USERS

INCLUYE VERSIÓN DIGITAL GRATIS

Bootstrap

DESARROLLE SITIOS WEB RESPONSIVE FÁCIL Y RÁPIDO

EL FRAMEWORK CLAVE PARA HTML5, JAVASCRIPT Y CSS3

DISEÑO MOBILE FIRST

por LUIS HERETER / VIVIANA ZANINI

PROGRAME SITIOS WEB ADAPTABLES PARA MÚLTIPLES DISPOSITIVOS

RU

BOOTSTRAP

PROGRAME SITIOS WEB ADAPTABLES
PARA MÚLTIPLES DISPOSITIVOS

por Viviana Zanini y Luis Hereter

RedUSERS

TÍTULO: Bootstrap
AUTORES: Viviana Zanini y Luis Hereter
COLECCIÓN: Seriado
FORMATO: 24 x 17 cm
PÁGINAS: 192

ISBN 978-987-734-049-5

Zanini, Viviana
Bootstrap / Viviana Zanini ; Luis Hereter - 1a ed. - Ciudad Autónoma de Buenos Aires : Fox Andina; Ciudad Autónoma de Buenos Aires: Dalaga; 2015.
192 p. ; 24x17 cm. - (Seriada; 21)

ISBN 978-987-734-049-5

1. Informática. I. Hereter, Luis II. Título CDD 005.1

Viviana Zanini

Es analista de Sistemas de Computación y profesora de Informática. Ha realizado varios cursos de capacitación en el área de Programación.

Se desempeña como profesora en diferentes niveles. Tras haber desarrollado su experiencia laboral en empresas, actualmente realiza proyectos de forma independiente.

Ha colaborado como autora en la colección *Excel Curso Visual y Práctico* y es autora de tres publicaciones de esta editorial: *Macros en Excel 2013*, *Tablas y Gráficos dinámicos en Excel 2013* y *jQuery Mobile* (2014).

Agradecimientos

A mi hermana Sandra, compañera en las derrotas y logros de mi vida. A mis sobrinos Víctor y Christian, que me impulsan a seguir adelante. A Luis, por empujarme hacia el camino de la superación. A mis amigos, por todo su apoyo. A nuestra editora Milagros, por guiarnos en la confección de este libro.

Luis Hereter

Es analista, profesor de Sistemas de Computación y, desde hace más de 8 años, webmaster. Ha desarrollado aplicaciones móviles y de escritorio. Es autor del libro *jQuery Mobile* de esta editorial.

Agradecimientos

A mi mujer Lorena, mi hijo Leandro, mi sobrino Jerónimo y mi hermana Sonia. A mi suegra Nanda, por todo su apoyo durante estos años y a mis tíos Jorge y Sabina. A Luis, Claudia y a la familia Sosa.

A mis hermanos de la vida, Charli, Maxi y Mingo, y a mis amigos de la *Poker League*. A Sergio, por todos los conocimientos compartidos durante tantos años. A mis compañeros de trabajo.

A todo el equipo de la editorial, por apoyar y creer en este proyecto.

Y un agradecimiento muy especial a mi amiga y coautora de este libro, Viviana, porque sin ella no hubiera sido posible, y a Gonzalo Suez, que gentilmente nos permitió hablar de su proyecto Master Bootstrap.

Prólogo

Con el correr de los años, la tecnología web ha mutado de diferentes formas, desde los primeros sitios web hasta las páginas que vemos hoy en día con diseño adaptable. Hace algún tiempo, cuando contábamos con máquinas estándar con procesadores Pentium, no se usaba JavaScript, ya que provocaba que la carga de la página web fuera más lenta. Hoy, con los procesadores estándar actuales podemos utilizar JavaScript junto con otras tecnologías como HTML5 y CSS3; y hemos pasado a realizar operaciones del lado del cliente y muy pocas del lado del servidor, como se venía haciendo tiempo atrás.

Todo va cambiando de acuerdo al avance de la tecnología. El Adobe Flash, que se usaba hace apenas tres o cuatro años, ya no se usa más. Programar para PC de escritorio ya no es lo más recomendado, porque ahora debemos hacerlo teniendo en cuenta todo tipo de resoluciones de pantalla. En cuanto al diseño, ocurrió algo similar: vimos surgir sitios casi sin diseño, pasamos por otros cargados de animaciones y gifs, hasta el diseño actual, minimalista, con pocos colores y, a veces, hasta con toda la información en una sola página.

Los programadores y diseñadores web fuimos mutando como la tecnología. Si hay algo que aprendí con los años es a no quedarme con un solo lenguaje de programación. Ayer podía ser Adobe Flash y hoy es JavaScript combinado con alguna otra cosa. Quizás —como les digo siempre a mis alumnos— el programador web sea el programador más completo, ya que en la Web constantemente todo cambia.

Presentamos hoy un libro destinado a programadores y diseñadores: para los programadores independientes que no quieran depender de un diseñador, para los diseñadores que no quieran depender de un programador y para los programadores y diseñadores que quieran trabajar en conjunto. Bootstrap es una de las herramientas más completas y más usadas para la creación de sitios web. Espero que disfruten de este libro tanto como nosotros disfrutamos de hacerlo.

Luis Hereter

El libro de un vistazo

En este libro aprenderemos a utilizar las herramientas que nos proporciona el framework de Bootstrap para desarrollar sitios web que se adapten a cualquier dispositivo, ya sea un teléfono inteligente, una tablet o una computadora de escritorio, siguiendo el concepto de diseño adaptable (*responsive design*).

*01 BOOTSTRAP

Nos introduciremos en los conceptos básicos sobre diseño web adaptable. Conoceremos qué es Bootstrap y todas las posibilidades que nos ofrece este framework. Realizaremos una breve descripción de las principales características de HTML5, JavaScript y las hojas de estilo CSS3 y PHP, que emplearemos para trabajar con Bootstrap.

*02 INICIANDO BOOTSTRAP

En este capítulo aprenderemos a descargar e instalar el framework de Bootstrap. Veremos cómo instalar un servidor web en nuestra computadora para poder probar los proyectos que realizaremos. Además, conoceremos diferentes editores de texto y finalmente realizaremos una plantilla básica.

*03 MAQUETACIÓN

Una vez creada la plantilla básica, comenzaremos a explicar cómo se utiliza el sistema de rejillas de Bootstrap basado en 12 columnas o rejillas. Veremos el comportamiento del framework de acuerdo a las diferentes resoluciones de pantalla. Finalmente, realizaremos un ejemplo práctico aplicando el sistema de rejillas y todo lo aprendido hasta el momento.

*04 COMPONENTES GRÁFICOS

Comenzaremos a explicar aquí los principales componentes gráficos que trae Bootstrap, que nos permitirán crear la interfaz de la aplicación: tipografía, encabezados, formatos de párrafo, tablas, botones, etiquetas, badges, imágenes e iconos. Presentaremos sus principales características y opciones de configuración.

*05 COMPONENTES GENERALES

A través de diferentes ejemplos, conoceremos y aprenderemos a utilizar los componentes de navegación en Bootstrap: menú de pestañas, menú de píldoras, migas de pan y paginación. También aprenderemos a utilizar y configurar barras de progreso, listas y paneles. Todos estos componentes nos permitirán personalizar nuestro sitio web y adaptarlo según nuestras necesidades e intereses.

*06 FORMULARIOS

En esta sección, nos dedicaremos a conocer el funcionamiento y la utilidad de los formularios, que nos permitirán introducir datos para luego ser procesados en el servidor. Explicaremos sus diferentes componentes y las distintas maneras de configurarlos.

*07 EJEMPLO PRÁCTICO: DISEÑO DE UN PORTFOLIO

Este capítulo consistirá en el diseño y armado de un portfolio. Utilizaremos el sistema de rejillas de Bootstrap y los diferentes componentes gráficos para el armado de las páginas que compondrán el sitio. Explicaremos también cómo realizar los distintos enlaces entre las páginas del sitio.

*08 EJEMPLO PRÁCTICO CON MASTER BOOTSTRAP

En este apartado, crearemos un blog sencillo dedicado a la programación, donde nuestro cliente deberá agregar artículos sobre diferentes lenguajes de programación.
Para ello, utilizaremos el gestor de contenidos Joomla! y la plantilla Master Bootstrap.

* ON WEB CMS Y BOOTSTRAP

Finalmente, realizaremos una introducción a los gestores de contenido (CMS). Explicaremos cuál es su utilidad y qué necesitamos para instalarlos. Para terminar, conoceremos cuáles son los gestores de contenido de código abierto más populares.

INFORMACIÓN COMPLEMENTARIA

A lo largo de este manual, podrá encontrar una serie de recuadros que le brindarán información complementaria: curiosidades, trucos, ideas y consejos sobre los temas tratados. Para que pueda distinguirlos en forma más sencilla, cada recuadro está identificado con diferentes iconos:


CURIOSIDADES E IDEAS


ATENCIÓN


DATOS ÚTILES Y NOVEDADES


SITIOS WEB

Contenido

*05
Componentes generales

*06
Formularios

*07
Ejemplo práctico: diseño de un portfolio

*08
Ejemplo práctico con Master Bootstrap

* ON WEB
CMS y Bootstrap

Introducción

En los últimos años, gracias al avance de la tecnología, accedemos a los sitios web no solamente a través de nuestra computadora de escritorio. Hoy en día podemos hacerlo desde diversos dispositivos, como los teléfonos inteligentes o tablets. Estos dispositivos poseen resoluciones de pantalla diferentes, que nos obligan a diseñar los sitios para que puedan ser visualizados perfectamente en cualquiera de ellos. Bootstrap es un entorno de trabajo (*framework*) que nos permite crear sitios web completamente adaptables a los diferentes dispositivos (lo que se conoce como *web responsive design*).

