

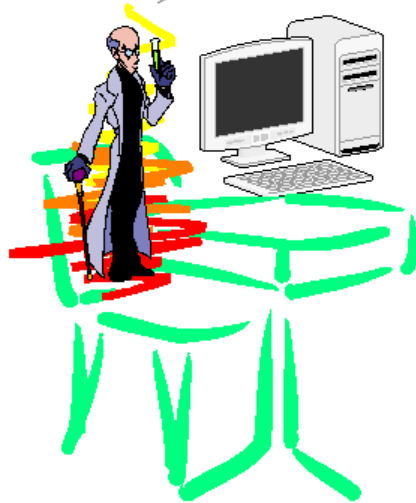


CONTROL DE FLUJO



Objetivos

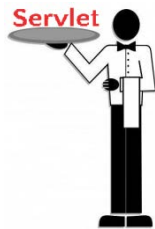
iii El objetivo de este módulo es Conocer las facilidades de control de flujo del API de servlets. Distinguir el uso correcto de los métodos `forward()`, `include()` y `sendRedirect()` .
!!!



!!!! también Conocer los aspectos fundamentales del manejo de atributos a nivel petición, sesión y aplicación en el API de servlets.
iiii



Componentes en una Aplicación Web



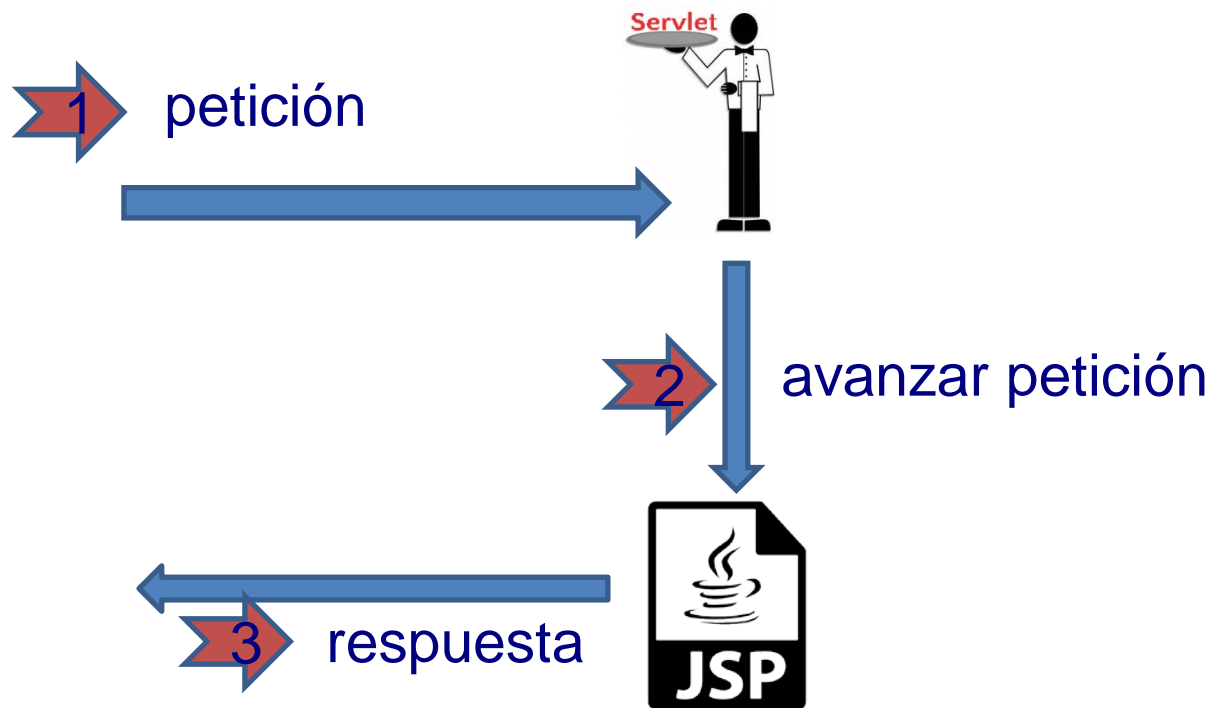
← servlets - ideales para el procesamiento de peticiones



