting at first if you've never worked this way before, but follow the listed links to learn more, and use the sample files as a guide.

Download the sample html file and css file to work on a copy locally. Once you have completed your masterpiece (and please, don't submit half-finished work) upload your .css file to a web server under your control. Send us a link to the file and if we choose to use it, we will spider the associated images. Final submissions will be placed on our server.

特典
Benefits

Why participate? For recognition, inspiration, and a resource we can all refer to when making the case for CSS-based design. This is sorely needed, even today. More and more major sites are taking the leap, but not enough have. One day this gallery will be a historical curiosity; that day is not today.

必要条件
Requirements

We would like to see as much CSS1 as possible. CSS2 should be limited to widely-supported elements only. The csz Zen Garden is about functional, practical CSS and not the latest bleeding-edge tricks viewable by 2% of the browsing public. The only real requirement we have is that your CSS validates.

Unfortunately, designing this way highlights the flaws in the various implementations of CSS. Different browsers display differently, even completely valid CSS at times, and this becomes maddening when a fix for one leads to breakage in another. View the Resources page for information on some of the fixes available. Full browser compliance is still sometimes a pipe dream, and we do not expect you to come up with pixel-perfect code across every platform. But do test in as many as you can. If your design doesn't work in at least IE5+/Win and Mozilla (run by over 50% of the population), chances are we won't accept it.

We ask that you submit original artwork. Please respect copyright laws. Please keep objectionable material to a minimum; tasteful nudity is acceptable, outright pornography will be rejected.

This is a learning exercise as well as a demonstration. You retain full copyright on your graphics (with limited exceptions, see submission guidelines), but we ask you release your CSS under a Creative Commons license identical to the one on this site so that others may learn from your work.

Bandwidth graciously donated by DreamFire Studios.

xhtml css cc 508 aaa

Masanori Kawachi，设计师

www.csszengarden.com/096

4.1 Japanese Garden

（日式花园）

理解图像文件格式以及优化方法

“和”「わ」在日语里是协调、融洽的意思。虽然大胆、生动的设计也许会让作品更有吸引力，但显然Masanori Kawachi在Japanese Garden中要表现的是一种更易理解的、全球范围内的美感。Masanori Kawachi尝试在各种风格鲜明的作品中脱颖而出，牢牢把握住“和”的宗旨，让作品能够被全球范围内的各种欣赏者所接受。

Kawachi所使用的设计元素显然带有日式风格，但痕迹却不是那么明显，任何国家的浏览者都会对其产生兴趣。

4.1.1 图像文件格式

Japanese Garden中所用到的种种精巧的透明效果都能够暗示出设计师对各种图像文件格式及其使用方式的深刻理解。计算机图形学领域中有上千种不同的图像格式，但能在Web页面中使用的并不多，最为常用的有三种格式：GIF、JPEG和PNG。这三种格式的图像均支持压缩，以节省浏览者下载时的网络带宽。

为什么优化如此重要

为什么要优化？随着宽带越来越普及，我们还需要如此在意图像的大小么？虽然宽带上网越来越普及，但这却不是任何地方的任何用户都能享受到的。很多用户仍旧使用56K的调制解调器上网，有些国家中连电话线都屈指可数。多下载1K的文件都要占用额外的时间，多上网一分钟也要花费额外的金钱。

同样需要考虑的还有服务器端。高流量的站点通常都有很多用户同时访问——每个用户耗费的带宽越多，服务器需要的总带宽也就越大，购买带宽的开销也会越大。无论对于用户还是网站运营者，图片文件大小的降低都意味着更少的开销。2K的优化看上去不算什么，但其累积的过程却非常快。例如，若日访问量为100 000的站点中某张图片的大小减少2K，那么每个月将节省6G的带宽。这种细微之处的优化显然可以实实在在地降低网站运营的开销。

Note

在2004年中期，美国国内家庭的宽带普及率已经超过50%。

4.1.2 理解并选择图像文件格式

每种图片文件格式都有自己的色彩模型、表示形式、压缩方案以及透明效果支持dent特性。只有充分地了解每一种格式的优劣之处，才能根据实际需要做出合理的选择。

无损压缩将保留原图像的所有细节，有损压缩会对细节有所取舍，以换来更小的文件。二者都有其适用的地方。

图4-1-1　256色的GIF图像，文件大小为8.6KB

图4-1-2　32色的GIF图像，文件大小为5.0KB

图4-1-3　4色的GIF图像，文件大小为2.1KB

图4-1-4　高品质（70）的JPEG图像，文件大小为8.3KB

图4-1-5　中品质（30）的JPEG图像，文件大小为3.7KB

图4-1-6　低品质（10）的JPEG图像，文件大小为2.5KB

色深（color depth）是指某种图像格式包含的颜色的多少。色深用位（bit）计量，更高的色深表示将包含更多的颜色种类。例如8位的色深只有256种颜色，而24位色深则能够包含一千六百万种颜色。

1. GIF

GIF出现的历史最早，大约在20年前就已经初具雏形。GIF使用一种叫做LZW的无损压缩格式，非常适合带有大块相同颜色区域的图像。单色插图、文本、色块等文件非常适合存为此格式。

GIF需要使用颜色索引来实现无损压缩，但这却是把双刃剑。该格式的色深只允许同一张图像中最多出现256种颜色，对于超过范围的若干种颜色只能忍痛割爱。但对于少于256种颜色的图像，其颜色索引也会相应地变小，这也就意味着图片文件的整体大小也将随之降低（图4-1-1~图4-1-3）。当然，对于超过256种颜色的图像，我们仍可以将其存为GIF格式，不过某些颜色信息将丢失——最终变成有损压缩。

GIF支持1位的透明度，即某个像素要么完全透明，要么完全不透明，没有任何中间值。若在图像编辑软件中使用了各种范围的透明度，那么在保存为GIF的时候将会受到限制并不得不进行妥协。但若是预先知道该图像在Web页面中所处区域的背景颜色，那么这也不会带来太大问题。此外，GIF还支持基于桢的简单动画以及隔行加载技术，但这两者都将增加文件大小。

2. JPEG

JPEG的24位色深让其能够很好地显示照片等色彩、细节丰富的图像。JPEG支持有损压缩且可以设置压缩等级，这样我们即可通过合理的压缩等级找到图像颜色和细节与文件大小之间的平衡点（图4-1-4~图4-1-6）。

压缩比很大的JPEG图像将会高度失真，并引发一种叫做非自然痕迹[1]（artifact）的小块像素缺陷问题。虽然图像中的丰富细节将会淡化非自然痕迹的影响，但在使用JPEG时仍要把握好图像质量和文件大小之间的平衡关系。JPEG不支持任何形式的透明度，但最新的“渐进JPEG（Progressive JPEG）”版本能够和GIF一样实现隔行加载，且已经被大多数浏览器所支持（请参考本节稍后的4.1.4节）。

[1] 有的翻译将其叫做膺像。——译者注

3. PNG

PNG格式在有些方面与GIF类似，但它提供了两种色深模式：24位和8位。24位色深的PNG文件通常都很大，但却能保证完全无损。原始图片的所有细节都将被精确地保留。8位色深的PNG文件使用了类似GIF的颜色索引。但很多时候，同样设置下用PNG压缩的图像要比GIF更小（图4-1-7~图4-1-9）。

由于PNG同时支持这两种色深模式，所以理论上它可以用于任何图像。但实际上最好还是用GIF和JPEG，前者用于带有大块相同颜色区域的图像，后者用于照片等色彩、细节丰富的图像。24位色深的PNG文件虽然能够做到完全无损，但这却是以增加文件大小的代价换来的。

PNG还支持两种模式的透明度设置：1位（bit）或8位。前者与GIF的行为一样，而后者则能够实现256级的透明度。当我们无法预知图像在Web页面中所处区域的背景颜色时，8位的透明度模式将非常实用。但不幸的是，浏览器支持问题却让PNG的透明度支持大打折扣。其中最明显的就是Windows平台上的IE浏览器，它无法正确呈现出PNG的透明度设置[1]。

图4-1-7　24位的PNG图像，文件大小为16.2KB

图4-1-8　8位、64色的PNG图像，文件大小为5.3KB

图4-1-9　8位、4色的PNG图像，文件大小为2.2KB

4.1.3　透明度限制

虽然计算机领域中仍有很多其他图像格式，但只有上面介绍的这三种才适合在Web中使用。JPEG根本不支持透明度，GIF的1位透明度设置又显得太简陋，但PNG所支持的8位透明度又不能被最流行的浏览器识别。这个问题已经由来以久了，甚至与CSS也没有什么关系。在过去的十年中，Web中的透明效果只能由一种方法实现，那就是模拟——预先设计并呈现好整个图像，包括其中的半透明效果，然后以无透明度的文件格式保存。

在Japanese Garden中，Kawachi将内容放在了看起来似乎半透明的栏中，透过这一栏甚至还能隐约看到作为背景的绿色的树叶（图4-1-10）。虽然并不支持这种类型的半透明设置，但Kawachi还是选用了GIF。其半透明效果的实现不是在浏览器中完成的，而是在Adobe Photoshop中直接使用图层技术做出的效果，然后直接保存为GIF。文本栏中作为背景的图像其实也根本没用到任何透明设定——不过是普通的完全不透明图像而已。虽然最终结果看上去像是有半透明效果，但这却是通过在图像处理软件中预先处理得到的。

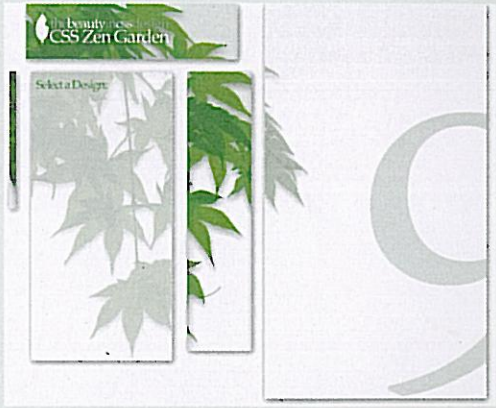


图4-1-10　Japanese Garden的半透明效果是由一些不透明或伪透明的图像组成的

[1] IE 7.0中已经可以正常识别并显示出PNG图像的透明度设置。——译者注

图4-1-11　为白色背景设计的透明GIF图像，置于白色背景之上

图4-1-12　为白色背景设计的透明GIF图像，置于淡绿色背景之上

图4-1-13　为白色背景设计的透明GIF图像，置于深绿色背景之上

在使用透明GIF图像时必须格外小心——因为它仅支持一个级别的透明度设定。所有其他半透明的像素都要恰当地与背景颜色融合在一起，即所谓的“蒙版颜色（matte color）”。蒙版必须与该GIF周围的背景颜色协调一致，否则将呈现出明显的不协调边缘（图4-1-11~图4-1-13）。

通常情况下，Web页面中只有这两种实现透明的方法，二者都经过测试且足够可靠。虽然PNG的半透明设定功能强大，但IE的漠视让我们不得不放弃使用它的想法。

虽然透明PNG图像在IE中显示得一团糟，但仍有一些办法让其正常工作。这些方法只应用于IE中，且自身都有一定的限制，但总归聊胜于无，迫不得已时仍可以考虑使用。

4.1.4　优化技巧

降低文件大小的诀窍在于适当牺牲一定的图像质量。图像压缩比越大，结果自然越难看。所以最好的方式是从较低品质（即较高压缩比）的图像开始，逐步提高品质，直到对图像质量满意为止。

1. GIF

优化GIF的步骤通常是从最低的颜色级别开始，逐步提高颜色数目，直至可以接受。若从8种颜色开始，那么接下来就依次查看16、32、64（依次类推）种颜色的样子，直到你觉得满意。某些图像确实需要256种颜色（若是超过256种颜色，那么就考虑使用JPEG代替GIF了），但还有一些图像却不那么敏感，哪怕是在8或16种颜色中看起来也和未经压缩过的差不多。

还有一些设定能够影响到GIF格式文件的大小。例如抖动技术（dither）即可在一块相同颜色区域中添加另一种颜色的像素点，这个技术可以用于模拟出一些本不在文件颜色索引中的颜色。但抖动技术也将影响文件的压缩比率。若是只考虑降低文件大小，那么某些时候增加颜色种类可能要比使用抖动技术更为有效。

Photoshop中有一个相关的新特性，即保存GIF文件时的有损（Lossy）压缩选项。无论已经将文件压缩到了多小，有损压缩选项都能在继续降低图像质量的代价下得到更大的压缩比率。甚至一个很小的有损压缩值（5%~6%，并不会明显影响到图像的质量），都有可能让GIF的文件大小降低20%~30%。

2. JPEG

优化JPEG则更为简单。在将原始图像保存为JPEG格式时，图像处理程序一般都会用滑动条表示图像的品质等级。高品质意味着更大的文件，低品质则意味着更小的文件。这样在保存时，我们即可从较低值开始，逐步增加图像的品质，直到可以接受为止。

JPEG所提供的另外一些高级特性目前已经被大多数浏览器支持。其中“优化（Optimized）”选项能够略微提高图像的压缩比。“渐进JPEG（Progressive JPEG）”对于庞大的照片文件来讲非常合适，只是需要添加一些额外的数据。而“Blur（模糊）”选项，顾名思义，则能够通过让图像略加模糊而降低文件大小。

3. PNG

优化8位PNG图像的步骤与优化GIF完全一致，提供了颜色级别以及抖动选项。24位PNG图像的优化更为简单：无需任何关注。因为PNG的唯一压缩方式固化在文件格式中，根本无法更改。

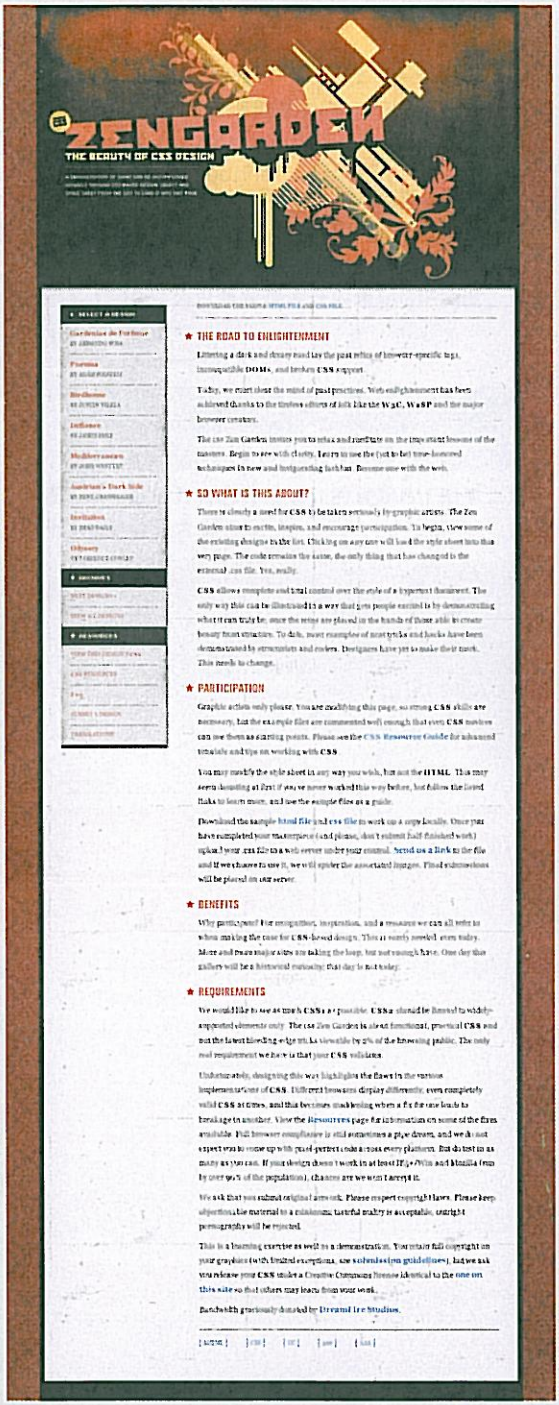
4.1.5 困难的选择

在所有浏览器都支持PNG的半透明行为之前，Masanori Kawachi预先生成半透明效果之类的技巧仍然会继续使用下去。

选择合适的图像文件格式也就是在透明支持、色深以及压缩比率三者之间取舍的问题。这三者并没有什么必要的联系，但限于文件类型，又不得不将其放在一起考虑。有时我们需要做出一定的妥协，只有了解了可选范围，我们才能够决定最适合当前情况的图像文件格式。

Note

下面一些文章介绍了让Windows上的IE也能正常显示PNG图像透明度信息的技巧：“PNG Behavior”（http://webfx.eae.net/dhtml/pngbehavior/pngbehavior.html）以及“Cross-Browser Variable Opacity with PNG: A Real Solution”（www.alistapart.com/articles/pngopacity）。



David Hellsing，设计师

www.csszengarden.com/102

4.2 Revolution!（革命！）

在CSS中为作品应用图像

图像在传达信息、营造氛围和吸引目光方面都能起到独到的作用。掌握正确、合理的应用图像方法将是我们刻不容缓的任务。

就像紧张、寂静的战场上的第一声枪响一样，Revolution!页首部分的那张大幅图片一下子就吸引了我们的注意。David Hellsing深谙这类表意性很强的图像的力量，他找到了一些从前的宣传海报，加上一些能体现历史感的图像，例如曾被浸湿的旧纸等，来营造作品的整体氛围。鲜血和标语、泥土和红星——这些元素组合在一起，让作品给浏览者以强大的视觉震撼。

但保证作品的平衡也不容忽视。虽然页首部分给人以好战、政治等相关暗示，但Hellsing却在内容区域选择了较为平静的风格。Hellsing首先在Adobe Illustrator中绘制出了图形和文字，然后用Adobe Photoshop将各个元素组合起来，最终完成了这幅充满活力的作品。

4.2.1 实际应用

在使用CSS进行布局时，要将那些预先设计好的图像（例如Hellsing的页首图像）应用到作品中。若某个设计师习惯使用<img>标签引入图片，那么在查看禅意花园的源代码时他定会感到些许疑惑，甚至不安——其中居然并没有什么显式的图像声明代码。其实，引入这些图像是在CSS文件中实现的。那么这又是如何完成的呢？

CSS提供了若干种将图片应用到页面元素上的方法，但只有背景图像方法得到了广泛支持，也就是说只有这一种方法才能用于实际开发中。CSS中为元素（例如禅意花园的h1标签，图4-2-1）应用背景图像的语法非常简单：只要定义图像的路径、位置以及重复显示方式（如果需要重复显示的话）即可：

```
h1 {
  background: url(path/to/image.gif) left top no-repeat;
}
```

若是设置了no-repeat，那么必须精确定义该背景图像的容器大小，以确保图像能够完整显示。若未指定图像的width或height属性，除非图像本身尺寸不大，否则很有可能因为容器中没有足够将其撑大到完整尺寸的文字，导致背景图像无法完整显示（图4-2-2）。所以大多数情况下，我们都要为容器元素显式指定大小，以便完整地显示出其背景图片。

```
h1 {
  background: url(image.gif) top left no-repeat;
  width: 400px;
  height: 20px;
}
```

4.2.2 图像替换

我们并不总是要文字呆在背景图像之上，很多禅意花园的作品都故意隐藏了标题等元素中的文本，取而代之的是一幅生动的图像。

2003年3月，Douglas Bowman在他的网站Stopdesign上发布了一个技巧（www.stopdesign.com/articles/replace_text），让设计师能够用一张背景图像替代某元素中的原始文字，以期显示出更美观的字体。为了表示对发明者

图4-2-1　h1元素应用了背景图像，其中文字仍然可见

Tip

注意背景属性中值的声明次序。例如对于背景图像的位置，我们必须要先声明水平位置，然后再是垂直位置，因此这段代码中left先于top出现。若使用left或top之类的位置关键字，那么其顺序要求并不是那么的严格，但若使用长度单位值，例如%或px，那么顺序就就变得至关重要。

图4-2-2　背景图像在文本结束时即被拦腰截断，只能显示出一部分

Note

我们同样可以用CSS生成的内容来为元素应用图像，这种技术可以让我们在CSS中用一段自定义的内容替换元素的原有内容：

```
h1 {
    content: url("aldente.gif");
}
```

在CSS 2引入:before以及:after两个伪元素（pseudo element）之后，CSS 3中又定义了上面代码中提到的content属性，以便更加全面地控制页面的表现。但大多数浏览器还不支持这个属性，所以目前还不能将其应用到实际开发中，只能用在实验性质的尝试中。

Note

屏幕阅读器是一种建立于浏览器（例如IE）之上的程序，所以它也不会读出浏览器中的不可见文本。

Todd Fahrner的尊重，这个技巧随后也被叫做“Fahrner图像替换（Fahrner Image Replacement，FIR）”。该技巧的实现非常简单：用span将元素中的文本包围起来，然后通过应用CSS样式隐藏这个span中的文本，最后将背景图像应用到该元素之上即可。例如，对于下面的这一段HTML标记：

```
<h1 id="pageHeader"><span>css Zen Garden</span></h1>
```

我们可以使用下面的这段CSS容易地实现图像替换：

```
#pageHeader {
    background: url(lemonfresh.gif) top left no-repeat;
    width: 400px;
    height: 20px;
}
#pageHeader span {
    display: none;
```

通过使用CSS的display: none或visibility: hidden，所有在#pageHeader元素中的span元素都将被隐藏起来。Hellsing同时使用了上述两种CSS设定——但它们的效果却是一致的。图像替换技术非常强大且很快开始流行——若是没有了它，我们甚至都无法建立禅意花园。该技术可以说是实现复杂、灵活CSS布局的一块最为重要基石。

4.2.3　图像替换的责任

图像替换技术使用display: none的本意并不只是想要替换文本，这样做还有一些更深层次的理由。实际上，若是没有任何提示或帮助，计算机就无法认出或读取图像中包含的文字。例如HTML中的img元素，若是没有了alt属性，那么对于Google等搜索引擎，以及辅助浏览设备（例如，屏幕阅读器即可阅读页面内容，并以声音的形式告知浏览者）之类无法呈现图像的客户端来说，将变得毫无意义。而图像替换技术则保留了被替换元素中的原有文本，因此无论对何种客户而言，理解页面内容都不成问题。

但图像替换技术还有一个问题：Google能够读取到隐藏的文本，而有些屏幕阅读器却不能。图像替换技术的本意是让视觉不便用户也能够获取图像传递的信息，但实际上它却没有达到预期的目标。不但如此，若用户在浏览器中禁用了图像，但却仍保留CSS，那么屏幕上将什么都不会显示。且这些额外的span也增加了页面文件的大小，虽然并没有太大的影响，但毕竟不

够理想。

4.2.4 更好的技术

随着图像替换技术一天天地流行起来，这些问题也渐渐开始受到越来越多的人的关注。有没有这样一种技术，既可以为普通浏览者隐藏一些不必要的元素，也能将这些信息提供给需要的人呢？幸运的是，几乎在图像替换技术开始流行的同时，另外一些替代的技术也开始走向成熟。其中有些在实现了一些最为基础的要求的同时，也留下了一两个悬而未决的显著问题；而另一些则需要大量的CSS技巧及hack来实现跨浏览器兼容。在撰写本书时，尚没有一种完美的替代方案出现，我们只能在权衡利弊之后，根据实际需要在这些现有的方案中进行挑选。

1. Leahy和Langridge的方案

Seamus Leahy和Stuart Langridge均独立地发现了这种方法。该方法让我们不必再添加那些多余的标签，且在保证屏幕阅读器可以正常阅读文本的同时，也能在页面中隐藏元素中的内容。但不幸的是，由于6.0版本之前的IE浏览器均不遵守盒模型，所以对于这类浏览器，我们还必须使用一些特殊的技巧来实现兼容。

• HTML标记

```
<h3 id="header">I like cola.</h3>
```

• CSS代码

```
#header {
  padding: 25px 0 0 0;
  overflow: hidden;
  background-image: url(cola.gif);
  background-repeat: no-repeat;
  height: 0px !important;
  height /**/:25px;
}
```

优点：屏幕阅读器可正常访问；没有多余的。

缺点：并没有解决浏览器禁用图像后空白页面的可访问性问题；实现需要hack。

Note

很多人都对图像替换技术发表了看法（例如Dave Shea的“In Defense of Fahrner Image Replacement”：www.digital-web.com/articles/in_defense_of_fahrner_image_replacement），并进行了研究（请参考Joe Clark的“Facts and Opinion About Fahrner Image Replacement”：www.alistapart.com/articles/fir）。很明显，图像替换技术的原本意图根本无法实现——这种在提高CSS地位中至关重要的、最为基础的技术实际上却是没用的。那我们又该怎么办呢？

Note

Dave Shea的个人网站上有一张页面（www.mezzoblue.com/tests/revised-image-replacement），专门用来跟踪时下最新的图像替换技术。

浏览器支持：Windows——IE 5.0及以上版本、Netscape 7、Opera 6及以上版本、FireFox；苹果机——IE 5.2、Safari、Firefox。

2. Rundle的方案

设计师Mike Rundle提出了一种使用负text-indent属性值的方法，将文本推到屏幕的左边缘之外。虽然该方案的适用性并不广泛——IE 5.0会将背景图片随文本一起推出浏览者的视线范围——但却相当简洁优雅。

• HTML标记

```
<h3 id="header">Apple pie with cheddar?!</h3>
```

• CSS代码

```
#header {
  text-indent: -5000px;
  background: url(sample-image.gif) no-repeat;
  height: 25px;
}
```

优点：屏幕阅读器可正常访问；没有多余的；简洁优雅的CSS。

缺点：并没有解决浏览器禁用图像后空白页面的可访问性问题；有时在IE 5.0中无法使用。

浏览器支持：Windows——IE 5.5及以上版本、Netscape 7、Opera 6及以上版本、FireFox；苹果机——IE 5.2、Safari、Firefox。

3. Levin的方案

Levin Alexander想出了一个绝妙的主意：不再将文本置于span中，而是将其从span中移出来，将文本和span一起放在父元素中，然后使用这个空白的span覆盖住文本，并将背景图像应用到该span之上。如果一切顺利的话，屏幕阅读器即可正常访问这段文本，且也充分考虑并解决了浏览器禁用图像后空白页面的可访问性问题。但新的问题也随之出现，那就是图像不能是透明的，否则用户将会看到下面的文本。且这种方案所需要的CSS极为冗长，让人难以理解。

• HTML标记

```
<h3 class="replace" id="myh1">And a dash of Thyme.<span>
</span></h1>
```

• CSS代码

```
.replace {
  position: relative;
  margin: 0px; padding: 0px;
  /* hide overflow:hidden from IE5/Mac \*/

  overflow: hidden;
  /* */
}
.replace span {
  display: block;
  position: absolute;
  top: 0px;
  left: 0px;
  z-index: 1; /*for Opera 5 and 6*/
}
#myh1, #myh1 span {
  height: 25px;
  width: 300px;
  background-image: url(thyme.png);
}
```

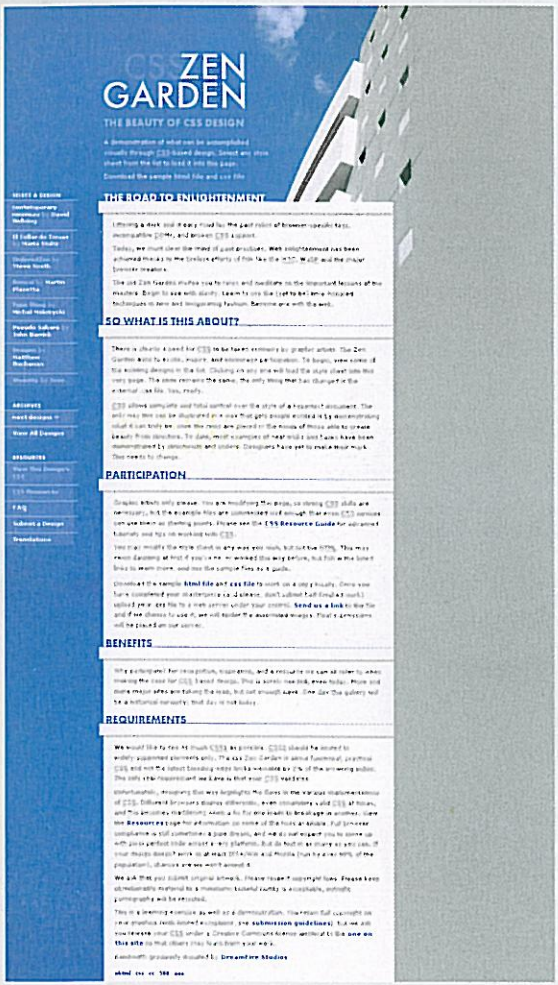
优点：屏幕阅读器可正常访问[1]；解决浏览器禁用图像后空白页面的可访问性问题。

缺点：无法使用透明图像；CSS代码较为冗长。

浏览器支持：Windows——IE 5.0及以上版本、Netscape 7、Opera 6及以上版本、FireFox；苹果机——IE 5.2、Safari、Firefox。

4.2.5 选择

应该使用哪种方法呢？这要自己选择。在大多数浏览器都支持前面提到的“CSS生成内容”的那一天之前，Leahy/Langridge、Rundle或Levin的方案仍然是最好的选择。若是图像替换技术没有出现，那么禅意花园中很多与Hellsing的Revolution！类似的作品也都不可能实现。在对未来的提交作品要求中，禅意花园也明确地提出了这一点，鼓励大家发现更具可访问性、兼容性的图像替代方案。



Marc Trudel，设计师

www.csszengarden.com/094

4.3 Deco

用最少的图像带来最大限度的视觉冲击

在开始创作之前，Marc Trudel总是希望他的设计过程能够像进行仪式一样有条不紊。Trudel首先访问了一些他欣赏的网站，用一个小时的时间欣赏他人的作品并收集内容素材，包括照片、图示和文本等。接下来他开始在Adobe Photoshop中创建原型。与其他图形设计师不同的是，Trudel没有将草稿打在纸上，而是直接选择了更易把握细节的Photoshop方式。

2004年初冬，Trudel却在弗罗里达州迈阿密闲逛时突然找到了灵感，他随即放弃了仪式一般的设计过程。Trudel钟情于南部海岸的装饰艺术建筑的特写效果，并拍摄了很多照片。Trudel更是对其中的一幅照片所表现出的简洁的线条赞不绝口，它也成为了禅意花园作品Deco的灵感来源。

4.3.1 用另一种方式去思考

对于那些曾用图像切片方式布局Web页面的“守旧”的设计师来说，在CSS中使用图像则是一种完全不同的思考方式。在图像切片的布局方式中（如果很幸运没有用过这种方式，那么也没有必要再了解它了），设计师通常需要首先在Photoshop中完成整幅图像，然后根据需要将其切成小片，最后在Web页面中用HTML表格将其再次组合起来。

这些表格组成了一个网格系统，且这种布局方式在20世纪90年代的Web设计中也处于支配地位（图4-3-1）。但不幸的是，这种阻碍进步、影响可访问性且影响网站功能的方法在今天仍未彻底消亡。

传统切片方式所带来的一个致命的限制却很少被提及，特别是在处理图像时，必须确保每张图像都恰好符合网格，也就是那一系列方框的大小。即使说这种布局方法彻底摧毁了Web作品的创造性也丝毫不为过。太多的设计师陷入到了方框设计的怪圈，甚至还陶醉于其中。

但CSS布局却将我们从这个限制中解放了出来。虽然所有CSS布局的实现仍旧基于盒模型，但我们仍可发现二者的显著差异。在CSS布局中，可以仍旧用网格的方式思考——但如果愿意的话，也可以从中彻底地跳出。通过巧妙的设计以及灵活的图像布局，设计师可以超越网格的限制，以更加流畅、开放的方式思考（图4-3-2）。

摆脱了表格的限制之后，设计师在创作时不再有束手束脚的感觉。这一转变所带来的最大的好处之一就是，我们能够以一种完全不同的方式使用图像，让作品远远好过从前。

严格、刻板且阻碍设计师发挥的网格布局年代已经过去。借助CSS的帮助，我们可以将图像以背景的形式应用到任何元素上，进而实现诸如图像分层、图像覆盖——甚至在某些浏览器中用PNG透明效果创建出一系列图片之间的交互效果等功能。

CSS不但让设计师变得更有创造力，同时也减轻了页面文件大小，让浏览者也倍感满意。

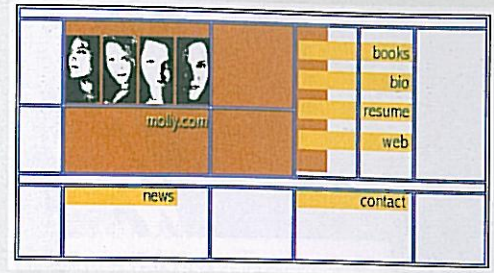


图4-3-1 基于表格的网格设计所带来的限制

图4-3-2 Deco的整体效果显得非常轻快、开阔。虽然在表格方式中也不是不可能实现，但在表格布局起支配地位的年代中，这样的创意却非常少见

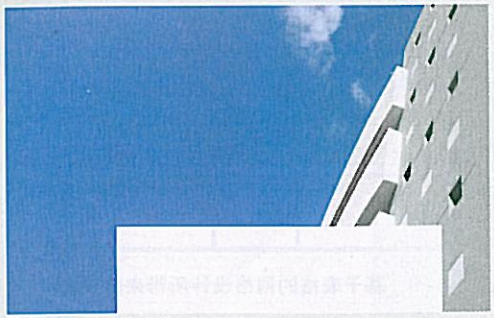


图4-3-3　Deco使用的主背景图

图4-3-4　作为标题图像的透明GIF

4.3.2　图像使用技巧

Deco除了在线条运用方面显得简洁、开阔之外，其用CSS应用图像的方法技巧也非常值得我们学习。无论是图像的大小、尺寸，还是图像的位置等相关选项，CSS布局方式中用到的图像都和传统表格布局中的作法大不相同。

1. 图像的大小和尺寸

在保证足够质量的前提下，图像文件的大小应该尽可能地降低。表格布局中的图像切片技术就是为了减少页面文件的总大小而产生的。

但CSS却允许更精确地跟踪图像的使用方式及使用位置，进而根据特定需要作出最好的选择和优化。有些情况下，我们将不得不选择一幅较大的图像，而有些时候只要用一张，或一系列较小的图片填充到某个较大区域中即可。

让我们看一下Deco作品中最主要的一张背景图像，Trudel拍下来的这张照片中包含了天空中的云以及该建筑的装饰线条。这张图像让作品显得轻快开阔，营造出了良好的意境，但同时其尺寸也相当让人吃惊（图4-3-3）。

这张图像宽788像素，高497像素。在传统的表格布局中，若设计师希望实现类似的效果，那么他一定会将该图像切分为若干小片。

该图像的大小为45.02KB，若是以表格布局的角度考虑，似乎有些太大了。但请记住，CSS布局中的图像使用方式更加明确专一，若页面中并没有太多其他图像，且这些图像大小也很合理的话，那么这样的背景图片大小则完全可以接受。接下来看看标题图像，该图像只有3.28KB，对标题而言其尺寸也很典型——266像素宽，103像素高（图4-3-4）。

2. 图像数量

同样一个作品，用CSS设计所使用图像的数量并不一定会比表格布局的少，但用CSS却能在使用更少图像的同时得到更好的效果。表4-3-1列举了Deco中用到的图像及其大小和尺寸。

表4-3-1 Deco中用到的图像

图像文件	尺寸（单位：像素）	大小	用途
bg.jpg	788 × 497	45.02 KB	主背景图
bg_blue1.gif	2000 × 1	0.14 KB	背景图
csszengarden.gif	266 × 1033	3.28 KB[1]	作品标题
thebeauty.gif	263 × 13	1.69 KB	作品副标题
h1.gif	537 × 39	3.02 KB	第一级标题
h2.gif	537 × 50	2.83 KB	第二级标题
h3.gif	537 × 50	2.46 KB	第三级标题
h4.gif	537 × 50	2.20 KB	第四级标题
h5.gif	537 × 50	2.52 KB	第五级标题
lightblue.gif	125 × 3	0.06 KB	列表背景
selectdesign.gif	95 × 7	0.37 KB	作品选择列表标题
archives.gif	95 × 7	0.28 KB	作品存档列表标题
resources.gif	95 × 7	0.31 KB	资源列表标题

那么将如何减少图像的数目，进而降低页面文件总大小呢？本章的主要目的之一就是让我们在使用图像时能够充分展开想象，思考从何处开始着手。下面就列出了一些能够有效减少图像使用的方法。

- 尽可能地使用文本样式代替图像。在Deco的页首部分我们可以看到，作品的主次标题均用图像得到期望的字体，但其下方quickSummary一节中的文字却并非用图像实现。合理地把握图像和文本之间的平衡将会得到立竿见影的效果。
- 只保留图像中为实现预期效果所必需的部分。在使用照片作为背景图或页面标题时，这一条所带来的好处显而易见——丝毫不影响图像质量的同时又降低了文件的大小。

[1] 原文为28KB，疑为笔误。——译者注

Note

当然，使用当代最新的图像压缩技术能够更进一步地降低图像大小。在Deco中，我们完全可以相信，在不损失过多图像质量的前提下，主背景图仍有相当大的压缩空间。尽管如此，由于该图片在作品中至关重要且支配了非常大的页面空间，设计师仍有足够的理由选择使用较大的文件，以期保证足够的质量，带来更好的视觉效果。

- 使用细条图像填充背景。bg_blue1.gif用来作为Deco中主背景之外的空白部分的背景，但其高度只有1个像素，以至于其文件大小甚至可以忽略不计——只有140字节。

- 尽可能地重复使用图像。在CSS中，我们可以将同一幅图像应用到不同的元素上。在Deco中，lightblue.gif就同时成为了三个截然不同元素的背景图像。借助CSS的后代选择器，我们只需书写一次规则即可，之后linkList中所有的ul元素都会自动应用该图像：

```
#linkList ul {
  margin: 0px;
  padding: 0px;
  background-image: url(lightblue.gif);
  background-repeat: repeat-y;
}
```

- 合理地选用图像或颜色。若CSS中的color或background-color属性也能实现设计效果，则你应该毫不犹豫地使用这些属性，而不是背景图像。

最后作为补充，若能不使用图像，那么就不要使用。图形设计中最大的诱惑就是为作品添加过多的非必要图像。每添加一幅图像之前都要先问问自己，这幅图像在作品中真的有其明确的目的，会起到预期的作用且能带来美感吗？如果并不能完全确定，那么就考虑将其移除。

3. 图像的分层

CSS所提供的另一个从前难以想象的功能就是以分层的形式显示不同的图像，并借此创建出惊人的视觉效果。现今看来，这种效果的确不难实现，但在从前的表格布局作品中却很难做到。

在从前的表格布局作品中，当某个透明GIF位于body或table中，且body或table应用了背景图像或背景颜色时，我们也能看到这样的效果。例如，如下的HTML代码就在body标签中指定了整个页面的背景颜色和背景图片：

```
<body bgcolor="#3366CC" background="images/bg.jpg">
```

接下来，那个透明的GIF图像将被置于body中的某处——最有可能在某张表格的单元格中：

```
<td><img src="images/csszengarden.gif" width="266 "
height ="103"alt="CSS Zen Garden"></td>
```

结果将类似于Deco中的效果（图4-3-5）。

但Trudel在Deco中的做法却是将主背景图像应用到#container元素上：

```
#container {
  background: url(bg.jpg) no-repeat scroll left top;
}
```

然后将该透明GIF图像应用在#pageHeader h1选择器中：

```
#pageHeader h1 {
  background: url(csszengarden.gif) no-repeat left top;
  width: 620px;
  height: 103px;
}
```

这样即完成了漂亮的页首部分。虽然与图4-3-5所示的图像与背景颜色混合达到的效果一样，但二者的实现方法却截然不同。

图4-3-5　用表格布局将透明GIF图像置于背景之上

4.3.3　最大化视觉冲击

Deco仅仅使用了少量的图像就实现了令人如此羡慕的效果。只要明智地选择图像放置的位置，仔细平衡图像和文本效果的使用，加上简单有效地借助图像或文本处理作品色彩，我们也能够创造出同样精彩的作品——不仅从技术实现角度上看足够优雅，作品本身也会变得超凡脱俗，给浏览者带来强大的视觉冲击。

Michal Mokrzycki，设计师

www.csszengarden.com/097

4.4 No Frontiers!

用圆角图像打破方框的束缚

CSS设计中占支配地位的趋势就是居中、固定宽度的布局加上阴影效果——但Michal Mokrzycki却希望在其作品No Frontiers!中打破这一惯例。Mokrzycki决定不限制他的三栏布局的宽度，让其可以随浏览器窗口大小的变化而自动调整。作品将尽可能地简洁，但仍足够优雅、独特。

与其他严格遵守设计流程的设计师不同，Mokrzycki在设计时更喜欢即兴创作。他首先在自己的照片库里找到了合适的素材，然后不停切换于CSS和Adobe Photoshop之间，以实现他的灵感。在这个过程中，Mokrzycki发现了CSS的强大之处就在于它能够让设计师完全无拘无束地进行创作，这也正是该作品被命名为No Frontiers!的原因。

4.4.1 方框问题

Web页面在历史上一直看起来如盒子一般四四方方，这并不是个意外。但具有讽刺意味的是有些人却批评CSS布局是引发这一问题的罪魁祸首！早期基于表格布局的作品均大量使用了边框和内边距，这也使此类作品看起来似乎是由一个个方框组成，页面元素之间也缺少了联系（图4-4-1）。

那时大多数个人计算机的屏幕分辨率只有640像素×480像素——与今天的大屏幕比起来显得非常小，自然空间也变得极为珍贵。加上浏览器方框界面和计算机屏幕方框外形带来的暗示，我们只能继续在方框中不停地嵌入另外的方框。

从视觉角度考虑，将信息埋没在一个又一个的方框中并不能使浏览者轻松找到。当然，那是一个随心所欲的时代——Web刚刚出现不久，无论是设计师还是用户都被其深深吸引，自然也就不会对界面样式太过于挑剔。

因此，当CSS这种新的、专注于视觉表现的布局方法出现时，大多数人都在考虑如何将他们从前熟悉的表格布局改为用CSS实现。人们对分栏布局的关注，以及随后为这类布局添加的一系列修饰（例如边框）均很自然地导致早期的CSS作品被人诟病为滥用方框（图4-4-2）。

说CSS对方框问题负有主要责任显得有失偏颇。随着我们越来越深入地了解CSS，包括聪明地使用图像（特别是在背景图像使用上），用层将图形、图案等装饰元素组合起来等，CSS布局模型能够实现的效果已经远远超过了从前表格布局的效果。

No Frontiers!就通过圆角形状的装饰替代了方框那单调的边缘，整个作品给人的感觉也显得更加柔和、统一。

4.4.2 实现曲线效果

No Frontiers!中演示了很多为作品添加曲线效果的方法，让作品摆脱“四四方方”的样子。这些方法都需要这样或那样的背景图像支持。

1. 简单曲线：单一背景图像实现

最简单的为作品添加曲线效果的方法就是创建一张带有曲线且内容部分空白的图像。然后用CSS将其设置为某元素的背景即可。

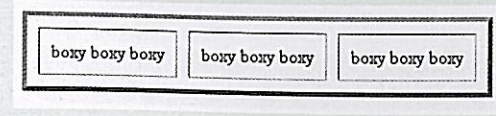


图4-4-1 早期的Web页面中经常滥用方框

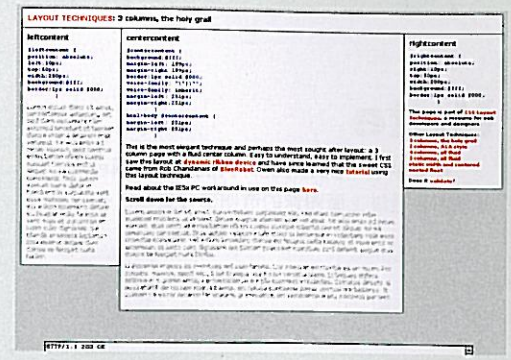


图4-4-2 早期的CSS布局：分栏布局的确能够带来四四方方的感觉。但考虑到当时的特殊情况，此类布局仍流行了起来

Note

幸运的是，随着Web设计技术的日趋完善，设计师们开始使用切片技术在表格布局的页面中添加更多的图形。一段时间之后，这种方法开始走向成熟且逐渐被大家接受，为Web设计带来了一丝新意。但在很大程度上，方框问题始终是Web设计中的一片挥之不去的阴影。

The Road to Enlightment

图4-4-3　No Frontiers!中的一幅圆角标题图像。注意右面的透明效果

Note

图像的透明设定将允许其他的背景颜色或背景图片透过上面图像的透明部分显示出来。

图4-4-4　圆角标题图像看上去位于条纹背景图像之上

No Frontiers!中演示了这种方法——页首部分“The Road to Enlightenment”的右边缘呈现出了漂亮的圆角效果（图4-4-3）。

该圆角图像就是以背景的方式应用于#preamble中的h3元素上的：

```
#preamble h3 {
    background: transparent url(enlightment.gif) no-repeat top left;
    width: 291px;
    height: 37px;
    margin-bottom: 10px;
    padding: 0;
}
```

