

JAVA: Servlets



Diseño de aplicaciones web

mperez@fi.upm.es



Servlets

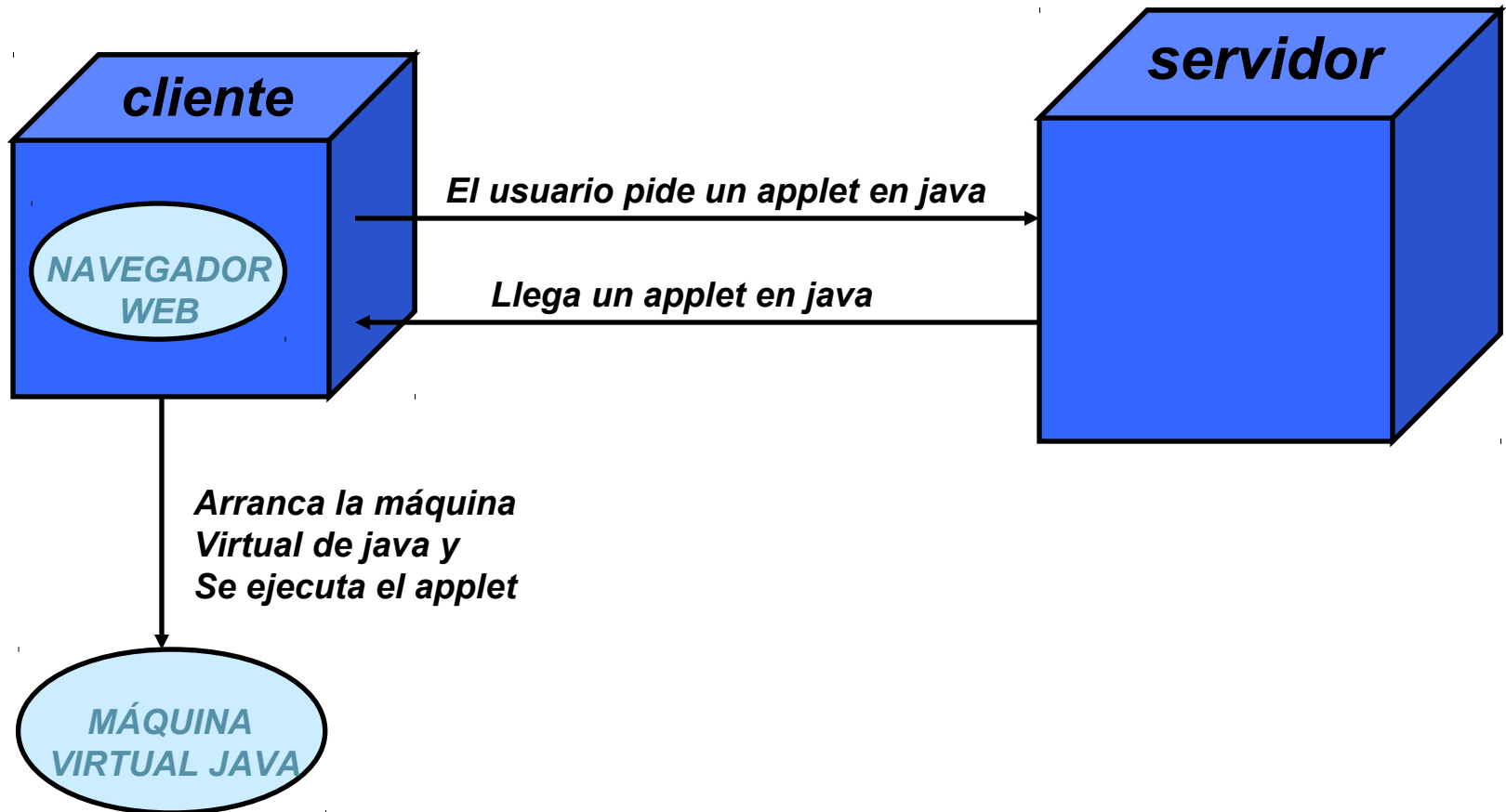
- Programas que se ejecutan en los servidores.
 - Añaden funcionalidad a un servidor web, del mismo modo que los applets añaden funcionalidad a los navegadores.
- Similares a los *scripts* CGI, salvo que ofrecen una independencia de la plataforma.



