Ambientes no propietarios

Introducción

Edwin Salvador

14 de abril de 2015

Sesión 2

Contenido I

Contenido II

Control de versiones con Git

- Git es un sistema de control de versiones (VCS) de código abierto y distribuido capaz de trabajar con proyectos grandes y pequeños de manera eficiente.

Control de versiones con Git

- Git es un sistema de control de versiones (VCS) de código abierto y distribuido capaz de trabajar con proyectos grandes y pequeños de manera eficiente.
- Git funciona estrictamente mediante la linea de comandos.

Control de versiones con Git

- Git es un sistema de control de versiones (VCS) de código abierto y distribuido capaz de trabajar con proyectos grandes y pequeños de manera eficiente.
- Git funciona estrictamente mediante la linea de comandos.
- Git facilita facilita la interacción con *GitHub* (un repositorio web con una interfaz amigable).

Control de versiones con Git

- Git es un sistema de control de versiones (VCS) de código abierto y distribuido capaz de trabajar con proyectos grandes y pequeños de manera eficiente.
- Git funciona estrictamente mediante la linea de comandos.
- Git facilita facilita la interacción con *GitHub* (un repositorio web con una interfaz amigable).
- Brevemente veamos como instalar y configurar Git en nuestro ambiente de trabajo.

Contenido I

Creando un repositorio remoto en GitHub

- Clic en el (+) en la esquina superior derecha (Nuevo repositorio).
- Ingresar nombre del repositorio (esfot_ambientesnopropietarios)
- Ingresar descripción del repositorio
- Seleccionar Público, no inicializar con README.md, .gitignore = none, license = none.
- Clic en crear

Contenido I

Instalando Git

- Crear una cuenta gratuita en http://github.com

Instalando Git

- Crear una cuenta gratuita en http://github.com
- Descargar Git bash desde:

Instalando Git

- Crear una cuenta gratuita en http://github.com
- Descargar Git bash desde:
  - http://git-scm.com (Trabajaremos con esta)

Instalando Git

- Crear una cuenta gratuita en http://github.com
- Descargar Git bash desde:
  - http://git-scm.com (Trabajaremos con esta)
- GitHub ofrece una interfaz gráfica (no necesario):

Instalando Git

- Crear una cuenta gratuita en http://github.com
- Descargar Git bash desde:
  - http://git-scm.com (Trabajaremos con esta)
- GitHub ofrece una interfaz gráfica (no necesario):
  - https://windows.github.com (Windows)

Instalando Git

- Crear una cuenta gratuita en http://github.com
- Descargar Git bash desde:
  - http://git-scm.com (Trabajaremos con esta)
- GitHub ofrece una interfaz gráfica (no necesario):
  - https://windows.github.com (Windows)
  - https://mac.github.com (Mac)

Configurando Git

Empezamos con la configuración del usuario para todos los repositorios locales.

- El nombre que queremos que vaya con nuestras transacciones `commit`:

```
$ git config --global user.name "[nombre]"
```

Configurando Git

Empezamos con la configuración del usuario para todos los repositorios locales.

- El nombre que queremos que vaya con nuestras transacciones `commit`:

```
$ git config --global user.name "[nombre]"
```

- El email que queremos que vaya con nuestras transacciones `commit`:

```
$ git config --global user.email "[email address]"
```

Configurando Git

Empezamos con la configuración del usuario para todos los repositorios locales.

- El nombre que queremos que vaya con nuestras transacciones `commit`:

```
$ git config --global user.name "[nombre]"
```

- El email que queremos que vaya con nuestras transacciones `commit`:

```
$ git config --global user.email "[email address]"
```

- Habilitar colores en la línea de comandos:

```
$ git config --global color.ui auto
```

Contenido I

Creando repositorios

Para crear un nuevo repositorio o obtener uno desde una URL ya existente.

- Crear un nuevo repositorio local con un nombre específico:

```
$ git init [project-name]
```

Creando repositorios

Para crear un nuevo repositorio o obtener uno desde una `URL` ya existente.

- Crear un nuevo repositorio local con un nombre específico:

```
$ git init [project-name]
```

- También podemos inicializar en un directorio ya existente, en este caso ejecutaremos solamente `git init`.

Creando repositorios

Para crear un nuevo repositorio o obtener uno desde una `URL` ya existente.

- Crear un nuevo repositorio local con un nombre específico:

```
$ git init [project-name]
```

- También podemos inicializar en un directorio ya existente, en este caso ejecutaremos solamente `git init`.
- Descargar un proyecto existente:

```
$ git clone [url]
```

Configurando el repositorio local

- Crear archivo README para el repositorio:

```
touch README.md
```

- Modificar el archivo README creado

```
echo texto descriptivo del repositorio » README.md
```

- Crear `.gitignore`

```
touch .gitignore
```

- Modificar y añadir los archivos que se desean excluir del versionamiento. (*.pdf, *.tex, etc)

Contenido I

Realizar cambios

Revisar las ediciones y empezar una transacción `commit`.

- Listar todos los archivos nuevos o modificados que necesitan ser subidos:

```
$ git status
```

Realizar cambios

Revisar las ediciones y empezar una transacción `commit`.

- Listar todos los archivos nuevos o modificados que necesitan ser subidos:

```
$ git status
```

- Mostrar las modificaciones en los archivos que no han sido subidas:

```
$ git diff
```

Realizar cambios

Revisar las ediciones y empezar una transacción `commit`.

- Listar todos los archivos nuevos o modificados que necesitan ser subidos:

```
$ git status
```

- Mostrar las modificaciones en los archivos que no han sido subidas:

```
$ git diff
```

- Añadir un archivo al control de versiones:

```
$ git add [archivo]
```

Realizar cambios

Revisar las ediciones y empezar una transacción `commit`.

- Listar todos los archivos nuevos o modificados que necesitan ser subidos:

```
$ git status
```

- Mostrar las modificaciones en los archivos que no han sido subidas:

```
$ git diff
```

- Añadir un archivo al control de versiones:

```
$ git add [archivo]
```

- Mostrar las diferencias entre el archivo a subir y la última versión subida:

```
$ git diff --staged
```

Realizar cambios

Revisar las ediciones y empezar una transacción `commit`.

- Listar todos los archivos nuevos o modificados que necesitan ser subidos:

```
$ git status
```

- Mostrar las modificaciones en los archivos que no han sido subidas:

```
$ git diff
```

- Añadir un archivo al control de versiones:

```
$ git add [archivo]
```

- Mostrar las diferencias entre el archivo a subir y la última versión subida:

```
$ git diff --staged
```

- Quitar el archivo del control de versiones pero conservar su contenido:

```
$ git reset [archivo]
```

Realizar cambios

Revisar las ediciones y empezar una transacción `commit`.

