

Taller de Tecnologías de Producción de Software

Cursada 2016

Docentes

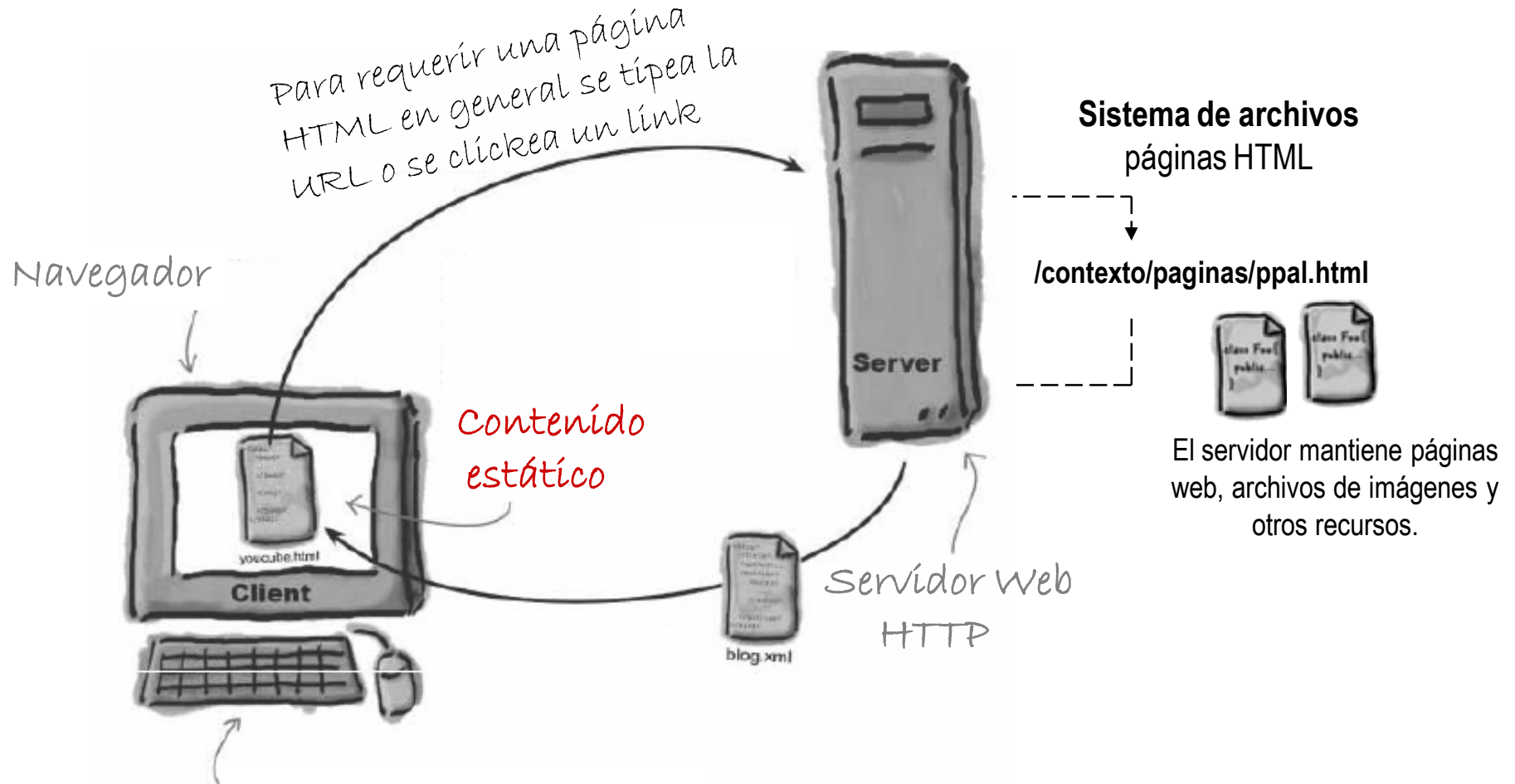
Laura Fava, Jorge Rosso, Vanessa Aybar, Luciano
Nomdedeu, Javier Corvi, Manuel Ortiz

Contenido

- Evolución de las aplicaciones web
- Tecnologías Java
 - Plataformas de desarrollo
 - Plataformas de ejecución
 - La plataforma empresarial Java (J2EE)
- Arquitectura de una aplicación
 - Capas lógicas (layers)
 - Capas físicas (tiers)
- Breve síntesis de HTTP y HTML
- Servidores J2EE
- IDEs para desarrollo de aplicaciones Java
- Frameworks para desarrollo de aplicaciones Java

Las Aplicaciones Web

Cuando un Cliente Web (Navegador) hace un requerimiento de una página Web, el Servidor Web lo recibe, busca la página HTML y la envía al cliente que hizo el pedido.



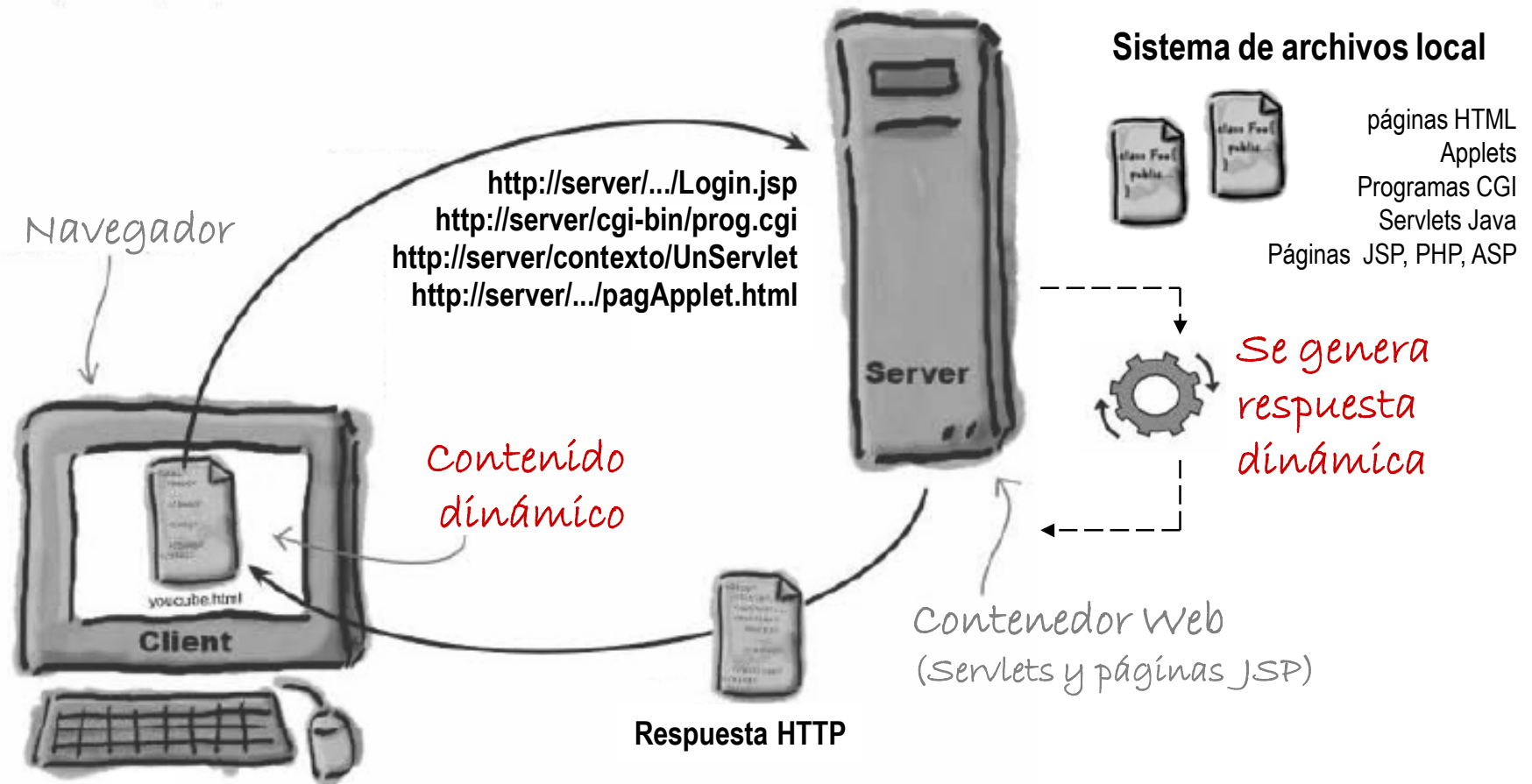
Las Aplicaciones Web

Tecnología del lado del cliente

- Programas del lado del Cliente: **Applets (1995)**

Tecnologías del lado del servidor

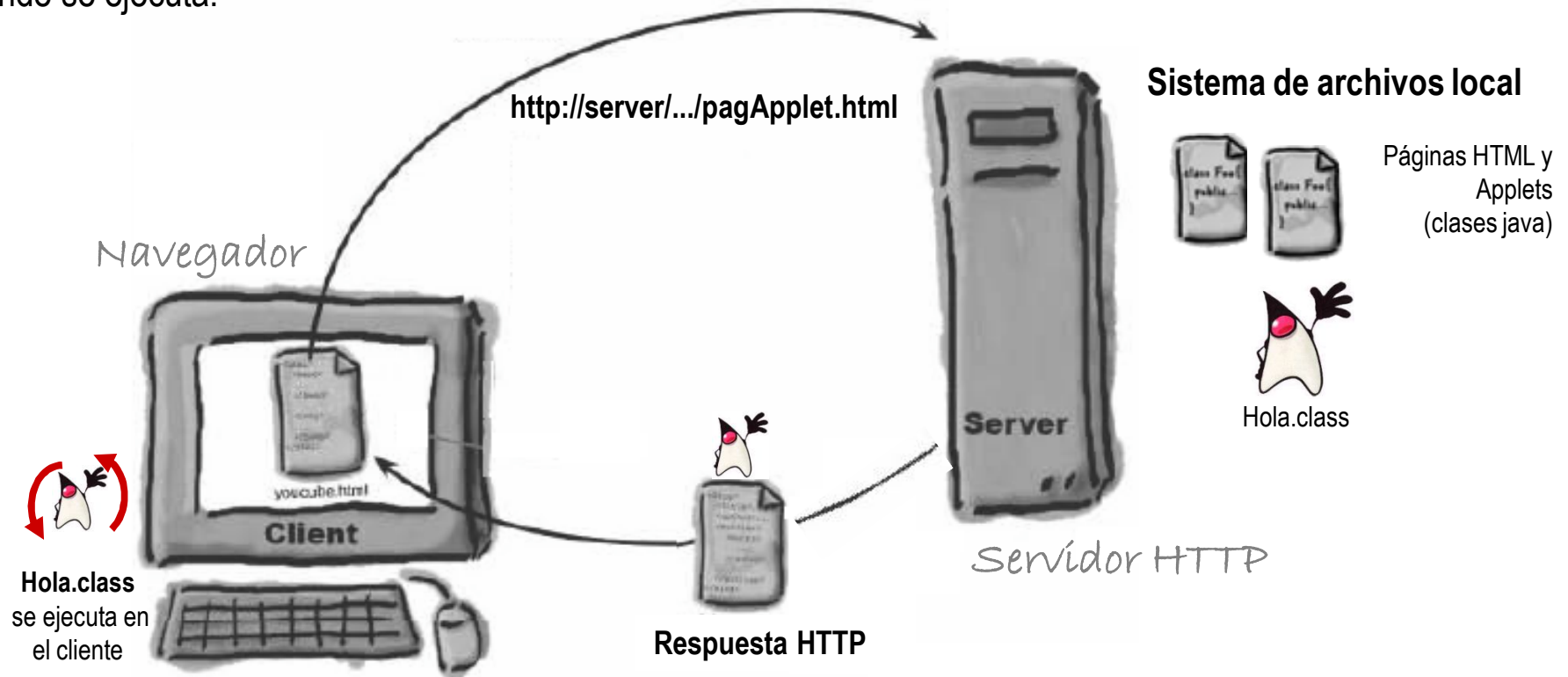
- Programas o Scripts: CGI y **Servlets (1996)**
- Páginas con Scripts: ASP, PHP y **JSP (1999)**



