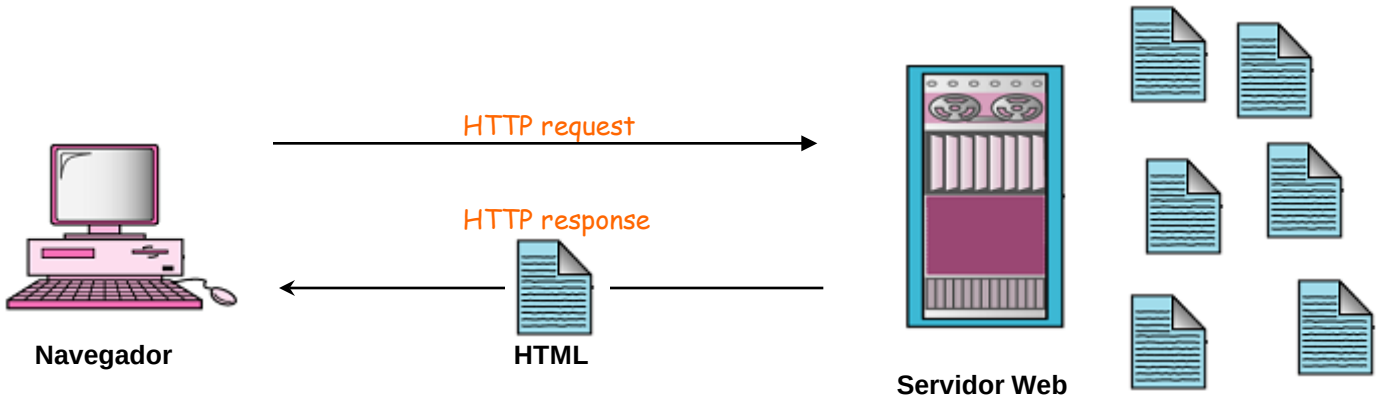
# Desarrollo de Aplicaciones Web

## *Servlets*. Introducción.

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

*Pablo García Tahoces*

# Aplicación Web.



# Servlets

*Servlet*.- programa java que se ejecuta en el contenedor web de un servidor de aplicaciones.

Contenedor web.- interfaz entre un componente y el sistema de inferior nivel que da soporte al componente.

Invocación.- un servlet es invocado por un programa cliente, mediante el uso del protocolo HTTP .

# Servlets. Características.

• Están escritos en Java → independiente de la plataforma.

• Consumen pocos recursos → se cargan la primera vez, las siguientes veces se crean hilos (multihilo).

• Se precompilan antes de su ejecución. No sucede así con otro tipo de tecnologías tales como CGI (*Common Gateway Interface*).

• Son seguros y portables → se ejecutan bajo la máquina virtual Java
utilizan el mecanismo de excepciones Java
utilizan el administrador de seguridad Java

• No requieren soporte para Java en el navegador.

# Servlets. Estructura.

Desde el punto de vista de Java → servlet es un objeto de alguna de las clases de la API Java Servlet, que implementa la interfaz Servlet.

API Java Servlet. Dos paquetes:

- javax.servlet.
- javax.servlet.http (específico de http).

Clases utilizadas para crear un servlet:

- GenericServlet → implementa un servicio genérico.
- HttpServlet → implementa servicios específicos HTTP.

java.lang.object
javax.servlet.GenericServlet
javax.servlet.http.HttpServlet
*mi servlet*

# Servlets. Construcción.

Para crear un *servlet*, únicamente se necesita:

- Escribir clases derivadas de las clases *GenericServlet* del paquete *javax.servlet*, o bien de la clase *HttpServlet* del paquete *javax.servlet.http.*

- Definir métodos asociados para inicializar, comenzar, detener y destruir el *servlet* → tanto la carga del *servlet* como la ejecución de sus métodos son responsabilidades del contenedor web.

Ejemplo:

```
public class MiServlet extends javax.servlet.http.HttpServlet
{
                    ……..
}
public void init()
{
                    ……..
}
public void start()
{
                    ……..
}

                    ……..
```

# Servlets. Ejecución.

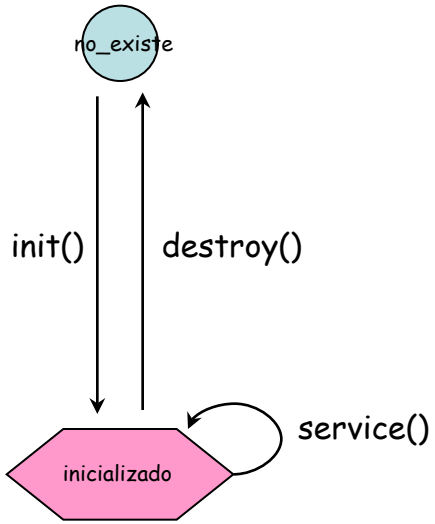
Interfaz Servlet. Tres métodos:

- init().- es invocado por el contenedor web, para iniciar la la ejecución del servlet.