← páginas JSP - ideales para el desplegado de información



Control del Flujo de una Petición



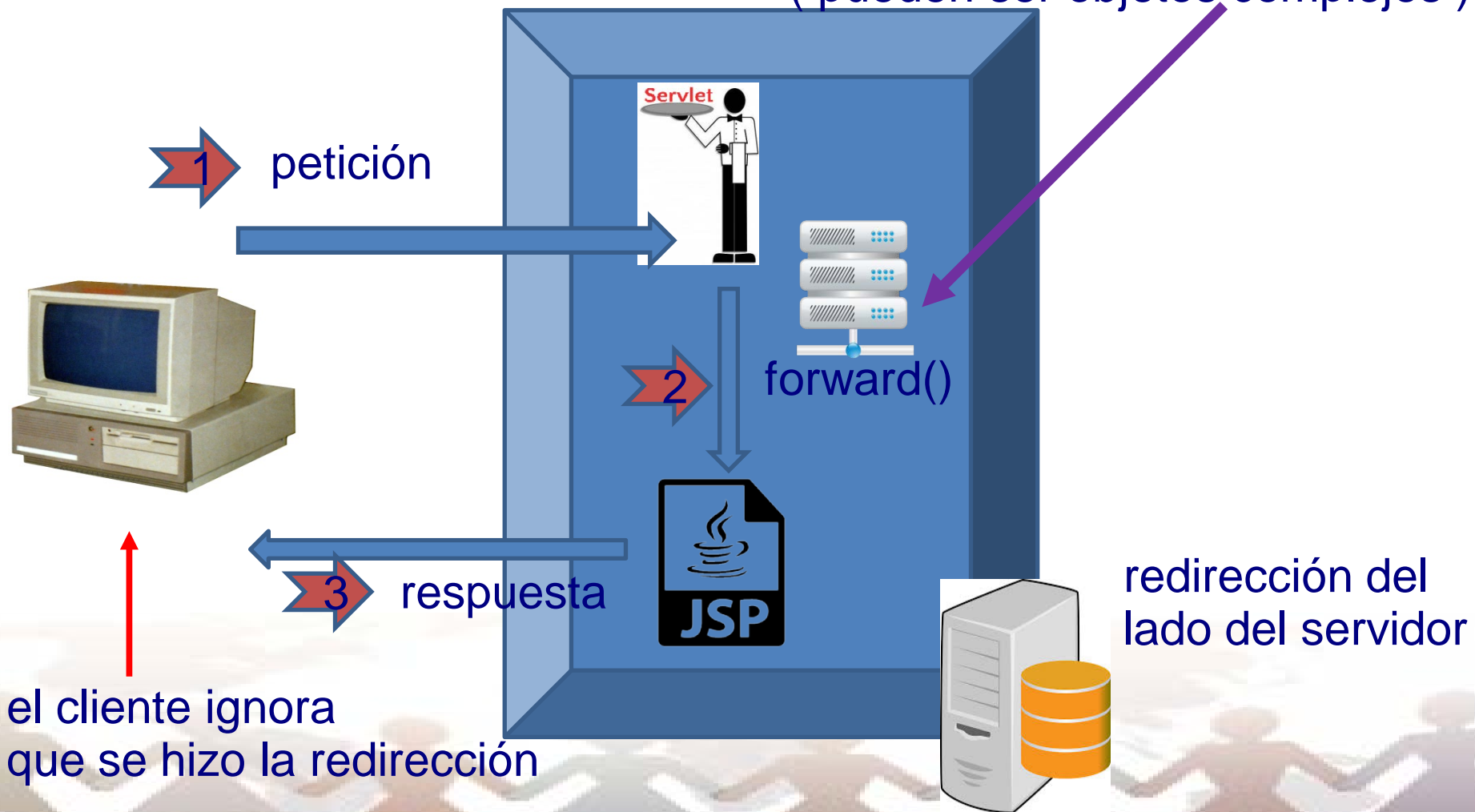
Tipos de Flujo Entre Componentes

- **forward()**
 - Redirección (avance) del lado del servidor.
- **include()**
 - Inclusión de página del lado del servidor.
- **sendRedirect()**
 - Redirección del lado del cliente.



forward()

paso de atributos de petición
(pueden ser objetos complejos)



Procedimiento - forward()

// 1.- Obtener referencia a recurso.

```
RequestDispatcher dispatcher =  
    request.getRequestDispatcher(  
        "/pagina.jsp" );
```

ruta dentro de la
aplicación web actual

// 2.- Avanzar petición.

```
dispatcher.forward( request, response );
```



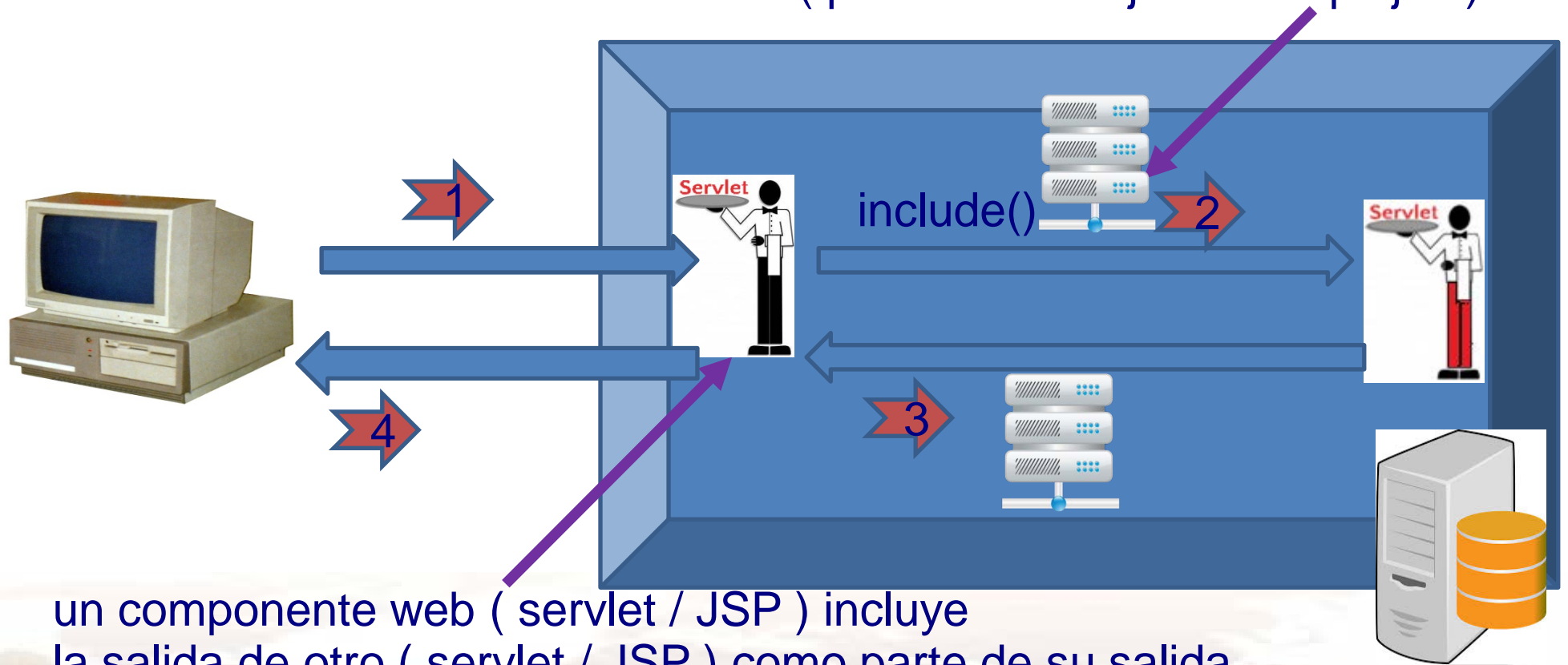
Ejemplo

```
protected final void doGet  
    ( HttpServletRequest request,  
      HttpServletResponse response )  
throws IOException, ServletException  
{  
    // Procesar petición.  
    RequestDispatcher dispatcher =  
        request.getRequestDispatcher( "/mostrar.jsp" );  
    forward( request, response );  
}
```



include()

paso de atributos de petición
(pueden ser objetos complejos)



un componente web (servlet / JSP) incluye
la salida de otro (servlet / JSP) como parte de su salida

