China-pub.com
下载

第5章　导航理论与实践

导航是科学，是艺术，它让人们从一个地方到另一个地方。当我们旅行到一个遥远的陆地或走进大厅喝一杯水时，我们在真实的世界里航行。 Web不是一个实际的地方，用户利用导航的提示在信息的空间里移动。许多来自真实世界的导航观点在 Web中被采用。然而，站点设计者应该注意，Web不是真实的世界：直接的转换并不总是奏效。 Web导航应该帮助用户理解他们在哪里，他们能去哪里，他们怎样获得其他的东西。为使用户得到真实的感觉，必须充分考虑可见性、标记和导航元素的布局。导航是复杂的学科，本章将介绍导航的基本理论和导航标记的使用，按钮布局，通用的导航扩展如帧的使用。下面的章节将讨论链接的细微差异和导航辅助如搜索引擎和站点映象。

5.1　导航

在现实生活中，人们经常需要从 A点到B点。可能我们需要在邮局里取一个包裹，或出去呼吸新鲜空气。我们想迅速有效地到达目的地，而不迷路。这就是导航的焦点。导航就是帮助人们找到他们的路。

在航行时，人们经常问下面的问题：

- 我在哪？
- 我能去哪？
- 我怎样才能到达我要去的地方？

他们也问第二种与基本问题相关的问题。例如，迷路的人经常问自己：

- 我以前到过这吗？

或

- 我能返回我住的某地吗？

有时如果旅行很长时间，或如果一个小孩独自坐在汽车的后座上，他们可能问：

- 还有多久到达那里？

所有的问题都是正常的。遗憾的是，在Web上准确地回答这些问题是不容易的。始终要记住，至少在正确的形式下， Web缺乏真实世界的物理性。事实上，考虑到物理性的丧失，在 Web上确切的位置不像你想像的那么重要；用户更多地关心可能的未来方向和怎样到达他们想去的地方。一些专家暗示用户在网上浏览时有点像找食物的动物。用户作为信息的使用者，嗅出他们寻找的信息的气味或他们尽力完成的任务。一旦找到痕迹，他们就同它保持联系直到该信息消失。如果开始找不到，他们坚持到信息的再次出现。如果信息完全丢失了，他们可以迅速退到一个已知的安全地方如某个主页。搜寻信息的观念暗示确切的位置与用户在正确的路径上的感觉和他们的基本位置相比不是十分重要的。优秀的导航总是通过显示用户在正确的路径上的线索，

China-pub.com
下载

尽力回答用户的导航问题。这些是通过使用导航帮助系统如 URL、网页标签、标记网页和导航条等工具实现的。

5.2 我在哪？

由于缺乏中心参考点，用户知道他们在 Web中的位置是非常困难的。在军舰的导航世界里，经度和纬度被用来标定航海的过程。然而，这些概念依靠有限的地球，并假定分别从英格兰的格林威治和地球的赤道开始测量， Web包含一个绝对的中心吗？让我们假定雅虎是 Web中心。在任何时刻我们从雅虎出来要多少次击键？答案是一次和许多次。如果允许人们直接接受事物，或如果你必须跟随一个路径，那么它是真正的依靠。讨论的中心是在主要时间内 Web上的位置和距离不是精确的。 URL将给定一个精确的位置，但是关于某个文档的位置相对于另一个文档位置的说法很少，并且 URL不能被用户理解。用户依靠 URL、网页标签、颜色甚至文档类型来理解他们在哪。

5.2.1 Web上的精确定位：URL

今天，URL（统一资源定位器）定义了 Web上的位置概念。用户浏览器可以识别正常的网页如：http://www.democompany.com/products/trainer.htm。不管是否喜欢， URL精确回答了用户“我在哪？”的问题。遗憾的是，答案可能是无用的，或不能被用户理解。考虑先前的 URL。这个地址表明用户接受了叫做 trainer.htm 的网页，该网页在名叫 www.democompany.com的机器的 products 目录下。当然，这地址没有告诉我们的位置相对于其他网页的距离。根据 URL不容易判断网页间的联系是在站点内或是在站点外。例如，该网页可能是来自外国服务器上的一个网页。例如，考虑如果在最后的网页同 http://www.robotparts.co.nz/robots/bodyparts.htm有一个链接，一个虚构的在新泽西的机器人部件商。用户将从 URL中收集一些信息，如基本的物理位置（就像在地理地域内一样），服务器名称，也可能是一个目录或文档名。然而，不要期望 URL总是能帮助决定人们在哪。许多动态产生的站点有如下形式的 URL。

```
http://www.robotsforsale.com/store/showprod.cfm?&DID=7&User_ID=6185
1&st=1985&st2=-78872300&st3=211170630&CATID=3&ObjectGroup_ID=18
```

这不能保证用户将容易地被告之他们在哪，你应该努力使 URL容易理解。

规则：使用简单且易记忆的URL去改善导航。

因为URL显示位置，除非你尽力避免直接外部链接，否则不要隐藏。一些站点利用构架，这将不显示被观看的文档的 URL，或打开的窗口没有地址栏。因为用户可看到 URL，理解他们在哪，你应该考虑不隐藏 URL，除非你尽力避免深层链接或一个松散的站点结构，如同第 4章讨论的那样。

规则：不隐藏URL —— 复杂的或简单的 —— 除非你尽力让人们避免直接链接。

5.2.2 网页和站点标签

除了URL的确切定位外，用户还可以通过网页标签得到精确度稍差的定位。大部分网页包

China-pub.com 下载

含网页标签，用来标识网页内容且提供站点内的一些位置。一般地，网页标签放在网页顶部。设计者确信Web标签外观和导航条或内容是不同的。通常，这意味着标签或者更大，是不同的字体和颜色，或者是独自成组。网页标签的位置在网页之间应该是连贯的。

规则：在站点内利用连贯且清楚的网页标签标识所有网页。

不仅标识用户访问的特定网页重要，而且标识他们所潜在访问的站点也很重要。最普通的方法是在整个站点内使用有限公司或组织的标识或名称。站点标签的位置是不同的，可能最常用的位置是在屏幕的左上角。如果这是用户主要的扫描路径，他将帮助真正的用户。一些站点把它放在右边，但Web习惯上好像更喜欢左上角，这和许多应用软件的标题栏上程序图标的位置是一致的。文字处理软件的标题栏和网站的比较如图 5-1所示。

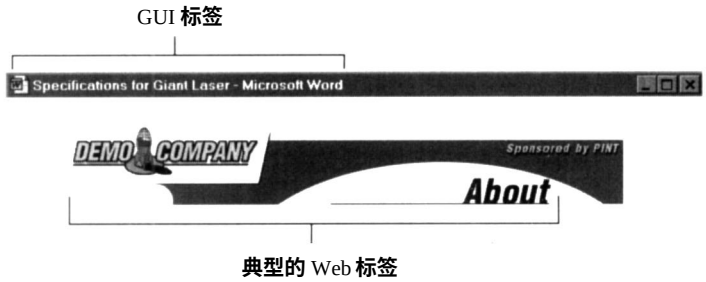


图5-1 标题栏的比较

注意，Web站点缺乏GUI应用程序的典型的最小化、最大化和关闭按钮，它不仅有表示站点的图标，而且有表示文档的标签。当然，Web站点和应用程序相比是高度风格化的，但标签趋向于一致。

不管站点标签的位置在哪里，应该有使用户返回站点主页的规定路径。对于迷路的用户，把它当作应急按钮，一个即时返回主页的方法。许多用户知道 Web习惯，但你应该考虑使用标题属性，以便当鼠标错过标识时，它会显示返回主页。这个观点将在下一章中详细讨论。记住，不必限制自己返回主页。许多站点提供更清晰的返回主页按钮，通常放在底部。

规则：在点击时，站点边侧的标签图标或者组织名或标识应该始终让用户返回站点主页。

另一个表示位置的方法是通过暗含的网页标签。当使用图形按钮时，正常的按钮将用不同方式表示。有时设计者将可以选择的按钮设置为明亮色，以便让用户知道他们处于特殊的网页上。然而，这并不是页面标签本身的角色，也不是可选状态的按钮的角色。事实上，按钮不应该是可以再选。用明亮的颜色修饰按钮，将暗示它比其他按钮更重要，而不是相反。这个常见的错误如图5-2所示。

回想典型的图形用户界面规则是使不再可选的按钮变为灰色，并且用户期望按钮按这种方式工作。

建议：按钮状态应被认为是网页标签的第二种形式，并且选择状态应该总是凹的，而不是凸的。

China-pub.com
下载

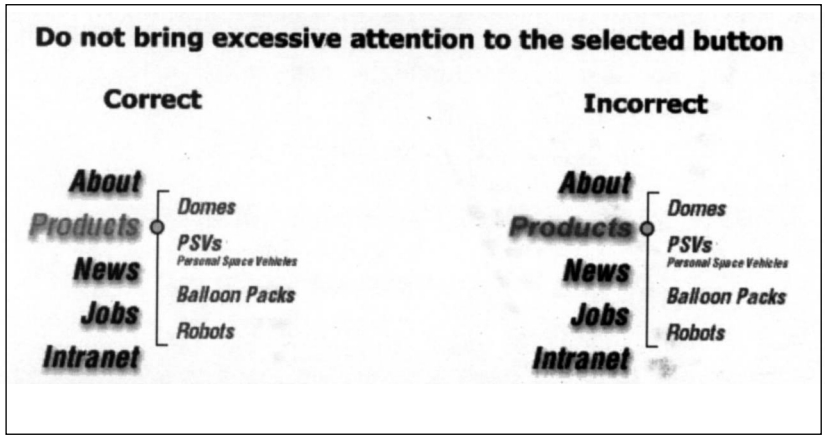


图5-2 被选的按钮应该变得暗淡，而不是更深

一个更高级的网页标签形式加入了关于位置的许多信息。这种类型经常被叫做深度测量，因为它显示了站点中用户所在位置，如下所示。

Home > Products > Robots > **Trainer**

注意，在这种情况下，在标签里前面三个链接是能够被选择的，而第四个是黑体，显示了我们现在的位置。一些人喜欢使用导管符号，而不更大的符号，如下所示。

Home | Products | Robots | **Trainer**

分隔符的选择看上去很精巧。但要注意第一种风格暗示了前进性。事实上，许多人错误地把网页标签的这种方式当做路径信息。而从主页引出了一个路径，这是真的。在这种情况下，对训练者来说，用户怎样结束主页是不必要的。由于混乱，一些人可能把网页标签作为路径表示，但深度测量看起来更精确显示了主页的距离。

标识网页的最新方法不是普通的技术，但是容易实现。浏览器的状态条不显示任何信息，除非用户在链接中滚动。在缺省的时间期间，它不仅能显示正确的网页标签，而且显示了 URL信息。考虑在状态条上显示正常网页和 URL信息，以便当它们集中在网页底部时，使用户能明白。在<Body>标志内使用Script声明，通过onload事件触发是一个较容易的设置，如果浏览器不执行JavaScript，将忽略该语句。

```
<BODY onLoad="window.defaultStatus='Current page: Robot Trainer
(http://www.democompany.com/products/robot/trainer.htm)';return true;">
```

缺省的状态条信息的唯一缺点是可能被用户忽略，或者当鼠标在链接上通过时，用户可能没有注意到状态条上的缺省信息和 URL目标信息之间的变化。，关于利用状态条显示链接信息的内容，将在第6章中介绍。图5-3显示了网页标签的各种形式及状态条上的显示信息。

China-pub.com 下载

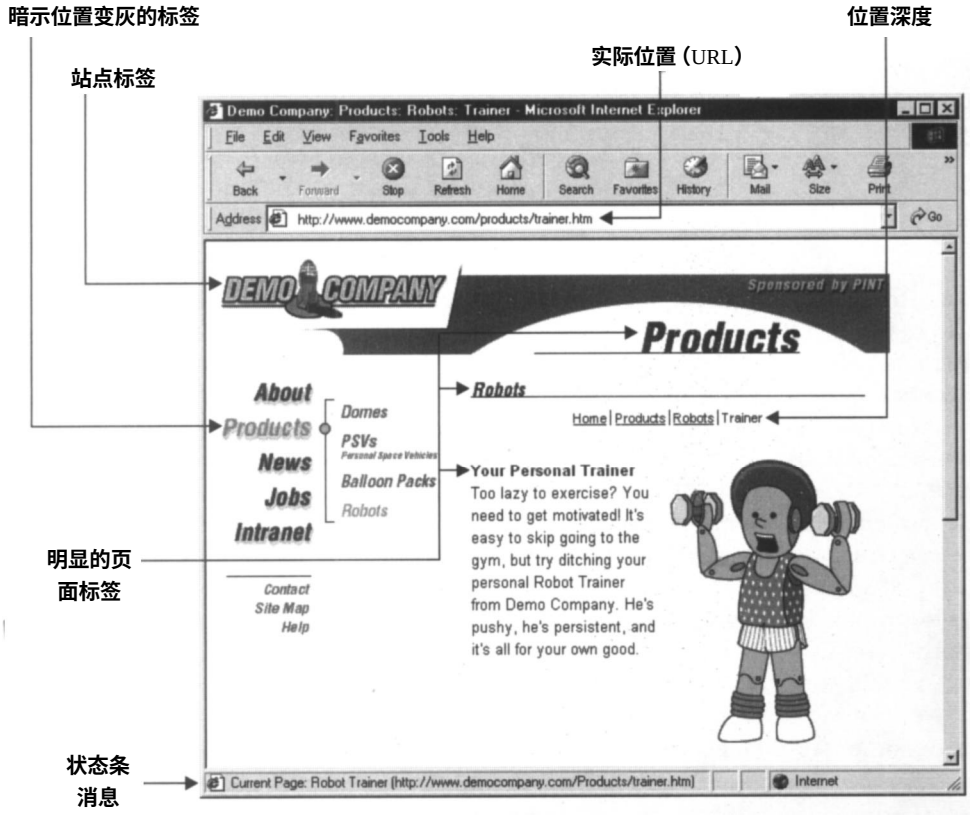


图5-3 网页标签的例子

5.2.3 网页、站点的样式和位置

让用户知道他们在哪的另一方法是看一个网页相对于另一个网页的关系。这种反馈形式是不准确的，但可以给出用户在网站中位置的一个相对概念。

许多站点使用颜色码来指示位置。使用这种方法，站点的每一节对按钮或图形花饰采用不同的颜色。例如，站点采用红色标识产品，绿色标识投资者关系，兰色标识技术支持，等等。在每节的网页中使用这种颜色来标定用户在哪。使用颜色标定位置的唯一限制是每一节的颜色必须不同，以便所有用户能注意到这些。如果人们能察觉到颜色的不同或观察到颜色有限制的环境，可以限制几种基本颜色，如：黑、红、黄、绿、青、兰和洋红。即使加入桔黄色，也会发现你把颜色限制在十到十二种间，因为颜色间要有实际的不同。有许多不同节的站点可发现这种方法的限制。

