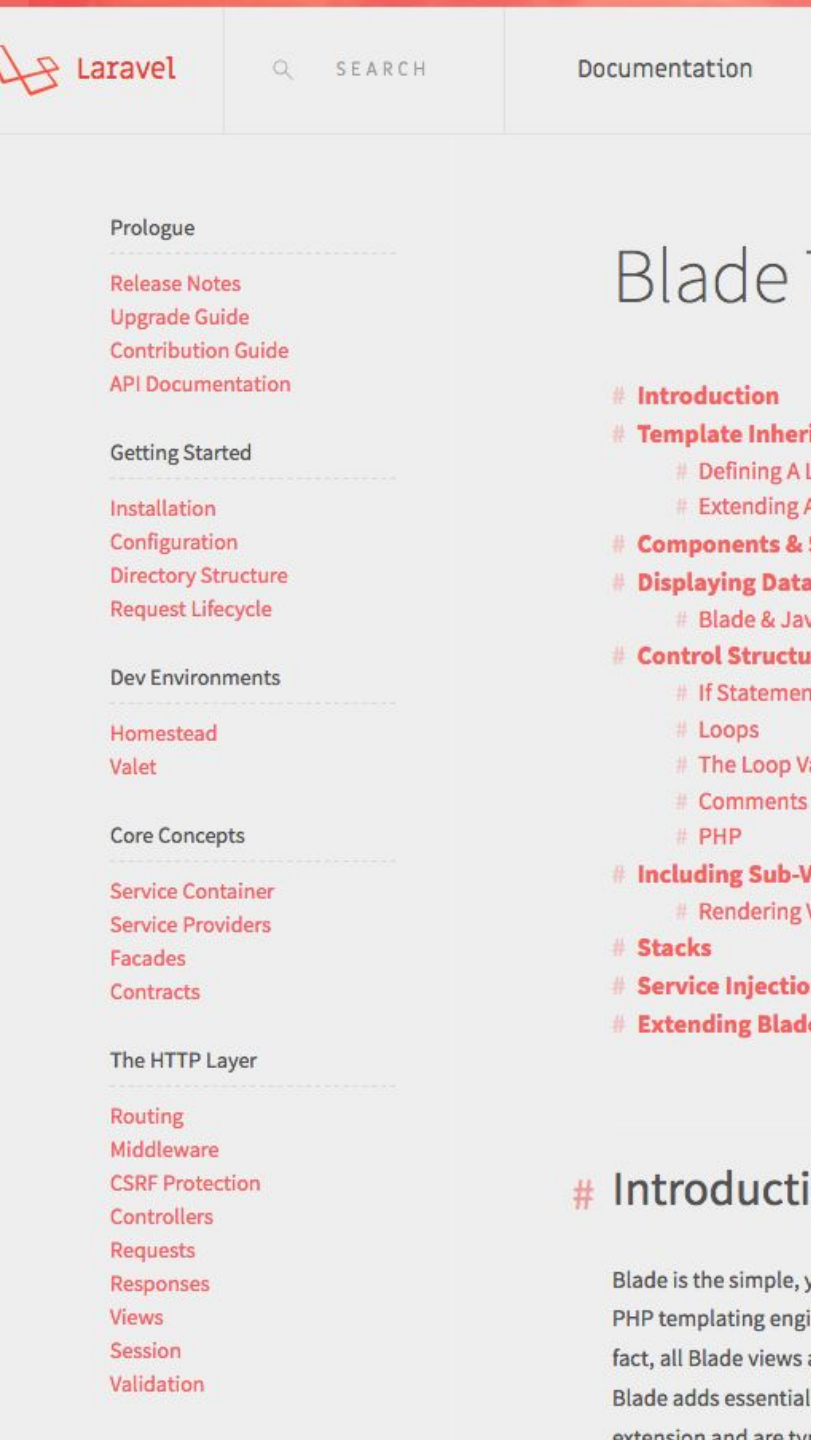


HTML, CSS Y PLANTILLAS BLADE

DISEÑO DE SISTEMAS SOFTWARE

HTML, CSS y plantillas Blade

HTML

HTML

- HTML (**H**yper**T**ext **M**arkup **L**anguage)
- Lenguaje de marcado predominante para la elaboración de páginas web
- Describe la estructura y el contenido en forma de texto
- Se escribe en forma de “etiquetas”, mediante las cuales podemos describir la estructura lógica y apariencia del contenido
- Se puede complementar con otros lenguajes como JavaScript o CSS
- Versiones: HTML 4.01 (1999) → HTML 5 (10/2014)

