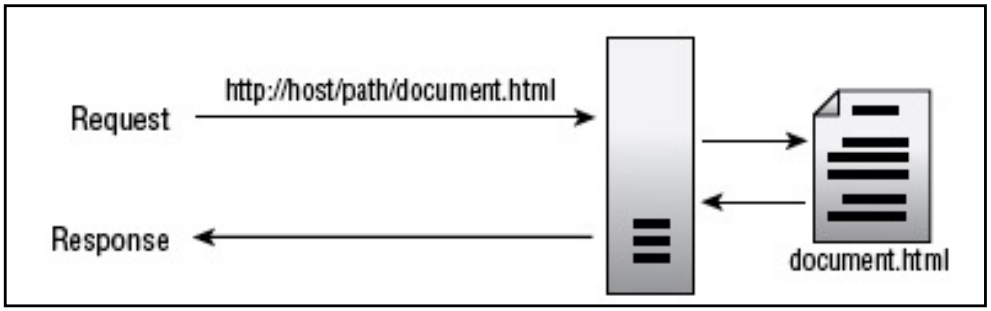
Página Web vs Aplicación Web

De páginas a Aplicaciones Web

- Arquitectura original de la WWW estaba basada en 3 estándares
  - documentos (páginas codificadas en HTML)
  - dirección única de las páginas mediante URLs
  - interacción cliente servidor mediante HTTP

El Rol de HTTP y HTML

Protocolo de solicitud (request) – respuesta (response)

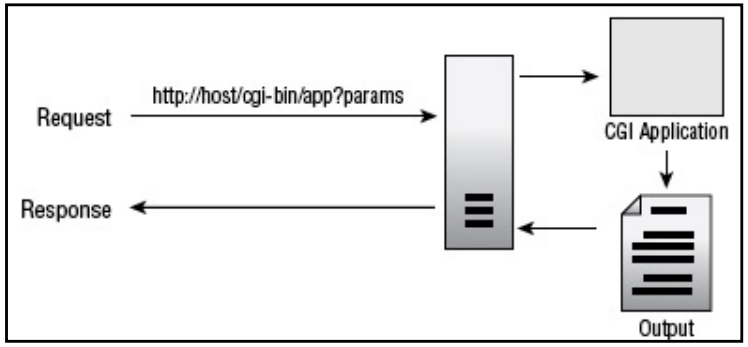


Principales Solicitudes HTTP

- GET Busque y entregue el recurso asociado al URI
- POST Acepte la entidad adjunta en el URI indicado
- PUT Acepte la entidad adjunta y genere un nuevo recurso
- DELETE Elimine el recurso asociado al URI

El Nacimiento de la aplicación Web

- Common Gateway Interface (CGI)
  - URL no corresponde a un documento sino a un programa
  - Servidor invoca ejecución de programa y espera el output
  - Output del programa es devuelto por el servidor como contenido



- Hoy se considera legado (nadie las hace)
  - problemas de performance
  - problemas de conveniencia

