

Google Web Toolkit

信息科学与技术丛书 程序设计系列

吉融 编著

# GWT

## 入门与经典 实例解析



随书配套源代码下载网址  
[www.cmpbook.com](http://www.cmpbook.com)

- 如何构建 GWT Web 界面
- GWT 中的 RPC 机制
- GWT 中的 JSNI 技术
- GWT 中的 JUnit 整合
- 在 GWT 中使用 JSON



机械工业出版社  
CHINA MACHINE PRESS

信息科学与技术丛书

# GWT 入门与经典实例解析

吉 融 编著



机械工业出版社

本书全面讲解了 GWT 技术, 主要内容包括: Widget、Panel、GWT 的事件和监听器, Image Bundle 技术, GWT 的 RPC 机制, RequestBuilder, 国际化, JSNI 技术, 在 GWT 中使用 JSON 的方法, JUnit 整合, 跨越同源策略, 部署 GWT 工程等。

全书深入浅出, 循序渐进, 从如何构建 GWT Web 界面讲起, 逐步深入到 GWT 中 RPC、JSNI、JSON 等高级技术。书中对所涉及的相关组件都介绍了基础知识和基本的使用方法, 并配有相关实例, 不仅能帮助读者掌握 GWT 技术本身, 也能掌握更多相关组件的知识。本书语言通俗易懂, 实例伴有相关讲解。如果读者有 Java 语言基础, 那么阅读本书更可以事半功倍, 迅速掌握 GWT 开发 Ajax 应用程序的方法。

本书不仅适用于具有一定 Java 开发经验的开发人员, 也同样适用于大中专院校的学生。

### 图书在版编目 (CIP) 数据

GWT 入门与经典实例解析 / 吉融编著. —北京: 机械工业出版社, 2009.1

(信息科学与技术丛书)

ISBN 978-7-111-25407-2

I. G… II. 吉… III. 计算机网络—程序设计 IV. TP393.09

中国版本图书馆 CIP 数据核字 (2008) 第 162924 号

机械工业出版社 (北京市百万庄大街 22 号 邮政编码 100037)

策划编辑: 丁 诚 吴鸣飞

责任编辑: 丁 诚

责任印制: 李 妍

保定市中国画美凯印刷有限公司印刷

2009 年 1 月 · 第 1 版第 1 次印刷

184mm×260mm · 17.75 印张 · 440 千字

0001—5000 册

标准书号: ISBN 978-7-111-25407-2

定价: 32.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

销售服务热线电话: (010) 68326294 68993821

购书热线电话: (010) 88379639 88379641 88379643

编辑热线电话: (010) 88379753 88379739

封面无防伪标均为盗版

## 出版说明

随着信息科学与技术的迅速发展,人类每时每刻都会面对层出不穷的新技术、新概念。毫无疑问,在节奏越来越快的工作和生活中,人们需要通过阅读和学习大量信息丰富、具备实践指导意义的图书,来获取新知识和新技能,从而不断提高自身素质,紧跟信息化时代发展的步伐。

众所周知,在计算机硬件方面,高性价比的解决方案和新型技术的应用一直备受青睐;在软件技术方面,随着计算机软件的规模和复杂性与日俱增,软件技术受到不断挑战,人们一直在为寻求更先进的软件技术而奋斗不止。目前,计算机在社会生活中日益普及,随着因特网延伸到人类世界的层层面,掌握计算机网络技术和理论已成为大众的文化需求。也正是由于信息科学与技术在电工、电子、通信、工业控制、智能建筑、工业产品设计与制造等专业领域中已经得到充分、广泛的应用,所以这些专业领域中的研究人员和工程技术人员将越来越迫切需要汲取自身领域信息化所带来的新理念和新方法。

针对人们对了解和掌握新知识、新技能的热切期待,以及由此促成的人们对语言简洁、内容充实、融合实践经验的图书迫切需要的现状,机械工业出版社适时推出了“信息科学与技术丛书”。这套丛书涉及计算机软件、硬件、网络、工程应用等内容,注重理论与实践相结合,内容实用,层次分明,语言流畅,是信息科学与技术领域专业人员不可或缺的图书。

现今,信息科学与技术的发展可谓一日千里,机械工业出版社欢迎从事信息技术方面工作的科研人员、工程技术人员积极参与我们的工作,为推进我国的信息化建设作出贡献。

机械工业出版社

## 前言

GWT 是 Google Web Toolkit 的缩写,它是搜索引擎巨头 Google 公司发布的基于 Java 语言开发 AJAX 应用的开发工具包。使用 GWT 可以通过编写 Java 代码和传统的 HTML 页面来构造 AJAX 应用界面,并与服务器实现异步通信。

借助于 GWT,开发者可以使用 Java 代码来开发 AJAX Web 应用程序,然后通过 GWT 提供的编译器将 Java 代码转换为 JavaScript 代码,从而大大简化了复杂的 AJAX 开发工作,并且提高了 AJAX 应用程序的浏览器兼容性。由 GWT 编写的应用程序可以支持 Internet Explorer、Mozilla、Opera 和 Safari 等主流浏览器。