HTML5

- Versión definitiva desde octubre de 2014, tras 5 años siendo un borrador
- Cambios en HTML5:
  - Añade semántica y accesibilidad implícitas
  - Establece nuevos elementos y atributos que reflejan el uso típico de los sitios web modernos
  - Otros elementos proporcionan nuevas funcionalidades a través de una interfaz estandarizada, como los elementos `<audio>` y `<video>`
  - Algunos elementos de HTML 4.01 han quedado obsoletos (`<font>` y `<center>`)

HTML5

- De los navegadores de escritorio, el que mayor soporte da es Google Chrome, seguido muy de cerca por Mozilla Firefox y Apple Safari
- El que menor compatibilidad ofrece es Internet Explorer
- Para comprobar la compatibilidad de un navegador podemos visitar la Web http://www.html5test.com/ donde se realiza un test de todas las funcionalidades de HTML5
- También existen librerías JavaScript que detecta si son compatibles elementos de HTML5 y CSS3: http://www.modernizr.com

Etiquetas

- Deben de ir encerradas entre corchetes angulares “<>”
- Pueden ser de dos tipos:
  - Apertura y cierre: `<p>texto</p>`
  - Sólo apertura: `<br>`
- Los atributos se deben de colocar en la etiqueta de inicio:

```
<etiqueta atributo1="valor1" atributo2="valor2">...</etiqueta>
```

- O para las etiquetas de solo apertura:

```
<etiqueta atributo1="valor1" atributo2="valor2">
```

Estructura básica de una web

```
<!DOCTYPE html>

<html>
    <head>
        <meta charset=”utf-8”>
        <title>Ejemplo</title>
    </head>
    <body>
        ¡Hola mundo!
    </body>
</html>
```

CABECERA

CUERPO

Elementos de la cabecera

- `<title>Título</title>`: define el título de la página, que aparecerá en la barra de título encima de la ventana
- `<link>`: para vincular el sitio con hojas de estilo CSS:

```
<link href="style.css" rel="stylesheet" type="text/css">
```

- `<style></style>`: estilo CSS en el mismo archivo:

```
...
<head>
    <style type="text/css">
        body { background-color: #EEEEEE; }
    </style>
</head>
...
```

Elementos de la cabecera

- `<meta>`: para indicar metadatos como: description, keywords, author…

```
 <head>
    <meta charset="UTF-8">
    <meta name="description" content="Free Web tutorials">
    <meta name="keywords" content="HTML,CSS,XML,JavaScript">
    <meta name="author" content="John Doe">
    <meta name="viewport" content="width=device-width,
        initial-scale=1.0">
</head>
```

Elementos de la cabecera

- `<script></script>`: permite incluir un script en la Web
- El código se puede cargar desde un fichero externo:

```
<script src="fichero.js" type="text/javascript"></script>
```

- O escribir directamente entre las etiquetas de <script>:

```
<script type="text/javascript">
    // Código de un script integrado en la página
</script>
```

Etiquetas básicas HTML

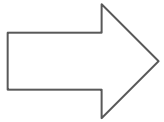
- `<h1></h1>` a `<h6></h6>`: encabezados o títulos del documento
- `<p></p>`: párrafo
- `<br>`: salto de línea
- `<strong></strong>`: **texto en negrita**
- `<em></em>`: *texto en cursiva*
- `<del></del>`: ~~texto tachado~~
- `<u></u>`: <u>texto subrayado</u>

Listas

- `<ol></ol>`: Lista ordenada (con numeración)
- `<ul></ul>`: Lista con puntos (o viñetas)
- `<li></li>`: Elemento de una lista

- Ejemplo:

```
<ol>
   <li>Elemento 1</li>
   <li>Elemento 2</li>
</ol>
```



1. Elemento 1
2. Elemento 2

Tablas

- `<table></table>`: define el inicio y fin de una tabla
- `<tr></tr>`: fila de una tabla
- `<td></td>`: celda de una tabla

```
<table>
    <tr>
        <td>Fila 1 izquierda</td>
        <td>Fila 1 derecha</td>
    </tr>
    <tr>
        <td>Fila 2 izquierda</td>
        <td>Fila 2 derecha</td>
    </tr>
</table>
```