Las Aplicaciones Web

Tecnologías java del lado del cliente: **Applets**

Son pequeños programas que requieren de un Navegador Web con una JVM embebida para su ejecución. Se hacen disponibles mediante páginas web escritas en el lenguaje HTML (HyperText Markup Language). Cuando se accede a una página web que tiene un tag **<OBJECT>** ó **<APPELT>** en el HTML, las clases necesarias para ejecutar el applet se descargan automáticamente desde un servidor a la máquina cliente donde se ejecuta.

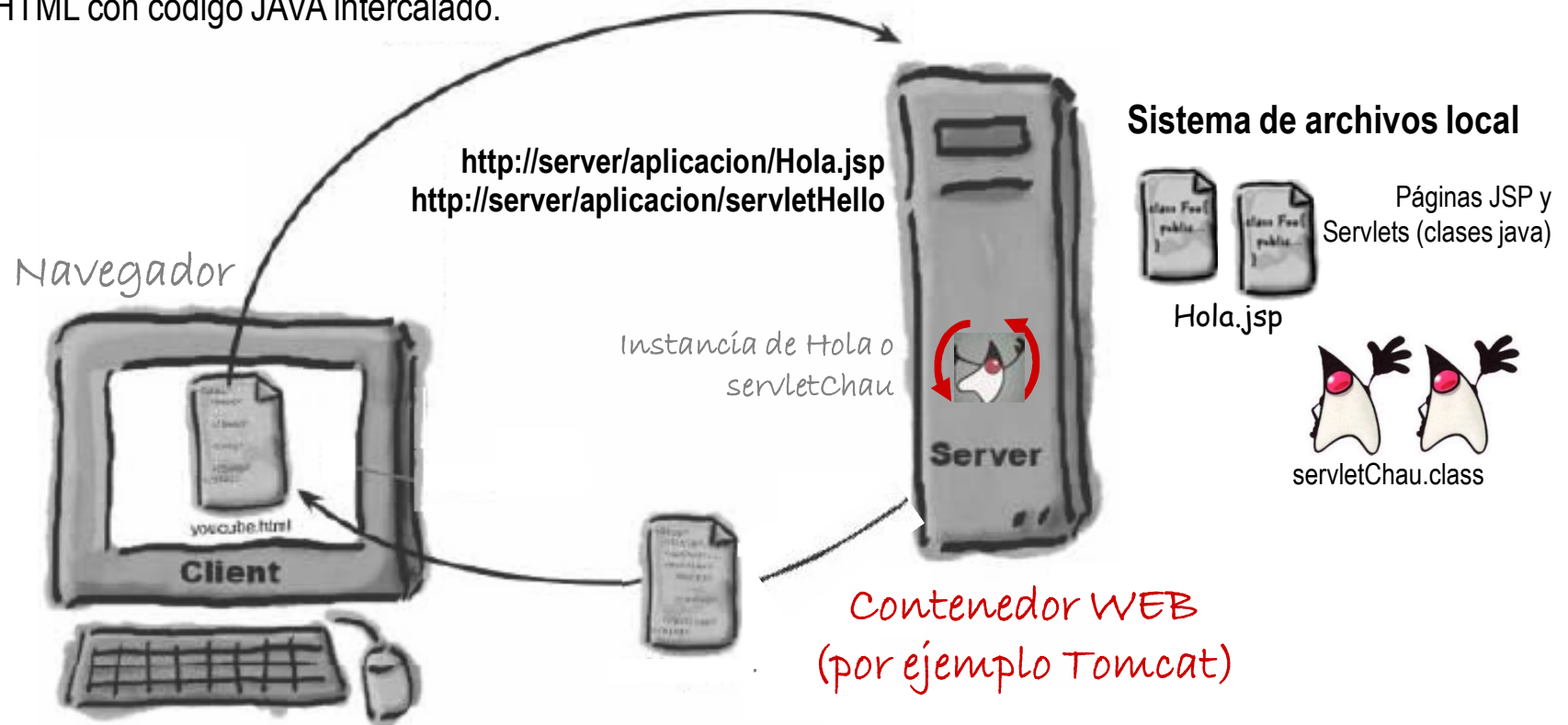


Las Aplicaciones Web

Tecnologías del lado del servidor: **Servlets y JSP**

Con el objetivo de proveer mecanismos para generar contenido dinámico SUN propone 2 tecnologías del lado del servidor: primero Servlets y más tarde JSP.

- Un **Servlet** es un programa Java que se ejecuta en un servidor. Requiere de un Contenedor Web o Contenedor de Servlet para ejecutarse.
- Una **JavaServer Page** también es una componente web Java gerenciada por el Contenedor Web. El código fuente de una JSP es un archivo de texto que combina tags HTML con *elementos de scripting* Java. Básicamente es un archivo HTML con código JAVA intercalado.



Tecnologías Java

Tipos de Plataforma

Plataforma de Desarrollo (Development Platform)

Es software usado para desarrollar software. Incluye un compilador o un intérprete y podría incluir otras herramientas como *debuggers* o editores.

Java EE	Java SE	Java ME
<i>Soluciones Empresariales</i> <ul style="list-style-type: none">▪ eCommerce▪ eBusiness	<i>Soluciones Desktop</i> <ul style="list-style-type: none">▪ Aplicaciones▪ Applets	<i>Soluciones para el Consumidor</i> <ul style="list-style-type: none">▪ Celulares, PDAs, TV▪ navegación <i>wireless</i>

Plataforma de Ejecución (Delivery Platforms)

Es software que provee los servicios necesarios para ejecutar una aplicación.

Java SE (J2SE) tiene una versión para ejecución de aplicaciones llamada Java Runtime Environment (JRE) el cual incluye la Java Virtual Machine (JVM).

La versión para ejecución de la **Java ME (J2ME)** es adaptado para cada tipo de dispositivo, los cuales tienen limitaciones: memoria limitada, procesador + lentos, conexión de red intermitente, pantalla reducida o sin pantalla, etc.

Las aplicaciones **Java EE** ejecutan en Servidores JEE, esto es, servidores que cumplen o implementan las especificaciones que forman parte del Java EE.

Tecnologías Java EE

- La Plataforma Java EE (Java Platform, Enterprise Edition) es el estándar para software empresarial impulsado por la Java Community Process (JCP). Cada nueva versión de la plataforma se define a través del JCP.
- La JCP tiene un comité ejecutivo que guía la evolución de las tecnologías Java. Participan de este comité expertos de la industria, organizaciones comerciales y de código abierto de Java, grupos de usuarios, etc. Todas las tecnologías Java tienen una especificación (JSR) desarrollada por la JCP.
- Cada versión integra nuevas características que resuelven necesidades del sector, mejoran la portabilidad de las aplicaciones y aumentan la productividad de los desarrolladores.
- Contar con una especificación, transforma a todas las **tecnologías JAVA en estándares**. Cada fabricante de software desarrolla su implementación respetando la especificación. De esta manera se garantiza compatibilidad y portabilidad.
- Una especificación comienza como un **Java Specification Request (JSR)**, que pasa por varios estados en la JCP antes de convertirse en la especificación definitiva. Cada JSR tiene asignado un número.

Algunas especificaciones que forman parte de la Plataforma JEE 7:

JSR 245: Especificación de JavaServer Pages 2.1

JSR 340: Especificación de Servlet 3.1

JSR 338: Especificación de Java Persistence 2.1

JSR 339: Java API for RESTful Web Services 2.0

Las especificaciones para J2SE, J2EE, y J2ME, junto con las APIs asociadas, son desarrolladas por el JCP.

Arquitectura de una aplicación

Layers (capas lógicas) y Tiers (capas físicas)

Layers

Son formas de organizar el código. Una aplicación, típicamente incluye como mínimo las capas de Presentación, Negocio y Datos.

¿Por qué usar layers?

- Para lograr beneficios de una organización lógica y agrupar por funcionalidad => reusabilidad.
- Reduce los costos de mantenimiento de la aplicación.
- Mejor diseño.

Tiers

Se refiere a donde corre el código. Las *tiers* son los lugares donde ejecutan las *layers*. Cada recurso ejecutando en un proceso es considerado un *tier*.