- Listar todos los archivos nuevos o modificados que necesitan ser subidos:

```
$ git status
```

- Mostrar las modificaciones en los archivos que no han sido subidas:

```
$ git diff
```

- Añadir un archivo al control de versiones:

```
$ git add [archivo]
```

- Mostrar las diferencias entre el archivo a subir y la última versión subida:

```
$ git diff --staged
```

- Quitar el archivo del control de versiones pero conservar su contenido:

```
$ git reset [archivo]
```

- Subir el archivo y los cambios al historial de versiones permanentemente:

```
$ git commit -m "[mensaje descriptivo]"
```

Subir los cambios

- Vincular con el repositorio remoto (en GitHub)

```
git remote add origin https://github.com/NOMBRE_DE_USUARIO/NOMBRE_DEL_REPOSITORIO.git
```

- Subir los cambios al repositorio remoto

```
git push -u origin master
```

- INGRESAR USUARIO
- INGRESAR CONTRASEÑA
- Para evitar que nos pida usuario y contraseña cada vez, seguimos estos pasos:

  https://help.github.com/articles/generating-ssh-keys/

Modificando nombres de archivos

Mover o eliminar archivos con control de versiones.

- Eliminar un archivo del directorio actual y registrar la eliminación.

```
$ git rm [archivo]
```

Modificando nombres de archivos

Mover o eliminar archivos con control de versiones.

- Eliminar un archivo del directorio actual y registrar la eliminación.

```
$ git rm [archivo]
```

- Eliminar el archivo del control de versiones pero mantener la copia local.:

```
$ git rm --cached [archivo]
```

Modificando nombres de archivos

Mover o eliminar archivos con control de versiones.

- Eliminar un archivo del directorio actual y registrar la eliminación.

```
$ git rm [archivo]
```

- Eliminar el archivo del control de versiones pero mantener la copia local.:

```
$ git rm --cached [archivo]
```

- Cambiar el nombre del archivo o moverlo a otro directorio:

```
$ git mv [archivo-original][archivo-renombrado]
```

Contenido I

Contenido II

Contenido web

- ¿Qué es la WWW? *World Wide Web*

Contenido web

- ¿Qué es la WWW? *World Wide Web*
- Gran cantidad de datos generados en el CERN por el uso de acelerador de partículas (Large Hadron Collider (LHC)).

Contenido web

- ¿Qué es la WWW? *World Wide Web*
- Gran cantidad de datos generados en el CERN por el uso de acelerador de partículas (Large Hadron Collider (LHC)).
- Esto derivó en el desarrollo de un método para navegar por la gran cantidad de computadores interconectados.

Contenido web

- ¿Qué es la WWW? *World Wide Web*
- Gran cantidad de datos generados en el CERN por el uso de acelerador de partículas (Large Hadron Collider (LHC)).
- Esto derivó en el desarrollo de un método para navegar por la gran cantidad de computadores interconectados.
- Este método se lo conoció como *Hypertext Transfer Protocol* (HTTP).

Contenido web

- ¿Qué es la WWW? *World Wide Web*
- Gran cantidad de datos generados en el CERN por el uso de acelerador de partículas (Large Hadron Collider (LHC)).
- Esto derivó en el desarrollo de un método para navegar por la gran cantidad de computadores interconectados.
- Este método se lo conoció como *Hypertext Transfer Protocol* (HTTP).
- Junto con HTTP se creó el HTML (*Hypertext Markup Language*).

Contenido web

- ¿Qué es la WWW? *World Wide Web*
- Gran cantidad de datos generados en el CERN por el uso de acelerador de partículas (Large Hadron Collider (LHC)).
- Esto derivó en el desarrollo de un método para navegar por la gran cantidad de computadores interconectados.
- Este método se lo conoció como *Hypertext Transfer Protocol* (HTTP).
- Junto con HTTP se creó el HTML (*Hypertext Markup Language*).
- Para unir estos dos componentes se crean los **servidores web** y los **exploradores web**.

Contenido web

- ¿Qué es la WWW? *World Wide Web*
- Gran cantidad de datos generados en el CERN por el uso de acelerador de partículas (Large Hadron Collider (LHC)).
- Esto derivó en el desarrollo de un método para navegar por la gran cantidad de computadores interconectados.
- Este método se lo conoció como *Hypertext Transfer Protocol* (HTTP).
- Junto con HTTP se creó el HTML (*Hypertext Markup Language*).
- Para unir estos dos componentes se crean los **servidores web** y los **exploradores web**.
- En un principio no se tenían las páginas web, los usuarios debían conectarse y suscribirse a boletines para acceder e intercambiar información.

Contenido web

- ¿Qué es la WWW? *World Wide Web*
- Gran cantidad de datos generados en el CERN por el uso de acelerador de partículas (Large Hadron Collider (LHC)).
- Esto derivó en el desarrollo de un método para navegar por la gran cantidad de computadores interconectados.
- Este método se lo conoció como *Hypertext Transfer Protocol* (HTTP).
- Junto con HTTP se creó el HTML (*Hypertext Markup Language*).
- Para unir estos dos componentes se crean los **servidores web** y los **exploradores web**.
- En un principio no se tenían las páginas web, los usuarios debían conectarse y suscribirse a boletines para acceder e intercambiar información.
- Luego con las páginas web se permitían visualizar textos, gráficos e hipervínculos.

Contenido estático

- Aún contando con contenido más visual como gráficos e hipervínculos no era suficiente para explotar todo el potencial de Internet.

Contenido estático

- Aún contando con contenido más visual como gráficos e hipervínculos no era suficiente para explotar todo el potencial de Internet.
- Era muy difícil satisfacer las necesidades variadas de todos los usuarios.

Contenido estático

- Aún contando con contenido más visual como gráficos e hipervínculos no era suficiente para explotar todo el potencial de Internet.
- Era muy difícil satisfacer las necesidades variadas de todos los usuarios.
- Generalmente el contenido de Internet era plano y no llamaba mucho la atención de los usuarios.

Contenido estático

- Aún contando con contenido más visual como gráficos e hipervínculos no era suficiente para explotar todo el potencial de Internet.
- Era muy difícil satisfacer las necesidades variadas de todos los usuarios.
- Generalmente el contenido de Internet era plano y no llamaba mucho la atención de los usuarios.
- La actualización de contenidos era muy tediosa.

Contenido estático

- Aún contando con contenido más visual como gráficos e hipervínculos no era suficiente para explotar todo el potencial de Internet.
- Era muy difícil satisfacer las necesidades variadas de todos los usuarios.
- Generalmente el contenido de Internet era plano y no llamaba mucho la atención de los usuarios.
- La actualización de contenidos era muy tediosa.
- No existía personalización para cada usuario.