Fila 1 izquierda	Fila 1 derecha
Fila 2 izquierda	Fila 2 derecha

Tablas

- Elementos opcionales para estructurar tablas:
  - `<thead></thead>`: cabecera
  - `<tfoot></tfoot>`: pie
  - `<tbody></tbody>`: cuerpo de la tabla
  - `<th></th>`: celda de cabecera

```
<table>
    <thead>
        <tr>
            <th>Nombre</th>
            <th>DNI</th>
        </tr>
    </thead>
```

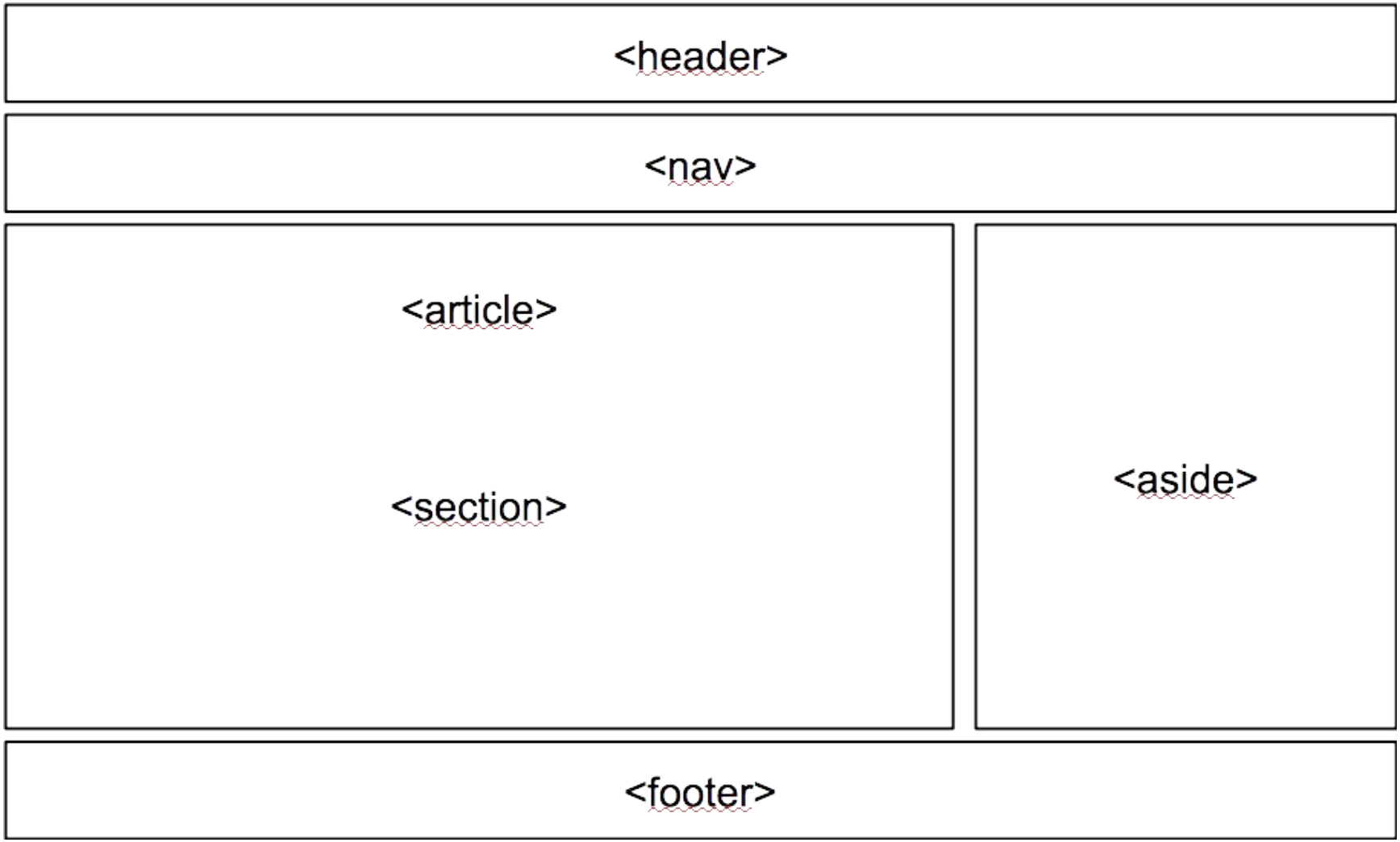
Cajas (etiqueta)

