Stripes Framework
Technology Preview

Rodrigo Malara

Araraquara Solution Centre - Brazil

Sun microsystems

Agenda

- Introdução
- Objetivos
- Introdução à execução de Jobs
- Visão Geral das Capacidades do Quartz
- Primeiro Exemplo
- Introdução ao Spring Framework
- Segundo Exemplo – Integração com o Spring
- Terceiro Exemplo – Injeção de Dependência com Spring
- Comparação com outras soluções
- Fechamento
- Q & A

Introdução

Rodrigo D. Malara

http://www.linkedin.com/in/rodrigomalara

Linux User ID 137855 – desde setembro de 1997

Engenharia de Computação – DC/UFSCar (2000)
Mestrado em Sistemas Distribuídos – IFSC/USP (2005)
Certificações Oracle OCJP, OCWCD, OCBCD, OCEA (step 1)

Coordenador dos Cursos de Computação da UNIARA desde 2004
Docente de disciplinas relacionadas a Computação desde 2003.
Arquiteto e sócio da Agnitia Soluções por 3 anos
Engenheiro de Sistemas - Nortel Networks por 4 anos
Software Specialist Senior na HP por 6 anos
Sócio-Diretor da Gourmex/Delivoro a 5 anos
www.gourmex.com / www.delivoro.com.br / www.gmxcheckout.com.br

Objetivos e Premissas

Demonstrar o Stripes Framework

- Usos, funcionalidades, configuração básica
- Integração com framework Spring (IoC container)
- Exibir alguns exemplos e realizar exercícios

- Premissa básica: a audiência conhece:
  - Java
  - Organização de uma aplicação WEB em Java
  - Eclipse EE IDE disponível.
    - Instalação e configuração do Tomcat Servlet Container
    - Não é necessário nenhum plugin extra.
    - Usaremos o template ‘Dynamic Web Project’

Desenvolvimento WEB

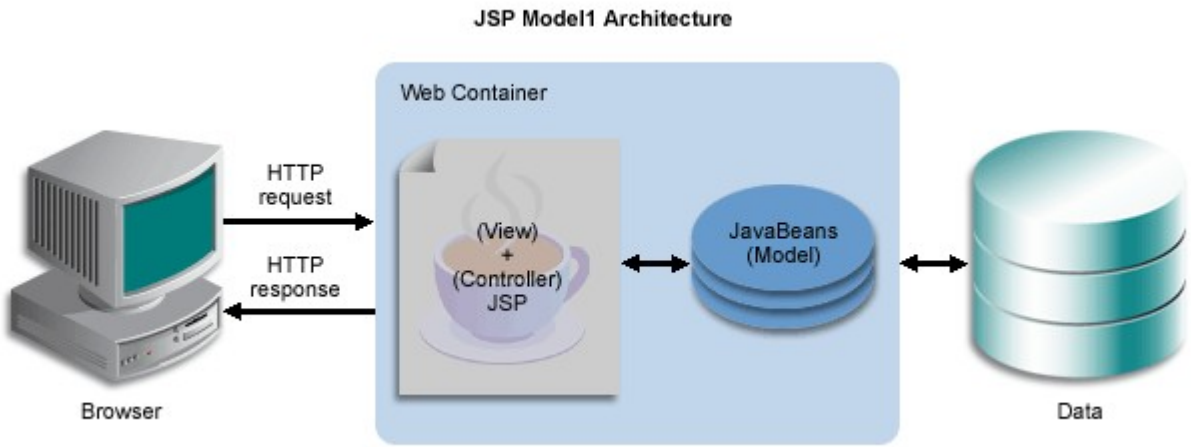
- Lado do cliente: Browser
  - HTML, Javascript, CSS
    - Captura de informações através de formulários
    - Parametros na URL
- Lado do servidor – Java Servlet Container
  - Servlets – não usaremos diretamente
  - JSP – importante e atualmente complexo
    - Scriptlet – Código Java embutido na página
      - » Estilo ASP.NET e PHP – Evitar
    - Expression Language
      - » Exibição de dados dinâmicos e aritmética básica
    - Taglibs
      - » Bibliotecas de funções com sintaxe parecida com HTML

