

Tema 2: Tecnologías para la Programación Web

Tecnologías para la Programación Web
Desarrollo Web en Entorno Servidor



Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

Tecnologías para la Programación Web
REPASO

- **URI**
 - Una **URI** (*Uniform Resource Identifier*) no es una **URL** (*Uniform Resource Locator*):
 - Las URL son un subconjunto de URI.
 - Todas las URL son URI pero no a la inversa.
 - Las URL, es un tipo de URI, que **describe la localización** de un documento específico.
 - Una URL no define el tipo de contenido que se encontrará (textos, imágenes, películas, etc.), sólo dice dónde y cómo encontrarlos.
- **Ejemplos de URI:**
 - mailto:joe@example.org
 - tel:+34-954-100-100
 - smsto:+15105550101?body=hola%20DAM
 - ftp://ftp.is.co.za/rfc/rfc1808.txt
 - Es un **tipo de URI** que describe la localización de un documento (**URL**)

- **URL**

- Estructura de una **URL** (*Uniform Resource Locator*):
 - El **protocolo** que se usa para comunicar.
 - La dirección del **servidor** (anfitrión) con el que se comunica.
 - El **puerto** de red en el servidor para conectarse.
 - La **ruta al recurso** en el servidor (p.e. nombre de archivo).

<http://es.wikipedia.org:80/wiki/Special:Search?search=tren&go=Go>

Donde:

- *http* es el protocolo,
- *es.wikipedia.org* es el anfitrión,
- *80* es el número de puerto de red en el servidor (siendo 80 el valor por omisión para el protocolo HTTP, esta porción puede ser omitida por completo),
- */wiki/Special:Search* es la ruta de recurso,
- *?search=tren&go=Go* es la cadena de búsqueda; esta parte es opcional.

JSON:

- Lenguaje de marcas descriptivo y ligero, subconjunto del lenguaje YAML.
- Más fácil, cómodo y rápido en su notación.
- **Características principales del lenguaje:**
 - Los elementos comienzan con llave de apertura “{” y terminan con llave de cierre “}”
 - Los subelementos de cada elemento comienzan con corchete de apertura “[” y terminan con corchete de cierre “]” → **lista ordenada**
 - Cada elemento del bloque, están separados por comas.
 - Pueden tratarse atributos característicos.
- JSON se apoya en dos tipos de **estructuras de datos**:
 - Una colección de pares nombre/valor.
 - En algunos lenguajes de programación se le denomina: objeto, registro o tabla hash.
 - Una lista ordenada de valores
 - En algunos lenguajes de programación se denominan: array, vector, lista o diccionario.

REPASO

Formatos de intercambio y representación de datos

```
{
  "empleados": {
    "empleado": [
      {
        "nombre": "Juan Menéndez",
        "categoria": "administrativo"
      },
      {
        "nombre": "Luisa Marín",
        "categoria": "jefe"
      }
    ]
  }
}
```

Lista ordenada de valores

Pares nombre-valor

EJEMPLO EQUIVALENTE EN XML

```
<?xml version="1.0" encoding="UTF-8"?>
<empleados>
  <empleado>
    <nombre>Juan Menéndez</nombre>
    <categoria>administrativo</categoria>
  </empleado>
  <empleado>
    <nombre>Luisa Marín</nombre>
    <categoria>jefe</categoria>
  </empleado>
</empleados>
```

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

REPASO

Formatos de intercambio y representación de datos



The screenshot shows a web-based converter tool. On the left, under 'XML', there is an input field containing the XML code provided earlier. On the right, under 'JSON', there is an output field displaying the equivalent JSON code. Between the two fields are two small buttons: one pointing from XML to JSON, and another pointing from JSON back to XML.

```
XML
<?xml version="1.0" encoding="UTF-8"?>
<empleados>
  <empleado>
    <nombre>Juan Menéndez</nombre>
    <categoria>administrativo</categoria>
  </empleado>
  <empleado>
    <nombre>Luisa Marín</nombre>
    <categoria>jefe</categoria>
  </empleado>
</empleados>

JSON
{
  "empleados": [
    "empleado": [
      {
        "nombre": "Juan Menéndez",
        "categoria": "administrativo"
      },
      {
        "nombre": "Luisa Marín",
        "categoria": "jefe"
      }
    ]
  ]
}
```

XML/JSON Converter: <http://www.utilities-online.info/xmltojson>

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

REPASO

Formatos de intercambio y representación de datos

XML/JSON Converter: <http://json.online-toolz.com/tools/xml-json-converter.php>

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

REPASO

Formatos de intercambio y representación de datos

```
<paises>
  <pais>
    <nombre>Brasil</nombre>
    <continente>América</continente>
    <animales>
      <animal>Pitón</animal>
      <animal>Caimán</animal>
    </animales>
  </pais>
  <pais>
    <nombre>Nigeria</nombre>
    <continente>África</continente>
    <animales>
      <animal>Tigre</animal>
    </animales>
  </pais>
</paises>
```

XML

Nº de caracteres de información: **42**
 Nº de caracteres totales (sin espacios): **262**
 Porcentaje de información con respecto al total:

16,03%

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

REPASO

Formatos de intercambio y representación de datos

```
{
  "paises": [
    {
      "pais": [
        {
          "nombre": "Brasil",
          "continente": "América",
          "animales": [
            "animal": [
              "Pitón",
              "Caimán"
            ]
          ]
        },
        {
          "nombre": "Nigeria",
          "continente": "África",
          "animales": { "animal": "Tigre" }
        }
      ]
    }
  ]
}
```

JSON

Nº de caracteres de información: **42**
 Nº de caracteres totales (sin espacios): **178**
 Porcentaje de información con respecto al total:

23,59%

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

REPASO

Formatos de intercambio y representación de datos

```
---
paises:
  pais:
  -
    nombre: "Brasil"
    continente: "América"
    animales:
      animal:
        - "Pitón"
        - "Caimán"
  -
    nombre: "Nigeria"
    continente: "África"
    animales:
      animal: "Tigre"
```

YAML

Nº de caracteres de información: **42**
 Nº de caracteres totales (sin espacios): **143**
 Porcentaje de información con respecto al total:

29,37%

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

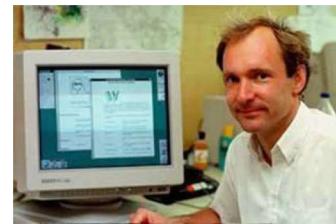
Tecnologías para la Programación Web

ORGANIZACIONES DESARROLLADORAS

- **Organización Internacional para la Estandarización (ISO) encargado de:**
 - El desarrollo de normas internacionales de fabricación, comercio y comunicación.
 - Buscar la **estandarización de normas** de productos y seguridad para las empresas u organizaciones a nivel internacional.
- **Consorcio World Wide Web (W3C):**
 - Director: **Tim Berners-Lee**
 - Creador del URL, el protocolo HTTP y HTML.
 - **URL → protocolo://servidor:puerto/recurso**
 - Consorcio de empresas de la World Wide Web liderado por Apple, Google, HP, IBM, Fujitsu, universidades internacionales, etc.)
 - Dirigir el crecimiento y organización de la web.



International
Organization for
Standardization



Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

Tecnologías para la Programación Web

UTILIZACIÓN DE LENGUAJES DE MARCAS EN ENTORNOS WEB

- Una página web es un documento electrónico adaptado para la World Wide Web que, normalmente, forma parte de la estructura de ficheros de un **sitio web**. Está compuesta, principalmente de:
 - Información (texto, imágenes y otros recursos multimedia)
 - **Contenido estático o dinámico**
 - **Enlaces** (hiperenlaces o hipervínculos)
 - Asocia los **estilos** a los datos para especificar su visualización
 - Atributos del lenguaje para mejorar la **accesibilidad** de la página.
 - **Embeber aplicaciones para hacerla interactiva** con el usuario (a través de la integración de módulos de librerías multimedia en la página) para conseguir:
 - Reproducción de vídeo y audio de múltiples formatos:
 - .webm, .ogg, .mp4 (H.264)
 - Acceder a recursos: webcam, micrófono, etc.
 - Trabajar **offline**.
 - Geolocalización
 - Optimizar las subidas de ficheros
 - *Drag and drop*
 - Generación dinámica de elementos gráficos
 - Complejos elementos de interacción con formularios



