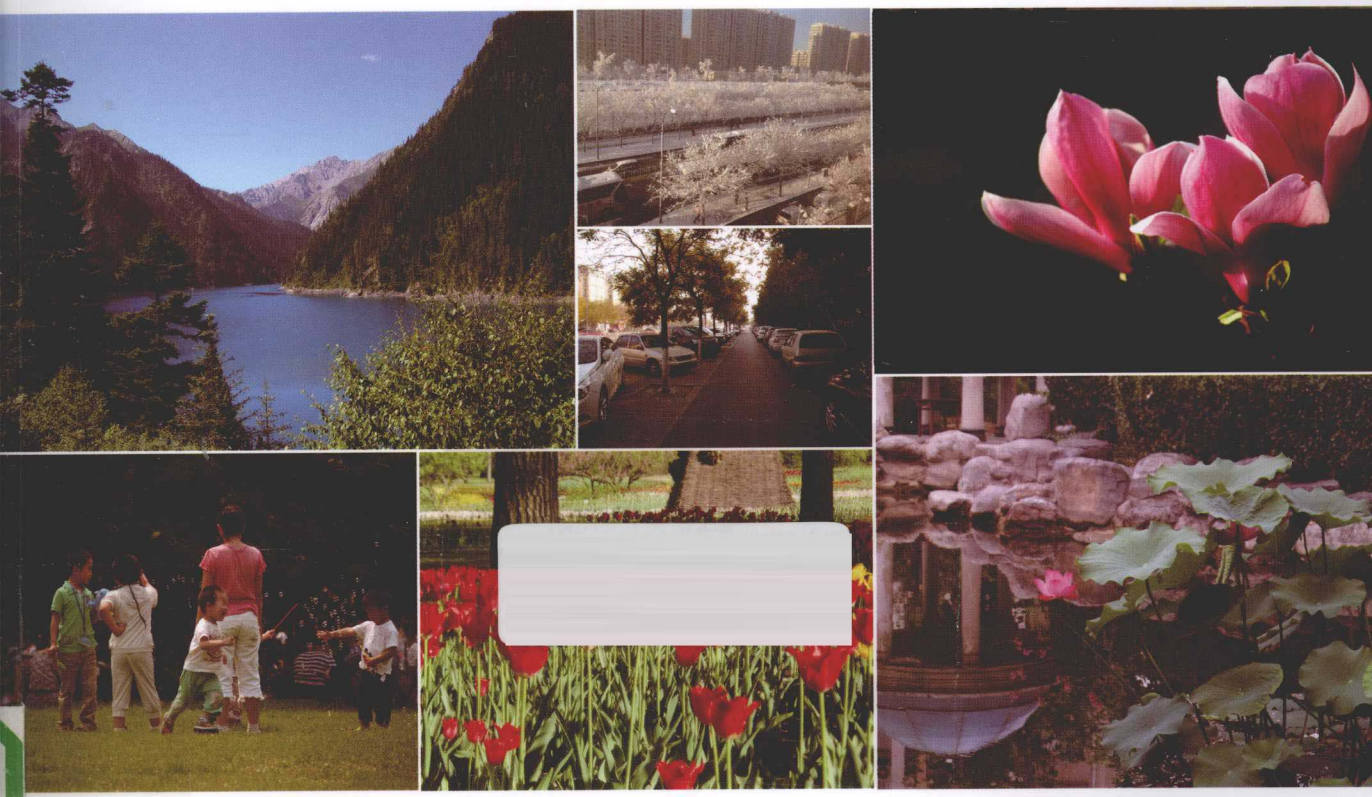
Sencha Ext JS JavaScript Framework for Rich Desktop Apps