Exemplo de JSP

```
<s:form beanclass="${actionBean.class}" focus="almExtServer.serverName" >
  <s:hidden name="editing" />
  <s:hidden name="menuStyleId" />
  <s:hidden name="entity.serverKey" />
  <table>
    <tr>
      <td><s:label for="almExtServer.serverName" /></td>
      <td>
        <s:text name="entity.serverName" size="30" maxlength="50"/>
        <s:errors field="entity.serverName" />
      </td>
    </tr>
  </table>
```

Arquitetura

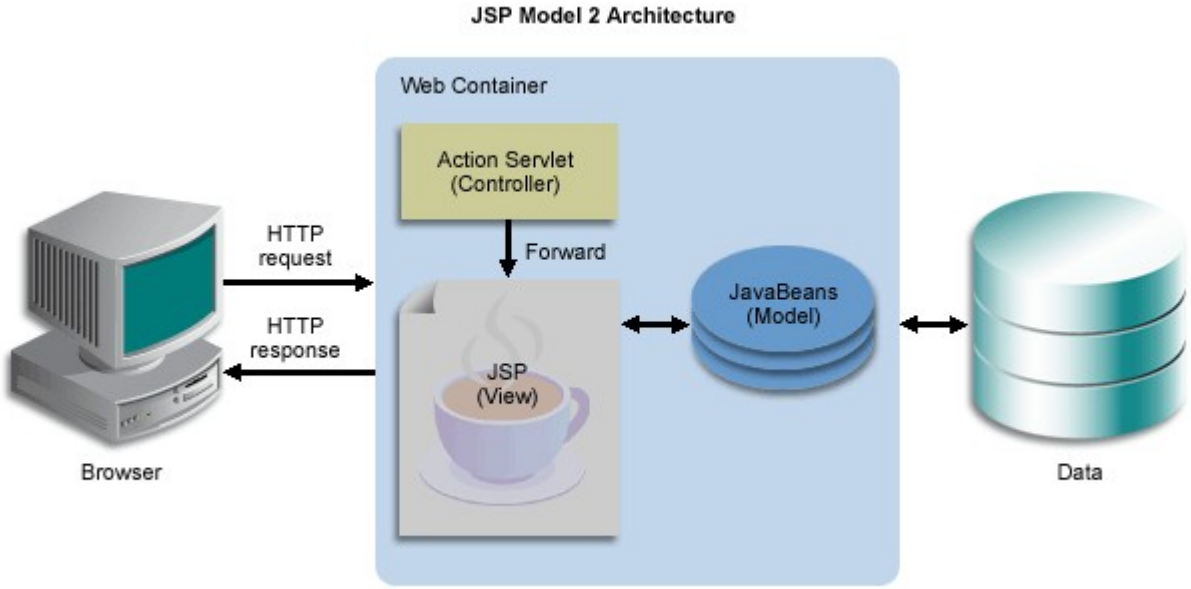
- Model 1
  - Arquitetura usada no início e em linguagens script
    - PHP, ASP.NET, PERL



- Problemas da arquitetura "Model 1"
  - Requisitos Não-funcionais
    - Manutenibilidade, Escalabilidade, Segurança, etc...

Arquitetura (2)

- Arquitetura frameworks Request/Response
  - Struts, Spring MVC, Stripes



Stripes Framework

- Framework para desenvolvimento WEB
  - Esconder complexidade e aumentar a eficiência
  - Disponibilizar soluções padronizadas para problemas conhecidos
- Objetivos (extraidos do web site)
  - Tornar o desenv. de app. Web em Java simples
  - Prover soluções simples e poderosas para problemas simples
  - Aprendizado em 30 minutos para um novo desenvolvedor
    - Que já desenvolva aplicações WEB em Java
  - Fazer com que seja muito fácil estender sem ter que reconfigurar tudo
- Verifique em www.stripesframework.org

Stripes Framework

- Uso massivo de Anotações (annotations)
  - Evitar configurações em arquivos XML
  - Configurações apenas no arquivo web.xml