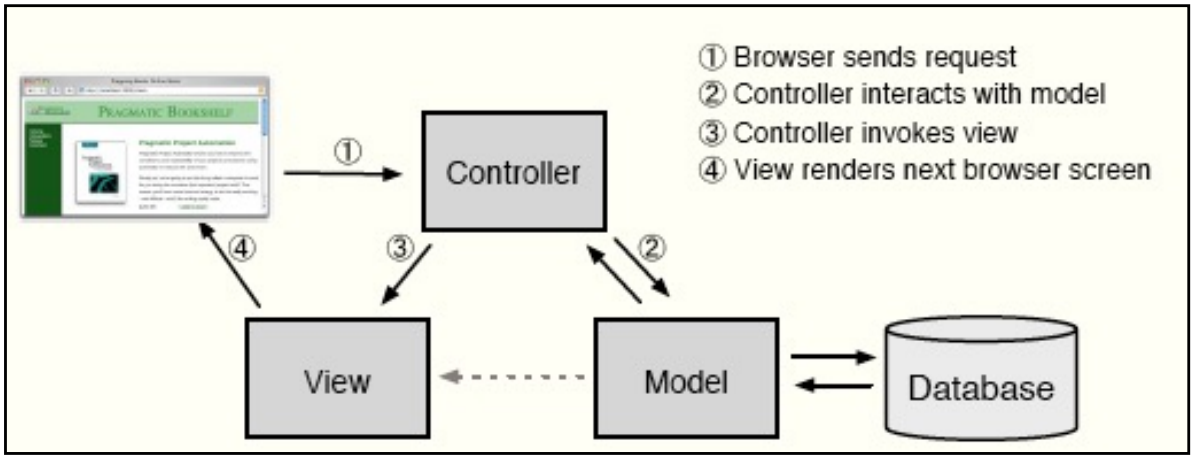
Plataformas Modernas

- Java (JSP, Sevlets, J2EE, Struts, Spring, Tapestry)
- .Net (C#, ASP.Net, VB.Net)
- PHP (Laravel, Symphony)
- Ruby (Rails, Sinatra)
- Python (Django, Flask)
- Node (Express, Sails, Hapi)

Surgimiento de Arquitectura MVC

- La aplicación que corre en el servidor se divide en 3 partes
  - Vista - código que aplicación usa para interactuar con el usuario (HTML, CSS, JavaScript)
  - Modelo - objetos y métodos con la lógica de la aplicación (PHP, Java, Ruby)
  - Controlador - código que sirve de nexo entre vista y modelo (PHP, Java, Ruby)

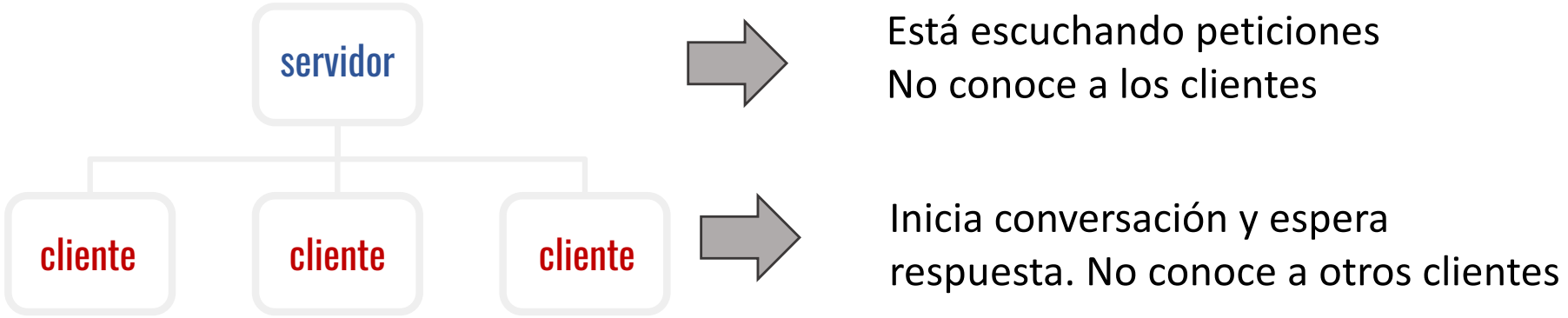
MVC



Web Server captura el request HTTP y lo dirige al controller (1)
Web Server toma la vista construida por la aplicación y la devuelve al solicitante (4)