Ext JS 4.2实战



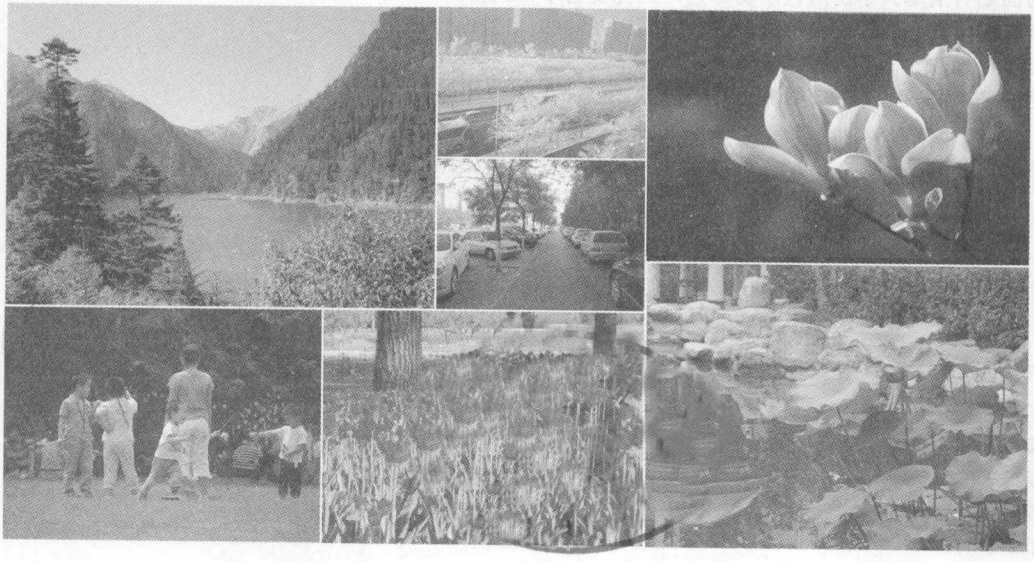
黄灯桥 编著

Ext JS从4.0开始，架构发生了重大变化，本书以4.2版本为基础，通过一个具体应用程序示例的实现，一步一步带领读者进入Ext JS 4.2的开发世界，使其掌握使用Ext JS 4.2开发应用程序的技术

本书示例项目源代码

清华大学出版社

Ext JS 4.2实战



黄灯桥 编著

清华大学出版社
北京

内 容 简 介

本书是一本 Ext JS 实战系列的书，主要通过简单的 CMS 系统的开发过程，介绍了使用 Ext JS 4.2 开发应用程序的新模式和新思路。本书也融入了作者使用 Ext JS 进行开发的实践经验。由于讲解 CMS 系统实现时，使用了 ASP.NET MVC 4 架构，所以本书也是学习 ASP.NET MVC 4 开发不可多得的书籍。

本书总共 15 章，前两章主要是一些介绍性的内容。从第 3 章到第 14 章，涵盖了简单的 CMS 系统从搭建开发环境到打包、发布和部署的整个开发过程。第 15 章介绍了 Ext JS 的跨平台特性。如果不是使用 C#进行开发的开发人员，可以先看第 15 章，了解 Ext JS 的跨平台特性，然后再慢慢地研读其他章节。

本书适合 Ext JS 4.2 富客户端开发人员、初学者，也可以作为高等院校和培训学校相关专业的教学参考书籍。

图书在版编目（CIP）数据

Ext JS 4.2实战 / 黄灯桥编著. - 北京：清华大学出版社，2014
ISBN 978-7-302-35339-3

I. ①E… II. ①黄… III. ①JAVA语言－程序设计 IV. ①TP312

中国版本图书馆CIP数据核字（2014）第021089号

责任编辑：夏非彼
封面设计：王　翔
责任校对：闫秀华
责任印制：何　芊

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：三河市李旗庄少明印装厂
经　　销：全国新华书店
开　　本：190mm×260mm　　**印　　张**：20.75　　**字　　数**：531 千字
附光盘 1 张
版　　次：2014 年 4 月第 1 版　　**印　　次**：2014 年 4 月第1 次印刷
印　　数：1～3500
定　　价：51.00 元

产品编号：056140-01

前　言

在《Ext JS 权威指南》出版后，为了弥补书中没有完整的开发示例的遗憾，我在博客中开了一个名为《Ext JS 开发》的专栏，结合 Ext JS 4.1.1 和 ASP.NET MVC 3 写了一个完整的示例，反应不错。不久，Ext JS 4.2 发布了，研究后发现 4.2 在 4.1 的基础上做了比较大的调整，而且在开发方式上又有了新思路。这时恰逢清华大学出版社编辑夏毓彦约我写一本与 Ext JS 实战有关的书，于是顺理成章地促成了本书的出版。

