Desarrollo de Aplicaciones Web

Servlets. Sesiones.

USC
UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Pablo García Tahoces

Servlets. Sesiones.

http es un protocolo sin estado → cada petición es tratada de forma independiente por el servidor → el servidor no puede saber si una petición proviene del mismo o diferentes casos y si están relacionados entre sí.

Solucion.- crear sesiones, que permiten identificar y guardar información individual de cada cliente.

Sesión:

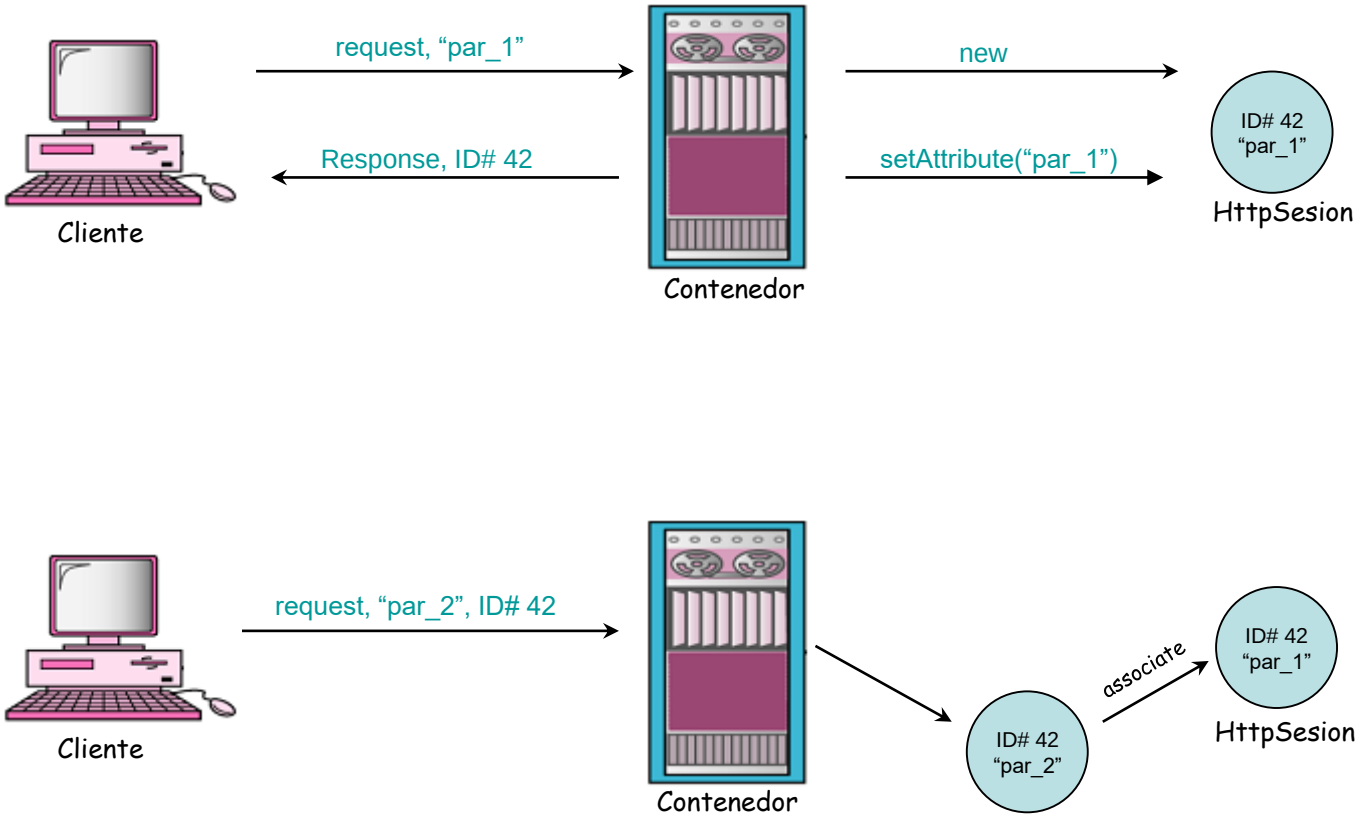
- Inicio.- al realizar una petición al servidor.
- Fin:
  - Al cabo de un tiempo de inactividad.
  - Al cerrarse la aplicación cliente.
  - Cierre por parte del cliente.