HTTP: Hypertext Transfer Protocol

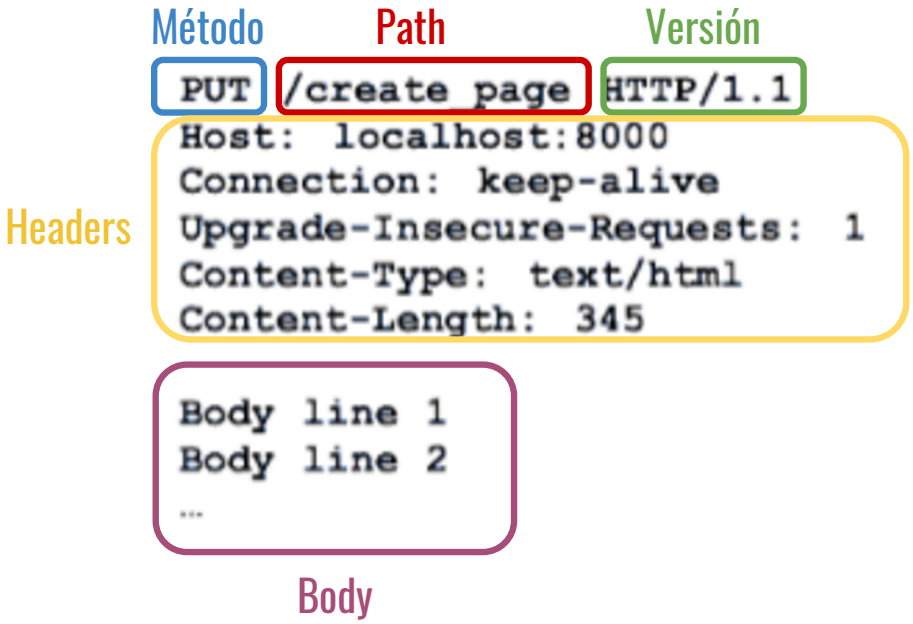
- Se basa en el modelo de arquitectura de software Cliente- Servidor



HTTP: Hypertext Transfer Protocol

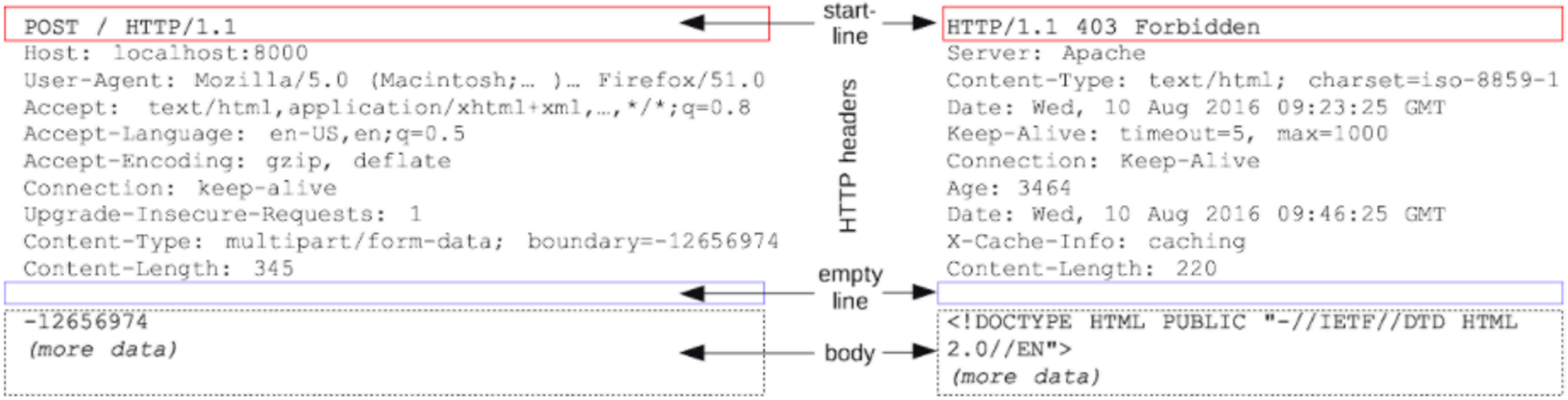
- Stateless - Sin estado
  - El servidor no guarda ningún estado entre dos requests
- Cliente y servidor pueden estar en la misma máquina
- Transmite HTML, imágenes, videos, css, etc
- HTTP 1.0 inicial, HTTP 1.1 en uso actualmente, HTTP 2 desplegándose

