DWZ\_SPRINGMVC架构标准规范

目录

[概述 3](#_Toc375510880)

[UI规范 3](#_Toc375510881)

[界面规范总体原则 3](#_Toc375510882)

[采用W3C标准规范Html,css代码目的 3](#_Toc375510883)

[UI组件化的思路 3](#_Toc375510884)

[三层系统架构 5](#_Toc375510885)

[表示层 5](#_Toc375510886)

[业务层 5](#_Toc375510887)

[持久化层 6](#_Toc375510888)

[服务器架构 8](#_Toc375510889)

[开发流程 9](#_Toc375510890)

[需求分析阶段 9](#_Toc375510891)

[设计阶段 9](#_Toc375510892)

[编码阶段 10](#_Toc375510893)

[测试流程 11](#_Toc375510894)

[单元测试 11](#_Toc375510895)

[集成测试 11](#_Toc375510896)

[功能测试 12](#_Toc375510897)

[压力测试 12](#_Toc375510898)

[测试人员的职责： 12](#_Toc375510899)

[版本控制 13](#_Toc375510900)

[版本控制基本流程： 13](#_Toc375510901)

[项目实施版本控制分为4个步骤： 13](#_Toc375510902)

[SVN版本库布局 13](#_Toc375510903)

[分支与标记 14](#_Toc375510904)

[Merge 14](#_Toc375510905)

[其它注意事项 15](#_Toc375510906)

[文件的删除、移动与重命名 15](#_Toc375510907)

[提交的时机 16](#_Toc375510908)

[数据备份 17](#_Toc375510909)

[Mysql备份策略大概可分为4种： 17](#_Toc375510910)

[数据备份分为三步： 17](#_Toc375510911)

[Crontab 定时任务管理 17](#_Toc375510912)

[常用工具 18](#_Toc375510913)

[Java Profiler 18](#_Toc375510914)

[Firebug 18](#_Toc375510915)

[Tamper Data 19](#_Toc375510916)

[Selenium IDE 20](#_Toc375510917)

[Html Validator 21](#_Toc375510918)

[Loadrunner 22](#_Toc375510919)

[ApacheBench - Apache 专用的压力测试工具 22](#_Toc375510920)

[Alexa Sparky 22](#_Toc375510921)

[SearchStatus 23](#_Toc375510922)

[IETester 23](#_Toc375510923)

[IE Developer Toolbar 24](#_Toc375510924)

[Office 中带的脚本调试器 24](#_Toc375510925)

[附录 26](#_Toc375510926)

[附录一 (mysqlFullBackup.sh) 26](#_Toc375510927)

[附录二 (ftpDownload.sh) 27](#_Toc375510928)

[附录三(dbPutToFtp.sh) 28](#_Toc375510929)

# 概述

DWZ\_SPRINGMVC框架是基于MyBatis+Spring3+Sitemesh模板基础上封装的一套java快速开发框架。采用Spring3自动扫描，简化了大量的java配置文件，提高开发效率。

DWZ\_SPRINGMVC框架采用Apache2.0开源协议，源码下载地址：

<https://code.csdn.net/dwzteam/dwz_springmvc>

**DWZ\_SPRINGMVC设计思想简单实用、扩展方便、灵活**

**联系方式**

DWZ微博 <http://weibo.com/dwzui>

DWZ博客<http://www.cnblogs.com/dwzjs>

# UI规范

## 界面规范总体原则

1. 以用户为中心。设计由用户控制的界面，而不是界面控制用户。
2. 简单且美观实用。界面应该很简单（不是过分单纯化）、易于学习、并且易于使用。每个页面简单明了，用户一看就能大概明白如何操作，不要有太多干忧用户视线的东西。
3. 清楚一致的设计。界面的风格保持一致，具有相同含义的术语保持一致，且易于理解。
4. 较快的响应速度。
5. 遵循W3C标准，提高界面灵活度，降低界面和程序的耦合度。
6. UI组件化。组件分为功能组件和布局组件。
7. 规范化CSS样式文件。将CSS文件划分为CSS文件和子模块CSS文件。核心CSS中定义系统核心样式和公共样式。子模块CSS在核心CSS的基础上进行扩展，可以调整样式风格，并定义本模块特有的CSS。

## 采用W3C标准规范Html,css代码目的

1. 改变页面布局和样式时只修改css,而不用改任何html代码就能实现。
2. 把各种界面布局和样式相关的，都由css来处理。（比如一个列表显示，可能是一列，二列，n列，程序不做任何控制，只改css）
3. 解决主流浏览器兼容问题.
4. 降低界面和程序的耦合度。

## UI组件化的思路

**UI组件分为2大类:**

1. UI环境 (UI整体修饰, 这个思路和windows XP 系统theme主题风格相似.当禁用theme时,windows XP 界面就和windows 2000差不多, 没有圆角和渐变色等)
2. UI组件 (表单, 导航条, 对话框, 提示框, 信息列表, tab等)