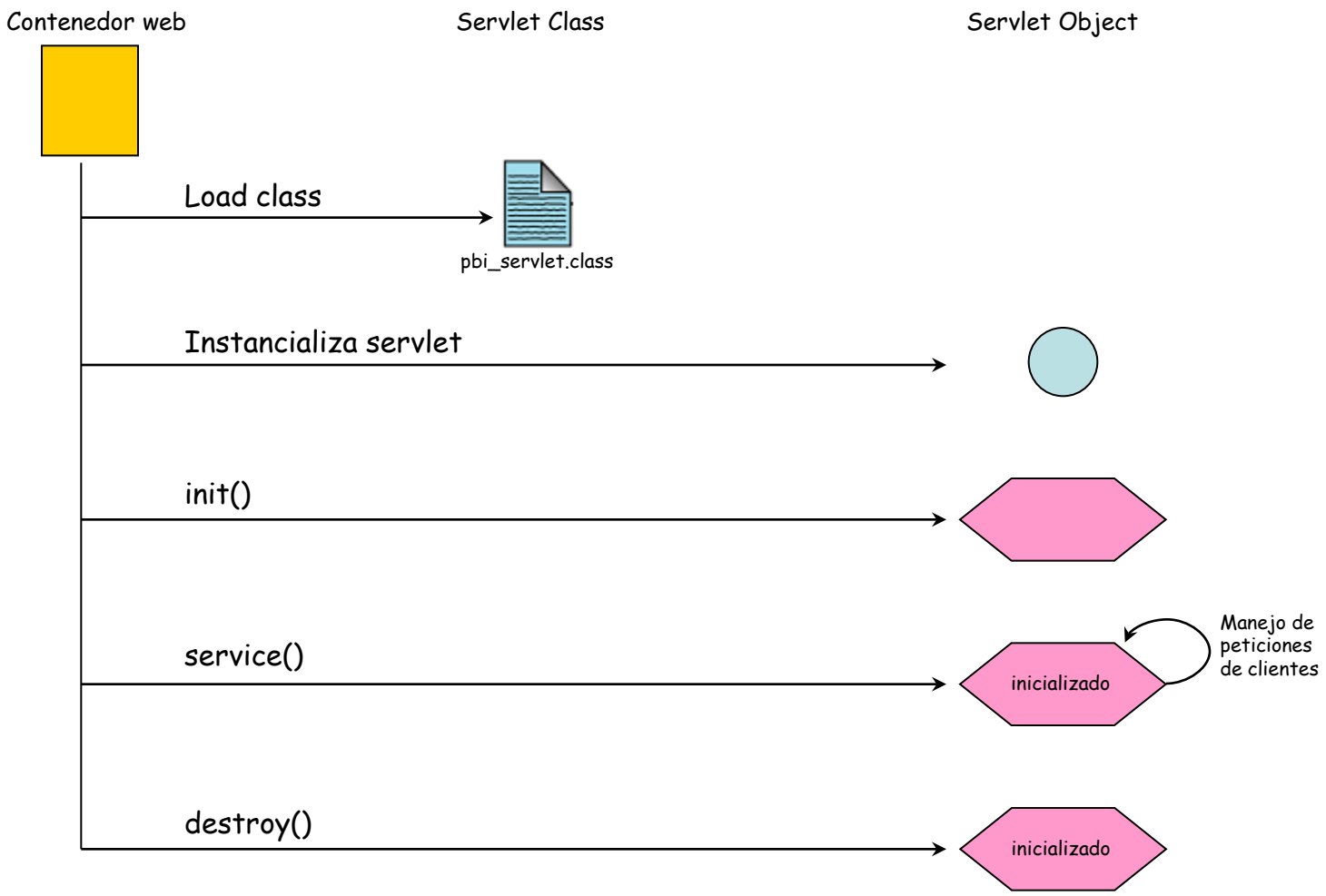
- service().- se invoca cada vez que el servidor recibe una petición para el servlet.