Contenido estático

- Aún contando con contenido más visual como gráficos e hipervínculos no era suficiente para explotar todo el potencial de Internet.
- Era muy difícil satisfacer las necesidades variadas de todos los usuarios.
- Generalmente el contenido de Internet era plano y no llamaba mucho la atención de los usuarios.
- La actualización de contenidos era muy tediosa.
- No existía personalización para cada usuario.
- ¿Alguien puede nombrar más desventajas?

Contenido dinámico

- Los carritos de compras, los motores de búsqueda, las redes sociales han cambiado completamente como utilizamos el Internet.

Contenido dinámico

- Los carritos de compras, los motores de búsqueda, las redes sociales han cambiado completamente como utilizamos el Internet.
- ¿Cuáles son los principales elementos qué han permitido la revolución del contenido de Internet?

Contenido dinámico

- Los carritos de compras, los motores de búsqueda, las redes sociales han cambiado completamente como utilizamos el Internet.
- ¿Cuáles son los principales elementos qué han permitido la revolución del contenido de Internet?
- JavaScript es unos de los más importantes. Permite más interacción con el usuario.

Contenido dinámico

- Los carritos de compras, los motores de búsqueda, las redes sociales han cambiado completamente como utilizamos el Internet.
- ¿Cuáles son los principales elementos qué han permitido la revolución del contenido de Internet?
- JavaScript es unos de los más importantes. Permite más interacción con el usuario.
- Ajax. Permite cargar contenido dinámico más rápido.

Contenido dinámico

- Los carritos de compras, los motores de búsqueda, las redes sociales han cambiado completamente como utilizamos el Internet.
- ¿Cuáles son los principales elementos qué han permitido la revolución del contenido de Internet?
- JavaScript es unos de los más importantes. Permite más interacción con el usuario.
- Ajax. Permite cargar contenido dinámico más rápido.
- HTML5 facilita la inserción de nuevos componentes.

Contenido dinámico

- Los carritos de compras, los motores de búsqueda, las redes sociales han cambiado completamente como utilizamos el Internet.
- ¿Cuáles son los principales elementos qué han permitido la revolución del contenido de Internet?
- JavaScript es unos de los más importantes. Permite más interacción con el usuario.
- Ajax. Permite cargar contenido dinámico más rápido.
- HTML5 facilita la inserción de nuevos componentes.
- CSS3. Permite dar estilos y animaciones.

Contenido dinámico

- Los carritos de compras, los motores de búsqueda, las redes sociales han cambiado completamente como utilizamos el Internet.
- ¿Cuáles son los principales elementos qué han permitido la revolución del contenido de Internet?
- JavaScript es unos de los más importantes. Permite más interacción con el usuario.
- Ajax. Permite cargar contenido dinámico más rápido.
- HTML5 facilita la inserción de nuevos componentes.
- CSS3. Permite dar estilos y animaciones.
- Las bases de datos.

Contenido dinámico

- Los carritos de compras, los motores de búsqueda, las redes sociales han cambiado completamente como utilizamos el Internet.
- ¿Cuáles son los principales elementos qué han permitido la revolución del contenido de Internet?
- JavaScript es unos de los más importantes. Permite más interacción con el usuario.
- Ajax. Permite cargar contenido dinámico más rápido.
- HTML5 facilita la inserción de nuevos componentes.
- CSS3. Permite dar estilos y animaciones.
- Las bases de datos.
- Otros elementos importante son: los applets de Java, Flash, JScript (variante de Microsoft de JS), ActiveX, WebGL (https://www.chromeexperiments.com/webgl)

Contenido dinámico

- Los carritos de compras, los motores de búsqueda, las redes sociales han cambiado completamente como utilizamos el Internet.
- ¿Cuáles son los principales elementos qué han permitido la revolución del contenido de Internet?
- JavaScript es unos de los más importantes. Permite más interacción con el usuario.
- Ajax. Permite cargar contenido dinámico más rápido.
- HTML5 facilita la inserción de nuevos componentes.
- CSS3. Permite dar estilos y animaciones.
- Las bases de datos.
- Otros elementos importante son: los applets de Java, Flash, JScript (variante de Microsoft de JS), ActiveX, WebGL (https://www.chromeexperiments.com/webgl)
- Lenguajes Script del lado del servidor (Perl, PHP, Python, etc) permiten insertar contenido dentro e otros archivos.

Contenido I

HTTP y HTML

- HTTP es un estándar de comunicación (protocolo).

HTTP y HTML

- HTTP es un estándar de comunicación (protocolo).
- Maneja las peticiones y repuestas entre clientes y servidores (exploradores y servidores web).

HTTP y HTML

- HTTP es un estándar de comunicación (protocolo).
- Maneja las peticiones y repuestas entre clientes y servidores (exploradores y servidores web).
- ¿Cuál es el trabajo del servidor?

HTTP y HTML

- HTTP es un estándar de comunicación (protocolo).
- Maneja las peticiones y repuestas entre clientes y servidores (exploradores y servidores web).
- ¿Cuál es el trabajo del servidor? aceptar peticiones de clientes y responder acorde a la solicitud,recursos como páginas web, imágenes, archivos, etc.

HTTP y HTML

- HTTP es un estándar de comunicación (protocolo).
- Maneja las peticiones y repuestas entre clientes y servidores (exploradores y servidores web).
- ¿Cuál es el trabajo del servidor? aceptar peticiones de clientes y responder acorde a la solicitud,recursos como páginas web, imágenes, archivos, etc.
- En el camino entre cliente y servidor normalmente tenemos otros dispositivos como routers, proxies, gateways, etc.