大家知道，Ext JS 是一个使用 JavaScript 编写的、用来开发富客户端的 AJAX 框架，主要用于创建前端用户界面，支持跨浏览器且与后台技术无关。它可以与 ASP.NET、Java、PHP 等各种语言结合开发各种应用。本书以一个简单的 CMS 系统开发为线索，详细讲解 Ext JS 4.2 的使用，同时，也给大家展示了 Ext JS 开发的一种最佳模式，这种模式一直存在，但由于网络上介绍得不多，相关的文章也很少，因而很少被人熟知而采用。这种模式是开发 Ext JS 应用程序的最佳方式，编者详细介绍了这种最佳模式的开发方法。这种模式的关键是要配合 Sencha Cmd 一起使用，利用 Sencha Cmd 的应用程序创建和生成功能，来创建应用程序并打包发布应用程序。

所谓熟能生巧，经过这么多年使用 Ext JS 的经验累积，也到达了从量变到质变的阶段，所以，本书除介绍新的开发模式以外，还对自己使用 Ext JS 的开发实践进行了总结，提出了一些新的开发思路，希望对大家有所启发。

本书另外一个特色，就是使用 ASP.NET MVC 4 作为后台开发语言。可以说，ASP.NET MVC 4 又是一种革新，它提供了新的思路来进行开发。它与 Ext JS 4.2 结合在一起是非常完美的组合，这样的结合可以说是高效开发的代表。

虽然本书的后台语言是使用 ASP.NET MVC 4 开发的，但并不意味着对于使用其他开发语言的开发人员就没有帮助。原因在于 Ext JS 是一种跨平台的框架，几乎不经任何修改就能迁移到不同的开发平台，这也是 Ext JS 的魅力所在。不过，要让 Ext JS 发挥跨平台的威力，还需要在开发思路上进行创新，而这正是本书将要讲解的重点。

希望本书能带给你全新的 Ext JS 开发体验，也希望对于使用 C#平台进行开发的读者会有所帮助。

本书的开发环境如下。

- 操作系统：Window 7
- 开发工具：Visual Studio 2012
- 数据库：Visual Studio 2012 本地数据库
- 浏览器：Firefox 24.0

为了能不受系统自身环境的影响，在使用 Sencha Cmd 时候，专门使用虚拟机搭建了一个干净

的 Window 7 系统进行安装和使用。如果在本机安装和使用 Sencha Cmd 碰到无法处理的错误时，建议读者也使用虚拟机搭建一个干净的 Window 7 系统再进行安装和使用，以避免自身环境的影响。

本书的源代码都在本书配套的光盘中。光盘中的每一个目录对应的是书中每一章的源代码，在 ch 后面的数字就是对应的章节号，例如 ch03，就是是第 3 章的源代码。

希望本书能给每位读者带来帮助，如果对本书有任何意见和建议，或者有任何技术上的问题，请发邮件到 huangdengqiao@outlook.com，或者加我微博 http://weibo.com/gerneal。如果想了解最新的 Ext JS 动态或编者的最新博文，可访问编者的博客：

http://blog.csdn.net/tianxiaode

http://dqhuang.blog.51cto.com/

在本书的出版过程中，得到了清华大学出版社图格事业部编辑的大力支持，在他们的努力下，促成了本书的出版，在此表示衷心的感谢。此外，还要感谢那些在互联网上默默耕耘的博客作者以及在各大论坛回复问题的大牛们，是他们的努力耕耘，才使我能找到解决问题的办法，是他们让我有了更进一步提高技能的机会。

编者

2014 年 2 月

目 录

第1章　Ext JS 4概述

Ext JS 4在短短一年多的时间内，从4.0、4.1到4.2，每一次版本号的更改都意味着重大的变化。如果按照正常的版本命名方式来算，应该算是4、5和6版本了。而每次这样的升级，造成的兼容性问题，会让开发人员有点迷惑，不知道应该在开发中选择哪个版本。本章将讲述这些版本之间的不同，并介绍了应该如何去选择这些版本。

1.1　从Ext JS 4.0到Ext JS 4.07

Ext JS 3是一个相当成功的产品，尤其是在浏览器的兼容性和性能方面，堪称经典。要说不足的地方，那就是Ext JS 4所带来的全新的开发体验，扩展更灵活、更快速，更适合于大型的项目。Ext JS 4对Ext JS 3进行了重构，主要的改进包括以下几个方面。

- 新的类系统
- 标准化类的命名规则
- 动态加载
- 混入功能（Mixins）
- 自动的配置功能
- 新的数据模型
- 全新的绘图和图表功能
- 重新架构的Grid组件
- 新的主题特性
- MVC开发模式