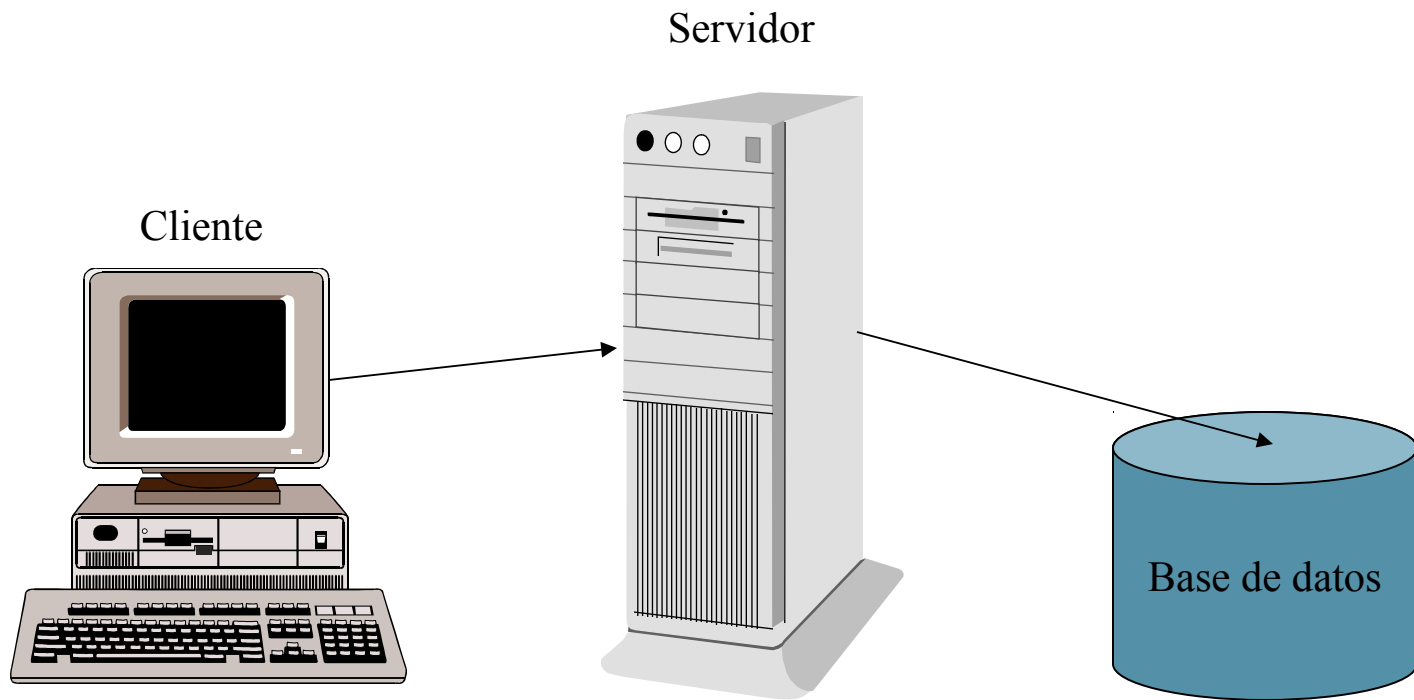
Cuándo se suelen utilizar

- La página web se basa en datos que proporciona el usuario.
 - Ejemplo: *e-commerce sites*.
- Los datos cambian frecuentemente.
 - Ejemplo: Partes meteorológicos.
- La página web utiliza información de BDs u otras fuentes.
 - Ejemplo: Aplicaciones comerciales.

Applets



Servlets





Servlets. Requisitos

- El servidor debe tener una máquina virtual Java.
- El servidor debe soportar la API de los servlets Java.
- Servidores capaces de ejecutar servlets:
 - Apache Tomcat
 - Java Web Server
 - O'Reilly WebSite Professional
 - Lotus Domino Go WebServer
 - Novel IntraNetWare
 - IBM Internet Connection Server
 - Otros



Instalando un servidor Web

- Jakarta Tomcat

- <http://java.sun.com/webservices/downloads/webservicespack.html>

- JSWDK 1.0

- <http://java.sun.com/products/servlet/2.1>



Java Servlet API

■ Extensión al JDK estándar

- Extensiones del JDK extensiones son empaquetados bajo javax

■ Paquetes:

- javax.servlet
- javax.servlet.http
 - Da soporte al protocolo HTTP y a la generación de HTML



Ejecución de un servlet

- Formas de iniciar la ejecución:
 - Introducir la dirección URL del servlet en un navegador web.
 - Llamar al servlet desde una página web.
 - Ejecutar un servlet llamándolo desde otro servlet.



Desde un navegador web

- `http://nombre_maquina:puerto/ruta_servlet/nombre_servlet`
- Las llamadas a servlets pueden contener parámetros:
 - `http://www.datsi.fi.upm.es/~mperez/servlets/serv1?num1=3&num2=4`



Desde una página web

- Un *servlet* también puede ser llamado desde el código de una página web, al igual que se llamaría a cualquier CGI.
- Ejemplo:
 - `<form action =`
 `“http://www.datsi.fi.upm.es/~mperez/servlet`
 `s/serv2” method=“post”>`



Desde otro servlet

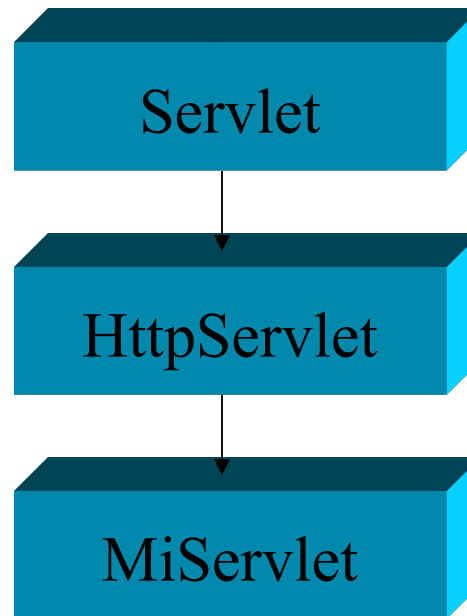
- Se puede lanzar la ejecución de un servlet desde otro.
- Pasos:
 - Conocer el nombre del servlet que queremos llamar.
 - Proporcionar acceso al objeto “Servlet” del servlet llamado.
 - Llamar al método público del servlet.