**组件化思路:**

1. 最大限度的减化html代码,html代码只基本的信息，然后css 和jquery去修饰页面. 因为html与后台程序关联相对较多，改界面时最好是不用修改html代码就能把界面风格换了．这样对后台程序没有影响.
2. 构建功能型表单

<form method="post" action="/login" **class="required-validate" onsubmit="return validateCallback(this, loginDone);"**>

<fieldset>

<legend>Login</legend>

<p>

<label>Email</label>

<input type="text" name="username" class="required email"/>

</p>

<p>

<label>Password</label>

<input type="password" name="password" class="required"/>

</p>

<p><button type="submit">submit</button></p>

</fieldset>

</form>

* 整理jquery表单验证和ajax表单提交通用方法, 简化前台代码.表单验证只需在form上加**class="required-validate"** ajax 提交只需在form上加**onsubmit="return validateCallback(this, loginDone);**
* 表单字段一般都有一个label和一个input, label上自动加for属性,点激label后相应的input获得焦点.
* 表单输入框获得焦点和鼠标经过样式修饰.

1. 列表信息隔行加背景色和hover加亮.
2. 在工作过程中进一些完善界面组件库. 整理一套完整的组件, 比如表单、导航条、对话框、提示框、信息列表、tab等等.
3. 把以上提到的做成一个通用的js库. 以后可以在其它项目中引入这个js库.

# 三层系统架构

## 表示层

Jsp 提供与用户交互的界面. Jsp页面不写java代码。Jsp页面只有标签（html标签，jsp标签，自定义标签）

Controller 根据用户指令调用业务层的接口, 并将数据传递给业务层, 操作完成后把数据返回给用户.

无需配制Controller，由Spring自动扫描

/dwz\_springmvc/WebContent/WEB-INF/web-app-servlet.xml

<!-- 扫描web包，应用Spring的注解 @Controller -->

<context:component-scan base-package=*"dwz.web"* />

## 业务层

在Spring的基础上封装了一个BusinessFactory. BusinessFactory 可得到BusinessObjectServiceMgr 和 Configuration.一个BusinessObjectServiceMgr管理一个或多个Business Object.

BusinessObjectServiceMgr是业务的核心,业务逻辑基本都在BusinessObjectServiceMgr中.

AppConfiguration是一个配制信息的封装,支持xml和properties文件.需要给系统增加一个配制文件时,只需在 app-config.xml中加入路径即可,无需再写java代码去读配制文件了.

**整合Spring**

1. 在WEB-INF/web.xml加入一个监听器AppContextInitListener. 负责在系统启动时初始化Spring和载入配制信息.

**public** **void** contextInitialized(ServletContextEvent event) {

ServletContext context = event.getServletContext();

String configFile = context.getInitParameter(*ETC\_FILE*);

**try** {

AppConfiguration.*init*(context.getRealPath(configFile));

} **catch** (ConfigurationException e) {

event.getServletContext().log("ConfigurationException: ", e);

}

}

1. 在Spring配制文件中加入各模块的BusinessObjectServiceMgr

/dwz\_springmvc/src/application-context.xml

<!-- 扫描业务层service实现类 -->

<context:component-scan base-package=*"dwz.\*\*.impl"* />

1. 可以从BusinessFactory获取BusinessObjectServiceMgr，在Controller也可以使用注入的方式

BusinessFactory.*getInstance*().getService("userServiceMgr");

**public** **class** UserController **extends** BaseController{

@Autowired

**private** UserServiceMgr userMgr;

}

**简化Spring 配制**

BusinessObject由BusinessObjectServiceMgr 来管理.业务逻辑都从BusinessFactory开始,先取取到某个BusinessObjectServiceMgr,再由BusinessObjectServiceMgr来操作业务对象.

## 持久化层

采用MyBatis实现DAO的设计模式封装持久化层。可以指定不同的session factory, DAO封装可以支持多个数据库.不需要写具体的DAO实现类.由Spring AOP 产生DAO实现,程序员开发时不需要写DAO实现类了.

**Spring DAO 配制示例**

<bean class=*"org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"*>

<property name=*"ignoreResourceNotFound"* value=*"true"* />

<property name=*"locations"* value=*"classpath\*:/jdbc.properties"*/>

</bean>

<bean id=*"dataSource"* class=*"org.apache.commons.dbcp.BasicDataSource"* destroy-method=*"close"*>

<!-- Connection Info -->

<property name=*"driverClassName"* value=*"${jdbc.driverClassName}"* />

<property name=*"url"* value=*"${jdbc.url}"* />

<property name=*"username"* value=*"${jdbc.username}"* />

<property name=*"password"* value=*"${jdbc.password}"* />

<!-- Connection Pooling Info -->

<property name=*"maxActive"* value=*"${dbcp.maxActive}"* />

<property name=*"maxIdle"* value=*"${dbcp.maxIdle}"* />

<property name=*"defaultAutoCommit"* value=*"false"* />

<property name=*"timeBetweenEvictionRunsMillis"* value=*"3600000"*/>

<property name=*"minEvictableIdleTimeMillis"* value=*"3600000"*/>

</bean>

<!-- enable autowire -->

<context:annotation-config />

<!--MyBatis integration with Spring as define sqlSessionFactory -->

<bean id=*"sqlSessionFactory"* class=*"org.mybatis.spring.SqlSessionFactoryBean"*>