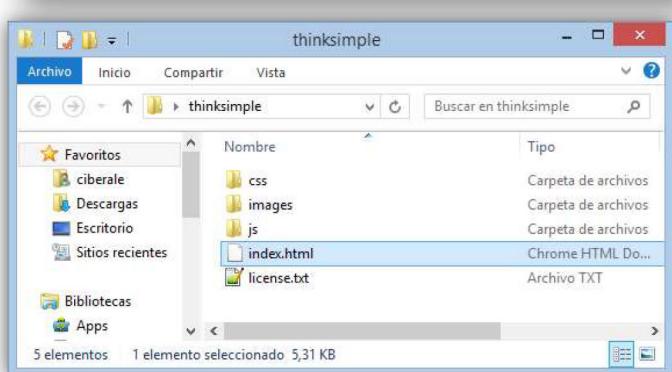
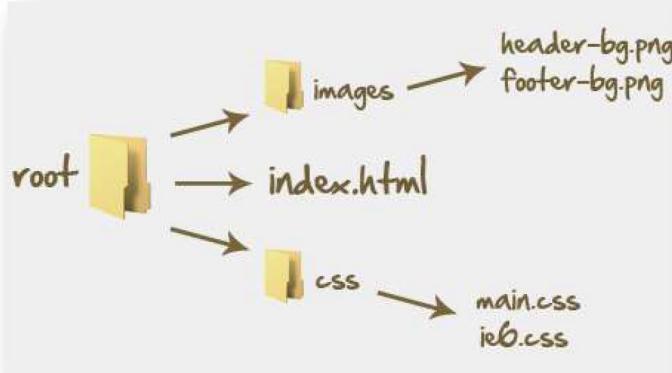
Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

Tecnologías para la Programación Web

Sitios Web

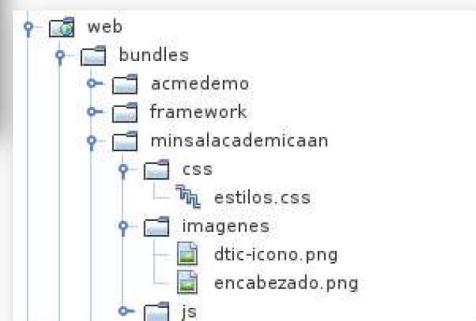
- **Estructura general de un sitio web:**



Alejandro Cardo Grau

- Conjunto de páginas web relacionadas, a las cuales se puede acceder a través de un mismo nombre de dominio DNS.

- El conjunto de sitios web en Internet constituye la WWW (World Wide Web).



Desarrollo Web en Entorno Servidor

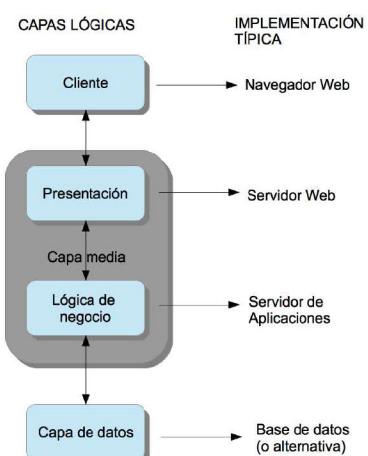
Tecnologías para la Programación Web

Aplicación Web

- **Aplicación web:**

- Aplicación software que los usuarios pueden utilizar accediendo a un servidor web, a través de Internet o intranet, mediante un navegador web para:
 - **Interactuar y acceder a diferentes servicios, recursos y necesidades** que ofrece el servidor web:
 - Ejemplos: buscador, cliente de correo electrónico, tienda electrónica, etc.
- Una aplicación web necesita de una estructura que permita su acceso desde diferentes lugares. Esta estructura es denominada:
 - **Arquitectura Web** (diseño de toda la estructura software).
 - Basadas en el modelo cliente/servidor y en una arquitectura software basada en *n-capas*.
 - Capa del cliente (*frontend*): El uso de HTML5 mejora la funcionalidad que puede ejecutarse nativamente en un navegador web.

Alejandro Cardo Grau

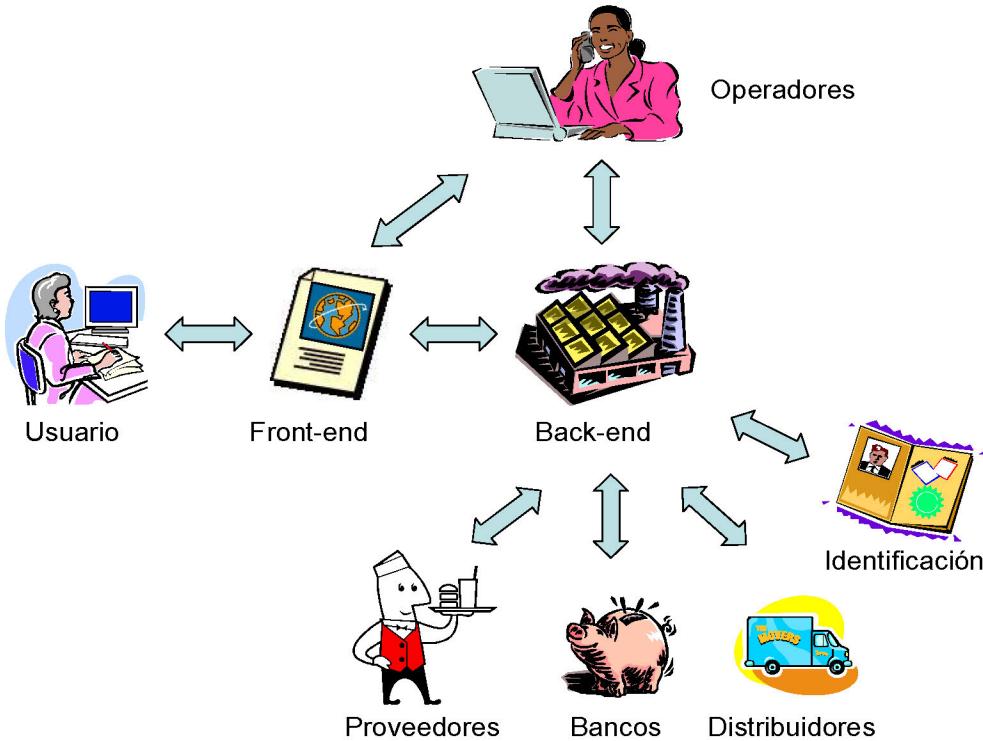


Desarrollo Web en Entorno Servidor

Tecnologías para la Programación Web

Clasificación de los lenguajes y tecnologías web

- ¿Qué tecnologías y lenguajes de programación debe conocer un programador web actual?



Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

Tecnologías para la Programación Web

Clasificación de los lenguajes y tecnologías web

- ¿Qué tecnologías y lenguajes de programación debe conocer un programador web actual?
 - **FRONTEND (lado del cliente):**
 - El lenguaje HTML (*Hypertext Markup Language*)
 - El uso de hojas de estilo CSS (*Cascading Style Sheets*)
 - Lenguajes orientados al lado del cliente (Javascript/JQuery Mobile)
 - Manejo de la Jerarquía de objetos DOM (*Document Object Model*)
 - **BACKEND (lado del servidor):**
 - Un lenguaje orientado al lado del servidor (PHP, Django, Node.js, Phyton)
 - Administración y programación de gestores de contenido (Drupal,Django)
 - El manejo de operaciones en base de datos de servidor (SQL / Oracle)
 - Características de configuración de servidores web (Apache / Tomcat)
 - Procesamiento de la información con XML (Java / C# / XPath / XQuery)
 - **ENTORNOS DE DESARROLLO:**
 - Netbeans
 - Eclipse
 - Webstorm
 - Otros IDE especializados

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

Tecnologías para la Programación Web

Historia, versiones y evolución

- W3C (consorcio de empresas de la *World Wide Web*) publica un nuevo estándar:
 - XML (*eXtended Markup Language*)
 - Basado en SGML y más sencillo
 - Más potente que HTML 4.01 (1999).
 - Sencillez de HTML + Flexibilidad SGML
 - Definición de lenguajes propios
- XHTML (*eXtensible HyperText Markup Language*):
 - HTML basado en XML.
 - Reformulación del HTML4.
 - Separación de **contenido** y **presentación** (aspecto/formato) para facilitar la **accesibilidad**.
- **Objetivo Inicial: Pensado para modificar la World Wide Web y adaptarla a XML en la evolución de la web semántica:**
 - Complicación del desarrollo web por la falta de apoyo en empresas, al ser restrictivo y con falta de semántica.
 - Nacen nuevas recomendaciones: **HTML5** (2009)