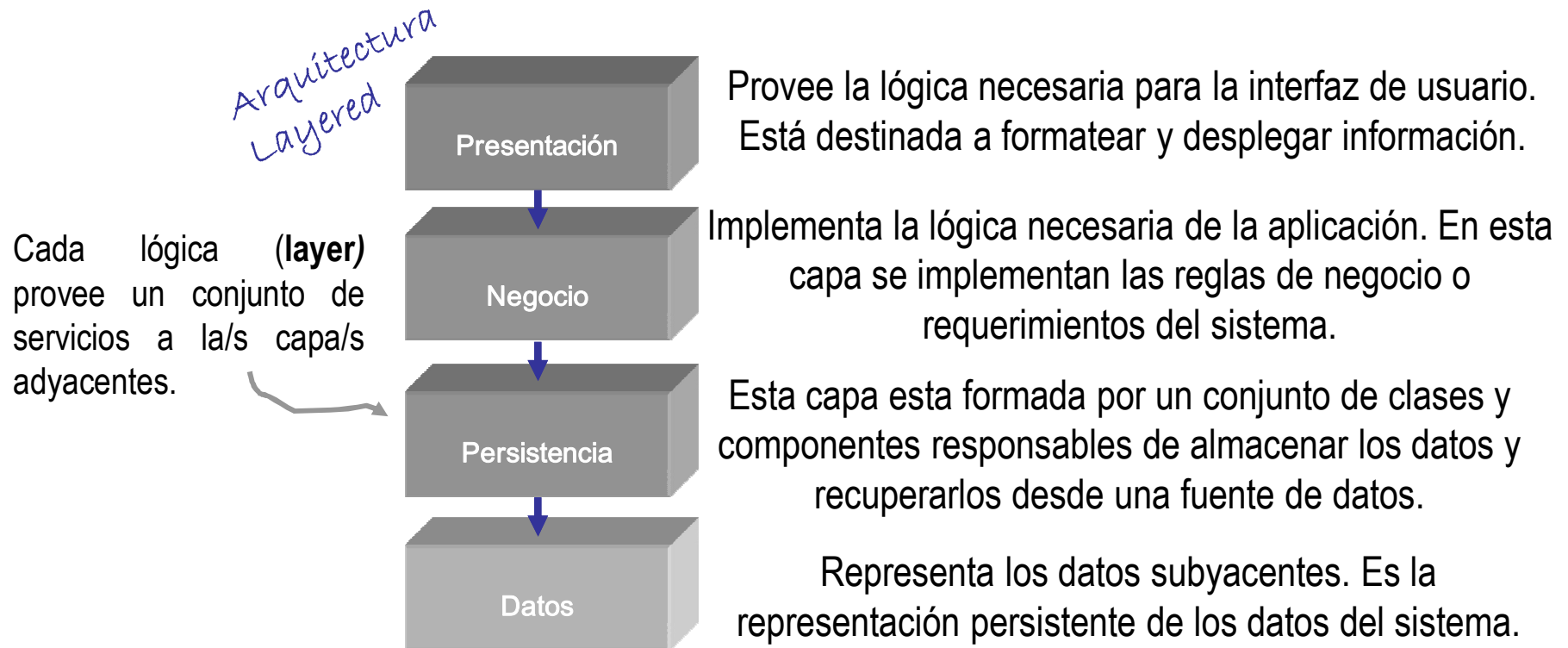
¿Cual es el beneficio de hacer el despliegue (deploy) y la ejecución de las capas lógicas en las capas físicas?

Para lograr escalabilidad, tolerancia a fallas y seguridad.

Arquitectura de una aplicación

Layers (capas lógicas)

La mayoría de las aplicaciones pueden ser organizadas para que soporten un conjunto de capas lógicas o layers. Una arquitectura de alto nivel para una aplicación empresarial, usa estas capas:

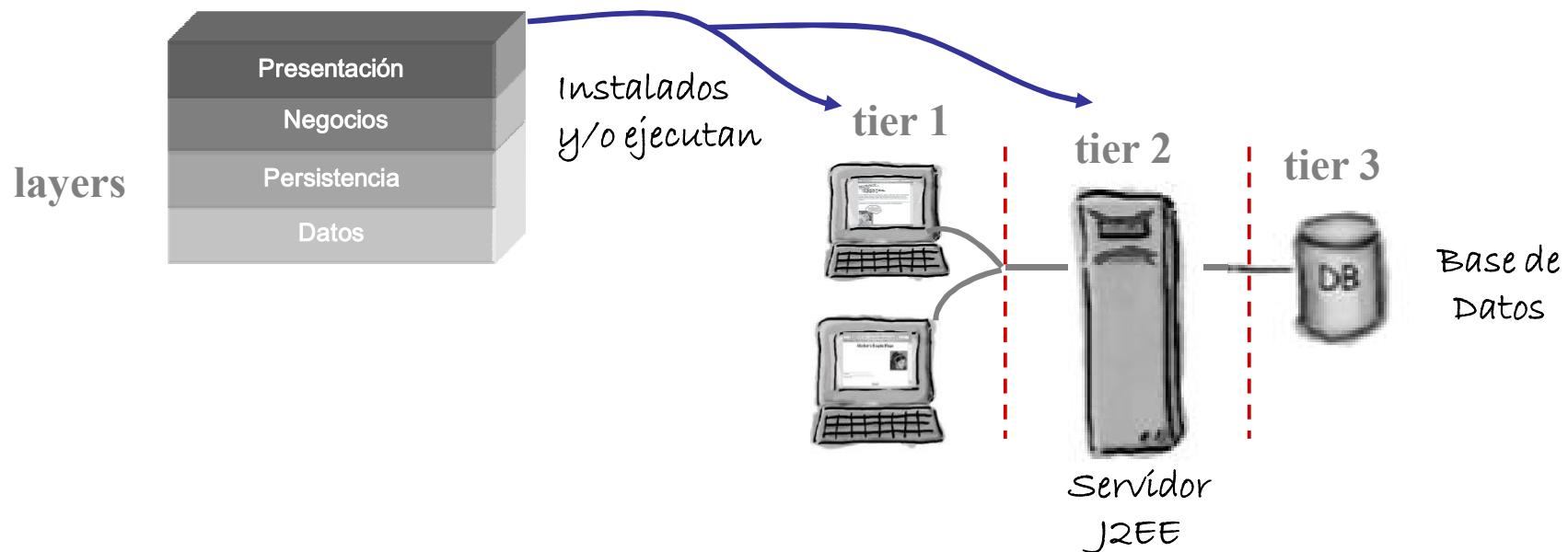


Con una arquitectura layered, los cambios en una de las capas no afecta o afecta poco al resto de las capas.

Arquitectura de una aplicación

Tiers (capas físicas)

Una plataforma *empresarial* es por naturaleza, una plataforma distribuida. Las tareas están distribuidas en distintas computadoras. Cada computadora implementa/ejecuta uno o más **layers**.



Una aplicación JEE se “despliega” en al menos 3 tiers: una máquina cliente, un servidor JEE y un servidor de datos.

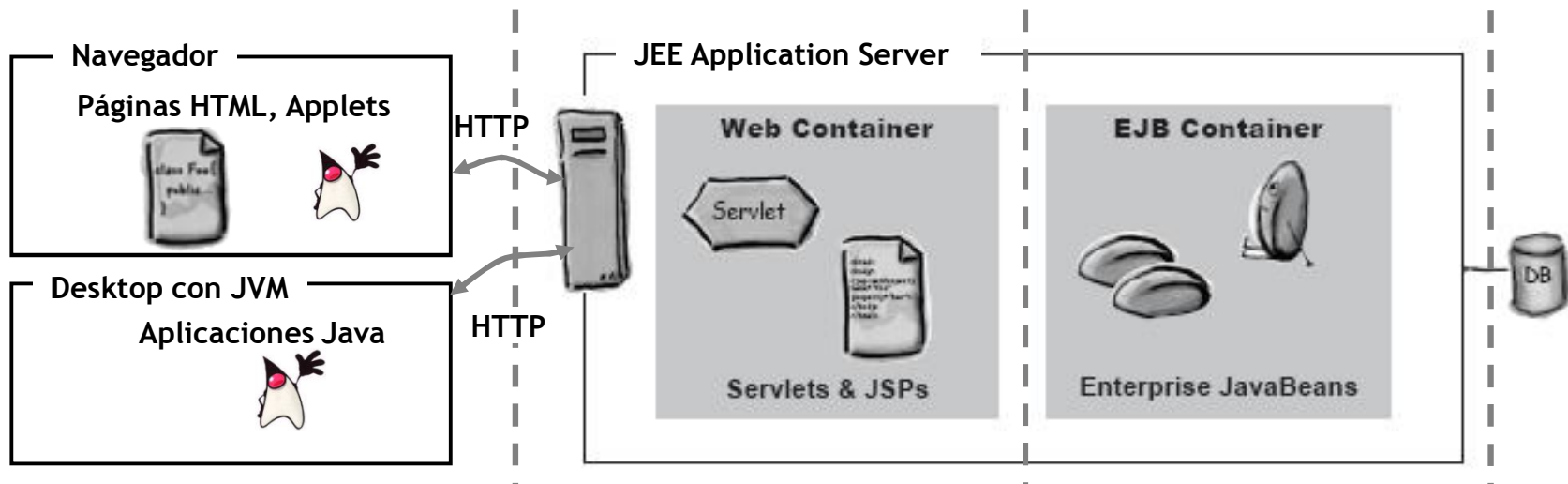
Arquitectura de Java EE

Componentes

La plataforma Java EE usa un modelo de aplicación “multitiered” distribuída. La lógica de la aplicación está dividida en componentes de acuerdo a su función, y las componentes que forman parte de la aplicación java EE son instaladas en varias máquinas dependiendo de la “tier” en el ambiente “multitiered” java EE a la que pertenece.

La especificación Java EE define las siguientes componentes:

- **Componentes Cliente:** Aplicaciones, Java Applets, páginas HTML.
- **Componentes Web:** Servlets (filtros, listeners) , JSPs y JavaServer Faces
- **Componentes Empresariales:** Enterprise JavaBeans



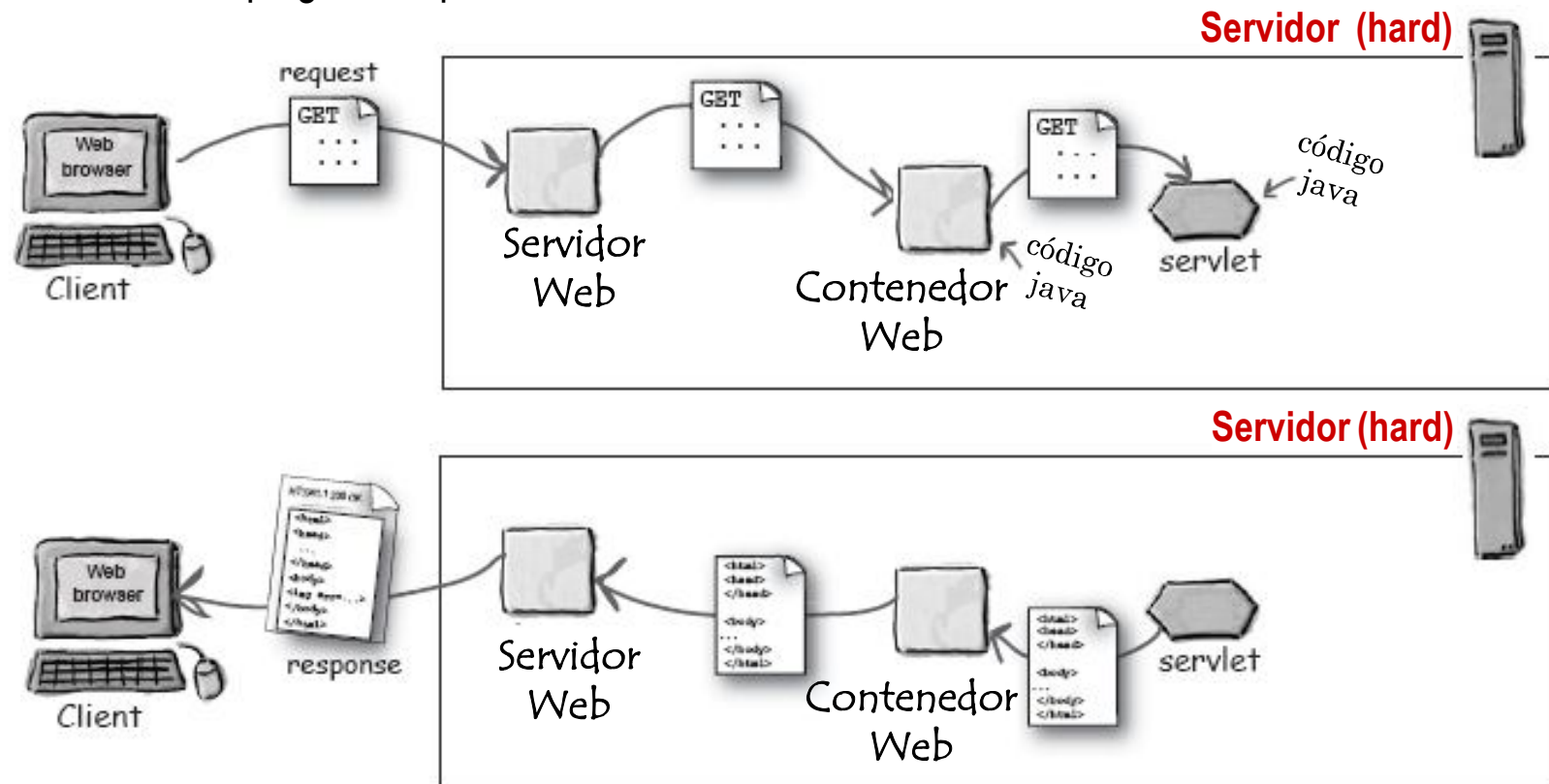
Una aplicación JEE está basada en componentes, donde una componente JEE es una unidad funcional de software, que debe correr dentro de un contenedor y que puede comunicarse con el resto de las componentes.