- Open-source e livre
  - Licença GPL

Configuração do Ambiente

- Familiarize-se com o site – https://stripesframework.atlassian.net
  - Encontre o produto para download
  - Link para documentação
    - Traduza com o Google se necessário
- Faça o download do Stripes
- Configuração do Apache Tomcat 8
  - http://tomcat.apache.org
- Faça o download e descompacte dentro do workspace do Eclipse
- Faça com que o Eclipse “reconheça” o seu Tomcat
- Crie uma instância do Tomcat para usar

Primeiro exemplo – Estrutura geral

- Configuração Inicial
  - Crie um projeto vazio Java Web project
  - Coloque os arquivos jar files no diretório WEB-INF/lib
    - cos.jar
    - commons-logging-1.1.jar
    - stripes.jar

Configuração do Stripes – web.xml

```
<filter-name>StripesFilter</filter-name>
<filter-class>net.sourceforge.stripes.controller.StripesFilter</filter-class>
<init-param>
<param-name>ActionResolver.Packages</param-name>
<param-value>br.com.uniara.action</param-value>
</init-param>
 </filter>

<filter-mapping>
<filter-name>StripesFilter</filter-name>
<url-pattern>*.jsp</url-pattern>
<dispatcher>REQUEST</dispatcher>
</filter-mapping>
```

Configuração do Stripes – web.xml (2)

```
<filter-mapping>
<filter-name>StripesFilter</filter-name>
<servlet-name>StripesDispatcher</servlet-name>
<dispatcher>REQUEST</dispatcher>
</filter-mapping>

<servlet>
<servlet-name>StripesDispatcher</servlet-name>
<servlet-class>net.sourceforge.stripes.controller.DispatcherServlet</servlet-class>
<load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
<servlet-name>StripesDispatcher</servlet-name>
<url-pattern>*.action</url-pattern>
</servlet-mapping>
```

Próximas tarefas

- - Calculadora com Stripes
  - Tags para campos
  - Validador required
  - Submissao
  - Exibicao do resultado usando EL
- Outros validadores

Primeiro exemplo – Job e config. web app

```
public class HelloWorldJob implements Job {

  public void execute(JobExecutionContext arg0) throws JobExecutionException {
    SimpleDateFormat fmt = new SimpleDateFormat("MM-dd-yyyy HH:mm:ss");
    System.out.println(fmt.format(new Date()) +" - Hello World !");
  }
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <listener>
   <listener-class>
      SchedulerContextListener
    </listener-class>
  </listener>

  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>
</web-app>
```

Primeiro exemplo - Logs

```
Jun 25, 2009 8:43:10 PM org.quartz.core.QuartzScheduler <init>
INFO: Quartz Scheduler v.1.6.5 created.
Jun 25, 2009 8:43:10 PM org.quartz.simpl.RAMJobStore initialize
INFO: RAMJobStore initialized.
Jun 25, 2009 8:43:10 PM org.quartz.impl.StdSchedulerFactory instantiate
INFO: Quartz scheduler 'DefaultQuartzScheduler' initialized from default :
Jun 25, 2009 8:43:10 PM org.quartz.impl.StdSchedulerFactory instantiate
INFO: Quartz scheduler version: 1.6.5
Jun 25, 2009 8:43:10 PM org.quartz.core.QuartzScheduler start
INFO: Scheduler DefaultQuartzScheduler_$_NON_CLUSTERED started.
Jun 25, 2009 8:43:10 PM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8080
06-25-2009 20:43:10 - Hello World !
Jun 25, 2009 8:43:10 PM org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8009
Jun 25, 2009 8:43:10 PM org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/32  config=null
Jun 25, 2009 8:43:10 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 724 ms
06-25-2009 20:43:11 - Hello World !
06-25-2009 20:43:12 - Hello World !
06-25-2009 20:43:13 - Hello World !
Jun 25, 2009 8:43:14 PM org.apache.coyote.http11.Http11Protocol pause
INFO: Pausing Coyote HTTP/1.1 on http-8080
06-25-2009 20:43:14 - Hello World !
Jun 25, 2009 8:43:15 PM org.apache.catalina.core.StandardService stop
INFO: Stopping service Catalina
Jun 25, 2009 8:43:15 PM org.quartz.core.QuartzScheduler shutdown
INFO: Scheduler DefaultQuartzScheduler_$_NON_CLUSTERED shutting down.
```

