U-tad
CENTRO UNIVERSITARIO DE TECNOLOGÍA Y ARTE DIGITAL

Tema 2. Tecnologías de desarrollo de aplicaciones web

Programación web

Boni García

Curso 2016/2017

Índice

Índice

Introducción

- El impacto de la Web ha propiciado la aparición de una gran cantidad de tecnologías, librerías, herramientas y estilos arquitectónicos para desarrollar una aplicación web
- Es conveniente conocer los elementos más importantes desde un punto de vista de alto nivel para tener una visión global de la programación web
- Existen dos enfoques principales en el desarrollo de aplicaciones web:
  - Creación de aplicaciones web con **tecnologías de desarrollo**
  - Creación de aplicaciones web con **sistemas gestores de contenido**

Introducción

- En las tecnologías de desarrollo, habrá que tener en cuenta:
  - **Lado cliente** (*frontend*). Tecnologías que permiten crear interfaces de usuario atractivos y permiten la comunicación con el servidor. Basadas en HTML, CSS y JavaScript.
  - **Lado servidor** (*backend*). Tecnologías que permiten implementar la lógica de negocio así como la integración con otros servicios (típicamente con un servidor de base datos)
- Existen aplicaciones web cuya principal funcionalidad es la publicación de contenido: **CMS** (*Content Management System*)
- En los últimos tiempos se ha popularizado el uso de **lenguajes de marcados ligeros** para la creación de sitios web estáticos

Índice

Arquitecturas de aplicaciones web

- Existen diferentes **arquitectura** de aplicación web en función de las tecnologías que usan y cómo se usan:

Arquitectura	Cliente	Servidor
Página web estática	Estático. HTML y CSS	Estático. Recursos en disco duro
Página web interactiva	Dinámico. JavaScript	Estático. Recursos en disco duro
Aplicación web con cliente estático	Estático. HTML y CSS	Dinámico. Ejecución Código
Aplicación web interactiva	Dinámico. JavaScript	Dinámico. Ejecución Código
Aplicación web con AJAX	Dinámico. JavaScript	Dinámico. Ejecución Código
Aplicación web SPA	Dinámico. JavaScript	Dinámico. Ejecución Código

Arquitecturas de aplicaciones web

Página web estática

- El navegador hace petición al servidor mediante HTTP
- El servidor transforma URL a ruta en disco
- El servidor devuelve el fichero de disco al navegador
- El navegador visualiza (*renderiza*) la página HTML con estilos CSS
- Cuando el usuario hace clic en un enlace, el navegador repite el proceso con la URL del link y recarga por completo la página web
- Con esta arquitectura el servidor siempre devuelve los mismos recursos
- Se suele usar esta arquitectura se usa para:
  - Páginas personales / proyectos
  - Documentación técnica

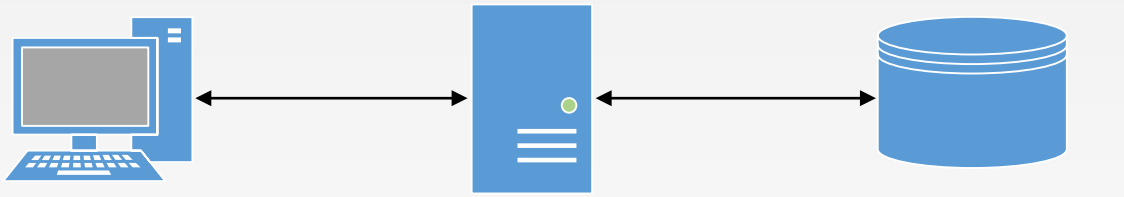
Arquitecturas de aplicaciones web

Página web interactiva

- El contenido de la página web está alojado en el disco duro del servidor (estático)
- El cliente es dinámico porque las páginas incluyen código **JavaScript** que se ejecuta en el navegador
- Este JavaScript se usa para incluir efectos gráficos:
  - Efectos gráficos que no se pueden implementar con CSS
  - Mostrar u ocultar información en función de los elementos que se seleccionan (para documentos largos)
  - Menús desplegables
  - Páginas adaptables para móviles (*responsive*)

Arquitecturas de aplicaciones web

Aplicación web con cliente estático



- Es un ejemplo de arquitectura de 3 capas:
  - Navegador: Capa de presentación
  - Servidor web: Capa de aplicación (lógica de negocio)
  - Base de datos: Capa de datos

Arquitecturas de aplicaciones web

Aplicación web con cliente estático

- Cuando el servidor web recibe una petición, dependiendo de la URL:
  - Devolver contenido del disco
  - Ejecutar código para generar el recurso dinámicamente
- Cuando se ejecuta código, normalmente se hacen consultas a una base de datos para recuperar la información
- Lo más habitual es que se genere la página HTML de forma dinámica (pero también puede generar imágenes, PDFs, etc…)
- El contenido es dinámico, porque se **ejecuta código en el servidor** para generar dicho contenido

Arquitecturas de aplicaciones web

- La mayoría de las aplicaciones web actuales son **dinámicas** tanto en **cliente** como en **servidor**
- Dependiendo de cómo se use el **JavaScript** en el cliente se diferencian tres arquitecturas:
  - Aplicación web **interactiva**
  - Aplicación web con **AJAX**
  - Aplicación web **SPA**