但若是想要在已经有背景的元素上再应用这样一幅带有曲线效果的图像，又该怎么做呢？

2. 在多个元素上应用背景

用CSS在多个元素上应用背景图像将会让我们受益匪浅。通过合理地将背景图像应用到若干元素上，在实现最终曲线效果的同时，图像文件大小也会有所降低。

在No Frontiers!中的“CSS Zen Garden”标题处可以看到该技术的实际应用。“CSS Zen Garden”标题的右下角带有圆角效果，且圆角下方还依稀露出了一部分条纹背景图像（图4-4-4）。

为了实现这样的效果，Mokrzycki首先将条纹背景图像（一幅细长的小图像，见图4-4-5）应用于body元素之上：

```
body {
    font: small "Trebuchet MS", Verdana, Arial, Helvetica, sans-serif;
    color: #676767;
    margin: 0px;
    padding: 0px;
    background: #fbfbf2 url(page_back.gif) repeat-x fixed;
}
```

然后，将圆角图像应用于容器元素之上：

```
#container {
   padding: 0px;
   margin: 0px;
   background: transparent url(body_back.png) no-repeat
   top left;
}
```

从概念上讲，此时背景图像将分层显示，既实现了圆角效果，又在其下显示了不同的图案。在作品的右上角处也能看到该技术的实际应用：链接列表的头部的圆角效果同样呈现于条纹背景之上（图4-4-6）。

3. 修饰内容区域的边角

通过使用这种层的概念，我们也可以将多个圆角甚至其他边缘修饰效果应用到内容区域上。用这种方法实现圆角效果的挑战在于，它必须依赖多个不同的HTML元素。

在No Frontiers!的#pageHeader节和#quickSummary节中可以看到该技术的实际应用。最终效果很漂亮——作品中该文字区域的右上角和右下角都以圆角方式显示（图4-4-7）。

若是没有了CSS，那么实现这种效果则需要一张单独的图像，或者用表格将若干张图像组合到一起。但在CSS中，我们完全可以仅仅为圆角部分创建图像，并将其应用到文本区域，这样在保证必要效果的同时也降低了图像文件的大小。

Mokrzycki为此创建了两幅图像，分别用于该区域的顶部（图4-4-8）和底部（图4-4-9）。

随后将这两幅图像分别应用于#pageheader和#pageheader h2元素上：

```
#pageHeader {
   position: absolute;
   left: 0px;
   top: 190px;
   width: 291px;
   height: 159px;
   background: transparent url(green_top.gif) no-repeat
   top left;
   z-index: 3;
}
```

图4-4-5　该细长的背景图像应用于body元素上，并在x方向上重复显示

图4-4-6　另一个分层背景图像的实例

图4-4-7　包含文本内容的多个元素组成的背景

Note

一些更复杂的相关技术，例如Douglas Bowman的“Sliding Doors of CSS”（www.alistapart.com/articles/slidingdoors和www.alistapart.com/articles/slidingdoors2）等都依赖于分层背景的概念。

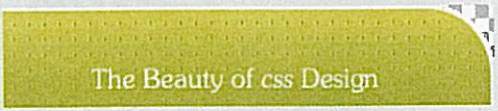


图4-4-8　实现该效果的顶部图像，图像自身包含文字

图4-4-9　实现该效果的顶部图像。注意，在这两张图像中灰白相间的格子图案表示图像的透明部分

```
#pageHeader h2 {
    position: absolute;
    top: 118px;
    background: transparent url(green_bottom.gif)
      ➥no-repeat bottom left;
    width: 291px;
    height: 135px;
    margin: 0;
    padding: 0;
}
```

但#pageHeader h1和#quickSummary这两个元素却并不包含任何背景图像定义。实际上，它们均应用了一些定位技术。h1元素用来实现背景颜色，其中的HTML文本内容也经过了一定样式的修饰：

```
#pageHeader h1 {
    position: absolute;
    top: 59px;
    background-color: #bbbf58;
    border-right: 2px solid #fff;
    width: 289px;
    height: 150px;
    margin: 0;
    padding: 0;
}

#quickSummary {
    position: absolute;
    top: 270px;
    left: 20px;
    width: 245px;
    line-height: 2;
    z-index: 3;
}

#quickSummary p {
    color: #fff;
    font-size: 11px;
    line-height: 1.5;
```

注意到上述代码中固定了各节的高度。这样即可避免该区域大小变化对圆角效果的影响。

4. 自定义列表项目的指示符号

这个技术同样可以将图像应用于元素的背景上——在这里为li元素。这些圆型、基于图像的列表项目指示符号能让列表内容更好地融入到No Frontiers!作品的整体意境中（图4-4-10）。仔细体会其融入作品主题的方式，我们也能够用自己的图像创建出类似的效果。

为了能在No Frontiers!中自定义所有项目列表指示符号，Mokrzycki首先创建了三个小图像。第一个使用淡绿色，用于主链接列表中。第二个使用淡蓝色。用于作品存档链接列表中。第三个使用浅红棕色，用于资源链接列表中。

然后，将这三个指示符号图像相应地应用到链接列表中各元素的背景上：

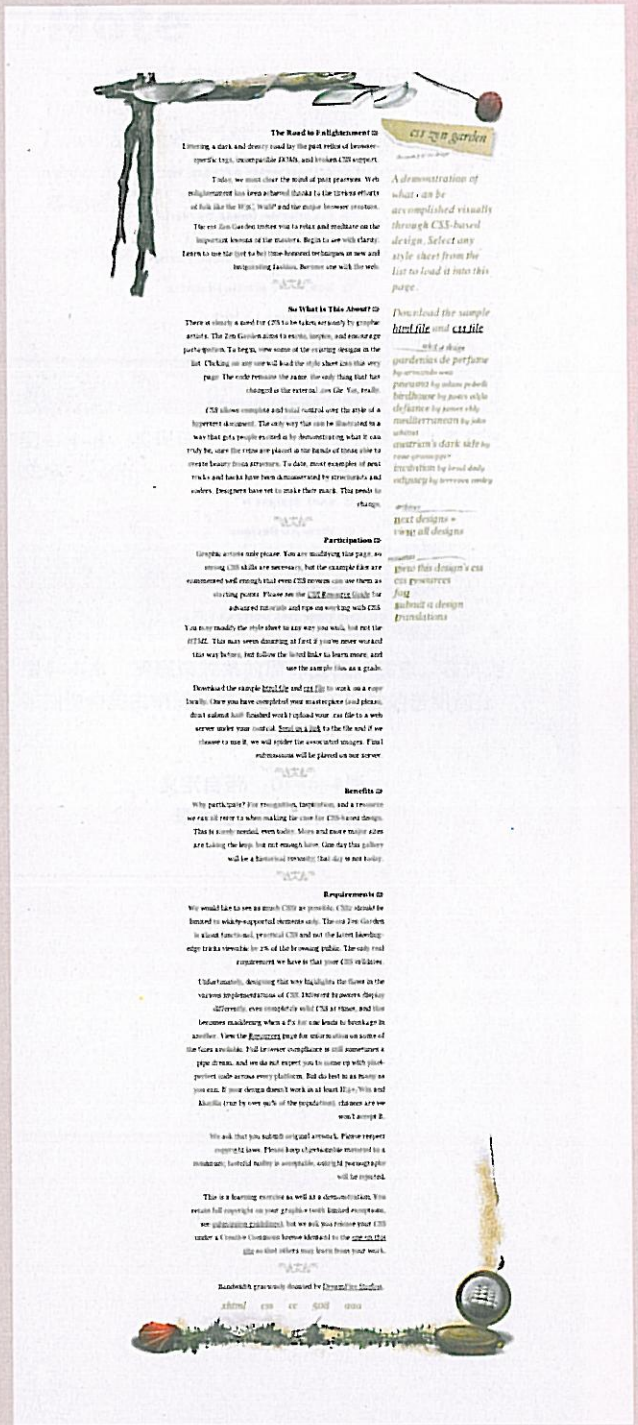
```
#lselect li {
   background: transparent url(link_dot.gif) 0 7px no-
   repeat;
}

#larchives li {
   background: transparent url(link_dot2.gif) 0 7px no-
   repeat;
}

#lresources li {
   background: transparent url(link_dot3.gif) 0 7px no-
   repeat;
}
```

使用不同颜色或不同样式的指示符号图像能够丰富作品的细节，且给用户一定的暗示，表明列表之间和列表中元素之间上下文关系。

图4-4-10　带自定义项目符号而链接列表

4.4.3　多姿多彩的图像

无论想通过圆角还是别的什么边缘效果打破那沿用已久的方框布局，恰当的图像应用技术总是能够让作品充满独到的新意。适当地用图像进行修饰将让整个作品的视觉效果变得截然不同，正如我们在No Frontiers!中所看到的一样。



Dave Shea，设计师

www.csszengarden.com/013

4.5 Coastal Breeze

为作品创建图像，并寻找额外的素材

被本地居民和著名作家Douglas Coupland称为“玻璃之城”的温哥华是一座繁荣的城市，坐落于加拿大西海岸的一片高耸入云的群山脚下。几根浮木，几片贝壳，加上一些玻璃碎片等海洋相关的作品主题让城市的形象呼之欲出，若是再来上几片寿司那就再好不过了。但设计师Dave Shea却更懂得在适当的时候戛然而止。

Coastal Breeze中的图像完全是手工在Adobe Photoshop中绘制完成的，没有用到任何照片素材。借助于Photoshop提供的各种绘画工具，Shea未使用任何摄影照片就实现了作品中栩栩如生、富有材质感的效果。

4.5.1 从这里开始

Coastal Breeze中的各个元素最初是为别的站点创建的，该站点在禅意花园发布之前几年就已经上线，但现在已经关闭。原Photoshop图像创建于2001年初（图4-5-1），其中使用了超过300个图层。Shea找出了这个文件，把其中需要的元素提取出来并布置在新作品中。调整图像相对来说较为简单，难点在于如何创建出能够包含住页面文本的布局。

原作品中这些元素构成了一个大致呈长方形的框架，并用框架中包含的HTML frame显示实际内容，而禅意花园的内容长度又让我们不得不使用垂直滚动的布局。这种种问题在开始新作品之前都要经过仔细斟酌。

Shea从一个方形的框架开始设计。将该框架应用到Coastal Breeze中的最干净的方法似乎就是将其劈成上下两部分。两个L字母型的对角既有力地让Coastal Breeze体现出框架的感觉，又给作品留出了必要空白。

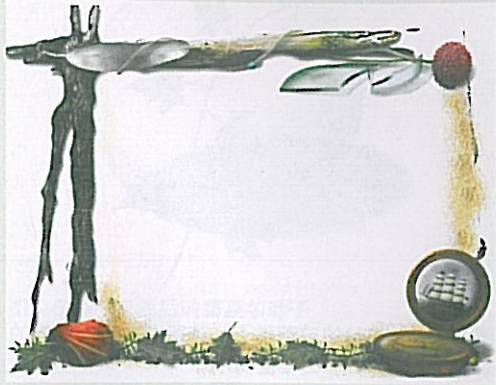


图4-5-1　作为Coastal Breeze起源的原Photoshop图像

图4-5-2　为玻璃碎片创建透明的阴影

4.5.2 绘图技巧

当然，并不是每个人都喜欢手工绘制整个作品的，但很多Coastal Breeze中用到的绘图技巧在其他情况中也同样适用，值得我们学习——正所谓胸中有丘壑，下笔如有神。注意下面的一些技巧适用于Adobe Photoshop。因为图形处理工具大都提供了类似的工具，所以即使不用Photoshop，也能用其他软件实现同样的效果。

1. 使用画笔和变换工具

右上角的玻璃碎片投射的阴影加上其透明效果，给人感觉如同阳光正透过它们一样（图4-5-2）。在Photoshop中实现这样的效果要比仅仅简单添加阴影图层复杂些，但也不会复杂太多。要想实现漂亮的阴影效果，首先应从复制一个新的前景图层（选择Layer > Duplicate Layer）开始，这里我们用一个玻璃弹子做示范。然后将该复制图层置于原图层下方，并填入单一颜色。借助于自由变换工具（Edit > Free Transform）所提供的放大、缩小、旋转和扭曲等功能，我们可以将该图层垂直压缩并向左扭曲，然后柔和一下其过于分明的边缘（图4-5-3）。

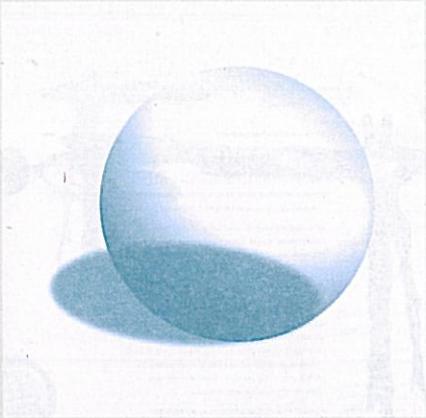


图4-5-3　简单的玻璃弹子及其实心阴影。倾斜的阴影暗示物体处于三维空间中

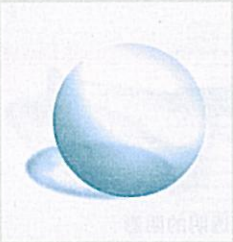


图4-5-4　使用橡皮擦和涂抹工具让玻璃弹子的部分阴影变得透明

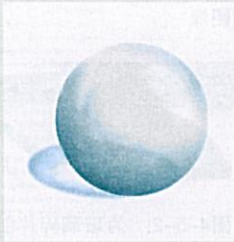


图4-5-5　完成后的玻璃弹子，添加了一些瑕疵用于表现真实感

图4-5-6　简单的叶子形状

图4-5-7　叶子上添加了高亮和阴影效果，用来模拟叶脉和材质

让阴影变得透明则需要使用橡皮擦工具和涂抹工具，二者的大小、力度均可设置。我们可以用较大且边缘柔和的橡皮擦手工擦去阴影的中部颜色，然后用较小且柔和的涂抹工具让剩下的部分变得不完整，模拟出实际阴影的样子（图4-5-4）。理想化的透明球体效果看起来固然美观，但现实中存在的物体却很少能如此完美无瑕。若不继续添加一些细节，则无法体现出物体的真实感。最后的效果是通过在玻璃弹子上添加一些微痕实现的（图4-5-5）。

2. 使用减淡和加深工具

虽然在后期修饰过程中完全可以抛弃加深和减淡工具，但我想我们不会这样做的。借助于加深和减淡工具，我们即可为复杂图像相应地添加极为逼真的阴影和高亮效果。减淡和加深工具类似于橡皮擦和涂抹工具，甚至它们的设定选项都差不多。从简单的单一颜色图形以及其下的图层（图4-5-6）开始，我们即可用减淡工具在叶子上画出浅色的叶脉。不同的力度设置也会给叶脉添加阴影的感觉，呈现出轻微的立体感。

即使有了网格状的叶脉，完全平整的叶子看起来仍显得不够真实。因此，我们接下来需要用加深工具在叶子表面添加更多的阴影效果，以实现立体的三维效果（图4-5-7）。

目前提到的这几种工具只不过是用于绘出一片叶子。随后，让我们将这片叶子复制若干次，调整每个副本的尺寸并适当旋转，最终组成一条蔓延的藤蔓的形状。在绘出藤蔓本身并对其进一步的高亮、阴影修饰之后，我们终于得到了魔术般的效果（图4-5-8）。

图4-5-8中很容易发现，随着叶子逐渐变小，上面的叶脉也变得渐渐模糊起来——图像尺寸变小之后，细节自然也将变得难以分辨。但需要注意的是无论如何模糊，肉眼仍旧能够隐约分辨出一些，因此无论如何，精益求精的细节总归都会将让整个作品更加具有说服力。缩小被精雕细琢过的作品对设计师来讲也是个不小的挑战，因为浏览者都会自然地希望它能够大一些，以便于仔细欣赏。

4.5.3 创建图像

虽然不是所有人都有那么多时间花费在手工绘制图像上，但像Coastal Breeze这样精美的作品仍让人觉得这些时间花费得物有所值，某些时候手工绘制也不失为一个好方法。

基于现有的图形作品进行图像创建或许是最直接的办法。在整站设计中，一般很少有人选择手工绘出所有页面，很多设计师都会先在纸上画出作品的草稿，并想好布局中将遇到的各种问题之后才打开计算机。日常积累的图画、照片等其他创造性的作品将会在我们厌倦现有资源或灵感枯竭时有效地起到启发作用。

虽说术业有专攻，但目前很多掌握技能各不相同的人们都在进行着Web设计。即使不懂绘画，仍可以借助现有的资源完成自己的作品。由于技术不断发展（技术也在不断贬值），知道如何使用哪怕是很便宜的扫描仪和数码相机也足够生成高质量的Web设计素材。虽然这些产品都提供了详尽的说明书，但若没有任何基础的话，仍无法避免开始时相当长的一段学习时间。

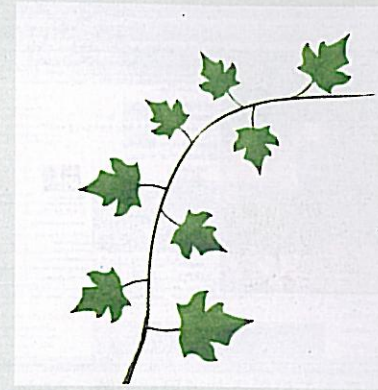


图4-5-8　完成后的藤蔓和叶子

4.5.4 获取图像

虽然有很多客户希望我们能够精心雕琢作品中的图像，要是能委托专门的设计团队制作那就更好了，但实际上，大多数设计师给出的预算都没有包含聘请专业设计团队的开销。

1. 委托设计和免专利使用费（royalty-free）资源

虽然良好的人际关系网络可能会帮你找到中意的摄影师或图像设计师，但有时我们仍免不了要在本地电话黄页中逐一搜寻。但也没必要将眼光局限在本地——很多现有的网站都为该领域专家为我们提供了互相发现和彼此交流的机会。例如Aquent（www.aquent.com）就是其中的佼佼者，其他同类网站也有它们自己的独到之处。

图4-5-9　iStockphoto鼓励艺术家们彼此分享作品

委托设计作品通常较为昂贵，而很多项目却并没有这么多的预算。因此我们对免专利使用费图像更加感兴趣。根据其声明条款，免专利使用费的含义就是只要一次购买，即可终身使用该资源——无论用于何处或重复使用多少次，都无需再额外付款。除了转手之外，大多数免专利使用费协议都允许随意使用购买的图像。这也就意味着购买之后你就可以在多个项目中重复使用。唯一的风险就在于别人可能也购买了同样的图像，所以很有可能你会发现有一天你的最新杰作“职业商务女性”与某人的“理疗师”是同一个人——这种情况真的发生过。

如果预算仍承受不起免专利使用费图像，那么也有一些网站提供免费的素材。为了让那些刚出道的摄影师和图像设计师有交流的平台，类似iStockphoto（www.istockphoto.com，图4-5-9）和stock.xchng（www.sxc.hu）之类的社区均提供了分类良好的索引，其中包含了大量的中高解析度的照片和图像，非常适合应用于Web页面中。只要花费极低的价钱即可得到这些图像，有些甚至可以免费使用，无论是否收费，都非常物超所值。

2. 其他可选择的资源

若将视野放宽，将找到更多的免专利使用费资源。因为通常来讲很多版权法律条款均有一定的时限（时限长度取决于所在的国家），所以很多失去版权保护的作品极有可能尚未过时。例如在美国，2004年年中时所有发布于1923年前的作品均失去了版权保护，且发布于1923~1963年间的作品也不能再延长其版权保护时间。

不受版权限制的作品可以被用于任何目的，无论商业还是个人，所以若想使用另一时代的作品，这将会是个非常值得考虑的方案。寻找不受版权限制的作品非常简单，只要在喜欢的搜索引擎中敲入“public domain”即可，图书馆和一些艺术品商店也大都提供了不受版权限制的作品目录供参考。某些图书馆中甚至还有专门有其馆藏的不受版权限制作品的目录。

但我们并不总是要上溯到如此久远的年代，当代也有很多不受版权限制的作品。创作共用组织（http://creativecommons.org）就是为了“扩大创造性作品的使用范围，让人能够基于前人的作品进一步创造并继续分享”（图4-5-10）而成立的。由于该站点在2004年以来的主要目标就是鼓励艺术家们分享他们的作品，所以我们能够容易地找到很多基于该许可协议发布的作品。但在使用这些作品时，不要忘记创作共用许可协议也有很多不同的授权等级——并不是所有作品都可以随意使用的。

3. 版权相关的注意事项

若某件作品不能使用，但在设计中却必须要用到，那么首先要做的就是向作者（很多情况下也可能是发布者）请求使用权。可能发生的最坏的结果就是他们不允许。要确保授权信息以书面形式来自合法的版权所有者，然后才能在授权范围内使用该作品。

同样还要注意的是，版权法律非常复杂且经常变化。无论作者是否显式声明了版权信息，创造性作品一问世就自动受到版权法律的保护。

要确保充分理解了法律的条款。若不是绝对地确信可以使用某件作品，那么或者去获取明确的许可，或者使用其他的替代品。未经许可就使用受版权保护的作品是违法的，将会因此被起诉。千万不要存在任何侥幸心理。

图4-5-10　创作共用等站点提供了不受版权限制的作品

Tip

在http://creativecommons.org/getcontent中可以搜索到创作共用作品。

4.5.5　结论：关于手工绘图

所谓勤能补拙，即使没有绝佳的天赋，只要肯花费时间，一样能够得到相当好的结果，Coastal Breeze就是一个有力的证据。为了完成这件作品，Shea估算出他在Photoshop中至少花费了80个小时。

寄希望于客户给你如此宽裕的时间无异于痴人说梦。在商业项目中，客户大多都希望你能从照片中取材，而不是等待类似Coastal Breeze这样漫长的手工绘制过程。若选用了恰当的资源，使用照片素材在保证预算的前提下同样能够营造出相似的意境。

Coastal Breeze作品看起来确实与众不同，但在现实中，项目中能做到什么，不能做到什么却通常是由预算决定的。除了一些细节修饰之外，Coastal Breeze看起来近乎完美。

Tip

一些有用的区域性版权资源[1]：美国版权局网站（www.copyright.gov）；加拿大著作权法网站（http://laws.justice.gc.ca/en/C-42）；英国专利局网站的版权信息页面（www.patent.gov.uk/copy）。

[1] 中华人民共和国国家版权局网站：http://www.ncac.gov.cn/。——译者注

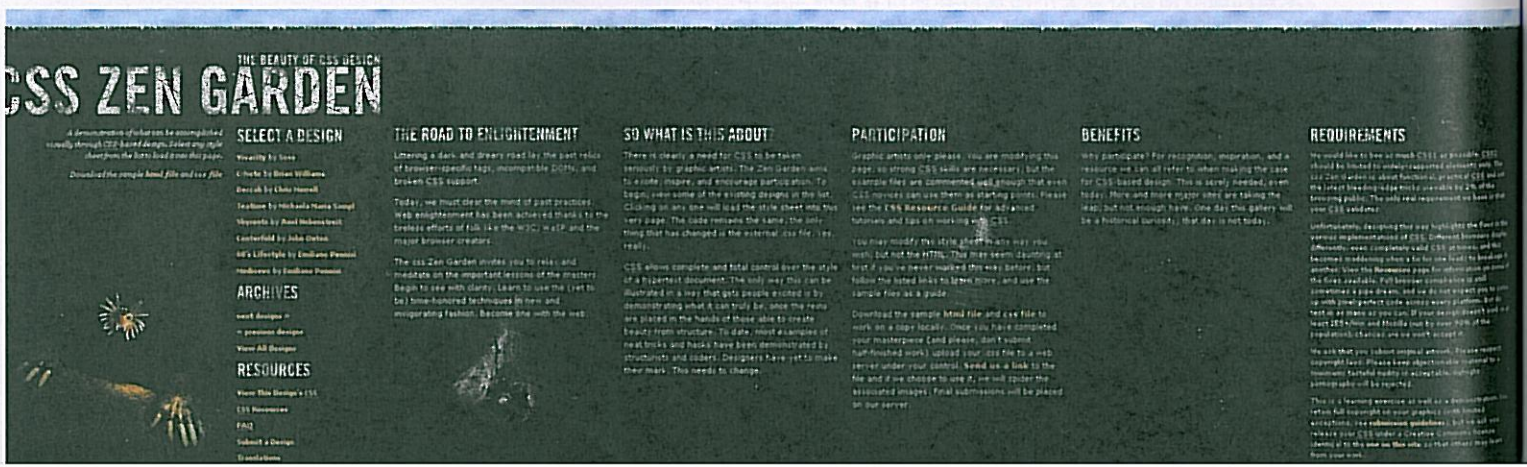
Michael Pick，设计师
www.csszengarden.com/019

4.6 What Lies Beneath

在二维平面中表现出三维的意境

头脑游戏和图像引发的联想让Michael Pick产生了该作品的灵感。Pick最开始考虑用纽约城中的公共花园作为该作品的素材，并拍摄了很多数码照片。但当他在Photoshop中处理这些图像时，却突然意识到了作品与蚁巢的某些共同之处。Pick随即放弃了创建普通的“地上”花园的想法，转而去描绘地下的场景。

Pick从一开始就在考虑能在地下发现的其他东西。灵感喷涌而至，最终完成了幽灵般奇异而居然同样美丽的作品——What Lies Beneath。



4.6.1 创造环境

大多数Web页面都主要关注于如何用足够可用且吸引人的方式传递信息。人们都要求设计师的作品有足够的创意，所以不得不承认Web站点最主要的需求就是按照惯例将图像和文字以二维的方式合理美观地组织起来。

当然也存在着一些例外：在Web发展的早期出现了一种名为VRML（Virtual Reality Modeling Language，虚拟实境标记语言）的技术。通过在模拟的环境中给出指定的坐标，开发者能够创建出可应用于Web页面中的三维模型。在安装了VRML插件之后，访问者也可以在Web页面中浏览这些模型。

之后不长时间，Apple公司也引入了QuickTime VR技术，设计师可以使用该技术将连续的全景图拼接起来。与VRML类似，QuickTime VR同样需要浏览者安装特定的插件。之后访问者即可在Web页面中看到这个模拟的全景环境。

当然，在得到浏览器支持或通过安装插件之后，三维电影或游戏（例如Flash、Shockwave以及其他格式的动画视频）同样也可应用于Web页面中。

偶然间，我们幸运地发现了Web媒介本身的潜力——无需任何虚拟现实工具或是特定的软件，即可在Web页面中表现出深度！无需再把想象禁锢在二维的平面国度中，我们自己——正如Pick做的那样——也能够在页面中表现出深度信息，且营造出相当真实的三维环境。

环境模拟所涉及的研究面非常广，包括复杂的理论科学等。若想开始学习这方面的相关知识，那么可以从计算机游戏开发图书开始入手。

计算机游戏的真实度不断提高，我们也越来越惊讶于其能达到的视觉效果。因此，学习游戏设计原理将显然给我们带来很多启发。

Note

VRML并没有消亡。实际上，该技术仍被研究并应用于某些场景中。www.openvrml.org就是开源版本VRML——OpenVRML的网站。

Note

更多关于Apple公司QuickTime VR的内容可在此访问：www.apple.com/quicktime/technologies/qtvr。

Note

Web3D协会（Web3D Consortium，www.web3d.org）是一个开放标准项目，致力于为Web平台研究并制定三维标准。

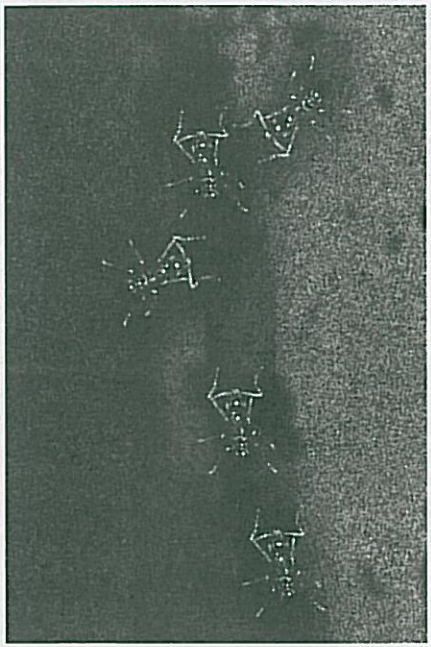


图4-6-1　将蚂蚁引入到What Lies Beneath中（为清晰起见，图像已被调亮）

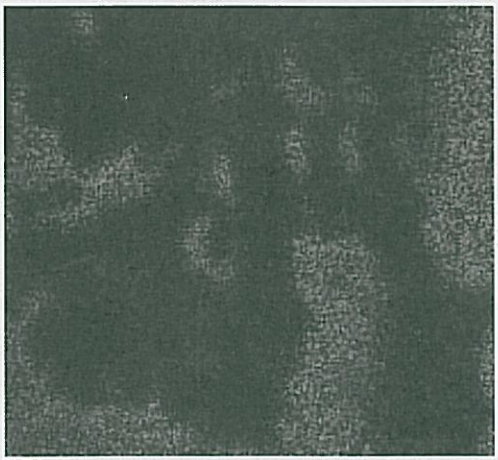


图4-6-2　边缘并不是线条，而是通过光影、线条以及空间的相互作用得到的（为清晰起见，图像已被调亮）

1. 理解细节

What Lies Beneath丰富的细节给作品带来了惊人的视觉效果。在明暗的阴影中，我们能够发现很多隐藏于地表以下的物体，有些是我们从前没有想到会出现在这里的。

设计师的主要职责之一就是仔细观察生活中的细节，以便在作品中充分地重现环境。或者像What Lies Beneath那样，为欣赏者营造出特定的意境。

这就意味着必须学会全面理解细节，而不仅仅停留在能够看到的水平上。例如，当我们走入一间房间时，立刻就会注意到其中最明显的陈设：墙壁、窗子（或许没有窗子）和家具等。

但这些东西的质地又是怎样的呢？木板上是否有裂纹？油漆是否已经斑驳？墙面的石灰是否有些脱落？玻璃窗是否有损坏或好久没有擦拭过？类似地，你是否注意到光影效果呢？房间的整体感觉又是如何呢？沉重？轻柔？开阔？还是狭小？所有这些问题的答案都不是那么显而易见，都需要我们的仔细观察才能够回答。但若你想让作品有足够的真实感，那么这些细节却都是必不可少的。

在What Lies Beneath中，细节无处不在，例如最初启发设计师应用当前风格的蚂蚁等（图4-6-1）。

2. 表现真实感

有意思的是，创建真实感不一定必须用照片作为素材。What Lies Beneath中用到的很多图像都不是来自于实拍照片，但仍旧表现出了非常强烈的真实感。

这些图片能够唤起人们对真实场景的想象。若要创建出能让人联想，而不是简单写实风格的图像，那么可以参考如下一些建议。

- 要特别注意图形边缘的效果。在学习绘画轮廓时，美术专业的学生通常都倾向于使用较为硬性的方式，例如用线条表示某人胳膊的轮廓。但这些学生很快就会知道，轮廓并不是这么简单的。事实上，边缘来自于线条、光影、正负空间之间的交汇融合，而正是这种融合才让我们体会到线条的感觉（图4-6-2）。

- 注意形状。看看四周找到一个物体——随便什么物体都行，然后仔细观察并说出它的形状。大多数物体都是由若干种形状组合而来的，这也正是区别于其他物体的关键之处。我们并不能简单地说某个茶壶是圆形的。组成茶壶的形状当然有圆形和弧形，但不同茶壶的外形设计千差万别，某些茶壶很有可能还包含其他一些独特的形状。

- 图像的材质能够为物体带来真实感。为了创建What Lies Beneath中的泥土效果，Pick特地去拍摄了纽约休斯顿大街上的一个交通灯控制台。有趣的是，这个与主题毫不相关的控制台材质却带给了人们极为真实的泥土效果（图4-6-3）。

- 深度的确可以表现出额外维度的感觉，即使是在二维环境中也是如此。在地理学中，x坐标表示水平方向，y坐标表示垂直方向，而z坐标则表示第三个维度。深度与z坐标密切相关——除了宽度和高度之外，第三个坐标轴就用来表示深度信息。在二维作品中，深度是通过边缘的光影体现出来的。图2中的深色区域看起来就像是凹陷了进去。

- 重力的模拟同样重要。在二维作品中，重力是通过不同的视觉重量（visual weight）体现的。在模拟的环境中，该原则同样适用，但作品的最远处必须要有一些起指示作用的元素。What Lies Beneath中的最远处就是地底深处，这部分使用了最深最浓重的颜色，以体现出重力感（图4-6-4）。

在用二维设计描述三维物体时，结合使用上面提到的几种方法将会得到我们期望的物体的体积感（mass）。

在视觉设计的上下文中，我们可以将体积感理解为三维物体的二维表现样式，也就意味着用某种方法让物体从其周围的空间中突显出来。请参考如下几条建议。

- 封闭区域是由颜色和材质组成的。

- 物体的体积是由长宽高组成的。

体积感是这些视觉元素组合的结果，由此营造出了真实感（如图4-6-5所示）。

最后，在真实世界中取材，然后将其应用于模拟环境中自然会大大增强最终效果。在What Lies Beneath中，那个奇怪的、长着星形鼻子的鼹鼠就来

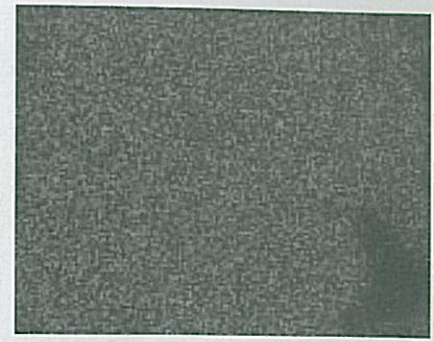


图4-6-3　交通灯控制台的表面材质却给浏览者带来了泥土的感觉（为清晰起见，图像已被调亮）

图4-6-4　作品中阴暗沉重的部分，表现了重力感（为清晰起见，图像已被调亮）

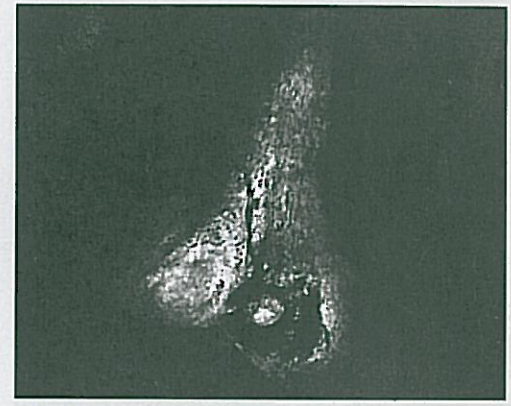


图4-6-5　人类腿骨埋在泥土中的图像，表现了体积感（为清晰起见，图像已被调亮）

Note

虽然设计领域中对体积与封闭区域的理解与数学上的相应概念有重合之处，但由于研究领域不同，二者在细节上仍有所区别。

图4-6-6　令人毛骨悚然的真实感：长着星形鼻子的鼹鼠

图4-6-7　作品的顶层——天空

图4-6-8　作品的中间层——草地（为清晰起见，图像已被调亮）

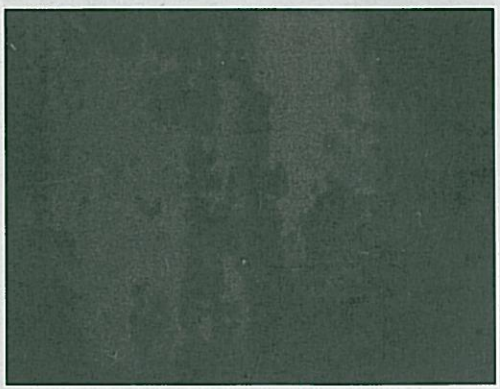


图4-6-9　作品的底层——地下

自于摄影作品——这画龙点睛的一笔毫无疑问地给浏览者带来了绝妙的真实感（图4-6-6）。

3. 处理层次

Web设计师的另一个很少考虑的问题就是作品的层次。顾名思义，层次表示环境的分层状态。联想某幢办公大楼能够帮助你理解层次的含义——每一个楼层都在不同的层次上。

在游戏开发领域中，人们非常重视在游戏环境中创建具有真实感的层次。游戏设计师经常会参考建筑师或环境设计师设计建筑的方法。作为Web设计师，我们也可以从建筑学、环境设计以及游戏设计中学到很多关于创建分层环境的方法。下面就列出了一些。

- 层次之间的线条、空间和大小要有区别。
- 层与层之间仍是一个整体，并以某种方式保持着联系。
- 层与层之间的过渡要自然。
- 作品中总是要包含焦点或重点。
- 正如前面提到过的那样，作品中绝对不能缺少重力感。人们都希望物体能够置于某个平面上。

What Lies Beneath中有三个层次：天空（图4-6-7）、草地（图4-6-8）以及地下部分（图4-6-9）。

仔细观察过这些图像之后，就会发现作品良好地运用了层次。每一个层次都能与其上一层（图4-6-10）和下一层（图4-6-11）良好地衔接起来，而重点则被放在了地下一层。

有趣的是，由于地下一层的高度在作品中占有最大的比重，加上其相对于上部和中部更深的颜色，所以作品很自然地带给人们相当的重力感。体积感就是由这些视觉元素组合起来形成的。良好的体积感也将带来绝佳的视觉体验，让浏览者就像是在欣赏真实物体一样（图4-6-5）。

4.6.2 模拟环境中的非真实存在元素

我们都知道，地下并没有页首文字、正文或链接之类的东西。尽管如此，Pick还是想方设法用一些独到的技术将这些非真实存在的元素和整个作品有机地结合了起来。

首先看一下页首文字（图4-6-12），Pick特别处理了字体，以便营造出低调、久远的感觉。将裂痕和泥土效果添加到这段文本上以后，页首文字也更好地融入到了整个作品中。

Pick在选择颜色时同样非常小心。链接的颜色来自于星形鼻子鼹鼠身上的较明亮部分，为作品添加了一丝明亮的意味，进而更好地将这些链接与作品基调融合起来（图4-6-13）。

作品中的链接并没有使用不同的鼠标悬停颜色，而只是在鼠标悬停时简单地显示出下划线。这并不是由于设计师的粗心，而是为了避免引入更多潜在的非真实存在元素。

若是Pick将链接设置为其他什么颜色，例如天空部分的蓝色，或是草地部分的绿色，那么整个作品的意境将荡然无存。原因很简单——这样做也就意味着把其他层次中独特的元素风格引入到与其毫不相关的层次中。

正因为Pick在设计时充分地考虑到了这些，整个作品的风格才仍旧能够保持高度统一。

4.6.3 空间相关的收获

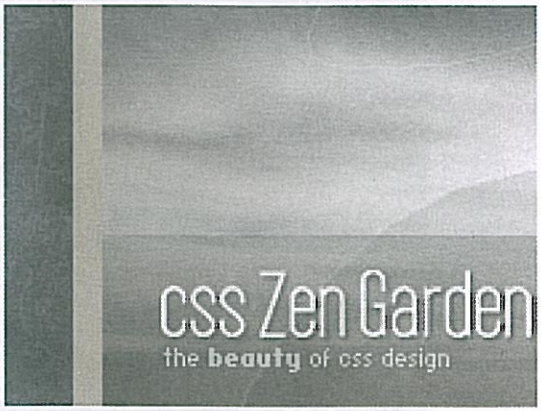
总体说来，禅意花园的本意就是鼓励大家进行实验性的创作，What Lies Beneath中描绘的地下环境正是这种创造力的一次绝佳体现。依靠着自己的聪明才智，加上一双善于发现的眼睛，Michael Pick最终完成了这幅如此独特的作品。

图4-6-10　从顶层到中间层的过渡。注意在草隙之间仍可以看到天空

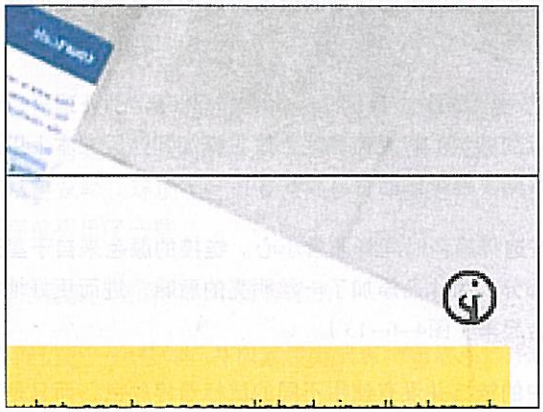
图4-6-11　从草地层到地下层的过渡是由逐渐加深的颜色实现的，表现出草生长在土地上的含义（为清晰起见，图像已被调亮）

图4-6-12　经过衰退陈化处理的文字能够更好地融入周围环境中

图4-6-13　坚持使用取自环境中的同一种颜色让这些非真实存在的文字不至于显得太过突兀

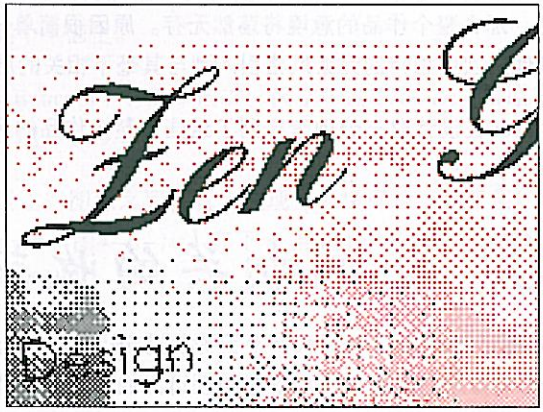


Oceans Apart（天水之间），154页



si6，160页

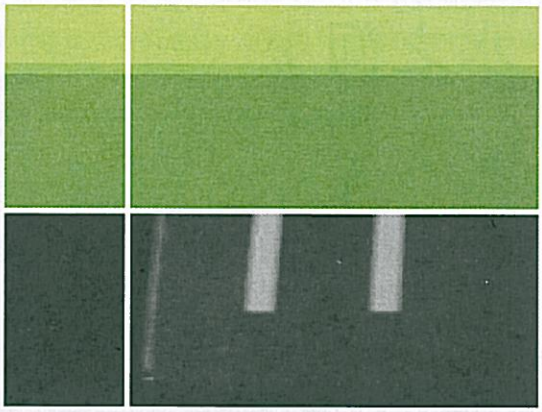
Dead or Alive（生与死），172页



Blood Lust（血色欲望），178页

第5章

文字排印（Typography）

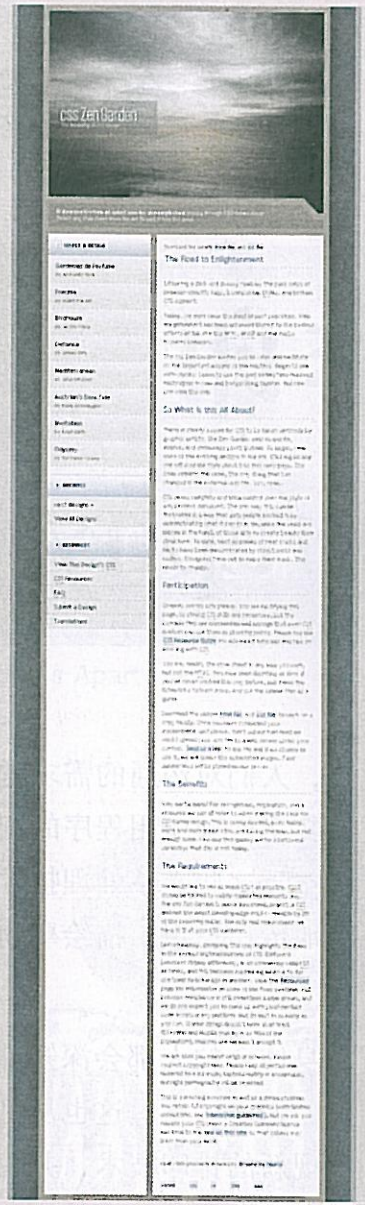


Release One（释放），166页

Golden Mean（黄金平衡），184页

随着现代社会的发展，人们对沟通的需求越来越强烈，这也不断促进着文字处理应用程序的创新和进步。当然，也正是得益于这些文字处理技术的不断发展，我们才能更好地传递思想，描绘事物和抒发情感。

任何对网站开发有亲身体会的人都会深知排印技术在网络环境中所面临的困境。不过这也从另一方面促使我们去完善和发现新的排印技术，并找出描述字体的形状、样式、色深和粗细的更好方法。我们希望本章中对文字排印介绍能帮助你更好地理解网页中处理文字的技术。



