Generació d'informes amb JasperReports

A qui va dirigit

Aquest howto va dirigit a tots aquells que hagin de desenvolupar una aplicació Canigó 3 i que necessitin disposar d'un sistema de generació d'informes.

Versió de Canigó

Aquest howto ha estat redactat per aplicacions Canigó 3.

Introducció

Amb aquest document es pretén proporcionar una guia per als desenvolupadors que necessitin que la seva aplicació Canigó 3 sigui capaç de generar informes.

Es parteix d'una plantilla d'aplicació Canigó 3 a la qual es vol integrar un sistema de generació d'informes. Existeixen diverses llibreries per a fer-ho. S'ha escollit JasperReports degut a que a Canigó 2 existeix un servei específic de reporting integrat amb aquesta llibreria i pel grau de maduresa d'aquest producte sent un dels més estesos en el món J2EE.

Configuració

Maven

Incloure dependència amb JasperReports i Spring MVC:

pom.xml

```
<dependency>
        <groupId>jasperreports</groupId>
        <artifactId>jasperreports</artifactId>
        <version>3.5.3</version>
</dependency>
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc</artifactId>
        <version>${org.springframework.version}</version>
</dependency>
```

Aplicació web

Cal modificar el fitxer *<app>/src/main/webapp/WEB-INF/web.xml* per tal que les peticions http://<host>:<port>/<app_context>/reports/* siguin ateses per el DispatcherServlet de Spring:

web.xml

```
<servlet>
```

```
        <servlet-name>dispatcher</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
        <servlet-name>dispatcher</servlet-name>
        <url-pattern>/reports/*</url-pattern>
</servlet-mapping>
```

Spring

Un cop feta aquesta modificació, ha de configurar-se el servlet creant el fitxer *<app>/src/main/webapp/WEB-INF/dispatcher-servlet.xml* amb la següent configuració:

dispatcher-servlet.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:context="http://www.springframework.org/schema/context"
        xmlns:aop="http://www.springframework.org/schema/aop"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:p="http://www.springframework.org/schema/p"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
                http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
                http://www.springframework.org/schema/aop
                http://www.springframework.org/schema/aop/spring-aop-3.0.xsd
                http://www.springframework.org/schema/context
                http://www.springframework.org/schema/context/spring-context-3.0.xsd">

        <!--
        Resolves view names based on the names declared on the declared xml location
        All our Jasper views are declared inside the specified xml location.
        Take note of the ordering of this ViewResolver.
         -->
        <bean class="org.springframework.web.servlet.view.XmlViewResolver"
                        p:location="classpath:spring/jasper-views.xml"  p:order="0" />

</beans>
```

També ha de crear-se el fitxer *<app>/src/main/resources/spring/app-integration-jasperreports.xml* amb el següent contingut:

app-integration-jasperreports.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:context="http://www.springframework.org/schema/context"
        xmlns:mvc="http://www.springframework.org/schema/mvc"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
                http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
                http://www.springframework.org/schema/context
                http://www.springframework.org/schema/context/spring-context-3.0.xsd
                http://www.springframework.org/schema/mvc
                http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd">

        <!-- Configures the annotation-driven Spring MVC Controller programming model.
        Note that, with Spring 3.0, this tag works in Servlet MVC only!  -->
        <mvc:annotation-driven />

        <import resource="classpath:spring/jasper-views.xml" />
```

```
</beans>
```

Exemple report PDF

Les vistes corresponents als reports han de definir-se al fitxer *<app>/src/main/resources/spring/jasper-views.xml*:

jasper-views.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:p="http://www.springframework.org/schema/p"
       xmlns:util="http://www.springframework.org/schema/util"
       xsi:schemaLocation="
              http://www.springframework.org/schema/beans
              http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
              http://www.springframework.org/schema/util
              http://www.springframework.org/schema/util/spring-util-3.0.xsd">

       <!--
              Declare Spring's View Resolvers for Jasper

              Based on the bean names we can infer that:
                     * pdfReport is for generating PDFs
       -->
       <!--
              id: the name to be used as a reference in the controller
              url: the path where the master JRXML/JASPER file is located
       -->
       <bean id="customerReportList"

class="org.springframework.web.servlet.view.jasperreports.JasperReportsPdfView"
              p:url="classpath:reports/customerReportList.jasper" />

</beans>
```

El controlador que rebrà la petició, crearà el model de dades i redigirà la petició cap a la vista:

CustomerController.java