对Ext JS进行重新架构可以说是前所未有的挑战，尤其是采用新的类架构，对性能来说，打击是巨大的。于是，在各个论坛上，充斥着对性能的抱怨。不过，主要的性能问题主要集中在IE 8以及之前的版本IE。为了解决这个问题，在发布到4.1的时候，又对架构进行了重大的修改，因而，造成了4.1与4.0.7之前版本的不兼容。

1.2　从Ext JS 4.1到Ext JS 4.1.1a

4.1版本对于4.0.x来说，主要是为了改善性能，在渲染模式上进行了大的修改。为了适应这种

修改，对布局和模板都进行了修改，而在 API 上也相应地做了修改，而这正是造成兼容性问题的主要的原因。

这样的后果，可想而知，正在使用 4.0.7 进行开发的，或已经完成开发的，就面临着是否要升级到 4.1 的问题。而升级对开发人员来说是灾难性的，因为这样做存在太多的未知数了。譬如升级后，需要修改多少代码？而这些修改，是否会影响到项目的进度？

但是，不升级也面临着风险，毕竟 4.0.7 只是产品完善过程中的一个版本，还存在大量的臭虫，有些已知的，而还有更多是未知的。项目如果不能随着版本的升级而升级，那么这些臭虫就可能一直存在，随时会威胁到项目的开发和维护，除非自己去修改框架，以修正这些臭虫，但这也无形中增加了项目的开发成本，需要找专人去研究和修正 Ext JS 的臭虫，这对项目来说也是难于负担的。最重要的是，关注性能的项目，若不升级，就要面对性能的影响，这也是需要慎重考虑的。

总的来说，这是一个两难的选择。

1.3 从 Ext JS 4.2 到 Ext JS 4.2.1

4.2 在延续 4.1 的性能修改的基础上，添加了海王星主题，并支持右对齐语言，还根据反馈修改了 MVC 架构类。

为了更便于主题的开发，又对 API 和开发模式进行了修改，尤其是为了配合 Sencha Cmd 的使用，做了不少调整，而这样的调整，又造成了 4.2 与 4.1 之间的不兼容。

总的来说，不兼容对于开发人员来说是灾难性的，他们将不得不面对升级的选择与挑战。

1.4 如何选择版本

基于上面的版本说明，大家对 Ext JS 4 的各版本有了一定的了解，要做出选择也不会太难。

对于新项目，应该毫不犹豫地选择的最新版本，这样，才能随着 Ext JS 的升级与更新而获得更新，主要目的就是减少 Ext JS 臭虫（Bug）对项目的影响。不过，最大的风险还是后续版本的兼容性问题，这是大家都控制不了的。

那么，对于新项目，选择 4.0.7 或者 4.1.1a 又如何呢？对于 4.0.7 这个版本，我们已经知道最大的问题是性能问题比较突出，而且也不太完善，如果确信这些对项目影响不大，那么用起来也是问题不大的，但一定要注意臭虫的影响，有时候还必须自己去修改框架的源代码来修正臭虫，如消息对话框按钮的本地化问题。对于 4.1.1a 这个版本，关键还是臭虫的影响，如表单提交时的遮罩问题。因而，为了减少臭虫的影响，选择最新版本是最适合不过的。

对于已经在使用 4.2 之前版本开发的项目或正在开发的项目，如果旧版本暂时能满足需求，臭虫的影响也不大，那么可以作为阶段性任务，先使用旧版本完成项目，等项目成功交付后，就可着手下一阶段任务，考虑将项目整体迁移到最新版本。迁移的目的是为了在未来的项目升级中减少臭虫或其他方面的影响。

在选择版本的时候，还涉及学习的问题，由于学习资料的滞后性，要获取最新版本的学习资料，只能在网络中搜索，尤其是关注官方论坛和博客，而最大的问题是，这些学习资料都是英文的，也是一个很大的学习障碍。可行的办法就是由项目中经验丰富且英语比较好的开发人员将所需的资料

翻译为中文，然后分发给项目组中的成员进行学习。这里特别要提一下的是官方论坛，值得多关注一下，因为各版本所存在的臭虫或其他问题基本都可以在这里找到，从中可以发现哪些问题对你的项目有影响，而哪些问题是无关紧要的。

如果英文水平可以，通过阅读 API 中的指南就可以学到很多东西。但正因为 API 是英文的，从而成了一大障碍。因而，对于新版本的学习曲线，也是项目选择版本时要考虑的问题。而这方面，最主要是集中在 API 的改变上。