GWT 通过 Java 语言编写 AJAX 应用,其设计参考 Java AWT 包设计,类命名规则、接口设计、事件监听等都和 Java AWT 非常类似,使开发者在学习和使用 GWT 时有一种轻车熟路的感觉,熟悉 Java AWT 的开发者不需要花费多大的力气就能够快速地理解 GWT 开发过程。

在用户界面方面,GWT 通过 Widget 封装了大量常用的 Web 组件和常见的 AJAX 组件,如 Menu、TabBar 和 Tree 等,并通过各种 Panel 为这些 Widget 提供页面布局方式,从而降低了 AJAX 页面开发的难度,提高了开发效率。

GWT 中提供了一种 JSNI 技术,它类似于 Java 中的 JNI 技术。JSNI 技术可以在 GWT 的 Java 代码中嵌入 JavaScript 代码,因此已有的 JavaScript 组件可以被无缝地嵌入到 GWT 应用当中,使 GWT 拥有极大的扩展性。

GWT 封装了 RPC(远程过程调用)机制,开发者可以方便地使用 RPC 完成异步通信,从而提供给用户丰富的 Web 2.0 应用体验。

本书结合大量实例全面介绍了 GWT 技术,全书共分 13 章,章节安排如下:

第 1 章讲解 GWT 的基础知识。介绍了 GWT 的特点、发展历程、基础知识以及如何创建 GWT 工程,使读者对 GWT 有一个初步了解。

第 2 章讲解事件和监听器。AJAX 应用重要的一环是对复杂用户事件的处理。介绍了 GWT 如何触发浏览器事件。

第 3 章讲解如何用 Widget 构建用户界面。介绍了 GWT 中提供的大量 Widget,通过这些 Widget,开发者可以构建基本的 AJAX 用户界面。

第 4 章讲解 Image Bundle 的使用方法。在构建用户界面的工程中,可能需要大量的图片,而这些图片可能很琐碎,并可能会影响到 AJAX 应用程序的性能。介绍了如何使用 Image Bundle 技术来处理这些小图片。

第 5 章讲解如何用 Panel 实现 UI 布局。Panel 用于为各种 Widget 提供页面布局,Widget 和 Panel 的结合使用才能形成完整的用户界面。介绍了 GWT 提供的常用 Panel。

第 6 章讲解 GWT 中的 RPC 机制。RPC 是 AJAX 技术实现异步通信的主要方式之一。介绍了 GWT 中如何使用 RPC 机制。

第 7 章讲解国际化的知识。GWT 提供的国际化策略可以让程序为不同国家或地区的用户选择当地的语言浏览页面。



第8章讲解 GWT 中的 JSNI 技术。JSNI 是 GWT 中非常有用的技术之一,通过它开发者可以将 GWT 与任何现有的 JavaScript 库整合,从而使 GWT 具有很强的扩展性,并提高了开发效率。详细介绍了 GWT 中的 JSNI 技术。

第9章讲解如何将 GWT 与 JUnit 进行整合。JUnit 是 Java 领域使用的重要的单元测试技术之一。介绍了 GWT 如何与 JUnit 整合,并且介绍如何用 JUnit 进行异步测试。

第10章讲解使用 RequestBuilder 传输数据的方法。RequestBuilder 是 GWT 中另外一种客户端与服务器端通信的方式。介绍了 RequestBuilder 的使用方法。

第11章讲解如何在 GWT 中使用 JSON。JSON 是一种客户端与服务器端交换数据的方式,介绍了如何在 GWT 中使用 JSON。

第12章讲解跨越同源策略。同源策略是现代浏览器的一种安全机制,但在 Ajax 应用中并不是什么时候都需要这种安全机制,有时候需要越过它来实现某种特殊的页面功能。介绍了跨越同源策略的几种方法。

第13章全面理解 GWT。当了解了 GWT 中的全部技术之后,有必要回过头来重新理解一下 GWT 的运行机制。总体介绍了 GWT 的运行机制,并且介绍了如何将 GWT 工程部署到 Web 服务器。