建议：当使用颜色代码来指示节的位置时，应保证使用的颜色彼此有明显的不同。

提示 一些设计者轻视用颜色代码标识节的方法，因为它经常造成一个花花绿绿的彩虹类的站点，在导航模板上的按钮是彩色的，并代表它们的节。尽管有这些批评，但这种设计类型仍被普遍使用。

China-pub.com
下载

指示位置的另外一种方法是通过主题。考虑在现实世界里，许多邻里利用树和行星为街道命名，他们也使用主街道符号或其他花饰来区别邻里。Web站点也能使用主题，一些站点可以使用某种形式的图例或图片在某节的所有网页上，让人们知道他们在同一节内，例如，在产品节内，他们可以在所有网页内放入销售人员的照片，在投资关系领域内放一张股票经纪人正在用股票收录机的图片，在技术支持领域，放一张技术操作的图片等等，主题概念很奏效，但设计者常常过于热衷它，而开始用“技术支持库”甚至“库”来代替“技术支持”。如果不小心，很容易因过多的使用主题而变成基于比喻的设计。这对某些情形有效，但考虑一下如果用户不能理解它，将会发生什么。一般来讲，隐喻更难趋于成功。

建议：不要过分热衷基于主题的位置暗示，以免掉入设计者定义的隐喻中。

像使用颜色一样，应用主题提示来标识位置的关键是确保提示或页面式样有足够的不同，使得用户能注意到这一点。做到这些可能是困难的，因为可用性建议页面在形式上应保持一致性。变化的程度应该能被察觉，又不致使差异太大而使每页都各不相同。如果要利用标签页的优点，这一点特别重要，进一步的讨论将在下一节展开。

5.2.4 我曾到过哪？

当迷失方向时，你也许担心如何返回原来的地方。同样，也许你感到在兜圈子，疑惑你是否见过这个页面或目前的链接。让用户知道他们曾经到过哪里的一个重要的方面是改变链接的颜色，如果这个链接被访问过。仿效著名的童话故事 Hansel and Gretel 中，小孩们通过扔面包屑的方法找到从森林中返回的路线，这种方法称为面包屑法。典型的做法是，未被访问的链接是蓝色的，而被访问过的链接是酱紫色的。因为用户会依赖于链接的颜色来知晓过去是否做过这一选择，改变链接的颜色是不明智的，这对一个站点访问者带来的害处就像森林中的动物吃光了小孩们的面包屑。关于导航中链接的颜色及效果的深层次讨论见第 6章。

1. 历史

除了带颜色的链接，Web浏览器用一个被称为历史的机制来记录用户所到各处的轨迹，每一个被访问页的参考信息储存在浏览器的历史栏中中，用户应用前进和后退按钮来遍历历史拦中的入口，从这种意义上讲，前进和后退只是当时的前进和后退。一些站点用链接标签如“后退”或用“JavaScript”来提供与历史机制同样的功能，如下所示：

```
<A HREF="javascript:window.history.back()">Back</A>
```

使用这样一个脚本并不是一个好的想法，因为后退链接带用户进入的页面将根据从哪里来而变化。用户不期望站点链接像浏览器的回退按钮一样工作。除去这些粗略的标签，用户期望一个后退链接，在站点内部结构中回退一页，并不像使用浏览器历史的回退的操作，甚至带用户退出站点。除了极少数的情形，当使用复杂的帧站点时，设计者对普通的站点链接不应当使用历史机制。

建议：不要试图在链接中模仿浏览器的历史机制。

为了避免对回退意思的不必要的困惑，鼓励设计者为回退链接设计详细完美的标签，例如，

China-pub.com 下载

“回退到产品”或“回退到机器人男管家”往往比简单的“回退”要好。

规则：避免链接简单的命名为“回退”，通常要精确地表示链接要回退到哪里。

不要低估了用户的历史注意力。显然一个新手最喜爱使用的按钮是回退按钮。当一些新手丢失方向时，他们很可能比一个在动作训练中饥饿的猴子击打食品发放机按钮还要快地触击回退按钮。当然，当用户想用浏览器的 Go 菜单快速遍历历史栏时，对于回退按钮的依赖是不能被忽视的。例如，一些站点通过改变方向带用户到另外一页甚至另一个站点。重定向的原因或许是浏览器的检测，或者只是调整一个错误的 URL拼写以重定向用户，或是处理一个过时的 URL。不管原因如何，所使用的方法经常没有延误，并且对所以重定向创建的页面，用户不使用标准的回退按钮是无法回退的。

