Trabajar con EJBS se ha convertido estos últimos años en algo bastante mas sencillo gracias a EJB 3.0 y EJB 3.1 .Aun así siempre quedan partes de que suelen generar bastantes dudas .En este post voy a hablar un poco de las anotaciones mas básicas @Stateless y @EJB y algunos de sus atributos que muchas veces generan dudas (name,beanName,mappedName). Vamos a suponer que tenemos el siguiente EJB con sus interfaces remotos y locales.

Interface Local

```
package com.arquitecturajava;

import javax.ejb.Local;

@Local
public interface HolaEJBLocal {

public String mensaje();
}
```

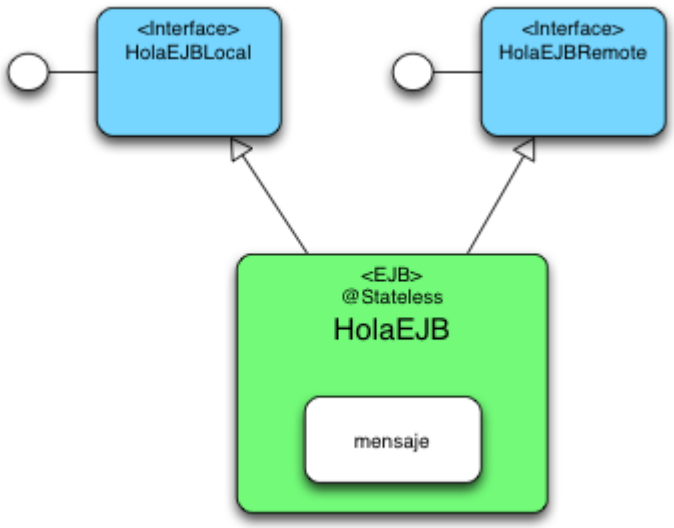
Interface Remoto:

```
package com.arquitecturajava;

import javax.ejb.Remote;

@Remote
 public interface HolaEJBRemote {

public String mensaje();
 }
```

EJB :

```
package com.arquitecturajava;

import javax.ejb.Stateless;
@Stateless
public class HolaEJB implements HolaEJBRemote, HolaEJBLocal {
 public HolaEJB() {

 }

@Override
 public String mensaje() {

 return "Hola EJB";
 }

}
```

Como podemos ver se trata del EJB de HolaMundo el siguiente diagrama clarifica la relación entre las partes

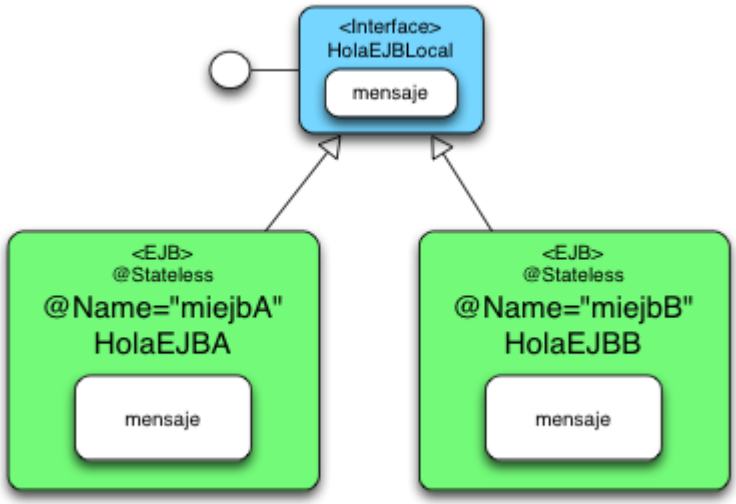


EJB Name

Vamos a modificar el EJB de la siguiente forma

```
@Stateless(name="miejbA" )
public class HolaEJBA implements HolaEJBRemote, HolaEJBLocal {
 public HolaEJBA() {

 }

@Override
 public String mensaje() {

 return "Hola EJB A ";
 }

}
```