Desde otro servlet

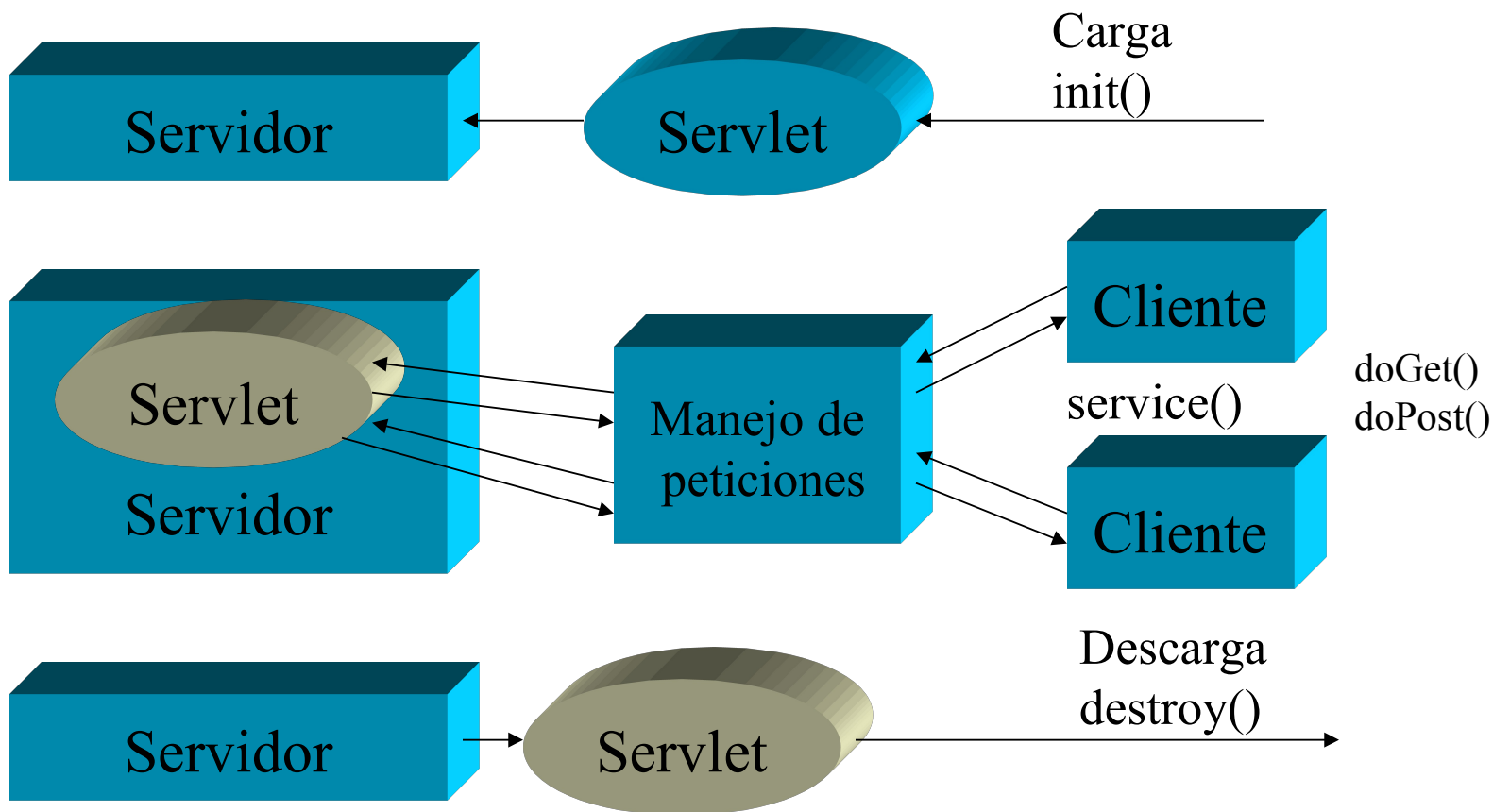
```
public class servletCalling extends HttpServlet {
    public void doGet (HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException
    {
        ...
        Servlet servlet_get = (Servlet)
        getServletConfig().getServletContext().getServlet("servl
        etCalled");
        String data_get = servlet_get.method1(data);
        ...
    }
}
```

Arquitectura de los servlets

- Biblioteca 'javax.servlet'



Ciclo de vida de un servlet





Inicialización de un servlet

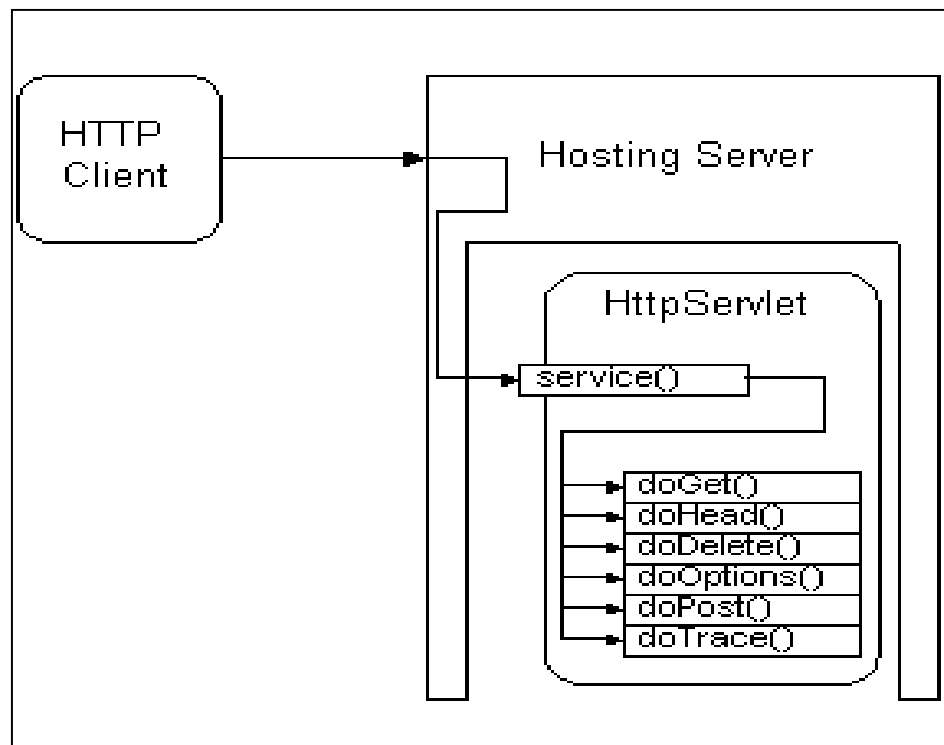
- `public void init (ServletConfig config)`
- Finaliza antes de la invocación de cualquier método sobre el servlet
- Sólo se invoca una vez, a menos que el servidor recargue el servlet
- `ServletConfig`: argumentos de inicialización para el servlet
- Ej.: abrir ficheros o establecer conexiones a los servidores