- destroy().- se invoca justo antes de la destrucción del servlet

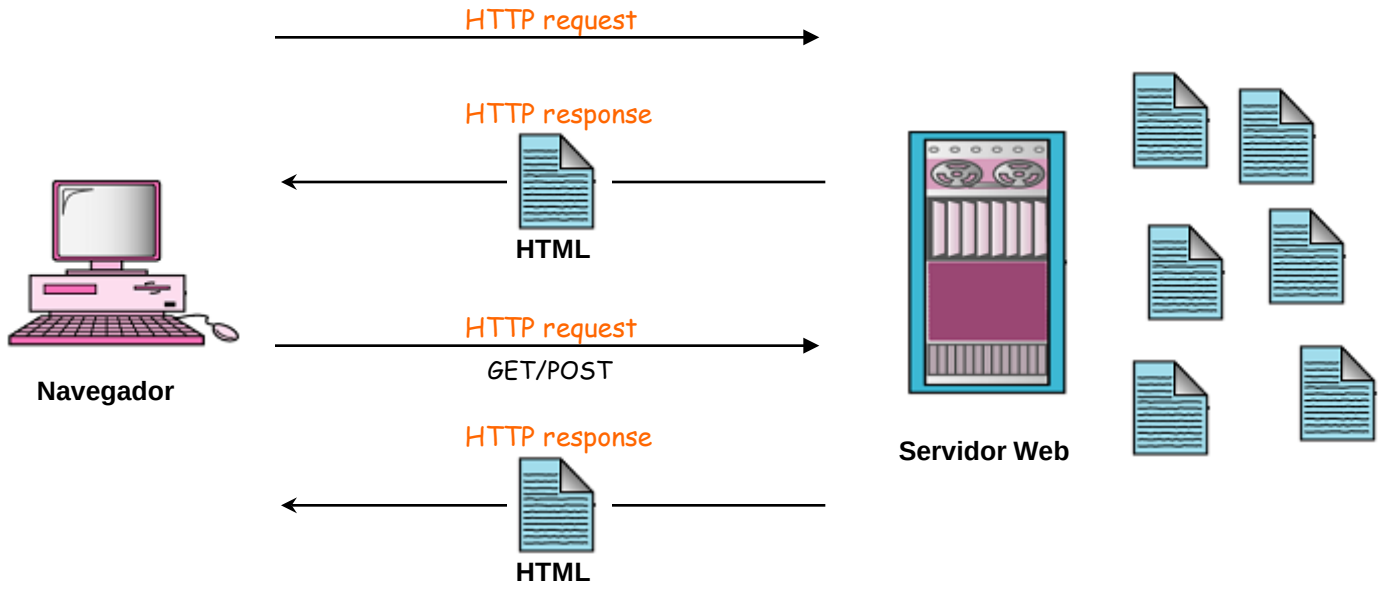
# Servlets. Ciclo de vida



# Servlets. Ciclo de vida.



# Aplicación Web.



# Métodos HTTP.

- HEAD
- TRACE
- PUT
- DELETE
- OPTIONS
- CONNECT
- GET
- POST

# Métodos HTTP.

## GET

- La cantidad de datos transferida es limitada.
- Los datos se envían en la cabecera del mensaje HTTP.
- Los datos enviados se añaden a la URL, apareciendo en la barra del navegador:
  - Es posible marcar (*bookmark*) una página obtenida mediante GET
  - No es un método seguro de envío de información.
- Pensado para “tomar” algo del servidor sin producir cambios (idempotente).
- Es el método invocado por defecto en un formulario HTML.

## GET vs URL

URL original: /mi_tienda/mi_compra.html
URL después de GET: /mi_tienda/mi_compra.html?color=rojo&tamaño=XXL

# Métodos HTTP.

Petición HTTP

*<línea de petición>*
*<cabecera>*
*<línea en blanco>*
*[<cuerpo de la petición>]*

Ejemplo petición HTTP

GET / HTTP / 1.1
Host: www.usc.es
User-Agent: Mozilla/5.0 (Windows; U; Windows XP; …..)
Gecko/20050225 Firefox/1.0.1
Connection: Keep-Alive

[<cuerpo de la petición>]

# Métodos HTTP.

Petición HTTP

*<línea de petición>*
*<cabecera>*
*<línea en blanco>*
*[<cuerpo de la petición>]*

Ejemplo petición HTTP

GET /cursos/?nombre=DAW&curso=3 HTTP/1.1
Host: www.usc.es
User-Agent: Mozilla/5.0 (Windows; U; Windows XP; …..)
Gecko/20050225 Firefox/1.0.1
Connection: Keep-Alive

[<cuerpo de la petición>]

# Métodos HTTP.

## POST

- La cantidad de datos transferida no está limitada.
- Los datos se envían en el cuerpo del mensaje HTTP.
- Los datos enviados no se añaden a la URL:
  - No es posible marcar (*bookmark*) una página obtenida mediante POST
  - Es un método seguro de envío de información.
- Pensado para enviar datos al servidor y que sean procesados, pudiendo generar cambios (BD) en el servidor (no idempotente).

# Métodos HTTP.

Petición HTTP

*<línea de petición>*
*<cabecera>*
*<línea en blanco>*
*[<cuerpo de la petición>]*

Ejemplo petición HTTP

POST / HTTP/1.1
Host: www.usc.es
User-Agent: Mozilla/5.0 (Windows; U; Windows XP; …..)
Gecko/20050225 Firefox/1.0.1
Content-Type: application/x-www-form-urlencoded
Content-Length:
Connection: Keep-Alive

[<cuerpo de la petición>]

# Métodos HTTP.

Petición HTTP

*<línea de petición>*
*<cabecera>*
*<línea en blanco>*
*[<cuerpo de la petición>]*

Ejemplo petición HTTP

```
POST / HTTP/1.1
Host: www.usc.es
User-Agent: Mozilla/5.0 (Windows; U; Windows XP; …..)
                Gecko/20050225 Firefox/1.0.1
Content-Type: application/x-www-form-urlencoded
Content-Length: 18
Connection: Keep-Alive

nombre=DAW&curso=3
```

# Métodos HTTP.

Respuesta HTTP

*<línea de respuesta>*
*<cabecera>*
*<línea en blanco>*
*[<cuerpo de la petición>]*

Ejemplo respuesta HTTP

HTTP / 1.1 *código respuesta*
Date: Tue, 17 Oct 2008 10:00:05 GMT
Content-Type: text/html; charset=ISO-8859-1
Content-Length: 122
Connection: Keep-Alive

```
<html>
        <head>
                .....
        </head>
        <body>
                .....
        </body>
</html>
```