Arquitecturas de aplicaciones web

Aplicación web interactiva

- El JavaScript se utiliza para crear efectos gráficos
- El dinamismo en el cliente se utiliza exactamente igual que en las páginas web interactivas
- JavaScript se diseñó, entre otras cosas, para añadir efectos gráficos básicos a las páginas cuando el CSS era muy limitado
- La gran mayoría de las aplicaciones web que existen en Internet siguen esta arquitectura

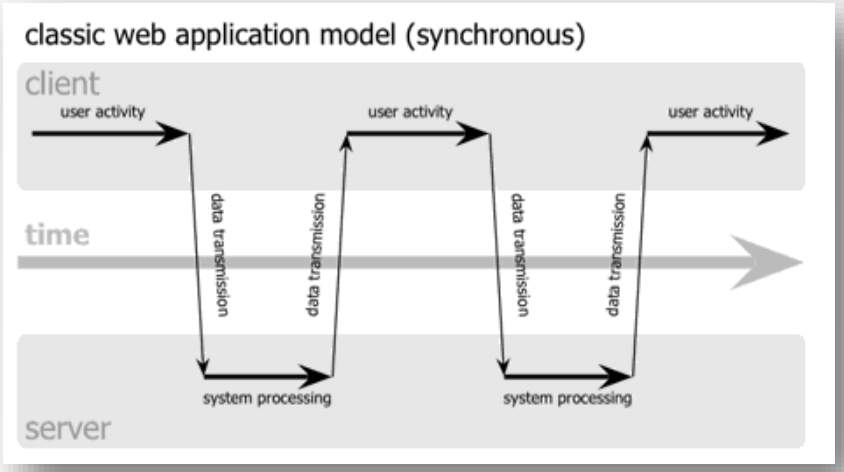
Arquitecturas de aplicaciones web

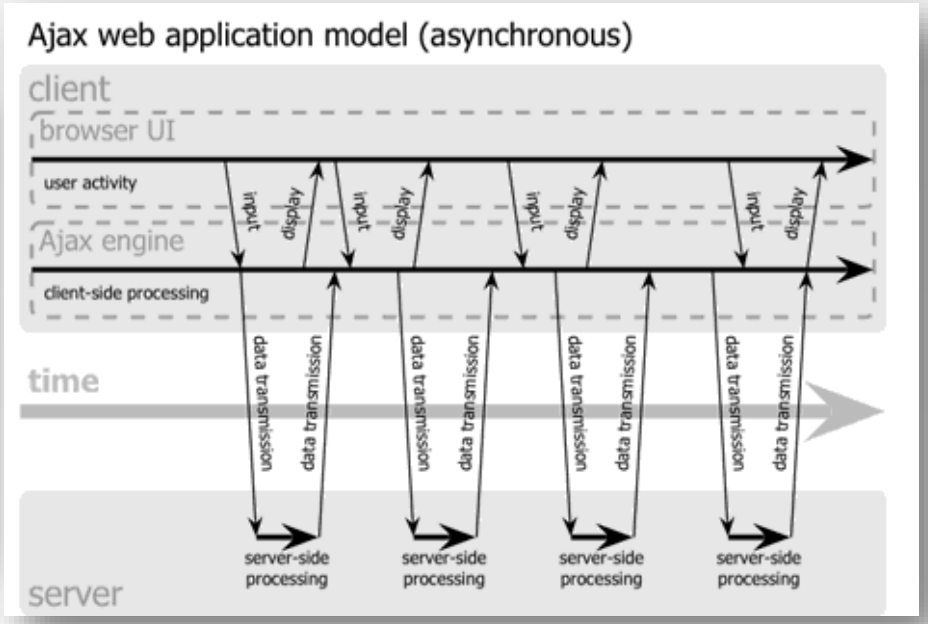
Aplicación web con AJAX

- JavaScript se usa para no tener que recargar completamente la página al pulsar un link
- Permite hacer petición al servidor web en segundo plano (oculta al usuario)
- Cuando llega al navegador el resultado de la petición, el código JavaScript actualiza aquellas partes de la página necesarias
- A esta técnica se la conoce como **AJAX** (*Asynchronous JavaScript And XML*)