Este libro está orientado a usuarios que tengan conocimientos previos sobre programación web (HTML y CSS3) y que, además, deseen aprender a utilizar una herramienta que les permitirá crear una aplicación web adaptable de manera rápida y fácil.

Comenzaremos desarrollando los conceptos básicos de diseño web adaptable y luego haremos una introducción al framework de Bootstrap. Explicaremos los primeros pasos para comenzar a trabajar con él: veremos cómo incluir en un archivo HTML las librerías de Bootstrap; aprenderemos cómo funciona el sistema de rejillas de 12 columnas y luego crearemos una página básica basándonos en este sistema.

Posteriormente, conoceremos los componentes gráficos que nos provee Bootstrap: tipografía, botones, imágenes, paneles, menús, barras de navegación, migas de pan y barras de progreso, que nos brindan una amplia variedad de opciones para personalizar nuestro sitio. A continuación, aprenderemos a usar formularios dentro de una aplicación web y veremos cómo utilizar los componentes incluidos dentro de un formulario para el ingreso de usuarios. Empleando los conocimientos adquiridos, crearemos un sitio web adaptable, a partir de un ejemplo sencillo.

Luego, mediante otro ejemplo, aprenderemos a crear un sitio web empleando el gestor de contenidos Joomla! y la plantilla deMaster Bootstrap. Por último, conoceremos los principales gestores de contenido de código abierto y su utilidad.

*01

Bootstrap

En este primer capítulo, definimos qué es Bootstrap y cómo fue desarrollado. Aprendemos en qué consiste el diseño web adaptable y cómo funciona el sistema de columnas en Bootstrap. Luego explicamos qué es el *front-end* o interfaz de usuario y, por último, realizamos una breve descripción de las principales características de HTML5, JavaScript y las hojas de estilo CSS3 y PHP que emplearemos para trabajar con Bootstrap.

¿Qué es Bootstrap?

Bootstrap es un **framework** creado por el equipo de desarrollo de la red social **Twitter** para realizar interfaces web adaptables (*responsive web design*) a cualquier dispositivo, ya sea una tablet, un teléfono o una PC de escritorio. Esto quiere decir que la interfaz se adapta automáticamente a cualquier tamaño y resolución de pantalla sin que el usuario tenga que modificar nada.

El framework utiliza hojas de estilo **CSS**, combinadas con el lenguaje de programación **JavaScript** (que veremos más en detalle al final de este capítulo) y, además, es compatible con la mayoría de los navegadores web, como Google Chrome, Mozilla Firefox y Safari.

Bootstrap es un software libre, por lo que los usuarios tienen la libertad de usarlo, mejorarlo y distribuirlo libremente.

Figura 1. Un mismo sitio web, visualizado en un monitor (arriba), en una tablet (izquierda) y en un teléfono inteligente (derecha).

Breve historia

Bootstrap fue creado en 2011 por Mark Otto y Jacob Thornton, del equipo de desarrollo de la red social Twitter, con el fin de crear un estándar de desarrollo interno a través de un framework que utilice JavaScript y CSS3. Antes de Bootstrap se utilizaba un conjunto de librerías que resultaban inconsistentes y cuyo mantenimiento costaba mucho trabajo a los desarrolladores. Con esta nueva herramienta, no solo se le dio más consistencia a la realización de proyectos web internos sino que también se ahorró mucho tiempo en el desarrollo de estos proyectos.

En agosto del mismo año, salió la primera versión como código abierto en **GitHub** y, en febrero de 2012, Bootstrap se convirtió en el proyecto más popular de desarrollo web.

¿Qué es el diseño web adaptable?

Antes de que profundicemos en el concepto que maneja el framework de Bootstrap, tenemos que entender qué es el **diseño web adaptable** o, en inglés, ***responsive web design*** (**RWD**).

Con el correr de los años, la tecnología creció y actualmente, a diferencia de años anteriores, hay más tablets y teléfonos inteligentes que computadoras de escritorio. Es decir que disponemos de tamaños y resoluciones de pantallas diferentes y, para que los sitios se visualicen bien en cada uno de estos dispositivos, los diseñadores o programadores deben adaptar el formato de sus sitios web a cada una de las pantallas disponibles en el mercado. Por esto surge la necesidad de desarrollar un formato de sitio web estándar que se adapte a todos los tamaños y resoluciones de pantalla: es allí que surge el concepto de "web adaptable".

FRAMEWORK

Un **framework** o **marco de trabajo** es un entorno para el desarrollo de aplicaciones que nos ayuda a resolver, en forma rápida y eficaz, un determinado problema. Utiliza un conjunto de librerías y módulos y le proporciona al desarrollador la opción de reutilizar código que es de uso común y, de esta forma, enfocarse en resolver solo el problema. Entre los frameworks más conocidos está **Django**, utilizado para el desarrollo web, y **jQuery**, para el desarrollo de JavaScript.

Podríamos definir al diseño web adaptable como la técnica que nos permite diseñar páginas web que pueden ser visualizadas perfectamente en todos los dispositivos disponibles. Gracias a este tipo de diseño, ya no necesitamos programar o diseñar una página web para cada tipo de dispositivo existente en el mercado, ya que el sitio web se adaptará automáticamente al tamaño, resolución y orientación del dispositivo que estemos utilizando para acceder. La página web detecta desde qué dispositivo se está conectando el usuario (mediante código de programación) y elige la versión que más se adapte a la resolución de pantalla de este.

El sistema de rejillas (12 columnas)

Para hacer que un sitio web se adapte a todo tipo de pantalla, ya sea una tablet, teléfono inteligente o computadora de escritorio, Bootstrap utiliza un sistema de **12 columnas** —llamadas también **rejillas** o **grids**— para adaptar el contenido de la página web a las distintas resoluciones y tamaños de los dispositivos que pueden utilizarse.

EL SISTEMA DE 12 REJILLAS PERMITE ADAPTAR UN SITIO WEB A DISTINTAS RESOLUCIONES

Por ejemplo, si diseñamos una página web basada en cuatro columnas, cuando accedemos a ella desde una tablet, el contenido puede reorganizarse a tres columnas, o a una sola si lo hacemos desde un teléfono inteligente.

En Bootstrap, ese sistema de columnas lleva un prefijo de clase (que veremos en el **Capítulo 3**) compuesto por cuatro tamaños diferentes, de acuerdo al dispositivo para el cual se esté programando. Por ejemplo, si tenemos

GITHUB

Es un servidor que nos permite alojar proyectos y trabajar en equipo junto con otros desarrolladores. También nos sirve para contribuir en el desarrollo de proyectos de otros. Para utilizarlo tenemos que crear una cuenta gratuita y subir nuestro código. También posee un controlador de versiones que nos permite saber quién modificó nuestro proyecto, cuándo y en qué parte del código lo hizo.

una resolución menor a **768 px**, estamos hablando de un teléfono inteligente, al que le corresponde el prefijo de clase **.col-xs-**.
Si tuviera una resolución mayor o igual a **768 px**, se trataría de una tablet, cuyo prefijo de clase es **.col-sm-**. Para resoluciones mayores o iguales a **992 px**, el prefijo **.col-md-** es el correspondiente a resoluciones medias, propias de las computadoras de escritorio.
Y, por último, para resoluciones mayores o iguales a **1200 px**, el prefijo de la clase es **.col-lg-** que corresponde a resoluciones mayores de computadoras de escritorio.

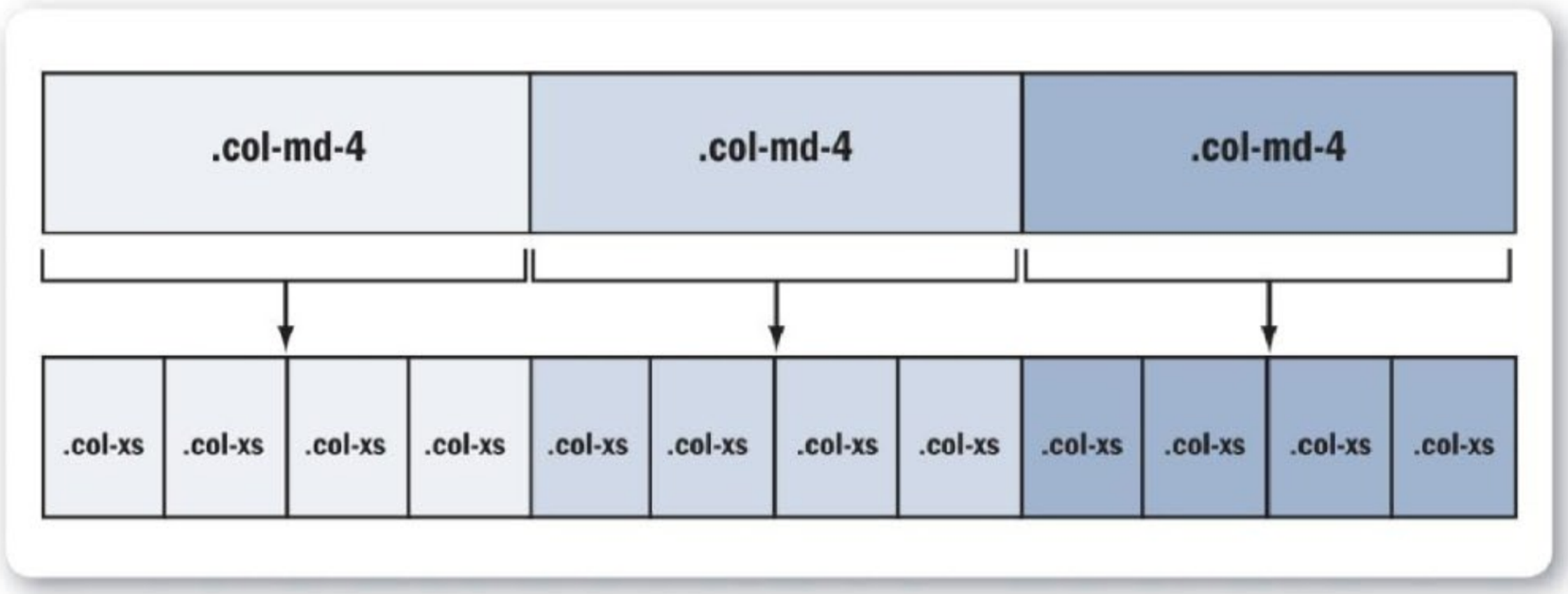


Figura 2. Sistema de grillas básico formado por tres columnas iguales, con el prefijo de clase **md** para resoluciones mayores o iguales a 992 px.