作为项目经理，可以先考虑让你的开发人员从中文材料充足的版本开始学习，然后逐步过渡到最新版本。学习过程是必须的，不然，让一些经验不足的开发人员直接使用 Ext JS 开发项目，所造成的问题就是项目的碎片化，可以说，就是临时用一堆 Ext JS 代码拼凑起来的项目，这样的系统，不容易维护，不健壮，不稳定，后果也就不用说了。在我所接触过的开发人员中，这个是普遍存在的问题。这些开发人员，基本都是临时转行过来使用 Ext JS 进行开发的，是项目组要求使用才使用的，基本上没有系统地学习过 Ext JS 的开发，尤其是 Ext JS 4 这种结构化很强的开发模式，从而造成他们往往习惯使用自己所熟悉的技术去处理 Ext JS 开发中的技术问题，产生了很多本来可以避免的问题。最常见的一个例子就是为标签页添加子组件的问题，习惯的模式是在标签页内使用 html 配置项放一个 DIV，然后将子组件渲染到这个 DIV 上，造成的问题就是子组件不能随标签页尺寸的改变而改变自身的尺寸。这个问题的症结所在，就是对 Ext JS 的布局和组件的使用了解不透彻。对于系统学习过 Ext JS 的开发人员，是不会出现这样的问题的。因而，作为项目经理，或项目的负责人员，应担起责任，为开发人员提供必要的 Ext JS 学习和培训，或为他们提供这样的环境，这样，对项目的好处是显而易见的，而且对于 Ext JS 的版本选择，也有更大的余地，因为，对于了解 Ext JS 的开发人员来说，版本升级中的 API 改变，问题是不大的，关键还是对于 Ext JS 功能特性的理解。

以上啰嗦了不少，不过要说的重点只有一个，尽量选择最新版本，把框架中的臭虫和其他方面因素对项目的影响降到最低程度，也便于项目以后的升级。

如果项目非要支持 IE 8 以下版本的 IE，建议还是使用 Ext JS 3，否则，面对兼容性问题真的会很头疼的。

1.5 基础知识

1.5.1 Ext JS 的一些专用术语

由于没有官方中文版本，因而 Ext JS API 文档中的一些术语都是直译的，在这里特意提醒一下，以避免混淆，主要包括以下几个。

- 配置项：API 的 Config options 中列出的用来定义组件的特性、功能或外观的属性。严格意义上来说，这本身就是属性，如果使用类似 Delphi、Visual Basic 等可视化开发工具 Sencha Architect 来进行开发的话，配置项就相当于这些工具属性窗口中的属性，可用来定义控件的特性、功能或外观。 不使用属性这一术语，是为了与 API 对应。在本书，只要说的是配置项，就可在 Config options 中找到。

- 配置对象：由配置项组成的对象，主要作用是用来定义组件的特性、功能或外观。在 Ext JS 中，一个配置对象就表示一个组件的定义，通过树状的层次结构来描述一个完整的界面。
- 属性：在本书中，把除了配置项之外的所有对象的属性都统称为属性，而对象中的键统称为属性名称，值称为属性值。
- 方法：在配置对象中，除了配置项外，值为函数的，都统称为方法。
- 字段：在 Ext JS 中，表单内的输入组件被统称为字段（field），因而，注意在本书中，字段有两个意思，一个是数据库中的字段，一个是表单的输入组件。
- 修改方法（setter）：在使用 Ext.define 定义类的时候，在 config 配置项中定义的属性，都会自动为它们创建以 set 为前缀，加上将第一个字母转换为大写后的属性名称的修改方法，如属性名称为 name，则修改方法的名称为 setName。修改方法的作用是设置属性的值。
- 访问方法（getter）：与修改方法类似，是框架为 config 配置项中定义的属性自动创建的用来获取属性值的方法，方法的名称会以 get 为前缀，再加上将第一个字母转换为大写后的属性名称。
- 更新方法（updater）：如果为属性定义了更新方法，那么在调用属性的修改方法时，会去执行更新方法。方法的名称会以 update 为前缀，再加上将第一个字母转换为大写后的属性名称。该方法会在调用修改方法后，新的属性值与旧的属性值不同时被调用。