Ryan Sims，设计师

www.csszengarden.com/085

5.1 Oceans Apart（天水之间）

用简单的字体造就恒久之美

Oceans Apart中绚丽的色调取材于Ryan Sims在加州南部拍下的照片，作品中宁静优雅的色调勾勒出了引人入胜的神秘之海的意境。

秉承简约的理念，Oceans Apart中不带有任何唐突和喧嚣的部分。作者剔出了作品中一切哗众取宠的元素，彰显出他所恪守的海明威式哲学。而这部作品中的美绝非来自浮华的修饰，而是源于无处不在的简约和质朴。

5.1.1 字体排印学的基础知识

我们知道，字体排印学是设计学中的一个非常特殊的领域。许多设计师毕生致力于学习和使用排印技术，并且不断创造新的排印方法，才最终造就了这个领域现今的成果。虽然设计领域现有的经验有足够的可选排印技术，因而我们并不需要将自己培养成专业的排印师，但那些相关领域的基础知识却是任何一个设计师都需要了解的。

1. 字体格式（Type Form）

字体格式指的是字样（typeface）表现出的粗细、宽度和姿态（posture）等特征。

字体粗细指文字笔画的宽度。在CSS中其对应的属性是font-weight，该属性用来设置文字表现出的粗细。表5-1-1列举了font-weight属性和其可选值。

在这里需要注意的是有些字体并没有粗细适中的字样，因而它们并不能提供和CSS中等同的粗细划分（图5-1-1）。而事实上，即使字体有足够多的粗细划分，浏览器也未必能找出最合适的字样。

比较现实的做法是仅仅依赖两种设置：normal和bold。图5-1-1是Mozilla对各种粗细设定下同一字体的显示，可以看到Mozilla中的确只用到了normal和bold两种设置。

字体宽度指字符的宽度。紧缩（condensed）字体会显得更为狭窄，而扩张（expanded）字体则显得宽大（图5-1-2）。CSS中对应的属性是font-stretch，不过这个属性却尚未被任何浏览器所支持。

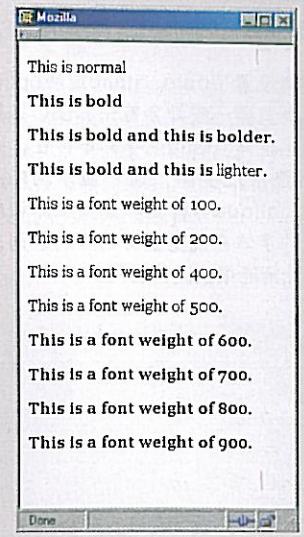


图5-1-1　在没有可用的中等粗细字体的情况下，浏览器会尽可能显示出相近的字体，一般均为normal或者bold

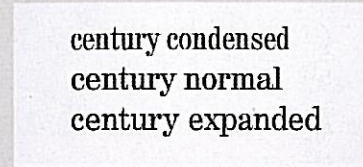


图5-1-2　同一字体的紧缩、正常和扩张显示

Note

如字体设置了bold、italic或者oblique等属性，浏览器一般都会首先尝试选择真实存在的衍生字体。若衍生字体不存在，浏览器就会调整常规字体，让它显示为bold、italic或者oblique等样式。真实字体和自动调整形成的字体在视觉上有明显的区别，有经验的设计师能很快区分它们。

表5-1-1　CSS中的字体粗细

属性和值	效果
font-weight：normal	用常规字体显示。和weight为400等价。
font-weight：bold	粗体显示，一般情况下对应于700的weight值，对少数字体对应于600的weight值。
font-weight：(300)	字体比正常粗细显得纤细，取值范围从最细的100到最粗的900。
font-weight：bolder	文字显示得比继承值更粗。例如一个段落的粗细已是bold，而这个段落内的strong元素的粗细为bolder，那这个strong元素中的文字将比段落内的其他文字更粗。唯一的例外是，假如继承得来的粗细已经是最高值900，那应用bolder后依然是900。
font-weight：lighter	文字显示得比继承值更细。如果继承的weight值为300，那计算所得的值将为200。与bolder类似的是，若继承值为100，计算所得的值还是100。
font-weight：inherit	使本段文字的字体粗细和父元素的一致。

另一个字体格式是姿态，与之对应的CSS属性是font-style。所谓的字体姿态实际上是指字体的倾斜程度。姿态有两种。

- Italic——一种在常规字体基础上做过特殊设计的变体，一般带有手写风格。
- Oblique——常规字体由浏览器作倾斜处理后的版本。和italics不同的是，oblique字体并不出自另外设计的变体，而仅仅是常规字体应用倾斜效果的结果。而多数人则错把oblique字体说成italic字体。

font-style属性的合法值有normal、italic、oblique和inherit。

2. 字体对比度（type contrast）

字体对比度指字型中粗的部分与细的部分之间的差异程度。不同字体的对比度相差很大，甚至有些字体没有任何粗细变化。

在印刷作品设计中，人们经常用一种字体撰写标题，而使用另一种不同的字体撰写正文。通常做法中，圆滑、缺乏粗细对比的sans serif字体被用作标题用字，而Serif字体则常用作正文字体，这种设定在长段文章中的应用尤其广泛。你平日阅读的书籍大多沿用了这种设计方式，不过这种传统在当今的Web时代却受到了很大的质疑。

3. 字体色深（type color）

字体色深指的是文字的深浅程度。例如Helvetica Light这样的字体相对较浅，这是由于此类字体较细因而含墨量较小。而Cooper或者Arial Black这类字体的含墨量就很高，因而显得重而深。当然，字体色深也不完全依赖于字体本身，其他因素如字距调整（kerning）、字间距（tracking）和行间距（leading）等也会影响人们感受到的字体色深。

除了对比度外，字体的颜色也至关重要。例如，字体色相（type hue）的不同也能造成字体的差别，这又给设计师带来了更多选项。

不过需要提示的是，通常情况下设计师都将作品中颜色的数量控制在一定范围之内。虽然诸如儿童主题比较适合加入多种颜色，但绝大多数情况下，在页面上使用过多颜色都会让人感觉不够专业。

5.1.2 Oceans Apart中的字体

在Oceans Apart中，我们可以看到Sims清晰而简洁的用字逻辑——所有的HTML中的文字都使用了Trebuchet字体，而备用字体也使用了几种sans serifs字体。

```
body {font: 11px "trebuchet ms", arial, helvetica, sans-
serif;}
```

Trebuchet本身也是一种sans serif字体，而且它还是Microsoft核心Web字体的一员，此外，Microsoft Office中也包含了这种字体，因而我们可以认为多数系统都能支持Trebuchet字体。但可以注意到，细心的Sims并没有遗漏设置备用字体这重要的一环。

Note

在网络上，很多人的想法恰恰相反，他们认为在显示器上Sans serif字体更适合作为正文字体，这也引发了一些争论。若想了解更多关于这类争论的信息，请参考www.digital-web.com/news/2004/06/serif_vs_sanserif。

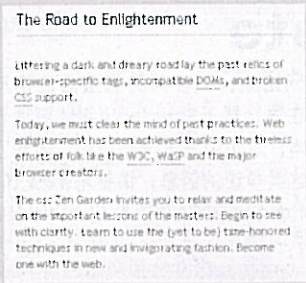


图5-1-3　作品中使用了对比度较低的文字

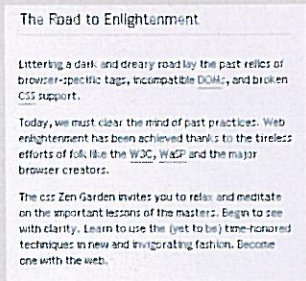


图5-1-4　使用蓝色字体未必能增强整体对比度

Tip

毋庸置疑，低对比度的字体会造成视觉不便人士阅读困难。因而在制作受众面较宽的公共网站时，使用传统的高对比度设计依然是最好的选择。

1. 色相和对比度的选择

对色相与对比度的选择至关重要，因为它不仅仅关系到设计的一致性，还直接影响到作品的可读性。

传统的文字设计理论认为提高前景和背景的色相对比——例如使用白底黑字的设计——会大大提高文本的可读性。

但在Oceans Apart中，Sims却抛弃了传统规则的束缚，使用了相对并不显眼的浅褐色作为正文字体：

```
color: #9F927F;
```

可以看到，这样的文字和白色的背景的对比度很低（图5-1-3）。而作品中无论正文部分还是侧栏中的链接都使用了深蓝色，这一点很值得玩味，因为多数设计师都下意识地选择纯蓝以提高对比度（图5-1-4）。

不可否认，使用纯蓝色能提高对比，但事实证明使用不显眼的褐色反而带来了更吸引人的效果——正是使用了褐色的正文字体，才使得页面显得更加柔和清晰，进而让读者能更好地阅读其中的内容。

而将蓝色用于链接则提高了作品整体的对比，增加了作品的视觉冲击力。

对比效果在作品头部的应用也十分突出：顶部图片上的文字使用了反差强烈的白色调（图5-1-5），而图片下的描述性文字则置成了和褐色背景相异的白色，这大大增强了作品的整体视觉效果（图5-1-6）。

2. 字号对比和字体颜色

使用不同大小的字体可以增加文字区块间的对比。在Oceans Apart中，这种使用简单的效果却非常有效。

可以看到，作品中正文和链接列表的一部分使用了11像素字体，链接列表中另一部分链接则使用了9像素字体作为辅助。图5-1-7中我们可以看到，这区区两个像素的差异却造成了相当强烈的视觉对比。

另外，Sims还在#linkList2中使用了橄榄绿色，这种色彩不但被链接列表的背景清晰地反衬出来，也同时和页面其他部分的蓝色、褐色形成了鲜明对比。

3. 字号对比和粗细

再回到图5-1-5中的头部图像，就会发现图像中同时使用了字号和粗细变化来增加对比度："css"使用了小写，从而和首字母大写的"Zen Garden"形成了对比，而在其下的"the beauty of css design"则特意将"beauty"加深为粗体，这又和周围的常规文字形成反差。

再看头部图片下方的文字，它们也使用了相同的技巧——前7个单词置为粗体，而后续单词都为常规粗细（图5-1-7）。

4. 标题和正文的关系

作品中最后一个亮点是正文和标题的关系。仔细观察可以发现，正文部分标题使用了图片，而不是简单的HTML文本，这为加入细微的文字阴影提供了坚实的基础，而经过阴影润色的文字毫无疑问将更加吸引眼球。

谈及吸引眼球之处，设计中还另有一处与众不同的地方——链接部分的标题（图5-1-8）。这些标题的文字全部使用了大写、深蓝的字体，它们和浅蓝底色与褐色整体背景形成鲜明的对比，很好地凸现了链接区域的导航作用。

5.1.3 从简洁设计中学习

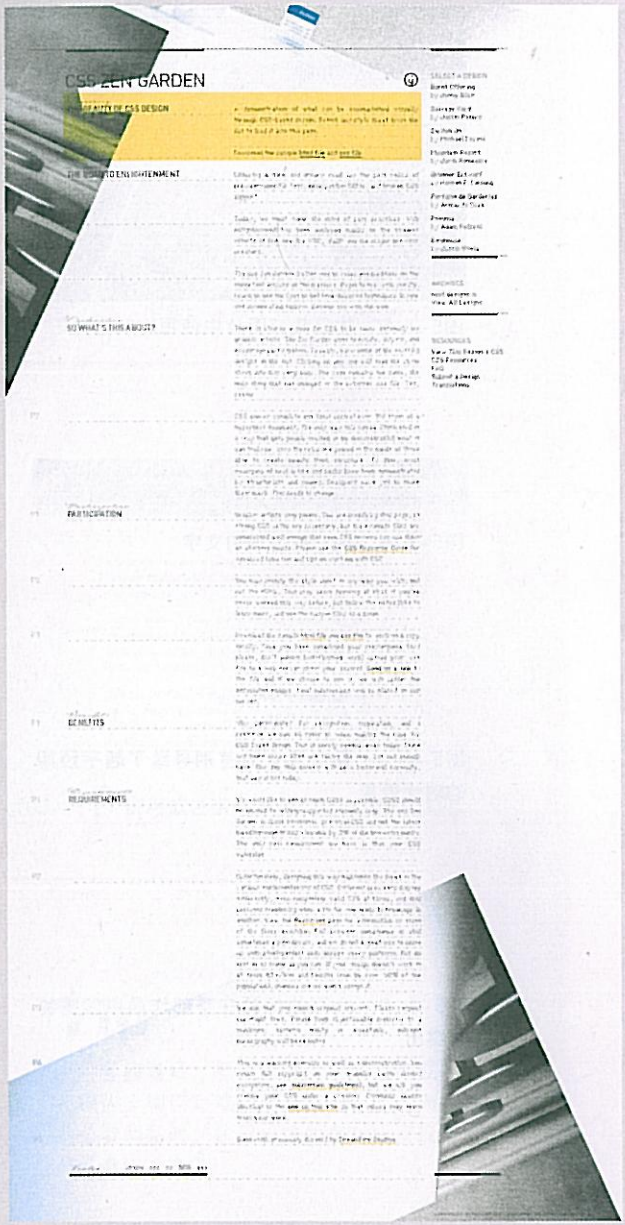
初见Oceans Apart，人们都会陶醉于其自然、平和却略带变化的简洁风格中。而作品字体运用中所触及的细微变化手段或许是我们更应该仔细学习的地方。

图5-1-5　作品头部图片中使用的对比技巧

图5-1-6　头部图片下的文字

图5-1-7　两个像素的差别导致了超乎预期的对比效果

图5-1-8　大写的标题在整部作品中显得特别突出



Shaun Inman，设计师

www.csszengarden.com/044

5.2 si6

应对字体的困扰

Shaun Inman在制作si6的过程中采用了一种特殊的网格化布局。这种网格式的布局在作品中形成了潜在的视觉边界，从而将页面水平划分为3栏。而文字间黯淡的线条则又将左侧的标题和中部的正文区域分割成多个部分。

在这部作品中，字体被细心地调整为和周围装饰线条相称的样式。使用这样的像素级的字体能较好地将作品中的线条与文字融合，不过这种方法也存在缺点。例如，用户可能因文字过小而手动改变浏览器字体大小，导致预定的字体与线条间的比例遭到破坏。

5.2.1 字体选择的限制

或许网站设计中最让人头疼的问题就是字体选择，因为用户可以随意更改计算机的字体库，而那些通用的字体又少之又少，因而设计师很难确定究竟哪些字体才可以使用。更让人沮丧的是，在未来的一段时间内，字体选择中面临的这个问题仍不大可能被完美解决。

我们知道，除了Flash之类的内嵌字体媒介，或者图片对象之外，所有站点文字都依赖于用户本地安装的字体。而由于用户并非设计师，他们所安装的字体非常有限，且很大程度上用户拥有的字体取决于他们的操作系统和相关安装的软件，例如，Microsoft Office用户会比非Office用户拥有更多的Office中捆绑的字体。

如果考虑操作系统的因素，我们会发现Windows和Mac OS X默认安装的字体并不一致，但还是可以从中发现一些共有的字体。可若是加入了Unix和Linux，情况就会变得更加难堪——我们几乎无法找到在Windows、OS X和Unix/Linux中共有的字体。表5-2-1列举了每个平台自带的字体。

表5-2-1　常用的Web字体（按操作系统平台分类）

	Windows	苹果机	Unix/Linux
Serif	Book Antiqua, Bookman Old Style, Garamond	New York, Palatino, Times	Bitstream Vera Serif, New Century Schoolbook, Times, Utopia
Sans Serif	Arial Narrow, Century Gothic, Lucida Sans Unicode, Tahoma	Charcoal, Chicago, Geneva, Helvetica, Lucida Grande	Bitstream Vera Sans, Helvetica, Lucida
Monospace	Courier,Lucida Console	Courier,Monaco	Bitstream Vera Mono,Courier

Tip

微软公司为其旗下产品所附带的字体给出了一个列表。（http://www.microsoft.com/typography/fonts/default.aspx）。但由于用户可以自行删除已安装的字体，或者选择不安装某些附加字体，因而我们并不能假定用户必然会全部拥有这些字体。

Tip

从2001年开始，有人为Web上的常用字体展开了一个调查。尽管调查的结果存在着一定不确定性而不完全可信，但至少可以作为一个参考的依据。关于这个调查的更多信息，请参考“Code Style font sampler”（http://www.codestyle.org/css/font-family/）。

Note

不幸的是，从2002年开始微软公司就取消了官方提供的核心Web字体下载。但目前仍可以从http://corefonts.sour ceforge.net下载到它们。由于这个页面是为Linux用户准备的，需要花些功夫找到Windows下的版本——“Original unaltered .exe files”。

图5-2-1　si6中的文字在（自上而下）Geneva、Arial和Tahoma下的显示

1996年微软公司专为屏幕显示发布了一系列的字体。之后，这些被称为核心Web字体的资源在网络上广为流传。如今它们已经被包含在微软的各种产品中。表5-2-2列举了这些核心Web字体。

表5-2-2　微软核心Web字体

Serif	Sans Serif	Monospace	Special
Georgia,Times New Roman	Arial, Arial Black, Trebuchet MS, Verdana	Courier New, Andale Mono	Comic Sans MS, Impact, Webdings

通常情况下，使用这些字体都是安全的。但这只是通常情况——仍然会有未安装这些字体的用户，因而这也不过是相对而言的安全。正是为了处理那些例外的情况，CSS标准中才引入了备用字体的机制。

5.2.2　通用字体族

例如在si6中，Inman最初设定的字体为Mac机上的Geneva，但是考虑到这种字体在Windows上未必存在，所以最终加入了额外的备选字体如下：

```
body {
 font: 9px/16px Geneva,Arial,Tahoma,sans-serif;
}
```

这段声明中的前两个值将字体大小设为9像素，将行高设为16像素，而后按次序设置了备选字体（图5-2-1）。这样，如果浏览器找不到Geneva字体，就会去寻找Arial字体，若无法找到Arial字体，则会去寻找Tahoma字体，如果连Tahoma字体也无法找到，那它将使用浏览器中sans-serif字体族的默认字体。

CSS内置了一系列的字体族，而所有CSS兼容的浏览器都能自动为每个字体族找到至少一个合适的默认字体（当然，用户也可以覆盖默认设置）。CSS并没有要求我们一定要在字体设定时设定一个字体族作为最后的备选字体，不过这样做却可以防止浏览器在没有匹配的情况下选择其默认的Times New Roman字体。

下面列出了所有的通用字体族并加以解释。

• serif

serif字体都有明显的装饰钩，所以在字母笔画的结尾处大都能看到细小的衬钩（图5-2-2）。Times New Roman是默认的serif字体，除非用户特意改变默认设置。

• sans serif

sans-serif相对serif的区别是它没有装饰钩（图5-2-3）。Arial是默认的sans-serif字体。

• monospace

monospace字体的每个字母都有相同的宽度（图5-2-4）。因而在monospace中字母“i”的宽度和字母“m”的宽度是相同的。这类字体在显示程序代码时特别有用。Courier（或者Courier New）是默认的monospace字体。

• cursive

cursive字体模仿手写的风格，通常用于标题（图5-2-5）中。从名字可以知道cursive多数为草体，但事实上各类手写字体都属于cursive范畴。cursive字体族没有默认的字体，不过最常用的莫过于Comic Sans了。

• fantasy

fantasy则完全是装饰用字体，多数用于标题（图5-2-6）。由于fantasy字族拥有太多的字体，几乎无法为这类字体的大小形状下一个定论，因而也很少有人会在正式的站点设计中使用它们。

5.2.3 传统虽好，但是……

现在假设被迫在每4个网站中的3个里都使用Verdana和Georgia字体，那么选择一种多数用户没有的字体一定会是件很有趣的事情。由于可以为多数用户设定备份字体，你可以安全地设置特殊字体以拓展新的交互体验。

Times New Roman

Georgia

Garamond

图5-2-2 serif字体：Times New Roman、Georgia和Garamond

Helvetica

Futura

Lucida Sans Unicode

图5-2-3 sans serif字体：Helvetica、Futura和Lucida Sans Unicode

Courier

Monaco

Andale Mono

图5-2-4 monospace字体：Courier、Monaco和Andale Mono

Comic Sans

Apple Chancery

Brush Script MT

图5-2-5 cursive字体：Comic Sans、Apple Chancery和Brush Script MT

Impact

Papyrus

图5-2-6 fantasy字体：Impact和Papyrus

Impact
Chicago
Verdana
Trebuchet MS
Lucida Grande
Geneva
Eurostile
Andale Mono
Helvetica
Arial
Arial Narrow
American Typewriter
New York
Monaco
Big Caslon
Palatino
Optima
Times New Roman
Georgia
Futura
Apple Chancery
Bell Gothic
Myriad Pro
COPPERPLATE
Gill Sans
Courier New
Baskerville
Brush Script MT

图5-2-7　在16像素下，不同字体有不同的x-height。注意列表顶端的字体和底端字体的尺寸差异

下面的字体组合都是将相对特殊的字体放在最前，随后是安全的备份字体：

```
body {
  font-family: "Lucida Grande", "Lucida Sans Unicode",
  Verdana,
  ➥sans-serif;
}
h2 {
  font-family: Garamond, "Times New Roman", serif;
}
dt, li {
  font-family: Optima, Arial, sans-serif;
}
.sidebarCallout {
  font-family: "Century Gothic", Verdana, sans-serif;
}
```

之前我们看到Inman将Geneva字体作为了站点的首选，同时将Arial和Tahoma作为备份。通常来说，这是一种很好的组合；因为这3种字体都有相同的x-height。这里需要谈及x-height，在Rob Carter、Ben Day和Philip Meggs所著的《Typographic Design:Form and Communication, 3rd ed.》（2002年由John Wiley & Sons出版社出版）一书中，将x-height定义为“小写字母去除高出部分[1]和低出部分[2]后的高度，而通常就是小写的x字母的高度”。拥有相同x-height的字体往往具有较好的相似性。

可能已经注意到，即便设置了相同的像素值，一些字体还是显得比其他字体更大。例如（图5-2-7）中，16像素的Verdana字体就比其他字体更大些。因而在设置字体组合时，需要记住font-size对所有字体的作用都是相同的。因而如果使用了一种x-height较大的字体，那在用户看来这个字体会显得比其他字体略小些。所以我们必须事先在各种平台和浏览器上进行足够多的测试。

1998年，CSS2引入了一个新的属性font-size-adjust，用以解决x-height差异。但是这个属性至今尚未被广泛地实现，所以在CSS 2.1标准中，甚至干脆删除了它。或许在CSS规范最终完成之后的某天该属性会被广泛支持，但就目前来说，我们还是应该避免使用。

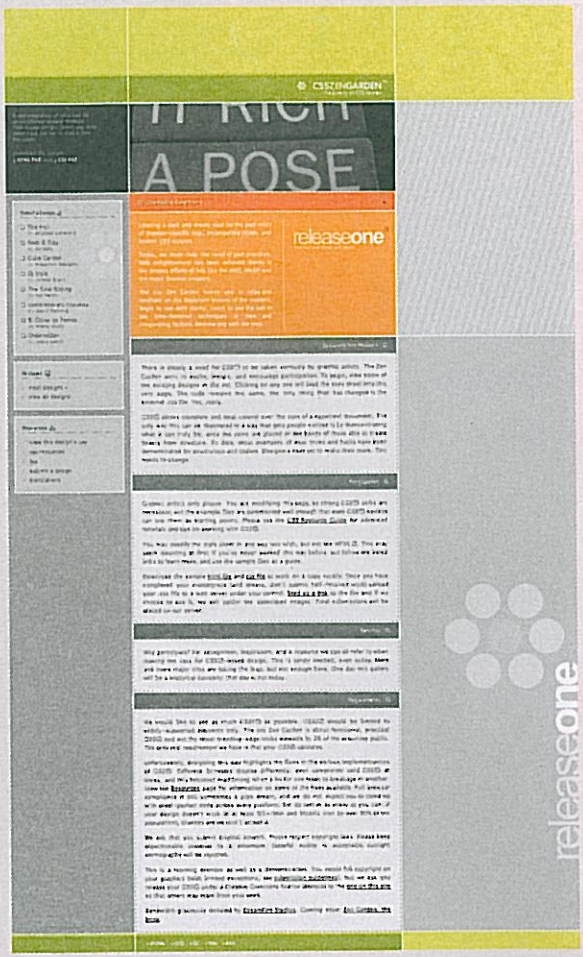
[1] 例如字母“d”上面的出头部分。——译者注

[2] 例如字母“y”下面的钩。——译者注

5.2.4 字体选择

由于用户安装的字体千差万别，我们无法预知用户所见文字的真实样式。因而现实中某些用户看到的文字形象与设计师看到的可能截然不同。

但这并不成为我们永远使用Arial字体的理由，字体的运用是一个创新和突破的过程。从Shaun Inman的作品中，我们也能看到突破核心Web字体束缚后作品效果所得到的显著增强。



Didier Hilhorst，设计师

www.csszengarden.com/035

5.3 Release One（释放）

适应文字大小的变化

Didier Hilhorst的Release One堪称禅意花园的又一力作。这次他使用了和以往类似的创作流程，同时也充分地考虑了禅意花园的受众（网站设计师）、作品要求（正确的代码、完美的设计）和竞争力（其他主要的设计站点），最终完成了这部杰出的作品。

为了使Release One成为禅意花园的另一部标志性作品，Hilhorst特地为作品制作了很多与众不同的元素，例如圆形的站点LOGO象征着高雅的雏菊（图5-3-1），再配以明亮的绿色和橙色作为前景和柔和的灰色调背景，最终营造出一种独特的设计风格。

5.3.1 字体大小

虽然书本的印刷和屏幕上的显示有一定差距，但我们还是能够发现Release One中的字体显得很小。查看样式表其可发现其中的原委：

```
p {
 font-size: 11px;
 text-align: justify;
}
#quickSummary p.p1 {
 font-size: 9px;
 background: transparent;
 color: #999;
 text-align: left;
}
```

这些样式限制了作品中最大的字号不会超过11像素，而有时甚至只有9像素。由于网页中大号字体的显示效果较差（12像素以上的Verdana显得尤为丑陋），网站设计师往往更倾向于小号字，而小号字也意味着同样的空间可容纳更多内容——这也成为了使用小字号的另一个原因。当然，设计师对小号字的偏好并不意味着普通阅读者也能接受，在这里我们只能期望用户不觉得反感。

5.3.2 浏览器的控制

所幸的是，浏览者还有另外的选择——主流浏览器都允许用户手动缩放字体到他们满意的效果。在多数浏览器中，这种功能位于View菜单中，你也可以使用Ctrl-+或者Ctrl--（减号）来缩放字体。

若是仅依靠这种功能就能达到目标，那本节的内容就可到此为止了。事实并不那么乐观——要知道，以像素为单位的字体在Windows平台上的IE中不能用前面所说的方法改变大小！

图5-3-1 别具一格的禅意花园图标

Note

如果使用Mac OS X或者Windows XP（且已经开启ClearType功能）这类带有字体平滑功能的操作系统，那辨识Release One中的文字还不算太困难（图5-3-2和图5-3-3）。但如果操作系统没有这类功能，就不得不使用额外的辅助工具来阅读作品中的文字了（图5-3-4）。因为只有一部分字体（例如Verdana）在小字号情况下才能较为清晰的显示。

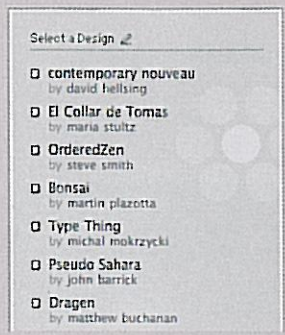


图5-3-2 Mac OS X的字体平滑功能使得Release One中的文字在较小情况下也能正常分辨

图5-3-3 Windows XP的字体清晰功能（ClearType）甚至在更小的字体下也能清晰显示

5.3.3 绝对单位和相对单位

在CSS中，绝对单位指预先可以确定单位长度的度量单位，它包括cm（厘米）、in（英寸）和pt（磅）等。它们中的每个都和真实世界中的长度对应，也可以将2.5厘米和1个半英尺做比较。

而相对单位则包括%、em、larger和smaller等，这是一种可变的尺寸。2em和200%是1em和100%的2倍大。但有一个疑问：1em究竟有多大，或者说100%是什么的100%呢？实际上，每个相对单位都和一个初始长度相关，但由于这个初始长度也是可以变化的，所以1em既可以定义为3英寸也可以定义为0.55厘米。

使用相对单位意味着存在一个默认的基值，也就是1em或者100%时的大小，然后再按照当前值成比例地缩放字体。比如页首标题可以设为150%或者2.2em，次要标题可以设为0.8em或者75%。这个基值可以自定义或者使用浏览器的默认设置，浏览者也可以手动改变字体大小，进而让页面文字自由缩放。用户可控的字体缩放是使用相对单位的一大优势，如果计算机未开启Windows XP的ClearType（字体清晰）功能，将发现Release One的字体小得几乎无法阅读，因此手动缩放功能的重要性不言而喻。

第三种单位类型是描述字体大小的关键字，它的合法值包括xx-small、medium和x-large等。在CSS的定义中，它们也属于绝对单位，但实际显示大小却都不是那么精确。

5.3.4 像素及其导致的问题

根据CSS的定义，px实际上是一个相对单位。这其中有一长套冗长、模糊、牵涉到视觉距离、视角等复杂概念的理论。不过一个较容易理解的例子就是，1280×960分辨率的15寸显示器上的1px比800×600分辨率的21寸显示器上的1px要小得多。由于10像素在后者中显示得足够大，而在前一台上却小得几乎无法分辨，因而我们不能说像素是个绝对单位。

那如果像素是个相对单位，用户是否就能手动调整像素字体的尺寸呢？事实并非如此，Windows平台上的IE就缺少这种功能，这意味着在IE中，像素事实上被作为绝对单位来处理。那是不是说我们只能用em或者%来满足手动缩放？别高兴得太早了——相对单位也没有我们想象中那么理想。

1. 起始值

应该还记得，前面提到相对单位都存在一个基值。但是如何设定这个基值呢？可以依靠浏览器的默认值吗？答案恐怕是不能的，要知道浏览器的默认为16像素字，这对多数人都显得太大了。

2. 不一致性和舍入误差

如果可以撇开浏览器兼容问题，那或许可以依赖浏览器的默认基值。但事实上由于操作系统和旧式浏览器的差异，同样的16像素默认字体，在各种浏览器所显示的实际效果却大相径庭。设计师Owen Briggs在数年前作过一项测试，他抓下了100多张不同环境下浏览器对16像素字的显示图像，结果发现要找到不同浏览器中的最终显示尺寸的关系几乎就是痴人说梦（图5-3-5）。而使用CSS关键字也有同样的问题。因而在很长一段时期内，没有人敢于使用相对的字体大小——1个em可能被显示为12像素到16像素之间的任意大小。

我们先前说到的还只是基值问题，字体缩放同样也存在问题。倘若某浏览器将1em显示为16个像素，0.9em又会有多大呢？答案是有些浏览器或许会把14.4像素截取为14像素，而有些却会舍入为15像素！

3. 相对单位的组合

若是使用相对单位不当的话，很有可能会把作品赶入死胡同。考虑下面的代码：

```
<div id="benefits">
  <h3>Benefits</h3>
  <p>Why participate? For recognition, inspiration, and
  a resource we can all refer to when making the case
  for CSS-based design. This is sorely needed, even
  today.</p>
</div>
```

如果应用下述的CSS样式，假设100%大小是16像素，段落的字体又会有多大呢？

```
#benefits {
 font-size: 75%;
}
p {
```

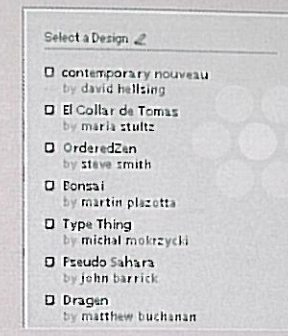


图5-3-4　Release One在不使用ClearType功能时几乎无法阅读

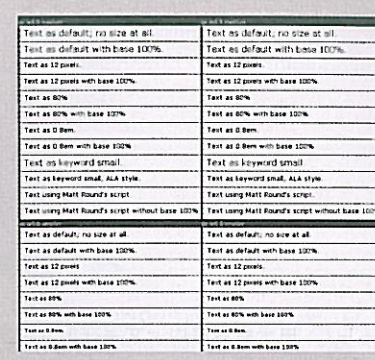


图5-3-5　Owen Briggs的屏幕截图的一部分，可以到各种浏览器下的16像素字尺寸相差很大

Tip

更多关于Owen Briggs的测试结果请参考“Text Sizing”（http://thenoodleincident.com/tutorials/box_lesson/font/index.html）。

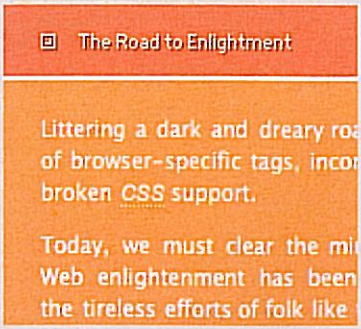


图5-3-6　文字在现代浏览器中的大小较为合理

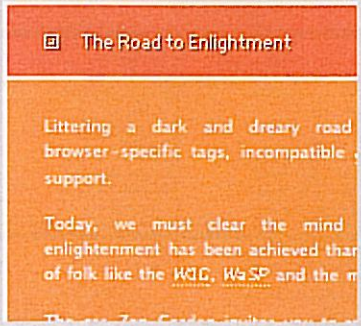


图5-3-7　由于组合原因，文字在旧式浏览器中变得难以阅读

Tip

互联网上有相当多关于文字大小的指导。例如MaxDesign（www.maxdesign.com.au/presentation/relative）css-discuss（http://css-discuss.incutio.com/?page=FontSize），以及Dive Into Accessibility（http://diveintoaccessibility.org/day_26_using_relative_font_sizes.html）等。

```
  font-size: 75%;
}
```

答案是12像素还是9像素呢？由于第一条样式将#benefit中的字体降至75%（12像素），而第二条样式又将12像素降至75%，所以正确的答案是12×75%＝9像素。

设想若不对作品做仔细检查，那这种组合方式将导致无法预测的后果——可能由于多次的收缩导致文字小得无法阅读。在那些较老的浏览器，如苹果机上的IE中，后果更加糟糕（图5-3-6和图5-3-7）。

5.3.5　合理地设置字体大小

所以无论何种方式，字体大小的设置过程中总会有些令人不快的问题。那设计师应该如何处理呢？似乎使用像素值是一个代价最小的方案，但这会让用户遇到太多的不便。所以下面介绍3种常用的解决方案，当然它们中的每一个都各有优劣。

1. 使用em

在Owen Briggs完成了测试之后，他发现了使用百分比的方式能够避免很多舍入误差。因而在对body元素应用了百分比单位后，我们就可以为其余元素使用em值。使用这种方式，字体的一致性相当不错：

```
body {
  font-size: 76%;
}
p {
  font-size: 1em;
}
```

Briggs建议不要将文字大小设置为1em以下。那上述设置将导致p元素中的文字大约为12像素大小（假定浏览器默认16像素字）。这对于多数浏览者都是相当舒适的字号，同时也允许他们手动改变字体大小。具体的em方法的介绍可见Briggs的个人站点中的“Sane CSS Sizes”一文（www.thenoodleincident.com/tutorials/typography/index.html）。而这个方案的唯一缺点就是它假定了所有浏览器都使用16像素为默认字号。尽管多数浏览器都是这样实现的，但若使用户悄悄改变了默认设置，那自然会导致问题。

2. 使用关键字

在使用字号关键字时，最容易出现问题的浏览器就是Windows平台上的IE 5，而在其余浏览器中都能很好地显示。因此，我们可以使用盒模型hack为Windows平台上的IE 5设置不同的字号值，这样即可正常使用CSS 字号关键字：

```
body {
 font-size: x-small;
 voice-family: "\"}\"";
 voice-family: inherit;
 font-size: small;
} html>body {
 font-size: small;
}
```

每个关键字之间的缩放比是1.2，所以如果medium的值为16像素的话，上述代码将让body中字号为13像素。

3. 使用样式切换方法

如果要精确控制屏幕上文字的大小，px是最为可靠的单位。可是因为IE的缘故，我们并不能依赖px，但仍可通过另行制作显示大号文字的样式表来解决IE中的这个问题。

对于某些浏览器来讲，切换样式表的功能是内置的，但这类浏览器中却并没有IE的身影。所以为了让用户可以切换文字大小，在页面中我们还必须加入相应的控制脚本。Paul Sowden在A List Apart上曾写过一个有名的实现方法。显然，除了用于字体之外，这种方式也可以用来处理各种其他类型的设置切换。

5.3.6 选用何种方法？

那我们在网站设计中应该如何选择呢？这取决于个人喜好。虽然普通用户大都不会拥有Didier Hilhorst之类图形设计师那样的好眼力，但只要我们合理地运用的字体大小，还是可以很容易地满足他们的需求。

Note

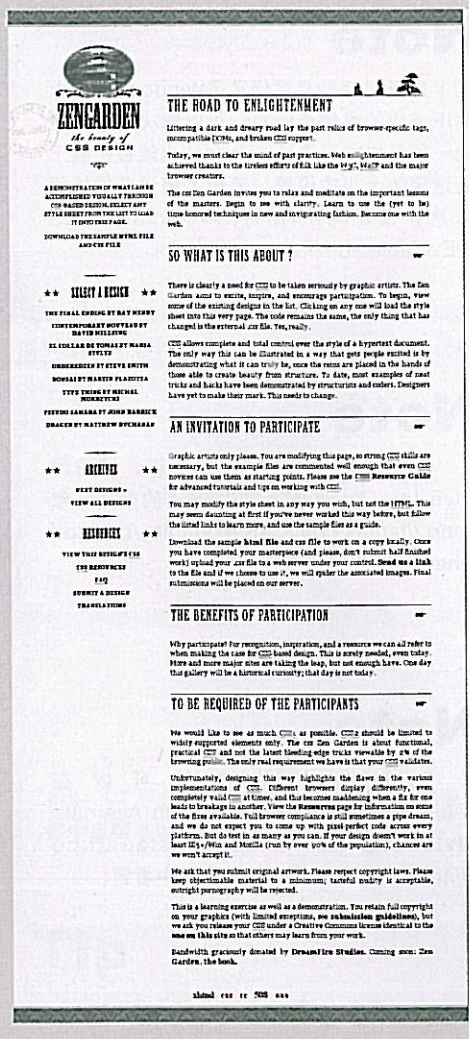
盒模型hack技术修正了Windows平台上的IE 5中的盒模型问题，可以解决器中存在的一部分样式问题。这个hack也能被扩展于处理字体大小的设置。若想了解更多的相关信息，请访问http://tantek.com/CSS/Examples/boxmodelhack.html。

Note

关于左边的代码的详细解释，请访问Dive Into Accessibility网站上的相关文章（http://tantek.com/CSS/Examples/boxmodelhack.html）。

Note

Paul Sowden的方法可以在“Alternative Style: Working with Alternate Style Sheets”（www.alistapart.com/articles/alternate）这篇文章中看到。



ZENGARDEN
the beauty of CSS DESIGN

A DEMONSTRATION OF WHAT CAN BE ACCOMPLISHED VISUALLY THROUGH CSS-BASED DESIGN. SELECT ANY STYLE SHEET FROM THE LIST TO LOAD IT INTO THIS PAGE.

DOWNLOAD THE SAMPLE HTML FILE AND CSS FILE

SELECT A DESIGN

THE FINAL ENDING BY RAY HENRY
CONTEMPORARY MOVEMENT BY DAVID HILLESHEIM
EL COLLAR DE TOMAS BY MARIA STYLTZ
ORIGEREDIEN BY STEVE SMITH
BONSAI BY MARTIN FLADUTTA
TYPE THINK BY MICHAL HOREJTCKI
PEYOTE SAMURAI BY JOHN BARRICK
DRAGON BY MATTHEW BUCHANAN

ARCHIVES

NEXT DESIGNS »
VIEW ALL DESIGNS

RESOURCES

VIEW THIS DESIGN'S CSS
CSS RESOURCES
FAQ
SUBMIT A DESIGN
TRANSLATIONS

THE ROAD TO ENLIGHTENMENT

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support.

Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators.

The CSS Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

SO WHAT IS THIS ABOUT ?

There is clearly a need for CSS to be taken seriously by graphic artists. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The code remains the same, the only thing that has changed is the external .css file. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. To date, most examples of neat tricks and hacks have been demonstrated by structurists and coders. Designers have yet to make their mark. This needs to change.

AN INVITATION TO PARTICIPATE

Graphic artists only please. You are modifying this page, so strong CSS skills are necessary, but the example files are commented well enough that even CSS novices can use them as starting points. Please see the CSS Resource Guide for advanced tutorials and tips on working with CSS.

You may modify the style sheet in any way you wish, but not the HTML. This may seem daunting at first if you've never worked this way before, but follow the listed links to learn more, and use the sample files as a guide.

Download the sample html file and css file to work on a copy locally. Once you have completed your masterpiece (and please, don't submit half finished work) upload your .css file to a web server under your control. Send us a link to the file and if we choose to use it, we will spider the associated images. Final submissions will be placed on our server.

THE BENEFITS OF PARTICIPATION

Why participate? For recognition, inspiration, and a resource we can all refer to when making the case for CSS based design. This is sorely needed, even today. More and more major sites are taking the leap, but not enough have. One day this gallery will be a historical curiosity; that day is not today.

TO BE REQUIRED OF THE PARTICIPANTS

We would like to see as much CSS1 as possible. CSS2 should be limited to widely supported elements only. The CSS Zen Garden is about functional, practical CSS and not the latest bleeding edge tricks viewable by 2% of the browsing public. The only real requirement we have is that your CSS validates.

Unfortunately, designing this way highlights the flaws in the various implementations of CSS. Different browsers display differently, even completely valid CSS at times, and this becomes maddening when a fix for one leads to breakage in another. View the Resources page for information on some of the fixes available. Full browser compliance is still sometimes a pipe dream, and we do not expect you to come up with pixel perfect code across every platform. But do test as many as you can. If your design doesn't work in at least IE5+/Win and Mozilla (run by over 90% of the population), chances are we won't accept it.

We ask that you submit original artwork. Please respect copyright laws. Please keep objectionable material to a minimum; tasteful nudity is acceptable, outright pornography will be rejected.

This is a learning exercise as well as a demonstration. You retain full copyright on your graphics (with limited exceptions, see submission guidelines), but we ask you release your CSS under a Creative Commons license identical to the one on this site so that others may learn from your work.

Bandwidth graciously donated by Dreamfire Studios. Coming soon: Zen Garden, the book.

valid CSS 1! 508 AAA

Michael Pick，设计师

www.csszengarden.com/009

5.4 Dead or Alive（生与死）

设计优雅、带有“重力感”的文字

在考虑提交何种类型的作品时，Michael Pick突然想到了一个有趣而古怪的念头，并跟随着这个奇怪的想法完成了他的Dead or Alive。

Pick的创意是将西方古朴的风格和日本武士道相结合，营造出令人着迷的历史反差感。为了引入西方古典派的主题，Pick使用了带有强烈西方风格的字体，并加以额外的润饰。最终完成的作品同时具有东西方双重风格，给人们带来了深刻的视觉印象。

5.4.1 装饰性元素

Dead or Alive中最引人瞩目的莫过于其中的装饰字体，也就是我们所知的符号（dingbat）字体。这是一种图案状的、起装饰作用的字体，它通常可分为如下几类：

- 商业
- 体育
- 动物
- 自然
- 创意/不规则元素

图5-4-1中列举了5种常用的符号字体

Pick在Dead or Alive中通过使用符号文字拓展了作品中的西式风格，进而创造了复杂而完整的视觉形象。

作品中第一处使用符号字体的地方是它的图标（图5-4-2）——在一系列西式字体的衬托下，一张日式阁楼的照片被置于椭圆的框架之中，周围同样环绕着西式的花体装饰图样，既凸现了东西方风格的差异，同时也确立了整个作品的形体风格。

而在其下的五星徽章也是西方人熟悉的视觉符号。在各个链接列表中，标题都点缀了五星符号和符号字体组成的分隔条（图5-4-3）。

若将视线移至设计的主体，我们还可以发现一些更加有趣的元素——两名日本武士和一坐日式盆景居然变成了标题上的装饰（图5-4-4）。

图5-4-1　符号字体往往遵从某种主题

图5-4-2　Dead or Alive中装饰效果极强的LOGO

图5-4-3　链接列表中的标题用五星和符号字体组成的分隔条装饰

图5-4-4　最上端的标题装饰有武士和盆景的形象

在随后的标题中，西方样式成为主导，我们可以看到正文的多个标题都使用了人们熟知的手指符号（图5-4-5）。

SO WHAT IS THIS ABOUT ?

