



# Grails 企业 web 应用开发与部署

作者：李茂圣

日期：2009 年 8 月 20 日

本人保留所有权利

# 序言

Grails 是个好东西，在中国算是个新东西，尝鲜总要付出代价。

我喜欢能快速解决问题的方式方法，有些时候即使付出代价，也值了。

在大家都还在犹豫的时候，综合分析后，决定在我们的项目中使用 Grails 来开发，为什么不选择 ROR(Ruby and Rails)，因为是 java 出生的，所以选择 GOG(Groovy On Grails)。

在将近一年的开发过程中，问题总是不断，总是在不断的尝试、咨询。国内目前大部分的开发都是居于个人尝试，所以项目实际应用中的问题基本没人解答，在完成我们的前期开发和部署后，我简单的把我一些经验分享给大家，希望大家在将来的实际应用中有有所帮助。如果大家在开发过程中遇到什么问题，也可以进入我们的论坛（Groovy 中文论坛：<http://groovycn.5d6d.com>）交流或者直接发 EMAIL([vottot@qq.com](mailto:vottot@qq.com))给我，我们可以一起探讨、解决问题。

本教程主要讲解企业开发环境和部署环境，不会涉及太深入的开发。

同时，感谢每一位支持我们的人，更感谢为 GOG 推广做出努力的、无私奉献的人。

# 目录

## 第1章 简介

- 1.1 Grails 介绍
- 1.2 Grails 应用

## 第2章 开发准备

- 2.1 安装 JDK
- 2.2 安装 Grails
- 2.3 安装数据库
- 2.3 安装开发工具

## 第3章 Grails WebMail

- 3.1 创建第一个 Grails 程序
- 3.2 Grails 工程结构
- 3.3 创建域(Domain)
- 3.4 创建控制器
- 3.5 创建视图
- 3.6 数据源配置
- 3.7 运行 Grails 程序
- 3.8 配置本地开发模式

## 第4章 Grails 服务器环境

- 4.1 Tomcat 安装
- 4.2 Apache 安装

4.3 Mod\_jk 连接配置

4.4 创建自己的服务器配置

4.5 虚拟主机的配置

## 第 5 章 总结

# 第 1 章 简介

Groovy 在 JAX 会议上获得最具创新和创造性项目，注定他将成为一颗新秀，与 JAVA 的无缝集成，注定是企业级领域的一颗新星。

## 1.1 Grails 介绍

Grails 是一套用于 Web 敏捷开发的开源框架，它基于 Groovy 编程语言，并构建于 Spring、Hibernate 和其它标准 Java 框架之上。2008 年 11 月 11 日，SpringSource（Spring 的所有者）宣布并购了 G2One Inc（Groovy 和 Grails 的所有者），这对 Groovy 和 Grails 的发展来说无疑是很好的消息。

## 1.2 Grails 应用

在学习本教程的过程中中，我将创建一个 WebMail 的应用程序来探讨 Grails 应用开发的各个方面。目前在企业应用的开发中，邮件发送是一个很基本的功能，在传统 J2EE 的开发中，要实现一个邮件发送的功能还是稍微有点麻烦。但是在 Grails 基于插件式开发中，实现一个邮件发送是很简单的，在后面的教程中我将逐步讲解。Grails 的应用很广泛，期待大家一起来发掘。

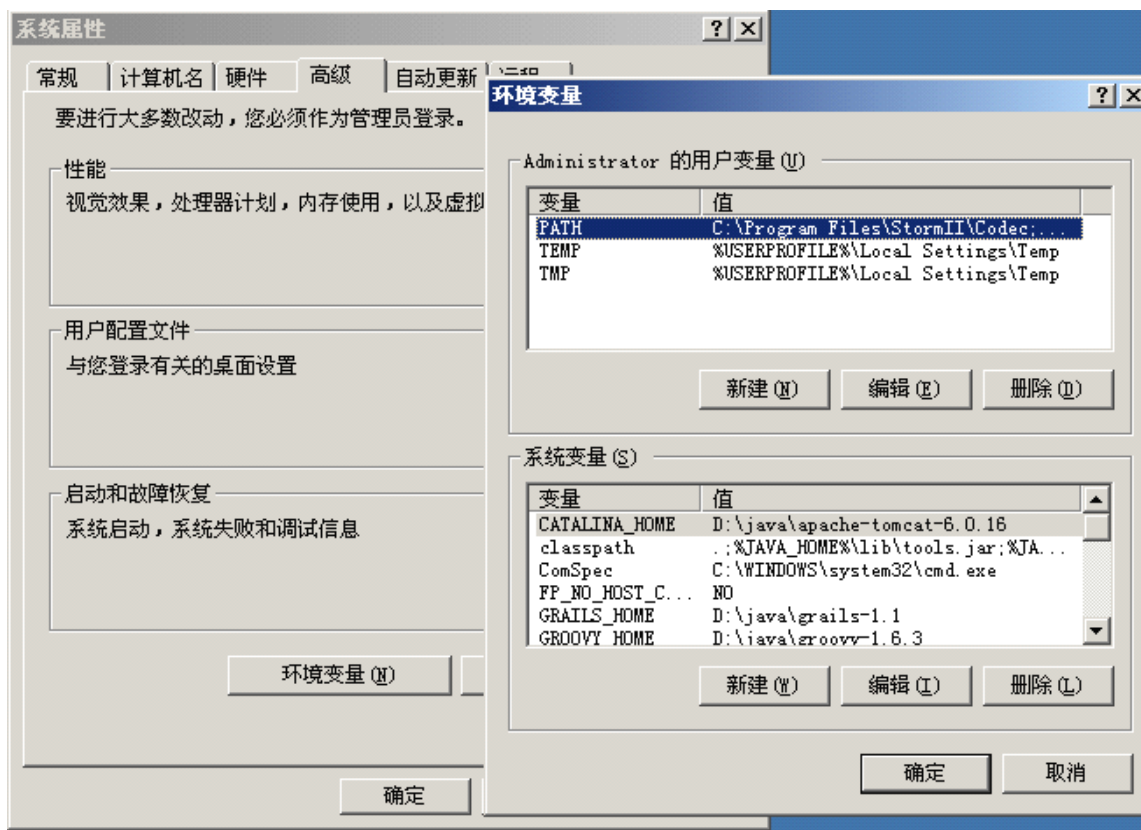
# 第 2 章 开发准备

本教程的开发环境是：Windows server 2003 EnterPrise  
+JDK6+IntelliJ IDEA 8.1+mysql-5.0.22-win32+Grails-1.1.