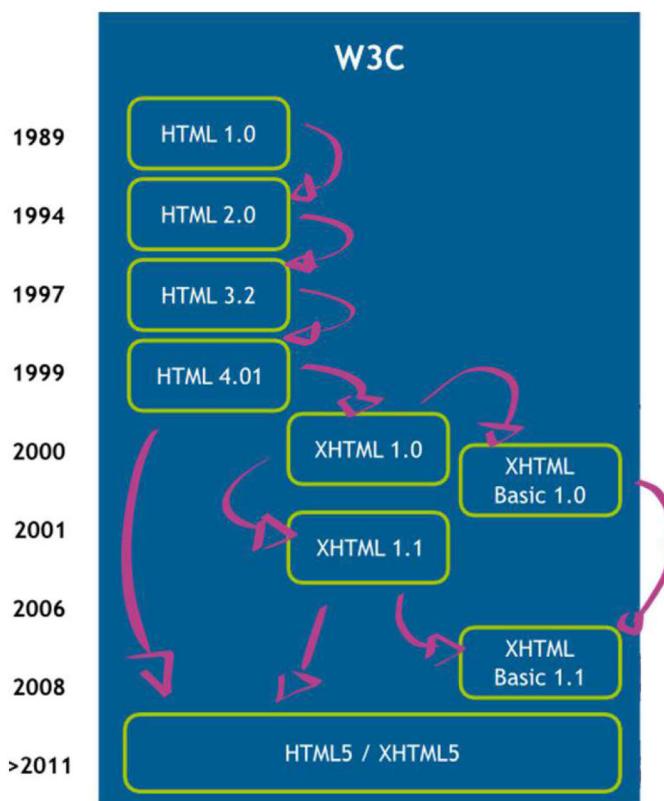


Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

Tecnologías para la Programación Web

Historia, versiones y evolución



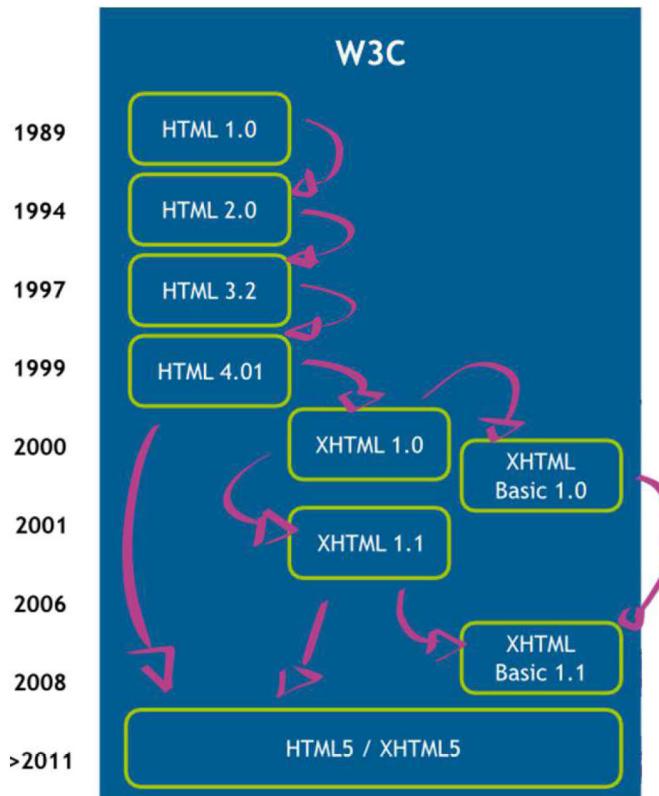
- **1994:** Se formaliza el HTML 2.0 y con ello la sintaxis y la mayoría de las reglas que se encuentran actualmente implementadas.
- **1996:** Nace CSS (*Cascading Style Sheets*) para asociar estilos a documentos web basados en HTML cuyo objetivo es separar contenido y presentación.
- **1997:** HTML 3.2 incorpora en su sintaxis: tablas, flujo del texto alrededor de las imágenes y compatibilidad con el anterior.
- **1999:** Se estabiliza la sintaxis y la estructura del HTML 4.0, convirtiéndose en el estándar definitivo para la web. Se promocionan navegadores web basados en dichos estándares. Se añade al lenguaje:
 - Accesibilidad usuario/dispositivos
 - Inclusión de objetos multimedia
 - Formularios
 - Internacionalización y *scripting*

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

Tecnologías para la Programación Web

Historia, versiones y evolución

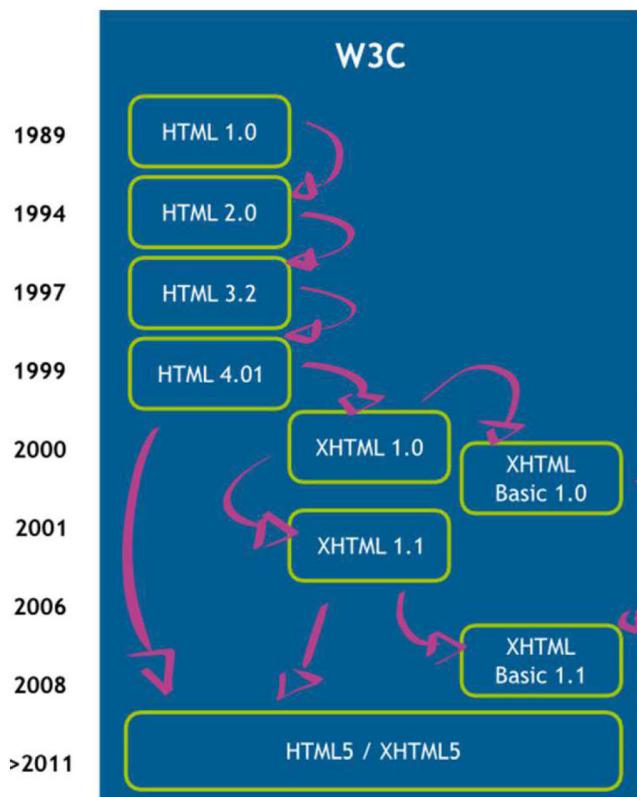


Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

Tecnologías para la Programación Web

Historia, versiones y evolución

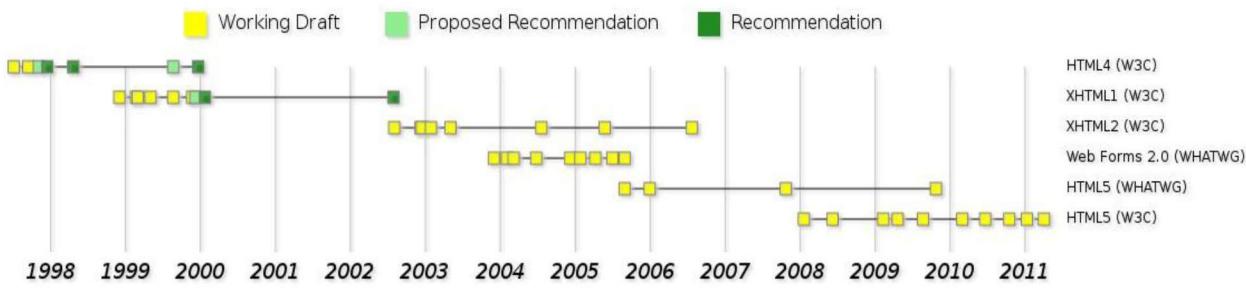


Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

Tecnologías para la Programación Web

Historia, versiones y evolución



Línea de evolución de las tecnologías de lenguajes de marcado (1998-2011)

(Fuente: <http://saurabhchalke.wordpress.com>)

- Las páginas web están escritas en el lenguaje de marcas HTML ó XHTML que:
 - Establecen **directivas o reglas** para **normalizar** el **diseño** del contenido y validar su gramática:
 - Se colocan al principio del documento para definir su tipo.
 - Definen el **tipo del documento web** (no es suficiente con la extensión y formato)
 - Es usado para identificar la **versión del XHTML ó HTML** para los navegadores web.

```
index.html
<!DOCTYPE html>
<html lang="en">
<head>
<title>HTML5: Up and
Running</title>
</head>
<body>
...