Front-end (lado del cliente)

Podemos definir el *front-end* como la parte del sistema que interactúa con el usuario, su parte visible. En la actualidad, este módulo generalmente es programado con HTML, hojas de estilo CSS y JavaScript. Desde el *front-end* es posible ver archivos multimedia, como videos o música, descargar archivos y guardar datos en el disco del usuario, entre otras cosas.

TÉRMINO "RESPONSIVE WEB DESIGN"

Este término fue usado por Ethan Marcotte, en su libro ***Responsive Web Design*** (2011), para explicar cómo diseñar un sitio web que se adapte a todos los dispositivos a través de los cuales podemos acceder a la Web, mediante el uso de rejillas, imágenes flexibles y *media queries*.

Bootstrap utiliza un sistema de programación del lado del cliente, es decir, *front-end*. También la parte *front-end* es la encargada de recolectar todos los datos de entrada del usuario, para luego ser procesados por lo que se denomina ***back-end***. Este otro concepto, que puede entenderse como "del lado del servidor", es realizado con lenguajes de programación como PHP, Rails y Java, entre otros. Cabe señalar que esta división entre *front-end* y *back-end* ayuda a mantener separadas ambas partes del sistema (parte del cliente y administrador).

Veamos un ejemplo. Un usuario hace una petición al servidor, llenando un formulario de login (usuario y contraseña) para entrar a una página web. El sistema le informa que su acceso es correcto y lo deja entrar; así, puede visualizar la página web solicitada. Analizando este ejemplo, vemos, por un lado, la parte del cliente (*front-end*): cuando el usuario ingresa sus datos para poder visualizar la página solicitada. Al escribir su usuario y contraseña, estos datos son procesados por el servidor con un lenguaje de programación (que podría ser PHP) que se encarga de comparar sus datos de login con los datos de la base de datos y, si son correctos —es decir, si coinciden ambos registros—, habilita al usuario para ver la página web. Esto último es lo que se llama lado del servidor o *back-end*. El usuario nunca supo qué es lo que pasó desde que ingresó su nombre y contraseña (*front-end*); simplemente observa si el sistema lo deja entrar o no.

Figura 3. ***Front-end*** es la interfaz que visualiza el usuario cuando accede al sitio web.

HTML5

HTML (*HyperText Markup Language*) es un lenguaje de etiquetas usado para darle una estructura al sitio web. Define las formas y la estructura que va a tener nuestra página web y cómo la visualizará el navegador.

HTML fue evolucionando con el tiempo, y su última versión es **HTML5**. Este lenguaje es regulado por el consorcio **W3C** (*World Wide Web Consortium*) que se encarga de estandarizar el lenguaje y hacerlo compatible con la mayoría de los navegadores.

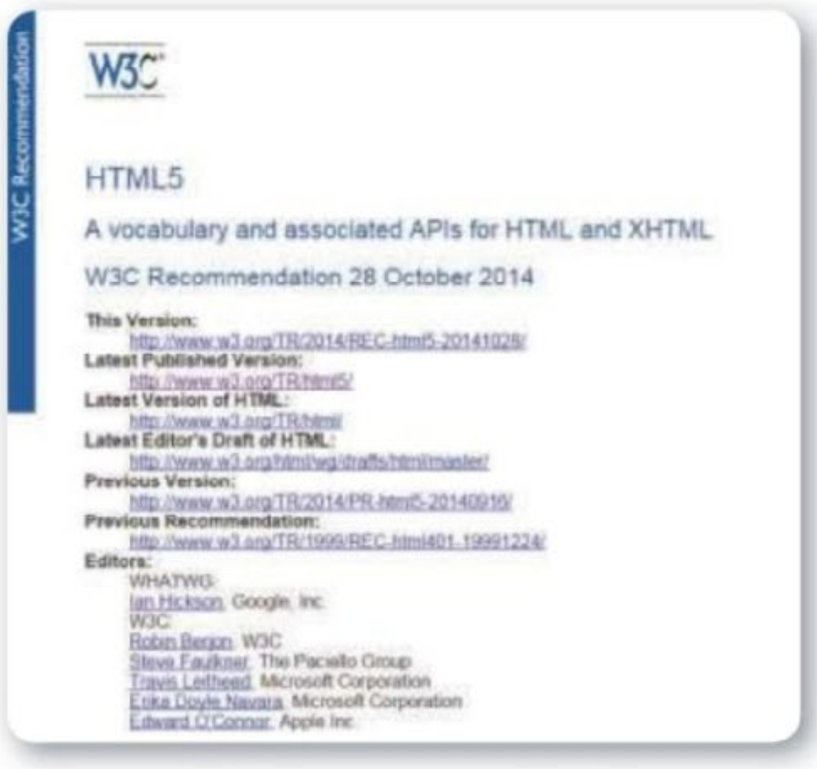


Figura 4. El ***World Wide Web Consortium*** (**www.w3.org**), creado en 1994, se ocupa, entre otras cosas, de estandarizar el lenguaje HTML.

Estructura básica de una página HTML5

Una página básica HTML5 cuenta con la siguiente estructura:

```
<!DOCTYPE html>
<html lang="es">
<head>
   <title>Mi sitio</title>
```

INTERFAZ DE USUARIO (GUI)

En inglés, ***Graphical User Interface*** (**GUI**) es la parte del programa que nos permite comunicarnos o hace de intermediario entre el usuario y nuestro sistema. Mediante el uso de botones, imágenes y menús, entre otros, nos permite a los programadores tomar decisiones sobre cómo debe actuar nuestro sistema de acuerdo a la elección del usuario.

```
    <meta charset="utf-8" />
</head>
<body>
    <header>
      <h1>Mi sitio web</h1>
      <p>Mi sitio web creado en HTML5</p>
</header>
<nav>
  <ul>
   <li><a href="#">Menú A</a></li>
   <li><a href="#">Menú B</a></li>
   <li><a href="#">Menú C</a></li>
  </ul>
</nav>
<section>
  <article>
     <h2>Título del contenido<h2>
     <p> Contenido </p>
  </article>
</section>
<aside>
  <h3>Título de contenido</h3>
  <p>contenido</p>
</aside>
<footer >
  <h2>Esto es un pie de página</h2>
</footer>
</body>
</html>
```

LENGUAJE DE MARCADO (MARKUP)

Los lenguajes de marcado están formados por **etiquetas** o ***tags***, que son instrucciones sobre cómo se verá el texto o imagen en una computadora. A medida que es interpretado por la computadora, se va a imprimir o mostrar de una determinada forma, de acuerdo a las instrucciones o marcas del código de nuestro archivo. El lenguaje de marcado más conocido es el HTML.

Figura 5. Aquí vemos una página web básica maquetada en HTML5.

El primer elemento que visualizamos en el documento es **<!DOCTYPE html>**, que siempre tiene que ir al comienzo de la página, antes de la etiqueta **<head>**, y es el encargado de decirle al navegador que lo que está por leer es una página HTML5. Se diferencia del conocido **<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">** que se emplea en las versiones anteriores de HTML.

A continuación, se ubican las siguientes etiquetas:

- **<html>** y **</html>**: se utilizan para indicar dónde comienza y dónde finaliza, respectivamente, la página web.
- **<head>** y **</head>**: dentro de estas etiquetas se incluye información que va a ser empleada por los navegadores, los servidores web y los buscadores, como, por ejemplo, el título y el autor. Esta información no es visible para el usuario.
- **<body>** y **</body>**: se emplean para definir el cuerpo del documento. Todo lo que se incluya dentro de estas etiquetas será visible para el usuario.

Dentro de las etiquetas **<body>**, podemos incluir otras que nos permiten definir las secciones lógicas que componen un documento HTML5:

- **<header> </header>**: estas etiquetas nos permiten definir el encabezado de la página: por ejemplo, un título, un logo o imagen de la empresa. Podemos tener más de una etiqueta **<header>** en todo el documento, es decir que podremos utilizarla, por ejemplo, como barra de navegación.

- **<nav> </nav>**: dentro de estas podemos incluir enlaces (links) a determinados lugares de nuestra web o enlaces externos, creando, de este modo, una barra de navegación. Debemos tener en cuenta que no todos los enlaces deben estar entre estas etiquetas; por ejemplo, los enlaces que se encuentran en el pie de página, que ya están contenidos por las etiquetas **<footer> </footer>**, no deben colocarse entre **<nav>** y **</nav>**.
- **<article> </article>**: empleamos estas etiquetas para agrupar contenido de una página web, como ser: comentarios de foros, entradas de blogs, artículos de periódicos o revistas, comentarios enviados por un usuario o cualquier otro contenido estático. Estos elementos deberían tener un título **<h1></h1>** que puede ir entre etiquetas **<header></header>**.
- **<section> </section>**: nos permiten agrupar el contenido que refiere a un mismo tema. Por ejemplo, si tenemos una página web en donde ofrecemos productos de computación y, a su vez, las distintas formas de pago, una sección sería la de los productos y otra, la de su forma de pago. Es fundamental que tengamos en claro qué vamos incluir en cada sección, para después poder dividirla en subsecciones.
- **<aside> </aside>**: las empleamos para incluir contenido tangencial, es decir, el contenido que no tiene que ver con el tema principal; por ejemplo: calendario de eventos, publicidad, enlaces externos, etcétera. Por lo general, estas etiquetas están representadas por barras laterales a la derecha o izquierda de nuestra página web.
- **<footer> </footer>**: estas etiquetas se ubican en la parte inferior de la página y sirven para crear un pie de página o sección final. Podemos incluir dentro de estas etiquetas links, publicidad, derechos de autor, mapa del sitio, entre otras opciones.

