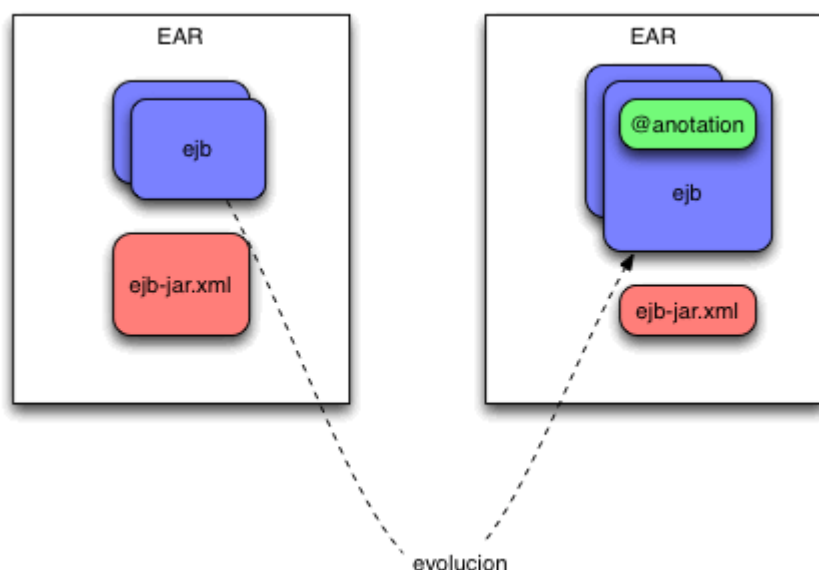


A partir de EJB 3.0 el uso del fichero ejb-jar.xml (deployment descriptor) se ha reducido significativamente ya que el uso de anotaciones ha simplificado sobremanera la forma de definir los distintos EJBs . De esta forma se ha eliminado la necesidad de usar este fichero que pasa a ser opcional y que se usa en algunas situaciones muy concretas.



Uso de ejb-jar.xml

Sin embargo el fichero sigue estando presente y en algunas ocasiones nos puede ser útil utilizarlo. Uno de los casos más habituales es la parametrización de valores que la aplicación necesita usar. Vamos a ver un ejemplo sencillo. Para ello nos vamos a crear un EJB que nos devuelva un número y este número lo vamos a inicializar en el ejb-jar.xml.

```
package com.arquitecturajava.ejb;
```

```
import javax.annotation.Resource;  
import javax.ejb.Stateless;
```

```

@Stateless
public class IniciarEJB implements IniciarEJBLocal {

@Resource(name="numero")
    private int numero;
    public int numero() {

        return numero;
    }

}

```

```

package com.arquitecturajava.ejb;

```

```

import javax.ejb.Local;

```

```

@Local
public interface IniciarEJBLocal {
    public int numero() ;
}

```

Como podemos ver el EJB hace uso de la anotación de @Resource para inyectar un valor en la variable. Para ello usaremos el ejb-jar.xml.

```

<?xml version="1.0" encoding="UTF-8">
<ejb-jar 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/ejb-jar_3_0.xsd"
version="3.0"&gt;
&lt;display-name&gt;EJB004Variable&lt;/display-
name&gt;
&lt;enterprise-beans&gt;
&lt;session&gt;
&lt;ejb-name&gt;IniciarEJB&lt;/ejb-name&gt;
&lt;env-entry&gt;
&lt;env-entry-name&gt;
numero
&lt;/env-entry-name&gt;
&lt;env-entry-type&gt;java.lang.Integer&lt;/env-entry-
type&gt;
&lt;env-entry-value&gt;5&lt;/env-entry-value&gt;
&lt;/env-entry&gt;
&lt;/session&gt;
&lt;/enterprise-beans&gt;
&lt;/ejb-jar&gt;

```

Hemos asignado a la variable el valor de 5. Es momento de usar un Servlet para inyectar un EJB e imprimir el resultado por pantalla.

```
package com.arquitecturajava.web;
```

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

```

import javax.ejb.EJB;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.arquitecturajava.ejb.IniciarEJBLocal;

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

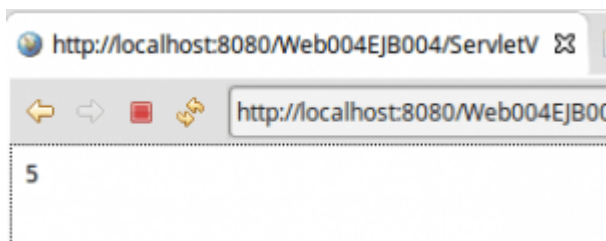
    @EJB
    IniciarEJBLocal miejb;
    protected void doGet(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {

        PrintWriter pw= response.getWriter();
        pw.println(miejb.numero());
    }

}

```

El resultado será que el navegador nos imprimirá el valor por la pantalla.



Otro de los usos más típicos del ejb-jar.xml es cuando queremos aplicar interceptores a varios de nuestros EJBs ya que recordemos las anotaciones se aplican a un solo EJB.

Otros artículos relacionados:

[Introducción a EJB](#)

[EJB Remotos](#)

[el concepto de EAR](#)