Servicio

- public void service (ServletRequest req, ServletResponse res)
- Lee la petición y produce el mensaje de respuesta
- Objeto '*ServletRequest*':
 - Comunicación que fluye del cliente al servidor
- Objeto '*ServletResponse*':
 - Encapsula la información enviada desde el servidor al cliente

HTTP Servlets



Método HTTP GET

■ Ejemplo:

```
- GET /servlet/MyServlet?nombre=Juan& institucion=FI
HTTP/1.1
Connection: Keep-Alive
User-Agent: Mozilla/4.0 (
compatible; MSIE 4.01; Windows NT)
Host: www.datsi.fi.upm.es
Accept: image/gif, image/x-xbitmap, image/jpeg,
image/pjpeg
```

- Limitación: Cuántos datos son pasados como parte del URL ⇒ Uso de HTTP POST

Método HTTP POST

- Permite al cliente enviar datos al servidor
 - Pasar más información que en una petición HTTP GET

- Ejemplo:

```
- POST /servlet/MyServlet HTTP/1.1
User-Agent: Mozilla/4.0 (
compatible; MSIE 4.01; Windows NT)
Host: www.datsi.fi.upm.es
Accept: image/gif, image/x-xbitmap, image/jpeg,
image/pjpeg, */
Content-type: application/x-www-form-urlencoded
Content-length: 39
nombre=Juan&institucion=FI
```

doGet() y doPost()

■ Sobrecribir métodos doGet() y doPost()

```
- public void doGet (  
    HttpServletRequest request,  
    HttpServletResponse response) ;  
  
- public void doPost (  
    HttpServletRequest request,  
    HttpServletResponse response) ;
```



Dstrucción del servlet

- `public void destroy()`
- Liberar los recursos (cerrar ficheros abiertos o cerrar conexiones con bases de datos). Puede ser un método vacío.
- El servidor espera a llamar al método `destroy()` hasta que todos los servicios se hayan completado o haya pasado una cantidad de tiempo determinada

Estructura de un Servlet

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class SomeServlet extends HttpServlet {
    // El servidor envía una página
    // web al cliente
    public void doGet      (HttpServletRequest
        request,
        HttpServletResponse response)
        throws ServletException,
        IOException
    {
        // Utilizar request para leer
        // datos procedentes del
        // cliente (ej: cookies,
        // otros datos)

        // Utiliza response para
        // especificar la respuesta
        // (tipo de contenido, establecer
        // cookies, ...)

        PrintWriter out = response.getWriter();

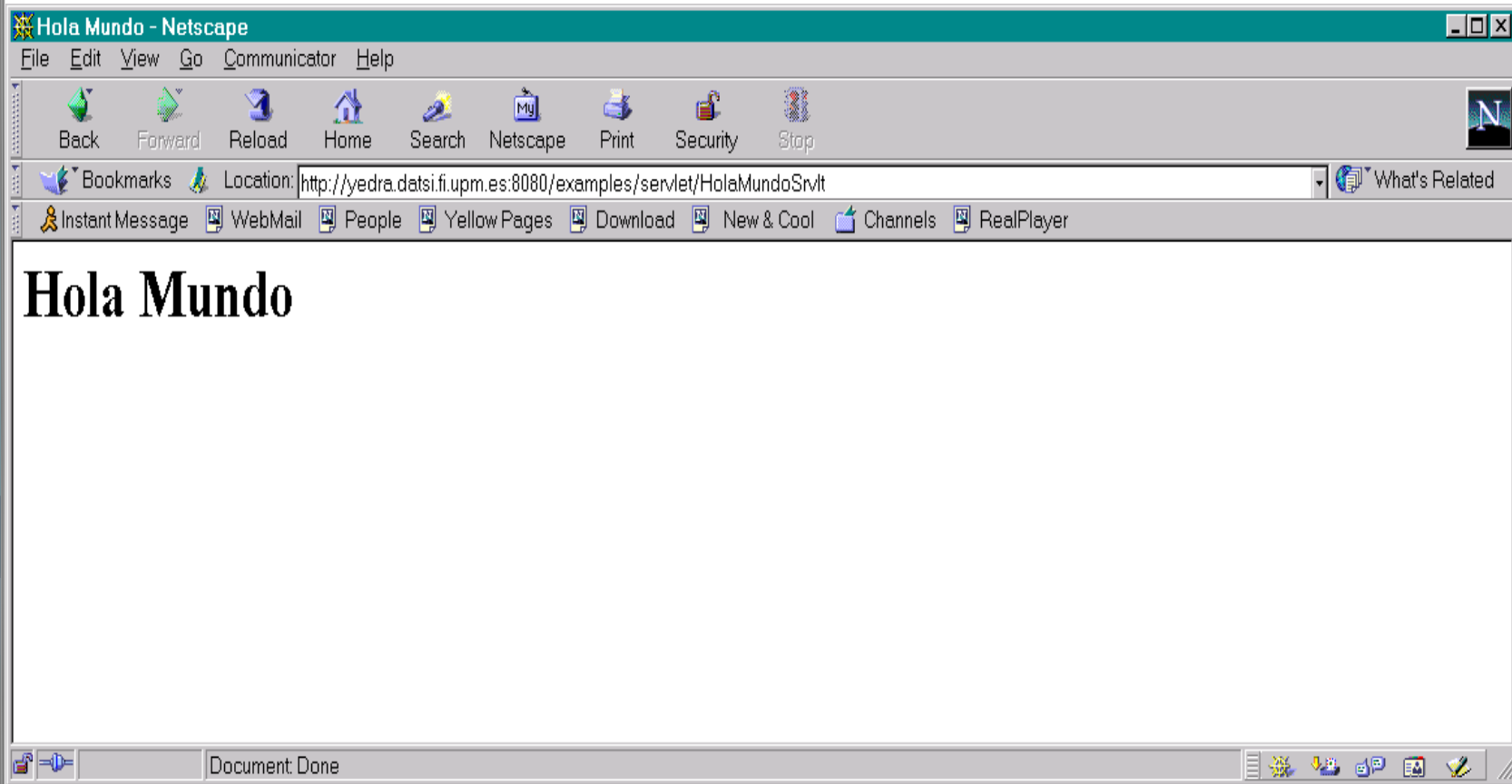
        // Utilizar out para enviar el
        // contenido al navegador