HTTP y HTML

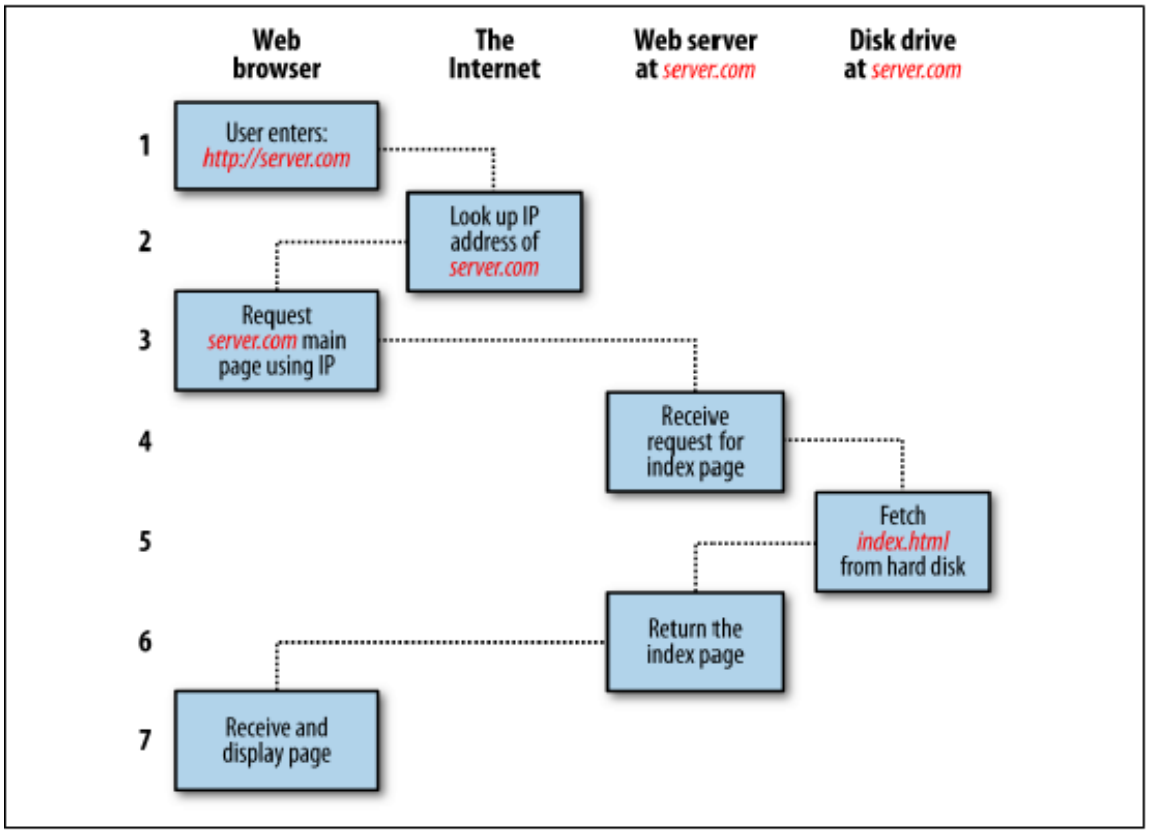
- HTTP es un estándar de comunicación (protocolo).
- Maneja las peticiones y repuestas entre clientes y servidores (exploradores y servidores web).
- ¿Cuál es el trabajo del servidor? aceptar peticiones de clientes y responder acorde a la solicitud,recursos como páginas web, imágenes, archivos, etc.
- En el camino entre cliente y servidor normalmente tenemos otros dispositivos como routers, proxies, gateways, etc.
- Un servidor normalmente puede aceptar múltiples peticiones de varios clientes. Y pasa el tiempo escuchando por un canal de comunicación hasta recibir peticiones.

El protocolo petición/repuesta

- Un explorador web realiza peticiones al servidor web que debe enviar como respuestas los recursos pedidos.

El protocolo petición/repuesta

- Un explorador web realiza peticiones al servidor web que debe enviar como respuestas los recursos pedidos.
- El cliente (explorador) se encarga de desplegar el contenido recibido (la página web).



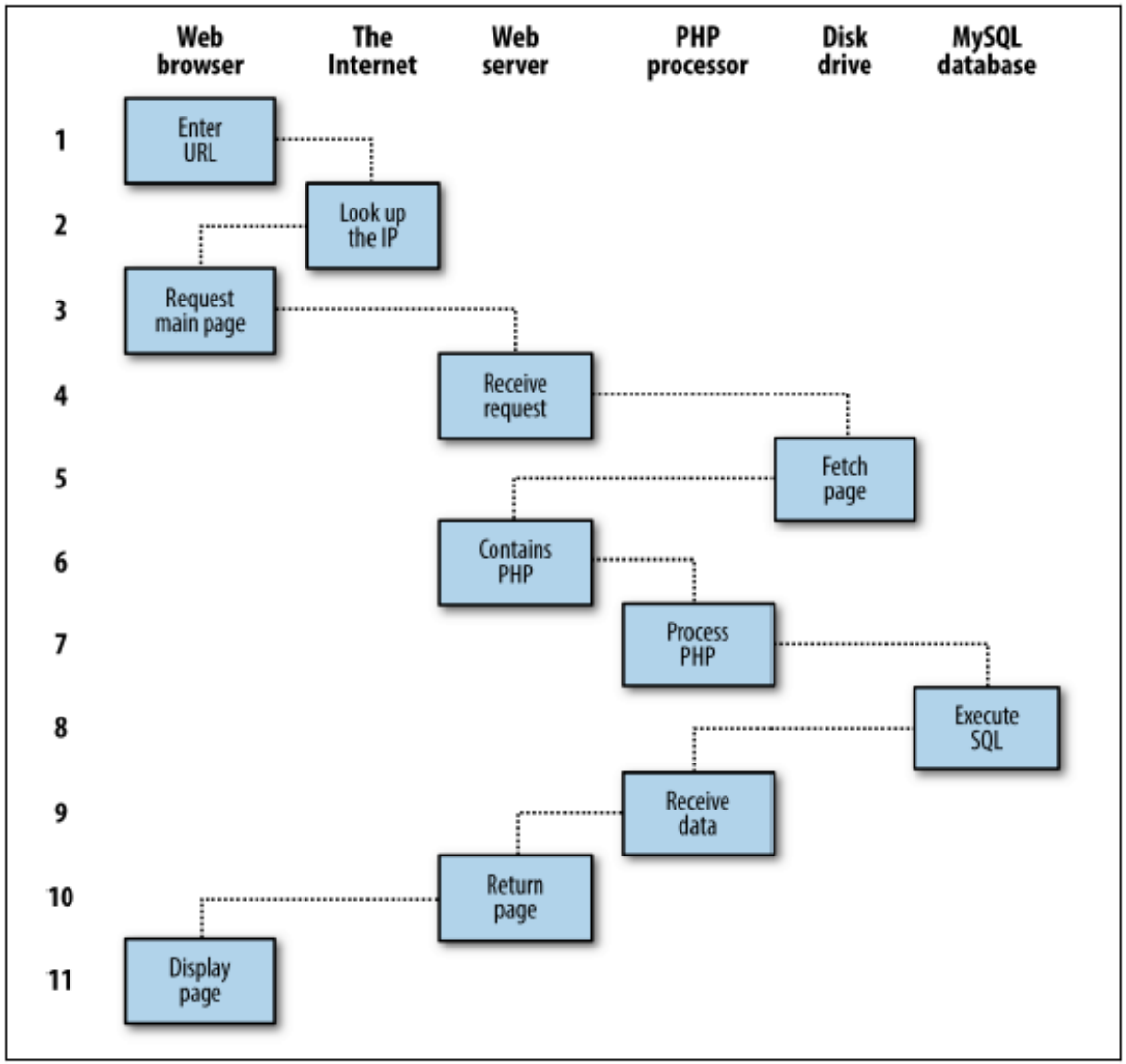
Protocolo petición/respuesta

- Este proceso se repite por cada uno de los componentes incluidos en la página web (figures, archivos .js, archivos .css, etc).