本书由吉融编著,读者可以通过电子邮件的方式与作者联系、交流,电子邮件地址为 gwtbook@163.com。

由于作者水平有限,书中遗漏和不足之处在所难免,敬请广大读者批评指正。

编 者



# 目 录

## 前言

|                                              |    |
|----------------------------------------------|----|
| 第 1 章 GWT 入门 .....                           | 1  |
| 1.1 GWT 概述 .....                             | 1  |
| 1.1.1 AJAX 简介 .....                          | 1  |
| 1.1.2 GWT 简介 .....                           | 2  |
| 1.2 搭建 GWT 开发环境 .....                        | 2  |
| 1.2.1 安装 JDK .....                           | 2  |
| 1.2.2 安装 GWT .....                           | 4  |
| 1.2.3 安装 Eclipse .....                       | 4  |
| 1.3 运行 GWT 样例程序 .....                        | 4  |
| 1.3.1 主机模式下运行 GWT .....                      | 4  |
| 1.3.2 Web 模式下运行 GWT .....                    | 4  |
| 1.4 创建 GWT 工程 .....                          | 6  |
| 1.4.1 用 applicationCreator 创建一个 GWT 应用 ..... | 6  |
| 1.4.2 创建一个用于 Eclipse 的 GWT 应用 .....          | 7  |
| 1.5 初识 GWT 应用程序 .....                        | 9  |
| 1.5.1 入口点类 .....                             | 9  |
| 1.5.2 HTML 页面 .....                          | 10 |
| 1.5.3 模块 .....                               | 12 |
| 1.6 projectCreator 命令详解 .....                | 13 |
| 1.7 applicationCreator 命令详解 .....            | 14 |
| 1.8 标准 GWT 工程结构 .....                        | 14 |
| 1.9 模块 .....                                 | 15 |
| 1.10 CSS 层叠样式表 .....                         | 17 |
| 1.11 GWT 的其他基础知识 .....                       | 20 |
| 1.11.1 客户端代码 .....                           | 20 |
| 1.11.2 服务器端代码 .....                          | 20 |
| 1.11.3 GWT 的 Java 语言兼容性 .....                | 20 |
| 1.12 本章小结 .....                              | 21 |
| 第 2 章 事件和监听器 .....                           | 22 |
| 2.1 事件 .....                                 | 22 |
| 2.1.1 GWT 事件 .....                           | 22 |
| 2.1.2 浏览器的事件处理机制 .....                       | 23 |
| 2.1.3 Widget/Panel 对事件的处理 .....              | 24 |





|        |                             |    |
|--------|-----------------------------|----|
| 2.2    | 监听器                         | 28 |
| 2.3    | 本章小结                        | 30 |
| 第 3 章  | 用 Widget 构建用户界面             | 31 |
| 3.1    | 传统 Web 组件                   | 32 |
| 3.1.1  | Label 标签                    | 32 |
| 3.1.2  | HTML 组件                     | 33 |
| 3.1.3  | Button 按钮                   | 34 |
| 3.1.4  | CheckBox 复选框                | 36 |
| 3.1.5  | RadioButton 单选框             | 39 |
| 3.1.6  | TextBox 文本框                 | 39 |
| 3.1.7  | TextArea 文本区                | 41 |
| 3.1.8  | PasswordTextBox 密码文本框       | 43 |
| 3.1.9  | TextBoxBase 文本框基类           | 44 |
| 3.1.10 | ListBox 列表框                 | 48 |
| 3.1.11 | Frame 框架                    | 51 |
| 3.1.12 | Hidden 隐藏标签                 | 52 |
| 3.2    | 高级 Web 组件                   | 52 |
| 3.2.1  | CustomButton 自定义按钮          | 52 |
| 3.2.2  | PushButton 推压按钮             | 54 |
| 3.2.3  | ToggleButton 绑定按钮           | 55 |
| 3.2.4  | RichTextArea 富文本            | 56 |
| 3.2.5  | Menu 菜单                     | 62 |
| 3.2.6  | TabBar 页卡栏                  | 66 |
| 3.2.7  | Grid 网格                     | 67 |
| 3.2.8  | FlexTable 伸缩表格              | 69 |
| 3.2.9  | HTMLTable HTML 表格           | 72 |
| 3.2.10 | Tree 树                      | 75 |
| 3.2.11 | Image 图片                    | 81 |
| 3.3    | Hyperlink 超链接和 History 历史管理 | 84 |
| 3.3.1  | Hyperlink 超链接               | 84 |
| 3.3.2  | 用 History 实现历史管理            | 86 |
| 3.4    | 本章小结                        | 89 |
| 第 4 章  | Image Bundle 的使用            | 90 |
| 4.1    | Image Bundle 简介             | 90 |
| 4.2    | 在 GWT 中实现 Image Bundle      | 90 |
| 4.2.1  | 定义 ImageBundle 接口           | 90 |
| 4.2.2  | ImageBundle 接口的使用           | 92 |
| 4.3    | ImageBundle 实例              | 93 |
| 4.4    | ImageBundle 的子接口            | 94 |





