



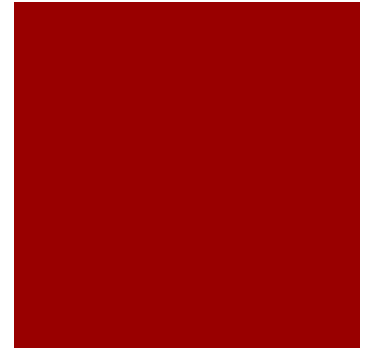
Aplicaciones Open Source

Semana01
24/03/2018

Carlos A. Quinto Cáceres
pcsicqui@upc.edu.pe

Agenda

- Fundamentos HTML
 - Elementos de página
 - Formularios Web
- Bootstrap
- Introducción a las aplicaciones Web
- Servlets

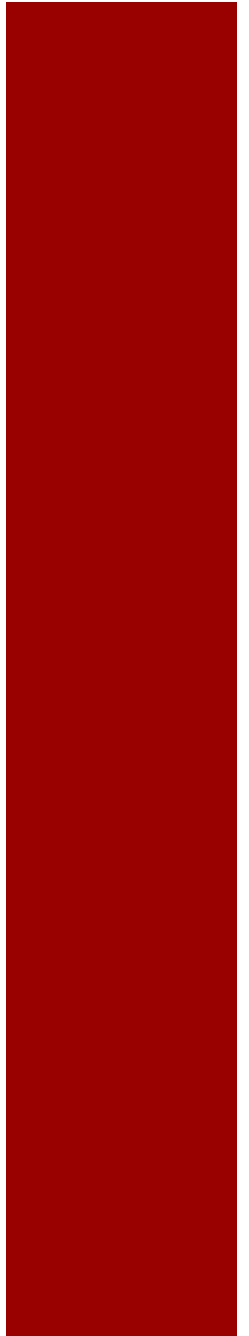


Logros del día



- Al finalizar la clase, el alumno podrá:
 - Hacer uso de un framework CSS para crear una página Web básica.
 - Conocer los principales conceptos relacionados a una aplicación Web.
 - Crear un proyecto Maven y desarrollar los primeros Servlets para recibir peticiones GET y POST

Fundamentos HTML



¿Qué es HTML?



- HTML son las iniciales HyperText Markup Language (lenguaje de marcado de hipertexto).
- Hipertexto quiere decir "texto que contiene enlaces."
- Es un conjunto de etiquetas, también conocidas como elementos.

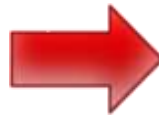
```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>

</body>
</html>
```

¿Qué es HTML?

- Los navegadores Web leen archivos de texto que contienen el conjunto de etiquetas, luego las interpretan para determinar como desplegar una pagina Web.

```
<!DOCTYPE html>
<html>
<head>
  <title>Titulo de la pagina</title>
</head>
<body>
<h2>Listado de productos</h2>
<ul>
  <li>Producto 1</li>
  <li>Producto 2</li>
  <li>Producto 3</li>
</ul>
</body>
</html>
```

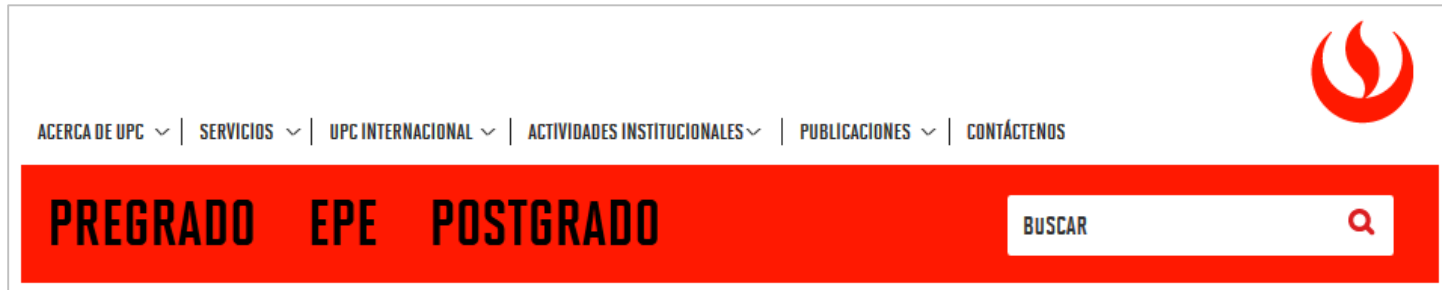


Listado de productos

- Producto 1
- Producto 2
- Producto 3

¿Qué es CSS?

- Son las iniciales para Cascading Style Sheets (hojas de estilo en cascada).
- Las hojas de estilo son un mecanismo para agregar estilo a las páginas Web, definen el look and feel de una página Web.



¿Cómo se usa CSS?

Los estilos se aplican en base a selectores.

- Selectores por id

```
#footer {background: #CCC; margin: 15px}
```

```
<p id="footer">
```

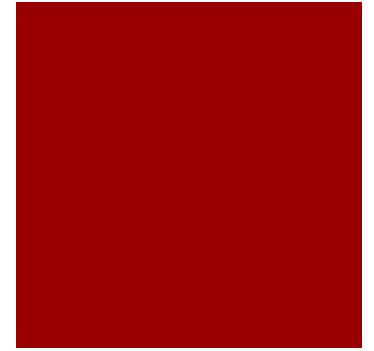
- Selectores por clases (class)

```
.introduction {font-size: small; color: white}
```

```
<p class="introduction">
```

- Selectores por elementos

```
p { margin-left: 25px; margin-right: 25px }
```



Ubicación de estilos



■ Estilos en línea

Pierde las ventajas de utilizar las hojas de estilo, puesto que los estilos se aplican directamente en el elemento HTML.

Para aplicar estilos directamente, hacer uso del atributo **style** del elemento y definir las propiedades a modificar.

```
<p style="font-weight: bold; font-style: italic; color: #FF0000">
```

Ubicación de estilos

■ Estilos internos

Debe ser usado cuando los estilos le pertenecen o van a ser usados solo en una página.

Se definen en la sección <head> y usando el tag <style>.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
  <head>
    <title>HTML 4.0 CSS Element Style Example</title>
    <style type="text/css">
      h1{text-align:center; color:blue;}
    </style>
  </head>
  <body>
    <h1>This text is centered and blue</h1>
  </body>
</html>
```

Ubicación de estilos



■ Hojas de estilo externas

Se debe usar cuando los estilos van a ser aplicados a varias páginas, lo que permite que al realizar un cambio en este único archivo se refleje en toda la Web.

Inclusión de hojas de estilo en las paginas

```
<head>  
<link rel="stylesheet" type="text/css" href="mystyle.css">  
</head>
```

Hojas de estilo

```
hr {color:sienna;}  
p {margin-left:20px;}  
body {background-image:url("images/background.gif");}
```

Formularios Web

Definición

- Los formularios permiten recolectar información.
- Actúan como contenedor de todos los elementos de entrada.
- La información recolectada puede ser entregada a un agente procesador – action.
- Las etiquetas son:

```
<form action="" method="" name="">
```

```
</form>
```

Definición

- Atributos de un formulario:
 - **action**: URL del programa que va a recibir la información del formulario.
 - **method**: como se enviarán los datos del formulario (GET o POST).
 - **name**: nombre del formulario.

```
<form action="" method="">
<p>Usuario: <input type="text" name="usuario" id="user" /></p>
<p>Clave: <input type="password" name="clave" id="pass" /></p>
<p><input type="submit" name="iniciar" id="login" value="Iniciar" /></p>
</form>
```

Envío de datos

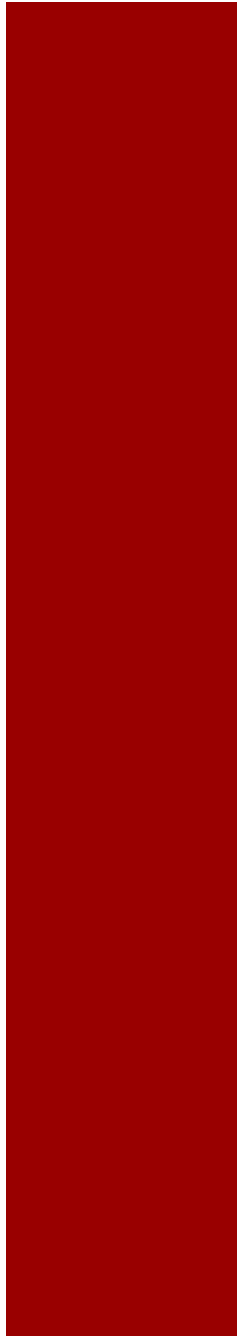
- Los datos se envían como una secuencia de pares **dato**(name) y **valor** (value).

```
<form action="" method="">
<p>Nombre: <input type="text" name="nombre" id="nombre" /></p>
<p>Sexo:
<input type="radio" name="sexo" id="sexo" value="M" />Masculino
<input type="radio" name="sexo" id="sexo" value="F" />Femenino
</p>
<p><input type="submit" name="guardar" id="guardar" value="Guardar" /></p>
</form>
```

Diferencias

	GET	POST
Seguridad	La información enviada al servidor va como parte de la URL	Los parámetros no se guarda en la historia del navegador o en los registros del servidor
Visibilidad	Los datos son visibles en la URL	Los datos no se visualizan en la URL
Tipos de datos	Solo se permiten caracteres ASCII	No hay restricciones.
Botón de recarga y atrás del navegador	Se usa sin problemas	Los datos son reenviados, pero se le avisa al usuario a través de una alerta
Bookmarked	Se puede realizar	No se puede realizar
Guardar en caché	Se puede almacenar	No se puede almacenar
Longitud de datos	Existe una longitud máxima dependiendo del navegador y servidor	Sin restricciones

Bootstrap



Bootstrap

- Es un framework HTML, CSS y JS utilizado para poder crear páginas Web de manera más rápida y que sean responsivas.
- Nos proporciona una variedad de CSS y JS listo para reutilizar en nuestros proyectos.



Bootstrap



- Descargar de la página oficial:
 - <http://getbootstrap.com/getting-started/>
- Incluir el archivo css y js.

```
<!DOCTYPE html>
<html lang="en">
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="bootstrap-3.3.7-dist/css/bootstrap.min.css">
<script src="bootstrap-3.3.7-dist/js/bootstrap.min.js"></script>
```

* Para hacer uso de los plugins, es necesario incluir jQuery.

Grid System



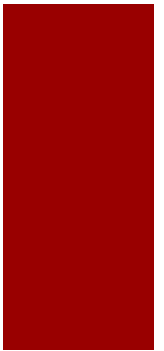
- Bootstrap incluye un grid system que se escala a 12 columnas.
- Los grid systems son usados para crear diseños a través de filas y columnas en donde se encuentra colocado nuestro contenido:
 - Las filas deben estar dentro de un .container
 - Las filas se usan para crear grupos horizontales de columnas
 - El contenido debe ir dentro de las columnas

Grid system



	Extra small devices Phones (<768px)	Small devices Tablets (≥768px)	Medium devices Desktops (≥992px)	Large devices Desktops (≥1200px)
Grid behavior	Horizontal at all times	Collapsed to start, horizontal above breakpoints		
Container width	None (auto)	750px	970px	1170px
Class prefix	.col-xs-	.col-sm-	.col-md-	.col-lg-
# of columns	12			
Column width	Auto	~62px	~81px	~97px

CSS - Tablas



```
<table class="table">  
  ...  
</table>
```

```
<table class="table table-striped">  
  ...  
</table>
```

```
<table class="table table-hover">  
  ...  
</table>
```

```
<table class="table table-bordered">  
  ...  
</table>
```

```
<div class="table-responsive">  
  <table class="table">  
    ...  
  </table>  
</div>
```

CSS - Formularios