    }
    // Análogamente, sobrescribir
    // doPost().
}
}
```

Ejemplo Servlets: Hola Mundo

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class HolaMundoServlet extends
    HttpServlet {
    // El servidor envía una página web al
    // cliente
    public void doGet
    (HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException,
    IOException
    {
        PrintWriter out;
        String titulo="Hola Mundo";
        // Utiliza response para
        // especificar la respuesta
        response.setContentType ("text/html");
        out = response.getWriter();
        out.println("<HTML><HEAD><TITLE>");
        out.println(titulo);
        out.println("</TITLE></HEAD><BODY>");
        out.println("<H1>"+titulo+"</H1>");
        out.println("</BODY></HTML>");
        out.close();
    }
}
```


Ejemplo Servlets: Hola Mundo





Obtención y envío de información

■ Obtención de datos del cliente:

- *getParameter()*
 - *getParameterValues()*
 - *getParameterNames()*
- *getReader*
- *getInputStream()*

■ Envío de datos al cliente:

- *getWriter*
- *getOutputStream*

Ejemplo: Lectura de parámetros

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class ThreeParam extends HttpServlet {
    public void doGet( HttpServletRequest
        request,
        HttpServletResponse response)
        throws ServletException,
        IOException {

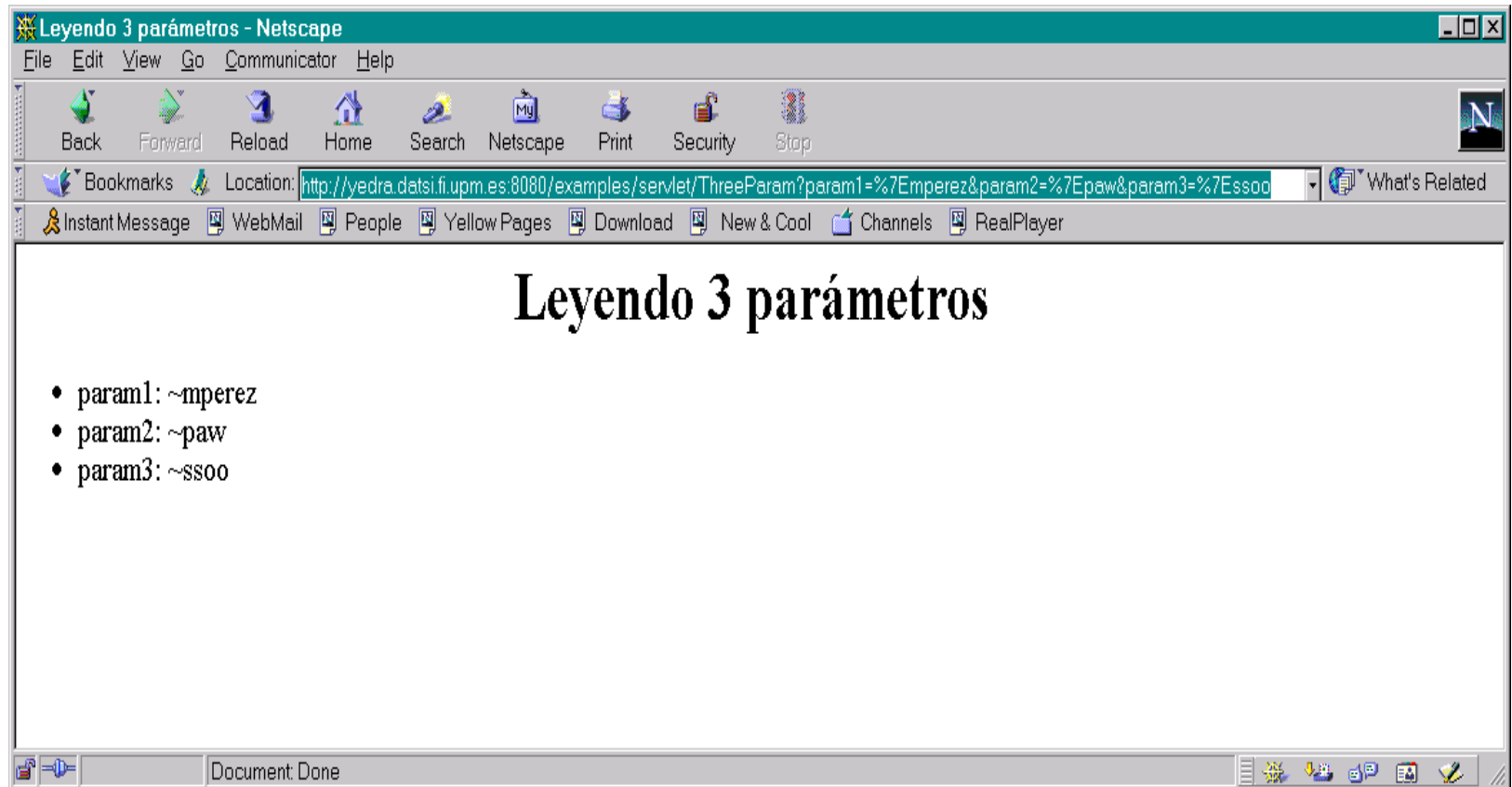
        PrintWriter out;
        String title="Leyendo 3 parámetros";

        response.setContentType
            ("text/html");
        out = response.getWriter();
        out.println("<HTML><HEAD><TITLE>");
        out.println(title);
```

