

**BOS物流项目**

# 教学计划

1、Quartz概述

* + Quartz介绍和下载
  + 入门案例
  + Quartz执行流程
  + cron表达式

2、在BOS项目中使用Quartz创建定时任务

3、在BOS项目中使用JavaMail发送邮件

4、HighCharts概述

* + HighCharts介绍
  + 在线演示

5、基于HighCharts实现区域分区分布图

# Quartz概述

## quartz介绍和下载

官网：http://www.quartz-scheduler.org/

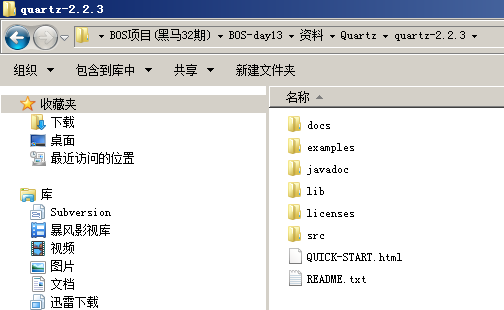


Quartz是OpenSymphony开源组织在Job scheduling领域又一个开源项目，它可以与J2EE与J2SE应用程序相结合也可以单独使用。Quartz可以用来创建简单或为运行十个，百个，甚至是好几万个Jobs这样复杂的程序。Jobs可以做成标准的Java组件或 EJBs。Quartz的最新版本为Quartz 2.2.3。