<property name=*"dataSource"* ref=*"dataSource"* />

<property name=*"configLocation"* value=*"classpath:mybatis-configuration.xml"*/>

<!-- <property name="mapperLocations" value="classpath\*:cn/com/ecointel/roomyi/authority/persistence/\*.xml" /> -->

</bean>

<bean id=*"mapperScanner"* class=*"org.mybatis.spring.mapper.MapperScannerConfigurer"*>

<property name=*"basePackage"* value=*"dwz.\*\*.persistence"* />

<property name=*"sqlSessionFactory"* ref=*"sqlSessionFactory"* />

</bean>

<!-- transaction manager, use JtaTransactionManager for global tx -->

<bean id=*"transactionManager"* class=*"org.springframework.jdbc.datasource.DataSourceTransactionManager"*>

<property name=*"dataSource"* ref=*"dataSource"* />

</bean>

<tx:annotation-driven transaction-manager=*"transactionManager"* />

# 服务器架构

一般来说刚开始规模不大，服务器压力不大，大多数情况下刚开始就1台或2台服务器，相对来说服务器架构也没有什么特殊要求。不过从长远考虑需要考虑服务器架构规模可扩展性。

下面介绍一下负载均衡主要有2个作用:

1. 分担压力
2. 容灾解决方案

多台服务器做负载均衡主要问题是解决各服务器之间的session共享问题. 可能需要把session存到内存数据库,hsqldb, mysql等. 自己实现一套session管理机制.

负载均衡网络架构规化:

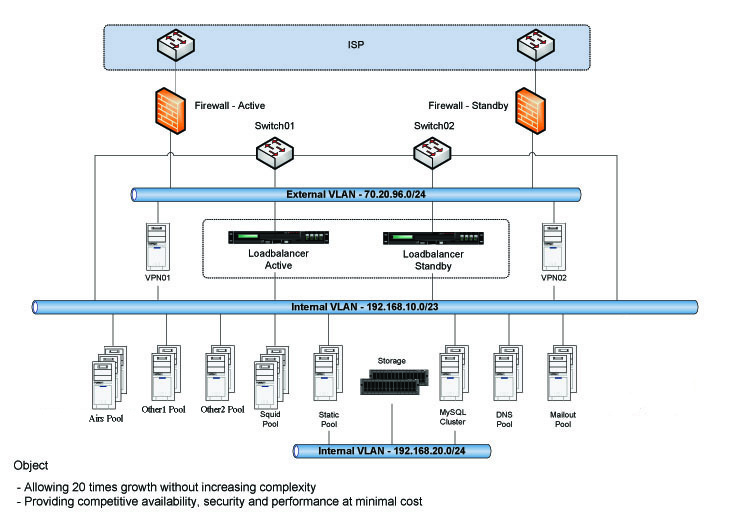


图3-1 网络架构

从服务器功能来分可分为几种：

1. Static 是静态服务器，负责所有静态文件的存和取. 和生成报表.
2. 前台展示和业务处理服务器.
3. 系统管理服务器

所有从客户端发过来的请求，全部交给Load Balance来处理．Load Balance来决定将请求发给哪台服务器．

# 开发流程

## 需求分析阶段

需求分析阶段在对需求调研和分析后，出来需求文档和用户界面UI。

**需求文档**必须能明确的描述业务流程，画出业务流程图。其目的是：

1. 有助于每个项目组成员能很快的理解需求。
2. 有助于和用户达成一致，保障项目能满足需求。

**用户界面UI**

所的业务都是为了满足用户需求。用户并不关心系统架构和服务器架构等，用户只是关软件是否好用，是否满足需求。

用户直接和UI打交道，UI对用户来说是非常重要的。而UI又直接软件的设计。

建议在这个阶段能把UI做出来（尽量做和最终的UI流程和风格一致）。很快把UI做出来让用户看，有什么反馈有意见可以马上修改。

## 设计阶段

软件设计原理: KISS(Simple but not simpler简单实用)。体系结构, 研发过程始终短小精悍。

**设计目标:**

1. 所有设计力求简单, 但不过于简化 (Simple but not simpler).
2. 从基本结构到多次优化.
3. 系统易于理解和维护, 必要的文档说明.
4. 优美的设计通常简单而不繁琐.
5. 不等于快速草率(Not quick and dirty).