COMENTARIOS

A veces es muy útil dejar comentarios en el documento HTML sobre el código que escribimos, ya que nos pueden servir para aclarar o recordar posteriormente lo que hicimos. Para escribirlos, debemos encerrar el comentario entre los símbolos **<!–** y **–>**. Todo lo que encerremos entre estos dos símbolos no se visualizará en pantalla.

Etiquetas más usadas

En la siguiente tabla, describiremos las etiquetas de HTML5 que usamos habitualmente para crear una página web.

ETIQUETAS MÁS USADAS DE HTML5

ELEMENTO	ETIQUETA	DESCRIPCIÓN
Título	** **	Permite definir un texto como título. Se pueden definir seis tamaños de títulos, que van desde **<h1>** hasta **<h6>.**
Párrafo	** **	Se emplea para definir bloques de texto independientes (párrafos).
Enlaces	**<a>**	La etiqueta **<a>** junto con el atributo **href** se utiliza para crear un enlace, ya sea interno o externo.
Imágenes	**<img>**	Esta etiqueta, junto al atributo **src**, se emplea para incluir una imagen en la página web.
Listas	** ** ** ** ** **	Las etiquetas ** ** y ** ** nos permiten crear listas ordenadas y desordenas, respectivamente. Mediante la etiqueta ** ** creamos los elementos de la lista.
División	** **	Se emplea para definir un contenedor, es decir, una división dentro de una misma página.
Script	**<script></script>**	Esta etiqueta se emplea para englobar acciones de otros lenguajes.
Formularios	**<form></form>**	Dentro de estas etiquetas se definen controles que permiten recoger información para luego ser procesados.

Tabla 1. Etiquetas de HTML5 que usaremos con más frecuencia para crear un sitio web.

OTRAS PUBLICACIONES ÚTILES

Para profundizar conocimientos que nos servirán para desarrollar sitios más complejos, recomendamos leer los manuales ***HTML5***, de Damián de Luca, y ***PHP6*** y ***Desarrollo PHP y MySQL***, de Francisco José Minera, así como el libro ***Sitio multiplataforma con HTML5 + CSS3*** de Eugenia Casabona y Ricardo Ceci, publicados por esta misma editorial.

CCS3

Las hojas de estilo **en cascada** o **CSS** (en inglés, ***Cascading Style Sheets***) se utilizan para definir la parte visual de un documento HTML.

Entre los elementos que podemos definir se encuentran, por ejemplo, el texto, el color, las imágenes y las posiciones de las secciones.

Las **instrucciones CSS** (reglas de formato) pueden escribirse directamente en el archivo HTML de la página, o bien en un archivo separado que luego deberemos vincular al archivo HTML.

Si agregamos el código CSS en el mismo documento HTML, debemos hacerlo entre las etiquetas **<head>** y **</head>** de nuestra página web.

Las siguientes líneas de código muestran un ejemplo de estilo CSS para definir el formato de un párrafo:

```
<style>
p{ font-family: Arial; font-size: 32px} ;
</syle>
```

En este ejemplo, vemos cómo, entre las etiquetas de apertura y cierre **<style></style>**, estamos modificando un párrafo ** **, cambiándole el tipo de fuente (**Arial**) y su tamaño (**32px**).

Una buena costumbre no escrita en programación, pero que recomendamos, es siempre utilizar un archivo separado para escribir las instrucciones de hojas de estilo (CSS). De esta forma queda bien separado, por un lado, el diseño y, por otro, la programación. Es más prolijo, sobre todo cuando el código de nuestra página web o proyecto es extenso. Para esto, lo que tenemos que hacer es escribir el conjunto de reglas de formato en un archivo con la extensión **.css** y, posteriormente, en el archivo de nuestra página web, vincular el archivo **.css**. al documento HTML.

Por ejemplo, en un archivo **estilo.css** creamos un formato para los encabezados **h1**: queremos que sean de color rojo, tengan fuente Verdana y un tamaño en pixeles de 15, como se muestra en el código:

```
h1{
font-family: Verdana;
font-size: 15px;
color: red;
}
```

Para vincular este archivo, escribimos entre las etiquetas **<head></head>** del documento HTML el siguiente código:

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

Las etiquetas **<link></link>** nos sirven para escribir un **enlace a un documento externo** y, mediante la propiedad **href**, indicamos dónde se encuentra ubicado el archivo (en nuestro ejemplo, está dentro de la carpeta **css**).

Con el atributo **type= "text/css"**, estamos indicando el contenido del archivo. Por último, con el atributo **rel** especificamos la relación que existe entre nuestra página y la de destino. Por eso, en este caso debemos poner el valor **StyleSheet**, porque nuestro destino es una hoja de estilo que va a contener los estilos del documento HTML.

Las opciones para personalizar una página web son muchas y las iremos viendo a medida que avancemos en los capítulos de este libro y realicemos los ejemplos finales.

Less

Es un preprocesador de hojas de estilo que nos permite utilizar las hojas de estilo como un lenguaje de programación **dinámico**, facilitando su uso.

Con un preprocesador podremos, entre otras cosas, utilizar funciones y variables, hacer cálculos y, de esta forma, crear las hojas de estilo más dinámicas organizando mejor el código. En otras palabras, un preprocesador nos permite manejar las hojas de estilo como cualquier lenguaje de programación y nos devuelve como resultado final una hoja de estilo que podrá ser interpretada por el navegador web que utilicemos.

LESS NOS PERMITE CREAR HOJAS DE ESTILO DINÁMICAS DE FORMA SENCILLA Y ORGANIZADA

Así, por ejemplo, en **Less** las variables siempre se definen con una **@** adelante y la asignación con dos puntos (**:**), como podemos observar en el siguiente ejemplo:

```
@color: #2a70e8;

#footer{
        Color: @color;
}
```

Bootstrap utiliza **Less** y **Sass**, los preprocesadores más importantes y más utilizados en la actualidad que nos permiten modificar el código fuente del framework.

JavaScript

JavaScript es un lenguaje de programación **interpretado**, lo que significa que es ejecutado por un intérprete (en este caso, un navegador web). También es llamado **lenguaje de scripting** porque no necesita ser compilado: a diferencia de otros lenguajes, a medida que el navegador web lo está leyendo, va interpretándolo y ejecutándolo.

JavaScript es, como dijimos anteriormente, un lenguaje del lado del cliente. Esto significa que las peticiones son realizadas y resueltas en el navegador, sin necesidad de ir al servidor constantemente y traer dichas peticiones, como lo hace un lenguaje como PHP.

Figura 6. JavaScript nos permite añadir dinamismo a una página web. Por ejemplo, podemos mostrar una ventana de alerta al cargar la página.

Configuración de un script

Un script es un programa que es ejecutado e interpretado por el navegador web (**intérprete**) del usuario (**cliente**). Las instrucciones del script se pueden ejecutar ya sea cuando se carga la página web o bien cuando se produce un determinado evento, como por ejemplo cuando hacemos clic sobre un botón de formulario.

Existen dos formas de crear un script de JavaScript: una es sobre la misma **página HTML** y la otra es en un **archivo aparte** que es llamado o incluido en una página HTML (la llamada al archivo) que tiene que tener la extensión **.js** (extensión de JavaScript).

Si quisiéramos crearlo desde la misma página web que estamos desarrollando, debemos hacer lo siguiente:

```
<script type= "text/javascript" >
</script>
```

Con las etiquetas **<script> </script>** le estamos informando al navegador que el contenido que va a leer o interpretar es un script. Y con el atributo **type** con el valor **"text/javascript"** le indicamos que pertenece al lenguaje JavaScript.

En cambio, si quisiéramos ejecutar un script con alguna función que se encuentra en otro documento **.js**, debemos escribir el siguiente código entre las etiquetas **<head></head>** de nuestra página HTML:

```
<head>
<script type="text/javascript" src="js/jquery-1.8.1.min.js"></script>
</head>
```


ECMASCRIPT

Brendan Eich, programador de la empresa Netscape Communications, crea el lenguaje **LiveScript** para su navegador web Netscape, con el fin de solucionar problemas de conexión al servidor, ya que este lenguaje se ejecutaba del lado del cliente. Posteriormente, Netscape se asocia con *Sun Microsystems* y le cambian el nombre a **JavaScript**. En 1997 se envía el lenguaje al organismo *European Computer Manufacturers Association* (ECMA) para crear un estándar que sea compatible con todos los navegadores web. ECMA crea un comité que define al lenguaje JavaScript con el nombre de **ECMAScript**.

En este caso, estamos llamando o incluyendo en nuestra página HTML un archivo JavaScript que se encuentra en un directorio llamado **js**. Como en el ejemplo anterior, entre las etiquetas **<script> </script>** se hace el llamado y, con el atributo **type**, se declara que es JavaScript pasándole el valor **"text/javascript"**. Por último, con el atributo **src** le estamos indicando la ruta del archivo, que puede ser en el mismo servidor o en un servidor externo (en el segundo caso, tenemos que pasarle la dirección URL del servidor).