Protocolo petición/respuesta

- Este proceso se repite por cada uno de los componentes incluidos en la página web (figures, archivos .js, archivos .css, etc).
- En el caso de páginas dinámicas el proceso debe incluir el procesamiento de PHP y la BDD.

Protocolo petición/respuesta



Protocolo petición/respuesta

- Es importante conocer como funciona el protocolo como conocimiento general pero no lo necesitaremos al desarrollar nuestras aplicaciones web.

Protocolo petición/respuesta

- Es importante conocer como funciona el protocolo como conocimiento general pero no lo necesitaremos al desarrollar nuestras aplicaciones web.
- En el caso de que la página contenga JS este código será ejecuta localmente por el explorador.

Protocolo petición/respuesta

- Es importante conocer como funciona el protocolo como conocimiento general pero no lo necesitaremos al desarrollar nuestras aplicaciones web.
- En el caso de que la página contenga JS este código será ejecuta localmente por el explorador.
- Similarmente con los CSS.

Contenido I

Las herramientas que utilizaremos

- De todas las tecnologías que existen o existieron dentro de la web finalmente destacaron unas pocas que lideran actualmente.

Las herramientas que utilizaremos

- De todas las tecnologías que existen o existieron dentro de la web finalmente destacaron unas pocas que lideran actualmente.
- Estas son:

Las herramientas que utilizaremos

- De todas las tecnologías que existen o existieron dentro de la web finalmente destacaron unas pocas que lideran actualmente.
- Estas son:
  - **PHP** por su facilidad de programación, y conexiones simples a **MySQL**.

Las herramientas que utilizaremos

- De todas las tecnologías que existen o existieron dentro de la web finalmente destacaron unas pocas que lideran actualmente.
- Estas son:
  - **PHP** por su facilidad de programación, y conexiones simples a **MySQL**.
  - **JavaScript** que es esencial que cualquier página web hoy en día. Y permite modificar los **CSS** muy fácilmente.

Las herramientas que utilizaremos

- De todas las tecnologías que existen o existieron dentro de la web finalmente destacaron unas pocas que lideran actualmente.
- Estas son:
  - **PHP** por su facilidad de programación, y conexiones simples a **MySQL**.
  - **JavaScript** que es esencial que cualquier página web hoy en día. Y permite modificar los **CSS** muy fácilmente.
  - **HTML** y su reciente evolución a HTML5.

Las herramientas que utilizaremos

- De todas las tecnologías que existen o existieron dentro de la web finalmente destacaron unas pocas que lideran actualmente.
- Estas son:
  - **PHP** por su facilidad de programación, y conexiones simples a **MySQL**.
  - **JavaScript** que es esencial que cualquier página web hoy en día. Y permite modificar los **CSS** muy fácilmente.
  - **HTML** y su reciente evolución a HTML5.
  - **Ajax** que permite manipular datos y enviar peticiones al servidor en segundo plano. Sin que el usuario sepa que algo está pasando.

Las herramientas que utilizaremos

- De todas las tecnologías que existen o existieron dentro de la web finalmente destacaron unas pocas que lideran actualmente.
- Estas son:
  - **PHP** por su facilidad de programación, y conexiones simples a **MySQL**.
  - **JavaScript** que es esencial que cualquier página web hoy en día. Y permite modificar los **CSS** muy fácilmente.
  - **HTML** y su reciente evolución a HTML5.
  - **Ajax** que permite manipular datos y enviar peticiones al servidor en segundo plano. Sin que el usuario sepa que algo está pasando.
- La combinación de todos estas tecnologías no provee de las herramientas suficientes para desarrollar aplicaciones web muy dinámicas e interactivas.

Contenido I

PHP

- Nos permite introducir contenido dinámico en páginas web HTML.

PHP

- Nos permite introducir contenido dinámico en páginas web HTML.
- Un archivo con extensión `.php` tiene acceso automático al lenguaje.

PHP

- Nos permite introducir contenido dinámico en páginas web HTML.
- Un archivo con extensión `.php` tiene acceso automático al lenguaje.
- Desde un punto de vista del programador se podría escribir el siguiente código:

```
<?php
echo "Hoy es " . date("l") . ".";
?>
Les presentamos las noticias de hoy.
```

PHP

- Nos permite introducir contenido dinámico en páginas web HTML.
- Un archivo con extensión `.php` tiene acceso automático al lenguaje.
- Desde un punto de vista del programador se podría escribir el siguiente código:

```
<?php
echo "Hoy es " . date("l") . ".";
?>
Les presentamos las noticias de hoy.
```

- En el código podemos ver las etiquetas de apertura `<?php` que le dicen al servidor web que el código que sigue debe ser interpretado por el interpretador de PHP hasta la etiqueta de cierre `?>`

PHP

- Nos permite introducir contenido dinámico en páginas web HTML.
- Un archivo con extensión `.php` tiene acceso automático al lenguaje.
- Desde un punto de vista del programador se podría escribir el siguiente código:

```
<?php
echo "Hoy es " . date("l") . ".";
?>
Les presentamos las noticias de hoy.
```

- En el código podemos ver las etiquetas de apertura `<?php` que le dicen al servidor web que el código que sigue debe ser interpretado por el interpretador de PHP hasta la etiqueta de cierre `?>`
- El texto "Les presentamos las noticias de hoy." es enviado al explorador como HTML.

PHP

- PHP es un lenguaje muy flexible y gracias a eso ha tenido tanto éxito.

PHP

- PHP es un lenguaje muy flexible y gracias a eso ha tenido tanto éxito.
- Se puede mezclar HTML directamente con código PHP así:

```
Hoy es <?php echo date("l"); ?>. Les presentamos las
    noticias de hoy.
```

PHP

- PHP es un lenguaje muy flexible y gracias a eso ha tenido tanto éxito.
- Se puede mezclar HTML directamente con código PHP así:

```
Hoy es <?php echo date("l"); ?>. Les presentamos las
    noticias de hoy.
```

- Existen más maneras de dar formato y presentar información que veremos más adelante.

PHP

- PHP es un lenguaje muy flexible y gracias a eso ha tenido tanto éxito.
- Se puede mezclar HTML directamente con código PHP así:

```
Hoy es <?php echo date("l"); ?>. Les presentamos las
    noticias de hoy.
```

- Existen más maneras de dar formato y presentar información que veremos más adelante.
- El principal beneficio de PHP es la manera en la que se integra con HTML.

