任务调度

凯盛软件

TimerTask Timer

凯盛软件

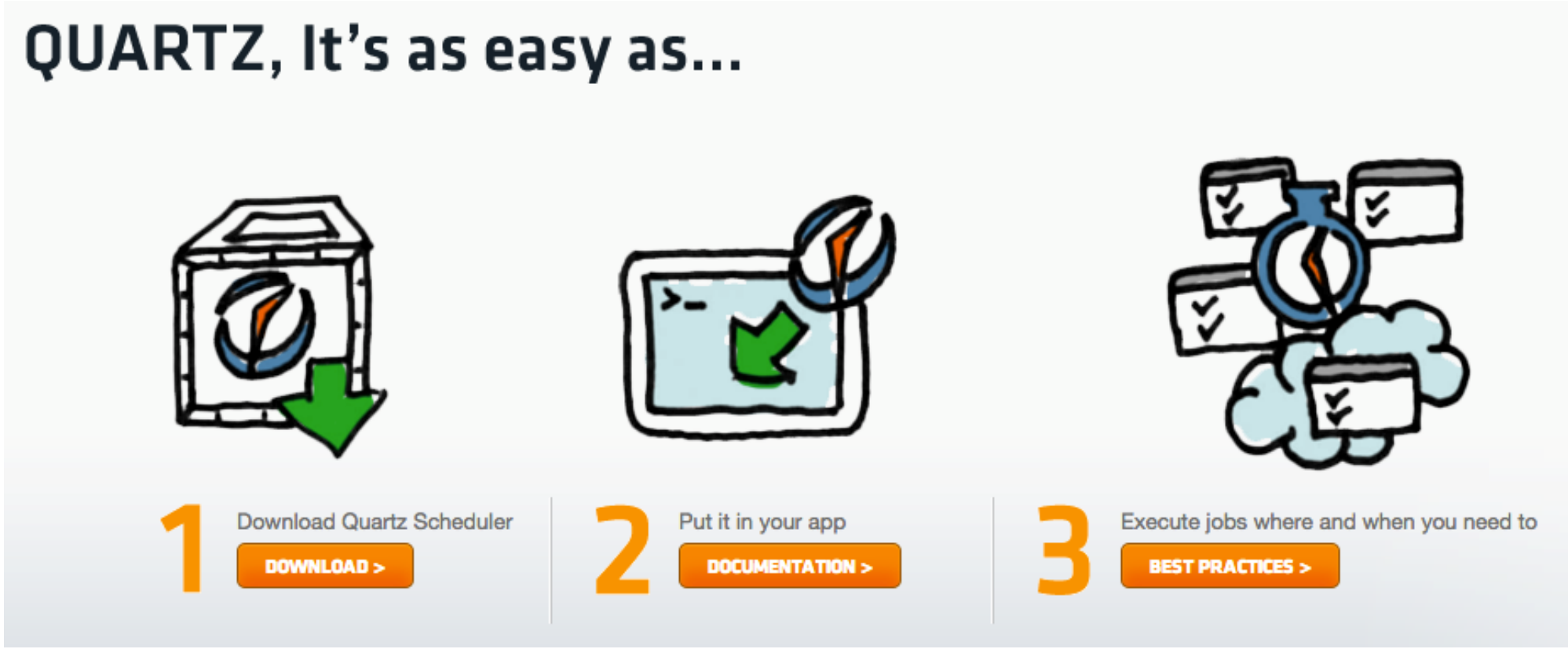
```
import java.util.TimerTask;

public class MyTimerTask extends TimerTask{

        @Override
        public void run() {
                System.out.println("Hello,Message");
        }
}

//每1000毫秒调用一次
Timer timer = new Timer();
timer.schedule(new MyTimerTask(), 0, 1000);
```