# Métodos HTTP.

Códigos Respuesta

200 OK

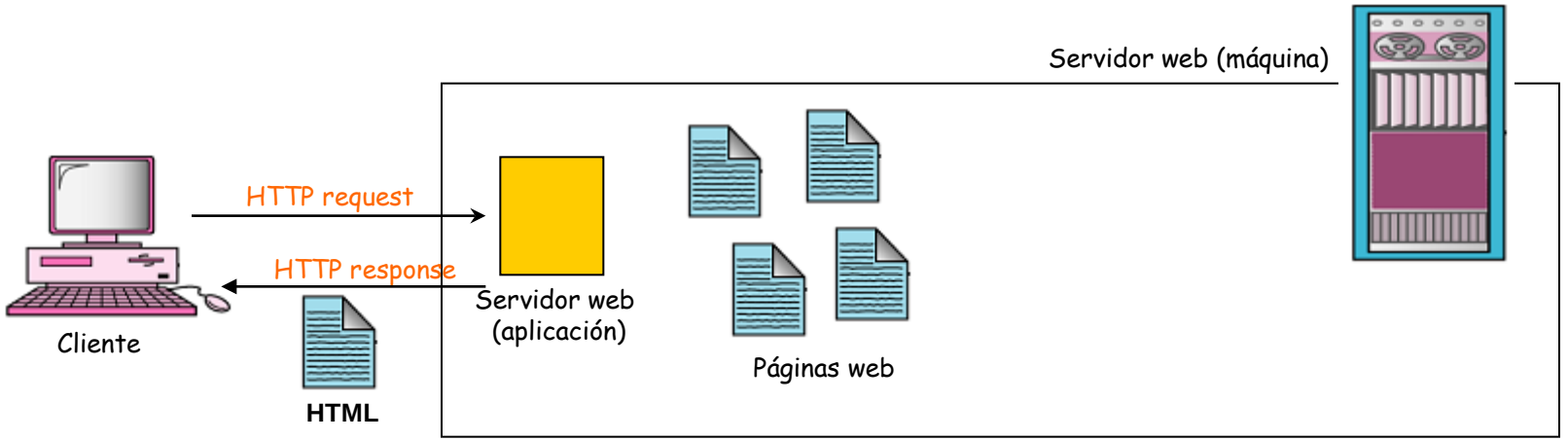
304 NOT MODIFIED

401 UNAUTHORIZED

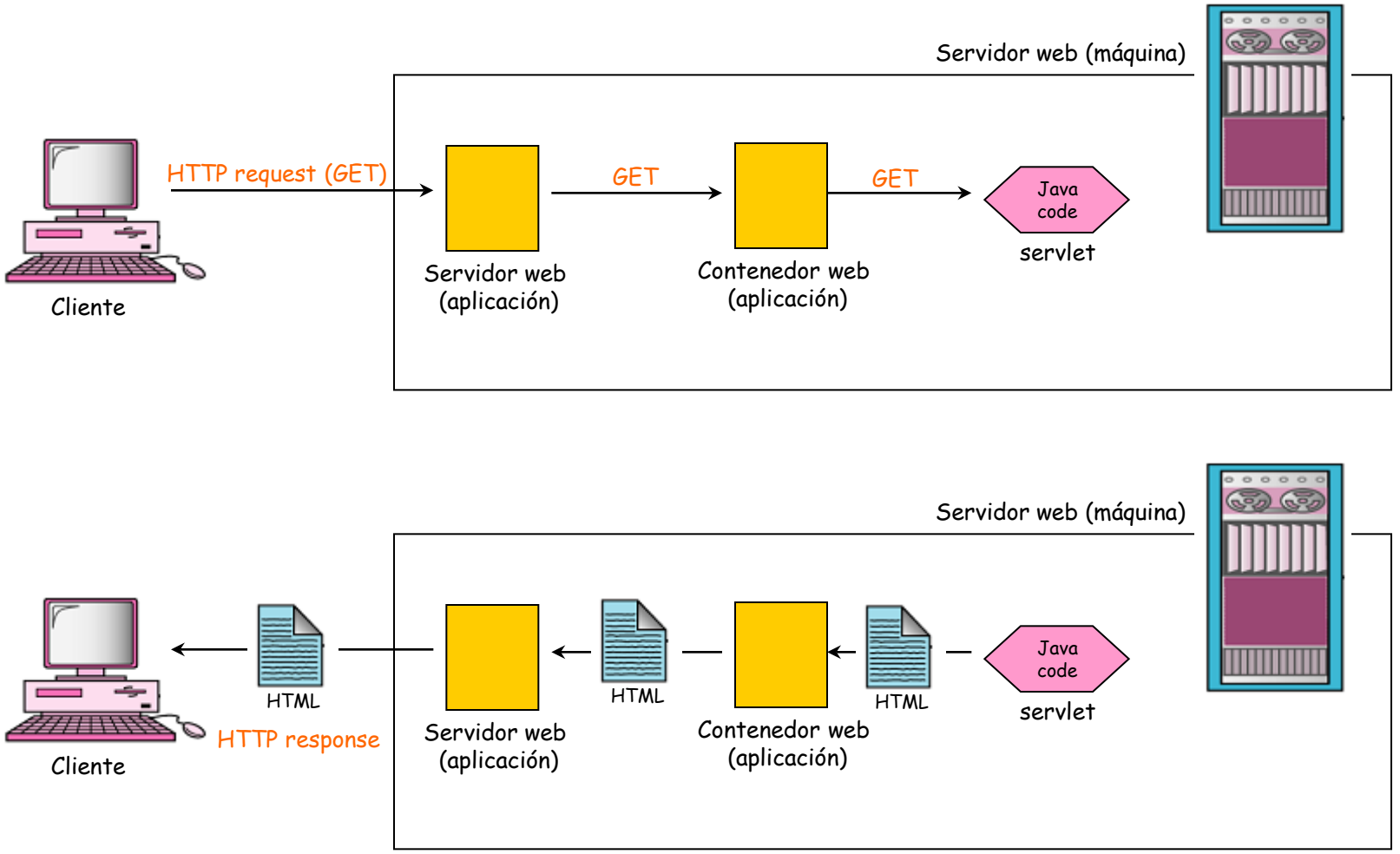
403 FORBIDDEN

404 NOT FOUND