Web Server Log saída exibindo:

1 – Inicialização Scheduler

2 – Execução Job

3 – Shutdown Scheduler

Getting more Flexibility - Cron Triggers

- Definir Triggers usando expressão Cron
  - São expressões usadas nos Unix
  - Muito poderosas e conhecidas na área de IT

```
try {
  scheduler = sf.getScheduler();
  Trigger trigger = new CronTrigger("1secTrigger",
      "defaultTGroup", "* * * * * ?");
  JobDetail jobDetail = new JobDetail("helloWorldJob",
      "defaultGroup", HelloWorldJob.class);
  scheduler.scheduleJob(jobDetail, trigger);
  scheduler.start();
} catch (Exception e) {
```

  - Para um tutorial consulte

http://www.opensymphony.com/quartz/wikidocs/CronTriggers%20Tutorial.html

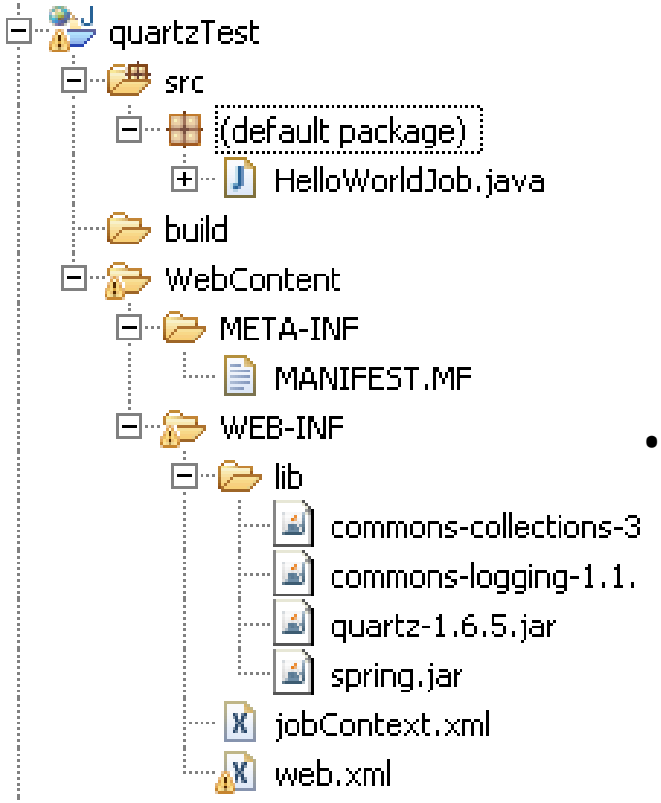
  - Alguns exemplos foram copiados e colados nos slides finais

Tuning Quartz options – quartz.properties

- Coloque o arquivo quartz.properties no classpath
  - Ex. Pasta src do seu projeto

```
# Default Properties file for use by StdSchedulerFactory
# to create a Quartz Scheduler Instance, if a different
# properties file is not explicitly specified.
org.quartz.scheduler.instanceName = DefaultQuartzScheduler
org.quartz.scheduler.rmi.export = false
org.quartz.scheduler.rmi.proxy = false
org.quartz.scheduler.wrapJobExecutionInUserTransaction = false
org.quartz.threadPool.class = org.quartz.simpl.SimpleThreadPool
org.quartz.threadPool.threadCount = 10
org.quartz.threadPool.threadPriority = 5
org.quartz.threadPool.threadsInheritContextClassLoaderOfInitializingT
hread = true
org.quartz.jobStore.misfireThreshold = 60000
org.quartz.jobStore.class = org.quartz.simpl.RAMJobStore
```