下载压缩包：

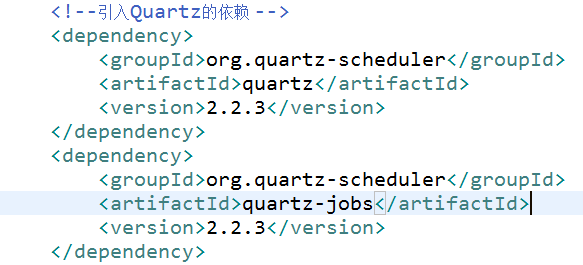


解压压缩包：

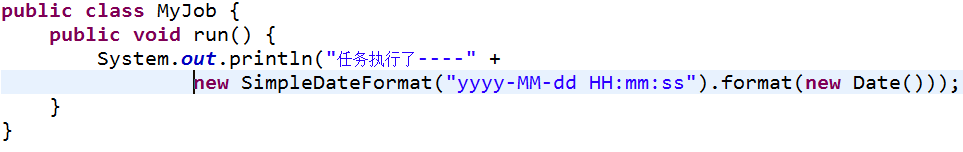


## 入门案例

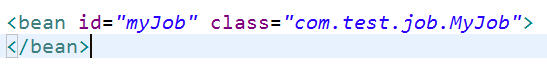
第一步：创建maven工程，导入spring和quartz相关依赖



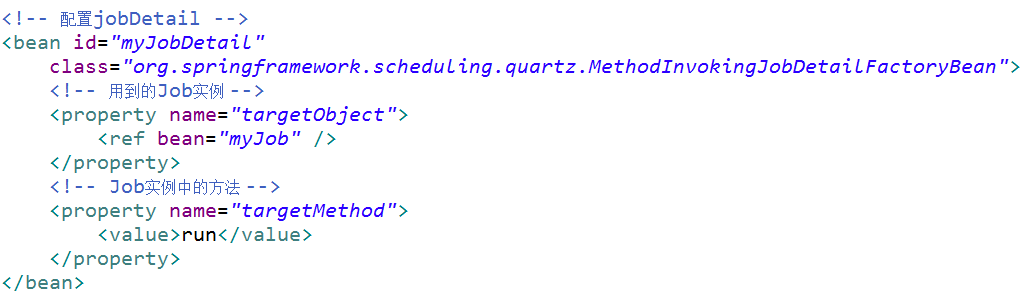
第二步：创建任务类



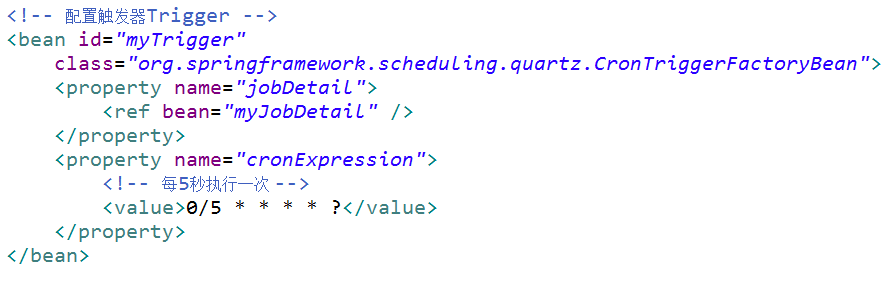
第三步：在spring配置文件中配置任务类



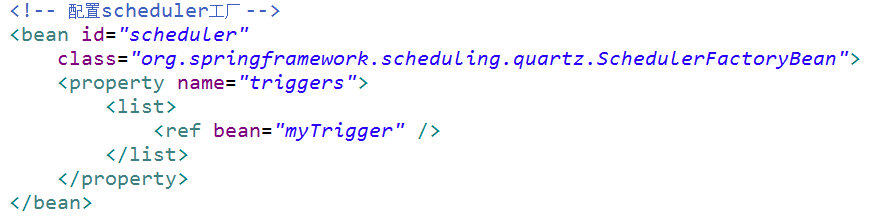
第四步：在spring配置文件中配置JobDetail



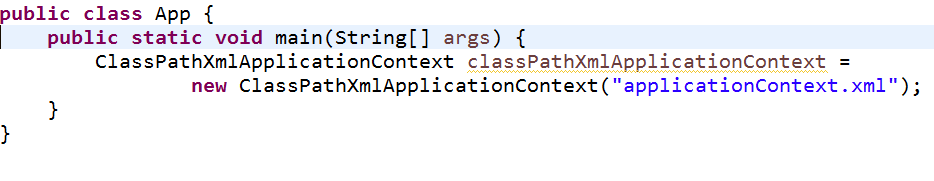
第五步：在spring配置文件中配置触发器



第六步：在spring配置文件中配置**scheduler**

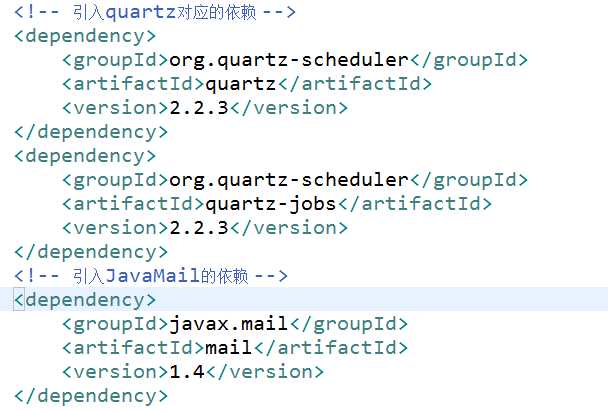


第七步：加载spring配置文件，创建spring工厂



# 在BOS项目中引入quartz

第一步：在pom.xml中引入quartz和JavaMail的依赖



第二步：提供一个作业类，用于为系统管理员发送邮件

**public** **class** MailJob {

@Resource

**private** IWorkbillDao workbillDao;

**private** String username;//发件人的邮箱账号

**private** String password;//密码

**private** String smtpServer;//服务器

**public** String getUsername() {

**return** username;

}

**public** **void** setUsername(String username) {

**this**.username = username;

}

**public** String getPassword() {

**return** password;

}

**public** **void** setPassword(String password) {

**this**.password = password;

}

**public** **void** execute() {

System.***out***.println("要发邮件了。。。" + **new** Date());

**try** {

//查询工单类型为新单的所有工单

List<Workbill> list = workbillDao.findAll();

**if**(**null** != list && list.size() > 0){

**final** Properties mailProps = **new** Properties();

mailProps.put("mail.smtp.host", **this**.getSmtpServer());

mailProps.put("mail.smtp.auth", "true");

mailProps.put("mail.username", **this**.getUsername());

mailProps.put("mail.password", **this**.getPassword());

// 构建授权信息，用于进行SMTP进行身份验证

Authenticator authenticator = **new** Authenticator() {

**protected** PasswordAuthentication getPasswordAuthentication() {

// 用户名、密码

String userName = mailProps.getProperty("mail.username");

String password = mailProps.getProperty("mail.password");

**return** **new** PasswordAuthentication(userName, password);

}

};