Arquitecturas de aplicaciones web

Aplicación web con AJAX





Arquitecturas de aplicaciones web

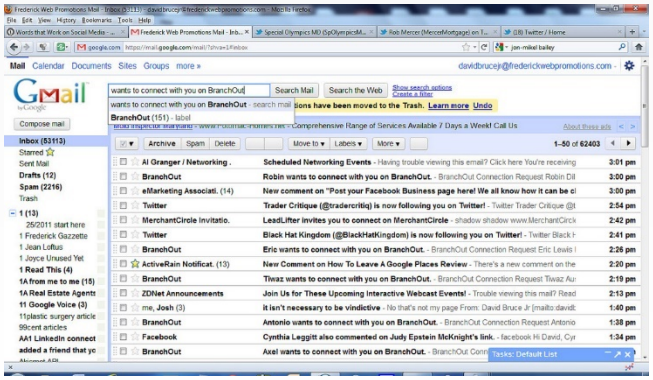
Aplicación web con AJAX

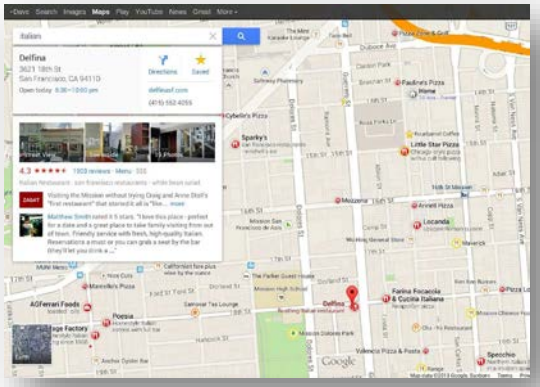
- Usar AJAX en una página mejora mucho la **experiencia de usuario**
- No es necesario recargar la página al completo, sólo aquellas partes que cambian (por ejemplo se puede dejar el menú fijo)
- La página se puede cargar por partes, primero la información importante y en segundo plano otros elementos complementarios (por ejemplo los botones de compartir, los comentarios en un blog…)
- Se puede dar realimentación al usuario de formas más adecuadas (cuadro de diálogo, error de validación en un formulario, quitar el icono de carga de un recurso, etc…)

Arquitecturas de aplicaciones web

Aplicación web SPA

- Las aplicaciones que siguen la arquitectura **SPA (*Single Page Application*)** llevan la técnica **AJAX** al extremo ya que todo el contenido se carga con **JavaScript en segundo plano**
- Existe una **única página** cuyo contenido va cambiando según el usuario interactúa con botones, pestañas, etc
- Google popularizó AJAX y SPA con Gmail y Maps





Índice

Tecnologías del cliente

Estándares web

- El cliente web por excelencia es el **navegador web**
- Existen un conjunto de **estándares web**, definidos por el **W3C**, que todo navegador debería implementar
- La familia de estándares W3C comprende:
  - HTML/CSS
  - Ajax
  - Servicios web
  - XML
  - Accesibilidad
  - ...

http://www.w3.org

Tecnologías del cliente

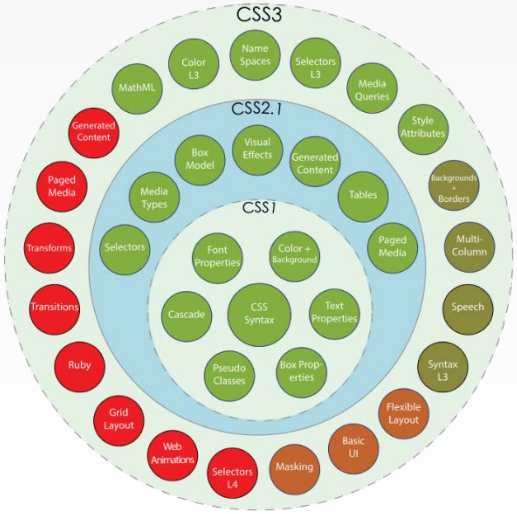
Estándares web - HTML

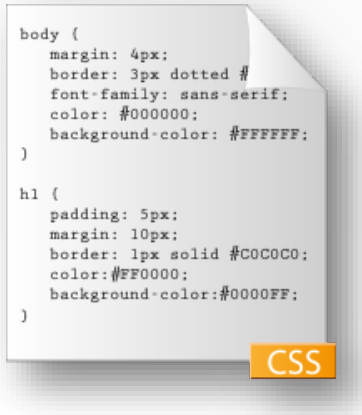
- La versión actual es HTML 5 (HTML 5.1 está en desarrollo)
- Ha supuesto una revolución para el dinamismo en el cliente porque ofrece muchas librerías/tecnologías avanzadas:
  - Multimedia: etiquetas vídeo, audio y canvas
  - Comunicaciones: websockets
  - Concurrencia: webworkers

Tecnologías del cliente

Estándares web - CSS

- CSS es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML
- Su versión actual es CSS 3





Tecnologías del cliente

Estándares web - JavaScript

- Las páginas web se pueden dinamizar con JavaScript
- Se puede modificar la página y ejecutar código cuando se interactúa con ella mediante la API DOM (*Document Object Model*)
- JavaScript es un lenguaje de programación basado en el estándar **ECMAScript** de ECMA (otra organización diferente al W3C)
- Hay ligeras diferencias en la implementación de JavaScript de los navegadores, aunque actualmente todos son bastante compatibles entre sí
- Aunque algunos elementos de la sintaxis recuerden a Java, el lenguaje es muy diferente a Java. El nombre JavaScript se eligió al publicar el lenguaje en una época en la que Java estaba en auge y fue principalmente por marketing

http://www.ecma-international.org/

Tecnologías del cliente

Librerías JavaScript

- Existen multitud de librerías/frameworks JavaScript para el desarrollo de aplicaciones. Algunas de las más populares:
  - **jQuery**: es un recubrimiento de la API DOM que aporta facilidad de uso, potencia y compatibilidad entre navegadores. Se usa para gestionar el interfaz (la página) y para peticiones **AJAX**
  - **Angular**: es un framework para la creación de aplicaciones SPA
  - **React**: librería para la creación de aplicaciones SPA