1.5.2 一些常见的配置项

在使用 Ext JS 编程的时候，下面的配置项是经常要用到的，需要熟记。

- requires：在自定义类时，需要引用其他类的时候，就要使用该配置项，该配置项类似于 C#的 using 语句或 JAVA 的 import 语句。主要作用就是让 Ext JS 在实例化这些类时，先自动去加载 requires 中的类，以避免这些类因未被加载而造成错误。
- extend：在自定义类时，用来指定该自定义类将从哪个类扩展。
- mixins：将一个类混入要定义的类。如果查看一下 Ext.Base 的 mixin 方法，就可知道混入的作用就是把混入类的方法复制到要定义的类的原型（prototype）。
- items：用来定义当前组件的子组件。
- xtype：用来说明当前配置对象使用的是哪个类，在 API 中具体的类说明中，在类名旁边就可以查到类的 xtype。
- config：在 config 中定义的属性，框架会自动为属性生成修改方法和访问方法。其实还有其他一些方法及事件，具体可以查看 Ext.Class 源代码中的 getConfigNameMap 方法。
- id：可以为组件定义一个唯一的 id，以方便查找组件。该 id 也会作为组件最终的 HTML 代码的 id。
- itemId：作用与 id 类似，不过不会将它作为组件最终的 HTML 代码的 id，而且，该 id 的使用范围也被限制在容器内。这样做的好处就是，避免由于 id 太多，造成 id 冲突。
- scope：用来定义作用域。要了解清楚此配置项，最好参阅《JavaScript 高级程序设计（第 2 版）》。
- defaults：用来简化组件定义代码的配置项。当父组件使用了 defaults 配置项的时候，在创

建子组件的时候，会将该配置项中定义的配置应用到子组件中。例如，工具栏中有许多按钮，如果为每个按钮都设置一次 ui 配置项，那么，要对 ui 进行修改的时候，就不得不逐个地修改，这很麻烦，而使用 defaults，则只修改一次就行了。如果某个按钮需要使用不同的配置，则直接定义 ui 配置项，在创建组件时就会使用这个直接定义的 ui 配置项来替换 defaults 中定义的，从而实现了差异化。

- defaultType：与 defaults 的作用类似，不过它是用来设置子组件的默认类型的，如表单中有许多输入字段，而使用最多的是文本字段，则可在定义表单面板时使用该配置项来指定默认的子组件类型是文本字段，这样就不必为每个文字字段编写 xtype 了。
- layout：用来定义容器的布局。

1.5.3 类的命名规则

在 Ext JS 中，为了配合自动加载机制，任何类的命名都必须符合唯一的一条规则，就是能根据类名找到类的定义文件。

命名空间作为类名的起始部分，必须在应用程序未开始执行之前使用 Ext.Loader 的 setPath 方法来为命名空间指定目录，这样，Ext JS 就可根据类名中的命名空间，在指定的目录下搜索到该类的定义文件，并进行自动加载。例如，本书 SimpleCMS 实例中包含的 bootstrap.js 文件定义如下：

```
Ext.Loader.addClassPathMappings({
  "SimpleCMS.Application": "app/application.js",
  "Ext.Msg": "ext/src/window/MessageBox.js",
  "Ext": "ext/src",
  "Ext.rtl.EventObjectImpl": "ext/src/rtl/EventObject.js",
  "SimpleCMS": "app"
});
```

定义中，方法 addClassPathMappings 的作用是为命名空间指定路径。在这里，包含 Ext 和 SimpleCMS 两个命名空间，其中，Ext 是 Ext JS 框架使用的命名空间，也就是说，所有框架的类文件都可以在 ext/src 目录下找到。而 SimpleCMS 则是应用程序的命名空间，一般情况下，该目录默认是在 app 目录下，可根据自己的喜好或设计要求进行修改。

在类名的定义中，命名空间指定了根目录，中间部分的名称则是根目录下的子目录的名称，而最后的名称加上扩展名“.js”就是定义类的文件的名称，例如类 Ext.window.MessageBox，根据规则，就可以在 ext\src 目录下的 window 目录找到 MessageBox.js 文件。

这个规则是必须熟悉和遵守的，否则就会发生找不到类的情况。反过来，如果在应用程序中出现找不到类的情况，那基本上说明类的名称定义错了或者是放置的目录错了。

1.6 关于调试

在本书中，如果没有特殊说明，一般所用的浏览器为 Firefox，调试工具为 Firebug。