HTTP: Ejemplo de Request



Ejemplo de Request

Ejemplo de Response



Códigos de respuesta HTTP

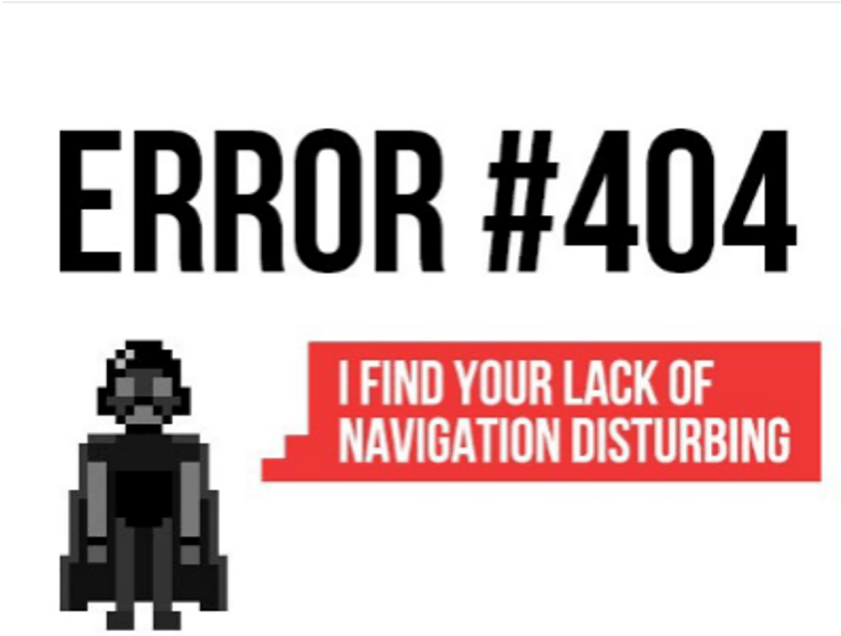
1xx Respuesta informativa

2xx Finalización exitosa de una tarea.
- 200 :ok
- 201 :created
- 204 :no_content