Tecnologías del cliente

Tecnologías no estándar en la Web

- Adobe Flash
  - Es una tecnología propietaria y cerrada
  - Es gratuita para los usuarios, pero los desarrolladores y servidores que usen ciertas características tienen que pagar licencia
  - Es una tecnología usada principalmente para incrustar contenido multimedia interactivo en páginas web
  - Durante muchos años fue la única forma de tener interactividad, animaciones, vídeos, juegos… en la Web
  - Fue acusada de que no era eficiente, ni abierta y por tanto, no es el futuro de la Web (Steve Jobs, Abril 2010)
  - Adobe lo acabó reconociendo y no la desarrolló más (Nov 2011)

Tecnologías del cliente

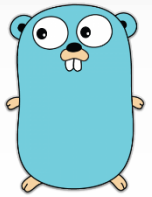
Conclusiones

- Si no hay un motivo importante, todas las aplicaciones web deberían implementarse con **estándares**
- En un mundo con multitud de dispositivos conectados a la red, es la única forma de la web sea accesible desde todos ellos
- HTML5 se ha convertido en la tecnología estándar para multitud de plataformas diferentes
- Para saber qué estándares soporta cada versión de cada navegador, se puede usar la web http://caniuse.com/

Índice

Tecnologías del servidor

- Los estándares son muy importantes en los navegadores web porque la web tiene que ser compatible con cualquier dispositivo
- En cambio los estándares no son necesarios en el servidor, porque cada organización desarrollará su aplicación en el servidor con la tecnología de su elección
- En el servidor, se pueden usar multitud de tecnologías



Tecnologías del servidor

Java Enterprise Edition

- 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
- Existen comunidades de desarrolladores y empresas que realizan complementos, bibliotecas, herramientas…

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

Tecnologías del servidor

Java Enterprise Edition

- Estándares en Java EE
  - Java tiene una organización que define estándares abiertos que cualquier empresa u organización puede implementar
  - Existen muchos estándares e implementaciones: Java EE, Servlets, JSP, JDBC, JPA, JSF, EJBs…
- Frameworks en Java EE
  - Existen multitud de implementaciones independientes de librerías y frameworks
  - **Spring** es el framework de desarrollo de aplicaciones empresariales basado en tecnologías Java más popular

Tecnologías del servidor

PHP

- Desarrollado en 1994 por Rasmus Lerdorf
- Fue una de las primeras **tecnologías libres** que se popularizaron para desarrollo web
- Tecnología con un lenguaje propio llamado PHP
- Desarrollada por PHP Group con licencia libre PHP license
- Es la tecnología de programación que más sitios activos tiene en Internet
- Se integra normalmente con Apache y MySQL en entornos Linux en un paquete llamado **LAMP**
- Facebook es sin duda una muestra importante de la popularidad de PHP
- CMSs como Drupal y Wordpress también están implementados en PHP

http://www.php.net/

Tecnologías del servidor

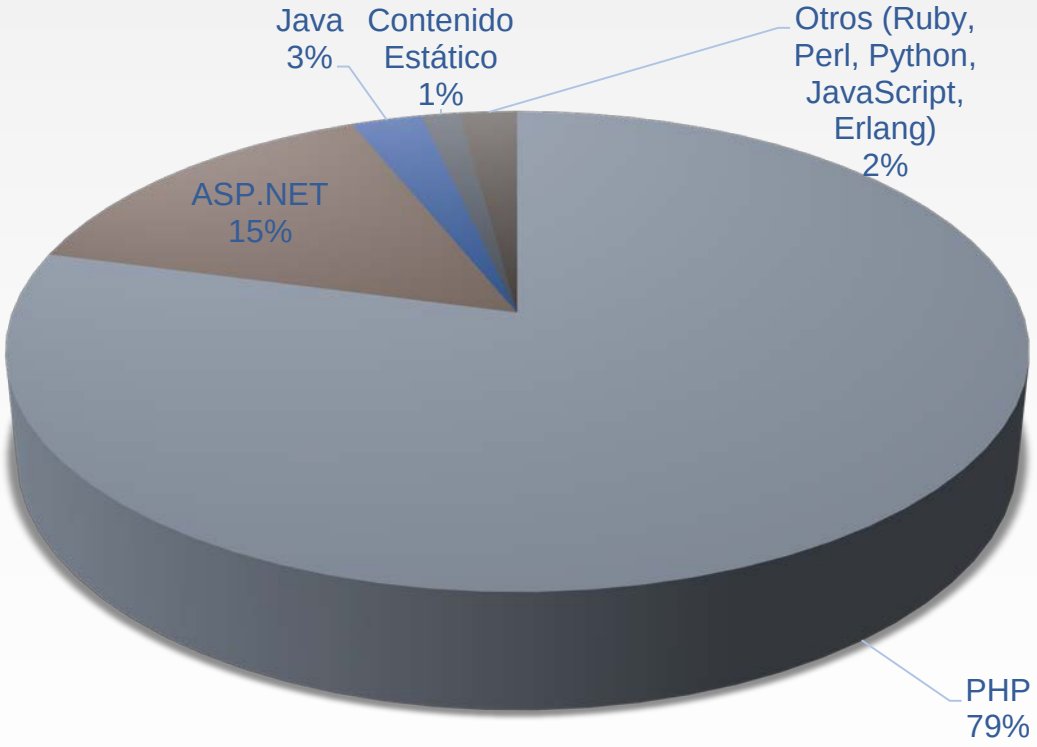
ASP.NET

- Versión evolucionada del ASP clásico
- Integrada en la tecnología **.NET** de Microsoft junto con el lenguaje **C#**
- Licencia propietaria y para plataformas **Windows**
- Tiene una comunidad de desarrolladores más limitada que las otras alternativas