```

Desarrollo Web en Entorno Servidor

Alejandro Cardo Grau

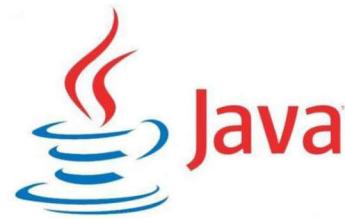
Tecnologías para la Programación Web

Servidor Web

- Un servidor web proporciona un servicio a través de una red:**
 - Ofreciendo los medios para permitir la comunicación entre aplicaciones web sin importar la tecnología usada para crear **interoperabilidad**.
 - La comunicación suele hacerse mediante el **protocolo HTTP** que está englobado en la capa de aplicación del modelo OSI.
 - Ejemplos de servidores web:
 - Apache, ISS, Nginx, Lighttpd, etc.
 - Ejemplos de servidores web de aplicación:
 - Tomcat, Wildfly, Glassfish, Websphere, Liferay (oncloud), etc.
- Inconvenientes:**
 - Latencia y poca fiabilidad del transporte (por ejemplo la red).
 - Problemas derivados de fallos parciales.
 - Gestión del acceso concurrente a recursos remotos.
 - Falta de memoria compartida entre las partes.
 - Problemas derivados de actualizaciones de alguna/s de las partes.
 - Búsqueda entre equilibrio de la **seguridad** y la **eficiencia** en la aplicación:
 - Demasiada seguridad puede ralentizar el uso, aumentar el tráfico, etc.

Alejandro Cardo Grau

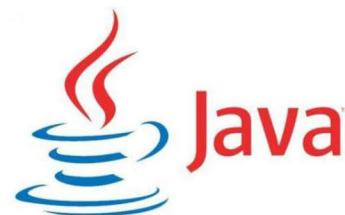
Desarrollo Web en Entorno Servidor



- **Java Enterprise Edition (Java EE)**

- Tecnología basada en **Java**
- Desarrollada por una coalición de empresas lideradas por **Oracle, IBM, Red Hat**, etc..
- Tecnología muy usada a nivel **empresarial**
- La mayoría de las **implementaciones y herramientas** para desarrollo son **software libre**
- Las aplicaciones se ejecutan en servidores web implementados en Java (**Tomcat, Glassfish, JBoss, Jetty...**)
- Estos servidores se integran en los servidores web **Apache, NginX e IIS**

<http://www.oracle.com/javaee>

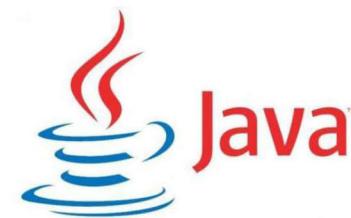


- **Estándares en Java EE**

- Java tiene una organización de estandarización propia llamada Java Community Process (JCP).
- En ella se definen estándares abiertos que se pueden implementar con licencia libre o propietaria
- Estándares web:
 - Java EE, Servlets, JSP, JDBC, JPA, JSF, EJBs...

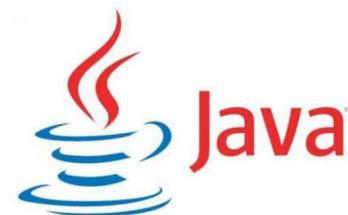
- **Bibliotecas y frameworks en Java EE**

- Existen multitud de implementaciones independientes que pueden seguir o no un estándar
- Ejemplos:
 - Spring, Struts, Hibernate, GWT, Vaadin, Google Closure, ...



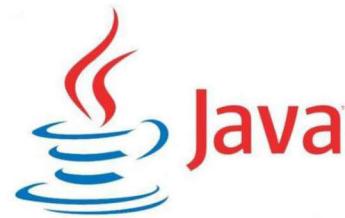
- **Estándares más importantes en Java EE**

- **Servlets:** Estándar para ejecutar código Java ante una petición web en un servidor Java EE
- **JSP (Java Server Pages):** Estándar que permite mezclar en un documento código Java y HTML para generar páginas web de forma dinámica
- **JDBC (Java Database Connectivity):** Estándar para conexión a bases de datos relacionales desde Java
- **JPA (Java Persistence API):** Estándar para la correspondencia objeto-relacional (ORM, *Object Relational Mapping*)
- **JSF (Java Server Faces):** Estándar de construcción de aplicaciones web basadas en componentes reutilizables
- **EJB (Enterprise JavaBeans):** arquitectura manejada para la construcción de aplicaciones web (transacciones, seguridad, distribución...)



- **Servidores Java EE**

- Toda aplicación web Java EE tiene que ejecutarse en una servidor de aplicaciones Java (aunque luego se integre en **Apache, NginX o IIS**)
- Existen muchos tipos de **servidores**, dependiendo de sus funcionalidades/rendimiento y de su licencia/coste.
- Ejemplos: **Glassfish** (Oracle), **Tomcat** (Apache), **Jetty** (Eclipse), **Wildfly** (RedHat), **WebSphere** (IBM), **WebLogic** (Oracle)

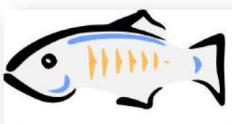


Servidor de aplicaciones

- Un **servidor de aplicaciones** es un framework que proporciona infraestructura para el despliegue, ejecución y gestión de aplicaciones
- Típicamente consiste en un servidor web (HTTP) que ejecuta aplicaciones dinámicas en el lado servidor
- En el mundo Java también se conoce a los servidores de aplicaciones como **contenedores** (containers)
 - Como Java EE es un superconjunto de Java SE, cualquier aplicación Java EE puede usar la API de Java SE.

Contenedores Java EE

Cumplen con la especificación Java EE completa (o al menos el perfil Web)



Glassfish



WildFly

Contenedores Web

ofrecen la APIs de Servlets y JSPs. Se le pueden añadir otras librerías Java EE (excepto EJB)



Apache Tomcat
(Servlets y JSPs)



Eclipse Jetty
(Servlets y JSPs)

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor



- Una **aplicación Java SE** se empaqueta en un **fichero JAR** (Java Archive)
 - Un JAR es un fichero comprimido que contiene clases Java compilados (bytecodes, ficheros .class, etc.) y otros recursos.
- Una **aplicación web Java EE** se empaqueta en un **fichero WAR** (Web Application Archive)
 - Un WAR es también un fichero comprimido que contiene:
 - Los componentes Java activos en el servidor (servlets, etc).
 - La aplicación web (HTML, CSS, Java-Scripts, imágenes, etc.).
- **Proporciona:**
 - **Seguridad:**
 - Se puede firmar digitalmente el contenido del fichero JAR y WAR.
 - **Compresión:**
 - JAR y WAR comprime los ficheros usando el propio IDE.
 - **Portabilidad:**
 - Los ficheros JAR y WAR son un estándar del API de Java y JavaEE.
 - **Empaquetado**
 - Para extensiones, sellado y versionado de aplicaciones.
 - **Recursos:**
 - Al descargarse todos los recursos necesarios de una aplicación empaquetados en un único artefacto generado durante el desarrollo.

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor



- **PHP**

- Desarrollado en 1994 por **Rasmus Lerdorf**
- La tecnología dispone de un lenguaje con tipos dinámicos llamado **PHP**
- Desarrollada por **PHP Group** con licencia libre **PHP license**
- Es multiplataforma
- Es una tecnología multiplataforma que se integra bien con servidores como **Apache**, **NginX** e **IIS**
- Se puede usar **Netbeans** o **Eclipse** para su desarrollo
- **Facebook** está implementado con PHP

<http://www.php.net/>



- **Bibliotecas y frameworks PHP**

- Existen multitud de **frameworks** para el desarrollo de aplicaciones PHP
- Ejemplos: Laravel, CakePHP, CodeIgniter, Zend, Symfony, Horde



<http://www.phpframeworks.com/>



• ASP.NET

- Versión evolucionada del **ASP clásico**.
- Forma parte de la **tecnología .NET de Microsoft**
- Se utiliza el lenguaje **C#** (muy similar a Java)
- Licencia **propietaria** y para plataformas **Windows**
- Usando **Mono** se puede usar limitado en Linux
- Se integra bien con el servidor **IIS**
- El desarrollo se realiza con **Visual Studio .NET**

<http://www.asp.net/>

• HTTP (Hypertext Transfer Protocol)

- Protocolo de red, **a nivel de aplicación**, que define las reglas que utilizan los componentes software (clientes, servidores y proxies) para comunicarse vía web.
- Es un protocolo que utiliza TCP y determina los **tipos de peticiones** que los clientes pueden enviar, formato y estructura de las respuesta, así como las de **metadatos**. Los clientes y servidores utilizan:
 - Versión actual: HTTP/1.1 (diferencia conexiones persistentes, etc.)
 - Versión experimental: HTTP/1.2
- Propuesto por Tim Berners-Lee en 1989
- Conexión TCP (puerto 80 que escucha pasivamente)
- Más información: <http://www.w3.org/protocol>

HTTP

Protocolo HTTP

- **HTTP (Hypertext Transfer Protocol)**

- Es un protocolo sin estado, es decir que no guarda ninguna información sobre conexiones anteriores.
- En HTTP/1.1 pueden mantenerse **conexiones persistentes** que realizan:
 - Múltiples peticiones y respuestas sobre la misma conexión TCP.
 - El cliente y servidor web mantienen las conexiones con caché abiertas por defecto.
 - Ventajas: reducción del número de conexiones y ahorro de recursos (memoria, CPU, etc.)
- La ejecución de aplicaciones web necesita frecuentemente mantener estados y almacenamiento temporal o persistente:
 - Para esto se usan las **cookies** que es información que un servidor puede almacenar en el sistema navegador web del cliente.
 - Permite rastrear a los usuarios ya que las cookies pueden guardarse en el cliente por tiempo indeterminado.