|                                    |     |
|------------------------------------|-----|
| 4.5 本章小结                           | 96  |
| 第 5 章 用 Panel 实现 UI 布局             | 97  |
| 5.1 Panel 的分类                      | 98  |
| 5.2 Simple panels 单一面板             | 98  |
| 5.2.1 ScrollPanel 滚动条面板            | 98  |
| 5.2.2 FocusPanel 焦点面板              | 100 |
| 5.2.3 FormPanel 表单面板               | 102 |
| 5.2.4 PopupPanel 弹出面板              | 111 |
| 5.2.5 DialogBox 对话框                | 121 |
| 5.3 Complex panels 复合面板            | 124 |
| 5.3.1 DockPanel 方位面板               | 124 |
| 5.3.2 HorizontalPanel 水平面板         | 128 |
| 5.3.3 VerticalPanel 垂直面板           | 131 |
| 5.3.4 FlowPanel 流面板                | 133 |
| 5.3.5 StackPanel 堆面板               | 134 |
| 5.3.6 DeckPanel 层叠面板               | 136 |
| 5.3.7 AbsolutePanel 绝对值面板          | 138 |
| 5.3.8 RootPanel 根面板                | 140 |
| 5.4 Composite Panels 合成面板          | 140 |
| 5.4.1 TabPanel 页卡面板                | 140 |
| 5.4.2 DisclosurePanel 展开面板         | 144 |
| 5.4.3 SuggestBox 提示框               | 147 |
| 5.5 Split Panels 分裂面板              | 151 |
| 5.5.1 HorizontalSplitPanel 水平分割面板  | 151 |
| 5.5.2 VerticalSplitPanel 垂直分隔面板    | 152 |
| 5.6 用 Composite 实现自定义 Widget/Panel | 153 |
| 5.7 其他布局方式                         | 155 |
| 5.8 本章小结                           | 156 |
| 第 6 章 GWT 中的 RPC 机制                | 157 |
| 6.1 服务接口                           | 158 |
| 6.2 服务实现类                          | 159 |
| 6.3 异步接口                           | 162 |
| 6.4 调用 RPC                         | 163 |
| 6.5 串行化                            | 173 |
| 6.6 本章小结                           | 175 |
| 第 7 章 国际化                          | 176 |
| 7.1 通过 Constants 实现国际化             | 176 |
| 7.2 通过 Messages 实现国际化              | 185 |
| 7.3 通过 ConstantsWithLookup 实现国际化   | 190 |



|        |                                           |     |
|--------|-------------------------------------------|-----|
| 7.4    | 通过 Dictionary 实现国际化                       | 190 |
| 7.5    | 静态本地化和动态本地化                               | 191 |
| 7.5.1  | 静态本地化                                     | 191 |
| 7.5.2  | 动态本地化                                     | 192 |
| 7.6    | i18nCreator 命令详解                          | 192 |
| 7.7    | 用 NumberFormat 国际化数值                      | 193 |
| 7.8    | 用 DateTimeFormat 国际化日期和时间                 | 195 |
| 7.9    | Image Bundles 和本地化                        | 201 |
| 7.10   | 本章小结                                      | 206 |
| 第 8 章  | GWT 中的 JSNI 技术                            | 207 |
| 8.1    | JSNI 简介                                   | 207 |
| 8.2    | 编写 JSNI 方法                                | 208 |
| 8.3    | 在 JavaScript 中访问 Java 域                   | 208 |
| 8.4    | 在 JavaScript 中调用 Java 方法                  | 208 |
| 8.5    | 在 Java 和 JavaScript 间传递对象                 | 211 |
| 8.6    | JSNI 与异常                                  | 213 |
| 8.7    | 封装 JavaScript 库                           | 213 |
| 8.7.1  | 在 GWT 中嵌入 script.aculo.us 的 effects 组件    | 213 |
| 8.7.2  | 在 GWT 中嵌入 script.aculo.us 的 dragdrop 拖放组件 | 216 |
| 8.7.3  | 在 GWT 中嵌入 ActiveWidgets 的 grid 组件         | 218 |
| 8.8    | 本章小结                                      | 224 |
| 第 9 章  | GWT 与 JUnit 整合                            | 225 |
| 9.1    | JUnit 基础                                  | 225 |
| 9.1.1  | 安装 JUnit                                  | 225 |
| 9.1.2  | 编写测试用例                                    | 225 |
| 9.1.3  | 运行 JUnit 测试                               | 226 |
| 9.2    | GWT 与 JUnit 整合                            | 226 |
| 9.2.1  | 建立测试类                                     | 227 |
| 9.2.2  | junitCreator 详解                           | 227 |
| 9.2.3  | 异步测试                                      | 229 |
| 9.3    | 本章小结                                      | 230 |
| 第 10 章 | 使用 RequestBuilder 传输数据                    | 231 |
| 10.1   | Get 方法和 Post 方法                           | 231 |
| 10.1.1 | URL 简介                                    | 231 |
| 10.1.2 | Get 方法和 Post 方法                           | 232 |
| 10.2   | 用 RequestBuilder 实现远程调用                   | 232 |
| 10.3   | 用 RequestBuilder 读取 XML 数据                | 236 |
| 10.4   | 本章小结                                      | 242 |
| 第 11 章 | 在 GWT 中使用 JSON                            | 243 |