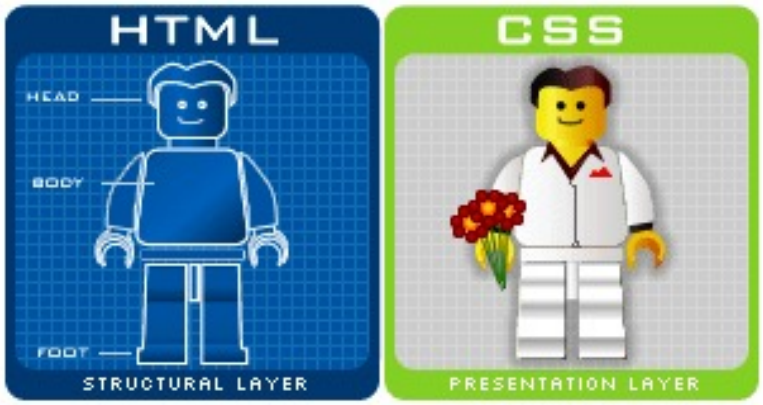
3xx Redirecciones

4xx Errores del cliente
- 400 :bad_request
- 402 :unauthorized
- 404 :not_found

5xx Errores del servidor

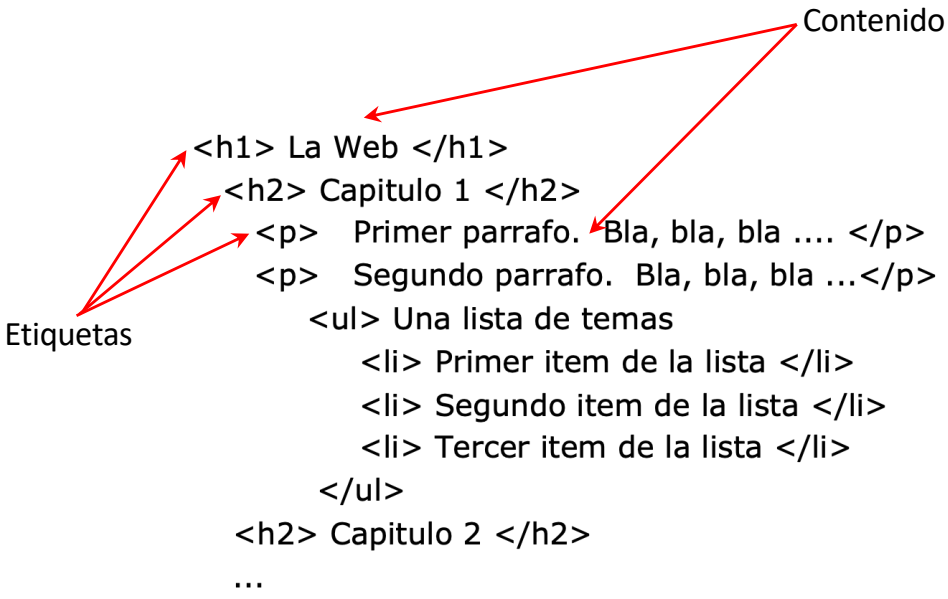


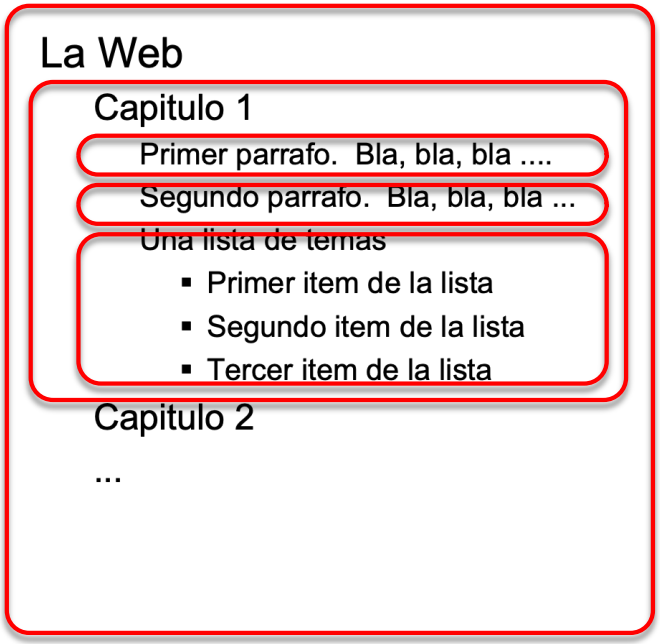
HTML y CSS



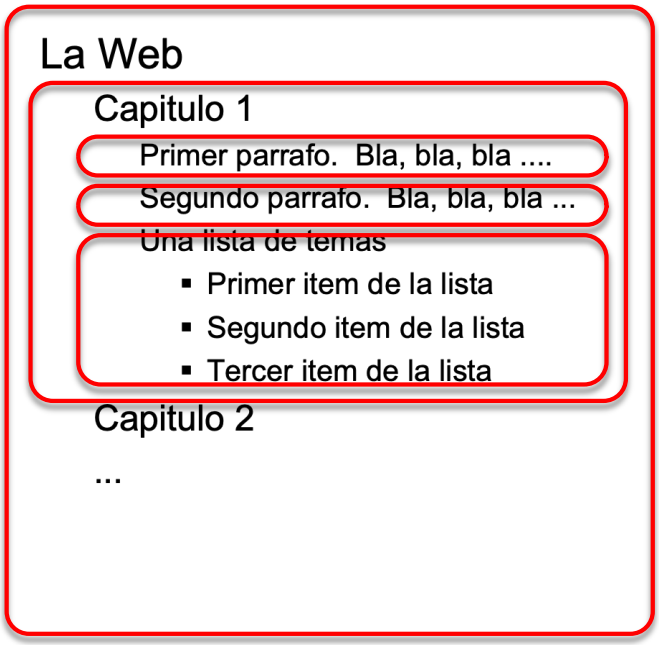
- Browser sabe como desplegar lo que le entrega el server porque el lenguaje es standard: HTML
- HTML se conoce como un lenguaje de marcas (*markup language*) porque combina contenido con instrucciones de estructura
- Antiguamente incluía también instrucciones de formato pero ellas se han separado ahora en otro standard: CSS
- CSS especifica la forma en que debe desplegarse un determinado elemento de HTML (selector)

Contenido Estructurado





Contenido Estructurado



Etiquetas,elementos, atributos

- etiquetas permiten separar contenido de instrucciones de estructura
- existen distintos tipos de etiqueta (h1, h2, img, p, etc.)
- delimitadores de etiqueta < >
- una etiqueta puede incluir atributos
- un atributo es un par clave: valor en que el valor va siempre entre comillas (ej. src = "img.jpg")