```
package cat.gencat.canigo3.jasperreports.bean;

import java.util.ArrayList;
import java.util.List;

import net.sf.jasperreports.engine.data.JRBeanCollectionDataSource;

import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import cat.gencat.canigo3.jasperreports.model.Customer;

@RequestMapping("/customer/**")
@Controller
public class CustomerController {

       @RequestMapping(value = "/list/pdf", method = RequestMethod.GET)
       public String fireReport(ModelMap modelMap) {
```

```
                JRBeanCollectionDataSource jrDataSource = new JRBeanCollectionDataSource(
                                getCustomerList(), false);
                modelMap.put("customerReportList", jrDataSource);
                return "customerReportList";
        }

        private List<Customer> getCustomerList() {
                List<Customer> customers = new ArrayList<Customer>();

                Customer c1 = new Customer();
                c1.setFirstname("Nom1");

                Customer c2 = new Customer();
                c2.setFirstname("Nom2");

                Customer c3 = new Customer();
                c3.setFirstname("Nom3");

                customers.add(c1);
                customers.add(c2);
                customers.add(c3);

                return customers;
        }

}
```

Per incloure un report al projecte cal seguir les següents passes:

1. *Disseny del report*: amb l'eina iReport es pot desenvolupar l'informe. Per no tenir problemes d'incompatibilitats es recomana utilizar la mateixa versió d'iReport que de JasperReports, en aquest exemple 3.5.3. El fitxer resultat té extensió .jrxml.

customerReportList.jrxml

```
<?xml version="1.0" encoding="UTF-8"?>
<jasperReport xmlns="http://jasperreports.sourceforge.net/jasperreports"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://jasperreports.sourceforge.net/jasperreports
http://jasperreports.sourceforge.net/xsd/jasperreport.xsd" name="customerList"
pageWidth="612" pageHeight="792" columnWidth="555" leftMargin="20" rightMargin="20"
topMargin="20" bottomMargin="20">
  <field name="firstname" class="java.lang.String"/>
  <columnHeader>
    <band height="31" splitType="Stretch">
      <staticText>
        <reportElement x="0" y="0" width="100" height="20"/>
        <text><![CDATA[FIRSTNAME]]></text>
      </staticText>
    </band>
  </columnHeader>
  <detail>
    <band height="24" splitType="Stretch">
      <textField>
        <reportElement x="0" y="0" width="100" height="20"/>
        <textElement/>
        <textFieldExpression
class="java.lang.String"><![CDATA[$F{firstname}]]></textFieldExpression>
      </textField>
    </band>
  </detail>
</jasperReport>
```

2. *Compilació*: per compilar el report es pot fer servir el plugin de maven jasperreports-maven-plugin. Aquesta és la configuració que hauria d'incloure's en el pom.xml

pom.xml

```
<plugin>
        <groupId>org.codehaus.mojo</groupId>
        <artifactId>jasperreports-maven-plugin</artifactId>
        <configuration>
           <sourceDirectory>src/main/resources/reports</sourceDirectory>
           <outputDirectory>src/main/resources/reports</outputDirectory>
        </configuration>
        <executions>
          <execution>
            <goals>
              <goal>compile-reports</goal>
            </goals>
          </execution>
        </executions>
       <dependencies>
         <dependency>
           <groupId>jasperreports</groupId>
           <artifactId>jasperreports</artifactId>
           <version>3.5.3</version>
         </dependency>
           <dependency>
             <groupId>log4j</groupId>
             <artifactId>log4j</artifactId>
             <version>1.1.3</version>
           </dependency>
       </dependencies>
</plugin>
```

Nota: com es pot veure s'ha configurat el plugin per a que generi el report compilat en el mateix directori que on està el font

Per compilar el report cal anar al directori arrel del projecte i executar la comanda mvn jasperreports:compile-reports. Com a resultat s'obtè el fitxer .jasper resultat de la compilació

I finalment, un cop desplegada l'aplicació, es pot obtenir el report mitjançant la següent URL: http://<host>:<port>/<app>/AppJava/reports/customer/list/pdf

Referències

- JasperReports: http://community.jaspersoft.com/project/jasperreports-library
- iReport: http://community.jaspersoft.com/project/ireport-designer
- Spring MVC – JasperReports: https://code.google.com/p/spring-mvc-jasper-integration-tutorial/