由于重定方向而产生的循环页是特别烦人的，用户不得不关闭浏览器，开一个新的窗口，或者想出如何应用GO菜单跳出重定向页的方法。

循环的原因通常与 <META>标签的误用有关。使用以下的代码可以把支持 JavaScript的浏览器中的问题降低到最小。

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<SCRIPT LANGUAGE="JavaScript1.1">
<!--
location.replace("redirectpage.html");
//-->
</SCRIPT>
<NOSCRIPT>
<META HTTP-EQUIV="Refresh" CONTENT="0; URL=redirectpage.html">
</NOSCRIPT>
</HEAD>
<BODY>
This page has moved <A href="redirectpage.html">to URL here</A>.
</BODY>
</HTML>
```

规则：避免创建使用浏览器回退按钮不能回退的页。

2. 用cookies表示过去的访问

表示用户以前曾经到过的站点的最先进的方法大概是用 cookie。

定义： cookies是站点分发给用户,并存储在用户系统中的少量的文本信息。

从某种意义上讲，一个 cookie就像一张洗衣票，当访问一个站点时，假定允许，你就分到一个cookie并存放在你的系统中。任何一个你设置的优先权都储存在 cookie中，每当你访问同一站点时，都会被重读。 cookie最基本的用处是为了先进的 Web设施如站点的个性化储存状态信息。便于用户跟踪的可编程 cookie的全面讨论超越了本书的范围。现在，就假定一个 cookie能够跟踪一个用户并且在导航中提供帮助。

China-pub.com
下载

使用cookie，一个站点能够有实力跟踪用户并且通过在屏幕上写出“欢迎回来”的信息让他们知道又回到同一站点或某一特殊页，甚至把上次用户访问特殊页的信息显示在该页上。

当然，用户也许会拒绝cookie，或不信任任何建立在cookie上的数据，因为最后的站点会跟踪用户并发布欢迎回来的信息而用户实际上以前根本没有去过哪里。如果想通过使他们认为他们已经在过去和站点打过交道而试图在用户中建立一种安全的虚假感觉，在某些情形下，用户会思索他们的记忆而认为这是一个真的诡计。

除了颜色，真的很少有信息让用户知道他们以前曾经到过这些地方。一个用户自己的记忆—长期的和短期的—是人们知晓他们是否到过某地的主要途径，考虑到人类记忆不准确的特性，用户可能不能够精确记住像文档的URL或内容这些细节，但他们能够记住一页或一个站点的一般特征。例如，用户也许能够记住站点或页面的颜色甚至布局，特别是它和他们曾经访问过的站点有显著的不同时。像这样能被很容易地记起的页称为标志页。

3. 标志

在脱机领域里，我们应用许多技术来寻找路径，最基本的可能是标志，一个标志是一块地形的显著的识别特征，基本上是独特的，容易被注意到并记住的事物，人们一直用标志物作为导航的参考点。例如，当给某人指路时你会说“我的房子刚过旗杆”或“当你到达肉饼屋后右拐”。由于我们用标志建筑相对于其他对象确定我们的位置，一个标志物应易于识别和记忆。

在Web上，用户趋向于识别两种主要的标志，一个是他们进入Web的页面：这被称为用户的主页或起始页。通常情况下，这一页被设置在用户的个人主页上，或公司的主页，或一个入口页如雅虎！这种标志不常改变，但是，Web上的另外一种标志形式是临时的，当一个用户进入站点开始浏览，站点的主页通常作为一个临时的标志。标志的关键是该页看起来与用户访问的其他页有很大的不同,而确认为一个标志。如果看起来很相似，用户就不能回忆起目前所在页是否是标志页。

但是，不能把用差异提高记忆力的方法用过头了。考虑一个实际的站点，用随意的描述作为主页，每次该页被装载时，都有显著的不同。这种想法使站点看起来是动态的。然而，访问过程中，迷失的用户往往回到主页重新开始，但在这种情形下，主页不再是他们感觉的那样。这种随意性引起了混乱。站点网页中令人舒服的标志的改变，使得用户对于自己在哪里感到迷惑，更糟糕的是，浏览器上的URL条被隐藏了，许多用户真的不知道在哪里了，只好放弃。标志必须是不同的，但如果它们被用作迷失者的参照点，它们必须是稳定的。

规则：用户把他们的起始页记作永久的标志，把站点的主页当作半永久性的标志。正因为如此，这些页面在生存期应该保持稳定，但要和其他所访问的页面有显著的不同。

5.3 我能去哪

大多数情形下，用户最重要的问题是，“我能去哪？”页面上各种各样的链接和标签预示着我们能去的地方。用户通常从出现的选择中选择目的地，除非他们对潜在的目标有些先前的知识，或以前曾去过该站点，或迂回地知道显示些什么，给用户展示各种各样的选择是很重要的。首先考虑的事情是保证选择项的可用性，这是显而易见的。一些站点的目标是把选择隐藏在下

China-pub.com 下载

拉菜单后，或者脱离屏幕的弹出式菜单。考虑到用户可能不会注意到这些选择，以至于它们很可能不被选择。

5.4 导航位置

导航元素的位置不仅仅是一个品味的问题；它还带来了许多使用上的问题。在屏幕上的一个Web页实际上仅有五个基本区域来放置导航元素：顶部、底部、左侧、右侧和中央。各个位置都有其优缺点。

5.4.1 顶部导航

许多站点把导航元素向屏幕的顶部放置。考虑到所有的导航元素很可能迅速地显示出来，这样做很有意义。同样，对于图形用户界面，传统上屏幕的上部是程序主要菜单的放置地，对Web站点为什么不同呢？考虑到对一个典型的网页最一般的扫描方向是从左到右，从上到下，这就是导航元素的一个比较好的位置。

1. 在屏幕的上部固定导航元素

一般来说，这一过程是用帧来实现的，所导致的一些使用上的问题将在本章的“帧问题”小节中讨论。也可能用DHTML来创建一个附着于屏幕顶部的浮动导航调色板，但仅在最新的浏览器上才支持这一点。

2. 使用“返回顶部”链接

一些站点使用“返回顶部”链接，使用户返回到有导航调色板的页面的顶部。这个底部的操作在用户漫游出网页以前需要额外的一击。“返回顶部”链接的上面部分也许是用户想简单地退回到顶部。这种形式的链接可以省去大量的滚动操作。相反，到底部的链接是不常用的，因为用户实际上不知道底部有什么，所以为什么要去哪里呢？就页面上部的链接，用户曾到过顶部，知道导航在那里，所以他们有理由选择向上的链接。也许有的用户会奇怪一个下部的“到达顶部”链接是回翻页面。建议使用一个好的标签如“回到页的顶部”或简单的“页的顶部”而不用“顶部”，一些设计者宁愿使用向上的箭头，如图5-4所示。

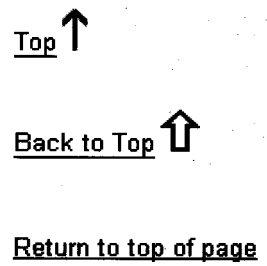


图5-4 使用向上箭头

在这种情形下，要确保至少ALT或TITLE文本显示链接的目标。在第6章将深入讨论创建可用的链接。

3. 在页面底部提供文本链接

模仿顶部链接的文本链接经常被添加到页面，一般在一条水平尺的下面。通常，文本链接用括号（[ ]）或管道标号（I）来明显区分导航链接，如下所示。

About | Products | News | Jobs | Intranet | Contact | Site Map | Help

这种在页面底部模仿顶部链接的导航方式通常被称为“标题和页尾”设计，这种方式很普

China-pub.com
下载

遍，以至于用户在厌倦了等待页面的图形按钮装载好之前，就快速地滚动到页面的底部去使用回退文本链接。

顶部导航的另一个可能引起麻烦的地方是标签和站点的品牌，如联合徽标，会丢失在导航元素中，例如，看一看如图5-5所示的导航条。

图5-5 导航条

注意到图标、表示位置的页面标签和按钮是如何竞相引起用户的注意的，应用顶部导航时，要确保不要让按钮拉长了上部的其他信息。考虑到在典型的软件应用程序中菜单条的应用，一般它们较窄并且占用了屏幕垂直方向上不到10%的空间。然而，在许多站点上，导航条做得很大，用户不滚动屏幕是不可能看到很多内容的，这样潜在地将更多的注意力集中在导航上，而非内容上。

技巧 当使用顶部导航时，注意不要用导航元素覆盖了标签和站点识别信息。

5.4.2 底部导航

在通常的情形下，底部导航看起来没有多大的意义，它可能增强滚动。这是因为导航元素可能不会在第一屏幕区域内完全展示出来，除非页面内容有限，或是用户有一个很高的屏幕。当然，应用帧有可能把导航固定在当前屏幕的底部。这些将在后面的小节中讨论。当然，即使没有帧，把导航放置在屏幕的底部也可以为页面标签和公司品牌腾出位置。

将导航置于页面底部的潜在问题是，导航不是用户的主要扫描方式。但是，如果用户的确扫描了整个页面，当他们就要移向下一个页面时，他们最终会到达导航所在的页面的底部。由于这种使用方式，许多站点在屏幕的底部提供了文本链接（如前一节所述）。但是，不建议放置基本的导航形式在屏幕的底部，如图形按钮，这不符合传统的软件位置控制。

建议：避免在屏幕的底部放置主要的导航，为辅助或增强导航保留这一区域。

5.4.3 左导航

页面的左边部分是导航元素的一般位置。因为英语和其他西方语言国家的读者从左向右扫描信息，这使得导航沿用用户阅读的途径。同样，屏幕的左侧在许多程序里是导航的通用位置，这在Web设计中有些约定成俗。即便在印刷设计中，左导航也是很普遍的，如，想想大多数表中的内容是如何读出来的。

左侧导航的主要问题是它妨碍了内容，并减少了预留给内容的屏幕空间的数量。考虑页面左侧的一个导航条创建了一种“导航篱笆”，致使用户必须跳过它去阅读内容。这即可看作一件分心的事，也可认为是页面的一个积极的限制区域。考虑到左导航为页面制造的空边，没有导航条就空着。另一个问题是导航条占用了宝贵的屏幕的真实空间，限制了一个典型页显示内容的空间。如，用户有一个640×480像素的屏幕，考虑到浏览器的设置和用户可能不会立即全屏

China-pub.com 下载

放大他们的浏览器，大约有 570～580个像素点可用于内容，即便有大一些如 600个像素点可用于内容，如图5-6所示，如果添加按钮，就只有很少的空间留给内容使用。

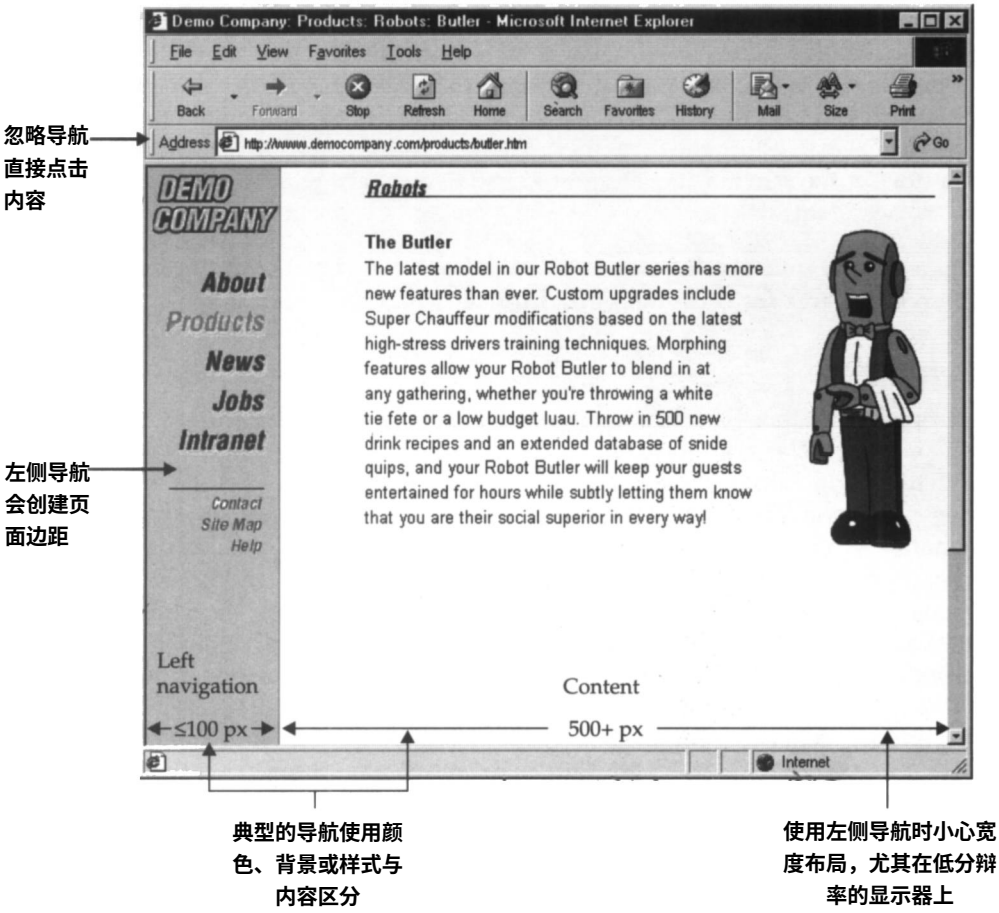


图5-6 左导航引起的屏幕样式问题

技巧 当使用左导航时，确保为窄屏幕重新格式化内容。

遗憾的是，重新格式化所有内容来处理左导航的屏幕限制问题也许是不可能的。解决这个问题仅有三种办法，下面来讨论。

1, 需要时让用户滚动屏幕

桌面系统中低分辨率系统的数目在急剧减少，随着 Web设备的增加和自动访问，在未来也许会引起戏剧性的变化，用户从左向右滚动并不觉得舒服，网站设计者不应在格式代内容时强迫这样做。

2. 为多的内容开新窗口

当用户需要浏览较宽的内容，许多站点倾向于开新窗口而不用左导航，如一个不能在低分辨率下，重新格式化需要全屏浏览的图表。当重新排列内容以便打印时，移动导航使内容占满

China-pub.com
下载

全屏也是站点的常用方法。而开一个新窗口的确有缺点，在低分辨率的环境下工作是必须的。有两种方法为较宽的内容开新窗口。一个依赖于使用 HTML的TARGET属性，另一个是使用JavaScript。用HTML开一个新窗口，如下所示简单地设置链接上的 TARGET属性到_Blank:

```
<A HREF="widepage.htm" TARGET="_blank">Product Table (new window)</A>
```

当然，通过这样一个明显的声明，让用户知道你打算开一个新窗口或许是一个不错的主意。以下的HTML方法就是当它工作时，确实用所有的浏览器控制创建了一个窗口。例如，这里的HTML展示了如何用JavaScript创建一个用于开一个特殊的最大化窗口链接的例子：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Window Opener</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
function createWindow(filename,width, height){

NewWindow = window.open(filename, "","toolbar=0, location=0,
directories=0,status=0,menubar=0,scrollbars=1,
resizable=1,copyhistory=0,width="+width+",height="+height);
}
//-->
</SCRIPT>
</HEAD>
<BODY>
<TABLE WIDTH="100%">
<TR>
<TD WIDTH="100" BGCOLOR="yellow">
  Navigation buttons in this column
</TD>
<TD>
<H2>Text links are going to open a new window</H2>
<HR>
<A HREF="http://www.yahoo.com"
onClick="createWindow('http://www.yahoo.com', 640,460); return
false">Yahoo!</A>
</TD>
</TR>
</TABLE>
</BODY>
</HTML>
```

技巧 Web链接http://www.weddesignref.com/chapter5/windowcreator.htm.

在这个例子中，我们把窗口的要求尺寸传递给 CreateWindow函数，赋予它等同于最大分辨率640×480下的设置。考虑到窗口任务条所占的空间仅用了 460个像素，也可能用 JavaScript去

China-pub.com 下载

检测屏幕的分辨率，或打开到屏幕的尺寸。甚至可能打开全屏占据整个桌面，但不建议这样做。

使用没有浏览器设置和导航的新窗口的主要缺陷是也许用户对新窗口感到困惑，不知道如何使用和关闭窗口，由于这些潜在的困扰，应该使用一个明确的关闭按钮甚至帧设置表明正在使用一个外部窗口。一些站点甚至应用这一方法同外部站点展开链接，图 5-7显示了其工作的一些例子。

图5-7 窗口样式应该显示辅助的窗口

3. 隐藏左导航

一些站点开始用动态HTML隐藏左导航，仅在需要的时候才显示。这种滑动或脱出的导航形式并不节省屏幕的实际资源，但把有关向用户隐藏导航的使用性问题和空间问题对换了。通过隐藏导航，我们限制了用户理解选择范围的能力，他们没有明显地脱出导航。另一方面，一个滑出式导航确实把注意力放在内容上。以下的 JavaScript和HTML演示了如何应用这种形式的导航：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Hidden Left Menu Demo</TITLE>
<STYLE>
<!--

#iemenu,
#netscapemenu    {font-weight: bold;
                  font-size: 12px;
                  font-family: Verdana, Sans-serif;
                  line-height:20px;
                  position:absolute;
```

China-pub.com
下载


```
                left:-75px;
                width:85px;
                top:10px;
                border:1.5px solid black;
                background-color: #ffff99;
                layer-background-color:#ffff99;}
.menulinks      {color: black; text-decoration: none;}

-->
</STYLE>
</HEAD>
<BODY>
<SCRIPT>
<!--
function generateLinks() {

var menuitems=new Array()
var menulinks=new Array()
// Add menu titles here
menuitems[0]="Home"
menuitems[1]="About"
menuitems[2]="Products"
menuitems[3]="News"
menuitems[4]="Jobs"

//Add corresponding menu links here
menulinks[0]="http://www.democompany.com/";
menulinks[1]="http://www.democompany.com/about/index.htm";
menulinks[2]="http://www.democompany.com/products/index.htm";
menulinks[3]="http://www.democompany.com/news/index.cfm";
menulinks[4]="http://www.democompany.com/jobs/index.htm";

for (i=0;i<=menuitems.length-1;i++)
document.write('  <A HREF='+menulinks[i]+'
CLASS="menulinks">'+menuitems[i]+'</A><BR>');

}

/ * output the menu */
if (document.all)
{
  document.write('<DIV ID="iemenu" STYLE="left:-75"
onMouseover="show()" onMouseout="hide()">')
  generateLinks();
  document.write('</DIV>');
}
else if (document.layers) {
  document.write('<LAYER ID="netscapemenu" onMouseover="show()"
```

China-pub.com 下载

```
onMouseout="hide()">');
  generateLinks();
  document.write('</LAYER>');
 }
else
 generateLinks();

//-->
</SCRIPT>

<SCRIPT LANGUAGE="JavaScript1.2">
<!--
/* Deal with Netscape resize bug */
function regenerate(){ window.location.reload() }

function NetscapeRegenerate() {

  if (document.layers)
   setTimeout("window.onresize=regenerate",400)
}
window.onload=NetscapeRegenerate;
/* end Netscape resize bug code */

if (document.all) {
  menu=document.all.iemenu.style;
  rightboundary=0;
  leftboundary=-75;
}
else {
  menu=document.layers.netscapemenu;
  rightboundary=70;
  leftboundary=10;
}

function show() {
if (window.hidemenu)
   clearInterval(hidemenu);
   showmenu=setInterval("doshow()",50);
}

function doshow() {
if (document.all&&menu.pixelLeft<rightboundary)
   menu.pixelLeft+=5;
else if(document.layers&&menu.left<rightboundary)
   menu.left+=5;
else if (window.doshow)
   clearInterval(doshow)
}
```

China-pub.com
下载


```
function hide() {
 clearInterval(showmenu)
 hidemenu=setInterval("dohide()",50)
}

function dohide() {
  if (document.all&&menu.pixelLeft>leftboundary)
    menu.pixelLeft-=5;
  else if(document.layers&&menu.left>leftboundary)
    menu.left-=5;
  else if (window.dohide)
     clearInterval(dohide);
}
// -->
</SCRIPT>

<H2 ALIGN="CENTER">Hide Left Navigation Demo</H2>
<HR>
<P>Put your site content here</P>