## 2.1 安装 JDK

Groovy 是基于 JVM 虚拟机的动态语言，所以开发前需要安装 JDK。从 Sun 的官方网站上下载 JDK1.6 以上版本，这里使用的是：  
jdk-6u3-windows-i586-p.Exe，根据提示安装，本机开发我把所有的 java 开发软件都安装在 D:\java 下面，JDK 的安装后的目录为：D:\java\jdk1.6.0\_03，安装 JDK 会自动提示安装 JRE，JRE 安装后的目录为：D:\java\jre1.6.0\_03。安装完成后开始设置 JDK 环境变量。

我的电脑->右键->属性->高级->环境变量：



新建环境变量：

变量名：JAVA\_HOME

变量值：D:\java\jdk1.6.0\_03 （这个值是你的 JDK 安装目录）

变量名：classpath

变量值：.;%JAVA\_HOME%\lib\tools.jar;%JAVA\_HOME%\lib\dt.jar ;

（注意前面有个".",一定不能缺少）

变量名：Path(这个值已经存在，添加下面的值就可以)

变量值：;%JAVA\_HOME%\bin;

设置完成后：开始菜单->运行->cmd->java -version:如下图显示，证明 JDK 安装成功。

```
C:\Documents and Settings\Administrator>java -version
java version "1.6.0_13"
Java(TM) SE Runtime Environment (build 1.6.0_13-b03)
Java HotSpot(TM) Client VM (build 11.3-b02, mixed mode, sharing)
```

## 2.2 安装 Grails

从 Grails 官方网站：<http://www.grails.org> 下载 Grails 版本，目前我们使用的是 Grails 1.1 正式版本，下载完成后解压到 D:\java 下面，解压后的目录结构为：D:\java\grails-1.1。

新建环境变量：

变量名：GRAILS\_HOME

变量值：D:\java\grails-1.1 （这个值是你的安装目录）

变量名：Path(这个值已经存在，添加下面的值就可以)

变量值：;%GRAILS\_HOME%\bin;

设置完成后：开始菜单->运行->cmd->grails -version:如下图显示，证明安装成功。

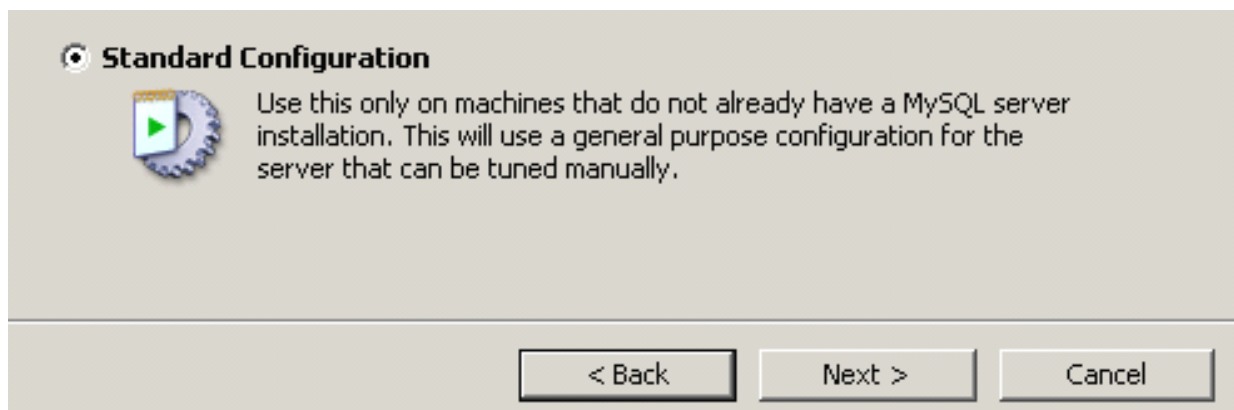
```
C:\Documents and Settings\Administrator>grails -version
Welcome to Grails 1.1 - http://grails.org/
Licensed under Apache Standard License 2.0
Grails home is set to: D:\java\grails-1.1

Base Directory: C:\Documents and Settings\Administrator
Running pre-compiled script
Script not found: Version
```

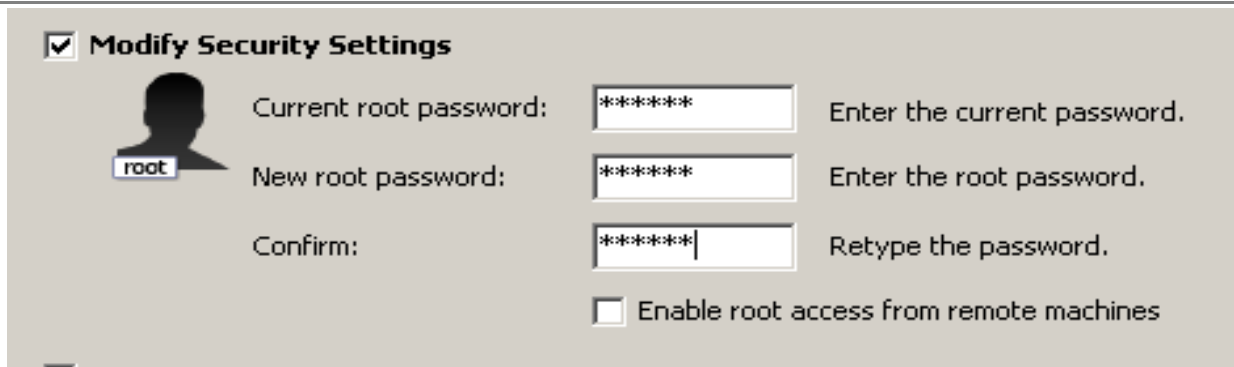
## 2.3 安装数据库

从 mysql 官方网站上下载 mysql 安装文件，本地使用的版本是 mysql-5.0.22-win32.Exe，我下载 mysql-5.1.37 没安装成功，没具体去看是什么问题。于是就使用 5.0 的版本。


安装 mysql 的时候注意设置访问密码：选择标准配置，如果是允许从其它电脑上登录这台 mysql 服务器的话需要选中那个复选框。







☒ **Modify Security Settings**

 **Current root password:**  Enter the current password.

**New root password:**  Enter the root password.

**Confirm:**  Retype the password.

☐ Enable root access from remote machines

配置完成后：开始菜单->运行->cmd->mysql -uroot -p123456:如下图显示，证明安装成功。(root 是用户名，123456 是密码)

```
C:\Documents and Settings\Administrator>mysql -root -123456
mysql: unknown option '-1'

C:\Documents and Settings\Administrator>mysql -uroot -p123456
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3 to server version: 5.0.22-community-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> _
```

## 2.3 安装开发工具

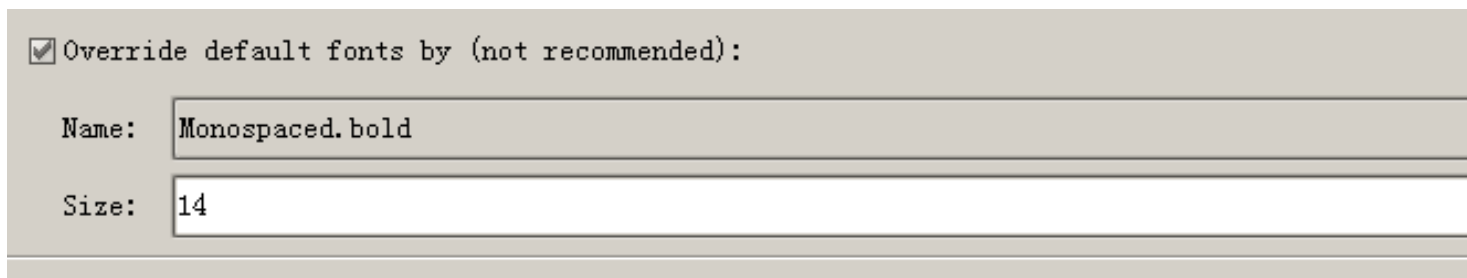
一门好的语言必定需要一个好的 IDE 支持，前期为了尽快熟悉 Grails，你可以使用记事本和 Grails 的 Scripts 来创建和开发项目( 可查看 Grails1.1 中文文档 )，以方便你了解 IDE 的原理。目前支持 Groovy 和 Grails 的 IDE 有 Eclipse, NetBeans, IntelliJ Idea，对于免费的开发工具 Eclipse 和 NetBeans 是不错的选择，当我每个开发工具都测试过后，我发现 IntelliJ Idea 对 Grails 的支持目前来说应该是最强大的了。举个小例子：在 Eclipse 和 NetBeans 里面你无法直接为你的 Domain(域)创建一个包，如果你手工创建后，自动生成的 controller(控制器)和 view(视图)也不会保留包的格式，但是 IntelliJ 能做到，最