Arquitectura JEE

Contenedores

Las componentes web JEE no tienen un método `main()` como las aplicaciones de escritorio. Ellas están bajo el control de otra aplicación java, llamada Contenedor.

Cuando el servidor HTTP recibe un requerimiento para una componente web (supongamos un Servlet), el servidor no maneja el requerimiento solo, sino que lo hace con ayuda del contenedor web en donde el servlet fue “desplegado” o puesto a funcionar.



Arquitectura JEE

Contenedores JEE

Servidores certificados que cumplen la especificación J2EE:






	Código Fuente Abierto	Propietarios
Contenedor web (solamente)	Tomcat (Apache) Jetty (Eclipse Foundation) Resin (Caucho Technology)	
Contenedor JEE completo (Application Server)	GlassFish JBoss AS JOnAS Geronimo (Apache) Resin	WebSphere Application Server (IBM) WebLogic (BEA System) iPlanet (SUN & Netscape) OC4J (Oracle) Resin Pro



Arquitectura JEE

Entornos para desarrollo de aplicaciones JEE

Los entornos de desarrollo o IDEs (Integrated Development Environment) para desarrollo de aplicaciones JEE:

IDE	Propietario	
Eclipse	-	
NetBeans 8.1	SUN	
JDeveloper	ORACLE	
IntelliJ IDEA	-	
IBM Rational Application Developer (RAD)	IBM	

Eclipse IDE para EE:

<https://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/neonr>

Arquitectura JEE

Frameworks para desarrollo de aplicaciones JEE

Los siguientes son algunos de los frameworks open source utilizados para facilitar el desarrollo de aplicaciones JEE. Todos necesitan de la plataforma JEE y de un servidor JEE para su funcionamiento

Framework de Código Fuente Abierto	Funcionalidad
Struts 2	Este framework implementa el modelo MVC. Struts fue uno de los framework web MVC más usado en java.
Spring	Es uno de los framework más populares, sus características mas importantes son la inyección de dependencias y programación orientada a aspectos.
Spring MVC	Es el framework web de Spring, está basado en el patrón MVC y ayuda a construir aplicaciones web poco acopladas.
Hibernate	Es un framework para persistencia de datos. Provee mapeo objeto-relacional (ORM)
JavaServer Faces	Este framework facilita el desarrollo de Interfaces de Usuario para aplicaciones JEE. Es el único estándar.

Clientes y servidores

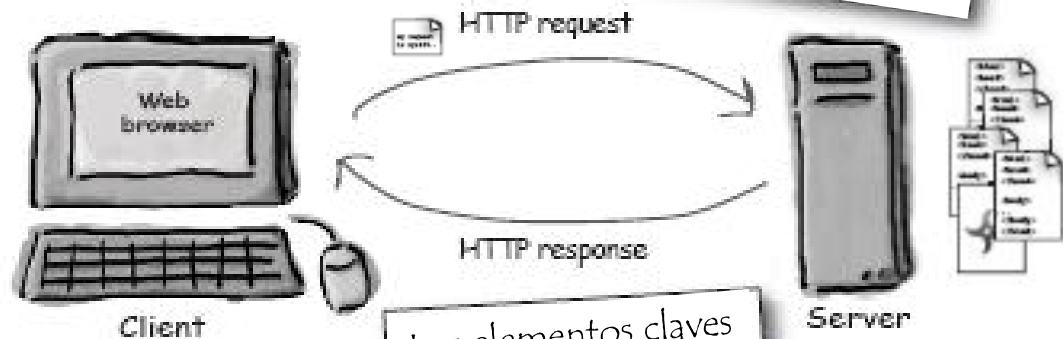
conocen el HTTP y HTML

¿qué es el protocolo HTTP?

La mayoría de las conversaciones que mantienen los navegadores con los servidores lo hacen usando el protocolo HTTP - HyperText Transfer Protocol-, el cual permite conversaciones del tipo requerimiento - respuesta. El cliente envía un requerimiento HTTP y el servidor responde con una respuesta HTTP.

Los elementos claves de un request:

- El método HTTP
- La página a acceder (URL)
- Los parámetros del formulario



¿ qué es HTML?

Cuando un servidor responde a un requerimiento, en general envía al navegador un conjunto de instrucciones escritas en HTML -HyperText Markup Language-. El HTML le dice al navegador **cómo** mostrar el contenido al usuario.

Los elementos claves de un response:

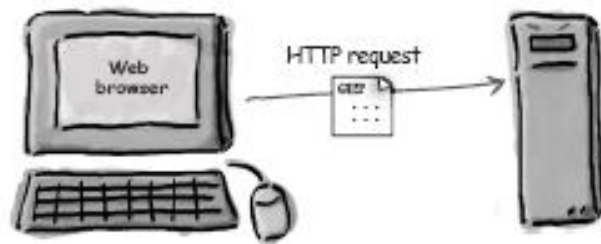
- El código de estado.
- El Content-type (text-picture-HTML etc.)
- El contenido

En términos generales los códigos de estados 5xx son errores internos, los 4xx son que no encontró lo que se pide, los 3xx son redirecciones y los 2xx ok

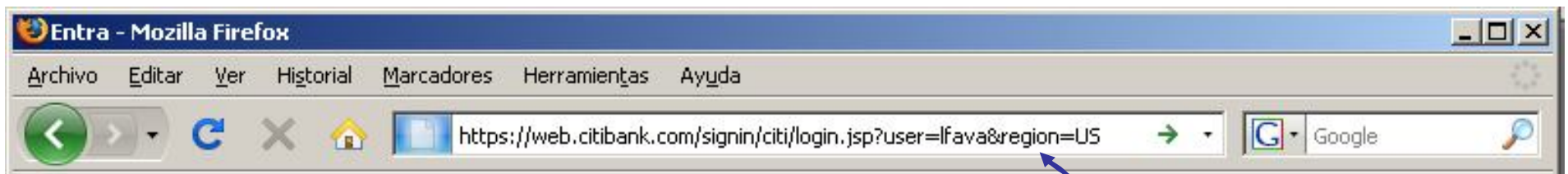
HTTP Response

¿qué tiene un requerimiento HTTP?

Un requerimiento HTTP tiene un nombre de método (no son métodos java). HTTP tiene varios métodos pero básicamente usaremos GET/POST.



La función de los métodos GET/POST es pedirle al servidor un recurso, que puede ser una página HTML, un JPEG, un PDF, etc. Ambos pueden enviar datos en el pedido: con el GET los datos aparecen en la URL y tiene limitaciones –cada servidor soporta una cantidad limitada de caracteres-, con POST no hay límite porque los datos viajan en el cuerpo del HTTP.



Cuando se tipea la URL en la barra del navegador (al igual que al presionar un link) se genera un método GET

HTTP Response

Anatomía de un requerimiento HTTP GET



HTTP Response

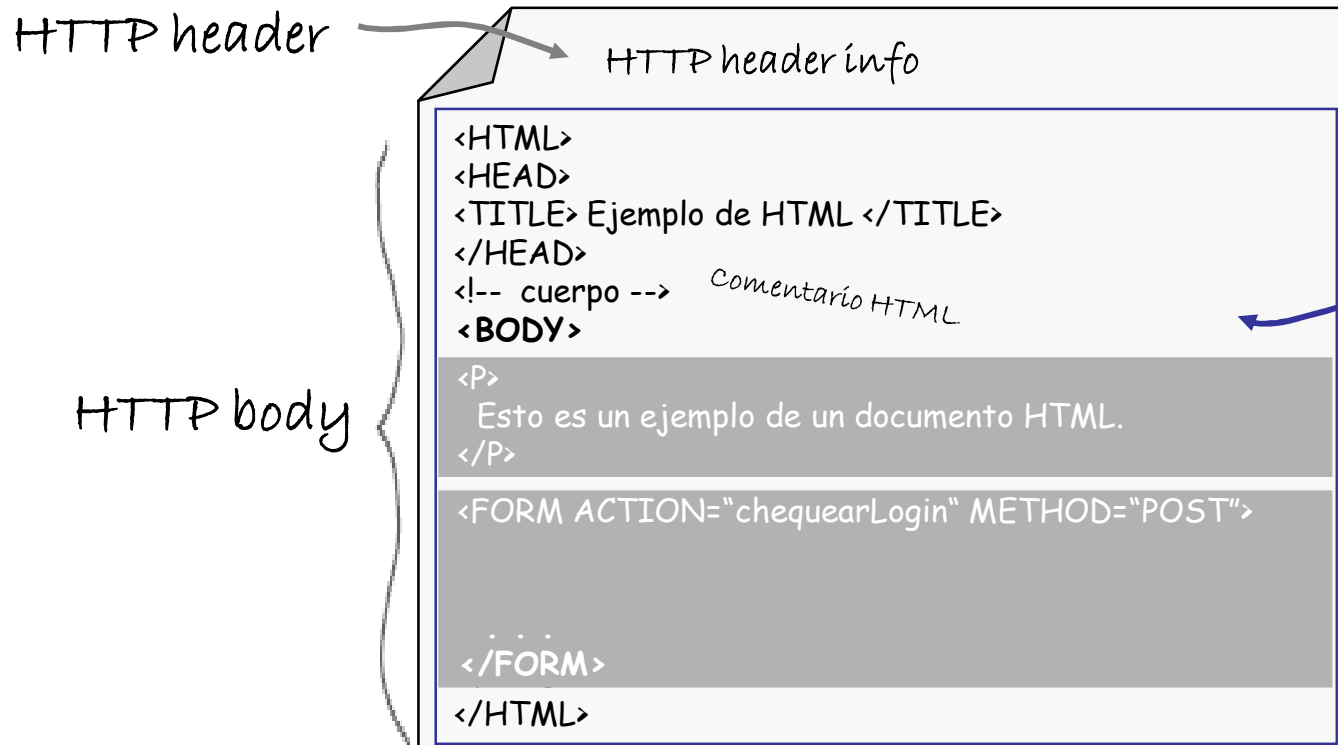
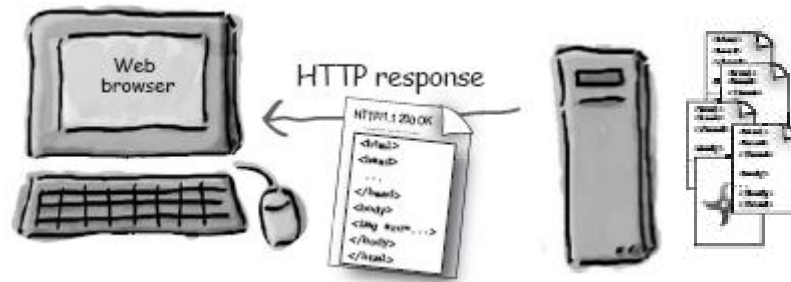
Anatomía de un requerimiento HTTP POST