Procedimiento - include()

// 1.- Obtener referencia a recurso.

```
RequestDispatcher dispatcher =  
    request.getRequestDispatcher(  
        "/fragmento.jsp" );
```

↑ ruta dentro de la
aplicación web actual

// 2.- Incluir respuesta.

```
dispatcher.include( request, response );
```



Ejemplo

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

```
{  
    // ...Iniciar respuesta...
```

```
    RequestDispatcher dispatcher =  
        request.getRequestDispatcher( "/servlet/menu" );  
    dispatcher.include( request, response );
```

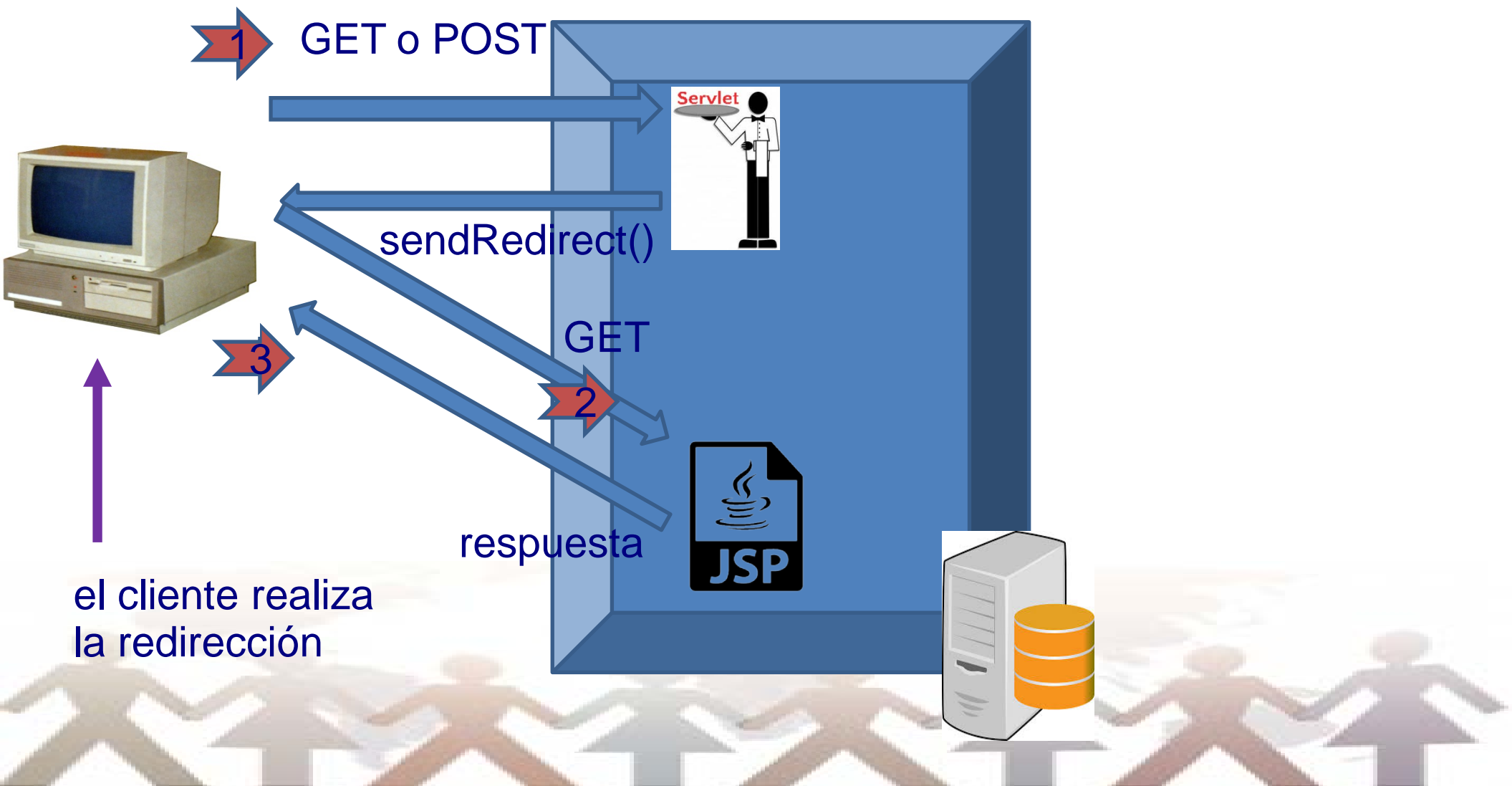
```
    // ...Finalizar respuesta...
```

```
}
```