**产品质量的各个方面**：

从工程设计角度：

1. 一致性（是指在有一组函数时，比如说一个接口的方法，对方法名字和参数的设计应该具有一致性。项目组各成员写出来的代码风格基本一致。遵循一致性设计的程序，会大大增加程序的可读性与编程效率。）
2. 可靠性
3. 性能
4. 可维护性
5. 功能可扩展性
6. 规模可扩展性

从用户界面角度：

1. 方便
2. 效用
3. 吸引力
4. 满意程度

**设计阶段的主要工作**

1. 定制任务优先级和时间表。
2. 定义接口, 业务层接口定义完后开会review一下。
3. 写概要设计文档：重点在于将模块分解为对象并阐明对象之间的关系。
4. 写详细设计文档：重点在于对模块进行实现，将模块的对象分解为属性和方法，并阐述如何实现。

## 编码阶段

**编码阶段的主要工作**

1. 编写接口实现类
2. 编写单元测试
3. 编写显示层代码

# 测试流程

## 单元测试

单元测试是开发者编写的一小段代码，用于检验被测代码的一个很小的、很明确的功能是否正确。通常而言，一个单元测试是用于判断某个特定条件（或者场景）下某个特定函数的行为.

用junit4做单元测试。创建一个命名为test的source forder，与src中的包名一一对应。在test中放的是src中业务对象测试类，test中的包名和src中的保持一致。

## 集成测试

集成测试的目的是确保各单元组合在一起后能够按既定意图协作运行，并确保增量的行为正确。它所测试的内容包括单元间的接口以及集成后的功能。使用黑盒测试方法测试集成的功能。并且对以前的集成进行回归测试。

junit 测试套件功能比较强大, 可以把整个项目的全部测试用例串起来. 也可以单独跑每个模块的测试套件.

添加junit 测试套件步骤:

1. 每个模块加一个测试套件Test Suite，把模块中的各junit测试类串起来。
2. 加一个总的测试套件，把各模块的测试套件串起来

测试套件示例:

@RunWith(Suite.**class**)

@SuiteClasses( { TestSuite.**class,** TestSuite2.**class,** TestSuite3.**class** })

**public** **class** AllTestSuite **extends** TestCase {

@BeforeClass

**public** **static** **void** init() {

}

}

@RunWith(Suite.**class**)

@SuiteClasses( { UserTest.**class,** ProductTest**.class** })

**public** **class** TestSuite **extends** TestCase {

@BeforeClass

**public** **static** **void** init() {

}

}

测试类示例:

**public** **class** UserTest {

@Test

**public** **void** testCreat() {

}

@Test

**public** **void** testDelete() {

}

}

## 功能测试

1. QA人工测试
2. 测试工具录制测试脚本（selenium，loadrunner等）

## 压力测试

1. ApacheBench
2. loadrunner录制测试脚本，设制访问用户数，和并发数量。

## 测试人员的职责：

* 在项目的前景、需求文档确立基线前对文档进行测试，从用户体验和测试的角度提出自己的看法。
* 编写合理的测试计划，并与项目整体计划有机地整合在一起。
* 编写覆盖率高的测试用例。
* 针对测试需求进行相关测试技术的研究。
* 认真仔细地实施测试工作。
* 进行缺陷跟踪与分析。
* 提交测试分析报告。

当然具体到某个角色的职责又有所不同，一般测试组内有测试经理、测试员、技术支持三个角色。

测试经理：组建测试组;协调测试组内部的沟通；代表测试组与其他角色组进行沟通（其他角色是指项目经理、开发人员）；编写测试计划；测试报告分析。

测试员：编写测试用例（有很多情况测试用例是由测试经理来编写的）；实施测试用例；执行测试。

技术支持：为测试工作提供技术支持。

# 版本控制

程序员编写程序的过程中，每个程序都会有很多不同的版本，这就需要程序员很好的管理代码，在需要的时间可以取出需要的版本，并且每个版本都有一个完整的说明。

使用Subversion（简称SVN）作为版本管理工具。在多个程序员管理同一段代码的过程中，版本的管理显得尤为重要，使用SVN可以方便的进行分支、合并，记录下所有的版本。

把版本划分为主线trunk 和分支branch和标记tag。**主线最干净的版本，是线上运行的稳定版本**。开发新功能从主线切出分支后，在分支上开发，开发结束并测试通过后才提交到主线上。Tag的主要作用是可以快速的回退到某一个稳定的时间点.

严格控制branch的数量，**保持trunk版本的稳定**，branch开发完成上线后merge回trunk，branch消除。有冲突解决机制，由冲突当事人仔细核对，敲定最终版本。branch merge到trunk由专人负责，避免错误修改提交到trunk。

Trunk稳定后, 上线时做一个tag. Trunk 上线后, 可能会有不尽人意的地方，也许需要回退到先前的一个版本. 这时就可以回退到前几次上线时做的tag.