图5-4-5　段落标题使用了西式风格

另外的两处装饰是在作品的顶部和底部类似钱币纹理的图样（图5-4-6）和一个写有“mikepick.com”的邮戳（图5-4-7）。

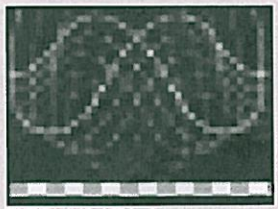


图5-4-6　重复的图样带来了很强烈的视觉吸引力（图像已被放大）

图5-4-7　邮戳体现出水印效果（图像颜色已被加深）

若是没有了装饰和符号的恰当运用，要体现出这部作品的独特风格将会变得很困难。而恰如其分的修饰点缀也的确在强化这部作品的设计效果和传达作品理念中起到了独到的作用。

5.4.2　文字排布

文字排布和页面元素的位置关系密切，先前我们已经有了很多相关讨论。不过在视觉元素布局的发展过程中，还有着更多科学理论和奇闻轶事。

1. 视觉行为

研究表明，人类眼睛使用非常特殊的方式进行阅读。因而排版专业人员推荐我们遵循人们的阅读方式进行页面布局（除非你要实现的是一种错乱的风格）。

在从左向右阅读的语言中，人们遵循一种特殊的模式阅读文字，这在纸面和网页设计中都是一样的。

排版的主流理论认为人类视觉会自然地偏向于页面的左上角，也就是所谓的视觉中心区。之后视线可以顺畅地向右移动，或是转向新的一行，这样逐渐趋近于页面底端。在这个过程中，视线会将页面的底端作为最终的停靠点。在日常的设计中，设计师们都会遵循上述视觉原理（图5-4-8）。

Note

在非自左及右阅读的文字，视觉的行为是不同的。例如希伯来文和阿拉伯文是两种自右至左阅读的文字，这时视线一般都会从右上角开始，并逐渐向左侧和下方延展。

2. 使用“引力感”来引导视线

视线被吸引的现象被形象地比喻为“引力”。就如地心的吸引作用那样，视线也会自然地被作品的布局导引。

因此，将重要的站点标志放置于视觉中心区，而把文字等其他元素置于页面下部形成停靠点的设计规则是非常合理的。而视线从右方转向左侧时形成的折点通常也是较易被浏览者关注的地方。于是我们可以得出如图5-4-9中所示的理想化的站点模版，这和Dead or Alive中遵循的规则完全一致。

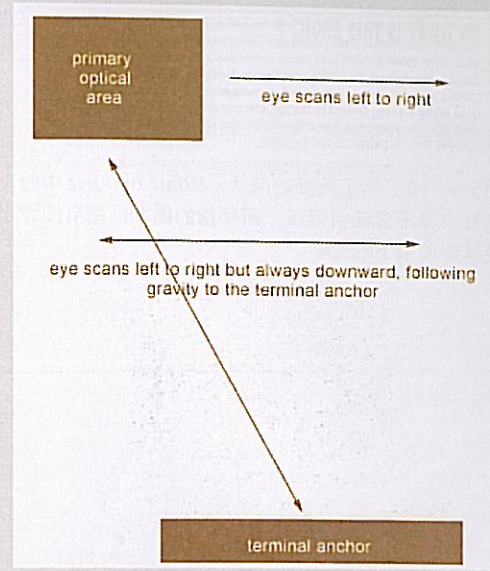


图5-4-8　人们浏览页面时的视觉行为

5.4.3　避免产生理解上的问题

排版中的另一个问题是如何保证人们能正确理解文字所传达的消息。当然，对那些专注于图像而不是文字的实验性作品还需另当别论。

1. 标题文字

标题文字用来指明段落的主要内容。如Dead or Alive中，标题的样式用于向浏览者传达作品的含义，同时也能起到强化作品整体效果的作用。

下面是一些关于处理标题的指导意见。

- □ 标题中使用的颜色必须是整个设计中较为强势的色彩。
- □ 标题的色彩必须鲜明。研究表明，在标题中亮色所带来的影响不如暗色。因而越是接近黑色的标题，越是能对浏览者产生强的冲击力。
- □ 越是重要的标题，越是需要带有强烈冲击力的色彩。
- □ 略为紧缩的字体更能凸现标题。所以设计师建议使用正常字体70%~90%宽的标题文字。
- □ 字体对比可以帮助增强标题效果。例如Dead or Alive中标题都使用了大写文字以区别于普通文字。
- □ 标题长度应该有所限制。过长的标题在视觉和理解上都容易让读者迷惑。

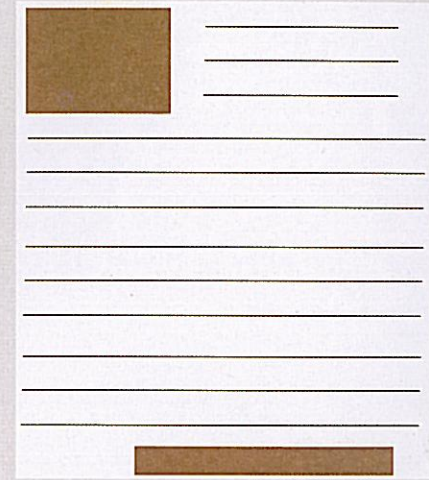


图5-4-9　根据视觉行为得出的页面模版

SO WHAT IS THIS ABOUT ?

There is clearly a need for CSS to be taken seriously by graphic artists. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The code remains the same, the only thing that has changed is the external .css file. Yes, really.

图5-4-10　除了此处的问号，Dead or Alive中的标题中都不含任何标点。而手指的图标则指引读者继续阅读余下的文字

在标题设计中另一个有趣之处是标点的作用，尤其是句号。通常情况下，应该避免在标题中使用句号，因为句号用来结束一段文字，而标题则是用来引出下面的内容文字（图5-4-10）。

2. 正文字体

正文是页面文字的主体部分，浏览者会仔细阅读，或者至少扫过正文中的文字。而不恰当的排版布局将可能错误地传达文字要表示的信息。

以下是关于正文布局的一些建议。

- 保持一行在60个字符以内。虽然在流式布局的网站设计中（尤其是CSS中问题百出的min-width和max-width属性）很难控制每行字数，设计师仍应将这条规则牢记在心。因为过长的一行文字不但会让读者难以理解，甚至会使其根本不愿阅读下去。
- 避免每行字数过少。如果将段落分布于20字的短行中，将导致行数激增，而过长的段落往往同样是人们所无法忍受的。因而最好只在需要强调的文字中使用短行。
- 英语书写的传统认为一个段落至少应由3到4句话组成。然而在Web中，过长的段落却不被看好。较短的段落在快捷的网络时代更适合迅速传达重要的讯息。
- 如果确实需要发布长篇文字，就得尽力让段落保持小巧，并且将长段分割为节并加以标题。这才能更好地引导读者，以便正确地理解文字。
- 若段落之间的距离过大，读者在阅读时会有被打断的感觉，造成不必要误解。
- 避免在正文中使用鲜艳的色彩。与标题一样，鲜艳的色彩在正文中往往会造成负面影响而不是正面的帮助。我们应该只在重点文字和链接上使用鲜艳色彩。在Dead or Alive中，唯一的色彩变化应就在正文的链接上（图5-4-11）。

3. 重要文字

相对于标题和正文来讲，重要文字显得更加灵活，当然，它们也会对帮助读者理解文字内容起到相当大的作用。一般来说，重点文字指的是侧栏、引用和说明部分的文字。在处理重要文字时，我们要注意以下几点。

- 由于重要文字的目的是引起读者的注意，因此为重要文字应用鲜艳的色彩自然没什么问题。但必须保证所有重要文字的色彩保持统一。
- 重点文字应当使用短行，且只能有很少的行数。
- 重点文字不应阻断正文，而应增强正文的效果。如果重点文字扰乱了读者的视觉流向，那也会影响到读者对页面内容的理解。页面插图同样也应遵循这条规律。
- 读者都希望插图附有相应的说明。说明文字能为插图提供上下文，起到帮助读者阅读的作用。

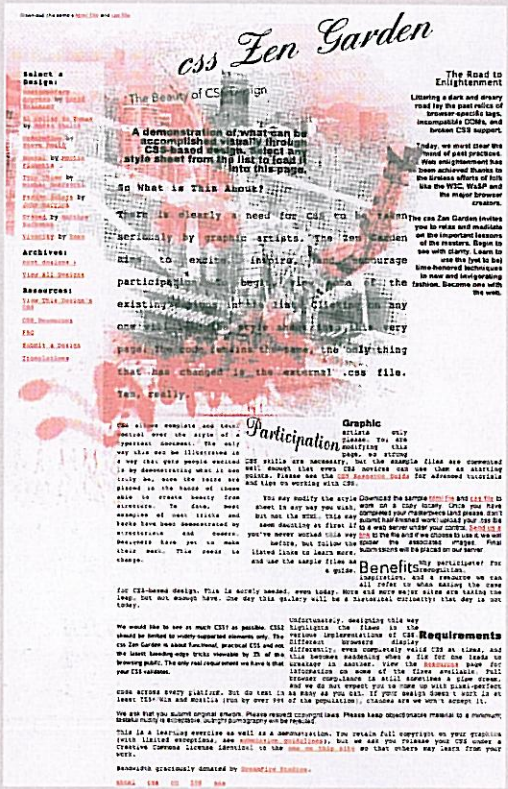
文字排布中另一个涉及可用性的问题是页面中能够容纳多少文字。虽然大多数用户不喜欢点击“下一页”按钮，但若是文字实在过多，设计师还是需要考虑适当地对内容分页。

5.4.4 仔细的排版，优雅的设计

Dead or Alive是一部相当优雅的作品，它的优雅来自于设计师对细节的精雕细琢和对装饰性符号字体的娴熟应用。加上作者无可挑剔的良好排版格式，最终造就了这部简洁但富有视觉冲击力的作品。

Download the sample **html file** and **css file** to work on a copy locally. Once you have completed your masterpiece (and please, don't submit half-finished work) upload your .css file to a web server under your control. **Send us a link** to the file and if we choose to use it, we will spider the associated images. Final submissions will be placed on our server.

图5-4-11　Dead or Alive中正文链接的色彩变化



Dave Shea，设计师

www.csszengarden.com/005

5.5 Blood Lust（血色欲望）

探索文本样式的各种选项

设计领域的不断发展得益于各种不同的因素，而有时这种前进的动力却来自于过去的事物。而本篇中看似混乱无序的Blood Lust正是借鉴了20世纪早期的未来主义设计理念，并用当代的电脑技术将各种过去的设计元素有效地组合了起来。

在设计过程中，Shea将他收藏的图片依次排布于多个图层中，形成了富有怀旧效果的丝质套印背景，在过去这种风格曾非常流行。而在背景之上则使用了黑、红、白三种色彩的组合，并没有太多的其他颜色。此外，为了增强锐化效果，Shea也避免了一切柔化操作，而是使用GIF抖动图样替代，这也为作品增添了一丝现代气息。

5.5.1 用字体设置页面基调

Blood Lust中的无序、混乱的效果主要来自文字样式的应用。可以看到作品中的段落使用完全不同的样式和布局，形成了混乱和差异化的风格。而这种不同字体的混杂和非和谐的文字排布也是未来主义设计师们常用的技巧（图5-5-1）。

为了在Blood Lust中创造与众不同的效果，Shea将多种样式的段落放置于不同的位置，形成了虽然混乱却仍能相互有机结合起来的风格。虽然这种效果不便阅读，但看起来却非常有趣——每个段落各不相同的文字间距和字体导致了不同的文字密度，从而让每个段落的基调都各不相同。图5-5-2将页面做了模糊化处理，但即使是在这种情况下，仍可以明显感觉到段落间基调的差异。

而整部作品在很大程度上依赖了基调的变化，以创造出独特的趣味。可以发现，作品中使用的行距、边距和字体调整都能够带来众多视觉上的变化。不过，让我们再回到易读性问题中——图5-5-3中的哪种文字更易辨认呢？这些都是在实现创意时需要考虑的问题。

5.5.2 格式化选项

Blood Lust使用了CSS中大量的文字格式化功能。若掌握了这些功能——即便只是设置字体和大小这样简单的操作，也可以形成各种不同的组合，进而影响到作品的最终效果。

通过阅读CSS说明文档或者一般的CSS书籍或指导材料，足够让我们在技术层面对CSS有一个简要的理解。可事实上，CSS的很多特性都来自于深层次的设计原则，这些原则却是无法在任何一份技术手册中找到的。

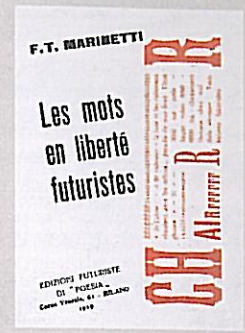


图5-5-1 未来主义设计的一个实例—F. T.Marinetti的“Les mots en liberté futuristes”（1919年）

Note

在排印学术语中，颜色并不单单指色彩的色相，而是指文字在所处环境下的整体基调。它牵涉到文字所在的页面、文字背景与文字主体之间的对比关系。在这层关系中，间距也起到了相当重要的作用，包括文字的笔画间距、字距、词距和行距等。

在多数情况下，设计的目标是保证文字在页面上拥有整体的一致性。虽然能用CSS控制的文字特性并不算太多，但在处理文字色彩之前，我们依然要首先考虑好可读性、每行字数等更加重要的因素。

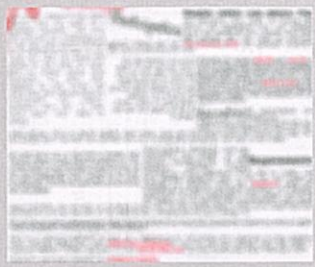


图5-5-2　在模糊处理后，文字块让人感觉出不同的基调

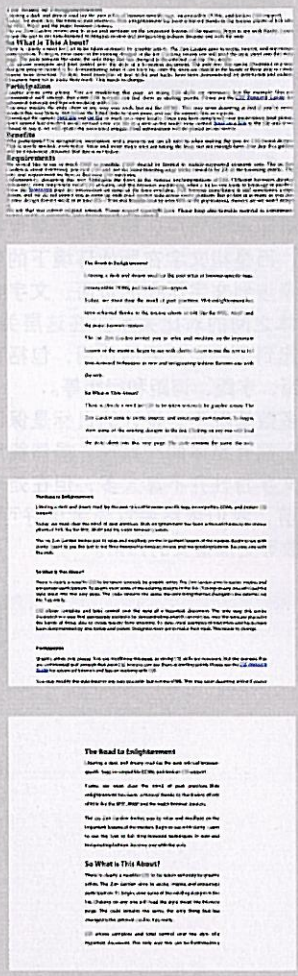


图5-5-3　不同粗细、间距的文字

以下的几页中将阐述CSS中通用的文字格式化属性的定义、来源和使用。当然，某些属性（font-family、font-size、font-style和font-weight）在其他章节已有介绍，因而在此略去。

1. 文本的样式

本段中涉及的属性均用于单个字符的调整。通常来讲，字体应该带有和属性值相对应的变体形式。系统字体列表中会说明每个字体拥有哪些变体，在决定使用之前，要确定浏览者同样拥有相同的变体。

• font-variant

这个属性的唯一作用是将文字用略小的大写字母表示。原先为大写的字母将仍保持原样，而原先小写的字母将由稍小的大写字母来显示。

这个属性的有两个合法值，分别为normal和small-caps。需要指出的是，多数字体没有small-caps变体，因而浏览器会将大写字母缩小。因为浏览器具有这种特性，font-variant也不强制要求字体必须有预设的变体。

font-variant的使用示例：

```
#quickSummary p {
 font-variant: small-caps;
}
```

效果：

The Beauty of *CSS* Design

A DEMONSTRATION OF WHAT CAN BE ACCOMPLISHED VISUALLY THROUGH *CSS*-BASED DESIGN. SELECT ANY STYLE SHEET FROM THE LIST TO LOAD IT INTO THIS PAGE.

DOWNLOAD THE SAMPLE HTML FILE AND CSS FILE

• text-transform

和font-variant类似，text-transform提供了更精确的字符大小写控制。它的可选值有capitalize、uppercase、lowercase和none。

capitalize将每个单词的首字母大写，而uppercase将单词所有字母大写，lowercase则将所有字母小写，none表示默认的正常状态。

text-transform的使用示例：

```
#quickSummary p {
 text-transform: uppercase;
}
```

效果：

The Beauty of *CSS* Design

A DEMONSTRATION OF WHAT CAN BE ACCOMPLISHED VISUALLY THROUGH CSS-BASED DESIGN. SELECT ANY STYLE SHEET FROM THE LIST TO LOAD IT INTO THIS PAGE.

DOWNLOAD THE SAMPLE HTML FILE AND CSS FILE

• text-decoration

自Web呱呱坠地的那一天开始，下划线就一直被用于表示链接。链接中的下划线也可通过使用text-decoration属性获得更多样式选项。text-decoration的可选值有none、underline、overline、linethrough和blink。其中none表示文字没有修饰，underline会为文字添加下划线，overline则会显示出上划线，line-through会显示为中划线（删除线），而blink将产生时隐时现的效果。

text-decoration同时也是最有争议的格式化属性之一，运用不当将让用户觉得相当不便。这里我们列出各个值的推荐用法：

none——默认值，在没有理由使用其他选项时，none就是最好的选择。

underline——仅在链接中使用underline。我们可能经常下意识地去点击带有下划线的文字，却困惑地发现它们并非链接。从这一点中就可以看到，带有下划线的非链接文字会对浏览者造成误导。虽然下划线在印刷设计中还用于书本的标题，但在网站设计中，下划线等同于链接却已是人们根深蒂固的观念，绝对不要尝试去打破。

overline——在数学中，上划线表示平均值，但其他领域却很少使用。

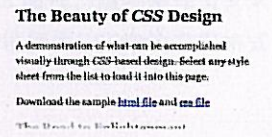
line-through——line-through仅用于表示已经不再相关、非事实或无效的文字，这些文字可能会很快被其他文字替代。理想的用法是用于那些需要被改写的部分——例如涉及版本跟踪和版本历史的文字中可以经常使用line-through。

blink——除非真的很想模仿录像机中闪烁的“12：00”，否则最好当作世上没有blink这个选项。在Web应用中，blink>标签早已被废弃，而且CSS中的blink值也并非所有浏览器都能支持。

text-decoration的使用示例：

```
#quickSummary p {
 text-decoration: line-through;
}
```

效果：



Tip

CSS说明档中更加详细地解释了每个字体样式，请参考“15 Fonts”（www.w3.org/TR/REC-CSS2/fonts.html#font-styling）。

Note

这里所述的所有属性都有一个通用的选项inherit。inherit将让该元素使用其父元素相同的属性值。这个规则对所有的CSS属性都有效，所以我们并没有在介绍属性时重复列出。

2. 间距的样式

我们在很多例子中都能看到背景对文字的影响。因此，页面中的间距也能成为一种有效的视觉工具。CSS中自然也提供了各类控制间距的属性。

• line-height

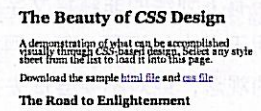
数世纪前，印刷师使用可延展的铅条控制两行文字间的距离，这也是行距（leading [1]）这个专有名词的起源。CSS提供了简单易用的行距控制属性line-height。该属性可被设置为绝对数值、相对大小、长度或者百分比，当然也可以设定为inherit。

line-height不需要单位。1.2和1.2em或者120%是同义的——也就是说未加单位的值表示基于当前文字大小的相对值。

line-height的使用示例：

```
#quickSummary p {
 line-height: 0.8;
}
```

效果：



• letter-spacing

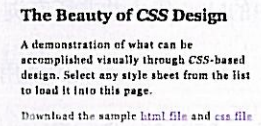
字符间距可使用letter-spacing来控制。我们可以将其设定为normal或一个长度值，该长度值将作为默认值的补充，添加到正常的字符间距上。

由于屏幕的分辨率较低，所以我们不应为大段的文字设置字符间距，否则文字将显得很长。因此最好只在标题和小段文字中使用letter-spacing。

line-height的使用示例：

```
#quickSummary p {
letter-spacing: 0.1em;
}
```

效果：



Note

著名的印刷大师Fredric Goudy曾说过，“对小写文字设置字符间距如同盗窃[2]”。在使用字符间距时，我们必须格外谨慎。

[1] lead在英语中也解释为铅。——译者注

[2] 对小写文字设置字距会使文字变得更加分散，这无疑浪费了有限的空间。——译者注

• word-spacing

和letter-spacing类似，word-spacing用于调整单词的间距。该属性可被设置normal、长度值、百分比值或者inherit。

需要注意的是诸如文字对齐（使用text-align）和white-space等设定均会影响到浏览器对word-spacing的处理。

word-spacing的使用示例：

```
#quickSummary p {
 word-spacing: 1em;
}
```

效果：

The Beauty of *CSS* Design

A demonstration of what can be accomplished visually through *CSS*-based design. Select any style sheet from the list to load it into this page.

Download the sample html file and css file

• text-align

text-align用于对齐文字。可选值为left、right、center和justify。

传统的对齐技术一般使用添加连字符等方式避免折行所造成的空隙，而浏览器中的调整则相当的不精确，其结果常如图5-5-4那样存在大量空隙。考虑到浏览器的不精确控制，我们更建议使用传统的right-align和left-align方式。而居中对齐则只适合标题、引用等小块文字，不要将其应用到长段文字中。

text-align的使用示例：

```
#quickSummary p {
 text-align: right;
}
```

效果：

The Beauty of *CSS* Design

A demonstration of what can be accomplished visually through *CSS*-based design. Select any style sheet from the list to load it into this page.

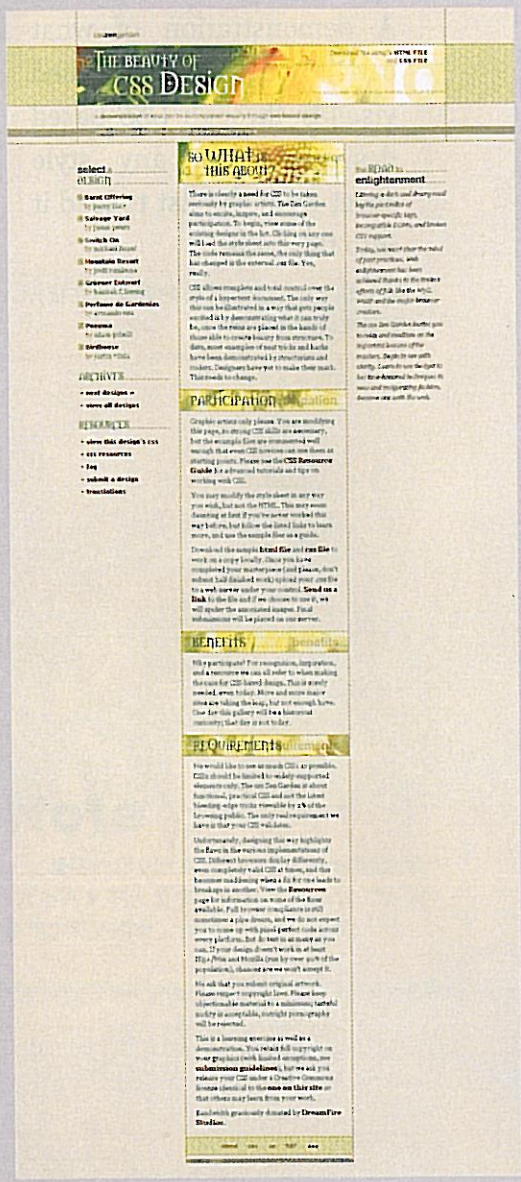
Download the sample html file and css

5.5.3 万事皆有度

虽然随心所欲地调整并混合不同的文字格式能创造类似Blood Lust那样的艺术效果，可真正的Web设计作品首先是用来让人阅读，而不是作为艺术品膜拜的。了解了这一点之后，相信你一定会恰当地使用CSS中的文字格式化属性，力求创造出清晰、合理的文本布局。

A demonstration of what can be accomplished visually through *CSS*-based design. Select any style sheet from the list to load it into this page.

图5-5-4　缺少连字符的调整将在折行时带来大块的空隙



Douglas Bowman，设计师

www.csszengarden.com/017

5.6 Golden Mean

（黄金平衡）

将方法和创意结合起来创造文字排印的平衡

虽然Golden Mean并非完整的网站项目，但Douglas Bowman在回忆其创作过程时说，他仍然按照了真实世界中的大型网站的制作流程完成了整个作品的创作。

作品本身也印证了一个道理——文字排印和图形设计一样，都是网站设计中重要的组成部分，而站点设计本身又是对创意、技巧和流程的有机结合。

5.6.1 选择字体

Bowman首先对作品的内容和含义进行了仔细的考虑。为了验证创意的可行性，他特意勾出了一幅页面的草图，在确信作品可行后才深入到页面文字的设计中。

与图像和色彩的运用相比，虽然字体的选择显得略微次要一些，但有经验的设计师却认为设计的成功与否取决于诸多的因素，而其中一个很重要的因素就是字体。字体往往能够给浏览者一些暗示，情绪就是其能够暗示的一种，常见的情绪有：

- 幽默；
- 淡泊；
- 狂野；
- 怪诞；
- 忧郁。

如果能够在作品中加入一种情绪，那么也可以用一种相应的字体将其表达出来。考虑到这个重要因素，设计师完全应该在选择字体时更加仔细，力求表现出更多的灵感和创意。

1. 在Illustrator中试验

为了确定合适的字体，Bowman打开了Adobe Illustrator，寻找能够与作品优雅格调相配的字体。

图5-6-1是Bowman当初试验时的一张截图。要在其中找寻找到能同时表达出机敏、趣味的独特设计也花费了不少的时间。关于这张图，值得我们借鉴的是，虽然Bowman仅从中挑选出了将用于“CSS Zen Garden”和“The Beauty of CSS Design”两段文字的字体，但像这样对字体列表的预览却帮助他把握了整个页面的风格。

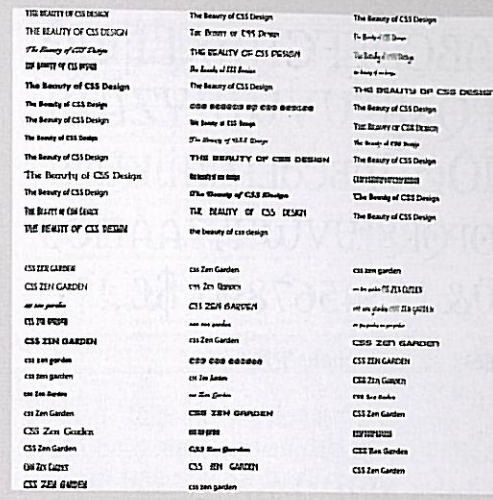


图5-6-1　在Golden Mean的创作过程中，设计师尝试了多种字体

Note

Illustrator是一款矢量作图软件。许多设计师都习惯在Illustrator中作出草图，再由Photoshop像素化，并最终整合到页面中（Illustrator和Photoshop都包含于Adobe的Creative Suite软件包中）。而Macromedia的对应产品是Freehand [1]。关于这些软件的信息，请参见www.adobe.co m/pro duc ts/creativesuite/main.html 和 http://w ww.adobe.com/products/studio/ [2]。

[1] Macromedia公司与Adobe公司已合并，Freehand现在已是Adobe公司旗下的产品。——译者注

[2] 原始链接www.macromedia.com/software/studio已失效。——译者注

图5-6-2　Morpheus字体

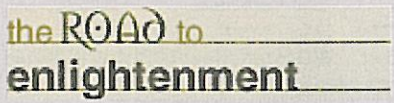


图5-6-3　标题中Morpheus和Helvetica字体的组合

图5-6-4　链接列表中同时使用了Verdana和Georgia字体。注意到所有的Geogia字体都设为小写，而Verdana则使用了正常字体

Bowman最终选择了Morpheus字体，该字体可在dafont.com网站中下载（http://www.dafont.com/font.php?file=morpheus [1]）（图5-6-2）。至于为什么要这样选择，他的解释如下：

- 特殊的M和N字型；
- 哥特式的拱形（A和U中特别明显）；
- K和R的自然延展；
- S和小写d的特殊字型；
- 小写i和D、O、Q中起装饰作用的菱形。

2. 制造视觉张力

之后Bowman又试图寻找一种能和Morpheus形成强烈对比的字体，以期形成视觉上的张力感。最后，他选择了Helvetica字体与Morpheus字体组合。Helvetica是一种现代派的sans-serif字体，它和Morpheus的哥特式风格形成了巨大的反差（图5-6-3）。其他带有强烈对比的字体组合还包含链接列表中sans-serif字族的Verdana字体与小写的serif字族的Georgia（图5-6-4）。最后，为了确保链接为小写，Bowman将text-transform属性设置为lowercase：

```
#linkList #lselect a.c:link, #linkList #lselect a.c:visited {
  display:inline;
  font-family:Georgia,Serif;
  font-weight:normal;
  color:#616623;
  background-color:transparent;
  text-transform:lowercase;
  }
```

需要注意的是，text-transform的可选值还包含将首字母大写的capitalize和将所有字母大写的uppercase。

链接列表中的Georgia字体常用于正文（图5-6-5），它的斜体则用于导言。这两种变体的运用也为作品添加了新的变化。

[1] 原始链接——Kiwi Media上的www.kiwi-media.com/fonts.html已失效。——译者注

3. 控制抗锯齿

若仔细观察图片中的文字本身，就会发现其边缘相当的清晰。但为什么放在在网页上之后却有时会变得模糊起来呢？这里我们可以解释一下导致模糊的几个主要原因。第一个种可能是图片的初始设置存在问题，或者缺少合理的优化。而另一个原因则是因为应用了抗锯齿（anti-aliasing）功能。

抗锯齿是图像处理软件的一种功能，它能在锯齿边缘加入半调色彩以达到柔化的效果。排版软件均提供了包括抗锯齿在内的各种强大的优化功能，但设计师若想得到清晰的字体边缘，则必须手动避免抗锯齿效果造成的字体模糊。

这个问题在小号文字中显得更为重要。字体中的笔画和曲线（注意到Morpheus之类的字体中细小的笔画特别多）不能被缩减到一个像素以下，因为抗锯齿功能对小字体会加入更多的半调色，进而造成更严重的模糊。而主流图像处理软件中的紧排（kern）功能则可以微弱地调整字体空隙，达到更好的柔化效果。若想了解抗锯齿和紧排的更多信息，请参考Web Page Design for Designers网站（www.wpdfd.com/wpdtypo3.htm）。

5.6.2 深入理解文字排印

如果上面的介绍让您渴望了解更多的文字排印相关知识，那也没有任何问题！各种图书和网络上都有足够的信息让你获取。

1. 推荐书籍

下次出行时一定要记得在喜爱的书店前留步，只要仔细浏览其中有关设计的书目，就一定会有适合的图书。这些图书可能是经典名著，也有可能专攻字体设计，还可能展示了某位设计师的相关作品。只要像阅读HTML和CSS那样翻阅一遍，即可找到很多全新的方法和创意。但这里还必须提醒的是，一定记得要去看Robin Williams的名著。

- *The Mac Is Not a Typewriter, 2nd Edition*（2003年由Peachpit Press出版社出版）和*The PC Is Not a Typewriter*（1995年由Peachpit Press出版社出版）。这两本中的一本针对苹果用户，而另一本则适合Windows用户，阅读二者都将在网页和印刷设计中受益匪浅。

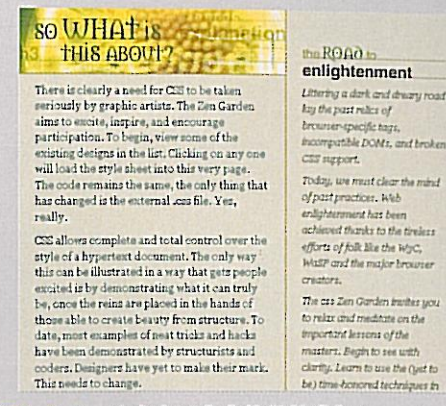


图5-6-5　注意在正文中使用的常规Georgia字体和在导言中使用的斜体Georgia字体，同样需要注意的还有标题中字体的运用

Tip

虽然你可以尝试使用不同的字体，但即便是有经验的设计师也很难在一个页面中运用不同的字体族。因而最好避免使用不同字体族。

Tip

关于这个问题有一篇更深入的讨论，请参考*Type: the Extra Mile*（www.mezzoblue.com/archives/2004/01/18/type_the_ext）。

- *The Non-Designer's Type Book*（1998年由Peachpit Press出版社出版）。无论你现在是一位职业设计师还是一位设计爱好者，这部书都能向你揭示字体设计中的惊人之处。包括设计文档时的技巧，以及突破陈规的各类有效方法等。

如果你是一位高级设计师，则或许会需要更多的字体相关书籍。这里我们列举了一些有价值的参考：

- Rob Carter、Ben Day和Philip Meggs所著的*Typographic Design: Form and Communication, 3rd Edition*（2002年由John Wiley & Sons出版社出版）。本书深入研究了字体的历史由来，也提供了一些通用字型、字体的参考。此书并非为初学者准备，但若想了解排印学的基础知识，那么阅读本书定会大有裨益。

- Robert Bringhurst所著的*The Elements of Typographic Style, 2nd Edition*（2002年由Hartley & Marks Publishers出版社出版）是一本排印学的高级参考。本书堪称排印学的百科全书，因而适合高级排印人员阅读。其中排印理论的实际应用是不可错过的篇章。

- Allan Haley所著的*Alphabet: The History, Evolution, and Design of the Letters We Use Today*（1995年由Pub Overstock Unlimited出版社出版）。此书出自一位排印领域的高产作家。它对字体的历史与字体在当今的使用有着相当独到的见解。而作者Haley的其他书籍也均是通俗易懂的佳作。

- 还可以阅读一些当代的作品，甚至是那些还存有有争议的著作，例如Lewis Blackwell和David Carson所著的*The End of Print: The Grafik Design of David Carson, Revised Edition*2000年由Chronicle Books出版社出版），就囊括了排印领域的众多有争议性的论题，或许能带来一些启发。

2. 在线的字体资源

如果更喜爱在网络上搜寻信息，那必须准备好足够的时间，因为网络上已有太多的相关资源。从网络字体库到对字体历史的介绍和指导，只要在Google中搜索“typography”、“type foundries”等关键词，就会得到一整天都无法看完的信息。这里我们列出了一些最有用的资源。

- Type Designers（http://www.myfonts.com/person/index.html）[1]。Luc Devroye创建的这张冗长的列表使用了A~Z的字母排序，囊括了几乎所有的当代字体设计师以及他们的著作、网站和联系方式。

- Identifont（www.identifont.com）。一个十分有趣的站点，该网站通过询问数个问题帮我们找到所需要的字体，同时也提供了很多其他的字体资源。

- MyFonts（www.myfonts.com）。这或许是互联网上最难看的字体站点了，但是它的内容却相当不错。所提供的搜索功能能灵活地模糊匹配的查询条件，而WhatTheFont工具则能通过上传的截图辨识出其中使用的字体。

- Adobe Type Library（http://store.adobe.com/type）。Adobe公司维护的商用字体库。

- Website Tips（www.websitetips.com/fonts）。这个站点几乎囊括了网络上所有可以找到的字体。它不但提供了共享和免费的字体和外文字体，同时也列出了字体相关的组织和新闻组，以及一系列的字体相关软件。

虽然字体的学习将是一个漫长的过程，但却不应因此忽视它。因为只有良好地掌握了字体知识，才能创作出精彩的设计作品。

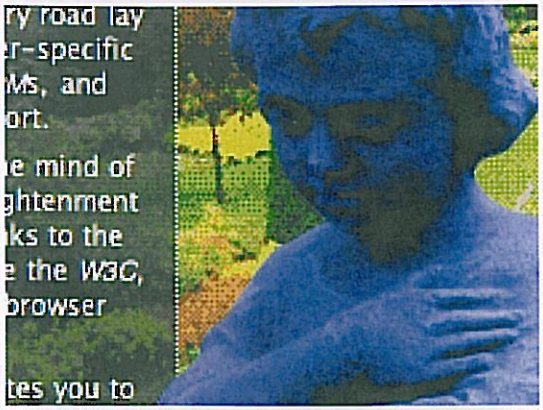
5.6.3 平衡才是关键

亚里士多德的“凡事中庸而行”是智慧的生存之道。虽然在设计中的一个思路是追求极限，但就实用的作品而言，平衡才是成功的关键。

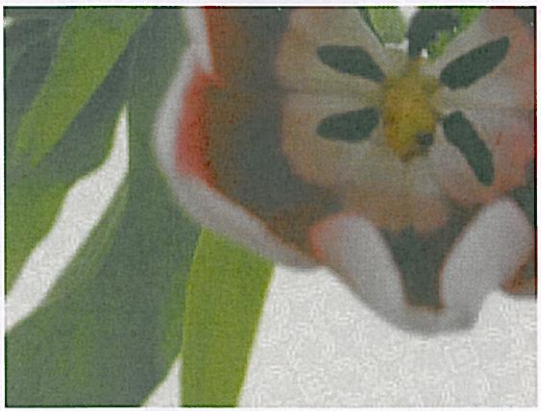
而Golden Mean中选用的那些富有对比的字体，既不脱离文字的实际作用，同时也给浏览者留下了深刻的视觉印象，堪称把握字体平衡的绝佳范例。

[1] 原链接http://cgm.cs.mcgill.ca/~luc/longdesigners.html已失效。——译者注

This is Cereal，192页



Gemination（双子座），198页



Tulipe（郁金香），210页

door to my garden（花园之门），216页

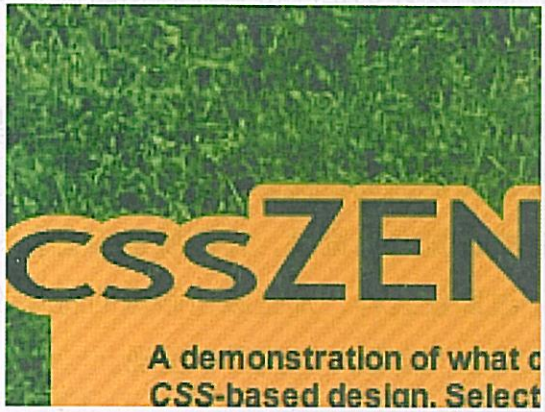
第6章

Bonsai Sky（盆景天空），204页

特效

在本章中，将学会用背景图片、透明和层等技术实现一些先前所无法想象的效果。

本章还将涉及网页设计中的那个最让人困惑的问题——浏览器兼容性问题。希望其中的案例能够帮助理解一些浏览器增强技术、浏览器兼容性解决方案及浏览器管理等相关的知识。



Elastic Lawn（绿草如茵），222页

Shaun Inman，设计师

www.csszengarden.com/057

6.1 This is Cereal

级联和分层效果

Shaun Inman一定非常喜爱均衡的饮食。想象一下这个场景：手捧着一盒“Post Fruity Pebbles”[1]，脑海中充斥着一股模仿的旋风和些许冲破陈规的冲动，Inman最终选择了早餐作为他的作品This is Cereal的基调。当然，借助现代浏览器强劲的特效处理能力，他的设计也正冲击着禅意花园视觉效果的巅峰。

由于目前为止，仍有很多用户正使用着那些不支持CSS高级功能的浏览器，所以Inman也竭力保证了在用这类浏览器浏览时，作品质量不会有太多的下降。不过即便这样，那些吸引眼球的特殊效果仍旧只能在那些“高级”浏览器中呈现。

禅意花园是一部收集了各种实验性和创意作品的集合，This is Cereal也如其他所有富有创意的作品一样，与禅意花园的主题衔接得天衣无缝。

[1] 一种零食。——译者注

6.1.1 级联

可以看到，只有在像Firefox这样的对CSS 2高度兼容的浏览器中，This is Cereal的菜单和透明效果才能正常显示。因为使用这类CSS的新增功能要涉及多种CSS过滤器，且还要CSS中“级联（cascade）”这种强大功能的支持。

那么级联样式表CSS中的“级联”究竟是什么意思呢？W3C是这样描述的：“CSS级联为每个样式规则都指派了权重，若元素应用了多个样式规则，那么将优先使用最高权重的样式”。这段定义尽管看上去复杂，但是它包含的两个相互关联的概念——特殊性（Specificity）和继承（Inheritance）将会成为学习和使用过程中的良师益友，当然，也可能变成让人头痛的噩梦……

CSS中的继承（Inheritance）是指只定义一次就能将同一样式赋予多个元素的能力。设想一下，若是没有了这种能力，对于每一种格式，都需要逐一将网页上的元素和CSS定义中的样式对应起来。天啊，这真是个脱离现实、无法操作的想法！

事实上，CSS还能把某一个样式赋予一类或者一组有着相同标准的元素。例如一个针对h3的定义会把此种样式赋予页面上所有的h3元素（图6-1-1）：

```
h3 {
 color: red;
}
```

这样，页面上所有的h3元素都会呈现为红色，是不是很简单呢？但是如何在覆盖某一些h3元素样式的同时又不影响到其他的h3元素呢？其实，我们只要编写一个额外的Class（图6-1-2）即可：

```
h3 {
 color: red;
}
h3.archives {
 color: green;
}
```

前面代码中的第一条规则会让所有h3元素显示成红色，而第二条规则会重写第一条规则，把该规则中的样式应用到Class为archives的那些h3元素上。

Tip

更多的关于特殊性和继承的信息可以在“CSS spec”中“Assigning pro perty values, Cascading, and Inheritance”（www.w3.org/TR/REC-CSS2/cascade.html）一章中查询。

图6-1-1　页面上的h3元素使用红色

图6-1-2　“archires”这个class中的h3元素使用绿色，但所有其他h3元素仍是红色

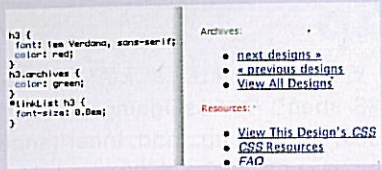


图6-1-3　样式规则的组合产生了字体为0.8em的Verdana的绿色标题

目前为止还没有遇到什么特别复杂的情况。接下来让我们尝试一个更有技巧性的例子：在下面的样式定义中，由#linkList标出的h3元素将会怎样显示呢（图6-1-3）？

```
h3 {
 font: 1em Verdana, sans-serif;
 color: red;
}
h3.archives {
 color: green;
}
#linkList h3 {
 font-size: 0.8em;
}
```

答案是它们将显示为红色，字体为0.8em的Verdana（或备用的sans-serif）。因为最后那条规则中没有覆盖字体设定，所以Verdana字体会从第一条关于h3的样式定义中继承下来，因此发生变化的只有字体的大小。但若是某个h3元素应用了名为archives的Class，那么其颜色将变为绿色。

1. 特殊性

前面的那个例子生动地说明了特殊性的行为——特殊性让最后一个规则比第一条规则有更高的优先权。因为其中定义了额外的Class，让整条规则比第一条更加精确，从而也就有了更高的优先权。

我们知道一个h3元素又不可能同时具有0.8em和1em的字体尺寸，如果CSS中没有特殊性这样的概念存在，那前面例子中的两条规则就必须互相竞争，从而决定究竟该使用哪个字体大小。有了特殊性之后，确定样式的优先顺序就变得非常简单。

尽管特殊性的行为看似简单，可它本身也存在着诸多问题。我们完全可能在不经意之间就造出了一些过于“特殊”的样式规则，若是想要覆盖这些的规则，就不得不通过编写更为“特殊”的规则！

请看下面的这个例子：

```
body>html #header ul.navigation li.home a {
 color: red;
}
```

假设需要把:hover状态的链接元素显示成蓝色，那又该怎么做呢？唯一的办法可能就是复制这一长串的选择器，然后在其后加上限定:hover状态的标记：

```
body>html #header ul.navigation li.home a:hover {
 color: blue;
}
```

• 控制特殊性

只要稍加考虑，我们就能避免写出这类复杂的、过于“特殊”的选择器。简单说来就是让样式规则的定义尽可能地通用化，而只有在迫不得已的特殊情况时再进行特殊化。

例如对于这样一个选择器：

```
#linkList ul li h3 {
 color: red;
}
```

与下面的声明比较：

```
#linkList h3 {
 color: red;
}
```

第一段代码中声明了很多额外的选择器：h3元素必须在无序列表（unordered list）以及一系列冗长的标记中，才能确定将其显示为红色。而第二个例子则摒弃了长串的额外约束，仅仅要求h3元素存在于linkList父元素中即可。使用类似的过程，在开始时以通用的方式定义样式，随后再根据需要逐渐特殊化，即可避免生成过于冗长的样式定义。

• 调试特殊化行为

调试CSS布局问题的一个诀窍就是：保证特殊化导致的冲突不会影响到想要的显示效果。如果某个样式不起作用，那么首先要检查的就是这个样式本身是否足够明确。一个很有用的检查方法就是，临时使用非常易于分辨的样式属性替代原先的样式属性，这样即可方便地确认选择器是否真正指向了期望中的元素。例如：

