Acabamos de ver como usar Java equals y hashcode

Hoy he sacado un hueco para hablar Spring Expression Language algo un contacto de linkedin (Fermín Martín) me sugirió. ¿Para que sirve Spring Expression Language? .Bueno se trata de un lenguaje de expresiones que permite trabajar de una forma mas flexible con el framework Spring .Vamos a ver un ejemplo sencillo Para ello diseñaremos un proyecto de Spring con Maven a continuación se muestran las dependencias.

```
&amp;lt;dependency&amp;gt;
 &amp;lt;groupId&amp;gt;org.springframework&amp;lt;/groupId&amp;gt;
 &amp;lt;artifactId&amp;gt;spring-core&amp;lt;/artifactId&amp;gt;
 &amp;lt;version&amp;gt;3.2.8.RELEASE&amp;lt;/version&amp;gt;
 &amp;lt;/dependency&amp;gt;
 &amp;lt;dependency&amp;gt;
 &amp;lt;groupId&amp;gt;org.springframework&amp;lt;/groupId&amp;gt;
 &amp;lt;artifactId&amp;gt;spring-webmvc&amp;lt;/artifactId&amp;gt;
 &amp;lt;version&amp;gt;3.2.8.RELEASE&amp;lt;/version&amp;gt;
 &amp;lt;/dependency&amp;gt;
 &amp;lt;dependency&amp;gt;
 &amp;lt;groupId&amp;gt;org.springframework&amp;lt;/groupId&amp;gt;
 &amp;lt;artifactId&amp;gt;spring-web&amp;lt;/artifactId&amp;gt;
 &amp;lt;version&amp;gt;3.2.8.RELEASE&amp;lt;/version&amp;gt;
 &amp;lt;/dependency&amp;gt;
 &amp;lt;dependency&amp;gt;
 &amp;lt;groupId&amp;gt;org.springframework&amp;lt;/groupId&amp;gt;
 &amp;lt;artifactId&amp;gt;spring-context&amp;lt;/artifactId&amp;gt;
 &amp;lt;version&amp;gt;3.2.8.RELEASE&amp;lt;/version&amp;gt;
 &amp;lt;/dependency&amp;gt;
 &amp;lt;dependency&amp;gt;
 &amp;lt;groupId&amp;gt;org.springframework&amp;lt;/groupId&amp;gt;
```

```
 &amp;lt;artifactId&amp;gt;spring-
expression&amp;lt;/artifactId&amp;gt;
 &amp;lt;version&amp;gt;3.2.8.RELEASE&amp;lt;/version&amp;gt;
 &amp;lt;/dependency&amp;gt;
 &amp;lt;dependency&amp;gt;
 &amp;lt;groupId&amp;gt;javax.servlet&amp;lt;/groupId&amp;gt;
 &amp;lt;artifactId&amp;gt;javax.servlet-
api&amp;lt;/artifactId&amp;gt;
 &amp;lt;version&amp;gt;3.0.1&amp;lt;/version&amp;gt;
 &amp;lt;/dependency&amp;gt;
```

Spring Expression Language

Vamos a diseñar un sencillo fichero de propiedades que contenga dos datos el porcentaje de iva y la retención que se aplica. El fichero se denomina impuestos.properties

```
iva=21
retencion=15
```

Vamos a integrar ahora esta información dentro de un bean de servicio de Spring para ello lo primero será configurar el framework Spring a nivel de web.xml.

```
&amp;lt;?xml version="1.0" encoding="UTF-8"?&amp;gt;
&amp;lt;web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
```

```
version="3.0"&amp;gt;
&amp;lt;display-name&amp;gt;Spring MVC&amp;lt;/display-name&amp;gt;
&amp;lt;listener&amp;gt;
&amp;lt;listener-class&amp;gt;
org.springframework.web.context.ContextLoaderListener
&amp;lt;/listener-class&amp;gt;
&amp;lt;/listener&amp;gt;
&amp;lt;/web-app&amp;gt;

&amp;amp;nbsp;
```