早的时候就因为这点我才决定一定要使用这个开发工具，尽管以前都没听过这个工具。

从 IntelliJ 官方网站( <http://www.jetbrains.com/idea/> )上下载 idea,IntelliJ 是收费的，所以如果是作为正式开发使用，请购买正版哦。

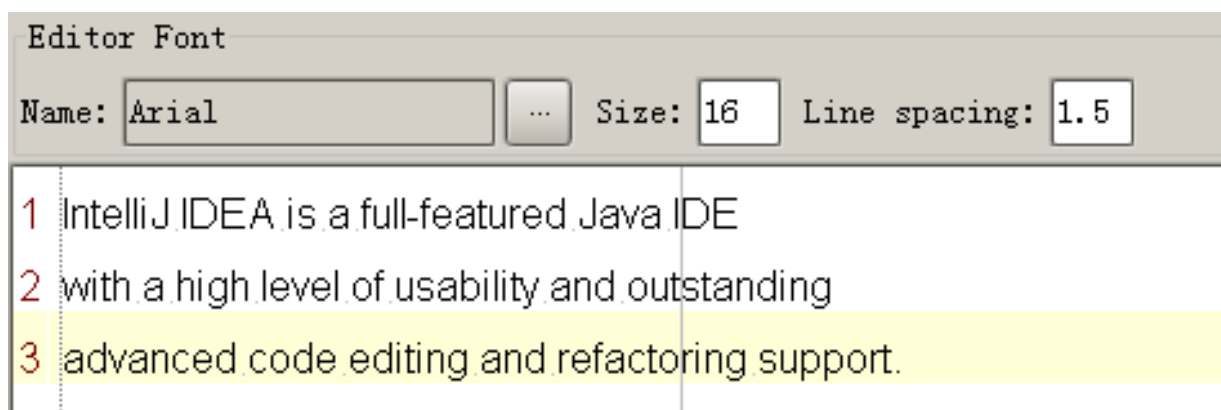
安装完成后你会发现默认的菜单英文字体很小，你可以设置：

File->Settings->Appearance 修改:修改 14 号字体后漂亮多了



你还可以设置代码的字体个大小:

File->Settings->Edition->Colors&Fonts->Font->Save As...->输入名称：china,然后设置字体、字号、间距等。