Seguimiento sesión:

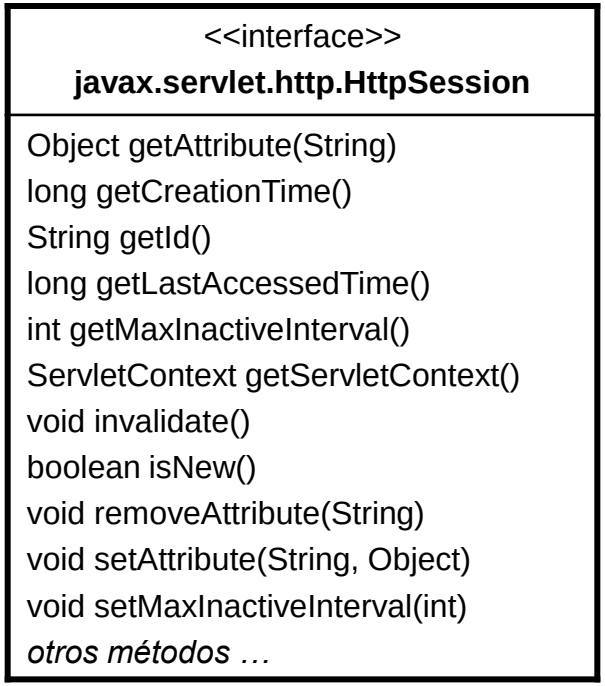
- Cookies
- Reescritura de URL
- Campos ocultos en formularios

Sesiones

Creación de una sesión

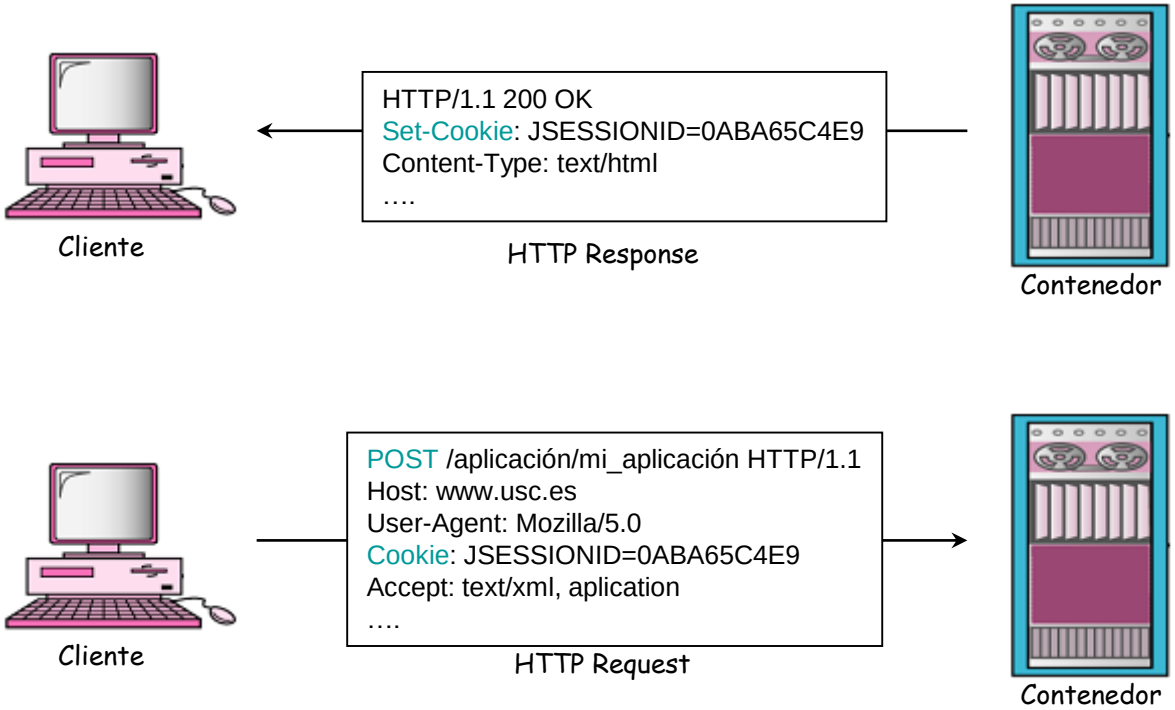


Sesiones. HttpSession Interface.



Sesiones. Cookies.

Cookie.- pequeña cantidad de información que se almacena en la máquina Cliente.



Sesiones. Cookies.

Crear una sesión:

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

Envío de un cookie de sesión en la respuesta:

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

Tomar el ID de sesión en la solicitud:

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

Consultar si la sesión creada es nueva:

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

if (session.isNew())
    _____
else
    _____
```

Sesiones. Cookies.

Consultar si la sesión creada es nueva:

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

if (session.isNew())
    _____
else
    _____
```

Consultar si existe una sesión previa:

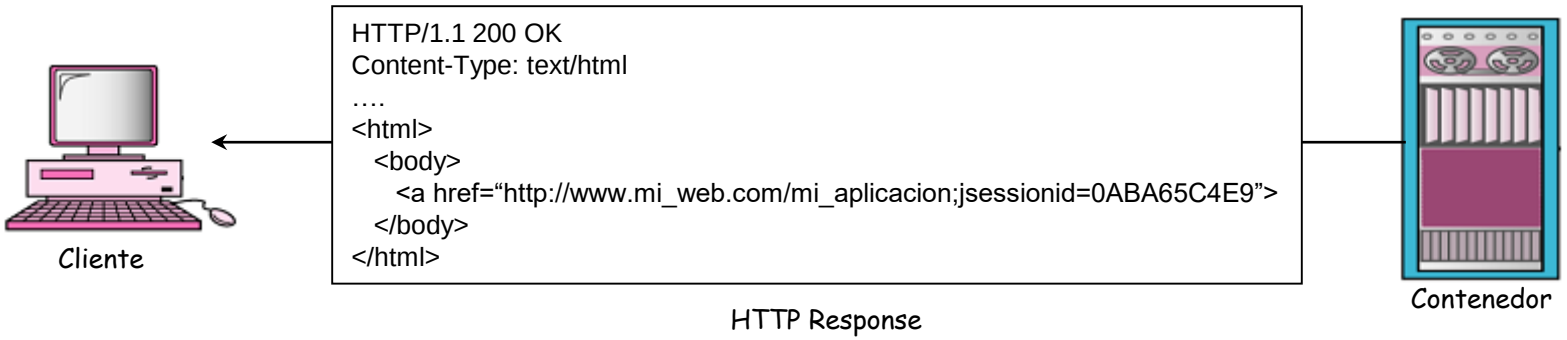
```
HttpSession session= request.getSession(false);

if (session==null)
    _____

else
    _____
```

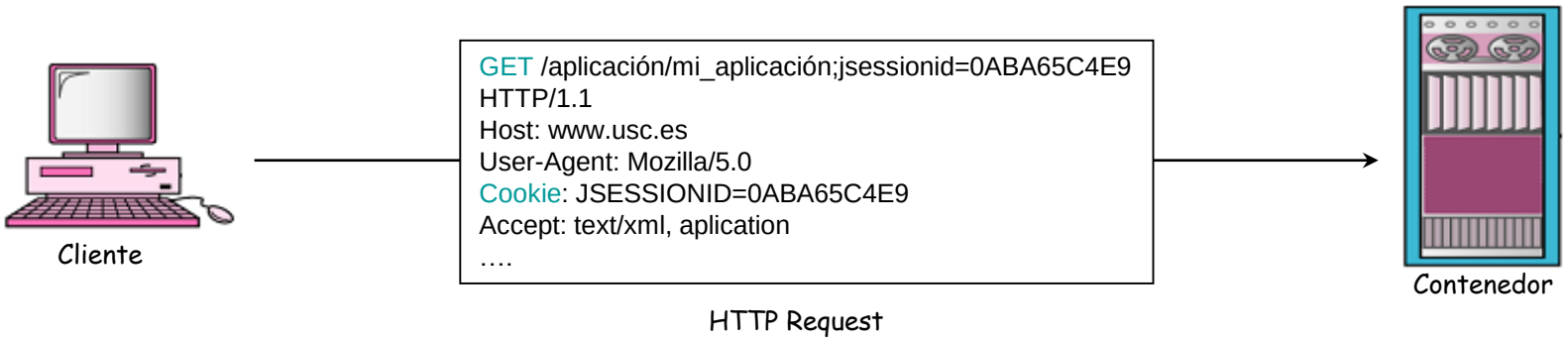
Sesiones. Cookies.

Ej.-

```
public void doGet(HttpServletRequest request, HttpServletResponse response)
                                        throws IOException, Servlet Exception{

        response.setContentType("text/html");
        PrintWriter out= response.getWriter();
        out.println("Testeo de sesiones");

        HttpSession session= request.getSession(false);

        if (session==null){
            out.println("No hay sesiones disponibles");
            out.println("Abriendo una sesión nueva…");
            session= request.getSession( );
        }
        else
            outprintln("Hay una sesión abierta");
}
```

Sesiones. Reescritura de URL.





Sesiones. Reescritura de URL.

Mediante los métodos: *encodeURL()* y *encodeRedirectURL()* del objeto *response*

Ej.-

```
public void doGet(HttpServletRequest request, HttpServletResponse response)
                                        throws IOException, Servlet Exception{

        response.setContentType("text/html");
        PrintWriter out= response.getWriter();
        HttpSession session= request.getSession( );

        out.println("<html><body>");
        out.println("<a href=\""+response.encodeURL("miaplicación")+"\">púlsame</a>");
        out.println("<body><html>");
}
```

Sesiones. *Timeout.*

Existen tres métodos:

i) Mediante configuración en xml de despliegue:

Ej.-

```
<web-app>
        <servlet>
            . . .
        </servlet>
        <session-config>
                <session-timeout>15</session-timeout>
        </session-timeout>
</web-app>
```

Sesiones. *Timeout.*

ii) Mediante especificación de tiempo máximo:

Ej.-

```
session.setMaxInactiveInterval(15*60);
```

iii) Mediante métodos de finalización de sesión:

Ej.-

```
session.invalidate();
```