incluir respuesta
externa

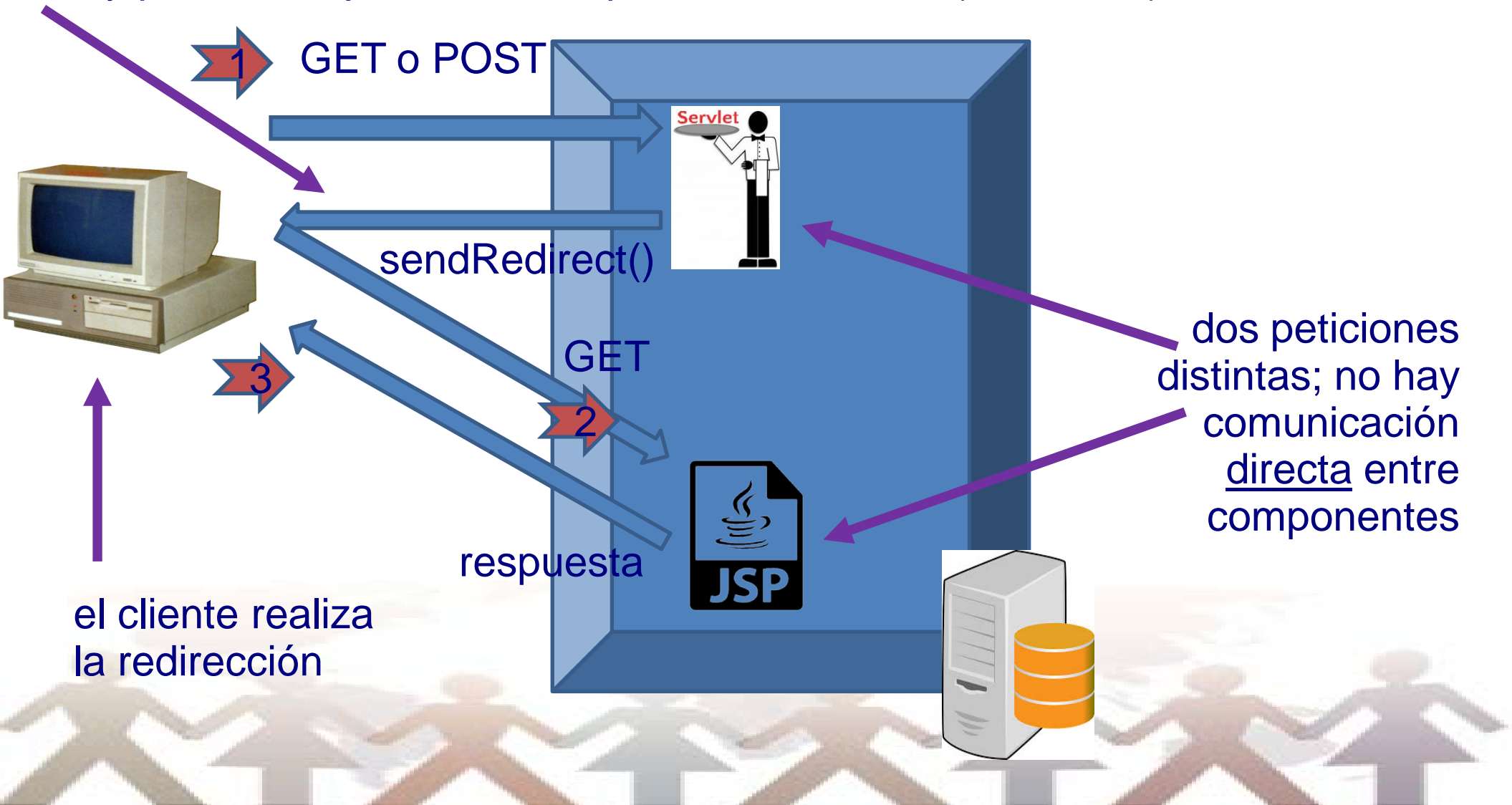


sendRedirect()



sendRedirect()

no hay paso de objetos; sólo de parámetros HTTP (cadenas)



Ejemplo - Recursos Externos

```
protected final void doPost  
    ( HttpServletRequest request, HttpServletResponse response )  
throws ServletException, IOException  
{  
    // Actualizar base de datos.  
    ...  
    response.sendRedirect  
        ( "http://localhost:7080/celulares/mostrar.html" );  
}
```

se puede redireccionar a cualquier lugar de la red



Ejemplo - Recursos Internos

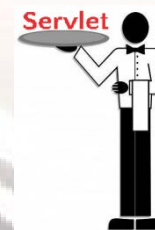
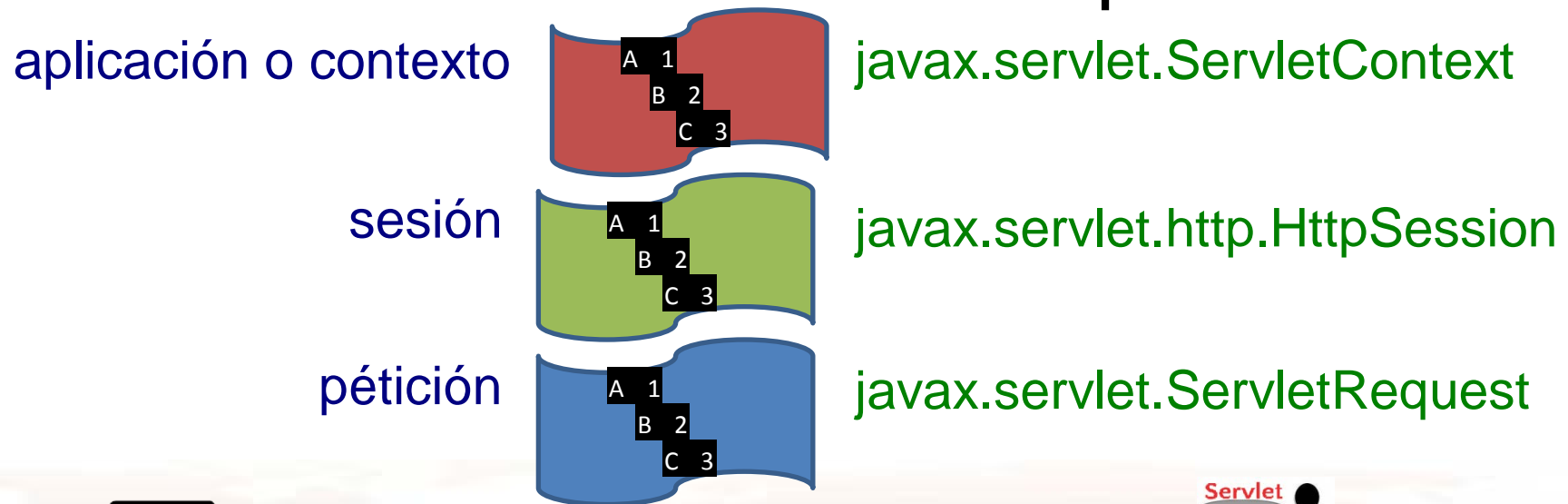
```
protected final void doPost  
    ( HttpServletRequest request, HttpServletResponse response )  
    throws ServletException, IOException  
{  
    // Actualizar base de datos.  
    ...  
    response.sendRedirect  
        ( request.getContextPath() + "/MostrarServlet" );  
}
```



a diferencia de `getRequestDispatcher()`,
aquí sí se debe usar el context path en rutas absolutas