Alejandro Cardo Grau

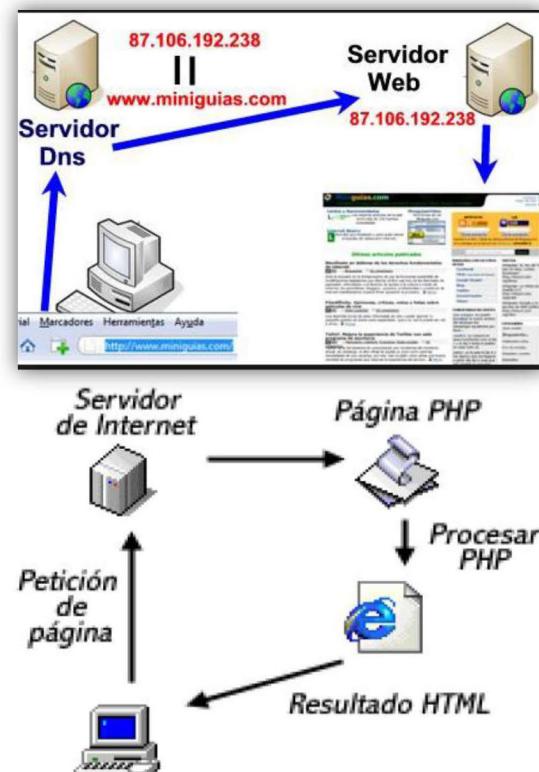
Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP

- **HTTP (Hypertext Transfer Protocol)**

- 1) El primer paso consiste en transformar el nombre del servidor de la URL en una dirección IP, utilizando la base de datos conocida como DNS.
- 2) La dirección IP permite contactar al servidor web y hacer la transferencia de los paquetes de datos.
- 3) Luego se envía una solicitud HTTP al servidor a fin de lograr acceder al recurso:
 - 1) Primero se pide el texto HTML y después se produce el análisis por parte del navegador web, que realiza otras peticiones reservadas a los recursos multimedia que formen parte de la página, sitio o aplicación web.



Interpretación de un script PHP en un servidor web de Internet

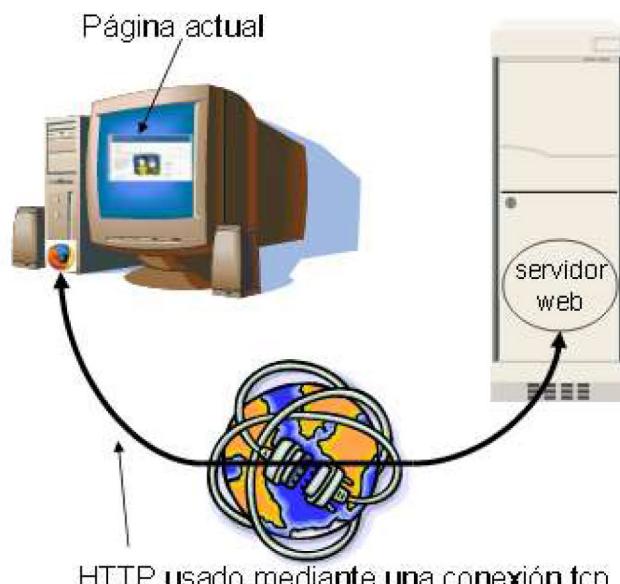
Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP**Protocolo HTTP**

- **HTTP (Hypertext Transfer Protocol)**

- El navegador solicita recurso
- Se determina la URL
- Se resuelve la IP (DNS)
- Se establece conexión TCP con puerto 80 de la IP destino
- Se transmite el método GET
`<URI> <protocolo>`
`(/archivo.html HTTP/1.1)`
- El servidor responde (según extensiones MIME y RFC822)
- Se cierra la conexión (HTTP 1.0)
- Se presenta el recurso en el "navegador"

**Transacción HTTP**

Alejandro Cardo Grau

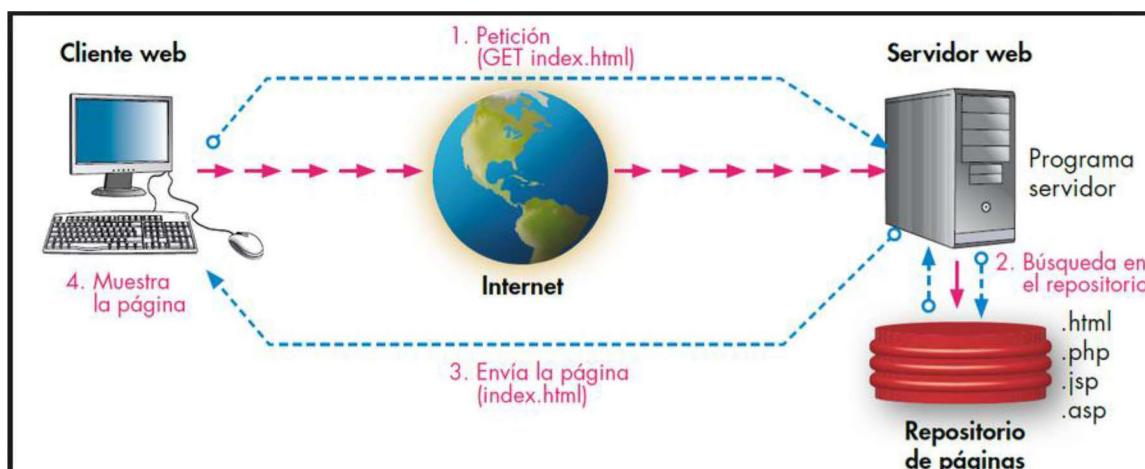
Desarrollo Web en Entorno Servidor

HTTP**Protocolo HTTP**

- **HTTP (Hypertext Transfer Protocol)**

1. CONEXIÓN
2. SOLICITUD
3. RESPUESTA
4. CIERRE (*)

- 1) El cliente realiza una petición o apertura activa (request) al servidor (puerto 80, por defecto)
- 2) Sigue la transacción con HTTP: GET, POST, HEAD, PUT, ...
- 3) El servidor envía la respuesta (response) en HTML
- 4) Se cierra la conexión (en HTTP/1.0, pero en HTTP/1.1 puede mantenerse persistente).