Ejemplo

```
<html>
<head>
        <title>Un Ejemplo</title>
</head>
<body>
        <h1>El Comienzo</h1>

        <p> Este es el primer parrafo </p>

        <ol>Y esta una lista de elementos
                <li> primero </li>
                <li> segundo </li>
        </ol>

        <a href="http://www.google.com"> Ir a Google </a>

        <img src="http://i1.wp.com/venturebeat.com/wp-content/google-logo.jpg" />

</body>
</html>
```

Control de la Presentación

- Se usan hojas de estilo
- Permiten especificar como debe ser presentado cada elemento HTML
- Estándar CSS (cascade style sheets)
- CSS -> CSS2 -> CSS3
- Especifica una lista de reglas
- Cada regla incluye un selector y las propiedades asociadas a él

Reglas CSS

```
selector { propiedad1:valor1;
           propiedad2:valor2
         }
```

Ejemplos

```
h1 {text-align: center}
h2 {text-align: center}
h3 {text-align: center}
h1, h2, h3 {text-align: center}
```

Ventajas de Usar CSS

- contenido puede ser presentado en diferentes formas
- documentos mas simples (crear y leer)
- estilo consistente en todas las páginas
- se facilita mantención del sitio
- las páginas cargan más rápido (más cortas)

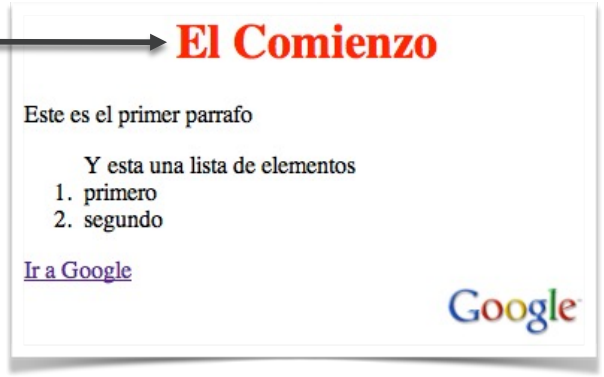
Tres lugares para CSS

- estilos inline
  - como atributo style de un elemento
  - Ejemplo <h1 style = "color: red; text-align: center">
  - específico para la etiqueta en que aparece
- estilos a nivel del documento
  - en el documento html mismo
  - entre etiquetas <style> y </style>
  - específico para esa página
- estilos en hoja externa
  - en un archivo externo (extensión .css)
  - Ejemplo <link rel="stylesheet" type="text/css" href="style.css">
  - aplica a todos quienes lo referencien

Inline

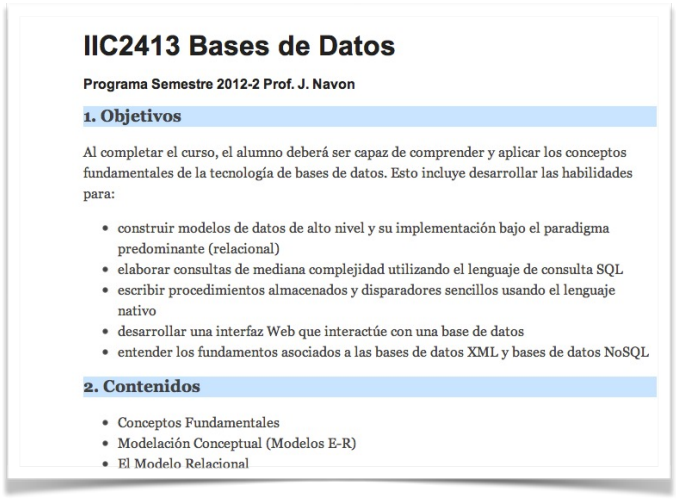
- atributo style de cualquier elemento
- lista de elementos atributo:valor;

```
<html>
<head>
    <title>Un Ejemplo</title>
</head>
<body>
    <h1 style="color:red; text-align:center">El Comienzo</h1>
    <p> Este es el primer parrafo </a>
    <ol>Y esta una lista de elementos
        <li> primero </li>
        <li> segundo </li>
    </ol>
    <a href="http://www.google.com"> Ir a Google </a>
    <img src="http://feutfsm.cl/wp-content/uploads/2012/04/google_1.jpg"
      align="right" width="100 px" />
</body>
</html>
```