```
<form>
  <div class="form-group">
    <label for="exampleInputEmail1">Email address</label>
    <input type="email" class="form-control" id="exampleInputEmail1" placeholder="Email">
  </div>
  <div class="form-group">
    <label for="exampleInputPassword1">Password</label>
    <input type="password" class="form-control" id="exampleInputPassword1" placeholder="Password">
  </div>
  <div class="form-group">
    <label for="exampleInputFile">File input</label>
    <input type="file" id="exampleInputFile">
    <p class="help-block">Example block-level help text here.</p>
  </div>
  <div class="checkbox">
    <label>
      <input type="checkbox"> Check me out
    </label>
  </div>
  <button type="submit" class="btn btn-default">Submit</button>
</form>
```

CSS - Buttons



Components

Home / Library / Data



Well done! You successfully read this important alert message.

Heads up! This alert needs your attention, but it's not super important.

Warning! Better check yourself, you're not looking too good.

Oh snap! Change a few things up and try submitting again.

Inbox 42

Messages 4

Dropdown ▾

Action

Another action

Something else here

JavaScript

- Dropdown
- Tab
- Tooltip
- Button
- Carousel

Collapsible Group Item #1

Collapsible Group Item #2

Anim pariatur cliche reprehenderit, enim non cupidatat skateboard dolor brunch. aliqua put a bird on it squid single-origin labore wes anderson cred nesciunt sapi beer farm-to-table. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est

VHS.

Collapsible Group Item #3

Home

Profile

Dropdown ▾

Raw denim you probably haven't heard of. Mustache cliche tempor, williamsburg car sweater eu banh mi, qui irure terry richardson
 apparel, butcher voluptate nisi qui.

Tooltip on the left

Tooltip on the top



First slide



Ejercicio

Some Favorites

[Celery Root](#)

[Spaghetti Squash](#)

[Killer Mushrooms](#)

Search Recipes

Uh oh! Have you had your daily dose of veggies today??

Wild & Wacky Vegetables

The beet is the most intense of vegetables. The radish, admittedly, is more feverish, but the fire of the radish is a cold fire, the fire of discontent not of passion. Tomatoes are lusty enough, yet there runs through tomatoes an undercurrent of frivolity. Beets are deadly serious.

— Tom Robbins

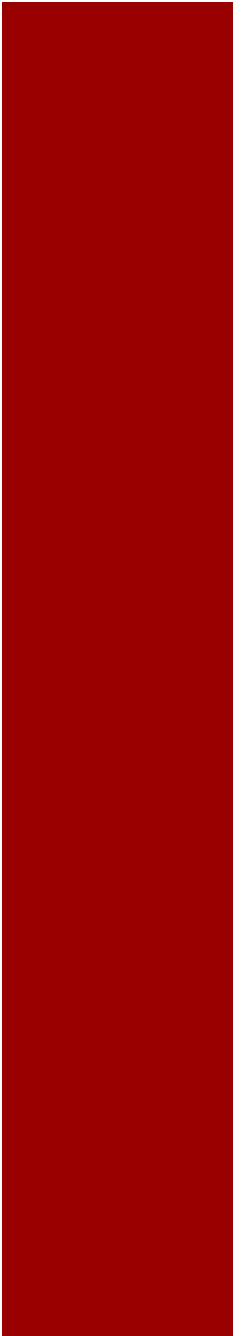
Some diseases

Veggie	Disease
Beets	Beeturia
Carrots	Carotenosis



World Wide Web

¿cómo se accede a un recurso en la Web?

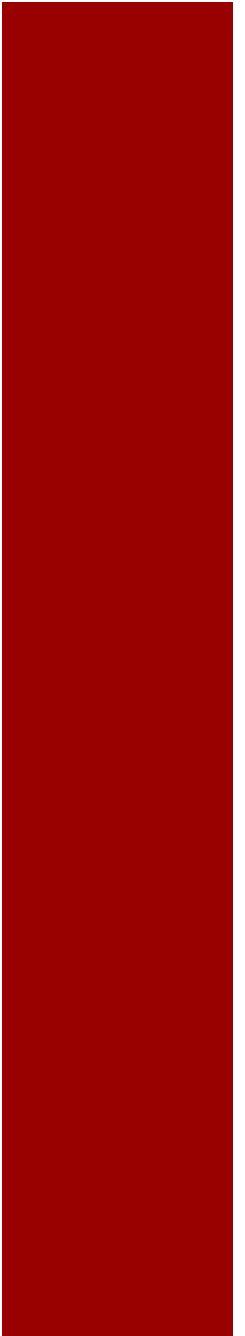


Video



Aplicación Web

Forma de acceso y conceptos relacionados a
una aplicación Web



¿Cómo se accede a una aplicación Web?

Usuarios de aplicaciones Web

Internet

Servidor Web



Servidor Web



- Un servidor web se encarga de suministrar páginas en respuesta a las peticiones de los navegadores Web.
- La petición se realiza a través de un link (enlace) o al presionar un botón de un formulario.

URL

- Localizador de recursos uniforme (Uniform Resource Locator)
- Permite localizar e identificar recursos en una red, el formato de una URL es:
esquema://maquina/directorio/recurso

Generalmente se convierte en:
protocolo://servidor:puerto/directorio/recurso



Dirección IP

- Identifica a un dispositivo dentro de una red que utilice el protocolo IP (Internet Protocol).
- El protocolo IP se usa para la comunicación entre un origen y un destino.



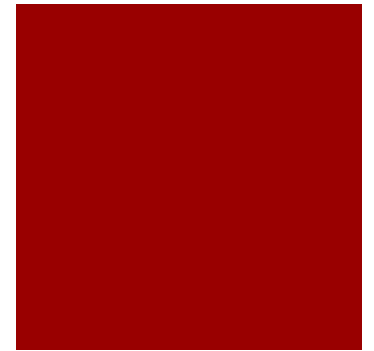
Dominio de Internet



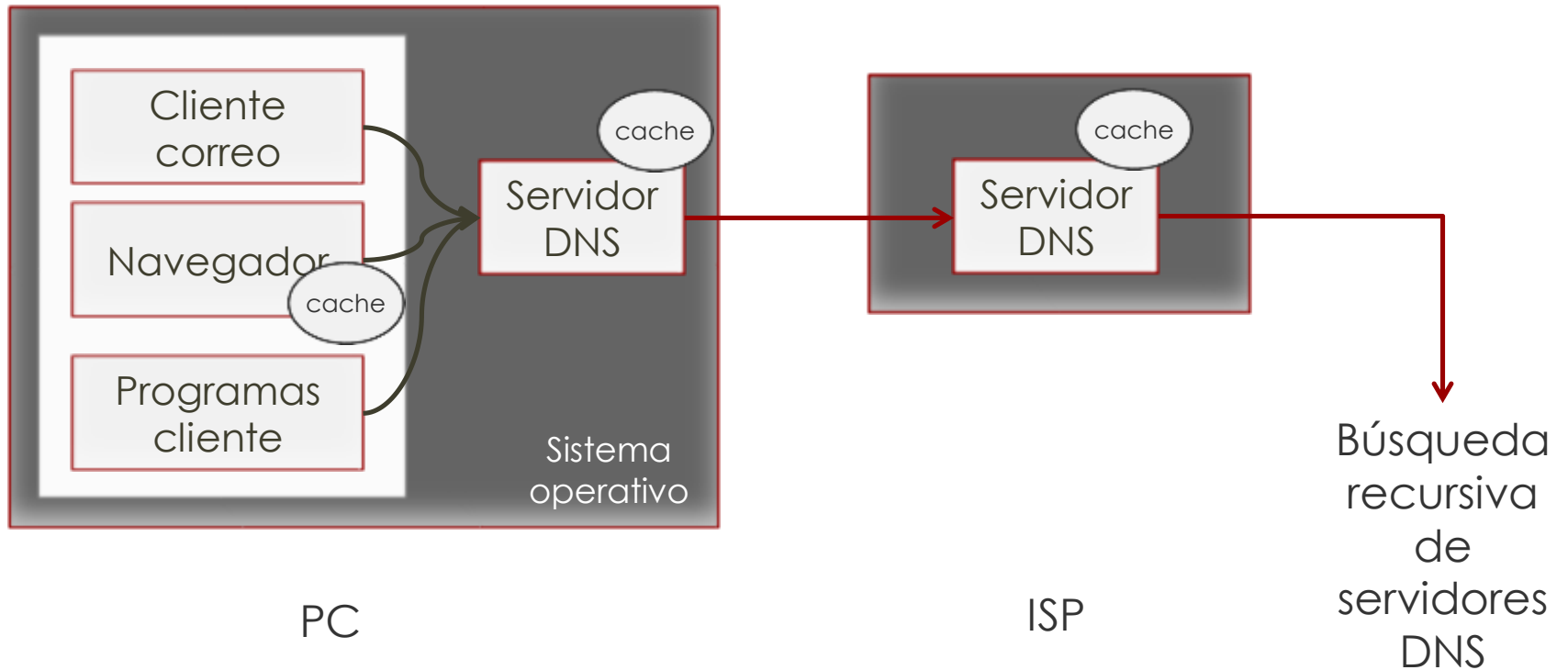
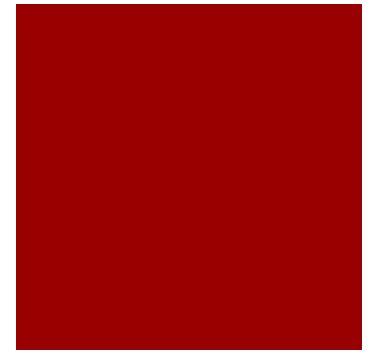
- Su principal objetivo es traducir una dirección IP en un término más fácil de recordar y encontrar.
- Niveles:
 - upc.edu.pe : nombre de dominio
 - pe : dominio de nivel superior geográfico (ccTLD)
 - edu : dominio de nivel superior genérico (gTLD)
 - upc : dominio de segundo nivel
 - www : subdominio

¿Qué es DNS?

- Sistema de nombres de dominio.
- Se utiliza principalmente para la resolución de nombres de dominio.
- Resuelve la pregunta:
¿Qué dirección IP le pertenece a un nombre de dominio?



DNS



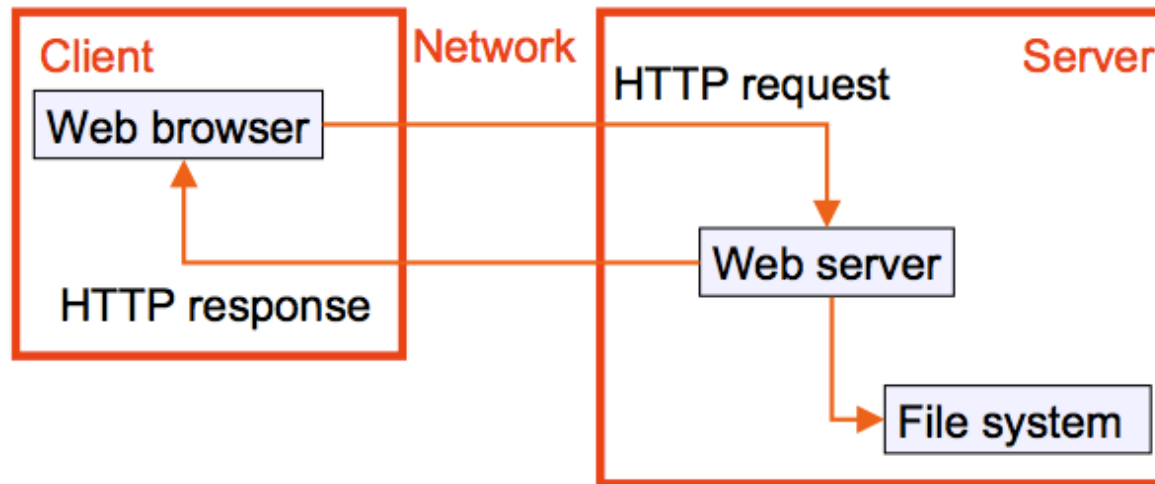
¿Qué es HTTP?

- Protocolo de transferencia de hipertexto.
- Protocolo orientado a transacciones.
- Se rige por un esquema request – response entre un cliente (navegador o user agent) y un servidor (servidor Web).



HTTP – Transacción

- Request: al solicitar un recurso desde el cliente.
- Response: al obtener la respuesta del servidor.

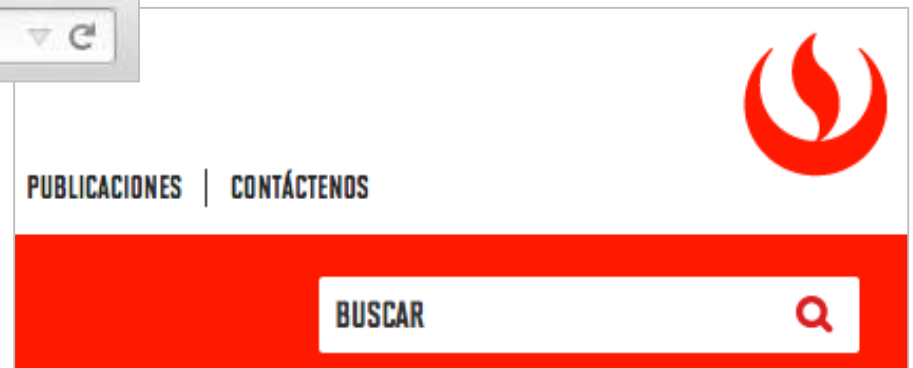
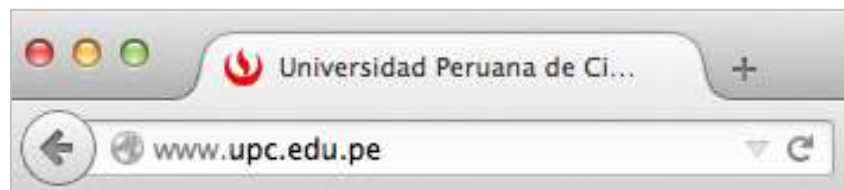


HTTP - Métodos

■ Método GET

Usado para solicitar un recurso u obtener información.

- <http://www.facebook.com/imagenes/foto1.gif>
- <http://www.upc.edu.pe/index.php>



HTTP - Métodos

■ Método POST

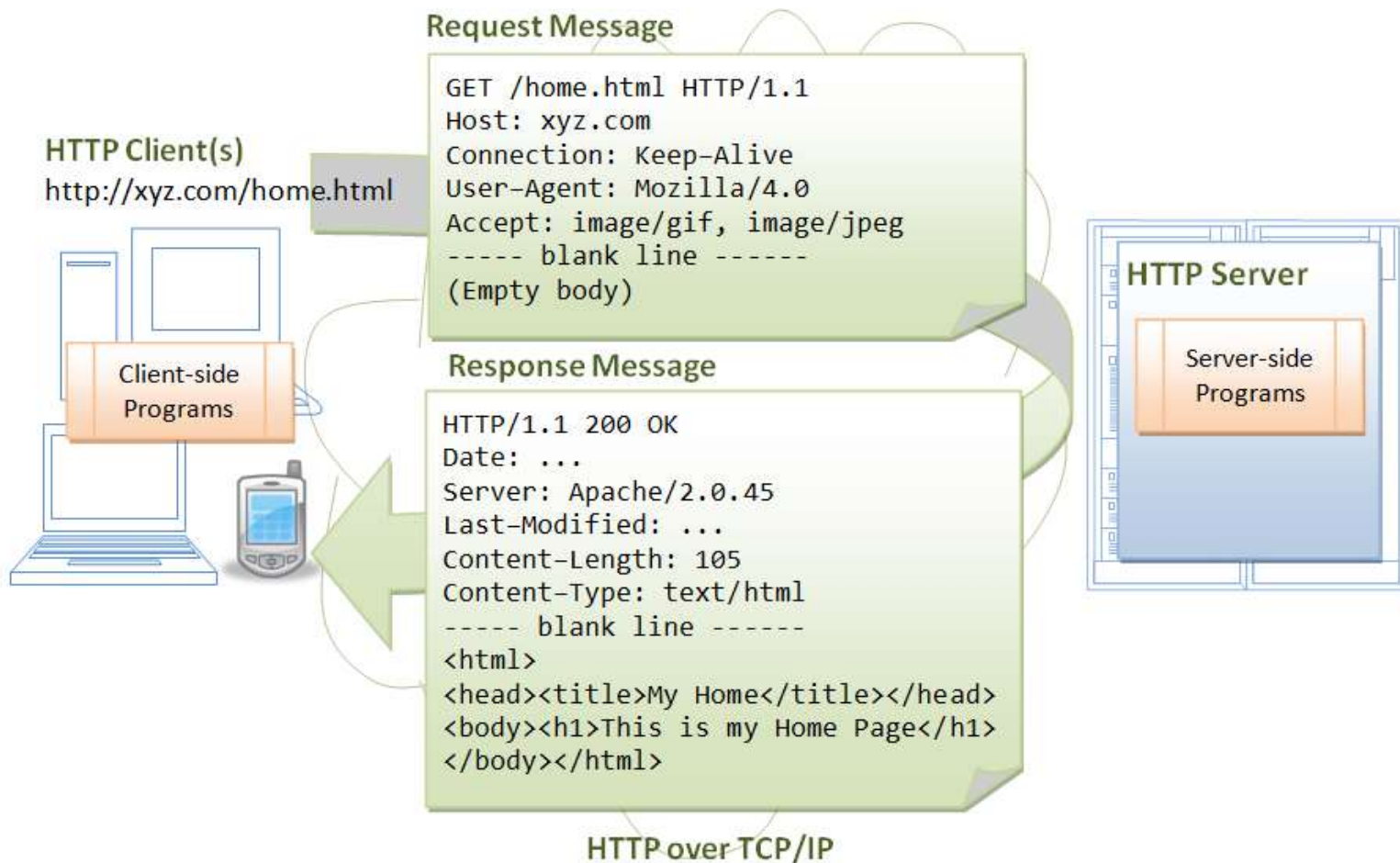
Usado para enviar datos que serán procesados por un recurso en el servidor.