- La etiqueta `<div></div>` define una caja contenedora
- Puede contener cualquier tipo de elemento (texto, imágenes, etc.) u otras etiquetas `<div>` para crear subdivisiones
- Se usan para organizar la disposición de elementos
- Podemos indicar su posición de forma absoluta o relativa
- **Se recomienda su uso junto con CSS cuando se desea alinear contenido (en vez de la etiqueta `<table>`)**

Etiquetas semánticas

- Se usan para estructurar el contenido
  - `header`: cabecera de la página
  - `footer`: pie de página
  - `nav`: bloque de enlaces de navegación (menús)
  - `article`: bloque de texto autocontenido
  - `section`: secciones (capítulos, cabeceras, …)
  - `aside`: contenido relacionado con el contenido principal

Etiquetas semánticas



Imágenes

- Para incluir una imagen se utiliza la etiqueta:

```
<img src="" alt=""/>
```

- Donde:
  - `src=“”` ruta de la imagen
  - `alt=“texto alternativo”`, texto a mostrar en caso de no poder cargar la imagen (también se utiliza en opciones de accesibilidad)
  - `width=””` y `height=””` para redefinir el ancho y el alto de la imagen
- Por ejemplo:

```
<img src="logo.jpg" alt="Logo"/>
```

Enlaces

- Vinculan partes de un documento con otros documentos o con otras partes del mismo documento

```
<a href="URL">Nombre del enlace</a>
```

  - `href=””` establece la dirección URL a la que apunta el enlace
- Por ejemplo:

```
<a href="es.wikipedia.org">Wikipedia</a>
```

- Podemos crear enlaces sobre otros objetos, como imágenes:

```
<a href="dirección_URL"><img src="imagen.jpg"/></a>
```

HTML, CSS y plantillas Blade

CSS

CSS

- Hojas de estilo en cascada: ***C**ascading **S**tyle **S**heets*
- Permite definir la apariencia de una Web
- Separa el contenido del formato
- El W3C (*World Wide Web Consortium*) es el encargado de formular su especificación:
  - 1ª versión → CSS 1 (1996)
  - Última versión → CSS 3 (2011)
- La última versión dividió el estándar en módulos que evolucionan por separado

CSS

- En versiones antiguas de HTML se añadía el formato dentro de las propias etiquetas:

```
<font face="verdana" color="blue" size="12">Texto</font>
```

- Desventajas:
  - Obligaba a especificar el mismo formato en todas las etiquetas para tener un diseño consistente
  - Al cambiar un formato hay que cambiarlo en todas las etiquetas
- Con CSS las etiquetas HTML no deben proporcionar información sobre apariencia

Adjuntar una hoja de estilo

- Puede ser adjuntada de tres formas diferentes:
  - Hoja de estilo externa

```
<head>
    <link rel="stylesheet" type="text/css" href="estilo.css"/>
</head>
```

  - Hoja de estilo interna

```
<head>
    <style type="text/css">
        H1 {color:blue; text-align:center}
    </style>
</head>
```

  - Estilo en línea (inline)

```
<h1 style="color: blue; font-size: 16pt;"> … </h1>
```

Definición de estilos para etiquetas HTML

- Para definir el estilo de una etiqueta HTML usamos la sintaxis:

```
etiqueta {
    estilo CSS 1;
    estilo CSS 2;
    …
}
```

- Podemos definir varias etiquetas a la vez separándolas con comas:

```
etiqueta1, etiqueta2, etiqueta3 {
    estilos CSS;
}
```

Definición de estilos para etiquetas HTML

- Definir etiquetas “dentro” de otras etiquetas

```
contenedor etiqueta {
    estilos CSS;
}
```

- Por ejemplo, definir el estilo de un enlace cuando esté dentro de un párrafo:

```
p a {
    estilos CSS;
}
```

Identificadores y clases

- Permiten crear “clases” de estilos aplicables a etiquetas HTML de dos tipos:
  - Identificador: único en todo el documento HTML

```
#identificador { estilos CSS; }
```