http://www.asp.net/

Tecnologías del servidor

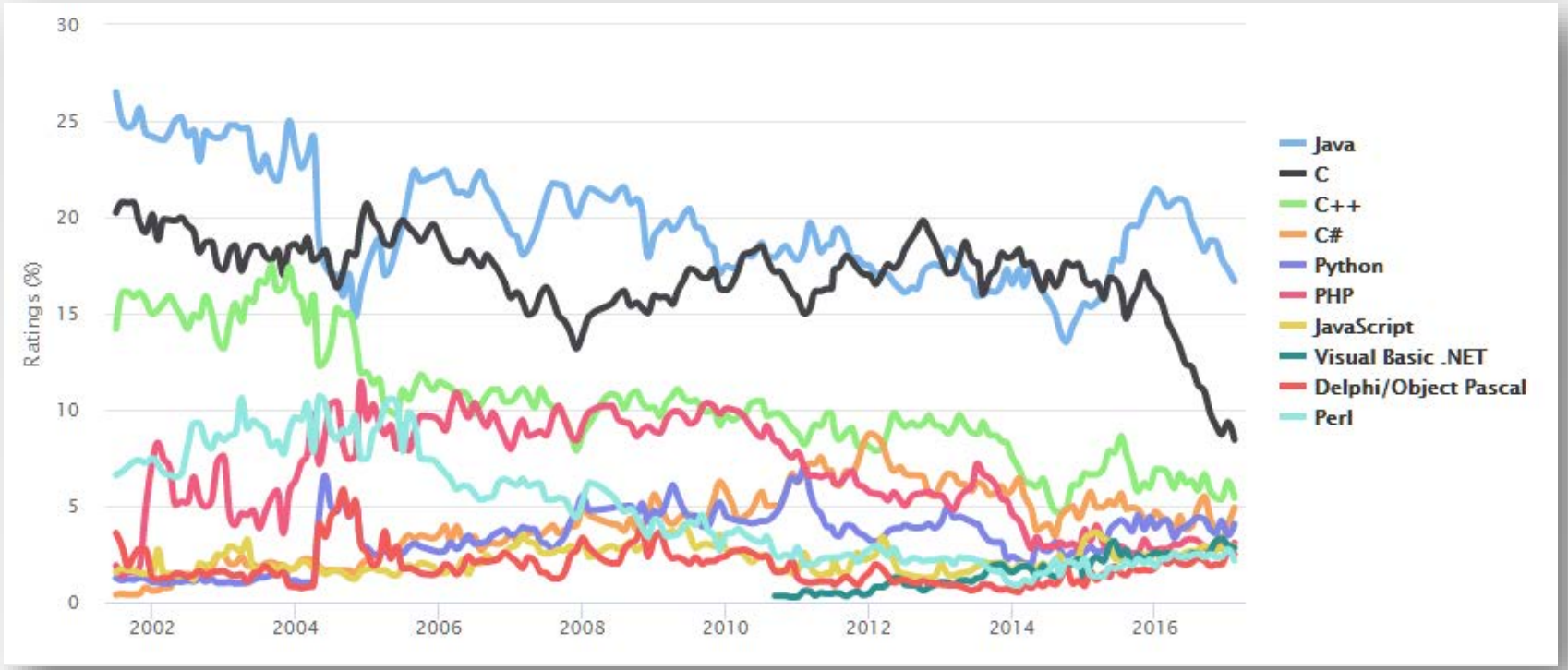
- Cuota de uso tecnologías del servidor (febrero 2017):



http://w3techs.com/technologies/overview/programming_language/all

Tecnologías del servidor

- Índice TIOBE (febrero 2017):



http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

Índice

Bases de datos

- Base de datos = conjunto ordenado de datos
  - La información está centralizada y es más sencillo realizar actualizaciones y copias de seguridad
- Sistema gestor de bases de datos (DBMS) = software que permite almacenar y consultar datos
- Existen muchos tipos de bases de datos, pero las más usadas son:
  - Bases de datos relacionales (RDBMS)
  - Bases de datos objeto-relacionales (ORDBMS)
  - Bases de datos NoSQL

Bases de datos

Bases de datos relacionales

- MySQL (Software Libre) - http://www.mysql.org
- Derby (Software Libre) - http://db.apache.org/derby
- H2 (Software libre) - http://www.h2database.com/
- HSQL (Software libre) - http://hsqldb.org/
- MS SQL Server (Comercial) - http://www.microsoft.com/sql
- PostgreSQL (Software Libre) - http://www.postgresql.org/
- Oracle (Comercial) - http://www.oracle.com

RDBMS

ORDBMS

Bases de datos

Bases de datos NoSQL

- El término NoSQL (“no sólo SQL”) define una clase de DBMS que difieren del clásico modelo relacional:
  - No utilizan estructuras fijas como tablas para el almacenamiento de los datos
  - No usan el modelo entidad-relación
  - Arquitectura distribuida (los datos pueden estar compartidos en varias máquinas mediante mecanismos de tablas Hash distribuidas)
- Pueden manejar gran cantidad de datos (***Big Data***)
- Permiten **escalabilidad horizontal** (para mejorar el rendimiento de estos sistemas simplemente se consigue añadiendo más nodos)
- Se recomienda usar una base de datos NoSQL cuando el volumen de los datos crece muy rápidamente (> Terabyte)