**Trunk 和tag 由专人负责, 开发人员只能往branch中提交代码, trunk 和tag对开发人员只有读的权限.**

## 版本控制基本流程：

1. 各开发小组建立自己的分支，并在此分支上开发
2. 各开发小组把分支合并到主干上并形成较为稳定
3. 各个小组重新从主线上建立新的分支，在此分支上开发(即回到第一步)
4. 循环往复，直到工程结束。

## 项目实施版本控制分为4个步骤：

1. 开发版本develop, 如果有几个小组同时开发，可以切出多个分支。
2. 内测版本alpha，开发完成后转为alpha版本，做内部测试。
3. 公测版本beta，内部测试通过后转为beta版本，对外公开测试。
4. 上线版本production，测试全部通过后转为production版本，并合并到主线上

## SVN版本库布局

一个版本库包含多个项目，通常按分支来安排布局:

/trunk/project1

/trunk/project2

/branches/project1

/branches/project2

/tags/project1

/tags/project2

或者按项目:

/project1/trunk

/project1/branches

/project1/tags

/project2/trunk

/project2/branches

/project2/tags

如果项目不是密切相关，而且每一个是单独被检出，那么按项目布局是合理的。对于那些你想一次检出所有项目，或需要将它们打成一个分发包的相关项目，按分支来布局通常比较好。这种方式你只要检出一个分支，而且子项目之间的关系也比较清楚。

## 分支与标记



版本控制系统的一个特性是能够把各种修改分离出来放在开发品的一个分割线上。这条线被称为分支。分支经常被用来试验新的特性，而不会对主干线干扰。当新的特性足够稳定之后，开发品的分支就可以混合回主分支里（主干线）.

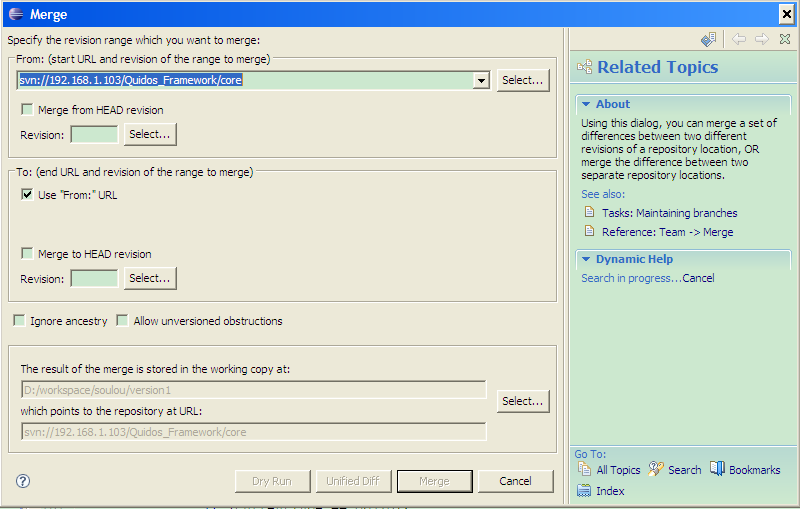
版本控制系统的另一个特性是能够标记特殊的版本（例如某个发布版本），所以你可以在任何时候重新建立一个特定的构件和环境。这个过程被称作标记。

Subversion 没有用于建立分支和标记的特殊命令，但是使用所谓的cheap copy来代替。cheap copy类似于Unix里的硬连接，它意思是代替一个版本库里的完整的拷贝，创建一个内部的连接，指向一个具体的版本树。结果分支和标记就迅速被创建，并且没有在版本库里占据任何额外的空间。

## Merge

当程序到达一个比较稳定的阶段，就需要把分支合并到主干上，下面讲述一下合并的流程。

1. Term1 从trunk切出branch (term1\_090420), Term2 从trunk切出branch (term2\_090420)
2. Term1与Term2分别修改自己分支上的代码.
3. Term1先把主干check out到本地。然后在主干的目录上右键选择Term->merge，弹出如下窗口：



此对话框的含义是把From指定的分支版本到To指定的分支版本之间的差异合并到主干上。

在这里分支选的是term1\_090420。

然后直接点击merge进行合并，你也可以通过dry run来看是不是两者之间有差异。由于没有其它人修改主干，所以合并的很顺利，合并完毕之后，由xb对主干进行提交。

1. 将term2\_090420分支合并到主干，解决冲突.

Term1合并完毕之后，Term2要将他的分支合并到主干上去，方法同上。但是由于Term1已经修改过主干，所以产生了冲突，会弹出一个冲突对话框。双击对话框中的产生冲突的文件名，就可以调出工具对此文件进行合并。