Variables en JavaScript

En lenguajes de programación, una **variable** es un espacio temporal, reservado por el sistema operativo, para almacenar algún dato. Ese dato puede ser un número, texto, fechas, entre otros.

A diferencia de otros lenguajes de programación, JavaScript interpreta qué tipo de variable es de acuerdo a su contenido, es decir que no hace falta declarar el tipo de dato que va a utilizar. Por ejemplo, si quisiéramos almacenar texto en una variable, lo haríamos de la siguiente forma:

```
var mensaje = "Mi nombre es Alberto";
```

Anteponiendo la palabra reservada **var**, estamos declarando que vamos a definir una variable; a continuación, escribimos el nombre de esta (en el ejemplo, **"mensaje"**) y le decimos que va a ser igual o va a tener el valor **"Mi nombre es Alberto"**. Como el contenido está encerrado entre comillas dobles —también podrían ser comillas simples—, significa que se trata de una cadena de texto; entonces la variable es del tipo texto.

DECLARAR VARIABLES

Aparte de declarar una variable anteponiendo la palabra reservada **var**, podemos también simplemente inicializar la misma con algún valor ya asignado, como: **Mi_numero = 5;**. En este caso, estamos diciendo que la variable **Mi_numero** es del tipo entero y tiene por valor **5**. A este tipo de declaración se la llama **declaración implícita**.

PHP

PHP, o **procesador previo al hipertexto** (*Pre-Hypertext Processor*), es un lenguaje de programación que, a diferencia de HTML, se ejecuta en el servidor. Es decir, todo lo que codifiquemos en PHP no se interpreta en el navegador, sino que es interpretado por un servidor web como, por ejemplo, Apache. El servidor se encarga de procesar las instrucciones y, una vez procesadas, devuelve el código HTML resultante al navegador.

PHP nos permite desarrollar sitios web dinámicos con la posibilidad de interactuar con un motor de base de datos, como MySQL, Oracle, Microsoft SQL Server y, de esta forma, introducir foros, blogs, carritos de compras, entre otros elementos.

A DIFERENCIA DEL LENGUAJE HTML, PHP SE EJECUTA DIRECTAMENTE EN EL SERVIDOR WEB

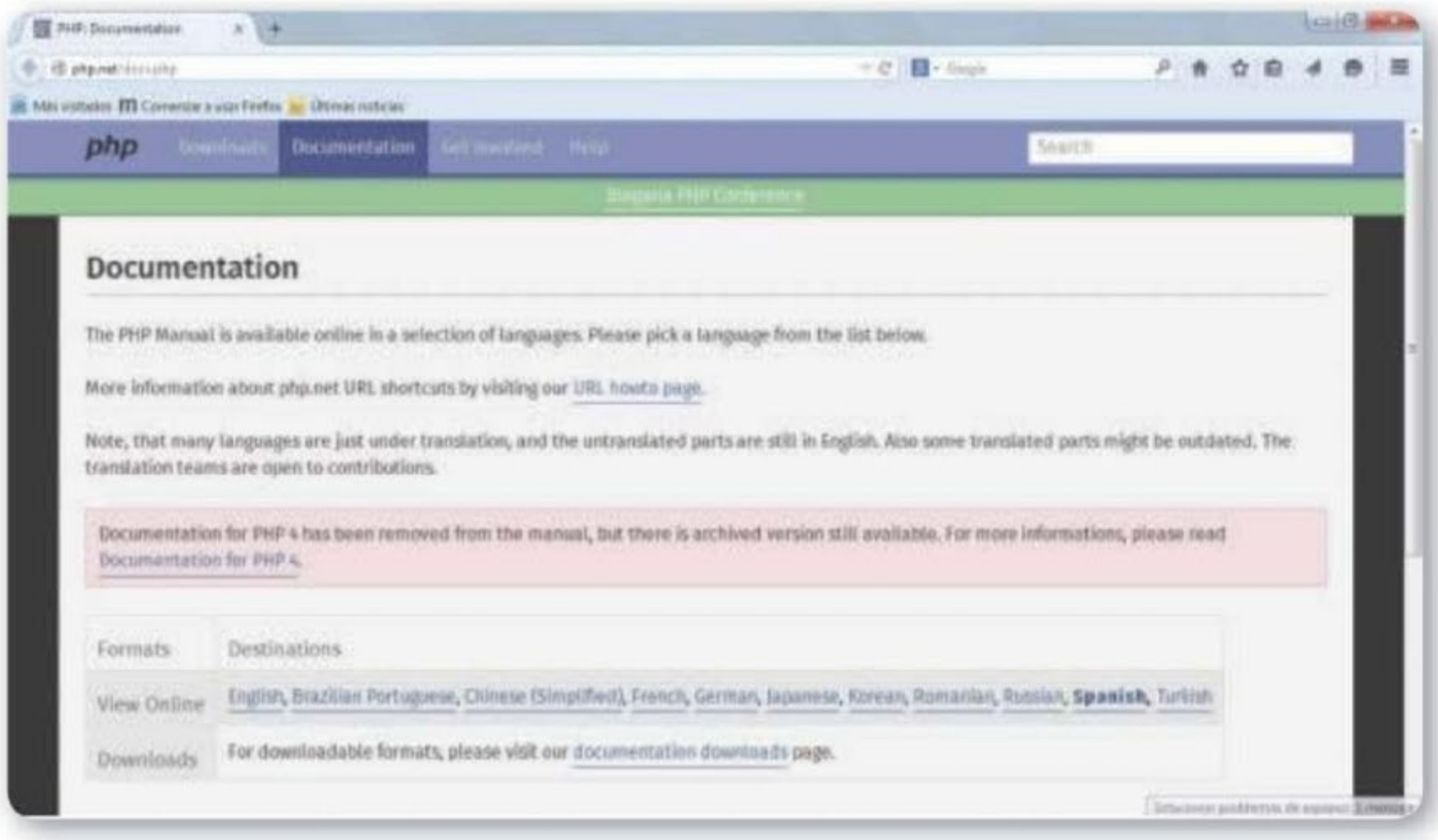


Figura 7. Desde el sitio web **www.php.net/docs.php** accedemos al manual oficial de PHP, disponible en varios idiomas.

SERVIDOR WEB

Es un programa que, a través del protocolo HTTP (*Hypertext Transfer Protocol*), transfiere datos de hipertexto (páginas HTML). Este programa se ejecuta en una computadora llamada "servidor". **Apache**, **Microsoft IIS**, **Google Web Server** son algunos ejemplos de servidores web.

Variables en PHP

El uso de variables en PHP es bastante sencillo. Siempre se comienzan a escribir con el signo de pesos (**$**), seguido por el nombre de la variable. Por ejemplo, si quisiéramos escribir una variable llamada "nombre", deberíamos escribirla de la siguiente manera: **$nombre**.

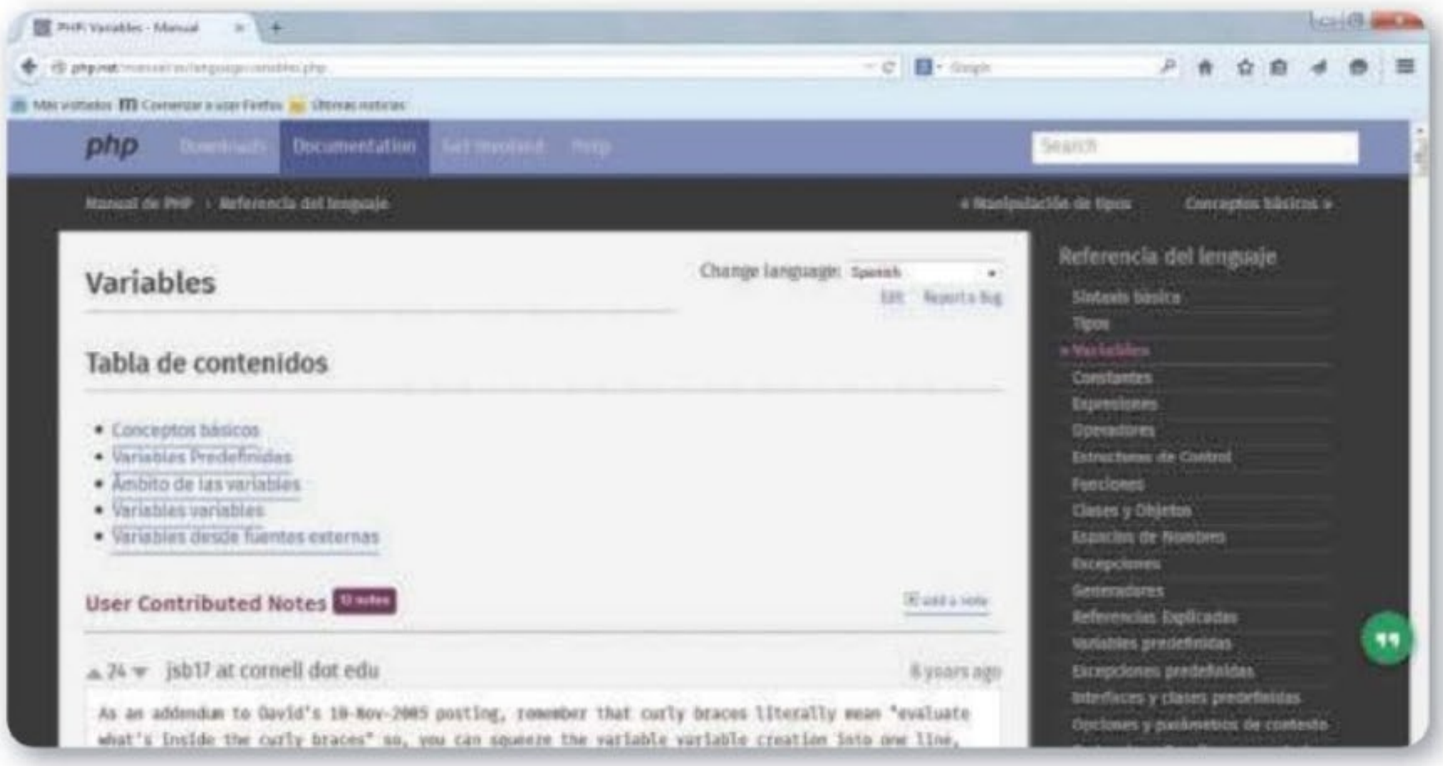


Figura 8. Podemos encontrar información acerca del uso de variables en **http://php.net/manual/es/language.variables.php**

El lenguaje es ***case sensitive***, es decir, es sensible a mayúsculas y minúsculas, por lo que siempre tenemos que respetar la forma de escribir las variables; si escribimos **$nombre** y **$Nombre**, estamos definiendo **dos variables distintas**, porque una está en minúscula y la otra comienza con una mayúscula, lo que el lenguaje interpreta como diferentes variables.