</BODY>
</HTML>
```

技巧 Weblink:http://www.Webdesignref.com/chapter5/dynamicleftnav.htm。

遗憾的是，以上提供的代码不得不处理一些与 Netscape相关的问题，如<LAYER>标签的使用，在未来，有关这方面的工作不再需要了。可改变代码来适应自己的站点设计 ,只是改变对象的位置并加注自己的标签和链接在 generateLink()函数中定义的队列。例子演示了隐藏菜单和一个外露式菜单，见图5-8。

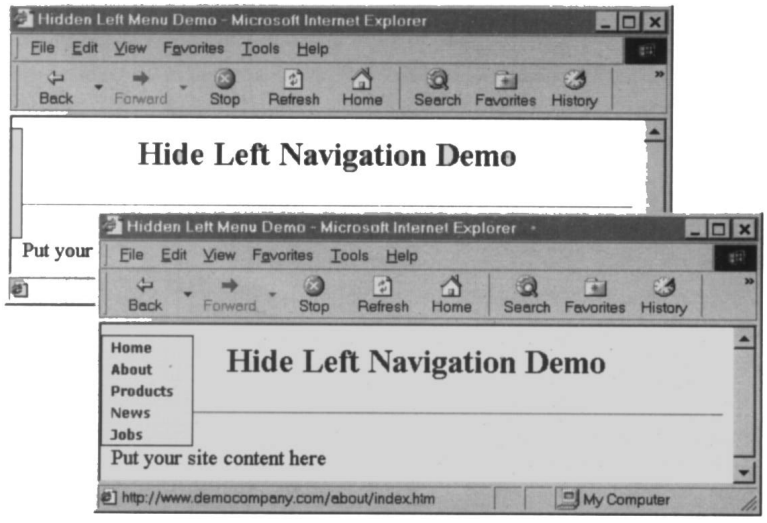


图5-8 弹出式左导航

China-pub.com 下载

5.4.4 右导航

最近，把导航元素放在右侧开始流行起来。一些人辩解道，在右侧放置导航不会影响内容，用户能立即投入到阅读中。右侧导航支持者认为导航按钮应靠近滚动条，为用户限制鼠标的移动。

尽管有以上的优点，右导航有许多潜在的缺点。首先，考虑一个简单的问题。哪里是确切的右侧？依赖于用户的监视器和浏览器的尺寸，从左侧到屏幕右侧的距离也许有很大的变化。在一个很大的监视器上，导航可能远离屏幕的左边缘，从而鼠标在导航元素和用户最喜爱的回退按钮之间的移动达到最大。同样，用这样灵活机动的右侧，屏幕的设计也应是灵活可变的。正因如此，一些设计者倾向于为右导航建立一个人为的右页边。一般来讲，他们趋于在 600到700像素之间制作右页边，而整个页面在 800像素宽的地方结束，在大多数的显示器上，这大约相当于标准信纸的尺寸。这是可以接受的，但在所有的监视器上，右导航的第二个好处不一定能真正享受到：靠近滚动条。当然，它比左导航要靠近些，但在大屏幕上距离仍然比较远。图 5-9展示了右导航的好处和一些问题。

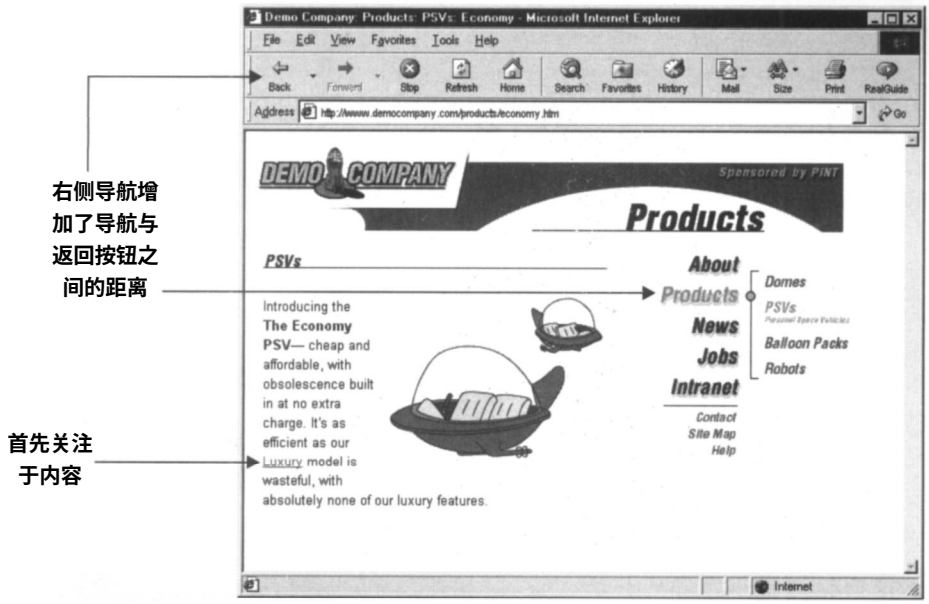


图5-9 右导航存在很多问题

可能不考虑使用右导航的最大因素只是因为习惯的问题。图形用户界面不倾向于右导航，大多数的站点也不赞成右导航。尽管在某些人看来左导航也不是最佳的，右导航当然也是非标准的。在实际上改变站点门户以前，要把这些都想好。是否这意味着从不使用右导航？不是，但确实不要把主要的导航元素放在那里，除非规则改变。一些站点开始用右侧来放置广告、交叉链接以及辅助的导航形式。

建议：避免把主要导航放在屏幕的最右侧。

China-pub.com
下载

5.4.5 中心导航

在窗口中放置导航元素的最后一个选择地是中心。一般来讲，只有对主页，才可把加重的导航元素如图形按钮或图像画面放在页面的中心。导航的中心设计在于加重强调，而并非为内容留出许多空间，因为导航位于用户的注意力集中区。然而，对主页而言，这也许不算什么问题。考虑到主页的主要作用是帮助用户决定到哪里。把导航的开始部分放在中间是很有道理的。这样设计也使得主页看起来有别于其他页，使得创建标志页容易些。

建议：主页和其他标志页应该考虑采用面向中心的导航方式，以区分同一站点的其他页面。

子页不要采用面向中心区域的导航设计，而用简单的文本链接。内容应该出现在屏幕的中心，从而使中心区域的导航元素能和内容交叉链接起来。给导航元素的最后一种选择就是脱离当前窗口放置。对帧和子窗口的讨论，经常称为远程，将在后面的“子窗口”小节中讲述。

5.5 导航的一致性

不管为导航选择了哪种位置，即便是几乎所有的位置都使用了，对不同类型的导航来说，所有的一切必须保持一致性。如果主导航在上部，辅助导航在左部，就仍保持原样。在标志页和其他页间，导航的变换是可能的，但一般来讲以下的 Web设计原则应给予考虑。

规则：导航的布置应和页面的布局保持一致。

导航稳定的重要性怎样强调也不过分。许多研究表明，一致性是实用性的关键，那种绕着屏幕跳来跳去的导航会迷惑或误导用户。考虑到即使导航的布置基本上相同，小的跳跃还是可能会发生。发现这一点的最简单的方法是对站点做一个“快速浏览”。执行一个快速浏览，在屏幕间快速点击，同时注意导航的移动，或许为了到达导航条的下一个选择你不得不大量地移动鼠标。

从屏幕到屏幕间导航区域内元素的位置和数目也应保持一致。当用户在许多站点的导航区域移动时，加上或移走按钮。想像一下，用你最喜爱的字处理软件工作时，突然加上或去掉主要菜单将会是什么样子。不要试图移去一个导航选择，因为用户选择了它；否则，可以使它不能被选择或变灰。考虑图 5-10所示的菜单选择。注意导航上被选择项的移去是如何改变区域尺寸的，破坏了稳定性，就使菜单看起来不再是同一个菜单。

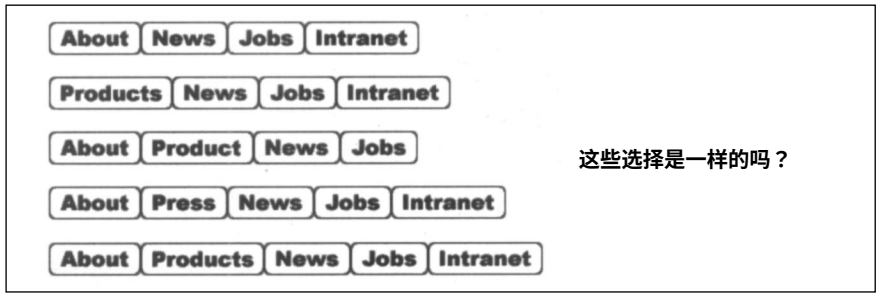


图5-10 带有各种选择的菜单令人困惑

China-pub.com 下载

仔细看看，就会发现某个菜单中仅有一个按钮有微小的变化。用户会发现按这种方式移动按钮和增加选择太令人困惑。

规则：导航应该是一致的，元素在位置、次序和内容上应当是稳定的。

以上的规则并不排斥增加导航元素的能力。然而，如果要这样做，必须让用户知道我们打算那样做，并明确知道增加了什么。例如，考虑一个树型的导航控制。在树型的导航控制中，用户能够用展开和收缩导航去显示和隐藏控制的选择项。然而，这种控制预示着导航将被改变，并试图区分新增加的导航项。例如，考虑在图5-11中的一个展开和没展开的树型控制，一旦展开，就通过缺口和不同的形式显示导航选择。

树型控制也并非尽善尽美。通常是树型控制扩展得太靠下太靠右。实际上，当扩展到第三、四级时，大多数站点的树型控制就显得有些笨拙了。

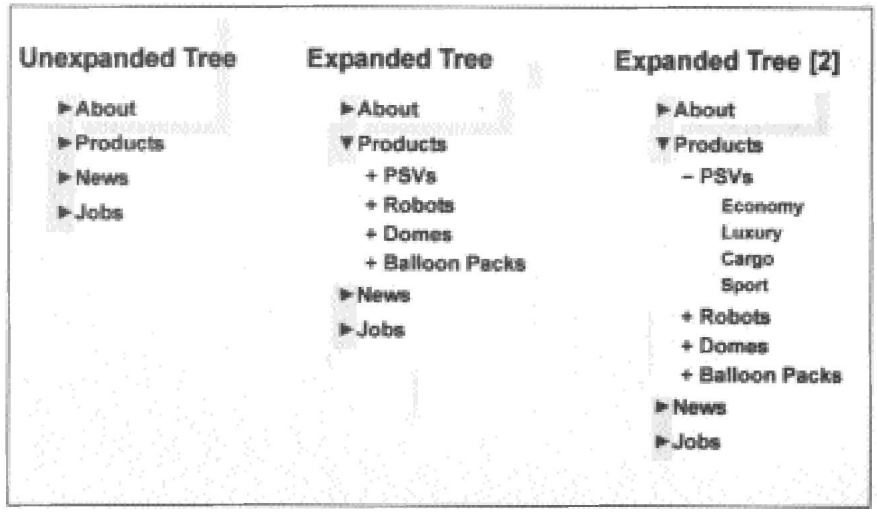


图5-11 树型导航允许导航更富有灵活性

用屏幕位置表示导航级别

除了用扩展/折叠形式的导航策略外，设计者通常倾向于在屏幕的其他部分显示新的导航选择。例如，在屏幕的上部放置主要的站点或导航片段，把补充的文本链接横放在屏幕的底部，辅助导航放在左部，三级导航以树型结构加放在左侧。对于有限制的情形，如果范围有限并且不和内容相冲突，就放在屏幕的中央区域。最有意思的是，当应用屏幕的不同位置等效于一个不同形式的导航方法时，站点导航就很难超过三、四级了。

建议：用当前屏幕的位置来区别导航的选择项时,要明白这四个位置是最顽固的障碍。

通常这种常用的导航形式包括回到主导航的文本链接。因为这是导航的一般位置，许多设计者宁愿使用这种称为 TLB或“顶部—左侧—底部”的导航方式。在这种方式下，主导航被横放在页面的顶部，辅助导航沿左侧放置。一般来讲，辅助导航不如主导航突出。一个简单的

China-pub.com
下载

TLB框图如图5-12 所示。

TLB方法赋予屏幕位置一定的导航意义，考虑到用户扫描的途径，这显得很有意义。如果用户首先浏览页面的上部，他们将横着读到主导航的选择项。一旦他们转到左侧，开始通读内容和选项，他们将到达页面的底部并在底部收到主导航的文本形式的问候。 TBL设计对许多设计者看起来似乎缺乏创意，但该方法确实方便有效。

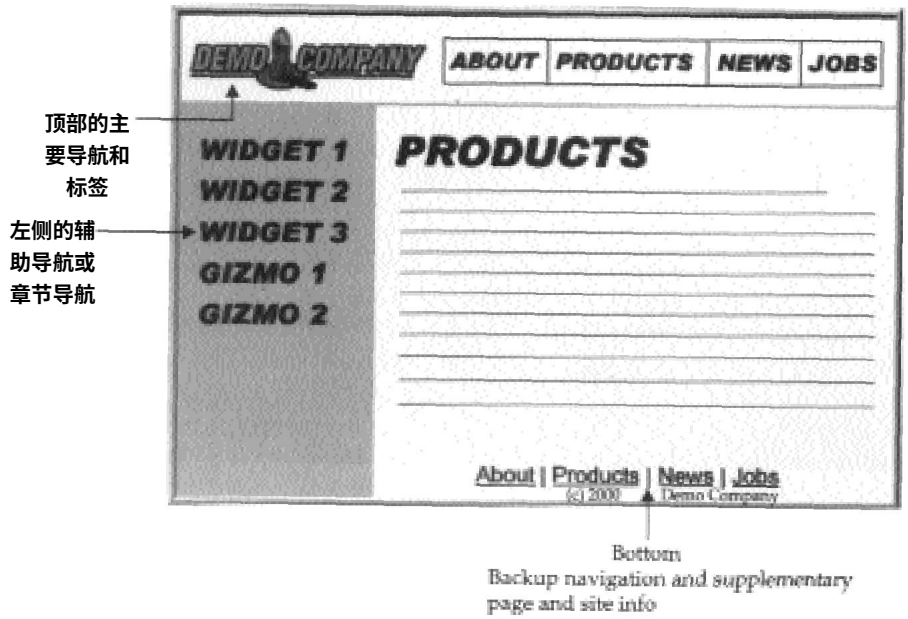


图5-12 TLB导航模板

5.6 导航和滚动

一个值得注意的问题是，是否需要滚动页面，特别是当包含导航元素的时候。非滚动倡导者认为按一个固定的页面尺寸来保持页面，可以使设计保持一致性并提高用户的可预见性。这是真的考虑到改变页面尺寸，用户真的不知道在按下一个典型的按钮后，他们将接收到的信息容量。当然，固定页长度的概念也有许多缺点。页面尺寸的讨论在第 9章中展开。现在，考虑那些用于导航到其他页面的页面，或许不应使用滚动。应该力图减少用户选择下一页面所花费的努力，并且屏幕向下滚动来寻找选择既增加了移动，又迫使用户回忆那些看不见的选择。这种想法引出了以下的Web设计建议。

建议：一有可能就应当把导航引导页竖放在屏幕上，就像所有其他页上主导航的放置一样。

这个原则特别地建议，那些仅用来引导别的页的页面应该集中在不需滚动的屏幕区域，或即便需要，如他们所说的“ 在折叠之上 ”。猜测一个允许的确切高度是不可能的，因为浏览器周围有不同数量的设置，用户按自己的辨别来确定屏幕的高度。但是，应用程序就能确定装载页

China-pub.com 下载

面屏幕的高度，这将在第9章中讨论，用户可以按自己的意愿来改变。在所有的情形下，特别当处理低分辨率如640×480时，设计者在估计上应当保守一些，许多站点把整个导航放在前300个像素点上，不去考虑屏幕的分辨率。第一屏作为真正的屏幕资源，这个区域外的导航需要用户滚动去激活它们。

5.7 导航和鼠标移动

除了尽量限制导航元素的滚动以外，设计者还应限制鼠标在导航选择项之间的移动。要经常考虑导航元素和回退按钮 —— 那些最经常使用的浏览按钮 —— 之间的距离。尽管高级用户会用右击或类似的导航快捷方式避免移动指针到屏幕的左上角，许多用户并不这样做。所以，当可能的时候应把导航项和回退按钮间的距离最小化，像在第3章中关于移动和可用性的讨论一样。

建议：把主Web导航按钮和回退按钮间的距离最小化。

当考虑到用户点击了导航选择又去做别选项时，关于鼠标距离的想法就会产生。如果用户仍想停留在该站点，这个选择或是回退按钮或是屏幕上的其他按钮；如果用户想离开该站点，他们会调用一个窗口键入新的URL或移动到地址栏，做同样的事情。如果用户选择了屏幕上的另一个按钮，限制用户与下一个选择间的移动距离是明智的。如果限制了距离，下一个按钮显得与刚按过的按钮很近，导航将会很省力。回忆第3章的规则。

建议：对连续的选择应使鼠标移动的距离最小。

限制鼠标的距离不仅仅是提高导航能力的一般意义上的方法。如第3章的讨论，再次考虑Fitts定律，它表明到达用户能点击按钮的地方的速度与按钮尺寸及按钮和鼠标的当前位置的距离成反比。基本上讲，如果按钮很小并且较远，用户就不能很快到达。如果按钮很大并靠近前面的选择，Fitts定律表明用户能够很快地使用界面。如果考虑到点击看上去彼此靠近的大的红色按钮是件多么容易的操作，你就会明白Fitts定律陈述了多么明显的道理。那么，为什么网页上有这么多跳跃的小按钮呢？为了消除Fitts定律效应，使选择项靠在一起，并把距离较远的按钮做得大一些。注意到许多界面，比如安装器和导航模式的界面，已经通过减小屏幕尺寸来限制鼠标的移动，并且下一个要点击的按钮与刚点击过的按钮相邻。

5.8 帧

一个能用以提高稳定性并有可能减少滚动的导航工具（也许甚至减少鼠标移动）是帧。遗憾的是，帧在Web中得到了某些坏名声，大致由于早期的实现问题和诸如可用性专家Jakob Nielsen的一些口头批评。事实上帧一般是被误用了，但在打算不使用它之前，应考虑到它们实际上确实有一些可以应用的特性。

帧的最大问题就是它们的用途被错误理解。很多用户偶尔使用帧作为页面布局工具。事实上帧是导航工具。帧的思想就是把屏幕分解为多个区域、面板或窗口。把浏览器窗口分解成许多相互独立的区域，就使得用户一次可看到多个文档。考虑如图5-13所示的简单的仅两个帧的

China-pub.com
下载

设计，实际上有三个文档在使用 —— 一个用于建立帧的文档，一个用于左帧的文档，一个用于右帧的文档。

Frameset Document:

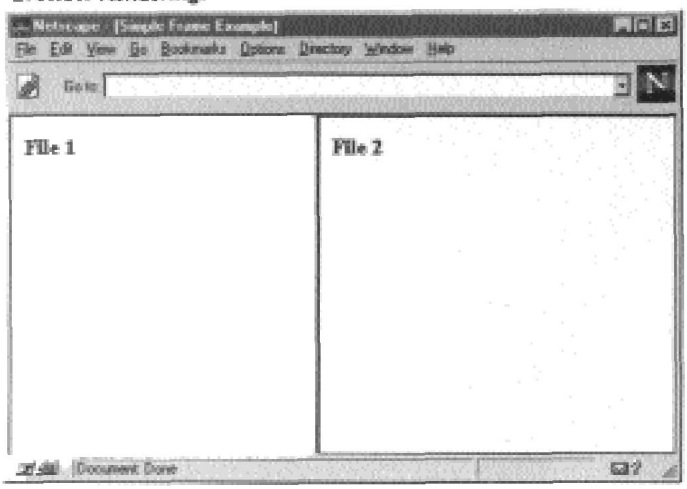
```
<HTML>
<HEAD>
<TITLE>Simple Frame Example</TITLE>
</HEAD>