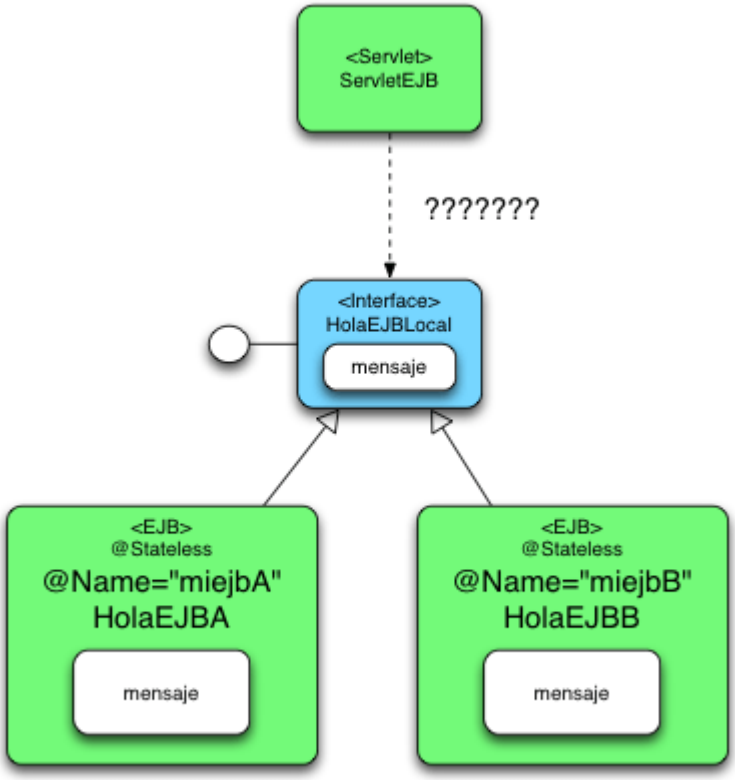
Hemos añadido la anotación name y hemos modificado el nombre del EJB (HolaEJBA) así como su implementación .Ahora bien ¿para que sirve la anotación name? . Sirve en el caso de que en el mundo de los EJB un interface tenga varios EJB que lo implementen y necesitemos diferenciarlos. Veamos el siguiente caso.



En este caso tenemos un mismo interface local implementado por dos EJB distintos . Es por lo tanto necesario diferenciar los EJBs de alguna forma y para ello se utiliza el atributo name de la anotación @Stateless.

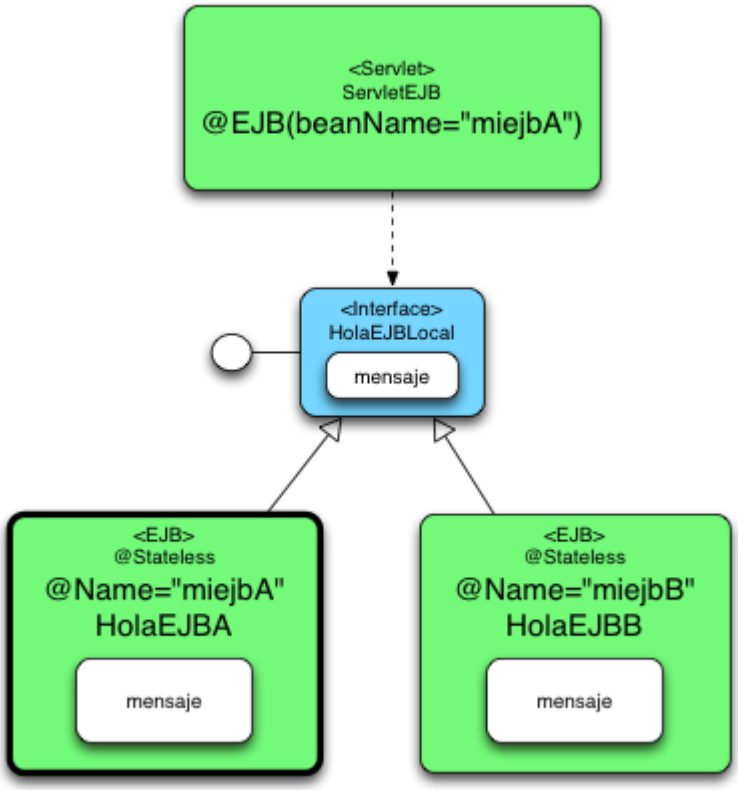
Cliente EJB

En estos momentos se nos plantea un problema . Como accedemos a los distintos EJBs desde un Servlet si el Servlet accede siempre a traves del interface local y existen dos implementaciones distintas de este.