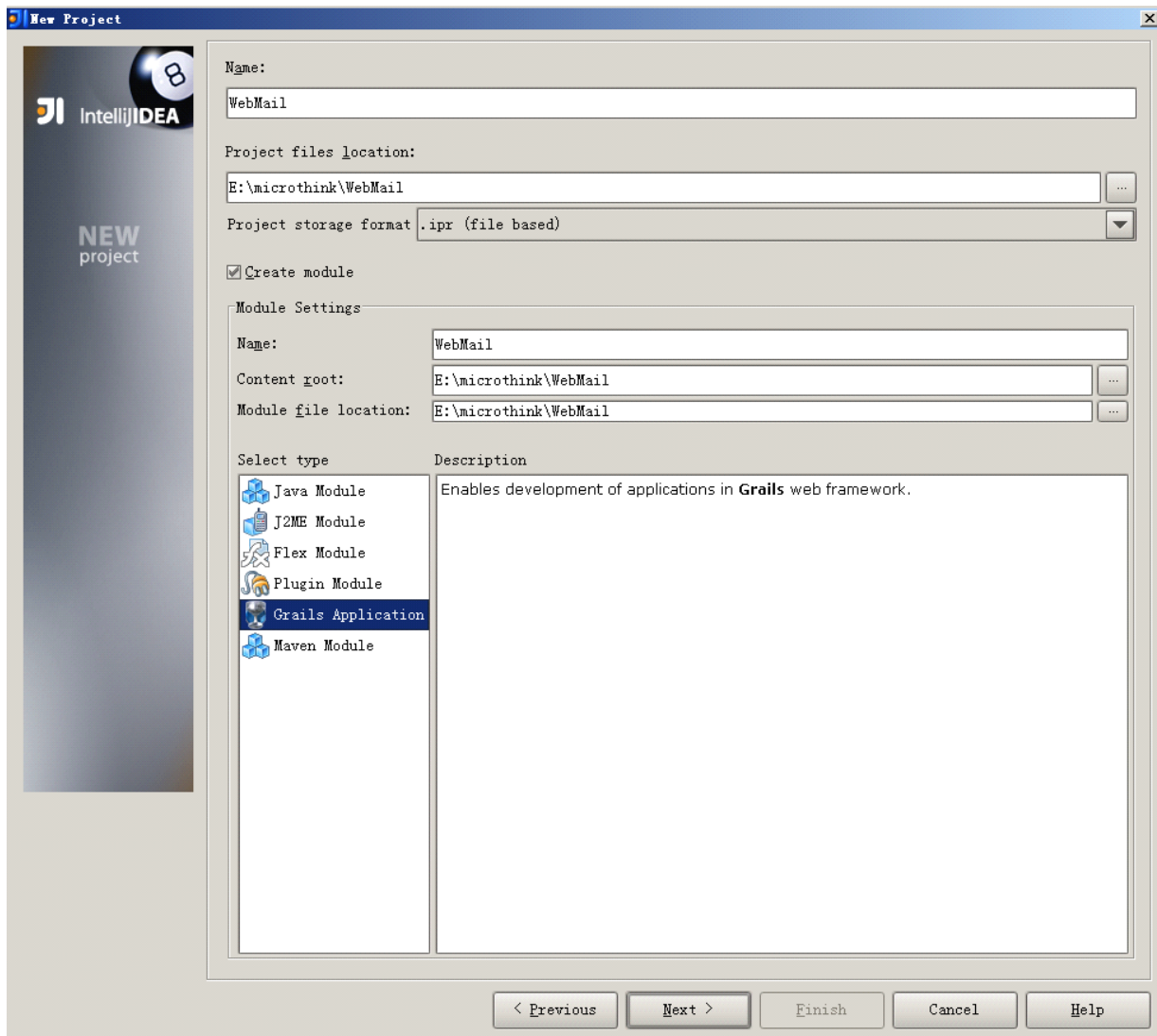
基本设置完成后可以开始开发了。

## 第 3 章 Grails WebMail

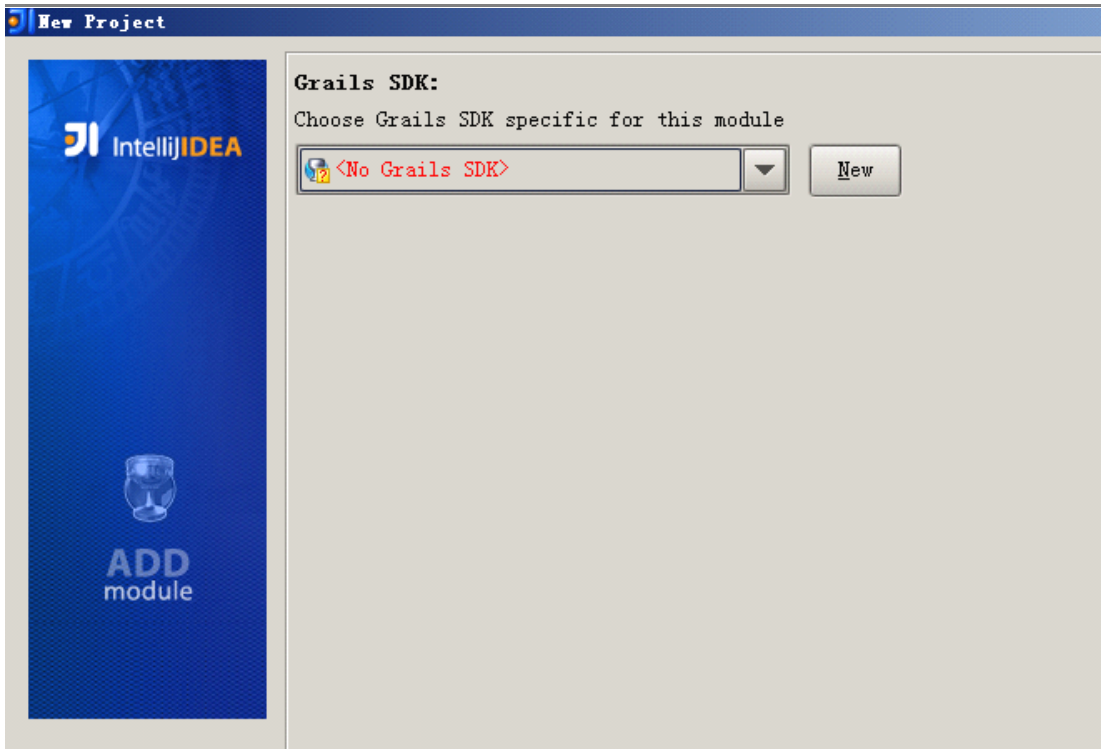
此处的教程将直接使用开发工具操作，不再讲解基本的操作命令。

### 3.1 创建第一个 Grails 程序

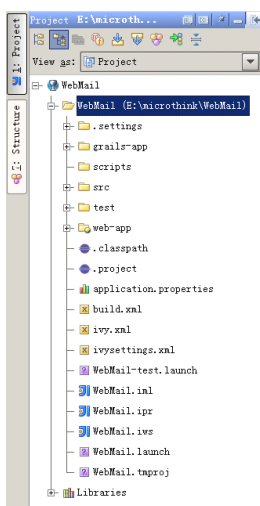
打开 IntelliJ->File->New Project->Create project from  
scrath:



点击下一步：



点击 New,找到 Grails 的安装目录并选择后：



点击下一步->都不用选择直接点击完成，这样我们的一个

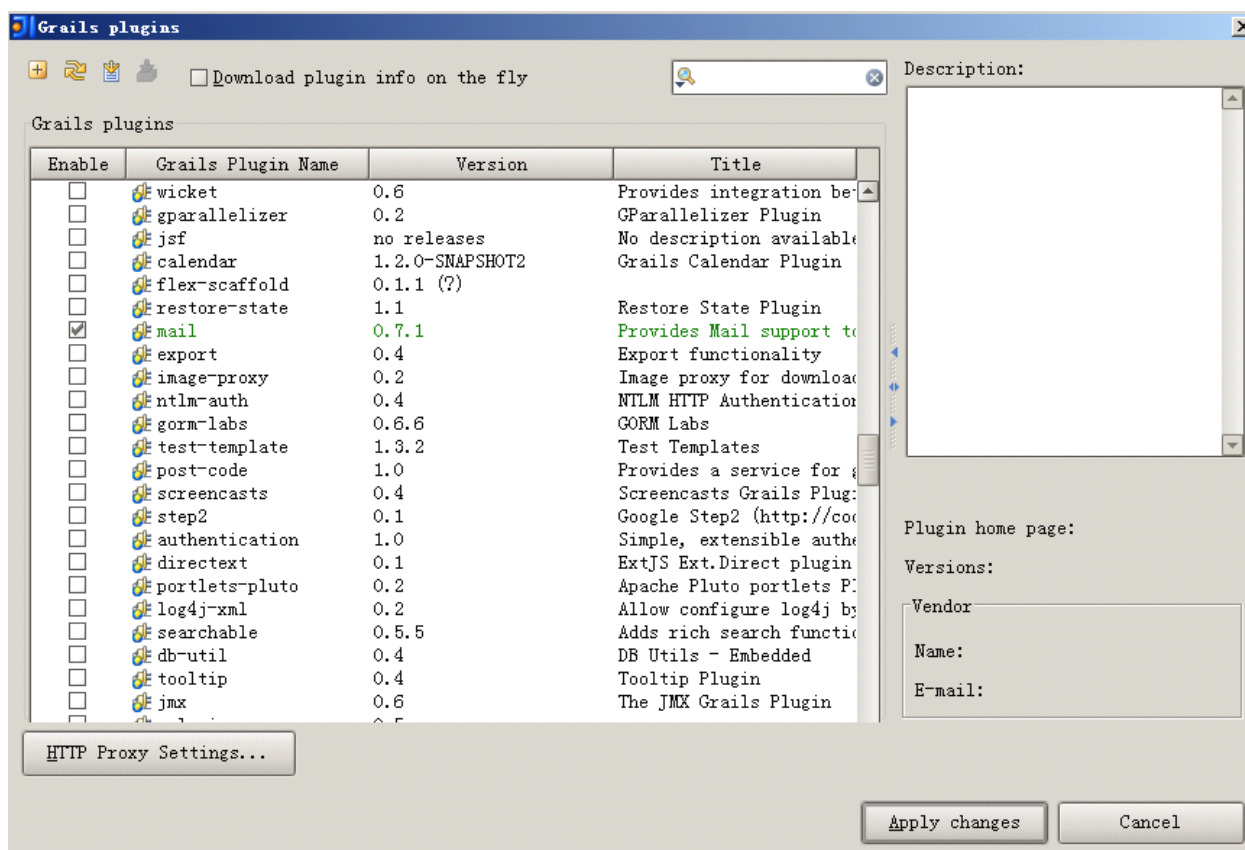
Grails 工程就自动创建完成了。

## 3.2 添加 Grails Mail 插件

在这里解释一下 Grails 插件 :Grails 开发者上传了很多自己开发的功能插件,你可以直接上官方网站找到你所需要的插件下载安装后,你就能在你的程序中直接使用相关的功能,目前官方网站上已经提供了很多很好用的插件,自己上去发现你需要的功能。