HTTP Response

¿qué tiene una respuesta HTTP?

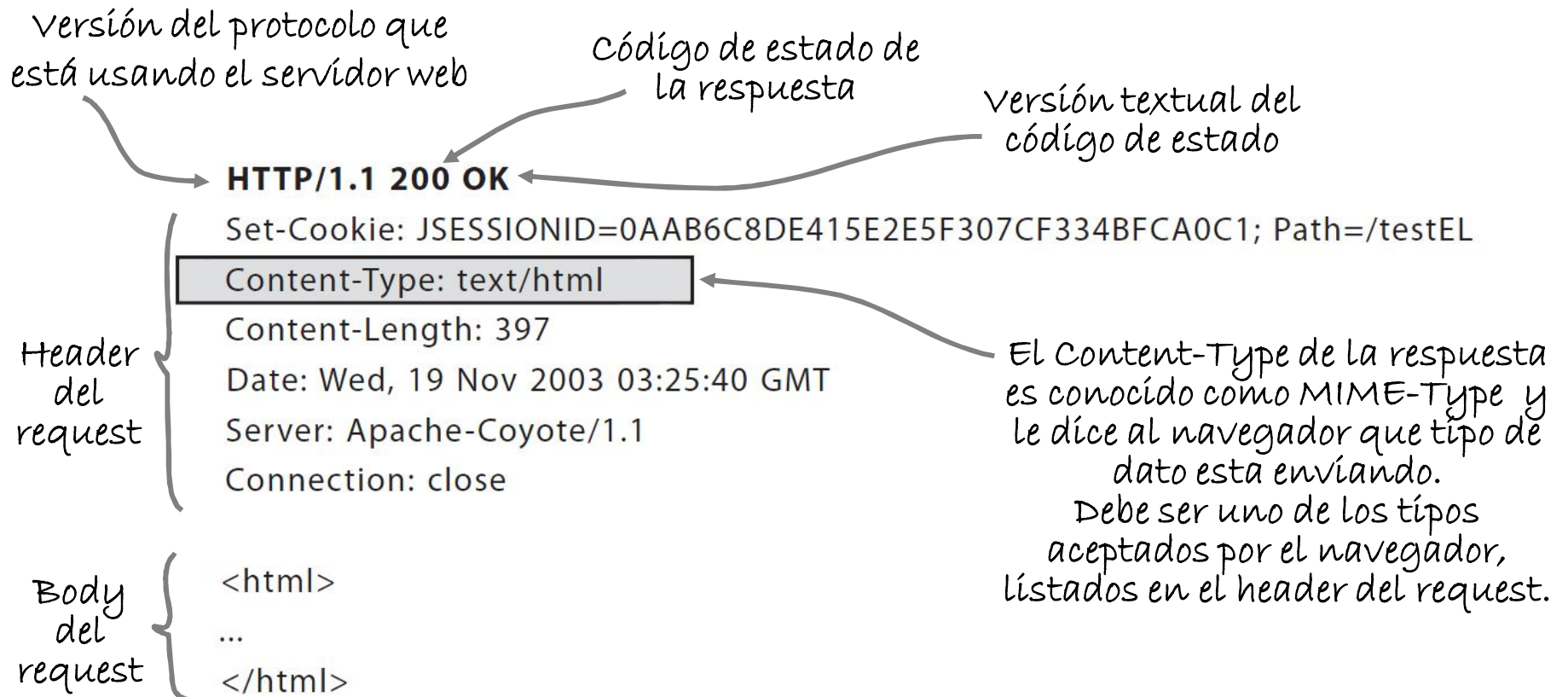
Una respuesta HTTP *puede* contener HTML. El HTTP le agrega un **header** con información arriba del contenido en la respuesta. Un navegador usa el **header** para procesar la respuesta y poder mostrarla.



Estructura de un documento HTML. Es parte de la respuesta HTTP

HTTP Response

Anatomía de un response HTTP



HTTP Request-HTTP Response

Todo junto

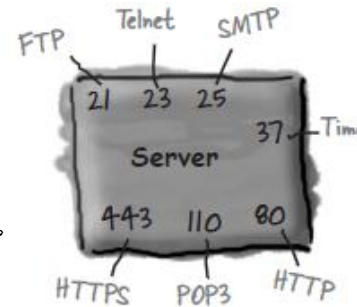


URL: Uniform Resource Locators

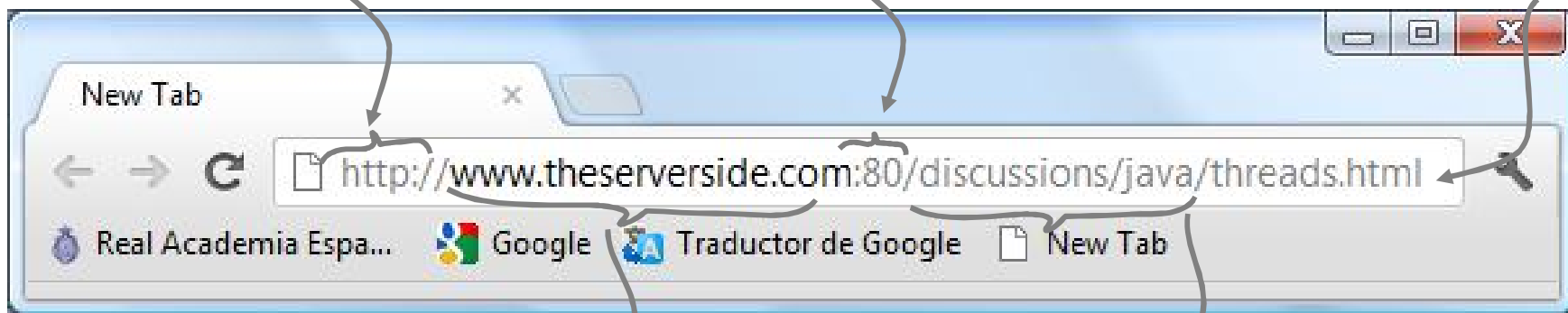
Cada recurso en la web tiene su propia y única dirección en el formato URL.

Protocolo: le indica al servidor que protocolo de comunicación será utilizado

Port (puerto): esta parte de la url es opcional. Cada servidor soporta muchos puertos. El puerto por defecto para servidores web, es 80.



Recurso: el nombre del recurso que se está pidiendo. Podría ser un HTML, una imagen, un servlet, etc. Si no se especifica nada, el servidor buscará un index.html



Servidor: nombre único del servidor que se está buscando. Este nombre mapea con una única dirección IP y se podría especificar la dirección IP en vez del nombre.

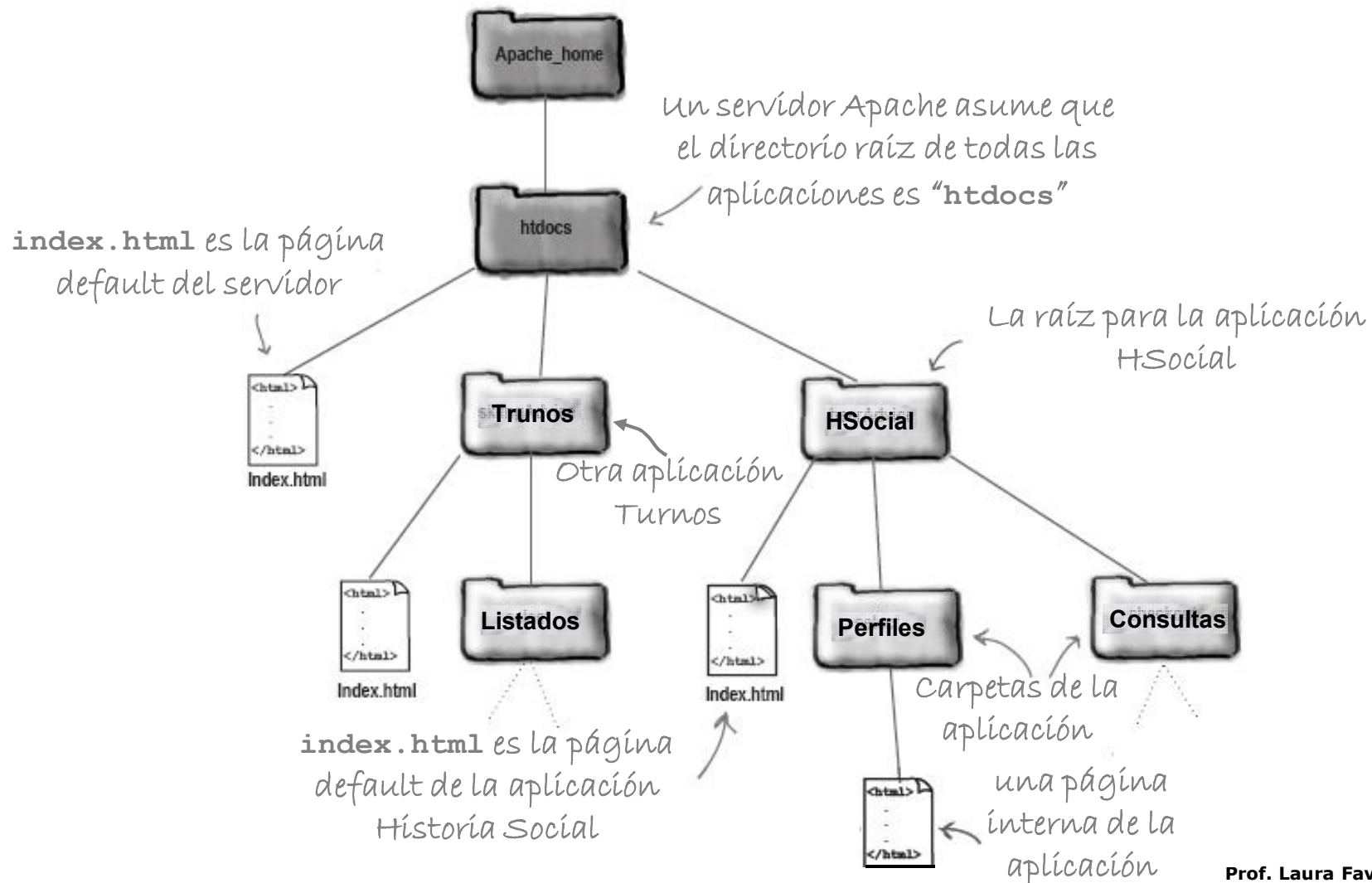
Path: camino de la ubicación del recurso buscado en el servidor.