Segundo Exemplo – Usar Spring Framework



- Configuração Inicial
  - Download Spring 2.5.6 SEC01
    - Zip menor é suficiente (5.8MB)
  - Colocar spring.jar no WEB-INF/lib
  - Remover ou apenas just esquecer nosso SchedulerContextListener
- Nós pretendemos
  - Continuar inicializando o Scheduler quando a aplicação web iniciar
    - Usar SpringBeanJobFactory e SchedulerFactoryBean
    - Vamos codificar apenas o Job
  - Passar um parâmetro para o job (*personName*)
  - Manter rodando de segundo em segundo

Segundo Exemplo - jobContext.xml – 1/2

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans-2.5.xsd">

  <bean name="HelloWorldJobBean"
    class="org.springframework.scheduling.quartz.JobDetailBean">
    <property name="jobClass" value="HelloWorldJob" />
    <property name="jobDataAsMap">
     <map>
      <entry key="personName" value="boss" />
     </map>
    </property>
  </bean>

  <bean id="HelloWorldTrigger"
    class="org.springframework.scheduling.quartz.CronTriggerBean">
    <property name="jobDetail" ref="HelloWorldJobBean" />
    <property name="cronExpression" value="* * * * * ?" />
  </bean>
```

Defining the Job and the Trigger

Segundo Exemplo - jobContext.xml – 2/2

```
<bean name="jobFactory"
  class="org.springframework.scheduling.quartz.SpringBeanJobFactory" />

<bean class="org.springframework.scheduling.quartz.SchedulerFactoryBean">
  <property name="jobFactory">
    <ref bean="jobFactory"/>
  </property>
  <property name="triggers">
      <list>
          <ref bean="HelloWorldTrigger" />
      </list>
  </property>
</bean>

</beans>
```

Creating Job Factory and Scheduler Beans

Segundo Exemplo – Mudanças no web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/jobContext.xml</param-value>
  </context-param>

  <listener>
    <listener-class>
      org.springframework.web.context.ContextLoaderListener
    </listener-class>
  </listener>

  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>
</web-app>
```

Replacing our home made Context Listener by Spring's

Segundo Exemplo - Mudanças no HelloWorldJob

```
public class HelloWorldJob implements Job {

  public void execute(JobExecutionContext ctx) throws JobExecutionException {
    JobDataMap config = ctx.getJobDetail().getJobDataMap();
    String personName = config.getString("personName");

    SimpleDateFormat fmt = new SimpleDateFormat("MM-dd-yyyy HH:mm:ss");
    System.out.println(fmt.format(new Date()) +
        " - Hello World and " + personName);
  }
}
```

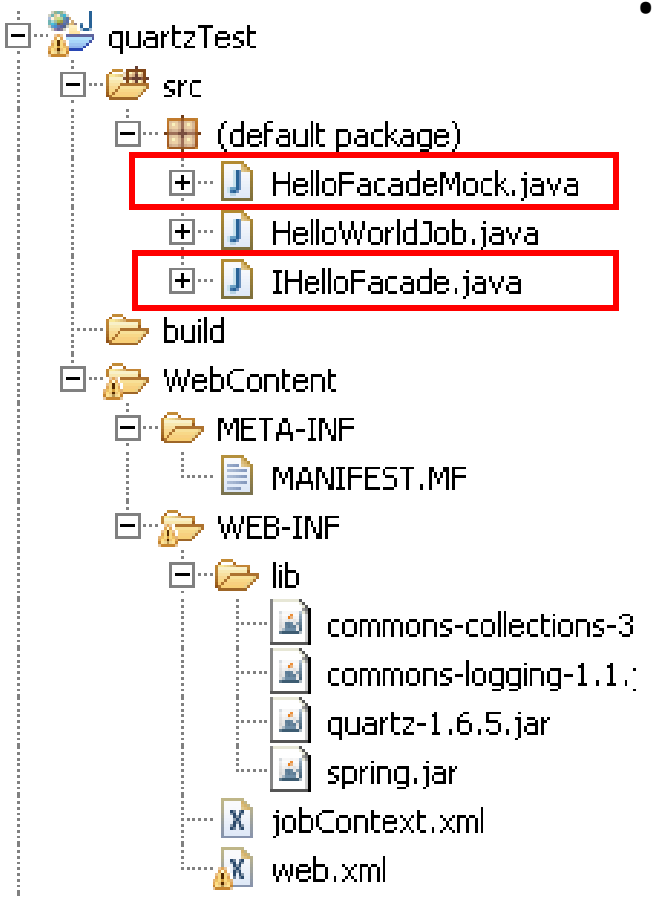
Getting the parameter set on jobContext.xml and printing it