## 其它注意事项

### 文件的删除、移动与重命名

你可以重命名、移动或删除你的文件或文件夹，但请使用SVN进行这些操作，否则之前的版本信息会丢失。

使用SVN删除、移动与重命名文件夹的方法是在文件/文件夹上点右键进行SVN操作，或直接在资源浏览器中使用右键拖放（会弹出SVN选项）。

文件的删除、移动与重命名之前，必须保证工作目录是最新的版本；进行这些操作之后，需要进行提交。

### 提交的时机

每个工程会有很多个小模块，当某个模块达到稳定的时候，你就需要提交一次，以免写下个模块代码的时候出现不可恢复的错误。

每一次提交需要前，需要通过Java编译检查，保证是一个编译没有错误的版本。当提交比较稳定的版本的时候，同时要修改你的版本号。

提交的时候要添加注释，若多人共同修改同一段代码我们就需要为注释添加上更加详细的说明

# 数据备份

## Mysql备份策略大概可分为4种：

1. 直接拷贝数据库文件（不推荐）
2. 使用mysqlhotcopy备份数据库
3. 使用mysqldump备份数据库
4. 使用主从复制机制(replication)（实现数据库实时备份）

一般mysqldump使用比较广泛，操作也方便，下面就具体描述mysqldump备份策略。

## 数据备份分为三步：

1. 定时导出备份数据库并打包成gz压缩包 （Linux shell 脚本 + crontab）

附录一：mysqlFullBackup.sh

1. 通过ftp 服务定时下载到本地 （Linux shell 脚本 + crontab）

或从服务器上备份数据库到ftp 服务器 （Linux shell 脚本 + crontab）

附录二：ftpDownload.sh

附录三：dbPutToFtp.sh

1. 定期刻录光盘 (人工操作)

## Crontab 定时任务管理

crontab -e

01 03 \* \* \* /var/proftpd\_data/backup/bin/mysqlFullBackup.sh

10 04 \* \* \* /var/proftpd\_data/backup/bin/ftpDownload.sh

前五个字段的整数取值范围及意义是：

0～59 表示分

1～23 表示小时

1～31 表示日

1～12 表示月份

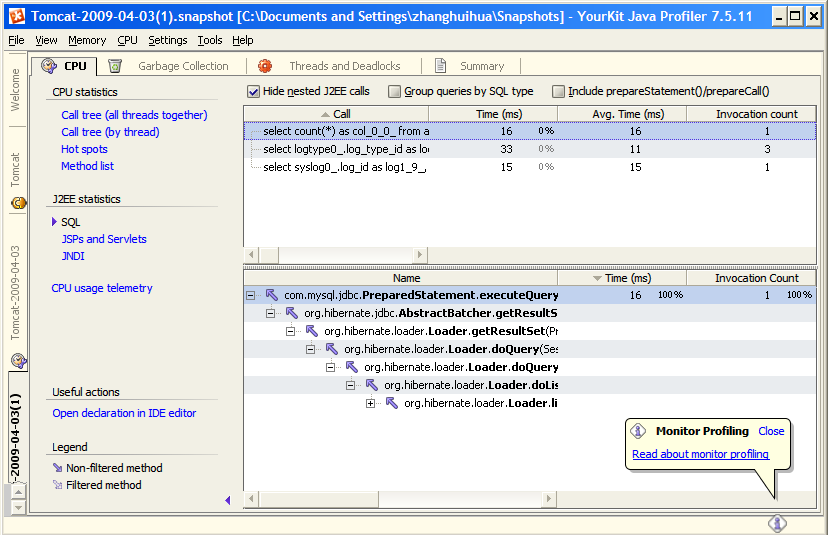
0～6 表示星期（其中0表示星期日）

# 常用工具

## Java Profiler

YourKit Java Profiler是业界领先的Java剖析工具。CPU，内存，sql语句执行效率等的剖析．

在线文档<http://www.yourkit.com/docs/index.jsp>



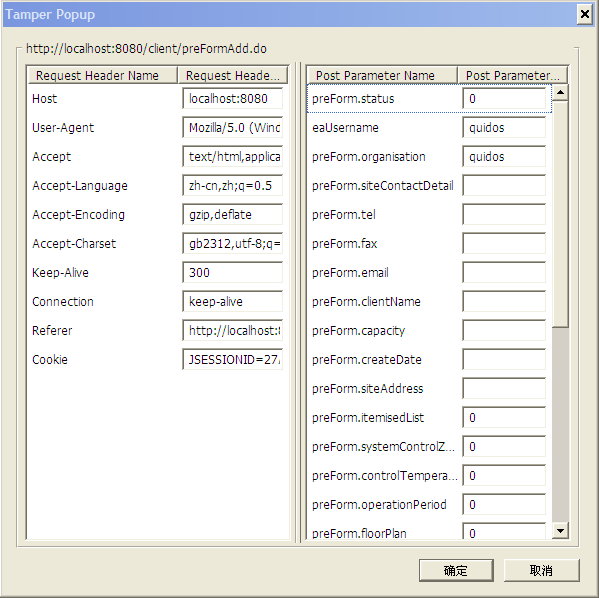
## Firebug

Firebug is implemented as a Firefox extension . Firebug integrates with Firefox to put a wealth of development tools at your fingertips while you browse. You can edit, debug, and monitor CSS, HTML, and JavaScript live in any web page...