到 Grails 官方网站上下载 Mail 插件: <http://www.grails.org/plugin/mail> 下载完成后开始安装。

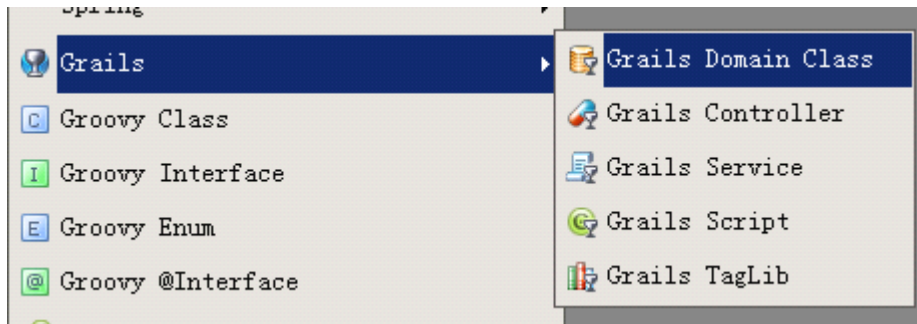
在 IntelliJ 里面右键项目名称->Grails plugins->点击+号->找到刚才下载的插件->点击 Apply changes->弹出的确认窗口点击 OK->完成 Mail 插件的安装。



### 3.3 创建域(Domain)

展开项目名称->grails-app->domain->右键->New->Package->输入

包名：cn.microthink.Mail->在包下面右键-New->



输入名称：SendMail，我们将看到 IDE 已经给我们创建好的域：

```
package cn.microthink.mail

class SendMail {

    static constraints = {
    }
}
```

里面有包名 cn.microthink.mail ,域类名 SendMail ,还有个约束条件的闭包：static constraints。

创建好域后，我们要考虑为我们新建的域添加些什么内容？

要发送邮件的话，至少需要对方的 email 地址，邮件主题，邮件内容，我们还可以增加一个邮件发送时间，以方便我们以后查询使用，所以我们添加 3 个字段：

String emailAdd;

String title;

String content;

Date sendTime;

现在我们要考虑，程序在运行的时候要判断是否是正确的 Email 地址并且不允许为空，邮件主题的话也不能为空，邮件内容的话会很长，我们必须保证足够的长度够用户输入，邮件发送的时间默认就为当前时间，加上我们的条件后程序是这个样子：

```
class SendMail {  
    String emailAdd;        //邮件地址  
    String title;           //主题  
    String content;         //内容  
    Date sendTime;          //发送时间  
    //映射数据库字段  
    static mapping = {  
        columns {  
            //映射 content 数据库字段为 text 类型  
            content type: 'text' //邮件内容  
        }  
    }  
    //约束条件  
    static constraints = {  
        //最大长度为 200，不允许为空，必须是 email 格式  
        emailAdd(maxLength: 200, blank: false, email: true);  
    }  
}
```



```
title(maxLength: 200, blank: false) ;

content(blank: false) ;

}

//在插入这条数据以前执行当前操作

def beforeInsert = {

    //设置该字段值为当前时间

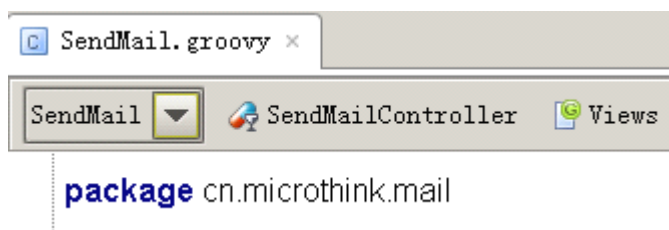
    sendTime = new Date() ;

}

}
```

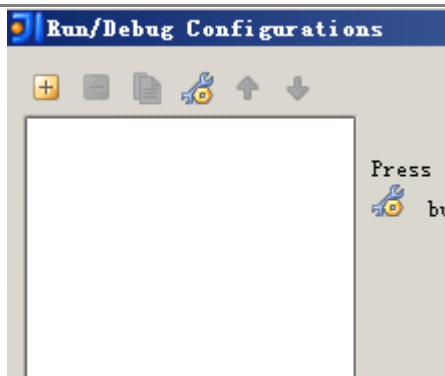
### 3.4 创建控制器和视图

上一小节我们已经创建了域，现在我们需要创建该域的控制器和视图。

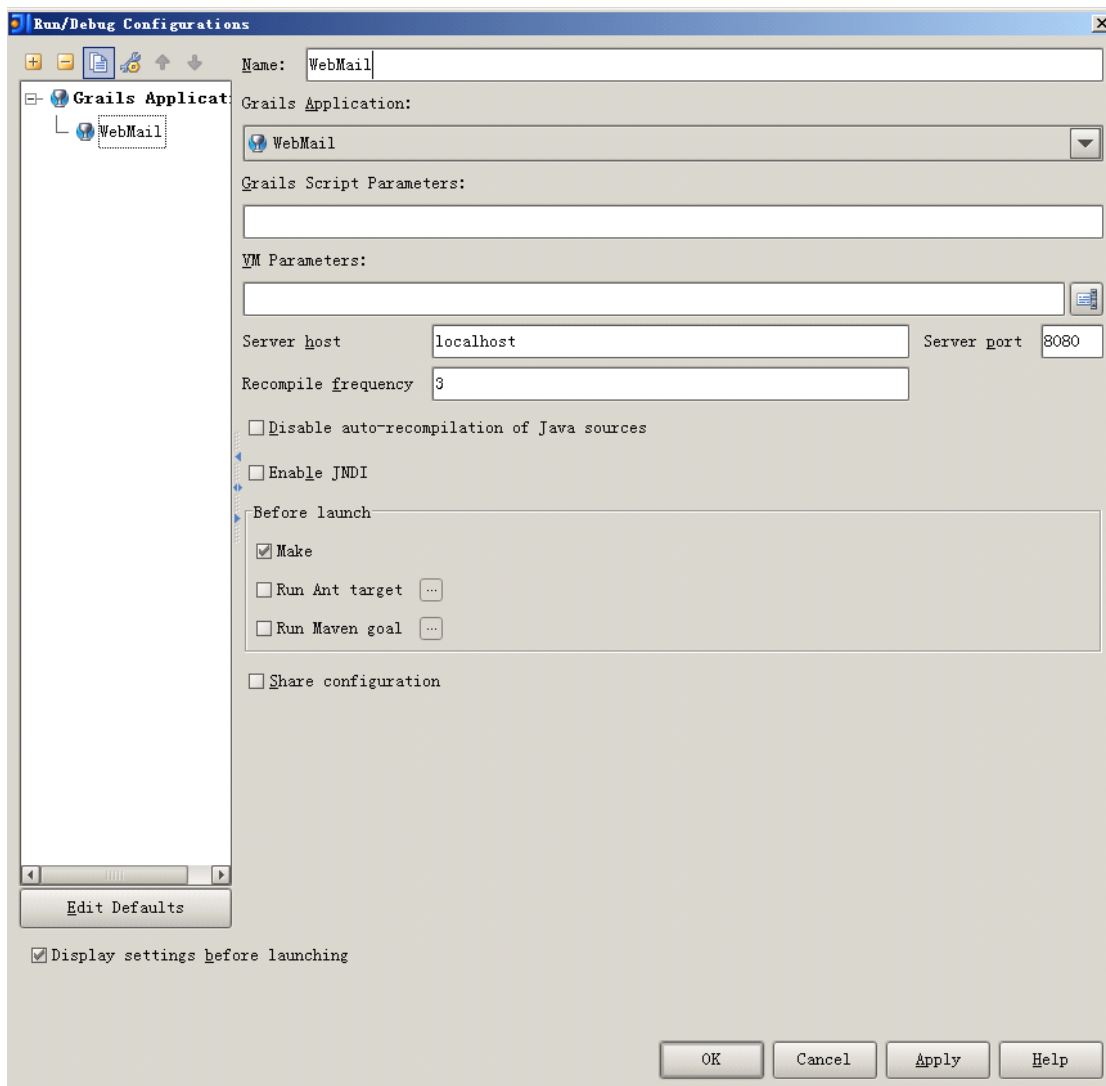


直接点击 SendMailController 后会提示 Generate Controller 将自动创建控制器，

直接点击 Views 后会提示 Generate View 将自动创建视图。这样基本的增、删、改操作就都完成了。我们现在先来运行看看是什么效果，点击菜单 :Run->Edit Configurations:



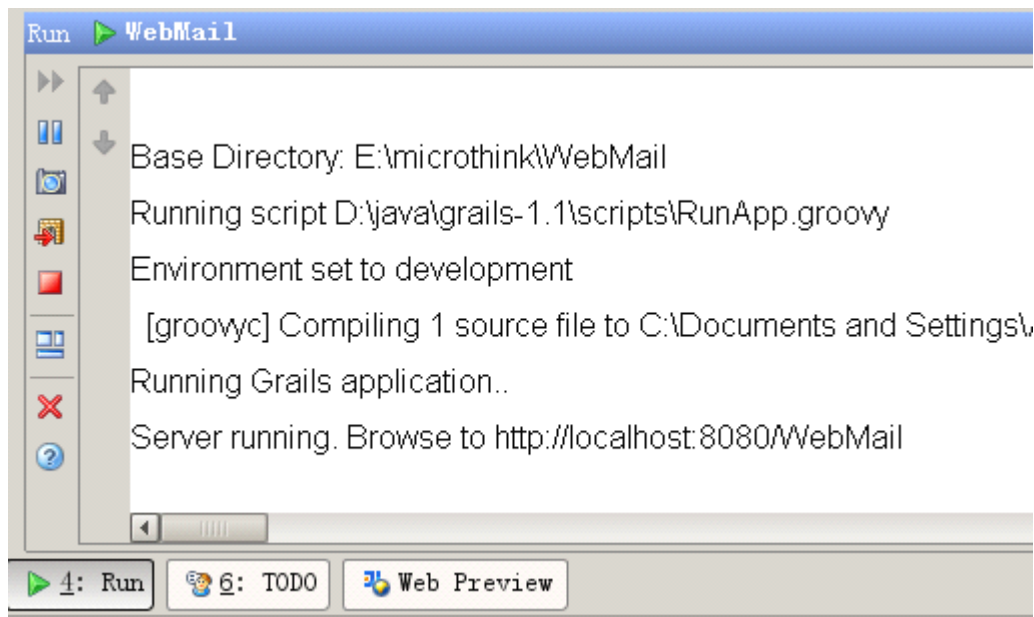
点击+号，选择 Grails Application,输入名称:WebMail,你可以修改端口等，设置完成点击 OK。



下面你可以直接点击 Run 就开始运行该程序，该运行方法和使用：grails run-app 命令是一样的，Grails 内置 Jetty 服务器，但是目前发布的新版本中，Spring 已经把内置服务器更换为 Tomcat.

当看到一下界面的时候：说明已经启动成功，在浏览器中输入：

http://localhost:8080/WebMail，将看到我们刚才新建的控制器，如图：



## Welcome to Grails

Congratulations, you have successfully started your first Grails application! At the momer display whatever content you may choose. Below is a list of controllers that are currently

- [cn.microthink.mail.SendMailController](#)

点击:cn.microthink.mail.SendMailController 后就可以进行默认的增、删、改操作，因为要保存数据，肯定有数据库，Grails 内置了 hsql 数据库，所有你可以不进行任何配置直接使用内置数据库测试，在下一节中我们将讲解如何配置数

数据库。

### 3.5 数据源配置

上一节的中，我们所有的测试数据都保存在 Grails 内置数据库中，现在我们需要使用我们自己的 mysql 数据库，请看下面的配置：

1.首先在 mysql 中创建一个名称为：WebMail 的数据库：

开始->CMD->mysql -uroot -p123456->执行：create database

WebMail;->执行：show databases; 看到一下结果表示操作成功：

```
mysql> create database WebMail
-> ;
Query OK, 1 row affected (0.05 sec)

mysql> show database
-> ;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
corresponds to your MySQL server version for the right syntax to use near 'ase' at line 1
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| test |
| webmail |
+-----+
```

2.下载 mysql 驱动：<http://www.mysql.com/products/connector/j/>，

选择：JDBC Driver for MySQL (Connector/J)下载，打开压缩包，将 mysql-connector-java-5.1.8-bin.jar 解压到项目文件夹下面的 lib 文件夹里面。

3.修改 grails-app\conf 文件夹下面的 DataSource.groovy

dataSource {

pooled = true

driverClassName = "com.mysql.jdbc.Driver"

username = "root"

```
password = "123456"
}

hibernate {
    cache.use_second_level_cache=true
    cache.use_query_cache=true

    cache.provider_class='com.opensymphony.oscache.hibernate.OSCachePr
    ovider'
}

// environment specific settings
environments {
    development {
        dataSource {
            dbCreate = "create" // one of 'create', 'create-drop','update'
            url =
            "jdbc:mysql://localhost/WebMail?useUnicode=true&characterEncoding=
            UTF8"
        }
    }

    test {
        dataSource {
            dbCreate = "update"
```

```
        url =  
        "jdbc:mysql://localhost/WebMail?useUnicode=true&characterEncoding=  
        UTF8"  
    }  
}  
production {  
    dataSource {  
        dbCreate = "update"  
        url =  
        "jdbc:mysql://localhost/WebMail?useUnicode=true&characterEncoding=  
        UTF8"  
    }  
}  
}
```

主要是修改：driverClassName、username、password、url，这里需要注意的是 grails 有几中运行模式：开发模式、调试模式、产品模式，所以 3 个地址的 URL 都需要修改，具体 3 个模式的区别可以看看 grails api. dbCreate 也有 3 种设置：

create-drop - 当 Grails 运行的时候删除并且重新创建数据库

create - 如果数据库不存在则创建数据库，存在则不做任何修改。删除现有的数据。

update - 如果数据库不存在则创建数据库，存在则对它进行修改更新。

4.重新运行程序，以后就可以使用 mysql 数据库了，关于 sql server 的配置这里也附上一份采用 jtds 连接的案例，使用前要下载 jtds 驱动：

```
dataSource {  
    pooled = true  
    driverClassName = "net.sourceforge.jtds.jdbc.Driver"  
    username = "sa"  
    password = "sa"  
    //logSql = true  
}  
  
hibernate {  
    cache.use_second_level_cache = true  
    cache.use_query_cache = true  
    cache.provider_class =  
'com.opensymphony.oscache.hibernate.OSCacheProvider'  
}  
  
// environment specific settings  
  
environments {  
    development {  
        dataSource {  
            dbCreate = "update" // one of 'create', 'create-drop','update'  
            url = "jdbc:jtds:sqlserver://192.168.0.48:1433/System"        }  
    }  
}
```

```
    }  
}  
test {  
    dataSource {  
        dbCreate = "update"  
  
        url = "jdbc:jtds:sqlserver://192.168.0.48:1433/System"  
  
    }  
}  
production {  
    dataSource {  
        dbCreate = "update"  
  
        url = "jdbc:jtds:sqlserver://192.168.0.48:1433/System"  
  
    }  
}  
}
```