```
<form action="[jsp/servlet]" method="post" name="formLogin" id="formLogin">  
  <input type="text" name="usuario" id="usuario" />  
  <input type="text" name="clave" id="clave" />  
  <input name="btnEnviar" type="submit" value="Iniciar Sesión" />  
</form>
```

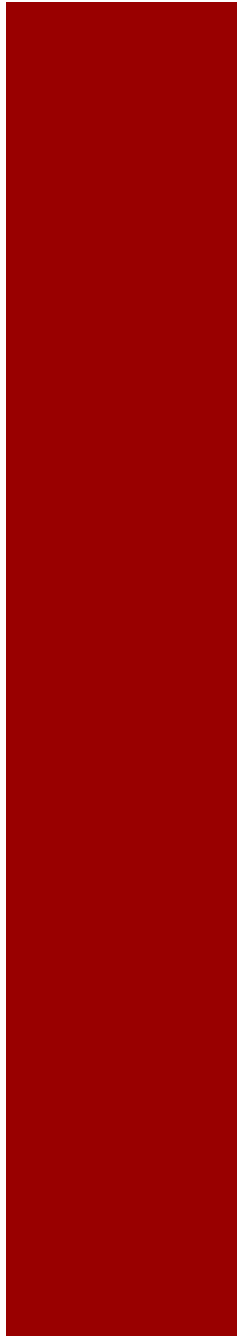


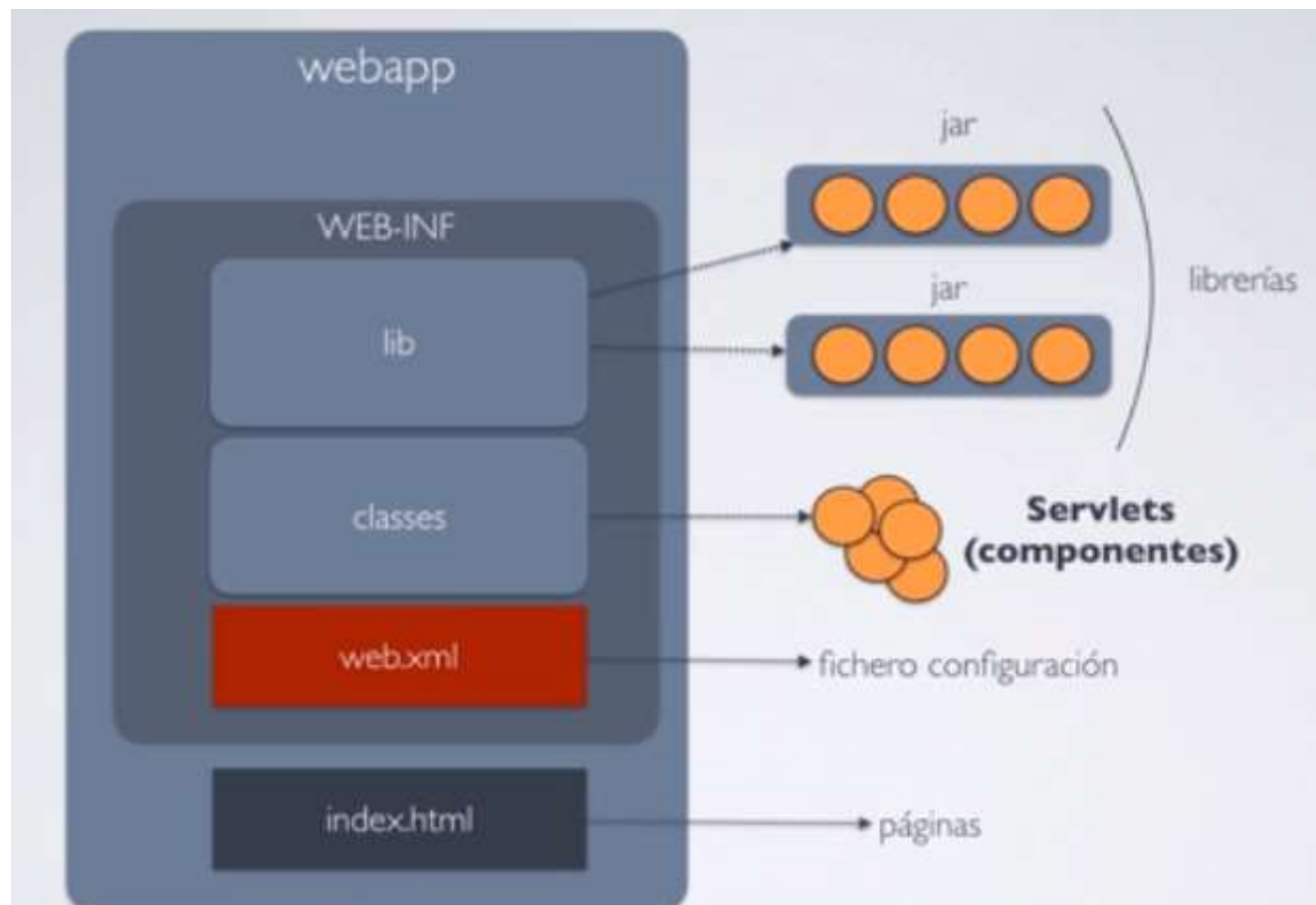
A screenshot of a Google Sign in form. The form is titled "Sign in" in the top left and has the Google logo in the top right. It contains two text input fields: "Username" and "Password". Below the "Password" field, there is a blue "Sign in" button and a checkbox labeled "Stay signed in".

HTTP - Mensajes



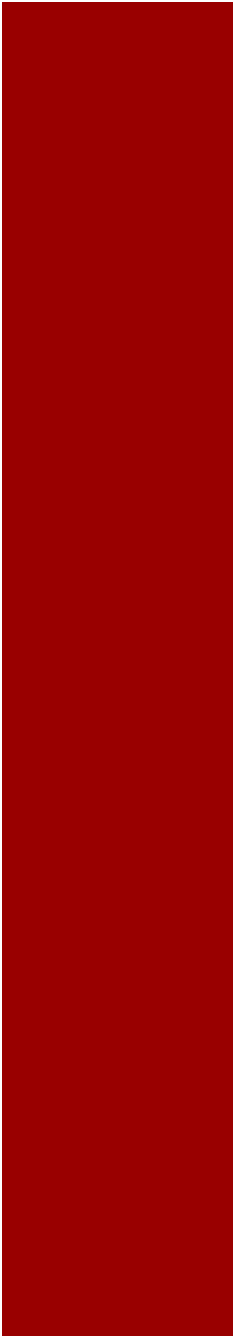
Aplicación Web Java





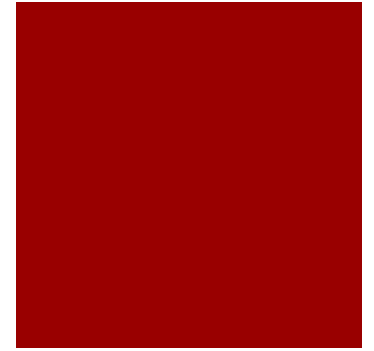
Servlets

Primeros pasos

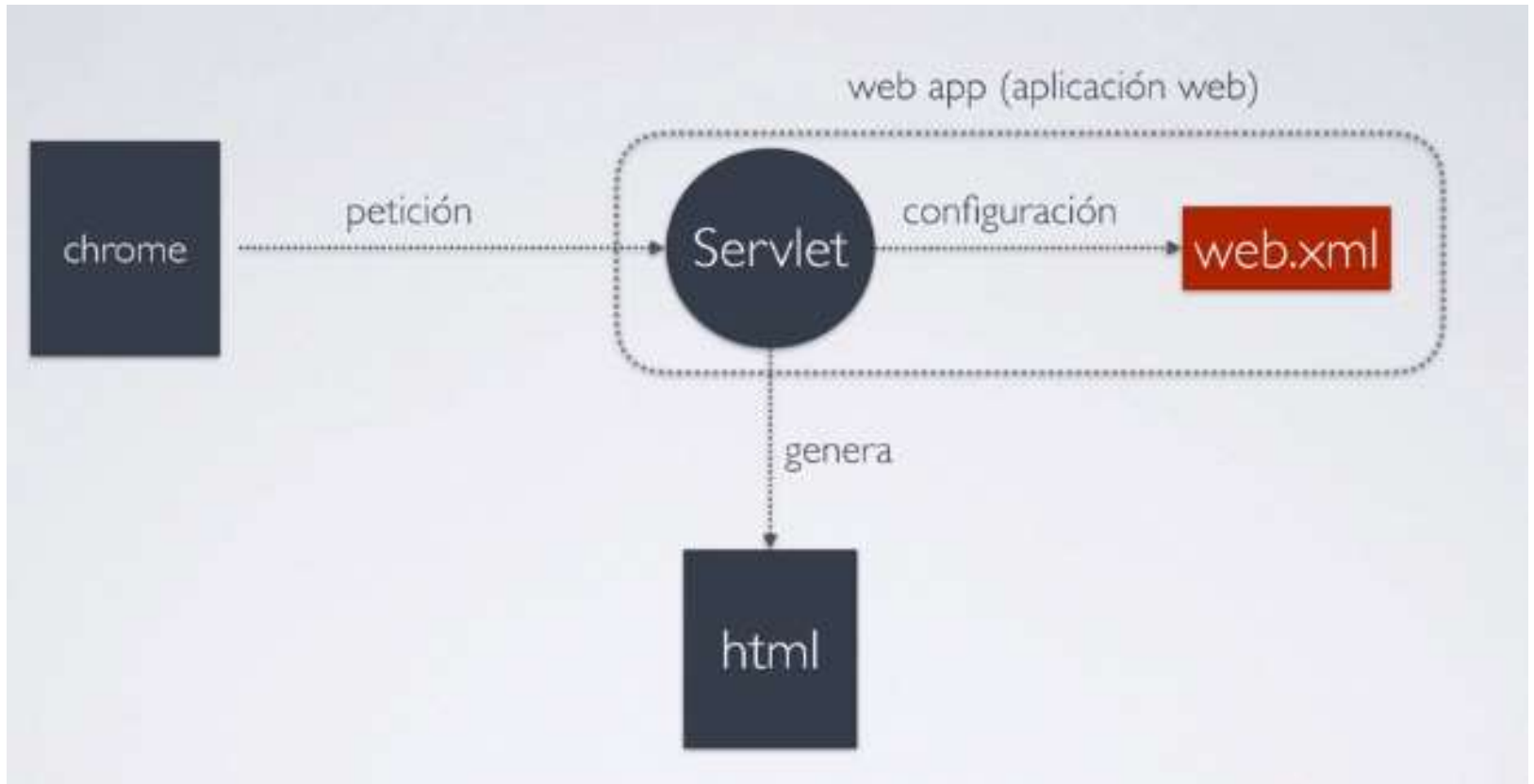


Concepto

- Se ejecutan dentro de un contenedor de servlets como el Apache Tomcat.
- Extiende la capacidad de proceso de un servidor y emplea el paradigma request – response.