|        |                                                   |     |
|--------|---------------------------------------------------|-----|
| 11.1   | JSON 基础                                           | 243 |
| 11.2   | 在 GWT 中使用 JSON                                    | 245 |
| 11.2.1 | 通过 JSONParser 解析 JSON 数据                          | 245 |
| 11.2.2 | JSONValue 对象                                      | 245 |
| 11.2.3 | JSONString, JSONBoolean, JSONNumber 和 JSONNull 对象 | 246 |
| 11.2.4 | JSONArray 对象                                      | 246 |
| 11.2.5 | JSONObject 对象                                     | 247 |
| 11.3   | JSON 实例                                           | 248 |
| 11.4   | 本章小结                                              | 253 |
| 第 12 章 | 跨越同源策略                                            | 254 |
| 12.1   | 同源策略简介                                            | 254 |
| 12.2   | 服务器端代理                                            | 254 |
| 12.3   | 通过 Google AJAX Feed API 实现跨域访问                    | 258 |
| 12.3.1 | Google AJAX Feed API 简介                           | 258 |
| 12.3.2 | Google AJAX Feed API 的使用方法                        | 258 |
| 12.3.3 | 用 Google AJAX Feed API 读取种子                       | 260 |
| 12.4   | 本章小结                                              | 263 |
| 第 13 章 | 全面理解 GWT                                          | 264 |
| 13.1   | GWT 工程文件详解                                        | 264 |
| 13.2   | GWT 工程的启动机制                                       | 267 |
| 13.2.1 | 标准应用                                              | 268 |
| 13.2.2 | 跨域应用                                              | 269 |
| 13.3   | 部署 GWT 工程                                         | 269 |
| 13.3.1 | 删除不需要的文件                                          | 269 |
| 13.3.2 | 将 GWT 工程部署到服务器                                    | 271 |
| 13.4   | 本章小结                                              | 273 |
| 附录     | keyCode 键码值表                                      | 274 |



# 第 1 章 GWT 入门

## 1.1 GWT 概述

### 1.1.1 AJAX 简介

AJAX 全称为 Asynchronous JavaScript and XML (异步 JavaScript 和 XML), 是指一种创建交互式网页应用的网页开发技术。

传统的 Web 应用往往是一个“提交—等待—重新显示”的过程。比如, 用户填写表单 (form), 当提交表单时就向 Web 服务器发送一个请求, 服务器接收并处理传来的表单, 然后返回一个新的网页。这种做法浪费了许多带宽, 因为在前后两个页面中的大部分 HTML 代码往往是相同的。由于每次应用的交互都需要向服务器发送请求, 应用的响应时间就依赖于服务器的响应时间。这导致了用户界面的响应比本地应用慢得多, 并且给用户一种停停走走的感觉, 极大地影响了用户体验。

与此不同, AJAX 应用可以仅向服务器发送并取回必需的数据, 并在客户端采用 JavaScript 处理来自服务器的响应。因为在服务器和浏览器之间交换的数据大量减少, 结果我们就能看到响应更快的应用。同时很多的处理工作可以在发出请求的客户端上完成, 所以 Web 服务器的处理时间也减少了。

AJAX 应用程序的优势表现在以下方面。

- 1) 通过异步模式, 提升了用户体验, 可以达到类似桌面应用程序的效果。
- 2) 优化了浏览器和服务器之间的传输, 减少不必要的数据往返, 减少了带宽占用。
- 3) AJAX 引擎在客户端运行, 承担了一部分本来由服务器承担的工作, 从而减少了大用户量下的服务器负载。

2005 年初, 许多事件使得 AJAX 被大众所接受。Google 在它著名的交互应用程序中使用了异步通信, 如大名鼎鼎的 Google Map、Google Groups、Gmail 等。自此不仅 Google 公司, 小到个人网站, 大到门户网站都或多或少地使用了 AJAX 技术, 互联网世界迅速掀起了一场 Web 2.0 风暴。这时还出现了大量的用 AJAX 技术开发的纯 Web 2.0 网站, 如 Flickr、YouTube、Digg 等。这些 Web 2.0 网站在给用户带来全新的用户体验的同时, 也创造了丰厚的利润, 一时间, Web 2.0 成为了互联网产业的一个新的金矿。

AJAX 不是指一种单一的技术, 而是有机地利用了一系列相关的技术。这些技术包括:

- 基于 Web 标准 (Standards Based Presentation) XHTML+CSS 的表示;
- 使用 DOM (Document Object Model) 进行动态显示及交互;
- 使用 XML 和 XSLT 进行数据交换及相关操作;
- 使用 XMLHttpRequest 进行异步数据查询、检索;



使用 JavaScript 作为 AJAX 应用程序的核心代码,帮助改进与服务器应用程序的通信。

AJAX 作为一种复合技术,由多种技术组成,这给开发带来了极大难度,尤其是要求开发人员必须掌握以上全部技术才可能开发出优秀的 AJAX 程序。于是,就像从低级语言发展到高级语言一样,面对巨大的市场前景,各软件公司开始对 AJAX 技术进行二次开发,封装 AJAX 的底层技术,开发出新的框架来简化 AJAX 开发过程。比如微软公司推出了 Ajax.net 平台,与微软体系对应的 Java 体系,由于 Java 的开源特性,涌现出了多种 AJAX 开发技术,如 Dojo、Backbase、GWT 等。其中,搜索引擎巨头 Google 公司开发的 GWT 技术自发布以来就在业界引起了强烈反响。GWT 可以使程序员像编写普通的 Java 代码一样编写 AJAX 程序,极大地简化了 AJAX 程序开发过程。

## 1.1.2 GWT 简介

GWT 是 Google Web Toolkit 的简写,它是搜索引擎巨头 Google 公司发布的基于 Java 语言开发 AJAX 应用的开发工具包。使用 GWT 可以通过编写 Java 代码和传统的 HTML 页面来构造 AJAX 应用界面,并与服务器实现异步通信。