Query String: si es un requerimiento GET la información extra (parámetros) son agregados al final de esta URL, comenzando con "?" y cada uno de los parámetros (nombre=valor) separado por "&".

Sitio Web

Estructura de directorio para un sitio simple en Apache

Apache es un servidor web HTTP *open source* muy popular. La imagen representa la estructura de directorios de un servidor Apache que contiene dos aplicaciones Turnos y HSocial.



HTML (HyperText Markup Language)

Evolución

HTML -“HyperText Markup Language”- especifica el formato de las páginas web, separando el contenido de las páginas, de su formato de presentación. Fue creado en los laboratorios CERN por Tim Berners-Lee.

Define un conjunto de símbolos (etiquetas o tags) que especifican la estructura lógica de un documento y de todos sus componentes.

Es independiente de la plataforma y su código es interpretado por los clientes web.

En el año 2004 comenzó a desarrollarse HTML5, actualmente el último estándar.

Los navegadores actuales soportan muchas de las características de HTML 5, los mejores posicionados para las versiones desktop son Google Chrome 36, Opera 22, Mozilla Firefox 30, para dispositivos móviles Tizen 2.2, BlackBerry 10.2, Chrome 35, etc.

La lista completa de navegadores y el soporte que brindan para HTML 5 puede consultarse en:
<http://html5test.com/>

Una página web tiene una **estructura** dada por los tags HTML y una **visualización** determina por las hojas de estilo.

HTML (HyperText Markup Language)

Estructura de un documento HTML

HTML es un lenguaje que utiliza etiquetas (tags) y texto para marcar y definir la estructura de un documento HTML. Los tags consisten de un nombre encerrado entre <> para la apertura y </> para el cierre. Por ejemplo:

```
<H1> un texto </H1>
```

A los tags con el contenido en el medio se lo llama elemento. En este caso podemos decir elemento <H1>

Los tags pueden contener atributos. Los atributos permiten especificar información adicional a un elemento. Por ejemplo:

```
<a href="top10.html"> Great Movies </a>
```

nombre del
atributo

valor del atributo. Siempre
entre comillas dobles

```

```

Hay algunos tags que excepcionalmente no
tienen tag de cierre

HTML (HyperText Markup Language)

Estructura de un documento HTML / Formularios

HTML define diferentes componentes, tales como encabezados, títulos, cuerpo, **formulario**, tablas, hipervínculos, imágenes, etc. Un **formulario** es un conjunto de campos que permiten la interacción entre el usuario y el servidor HTTP. Se define con los tags <FORM> ... </FORM> y entre ellos se encuentran las definiciones de los campos del formulario, que viajarán como parámetros hacia el servidor.

Especifica la URI (Universal Resource Identifier) que atenderá el requerimiento

Especifica la forma en que se transferirán los datos, en general GET o POST

```
<html>
<body>
<FORM ACTION="chequearLogin" METHOD="POST">
  Nombre:<input type="text" name=nombre><br>
  Contraseña:<input type="text" name=contra><br>
    <input type="submit" name=b1 value="Enviar">
</FORM>
</body>
</html>
```

Cuando el navegador "submite" un formulario, crea un parámetro en el requerimiento para cada campo en el formulario (FORM)

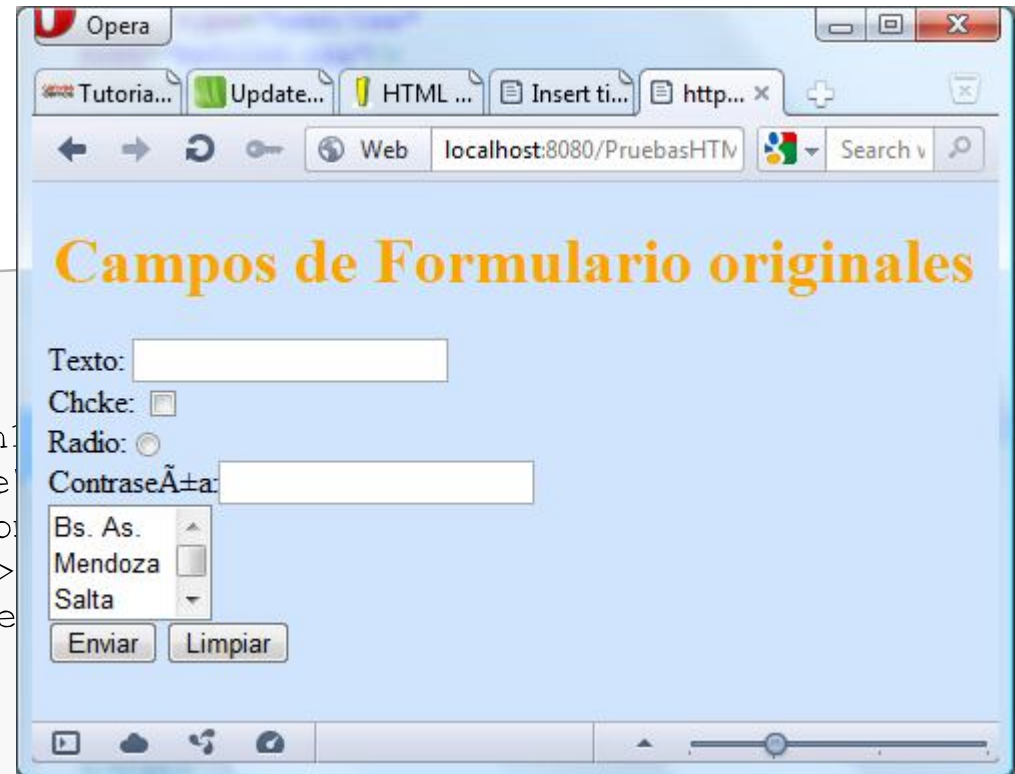
HTML (HyperText Markup Language)

Campos de formulario originales

Los campos "input" originales que están disponibles desde las primeras versiones del HTML son:

- **text** : campos de búsqueda
- **radio**: campos cuyo valor es una URL
- **checkbox** : direcciones de correo
- **password**: fechas
- **reset**: números
- **select**: un color hexadecimal

```
<!DOCTYPE html>
<html><body>
. . .
<h1>Campos de Formulario originales</h1>
Texto: <INPUT type="text" name="nombre">
Chcke: <INPUT type="checkbox" name="correo">
Radio: <INPUT type="radio" name="sexo">
Contraseña: <INPUT type="password" name="pass">
<SELECT size="3" name="provSel">
  <OPTION value="BA"> Bs. As. </OPTION>
  <OPTION value="MZ"> Mendoza </OPTION>
  <OPTION value="CB"> Salta </OPTION>
  <OPTION value="MZ"> Misiones </OPTION>
</SELECT><BR>
<INPUT type="submit" value="Enviar">
<INPUT type="reset" value="Limpiar">
</body></html>
```



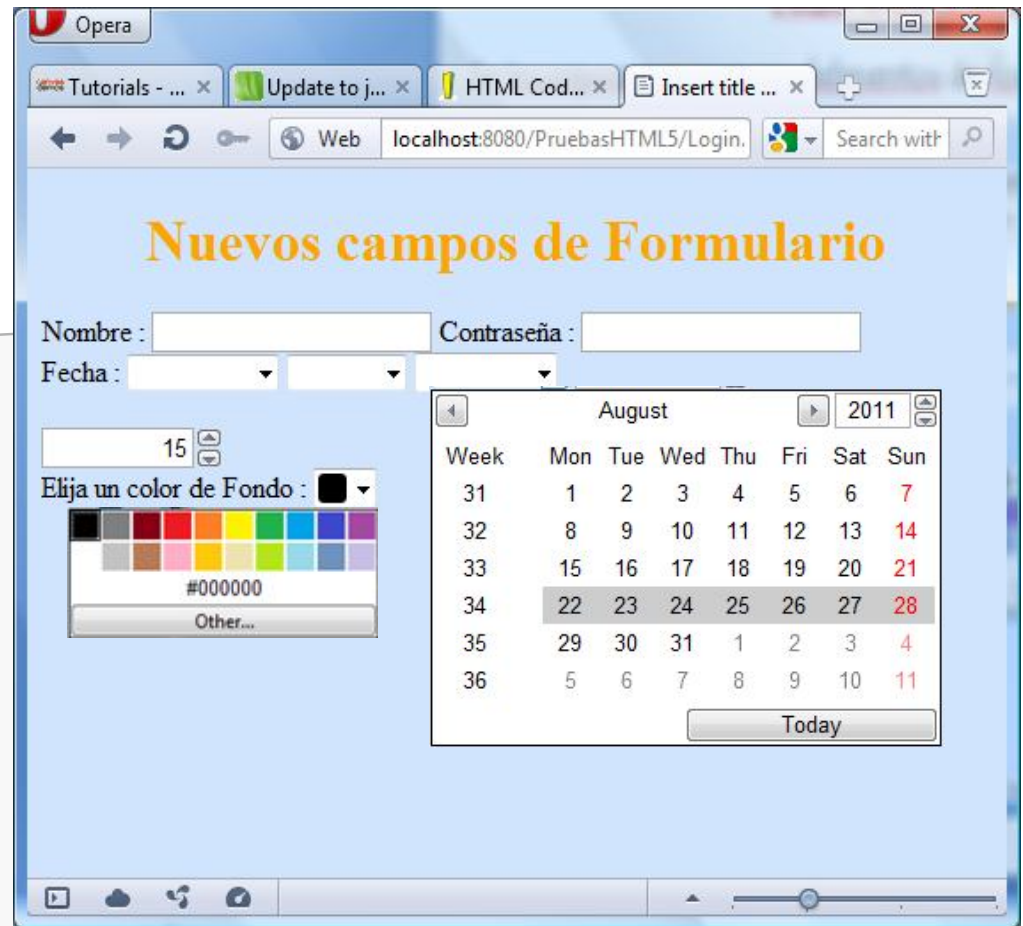
HTML (HyperText Markup Language)