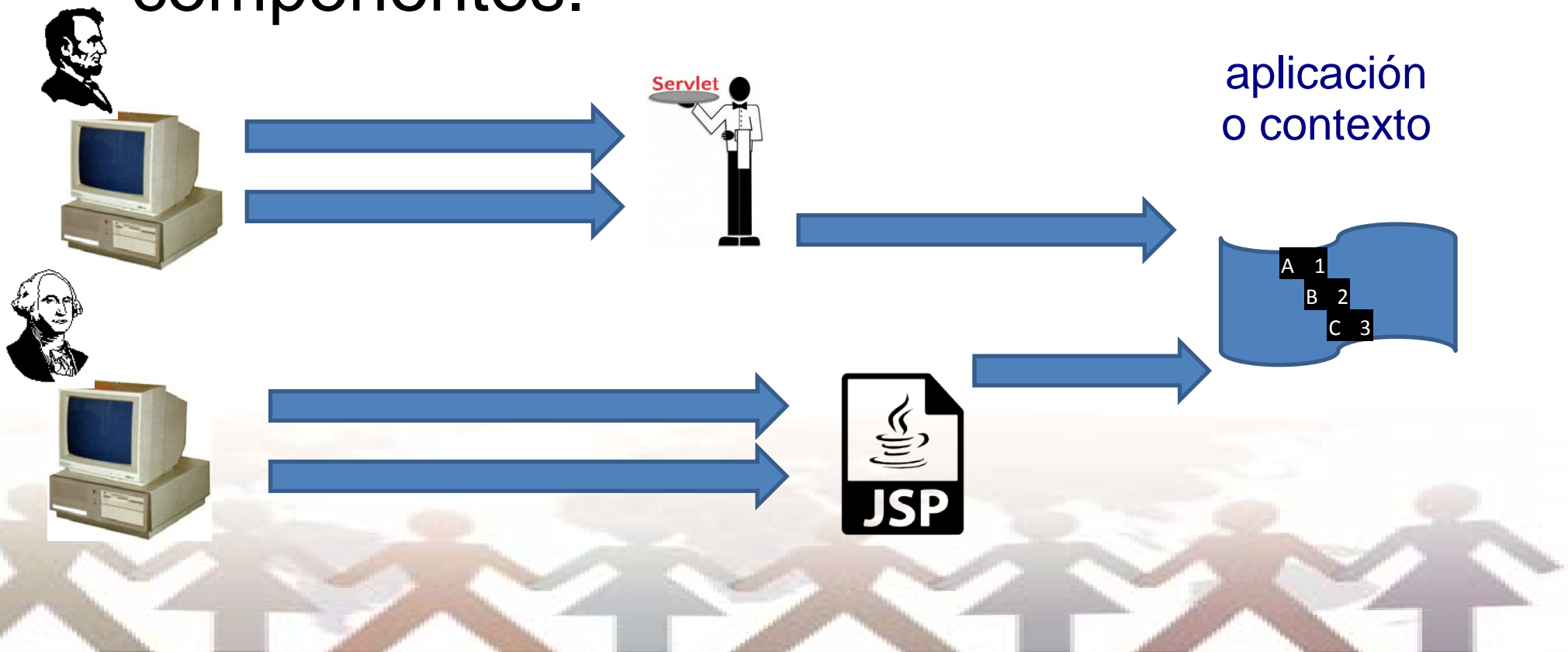
Atributos

- Un componente web puede almacenar información a varios niveles para comunicarse con otros componentes.



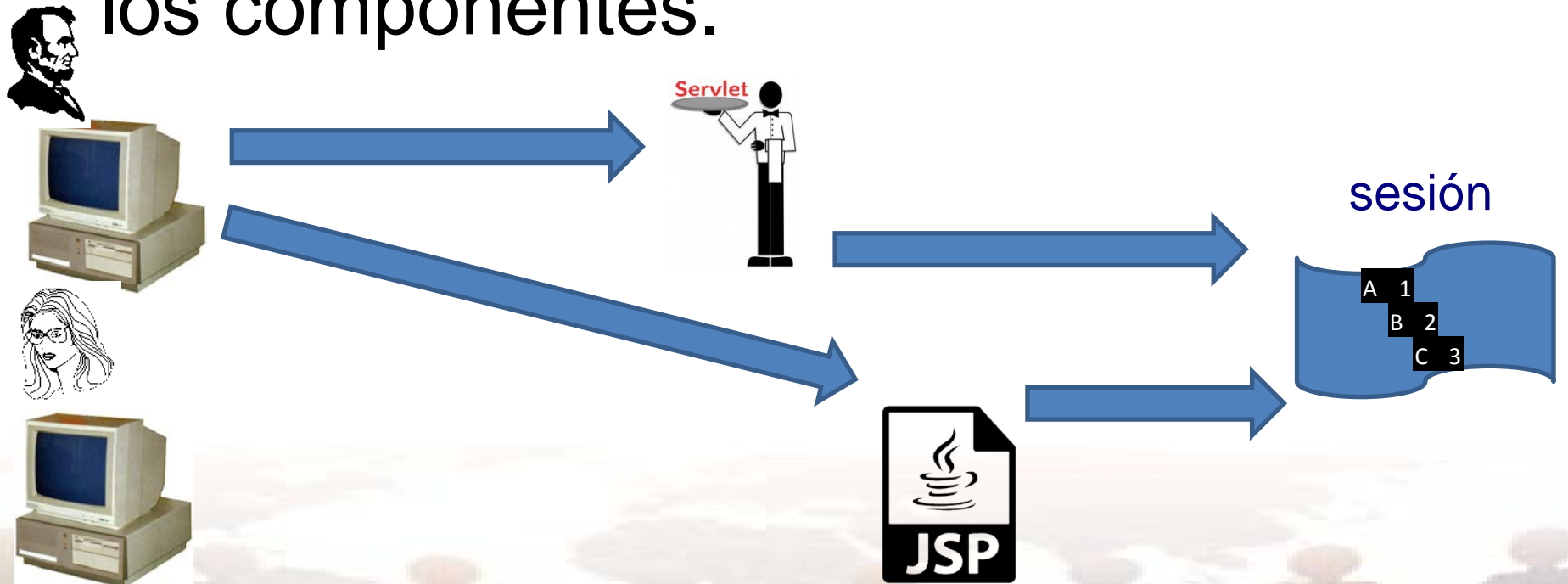
Atributos de Aplicación o Contexto

- Son visibles por todas las peticiones de todos los usuarios hacia todos los componentes.