- Es una clase Java que recibe requerimientos de un cliente para cumplir con un servicio; luego de cumplir con el servicio, envía la respuesta hacia el cliente.



Servlets



Concepto

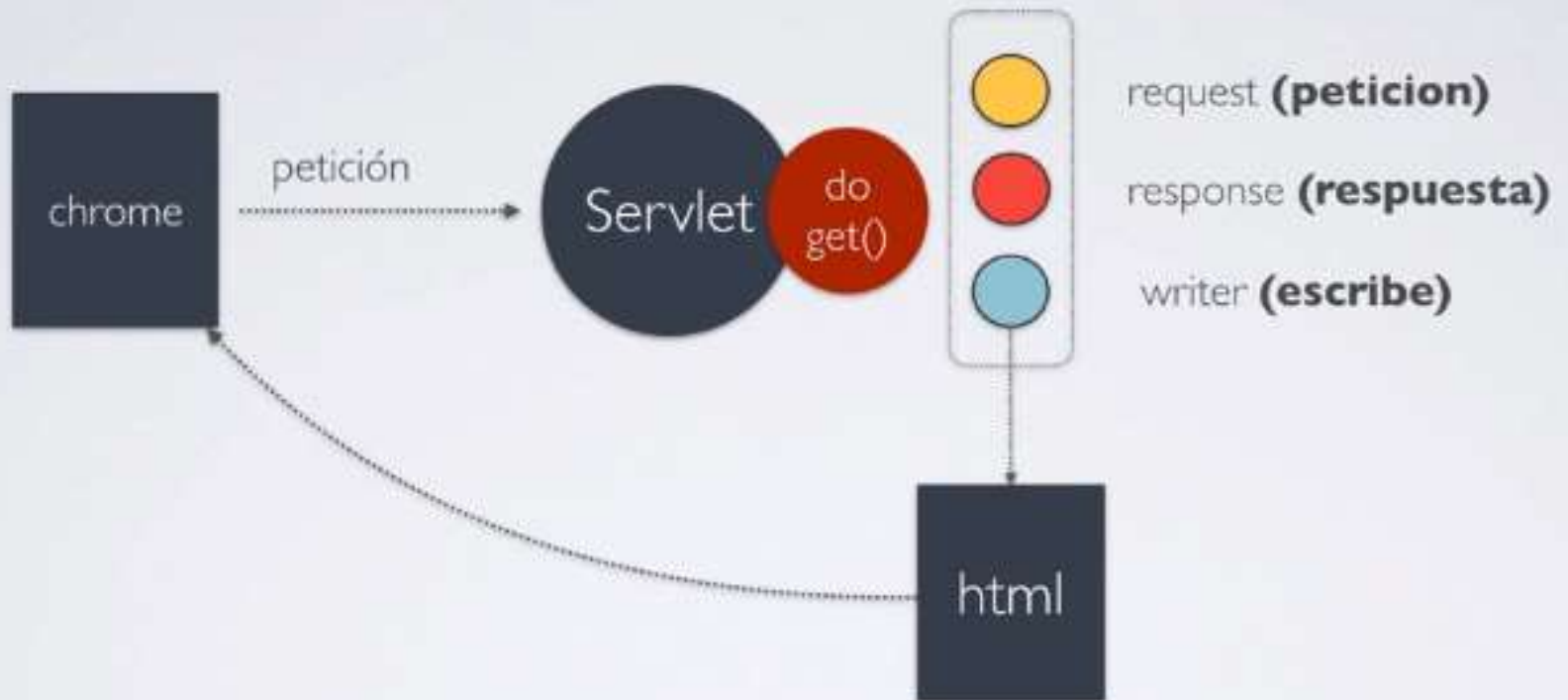


- El contenedor Web es el encargado de:
 - Capturar los requerimientos HTTP y traducirlos en objetos java que el Servlet entiende.
 - Pasar el requerimiento al Servlet (Interfaz `HttpServletRequest`).
 - Devolver las respuestas (Interfaz `HttpServletResponse`).
 - Manejar el ciclo de vida del Servlet.

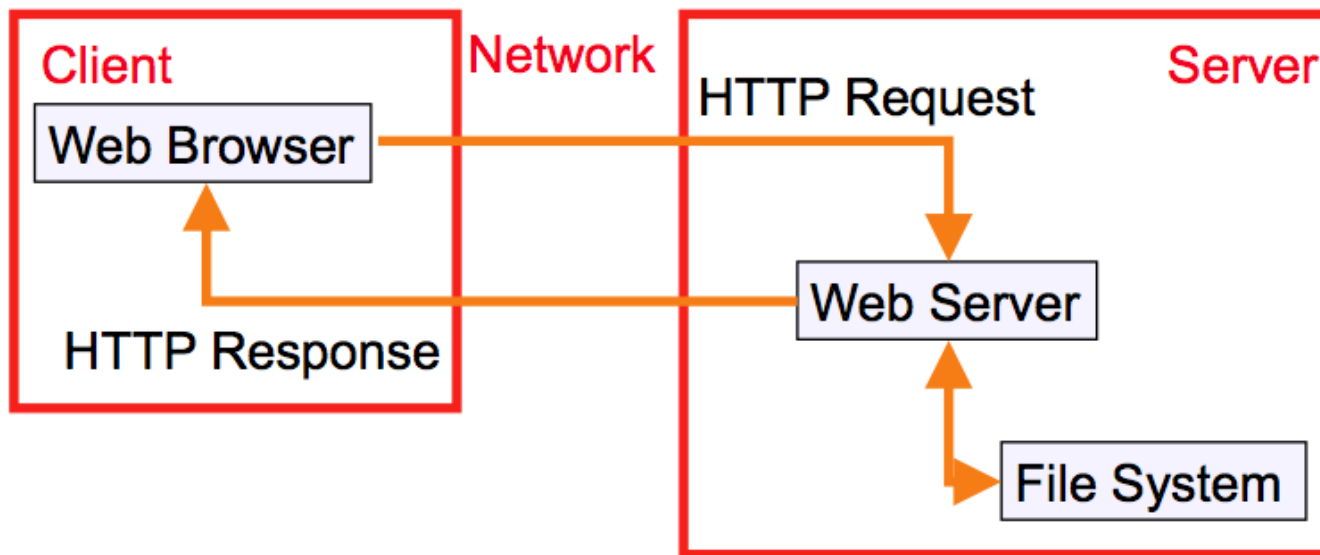
Ciclo de vida



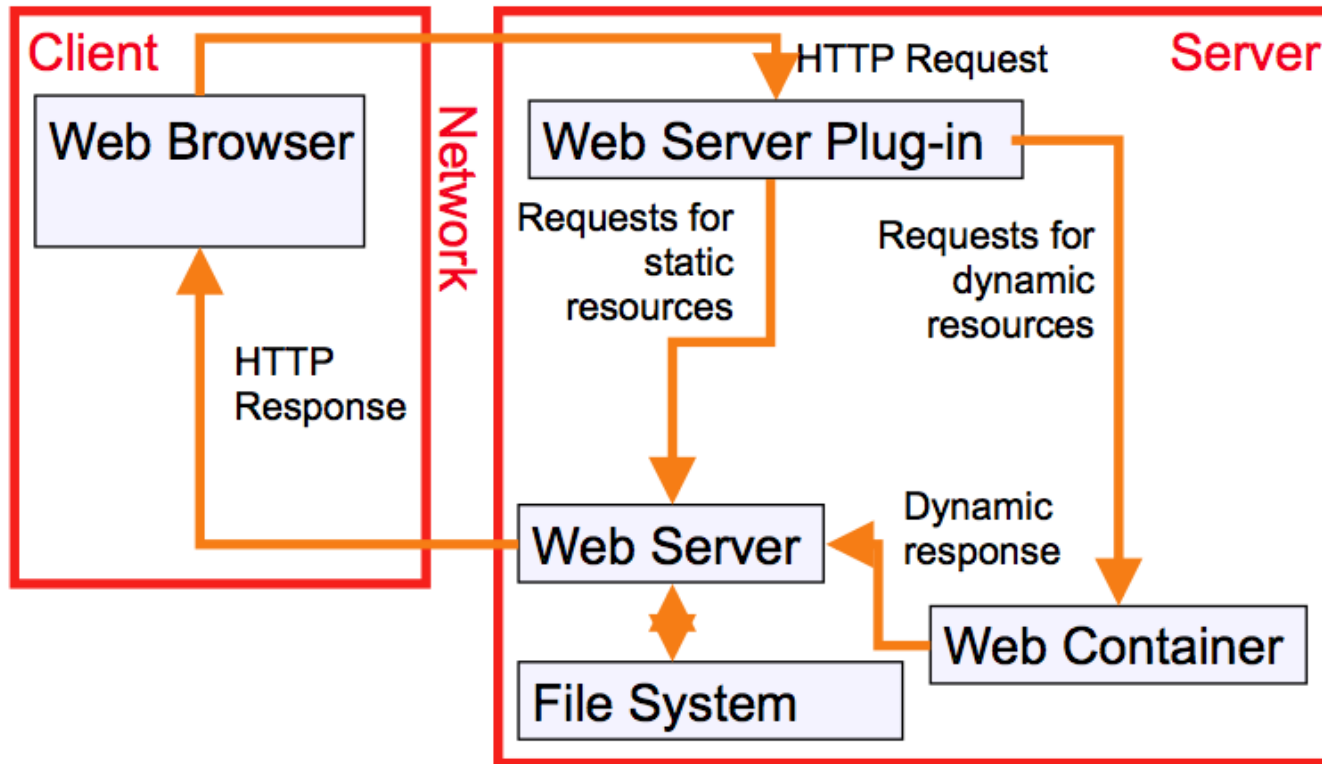
Servlets



Aplicaciones Web



Aplicaciones Web



Acceder a un Servlet



- Por método GET:

Escribiendo la URL en el browser, haciendo clic en un enlace o a través de un formulario:

`http://localhost:8080/aplicacionweb1/helloWorld`

- Por método POST:

Al hacer clic en un formulario que tiene definido como método **post**:

```
<form action="helloWorld" method="post" name="formLogin"  
id="formLogin">
```

```
    <input type="text" name="usuario" id="usuario" />
```

```
    <input type="text" name="clave" id="clave" />
```

```
    <input name="btnEnviar" type="submit" value="Iniciar Sesión" />
```

```
</form>
```

¿Cómo acceder a un Servlet?

1. `http://localhost:8080/aplicacionweb/login.html`

Usuarios de aplicaciones Web



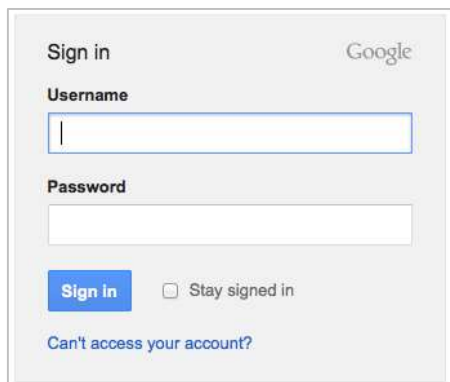
GET request
Recurso: login.html

Servidor Web



Response
retorna HTML

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



Sign in Google

Username

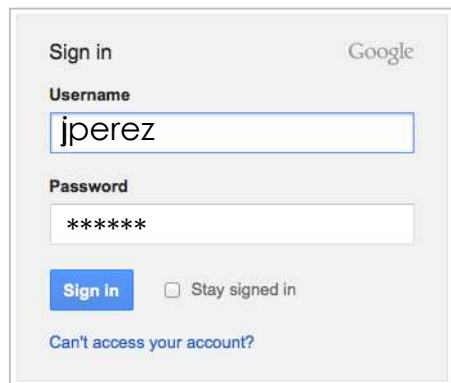
Password

☐ Stay signed in

[Can't access your account?](#)

¿Cómo acceder a un Servlet?

2. `http://localhost:8080/aplicacionweb/servletLogin`



Sign in Google

Username
jperez

Password

☐ Stay signed in

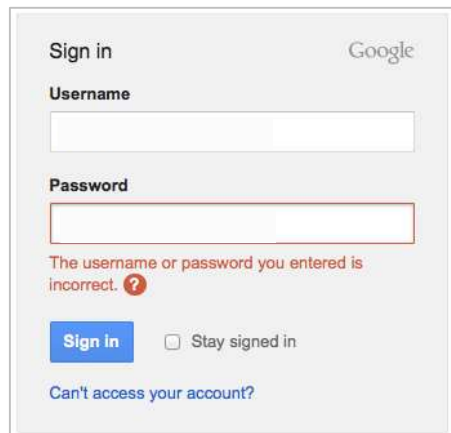
[Can't access your account?](#)

POST request
Recurso: `servletLogin`

Servidor Web



Response
El Servlet genera la respuesta
HTML en base al código
ejecutado.



Sign in Google

Username

Password

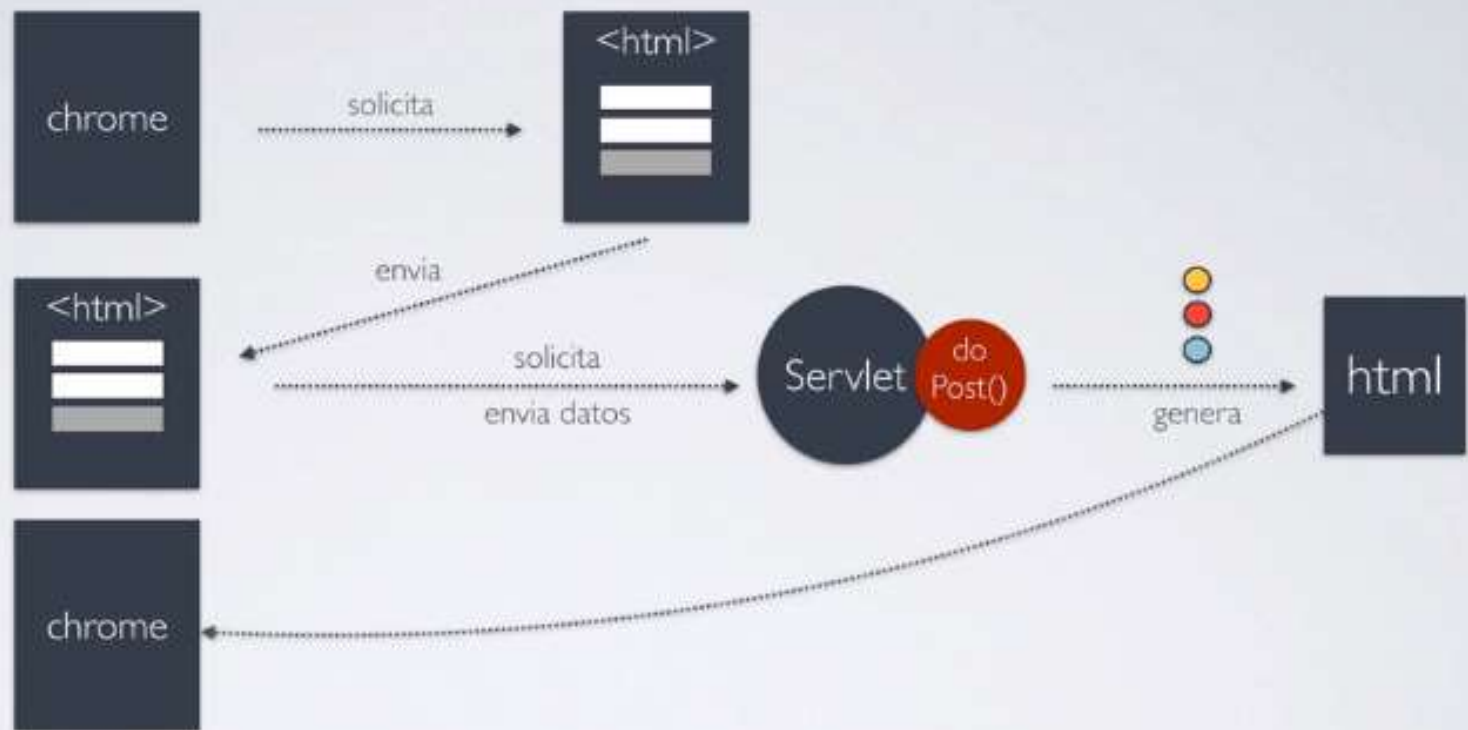
The username or password you entered is incorrect. ?

☐ Stay signed in

[Can't access your account?](#)

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

Peticiones



Ejercicio

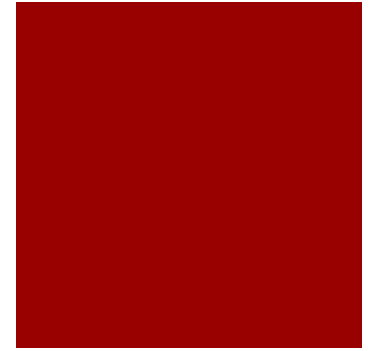


1. Acceso a Servlets

- Conocer la herramienta de trabajo
- Crear un Servlet
- Acceder al Servlet creado haciendo uso de los métodos HTTP GET y POST
- El Servlet devolverá un mensaje indicando el método utilizado según la forma como fue accedido

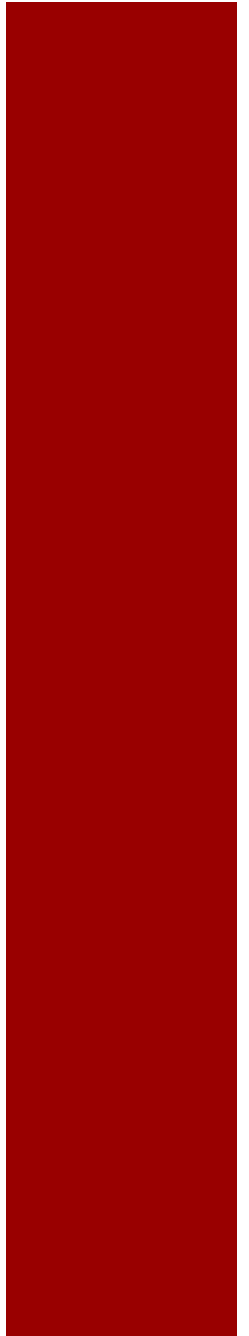
Conclusiones

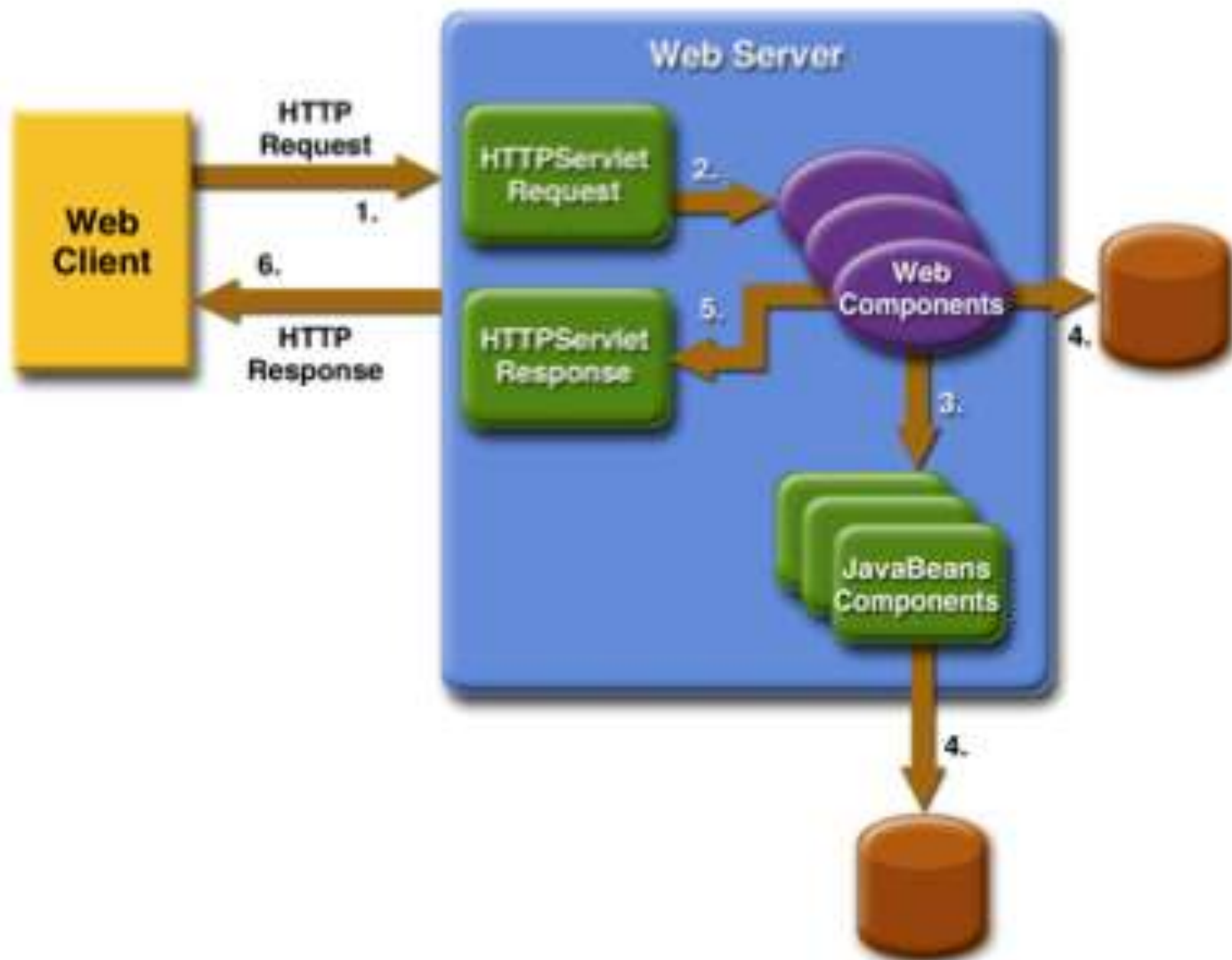
- Usar el método GET para solicitar un recurso u obtener información, existen hasta 3 formas para utilizarlo.
- Cuando sea necesario enviar datos que serán creados o modificados en el servidor, se hará uso del método POST.
- Los Servlets nos permiten recibir peticiones a través de la cuales se le puede enviar información, que permitirá crear repuestas dinámicamente en base a la información recibida por el Servlet.



HttpServletRequest

Información de la petición





HttpServletRequest



- Este objeto esta disponible para los métodos doGet y doPost de los Servlets.
- Proporciona a los Servlets la información completa de la petición (request) que realizan los clientes (navegador/browser), información enviada como parte de la URL o a través de formularios.
- La información proporcionada se presenta como pares clave-valor y sirven para generar el contenido de la respuesta.

HttpServletRequest



- `public String getParameter(String clave)`
Devuelve el valor de un parámetro simple.
Ej: `request.getParameter("clave");`
- `public String[] getParameterValues(String clave)`
Devuelve todos los valores de un parámetro compuesto.
Ej: `request.getParameterValues("clave");`
- `public Enumeration getParameterNames()`
Para obtener todos los nombres de los parámetros enviados desde el cliente.

POST

Settings

[General](#) [Labels](#) [Accounts and Import](#) [Filters](#) [Forwarding and POP/IMAP](#) [Chat](#) [We](#)

Language: Gmail display language: English (US)

Maximum page size: Show 100 conversations per page
Show 250 contacts per page

Keyboard shortcuts: ☒ Keyboard shortcuts off
[Learn more](#) ☐ Keyboard shortcuts on

Undo Send: ☒ Enable Undo Send
Send cancellation period: 10 seconds

Browser connection: ☒ Always use https
[Learn more](#) ☐ Don't always use https

```
<form id="login_form" action="alias_servlet" method="POST">
```

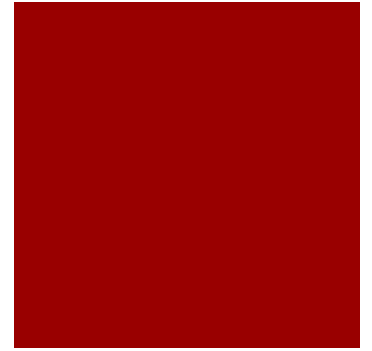
```
<select name="idioma" id="idioma">  
  <option value="en">English (US)</option>  
  <option value="es">Español</option>  
</select>
```

```
<input type="radio" id="ksoff" checked="" value="0" name="kso">  
<input type="radio" id="kson" checked="" value="1" name="kso">
```

```
<input type="checkbox" name="usend" id="usend" value="1">
```

```
</form>
```

POST



```
public class helloWorld extends HttpServlet {  
    protected void doPost(HttpServletRequest request,  
                           HttpServletResponse response) {  
        String var1 = request.getParameter("idioma");  
        String var2 = request.getParameter("kso");  
        String var3 = request.getParameter("usend");  
    }  
}
```


POST

Fecha de nacimiento: ☐ Mostrar el anuncio el día del cumpleaños del usuario
Inclinación sexual: [?] ☒ Todos ☐ Hombres ☐ Mujeres
Relación: [?] ☒ Todos ☐ Soltero(a) ☐ Comprometido(a)
☐ Tiene una relación ☐ Casado(a)
Idiomas: [?]