<FRAMESET COLS="250,*">

<FRAME SRC="fileone.htm" NAME="1">
<FRAME SRC="filetwo.htm" NAME="2">

</FRAMESET>
</HTML>
```

Browser Rendering:



File 1 Code:

```
<HTML>
<HEAD>
<TITLE>File One</TITLE>
</HEAD>

<BODY>
<H2>File 1</H1>
</BODY>
</HTML>
```

File 2 Code:

```
<HTML>
<HEAD>
<TITLE>File Two</TITLE>
</HEAD>

<BODY>
<H2>File 2</H1>
</BODY>
</HTML>
```

图5-13 帧的总览

应用帧可能有更大的优点。首先，有可能使用帧始终把导航固定在屏幕上。不管放置在帧的导航按钮在上部、左部、底部，甚至屏幕的右部，按钮都能固定因而不能滚动。

技巧 Web链接：例子的展示可在在线网址上找到：

`http://www.Webdesignref.com/chapter5/framefixednav.htm`。

帧的另一个优点是它们产生看上去速度很快的效果。考虑上例展示的固定导航的情形。如

China-pub.com 下载

果点击屏幕上导航条的不同链接，注意到仅有屏幕右侧部分更新，而左侧依然不变。因为屏幕更新变少，站点对用户来说显得速度很快。如果帧更大一些，这种幻觉更明显。

技巧 Web链接：例子的展示可在在线网址上找到：

`http://www.Webdesignref.com/chapter5/framespeed.htm`。

帧也提供了允许多个文档同时在一个窗口内显示的优点。例如，如果想比较不同的项目，建立一个帧的运行环境以使用户能够点击按钮，调出含有不同产品的页来，比较就变为可能，如图5-14所示。这个例子说明用帧能够导致非常复杂的导航。

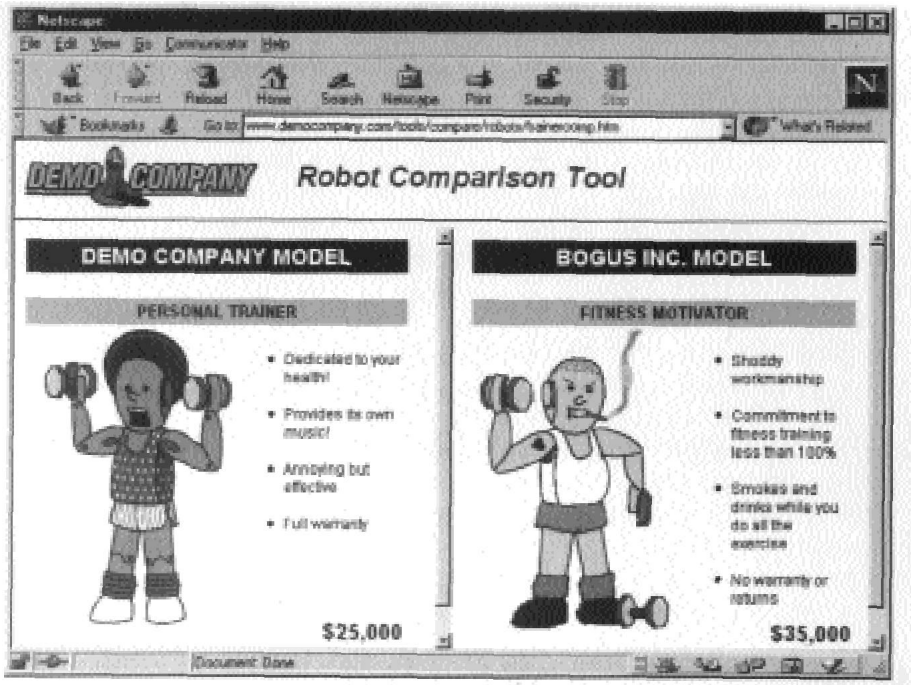


图5-14 帧用于比较

5.8.1 帧问题

确实，在使用帧以前必须认识到帧有许多严重的缺点。首先，要理解帧打乱了Web基本页的隐喻。在大多数站点上，一个单独的URL等同于一个单独的文件。点击链接或者输入一个新的URL，得到一个全新的满屏数据。一个帧站点不服从“一个URL对应一个文档”的Web模式，或至少看起来不是这样。实际上，在大多数帧环境下，URL仍似乎不变。重新回顾在http://www.Webdesignref.com/chapter5/framesfixednav.htm的例子，并注意到不管选择什么样的链接，URL根本没变。当考虑到用户可能会使用URL决定他们的位置时，这是一个很严重的问题。

因为URL在帧环境下根本不变化，用户也许很难给内部页面做书签。也许通过设计来解决，

China-pub.com
下载

正如第4章讨论固定网点结构时所提及的一样。但是，如果用户感到他们能够为一个特殊的新闻版本做标签，但发现加上标签的是顶层的页，他们就变得困惑起来。即便高级用户能够在新窗口内打开一个帧页，也给深处的 URL加了标签，当他们发现所做标签并不包括页面上的所有元素如导航时，也变得困惑起来。

和帧相关的另一个问题是它们打印起来很困难。打印一个帧页时，用户必须知道在打印前点击帧区域。许多应用帧的站点没有把帧的不同区域明显地区分出来，所以用户也许不知道打印时应该点击哪个区域。

最后，许多设计者发现帧设计中搜索引擎不能很好地工作，并且不能索引站点内容或跟随链接。单根据这一条限制，许多设计者就放弃了帧设计，这是很遗憾的。事实上，真正的帧问题是实现起来很困难 —— 特别是说清它们所有的缺点。当用帧时，很容易加强站点。

尽管帧对许多用户、设计者和使用者可能比较困难，但仍被使用得越来越多；一些普通形式的帧已经出现。更进一步的是，已经可能避开前面提及的许多帧的局限，比如做书签和固定URL等。遗憾的是，问题的解决是以牺牲帧的优点如屏幕的刷新或对代码依赖的增加等为代价的。对Web浏览器的改进也要解决一些帧问题。因特网已经对打印和帧标签提供了许多好的支持。但是，撇开帧技术的所有潜在的先进之处不谈，主要的问题仍然是执行起来缓慢。

5.8.2 使用帧

本小节将讨论帧的使用和避免问题的技巧。然而，对帧语法的全面讨论，读者可以参阅HTML的配套书：参考大全。要考虑的第一个问题是你是否需要使用帧。记住，帧是导航工具，只有当试图创建区域如装载屏幕的其他部分的控制条时，你才使用它，而不是当你试图创建一个复杂的布局时使用它。

建议：避免把帧用于布局，要在导航设计时使用它。

如果要使帧的应用有意义，坚持图5-15的模式。用户将会用到那些普通模式中的一个；这样，当点击时，不知道什么将被更新的负面效应将会因为这些经验而被改善。