# Contenedores

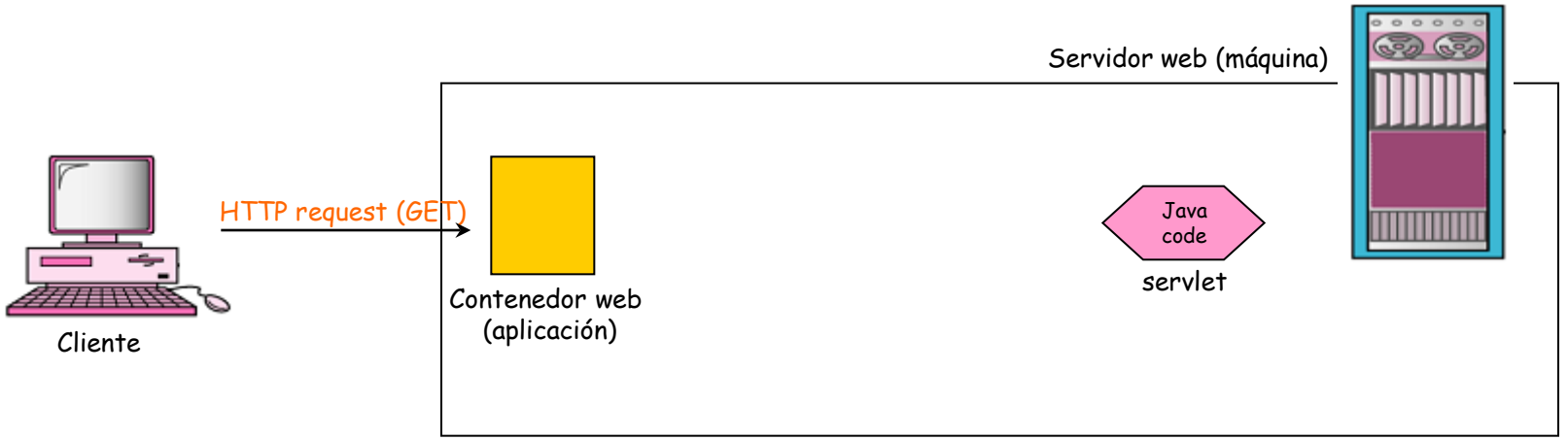


# Contenedores



# Contenedores

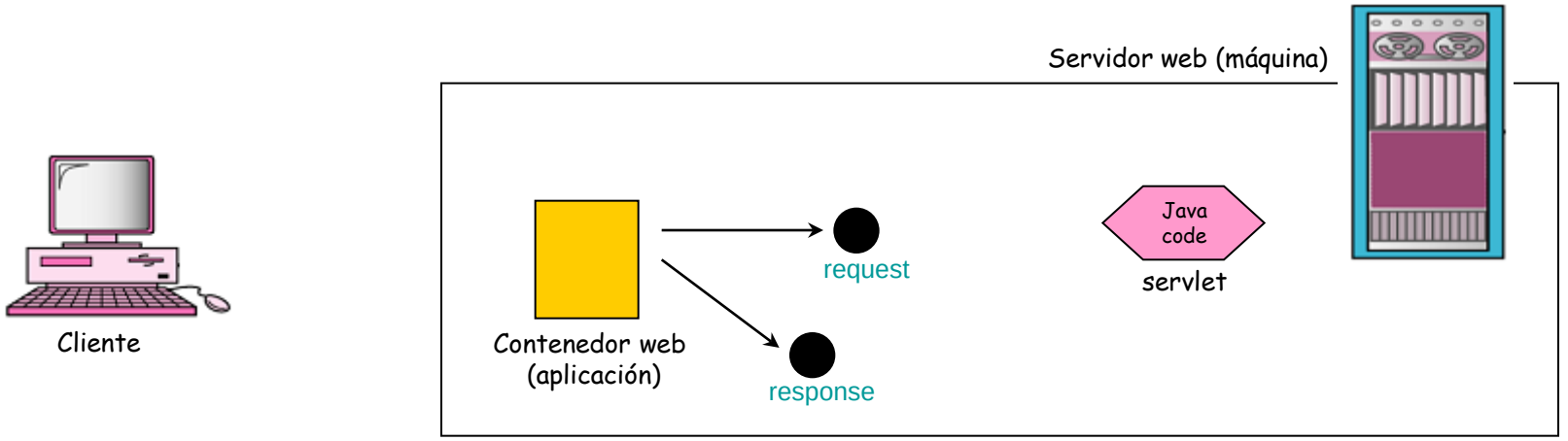
¿Cómo maneja el contenedor una petición (*request*)?