```
<form id="login_form" action="alias_servlet" method="POST">
  <input type="checkbox" name="birth" id="birth" value="1">

  <input type="radio" id="is1" checked="checked" value="0" name="is">
  <input type="radio" id="is2" checked="" value="1" name="is">
  <input type="radio" id="is3" checked="" value="2" name="is">

  <input type="checkbox" name="rel" id="rel" value="0">
  <input type="checkbox" name="rel_det" id="rel_det1" value="1">Sol
  <input type="checkbox" name="rel_det" id="rel_det2" value="2">Comp
  <input type="checkbox" name="rel_det" id="rel_det3" value="3">Tiene
  <input type="checkbox" name="rel_det" id="rel_det4" value="4">Cas

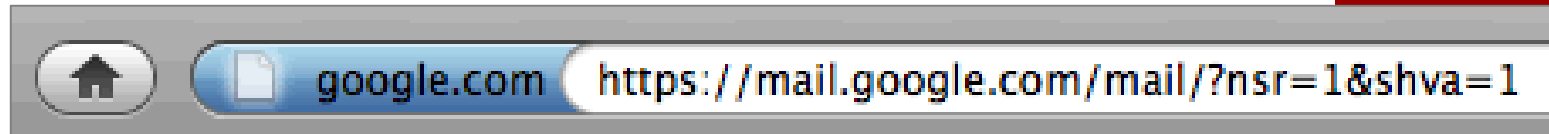
  <input type="text" name="idioma" id="idioma" value="">
</form>
```

POST



```
public class helloWorld extends HttpServlet {  
    protected void doPost(HttpServletRequest request,  
                           HttpServletResponse response) {  
        String var4 = request.getParameter("birth");  
        String var5 = request.getParameter("is");  
        String var6 = request.getParameter("rel");  
        String var7[] =request.getParameterValues("rel_det");  
        String var8 = request.getParameter("idioma");  
    }  
}
```

GET

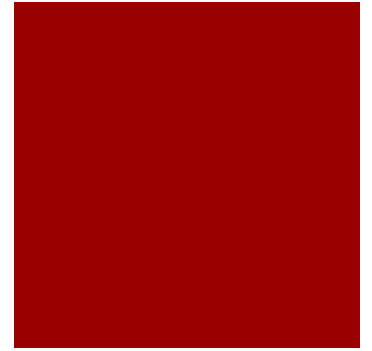


[Regístrate](http://www.facebook.com/help/?page=173)

[Inicio de sesión y contraseña](http://www.facebook.com/help/?page=174)

[Bloquear](/help/?search=block)

GET



```
public class helloWorld extends HttpServlet {  
    protected void doGet(HttpServletRequest request,  
                           HttpServletResponse response) {  
        String var10 = request.getParameter("nsr");  
        String var11 = request.getParameter("shva");  
    }  
}
```

Ejercicio

1. Crear la pagina registro.html
2. Crear el Servlet Registro
3. Colocar el código en el método correspondiente del Servlet para capturar y mostrar la información enviada del formulario.

Registro de usuarios

Correo*

Clave*

Repetir clave*

Sexo*

☐ Masculino ☐ Femenino

Carrera*

Ingeniería de Sistemas ▾

Intereses*

☐ Noticias ☐ Deportes ☐ Entretenimiento ☐ Automoviles ☐ Tecnologia

Lenguajes*

Inglés
Francés
Español
Italiano

Mensaje*

Ejercicio

1. Crear la pagina noticias.html
2. Crear el Servlet Noticias
3. Colocar el código en el método correspondiente del Servlet para capturar y mostrar la información enviada del formulario.



50+ Events in Social Media, Social Intelligence and More

1.4K SHARES / Jul 25, 2013



Twitter Rolls Out TV Ad Targeting to All National Advertisers

892 SHARES / Jul 23, 2013



40+ Events in 3D Printing, Marketing and More

617 SHARES / Jul 18, 2013

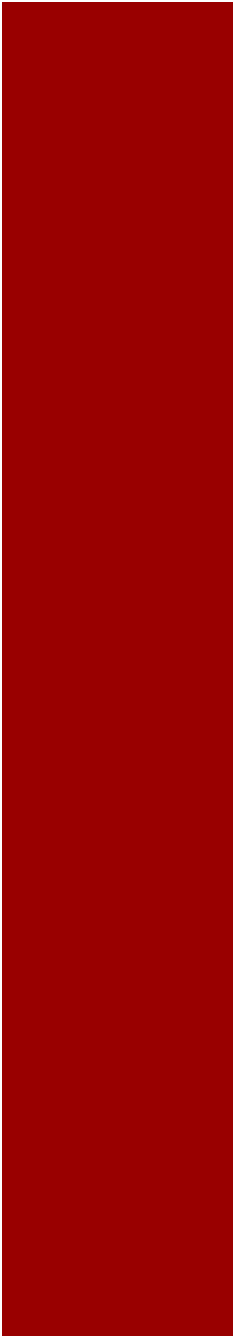


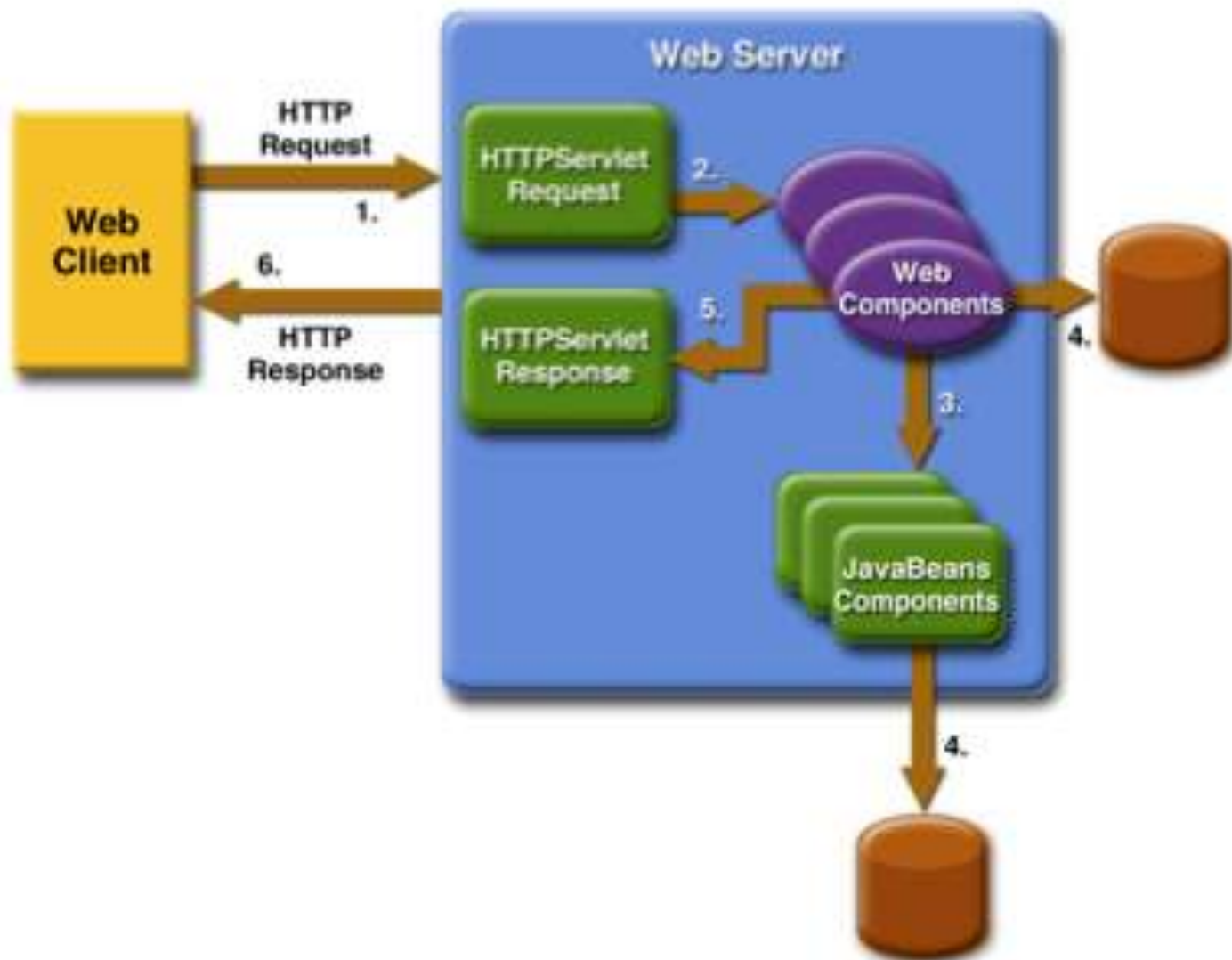
7 Hot Management Jobs in Miami, New York and More

499 SHARES / Jul 17, 2013

HttpServletResponse

Información de la respuesta





HttpServletResponse



- Este objeto encapsula toda la información que debe ser retornada desde el servidor al cliente:
 - Permite enviar información de respuesta desde el servidor al cliente (navegador) (PrintWriter).
 - Podemos indicar el tipo de contenido que será enviado como respuesta al cliente.
 - Se puede especificar el código de estado y la cabecera de respuesta.
- Nos permite hacer redirecciones.

HttpServletResponse



- A través de este objeto, podemos usar el método `setContentType(String tipo)` para indicar el tipo de contenido a responder.
- Desde el Servlet podemos indicar el tipo de contenido de respuesta:
 - `text/html`
 - `image/gif`
 - `image/jpeg`
 - `application/pdf`
 - `application/zip`

HttpServletResponse

```
package prog1.sesion2;
import java.io.*;

public class PrimerServlet extends HttpServlet{
    protected void doGet
        (HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        resp.setContentType("text/html");

        PrintWriter out = resp.getWriter();
        out.print("<html>");
        out.print("<body>");
        out.print("<p>mensaje desde el servlet</p>");
        out.print("</body>");
        out.print("</html>");
        out.close();
    }
}
```

HttpServletResponse



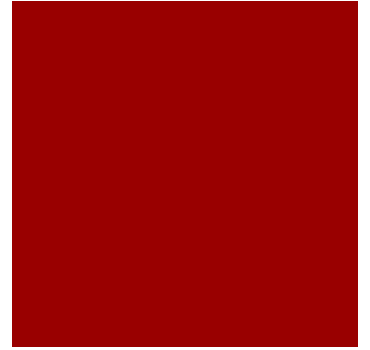
- Con este objeto, también podemos indicar el código de estado que se va a devolver al cliente:
 - 200:OK, todo es correcto
 - 301:Movido permanentemente
 - 404: No encontrado
 - 405:Método no permitido (GET, POST)
 - 500: Error interno

HttpServletResponse

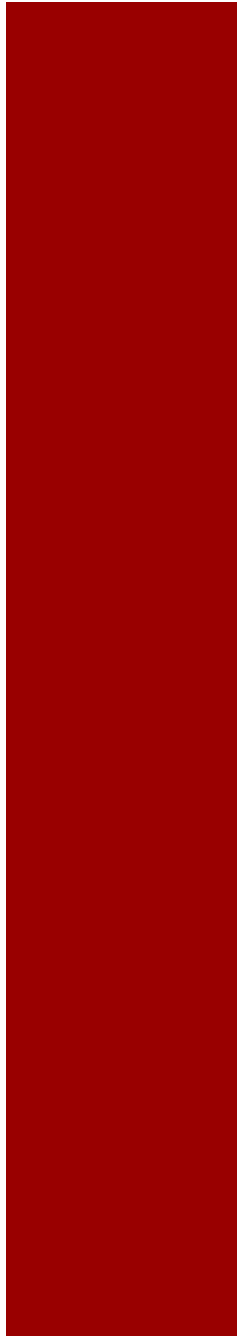
```
public class ServletResponse extends HttpServletResponse {  
    public void doGet(HttpServletRequest req,  
        HttpServletResponse res)  
        throws ServletException, IOException {  
  
        res.setStatus(HttpServletResponse.SC_OK);  
        res.setContentType("text/html");  
        PrintWriter out = res.getWriter();  
  