Una vez hecho esto el siguiente paso es registrar el fichero de propiedades a nivel de Spring framework usando el applicationContext.xml

```
&amp;lt;?xml version="1.0" encoding="UTF-8"?&amp;gt;
&amp;lt;beans xmlns="http://www.springframework.org/schema/beans"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:util="http://www.springframework.org/schema/util"
 xmlns:context="http://www.springframework.org/schema/context"
 xsi:schemaLocation="
http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/util
http://www.springframework.org/schema/util/spring-util.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context.xsd"&amp;
gt;

&amp;lt;context:component-scan base-package="com.arquitecturajava"
```

```
/&amp;gt;
&amp;lt;util:properties id="impuestos"
location="classpath:impuestos.properties"/&amp;gt;
&amp;lt;/beans&amp;gt;
```

Como vemos se ha usado para registrar el fichero la etiqueta <util:properties> y se han registrado todas las clases que se encuentren en al paqueta “com.arquitecturajava” . Realizada esta operación el siguiente paso es asignar los valores del fichero de properties a variables de clases de Servicio.

```
package com.arquitecturajava;

public interface ServicioImpuestos {
 public int getIva();
 public int getRetencion();
}
```

```
package com.arquitecturajava;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Service;

@Service("servicioImpuestos")
public class ServicioImpuestosImpl implements ServicioImpuestos {

 @Value("#{impuestos.iva}")
```

```
 private int iva;
 @Value("#{new Integer(impuestos.retencion)+3}")
 private int retencion;

 public int getIva() {
 return iva;
 }
 public void setIva(int iva) {
 this.iva = iva;
 }
 public void setRetencion(int retencion) {
 this.retencion = retencion;
 }
 public int getRetencion() {
 return retencion;
 }
}
```

Podemos ver aquí como se cargan automaticamente los valores del fichero en el bean de Servicio a través de la anotación @Value y se rellena la propiedadde iva

```
@Value("#{impuestos.iva}")
 private int iva;
```

De igual manera se puede usar Spring Expression Language para realizar operaciones . Como sucede en la retención que la incrementamos.

```
 @Value("#{new Integer(impuestos.retencion)+3}")
```

```
 private int retencion;
```

Así pues hemos usado Spring Expression Language para parametrizar datos que afectan a nuestros servicios.

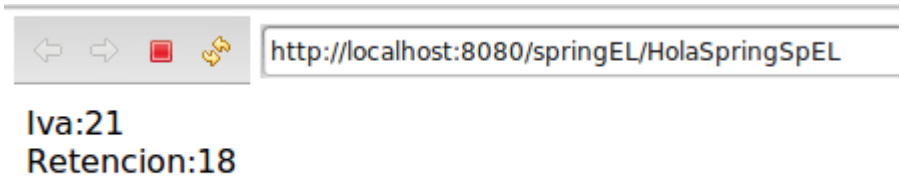
Una vez utilizada la anotación podremos generar un Servlet que invoque al servicio y nos presente por pantalla los datos de nuestro servicio de impuestos.Para ello inyectamos el servicio con @Autowired.

```
package com.arquitecturajava;

import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.beans.factory.annotation.Autowired;
import
org.springframework.web.context.support.SpringBeanAutowiringSupport;
```

```
public class HolaSpringSpEL extends HttpServlet {
private static final long serialVersionUID = 1L;

@Autowired
private ServicioImpuestos miservicio;

public HolaSpringSpEL() {
// TODO Auto-generated constructor stub
}

protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {

response.setContentType("text/html");
PrintWriter pw=response.getWriter();
pw.print("Iva:"+ miservicio.getIva()+" &amp;lt;br/&amp;gt;");
pw.print("Retencion:"+ miservicio.getRetencion());

}

public void init(ServletConfig config) {
try {
super.init(config);

SpringBeanAutowiringSupport.processInjectionBasedOnServletContext(this
,
config.getServletContext());
} catch (ServletException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
```

```
}

}

&amp;amp;nbsp;
```

Realizada esta operación el Servlet se apoya en Spring Expression Language y nos genera la siguiente página.