En PHP, al igual que en JavaScript, no hace falta definir el tipo de variable: el lenguaje interpreta de qué tipo es, de acuerdo a su

ETIQUETAS DE PHP

Cuando programamos un archivo en PHP, ya sea externo o incrustado en código HTML, siempre debe comenzar con la etiqueta **<?php** y terminar con la etiqueta **?>**. Los comentarios de una línea se realizan con doble barra (**//**) y, si son multilínea, deben iniciar con barra asterisco (**/***) y terminar con asterisco barra (***/**), como en el siguiente ejemplo: **/* esto es un comentario multilínea */**.

contenido. Por ejemplo, si quisiéramos definir una variable del tipo **interger** (entero, número), basta con hacer lo siguiente:

```
$miNumero = 15;
```

Utilizar archivos externos

Para utilizar archivos externos en PHP, estos deben tener la extensión **.php** y ser llamados desde nuestro documento HTML de la siguiente manera:

```
<?php include "miArchivo.php";?>
```

Esto nos va a ser útil cuando diseñemos el ejemplo del **Capítulo 7** de este libro. Abriendo y cerrando las etiquetas **<?php ?>** le estamos indicando al navegador que lo que va a leer es **código PHP** y, con la palabra reservada **include**, que queremos que busque en el servidor y lea el archivo **"miArchivo.php"**. Esto es muy útil y prolijo, cuando utilizamos un código muy extenso.

RESUMEN

En este primer capítulo, aprendimos los fundamentos básicos de la técnica de diseño web adaptable (*responsive web design*) y conocimos los conceptos de *front-end* (lado del cliente) y *back-end* (lado del servidor). Posteriormente, hicimos un recorrido por las características más importantes de los lenguajes HTML5, CSS3, JavaScript y PHP, lenguajes que, junto a Bootstrap, nos van a permitir diseñar páginas web que puedan visualizarse perfectamente ya sea en una computadora de escritorio, en una tablet, o en un teléfono inteligente.

Actividades

TEST DE AUTOEVALUACIÓN

1. ¿Qué es Bootstrap?
2. ¿Qué proporciona un sistema de rejillas?
3. ¿A qué se denomina *front-end*?
4. ¿Para qué se utilizan las etiquetas **<header>** y **</header>** en HTML?
5. ¿Cuál es la forma más adecuada de escribir un estilo en una página HTML?
6. ¿Cómo se configura un script?

EJERCICIOS PRÁCTICOS

1. Ingrese a **www.w3.org/TR/html** para obtener más información sobre HTML5.
2. Ingrese a **www.php.net/docs.php** y acceda al manual de PHP en español.
3. Establezca qué prefijo de clase debe tener el sistema de rejillas de Bootstrap para una resolución menor a 768 px.
4. Defina cuáles son las etiquetas de apertura y cierre, que le indican al navegador que lo que escribiremos es código PHP.
5. Realice un documento HTML que tenga un enlace externo a un documento **.css** que cambie todos los párrafos ** ** a fuente Arial de tamaño 14 px.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

*02

Iniciando Bootstrap

En este segundo capítulo, observamos cómo funciona la página web oficial del proyecto Bootstrap y cómo aprovechar sus recursos. También aprendemos todo lo necesario para instalar Bootstrap en nuestra computadora y utilizarla como un servidor web. Como vamos a necesitar utilizar un editor, presentamos distintas alternativas gratis, como Sublime Text y Brackets. Finalmente, hacemos nuestra primera plantilla y conocemos otros modelos disponibles.

Crear un servidor con XAMPP

Ya conocimos qué es y para qué sirve Bootstrap. Ahora, para comenzar a utilizar este framework, lo primero que tenemos que hacer antes de instalarlo es tener un servidor web que nos permita alojar allí nuestros proyectos web. De esta manera, podremos probar y ver los resultados de nuestros scripts sin necesidad de tener un **hosting**.

Para utilizar un servidor web en nuestra computadora y probar los proyectos que vamos a realizar a lo largo de todo el libro, recomendamos instalar Apache (**www.apache.org**).

Apache es un servidor web que soporta diferentes plataformas y puede trabajar perfectamente con PHP, Python y Perl, entre otros lenguajes de programación.

Hay en la actualidad varios programas que nos permiten instalar Apache en nuestras computadoras. Dependiendo del sistema operativo que utilicemos, podemos optar por los siguientes:

- **WAMP** (**W**indows, **A**pache, **M**ySQL y **P**HP)
- **MAMP** (**M**ac, **A**pache, **M**ySQL y **P**HP)
- **LAMP** (**L**inux, **A**pache, **M**ySQL y **P**HP)
- **XAMPP** (**X**—cualquier sistema operativo—, **A**pache, **M**ySQL, **P**HP y **P**ERL)

Figura 1. Podemos encontrar más información sobre Apache en su sitio web oficial: **www.apache.org**

Entre varias opciones de programas que nos sirven para instalar Apache, en este libro optamos por uno de los programas más conocidos y populares, XAMPP, que cuenta con todos los requisitos que necesitamos para armar nuestro servidor web.

XAMPP es un software de distribución de Apache gratuito que pertenece a la organización **Apache Friends**. Como bien se indica en su página web oficial, es fácil de instalar y contiene, entre otros componentes, MySQL, PHP y Perl.

Figura 2. Sitio web oficial de la organización **Apache Friends**, en el que se distribuyen de forma gratuita las distintas versiones disponibles de XAMPP: **www.apachefriends.org/es/index.html**

Para instalarlo, una vez que ingresamos al sitio oficial, podemos optar por descargar el archivo de instalación desde la barra de menú principal presionando la opción **Descargar**, o desde la pantalla principal del sitio, tal como se observa en la **Figura 2**.

SERVIDOR WEB O SERVIDOR HTTP

Un servidor web es un programa que nos permite alojar sitios web y transferir esos datos por medio del protocolo **HTTP**. Estos datos son accedidos por medio de un navegador web; es decir que nos permiten visualizar páginas web a través de un navegador, como **Chrome** o **Safari**, entre otros. Estos servidores deben ser alojados en un ordenador con conexión a Internet. El servidor web más utilizado en la actualidad es Apache.

De acuerdo a nuestro sistema operativo —al momento de escribir este libro—, contamos con la versión para:

- **Windows** versión 5.6.6 (PHP 5.6.3)
- **Linux** versión 5.6.6 (PHP 5.6.3)
- **OS X** versión 5.6.6 (PHP 5.6.3)

Una vez que descargamos el archivo de instalación de XAMPP, procedemos a instalarlo como cualquier otro programa, siguiendo las instrucciones de instalación. Entre otras cosas, nos preguntará qué componentes queremos instalar y en qué directorio se instalará XAMPP.

Al finalizar el proceso de instalación veremos que, en el caso de **Windows** (**WAMP**), se instala una carpeta en el directorio **C:** (suponiendo que es nuestro directorio principal del sistema operativo), denominada **xampp**. En el caso de Linux (**LAMP**), generalmente se instala en el directorio **opt/lamp** y, en OS X (**MAMP**), nos va a crear un directorio **xampp** en el directorio principal del disco, con un acceso directo a la carpeta **htdocs** en **Aplicaciones**.