  - Clases: pueden repetirse

```
.clase {  estilos CSS;  }
```

Identificadores y clases

- En HTML usamos `id=""` para asignar el identificador de una etiqueta y `class=""` para usar una clase:

```
<div id="capitulo">
    <p>...</p>
    <p class="destacado">....</p>
</div>
```

- En la hoja de estilos tendríamos:

```
#capitulo {
    estilos CSS;
}
```

```
.destacado {
    estilos CSS;
}
```

Estilos CSS

- La sintaxis básica para definir un estilo es:

```
    atributo: valor;
```

- Los estilos se separan con punto y coma (;)
- Después del atributo se ponen dos puntos (y no un igual)
- Unidades de medida:
  - **píxeles (`px`)**
  - puntos (`pt`)
  - centímetros (`cm`)
  - pulgadas (`in`)

Estilos CSS básicos

- color: valor RGB | nombre de color (white, black, gray, blue, red, green, yellow)

```
color: #009900;          color: red;
```

- font-size: xx-small | x-small | small | medium | large | x-large | xx-large | Unidades de CSS

```
font-size: 12pt;         font-size: x-large;
```

- text-align: left | right | center | justify

```
text-align: right;       text-align: center;
```

- background-color: Un color, con su nombre o su valor RGB

```
background-color: green; background-color: #000055;
```

- width o height: Unidades CSS | Porcentaje

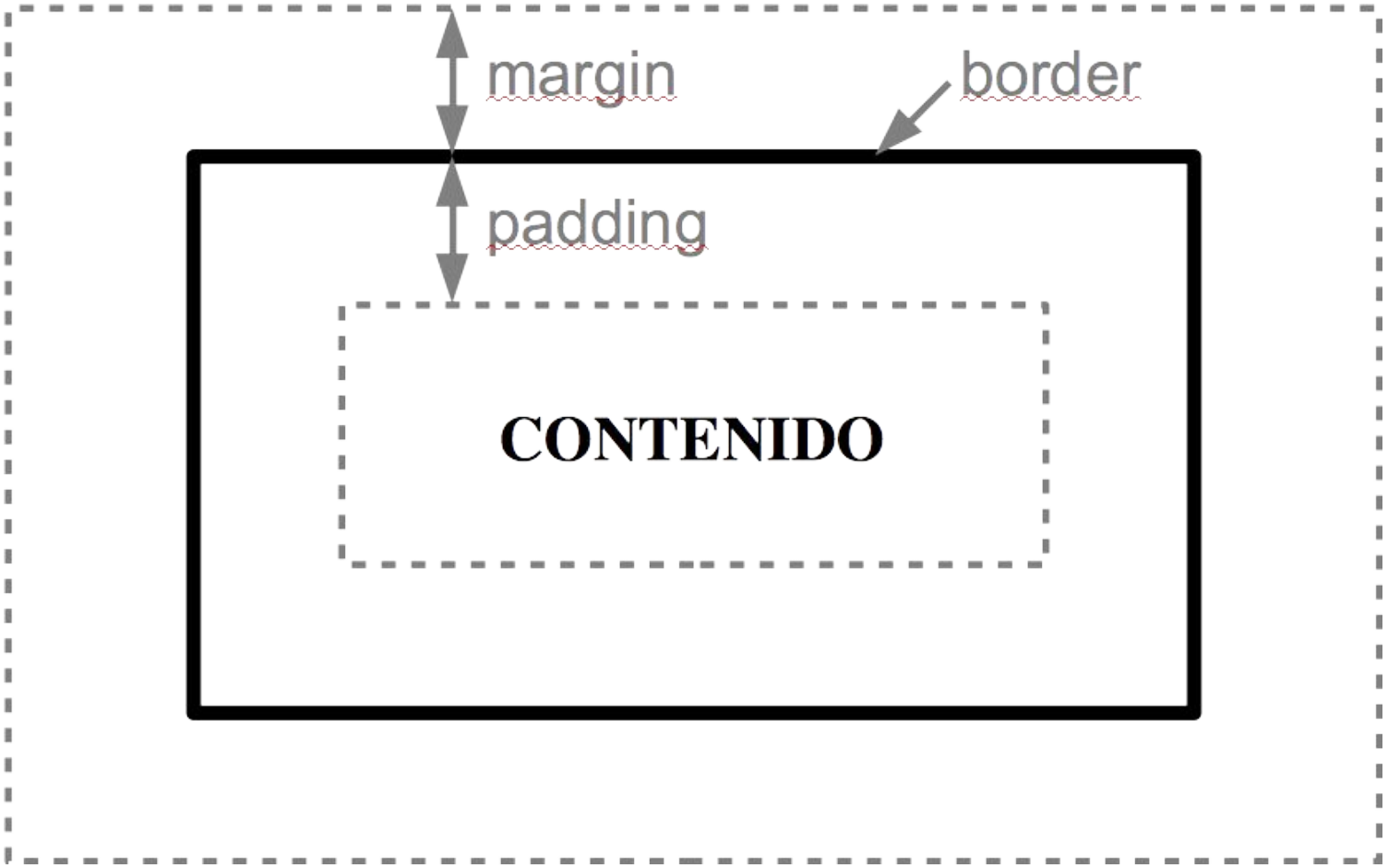
```
width: 50px;             width: 100%;
```