// 使用环境属性和授权信息，创建邮件会话

Session mailSession = Session.*getInstance*(mailProps, authenticator);

**for**(Workbill workbill : list){

// 创建邮件消息

MimeMessage message = **new** MimeMessage(mailSession);

// 设置发件人

InternetAddress from = **new** InternetAddress(mailProps.getProperty("mail.username"));

message.setFrom(from);

// 设置收件人

InternetAddress to = **new** InternetAddress("test@itcast.cn");

message.setRecipient(RecipientType.***TO***, to);

// 设置邮件标题

message.setSubject("系统邮件：新单通知");

// 设置邮件的内容体

message.setContent(workbill.toString(), "text/html;charset=UTF-8");

// 发送邮件

Transport.*send*(message);

}

}

} **catch** (Exception ex) {

ex.printStackTrace();

}

}

**public** String getSmtpServer() {

**return** smtpServer;

}

**public** **void** setSmtpServer(String smtpServer) {

**this**.smtpServer = smtpServer;

}

}

第三步：在spring配置文件中配置

<!-- 注册自定义作业类 -->

<bean id=*"myJob"* class=*"com.itheima.jobs.MailJob"*>

<property name=*"username"* value=*"itcast\_server@126.com"*/>

<property name=*"password"* value=*"147963qP"*/>

<property name=*"smtpServer"* value=*"smtp.126.com"*/>

</bean>

<!-- 配置JobDetail -->

<bean id=*"jobDetail"* class=*"org.springframework.scheduling.quartz.MethodInvokingJobDetailFactoryBean"*>

<!-- 注入目标对象 -->

<property name=*"targetObject"* ref=*"myJob"*/>

<!-- 注入目标方法 -->

<property name=*"targetMethod"* value=*"execute"*/>

</bean>

<!-- 配置触发器 -->

<bean id=*"myTrigger"* class=*"org.springframework.scheduling.quartz.CronTriggerFactoryBean"*>

<!-- 注入任务详情对象 -->

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

<!-- 注入cron表达式，通过这个表达式指定触发的时间点 -->

<property name=*"cronExpression"*>

<value>0/5 \* \* \* \* ?</value>

</property>

</bean>

<!-- 配置调度工厂 -->

<bean id=*"schedulerFactoryBean"* class=*"org.springframework.scheduling.quartz.SchedulerFactoryBean"*>

<!-- 注入触发器 -->

<property name=*"triggers"*>

<list>

<ref bean=*"myTrigger"*/>

</list>

</property>

</bean>

# Highcharts

## Highcharts简介

Highcharts 是一个用纯[JavaScript](http://baike.baidu.com/view/16168.htm" \t "_blank)编写的一个图表库， 能够很简单便捷的在web网站或是web应用程序添加有交互性的图表，并且免费提供给个人学习、个人网站和非商业用途使用。HighCharts支持的图表类型有曲线图、区域图、柱状图、饼状图、散状点图和综合图表。