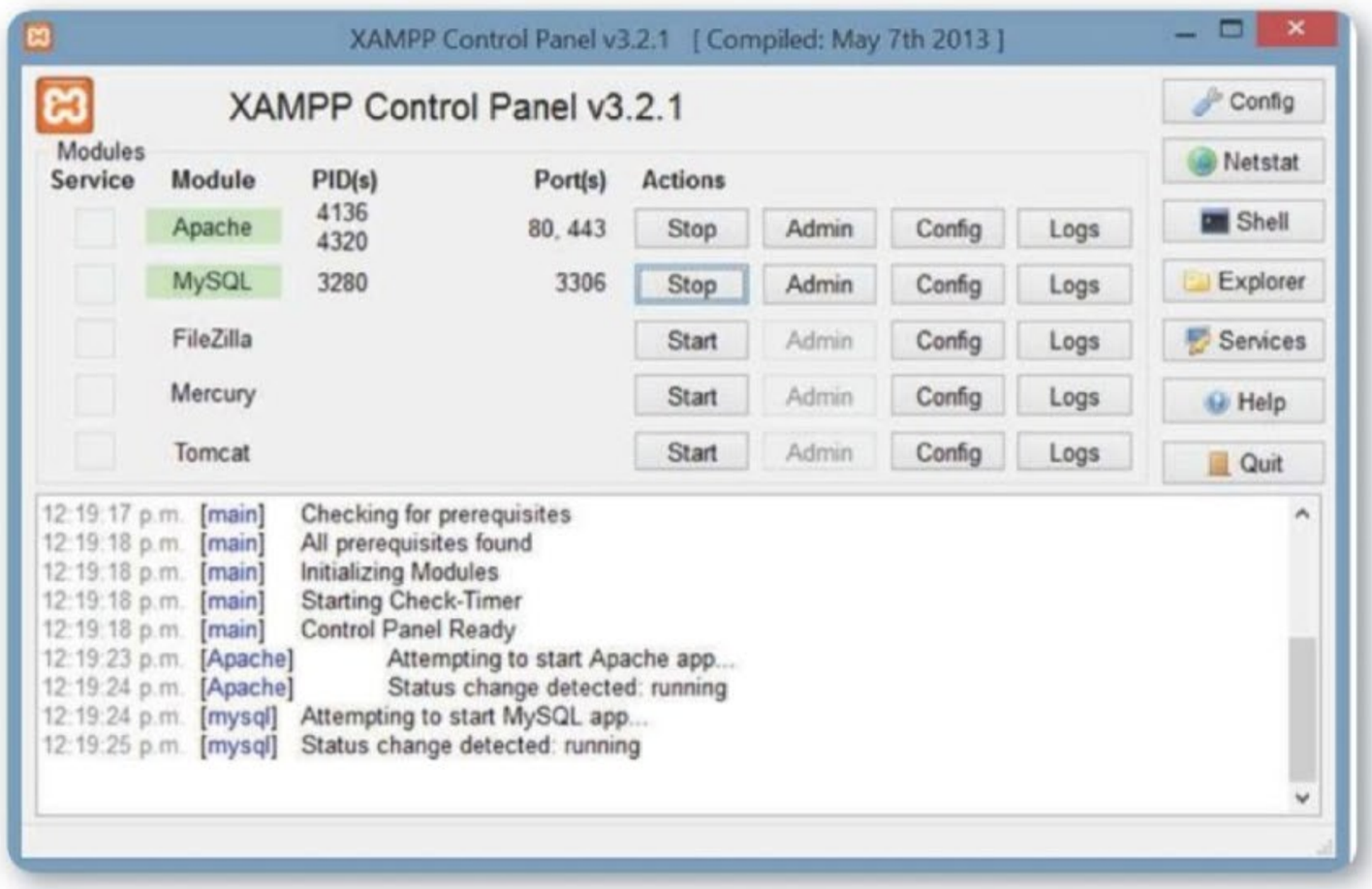


Figura 3. Siempre que trabajemos con un servidor web, debemos verificar que el servicio esté activo. De lo contrario, nos dará error al querer acceder a una URL que tengamos alojada.

Una vez que instalamos el servidor, para alojar nuestras páginas web siempre las crearemos en el directorio **C:\xampp\htdocs**. Como podemos

trabajar con varios proyectos, es conveniente que dentro de la carpeta **htdocs** creemos una carpeta para cada uno, así podremos organizar y administrar los archivos que componen cada proyecto.

Para acceder y visualizar el resultado de nuestra página desde el navegador, deberemos escribir **http://localhost/** seguido del nombre de la carpeta del proyecto. Por ejemplo, si la carpeta del proyecto se llama **miprimerproyecto**, tendremos que escribir en la barra de dirección del navegador **http://localhost/miprimerproyecto**, como podemos observar en la siguiente imagen.

Figura 4. Directorio **htdocs** en donde se encuentra alojado nuestro proyecto web con el nombre **miprimerproyecto**. Aquí lo vemos con el navegador Google Chorme.

Sitio web oficial

En la página web oficial de Bootstrap (**http://getbootstrap.com**), además de la posibilidad de descargar el framework, podemos encontrar diferentes recursos para utilizar con Bootstrap.

En la página de inicio nos encontramos con un botón, **Built with Bootstrap**, que, al presionarlo, nos dirigirá hacia una página que contiene una serie de ejemplos de sitios web desarrollados con Bootstrap. Además, desde la barra de menú de la página principal

USANDO BRACKETS

Hacemos clic en el menú **Archivo** y seleccionamos **Abrir carpeta**. Al seleccionar la carpeta, nos abre un árbol del proyecto en el costado izquierdo, donde vamos a tener todos los directorios y subdirectorios de nuestro proyecto tal cual lo organizamos. Si seleccionamos el archivo **index.html**, en la parte central nos muestra todo el código HTML de este.

podemos acceder a distintos apartados que nos mostrarán ayuda para descargar y utilizar Bootstrap, recursos como plantillas CSS y distintos componentes que podremos utilizar en nuestros proyectos.

Figura 5. Sitio web oficial de Bootstrap, ***Get Bootstrap*** (en español, "conseguir Bootstrap"): **http://getbootstrap.com**

Descargar Bootstrap

Para descargar e instalar Bootstrap, tenemos que acceder a su página web oficial. Una vez allí, lo descargamos desde el botón central **Download Bootstrap** (en español, "descargar Bootstrap") o bien desde el menú principal haciendo clic en **Getting started** (en español, "comenzando").

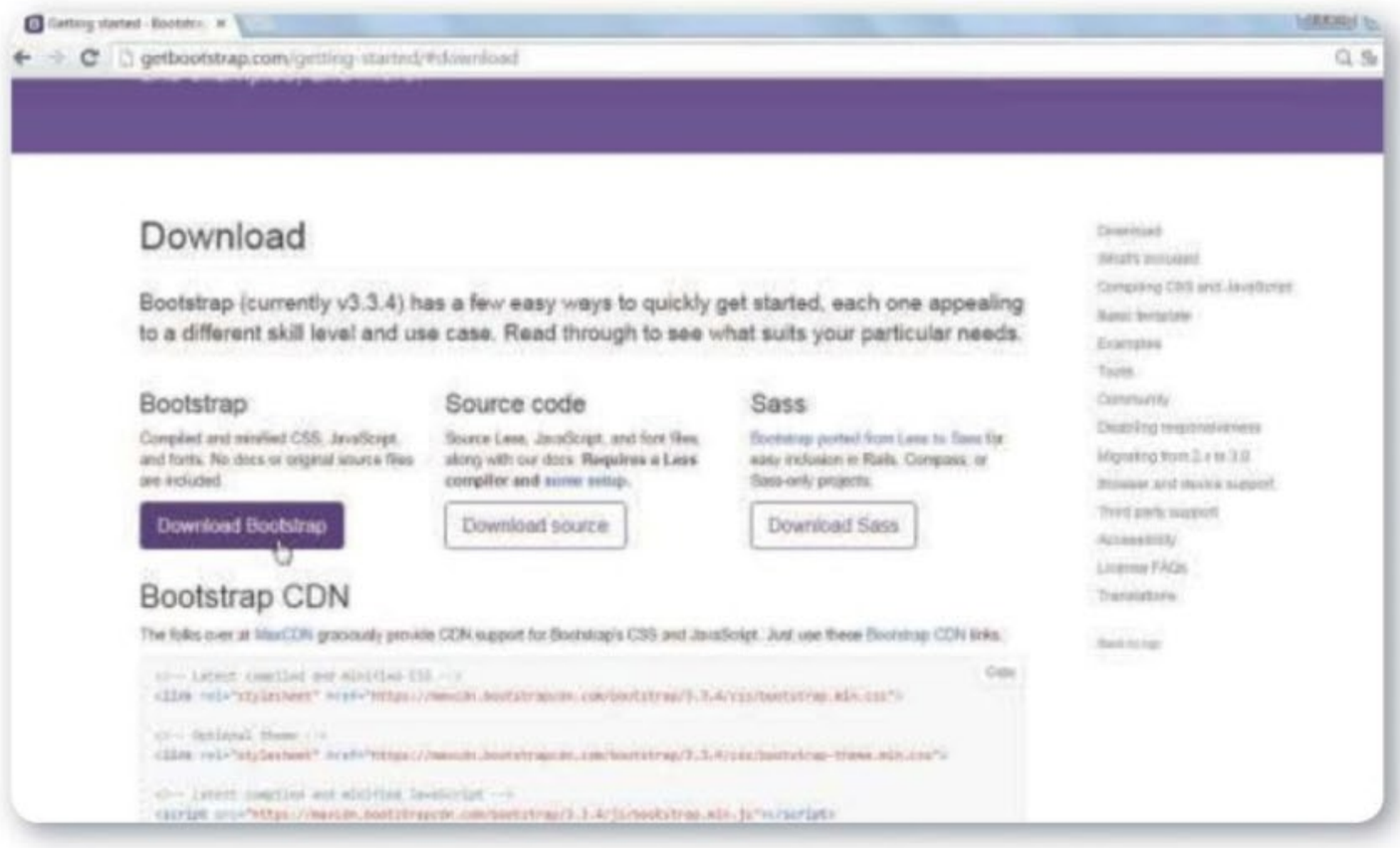


Figura 6. Presionando el botón **`Download Bootstrap`**, descargamos la última versión de los archivos necesarios para trabajar con el framework.

De esta forma, se nos descargará un archivo comprimido que contiene las librerías necesarias para trabajar de manera independiente en nuestro servidor. El nombre del archivo es **bootstrap-X.X.X-dist.zip**, donde la letra **X** nos indica la versión (al momento de escribir este libro, es **3.3.4**).

Para trabajar localmente, simplemente tendremos que descomprimir este archivo en el directorio raíz de nuestro proyecto. Al descomprimir el archivo Zip que descargamos del sitio de Bootstrap, encontraremos tres carpetas: **CSS**, **js** y **fonts**, las cuales contienen archivos con hojas de estilo, librerías de JavaScript e iconos gráficos con los cuales trabajaremos en nuestros proyectos de Bootstrap.