### 3.6 邮件模块的配置

当我们基本的框架已经搭建好后，现在要实现一个发送邮件的功能，邮件发送需要一个单独界面，但是目前为了测试方便，不在做单独的界面，直接使用自



动生成的添加数据界面，当添加一条数据的时候同时把内容自动发送到这个邮件地址中。

接下来我们开始配置邮件发送模块：在前面的讲解中我们已经安装了 Grails mail 插件，现在我们打开 grails-app/Config.Groovy 文件，在最下面增加：

```
grails {  
    mail {  
        host = "smtp.gmail.com"  
        port = 465  
        username = "用户名@gmail.com"  
        password = "密码"  
        props = ["mail.smtp.auth":"true",  
                "mail.smtp.socketFactory.port":"465",  
                "mail.smtp.socketFactory.class":"javax.net.ssl.SSLSocketFactory",  
                "mail.smtp.socketFactory.fallback":"false"]    }  
}
```

修改你的发送邮箱的 smtp 地址，端口，用户名，密码，如果你使用 163 邮箱，请注意你是否有 SMTP 的功能，163 对新注册的邮箱都没有开启 smtp 功能。

配置完成后，下面我们将实现添加数据的时候同时发送内容到相应的邮箱地址里面：

用 idea 打开

grails-app\controllers\cn\microthink\mail\SendMailController.Groovy 文

件，找到 save 方法，修改：

```
def save = {  
    def sendMailInstance = new SendMail(params)  
    if(!sendMailInstance.hasErrors() && sendMailInstance.save()) {  
        flash.message = "SendMail ${sendMailInstance.id} created"  
        redirect(action:show,id:sendMailInstance.id)  
    }  
    else {  
  
render(view:'create',model:[sendMailInstance:sendMailInstance])  
  
    }  
}
```

为：

```
def save = {  
    def sendMailInstance = new SendMail(params)  
    if (!sendMailInstance.hasErrors() && sendMailInstance.save()) {  
        //如果保存数据成功，就发送到保存的邮箱地址中  
        sendMail {  
            to    sendMailInstance.emailAdd    //发送到的邮箱地址  
            subject  sendMailInstance.title    //邮件主题  
            body    sendMailInstance.content    //邮件内容  
            //如果邮件包含 html 代码的话，请使用下面的语句
```

```
// html    sendMailInstance.content           //邮件内容
}

flash.message = "SendMail ${sendMailInstance.id} created"

redirect(action: show, id: sendMailInstance.id)

}

else {

    render(view: 'create', model: [sendMailInstance: sendMailInstance])

}

}
```

修改完成后我们可以运行看看测试是否实现了邮件的发送功能，运行程序后，  
点击 New SendMail：



Home SendMail List

### Create SendMail

Email Add:

Title:

Content:

Send Time:     :

 Create

输入邮箱地址，主题，输入的内容，因为前面我们已经添加了邮箱地址等约束条件，如果我们输入错误的邮箱地址，将会看到以下提示：


❗ [class cn.microthink.mail.SendMail]类的属性[emailAdd]的值[saas@sfsdf]不是一个合法的电子邮件地址

Email Add:

Title:

Content:

Send Time:     :

 Create

当我们输入正确的地址后，点击 create,数据首先会保存到我们的数据库中，然后发送到相应的邮箱：

Email Add:

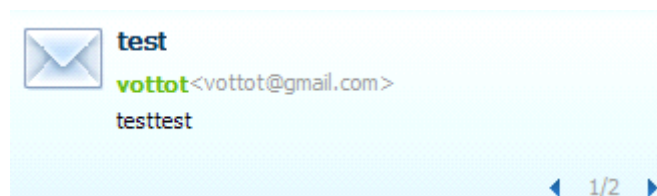
Title:

Content:

Send Time:     :

 Create

马上就会收到从 gmail 发过来的邮件了。



接下来我们把 Content 的 input 修改为 textarea:

```
<textarea id="content" name="content" col="10"
```

```
row="6">${fieldValue(bean:sendMailInstance,field:'content')}</textarea>
```

### Create SendMail

Email Add:

Title:

Content:

Send Time:     :

 Create

效果如图：

当我们在 Content 里面输入中文内容的时候，grails 会提示错误：Data truncation: Data too long for column 'content' at row 1，这是因为 mysql 的编码不正确：mysql 采用默认的编码 latin1 不支持中文：

修改 mysql 安装目录下面的 :my.ini 文件，把 default-character-set=latin1 修改为 default-character-set=utf8，记得有 2 个地方都要修改，为了程序字符的兼容性，开发过程中我们都是全部使用 UTF8，删除 webmail 数据库，重新创建一个 webmail 数据库，现在重新运行程序，中文也支持了，基本开发就到此完成了。

## 3.7 开发小节

在我们的一些 web 开发过程中，经常因为一些图片路径或者其它路径导致移植问题，因为 Grails 默认部署路径加上了应用程序的名称，而在我们正式发布的过程中每一个应用程序都对应了一个独立的网址，所以最好能开发环境跟部署环境一致的访问方式，避免后期的发布问题，这个问题也困扰了了我好几天，因为我们的程序在发布的时候好多路径都存在问题，也查找了一些资料，都没看到解决方式，后来就自己写了个 bat 文件测试，原理是在 tomcat 里面配置好一个访

问路径，不包含应用程序名称，然后把项目打包成 war 放置于 tomcat 能访问到的目录，最后启动 tomcat 服务实现访问，这样下来问题能解决了，但是每次修改都需要打包，而且项目文件越来越大的时候打包时间越来越长，最终实在没有什么好方法后到官方邮件里面求助，得到解决方法：直接修改 application.Properties 文件，在里面添加一行：app.context=/。就解决这个问题了，郁闷我了。

## 第 4 章 Grails 服务器环境

当我们把程序都开发完成后，现在需要正式发布出去了，下面主要讲解服务器环境的配置，服务器采用经典的 Apache+Tomcat 配置。

### 4.1 Tomcat 安装

下载 apache-tomcat-6.0.16.Zip，解压 apache-tomcat-6.0.16.Zip 到 E:\Server 下面，解压后的结构为：E:\Server\apache-tomcat-6.0.16。

设置 tomcat 环境变量：

CATALINA\_HOME: E:\Server\apache-tomcat-6.0.16

path: %CATALINA\_HOME%\bin

classpath: %CATALINA\_HOME%\lib\servlet-api.Jar

### 4.2 Apache 安装

下载 apache\_2.2.4-win32-x86-no\_ssl.Zip，根据提示进行安装，选择安装目录：E:\Server\Apache2.2，要求输入：Network Domain、Server Name、