```
        out.println("</TITLE></HEAD><BODY>");
        out.println("<H1 ALIGN=CENTER>"+ title
            +"</H1>");
        out.println("<UL>");
        out.println("<LI>param1: "+
            request.getParameter("param1"));
        out.println("<LI>param2: "+
            request.getParameter("param2"));
        out.println("<LI>param3: "+
            request.getParameter("param3"));
        out.println("</UL>");
        out.println("</BODY></HTML>");
        out.close();
    }

    public void doPost(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        doGet(request, response);
    }
}
```

Ejemplo: Lectura de parámetros



Ejemplo: Lectura de parámetros II

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;

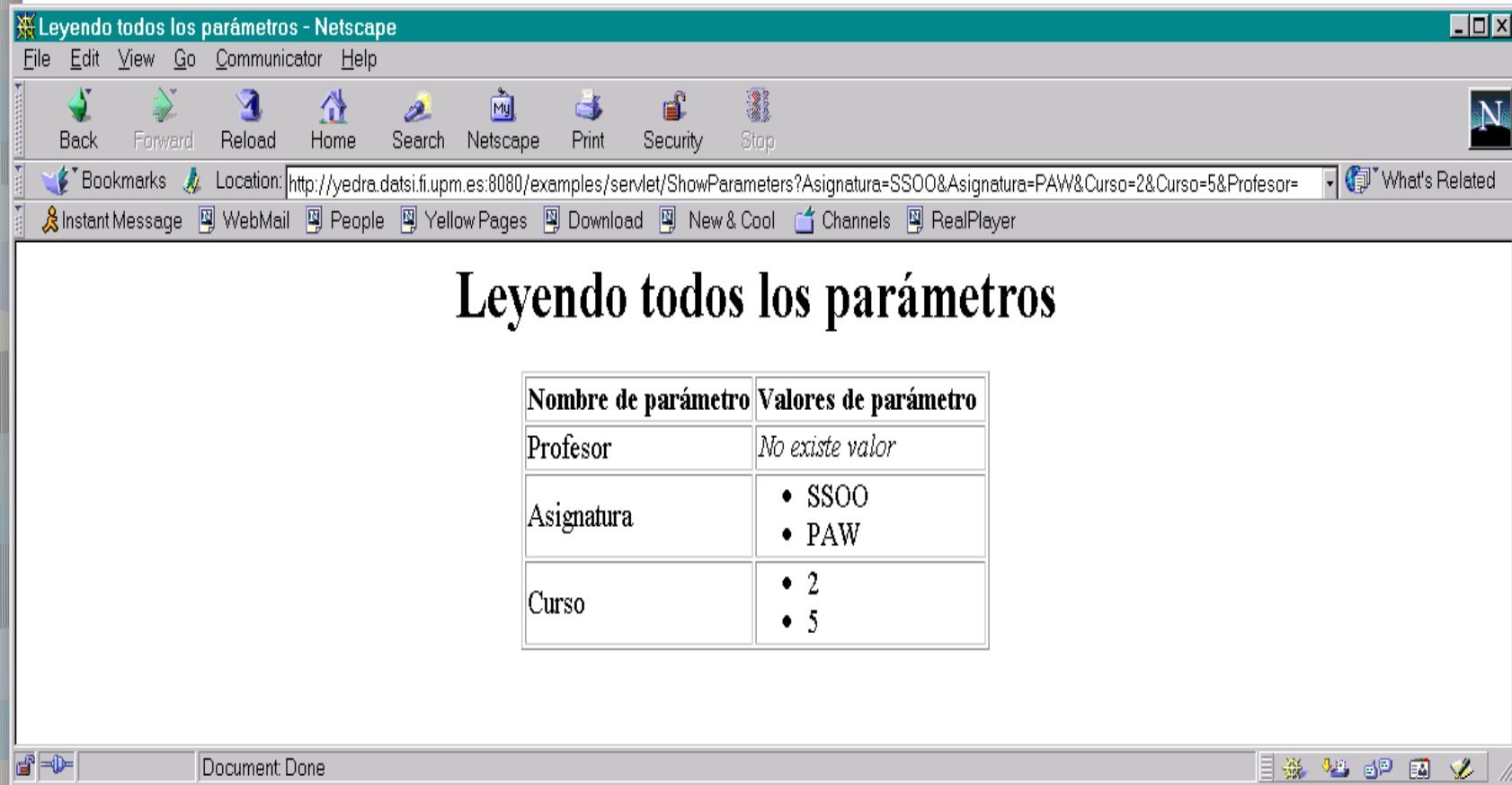
public class ShowParameters extends HttpServlet {
    public void doGet( HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException
    {
        PrintWriter out;
        String title="Leyendo todos los parámetros";
        response.setContentType ("text/html");
        out = response.getWriter();
        out.println("<HTML><HEAD><TITLE>");
        out.println(title);
        out.println("</TITLE></HEAD><BODY>");
    }
}
```