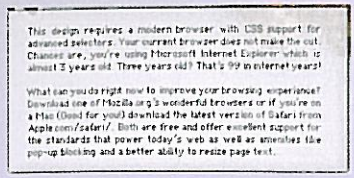


图6-1-4　用户在使用IE浏览This is Cereal时将看到的消息

```
#linkList h3 {
 border: solid 20px red;
}
```

如果h3元素没有出现红色边框，我们就要把选择器写得再明确些。但若是正常显示出了边框，那就不是选择器不够明确的问题。这时即可把该临时边框样式改回先前的样式，然后在其他方面查找问题的原因。

• 覆盖特殊性

CSS对覆盖样式规则的特殊化行为有着内建的支持，即!important声明。!important声明可以被赋于单个样式属性，从而保证即使样式表中的其他样式的选择器更为明确，该带有!important声明的规则也不会被覆盖。例如：

```
h3 {
 color: red !important;
}
#linkList h3 {
 color: blue;
}
```

即使上述的第二条规则更加明确地指向了#linkList元素中的h3元素，该元素仍将显示为红色，这正是因为先前的那条规则中的!important声明让CSS忽略了特殊化作用。需要注意的是Windows平台上的IE浏览器并不支持该!important声明，所以不要尝试在IE中使用。

然而在CSS的调试过程中，IE的这个缺陷却成为了一个优点。使用!important声明有选择地输出CSS样式将会为非IE浏览器提供一种强大的过滤方法。例如在前面的那个例子中，h3元素将在Firefox、Opera和Safari等浏览器中显示为红色，而在IE中显示为蓝色。

This is Cereal中大量地使用了级联特性，用来弥补某些浏览器中部分特殊效果的缺失。事实上，Inman对样式做了大量的特殊考虑，甚至当用户使用某些对特效支持不够的浏览器时，将会收到一条“唐突”的消息（图6-1-4）。

6.1.2 分层

This is Cereal中的所有元素（包括那条将在IE中显示出消息）都使用了绝对定位，除此之外，作者还特地小心设定了层的顺序以避免元素相互覆盖。

CSS中的z-index属性能够有效地控制绝对定位元素的层叠顺序。由于绝对定位元素不属于页面的常规文档流，所以默认的控制层叠方法已经不能满足需要。若是没有了z-index，绝对定位元素将自下而上层叠，因而在HTML中先出现的元素会被堆叠于底部，而后出现的元素会被堆叠于上部。

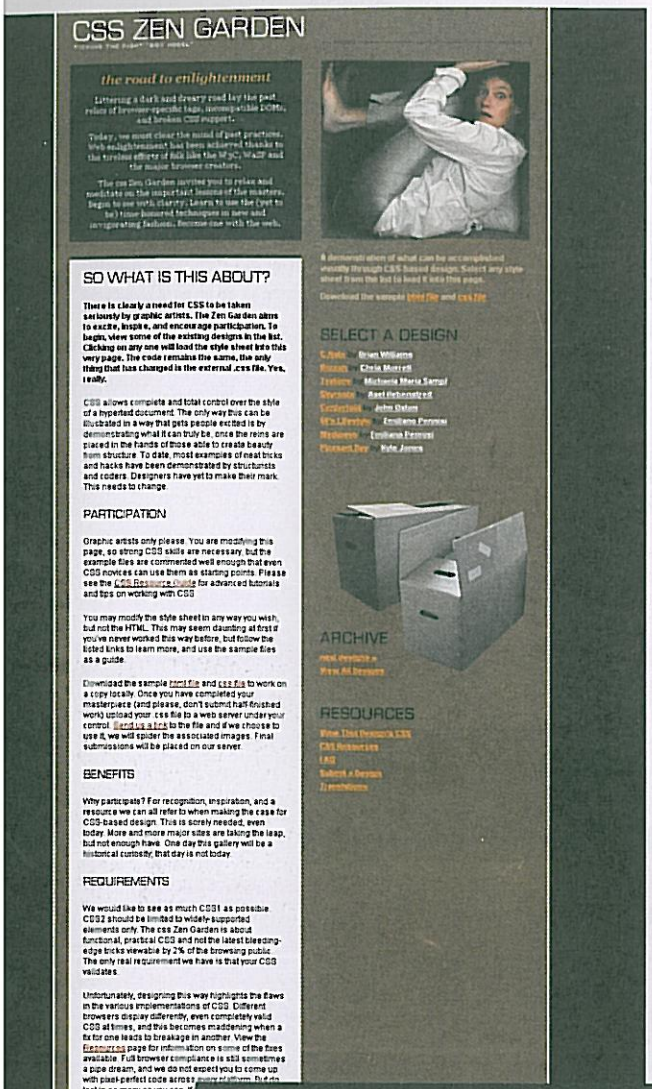
通过使用z-index，设计师可以以任意的顺序堆叠页面元素。z-index中合法的值是从0开始的任意整数。例如：

```
#linkList h3 {
 position: absolute;
 top: 50px;
 left: 50px;
 z-index: 10;
}
#linkList ul {
 position: absolute;
 top: 50px;
 left: 50px;
 z-index: 20;
}
```

因为h3元素的z-index较低，所以不管其声明在HTML源文件中的何处，在页面中都会显示于ul元素之下。

6.1.3 结合

图形设计师应该很熟悉CSS中的分层——各种图像处理软件中都有类似的概念。而级联的概念则更像是一种程序设计概念，无法映射于设计领域中。但一旦我们牢固掌握了这个概念，即可有效地控制页面元素之间的关系，进而实现优美的页面布局——就像This is Cereal那样。



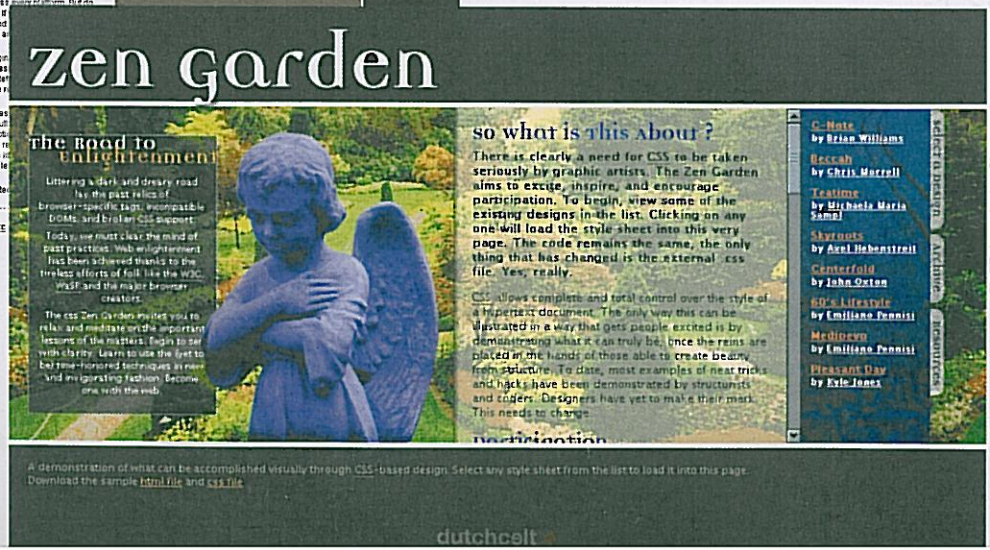
Egor Kloos，设计师

www.csszengarden.com/062

6.2 Gemination(双子座)

基于同一样式表的两种设计

禅意花园的自由风格让Egor Kloos有机会证明CSS是一种能够实现任何创意的有效工具。Egor对CSS浓厚兴趣让他开始学习，并最终能够娴熟地控制浏览器，从而造就出了Gemination这部用一张样式表表现两种迥异风格的作品。在创作中他也体会到了一些自己的想法——那就是聪明地运用代码可以成就让人惊艳的艺术作品。



6.2.1 MOSe

早在Gemination之前，人们就已经讨论过了通过强化CSS支持而改善页面设计的想法。这个想法本身非常简单：首先发掘出浏览器对特定CSS选择器支持的缺陷，然后基于这些CSS选择器分别为每种支持它的浏览器（包括Mozilla及其衍生的Firefox、Opera和Safari等）编写不同的CSS代码，以实现不同的设计。

MOSe特别关注那些IE不能支持的选择器。其中最常见有子选择器（child selector），邻接选择器（adjacent sibling selector）和属性选择器（attribute selector）等。

1. 子选择器

顾名思义，子选择器用来选择给定元素的子元素。它的行为有些类似后代选择器（desendant selector），但是后代选择器却能够选择父元素的任意子孙元素，即便该元素已经嵌套于更深层次的子元素中。

若想编写子选择器，我们可以先写出父元素，然后加上“>”连接器（combinator）（连接器是指那些用来连接选择器的符号），最后添上子元素。例如：

```
div#content > p {color: orange;}
```

基于上述CSS代码，所有#content元素的直接子段落元素都将显示为橙色，如图6-2-1所示。

2. 邻接选择器

邻接选择器用来选择元素的兄弟元素。具体做法是：首先定义第一个元素，然后使用“+”连接器，最后定义邻接元素完成样式规则，例如：

```
div#warning p + p {color: red;}
```

Note

术语MOSe是"Mozilla Opera Safari enhancement"的简称。请参考www.mezzoblue.com/archives/2003/06/25/mose。

Note

子选择器只能够选择父元素的直接子元素，而不会选择父元素所含嵌套结构中的更深层的子孙元素。

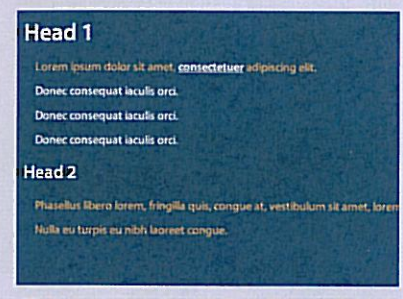


图6-2-1　注意，子选择器只能将样式应用于元素的直接子元素上，而不会应用到所有子孙元素上

Note

关于CSS选择器，请参考OPAL Group站点（http://gallery.theopalgroup.com/selectoracle）以获得详细信息。

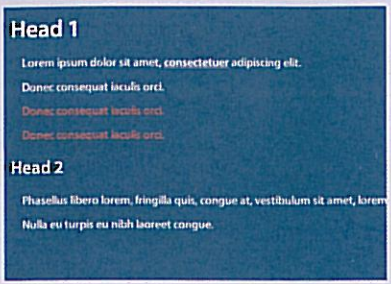


图6-2-2　注意，邻接选择器选择了第一个元素的所有兄弟元素

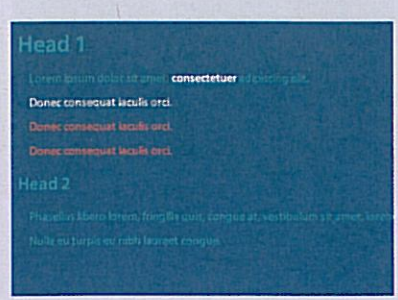


图6-2-3　在支持属性选择器的浏览器中，该选择器的作用结果

这条规则会让第一个在#warning区域中的段落元素的所有兄弟段落元素都显示为红色。如将这条规则应用于前面的那段HTML标记中，显示结果将如图6-2-2所示。

3. 属性选择器以及模式匹配

有趣的才刚刚开始！或许因为IE不支持属性选择器的缘故，导致很多朋友都对这个概念不甚了解，这也让很多网页设计师根本没有机会使用这种特性。但不可否认的是，它所提供的选择功能确实非常强大有效。

属性选择器允许我们根据元素所应用的属性来选择特定元素，且更为强大的是，它能仅选择含有某个特定属性值的元素。譬如对于一个特定的URL链接，可以根据元素是否存在href属性来选择此元素，甚至还可以根据此href属性是否为某个特定的值来决定是否选择。

考虑如下CSS样式：

```
[id] {color: teal;}
[id="warning"] {color: red;}
[href~="http://www.molly.com/"] {text-decoration: none;}
```

图6-2-3演示了这些选择器是如何依据元素的属性值来更改其显示样式的。

Kloos在Gemination中成功地为IE和其他兼容浏览器[1]分别定制了两套截然不同的风格。在实现过程中，他主要借助的就是属性选择器。

6.2.2　CSS签名（CSS Signatures）

CSS签名正在当前的网站设计领域中变得越来越流行，它的主要思想是为页面的body元素明确指定id，这样即可用这个额外的样式来统一站点风格，定义用户样式表（user style sheets），以及实现多CSS样式文件（multiple CSS files）。

CSS签名的真正益处在于，它允许一些资深的用户能够自行调整站点的风格。例如某个经常访问的站点的使用了9个像素大小、又近乎白色的字体，

[1] 指支持前面所述3种特殊选择器的“MOSe”浏览器。——译者注

让人觉得很难分辨。以往浏览者只能发信给站长，建议他修改站点的字体以便阅读。但有了CSS签名之后，用户只需在浏览器中添加一张自定义的样式表，并在其中自行更改字体配置即可达到修改网站样式的目的。

Gemination中就演示了禅意花园中对CSS签名的应用：

```
<body id="css-zen-garden">
```

为了优化设计，Kloos改写了带有CSS签名的body元素的样式。若仔细观察Gemination中的CSS样式，将会发现一些并不常见的CSS语法：

```
body[ id=css-zen-garden] a:link   { color: #f90; }
body[ id=css-zen-garden] a:visited { color: #f90; }
body[ id=css-zen-garden] a:hover,
body[ id=css-zen-garden] a:active  { color: #fff; }
```

通过上面的这几行语句，Kloos定义了带有禅意花园CSS签名的页面中所有a标签的样式。虽然只是短短数句，却有效地保证了带有签名的body元素的所有子元素都将应用相同的样式属性（图6-2-4）。

6.2.3 为IE实现独特的样式

想让IE显示出完全不同的风格也非常简单，只要写出IE能够理解的样式即可。

以下是IE版的CSS样式定义：

```
a:link        { color: #f90; }
a:visited     { color: #f90; }
a:hover, a:active      { color: #fff; }
```

可以看到，这几行样式定义和先前的定义完全一致，区别仅在于IE能够理解这样的语法，显示效果如图6-2-5所示。

6.2.4 改进后的Gemination

改进后Gemination的效果令人称道，其改进过程类似于设计师在Adobe Photoshop中的常用创作方式——首先在背景中使用了重复的图案，然后再在前景部分中添加不同的设计元素。

Note

MOSe浏览器还支持CSS 3.0中的一些新增的选择器，IE自然仍无法理解。若想了解更多MOSe的示例及应用，请参考www.mezzoblue.com/archives/2003/06/25/mose。

图6-2-4　Gemination中改进后的链接的行为

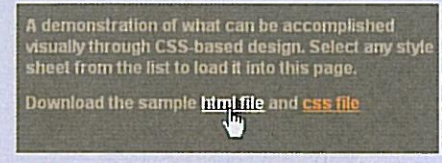


图6-2-5　未改进版本的Gemination中链接的标准行为

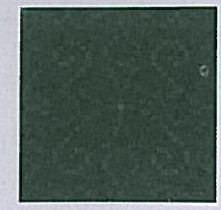


图6-2-6　从squidfingers.com上取得的重复图案构成了Gemination的背景图像

图6-2-7　优化版Gemination中的背景图像

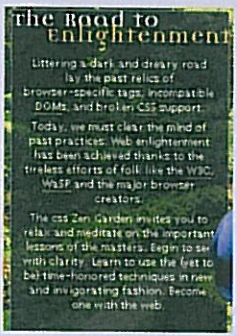


图6-2-8　应用透明背景，并将图像分层叠加显示的效果

首先，Kloos为body元素设置了一个重复的背景图案（图6-2-6）

相应的CSS样式：

```
body[ id=css-zen-garden]
{ margin: 100px 0 0 0; padding: 0; text-align: center;
background: transparent url(squidback.gif); }
```

然后再用#container规则为正文区域应用背景图像（图6-2-7）。

在序言部分，Kloos使用了浅灰色的背景层，透过该背景层还可以隐约看到底层的图片（图6-2-8）。

相应的CSS样式：

```
body[ id=css-zen-garden] #preamble { position:
absolute; top: 30px; left: 20px; display: block; margin:
0; border:
1px dotted #fff; padding: 0; width: 196px; height:
290px; background:
transparent url(blk35.png) repeat; overflow: hidden; }
```

同样使用该样式的还有正文部分，该背景区域部分还额外添加了一个滚动条。最后，Kloos又用:hover选择器给页面最右侧的导航标签实现了一个非常漂亮的图像翻转效果（图6-2-9）。这种翻转效果只能在支持将:hover应用到任意元素上（IE只能将其应用到链接元素上）的浏览器中正常显示。且可以肯定的是，支持高级MOSe选择器的浏览器都能够正常显示该效果。

对于IE版本的Gemination，使用更多的是传统的CSS设计风格，因此也不没有太多的“特效”值得一提，其中唯一亮点是个幽默的比喻——作品中一语双关地引用了CSS中的盒模型——图中的那个面色惊恐、蜷缩在箱中的女人暗喻了IE 5.x系列浏览器对标准CSS盒模型的歪曲，同样也是一个对IE浏览器拙劣兼容性的有力批评（图6-2-10）。（盒模型是浏览器用来计算元素显示位置的一种视觉模型，详细信息请参阅www.w3.org/TR/REC-CSS2/box.html）

6.2.5 能经受住未来考验的设计

在如今的网站设计领域中，人们经常讨论的一个话题就是如何保证作品能够符合未来的变化趋势，也就是如何让今天成功的作品在未来新技术下依旧能完美显示。

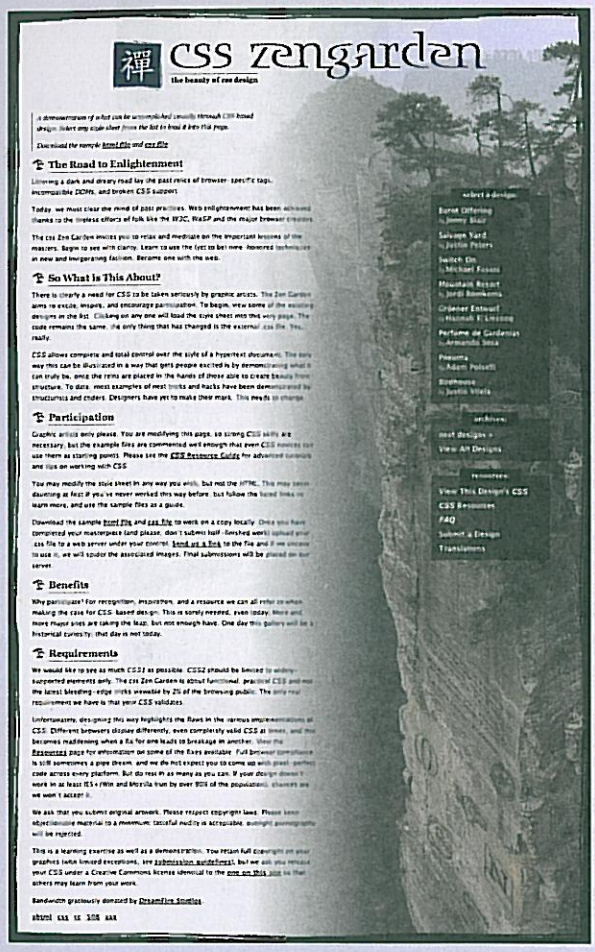
当然，Gemination整个的设计理念如同所有基于MOSe的作品一样，都依赖于IE对CSS 2.1规范中选择器的贫乏支持，而这种支持又是各种其他的当代浏览器所已经实现了的。那么，假若某天微软的浏览器支持了所有这些特性呢，这将会对作品产生负面影响吗？——从理论上说不会，如果新版IE支持所有CSS 2.1中的选择器，那它所显示的网页将和Mozilla等浏览器中的完全一致，这反倒实现了我们梦寐以求的目标。当然若这样的假设成为现实的话，也会导致浏览器市场的冲突，所以这样的想法也仅仅是臆测而已。

Gemination的作者Kloos同样坚信设计必须经受未来考验的论调完全是某些人过激的想法，未来的变化实在太多，谁又能有十足的把握呢。不过，无论这些论点谁是谁非，我们目前还是能借助IE对选择器的贫乏支持，创作出Gemination这样独特亮丽的双重[1]站点。

图6-2-9　使用伪类（pseudo class）选择器:hover实现翻转效果，无须借助任何脚本

图6-2-10　语双关的“盒模型”

[1] 指在同一站点中为Internet浏览器和MOSe浏览器实现两种不同的样式。——译者注



Mike Davidson，设计师

www.csszengarden.com/069

6.3 Bonsai Sky（盆景天空）

用富有创造性的图像和代码突破浏览器的限制

为了将“启蒙、开导”的主题推向一个新的高度，Mike Davidson在他的Bonsai Sky作品中把禅意花园修建在了无涯的天际。他以山石嶙峋的峭壁为页面核心，加上树木的点缀，最终完成了这部具有挑战性的作品。而在这副以图片为主导的作品中，Davidson首先面临的就是如何处理文字和图片的平衡。

考虑到这个问题，Davidson将悬崖置于页面的右上方，再加以淡出的效果，从而和左端的文字天衣无缝地衔接了起来。文字部分则使用了宽大的字符间距，逐渐地把浏览者引向了悬崖深处在挺拔的峭壁上镌刻有各种浏览器的标志。更让人为之惊喜的是，悬崖上竟还有一个登山者俯正瞰着深渊——这个人就是Davidson——深刻掌握了要诀的他已经到达了CSS领域中巅峰。

6.3.1 版本问题

然而Bonsai Sky中的一些吸引人的图片却不能在某些浏览器中正确显示，原因在于Davidson使用了一些特别的CSS功能，而这些功能却并不是所有浏览器都能支持的。因此，不同浏览器的用户将会看到不同的视觉效果。但Davidson并没有因为浏览器的兼容性问题而选择放弃，他灵活地运用了一些CSS技巧，成功地避开了这些浏览器的问题。

1. 黑色边框的定位

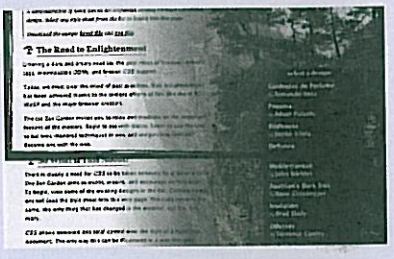
Bonsai Sky中的黑色边框使用了固定（fixed）定位：

```
#extraDiv2 {
  background-image: url(edge_bottom_black.gif);
  background-repeat: repeat-x;
  position: fixed;
  bottom: 0;
  left: 100%;
  width: 100%;
  height: 20px;
  margin-left: -100%;
}
```

固定定位让元素能够精确地固定在页面中的某个位置上，即使用户滚动页面，固定定位元素在浏览器可视区域中的位置仍然不会改变。其显示效果就如Bonsai Sky中的黑色边框一样，俨然悬浮于页面之上（如图6-3-1~图6-3-3所示）。

图6-3-1~图6-3-3　Bonsai Sky中不随页面滚动而移动的边框

可固定定位的功能在某些浏览器（尤其是Windows平台上的IE）中却不能正确地显示——在这些浏览器中绝对定位元素会跟随其他元素移动。不过若是经过了巧妙的设计，该效果不但在支持固定定位的浏览器中能正确显示，在不支持这种功能的浏览器中的效果也能差强人意。



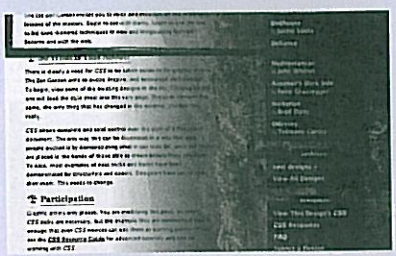


图6-3-4~图6-3-6　滚动时边框将与正文发生重叠

2. 黑色边框的处理技巧

不幸的是，若浏览器不支持的话，仍坚持使用固定定位会让Bonsai Sky中的边框干扰到正文内容的显示（图6-3-1~图6-3-6）。解决这个问题的方法有很多，比如可以使用多种过滤方法隐藏该边框。但是Davison却并没有简单地这样做，而是找到了一个更加有技巧性的方法——他巧妙地利用了IE浏览器的另一个缺陷，把黑色边框整个移出了画面。

Davison为横向伸展的上下边框设置了100%的左偏移量。如此操作之后，这两条边框将位于浏览器右边框的右侧，很明显此时的边框已经在浏览器的显示区域之外了。接下去要把处于显示区外的边框"拽"回来，所以Davison又为其设置了-100%的左外边距（margin-left）值。

```
#extraDiv2 {
    background-image: url(edge_bottom_black.gif);
    background-repeat: repeat-x;
    position: fixed;
    bottom: 0;
    left: 100%;
    width: 100%;
    height: 20px;
    margin-left: -100%;
}
```

由于绝大多数支持负外边距的浏览器都能够成功"拽"回边框，所以在这些浏览器中边框的显示不会有任何问题。而对于那些不能支持负外边距的浏览器，自然无法"拽"回边框，让边框留在画面之外。很幸运的是，有这类缺陷的浏览器又恰好不能正确显示固定布局带来的"悬浮"效果。这样一来，在使用此类浏览器时，原本会被错误显示的边框反而却被很好地隐藏了。

3. 佛像

在处理完边框这个棘手的问题之后，Davidson在场景中加入了另一个视觉上的亮点——一尊安坐于悬空石上的金色佛像。有了先前的成功经验，Davidson再次使用了固定布局技术，把佛像精确地定位在页面的右边缘。

由于使用了类似的技巧，在Windows平台上的IE中佛像的图片也不会出现。因此，Davidson大胆地决定再使用一种IE不支持的效果——透明PNG [1]图像。

如果读者仔细观察，可以发现在悬空石的背景上带有隐约的阴影效果（图6-3-7）。由于有了这层阴影效果的存在，滚动页面时悬空石在视觉上不会混杂于背景的山峦之中，这样一来石像的立体效果就显得格外明显。

这种简单而细微的效果在只支持简单透明度的GIF图像 [2]中是无法良好表现的，而在那些不支持PNG图片的浏览器中则根本完全不能呈现。

虽然悬空石和金色佛像能为整个画面添加了不少亮色，Davidson却觉得作品的"启蒙、开导"主题已经足够完美了，所以他决定只让那些对CSS充分支持的浏览器显示悬空石的效果。因此他又使用了和边框效果相同的方法为悬空石设置了负外边距，这样在Windows平台上的IE这类不支持PNG高级透明的浏览器中，石像同样不会出现。

尽管使用了各种技巧，但原作中想要融入的高级效果仍旧不能在Windows平台上的IE浏览器中呈现，所以Davidson还得努力保证作品的剩余部分也具有较高的品质。当然，若想要得到最好的显示效果，总归需要在支持固定布局和PNG高级透明的浏览器中欣赏。

4. 透明的边栏

在成功地完成了文字和悬崖图片后，Davidson再次陷入了窘境：如何布置"select a design"、"archives"和"resources"这3张列表呢？难道还要像前面对图片的处理那样把它们隐藏起来？

或许最直接的解决办法莫过于把3张列表以透明方式叠加到背景图片上了，但是这次却没法再借助PNG的透明支持了，因为在Windows平台上的

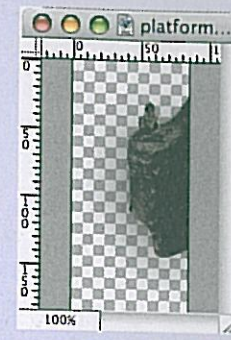


图6-3-7　PNG透明效果——悬空石和佛像后的阴影

[1] PNG格式图片支持alpha通道，可以显示多层级的半透明效果，但是Windows平台上的IE 6.0浏览器尚不能支持。——译者注

[2] GIF图像只支持1位的透明效果，也就是说GIF中的一个像素仅仅存在完全透明和完全不透明两种状态，而PNG图像对透明的支持是多级的，能够很好地显示半透明的过渡效果。——译者注

图6-3-8　放大后的GIF图案类似于一个棋盘，在实际应用中能很好地模拟50%的半透明效果

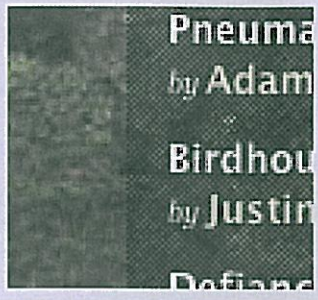


图6-3-9　图6-3-8中GIF图像与背景重叠后的效果

IE中也必须显示出这些列表。Davidson想到了一个设计师中流传已久的诀窍——使用"半"透明的GIF图片重叠于背景之上，若是设计巧妙的话也能够不错地骗过浏览者的眼睛。

半透明GIF的工作原理类似于一个国际象棋棋盘，其中黑色的不透明部分和透明部分交错排列。若平铺在一个较大的平面上，远远看去即可模拟出50%透明的效果（如图6-3-8所示），这类似于喷墨打印机使用稀疏的小黑点进行仿色的原理。

不过屏幕上像素的尺寸比最差的打印机所能绘出的黑点还要大不少，所以稍微留神即可看出这个点阵效果。好在尽管仿真度不高，在该作品中使用还是绰绰有余（图6-3-9）。事实上Davidson把这种棋盘效果定义成了作品中的一个常用模式，正文部分的标题上也都装饰有类似的图案，让整部作品的风格更加趋于协调统一。

5. 特殊技巧

发现固定布局效果的浏览者几乎都会花额外的时间滚动页面，来测试能否正确显示。这或许是浏览者的一种本能——一旦看到了那座浮动的佛像，他们都会迫不及待地把页面翻滚到底部。

作为对他们的奖励，此时浏览者可以看到悬崖上雕刻的各种浏览器的徽标，以及徽标上的一些CSS代码（图6-3-10）。

这些额外的图像可以说是一个新的尝试。很多站点设计师都偏重页面的顶部——由于这个位置最容易被浏览者注意，设计师们也理所当然地把复杂的设计元素都集中在此处。但若是偶尔打破陈规，将重要的元素置于非常规的位置，并由此激发浏览者去寻找它们，也不失为一种推陈出新的好办法。

6.3.2　评论

若某个作品中使用了过多的高级CSS效果，而多数浏览者又无法用主流浏览器欣赏到的话，那作品的定位未免有些曲高和寡。

所以一个很重要的设计原则就是，在尝试新技术时仍有选择地略微降低标准，以期仍能为那些不支持此类高级效果的浏览器提供足可接受的备用方案。

设计师们在挑战CSS效果的极限时的确需要一些运气，好在浏览器兼容性导致的问题也不是不可避免。只要足够谨慎和细致，就能在带给大部分用户新奇体验的同时，也不让小部分人失望而归。

图6-3-10　雕刻于峭壁上的浏览器标志

Eric Shepherd，设计师

www.csszengarden.com/088

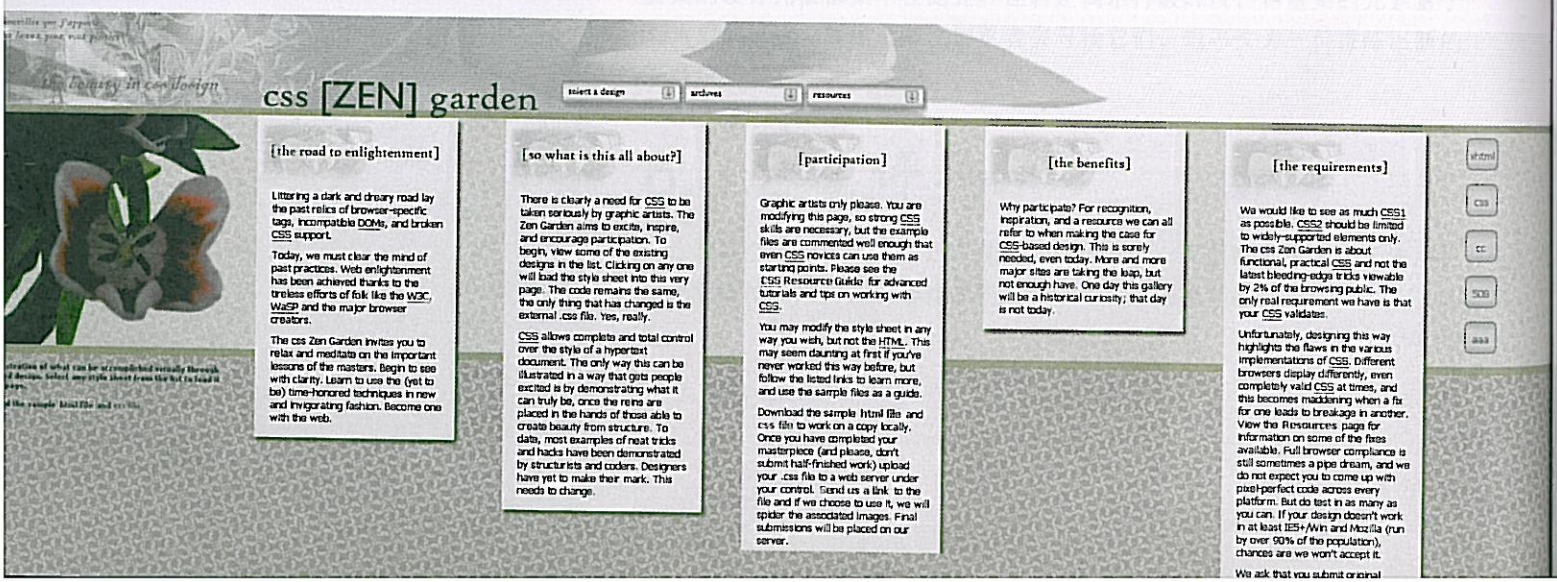
6.4 Tulipe（郁金香）

为IE提供了后备选项的CSS下拉菜单

一枝凋落了一叶花瓣的郁金香随风飘曳，勾起了Eric Shepherd的无限遐思。在摄下了这副图景后，Eric Shepherd大胆使用了通贯全页的横向布局，缀以页面上端的下拉菜单，再配用唯美的色调，最终创作出了这部引人入胜的作品——Tulipe。

尽管在数年前仅依靠CSS创建下拉菜单的方法已经出现于Eric Meyer的站点CSS/Edge上（http://meyerweb.com/eric/css/edge/menus/demo.html），人们还是需要一些额外的技巧让IE也能正确显示出这种效果。

这些技巧中最棘手的问题就是，如何在创造理想设计的同时，也为IE提供一个仍可正常使用页面功能的备份设计。Shepherd在该作品中巧妙地解决了这个问题。



6.4.1 探究纯粹的CSS菜单

创建纯CSS菜单是件让人快乐的事。它不但容易操作，还能使用大量的样式和定位特性。而最让人高兴的是：根本不用再书写任何JavaScript脚本（图6-4-1）！

在下面的代码中，可以看到本作品中下拉表的基本实现。这个实现是最终显示的一个简化版本。仔细观察可以发现，代码并不像Zen Garden中众多div标签嵌套那么复杂，只是一个div标签和一些表示了实际菜单的链接而已。

```
<div id="menu">
<h3>Select a Design:</h3>
<ul>
<li><a href="#">Gardenias de Perfume</a> by <a
href="#">Armando
➥Sosa</a></li>
<li><a href="#">Pneuma</a> by <a href="#">Adam
Polselli</a></li>
<li><a href="#">Birdhouse</a> by <a href="#">Justin
Vilela</a></li>
<li><a href="#">Defiance</a> by <a href="#">James
Ehly</a></li>
<li><a href="#">Mediterranean</a> by <a href="#">John
Whittet</a></li>
<li><a href="#">Austrian&#8217;s Dark Side</a> by <a
href="#">Rene
➥Grassegger</a></li>
<li><a href="#">Invitation</a> by <a href="#">Brad
Daily</a></li>
<li><a href="#">Odyssey</a> by <a href="#">Terrence
Conley</a></li>
</ul>
</div>
```

可以看到，去除样式后这个片断在浏览器（包括IE）中仅仅显示为一个无序列表（图6-4-2）。

虽然这样一段文字在使用起来没什么问题，但它还不够漂亮——添加一些CSS效果就可以解决这个问题。下面的CSS代码指定了body元素和链接元

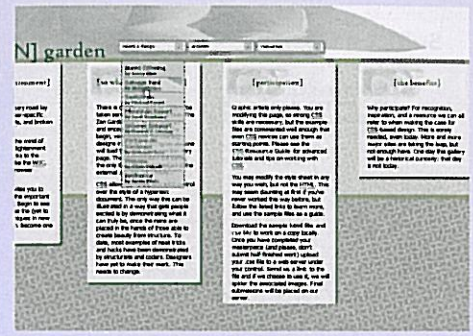


图6-4-1　Tulipe中的下拉菜单

Select a Design:

- Gardenias de Perfume by Armando Sosa
- Pneuma by Adam Polselli
- Birdhouse by Justin Vilela
- Defiance by James Ehly
- Mediterranean by John Whittet
- Austrian's Dark Side by Rene Grassegger
- Invitation by Brad Daily
- Odyssey by Terrence Conley

图6-4-2　未应用任何样式的片断将显示为一个无序列表

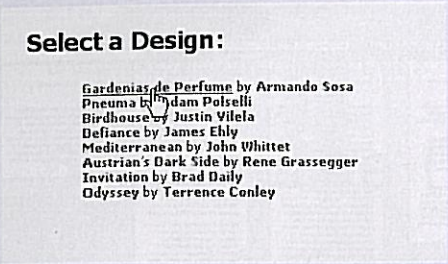


图6-4-3　加入基本样式（包括在IE中也支持的鼠标悬停样式）后的显示效果

素的样式。

```
body {
    font-family: tahoma, arial, helvetica, sans-serif;
    font-size: small;
    background-color: #F2F4EE;
}
a:link, a:visited {
    font-weight: bold;
    text-decoration: none;
    color: #998;
}
a:hover {
    color: #887;
    text decoration: underline;
}
```

在图6-4-3中可以看到列表应用了相应的样式。值得庆幸的是，即使在IE中，当鼠标悬停于链接上方时，下划线也能正常显示。

目前为止，一切都非常好。不过下面的这个步骤却有些玄机：现在需要把ul元素的display属性设置成none。这可以在鼠标未悬停于ul元素上方时隐藏掉该元素。

```
ul {
    display: none;
}
```

在加入这个样式后，列表会在所有支持CSS的浏览器中隐藏起来。要使列表在鼠标移过时重新显现，我们需要再添加一个伪类:hover，然后在其中将display属性设为none以外的某个值。请参考下面的代码（注意其中对ul和li项应用了一些额外的样式）：

```
#menu:hover ul {
    display: block;
    border: 1px solid #776;
    margin: 5px;
    padding: 5px;
    width: 175px;
    list-style-type: none;
}
```

```
li {
    margin: 0;
    padding: 3px;
    border-bottom: 1px solid #887;
}
```

在添加上述样式后，用任意一种能够全面支持:hover伪类的浏览器打开页面，会发现在鼠标悬于列表上方时，列表会从隐藏转为显现。接下来把列表独立出来并继续添加一些额外的样式，Tulipe的下拉菜单就逐渐成型了（图6-4-4）。不过在IE中它一直处于隐藏状态，用户所能见到的始终只是一个标题（图6-4-5）。

图6-4-4　在鼠标悬停时，Mozilla或者Opera之类的高级浏览器能够显示出隐藏的列表

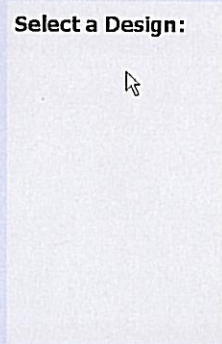


图6-4-5　因为IE只对链接元素支持:hover伪类，所以当鼠标悬停于其上时，不会显示出该列表

6.4.2　Tulipe中的下拉菜单

如果仔细研究过Tulipe运用的渐进改善的创作模式，就会认同前面曾说过的那个前提——HTML列表打造了菜单结构，而一系列的样式设定了鼠标悬停时的行为，并逐渐美化了作品。但又因为IE在整个过程中所表现出的问题，Shepherd必须周密地考虑作品中CSS的运用。不但要能在良好支持CSS的浏览器中展示出光彩夺目的复杂效果，也要在IE中将其隐藏起来，同时使用某种较为简单的样式替代。这样才既能弥补了IE的不足，又不至于破坏整体的设计。

1. 以其人之道还治其人之身

至此，我们已经发现了IE的两个重要的问题。第一是它只对链接提供:hover伪类支持，这也是无法用:hover实现下拉菜单效果的罪魁祸首。第二是它不支持某些选择器，例如子选择器等——我们接下来正是要利用这个缺陷在IE中隐藏该样式：

```
#linkList2 ul > li
```

另外一个IE不支持的伪类是:first-child：

```
#linkList2 #lselect li > a:first-child
```

这条规则在CSS中的解释是：linkList2的子孙节点中所有id为lselect的元素所含的任意一个li元素下的第一个子链接元素应用样式。因为IE不支持这些伪类，所以这条规则在该浏览器中会被忽略。

对上述选择器和伪类缺乏应有的支持可以说是让众多IE下的开发人员最为头疼的地方。不过这种缺失也带来了好的方面，那就是可以利用它隐藏不想要的样式。

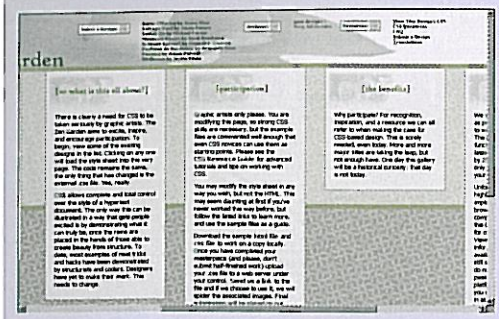


图6-4-6　IE版的菜单。虽然没有下拉菜单那么漂亮，但足够使用

2. 将样式分开

考虑到前面所说的问题，我们在使用绚丽的下拉菜单的同时，也必须要为IE创建备份样式，例如：

```
#linkList2 {
    position: absolute;
    top: 65px;
    left: 750px;
z-index: 2000;
    height: 150px;
    margin: 0;
    padding: 0;
    width: 1350px;
}
#linkList2 a {
    border: 0;
}
```

这仅仅是Tulipe的CSS文件中定义的众多样式中的一小部分。在真正的CSS文件撰写过程中，总是先定义IE所支持的样式，然后在加入其他高级浏览器所特别支持的CSS样式，因为存在子选择器，这些样式也很容易区分：

```
#linkList2 #lselect li > a:first-child {
    display: block;
}
html > body #linkList2 li {
    margin: 0;
    padding: 3px;
    border-bottom: 1px solid #887;
}
```

区分开样式后，Shepherd即可着手在IE中实现可用的菜单了（图6-4-6）。虽然这些菜单不像另一个版本中的那么漂亮，但是它们导航的功能依然完全可用，且足够清晰。

3. 更高级的CSS

在Tulipe的菜单中，还有一项值得注意的CSS特性：

```
#linkList2 ul > li:last-child
```

这里已不仅仅是个普通的子选择器，它还使用了一个定义于CSS 3标准中的伪类:last-child。与:first-child能够选择菜单中的第一个子链接类似，:last-child可以选择菜单中最后一个子链接，从而将其设置为和第一个子链接不同的样式。由于部分浏览器已经实现了这些尚未确定的CSS 3中的新特性，Eric才有机会用它们实现这样让人满意的完美效果。

但在本书编写时，CSS 3推荐标准尚未完成，而且正如前面所见的那样，即使是现在已经完成了的CSS标准，在某些浏览器中的支持仍旧不敢恭维。那么有没有在IE中同样管用的办法呢？回答自然是有，不过这些方法的效果究竟如何，还需要自己去评判。

第一种方法是使用JavaScript创建动态菜单。从本质上说，使用脚本没有任何过错，但是使用CSS实现却不需要考虑诸如客户端JavaScript引擎是否打开，或者CSS兼容性等这样那样的问题，所以CSS的实现方式更为简洁。

另一种方法是使用所谓的.htc文件。这种文件只有IE才能支持。在.htc中撰写的脚本能够改变浏览器的特性。使用这种方法最有名的案例是Dean Edward的“IE7”项目，该项目提供了一系列的.htc文件，用来修正了IE中的部分缺陷并且提供了对高级CSS效果的支持。另外一个关于:hover的修正方法是“whatever:hover”技术，它的实现也是基于.htc文件。