开发者可以使用任何 Java 开发工具来编写基于 GWT 的 Java Web 应用。GWT 自带的编译器可以将 Web 客户端的 Java 代码编译成 JavaScript 代码,这些 JavaScript 代码有很好的浏览器兼容性,可以支持 Internet Explorer、Mozilla、Opera、Safari 等主流浏览器。

GWT 的设计参考 Java AWT 包设计,类命名规则、接口设计、事件监听等都和 AWT 非常类似。熟悉 Java AWT 的开发者不需要花费多大的力气就能够快速地理解 GWT 开发过程。

GWT 通过 Widget 提供了各种传统 HTML 组件,如 Button 按钮、ListBox 下拉列表框等;在此基础上又通过 Widget 封装了很多桌面组件,如多级菜单、树形结构等。通常,在 Web 中实现这些组件需要编写大量的 JavaScript 和 CSS 代码,借助 GWT 提供的 Widget,开发者可以像使用 Button 按钮一样轻易地使用这些组件;另外,GWT 通过 Panel 提供了布局管理器和一些常用的布局,Panel 就像 Java 桌面应用中的布局管理器一样,可以管理各组件的布局;GWT 还提供了各种事件监听器,以处理用户触发的各种 Web 事件。通过 Widget、Panel 和监听器的结合使用,开发者可以像编写普通的 Java 桌面程序一样编写 AJAX 应用。

GWT 封装了 RPC (远程过程调用) 机制,开发者可以方便地使用 RPC 完成异步通信,从而提供给用户丰富的 Web 2.0 应用体验。

## 1.2 搭建 GWT 开发环境

工欲善其事,必先利其器。开发任何应用程序都要搭建一个方便好用的开发环境,这一节我们将详细讲述 GWT 开发环境的搭建过程。

### 1.2.1 安装 JDK

安装 JDK 的过程非常简单,只需从 Sun 公司的网站 <http://java.sun.com/javase/downloads/index.jsp> 下载需要的 JDK 版本,下载完成后运行此安装程序,然后按照安装向导一步一步地操作即可完成安装。需要注意的是,GWT 目前只支持 1.4.2 版本以下的 JDK。本书使用的是 1.4.2\_13 版的 JDK。



JDK 安装完成后需要配置 JDK 的环境变量，包括以下两项设置：一是将 JAVA\_HOME 变量设置为 JDK 的根目录；二是将 JDK 可执行文件的路径添加到 Path 中。

右键单击“我的电脑”，选择“属性”，在随后弹出的“系统属性”对话框中选择“高级”选项卡，单击“环境变量”按钮，便弹出“环境变量”对话框，如图 1-1 所示。

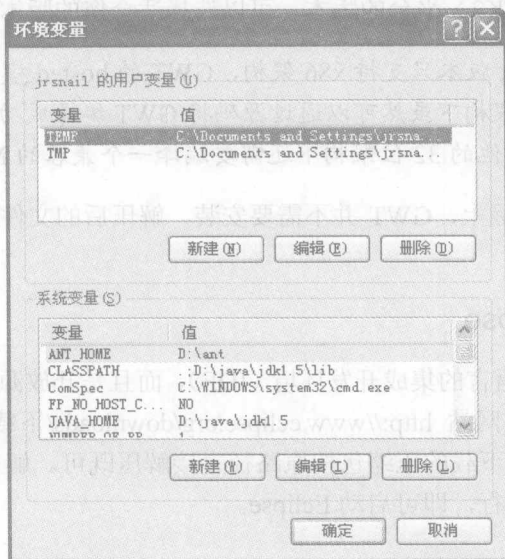


图 1-1 “环境变量”对话框

“环境变量”对话框包括“用户变量”和“系统变量”两部分，前者定义的环境变量只对当前用户起作用，而后者定义的环境变量对所有用户都起作用。这里，我们以系统变量为例。

单击“新建”按钮，在弹出的“新建系统变量”对话框中添加“JAVA\_HOME”变量，并将其值设为 JDK 的安装路径，如图 1-2 所示。

单击“确认”按钮完成 JAVA\_HOME 变量的添加。之后修改 Path 变量。在“环境变量”对话框的“系统变量”列表中找到“Path”项，双击或选中后单击“编辑”按钮，弹出“编辑系统变量”对话框，在“变量值”文本框中添加“%JAVA\_HOME%\bin;”，注意各变量值之间以分号分隔，如图 1-3 所示。

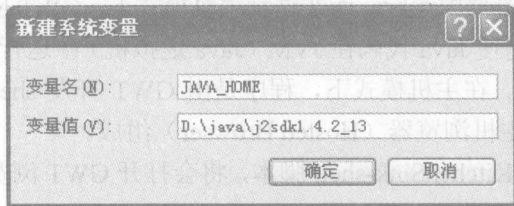


图 1-2 “新建系统变量”对话框

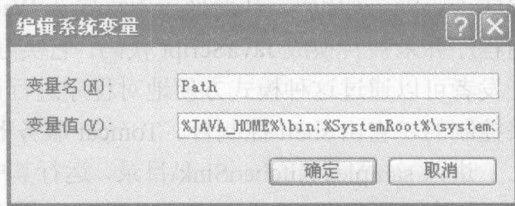


图 1-3 “编辑系统变量”对话框

单击“确认”按钮完成 Path 变量的修改并返回“环境变量”对话框，单击“确认”，完成环境变量的设置和修改工作。





## 1.2.2 安装 GWT

首先到 GWT 的官方网站 <http://code.google.com/webtoolkit/download.html> 下载最新版本的 GWT 开发包,本书所使用的是 gwt-windows-1.4.61 版。官方网站上目前提供 Windows、Mac OS X 10.4 和 Linux (i386 GTK2) 平台的版本,可以选择适合您的版本下载。