        out.println("<HTML><BODY><H1>Today is " + (new Date()));  
        out.println("</H1></BODY></HTML>");  
    }  
}
```

Ejercicio

1. En los Servlets creados anteriormente, definir el tipo de contenido a devolver como respuesta.

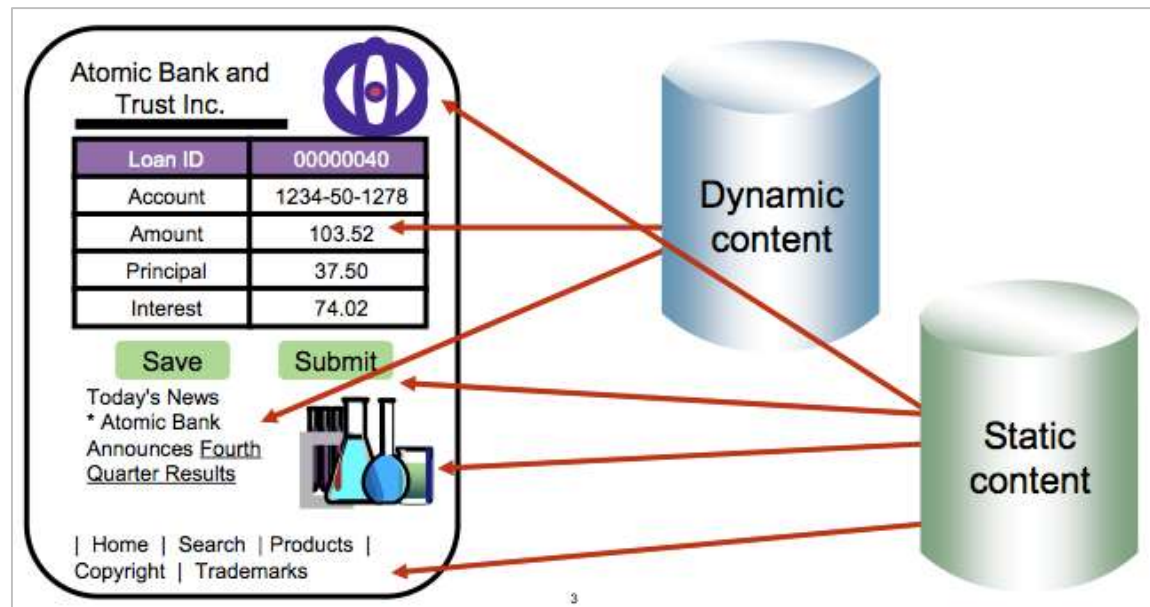


JSP



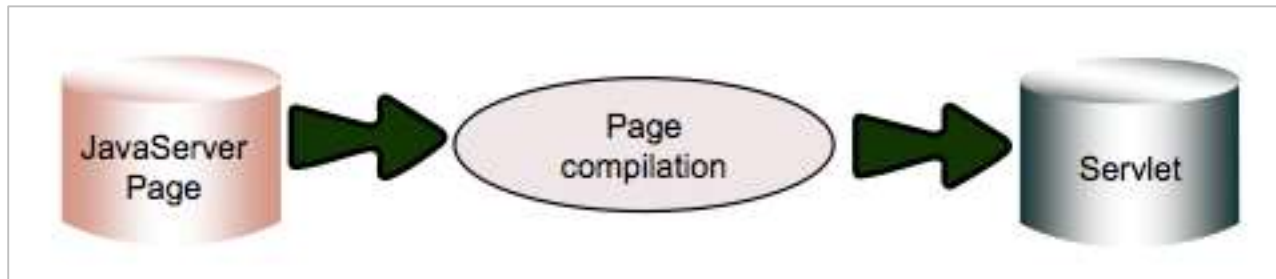
JSP

- Es la tecnología Java que permite combinar HTML estático con HTML generado dinámicamente.
- Podemos utilizar código de servidor, en este caso código Java para generar dinámicamente código HTML.



JSP

- Proceso de compilación de las páginas JSP:
 - El código del JSP es analizado.
 - El código del Servlet es generado.
 - El Servlet generado es compilado, cargado y ejecutado.



1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024, 2025, 2026, 2027, 2028, 2029, 2030, 2031, 2032, 2033, 2034, 2035, 2036, 2037, 2038, 2039, 2040, 2041, 2042, 2043, 2044, 2045, 2046, 2047, 2048, 2049, 2050, 2051, 2052, 2053, 2054, 2055, 2056, 2057, 2058, 2059, 2060, 2061, 2062, 2063, 2064, 2065, 2066, 2067, 2068, 2069, 2070, 2071, 2072, 2073, 2074, 2075, 2076, 2077, 2078, 2079, 2080, 2081, 2082, 2083, 2084, 2085, 2086, 2087, 2088, 2089, 2090, 2091, 2092, 2093, 2094, 2095, 2096, 2097, 2098, 2099, 2100, 2101, 2102, 2103, 2104, 2105, 2106, 2107, 2108, 2109, 2110, 2111, 2112, 2113, 2114, 2115, 2116, 2117, 2118, 2119, 2120, 2121, 2122, 2123, 2124, 2125, 2126, 2127, 2128, 2129, 2130, 2131, 2132, 2133, 2134, 2135, 2136, 2137, 2138, 2139, 2140, 2141, 2142, 2143, 2144, 2145, 2146, 2147, 2148, 2149, 2150, 2151, 2152, 2153, 2154, 2155, 2156, 2157, 2158, 2159, 2160, 2161, 2162, 2163, 2164, 2165, 2166, 2167, 2168, 2169, 2170, 2171, 2172, 2173, 2174, 2175, 2176, 2177, 2178, 2179, 2180, 2181, 2182, 2183, 2184, 2185, 2186, 2187, 2188, 2189, 2190, 2191, 2192, 2193, 2194, 2195, 2196, 2197, 2198, 2199, 2200, 2201, 2202, 2203, 2204, 2205, 2206, 2207, 2208, 2209, 2210, 2211, 2212, 2213, 2214, 2215, 2216, 2217, 2218, 2219, 2220, 2221, 2222, 2223, 2224, 2225, 2226, 2227, 2228, 2229, 2230, 2231, 2232, 2233, 2234, 2235, 2236, 2237, 2238, 2239, 2240, 2241, 2242, 2243, 2244, 2245, 2246, 2247, 2248, 2249, 2250, 2251, 2252, 2253, 2254, 2255, 2256, 2257, 2258, 2259, 2260, 2261, 2262, 2263, 2264, 2265, 2266, 2267, 2268, 2269, 2270, 2271, 2272, 2273, 2274, 2275, 2276, 2277, 2278, 2279, 2280, 2281, 2282, 2283, 2284, 2285, 2286, 2287, 2288, 2289, 2290, 2291, 2292, 2293, 2294, 2295, 2296, 2297, 2298, 2299, 2300, 2301, 2302, 2303, 2304, 2305, 2306, 2307, 2308, 2309, 2310, 2311, 2312, 2313, 2314, 2315, 2316, 2317, 2318, 2319, 2320, 2321, 2322, 2323, 2324, 2325, 2326, 2327, 2328, 2329, 2330, 2331, 2332, 2333, 2334, 2335, 2336, 2337, 2338, 2339, 2340, 2341, 2342, 2343, 2344, 2345, 2346, 2347, 2348, 2349, 2350, 2351, 2352, 2353, 2354, 2355, 2356, 2357, 2358, 2359, 2360, 2361, 2362, 2363, 2364, 2365, 2366, 2367, 2368, 2369, 2370, 2371, 2372, 2373, 2374, 2375, 2376, 2377, 2378, 2379, 2380, 2381, 2382, 2383, 2384, 2385, 2386, 2387, 2388, 2389, 2390, 2391, 2392, 2393, 2394, 2395, 2396, 2397, 2398, 2399, 2400, 2401, 2402, 2403, 2404, 2405, 2406, 2407, 2408, 2409, 2410, 2411, 2412, 2413, 2414, 2415, 2416, 2417, 2418, 2419, 2420, 2421, 2422, 2423, 2424, 2425, 2426, 2427, 2428, 2429, 2430, 2431, 2432, 2433, 2434, 2435, 2436, 2437, 2438, 2439, 2440, 2441, 2442, 2443, 2444, 2445, 2446, 2447, 2448, 2449, 2450, 2451, 2452, 2453, 2454, 2455, 2456, 2457, 2458, 2459, 2460, 2461, 2462, 2463, 2464, 2465, 2466, 2467, 2468, 2469, 2470, 2471, 2472, 2473, 2474, 2475, 2476, 2477, 2478, 2479, 2480, 2481, 2482, 2483, 2484, 2485, 2486, 2487, 2488, 2489, 2490, 2491, 2492, 2493, 2494, 2495, 2496, 2497, 2498, 2499, 2500, 2501, 2502, 2503, 2504, 2505, 2506, 2507, 2508, 2509, 2510, 2511, 2512, 2513, 2514, 2515, 2516, 2517, 2518, 2519, 2520, 2521, 2522, 2523, 2524, 2525, 2526, 2527, 2528, 2529, 2530, 2531, 2532, 2533, 2534, 2535, 2536, 2537, 2538, 2539, 2540, 2541, 2542, 2543, 2544, 2545, 2546, 2547, 2548, 2549, 2550, 2551, 2552, 2553, 2554, 2555, 2556, 2557, 2558, 2559, 2560, 2561, 2562, 2563, 2564, 2565, 2566, 2567, 2568, 2569, 2570, 2571, 2572, 2573, 2574, 2575, 2576, 2577, 2578, 2579, 2580, 2581, 2582, 2583, 2584, 2585, 2586, 2587, 2588, 2589, 2590, 2591, 2592, 2593, 2594, 2595, 2596, 2597, 2598, 2599, 2600, 2601, 2602, 2603, 2604, 2605, 2606, 2607, 2608, 2609, 2610, 2611, 2612, 2613, 2614, 2615, 2616, 2617, 2618, 2619, 2620, 2621, 2622, 2623, 2624, 2625, 2626, 2627, 2628, 2629, 2630, 2631, 2632, 2633, 2634, 2635, 2636, 2637, 2638, 2639, 2640, 2641, 2642, 2643, 2644, 2645, 2646, 2647, 2648, 2649, 2650, 2651, 2652, 2653, 2654, 2655, 2656, 2657, 2658, 2659, 2660, 2661, 2662, 2663, 2664, 2665, 2666, 2667, 2668, 2669, 2670, 2671, 2672, 2673, 2674, 2675, 2676, 2677, 2678, 2679, 26

- **Directivas**, son instrucciones para el compilador y motor JSP:
 - `<%@ import="clase" %>` `<%@ include file="archivo.jsp" %>`
 - `<%@ page isErrorPage="true" %>` `<%@ page errorPage="file.jsp" %>`
- **Scripting**:
 - **Declaraciones**, sirven para declarar métodos y variables
 - `<%! codigo %>`
 - **Scriptlets**, son líneas de código Java
 - `<% codigo %>`
 - **Expresiones**, es código Java que resulta en una cadena
 - `<%=variable%>`

JSP



- **Comentarios:**

- `<%-- texto de comentario --%>`

- **Actions**

- `<jsp:include page="StdHeader.jsp" flush="true" />`
- `<jsp:forward page="ExtraInfo.jsp" />`

JSP



- Servlet = Código Java + HTML
- JSP = HTML + Código Java

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HEAD><TITLE></TITLE></HEAD>
<BODY>
<%TestAreas tests=
    (TestAreas)request.getAttribute("tests");%>

<TABLE cellpadding="10"><TBODY>
<TR>
<TD><FONT color="#0000cc" size="5" face="Comic Sans MS">
The following areas have tests available</FONT></TD>
</TR>

<%for (int i = 0; i < tests.getTestAreas().length; i++) {%>
    <TR><TD align="center">

        <!-- The following two lines are really 1 line -->
        <A href="/Course/ExamCommand?cmd=displayTestsByArea&testArea=
        <%= tests.getTestAreas(i).getKey() %>">

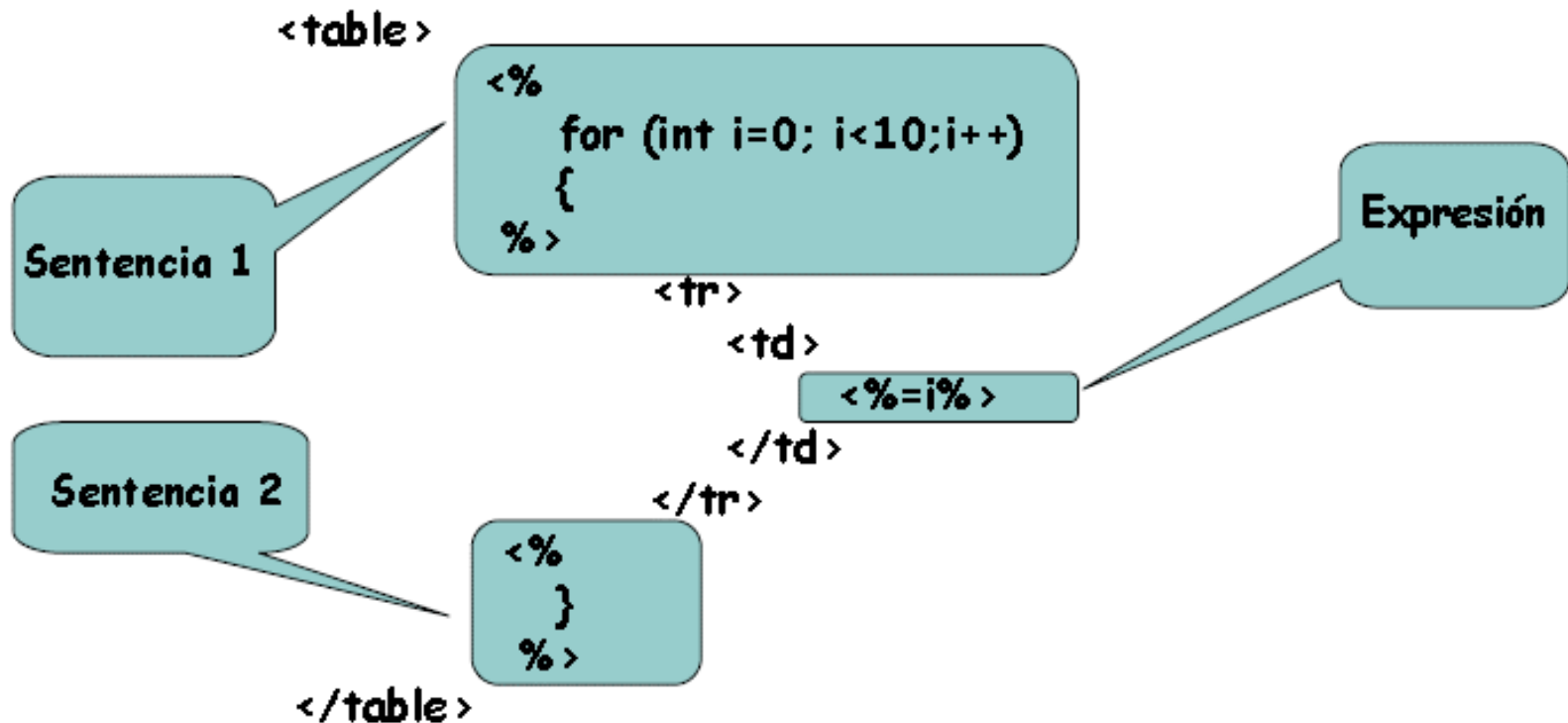
        <IMG border="0"
            src="<%= tests.getTestAreas(i).getImageFilePath() %>">
        </A>
    </TD></TR>
<%}%>

</TBODY> </TABLE> </BODY>
```