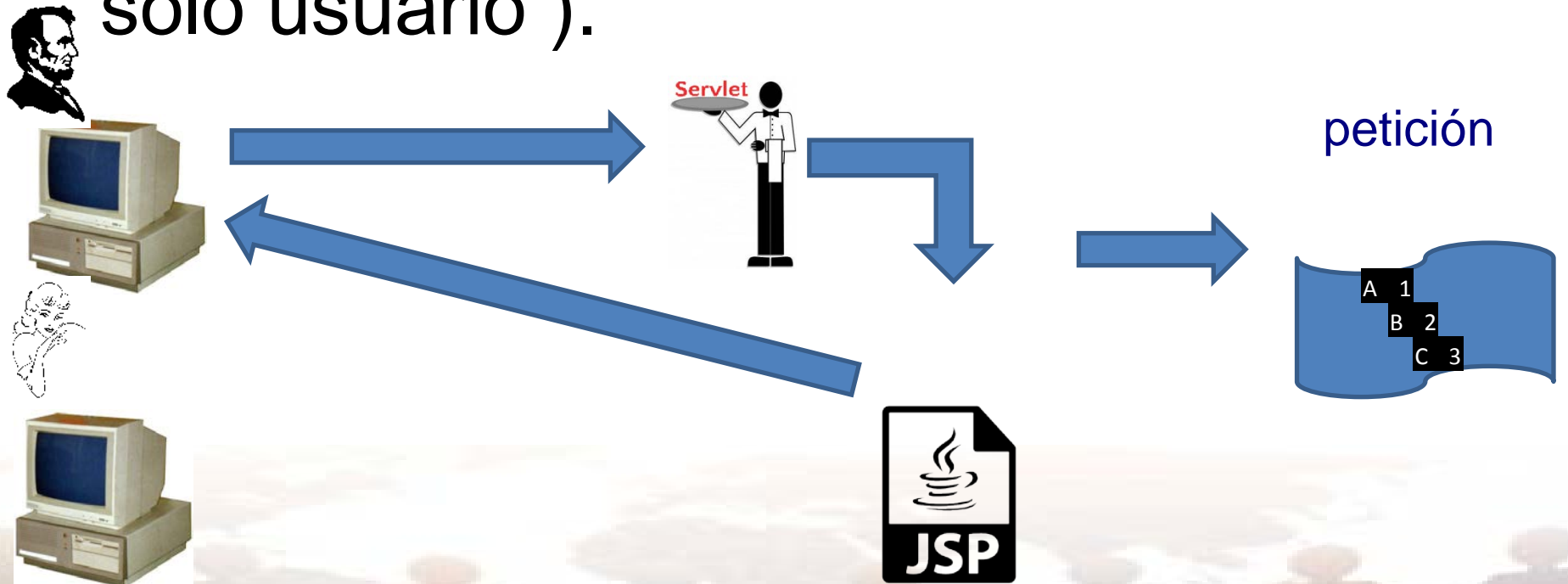
Atributos de Sesión

- Son visibles por todas las peticiones de una sola sesión de usuario hacia todos los componentes.



Atributos de Petición

- Son visibles por todos los componentes que componen una sola petición (de un solo usuario).

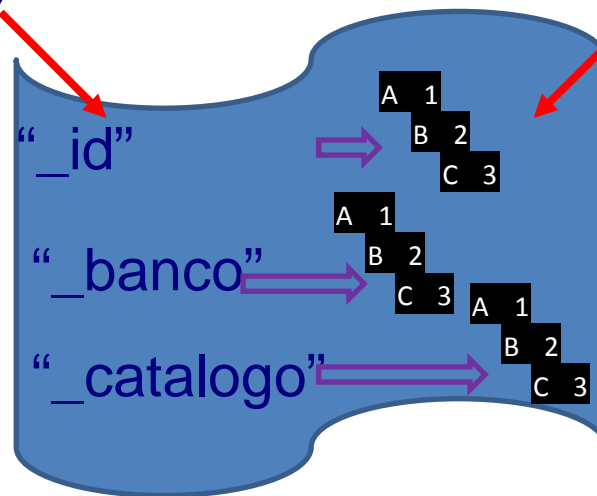


Manejo de Atributos

- Todos los atributos se referencian por nombre.

llave (`java.lang.String`)

valor (`java.lang.Object`)

