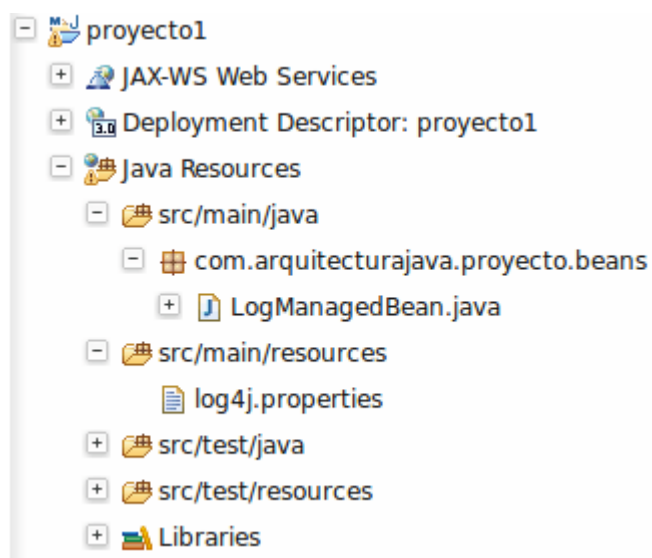


En el post anterior hemos añadido las dependencias que necesitamos para nuestra aplicación .Ahora simplemente vamos a construirla, para ello necesitaremos un ManagedBean que escriba mensajes a log y el fichero log4j.properties que se encarga de configurar nuestro sistema de Log. En este caso la configuración del log será muy sencilla apoyándonos en la consola. Vamos a ver donde ubicar estos dos ficheros.



El fichero de log4j.properties le hemos ubicado en la carpeta de recursos y tiene el siguiente contenido:

```
log4j.rootLogger=INFO, stdout
```

```
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
```

```
log4j.appender.stdout.Target=System.out
```

```
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
```

```
log4j.appender.stdout.layout.ConversionPattern= %m%n
```

El fichero de código fuente le hemos ubicado en la carpeta src/java/main y tiene el siguiente contenido:

```
package com.arquitecturajava.proyecto.beans;

import javax.faces.bean.ManagedBean;

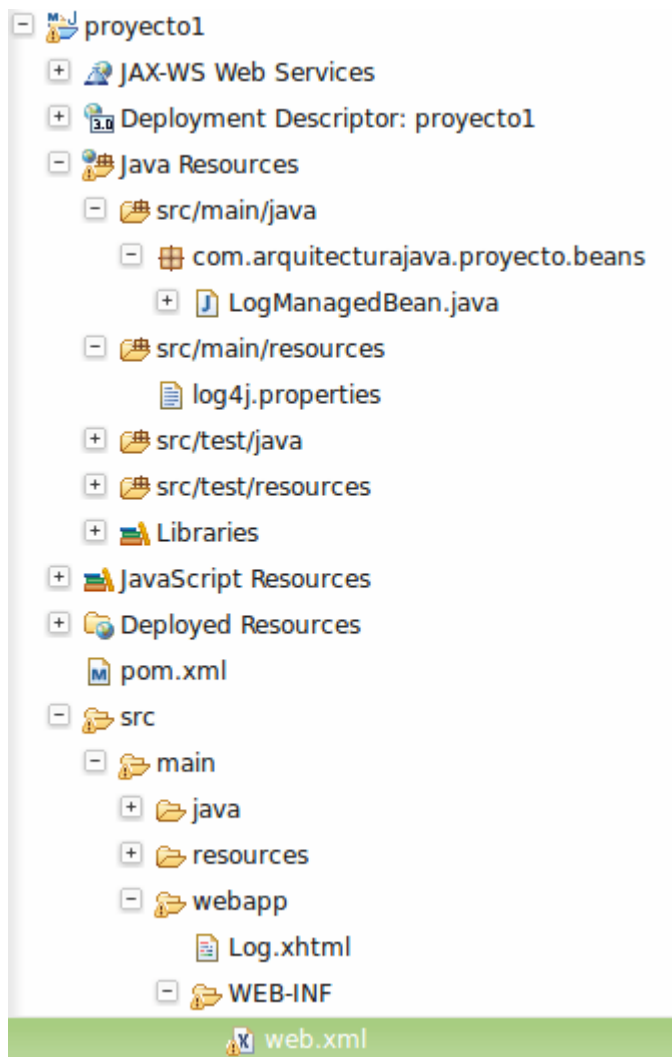
import org.apache.log4j.Logger;

@ManagedBean(name="logManagedBean")
public class LogManagedBean {
    public String mensaje(){

        Logger milogger=
        Logger.getLogger(this.getClass().getPackage().getName());

        milogger.info("hola desde jsf al log");
        return "log";
    }
}
```

Una vez tenemos estos dos ficheros vamos a ver donde hemos ubicado los otros dos ficheros que necesitamos web.xml y Log.xhtml .