Segundo Exemplo - Log

```
Jun 26, 2009 11:55:14 AM org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
INFO: Loading XML bean definitions from ServletContext resource [/WEB-INF/jobContext.xml]
Jun 26, 2009 11:55:17 AM org.springframework.context.support.AbstractApplicationContext obtainFreshBeanFactory
INFO: Bean factory for application context [org.springframework.web.context.support.XmlWebApplicationContext@9e
Jun 26, 2009 11:55:17 AM org.springframework.beans.factory.support.DefaultListableBeanFactory preInstantiateSin
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListableBeanFactory@e646
Jun 26, 2009 11:55:19 AM org.quartz.core.SchedulerSignalerImpl <init>
INFO: Initialized Scheduler Signaller of type: class org.quartz.core.SchedulerSignalerImpl
Jun 26, 2009 11:55:19 AM org.quartz.core.QuartzScheduler <init>
INFO: Quartz Scheduler v.1.6.5 created.
Jun 26, 2009 11:55:19 AM org.quartz.simpl.RAMJobStore initialize
INFO: RAMJobStore initialized.
INFO: Root WebApplicationContext: initialization completed in 4719 ms
Jun 26, 2009 11:55:19 AM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8080
06-26-2009 11:55:19 - Hello World and boss
06-26-2009 11:55:19 - Hello World and boss
Jun 26, 2009 11:55:19 AM org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8009
Jun 26, 2009 11:55:19 AM org.apache.jk.server.JkMain start
Jun 26, 2009 11:55:29 AM org.apache.coyote.http11.Http11Protocol pause
INFO: Pausing Coyote HTTP/1.1 on http-8080
06-26-2009 11:55:30 - Hello World and boss
Jun 26, 2009 11:55:30 AM org.quartz.core.QuartzScheduler standby
INFO: Scheduler org.springframework.scheduling.quartz.SchedulerFactoryBean#0_$_NON_CLUSTERED paused.
Jun 26, 2009 11:55:30 AM org.quartz.core.QuartzScheduler standby
INFO: Scheduler org.springframework.scheduling.quartz.SchedulerFactoryBean#0_$_NON_CLUSTERED paused.
Jun 26, 2009 11:55:30 AM org.quartz.core.QuartzScheduler shutdown
INFO: Scheduler org.springframework.scheduling.quartz.SchedulerFactoryBean#0_$_NON_CLUSTERED shutdown complete.
Jun 26, 2009 11:55:31 AM org.apache.coyote.http11.Http11Protocol destroy
INFO: Stopping Coyote HTTP/1.1 on http-8080
```

Quartz Startup, job output and shutdown

Final example - Injecting Bean Instance on Job



- We intend to
  - Add IHelloFacade interface
    - 2 methods for generating hello messages
  - Add HelloFacadeMock class
    - Implements IHelloFacade
    - Just enough for testing our configuration
  - Initialize a HelloFacadeMock singleton
  - Inject it into HelloWorldJob for being used

Final Example - IHelloFacade and it's mock

```
public interface IHelloFacade {

  String getHelloMessage(String who, Locale locale);

  String getHelloMessage(String who);
}
```

Interface used By HelloWorldJob

```
public class HelloFacadeMock implements IHelloFacade {

  public String getHelloMessage(String who, Locale locale) {
    return getHelloMessage(who);
  }

  public String getHelloMessage(String who) {
    return "Hello World and " + who;
  }
}
```