Ejemplo: Lectura de parámetros II

```
out.println("<H1 ALIGN=CENTER>" + title
+"</H1>");
out.println("<TABLE BORDER=1 ALIGN=CENTER>");
out.println("<TR><TH>Nombre de
parámetro<TH>Valores de parámetro");
Enumeration nombresDeParam =
request.getParameterNames();
while (nombresDeParam.hasMoreElements()) {
    String nombreParam = (String)
nombresDeParam.nextElement();
out.println("<TR><TD>" + nombreParam);
out.println("<TD>");
String[] valoresDeParam=
request.getParameterValues(nombreParam);
if (valoresDeParam.length == 1) {
    String valorParam =
valoresDeParam[0];
    if (valorParam.length() == 0)
        out.print("<I>No existe valor</I>");
```

```
else
    out.print(valorParam);
} else {
    out.println("<UL>");
    for (int i=0;
i<valoresDeParam.length; i++) {
        out.println("<LI>" +
valoresDeParam[i]);
    }
    out.println("</UL>");
}
}
out.println("</TABLE>");
out.println("</BODY></HTML>");
out.close();
}
public void doPost(HttpServletRequest request,
HttpServletResponse response)
throws ServletException, IOException {
    doGet(request, response);
}
}
```

Ejemplo: Lectura de parámetros II



The screenshot shows a Netscape browser window with the title 'Leyendo todos los parámetros - Netscape'. The address bar contains the URL: `http://yedra.datsi.fi.upm.es:8080/examples/servlet/ShowParameters?Asignatura=SSOO&Asignatura=PAW&Curso=2&Curso=5&Profesor=`. The main content area displays the title 'Leyendo todos los parámetros' and a table with the following data:

Nombre de parámetro	Valores de parámetro
Profesor	<i>No existe valor</i>
Asignatura	<ul style="list-style-type: none">• SSOO• PAW
Curso	<ul style="list-style-type: none">• 2• 5

Ejemplo: Uso de formulario

```
<HTML>
<HEAD>
  <TITLE> Un ejemplo de formulario que utiliza POST </TITLE>
</HEAD>

<H1 ALIGN=CENTER> Un ejemplo de formulario que utiliza POST </H1>

<FORM ACTION="../servlet/ShowParameters" METHOD="POST">
  Número Artículo:
  <INPUT TYPE="TEXT" NAME="numItem"> <BR>
  Cantidad:
  <INPUT TYPE="TEXT" NAME="cantidad"> <BR>
  Precio Unidad:
  <INPUT TYPE="TEXT" NAME="precio" VALUE="$"> <BR>
  <HR>
  Nombre:
  <INPUT TYPE="TEXT" NAME="nombre"> <BR>
  Apellidos:
  <INPUT TYPE="TEXT" NAME="apellidos"> <BR>
```


Ejemplo: Uso de formulario

Dirección:

```
<TEXTAREA NAME="direccion" ROWS=3 COLS=40> </TEXTAREA><BR>
```

Tarjeta de crédito:


```
<INPUT TYPE="RADIO" NAME="tipoTarjeta" VALUE="Visa">Visa<BR>
```

```
<INPUT TYPE="RADIO" NAME="tipoTarjeta" VALUE="Master Card">Master Card<BR>
```

```
<INPUT TYPE="RADIO" NAME="tipoTarjeta" VALUE="Amex">American Express<BR>
```

```
<INPUT TYPE="RADIO" NAME="tipoTarjeta" VALUE="Discover">Discover<BR>
```

Número de tarjeta de crédito:

```
<INPUT TYPE="PASSWORD" NAME="numTarjeta"> <BR>
```

Repetición del número de tarjeta de crédito:

```
<INPUT TYPE="PASSWORD" NAME="numTarjeta"> <BR><BR>
```

```
<CENTER>
```

```
<INPUT TYPE="SUBMIT" VALUE="Enviar solicitud">
```

```
</CENTER>
```

```
</FORM>
```

```
</BODY>
```

```
</HTML>
```

Ejemplo: Uso de formulario

Un ejemplo de formulario que utiliza POST - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security Stop

Bookmarks Location: <http://yedra.datsi.fi.upm.es:8080/examples/servlets/PostForm.html> What's Related

Instant Message WebMail People Yellow Pages Download New & Cool Channels RealPlayer

Un ejemplo de formulario que utiliza POST

Número Artículo:

Cantidad:

Precio Unidad:

Nombre:

Apellidos:

Dirección:

Tarjeta de crédito:

Visa

Master Card

American Express

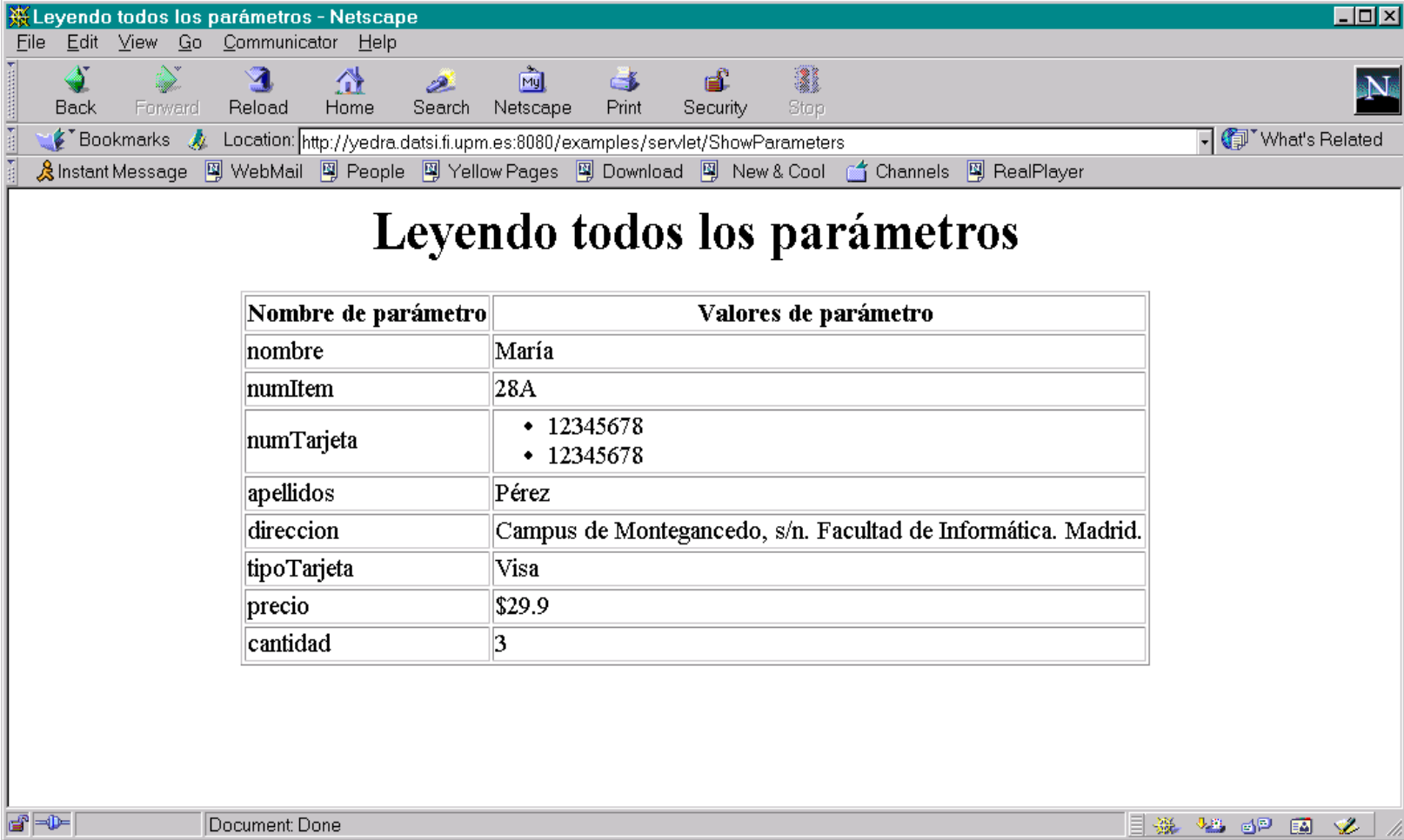
Discover

Número de tarjeta de crédito:

Repetición del número de tarjeta de crédito:

Document: Done

Ejemplo: Uso de formulario



The screenshot shows a Netscape browser window with the title "Leyendo todos los parámetros - Netscape". The address bar contains the URL "http://yedra.datsi.fi.upm.es:8080/examples/servlet/ShowParameters". The page content features a table with the following data:

Nombre de parámetro	Valores de parámetro
nombre	María
numItem	28A
numTarjeta	<ul style="list-style-type: none">• 12345678• 12345678
apellidos	Pérez
direccion	Campus de Montegancedo, s/n. Facultad de Informática. Madrid.
tipoTarjeta	Visa
precio	\$29.9
cantidad	3



Almacenamiento del estado de un cliente

■ Estado del cliente:

- Situación en la que se encuentra un cliente en sucesivas peticiones al servidor
- Ejemplo típico de aplicación: carrito de la compra (almacenamiento de productos por parte del cliente).

■ Mecanismos:

- “Cookies”
- Seguimiento de sesiones (*session tracking*)

Uso de cookies

- `Cookie miCookie = new Cookie("Ciudad", "Madrid");`

- **Métodos sobre cookies:**

- `setValue() / getValue()`: Valores de una cookie
- `setComment() / getComment()`: Comentario de una cookie
- `setMaxAge() / getMaxAge()`: Tiempo de caducidad de una cookie
- `getName()`: Nombre de una cookie

- **Las cookies se envían en la cabecera de la respuesta al cliente**

- `addCookie()`, sobre el objeto de la clase `HttpServletResponse`

Establecimiento de una cookie

```
public void doPost (HttpServletRequest request,
HttpServletRequest response) throws
ServletException, IOException {
    PrintWriter out;
    ...
    Cookie miCookie = new Cookie("Ciudad", "Madrid");
        miCookie.setComment("Cookie para
establecer la ciudad de origen");
miCookie.setMaxAge(3600);
response.addCookie(miCookie);
out = response.getWriter();
    ...
}
```

Recuperación de una cookie

- `getCookies()`: devuelve un array de objetos de tipo `Cookie`

```
public void doPost (HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
    String valorCookie;
    boolean encontrado = false;
    int i = 0;
    Cookie[] misCookies;
    Cookie miCookie;
    misCookies = request.getCookies();
    while (!encontrado && i<misCookies.length)          {
    miCookie = misCookies[i];                            encontrado =
    miCookie.getName().equals("Ciudad");                 if
    (encontrado)                                       valorCookie =
    miCookie.getValue();                               i++;
    }...
}
```



Acceso a bases de datos

- JDBC, interfaz de acceso a un sistema de gestión de bases de datos o RDBMS
- Paquete `java.sql`: contiene un gran número de clases e interfaces útiles para la programación del acceso a bases de datos mediante JDBC
- Clases más útiles:
 - Driver
 - Connection
 - Statement
 - ResultSet



Servlets vs CGI

■ Eficiencia

- CGI inicia un nuevo proceso para cada petición HTTP. (Solución: FastCGI)
- El servlet se ejecuta solamente la primera vez que es llamado. Permanece en memoria una vez cargado y puede compartir información entre varias llamadas de clientes. Uso de threads para las distintas peticiones.

■ Portabilidad

■ Modularidad

■ El uso de parámetros es más sencillo en el caso de los servlets

Servlets y JSP

■ JSP (Java Server Pages)

- Páginas dinámicas añadiendo funcionalidad al código HTML
- Similar a ASP o PHP, aunque puede utilizar diferentes plataformas como servidores
- Extensión de la tecnología de servlets
 1. Cliente realiza una petición de una página JSP al servidor web
 2. Éste envía la solicitud al motor de JSP
 3. Compila la página JSP, convirtiéndola en un servlet
 4. Ejecuta el servlet
 5. Devuelve los resultados al cliente en formato HTML.