Campos de Formulario del nuevo HTML 5

Algunos nuevos tipos de `<input>` son:

- **tel**: números telefónicos
- **search**: campos de búsqueda
- **url**: campos cuyo valor es una URL
- **email**: direcciones de correo
- **date**, **month**, **week**: fechas
- **number**: números
- **color**: colores en hexadecimal

```
<!DOCTYPE html>
<html><body>
. . .
<h1>Nuevos campos de Formulario</h1>
Fecha:
  <input type="date" name="dia"/>
  <input type="month" name="mes"/>
  <input type="week" name="semana"/>
<br>
  <input type="number" min="0" max="15"
    step="2" Value="15">
Elija un color de Fondo :
  <input type="color" name="colores"/>
</body>
</html>
```



Ejemplos en: http://www.w3schools.com/html/html_forms.asp

HTML (HyperText Markup Language)

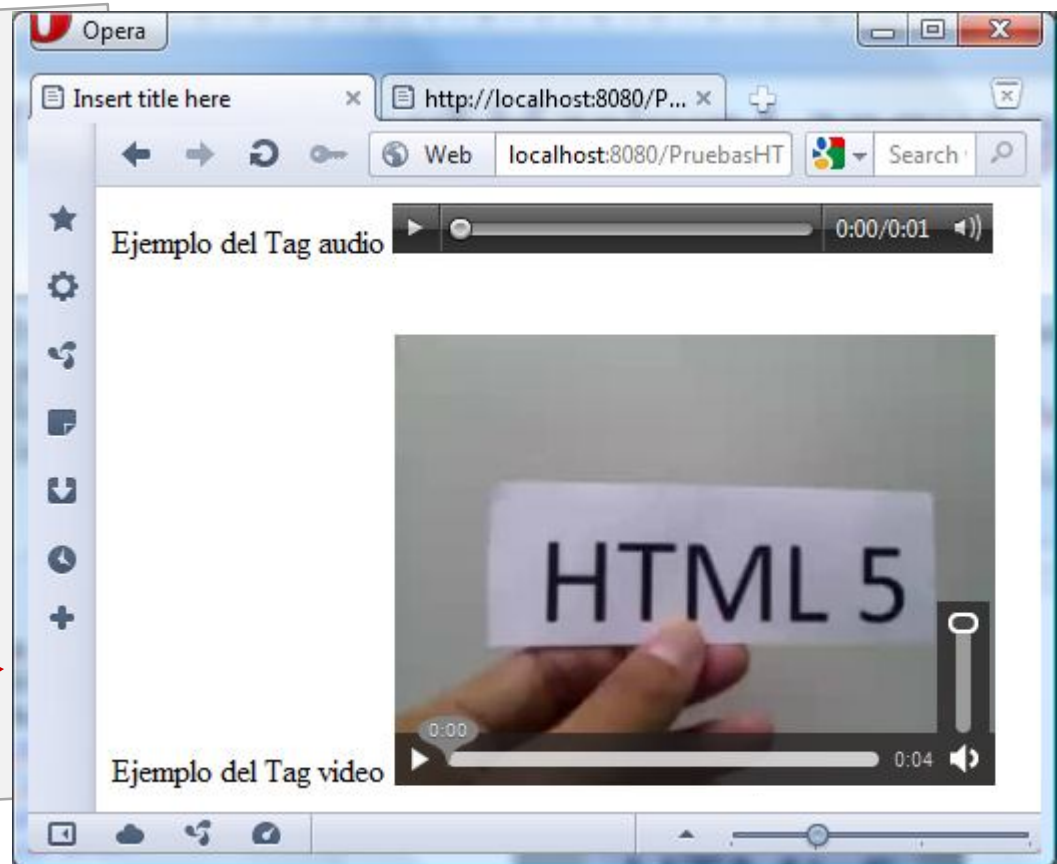
Otras componentes del nuevo HTML 5

También se incorporaron dos elementos para **contenido multimediales** :

- **<audio>**: Distintos tipos de sonidos, música, transmisión de audio. Los formatos propuestos son: ogg, mp3, wav
- **<video>**: Este tag permite integrar el video con un tag del HTML. Aún no hay una postura definitiva sobre los formatos aceptados.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>