EJB beanName

Para solventar este problema deberemos usar la propiedad beanName de la anotación @EJB que nos permite diferenciar que implementación en concreto queremos elegir.



Vamos a mostrar su código fuente

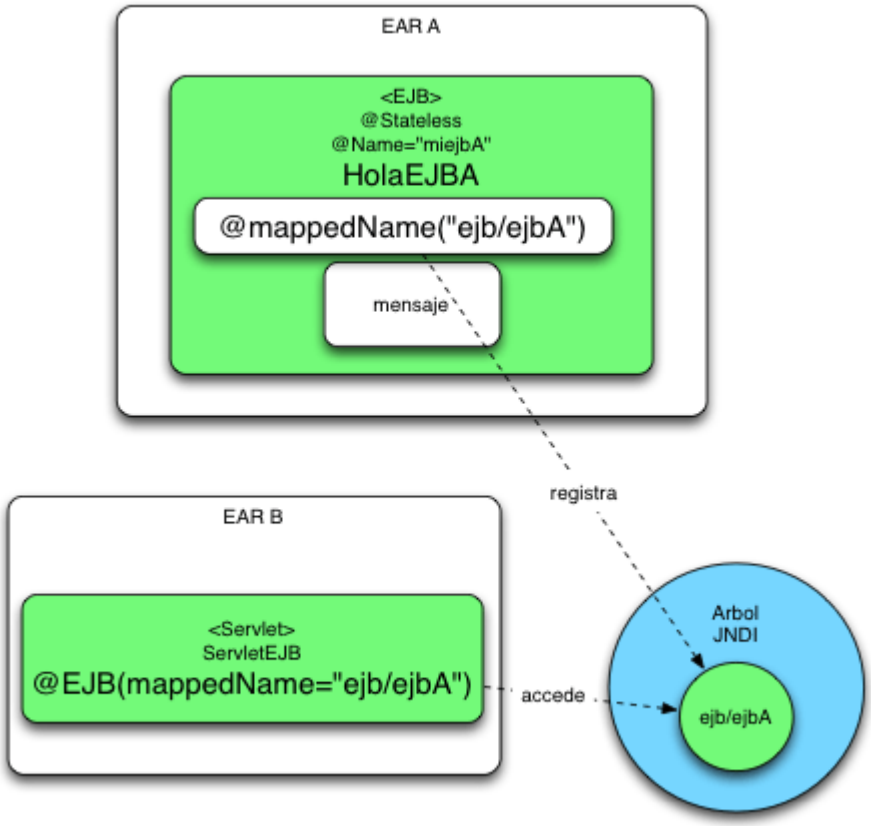
```
package com.arquitecturajava;

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

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

```
@WebServlet("/ServletEJB")
public class ServletEJB extends HttpServlet {
 private static final long serialVersionUID = 1L;

 @EJB(beanName="miejbA")
 private HolaEJBLocal miEJB;

 public ServletEJB() {
 super();

 }

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

 PrintWriter pw= response.getWriter();
 pw.println(miEJB.mensaje());
 }

}
```

Acabamos de explicar las diferencias entre name y beanName .Vamos a abordar ahora la anotación mappedName.

EJB mappedName

A diferencia de otros componentes los EJBs pueden ser accedidos de forma local y también de forma remota .Cuando son accedidos de forma remota deben de estar registrados a nivel de JNDI Global . La anotación mappedName nos permite asignarles un nombre sencillo para que puedan ser accedidos de forma remota a traves de él.



Ahora bien la especificación de EJBs no obliga a los servidores a implementar esta funcionalidad .Es mas permite que cada servidor la implemente de la forma que el considere mas oportuna. Por lo tanto para usar esta anotación hay que tener en cuenta que aunque nos permite un mapeo sencillo a la hora de acceder de forma remota al recurso . Puede que nos complique la portabilidad de la aplicación entre los distintos servidores. JEE 6 nos proveerá de soluciones para este tipo de problemas que cubriremos en articulos posteriores.