Mock which implements IHelloFacade

Final Example - HelloWorldJob changes

```
public class HelloWorldJob implements Job {

  private IHelloFacade helloFacade;

  public void execute(JobExecutionContext ctx) throws JobExecutionException {
    JobDataMap config = ctx.getJobDetail().getJobDataMap();
    String personName = config.getString("personName");

    SimpleDateFormat fmt = new SimpleDateFormat("MM-dd-yyyy HH:mm:ss");
    System.out.println(fmt.format(new Date()) +
        " - " + helloFacade.getHelloMessage(personName));
  }

  public IHelloFacade getHelloFacade() {
    return helloFacade;
  }

  public void setHelloFacade(IHelloFacade helloFacade) {
    this.helloFacade = helloFacade;
  }
}
```

Property to hold Injected reference to the HelloFacadeMock

Invoking method on IHelloFacade

Property getter and setter

Final Example - jobContext.xml changes

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans-2.5.xsd">

  <bean name="HelloFacadeBean" scope="singleton"
    class="HelloFacadeMock">
  </bean>

  <bean name="HelloWorldJobBean"
    class="org.springframework.scheduling.quartz.JobDetailBean">
    <property name="jobClass" value="HelloWorldJob" />
    <property name="jobDataAsMap">
     <map>
      <entry key="personName" value="boss" />
      <entry key="helloFacade">
        <ref bean="HelloFacadeBean"/>
      </entry>
     </map>
    </property>
  </bean>
```

Initializing the Hello World Bean

Injecting the Bean into the Job Bean

Final Example - Output log