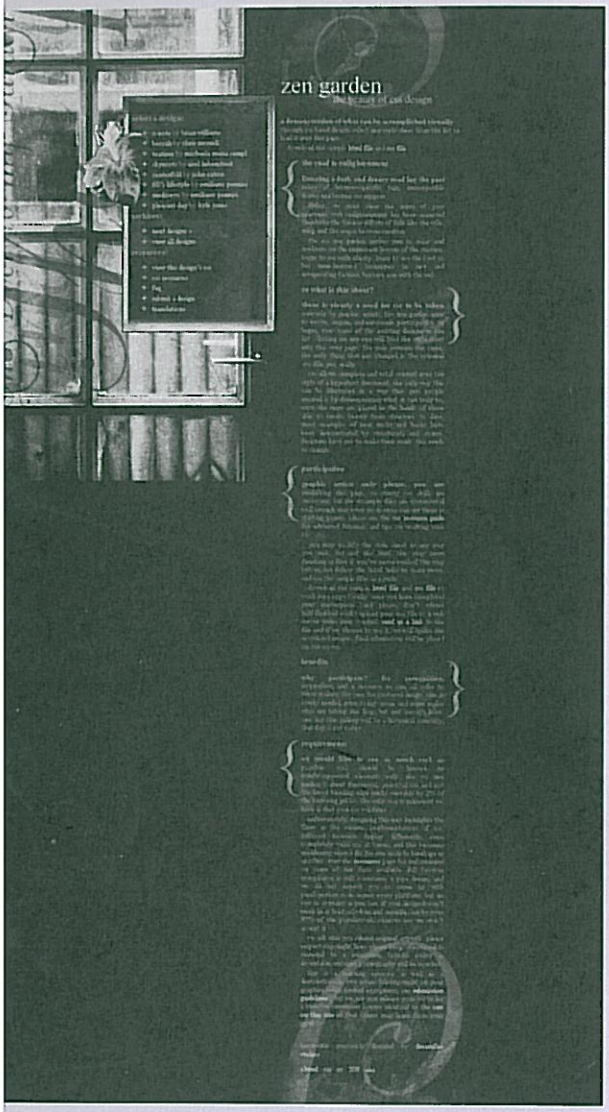
在使用这些方法之前需要牢记这些方法都不受官方支持，所以在使用时一定要慎重。人们一直对是否使用此类方法存有争议。很明显，对标准的支持应该是浏览器内建的而不是外加的，一些纯粹主义者甚至认为所有使用hack、过滤规则的方式都是对标准的亵渎。

6.4.3 从作品中学到的

从起初的灵感乍现，到使用复杂的方法实现CSS下拉菜单（虽然在符合标准的浏览器中这非常容易），再到使用让人困惑的过滤规则等间接方法，Tulipe确实称得上智慧而且负责地使用CSS的典范之作。虽然所有的CSS设计者都渴望无需旁敲侧击即可实现特效功能的那天早日到来，但渐进的设计和后向兼容性的考虑却让我们又不得不必需掌握这些方法。不过在Tulipe的浏览者眼中，无论使用何种浏览器，都能够看到一部足够唯美且完整的作品。

Tip

若想了解更多关于Dean Edward的IE 7项目的内容，请访问http://dean.edwards.name/IE7/。关于对:hover技术的讨论，请访问www.xs4all.nl/~peterned/cs-shover.html。



Patrick H. Lauke，设计师

www.csszengarden.com/041

6.5 door to my garden（花园之门）

用简单的CSS实现完善、震撼的视觉效果

作为一个摄影爱好者，Patrick Lauke喜欢亲自拍摄照片，而不是去借鉴图片库中的资源。他的禅意花园作品door to my garden自然也不例外。在该作品中，Lauke用他拍摄的一张让人沉醉的照片作为了页面的主要背景。

在酷爱摄影的同时，Lauke也痴迷于灵感设计（incidental design），这种设计的方式又被称为拾得艺术（found art），讲求在创作中大量使用偶然发现的设计元素。在本作品中Lauke就细致地运用了他在工作地附近拍摄的锈蚀铁器。借助于他高超的摄影技巧，这些偶得的元素极大增强了作品陈旧斑驳的怀旧风格。

6.5.1 灵巧地应用背景

虽然图片和摄影的运用对营造阴暗寒冷的作品气氛有着不可忽视的作用，但是体现效果的本身却更加依赖于背景图像的制作。当然，禅意花园中的很多作品都可以体现出这一点，但door to my garden却更为清晰地向读者展示了背景定位、滚动行为所能创造出的与众不同的效果。

在本作中，作品主体效果的创制方法非常简单，Lauke仅仅是在body元素中加入了固定背景属性（图6-5-1）而已，且这种方式在多种浏览器下都能正常显示。然后将正文区域布置在页面的右侧，让其与背景图像重叠，这样即可让正文在固定背景上滚动。在CSS设计中处理背景是设计师的一项必备需求，掌握背景样式的属性和它们相应的效果对实现作品的设计风格来讲至关重要。

图6-5-1　body元素的背景图像

1. CSS中的背景属性

加上background的简写形式，CSS中的背景属性一共有6种。表6-5-1列出了这些属性的功用。

表6-5-1　CSS中的背景属性

属性	描述
background-color	设定背景颜色
background-image	应用背景图像
background-repeat	设定背景是否重复，以及重复的方向
background-position	设定背景图像在元素中的显示位置
background-attachment	设定背景图像是否随页面内容滚动
background	上述5种属性的缩略写法

2. 添加背景图片和背景颜色

若想使用CSS添加背景图片，我们要将background-image属性设置为指向图片位置的URL，这个URL可以是本地服务器上的某个相对位置，也可以是指向另一个服务器的绝对路径：

```
background-image: url(images/my.jpg);
```

而若是想设定背景颜色，则需要将background-color属性设置为颜色值（十六进制颜色值、十六进制颜色缩写、RGB颜色或者颜色名称均可）。

```
background-color: black;
```

也可以使用background缩略属性同时定义背景颜色、背景图像以及其他属性：

```
body {
    background: #000 url(background.png)
    -30px 0px no-repeat fixed;
}
```

door to my garden中另外一个常用的效果是将background-color设置为transparent，以便让元素背景变成透明。例如该作品中链接就应用了透明背景：

```
a {
    color: #eee;
    background: transparent;
    text-decoration: none;
}
```

3. 背景的黏附（Attaching）属性

background-attachment属性可以接受scroll和fixed两个值，浏览器的默认值是scroll，也就是说背景图像会跟随页面一起滚动。若是设定为fixed，背景将不会随页面内容一起滚动（图6-5-2）。Lauke设置的作为页面背景的图像就是一个例子：

```
body {
    background: #000 url(background.png) -30px 0px no-
    repeat fixed;
}
```

4. 背景的定位属性

background-position属性定义了背景图片在父元素中的相对位置，它可以接受两个百分比、长度或关键字值，分别用来表示两个坐标值中的一个。如果同时设置了两个，那么第一个值表示水平位置，第二个值表示垂直位置：

Note

许多设计师会同时定义背景色和背景图片。背景色由浏览器直接控制，会和CSS中的其他定义一起快速显示于页面上，而背景图片需要相对较长的时间加载。因此，同时定义二者有助于让页面的加载过程变得更为平滑。

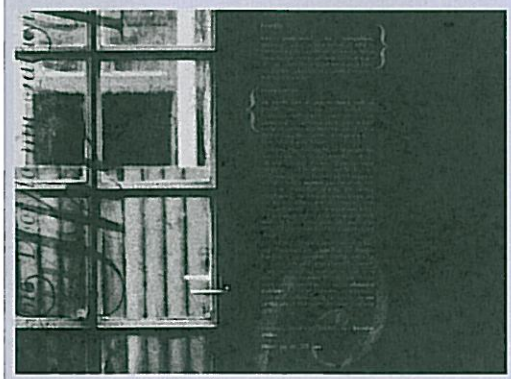


图6-5-2　由于背景图片的位置在页面滚动时保持固定，正文底部和背景图片即可同时显示于屏幕上

```
background-position: 10px 20px;
```

如果只设定了一个值，那么它将表示水平位置，而垂直位置将取默认的50%或者居中。上面的声明将把图片置于水平10个像素、垂直20像素的位置上。可以看到，Lauke的作品中均使用了长度值：

```
body {
  background: #000 url(background.png) -30px 0px no-repeat fixed;
}
```

其中有一条很有意思的设置，将水平位置定为-30px。负值在背景定位中是合法的，它的效果和设定正值相反，会将图片向左平移30个像素。而垂直方向的0像素设定也会让背景图像恰好位于body元素的上边界。图6-5-3和图6-5-4比较了设定负值后的变化，图6-5-3是正常情况下浏览器中显示的样式，图6-5-4是水平位置为负值时的样式。

5. 控制背景平铺

单纯使用HTML无法控制背景重复——默认情况下浏览器会把背景图片同时在x轴和y轴上平铺。所幸CSS的background-repeat属性给了我们更多选项。

background-repeat属性可以设定为以下几个值：repeat-x使背景仅在x轴上平铺；repeat-y使背景仅在y轴上平铺；no-repeat使背景不平铺；repeat使背景同时在x、y轴上平铺，这也是该属性的默认值。Lauke在background属性中使用了no-repeat：

```
body {
  background: #000 url(background.png) -30px 0px
  no-repeat fixed;
}
```

这样的设定保证了图片不会被平铺。虽然书中并没有逐一列出其他部分的代码，但door to my garden中所有的背景均没有平铺设定。

Tip

如果某个值为0，就不需要再额外给出单位。这一点是用于各种CSS属性。所以正文中的声明同样也可以合法地写为background:#000url(background.png) -30px 0 no-repeat fixed;

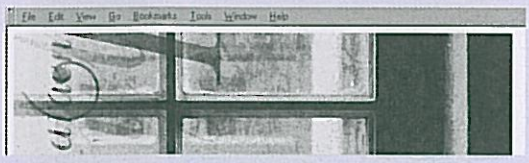


图6-5-3　通常的图片位置

图6-5-4　设定了负值background-position后的背景位置

6-5-5　使用内边距让container元素和浏览器左侧之保持470像素的间距

图6-5-6　用Web Developer extension插件可以看到#container区域的样式

6.5.2　布局选项

为了创建一个300像素宽，位于显示区左侧470像素的#container元素，Lauke给body元素设置了470像素的内边距（padding）：

```
body {
    padding: 0 0 0 470px;
}
```

记住内边距和外边距属性在缩略写法中的顺序是top、right、bottom、left。在这个例子中除了left值之外，所有其他值都设为0（同样值得注意的是，这里Lauke也省去了其长度单位），为页面左边留出了470像素宽的一块空白（图6-5-5）。

而#container元素本身仅仅应用了背景图片、颜色和宽度3个属性：

```
#container {
    background: #000 url(bottom_corner.png) no-repeat
    bottom right;
    color: inherit;
    width: 300px;
}
```

若安装了Mozilla或者Firefox中的Web Developer extension插件，可以打开其中的“outline block element”选项，这样即可看到每个元素的Class和ID等信息。图6-5-6就演示了#container区域在应用样式后的效果。

6.5.3　额外的样式效果

door to my garden使用的CSS样式普遍非常简单易懂，没有什么hack，所以真正具有挑战性的就是用花瓣衬托菜单（图6-5-7）。在不同浏览器中，该菜单存在着较大的兼容问题。

就菜单本身而言，其实并没有多大挑战性，Lauke为#linkList创建了一个容器，并且为其中的每个元素设置了不同的样式以构成导航条的框架。较为复杂的部分在于，如何使花的图像浮于框架的外侧并且能够透过其边缘看到下面的背景。

```
#linkList2>#lselect {
    background: url(flower.png) no-repeat top left;
    margin-left: -65px;
    padding-left: 80px;
    min-height: 150px;
}
```

由于使用了子选择器，所以这段CSS只能在Mozilla、Opera和Safari等浏览器中正常工作。为了能在IE中也显示出相应的信息，Lauke使用了禅意花园HTML中的额外div元素来绝对定位花的图像：

```
#extraDiv1 {
    position: absolute;
    top: 165px;
    left: 142px;
    background: url('flower.png') no-repeat top left;
    width: 115px;
    height: 150px;
}
```

这样一来，完全兼容CSS的浏览器将同时应用两种样式。所以Lauke还必须在这些浏览器中隐藏上述#extraDiv1的效果，因为#linkList2>lselect样式已经足够用了：

```
body>#extraDiv1 {
    display: none;
}
```

大功告成！现在花和菜单在任何一种现代浏览器中都能正确显示了。

6.5.4 完美的设计，简单的CSS

很明显door to my garden的漂亮外观得益于其中独特的图片、作者的摄影技巧以及精心的元素排布。第一眼见到这部作品时，或许你会觉得非常复杂，但其实大部分效果的CSS实现都非常简单，在兼容性方面使用的技巧仅有一处而已。

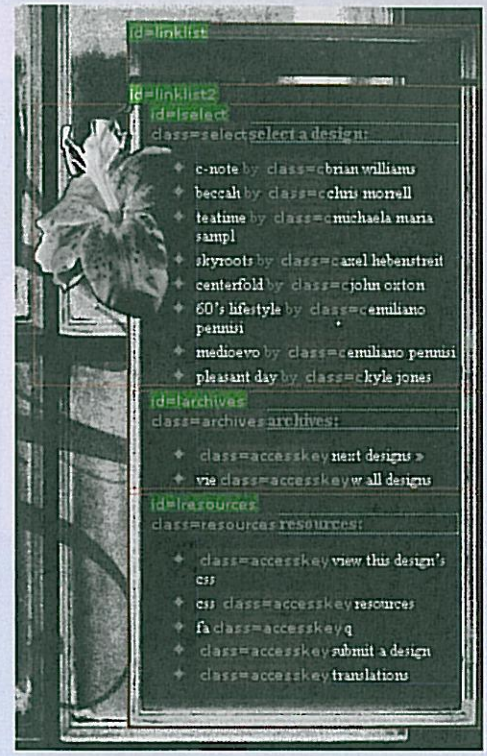
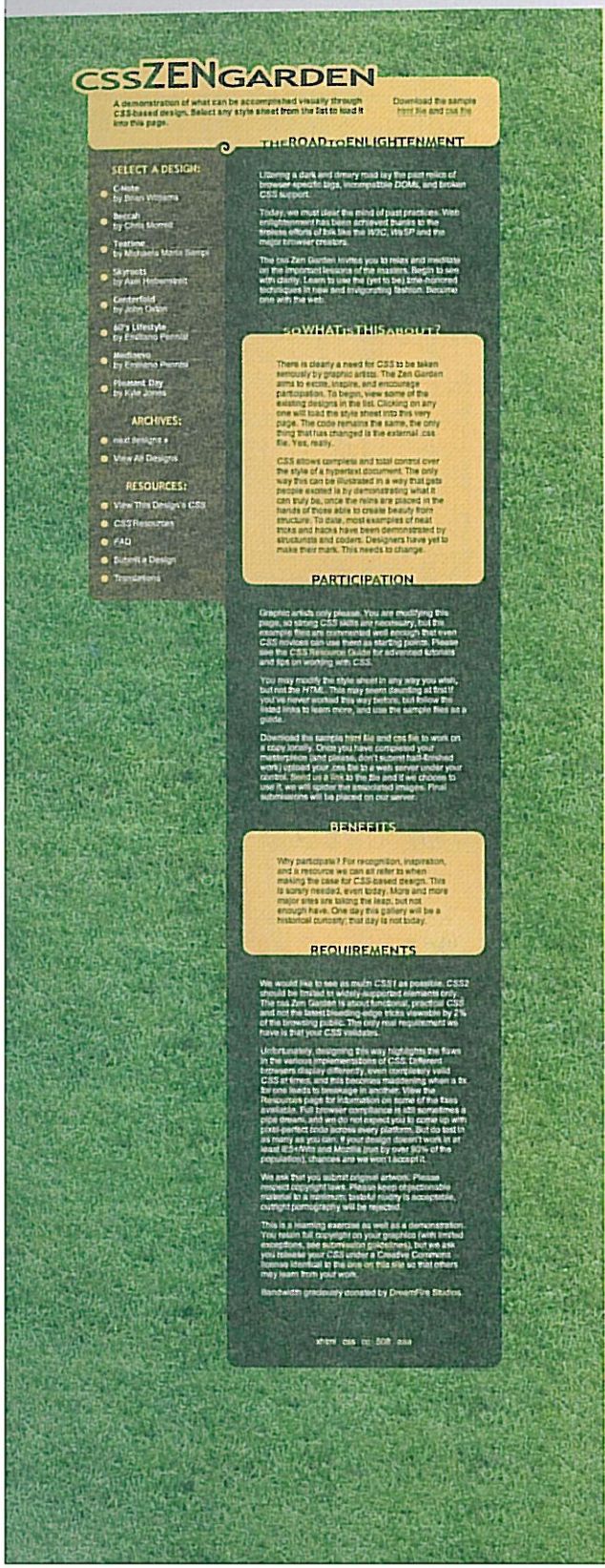


图6-5-7　菜单和花的图片，我们在其周围添加了边框，以便读者识别

Tip

Web Developer extension插件可以在chrispederick.com（www.chrispederick.com/work/firefox/webdeveloper/index.php）网站中免费下载。



Patrick Griffiths，设计师

www.csszengarden.com/063

6.6 Elastic Lawn（绿草如茵）

处理视觉效果问题有时并非那么简单

一个简单的设计日程引发了Patrick Griffiths的灵感，他开始尝试一种新的布局方法并用这种方法来约束图形元素。很快，Griffiths就发现这种从底向上的布局技术会导致设计过程乱作一团。于是他很快聪明地放弃了这个想法，并拿起铅笔和速写本勾画出了页面草图，随后开始用CSS实现，最后加入了图像元素——这正是Elastic Lawn的诞生过程。

设计CSS布局的过程并没有什么“金科玉律”。有些设计师喜欢先在Macromedia Fireworks之类的图像设计软件中绘出草图，然后再将其转化为代码。而另外一些设计师则喜欢双管齐下，同时处理代码和图像。考虑到市面上为数众多的代码和图像编辑软件，无论使用何种方式，总会发现有无数种方式达到最终的效果。

6.6.1 固定还是可变？

在建站时，设计师通常会面临两种布局上的选择：一种是给内容区域设置固定的像素宽度，把页面限制在这个宽度以内；另一种则是让内容区域不受限制地填满整个浏览器的显示区域。

这两种技术都有众多的别名：前者有被称为定宽（fixed width）布局，受限（constrained）布局，或者冰块布局（ice layout）；而后者则被称为变宽（variable width）布局，全屏（full screen）布局，或者流式布局（liquid layout）。无论称呼为什么，这两者都是现代网页设计中最基本的两种布局方式。

当然，这两者各有优劣。而网页设计师更为困惑的问题是，网站的画布尺寸不是固定的。在设计印刷品时，我们总可以确定最终成品的比例和尺寸。而在浏览网页时，访问者完全可以使用640×480像素的非最大化浏览器窗口，或者一个2048×1536的超大型全屏窗口，或者介于其间的任何尺寸的窗口。

1. 固定宽度

固定宽度布局将画布的宽度限制于某个固定的像素大小之内。在需要精确控制图片这类宽度大小不会变动的元素时，该方法特别有效。但是由于限定宽度后作品的尺寸往往会超过较小窗口的可显示范围（图6-6-1和图6-6-2），这就意味着用户将看到水平滚动条。而对于那些使用大尺寸浏览器的用户，却反而会看到很大的一片空白。

2. 可变宽度

可变宽度的布局在任何分辨率的屏幕下都能填满整个窗口（图6-6-3和图6-6-4）。理论上说这是一个解决窗口尺寸问题的好办法，但要在600像素宽和200像素宽的窗口间做平衡也是一件很困难的事情。所以妥协的结果经常是使用最小公分母（lowest common denominator）原则，即让页面在大窗口中自动扩展，但并不在多余的空白区域填入其他视觉元素。因为纯CSS的方式局限性太大，Cameron Adams曾试图用JavaScript更好来解决这个问题（www.themainblue.com/experiment/Resolution-Layout/）。

图6-6-1　Kevin Davis的固定尺寸站点在1024×768的屏幕上显示得非常不错

图6-6-2　但在较小的窗口中却会出现水平滚动条

6-6-3　由于使用了变宽布局，Digital Web Magazine（www.digital-web.com）在1024*768的分辨率下显示良好

图6-6-4　在小分辨率下也有不错的质量，注意页面底部没有出现水平滚动条

Note

比例（ratio）指两个对象的相对大小。1.6:1的比例意味着若第一个对象的宽度为1.6英尺，那么第二个对象的宽度将为1英尺。

Note

Griffiths在2004年1月的*A List Apart*电子期刊上撰写了一篇文章，其中阐释他的"弹性设计"技术。

不仅如此，流式布局在处理比例时存在着更大的问题，在宽度发生变化时这种方法甚至将变得完全不可靠。例如若一块色彩区域和一张图片的原始比例是1.6:1的话，在窗口宽度增大后则可能变为2:1、3:1甚至更大。原因就在于图片大小是不会自动缩放的，这一点我们必须牢记，且在流式布局中也需要尽量避免过于精确的比例关系。

6.6.2　弹性设计

前面所说的所有内容都是关于布局技术，是不是会怀疑本节接下来还会不会有特殊效果的介绍？我们可以很肯定的说，是的！Elastic Lawn中就展示了这样一种既非固定宽度亦非可变宽度，而是结合了两者优点的方案，即所谓的"弹性设计"。

而Griffiths的结合技术也存在其合乎逻辑的理论基础。我们知道文字的尺寸在使用%或者em这样的相对单位时可以改变大小。那如果块级元素也使用此类单位设定大小时，它们是不是也应该能改变大小呢？这正是弹性设计的理论根据。

下面的代码展示了Elasitc Lawn中使用的相对单位。宽度、内边距、外边距以及其他的度量单位都使用了em而不是px，这样就让它们能随字体大小的变化而相应地改变。图6-6-5~图6-6-7显示了在改变浏览器字体尺寸时页面比例的变化。

```
body {
  padding: 0 0 6em 0;
}
#container {
  width: 48em;
}
#preamble, #supportingText {
width: 32em;
  margin-left: 16em;
}
```

看上去是不是很简单呢？不过是不是各种宽度和长度属性都能够应用同样的方法呢？并不完全正确。因为图像的尺寸是固定的，它们不会随着字符大小的改变而改变。所以要彻底地应用弹性设计还需要更多的考虑。

这就需要提到两个能将图像加入到弹性设计中的方法，其中第一个方法用到了重复背景，而另一个则和图像裁减有关。

1. 图像重复

默认情况下用CSS设置的背景图片会在整个元素的横向和纵向平铺。多数情况下关闭这种平铺是有益的。但是弹性设计却需要借助于平铺的实现，因为无论元素本身有多大，应用了平铺效果的背景总会重复并且填满元素区域，而不是在图片结束后就戛然而止。

Elastic Lawn用绿色的草地背景填满了整个body元素。#linkList2，#preamble，#supportingText元素则使用了这个草地背景的变体（图6-6-8）。这些图片被透明的禅意花园正文与链接栏所覆盖，最终成就了那怡人的阴影和透视效果。

如果仅仅使用默认的方形盒子来承载图像，那自然极为简单。但是Griffiths却特意使用了圆角的图片，这样也便于在改变字体大小时反映出其与众不同的特殊效果。

不过还有一个问题，那就是并非所有的图像都是可以平铺的，除非是那些四条边缘都有相同色彩的图像，否则在图像衔接的边缘会出现明显的条缝。图6-6-9就显示了未经处理过的草地图片，从中可以明显地看到水平和垂直结合部分的线条痕迹。

要解决这个问题，只要将图像在Adobe Photoshop中做一些前期处理即可。使用Offset滤镜（Filter>Other>Offset）将一边的图像推至另一边，即可让其边缘完美地融合起来。

图6-6-5　小号字体下，主体部分会显得较小

图6-6-6　略大的字体下，主体部分会变得更宽

图6-6-7　更大的字体下，黄色标题的外边距也会增大

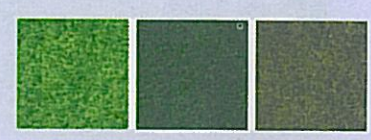


图6-6-8　Elastic Lawn中的草地图片。亮绿色用作body元素的背景，褐色用于工具条，深绿色用于正文部分

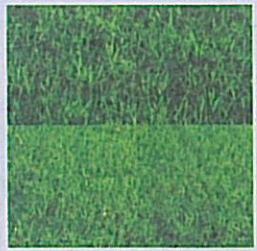


图6-6-9　未经处理的图片在平铺时会有明显的缝隙

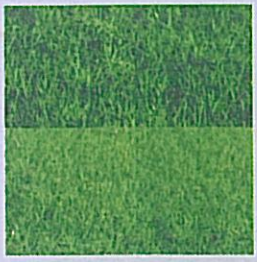


图6-6-10　使用Offset滤镜后，仍旧有色相上的差异

图6-6-11　复制周围区域，以消除缝隙

图6-6-12　完成后的结果

图6-6-13　很难再看到明显的缝隙了

图6-6-10~图6-6-12展示了如何使用采样像素覆盖这条缝隙，图6-6-13显示了最后的效果。至于是用橡皮图章（Clone Stamp）还是用修复笔刷（Healing Brush），则完全取决于习惯，只要能将无缝区域的像素覆盖到缝隙上即可。具体的图像制作技术可以参见第4章，也可以自由选择修改或是重绘这些图案，抑或是用任何可以达到期望效果的图像调整技术。

2. 图像裁减

在Griffiths的“Elastic Design”一文中引用了Haiko Hebig的Blog（www.hebig.org）作为例子（图6-6-14和图6-6-15），其目的是为了说明弹性设计中用到的一个重要的、名为可变裁减（variable cropping）的技术，该技术可以在页面空白的大小变化时自动调整图像显示出来部分的尺寸。

要创建弹性的图片元素有两条途径：第一条是为承载这个图像的可变长元素应用overflow:hidden属性；第二条则是为背景元素添加可变长度。例如，下列代码中就定义了一个原始尺寸为600像素的图像，嵌套于div元素中：

```
<div class="wrapper">
  <img src="skyline.jpg" width="600" height="200"
alt="Skyline"/>
</div>
```

如果这个div元素是用em来标识长度的，则图像的超出部分可以用overflow属性来隐藏：

```
#wrapper {
  width: 10em;
  overflow: hidden;
}
```

另外一种方式是直接将如下样式赋予div元素：

```
#wrapper {
  width: 10em;
  background: url(skyline.jpg) no-repeat;
}
```

如果图像本身就可以重复，no-repeat属性也可以被安全地移除，让图片能够在较大的页面上平铺直至填满更大的区域。

3. 向前跃进

弹性设计是对网页布局理论的一次积极且引人注目的创新，但正如前面讨论的那样：由于设计元素的比例变化与浏览器窗口大小无关，网页很可能变得比浏览器窗口更大，此时页面就会被裁减并出现滚动条，这都不是我们所希望看到的。当然，max-width属性可能会帮助我们改善这种情况：

```
#container {
  max-width: 700px;
}
```

但Windows平台上的IE 6浏览器却并不支持max-width，现有的宣称能够打破该限制的办法也并不如期待中的那样有效。诸如此类莫名其妙的异常的确在不停地打击着人们在尝试新鲜方法时的信心，但即使如此，这些尝试也仍会不时造就一些成功案例，这当然不是件坏事。

在用新技术推进网站设计的浪潮中，成败都难以预料，网站设计师们需要在其中确定一个平衡点。而作为一个成功的例子，Elastic Lawn无疑向我们展示了勇于尝试带给其他设计师们的有益结果和无限启发。

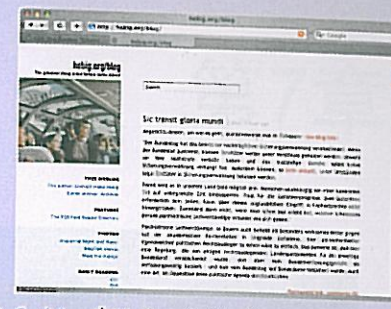


图6-6-14　在Haiko Hebig的Blog中，左侧图像显示的尺寸取决于空白区域的大小。此图是800×600分辨率下的样子

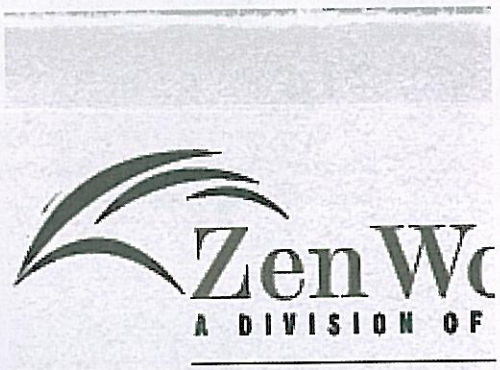
图6-6-15　同样的图片，在1100×600分辨率下成比例地变宽

Tip

Svend Tofte制作了一个非标准的方案，可以有效地修复IE 6不支持max-width的问题（www.svendtofte.com/code/max_width_in_ie）。

Hedges（树篱），230页

Radio Zen，236页



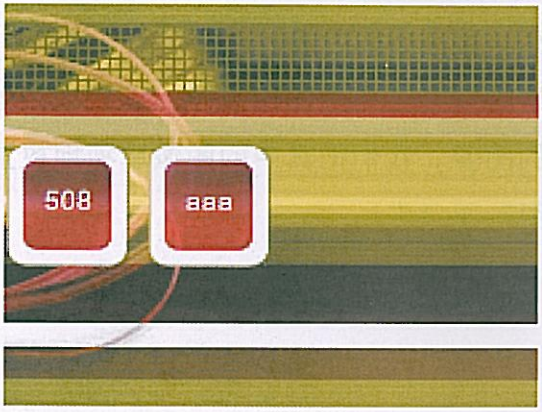
Corporate Zen works，248页

Open Window（敞开窗口），254页

第7章

重构

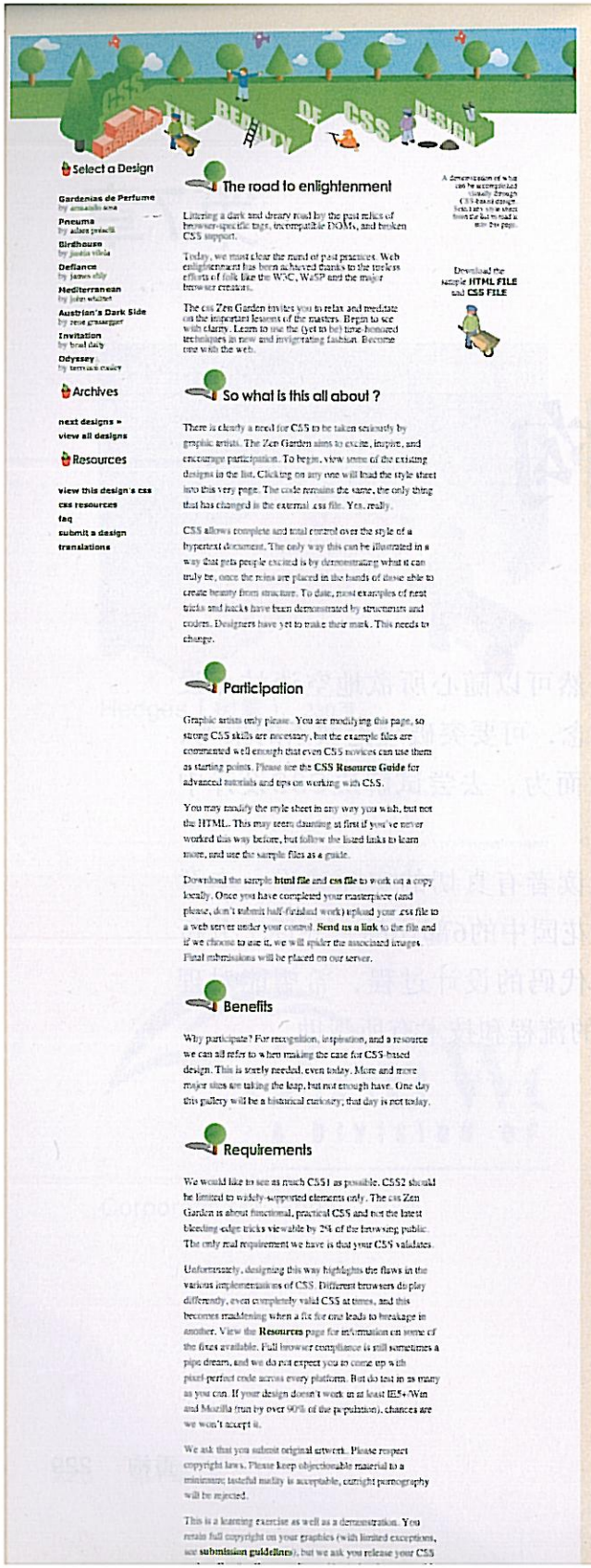
South of the Border（南部边境），242页



mnemonic（记忆），260页

我们当然可以随心所欲地空谈站点设计的理论概念，可要突破理论教条的束缚，却必须亲力而为，去尝试解决CSS设计中的实际问题。

为了让读者有真切的实际感受，本章选取了禅意花园中的6部作品，并逐个分析了其中CSS代码的设计过程，希望能对理解CSS设计的流程和技术有所帮助。



Kevin Mears，设计师

www.csszengarden.com/031

7.1 Hedges（树篱）

让任意的前景都能与背景图像完美融合

在禅意花园的众多提交作品中，有关花园的主题层出不穷。但Kevin Mears却从设计师和插画师的角度出发，抛开了围绕植物的传统手法，设计出具有卡通风格的“田园风光”。

在这部作品中，Mears将自己装扮成手推小车的园丁，加之各种有趣的卡通元素的妆点，最终让他的花园充满着亮丽的色彩。

而在这幅以插画为主的作品中，理所当然地运用了大量的图像。我们把其中的图像单独地放置在后一页的图7-1-1中。

7.1.1 设置边界

在加入这些有趣的图像之前，需要为页面设置一些基础的参数。Mears首先使用了定宽布局，所以尽管页面头部背景横向平铺到了整个窗口，页面中心部分却始终被限制在732像素宽的区域中。其次，#container将在页面中包围所有的其他元素，所以要为其应用如下的样式：

```
body {
 margin: 0;
 padding: 0;
 color: #888;
 text-align: center;
}
#container {
 position: relative;
 padding: 0;
 width: 732px;
 margin: 0 auto;
 text-align: left;
}
```

#container被限定于732像素宽，而将外边距设置为auto则会让#container内容部分水平居中于整个页面中[1]。由于body元素的text-align属性被设为center，所以我们要将#container的text-align属性设置为left，以覆盖body元素的设定。Windows平台上的IE 5.0和5.5不能辨认当前应用的margin值，因此这是确保#container居中的一个必要手段。在应用了链接样式之后，我们可以看到图7-1-2中的效果。

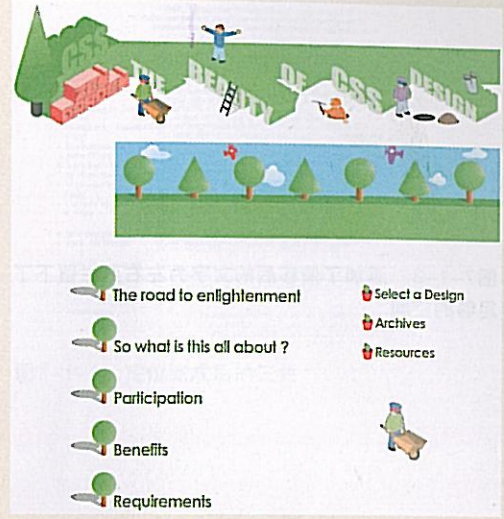


图7-1-1　Hedge中用到的所有图像

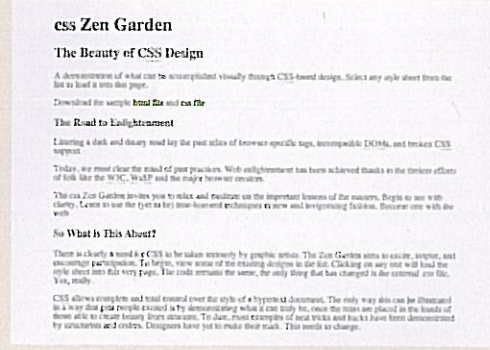


图7-1-2　未加入图像前居中的一栏

7.1.2 打造主体

下一步就是隐藏复杂的头部元素，以免它们影响到创建主体部分的分栏布局：

[1] 具体作用如下：auto被设置于右外边距，由于左外边距未被设置，它将继承右外边距的auto值，而auto值会导致#container的元素外框撑满body区域，而去除了#container width部分设置的宽度，剩余的空白边界将由左右外边距平分，最终让#container的内容区域居中于body元素中。——译者注

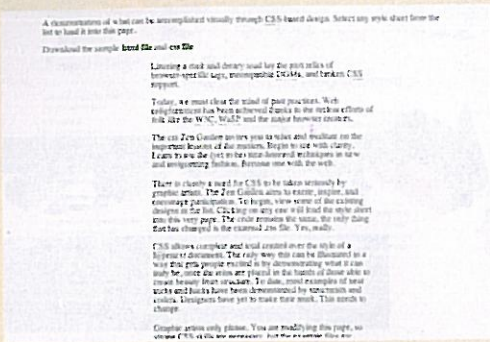


图7-1-3　添加了偏移后的文字为左右两栏留下了足够的空间

```
h1 span, h2 span, h3 span, #quickSummary p.p span {
 display: none;
}
```

这样，Mear就完成了主内容区的定义，可他在主内容区中三栏设计中却遇到了麻烦——Mear发现禅意花园预先定义的标签还不够灵活——可以看到，在主栏中有#preamble和#supportingText两个标签，但这它们却缺少一个可供统一设置样式的共同的父元素。所以我们只能分别定义其样式，并努力让它们在视觉上保证融合：

```
#preamble {
 position: relative;
 left: 195px;
 padding-bottom: 0px;
 width: 400px;
}
#supportingText {
 position: relative;
 left: 195px;
 padding-bottom: 0px;
 border-bottom: 2px solid #363;
 width: 400px;
}
```

可以看到，每个区块的宽度都设置为400像素，并且用195像素的左偏移以形成中间一栏。当然使用margin-left也能产生相同的偏移效果，只是源代码中使用了当前的方法而已。在完成这个步骤之后，效果如图7-1-3所示。

接下来要把工具条移动到合适的位置。左侧的列表现在处于页面的底端，因此我们可以借助绝对定位将其挪动到页面上部。

```
#linkList {
 position: absolute;
 top: 200px;
 left: 0;
 width: 190px;
}
#quickSummary p.p1 {
 position: absolute;
 top: 12em;
 right: 15px;
```

```
  width: 120px;
  margin: 75px 0 0 580px;
  font-size: 80%;
  text-align: right;
}
#quickSummary p.p2 {
  position: absolute;
  top: 25em;
  right: 15px;
  width: 120px;
  margin: 0;
  font-size: 93%;
  text-align: center;
  padding-bottom: 90px;
  background: url("barrow.gif") no-repeat bottom;
}
```

上述CSS设置为作品中的多个元素均应用了样式。例如，为#linkList设置了宽度，并且将其移至页面上部，#container的左侧。还用相同方法将两个标有#quickSummary的段落移至页面的右侧（作为改进，可以为两个段落的父元素#quickSummary设置部分定位属性以减少冗余代码）。完成以上步骤后，显示效果将如图7-1-4所示。

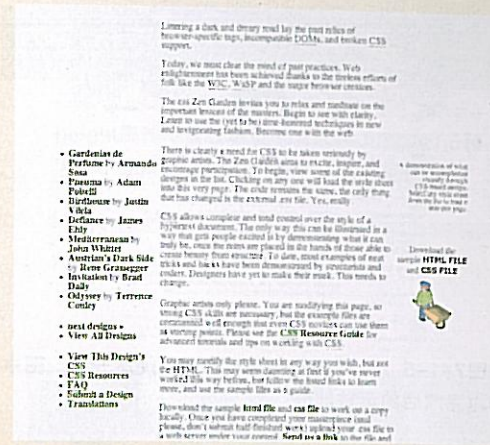


图7-1-4　定位完成后的三栏

7.1.3　图像处理中的诀窍

到目前为止，整个布局的框架已经完成。接下来要考虑的就是更为直观的图形区块的处理。Mear先从页首部分开始，他以一行重复的树木作为背景，而后在树木的上方手工绘制出站点标题。这里我们先从相对简单的背景图案入手：

```
body {
  margin: 0;
  padding: 0;
  background: #FFF url("bg_tree.gif") repeat-x left top;
  color: #888;
  text-align: left;
}
```

在上面这段代码中，Mear为现有的body样式添加了背景图像，且将background属性设置为repeat-x。简单修改之后，图像即可如我们所愿地水平伸展。

图7-1-5　少量的代码即可让页面焕然一新。图中为完成后的页首部分

图7-1-6　在浏览器窗口大小变化时，前景中的人物会随之移动，但其总是能很好地与背景图像相融合

页面顶端的h1元素应用了另一个非重复显示的图像，这张图像将位于前面设定的背景图像之上。还记得先前为所有头部元素的span设置的display:none吗？该设置将会隐藏原始文本，因此背景图像即可不受干扰地呈现出来。接下来就是给h1元素指定合适的尺寸：

```
#pageHeader h1 {
 height: 200px;
 background: url("nutitle.gif") no-repeat;
 padding: 0;
}
```

由于图像会占用很多水平空间，所以我们允许h1元素填满整个包含它的容器（也就是前面那个宽度设为732px的#container），至于高度并没有太多要求，只要调整到一个较为合适的尺寸即可。注意到先前处理的几栏也会同时向下平移，这是由于刚刚加入的height属性会影响到h1下方元素的缘故。然而左右的两列却不受影响，因为它们应用了绝对定位，从而独立于常规文档流之外。在最终实现的效果中，这两个细微的调整却让整个站点焕然一新（图7-1-5）。

由于body元素的背景固定，而页首的位置却是可变的，所以我们无从断定页首的各个图像将重叠于背景中的树木还是天空上。例如图中张开双手的那个人，当缩放浏览器窗口时可以看到他相对于背景的位置也随之变化。好在该图像的上半部分使用了透明的像素，所以能够完美地融合于背景中。无论浏览器的尺寸设为多大，背景和前景图像都能很好地融合在一起（图7-1-6）。

7.1.4　格式化文字

由于完成后的作品使用了浏览器的默认字体，所以并不需要太多的额外设置。我们先从中间一栏开始分析：

```
#supportingText p {
 line-height: 1.5em;
}
#preamble p {
 width: 350px;
}
#footer {
 background: #D9D98B;
 color: #fff;
 padding: 10px 20px;
```

```
  border-top: 1px solid #363;
  font: 85% Verdana,Arial,Sans-serif;
  text-align: center;
}
#footer a:link, #footer a:visited {
  padding: 0 5px;
  font-weight: normal;
}
```

第一条规则用line-height改变了文字间的行距。接下来将#preamble的宽度限定于350px，以便为#preamble和右栏留出足够的间隔。

页脚部分需要更多的格式设定。前面我们已经为#supportingText设置了绿色的底部边框，现在则要为#footer设置border-top属性，以便添加新的边框颜色。完成了这些步骤之后，作品效果将如图7-1-7所示。

7.1.5 收尾工作

最后，不要忘记对隐藏的头部进行一些处理。

```
#supportingText h3 {
   height: 70px;
}
#explanation h3 {
   background-image: url("what.gif");
   background-repeat: no-repeat;
}
#linkList h3 {
   height: 41px;
}
#lselect h3 {
   background-image: url("select.gif");
}
```

上述代码非常简单易懂，不过这也没什么值得大惊小怪的——链接列表都由图像组成，自然免除了字体、颜色的设置——在上述代码中，我们仅仅为#supportingText中的h3元素设置了公共高度和各自的背景图像而已，就这样整部作品大功告成。