En el documento

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
   <title>IIC2413 Bases de Datos - Programa</title>
   <style type="text/css">
      h1, h2 {
         color: #212121;
         font-family: Arial, Helvetica, Geneva, sans-serif;
      }
      h2 {
         font-size: medium;
      }
      body {
         color: #424242;
         text-align: left;
         left: 5em;
         position: absolute;
         line-height: 1.5em;
         font-family: Georgia, "Times New Roman", Times, serif;
      }
      h3 {
         color: #424242;
         margin-top: 2px;
         background-color: #c6e5fe;
      }
     </style>
</head>
<body>
<h1>IIC2413 Bases de Datos </h1>
<h2>Programa Semestre 2012-2 Prof. J. Navon</h2>
```



IIC2413 Bases de Datos

Programa Semestre 2012-2 Prof. J. Navon

1. Objetivos

Al completar el curso, el alumno deberá ser capaz de comprender y aplicar los conceptos fundamentales de la tecnología de bases de datos. Esto incluye desarrollar las habilidades para:

- construir modelos de datos de alto nivel y su implementación bajo el paradigma predominante (relacional)
- elaborar consultas de mediana complejidad utilizando el lenguaje de consulta SQL
- escribir procedimientos almacenados y disparadores sencillos usando el lenguaje nativo
- desarrollar una interfaz Web que interactúe con una base de datos
- entender los fundamentos asociados a las bases de datos XML y bases de datos NoSQL

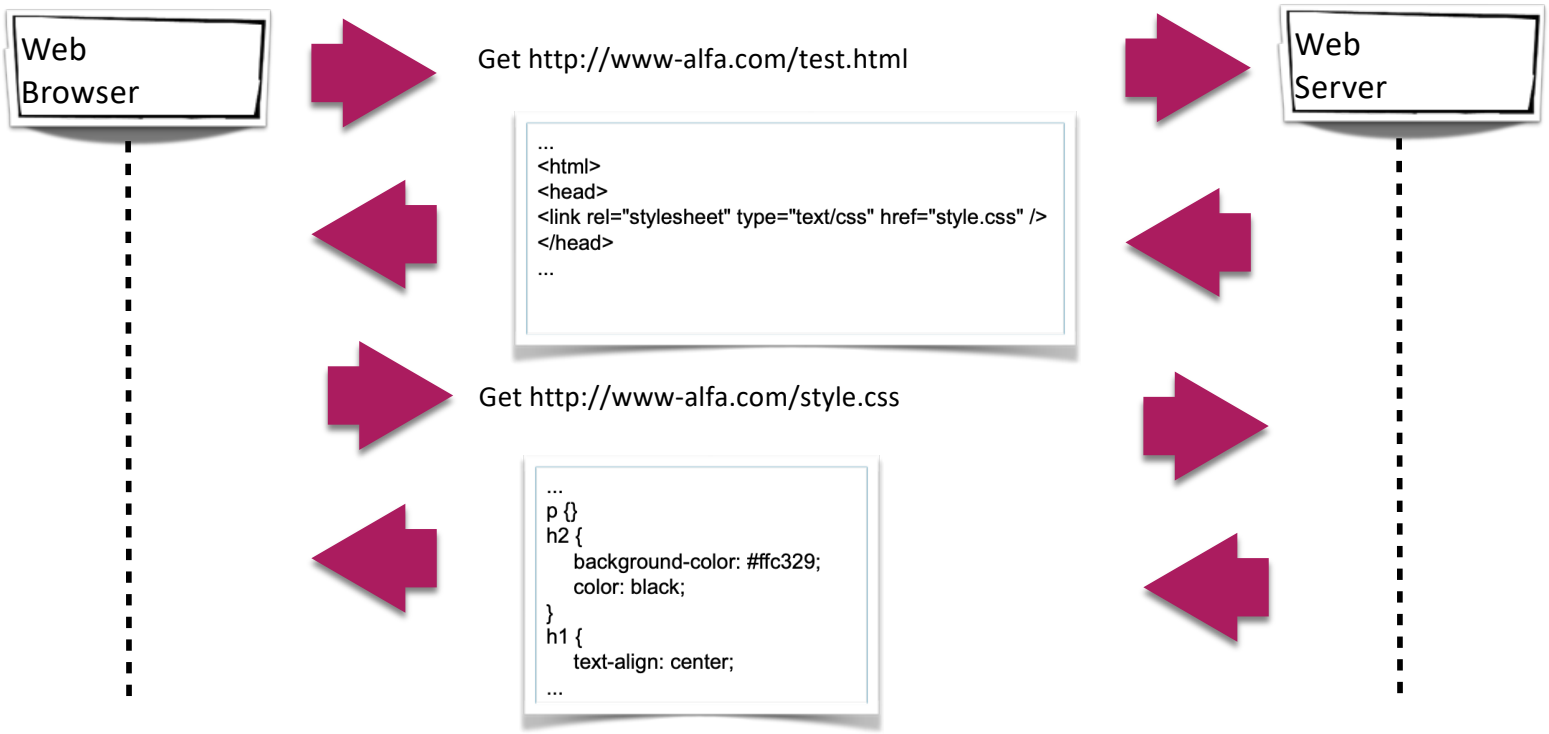
2. Contenidos

- Conceptos Fundamentales
- Modelación Conceptual (Modelos E-R)
- El Modelo Relacional

En hoja externa

- en la sección correspondiente al <head>