Pseudo-clases

- Permiten definir un estilo solo cuando suceda un determinado evento o se den unas condiciones:
  - `a:hover` - enlace con el puntero del ratón encima, pero no seleccionado
  - `a:visited` - enlaces ya visitados
- Por ejemplo:

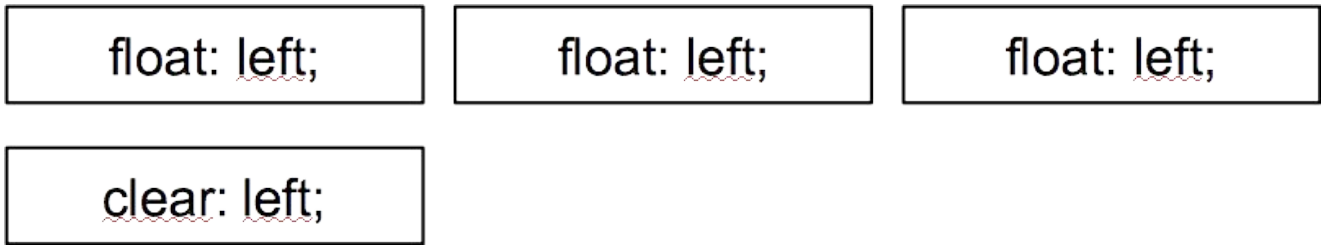
```
a:hover { color: blue; }
a:visited { color: darkgreen; }
```

Estilos para cajas



Capas ( )

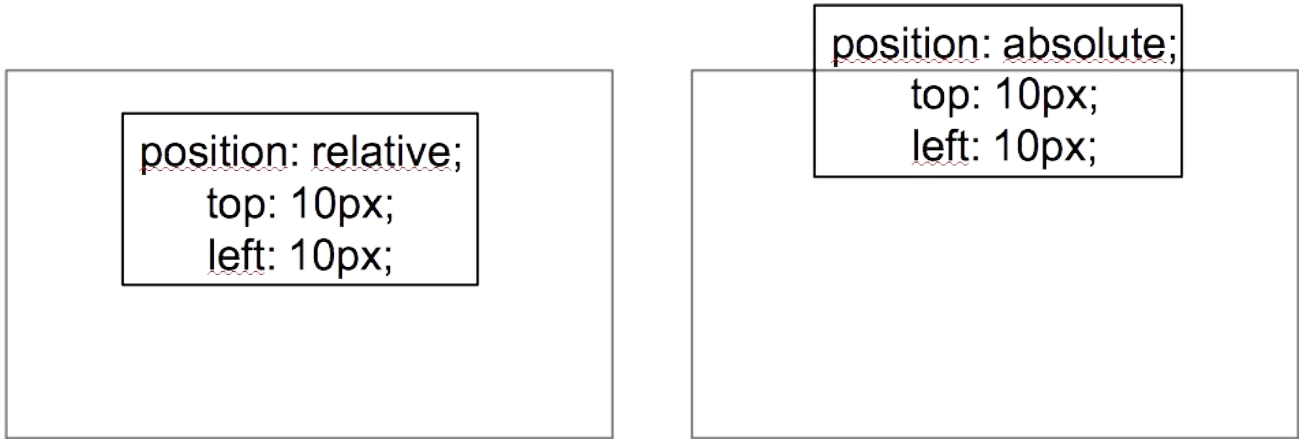
- `float: none | left | right`
  Sirve para alinear un elemento a la izquierda o la derecha de un elemento anterior
- `clear: none | right | left`
  Indica que no se permiten elementos por ese lado del objeto




```
<div style="float:left;width:50px;border:1px solid black">1</div>
<div style="float:left;width:50px;border:1px solid black">2</div>
<div style="float:left;width:50px;border:1px solid black">3</div>
<div style="clear:left;width:50px;border:1px solid black">4</div>
```