注意: GWT 的 Linux 版本只支持 x86 架构, GWT 的 hosted 模式需要 32 位的 JVM 虚拟机和库,在其他的 32 位架构下虽然可以通过源码将 GWT 编译成功,但实际上 GWT 是不支持这些架构的,而且在其他的 32 位架构下还需要编译一个兼容的 Mozilla 1.7.12 版浏览器。

然后解压 GWT。实际上, GWT 并不需要安装,解压后的文件夹中包含了开发 GWT 应用程序所需的全部组件。

## 1.2.3 安装 Eclipse

Eclipse 是一个 Java 语言的集成开发环境 (IDE),而且是开放源代码的项目,目前已经得到广泛使用。可以到官方网站 <http://www.eclipse.org/downloads/> 下载最新版本,安装 Eclipse 的步骤非常简单,只需将下载的压缩包按原路径直接解压既可。解压缩后,到解压路径中找到 Eclipse.exe 文件双击运行,即可启动 Eclipse。

## 1.3 运行 GWT 样例程序

搭建完 GWT 的开发环境之后,我们可以试着运行 GWT 的一些样例程序,以对 GWT 的使用过程有一个初步印象。

在 GWT 的根目录下有一个名为 “samples” 的文件夹,这就是 GWT 自带的样例程序文件夹,里面存放着若干由 GWT 开发人员编写的经典小程序,可以帮助初学者了解 GWT 的功能。下面我们以 “Kitchen Sink” 这个样例为例来看看如何运行这些样例程序。

### 1.3.1 主机模式下运行 GWT

顾名思义,主机模式 (Hosted Mode),就是不通过 Web 而在本地运行。前面介绍过, GWT 之所以可以通过 Java 代码开发 Web 界面,是因为它可以通过 GWT 编译器将这些 Java 代码转换成 JavaScript 代码,从而使应用程序在 Web 浏览器中运行。而在这种主机模式下, GWT 应用程序并未被转换成 JavaScript 代码,它实际上是使 Java 代码在 JVM (Java 虚拟机) 中运行,开发者可以通过这种模式方便地对程序进行调试。在主机模式下,程序是由 GWT 开发 shell 来控制的,它由 shell 控制台、Tomcat 服务器、主机浏览器 (Hosted Browser) 组成。

进入 samples\KitchenSink\ 目录,运行其中的 KitchenSink-shell 脚本,将会打开 GWT 浏览器,并在里面运行 Kitchen Sink 应用,如图 1-4 所示。

### 1.3.2 Web 模式下运行 GWT

Web 模式 (Web Mode) 就是通过通用 Web 浏览器来运行 GWT 应用程序。在这种模式下, Java 代码已经被转换成 JavaScript 代码,通过这种模式,可以看到此程序在实际应用中的效果。

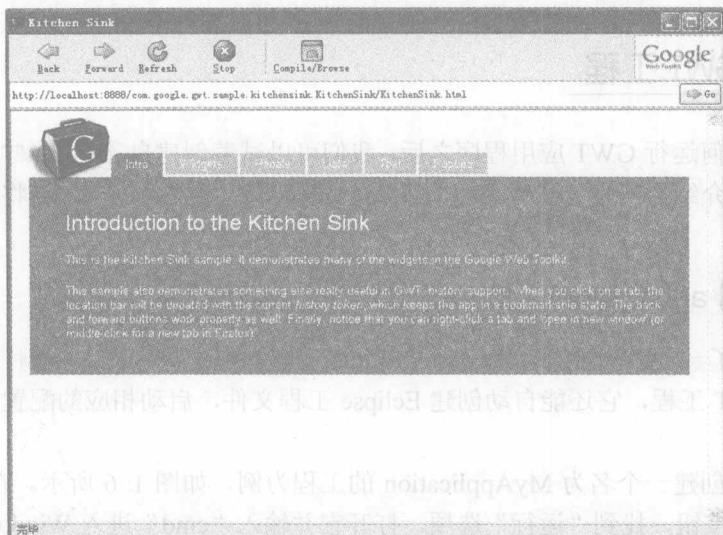


图 1-4 “Kitchen Sink” 样例主机模式效果图

有以下两种方式可以使程序以 Web 模式运行。

第一种方式相当于手动方式。首先用 samples\KitchenSink\目录下的 KitchenSink-compile 脚本对应用程序进行编译, GWT 编译器会根据 KitchenSink 的 Java 源代码产生一些 JavaScript 和 HTML 文件, 放到 samples\KitchenSink\下的 www 目录中。然后用浏览器打开 www\com.google.gwt.sample.kitchensink.KitchenSink\KitchenSink.html 文件即可用 Web 模式运行此应用程序。