所有这些帧的布局的共同特性是小的区域控制着大的区域。考虑到导航一般应小于当前的内容，这一点很有意义。同样，这些区域能够控制相邻的或下面的区域。

建议：当使用帧的时候，要使小帧控制相邻的大帧。

除了两帧、三帧或四帧的布局外，许多设计者倾向于使用与图片帧相似的固定帧布局，把内容固定在屏幕的中央，因为它的突出性布局超越了导航的作用。遗憾的是，虽然这种独特的布局代码深具潜力，这样的设计可能有很多问题 —— 特别是在不同的浏览器中，帧布局中存在一些细微的问题。图 5-13展示了一个固定帧设计的例子。应注意到两个浏览器屏幕间的细微差别。尽管未来也许能解决这些问题，但不要低估了浏览器中并非如此细微的帧显示问题。

微软浏览器支持 <IFRAME>标记，使得像图 5-16所示的布局做起来很容易。这样有可能用CSS和JavaScript相结合创造一个类似的布局。一段时间以后将很有可能很少依赖帧做布局。

China-pub.com 下载

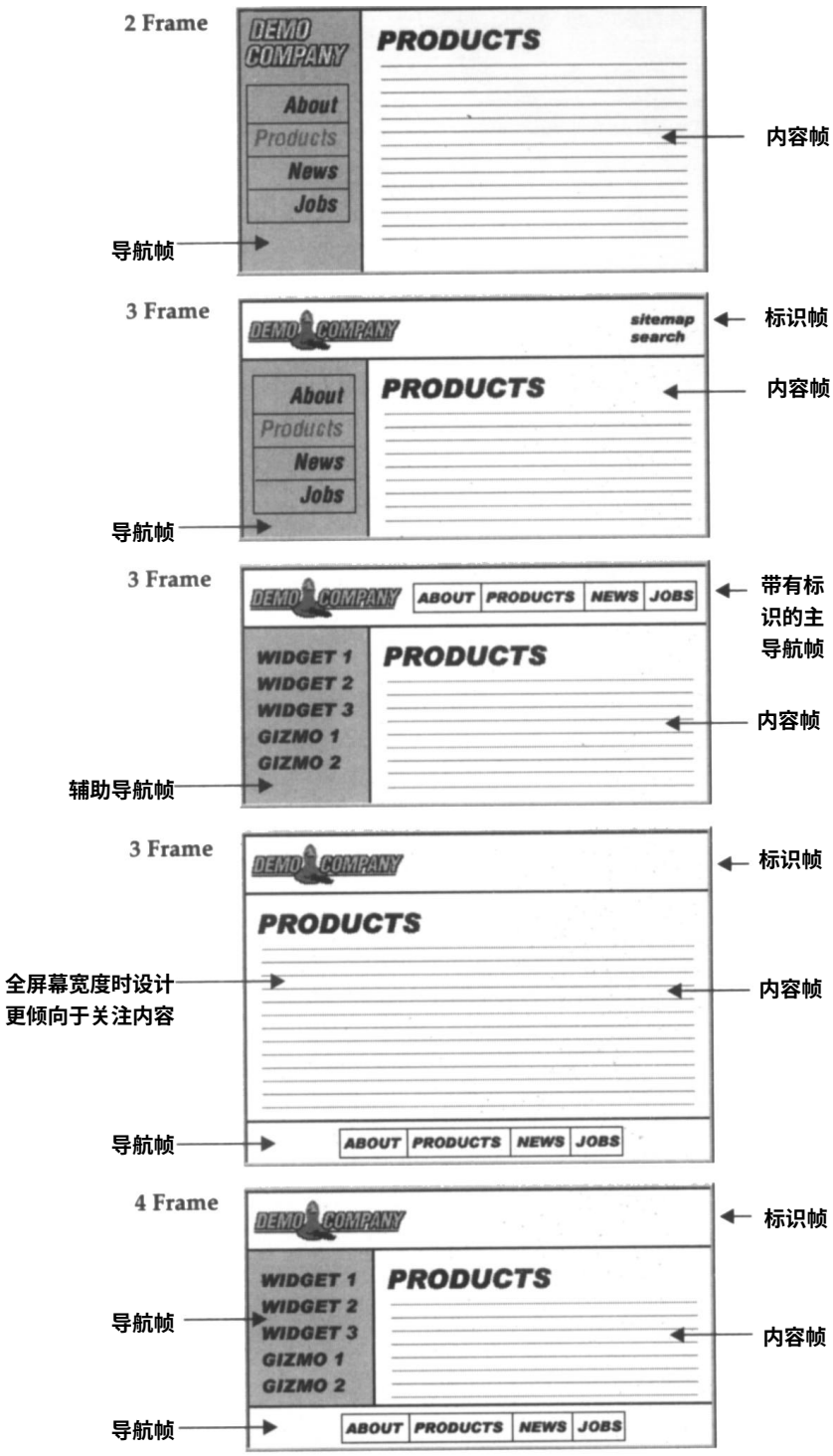


图5-15 一般的帧设计

China-pub.com
下载

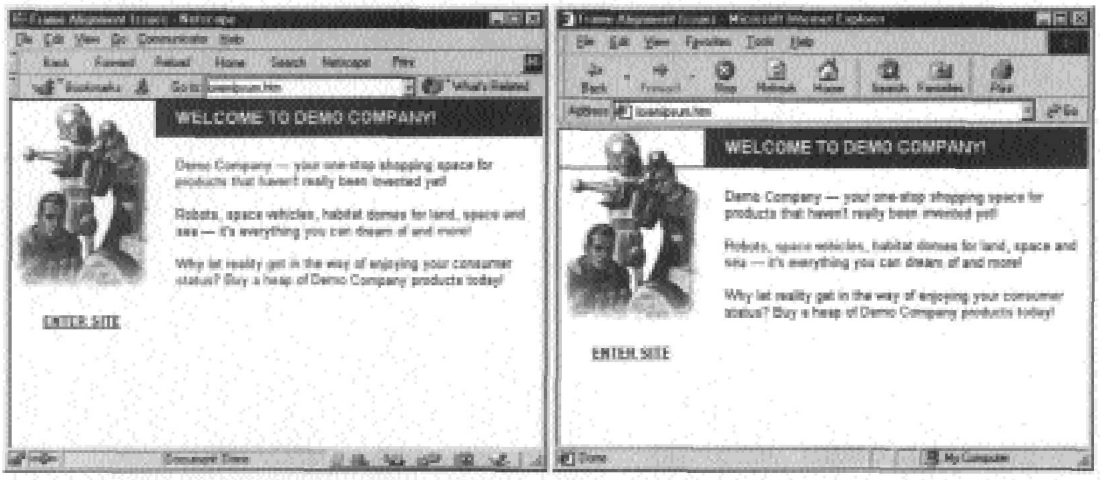


图5-16 复杂的帧使用可能会产生布局问题

1. 打印帧

关于打印帧的最好建议是使用户清楚地知道屏幕的什么区域在什么帧内。当打印时点击相应的区域，用户可以很快对这个问题熟悉起来。也可以像图 5-17所示的那样，在帧页上添加一个打印按钮。

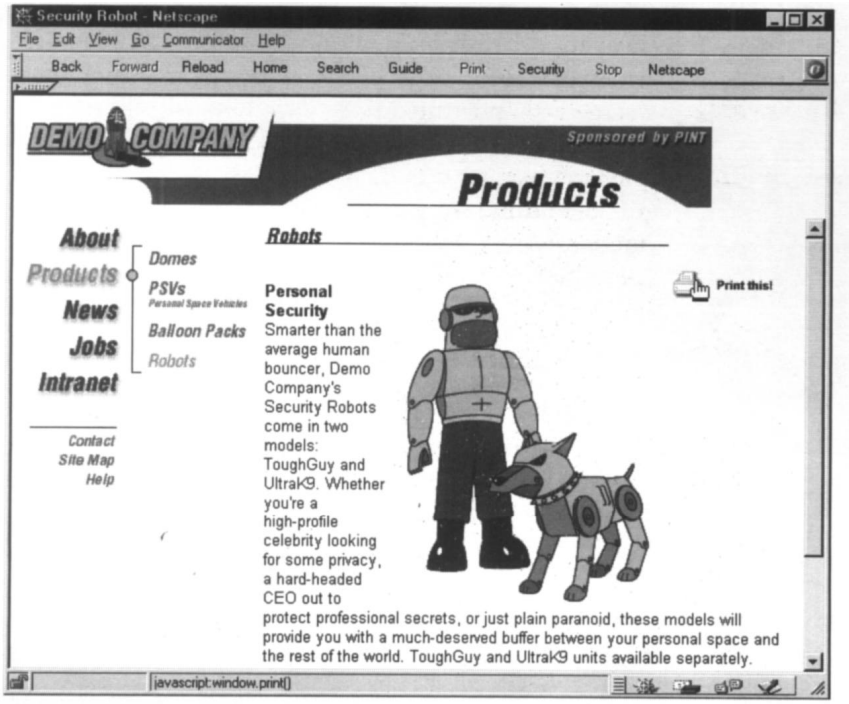


图5-17 使用帧时用明显的打印按钮

打印按钮能用于打开一个新的没有帧的页面，或使用代码语言立即激活一个打印对话框。

China-pub.com 下载

下面的代码说明如何添加按钮：

```
<SCRIPT LANGUAGE="JAVASCRIPT">
<!--

if (window.print)
 {
   document.write('<A HREF="javascript:window.print()">');
   document.write('<img src="../images/printer.gif" width="110"
height="40" border="0" align="right"></A>');
 }
else
 {
   document.write('<A
HREF="javascript:window.top.location=window.location">');
   document.write('<img src="../images/printer.gif" width="110:
height="40" border="0" align="right"></A>');
 }

// -->
</SCRIPT>
```

2. 书签问题

由于帧显示的是帧文件的URL而并非实际的内容，所以对用户来说为页面做书签一般是困难的。首先，你也许不希望用户对某些页面做书签。许多复杂的商业站点特意把内部页做成帧，因为它们经常动态地产生，可能有十分复杂的URL。

但是，经常是用户想试图给某一帧页做标签，而浏览器没有显示正确的页。举个例子，如图5-18 所示，经常是用户给左面的页做了书签，离开后再返回，他们发现了在右面的初始化的帧页。注意到在两个屏幕上URL是一样的。

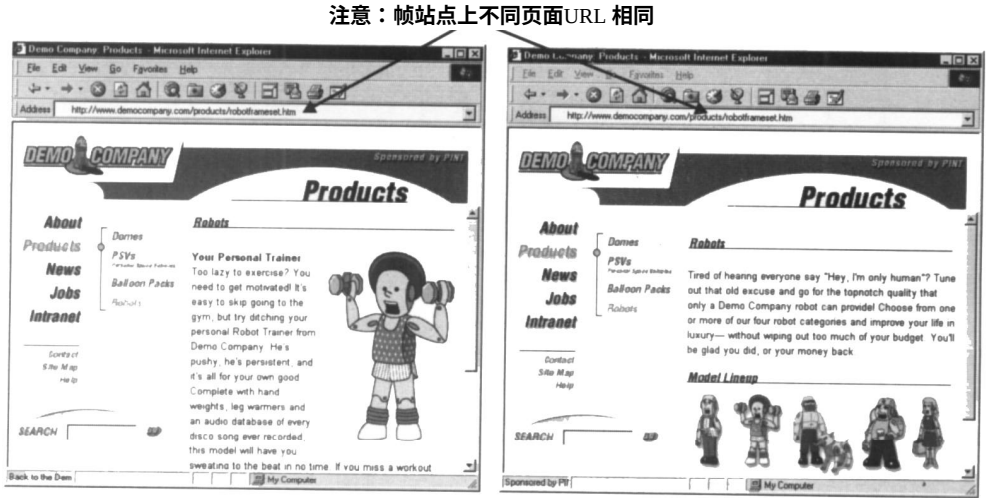


图5-18 因为URL是固定的，书签帧设置可能有困难

China-pub.com
下载

幸运的是，一些新的浏览器如IE5就能够处理帧书签问题。当然，某些老练的用户将想出在不顾及浏览器支持的情况下只对帧页做书签的办法。遗憾的是，他们将发现丢失了所有的内容信息，如导航，当他们返回时发现他们不经意已创建了一个孤立页，用户不结合 URL就不能导航，如图5-19所示。

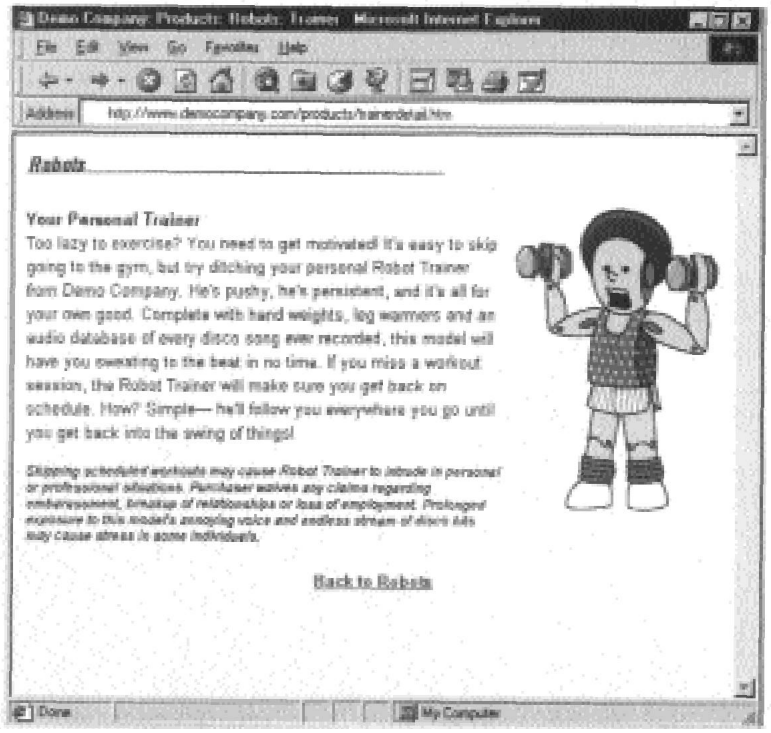


图5-19 给帧内容设置书签实际上会产生孤立页面

有两种方法处理帧书签问题。第一种是创建多重帧设置，每一帧有各自的可以做书签的不同的URL。这种处理问题的办法会降低帧屏幕刷新的优点，你会被迫创建多重文件。

书签问题的另一个较好的解决方法是用程序语言检验用户是否进入了一个无帧页，并且如果需要就动态地建立相应的帧。例如，如果用文件 Frameset.htm为站点的某一节定义了帧设置，就能够用JavaScript来判断一个页面是否在文件所定义的帧内。如果不在，就把地址设置为帧文档的初始地址。以下的代码显示了这是如何做的。仅仅在所有的帧文件内放置 <HEAD>标记。