Ejemplo del Tag audio
<audio src="elegirCuento.wav"
controls></audio>
<BR>
Ejemplo del Tag video
<video src="html5.mp4" controls
width="180" height="260"></video>
</body>
</html>
```

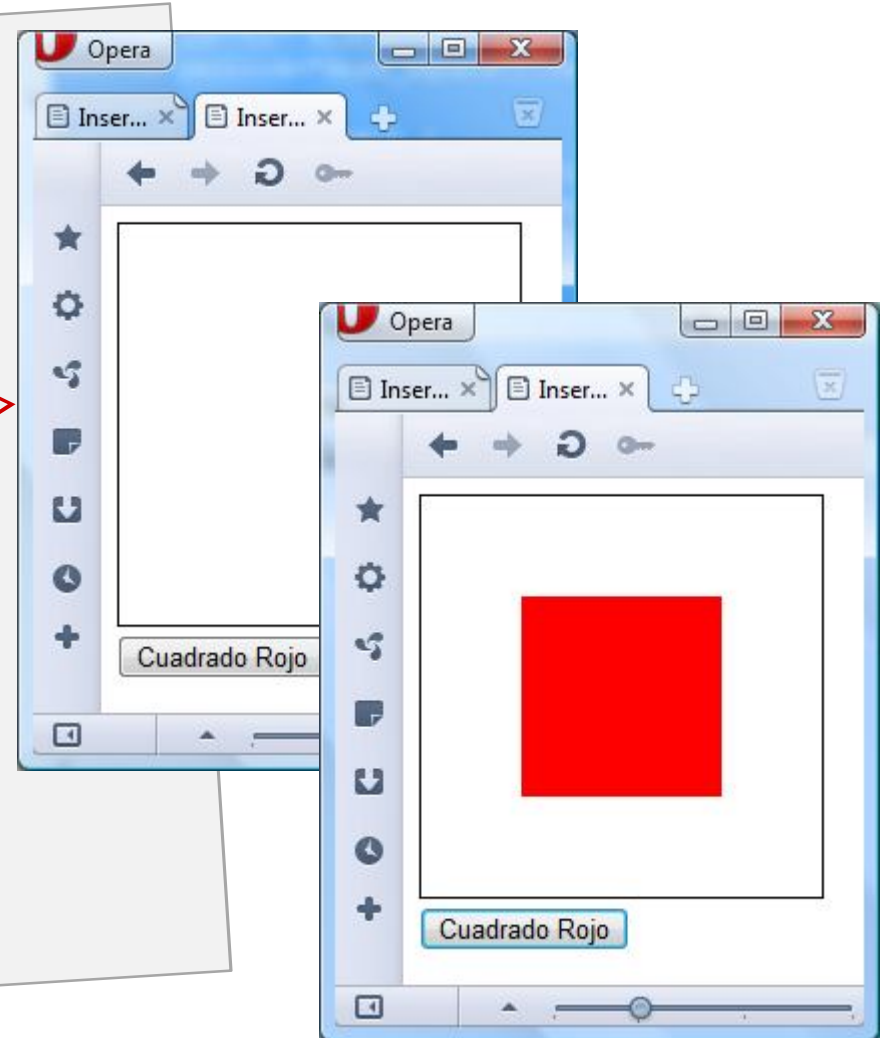


HTML (HyperText Markup Language)

Otras componentes del nuevo HTML 5

Una de las incorporaciones más interesantes en HTML5 es la componente **Canvas**. Un canvas es un espacio en blanco y javascript, cumpliendo la función del lápiz y el pincel para dibujar y animar un gráfico.

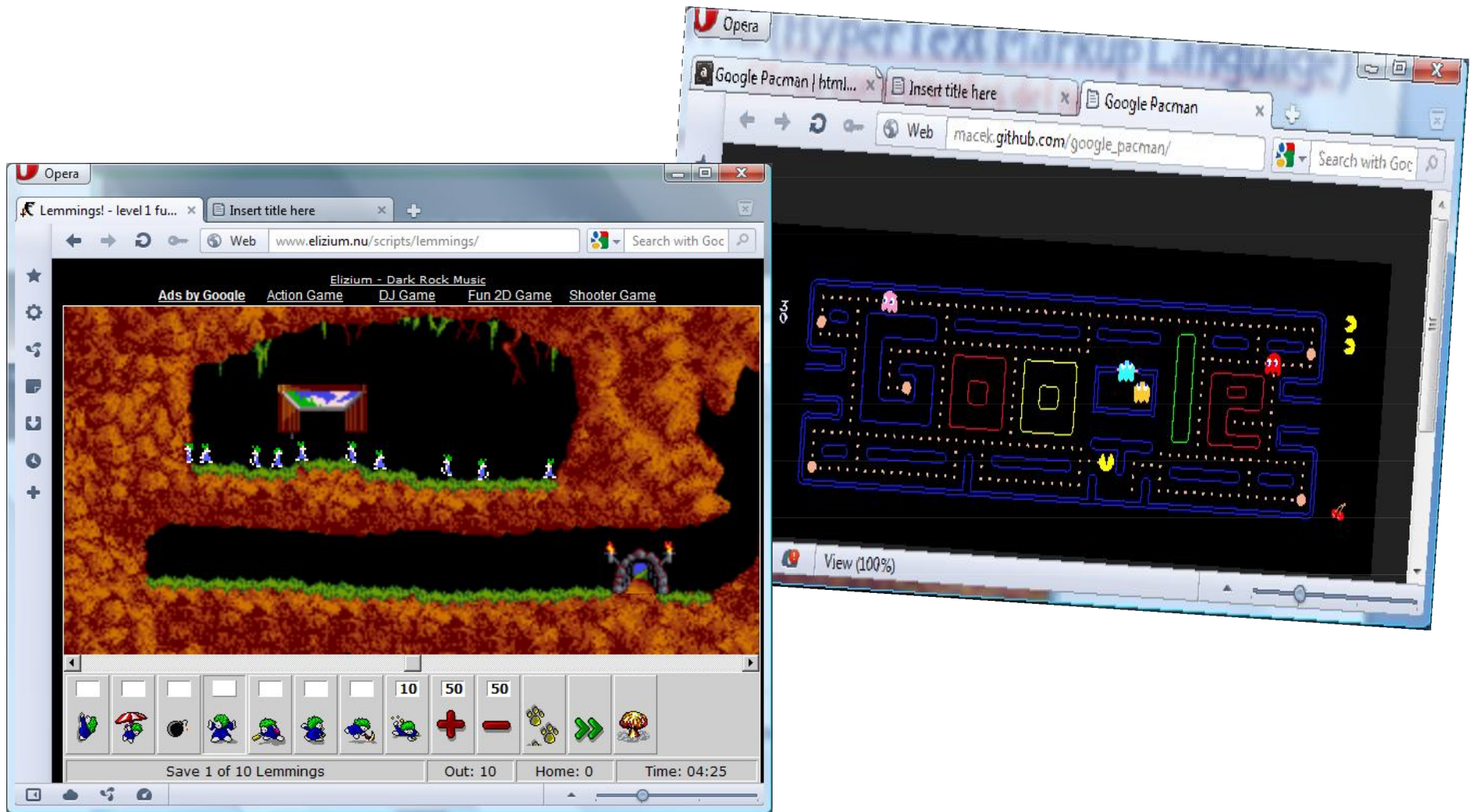
```
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<canvas id="c1" width="200" height="200"
style="border:solid 1px #000000;"></canvas>
<button onclick="draw_square();return
true;">Cuadrado Rojo</button>
<script>
function draw_square() {
  var c1 = document.getElementById("c1");
  var c1_context = c1.getContext("2d");
  c1_context.fillStyle = "#f00";
  c1_context.fillRect(50, 50, 100, 100);
}
</script>
</body>
</html>
```



HTML (HyperText Markup Language)

Otras componentes del nuevo HTML 5

Ejemplos de canvas con programación más avanzada.



HTML (HyperText Markup Language)

Cascading Style Sheets(CSS)

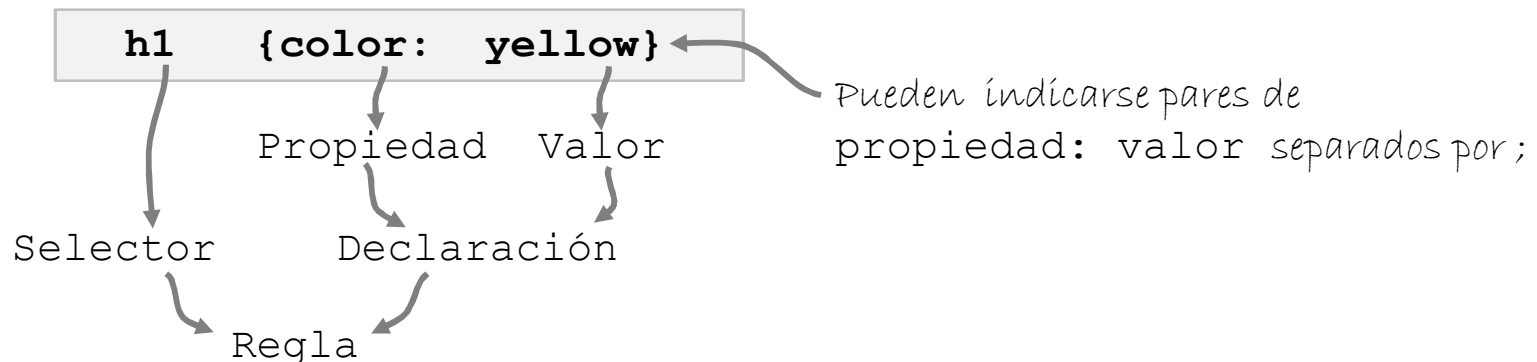
Las hojas de estilo describen o determinan la visualización de los elementos de una página web. El estándar más utilizado para la confección de los archivos de estilos es el de estilo en cascada o CSS (Cascading Style Sheets).

El objetivo de trabajar con hojas de estilos es mantener el mismo aspecto en todas las páginas pertenecientes a un sitio o aplicación web.

Una hoja de estilo consiste de un **conjunto de reglas**, que definen un estilo para cada elemento o grupos de elementos HTML.

Una regla de estilo tiene dos partes:

- Un selector, que identifica el elemento o grupo al que el estilo se aplicará.
- Una declaración de propiedades que se aplicarán al selector.



HTML (HyperText Markup Language)

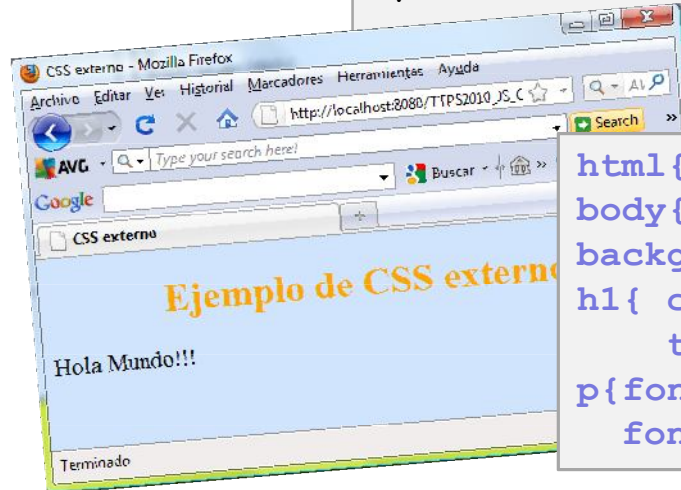
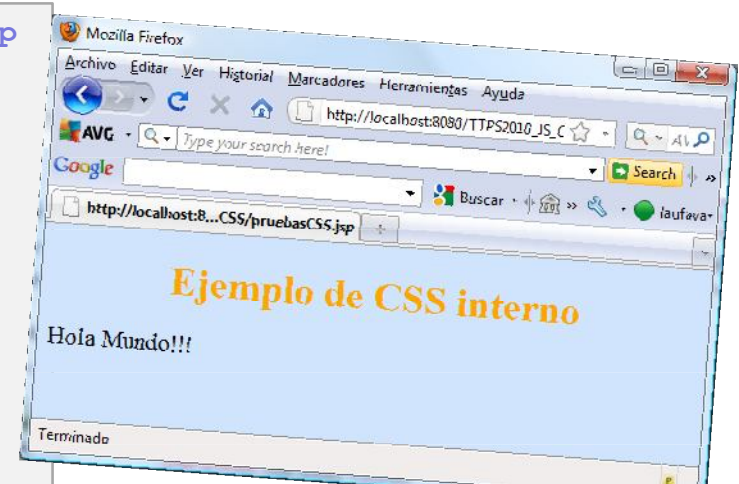
Cascading Style Sheets(CSS)

La definición de la visualización puede estar adentro de la página web o puede ser importada desde una fuente externa.

Los tags

`<style>`, `</style>`
permiten definir un
estilo embebido

```
<html><head> pruebaCSS.jsp
<style type="text/css">
body { background-color:#d0e4fe;}
h1{ color:orange;
    text-align:center;}
p{ font-family:"Times New Roman";
   font-size:20px;}
</style>
</head><body>
<h1>Ejemplo de CSS interno</h1>
<p>Hola Mundo!!!</p>
</body>
</html>
```



```
<html><head>
<link rel="stylesheet"
      type="text/css"
      href="unEstilo.css"/>
<title>CSS externo</title>
</head>
<body>
<h1>Ejemplo de CSS externo</h1>
<p>Hola Mundo!!!</p>
</body></html>
```

pruebaCSS.jsp

CSS externo

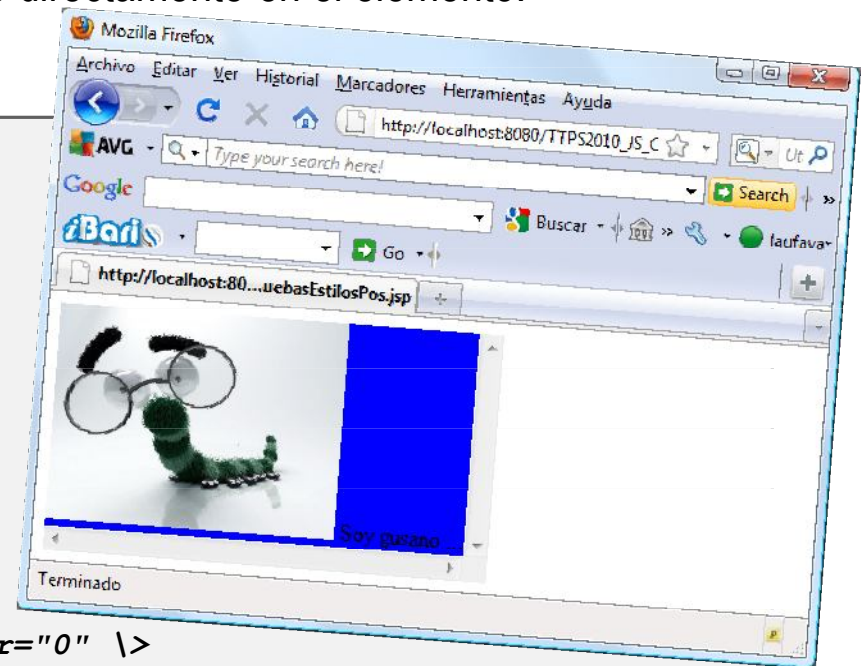
unEstilo.css

HTML (HyperText Markup Language)

Cascading Style Sheets(CSS)

Es muy común agrupar un conjunto de elementos o tags HTML para aplicarles a todos un mismo estilos. Para ello existe el tag <div> que define una sección en un documento HTML. También acá se puede ver que se puede definir el estilo directamente en el elemento.

```
<html>
<head>
<script type="text/javascript">
  function imagen() {
    document.getElementById('imagen').src =
      "imagenes/minibuscador.jpg";
  }
</script>
</head>
<body onload="imagen()">
<div style="position:absolute; top:100; left:100;
  text-align:center; background-color: blue;
  overflow:scroll">
  <img id='imagen' height="150" width="200" border="0" \>
    Soy gusano ...
</div>
</body>
</html>
```



Referencias

Tecnologías JEE (Java Enterprise Edition) 6

- <http://www.oracle.com/technetwork/java/javaee/tech/index.html>

El Lenguaje HTML

- <http://www.w3c.org/html>
- <http://www.htmlquick.com/es/reference.html>
- <http://www.w3.org/TR/html401/struct/tables.html#h-11.2.3>

HTML5 y CSS

- <http://www.w3.org/html/wg/html5/>
- <http://www.w3schools.com/css/default.asp>
- *HTML5 Designing Rich Internet Applications*, Mathew David

El Consorcio de la Web (W3C) fue creado y es presidido por Tim Berners-Lee, con el objetivo de guiar el desarrollo de protocolos y pautas estándares para la web.

- <http://www.w3c.es/Consortio/>
- <http://www.w3.org/History/1989/proposal.html>