Contenido I

MySQL

- Muchos sitios web solían utilizar archivos de texto para almacenar datos (incluso usuarios y claves).

MySQL

- Muchos sitios web solían utilizar archivos de texto para almacenar datos (incluso usuarios y claves).
- Era difícil proteger estos archivos y eran propensos a sufrir daños por los accesos simultáneos.

MySQL

- Muchos sitios web solían utilizar archivos de texto para almacenar datos (incluso usuarios y claves).
- Era difícil proteger estos archivos y eran propensos a sufrir daños por los accesos simultáneos.
- Muchos problemas cuando los archivos crecían mucho al hacer consultas complejas.

MySQL

- Muchos sitios web solían utilizar archivos de texto para almacenar datos (incluso usuarios y claves).
- Era difícil proteger estos archivos y eran propensos a sufrir daños por los accesos simultáneos.
- Muchos problemas cuando los archivos crecían mucho al hacer consultas complejas.
- Por este motivo las BDD relacionales se han apoderado del mercado.

MySQL

- Muchos sitios web solían utilizar archivos de texto para almacenar datos (incluso usuarios y claves).
- Era difícil proteger estos archivos y eran propensos a sufrir daños por los accesos simultáneos.
- Muchos problemas cuando los archivos crecían mucho al hacer consultas complejas.
- Por este motivo las BDD relacionales se han apoderado del mercado.
- Dentro de estas BDD relacionales MySQL destaca sobretodo por ser muy potente, robusto, rápido y gratuito.

MySQL

- Un rápido ejemplo de uso de MySQL.

MySQL

- Un rápido ejemplo de uso de MySQL.
- Ingresar a http://localhost/phpmyadmin

MySQL

- Un rápido ejemplo de uso de MySQL.
- Ingresar a http://localhost/phpmyadmin
- Creamos una base de datos “`prueba`”

MySQL

- Un rápido ejemplo de uso de MySQL.
- Ingresar a http://localhost/phpmyadmin
- Creamos una base de datos “`prueba`”
- Creamos una tabla llamada “`usuarios`” con los campos:

MySQL

- Un rápido ejemplo de uso de MySQL.
- Ingresar a http://localhost/phpmyadmin
- Creamos una base de datos “`prueba`”
- Creamos una tabla llamada “`usuarios`” con los campos:
  - `apellido (varchar 30)`

MySQL

- Un rápido ejemplo de uso de MySQL.
- Ingresar a http://localhost/phpmyadmin
- Creamos una base de datos “`prueba`”
- Creamos una tabla llamada “`usuarios`” con los campos:
  - `apellido (varchar 30)`
  - `nombre (varchar 30)`

MySQL

- Un rápido ejemplo de uso de MySQL.
- Ingresar a http://localhost/phpmyadmin
- Creamos una base de datos "`prueba`"
- Creamos una tabla llamada "`usuarios`" con los campos:
  - `apellido (varchar 30)`
  - `nombre (varchar 30)`
  - `email (varchar 50)`

MySQL

- Un rápido ejemplo de uso de MySQL.
- Ingresar a http://localhost/phpmyadmin
- Creamos una base de datos "`prueba`"
- Creamos una tabla llamada "`usuarios`" con los campos:
  - `apellido (varchar 30)`
  - `nombre (varchar 30)`
  - `email (varchar 50)`
- Vamos a la pestaña "`SQL`" y ejecutamos:

```
INSERT INTO usuarios VALUES('Perez', 'Juan',
'jperez@mail.com');
```

MySQL

- Un rápido ejemplo de uso de MySQL.
- Ingresar a http://localhost/phpmyadmin
- Creamos una base de datos "`prueba`"
- Creamos una tabla llamada "`usuarios`" con los campos:
  - `apellido (varchar 30)`
  - `nombre (varchar 30)`
  - `email (varchar 50)`
- Vamos a la pestaña "`SQL`" y ejecutamos:
  ```
  INSERT INTO usuarios VALUES('Perez', 'Juan',
  'jperez@mail.com');
  ```
- En los siguientes capítulos veremos más detalles sobre MySQL pero es tan simple como ejecutar sentencias SQL.

MySQL

- Un rápido ejemplo de uso de MySQL.
- Ingresar a http://localhost/phpmyadmin
- Creamos una base de datos "`prueba`"
- Creamos una tabla llamada "`usuarios`" con los campos:
  - `apellido (varchar 30)`
  - `nombre (varchar 30)`
  - `email (varchar 50)`
- Vamos a la pestaña "`SQL`" y ejecutamos:
  ```
  INSERT INTO usuarios VALUES('Perez', 'Juan',
  'jperez@mail.com');
  ```
- En los siguientes capítulos veremos más detalles sobre MySQL pero es tan simple como ejecutar sentencias SQL.
- Para buscar un datos ejecutamos:
  ```
  SELECT apellido,nombre FROM usuarios WHERE
  email='jperez@mail.com';
  ```

MySQL

- Podemos ejecutar muchos más comandos SQL en MySQL: `JOIN`, `ORDERBY`, buscar por coincidencias parciales de texto, etc.

MySQL

- Podemos ejecutar muchos más comandos SQL en MySQL: `JOIN`, `ORDERBY`, buscar por coincidencias parciales de texto, etc.
- Al utilizar PHP podemos ejecutar todos estos comandos sin necesidad de interactuar con MySQL directamente.

MySQL

- Podemos ejecutar muchos más comandos SQL en MySQL: `JOIN`, `ORDERBY`, buscar por coincidencias parciales de texto, etc.
- Al utilizar PHP podemos ejecutar todos estos comandos sin necesidad de interactuar con MySQL directamente.
- Podríamos utilizar arreglos de PHP (`array`) para mantener los registros, modificarlos, realizar búsquedas, etc directamente desde PHP.

Contenido I

JavaScript

- Provee una interacción dinámica con el usuario: validación de datos como email, textos, mostrar mensajes de confirmación, etc.

JavaScript

- Provee una interacción dinámica con el usuario: validación de datos como email, textos, mostrar mensajes de confirmación, etc.
- Sin embargo todas las validaciones realizadas con JS deben estar respaldadas con validaciones en el lado del servidor.

JavaScript

- Provee una interacción dinámica con el usuario: validación de datos como email, textos, mostrar mensajes de confirmación, etc.
- Sin embargo todas las validaciones realizadas con JS deben estar respaldadas con validaciones en el lado del servidor.
- Al combinar JS con CSS tenemos la herramienta perfecta para crear páginas web dinámicas e interactivas sin necesidad de interactuar con el servidor.