Podemos ver que dentro de la carpeta de webapp hemos alojado el fichero Log.xhtml para después crear la carpeta WEB-INF y alojar dentro de ella el fichero web.xml .El contenido de estos ficheros es el siguiente:

```
&lt;?xml version="1.0" encoding="UTF-8"?>
&lt;!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
&lt;html xmlns="http://www.w3.org/1999/xhtml"
xmlns:h="http://java.sun.com/jsf/html">
```

```

&lt;h:head&gt;
&lt;title&gt;Bienvenido&lt;/title&gt;
&lt;/h:head&gt;
&lt;h:body&gt;
&lt;h:form&gt;
&lt;h:commandButton value="Aceptar"
action="#{logManagedBean.mensaje}" /&gt;

&lt;/h:form&gt;
&lt;/h:body&gt;
&lt;/html&gt;

```

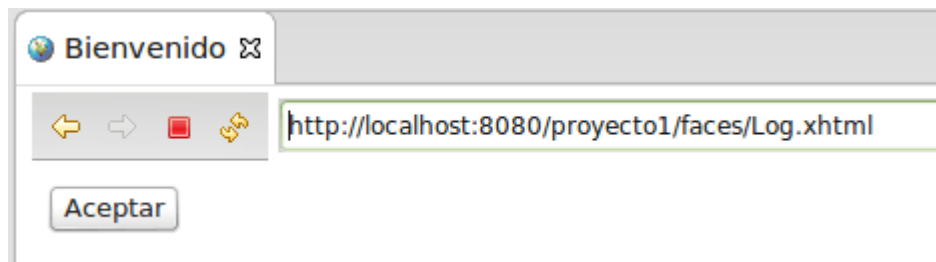
web.xml:

```

&lt;web-app xmlns="http://java.sun.com/xml/ns/javaee"
version="3.0"&gt;
  &lt;servlet&gt;
    &lt;servlet-name&gt;Faces Servlet&lt;/servlet-
name&gt;
    &lt;servlet-
class&gt;javax.faces.webapp.FacesServlet&lt;/servlet-
class&gt;
    &lt;load-on-startup&gt;1&lt;/load-on-startup&gt;
    &lt;/servlet&gt;
    &lt;servlet-mapping&gt;
      &lt;servlet-name&gt;Faces Servlet&lt;/servlet-
name&gt;
      &lt;url-pattern&gt;/faces/*&lt;/url-pattern&gt;
      &lt;/servlet-mapping&gt;
    &lt;/web-app&gt;

```

Realizadas todas estas operaciones podemos usar mvn package y empaquetar la aplicación para desplegarla en un Tomcat. Al desplegarla en Tomcat solicitaremos la página Log.xhtml



Hecho esto pulsaremos sobre el botón y nos realizará log por consola :

```
INFO: Initialization processed in 1255 ms
may 01, 2013 6:31:33 PM org.apache.catalina.core.StandardService sta
INFO: Arrancando servicio Catalina
may 01, 2013 6:31:33 PM org.apache.catalina.core.StandardEngine sta
INFO: Starting Servlet Engine: Apache Tomcat/7.0.39
may 01, 2013 6:31:36 PM com.sun.faces.config.ConfigureListener conte
INFO: Inicializando Mojarra 2.1.20 ( 20130304-1520 https://svn.java
may 01, 2013 6:31:37 PM com.sun.faces.spi.InjectionProviderFactory c
INFO: JSF1048: hay presentes anotaciones PostConstruct/PreDestroy.
may 01, 2013 6:31:37 PM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler ["http-bio-8080"]
may 01, 2013 6:31:37 PM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler ["ajp-bio-8009"]
may 01, 2013 6:31:37 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 3691 ms
hola desde jsf al log
```

Tenemos correctamente desplegada una aplicación web con estructura y dependencias de Maven en un Tomcat. Sin embargo nos queda un tema puntual. El plugin que tenemos instalado m2e no soporta una integración con el Tomcat interno del Eclipse. En el siguiente post cubriremos este problema.