```
Jun 26, 2009 11:55:14 AM org.springframework.beans.factory.xml.XmlBeanDefinitionReader loadBeanDefinitions
INFO: Loading XML bean definitions from ServletContext resource [/WEB-INF/jobContext.xml]
Jun 26, 2009 11:55:17 AM org.springframework.context.support.AbstractApplicationContext obtainFreshBeanFactory
INFO: Bean factory for application context [org.springframework.web.context.support.XmlWebApplicationContext@9e
Jun 26, 2009 11:55:17 AM org.springframework.beans.factory.support.DefaultListableBeanFactory preInstantiateSin
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListableBeanFactory@e646
Jun 26, 2009 11:55:19 AM org.quartz.core.SchedulerSignalerImpl <init>
INFO: Initialized Scheduler Signaller of type: class org.quartz.core.SchedulerSignalerImpl
Jun 26, 2009 11:55:19 AM org.quartz.core.QuartzScheduler <init>
INFO: Quartz Scheduler v.1.6.5 created.
Jun 26, 2009 11:55:19 AM org.quartz.simpl.RAMJobStore initialize
INFO: RAMJobStore initialized.
INFO: Root WebApplicationContext: initialization completed in 4719 ms
Jun 26, 2009 11:55:19 AM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8080
06-26-2009 11:55:19 - Hello World and boss
06-26-2009 11:55:19 - Hello World and boss
Jun 26, 2009 11:55:19 AM org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:8009
Jun 26, 2009 11:55:19 AM org.apache.jk.server.JkMain start
Jun 26, 2009 11:55:29 AM org.apache.coyote.http11.Http11Protocol pause
INFO: Pausing Coyote HTTP/1.1 on http-8080
06-26-2009 11:55:30 - Hello World and boss
Jun 26, 2009 11:55:30 AM org.quartz.core.QuartzScheduler standby
INFO: Scheduler org.springframework.scheduling.quartz.SchedulerFactoryBean#0_$_NON_CLUSTERED paused.
Jun 26, 2009 11:55:30 AM org.quartz.core.QuartzScheduler standby
INFO: Scheduler org.springframework.scheduling.quartz.SchedulerFactoryBean#0_$_NON_CLUSTERED paused.
Jun 26, 2009 11:55:30 AM org.quartz.core.QuartzScheduler shutdown
INFO: Scheduler org.springframework.scheduling.quartz.SchedulerFactoryBean#0_$_NON_CLUSTERED shutdown complete.
Jun 26, 2009 11:55:31 AM org.apache.coyote.http11.Http11Protocol destroy
INFO: Stopping Coyote HTTP/1.1 on http-8080
```

Quartz Startup,
job output
and shutdown
* Same output but
using injected bean

Você pode fazer mais com Quartz

- Manter dados do Scheduler no banco de dados
  - Jobs, Triggers, Calendars
  - org.quartz.impl.jdbcjobstore.JobStoreTX or JobStoreCMT
- Use JDBC DataSources
  - Gerenciados pelo container JEE container ou
  - Pelo próprio Quartzu
- Configure ThreadPool priorities, threads amount, etc
- Use of Quartz plugins
  - Use the JobInitialization plugin to load jobs from XML file
  - Keep the history of trigger events created
  - Shutdown Hook plugin (all the JVM)
- Job Execution Clustering with JDBC JobStore

Other Job Scheduling Solutions

- Built in Java Timer and TimerTask
  - Available in Java Standard Edition since release 1.3
  - Inflexible scheduling (only start-time and repeat interval)
    - Can’t use dates, week days, etc.
    - No support for CRON expressions
  - Don’t have runtime management support.
  - No plugin support

- EJB Timers
  - Available since EJB 2.1
  - Requires an EJB container
  - No plugin support
  - CRON expression support present on release 3.1

Does Quartz met our initial expectations ?

- Job Scheduling Software are expected to support
  - ☑ Configurable Job execution schedules
  - ☑ Configurable parameters to Jobs on deployment basis
  - ☑ Easy integration with already existing (or legacy) code
  - ☒ Job sequencing and dependencies (workflow like)
  - ☑ Context isolation via a sandbox model
  - ☒ Interface for Monitoring Job execution
  - ☑ Job queues and deal with job priorities *(not covered)*

- Positive aspects
  - Free, open-source, mature and actively maintained
  - Easy to use and integrate with widely used technology
    - Servlet containers, Spring Framework, …

Wrap Up

- As expected, we've covered
  - Uses, Features, Basic configuration
  - Integration with Spring IoC container
  - Provided you some examples
- What you should take from here
  - Quartz Scheduler is alone a good and simple solution
  - Spring integration gives you extra power
    - Leverage Spring Dependency Injection (DI)
    - Inject static parameters and other beans instances
  - Allow easily create and maintain scheduling info
    - No code changes required
  - EJB Timers are interesting also
    - But requires EJB Container infrastructure

Questions ?

NOW is where we do business.
It's when we do business.
It's how we do business.

ARE YOU READY FOR **NOW**?

Presentation by Rodrigo Malara
EDS Top Gun Program

rodrigo.malara@hp.com
Araraquara, Brazil

Cron Expression Examples – extra slide

0 0 12 * * ? → Fire at 12pm (noon) every day

0 15 10 * * ? 2005 → Fire at 10:15am every day during the year 2005

0 * 14 * * ? → Fire every minute starting at 2pm and ending at 2:59pm, every day

0 0/5 14 * * ? → Fire every 5 minutes starting at 2pm and ending at 2:55pm, every day

0 0/5 14,18 * * ? → Fire every 5 minutes starting at 2pm and ending at 2:55pm, AND fire every 5 minutes starting at 6pm and ending at 6:55pm, every day

0 10,44 14 ? 3 WED → Fire at 2:10pm and at 2:44pm every Wednesday in the month of March.

0 15 10 ? * MON-FRI → Fire at 10:15am every Monday, Tuesday, Wednesday, Thursday and Friday

0 15 10 L * ? → Fire at 10:15am on the last day of every month

0 15 10 ? * 6L → Fire at 10:15am on the last Friday of every month

0 15 10 ? * 6L → Fire at 10:15am on the last Friday of every month

0 15 10 ? * 6L 2002-2005 → Fire at 10:15am on every last Friday of every month during the years 2002, 2003, 2004 and 2005

0 15 10 ? * 6#3 → Fire at 10:15am on the third Friday of every month

0 0 12 1/5 * ? → Fire at 12pm (noon) every 5 days every month, starting on the first day of the month.