Capas ( )

- `position:relative;`
  Su posición depende de la de los demás
- `position:absolute;`
  Coloca los elementos en una posición exacta
- `top, bottom, left, right`
  Para indicar la posición



HTML, CSS y plantillas Blade

PLANTILLAS BLADE

Plantillas Blade

- Laravel utiliza Blade para la definición de plantillas en las vistas
- Permite realizar todo tipo de operaciones con los datos: sustitución de variables o de secciones, herencia entre plantillas, definición de *layouts*, etc.
- Los ficheros de Blade tienen que tener la extensión `.blade.php`
- Para hacer referencia a una vista que utiliza Blade no tenemos que indicar la extensión, por ejemplo:

`“home.blade.php”` → `“view('home');”`

Comentarios / Mostrar valores

- Comentarios con `{{-- comentario --}}`:

```
{{-- Este comentario no se mostrará en HTML --}}
```

- Para mostrar datos usamos `{{ var / función() }}`:

```
Hola {{ $name }}
La hora actual es {{ time() }}
{{-- Si no queremos escapar los datos (CUIDADO): --}}
Hola {!! $name !!}
```

- Operador ternario:

```
{{ isset($name) ? $name : 'Default' }}
{{ $name or 'Default' }}
```

Estructuras de control

- Estructuras de control `if`:

```
@if( count($users) === 1 )
    Solo hay un usuario!
@elseif (count($users) > 1)
    Hay muchos usuarios!
@else
    No hay ningún usuario :(
@endif
```

- Estructuras de control `unless`:

```
@unless (Auth::check())
    Usuario no identificado
@endunless
```

Estructuras de control

- Bucles:

```
@for ($i = 0; $i < 10; $i++)
    El valor actual es {{ $i }}
@endfor

@while (true)
    <p>Soy un bucle while infinito!</p>
@endwhile
```

- Dentro de un bucle se puede usar `@continue` y `@break`

Estructuras de control

- Más bucles:

```
@foreach ($users as $user)
    <p>Usuario {{ $user->id }}</p>
@endforeach

@forelse ($users as $user)
    <li>{{ $user->name }}</li>
@empty
    <p>No hay usuarios</p>
@endforelse
```

Sub-vistas

- Incluir una plantilla dentro de otra plantilla:

```
@include('view_name')

{{-- También podemos pasarle un array de datos
     como segundo parámetro: --}}

@include('view_name', array('some'=>'data'))
```

Componentes

- Los componentes son pequeñas sub-vistas parametrizables

```
<!-- /resources/views/alert.blade.php →
<div class="alert alert-danger">
    {{ $slot }}
</div>
```

- Al incluir un componente podemos incluir contenido que se mostrará en el lugar de la variable `$slot`

```
@component('alert')
    <strong>Whoops!</strong> Something went wrong!
@endcomponent
```

Componentes

- Se pueden definir slots adicionales

```
<!-- /resources/views/alert.blade.php →
<div class="alert alert-danger">
    <div class="alert-title">{{ $title }}</div>
    {{ $slot }}
</div>
```

- Para asignarles valor se usa `@slot('nombre')`

```
@component('alert')
    @slot('title')
        Forbidden
    @endslot
    You are not allowed to access this resource!
@endcomponent
```

Layouts

- Definición de un Layout

```
<!-- /resources/views/layouts/master.blade.php →
<html>
    <head>
        <title>@yield('title')</title>
    </head>
    <body>
        @section('menu')
            Contenido del menu
        @show
        <div class="container">
            @yield('content')
        </div>
    </body>
</html>
```

Layouts

- Para extender una plantilla

```
@extends('layouts.master')
@section('title', 'Título de la página')
@section('menu')
    @parent
    <p>Este contenido es añadido al menú principal.</p>
@endsection
@section('content')
    <p>Este apartado aparecerá en la sección
       "content".</p>
@endsection
```