EMAIL 的时候你可以自己输入。

安装 apache 的时候一定要注意版本问题 ,因为 mod\_jk 有些版本使用不了。

### 4.3 Mod\_jk 安装配置

下载 mod\_jk-1.2.26-httpd-2.2.4 的版本 , 拷贝  
mod\_jk-1.2.26-httpd-2.2.4.so 到 E:\Server\Microthink-Apache\modules 里  
面并且修改名称为 : mod\_jk.so。

### 4.4 创建自己的服务器配置

在 E:\Server\Apache2.2\conf\httpd.conf 里面增加一行 :

include E:\Server\Microthink-Apache\conf\mod\_jk.conf , 这里我是自己创建  
了一个 Microthink-Apache 的文件夹 , 我把一些配置和日志文件都专门放在这个  
目录里面方便管理。在 E:\Server\Microthink-Apache\conf\里面新建  
mod\_jk.conf 文件 , 添加以下内容 :

LoadModule jk\_module E:\Server\Microthink-Apache\modules\mod\_jk.so

JkWorkersFile "E:\Server\Microthink-Apache\conf\workers.properties"

#指定 tomcat 监听配置文件地址

JkLogFile "E:\Server\Microthink-Apache\conf\mod\_jk.log"

#指定日志存放位置

JkLogLevel info

#指定日志级别

在 E:\Server\Microthink-Apache\conf 新建 : workers.properties 文件。

在 E:\Server\Microthink-Apache\conf 新建：mod\_jk.log 文件。

配置 workers.properties

在 E:\Server\Microthink-Apache\conf\workers.properties 文件，添加以下内容

workers.tomcat\_home=E:\Server\apache-tomcat-6.0.16 #让 mod\_jk 模块知道 Tomcat 的位置

workers.java\_home=E:\Server\jdk1.6.0\_05 #让 mod\_jk 模块知道 jre 的位置

ps=\

worker.list=ajp13 #模块版本

worker.ajp13.port=8009 #工作端口,若没占用则不用修改

worker.ajp13.host=localhost #本机,若上面的 Apache 主机不为 localhost, 作相应修改

worker.ajp13.type=ajp13 #类型

worker.ajp13.lbfactor=1 #代理数,不用修改

如果要想实现负载均衡的话就可以在此处配置多个 tomcat 服务器，开启 session 复制就可以了，具体的情况可以 google 搜索一下，很多的配置方法。

## 4.5 虚拟主机的配置

### [1].Apache 虚拟主机配置：

Httpd.conf 文件最后添加

Include E:\Server\Microthink-Apache\conf\vhosts.conf

在 vhost.conf 里面添加：



NameVirtualHost \*:80

#网站 1

<VirtualHost \*:80>

ServerAdmin web1@microthink.cn

DocumentRoot F:\WebApp\web1

ServerName web1.microthink.cn

DirectoryIndex index.html index.htm index.jsp

ErrorLog logs\web1-error\_log.txt

CustomLog logs/web1-access\_log.txt common

JkMount /servlet/\* ajp13

JkMount /\*.jsp ajp13

JkMount /\*.do ajp13

</VirtualHost>

#网站 2

<VirtualHost \*:80>

ServerAdmin webmaster@web1.web.mt

DocumentRoot F:\WebApp\web2

ServerName web2.microthink.cn

DirectoryIndex index.html index.htm index.jsp

ErrorLog logs\web2-error\_log.txt

CustomLog logs/web2-access\_log.txt common

JkMount /servlet/\* ajp13

JkMount /\*.jsp ajp13

JkMount /\*.do ajp13

</VirtualHost>

添加完成后可以使用你的域名访问，如果提示没有访问权限，修改 Httpd.conf 里面的：

<Directory />

Options FollowSymLinks

AllowOverride None

Order deny,allow

deny from all

Satisfy all

</Directory>

把 allow from all 修改成为 allow from all。

## [2].tomcat 配置

修改 D:\java\apache-tomcat-6.0.20\conf\server.Xml

<Engine name="Catalina" defaultHost="localhost">

<Realm className="org.apache.catalina.realm.UserDatabaseRealm"

resourceName="UserDatabase"/>

```
<Host name="localhost" appBase="webapps"
```

```
    unpackWARs="true" autoDeploy="true"
```

```
    xmlValidation="false" xmlNamespaceAware="false">
```

```
    <Context path="" docBase="F:\WebApp" reloadable="true"
```

```
crossContext="true"/>
```

```
</Host>
```

```
<Host name="web1.microthink.cn" appBase="webapps"
```

```
    unpackWARs="true" autoDeploy="true"
```

```
    xmlValidation="false" xmlNamespaceAware="false">
```

```
    <Context path="" docBase="F:\WebApp\web1" reloadable="true"
```

```
crossContext="true"/>
```

```
</Host>
```

```
<Host name="web2.microthink.cn" appBase="webapps"
```

```
    unpackWARs="true" autoDeploy="true"
```

```
    xmlValidation="false" xmlNamespaceAware="false">
```

```
    <Context path="" docBase="F:\WebApp\web2" reloadable="true"
```

```
crossContext="true"/>
```

</Host>

</Engine>

### [3]将 tomcat 注册成为系统服务

```
cmd->E:\Server\apache-tomcat-6.0.16\bin
```

```
service install
```

这里如果提示：Windows 不能在 本地计算机 启动 Apache Tomcat。有关更多信息，查阅系统事件日志。如果这是非 Microsoft 服务，请与服务厂商联系，并参考特定服务错误代码 1。将 JDK 中 BIN 目录下的 msxcr71.dll 复制到 TOMCAT 的 BIN 下就可以解决了。

### [4]指定访问

在 vhosts.Conf 配置中，我们这样指定了 JkMount /\*.jsp ajp13，意思就是把以 jsp 为后缀的访问都交给 tomcat 处理，然而 Grails 全部采用 URL 映射的访问方式，在这里我们就采用排除法，把一些静态文件交给 apache 处理，其余的全部交给 tomcat 处理,这样的话就直接把刚才的配置修改成为：

```
JkMount /* ajp13
```

```
JkUnMount /*.gif ajp13
```

```
JkUnMount /*.jpg ajp13
```

```
JkUnMount /*.png ajp13
```

```
JkUnMount /*.css ajp13
```

```
JkUnMount /*.js ajp13
```

```
JkUnMount /*.htm ajp13
```

```
JkUnMount /*.html ajp13
```

JkUnMount /\*.swf ajp13

接下来我们的整个服务器环境就配置完成了，为了检查这样的配置是否正确，我们可以把 tomcat 关闭掉，然后单独访问一个图片文件或者静态文件，一样的可以访问，因为还有 apache 存在。

## 第 5 章 总结

写到最后，最初本来计划认真的扩展一些基础知识讲解，但是后来考虑目前的入门教程国内已经有一本 infoq 翻译的教程：《Grails 入门指南》，而中文的 Grails API 也讲解得比较详细，所以就不再过多的讲解相关开发技术，而只是简单的做个引导工作，所以很多地方讲得都比较浅。

而学习新的东西，更多的需要靠实践去操作，所以需要大家更多的努力，目前国内资料还是比较缺乏，出于自己的英文水平问题，所以一直都不敢翻译去误导大家，当自己真正理解的时候我会慢慢的把一些经验用自己的方式跟大家分享出来。也希望大家有好的经验和问题多上论坛交流：

<http://groovycn.5d6d.com> .