第二种方式相当于自动方式。如果正在以主机模式运行此程序, 那么可以直接单击主机浏览器上的“Compile/Browse”按钮, GWT 会自动编译工程并用通用 Web 浏览器打开 KitchenSink.html 文件。

Kitchen Sink 样例在 Web 模式下的运行效果如图 1-5 所示。

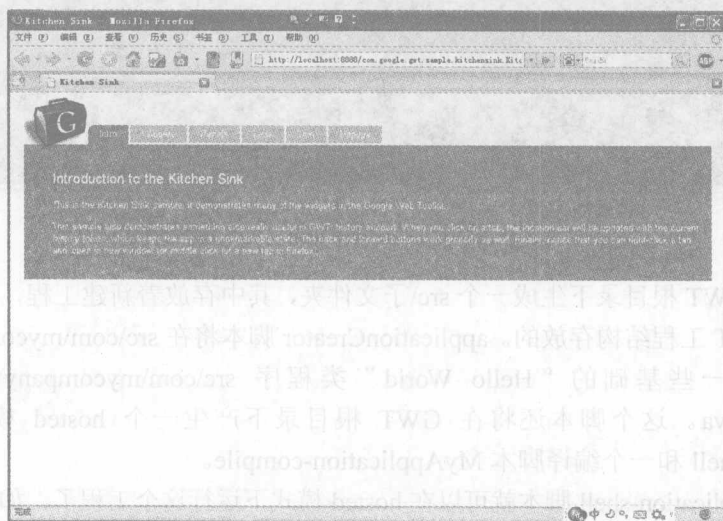


图 1-5 “Kitchen Sink” 样例 Web 模式效果图





## 1.4 创建 GWT 工程

在了解了如何运行 GWT 应用程序之后，我们可以试着创建自己的 GWT 应用程序了，在这一节，我们将介绍如何用 GWT 命令行工具创建 GWT 工程，以及如何将这个工程添加到 Eclipse 中。

### 1.4.1 用 applicationCreator 创建一个 GWT 应用

GWT 提供了一个命令行工具——applicationCreator，位于 GWT 的根目录下。通过它可以生成一个 GWT 工程，它还能自动创建 Eclipse 工程文件，启动相应的配置文件，便于主机模式下的调试。

这里我们以创建一个名为 MyApplication 的工程为例，如图 1-6 所示。在 Windows 系统中单击“开始”按钮，找到“运行”选项，打开它并输入“cmd”进入 Windows 终端控制台，在控制台中进入 GWT 的根目录，通过以下命令创建一个新工程：

```
applicationCreator com.mycompany.myproject.client.MyApplication
```

注意：根据 GWT 的标准工程结构，GWT 应用类应该建立在 client 子文件夹中。

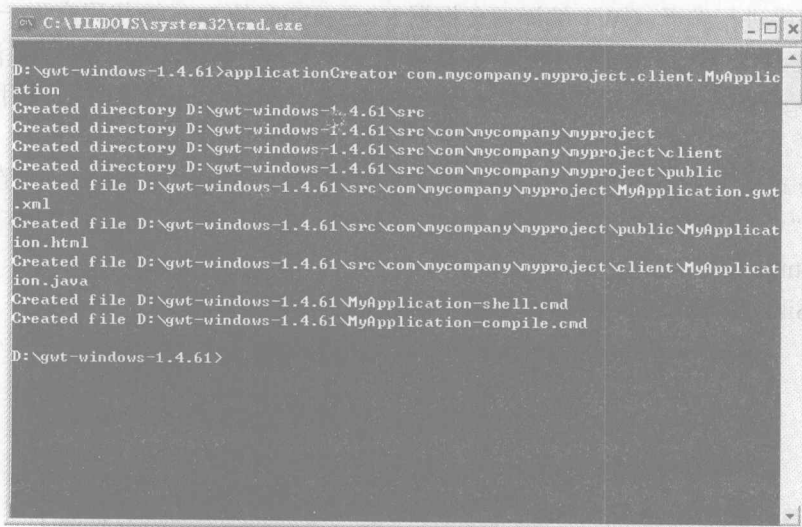


图 1-6 用 applicationCreator 创建一个 GWT 应用原型

这时会在 GWT 根目录下生成一个 src\子文件夹，其中存放着新建工程，并且这个工程是按照标准的 GWT 工程结构存放的。applicationCreator 脚本将在 src\com\mycompany\下创建一些文件，包括一些基础的“Hello World”类程序 src\com\mycompany\myproject\client\MyApplication.java。这个脚本还将在 GWT 根目录下产生一个 hosted 模式的启动脚本 MyApplication-shell 和一个编译脚本 MyApplication-compile。

运行 MyApplication-shell 脚本就可以在 hosted 模式下运行这个工程了，如图 1-7 所示。通过修改 src\com\mycompany\myproject\client\MyApplication.java 和 src\com\mycompany\myproject\