Usuario hace click en una URL asociada a un *servlet*

# Contenedores

¿Cómo maneja el contenedor una petición (*request*)?

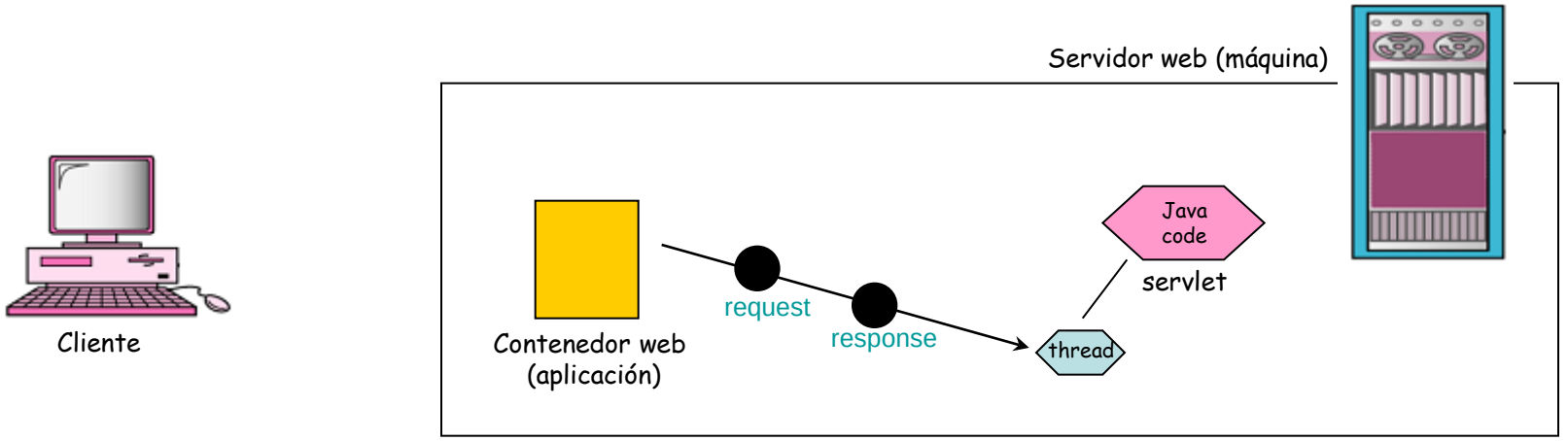



Contenedor, mira si *request* es para un *servlet.* En ese caso crea los objetos:

- HttpServletResponse
- HttpServletRequest

# Contenedores

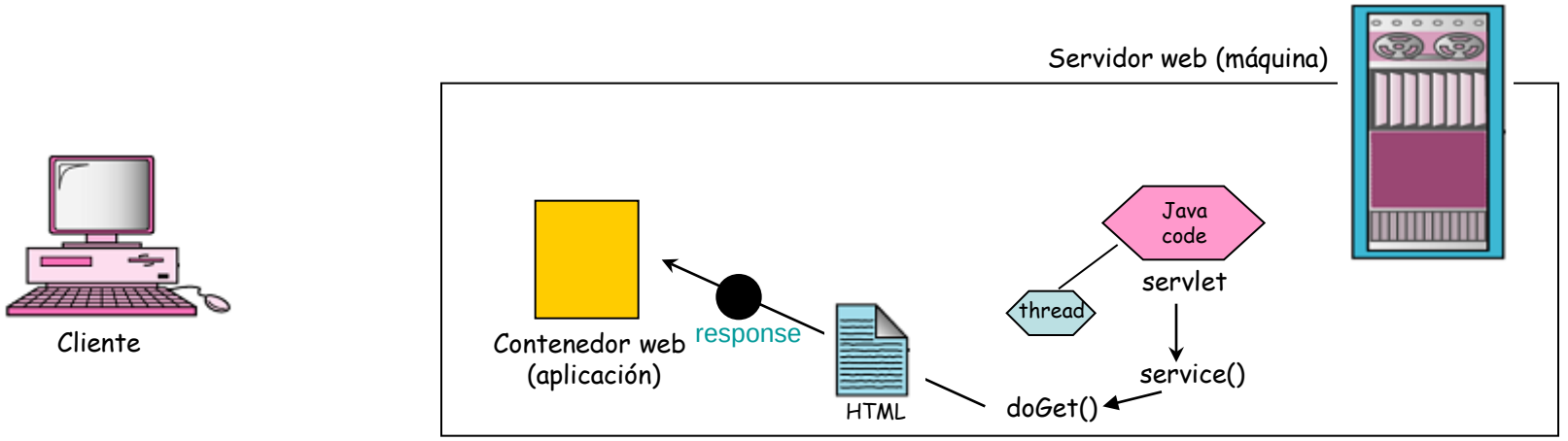
¿Cómo maneja el contenedor una petición (*request*)?