```
<!--
 var containingwindow =
top.location.pathname.substring((top.location.pathname.lastIndexOf(
"/"))+1).
toLowerCase();

  if (containingwindow!="frameset.htm")

    top.location.replace("frameset.htm");
```

China-pub.com 下载

```
//-->
</SCRIPT>
```

如果用户深入到帧设置的内部，这段程序就不起作用，重新生成的帧将指向帧设置中的初始文件。取而代之的是，我们不得不根据不在帧内的页动态创建它自己的帧设置。以下的文件解释了这个过程。确保在真的Web服务器上运行这个例子。否则，因为URL不会正常生成，这段程序就不会运行。

File: frameset.htm

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Dynamic Frames Demo</TITLE>
</HEAD>
<SCRIPT>
<!--
function getPage() {
   return unescape(
window.location.search.substring(window.location.search.
indexOf("=")+1));
  }

  document.write('<FRAMESET COLS="100,*">');
  document.write('<FRAME SRC="controls.htm" NAME="controls">');
  if (window.location.search=="")
    document.write('<FRAME NAME="display" SRC="page1.htm">');
  else
    document.write('<FRAME NAME="display" SRC="'+getPage()+' ">');
  document.write("</FRAMESET>");
// -->
</SCRIPT>
</HTML>
```

File: controls.htm

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Control Frame</TITLE>
</HEAD>
<BODY>
<A HREF="page1.htm" TARGET="display">Page 1</A><BR>
<A HREF="page2.htm" TARGET="display">Page 2</A><BR>
<A HREF="page3.htm" TARGET="display">Page 3</A><BR>
</BODY>
</HTML>
```

File: Pagel.htm

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
```

China-pub.com
下载


```
<HTML>
<HEAD>
<TITLE>Page 1</TITLE>
<SCRIPT>
<!--
var container = "frameset.htm";
  var wname =
top.location.pathname.substring((top.location.pathname.
lastIndexOf("/"))+1).toLowerCase();
  if (wname!=container)
    parent.location.replace(container +
"?display="+escape(this.location));
// -->
</SCRIPT>
</HEAD>
<BODY>
<H1 ALIGN="center">Page 1</H1>
</BODY>
</HTML>
```

另一些文件，如Page2.htm和Page3.htm，与Page1.htm是一样的，仅仅更改了它们的标题，你能够找出差别。这个演示的关键是给帧页做标签。选择一个如 Page1.htm的帧页直接为其做标签。现在，当你返回到该页，它将自动地产生环境设置 ,所以该页不是孤立页。这个技巧也可以采用服务器端的技巧实现，它的唯一不足之处是，它将引起服务器在建立帧设置时产生额外的回路。

技巧 Web链接：动态帧设置的例子参见：

`http://www.Webdesignref.com/chapter5/dynamicframes.htm`

3. 布局问题

当用帧布局或导航时，不要过多限制是重要的。许多设计者关掉边框，关掉除去内容帧的所有帧的滚动，并限制用户改变帧的尺寸。尽管这样做会产生一个漂亮的布局，但考虑一下用户的屏幕并非足够大，不能整个装下帧设置时，会有什么发生。如果没有合理使用屏幕的分辨率，当关掉帧的尺寸设置和滚动时，你一定要小心。只有当设置的导航帧的尺寸恰好等于按钮的最小尺寸时，才能考虑这样做。

建议：除非显示器的分辨率给出了很好的说明，否则不要关掉帧的尺寸设置和滚动。

帧边界是另外一个问题。在帧环境下保持边界能够很好地显示出帧在哪里，但可能会导致较差的布局，如图5-20的例子所示。

但是，如果用户不得不重新设置帧尺寸，保持帧的边界是一个不错的主意。

4. 帧的busting

帧的一个普遍问题是，一个帧设置在另一个已存在的帧内开始显示。有些时候，外面的站点为了吸引用户故意这样做。另外一些时候，帧在帧中出现仅仅是一个简单的错误。这个演变出来的错误被冠以“俄国木偶问题”的绰号，就像木偶中包含同样的小一些的木偶一样。两个例子如图5-21所示。

China-pub.com 下载

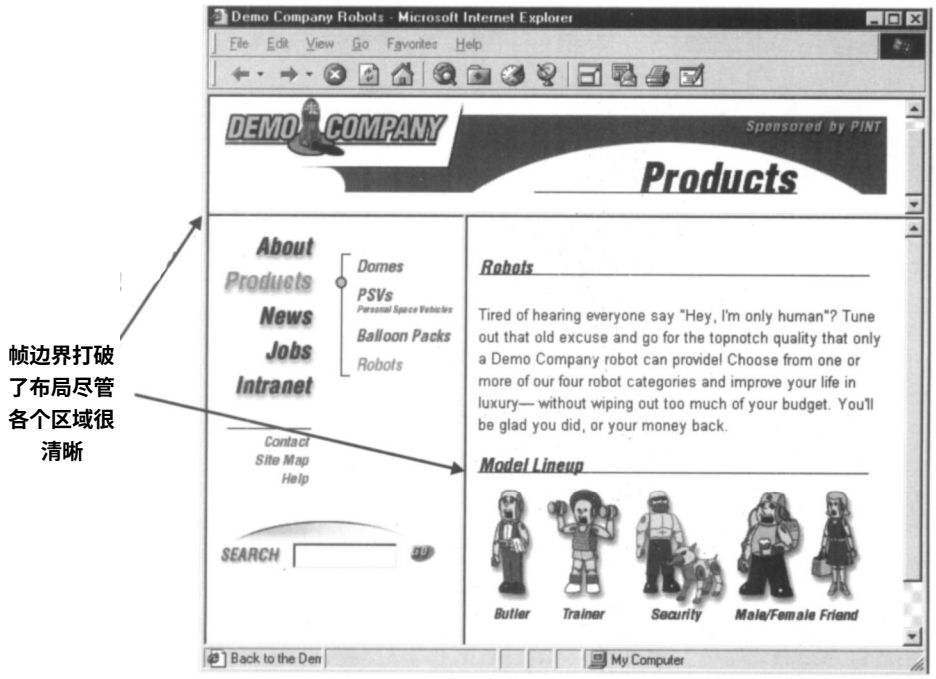


图5-20 帧边界可以变得很难看

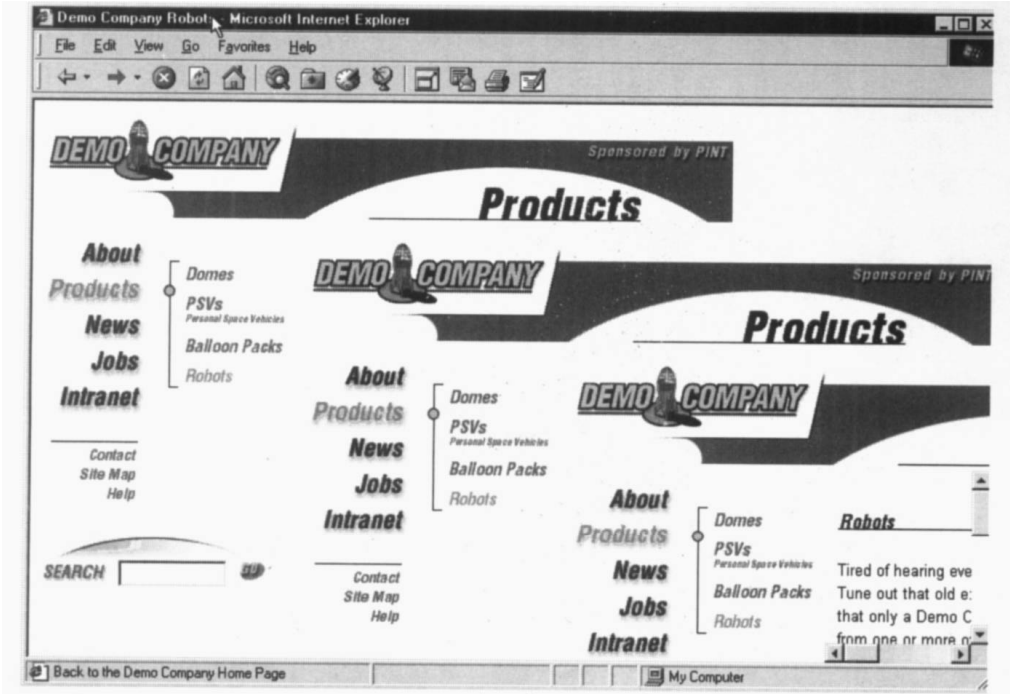


图5-21 帧中帧的例子

China-pub.com
下载

尽管一个例子只是设计的错误，你也许想知道如何防止用户重叠你的站点。如果在自己的站点内没有使用帧，一个简单的方法是，只需写出 HTML来保证站点中的每一个链接都把TARGET属性设置为_top,如下所示：

```
<A HREF="robots.htm" TARGET="_top">Robots</A>
```

应用这种办法，任何时候当用户在帧设置之中时，要装载的下一个链接被放置在已存在的帧的上部。这种思想被称为“帧busting”。另一种方法是应用一小段JavaScript来检查一页是否加有帧，然后再绘制帧。这也可以用于自己的帧文件中，你可以在自己的站点中安全地使用帧。

```
<SCRIPT LANGUAGE="JavaScript">
<!--
if (window != top) top.location.href = location.href;
// -->
</SCRIPT>
```

然而，要小心对待帧busting。尽管的确可以用额外得到的资源改善布局，但用户也许需要帧的环境。考虑到也许用户为了方便地返回老站点，只是经过你的站点，需要的是你周围的其他站点的导航。这样控制问题又出现了。当然能够写一段程序来检测用户是否想取消帧，但这将打扰用户。

```
<SCRIPT LANGUAGE="JavaScript">
<!--
if (window != top)
 if (confirm("Remove framing document?"))
   top.location.href = location.href;
// -->
</SCRIPT>
```

<NOFRAMES>

帧的一个潜在问题是所有的浏览器都不支持帧。许多老的浏览器如 Netscape或Mosaic,还有

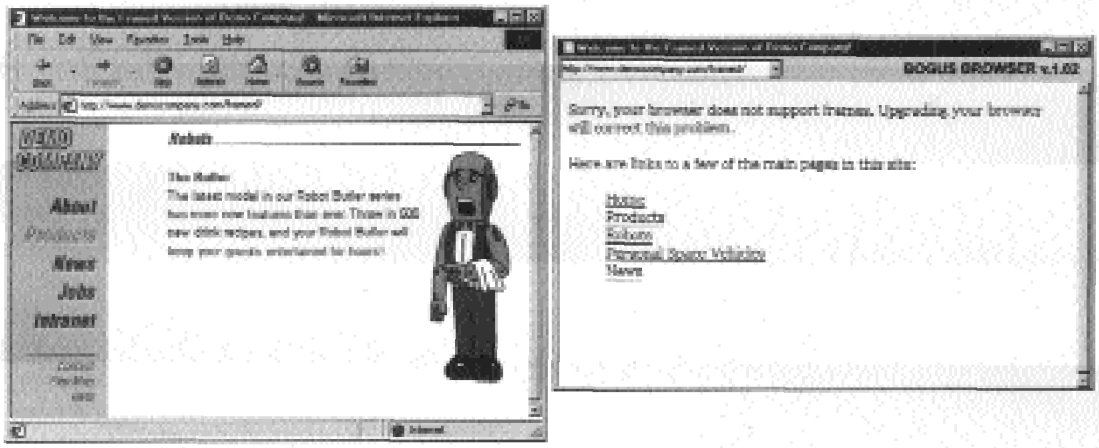


图5-22 不支持帧的用户将看到一个不同的站点

China-pub.com 下载

Web设备的上浏览器或手持设备，一般都有与帧页面相关的限制。更进一步地，许多搜索引擎不能用帧索引站点，这将严重限制用公共搜索引擎的能力，这一点将在第 7章中讨论。如果使用了帧，要确保至少一些出现过的内容在一个帧设置文件中应用 <NOFRAMES>标记。用户没有帧和搜索引擎的支持，将能看到内容，然而典型的用户能看到帧站点，如图 5-22所示。