- 每天晚上12：30备份数据库
- 每周五下午2：00给全部人员发邮件通知开会
- 每月1号1：00发送上个月的统计数据

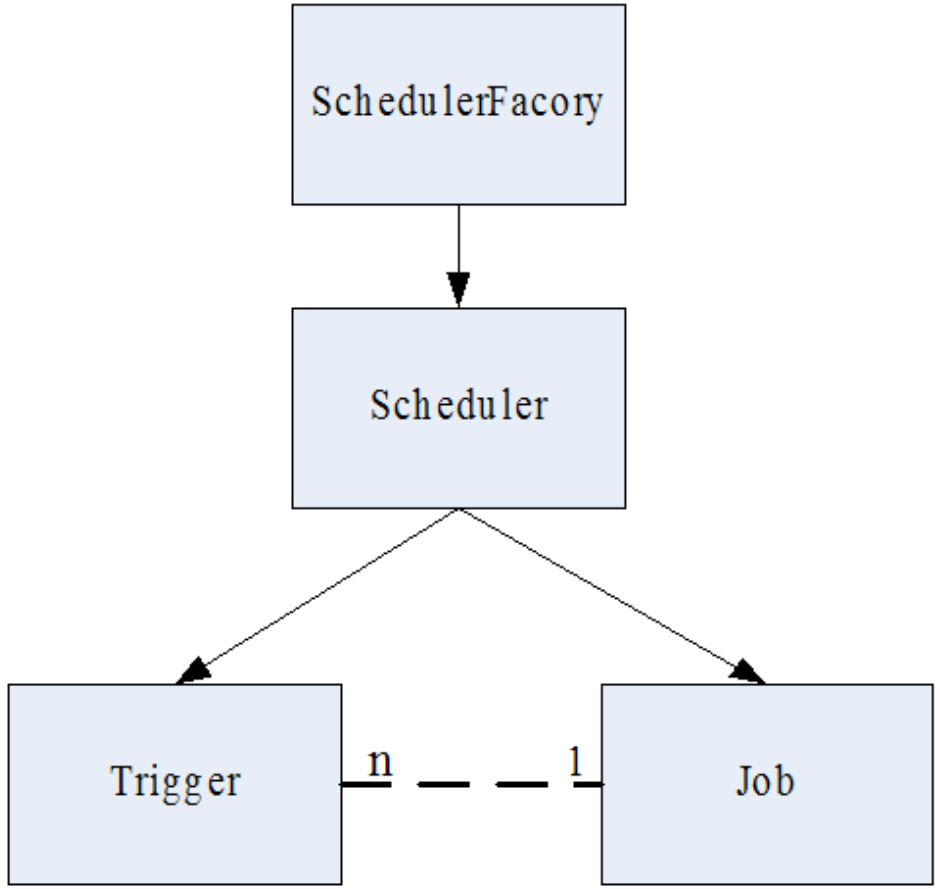
Quartz

凯盛软件

http://quartz-scheduler.org/



核心组件



QuartzJob

凯盛软件

```
import org.quartz.Job;
import org.quartz.JobExecutionContext;
import org.quartz.JobExecutionException;

public class MyQuartzJob implements Job{

        public void execute(JobExecutionContext context) throws JobExecutionException {
                System.out.println("Hello,Quartz");
        }
}
```

SimpleTrigger

凯盛软件

```
JobDetail detail = JobBuilder.newJob(MyQuartzJob.class).build();

SimpleScheduleBuilder ssb = SimpleScheduleBuilder.simpleSchedule();
ssb.withIntervalInSeconds(5);//间隔5秒钟
ssb.repeatForever();//永远执行下去
Trigger simpleTrigger = TriggerBuilder.newTrigger().withSchedule(ssb).build();

Scheduler scheduler = new StdSchedulerFactory().getScheduler();
scheduler.scheduleJob(detail, simpleTrigger);
scheduler.start();
```

CronTrigger

凯盛软件

```
JobDetail detail = JobBuilder.newJob(MyQuartzJob.class).build();

ScheduleBuilder ssb =  CronScheduleBuilder.cronSchedule("0/5 * * * * ? *");
Trigger cronTrigger = TriggerBuilder.newTrigger().withSchedule(ssb).build();

Scheduler scheduler = new StdSchedulerFactory().getScheduler();
scheduler.scheduleJob(detail, cronTrigger);
scheduler.start();
```

Cron 表达式

凯盛软件

在线生成器

- http://www.g2room.com/subject/cron/
- http://www.cronmaker.com/

在Spring中使用

凯盛软件

在Spring版本3.x中不支持Quartz2.x，需要将Quartz的版本将为1.x

定义任务类

```
public class MySpringQuartzJob {

    public void sayHello() {
        System.out.println("Hello,Spring + Quartz");
    }
}
```

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context-support</artifactId>
</dependency>
```

配置applicationContext.xml

```
<!--定义任务类-->
<bean id="mySpringQuartz" class="com.kaishengit.MySpringQuartzJob"/>

<!--定义调用对象和方法-->
<bean id="jobDetail" class="org.springframework.scheduling.quartz.MethodInvokingJobDetailFactoryBean">
    <property name="targetObject" ref="mySpringQuartz"/>
    <property name="targetMethod" value="sayHello"/>
</bean>