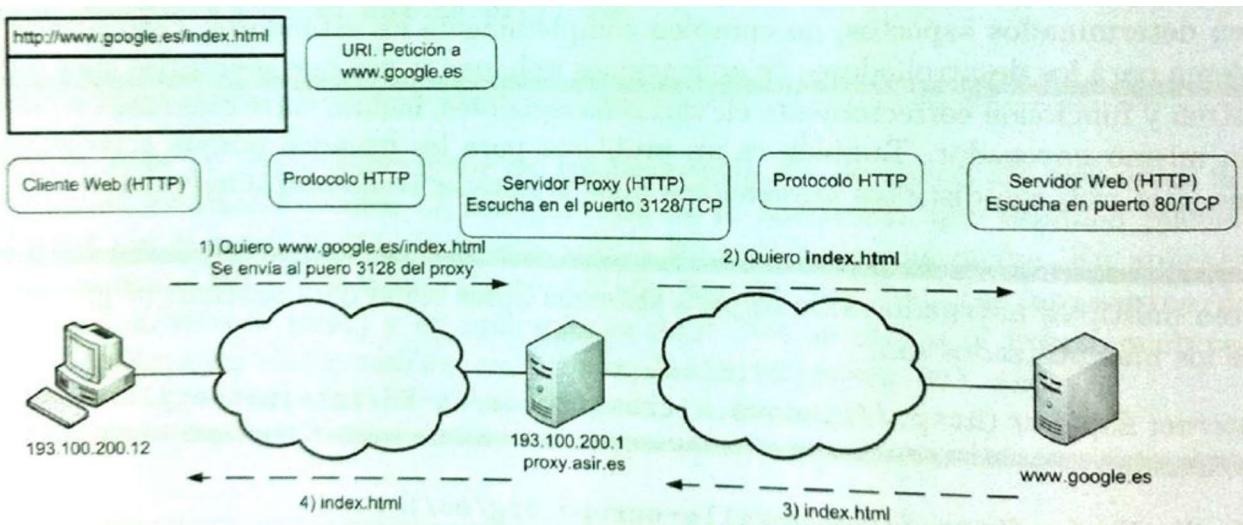
Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP

- **HTTP (Hypertext Transfer Protocol)**



Intermediarios entre clientes y servidores web (*proxies*)

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP

- **HTTP: Tipos de Mensajes**
- Dos tipos de mensajes usando ASCII (texto plano):
 - Request (Solicitud)
 - Response (Respuesta)
- Solicituds HTTP/1.0

Línea de solicitud → `GET /path/pagina.html HTTP/1.0`
 (método, URI, ver.)

Líneas de cabecera → {
 - `User-agent: Mozilla/6.0`
 - `Accept: text/html, image/gif, image/jpg`
 - `Accept-language: es`
 Fin de la cabecera → `<CR><LF>`

Datos (opcional) →

- Existen 3 métodos distintos:

- GET: Se utiliza para recoger cualquier tipo de información del servidor. Esta información va en el cuerpo de la respuesta.
- HEAD: Sigue el mismo formato que GET, pero la información va contenida en la cabecera de la respuesta. NUNCA TIENEN CUERPO.
- POST: Se utiliza para enviar información al servidor, por ejemplo los datos contenidos en un formulario.

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

- **HTTP: Tipos de Mensajes**

■ Respuestas HTTP/1.0

Línea de estado → **HTTP/1.0 200 OK**
 (ver., código, frase)

Líneas de cabecera {
Date: Thu, 23 Nov 2004 12:00:15GMT
Server: Apache/2.0 (Unix)
Last-Modified: Mon, 10 Sept 2004
 ...
Content-Length: 6821
Content-Type: text/html
 ...

Datos → <html> </html>

- **HTTP (Hypertext Transfer Protocol)**

```
C: telnet www.w3.org 80
T: Trying 18.23.0.23 ...
T: Connected to www.w3.org.
C: GET /hypertext/WWW/TheProject.html HTTP/1.0
C:
S: HTTP/1.0 200 Document follows
S: MIME-Version: 1.0
S: Server: CERN/3.0
S: Content-Type: text/html
S: Content-Length: 8247
S:
S: <HEAD><TITLE>The World Wide Web Consortium </TITLE></HEAD>
S: <BODY>
S: <H1>...</H1>
S: ...
S: </BODY>
```

Ejemplo real de mensajes HTTP usando Telnet/23

HTTP

Protocolo HTTP

- **HTTP: Códigos de Estado**

- Identificador del estado de la petición.
- Los envía el servidor web como respuesta.
- Entero de 3 dígitos:
 - 1xx: Informativos.
 - 2xx: Operación realizada con éxito.
 - 3xx: Redireccionan al cliente a otra URL.
 - 4xx: Error del cliente.
 - 5xx: Error del servidor.
- Los más usuales:
 - 200 OK: solicitud exitosa, la respuesta va en el cuerpo.
 - 404 Not Found: el recurso no existe.
 - 303 See Other: el recurso se ha movido a otra URL (ver Header Location).
 - 500 Server Error: error interno del servidor.

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP**PETICIÓN HTTP (página web)****GET / HTTP/1.1**

```

Host: www.24x7linux.com
User-Agent: Mozilla/5.0 (X11; U; Linux i686; en-US;
rv:1.2b) Gecko/20021016
Accept:text/xml,application/xml,application/xhtml+xml,te-
xt/html;q=0.9, text/plain;q=0.8,video/x-
mng,image/png,image/jpeg,image/gif;q=0.2,
text/css,*/*;q=0.1
Accept-Language: es-es, en-us;q=0.66, en;q=0.33
Accept-Encoding: gzip, deflate, compress;q=0.9
Accept-Charset: ISO-8859-15, utf-8;q=0.66, *;q=0.66
Keep-Alive: 300
Connection: keep-alive
  
```

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP

RESPUESTA DEL SERVIDOR

HTTP/1.1 200 OK

Date: Sun, 10 Nov 2002 22:50:55 GMT

Server: Apache/1.3.26 (Unix) mod_bwlimited/1.0

PHP/4.2.2 mod_log_bytes/0.3

FrontPage/5.0.2.2510 mod_ssl/2.8.9 OpenSSL/0.9.6b

Content-Type: text/html

Age: 130

Connection: close

<-- archivo index.html que contiene la página principal
del sitio -->

HTTP

Protocolo HTTP

• Declaración de formularios web:

- Los atributos más importantes de <form> son (opcionales dependiendo del tratamiento y procesamiento que necesite):

- **action:**

- Indica la URI donde se especifican las acciones a seguir para procesar los datos del formulario.
- Es el agente procesador del formulario que reside en el servidor (p.e. script JSP).
- Se encarga de generar la respuesta tras la solicitud del cliente.

- **Ejemplos:**

```
<form>
  <!-- Controles del formulario que sólo se representarán y no se procesan -->
</form>
```

1º CURSO

```
<form action="registro.php">
  <!-- Controles del formulario cuyos datos se procesarán por un script PHP-->
</form>
```

2º CURSO

• Declaración de formularios web:

- **method:** Establece la forma en que se enviarán los datos al servidor. Los valores posibles de este atributo son:
 - GET (**por defecto**):
 - No permite el envío de archivos adjuntos al fichero.
 - Admite como máximo 500 bytes de información.
 - Se envían los datos del <form> al agente servidor incrustados dentro de la URL:

```
http://www.web.com/registro.php?nombre=pepe&email=pepe@gma.es
```

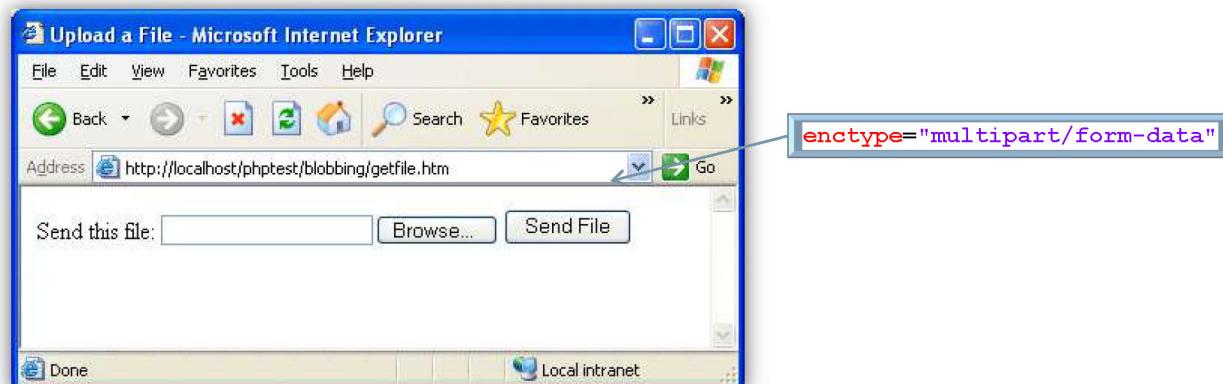
- Los datos se añaden a la URL del agente servidor especificado en action después del carácter ? en parejas name=valor separadas por &.