Bases de datos

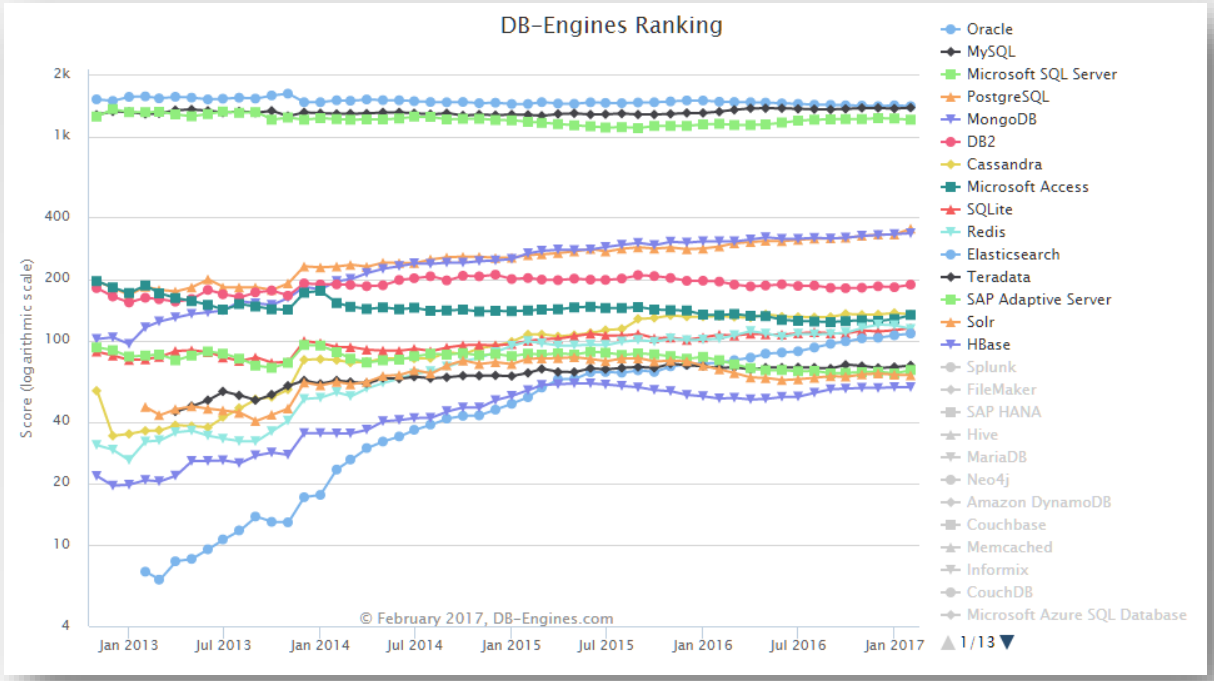
Bases de datos NoSQL

- Hay 4 tipos principales de bases de datos NoSQL:

Documento	Clave-Valor	Columna	Grafo
MongoDB	Redis	Cassandra	Neo4J
CouchDB	Membase	BigTable	FlockDB
RavenDB	Voldemort	Hbase (Hadoop)	InfiniteGraph
Terrastore	MemcacheDB	SimpleDB	InfoGrid
	Riak	Cloudera	Virtuoso

Bases de datos

Ranking de uso de bases de datos



http://db-engines.com/en/ranking_trend

Índice

Sistemas gestores de contenido

- Los sistemas gestores de contenido (**CMS**, ***Content Management System***) son aplicaciones web genéricas que permiten la creación y administración de contenidos vía web
- El sistema permite manejar de manera independiente el contenido y el diseño, permite el cambio de diseño (con *templates* o *themes*)
- Los CMSs han evolucionado para convertirse en un nuevo modelo de desarrollo de aplicaciones web configurando y adaptando módulos con un interfaz web

Sistemas gestores de contenido

- Existen multitud de CMSs con enfoques y objetivos diferentes
- Ejemplos: Drupal (PHP), Joomla (PHP), Wordpress (PHP), Plone (JavaSript), Liferay (Java)…

- Existe CMS específicos para comercio electrónico (*eCommerce*), por ejemplo: Magento (PHP), PrestaShop (PHP), OsCommerce (PHP)…

http://en.wikipedia.org/wiki/List_of_content_management_systems

Índice

Lenguajes de marcado ligeros

- Un **lenguaje de marcado** permite añadir información semántica a un conjunto de datos mediante **etiquetas**. Por ejemplo:

HTML

```
<!DOCTYPE html>
<html lang="en">
<body>

<h1>This is heading 1</h1>

<p>This paragraph one.</p>
<p>This paragraph two.</p>

</body>
</html>
```

XML

```
<project>
   <dependencies>
      <dependency>
        <groupId>junit</groupId>
         <artifactId>junit</artifactId>
         <version>4.12</version>
         <scope>test</scope>
      </dependency>
   </dependencies>
</project>
```

Lenguajes de marcado ligeros

- El problema que tienen los lenguajes de marcados es que habitualmente se hacen demasiado **complejos**
- Por esta razón, en los últimos años se está popularizando el uso de **lenguajes de marcado ligeros**
- El objetivo de estos lenguajes es ser más sencillos (mejor legibilidad por parte de las personas). Por ejemplo:

JSON

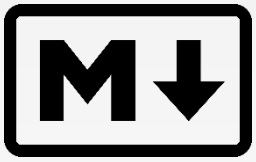
```
{
  "name": "hello-world",
  "version": "1.0.0",
  "dependencies": {
    "cookie-parser": "^1.3.5",
    "minimist": "^1.1.1"
  }
}
```

YAML

```
language: java
sudo: true
dist: trusty

script:
    - mvn verify
```

Lenguajes de marcado ligeros

- Existen ciertos lenguajes de marcado ligeros que acaban generando HTML:
  - Markdown
  - MediaWiki
  - reStructuredText
  - AsciiDoc



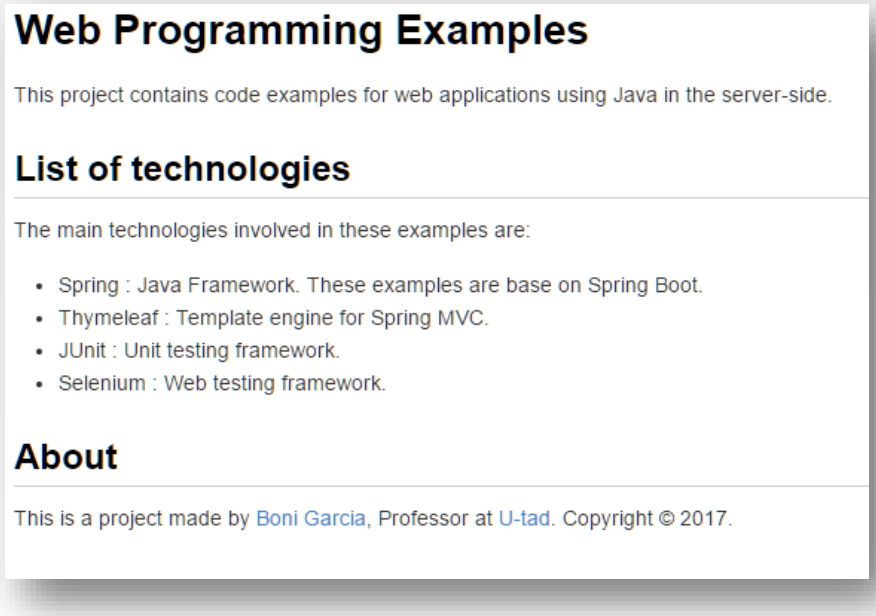
reStructuredText

AsciiDoc

Lenguajes de marcado ligeros

- Ejemplo de Markdown:

```
# Web Programming Examples

This project contains code examples for web
applications using Java in the server-side.

## List of technologies

The main technologies involved in these examples
are:

* Spring : Java Framework. These examples are
based on Spring Boot.
* Thymeleaf : Template engine for Spring MVC.
* JUnit : Unit testing framework.
* Selenium : Web testing framework.

## About

This is a project made by [Boni Garcia], Professor
at [U-tad]. Copyright &copy; 2017.

[U-tad]: http://www.u-tad.com/
[Boni Garcia]: http://bonigarcia.github.io/
```



Web Programming Examples

This project contains code examples for web applications using Java in the server-side.

List of technologies

The main technologies involved in these examples are:

- Spring : Java Framework. These examples are base on Spring Boot.
- Thymeleaf : Template engine for Spring MVC.
- JUnit : Unit testing framework.
- Selenium : Web testing framework.

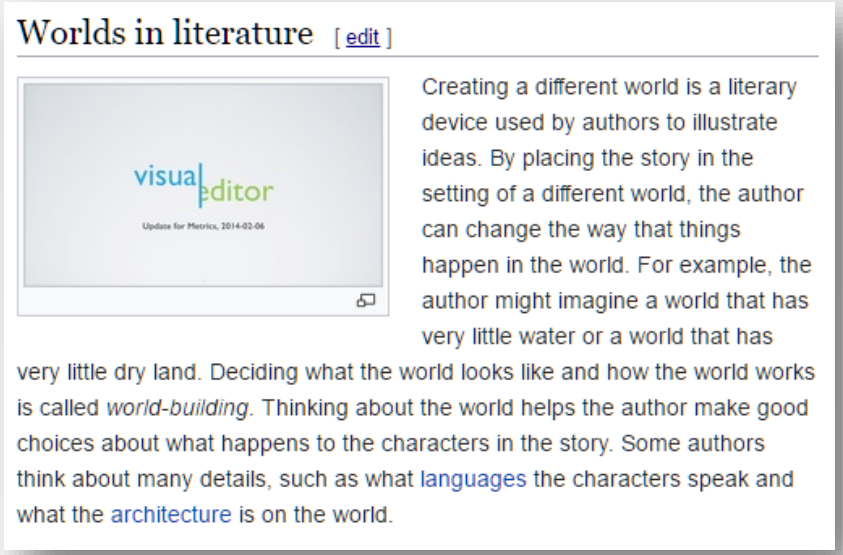
About

This is a project made by Boni Garcia, Professor at U-tad. Copyright © 2017.