<!--定义Trigger-->
<bean id="cronTriggerBean" class="org.springframework.scheduling.quartz.CronTriggerBean">
    <property name="jobDetail" ref="jobDetail"/>
    <property name="cronExpression" value="0/5 * * * * ? *"/>
</bean>
```

```
<!--调度器-->
<bean id="scheduler" lazy-init="false" class="org.springframework.scheduling.quartz.SchedulerFactoryBean">
    <property name="triggers">
        <list>
            <ref bean="cronTriggerBean"/>
        </list>
    </property>
</bean>
```

将任务调度持久化

凯盛软件

1. 下载Quartz的源代码包，执行docs/Tables/xxx.sql ， 在数据库中创建表。重要的是qrtz_job_details和qrtz_triggler两个表，分别存放任务和触发器
2. 创建Quartz配置文件，quartz.properties

```
#表前缀
org.quartz.jobStore.tablePrefix = QRTZ_
#集群模式
org.quartz.jobStore.isClustered = true
```

3. 修改Spring中JobDetails的创建方式

```
<bean id="jobDetail" class="org.springframework.scheduling.quartz.JobDetailFactoryBean">
    <!--固定重复的Job-->
    <property name="durability" value="true"/>
    <!--Job类的完全限定名称 Job接口的实现类-->
    <property name="jobClass" value="com.kaishengit.quartz.jobs.WeixinNotifyJob"/>
    <!--JobMapDate 需要时设定-->
    <property name="jobDataAsMap">
        <map>
            <entry key="to" value="1"/>
            <entry key="message" value="init message ........."/>
        </map>
    </property>
</bean>
```

4. 修改SchedulerFactoryBean的创建

```
<!--定义调度器工厂-->
<bean id="stdScheduler" class="org.springframework.scheduling.quartz.SchedulerFactoryBean">
    <!--数据源-->
    <property name="dataSource" ref="dataSource"/>
    <!--事务管理器-->
    <property name="transactionManager" ref="transactionManager"/>
    <!--配置文件-->
    <property name="configLocation" value="classpath:quartz.properties"/>
    <!--是否覆盖已有的job-->
    <property name="overwriteExistingJobs" value="true"/>
    <property name="triggers">
        <list>
            <ref bean="cronTrigger"/>
        </list>
    </property>
</bean>
```

动态添加任务

凯盛软件

```
@Autowired
private SchedulerFactoryBean schedulerFactoryBean;


JobDataMap dataMap = new JobDataMap();
dataMap.putAsString("to",task.getAccountId());
dataMap.put("message",task.getTitle());

Scheduler scheduler = schedulerFactoryBean.getScheduler();

//jobDetail
JobDetail jobDetail = JobBuilder.newJob(WeixinNotifyJob.class)
        .withIdentity(new JobKey("account:"+task.getAccountId()+":"+task.getId(),"weixinGroup"))
        .setJobData(dataMap).build();
```

```
//字符串日期格式转换为JodaTime的DateTime类对象
DateTimeFormatter formatter = DateTimeFormat.forPattern("yyyy-MM-dd HH:mm");
DateTime dateTime = formatter.parseDateTime(task.getRemindTime());
//根据日期生成cron表达式
Cron cron = CronBuilder.cron(CronDefinitionBuilder.instanceDefinitionFor(CronType.QUARTZ))
        .withYear(on(dateTime.getYear()))
        .withMonth(on(dateTime.getMonthOfYear()))
        .withDoM(on(dateTime.getDayOfMonth()))
        .withHour(on(dateTime.getHourOfDay()))
        .withMinute(on(dateTime.getMinuteOfHour()))
        .withSecond(on(dateTime.getSecondOfMinute()))
        .withDoW(questionMark())
        .instance();

String cronExpression = cron.asString();
```

https://github.com/jmrozanec/cron-utils

```
CronScheduleBuilder scheduleBuilder = CronScheduleBuilder.cronSchedule(cronExpression);
Trigger trigger = TriggerBuilder.newTrigger().withSchedule(scheduleBuilder).build();

try {
    scheduler.scheduleJob(jobDetail, trigger);
    scheduler.start();
} catch (SchedulerException ex) {
    throw  new ServiceException("添加定时任务异常",ex);
}
```

动态删除任务

凯盛软件

```
Scheduler scheduler = schedulerFactoryBean.getScheduler();
try {
    scheduler.deleteJob(new JobKey("account:" + task.getAccountId()+":"+task.getId(), "weixinGroup"));
} catch (SchedulerException ex) {
    throw new ServiceException("删除调度器任务异常",ex);
}
```