- POST:

- Los valores se envían de forma separada ofreciendo más seguridad:
 - Los datos no pueden ser observados en la URL (como ocurre en GET).
 - Permite el envío de ficheros adjuntos.

enctype: especifica el formato de codificación de los datos cómo se envían

```
<form action="acceso.asp" method="post">
<form action="formulario.php" method="get" enctype="text/plain">
```

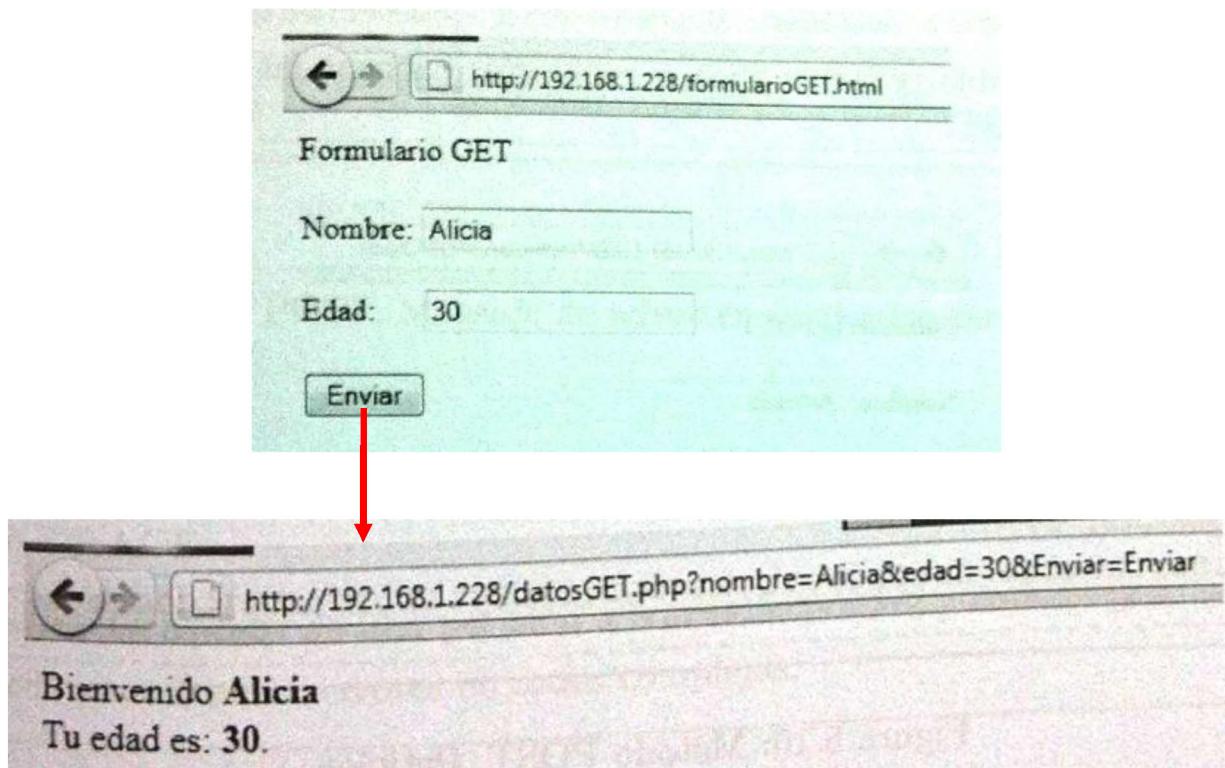


Uso del método GET y POST en formularios web

HTTP

Protocolo HTTP

- Método GET



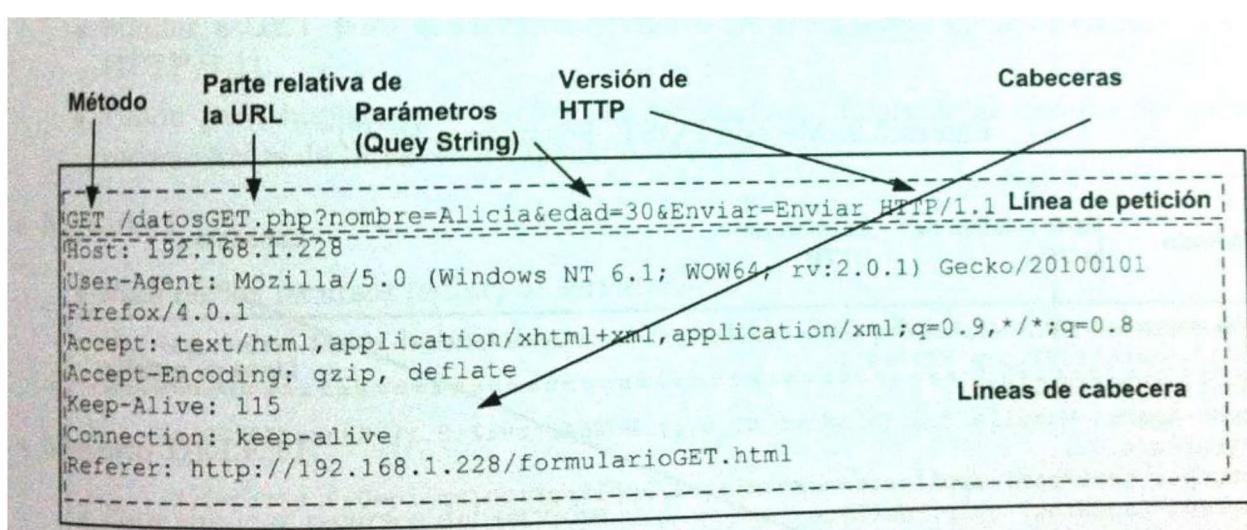
Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP

- Método GET



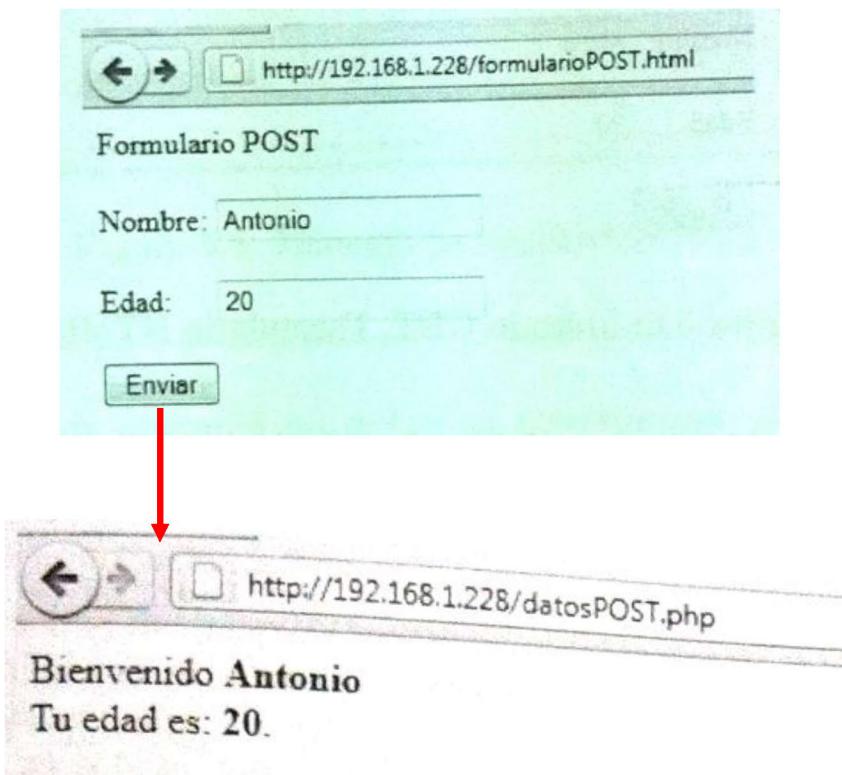
Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP

- **Método POST**



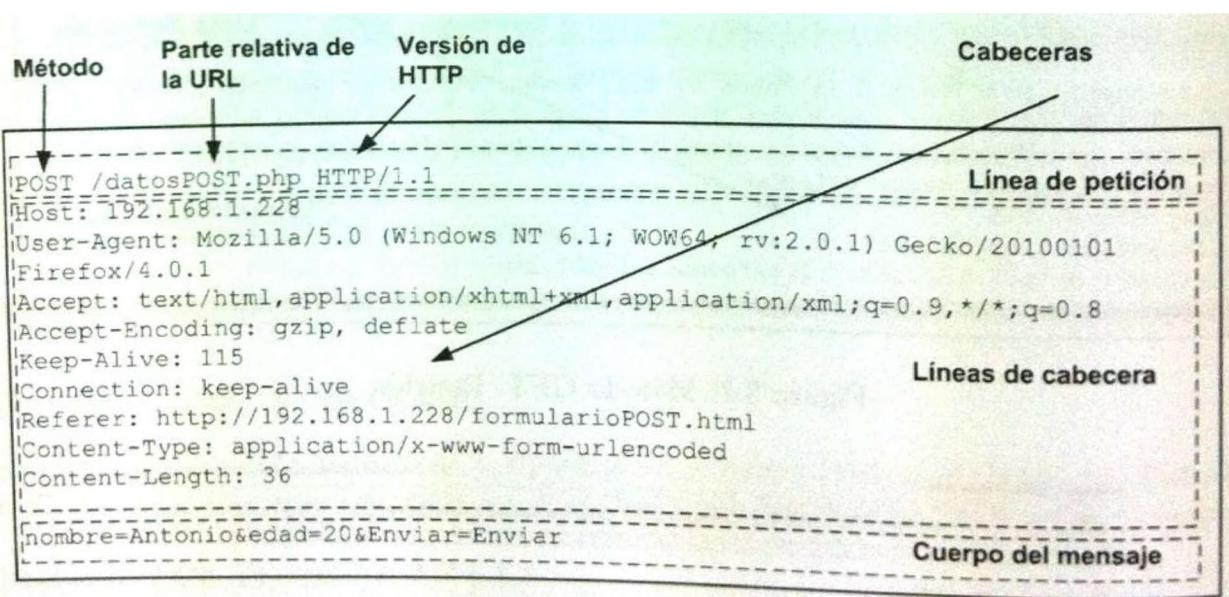
Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP

- **Método POST**



Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

Protocolo HTTP

- **Autenticación web:**

- Es un método para verificar la identidad del usuario y determina el nivel de acceso al recurso que posee el sitio o aplicación web.

- **Tipos de autenticación web:**

- **Anónima**
- **Basic:** El cliente envía usuario y clave codificados en algoritmo Base 64:
 - Codificación simple de 6 bits que une en la misma cadena login.password (separado por punto) → Método inseguro, se captura fácilmente la clave con sniffer.
- **Digest:** El cliente envía usuario y una función hash (resumen) de la clave al servidor usando el algoritmo MD5:
 - Codificación de 128 bits para hallar el compendio del password → Más seguro que Basic, pero presenta vulnerabilidades a ataques.

- **HTTP no es un protocolo seguro y existen varias alternativas:**

- 1) Tratamiento desde el *back-end* a partir de scripting: JSP, PHP, etc.
- 2) Uso de certificados digitales y HTTPS (Hypertext Protocol Secure) usando protocolos TSL/SSL.

Protocolo HTTP

- **HTTPS (HyperText Transfer Protocol Secure)**

- Protocolo de red, a nivel de aplicación, que garantiza la confidencialidad e integridad de la información transmitida, usando algoritmos criptográficos y certificados.
- Los clientes usan `https://` en las URL.
- Los servidores web escuchan peticiones por el puerto **443/TCP**.

- **Requisitos de configuración de un servidor web HTTPS:**

- Dirección IP fija.
- Servidor HTTP previamente configurado.
- Configuración del protocolos TTS ó SSL para:
 - Encapsular y cifrar los mensajes HTTP.
- Certificado (ya sea autofirmado, propio o externo)
- Sitios web enlazados.

HTTP

Protocolo HTTP

- **Cookies**

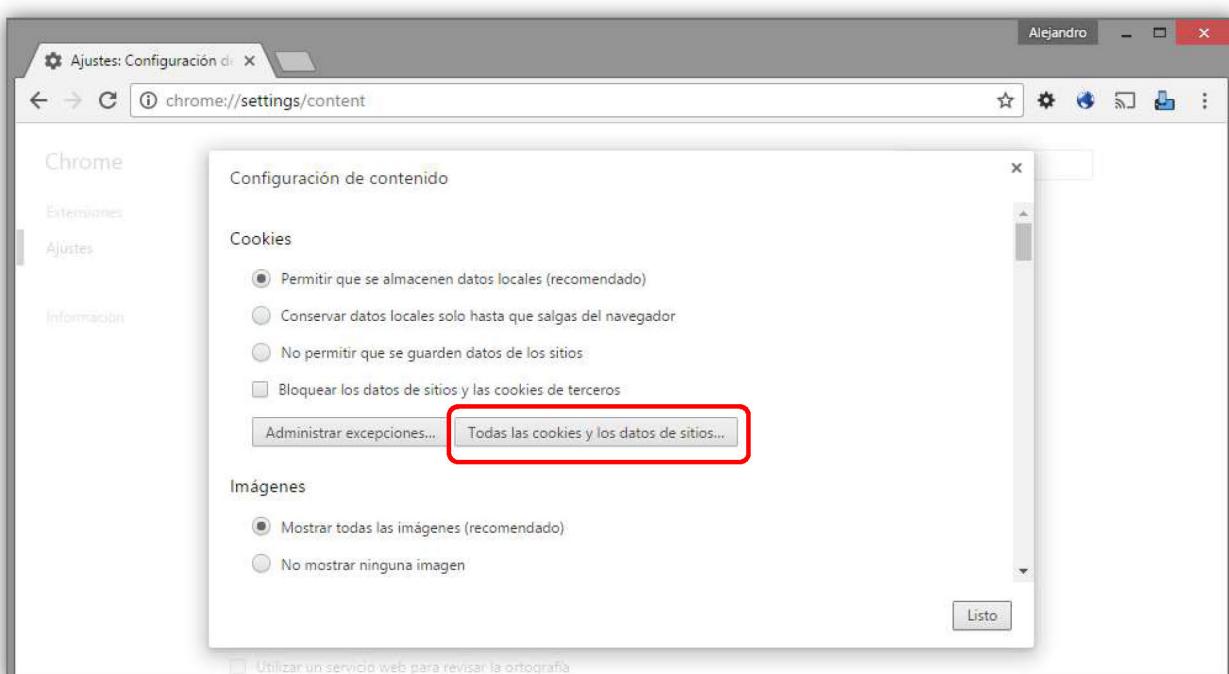
- Fragmento de información textual que envía un servidor web en una respuesta HTTP y es almacenada, si la configuración del navegador web cliente es permitida.
 - Son utilizadas por los servidores web para diferenciar usuarios y conexiones y actuar en consecuencia.
 - Pueden ser utilizados con fines malintencionados para obtener datos privados de los usuarios como hábitos de navegación y sitios web visitados.
- HTTP es un protocolo sin estado, cada transferencia de datos es independiente :
 - Ejemplos: Memorización del username-password, anuncios y publicidad personalizada según historial y navegación, etc.
- Las cookies pueden ser aceptadas, bloqueadas o borradas según la configuración del navegador web.

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP



Configuración de cookies en Google Chrome

Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP

- **Cookies**

- Los servidores envían las cookies usando cabeceras con los siguientes detalles:
 - **Name:** Nombre de la cookie:
 - **Value:** Texto con el valor del contenido de la cookie.
 - **Expires:** Fecha/hora de expiración en la que la cookie deberá ser descartada por el navegador web.
 - **Path:** Lugar o ubicación donde la cookie será guardada por el navegador web.
 - **Domain:** Nombre de dominio de procedencia de la cookie.

```

HTTP/1.1 200 OK
Date: Wed, 18 May 2011 11:23:39 GMT
Server: Apache/2.2.16 (Debian)
X-Powered-By: PHP/5.3.3-7
Set-Cookie: visitas=1; expires=Thu, 17-May-2012 11:23:39 GMT
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 87
Keep-Alive: timeout=15, max=99
Connection: Keep-Alive
Content-Type: text/html
  
```

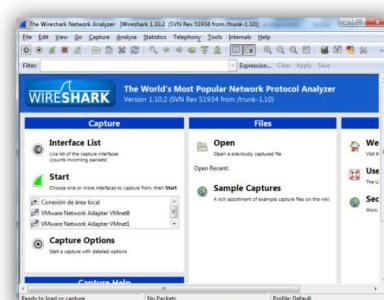
Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor

HTTP

Protocolo HTTP

- **Wireshark**, es un analizador *sniffer* de protocolos de red utilizado para:
 - Realizar análisis de protocolos y solucionar problemas en redes de comunicaciones.
- Permite ver todo el tráfico que pasa a través de una red (usualmente una red Ethernet, aunque es compatible con algunas otras) estableciendo la configuración en modo promiscuo.
- Permite examinar datos de una red local o Internet o a partir de un archivo de captura almacenado previamente.
- Es software libre y multiplataforma:
 - Se ejecuta sobre la mayoría de sistemas operativos Unix y Windows.
 - También incluye una interfaz gráfica y una versión basada en texto llamada *tshark*.



Alejandro Cardo Grau

Desarrollo Web en Entorno Servidor