Tip

和一般的CSS属性不同的是：line-height属性不需要单位——line-height:1.5和line-height:1.5em会得到相同的行间距。

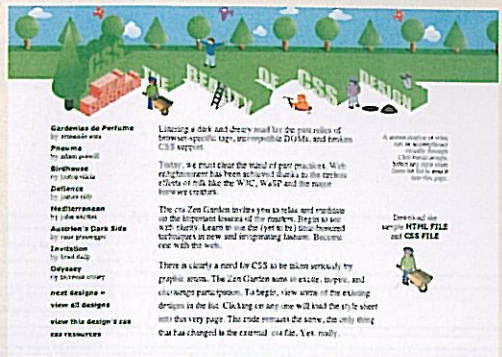


图7-1-7　近乎完成的作品，唯一缺少的就是页首部分

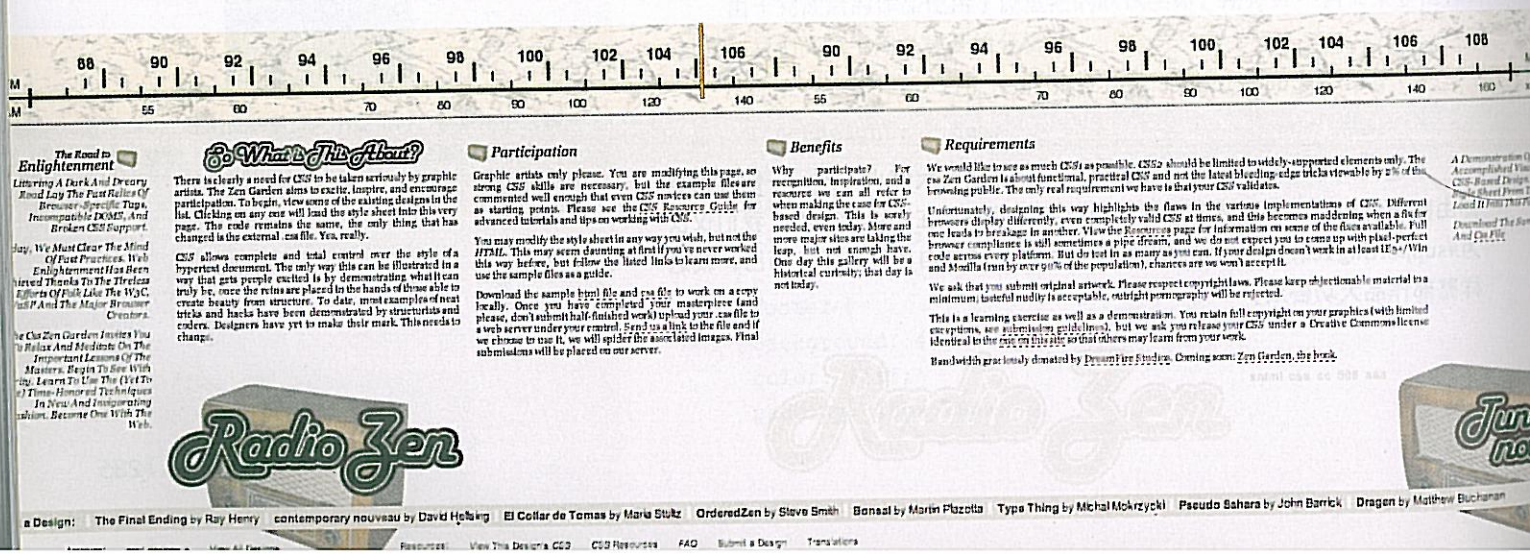
Marc van den Heuvel，设计师
www.csszengarden.com/058

7.2 Radio Zen

用固定的背景图像实现类似收音机频道调节器的效果

Marc van den Heuvel从他的老式收音机中获得了作品的灵感。通过使用固定背景图像，他模拟出了带有活动指针的频道调节器的效果。由于页首的位置需要改变，且指针也要能够水平移动，所以作者采用了少见的水平滚动布局。

除此之外，van den Heuvel还采用了带有强烈视觉张力的retro字体，加之老式收音机的背景与泛黄的整体色调，让整个作品散发出一种怀旧的气息。



7.2.1 构建基础

除了页首图像之外，Radio Zen在实现过程中只运用了很少的图像资源（图7-2-1），而作品中水平滚动的布局则主要依赖于CSS的定位技术。

和很多其他设计师一样，van den Heuvel在开始之前就隐藏了页面两个头部中的文字。因为这些文字最终会用背景图像替代，所以现在理所当然应该将其隐藏：

```
#pageHeader h1 span, #pageHeader h2 span {
 display: none;
}
```

这样做的另一个目的是为将加入顶部的收音机频道调节器预留出空间（图7-2-2）。这里我们并没有列出所有的样式代码，读者可在原始的CSS文件中获得。

图7-2-1　Radio Zen中用到的图像

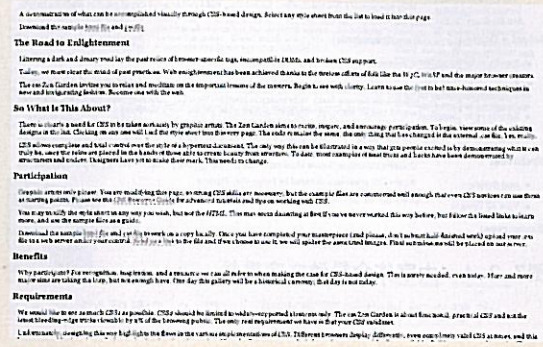


图7-2-2　使用基本文字样式时的Radio Zen

布局框架

在Radio Zen中，多数元素使用了绝对定位，这就意味着此类元素将被排除于文档流之外。而其他在文档流之内的元素并不能知晓绝对定位元素的始末位置，以致元素之间有可能会发生重叠。眼下的解决方法只能是将这些元素的边缘相互对齐，相关的定位规则如下：

```
#quickSummary {
 position: absolute;
 right: 0px;
 text-align: left;
 width: 165px;
 height: 340px;
 top: 95px;
 padding-top: 40px;
}
#preamble {
 position: absolute;
 left: 10px;

 text-align: right;
 width: 165px;
```

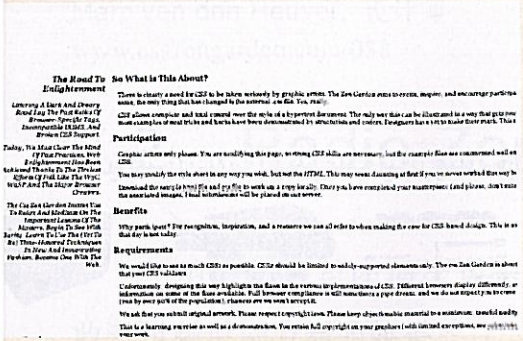


图7-2-3　设定了基本位置属性后的页面

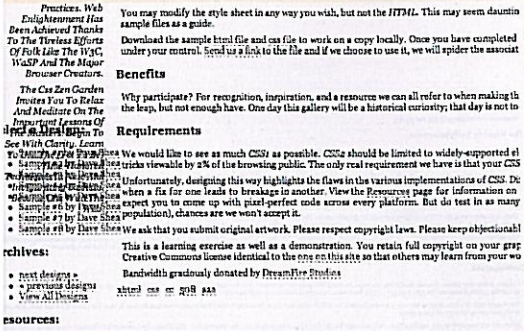


图7-2-4　大字体会导致文字重叠

Note

在Windows平台上的IE中，固定背景只能用于body元素。

```
  top: 95px;
}
#supportingText {
  position: absolute;
  width: 1230px;
  padding: 95px 0 0 185px;
  text-align: justify;
  z-index: 3;
}
#linkList {
          font-family: Arial, Helvetica, sans-serif;
  position: absolute;
  top: 0;
  padding-top: 480px;
  text-align: justify;
  z-index: 1;
}
```

若浏览器的字体能始终保持前面设置的10px，那么这些规则将没有任何问题（图7-2-3）。但很多浏览器支持动态改变字体大小，所以在大字体情况下，文字依然会发生重叠（图7-2-4）。稍后我们将介绍如何解决这个问题。

7.2.2 分层和附着（Attachment）

我们强烈建议您打开原始网页试着水平滚动（图7-2-5A和图7-2-5B），所看到的情况或许和原先的猜想有所不同。事实上，指针在频道调节器上移动的效果并非由脚本完成。Radio Zen巧妙地使用了分层功能，并利用background-attachment:fixed属性实现了最终的效果。作者将带有FM刻度的频道调节器作为静态图像应用到背景元素之上，从而使在滚动时该图像在窗口中的位置保持固定。而此时，页面其他部分则能相对频道调节器板滚动。我们再仔细分析body元素的样式定义，可以发现在一行额外的规则中同时给出了background-image和background-position的设定：

```
body {
  font: 10px Georgia, Arial, Helvetica, sans-serif;
  color: #353638;
  background: #F8F8EE url(schaal.jpg) no-repeat fixed top
  center;
  margin: 0px;
}
```

而指针则应用到#supportingText元素中：

```
#supportingText {
 position: absolute;
 width: 1230px;
 padding: 95px 0 0 185px;
 text-align: justify;
 background: url(naald.gif) top center no-repeat;
 z-index: 3;
}
```

现在终于可以明白为什么要将#supportingText设置为一个如此“巨大”的width值了，这是因为它不但要装入水平排布的众多图像元素，还要强制页面出现水平滚动条，以便让指针有足够大的空间移动到调板的左侧。

7.2.3 列的定位

目前为止，body中的文字包含了一系列的宽大段落，这些段落将被水平排布在浏览器窗口中：

```
#preamble {
 position: absolute;
 left: 10px;
 text-align: right;
 width: 165px;
 top: 95px;
}
#supportingText h3 {
 margin: 0 8px 0 3px;
}
#explanation, #participation {
 float: left;
 width: 290px;
}
#benefits {
 float: left;
 width: 150px;
}
#requirements {
 float: left;
 width: 500px;
}
```

这样的布局看上去显得更加美观（图7-2-6）。接下来我们再讨论一下前面所说的字号变化问题。

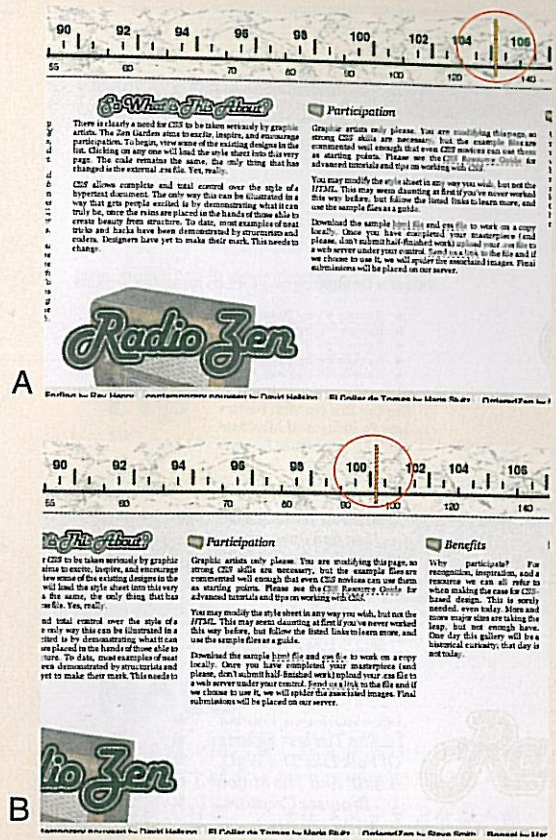


图7-2-5A和图7-2-5B　滚动页面时指针随之移动

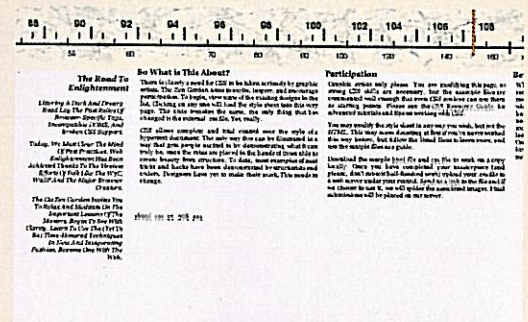


图7-2-6　原先垂直堆砌的段落现在被水平隔开

图7-2-7A ~图7-2-7C　字号变化所导致的重叠

字号变化所导致的问题

由于绝对定位将会把元素移出常规文档流，所以在某些情况下，文字将变得过长，甚至遮盖住其他元素。若使用的是如Firefox之类的可动态设置字体大小的浏览器，就能很清楚地看到页面底部的列表部分在字号改变时发生的变化（图7-2-7A~图7-2-7C）。

虽然目前来说这块区域还没有应用样式，但是如果现在不着手处理的话，也就意味着在未来埋下一颗定时炸弹。我们能会注意到，前面我们将#linkList元素的top属性设置为0，将其定位于页面顶部；之后又设置了480px的padding-top值，将其移动到下方。这个padding的设置可以使作品列表区域暂时位于页面中其他元素的下方，方便我们设计页面。但从本质上说，它和列表内部的div元素定位却没有任何关系，因为这些元素实际上都是通过绝对定位来控制的：

```
#lselect, #larchives, #lresources {
 position: absolute;
 text-align: left;
 padding: 5px;
}
#lselect {
 width: 1590px;
 top: 46em;
 background-color: #E8E9CF;
}
#larchives {
 width: 300px;
 top: 49em;
 left: 75px;
}
#lresources {
 width: 800px;
 left: 400px;
 top: 49em;
}
```

可以看到，由于div的定位都是以em，而不是px为单位，所以它们的顶端位置会随着字体大小的改变而同样发生变化，这也就避免了重叠问题。而

在继续设置如下规则之后，作品列表和作品存档区域将出现在页面底部，如图7-2-8所示：

```
#lselect ul, #larchives ul, #lresources ul {
 margin: 0px;
 padding: 0px;
}
#lselect h3 {
 margin: 0 10px 0 0;
 color: #555F44;
 font-size: 8pt;
 float: left;
 font-weight: bold;
}
#lselect li {
 color:#000000;
 list-style-type: none;
 float: left;
 border-left: 4px solid #F8F8EE;
 padding: 0 5px;
}
#lselect a:link, #lselect a:visited {
 border: none;
 text-decoration: none;
 font-size: 8pt;
 font-weight: bold;
}
#lselect a:hover, #lselect a:active {
 text-decoration: underline;
}
#lselect a.c, #lselect a:link.c , #lselect a:visited.c {
 color: #422222;
 font-weight: normal;
}
```

7.2.4 探索一切可能

回顾Radio Zen的设计过程：只不过是简单地通过为body元素应用了固定背景，就实现了调板和移动元素之间的有趣交互。可见，只要悉心研究CSS，即使一些简单的CSS样式也能给用户带来相当独特的视觉体验——Radio Zen就是其中一个让人称道的尝试。CSS更多的纷繁复杂的特性始终敞开着大门，等待我们进一步去发掘。

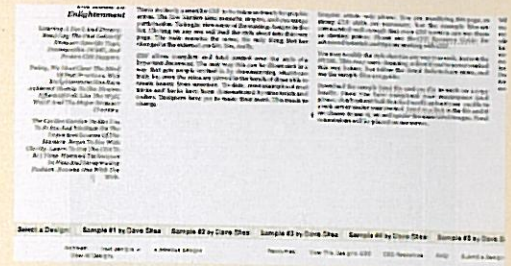


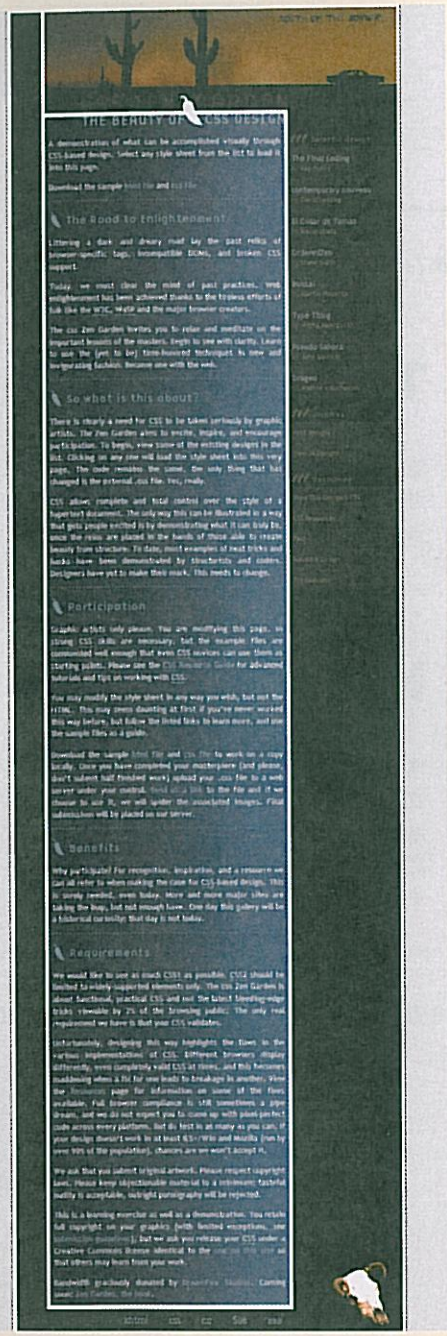
图7-2-8 应用样式后的页脚

Tip

在绝对定位中使用em为单位会导致定位随字体大小改变而自动调整。但若是使用px为单位，虽然可能会导致文字重叠，但却能够对元素位置进行更为精确的控制。

Note

值得一提的是，作者最后使用了图像对作品进行了修饰，例如在页首和页脚加入的收音机图样等。这些在本文中并没有深入阐述，而是作为练习留给了读者。所有的样式定义的终稿都可以在Radio Zen的CSS文件中找到。



Rob Sheilds，设计师

www.csszengarden.com/093

7.3 South of the Border（南部边境）

用外边距和定位体现框架之感

咖啡在某种程度上是一种普遍的创意源泉，可在South of the Border中引发Rob Sheilds灵感的则是一杯热茶。对Shield而言，无论酝酿什么样的作品，总能找到50多个初始的创意，其中每个创意包含的一系列关键字都能帮助他勾画出不同的站点主题。

Shields在这部作品中使用了他的第三个创意，这是个关于美国西南部风情的主题。为了把握住这转瞬即逝的灵感，Shields在灵感乍现后，急速地打开了Adobe Photoshop处理图像，完成了图7-3-1中的组合效果。进一步修饰之后，Shields开始着手CSS代码的编写，直到最终完成了主题鲜明的South of the Border。

7.3.1 由基础开始

Shields的设计中使用了独特的色彩基调，他的灵感来自于一件带有蓝色条纹的褐色外衣。为了模拟出这种色彩基调，Shields将body元素的背景颜色设置为蓝色，又用浅褐色和深蓝色制作链接，而主要文字部分则使用了白色，与深蓝色背景形成足够的反差：

```
body {
    font: 83%/150%;
    color: #fff;
    background: #EDEDED url(background.gif) repeat-y fixed bottom
    ➡center;
    margin: 0px;
    padding: 0px;
}
```

可以看到，Shields特意将字体大小用百分比表示，以确保它们可被自由缩放。下一页将给出字体和背景的效果。

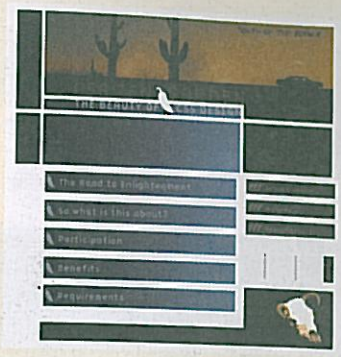


图7-3-1 South of the Border中用到的图像

7.3.2 为容器设置样式

在基本样式定义完成之后（可在源文件中找到这里跳过的部分CSS代码），接下来就是定义#container div的样式。下面声明中的样式定义了容器的背景图像、外边距和宽度：

```
#container {
    position: relative;
    background: url(bg-central.gif) repeat top left;
    margin: 0px auto 0px auto;
    width: 655px;
}
```

#container元素的position被设定为relative。由于并没有再定义其top和left属性，所以#container div还是会留在常规文档流中。许多CSS新手都会被relative一词所迷惑，以为相对定位是指元素的定位会相对于另外某个元素的位置。如此解释这个英语单词自然非常合理，但它在CSS中却不是这个意思。

Note

常规文档流是指在没有任何强制定位元素时，浏览器对页面的解释和布局。若移除了禅意花园中的所有样式，我们可以看到所有的文字和标题都会以从左到右自上至下的方式排布，并且浏览器也会根据不同的分辨率做相应的调整。

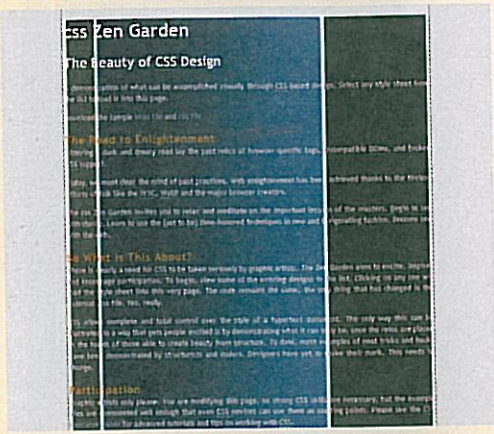


图7-3-2　设定了位置、宽度、外边距后的#container div

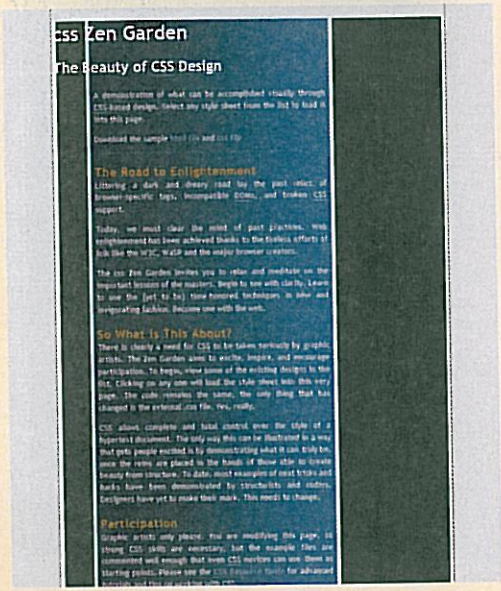


图7-3-3　为body中的文字设定外边距

相对定位让设计者能够在定位某元素的同时，仍保证其位于常规文档流中。这和使用如绝对定位等其他定位方式有很大的差别，我们将很快提到。

以前，我们的目标是创造一个在视觉上囊括所有其他元素的容器——#container。而既然现在已成功设定了#container的样式（详细样式可在CSS源文件中看到），我们即可着手为其中的子元素设置更多的样式了（图7-3-2）。

7.3.3　驾驭文本

当然，现在的首要任务就是把位于#container中的文本拉回到容器范围内。我们可以通过为各个div元素设置外边距来实现该目的：

```
#intro {
    margin:0px;
}
#quickSummary, #preamble {
    margin: 0px 200px 0px 68px;
}
#pageHeader {
    margin: 0px;
}
#supportingText {
    margin:0px 200px 0px 68px;
}
```

上面代码中定义的这些外边距即可将文本拉至容器的范围内（图7-3-3）。

7.3.4　加上标题

添加图形化的标题，尤其对于#pageHeader中的h1来讲，将大大有助于平衡文字和标题的整体视觉效果：

```
#pageHeader h1 {
    margin: 0px 0px 0px 0px;
    background: transparent url(top.gif) no-repeat top left;
    width: 655px;
    height: 206px;
}
```

图7-3-4显示了#pageHeader中h1样式所设置的体现主题的背景图像。

这里我们摘录了一部分标题的样式，用来说明这部分是如何通过设置外边距、背景图像和长宽来有效装载图像和文字的：

```
#preamble h3 {
    background: transparent url(ch1.gif) no-repeat top left;
    margin: 10px 0px 5px 0px;
    width: 388px;
    height: 44px;
}
#benefits h3 {
    background: transparent url(ch4.gif) no-repeat top left;
    margin: 10px 0px 5px 0px;
    width: 388px;
    height: 44px;
}
#requirements h3 {
    background: transparent url(ch5.gif) no-repeat top left;
    margin: 10px 0px 5px 0px;
    width: 388px;
    height: 44px;
}
```

现在标题部分和页面其他部分协调得非常不错（图7-3-5）。

这里我们再查看蓝色边框的样式：

```
#benefits {
    border-bottom: 1px dashed #8095AF;
}
#preamble {
    border-bottom: 1px dashed #8095AF;
}
```

这样设置之后，边框将呈点划线状，起到分割各个文字块的作用（图7-3-6）。

图7-3-4　#pageHeader图像确定了整个作品的基调

图7-3-5　标题图像中带红辣椒的图案，以配合美国西南部的主题

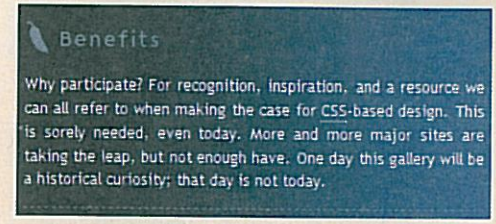


图7-3-6　边界线分开了各个文字区域

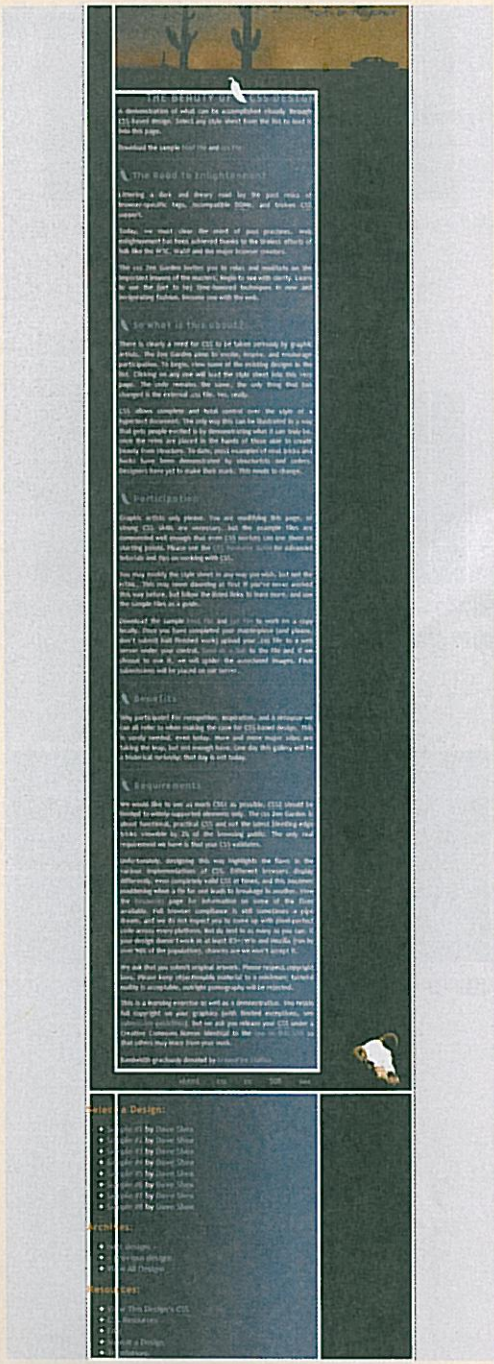


图7-3-7：将页脚设定到正确位置

7.3.5 转到页脚

在中心部分顺利完成之后，我们现在需要将页脚部分布置到正确的位置。页脚的内容非常简单，仅仅是一些用于验证的链接和一具装饰性的牛头骨而已：

```
#footer {
    text-align: center;
    background: transparent url(bottom.gif) no-repeat top left;
    height: 37px;
    margin: 0px -200px 0px -68px;
    padding-top: 88px;
}
#footer a:link, #footer a:visited {
    margin-right: 20px;
}
```

注意到样式中使用了负的左右外边距，让页脚能够更好地置于包含它的元素中（图7-3-7）。

7.3.6 绝对定位

还记得先前的#container div使用的是相对定位么？这里的链接列表却在相对定位的#container div中使用了绝对定位：

```
#linkList {
    position: absolute;
    top: 207px;
    right: 6px;
    width: 181px;
}
```

这条样式会将列表布局在正文旁边。和相对定位不同的是，绝对定位以定位容器元素为参照设置位置。若没有定位容器的话，绝对定位的元素会以根元素html为参照。

若想实际感受一下该效果，可以暂时将相对定位的属性从#container div中移除（图7-3-8）。

绝对定位的本质是这样的：一个绝对定位元素会参照它的定位容器元素

来定位，如果没有定位容器元素，则会以html根元素为参照。但由于设置了相对定位，绝对定位的#linkList就会定位在该元素内。所以一旦将#container div的position属性的值重新设为relative[1]，就会再次回到我们期望中的布局。

7.3.7 修饰链接的样式

解决完#linkList的定位问题，我们可以开始考虑链接的样式了。第一步需要做的就是将列表的内外边距设为0：

```
#linkList ul {
    margin: 0px;
    padding: 0px;
}
```

这给导航条的进一步精确配置铺平了道路。接下来的代码设置了单个列表项的样式：

```
#linkList li {
    list-style-type: none;
    background: transparent url(li-bg.gif) repeat top center;
    padding:8px 6px 8px 6px;
    margin:0px;
    border-top: 1px solid #5D4831;
    border-bottom: 1px solid #221609;
    display: block;
}
```

这些额外的对链接的标题、色彩、内外边距的设置为整个作品添加了不少有趣的细节，这些代码同样可以在CSS源文件中找到。

7.3.8 完成

外边距是控制文字和图像最有效的工具。相对定位也能让我们在元素中加入绝对定位的子元素，从而有效地处理那些棘手的布局问题。本节中的South of the Border就用简单直观的样式表现出了极其生动的主题，这也从一个侧面证明了CSS的强大功能。

[1] 将某元素的position设定为relative将使之成为其中绝对定位元素的定位容器元素。——译者注

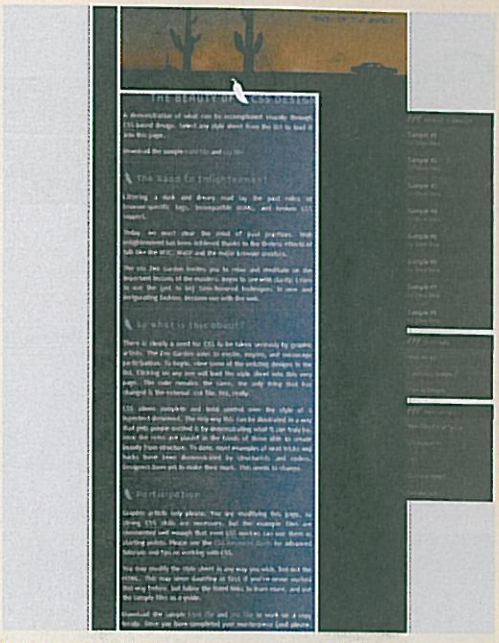
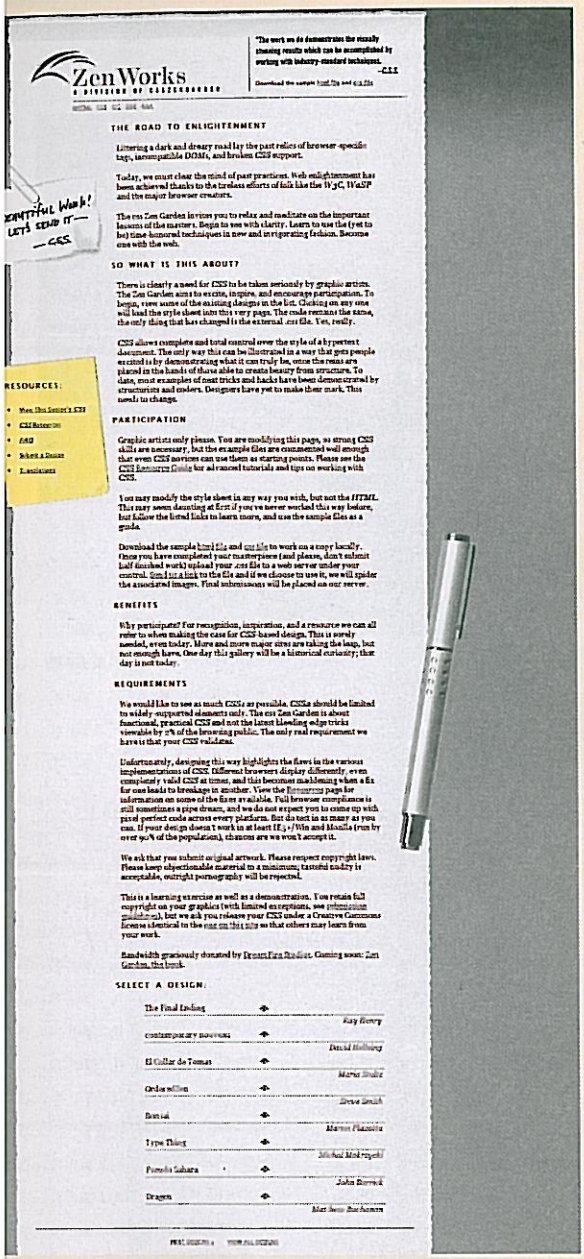


图7-3-8 和#container div失去定位关系后的#linkList

Tip

定位选项在布局中十分有效。它能够直接改变绝对定位的参照点。如果绝对定位元素的任意一个祖先元素都没有应用某种定位，那这个元素的参照点将从根元素html的边界开始。然而若是某个祖先元素已经应用了某种定位，那绝对定位就将以这个祖先元素的位置为参照点。

如何利用这个功能来处理父元素没有设置定位属性的情况呢？只需要为父元素设置position:relative而不为其设置top或者left值即可。这样既不会改变该父元素的原始位置，同时又能将其设为当前元素定位的参考点。关于该功能的更多细节，请参考Doug Bowman的文章“Making the Absolute,Relative”（www.stopdesign.com/articles/absolute）。



Derek Hansen，设计师
www.csszengarden.com/095

7.4 Corporate ZenWorks

从竞争到和谐

在第一次接触CSS禅意花园时，Derek Hansen就深深折服于人们在其中付出的努力及其体现出的创意。此时他正致力于发掘出一种标准化的网页制作方案，自然很快想到了向禅意花园提交自己的作品。

在灵感的指引下，Hansen使用了一组崭新的创意为禅意花园带来了新的亮点。他首先从仔细地研究花园中的现有作品开始。在被各种主题所深深震撼的同时，他决定让自己的作品讲述一个故事，力求将其打造成禅意花园中的另一部标新立异之作。

7.4.1 从一封信开始

就如所有典雅精致的文案一样，Corporate ZenWorks有着简洁工整的页面布局（图7-4-1），使用了银色的页面背景和简洁的字体作为基调：

```
body {
  background: #999 url(bk.jpg) repeat-x fixed;
  margin: 0;
  padding: 0;
  color: #33140E;
  font-family: Georgia, Times, serif;
  font-size: 80%;
}
h1,h2,h3,h4,h5,h6 {
  border: 0;
  margin: 0;
  padding: 0;
  font-family: "Lucida Grande", "Lucida Sans Unicode", Verdana,
  ➥sans-serif;
  font-weight: normal;
}
h3 {
  text-transform: uppercase;
  font-size: 90%;
  word-spacing: 2px;
  letter-spacing: 2px;
  margin-left: -10px;
  font-weight: bold;
}
```

注意到h3元素使用了负的外边距（图7-4-2）。看上去虽然有些怪，但它却给最终的作品带来了亮色，至于其中的原因，我们将在稍后阐释。

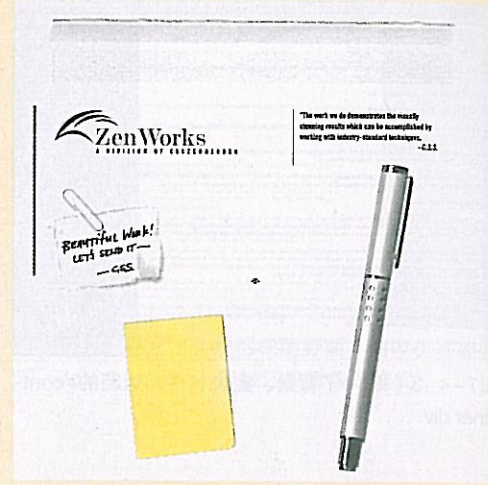


图7-4-1　Corporate ZenWorks中的图像

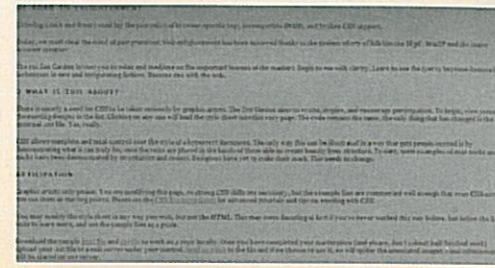


图7-4-2　背景和字体一角。可以看到由于负外边距的缘故，文字非常靠近窗口的边缘，标题的一部分则已经被边缘截去

Note

标题使用了Lucida Grande字体、正文则用Georgia字体，这让Corporate ZenWorks的文字显得简洁而优雅。Hansen解释说，他之所以偏重用字体而不是图像实现设计效果，是因为感觉到了一丝约束——担心由于过度使用图像而导致页面文件过大。而正是得益于这些限制，作品中的字体才更真实地切合企业文书中的真实风格。

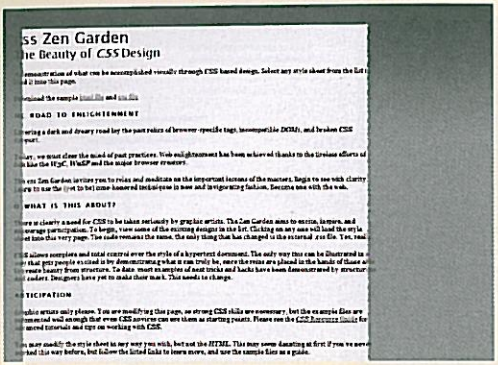


图7-4-3　应用了背景、宽度、内外边距的#container div

7.4.2　纸张的设计

下一步开始着手纸面部分，将文字排列成合适的顺序。我们可以通过为#container div定义如下的样式定义来完成：

```
#container {
  width: 750px;
  background: url(paper_mid.gif) repeat-y;
  margin: 50px 0 0 25px;
  padding-bottom: 20px;
}
```

以上样式将container区块设置为750像素宽，同时加入了纸张的图像并将其设置为横向平铺的背景图案，另外也为#container设置了外边距和内边距以留有一定的空白。不过外边距的使用也导致了h3元素过于靠近纸面的边缘（图7-4-3），我们将在稍后修正这个问题。

7.4.3　企业标识

由于页面模仿信纸的样式，所以还需要提供公司标志和相关的图案。作品中把标志置于#pageHeader的h1元素上：

```
#pageHeader {
  height: 100px;
  padding: 20px 0 20px 50px;
}
#pageHeader h1 {
  height: 90px;
  width: 340px;
  background: url(zen_logo.gif) no-repeat;
}
```

随后，下面的CSS将文字隐藏，从而只显示出标志图案：

```
#pageHeader h2 span, #pageHeader h1 span {
  display: none;
}
```

此时正好可以为标题和正文添加外边距，将其移至页面的中心部位：

```
#linkList, #supportingText, #preamble {
```

```
    margin: 10px 100px 10px 200px;
  }
```

这样，企业标志也恰好位于我们想要的地方，即页面的左上角（图7-4-4）。在标志旁显示了那条关于CSS的引用。而这也是通过为#quickSummary应用样式实现的，包括设置引用的位置、样式，并且隐藏了默认的文字：

```
  #quickSummary {
    position: absolute;
    top: 50px;
    left: 450px;
    background: url(subtext.gif) no-repeat;
    height: 100px;
    width: 280px;
  }
  #quickSummary p.p1 {
    display: none;
  }
  #quickSummary p.p2 {
    padding: 78px 0 0 15px;
    font-size: 10px;
    font-family: "Lucida Grande", "Lucida Sans Unicode", Arial,
    ➥sans-serif;
  }
```

在应用了如上的图像和样式后，页面的整体感觉就如同一张正式的企业信函（图7-4-5）。不过美中不足的是信纸的顶部边缘略显僵硬。所以Hansen又通过html源文件中定义的额外的div为僵硬的边缘增添了一些活泼：

```
  #extraDiv1 {
    position: absolute;
    top: 10px;
    left: 23px;
    width: 750px;
    height: 50px;
    background: url(paper_top.gif) no-repeat;
  }
```

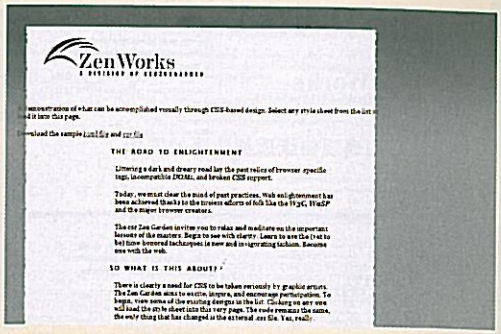
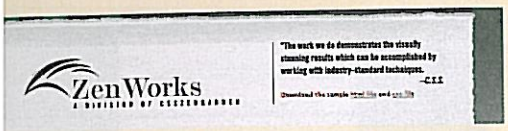
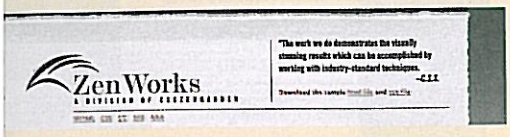


图7-4-4　添加了企业标识的页面。图中也可能看到，#quickSummary区域还没有应用样式

图7-4-5　强化后的信纸头部



7-4-6 添加表现信纸顶部边缘的图像



7-4-7 让#footer作为页首的一部分

Note

如果能够修改页面的XHTML，那么最好将#footer改名为一个符合它实际功能的标记。在该作品中，之所以#footer元素奇怪地出现在页面的顶部，纯粹是因为禅意花园的HTML源代码就是这样组织的。不过这也从侧面说明了使用CSS可以把元素定位在页面的任何一个位置。

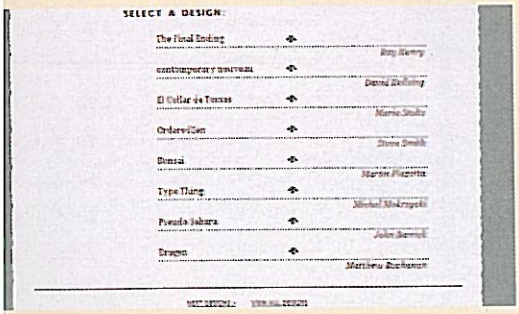


图7-4-8 应用了样式之后的作品存档链接

注意到绝对定位保证了信纸能和页面其他内容正确对齐（图7-4-6）。而作为标识部分的结束，#footer部分被移动到了页首图像的下方：

```
#footer {
  position: absolute;
  top: 160px;
  left: 150px;
  font-size: 80%;
  border-top: 1px solid #6E512F;
  padding-top: 3px;
  width: 575px;
  text-transform: uppercase;
  text-decoration: none;
  font-family: "Lucida Grande", "Lucida Sans Unicode", Arial,
  ➥sans-serif;
}
```

上述样式为#footer添加了链接和分割线，也为作品的标识部分画上了句号（图7-4-7）。

7.4.4 加入链接

Hansen决定让每个链接区域表现出不同的视觉效果。由于已经设定过#linkList的外边距，所以我们可以直接为其添加“Select a Design”的样式。读者可以通过在CSS源文件中查找#lselect ul和#larchives ul选择器找到这部分代码。这些样式让页脚部分的作品列表和作品存档链接显得极其引人注目（图7-4-8）。

7.4.5 人性化气息

谁说以企业为主题就必须要极端庄重？Hansen就在他的作品中加入了一些人性化的设计。他首先用#pageHeader h2选择器在页面中加入了一张小纸片：

```
#pageHeader h2 {
  background: url(paper_clip.gif) no-repeat;
  width: 320px;
  height: 262px;
  position: absolute;
  top: 250px;
```

```
    left: -6px;
    z-index: 5;
  }
```

Hansen又一次使用绝对定位来将纸片精确定位在他选择的地方。而在纸片下还有一张便条纸，用来提供资源链接。便条纸的样式定义可以在CSS源文件中找到，下面的代码仅仅用来演示便条纸是如何加入到页面中的：

```
  #lresources {
    position: absolute;
    top: 600px;
    left: 10px;
    width: 200px;
    height: 257px;
    background: url(sticky.gif) no-repeat;
     font-family: "Lucida Grande", "Lucida Sans Unicode",
Arial,
    ➡sans-serif;
    color: #000;
  }
```

现在，页面的左侧加入了两项绝对定位元素，极大地增强了页面的真实感（图7-4-9）。而最后加入的钢笔则使作品的整体形象更为丰满，钢笔本身是通过#extraDiv2选择器绝对定位于页面右部的（图7-4-10）。

```
  #extraDiv2 {
    position: absolute;
    top: 900px;
    left: 700px;
    background: url(pen.gif) no-repeat;
    width: 140px;
    height: 547px;
  }
```

7.4.6 完稿

Corporate ZenWorks依靠浓重的主题色彩构建出了一个绝无仅有的独特站点。尽管之前Derek Hansen曾遇到过诸如字体限制、GIF图像质量和PNG文件兼容性等问题，但作品的最终整体效果仍非常精致优雅。