JavaScript

- Provee una interacción dinámica con el usuario: validación de datos como email, textos, mostrar mensajes de confirmación, etc.
- Sin embargo todas las validaciones realizadas con JS deben estar respaldadas con validaciones en el lado del servidor.
- Al combinar JS con CSS tenemos la herramienta perfecta para crear páginas web dinámicas e interactivas sin necesidad de interactuar con el servidor.
- La incompatibilidad entre exploradores es el mayor problema con el que nos toparemos al utilizar JS (Algunos añadieron funcionalidades extra)

JavaScript

- Provee una interacción dinámica con el usuario: validación de datos como email, textos, mostrar mensajes de confirmación, etc.
- Sin embargo todas las validaciones realizadas con JS deben estar respaldadas con validaciones en el lado del servidor.
- Al combinar JS con CSS tenemos la herramienta perfecta para crear páginas web dinámicas e interactivas sin necesidad de interactuar con el servidor.
- La incompatibilidad entre exploradores es el mayor problema con el que nos toparemos al utilizar JS (Algunos añadieron funcionalidades extra)
- Estos problemas de van desapareciendo gracias a los estándares que deben seguir los exploradores web. Versiones antiguas aún en funcionamiento.

JavaScript

- Provee una interacción dinámica con el usuario: validación de datos como email, textos, mostrar mensajes de confirmación, etc.
- Sin embargo todas las validaciones realizadas con JS deben estar respaldadas con validaciones en el lado del servidor.
- Al combinar JS con CSS tenemos la herramienta perfecta para crear páginas web dinámicas e interactivas sin necesidad de interactuar con el servidor.
- La incompatibilidad entre exploradores es el mayor problema con el que nos toparemos al utilizar JS (Algunos añadieron funcionalidades extra)
- Estos problemas de van desapareciendo gracias a los estándares que deben seguir los exploradores web. Versiones antiguas aún en funcionamiento.
- Estos problemas son menores hoy en día gracias a las librerías JS existentes como veremos más adelante con jQuery.

Ejemplo JS

- Un pequeño ejemplo JS, dentro del mismo archivo `index.php` al final escribimos el siguiente código:

```
<script type="text/javascript">
  document.write("Today is " + Date() );
</script>
```

Contenido I

Ajax

- JS fue creado especialmente para aumentar la interacción con el usuario pero hoy en día se lo utiliza mucho para aplicaciones que utilizan **Ajax** *Asynchronous JavaScript and XML* (aunque ya esta definición está desactualizada).

Ajax

- JS fue creado especialmente para aumentar la interacción con el usuario pero hoy en día se lo utiliza mucho para aplicaciones que utilizan **Ajax** *Asynchronous JavaScript and XML* (aunque ya esta definición está desactualizada).
- Hoy en día no se utiliza `XML` con Ajax. Ahora los mensajes son enviados mediante `JSON`.

Ajax

- JS fue creado especialmente para aumentar la interacción con el usuario pero hoy en día se lo utiliza mucho para aplicaciones que utilizan **Ajax** *Asynchronous JavaScript and XML* (aunque ya esta definición está desactualizada).
- Hoy en día no se utiliza `XML` con Ajax. Ahora los mensajes son enviados mediante `JSON`.
- Ajax a sido clave para el paso a la Web 2.0, donde las páginas web se han vuelto prácticamente programas independientes.

Ajax

- JS fue creado especialmente para aumentar la interacción con el usuario pero hoy en día se lo utiliza mucho para aplicaciones que utilizan **Ajax** *Asynchronous JavaScript and XML* (aunque ya esta definición está desactualizada).
- Hoy en día no se utiliza `XML` con Ajax. Ahora los mensajes son enviados mediante `JSON`.
- Ajax a sido clave para el paso a la Web 2.0, donde las páginas web se han vuelto prácticamente programas independientes.
- Una llamada Ajax es capaz de cargar contenido y actualizar un solo elemento de la página web como un `div, tabla, p, a, etc`. Ej. cargar una foto, reemplazar un botón, mostrar una barra de progreso, etc.

Contenido I

CSS

- El desarrollo del estándar CSS3 ha permitido de los CSS ofrezcan un mayor nivel de interactividad que antes era posible solo con JS.

CSS

- El desarrollo del estándar CSS3 ha permitido de los CSS ofrezcan un mayor nivel de interactividad que antes era posible solo con JS.
- CSS3 ofrece no solo la opción de cambiar tamaño, color, bordes, espaciado si no que permite realizar animaciones de transiciones y transformaciones a los elementos.

CSS

- El desarrollo del estándar CSS3 ha permitido de los CSS ofrezcan un mayor nivel de interactividad que antes era posible solo con JS.
- CSS3 ofrece no solo la opción de cambiar tamaño, color, bordes, espaciado si no que permite realizar animaciones de transiciones y transformaciones a los elementos.
- Un ejemplo de CSS (podemos añadir un elemento <p> a nuestro index.php) y añadir el siguiente código al inicio:

```
<style>
  p {
  text-align:justify;
  font-family:Helvetica;
  color: red;
  }
</style>
```

CSS

- Los CSS pueden escribirse directamente en las etiquetas HTML o escribir un archivo aparte con las reglas de estilo e incluir el archivo en la cabecera del HTML (esta es la mejor opción).

CSS

- Los CSS pueden escribirse directamente en las etiquetas HTML o escribir un archivo aparte con las reglas de estilo e incluir el archivo en la cabecera del HTML (esta es la mejor opción).
- Con CSS se pueden manejar eventos del mouse para cambiar los estilos de los elementos.

CSS

- Los CSS pueden escribirse directamente en las etiquetas HTML o escribir un archivo aparte con las reglas de estilo e incluir el archivo en la cabecera del HTML (esta es la mejor opción).
- Con CSS se pueden manejar eventos del mouse para cambiar los estilos de los elementos.
- Se puede acceder a los estilos CSS a través de JS.

Contenido I

HTML5

- Aunque todos los estándares y elementos que hemos mencionado son de gran utilidad, no ha bastado para las exigencias de los usuarios.

HTML5

- Aunque todos los estándares y elementos que hemos mencionado son de gran utilidad, no ha bastado para las exigencias de los usuarios.
- Se necesitaban mayor facilidad para manipular gráficos en un explorador sin necesidad de instalar plug-ins como Flash.

HTML5

- Aunque todos los estándares y elementos que hemos mencionado son de gran utilidad, no ha bastado para las exigencias de los usuarios.
- Se necesitaban mayor facilidad para manipular gráficos en un explorador sin necesidad de instalar plug-ins como Flash.
- Otro problema era la inserción de audio y video en las páginas web.