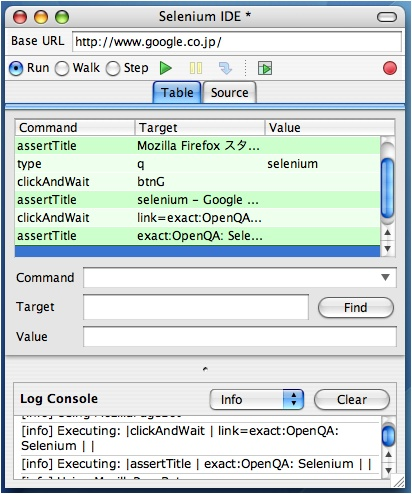
## Tamper Data

Tamper Data is implemented as a Firefox extension．Use tamperdata to view and modify HTTP/HTTPS headers and post parameters...



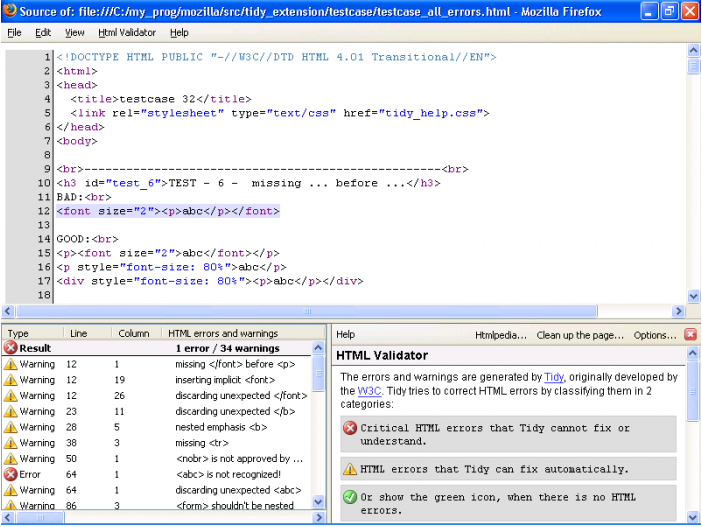
## Selenium IDE

Selenium IDE is an integrated development environment for Selenium tests. It is implemented as a Firefox extension, and allows you to record, edit, and debug tests. Selenium IDE includes the entire Selenium Core, allowing you to easily and quickly...



## Html Validator

HTML Validator is a Mozilla extension that adds HTML validation inside Firefox and Mozilla. The number of errors of a HTML page is seen on the form of an icon in the status bar when browsing.  
  
The extension can validate the HTML sent by the server or the HTML in the memory (after Ajax execution)



## Loadrunner

LoadRunnner这一强大的性能测试工具, 可以录制脚本,设并发数后去跑流程.

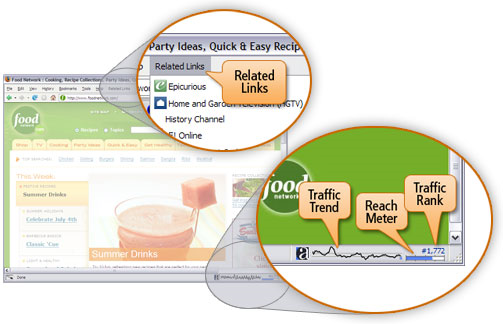
## ApacheBench - Apache 专用的压力测试工具

在阿帕奇服务器的套件中，有一个叫做 ab (ApacheBench) 的工具。ApacheBench 主要是用来测试阿帕奇服务器执行效率用的.

ApacheBench 可以针对某个特定的 URL 仿真出连续的联机请求，同时还可以仿真出同时间点数个相同的联机请求，因此利用 ApacheBench 可帮助我们在网站开发期间仿真实际上线可能的情况，利用仿真出来的数据做为调整服务器设定或程序的依据.

## Alexa Sparky

查看网站全球排名的firefox插件．



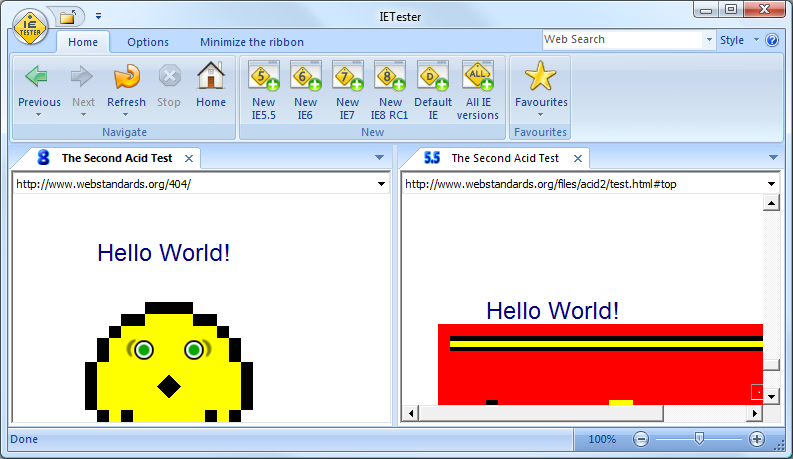
## SearchStatus

Display the Google PageRank, Alexa rank, Compete ranking and SEOmoz Linkscape mozRank anywhere in your browser, along with fast keyword density analyser, keyword/nofollow highlighting, backward/related links, Alexa info and more.