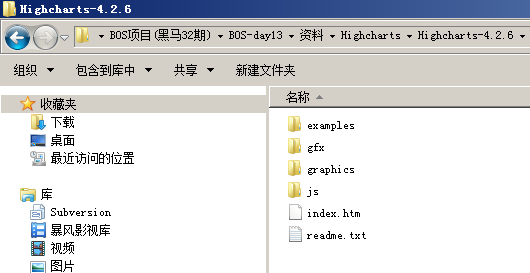
基于jQuery开发一个图形报表工具插件

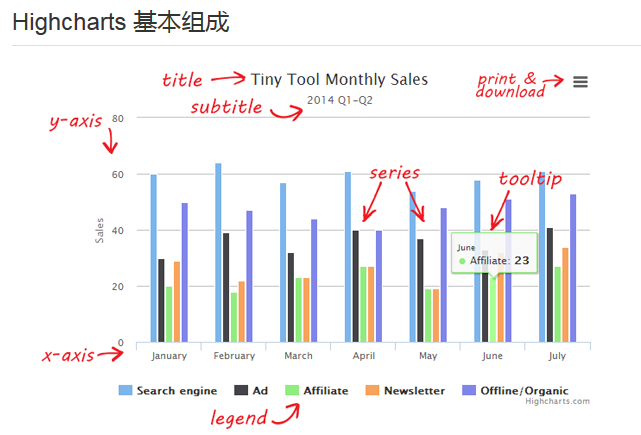
官网：http://www.hcharts.cn/

下载zip开发文档：



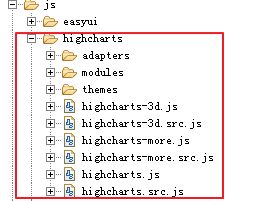
解压zip文件：





## 入门案例

第一步：将Highcharts相关资源文件复制到项目中



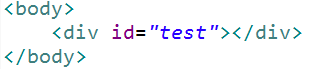
第二步：在页面中引入相关js文件

<script type=*"text/javascript"* src=*"*${pageContext.request.contextPath }*/js/jquery-1.8.3.js"*></script>

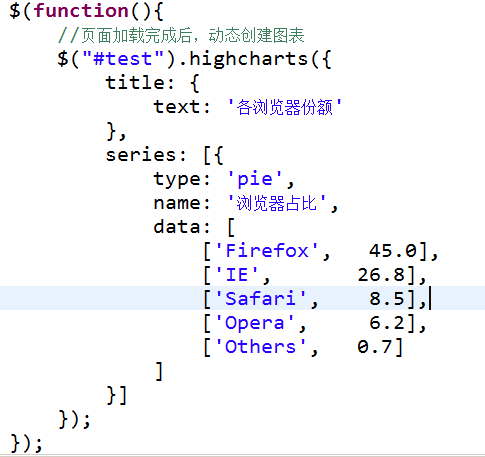
<script src=*"*${pageContext.request.contextPath }*/js/highcharts/highcharts.js"*></script>

<script src=*"*${pageContext.request.contextPath }*/js/highcharts/modules/exporting.js"*></script>

第三步：在页面中提供一个div，并指定id属性



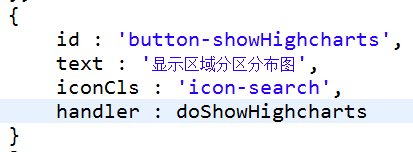
第四步：调用Highcharts提供的方法，动态创建图表

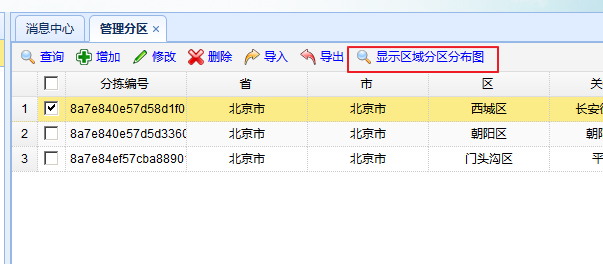


# 在BOS项目中展示区域分区分布图

第一步：在subarea.jsp页面中引入Highcharts资源文件

第二步：在jsp页面中提供按钮，并提供div窗口，在这个窗口中展示图表





第三步：定义function

**function** doShowHighcharts(){

$("#showSubareaWindow").window("open");

//页面加载完成后，动态创建图表

$.post("subareaAction\_findSubareasGroupByProvince.action",**function**(data){

$("#test").highcharts({

title: {

text: '区域分区分布图'

},

series: [{

type: 'pie',

name: '区域分区分布图',

data: data

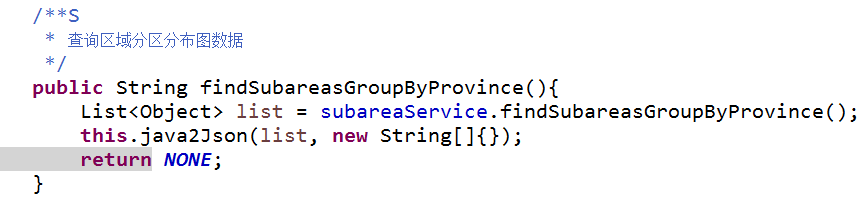
}]

});

});

}

第四步：在服务端Action中提供方法



Dao代码：

@Repository

**public** **class** SubareaDaoImpl **extends** BaseDaoImpl<Subarea> **implements** ISubareaDao {

**public** List<Object> findSubareasGroupByProvince() {

String hql = "SELECT r.province ,count(\*) FROM Subarea s LEFT OUTER JOIN s.region r Group BY r.province";

**return** (List<Object>) **this**.getHibernateTemplate().find(hql);

}

}