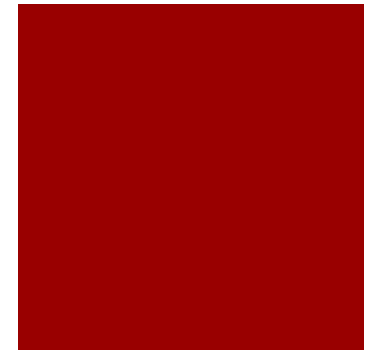
```
<%@ page language="java" %>
```

```
<%@ include file="companyBanner.html"%>
```

Ejemplos de JSP



Ejemplos de JSP



Sentencia

<%

```
out.println("<table>");  
for(int i=0;i<10;i++)  
    out.println("<tr><td>" + i + "</td></tr>");  
out.println("</table>");
```

%>

Objeto implícito out

Ejemplo de JSP

**Declaración
Metodo**

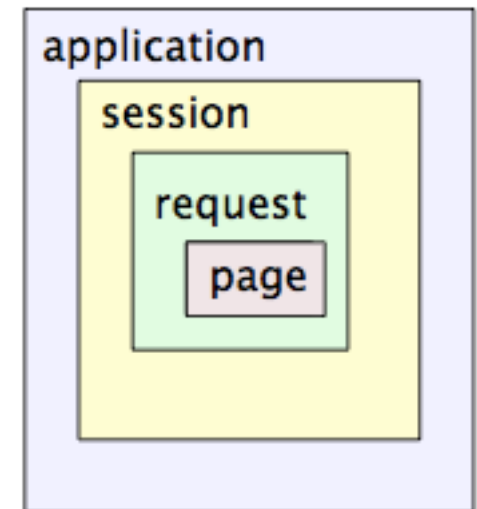
```
<%!  
    private String ahora()  
    {  
        return ""+new java.util.Date();  
    }  
%>
```

Expresion

```
<html>  
    <body>  
        <%=ahora() %>  
    </body>  
</html>
```

Ámbitos

Servlet	JSP
PageContext	Page
HttpServletRequest	request
HttpSession	session
ServletContext	Application



JSP



- **Page**, el alcance es corto, es parecido a tener una variable local.
- **Request**, el alcance se mantiene a través del request hasta que la respuesta es enviada al cliente (HttpServletRequest).
 - request.getAttribute()
 - request.setAttribute()
- **Session**, se tiene el alcance mientras se mantenga la sesión del usuario (HttpSession).
 - session.getAttribute()
 - session.setAttribute()
- **Application**, permite compartir información para toda la aplicación (ServletContext).
 - application.getAttribute()
 - application.setAttribute()

JSP



- Variables predefinidas:
 - **request**, objeto HttpServletRequest
 - **response**, objeto HttpServletResponse
 - **session**, objeto HttpSession
 - **application**, objeto ServletContext
 - **config**, objeto ServletConfig
 - **out**, JspWriter

JSP



- La página JSP puede ser invocada:
 - Por URL
 - Por un Servlet
 - Por otra página JSP
- Una página JSP puede invocar:
 - Un Servlet
 - Otra página JSP

¿JSP o Servlet?



■ Servlets

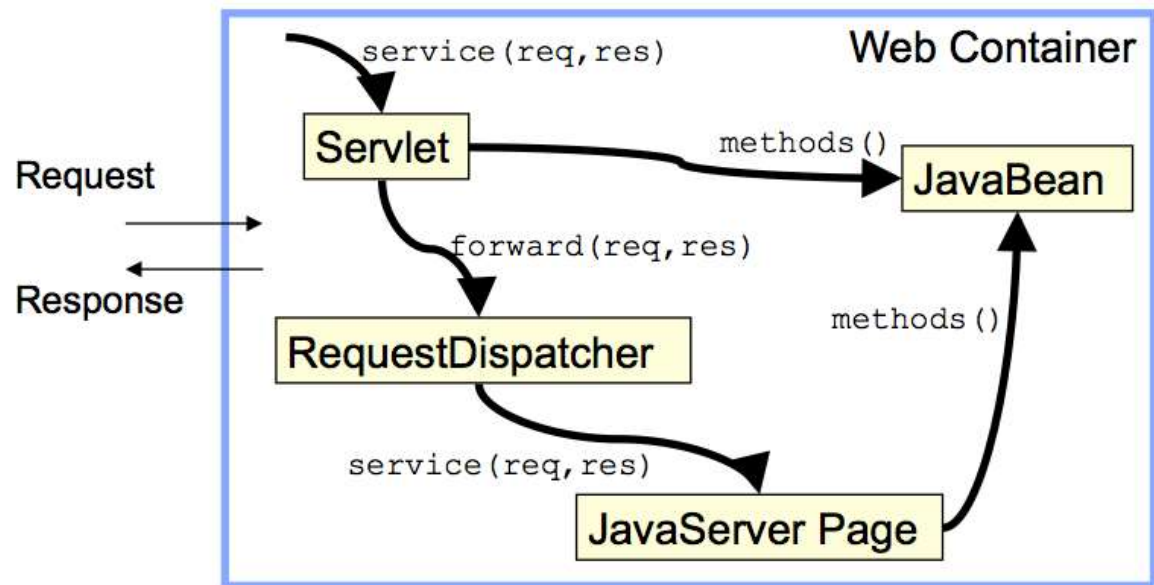
- Determinar los procesos requeridos para cumplir con la petición
- Validar la información enviada en la petición
- Acceder a los datos y realizar el proceso requerido por la petición
- Controlar el flujo de la aplicación Web

■ JSP

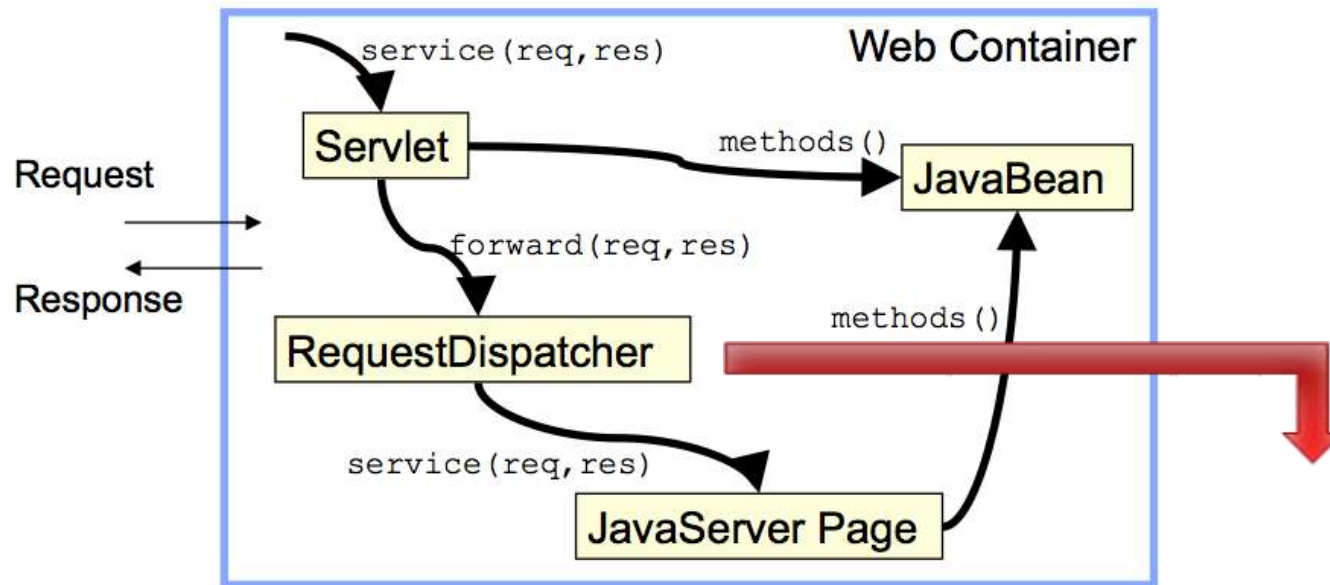
- Mostrar el contenido generado por la aplicación Web

Aplicaciones con JSP

1. El Servlet recibe las peticiones del cliente.
2. El Servlet ejecuta la lógica de negocio que tiene desarrollado.
3. Realiza una redirección a la página JSP utilizando RequestDispatcher.
4. El JSP contiene el HTML de respuesta.



Aplicaciones con JSP



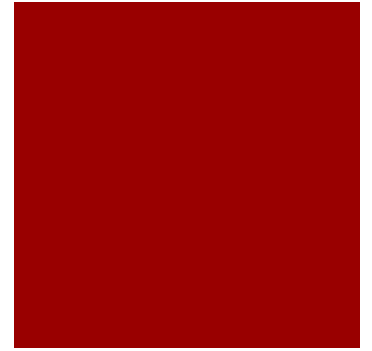
Recurso, puede ser:

- Alias de Servlet
- Página JSP



```
request.getRequestDispatcher("/recurso")  
    .forward(request, response);
```

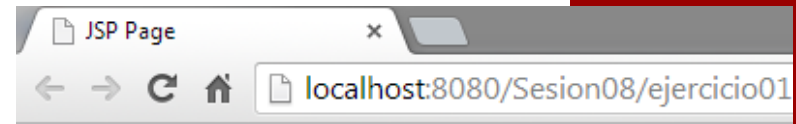
Ejercicio



1. Crear la página JSP index.jsp
 - Hacer uso de scriptlets
 - Crear una variable y mostrar su valor, hacer uso de declaraciones y expresiones
 - Incluir comentarios en el JSP

Ejercicio

- Realizar la siguiente pagina JSP, usando los diferentes elementos aprendidos en clase.



Cabecera de la pagina

La hora actual es : Sun Mar 16 19:57:05 COT 2014

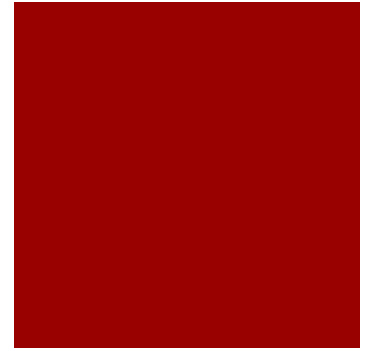
Hola tu edad es: 50

Tabla del 12 usando FOR

12 * 0 = 0
12 * 1 = 12
12 * 2 = 24
12 * 3 = 36
12 * 4 = 48
12 * 5 = 60
12 * 6 = 72
12 * 7 = 84
12 * 8 = 96
12 * 9 = 108
12 * 10 = 120
12 * 11 = 132
12 * 12 = 144

Pie de la pagina usando include

Ejercicio



1. Crear las páginas JSP siguientes en base a la página index.jsp creada anteriormente:
 - quienes_somos.jsp
 - contactenos.jsp
2. Hacer uso de la directiva include para incluir secciones repetitivas dentro de las páginas.

Enlaces de interés



- HTML
 - <https://developer.mozilla.org/en-US/docs/Web/Guide/HTML>
- HTTP

http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol
- Métodos HTTP

<http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html>
- HttpServletRequest
 - http://docs.oracle.com/cd/E17802_01/products/products/servlet/2.1/api/javax.servlet.http.HttpServletRequest.html#_top_
- HttpServletResponse
 - http://docs.oracle.com/cd/E17802_01/products/products/servlet/2.1/api/javax.servlet.http.HttpServletResponse.html#_top_
- JSP
 - <http://www.oracle.com/technetwork/java/javaee/jsp/index.html>