Contenedor, encuentra el *servlet* correcto (basado en URL), crea un hilo para la petición y pasa los objetos *request* y *response* a dicho hilo.

# Contenedores

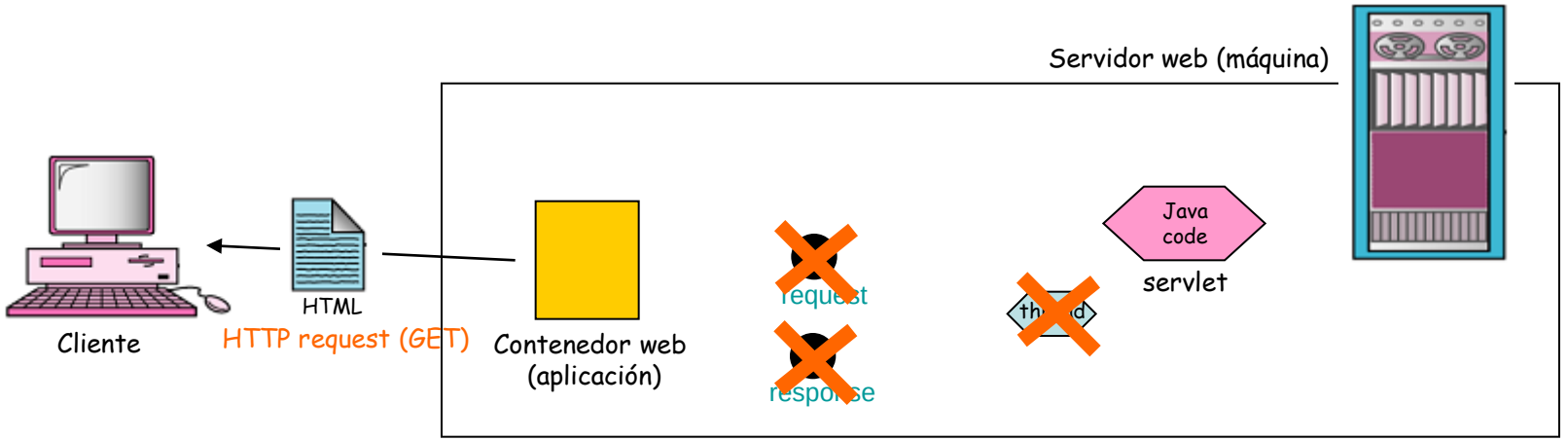
¿Cómo maneja el contenedor una petición (*request*)?




El método *doGet() (o doPost()),* genera una página HTML e incluye dicha página en el objeto *response*.

# Contenedores

¿Cómo maneja el contenedor una petición (*request*)?




El hilo completa su ejecución. El contenedor convierte el objeto respuesta en una respuesta HTTP y se la devuelve al cliente. Finalmente el contenedor borra los objetos *request* y *response.*