- usar etiqueta link con atributos
  - rel = stylesheet"
  - type = "text/css"
  - href = "nombrearchivo.css"
- Ejemplos
  - <link rel="stylesheet" type="text/css" href="style.css" />
  - <link rel="stylesheet" type="text/css" href="http://www.google.com/uds/css/gsearch.css" />

Pagina = HTML + CSS

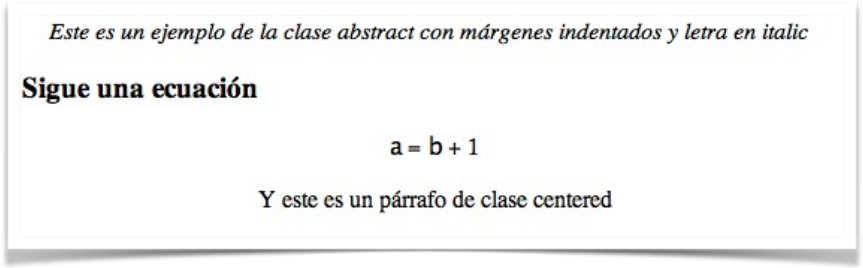


Clases

- selectores asociados a etiquetas html hace que todos queden iguales
- se puede manejar una individualidad con estilo inline
- que hacer si por ejemplo hay dos grandes grupos de párrafos
- solución: párrafos de distintas clases
- atributo class puede agregarse a cualquier etiqueta
- el valor del atributo es el nombre de la clase

Ejemplos de Clases

```
<html>
<head>
   <title>Ejemplo de Clases </title>
<head>
   <style type="text/css">
      p.abstract {font-style: italic; margin-left: 0.5cm;  margin-right: 0.5cm}
      p.equation {font-family: Symbol;  text-align: center}
      h1, p.centered {text-align: center; margin-left: 0.5cm;  margin-right: 0.5cm}
   </style>
   </head>
</head>
<body>
<p class="abstract">Este es un ejemplo de la clase abstract con m&aacute;rgenes indentados y letra en italic</p>
<h3>Sigue una ecuaci&oacute;n</h3>
<p class="equation">a = b + 1</p>
<p class="centered">Y este es un p&aacute;rrafo de clase centered</p>
</body>
</html>
```



Frameworks

- Permiten no tener que escribir TODO el código de una aplicación Web
- Proporcionan una estructura que permite construir una aplicación más fácil de mantener en el futuro
- La mayoría está orientado a producir una arquitectura MVC
- Existen *frameworks* para cada plataforma
  - Ruby – Rails
  - Python – Django
  - PHP - Laravel