当然，处理支持帧和不支持帧的浏览器将意味着对一种页面创建两种版本。尽管能够动态地建立，但代价昂贵。但是，在简单地创建一个告知用户更新浏览器的〈 NOFRAMES>页之前，应考虑到许多用户看上去很不喜欢帧而欣赏不使用帧的站点。

5.9 子窗口

另一种基于“ 远程 ”思想的导航方案不如帧流行。远程基本上是从浏览器分离出来的一个小窗口，能够用来向主窗口装载内容。图 5-23展示了一个远程的例子。

从某种意义上讲，“ 远程 ”在适用性方面和帧是类似的，只是它不依附于主窗口。远程可以认为与一个下拉式菜单等同。创建一个远程是相当容易的，只是命名一个窗口然后创建一个二级窗口或远程，使之与目标主窗口链接起来。例如，文件 remotetester.htm开始运行一个远程，而remote.htm 包含用于把TARGET属性设置为主窗口的链接。

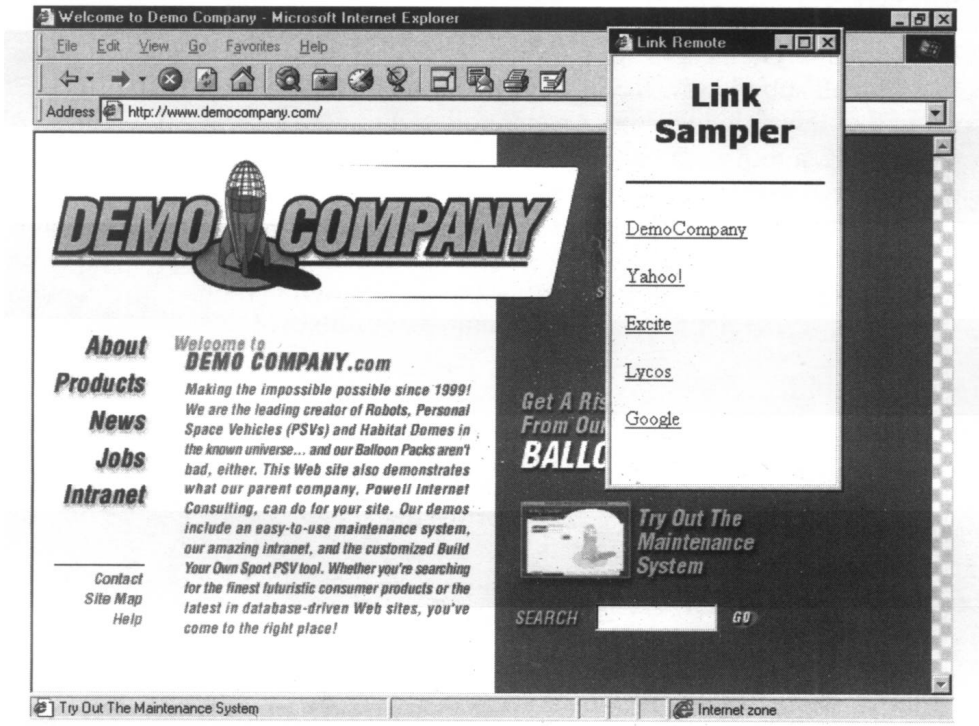


图5-23 远程的使用

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0
Transitional//EN"><HTML>
```

China-pub.com
下载


```
<HEAD>
<TITLE>Remote Tester</TITLE>
<SCRIPT LANGUAGE="JAVASCRIPT">
<!--
window.name = "mainwindow";

function spawn()
 {
  remotewindow=window.open("remote.htm", "remote",
"toolbar=0,location=0,directories=0,status=0,menubar=0,resizable=1,
copyhistory=0,width=200,height=400", false);
 }
// -->
</SCRIPT>
</HEAD>
<BODY BGCOLOR="#FFFFFF">
<H2 ALIGN="center">Remote Tester</H2>
<HR>
<P>
<A HREF="javascript:spawn()">Spawn Remote</A>
</P>
</BODY>
</HTML>
File: remote htm
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>Link Remote</TITLE>
</HEAD>
<BODY BGCOLOR="#000000" TEXT="#FFFFFF" LINK="#FFFF33"
VLINK="#FF0000" ALINK="#FF0000">
<H2 ALIGN="center"><FONT COLOR="#FF6600" FACE="Verdana, Arial,
Helvetica, sans-serif">Link Sampler</FONT></H2>
<HR>
<A HREF="http://www.democompany.com/"
TARGET="mainwindow">DemoCompany</A><BR>
<A HREF="http://www.yahoo.com/" TARGET="mainwindow">Yahoo!</A><BR>
<A HREF="http://www.excite.com" TARGET="mainwindow">Excite</A><BR>
<A HREF="http://www.lycos.com" TARGET="mainwindow">Lycos</A><BR>
<A HREF="http://www.google.com" TARGET="mainwindow">Google</A><BR>
</BODY>
</HTML>
```

远程的主要问题是它们很容易丢失。一些用户把他们的操作环境设置成一些自动的低矮窗口，这样做很容易把远程隐藏在主窗口的后面而丢失，另一个可能出现的问题是，远程反而起了妨碍作用，在一个很小的屏幕上，由于没有位置用来放置远程，它可能会悬挂覆盖在页面内容之上。对于经常使用的站点，远程是很有意义的，但其他站点只把它们当做一种参考模式。

China-pub.com 下载

建议：不要把远程做为导航的强制形式。

5.10 书签设置

书签是导航的一个重要方面。通常，用户对某一页加上标签，将来返回到该页。在一些浏览器上，用户甚至对某页加标签以便能下载该页，在离线的情况下浏览，或在一个规则的基础上检查有哪些自动的变换。传统上讲，设计者不可能影响书签，除了使用户用帧对某页做书签变得更困难了。但是，对 IE 5或以上版本，在某种程度上定做书签是可能的，用微软的说法叫“偏好”。

最简单的定制方式是创建一个自定义的图标与一个书签相联系。首先，使用一种叫图标创建的工具创建一个希望用户看到的图标。图标文件必须以 .ico形式存储，维数必须是为 16×16像素；否则，浏览器将会忽略它。下一步，复制图标到为一个特殊域处理页的 Web服务器的根目录下。因特网浏览器将会自动使用这一图标，在任何时候为用户设置一个“偏好”，或快速链接站点。

注意 使用标准的书签文件会发生潜在的个人隐私风险，一些人对此表示了关注，考虑到对一个标准图形文件如 favicon.ico的请求可能从日志文件中筛选得到，并用于暗示用户对某页做了书签。

应用HTML标记逐页设置“偏好”是可能的。例如：

```
<LINK REL="SHORTCUT ICON"
      HREF="http://www.democompany.com/icons/robot.ico">
```

大多数用户会使用浏览器的书签菜单，但也可能加载自己定制的按钮，以便用户能自动地把页加载到他们的偏好栏中。以下的程序显示了在激活一个对话框把当前页加载到偏好栏中时如何增加一个链接：

```
<SCRIPT>
<!--
if ((navigator.appVersion.indexOf("MSIE") > 0) &&
(parseInt(navigator.appVersion) >= 4)) {
   document.write("<U><SPAN STYLE='color:blue;cursor:hand;'
   onclick='window.external.AddFavorite(location.href,
   document.title);'>Add this page to your favorites</SPAN></U>");
}
//-->
</SCRIPT>
```

制作书签的一个重要考虑是确保动态数据的处理。例如，考虑到有一个特殊的 URL，能够在http://www.democompany.com/latestnews储存最新的新闻发布。在一个特殊项后不再是最新项，它被移到一个存档处，并用一个新项来代替。想像一下，如果用户对某一特殊的内容加了标签却发现它们已改变时，他们表现的困惑。如果有可变的数据，你也许考虑在能够产生一个新窗口的页面上创建一个特殊的书签按钮。使用 javaScript对前一段代码稍做修改，如下所示：

China-pub.com
下载


```
<SCRIPT>
<!--
archivelocation='http://final-url-goes-here';
if ((navigator.appVersion.indexOf("MSIE") > 0) &&
(parseInt(navigator.appVersion) >= 4)) {
   document.write("<U><SPAN STYLE='color:blue;cursor:hand;'
   onclick='window.external.AddFavorite(archivelocation,
   document.title);'>Add this page to your favorites</SPAN></U>");
}
//-->
</SCRIPT>
```

始终考虑某些页面是否是站点的入口点。如果是，要保证该页容易被加上书签。

导航的禁忌

在对本章做总结时，让我们简要地回顾一下导航的禁忌，我们已经涉及到了许多点。这些问题中没有一个会彻底摧毁一个站点的导航，但每一个都能极大地迷惑用户。

1. 回退按钮劫持

用户最喜爱的浏览按钮是回退按钮，特别是对新手。遗憾的是，许多站点通过重定向的使用而关掉了回退按钮。在制作那些仅仅是立即重定向用户到另外一页的特殊页时，一定要小心。这样做经常是用JavaScript检测浏览器。在服务器端检测浏览器更明智。

2. 弹出式窗口

当用户准备离开站点时，许多站点开始弹出窗口。通常，这些窗口包含其他站点的广告或试图阻止用户离开站点。十分不道德的设计者也许试图纠缠用户，又送他们回到他们试图离开的站点。如果用户想走就让他们走，不要讫求。

3. 独特的导航

用户整天在网上漫游，对那些太离普的站点，他们很少有足够的耐心。当用户习惯于这些现有的方式时，一些设计者对由此导致的导航雷同性表示哀叹。他们辩解道： Web导航开始变得看起来很类似，用户都不能辨别出站点的区别，并且为创造的灵活性留下了太小的空间。然而，站点导航的一致性对稳定性起着重要的作用。用户开始理解如 Amazon和Yahoo的设计！为什么能够利用用户先前的经验时却与这些设计偏离很远呢？人们知道这些网点如何工作，就像他们理解基本的Web惯例如蓝色作为链接的颜色，以及GUI惯例。进一步说，考虑大多数文字处理器和电子表格工作方式一样。为什么大多数商业网点不遵循这一点呢？记住，你不是向用户兜售导航！

4. 重品牌按钮

过于强调导航元素而不注重内容，经常导致设计者试图“按钮品牌化”并使导航精于设计。这里的意思是说，应力图创建一个看起来有明显不同的外观来展示品牌。一个看上去很特别的按钮也许能被用户记住，但又觉得异常而反对这样做。设想上次你真的坐着欣赏电梯里的按钮。好，也许你是按钮的设计者，问问和你同乘电梯的人有什么想法，他们会认为你有些稀奇古怪。再说一次，记住用户使用导航只是为完成他们已开始着手的任务。

China-pub.com 下载

5. 依赖回退按钮

如果主要依靠回退按钮来导航，特别是对于内容页，你有可能会产生一个无法退出的孤立页。如果用户通过一条特殊的路径穿过站点，回退按钮将很好地工作。但是，用户可能不是按你所想的途径进入站点。设想用户对某页加了书签，另一天返回时，回退按钮不能让它退出。

6. 不要让用户工作太辛苦

不管喜欢不喜欢，人们可能是有些懒的，一般只愿意浏览不太费力的站点。当用户继续使用这一站点时这一点是很重要的，按钮放置得很明，显并可识别。不要迫使用户在使用站点时视觉上、脑力和身体上过于疲劳。用户不应该回忆要选择按钮的信息，他们应该简单地认识这些选择。最后，不要使用户在环游站点时使用过多的身体动作。导航应尽可能地省劲。再强调一次，注意检查鼠标移动的量，仔细注意随后的选择。

规则：在导航中尽可能地限制滚动和鼠标的移动。

同样，应计算到达一个目标页需要点击鼠标的次数。你经常认为三次是点击的最多次数，但不应该仅看中点击；也许是装载页时间过多，经常是 Web的来回路程时间使用户感到挫折。考虑限制导航深度到装载三个页面。

规则：在一个结果之前最多装载三个页面。

创建一个糟糕的导航很容易，但有时要设计者检查出失误之处却很困难。原因是，作为一个设计者，你应该知道如何浏览自己的站点。牢记你收到的用户对导航问题的任何抱怨。如果你怀疑某个问题，就迅速对站点的导航进行评估，这一点将在附录 B中讨论。

5.11 小结

在现实世界里，人们依赖于不同的环境使用不同的导航方法。例如，人们在参观一个博物馆，逛公园，光顾一个商店或寻找一个朋友的房间时方式是不同的。根据手头的不同任务，导航的技巧也在变化。不管站点的形式如何，导航的目标是能够简单地帮助用户找到他们的路线。一个好的导航方式能够帮助用户回答诸如“我在哪？”,“我能去哪？”,“如何到达想去的地方？”“我以前到过这里吗？”和“如何回到曾经到过的地方？”等问题。页面标志、 URL、标志页、页面方式和颜色能够帮助用户识别位置。应能够加入导航元素帮助用户对将来的目标做出选择。然而，导航项的放置和稳定性应计算好。高级技巧如隐藏菜单、帧和远程会产生某些问题，在执行前要考虑它们带来的问题。最后，要一直记住，导航是结束的途径，而不是结束本身。一般来说，用户并不是为导航系统的美丽而赞叹；事实上，他们有可能把站点作为他们在Web旅游中的一个小的驿站。站点不能把他们的导航注意力引向无价值的地方。实际上，如果用户注意得太多，我们就不能做好每一项工作。下一章将讨论各种导航元素如链接和搜索引擎，导航帮助和站点映像的使用。