Acabamos de construir nuestro primer ejemplo con Spring Expression Language .Sin embargo podemos seguir avanzando y realizar otro tipo de operaciones con este lenguage de expresiones como por ejemplo usarlo en JSP a traves de los taglib de Spring.

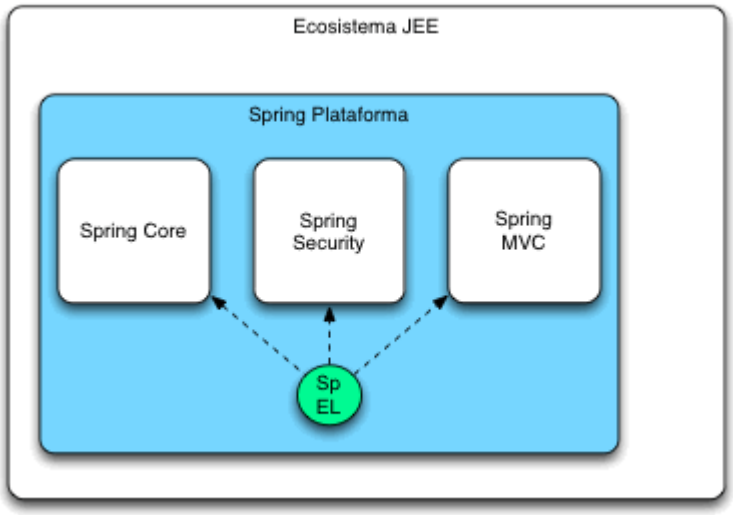
```
&amp;lt;%@ page language="java" contentType="text/html; charset=UTF-8"
 pageEncoding="UTF-8"%&amp;gt;
&amp;lt;%@ taglib prefix="spring"
uri="http://www.springframework.org/tags" %&amp;gt;
&amp;lt;!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd"&amp;gt;
&amp;lt;html&amp;gt;
&amp;lt;head&amp;gt;
&amp;lt;meta http-equiv="Content-Type" content="text/html;
charset=UTF-8"&amp;gt;
&amp;lt;title&amp;gt;Insert title here&amp;lt;/title&amp;gt;
```

```
&amp;lt;/head&amp;gt;
&amp;lt;body&amp;gt;
&amp;lt;spring:eval
expression="@servicioImpuestos.retencion"&amp;gt;&amp;lt;/spring:eval&
amp;gt;&amp;lt;/br&amp;gt;
&amp;lt;spring:eval
expression="@impuestos.iva"&amp;gt;&amp;lt;/spring:eval&amp;gt;
&amp;lt;/body&amp;gt;
&amp;lt;/html&amp;gt;
```

Por lo tanto podemos usar Spring Expression Language en muchas partes de nuestro código para realizar operaciones de parametrización ,conversiones etc .Aún así mucha gente no acaba de entender bien para que se usa vamos a intentar aclararlo.

¿Es Spring un Framework?

La respuesta suele ser clara ,SI es uno de los frameworks de referencia en la plataforma JEE. Para mi esto no es tan cierto, Spring ya hace tiempo que no es un framework .Ahora mismo es una PLATAFORMA de desarrollo gigantesta. Una plataforma que dispone de muchos productos y que necesita generar homegeneidad entre todos ellos . Esta plataforma se encuentra ubicada dentro del Ecosistema de Java EE .¿Para que sirve SpEL? .Para generar homegenidad entre los distintos productos como son Spring Security ,Spring MVC etc.



Aprendamos a usar este lenguaje de expresiones que nos será util en nuestros desarrollos.

Descargar

:springEL