## IETester

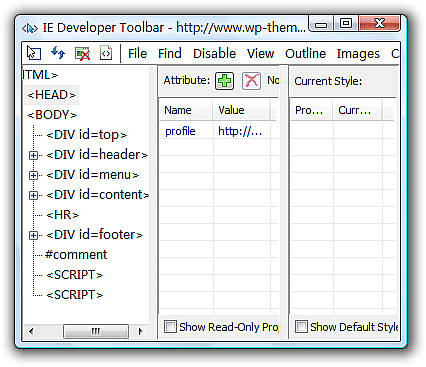
主要是用来调试UI在IE各个版本中的兼容性问题.

IETester is a free WebBrowser that allows you to have the rendering and javascript engines of IE8, IE7 IE 6 and IE5.5 on Vista and XP, as well as the installed IE in the same process.



## IE Developer Toolbar

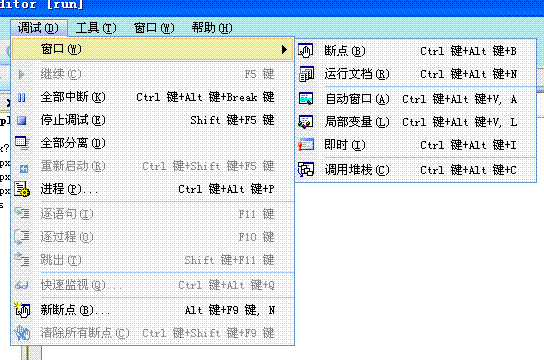
相当于firefox下的firebug插件.



## Office 中带的脚本调试器

打开添加/删除程序->添加/删除 Windows 组件，然后可以看到一个“脚本调试器”

在Word中，选择"工具→"宏→"Microsoft 脚本编辑器"命令



# 附录

## 附录一 (mysqlFullBackup.sh)

#!/bin/sh

# Use mysqldump --help get more detail.

#

# 定义变量，请根据具体情况修改

# 定义脚本目录

scriptsDir=`pwd`

# 定义用于备份数据库的用户名和密码

user=root

userPWD=123456

# 定义备份数据库名称

dbNames=(database1 database2 database3)

# 定义备份目录

dataBackupDir=/home/zhanghuihua/backup

# 定义邮件正文文件

eMailFile=$dataBackupDir/log/email.txt

# 定义邮件地址

eMail=Huihua.Zhang@quidos.co.uk

# 定义备份日志文件

logFile=$dataBackupDir/log/mysqlbackup.log

# DATE=`date -I`

DATE=`date -d "now" +%Y%m%d`

echo `date -d "now" "+%Y-%m-%d %H:%M:%S"` > $eMailFile

for dbName in ${dbNames[\*]}

do

# 定义备份文件名

dumpFile=$dataBackupDir/db/$dbName-$DATE.sql.gz

# 使用mysqldump备份数据库，请根据具体情况设置参数

mysqldump -u$user -p$userPWD $dbName | gzip > $dumpFile

done

if [[ $? == 0 ]]; then

echo "DataBase Backup Success!" >> $eMailFile

else

echo "DataBase Backup Fail!" >> $emailFile

fi

# 写日志文件

echo "================================" >> $logFile

cat $eMailFile >> $logFile

echo $dumpFile >> $logFile

# 发送邮件通知

#cat $eMailFile | mail -s "MySQL Backup" $eMail

## 附录二 (ftpDownload.sh)

#!/bin/bash

# 定义变量

DATE=`date -d "now" +%m%d%y`

ftpServer=192.168.1.103

ftpUser=zhanghuihua

ftpPasswd=123456

remoteDir=/backup/db

localDir=/home/zhanghuihua/backup

#开始ftp操作

/usr/bin/ftp -niv <<!

open $ftpServer

user $ftpUser $ftpPasswd

binary

cd $remoteDir

lcd $localDir

mget database1-$DATE.sql.gz

mget database2-$DATE.sql.gz

mget database3-$DATE.sql.gz

bye

!

## 附录三(dbPutToFtp.sh)

#!/bin/bash

# 定义变量

DATE=`date -d "now" +%Y%m%d`

ftpServer=192.168.1.103

ftpUser=zhanghuihua

ftpPasswd=123456

remoteDir=/backup/db

localDir=/home/zhanghuihua/backup/db

#开始ftp操作

/usr/bin/ftp -niv <<!

open $ftpServer

user $ftpUser $ftpPasswd

binary

cd $remoteDir

lcd $localDir

mput database1-$DATE.sql.gz

mput database2-$DATE.sql.gz

mput database3-$DATE.sql.gz

bye

!