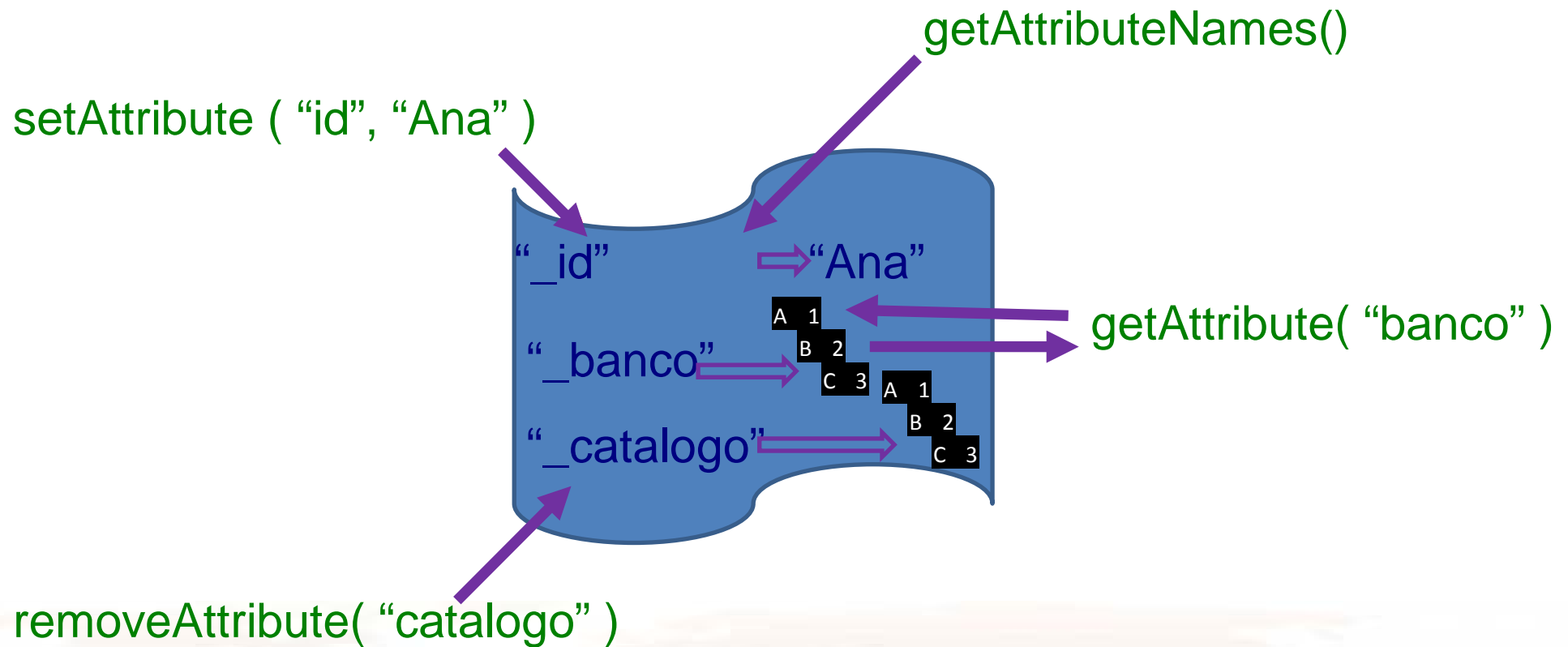


request - `javax.servlet.ServletRequest`

session - `javax.servlet.http.HttpSession`

application - `javax.servlet.ServletContext`

Manejo de Atributos



Ejemplo - Atributos de Petición

```
protected final void doPost  
    ( HttpServletRequest request,  
      HttpServletResponse response )  
throws ServletException, IOException
```

```
{
```

```
    request.setAttribute( "banco", new Banco() );
```

```
    Banco banco = ( Banco )request.getAttribute( "banco" );
```

```
    ...
```

```
}
```

establecer un
atributo

leer un atributo



Ejemplo - Atributos de Sesión

```
protected final void doPost
```

```
( HttpServletRequest request,  
  HttpServletResponse response )
```

```
throws ServletException, IOException
```

```
{
```

```
HttpSession session = request.getSession();
```

```
session.setAttribute( "banco", new Banco() );
```

```
Banco banco = ( Banco )session.getAttribute( "banco" );
```

```
...
```

```
}
```

obtener referencia
a la sesión
del usuario



Ejemplo - Atributos de Contexto / Aplicación

```
protected final void doPost
```

```
( HttpServletRequest request,  
  HttpServletResponse response )
```

```
throws ServletException, IOException
```

```
{
```

```
ServletContext context = getServletContext();
```

```
context.setAttribute( "banco", new Banco() );
```

```
Banco banco= ( Banco)context.getAttribute( "banco" );
```

```
...
```

```
}
```

obtener referencia
al contexto



Preguntas



Resumen





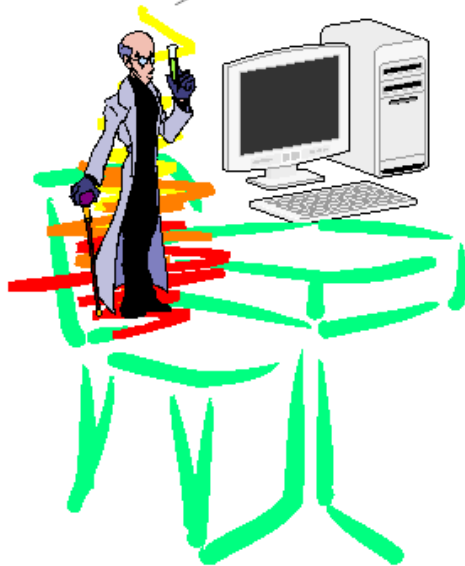
Eclipse 2019-06

FLUJO



Objetivos

iii El objetivo de este módulo es Emplear `forward()` y `sendRedirect()` para pasar una petición de un servlet a una página JSP ,
!!!

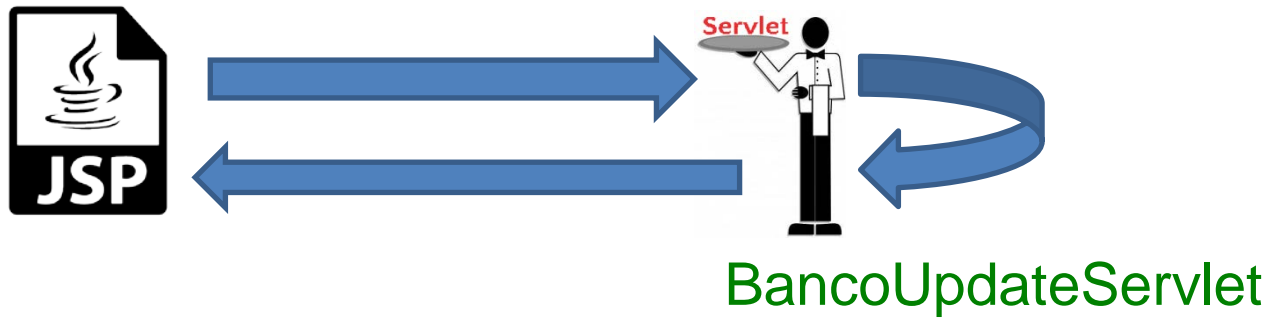


!!!! también comparar ambos enfoques.
iiii



Ejemplo

banco_update_form.jsp



BancoUpdateServlet - Avance con forward()


```
public void doPost  
    ( HttpServletRequest request, HttpServletResponse response )  
throws ServletException, IOException  
{  
    log( "Actualizando Información del Banco" );  
  
    RequestDispatcher dispatcher = request.getRequestDispatcher  
        ( "/bancos_update_form.jsp" );  
    dispatcher.forward( request, response );  
}
```