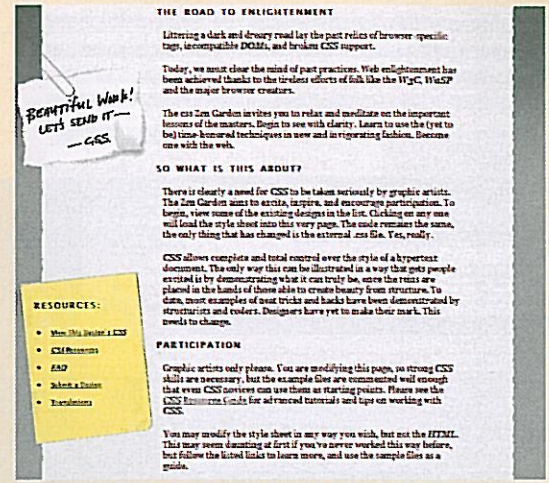


图7-4-9　使用绝对定位的纸片和便条纸

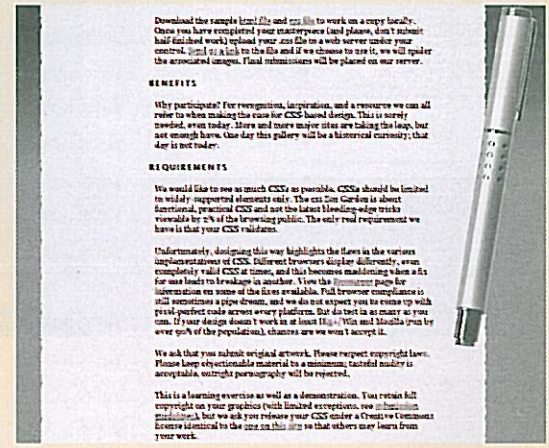
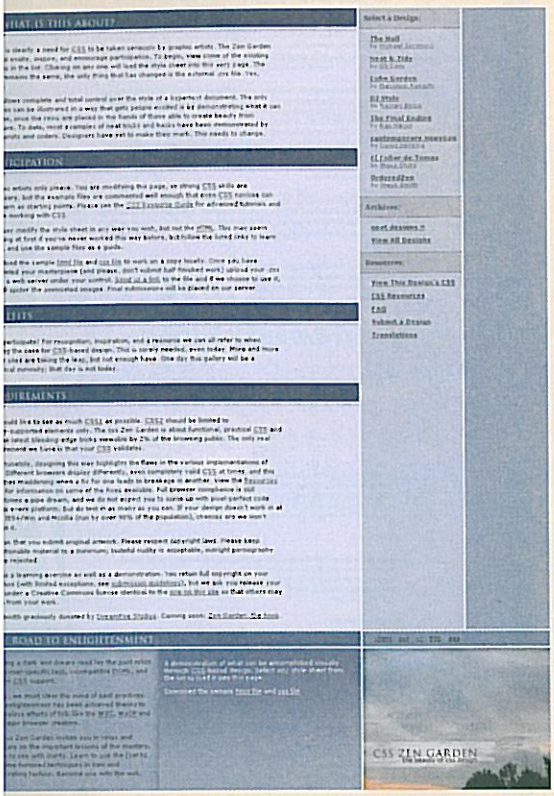


图7-4-10　钢笔的造型打破了作品中的严肃气氛



Ray Henry，设计师
www.csszengarden.com/090

7.5 Open Window（敞开窗口）

同一设计，三种布局

Ray Henry在抓拍景色的那一刹突然捕获了创作Open Window的灵感。为了避免作品落入俗套，Ray Henry在创作过程中选择了另辟蹊径。他摒弃了固定导航菜单、广告条幅等传统模式，尝试将关键元素置于页面底部，从而造就了一种令人耳目一新的风格。

但由于Windows平台上的IE浏览器不支持固定定位，所以Henry在创作中不得不花费额外的精力以解决浏览器兼容性问题，为了确保作品在各种现代浏览器都能正常显示，最终他还是不得不使用了一些CSS hack（图7-5-1）。

7.5.1 基本部分

Henry从body和container元素入手勾画整部作品。他首先禁用了默认的body内外边距和字体设置：

```
body#css-zen-garden {
    background:#A1B4CD;
    margin:0;
    padding:0;
    font-family:verdana, arial, sans-serif;
    font-size:11px;
}
```

之后，Henry将#container的width设置为100%，实现能够填满整个窗口的流式布局：

```
#container {
    width:100%;
    text-align:left;
}
```

在开始下面的内容之前，页面布局在各个浏览器下的样式都是一致的（图7-5-2）。

7.5.2 修正导言部分

在对#intro进行布局之后，作品的设计的过程开始变得耐人寻味起来：

```
#intro {
    width:100%;
    height:276px;
}
html>body div#intro {
    position:fixed;
    bottom:0px;
    right:0;
}
```

图7-5-1　Open Window中使用的图像。包括那张带给作者创意灵感的照片

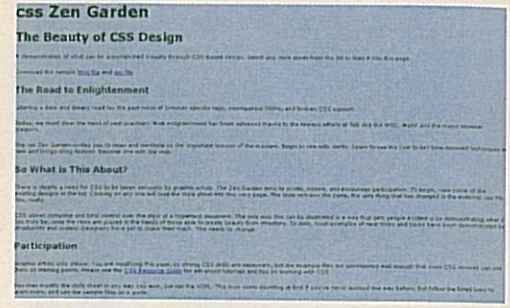


图7-5-2　在完成基本设置之后，所有的浏览器中的样式都能保持一致

图7-5-3　在IE中，#intro出现页面的头部，且底部有一条白色边线

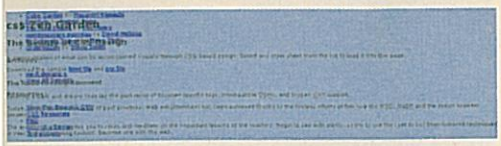


图7-5-4　在Mozilla（和Safari）中，#intro固定于页面底部，可以看到它目前遮挡住了部分其他文字

```
html*#intro {
    border-bottom:3px solid #fff;
    position:fixed;
    bottom:-10px;
}
#intro {
    height:286px;
    voice-family: "\"}\"";
    voice-family: inherit;
    height:auto;
}
```

在上面这些CSS代码中，我们可以看到很多专为某个特定浏览器设计的特殊样式。第一条样式中的#intro选择器对所有浏览器都有效。接下去的两条样式则只对支持固定定位的“高级”浏览器有效。第三条定义了border属性，Windows平台上的IE 6也能正常支持。而在最后一个样式中，Henry使用了一些盒模型的hack，用来修正IE 5对#intro区块height属性的错误理解。

可以看到，目前在IE中#intro区块位于页面上方（图7-5-3），而在Mozilla和Safari中，#intro却位于页面的底端（图7-5-4）。

7.5.3　上、下、左、右

目前为止，页面的顶部和底部均已定义完成。当然，事实上顶部和底部是同一个部分，只不过通过使用一些过滤规则让其在不同浏览器中出现在不同的位置而已。

1. 页面右侧

在制作页面右侧内容时，Henry首先加入了一张图像，以确定页面的色彩基调和图像布局：

```
#pageHeader {
    background:url(pageHead_bg.jpg) no-repeat top left;
    width:332px;
    height:283px;
    position:absolute;
    right:0;
    top:0;
}
```

以上代码在IE（图7-5-5）和Mozilla中都能正常地定位#pag-eHeader元素，不过其具体位置则取决于浏览器对固定元素的支持程度（图7-5-6）。Opera则尚不能显示这张图像，我们还需要继续努力。

2. 页面左侧

下面的CSS代码定义了#preamble的样式，这里我们对次要细节进行了删减，意在重点演示内容部分的布局方法：

```
#preamble {
    background:#A1B4CD url(preamble_bg.gif) repeat-y top right;
    height:283px;
    width:312px;
    margin:0;
    padding:0;
    position:absolute;
    left:0;
    top:0;
}
html>body #preamble {
    position:absolute;
    left:0;
}
```

可以看到，其中第一个#preamble选择器设置了区块的位置（图7-5-7）。之后的那条样式使用了子选择器，因此在支持子选择器的浏览器中，#preamble将会被绝对定位于页面的左侧。（图7-5-8）。

3. 页面中部

随后，Henry又将#quickSummary区块放置在#pageHeader和#preamble之间：

```
#quickSummary {
    background:url(quickSum_bg.gif) repeat-x top right;
    border-right:3px solid #fff;
    margin:0 332px 0 312px;
    padding:52px 0 0 0;
    height:283px;
    voice-family: "\"}\"";
    voice-family: inherit;
    height:231px;
}
```

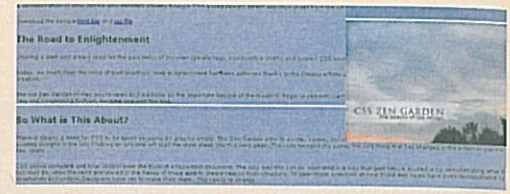


图7-5-5　图像在IE浏览器中位于页面右上方

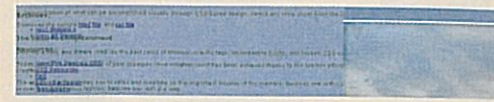


图7-5-6　图像在Mozilla中也位于右上方，不过尚不能完全显示出来

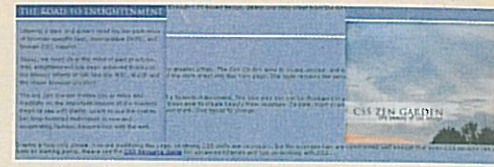


图7-5-7　#preamble区块在IE中位于页面的左上角

图7-5-8　图像在Mozilla（和Safari）中位于页面的左下角。#pageHeader部分尚无法完全显示

Note

若想了解更多“Box Model Hack”的细节，请参考http://tantek.com/CSS/Examples/boxmodelhack.html。

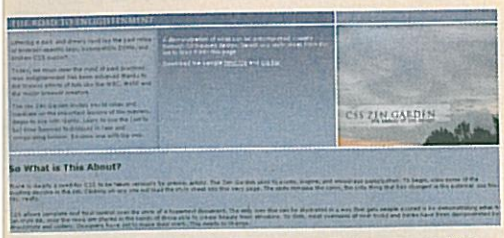


图7-5-9　IE中的效果，注意标题位于页面顶部

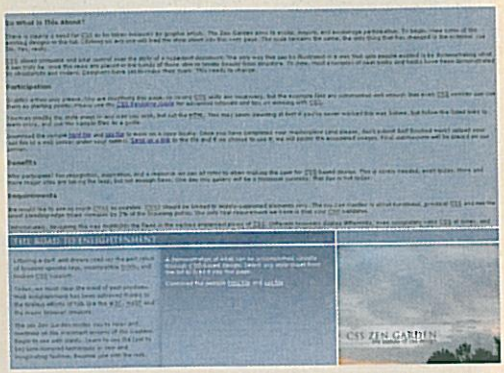


图7-5-10　在Mozilla中，相应的区域则显示在页面底部

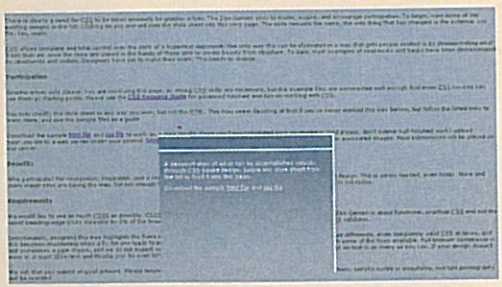


图7-5-11　Opera中则只出现了中间的一小部分

这里唯一用到的技巧就是#quickSummary声明中出现的盒模型hack，用来修正无法正确支持盒模型的浏览器中对宽度的解析。

在使用了盒模型hack后，3种布局的显示问题得到了初步解决。可以看到：在IE中，图像将显示在页面的顶端（图7-5-9），而Mozilla中将显示于页面底部（图7-5-10）。你同样会发现在Opera中，问题也似乎有了好转的迹象（图7-5-11）。

7.5.4　为内容和链接应用样式

接下去需要为页面的内容和链接部分应用样式。读者可以在禅意花园的站点上下载到这部分的完整样式规则，书中限于篇幅，只能列出一部分最有价值的代码。其中的第一条使用了浮动特性，将内容部分置于页面的左侧。

```
#supportingText {
    width:100%;
    float:left;
    margin-right:-332px;
    margin-top:-3px;
}
```

还有一处值得关注的地方就是有关外边距的设置。在上面的样式中我们可以发现，即便宽度被设为100%，仍可以用负的外边距调整文字区域的显示样式。负的右外边距的作用和正的右内边距相同，都可以保证文字不会超过右侧图像的左边缘。但在这里，我们却不能将右内边距应用于width为100%的情况，否则将导致#supportText的宽度为100% + 332px。接下来是#footer的样式，可以看到其中也考虑到了浏览器兼容问题：

```
#footer {
    background:#d6e0ee;
    margin:0 332px 0 0;
    border-right:3px solid #fff;
    border-top:3px solid #fff;
    border-bottom:3px solid #fff;
    padding:15px 0;
    text-align:center;
}
html>body #footer {
    background:transparent;
    position:fixed;
    right:0;
    bottom:261px;
```

```
    padding:0;
    margin:0 160px 0 0;
    border-right:0;
    border-top:0;
    border-bottom:0;
    text-align:left;
}
```

这个样式在IE中会将页脚部分定位于页面的顶端（图7-5-12），而在Mozilla中则会将其定位于页面的底部（图7-5-13）。

7.5.5 解决Opera中的问题

目前为止，整个设计在IE、Mozilla和Safari中都没什么问题。但前面曾提到过，我们还要为Opera进行一些调整，以便让顶部区域能够正常显示。下面这段CSS代码就是专门为Opera准备的：

```
@media all and (min-width: 0px){
  div#intro {
    position: absolute;
    right:0;
    width:100%;
    height:283px;
    top:0;
    z-index: 0;
    border-bottom:3px solid #fff;
      {
}
```

这里用到了CSS 3规范中的“Media Queries”技术，直接向Opera 7发送了属性设置。可以看到在应用了这段代码之后，作品在Opera中的效果也非常令人满意。

7.5.6 三位一体

Henry的最初目标是使作品在Mozilla和Safari中能正确显示；然后又用一些技巧满足了IE的要求；最后还为Opera找到了折衷的方案，将图像顺利地置于顶部。不过还有一点需要注意的是，Opera版本的正文只是在顶部图像的下方滚动，图像本身的位置则始终保持固定，这和IE中的情况又有所不同。这也正是Henry在实验中发现的一种令人耳目一新的布局方法，并最终让他确定了作品的名称——Open Window。

IE中的#footer

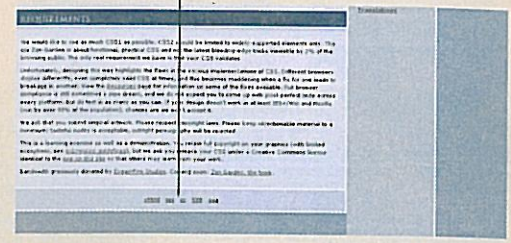


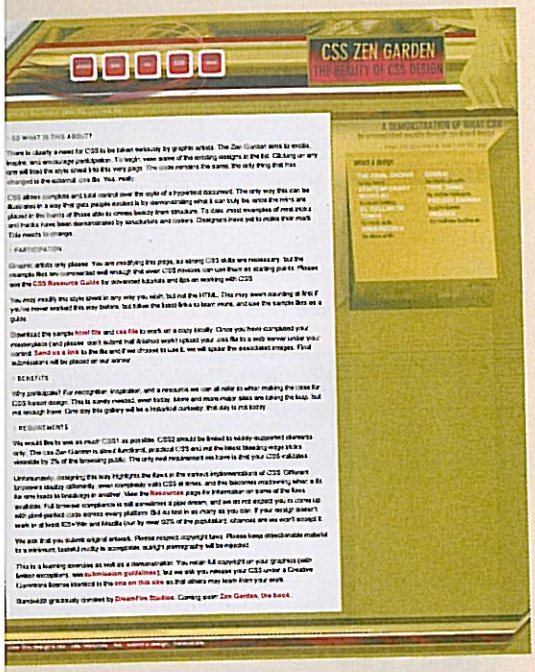
图7-5-12　在IE中，#footer位于正文部分的底端

Mozilla中的#footer

图7-5-13　在Mozilla中#footer是固定底部的一部分

Note

若想了解Opera 7中CSS 3 @media技术的更多信息，请参考www.virtuelvis.com/archives/2004/01/css-for-opera-7。



Dave Shea，设计师
www.csszengarden.com/025

7.6 mnemonic（记忆）

流式布局和高级效果的技巧

明亮的线条和几何造型交织流动于背景中，束状的细线覆盖于水平的条带之上组成了标题，设计师Dave Shea正是用这样一组明快的亮色构成了作品mnemonic的基调。

Shea试图用这部作品带来一些新的设计理念，而他的作品也的确展现了一种独特的布局。或许mnemonic并不像其他作品那样有特别深刻的内在美，但其中一致的视觉观感和显眼的主体内容使作品仍称得上相当优秀。

在这部作品的设计和编码过程中体现了作者很多灵感和创意。在这里，我们强烈推荐阅读带有注释的原始CSS文件（www.csszengarden.com/025/025-comments.css）。至于作品中所需的图像，可在下一页的图7-6-1中找到。

7.6.1 解决问题

在完成基本样式（下面的代码没有列出这部分样式，但读者可以在图7-6-2中看到应用样式后的效果）之后，第一个问题随之出现——这里我们需要2张不同的主体背景，一张水平平铺的背景和另一张叠放于其上的垂直平铺背景。

```
body {
 font-size: small;
 font-family: arial, sans-serif;
 background: #B0A40B url(bg-1.gif) top left repeat-x;
 margin: 0px;
 padding: 0px;
}
```

在应用上述CSS之后，body元素即被赋予了水平平铺的背景（图7-6-3）。但CSS不支持为同一元素设置多个背景图像，所以唯一的选择只能是使用两个元素。所幸的是，#container元素恰好位于body内且包围着其余元素，非常适合放置第二张图像：

```
#container {
 background: transparent url(bg-2.gif) top right repeat-y;
}
```

这样也就成功地完成了该技巧。在这部作品中，页首部分聚集了大量的图像，所以下一个挑战就是如何将这些图像聚合成一条位于页面顶端的、可伸展的条带，且同时还必须保证所有图像都有统一的视觉效果。

制作图像

通过观察最终的作品，我们可以发现mnemonic中包含了大量视觉元素。那么究竟是该选择绝对定位方式，将每个元素置于其他元素之上，还是把所有的元素合并为在同一张大图像中呢？

要想解决这个问题，我们就必须在两个极端中做出选择——要么找出相关元素合并成一个JPG文件，要么就将不同元素分别存为各自的GIF文件。这个问题的最终取舍取决于个人喜好和站点的限制，而没有什么绝对“正确”的答案，因此不同的设计师很有可能得出迥异的结果。

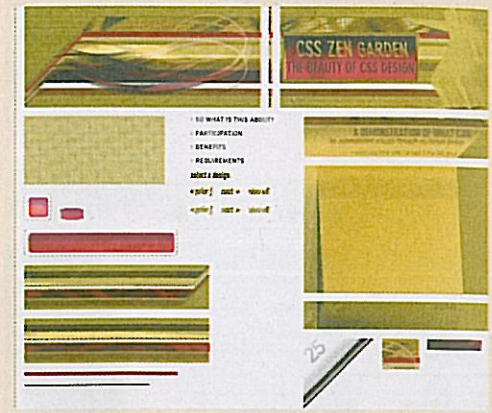


图7-6-1　mnemonic中用到的所有图像

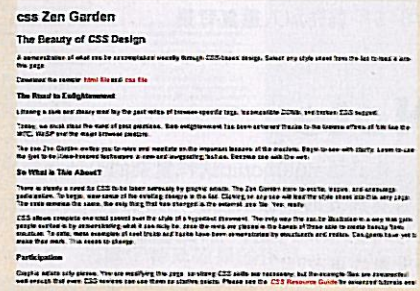


图7-6-2　基本样式，包括链接颜色和字体

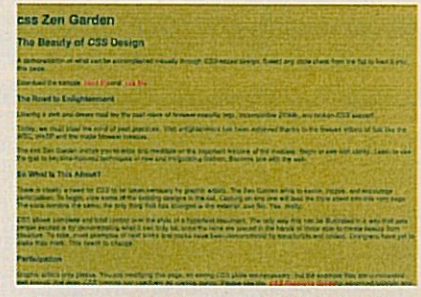


图7-6-3　应用于body元素的背景图像

图7-6-4　mnemonic中的页首图像

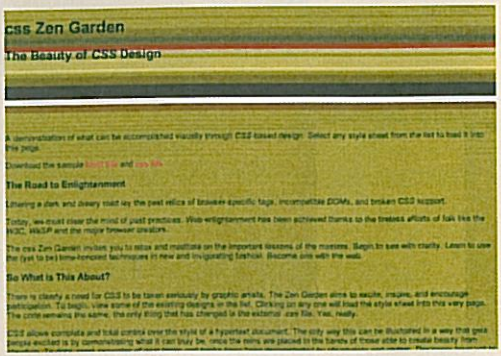


图7-6-5　首先加入重复背景

Note

在开始mnemonic这样复杂的设计之前，要有着良好的布局规划。如果到CSS编码阶段才考虑布局问题，将难免需要不停返工，做大量的更改。

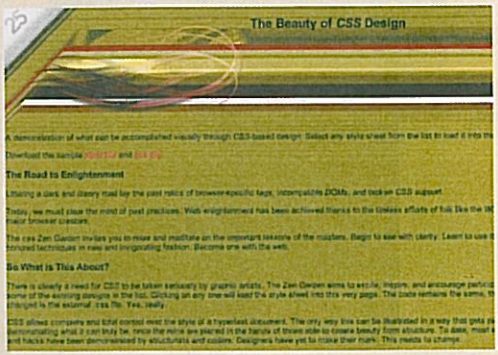


图7-6-6　页首左侧的图像被放置于一个闲置的标题元素中

我们知道，禅意花园的HTML文件不可更改，所以mnemonic只能使用#pageHeader中的两个标题元素（h1、h2等）和一个div元素载入图像。由于总共只有这3个元素，所以我们最多也只能放入3张图像。考虑到这些，作者将顶部左侧的图样（除去左上角）保存于一张图像中，将顶部的右侧存为另一张图像，而第三张则是个较窄的GIF图像，用于页首部分的重复背景。

有一点可能让人疑惑，就是mnemonic如何通过使用固定尺寸元素创建变宽布局。一般情况下，固定尺寸元素更适合固定宽度布局，而对于流式布局的mnemonic，位于页面左右两侧的定宽图像会在中间形成一段空白间隙。Shea聪明地使用了条带状的第三张背景图像填充了这条空隙（图7-6-4），这样也就保证了流式布局不会被破坏。

在确定了使用哪些图像之后，接下来就是将它们合成起来。其中页首重复的图案应被布置在其他元素之后（图7-6-5）：

```
#pageHeader {
 height: 171px;
 padding: 0px;
 background: transparent url(bg-4.gif) top left repeat-x;
}
```

虽然禅意花园预先定制HTML对图像的应用会产生或多或少的限制，但在将一些图像合并之后，我们依然能够留出足够多的用于装载图像的HTML元素。由于在mnemonic的设计中，#pageHeader里的h1和h2的文字只需绘于一幅图像（位于头部的右侧的那幅）中即可，所以另一个标题元素就能节省下来以放置其他图像。在该作品中，作者用它放置页面左侧的图像（图7-6-6和图7-6-7）：

```
#pageHeader h1 {
 padding: 0px;
 margin: 0px;
 float: left;
 width: 396px;
 height: 171px;
 background: transparent url(cr-tl.jpg) top right no-repeat;
}
#pageHeader h1 span {
 display: none;
```

```
}
```

在加入左侧图像后，Shea将右侧带有文字的图像置于h2元素中：

```
#pageHeader h2 {
 padding: 0px;
 margin: 0px;
 position: absolute;
 top: 0px;
 right: 0px;
 width: 350px;
 height: 171px;
 background: transparent url(cr-tr.gif) top right no-repeat;
}
#pageHeader h2 span {
 display: none;
}
```

7.6.2 向前推进

在这里我们略去了布局构建的详细过程，而直接转向作品中应用的高级效果。如果曾查看过作品CSS源文件的话，可以发现在文件的尾部有一处注释标识了高级效果的开始。若是移除了该标识后的代码，页面效果将如图7-6-8所示。事实上，这也是作品在IE中的最终效果。而如果使用的是Firefox或者Safari，那看到的效果将如图7-6-9所示。

1. 问题的本质

对于某些没有赋予class或者id的元素，可以使用CSS 2中新增的各种选择器为它们设置样式，当然并不是所有的浏览器都支持这类功能。但Shea在创作中使用这种技术的原因也很简单——他无法用CSS1中的常规选择器选中某个元素。若是能够修改禅意花园的HTML源文件的话，则只需为HTML元素加入更多标识符，问题即可迎刃而解。

因而在撰写HTML时，需要在简洁的HTML代码和足够多的标识符之间找到平衡点。禅意花园在设计HTML文件之初已经尽力为选择元素留下了相当高的灵活性，但事后人们却发现这样的灵活性在某些情况下依然不能满足需求。当然，在通常的项目中，HTML和CSS的设计是同步的，读者可以自

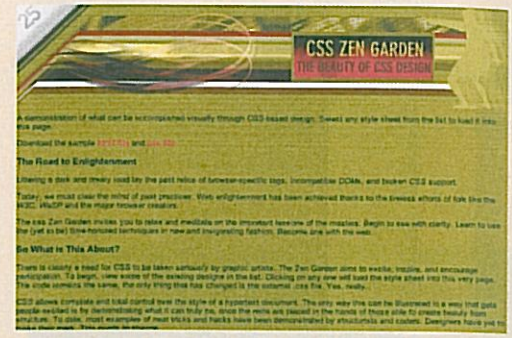


图7-6-7　h1和h2中的文字被置于同一个元素中

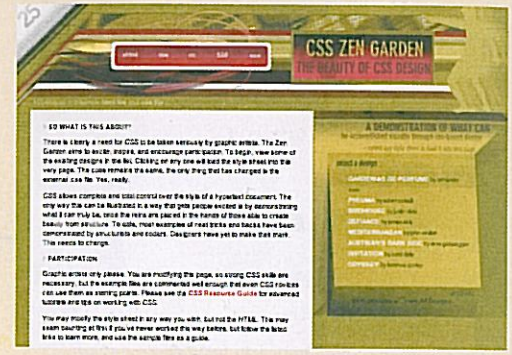


图7-6-8　去除了特殊效果的作品。这和IE浏览器中的最终效果是一致的

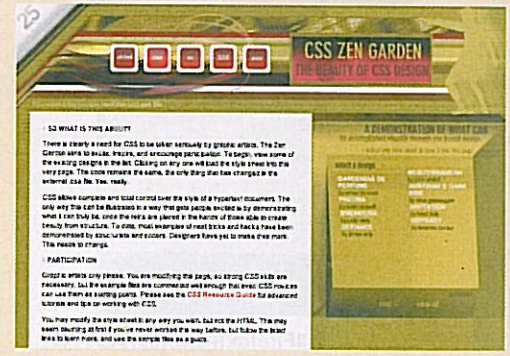


图7-6-9　包含特殊效果的作品

图7-6-10　IE中显示的作品列表

行把握标识符的多少，以实现最终效果。

2. 作品列表

从图7-6-10中可以看到，IE中作品列表呈垂直排列，而在Firefox和Opera中则成对排列且略带倾斜（图7-6-11）。若想实现第二种效果，我们就必须逐个选择每个列表项，然后将其分组排列。作者所使用的方式是一组逐渐增长的子选择器和邻接选择器。这两种选择器均定义于CSS 2中，但不为IE所支持：

```
#lselect .ul>li {
 position: relative; left: -14px;
}
#lselect ul>li+li {
 position: relative; left: -12px;
}
#lselect ul>li+li+li {
 position: relative; left: -10px;
}
...
```

Note

省略号代表之后还有部分代码。

在上述代码中，第1个选择器选择了#lselect中的ul元素的任意li子元素。第2个选择器和第一个没有太大区别，只是规定这个li元素必须有一个li元素作为前驱。第3个则规定必须有2个li元素作为前驱，第4个需要3个li前驱……依此类推。就是通过这样一组层叠的设置，将每条设置覆盖前一条设置中的属性，最终为8个li元素逐个定位。回过头来看，不得不说这个复杂的方案仅仅是用来解决一个原本应该极其简单的问题——因为在多数情况下，都可以用加入额外的id和class解决该问题，而无须如此冗长的CSS结构。我们也不建议在实际开发中书写类似的CSS。

图7-6-11　Safari和Firefox中显示的作品列表

3. Previous/Next链接

作品中遇到的最后一个难题是，如何用一图像表示的文字替换作品列表中呆板的默认链接（图7-6-12和图7-6-13）。而这个需求也同样面临着缺少元素标记的问题，因此，作品最终的实现已不仅限于子选择器和邻接选择器，还使用了一些更加高级的属性选择器。在禅意花园的HTML源代码中，可以看到Previous/Next链接都被赋予了不同的HTML accesskey：

```
<a href="http://www.mezzoblue.com/zengarden/alldesigns/"
title="View every submission to the Zen Garden. AccessKey:
w" accesskey="w">
```

找到了这样的特征之后，问题即可迎刃而解。下面是其具体的实现方式：

```
#larchives ul>li+li a[accesskey="w"] {
  display: block;
  width: 35px;
  height: 0;
  overflow: hidden;
  padding-top: 18px;
  background: transparent url(archive-b3a.gif) top left
no-repeat;
  position: absolute;
  top: 16px;
  left: 170px;
  color: #958500;
}
```

顾名思义，属性选择器用于选择具有指定属性值的元素。在上面的代码中，相关的语句已经高亮显示。由于需要应用样式的a元素的accesskey属性值为w，所以上述代码即可轻松地找到相应元素并随后应用样式。

这时，细心的读者或许会产生这样的疑惑：之前的作品列表同样也有类似的accesskey，可为什么还是使用了一串冗长的子选择器呢？如果您能想到这一点，那实在太棒了。事实上，用accesskey选择元素也确实更为合理。

7.6.3 做出最恰当的选择

mnemonic的设计就是一个选择的过程。为了达到同样的效果，Shea完全可以在Photoshop中做出不同的图像切分，或者用其他什么方法来选取作品列表中的元素。而在网页设计的过程中，也总是充满了类似的选择和岔路，如果能明察所有的选项，那么定会做出最恰当的选择。

图7-6-12 IE中“Next Design”和“View All Design”的显示效果

图7-6-13 “Prior”、“Next”和“View All”在Safari和Firefox中的显示效果

结语

现在到了独自学习禅意花园的时候了，接下来应该走向哪里呢?

你或许已经在此浸淫多年，在CSS刚刚出现，还不甚实用的那些日子里就开始了尝试。你也可能是个新手，通过这本书才第一次认识了CSS。

无论属于哪种情况，你的目标都是一样的：希望能够让CSS做得更多、做得更好。本书的这一部分正是为了帮助你达到这个目标而作。

1. “CSS Crib Sheet”

如果曾用CSS创建过一两个网站的话，那么很容易发现浏览器对CSS的支持仍远远称不上完美。取决于作品的复杂程度，最终很有可能导致这样的结局：深陷于CSS的泥沼中，气愤得只想撞墙，并发誓今后再也不会去做Web设计，甚至会觉得哪怕是城市规划或者核物理，都要比CSS简单、友好得多。

若正处于这样的情绪中，那么请相信自己并不孤单。在我们背后，整个社区都在一刻不停地与基于标准设计中遇到的各种问题做着斗争，并分享着他们的成功经验。基于很多的现有资源，本书的合著者之一Dave Shea也整理出了一份资料，叫做“CSS Crib Sheet”。“CSS Crib Sheet”旨在为很多常见的CSS问题给出“一站式”的解决方案，下面我们就将其完整列出。“CSS Crib Sheet”的最新版本可以到www.mezzoblue. com/css/ cribsheet访问。

• 若是不确信的话，那么就去验证

在调试之前若能简单地验证一下代码，将为我们减少很多头痛的经历，节省很多时间。格式不正确的XHTML或CSS将导致很多布局问题。这时，使用验证工具可以很快找到并纠正这些错误。W3C提供了XHTML和CSS验证工具，可分别在http://validator.w3.org和http://jigsaw.w3.org/css-validator访问。

需要注意的是，由于不恰当的嵌套等原因，文档中较早出现的错误可能会导致随后更多处的验证失败，而这些验证失败之处却并不一定是真正的错误。我们建议首先修复一些明显的错误，然后重新验证：这时错误总数可能会直线下降。

• 创建并测试CSS时，首先在最高级、最先进的浏览器中进行，然后再考虑其他浏览器

如果首先基于那些较为古老的浏览器创建并测试CSS的话，那么代码将无法避免地依赖于这些浏览器中落后、错误的呈现规则。这样随后在那些更先进、更遵守标准的浏览器中调整时，将变得举步维艰——这无疑是一种倒退行为。

因此，若是首先基于先进浏览器编写，然后再为那些非标准浏览器调整的话，代码将从一开始就非常标准，调整的过程也将会非常容易。就目前的实际情况看来，应该首先在Mozilla（Firefox）、Safari或者Opera中测试，然后再在IE中调整。

• 若用浮动实现布局，请确保正确地清除了浮动

浮动的使用非常富有技巧性，而且它也并不总是那么“听话”。若某个浮动元素超过了预期的边界，或者不管由于什么别的原因与预期的表现行为不同的话，那么首先应该确认的就是预期目标的合理性与正确性。关于这个问题，请查看Eric Meyer在他的网站Complex Spiral Consulting上的教程（www.complexspiral. com/publications/containing-floats）。

• 为元素应用内边距或边框来避免外边距重叠

有时候元素之间留出了一条并不希望看到的缝隙，而另一些时候却无法让元素之间保持一段距离。若效果是由外边距实现的话，那么罪魁祸首十有八九就是外边距重叠现象。Andy Budd在他的网站中详细解释了这一现象（www.andybudd.com/archives/2003/11/no_margin_for_error）。

• 避免同时为元素指定明确的内边距/边框值与宽度/高度值

Windows平台上的IE 5.0和IE 5.5浏览器对盒模型的理解是错误的，因此当为元素指定了高度或宽度时，其结果往往不能如我们所愿。虽然对于这个问题已经有了一些权宜之计（请参考http://tantek.com/CSS/Examples/boxmodelhack.html），但最好还是从源头上避免。例如，若某元素需要固定的宽度/高度，那么既可以用其子元素的外边距值来撑大，也可以用其父元素的内边距值来约束。

• 避免IE 6的“无样式内容瞬间”现象

若用@import语句引入外部样式表，那么迟早会发现IE 6的“闪烁”问题，即未经CSS格式化的HTML代码短暂地出现在浏览器中。我们可以通过使用link元素，或在页面中添加script元素来避免这个问题。详情请参考BlueRobot.com网站上的这篇文章：“Flash of Unstyled Content”（www.bluerobot.com/web/css/fouc.asp）。

• 不要依靠min-width或min-height属性

Windows平台上的IE并不支持这两个属性，但它在某些情况下却把width和height属性认为是min-width或min-height属性。因此，借助于一些只为Windows平台上的IE设计的过滤规则，我们用width和height属性也可以实现min-width或min-height的效果。mezzoblue.com上的文章“min-height: fixed”（mezzoblue.com/archives/2004 /09/16/minheight_fi）给出了示范。

• 尝试减少百分比值

浏览器计算中的舍入误差有时会让50% + 50%等于100.1%，导致布局被破坏。这时请尝试略为减少百分比的值，例如将50%改为49%或者49.9%。

• IE中没有完整显示出内容?

若页面中的内容没有显示完全，但在鼠标指针悬停在某链接上时却能正常显示的话，那么十有八九是遇上了IE的“Peekaboo bug”。解决方案请参考Position Is Everything网站上的文章：www.position iseverything.net /explorer/peekaboo.html。

• 确保要实现的效果确实存在

有些浏览器对CSS进行了一些并不在官方规范中的扩展，例如IE中的滤镜和滚动条样式等，这些属性不能应用在其他浏览器中。若验证工具显示某些代码没有定义的话，那么这些效果将很有可能不会在各个浏览器中表现出一致的行为。

• 牢记“LoVe/HAte（爱/恨）”链接规则

在为链接元素指定伪类时，牢记以下的顺序：link、visited、hover和active。任何其他的声明顺序都可能会导致问题。若是还要考虑:focus伪类，那么可将该规则修改为LVHFA（可以用Matt Haughey给出的记忆法“Lord Vader’s Handle Formerly Anakin[1]”记住这个顺序）。

• 牢记“TRouBLED（麻烦的）”边框简写顺序

简写边框、外边距或内边距值时，必须按照从顶部开始的顺时针方向进行，即 top、right、bottom和left。例如margin: 0 1px 3px 5px;表示上边距为零，右边距为1px，依此类推。“TRouBLED”记忆法能够帮助我们牢记正确的顺序。

• 为非零值标明单位

在用CSS指定字体、边距或元素尺寸的值时，必须标明所用的单位（唯一的例外是 line-height属性，虽然我们也可以为其指定单位，但这并不是必须的）。某些浏览器对未标明单位值的处理方法不足为凭，我们不能认为其他浏览器也会这样做。零就是零，没有必要分辨是px还是em。但非零值却一定要标明计量单位。例如：padding: 0 2px 0 1em;

---

[1] 这句话描述的是著名科幻小说/科幻电影《星球大战》中黑武士Vader和天行者Anakin的人物关系。——译者注

• 测试不同的字体大小

无论在指定字体原始大小时使用了什么单位，Mozilla和Opera之类的高级浏览器都允许用户调整字体大小。肯定会有一些用户所使用字体大小与我们的不同，所以在声明字体大小时，我们就应该尽量考虑到这些可能出现的变化。

• 在HTML和CSS中统一大小写

在大多数浏览器中，CSS都是大小写敏感的。若是CSS和HTML中都使用了同样的homePage作为某Class的名称，则自然会良好工作。但若在CSS中将该规则改名为“homepage”，那么在一些严格遵守CSS规范的用户代理（例如Mozilla浏览器）中就会出现问题。

• 测试时使用嵌入样式，发布时再改为外部输入

若在测试时将样式表嵌入到HTML源代码中的<style>标签中，即可避免许多因浏览器缓存导致的不正常现象。对于某些较老的浏览器来讲，这一点至关重要。但在发布前，记住一定要将这些样式移至外部文件，然后用@import或<link>引入，否则将无法得到外部CSS带来的好处。

• 用明显的边框辅助调试布局

div {border: solid 1px #f00;}之类的全局规则可以帮助检查出很多细微的布局差错。为某个特定的元素加上边框也便于找到一些难以发觉的元素重叠或多余空白问题。

• 指定图片路径时不要使用单引号

设置背景图片时的引号并不是必须的。不但如此，若用单引号给出图片路径，那么苹果机上的IE将无法正确加载该图像。因此要么使用双引号，要么根本不用任何引号。

• 若使用了锚点（anchor），那么在为超链接应用样式时要格外小心

如果代码中使用了传统的锚点（例如<a name=“anchor”>），将发现:hover和:active伪类的样式也会应用于其上。我们可以使用独一无二的id（例如<id=“anchor”>），或是一种鲜为人知的奇怪语法:link:hover

和:link:active来避免这种情况。请参考www.dbaron.org/css/1999/09/links。

- 不要未雨绸缪地链接至作为“占位符”的空样式表（例如手持设备或打印用样式表）

苹果机上的IE 5对空的样式表比较敏感，页面的加载时间将因此变长。因此，我们建议样式表中至少应该有一条规则（哪怕注释也好），或者干脆就不要链接。

理解问题

理解问题才是修复问题的关键，这或许是我们能够提供的最有用的一个建议。想要了解CSS布局中的常见问题别无他法，唯有用时间为代价换来经验。

假设在Windows平台上的IE中发现某个浮动元素旁多出了3个像素的水平间距，且只有在该浏览器中才出现。如果并不知道这是IE在处理浮动元素时的一个缺陷，则可能会尝试使用负外边距值、相对定位或者为其他环绕元素应用一些CSS hack，以期弥合这个令人费解的间距。

但如果事先就了解了Windows平台上的IE的这个独特的缺陷，即自动为某些浮动元素添加3个像素的额外外边距，那么就不需要猜测并逐一尝试上述方法了。

尝试修复这类难以理解的缺陷所带来的另一个问题就是，这类缺陷是可以叠加的。如果已经在页面中添加了很多这样令人费解的补丁，那么整个页面结构将充满显示问题，再也无法稳定下来。若是早先的问题都没有妥善解决的话，那么开发中随后的调试过程将变得更加举步维艰。

但如果搞清楚了导致所遇到的某个缺陷的真实原因，那么除错的过程将会变得相当简单直观。“CSS Crib Sheet”只是一个开始，很多其他网站也都详细讨论了特定浏览器在呈现过程中的问题。这里我们强烈推荐从Position Is Everything（www.positioniseveryt hing.net）网站开始接下来的学习。

2. 其他资源

创建Web站点的人自然都喜欢以Web作为沟通方式，我们喜欢的一些站点和期刊都非常值得推荐。此外，大量的CSS资源以及面向标准的网站也正

等待着你的发现。

• CSS设计类Web站点

□ CSS Discuss

www.css-discuss.org

一个大容量的邮件列表，关注CSS新闻并提供CSS的相关帮助。无论是专家还是新手，都能够以同样的方式提出问题或分享技巧。

□ CSS at MaxDesign

http://css.maxdesign.com.au

集中了很多CSS相关的内容，特别专注于列表、浮动等问题。

□ CSS/Web Standards Links

www.andybudd.com/links/cssweb_standards

Andy Budd的CSS相关文章、技术以及缺陷修复列表，更新频繁。

□ Holy CSS Zeldman!

www.dezwozhere.com/links.html

Andrew Fernandez收集的CSS相关资源列表，这或许是能在网上找到的最全最好的资源集合。

□ Position Is Everything

www.positioniseverything.net

遇到了CSS布局问题？发现了新的浏览器缺陷？去找Big John和Holly吧——无论有什么问题，他们都能帮助解决。

• 期刊出版物

□ A List Apart

www.alistapart.com

Note

本书提及的所有URL，包括本页右边的这些，我们都给出了一个更加方便的可以直接点击进入的列表：www.mezzoblue.com / zengarden/book/links。

长时间以来一直是Web设计师首选的杂志。

□ Digital Web Magazine

www.digital-web.com

提供Web设计、软件以及书评等，内容丰富。

□ Web Standards Project Buzz

www.webstandards.org/buzz

Web标准相关的所有新闻，更新频繁。

• 图书

无论是专家还是新手，都应该仔细阅读如下图书。

- □ *Eric Meyer on CSS: Mastering the Language of Web Design*（2002年由New Riders出版社出版，ISBN 073571245X）以及*More Eric Meyer on CSS*（2004年由New Riders出版社出版，ISBN 0735714258），均为Eric Meyer所著。
- □ *Web Standards Solutions: The Markup and Style Handbook*，Dan Cederholm著（2004年由Friends of ED出版社出版。ISBN 1590593812）。
- □ *Special Edition: Using XHTML and HTML*，Molly Holzschlag著（2002年由Que出版社出版。ISBN 0789727315）。

提交自己的作品

在本书出版后，禅意花园仍旧欢迎各位提交新的作品。在研究了本书中各个示例作品之后，没有什么理由阻止你提交自己的创意（提交时别忘了提到这本书哦！）。

每一件新提交的作品都将经历严格的审查过程，首先按照一些因素进行评分，从1~37分不等，然后再由一群训练有素的猴子进行手工筛选……唔，开玩笑而已。实际上这只是个收录或不收录的选择——没有什么严格的标准。

the Zen of CSS design

Web视觉艺术设计的王者之书

通过结合使用少量的标记，级联样式表（CSS）可提供引人注目的视觉设计。与HTML驱动的老式布局相比，使用的代码更少，而效率和灵活性却高得多。受这种Web设计方法的创意启发，Dave Shea创建了“CSS禅意花园”（www.csszengarden.com）。其理念简单而独特：召集设计师在设计良好的HTML的基础上，通过应用CSS技巧创建极富想象力的作品，然后将其CSS代码粘贴在网站上，供他人研究和使用。在2003年创建之初，“CSS禅意花园”还默默无闻，而现在已发展到包含将近150件设计作品的规模，这些作品来自世界各地，树立了使用CSS设计高质量网站的标准。本书将引领读者探索“CSS禅意花园”中最激动人心的36件设计作品。

“这是我一直期待的书，将设计理念的精髓和网页设计的技巧巧妙揉合。这归功于独一无二的网站CSS Zen Garden（CSS禅意花园），它绝不是仅仅在旧蛋糕上裹了一层糖衣——这是一个全新的蛋糕，有着全新的美味，现在就开始享受吧！”

—— Eric Meyer

在这个花园中，您将体验到：

- 36件效果惊人的设计艺术作品
- 来自顶级Web设计师和开发人员的CSS秘诀
- 高超的文字使用技巧，使作品生气勃勃
- 绝妙的CSS技巧，轻松适应未来的变化
- 高效使用图形以提高网页的下载速度

本书奉献给：

设计师	迈向完美，走向巅峰
程序员	无设计知识，亦能运用自如
CSS新手	未丧失拓展知识的勇气
CSS高级用户	即使了如指掌，亦能受益良多

封面设计：胡平利

ISBN 978-7-115-16035-5

9 787115 160355 >

ISBN 978-7-115-16035-5/TP

定价：49.00 元

分类建议：计算机/网页制作

人民邮电出版社网址 www.ptpress.com.cn