HTML5

- Aunque todos los estándares y elementos que hemos mencionado son de gran utilidad, no ha bastado para las exigencias de los usuarios.
- Se necesitaban mayor facilidad para manipular gráficos en un explorador sin necesidad de instalar plug-ins como Flash.
- Otro problema era la inserción de audio y video en las páginas web.
- Todos estos problemas han llevado a una evolución del HTML.

HTML5

- Aunque todos los estándares y elementos que hemos mencionado son de gran utilidad, no ha bastado para las exigencias de los usuarios.
- Se necesitaban mayor facilidad para manipular gráficos en un explorador sin necesidad de instalar plug-ins como Flash.
- Otro problema era la inserción de audio y video en las páginas web.
- Todos estos problemas han llevado a una evolución del HTML.
- El desarrollo de HTML5 empezó en 2004 y fue liderado por Mozilla Foundation y Opera Software.

HTML5

- Aunque todos los estándares y elementos que hemos mencionado son de gran utilidad, no ha bastado para las exigencias de los usuarios.
- Se necesitaban mayor facilidad para manipular gráficos en un explorador sin necesidad de instalar plug-ins como Flash.
- Otro problema era la inserción de audio y video en las páginas web.
- Todos estos problemas han llevado a una evolución del HTML.
- El desarrollo de HTML5 empezó en 2004 y fue liderado por Mozilla Foundation y Opera Software.
- Recién en 2013 se emitió un documento final a la W3C (el encargado de controlar los estándares de la web).

HTML5

- Aunque todos los estándares y elementos que hemos mencionado son de gran utilidad, no ha bastado para las exigencias de los usuarios.
- Se necesitaban mayor facilidad para manipular gráficos en un explorador sin necesidad de instalar plug-ins como Flash.
- Otro problema era la inserción de audio y video en las páginas web.
- Todos estos problemas han llevado a una evolución del HTML.
- El desarrollo de HTML5 empezó en 2004 y fue liderado por Mozilla Foundation y Opera Software.
- Recién en 2013 se emitió un documento final a la W3C (el encargado de controlar los estándares de la web).
- El desarrollo de nuevos estándares y mejoras de las especificaciones es continuo y es un ciclo sin fin. Pero por ahora todos los sitios web deben ser compatibles con HTML5.

Novedades en HTML5

- *Markup* nuevos elementos como `<nav>` y `<footer>` y elementos obsoletos como `<font>` y `<center>`.

Novedades en HTML5

- *Markup* nuevos elementos como `<nav>` y `<footer>` y elementos obsoletos como `<font>` y `<center>`.
- *Nuevas APIs* como `<camvas>` (para escribir y dibujar), `<audio>` y `<video>`, aplicaciones web offline, microdata y almacenamiento local.

Novedades en HTML5

- *Markup* nuevos elementos como `<nav>` y `<footer>` y elementos obsoletos como `<font>` y `<center>`.
- *Nuevas APIs* como `<camvas>` (para escribir y dibujar), `<audio>` y `<video>`, aplicaciones web offline, microdata y almacenamiento local.
- *Aplicaciones* incluye nuevas tecnologías de renderizado: MathML (Math Markup Language) para fórmulas y `SVG` (Scalable Vector Graphics) para crear elementos fuera del `<canvas>`.

Novedades en HTML5

- *Markup* nuevos elementos como `<nav>` y `<footer>` y elementos obsoletos como `<font>` y `<center>`.
- *Nuevas APIs* como `<camvas>` (para escribir y dibujar), `<audio>` y `<video>`, aplicaciones web offline, microdata y almacenamiento local.
- *Aplicaciones* incluye nuevas tecnologías de renderizado: MathML (Math Markup Language) para fórmulas y `SVG` (Scalable Vector Graphics) para crear elementos fuera del `<canvas>`.
- Con HTML5 ya no es necesario utilizar las etiquetas de compatibilidad con XHTML. Es decir los elementos que no tienen etiquetas de cerrado como el `<br>` ya no necesitan escribirse así: `<br />`.

Contenido I

Otras características de Apache

- Apache no solamente sirve como un servidor web que provee recursos HTML.

Otras características de Apache

- Apache no solamente sirve como un servidor web que provee recursos HTML.
- Es capaz de manejar documentos como imágenes, archivos Flash, MP3, RSS, etc.

Otras características de Apache

- Apache no solamente sirve como un servidor web que provee recursos HTML.
- Es capaz de manejar documentos como imágenes, archivos Flash, MP3, RSS, etc.
- Apache y PHP proporcionan también módulos precompilados que pueden ser llamados en tiempo de ejecución para generar imágenes u otros archivos. Ej: GD (Graphics Draw).

Otras características de Apache

- Apache no solamente sirve como un servidor web que provee recursos HTML.
- Es capaz de manejar documentos como imágenes, archivos Flash, MP3, RSS, etc.
- Apache y PHP proporcionan también módulos precompilados que pueden ser llamados en tiempo de ejecución para generar imágenes u otros archivos. Ej: GD (Graphics Draw).
- Módulos importantes que proporciona PHP es el de seguridad, Rewrite (opción para el servidor web de manejar tipos de URL y reescribirlos a un estándar interno).

Contenido I

Contenido II

El código abierto

- Herramientas como PHP, MySQL y Apache son los más populares dentro de sus categorías.

El código abierto

- Herramientas como PHP, MySQL y Apache son los más populares dentro de sus categorías.
- Cabe destacar que son de código abierto y han sido desarrollados por comunidades de programadores.

El código abierto

- Herramientas como PHP, MySQL y Apache son los más populares dentro de sus categorías.
- Cabe destacar que son de código abierto y han sido desarrollados por comunidades de programadores.
- ¿Cuál es la principal ventaja del código abierto?

El código abierto

- Herramientas como PHP, MySQL y Apache son los más populares dentro de sus categorías.
- Cabe destacar que son de código abierto y han sido desarrollados por comunidades de programadores.
- ¿Cuál es la principal ventaja del código abierto?
- Se detectan bugs rápidamente antes de que representen un peligro.

El código abierto

- Herramientas como PHP, MySQL y Apache son los más populares dentro de sus categorías.
- Cabe destacar que son de código abierto y han sido desarrollados por comunidades de programadores.
- ¿Cuál es la principal ventaja del código abierto?
- Se detectan bugs rápidamente antes de que representen un peligro.
- En este caso, las herramientas que utilizaremos son todas gratis.

Contenido I

Contenido II

Ejercicio

- Presentar ejemplos de sitios web que utilicen Ajax y explicar su funcionamiento.

Ejercicio

- Presentar ejemplos de sitios web que utilicen Ajax y explicar su funcionamiento.
- Escribir una página web utilizando todos los recursos que conozcan (HTML, HTML5, PHP, MySQL, JS, CSS, Ajax).