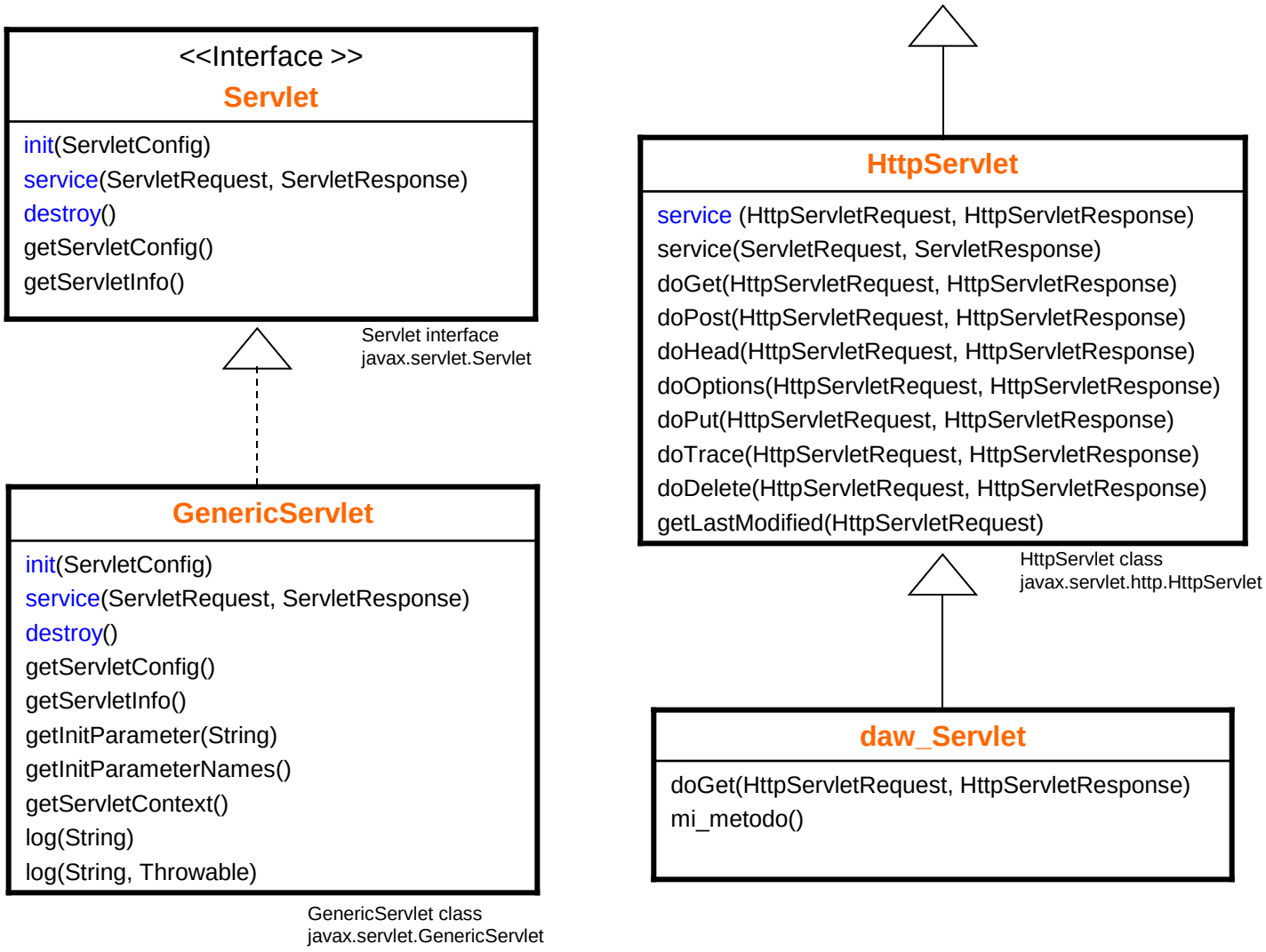
# Código servlet

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

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

            PrintWriter out= response.getWriter();
            java.util.Date today= new java.util.Date();
            out.println("<htlm>" +
                      "<body>" +
                      "<h1>" +
                      "Desarrollo de Aplicaciones Web" +
                      "</h1>" +
                      "<p>"´+ today + "</p>" +
                      "</body>" +
                      "</html>");
    }
}
```

# Servlets. Métodos.



# Servlets. *Service().*

service(HttpServletRequest, HttpServletResponse)

HttpServletRequest.- objeto de la clase HttpServletRequest que encapsula los datos enviados por el cliente al servidor.

HttpServletResponse.- objeto de la clase HttpServletResponse que encapsula los datos enviados por el servidor al cliente. Existen dos formas de envío:

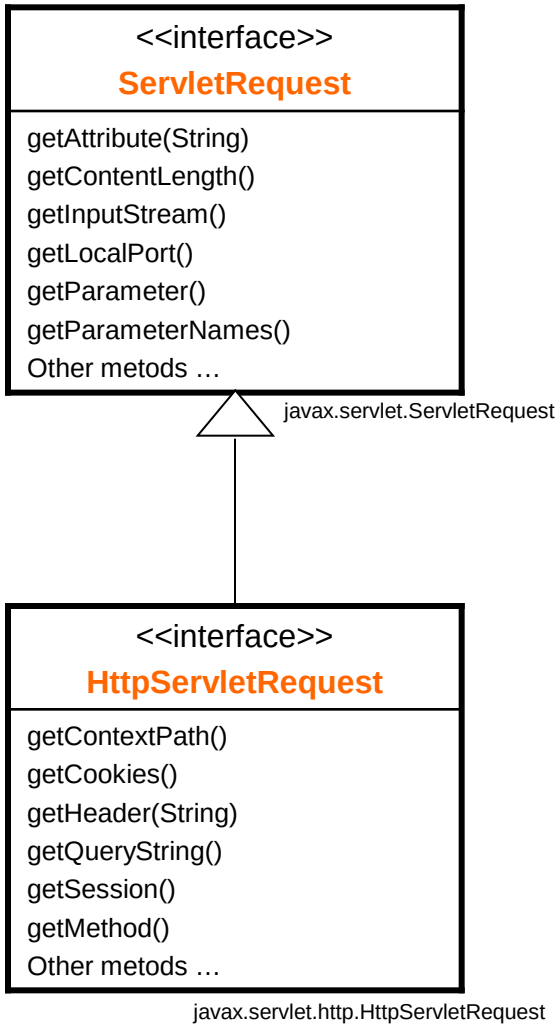
- getWriter().- devuelve cadenas de caracteres.

  Ej.- PrintWriter writer= response.getWriter();
  writer.println("Este texto va de vuelta");

- getOutputStream().- devuelve cualquier cosa. Pensado para devolver datos binarios.

  Ej.- ServletOuputStream out= response.getOutputStream();
  out.write(ByteArray);

# Servlets. *ServletRequest.*



# Servlets. *ServletResponse.*