Lenguajes de marcado ligeros

- Ejemplo de MediaWiki:

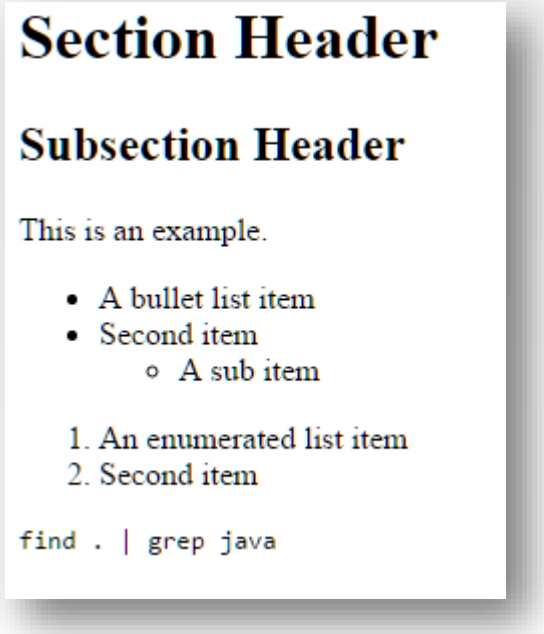
```
== Worlds in literature ==
[[File:VisualEditor - 2014-02 Metrics
deck.pdf|left|thumb]]
Creating a different world is a literary
device used by authors to illustrate ideas. By
placing the story in the setting of a
different world, the author can change the way
that things happen in the world. For example,
the author might imagine a world that has very
little water or a world that has very little
dry land. Deciding what the world looks like
and how the world works is called ''world-
building''. Thinking about the world helps the
author make good choices about what happens to
the characters in the story. Some authors
think about many details, such as what
[[languages]] the characters speak and what
the [[architecture]] is on the world.
```



Lenguajes de marcado ligeros

- Ejemplo de reStructuredText:

```
Section Header
==============

Subsection Header
-----------------

This is an example.

- A bullet list item
- Second item

  - A sub item

1) An enumerated list item

2) Second item

.. code:: bash

    find . | grep java
```



Section Header

Subsection Header

This is an example.

- A bullet list item
- Second item
  - A sub item

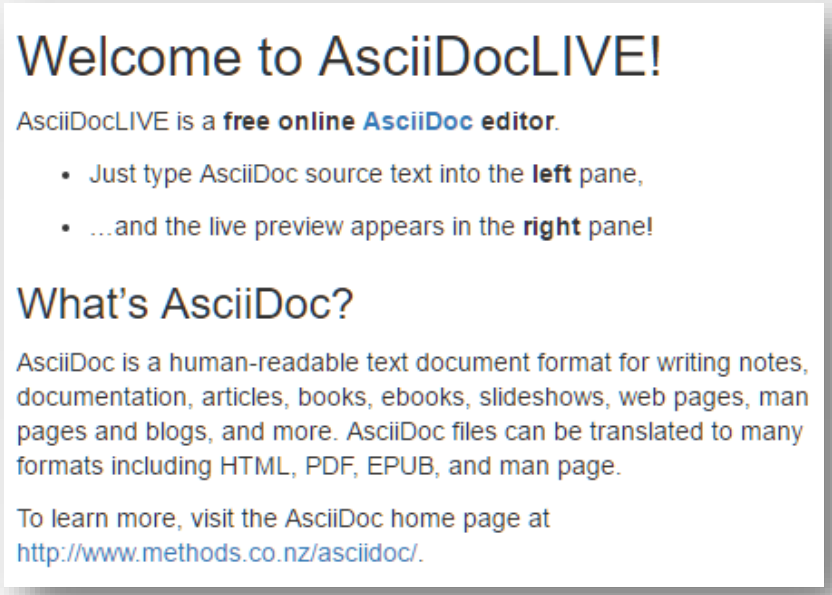
1. An enumerated list item
2. Second item

```
find . | grep java
```

Lenguajes de marcado ligeros

- Ejemplo de AsciiDoc:

```
Welcome to AsciiDocLIVE!
------------------------

AsciiDocLIVE is a *free online
http://www.methods.co.nz/asciidoc/[AsciiDoc^]
editor*.

* Just type AsciiDoc source text into the *left*
pane,
* ...and the live preview appears in the *right*
pane!

What's AsciiDoc?
~~~~~~~~~~~~~~~~

AsciiDoc is a human-readable text document format for
writing notes, documentation, articles, books,
ebooks, slideshows, web pages, man pages and blogs,
and more. AsciiDoc files can be translated to many
formats including HTML, PDF, EPUB, and man page.

To learn more, visit the AsciiDoc home page at
http://www.methods.co.nz/asciidoc/[^].
```



Lenguajes de marcado ligeros

- Últimamente se ha popularizado el uso de estos lenguajes para la generación de sitios web estáticos
- Herramientas útiles:
  - Jekyll: Convierte Markdown (entre otros formatos) a HTML
  - AsciiDoctor: Convierte AsciiDoc a HTML, PDF, EPUB
  - Sphinx: Convierte reStructuredText a HTML, PDF, EPUB
  - Mkdocs: Converte Markdown a HTML/CSS
- Alojamiento gratuitos de sitios web estáticos:
  - GitHub Pages
  - Readthedocs

Asciidoctor

MkDocs

GitHub Pages

Read the Docs