Actividad - Probar

BancoUpdateServlet.java Catálogo de Bancos

http://localhost:7080/cel/bancos_update_form.jsp



Tienda de Celulares

[Principal](#) | [Misión](#) | [Visión](#) | [Casos de exito](#) | [Contactano](#)

[Logout](#)

Catálogo de Bancos

Actualiza los Campos que se Requieran Modificar

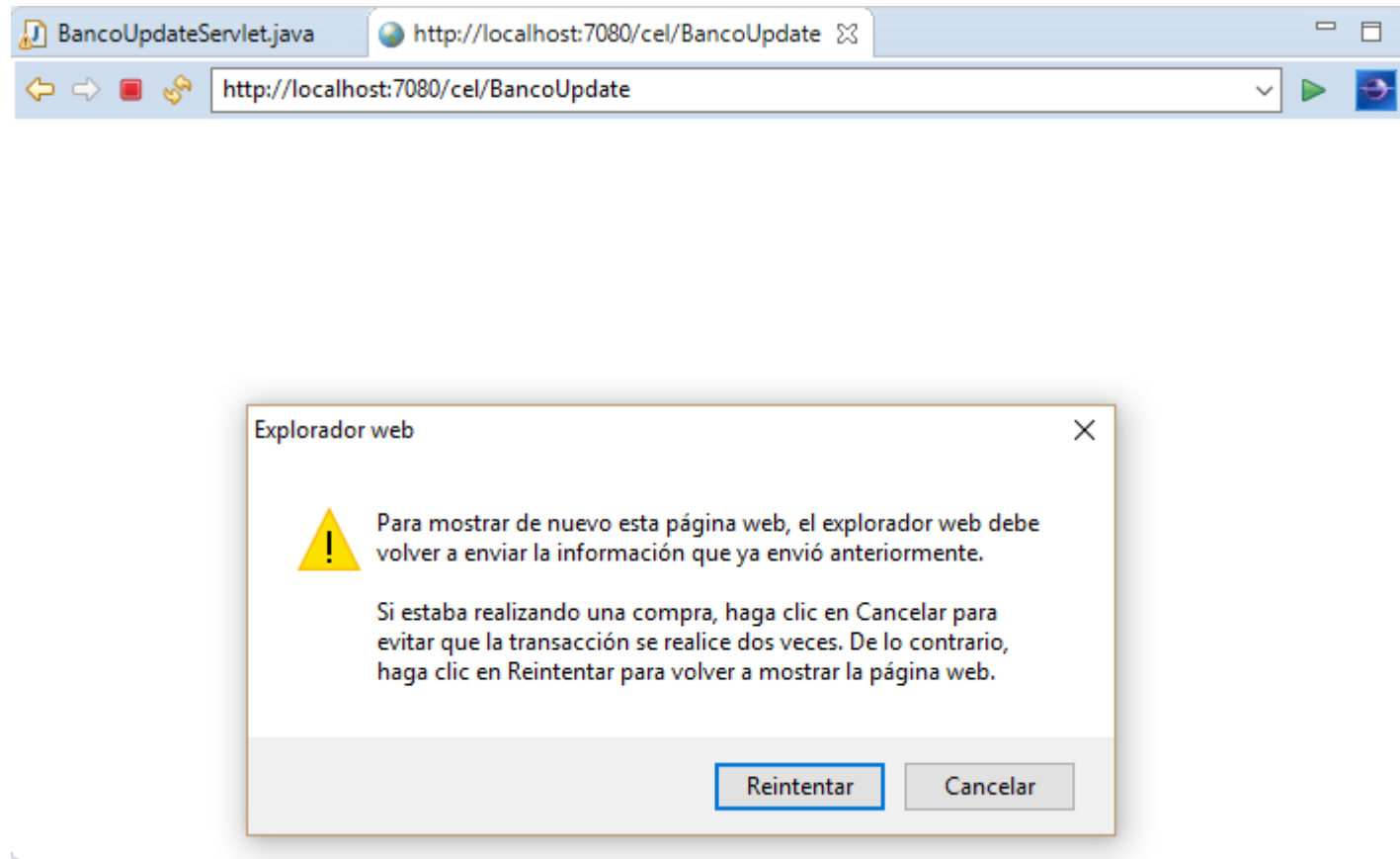
Clave	Nombre	
BANN	<input type="text" value="Ban Norte"/>	<input type="button" value="Modificar"/> <input type="button" value="Borrar"/>
BANX	<input type="text" value="Banamex"/>	<input type="button" value="Modificar"/> <input type="button" value="Borrar"/>

Preguntas

- ¿ Qué se imprime en la bitácora como resultado de la prueba anterior ?
- ¿Cuál es el URL que se despliega en la página resultante ? ¿ Por qué ?



Actividad - Recargar Página



reportar qué sucede, y ¿ porqué ?

BancoUpdateServlet - Avance con sendRedirect()

```
public void doPost  
    ( HttpServletRequest request, HttpServletResponse response )  
throws ServletException, IOException  
{  
    log( " Actualizando Información del Banco " );  
  
    String base = request.getContextPath();  
    response.sendRedirect( base + "/banco_update_form.jsp" );  
}
```



Actividad - Probar

Repetir el escenario de modificación y recarga.

¿Cuál es el URL que se despliega en el servidor?

¿Se repite la carga utilizando el botón **Back**?

¿Porqué?

BancoUpdateServlet.java Catálogo de Bancos

http://localhost:7080/cel/banco_update_form.jsp

Tienda de Celulares

[Principal](#) | [Misión](#) | [Visión](#) | [Casos de exito](#) | [Contactano](#)

Catálogo de Bancos

Actualiza los Campos que se Requeran Modificar

Clave	Nombre	
BANN	Ban Norte	<input type="button" value="Modificar"/> <input type="button" value="Borrar"/>
BANX	Banamex	<input type="button" value="Modificar"/> <input type="button" value="Borrar"/>

Código Completo - BancoUpdateServlet



Resumen