Figura 7. Directorios del contenido de la descarga de la versión 3.3.4 de Bootstrap.

Otra manera de trabajar es enlazando las librerías de Bootstrap por **CDN**. Para eso, desde la página oficial tenemos un apartado denominado **Bootstrap CDN**, con los diferentes enlaces de las librerías.

También podemos descargar el código fuente del framework en el apartado **Source code** (en español, "código fuente").

Instalación

Una vez que descargamos el framework, solo nos resta instalarlo. Tenemos varias formas de enlazar las librerías para comenzar a trabajar con Bootstrap. Si lo hacemos por medio de CDN, tenemos que escribir el siguiente código entre las etiquetas **<head></head>** de nuestro documento HTML:

CDN (CONTENT DELIVERY NETWORK)

Content delivery network (CDN) —en español, "red de entrega de contenidos"— son servidores con distintas ubicaciones en todo el mundo que distribuyen un contenido estático en particular (archivos, imágenes, aplicaciones web, entre otros), permitiendo una actualización permanente y una reducción del tiempo de respuestas entre cliente-servidor.

```
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/css/
bootstrap.min.css">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/css/
bootstrap-theme.min.css">
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.
js"></script>
```

PODEMOS ENLAZAR LAS LIBRERÍAS POR MEDIO DE SERVIDORES EXTERNOS (CDN)

Los enlaces anteriores son a servidores externos (CDN), los dos primeros son a los estilos CSS del framework y, el último, a la librería de Bootstrap, con la extensión **.js** (JavaScript).

En cambio, si descargamos el framework para alojarlo en nuestro servidor —ya sea un hosting contratado o nuestra computadora de manera local—, tenemos que copiar, primero, los archivos que conforman la distribución de Bootstrap dentro de la carpeta de nuestro proyecto. Luego podemos enlazar, desde las etiquetas **<head></head>** de nuestro documento HTML, los dos directorios descargados (**js** y **css**) con sus correspondientes archivos y ubicar en el mismo directorio la carpeta **fonts**, ya que es utilizada por los estilos CSS. De esta forma, tenemos acceso a las librerías y estilos de Bootstrap, como podemos apreciar en el siguiente ejemplo de código:

```
<link rel="stylesheet" href="css/ bootstrap.min.css">
<link rel="stylesheet" href="css/bootstrap-theme.min.css">
<script src="js/bootstrap.min.js"></script>
```

Distintos editores

Después de descargar e instalar el framework, tenemos que elegir un editor para poder comenzar a programar. Existe gran variedad y opciones de todo tipo; en este libro recomendamos dos: **Sublime Text** y **Brackets**, ambos gratuitos y compatibles con los sistemas operativos de Linux, Windows y OS X.

Sublime Text

Este editor se puede descargar de forma gratuita desde su sitio web: **www.sublimetext.com**. La última versión, al momento de escribir este libro, es la versión 2, disponible para Linux, Windows y Mac.

Sublime Text es un editor de código potente con una interfaz sencilla y fácil de personalizar (podemos cambiar desde el tamaño y tipo de fuente hasta los atajos de teclado). Nos permite escribir código para diversos lenguajes, como por ejemplo HTML, PHP, Perl, JavaScript, entre otros. Además nos ahorra tiempo al escribir código, ya que incorpora la herramienta de autocompletado de sintaxis. Esto quiere decir que, a medida que vamos escribiendo, por ejemplo, código HTML, Sublime Text anticipa lo que estamos escribiendo y lo completa por nosotros. Es un editor sencillo de usar y tiene la ventaja de que resalta con colores la sintaxis propia de cada lenguaje, lo que facilita la lectura de nuestro código.

Figura 8. Sitio web oficial de **Sublime Text** desde donde podemos descargar la versión 2, acceder al foro y al soporte técnico, entre otras cosas: **www.sublimetext.com**

Brackets

Es un editor desarrollado por Adobe, que se encuentra en la versión 1.2 y, al igual que Sublime Text, es gratuito.

Brackets soporta una vista en vivo en tiempo real, a través del navegador Google Chrome. Al igual que el editor que mencionamos antes (Sublime Text), al empezar a escribir una etiqueta o palabra reservada, Brackets nos va sugiriendo código, lo que facilita y agiliza notablemente la escritura.

Una de las ventajas de Brackets es que, al ser de código abierto, cualquier usuario con experiencia puede modificar el código y, de esta manera, extender las funcionalidades de este software.

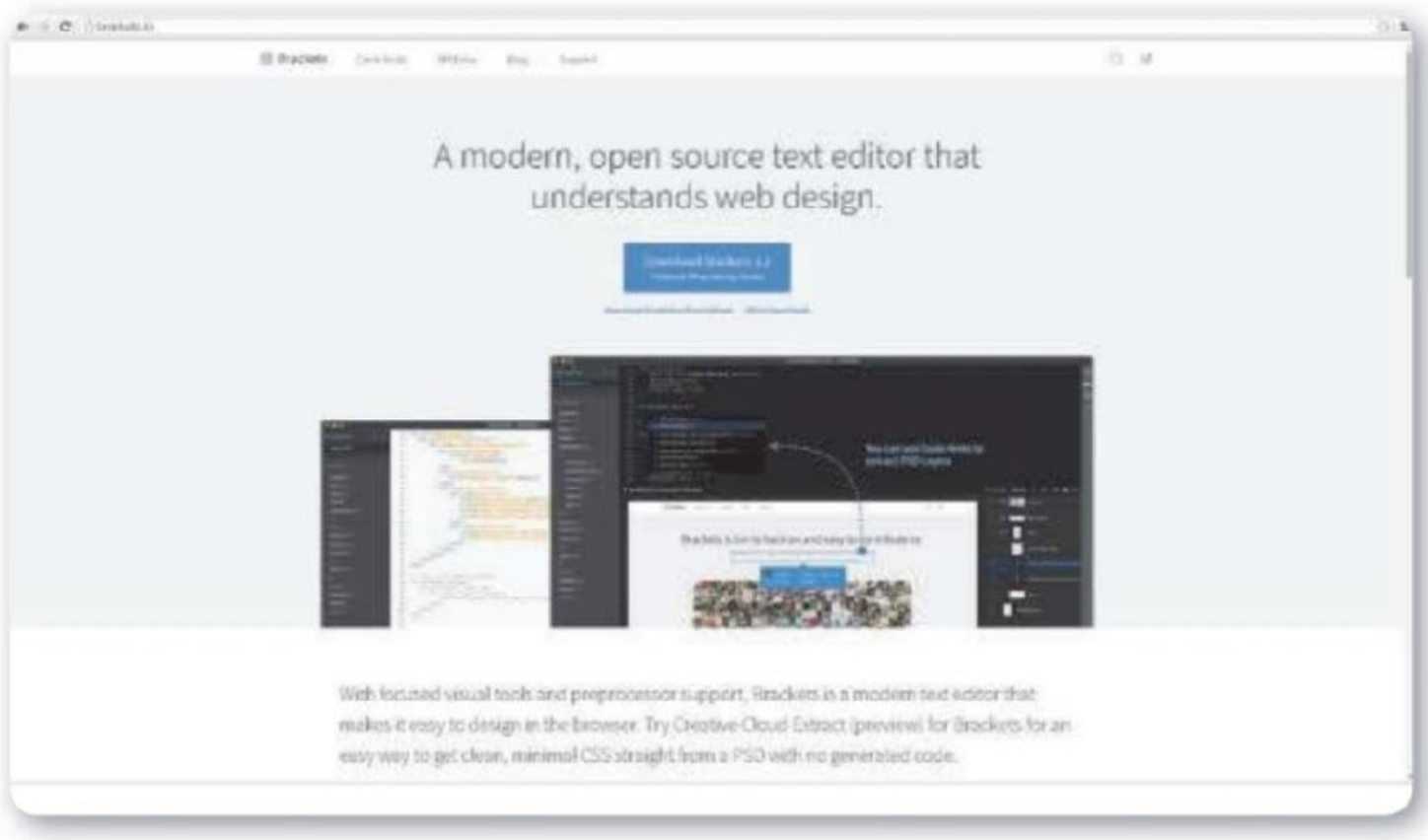


Figura 9. Web oficial de **Brackets by Adobe**, con su correspondiente documentación, blog y soporte: **http://brackets.io**

Ambos editores nos permiten instalar **extensiones** (en inglés, *plugins*), entre las cuales recomendamos instalar Emmet.

Emmet es una extensión que, entre otras cosas, nos facilita la escritura de código y, por lo tanto, nos ahorra mucho tiempo en esta tarea.

Por ejemplo, si queremos comenzar a escribir una página en HTML5, simplemente escribiendo en el editor **html:5** y pulsando la tecla **TAB**, tenemos el siguiente resultado:

```
<!DOCTYPE html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <title>Document</title>
</head>
<body>

</body>
</html>
```

O si queremos escribir una lista desordenada con nueve ítems, simplemente escribiendo **ul>li*9** y pulsando la tecla **TAB**, obtenemos el siguiente resultado:

```
<ul>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
        <li></li>
</ul>
```

Como estos ejemplos, hay muchos más que nos permiten ver cómo esta extensión facilita la —a veces— tediosa tarea de copiar, pegar o simplemente repetir líneas de código.

En el sitio oficial de Emmet, encontraremos toda la documentación correspondiente sobre el uso de esta herramienta que nos va a facilitar enormemente la escritura de código.

Figura 10. Sitio web oficial de **Emmet**, desde donde podemos descargar esta extensión para nuestro editor, ya sea Brackets o Sublime Text: **http://emmet.io**

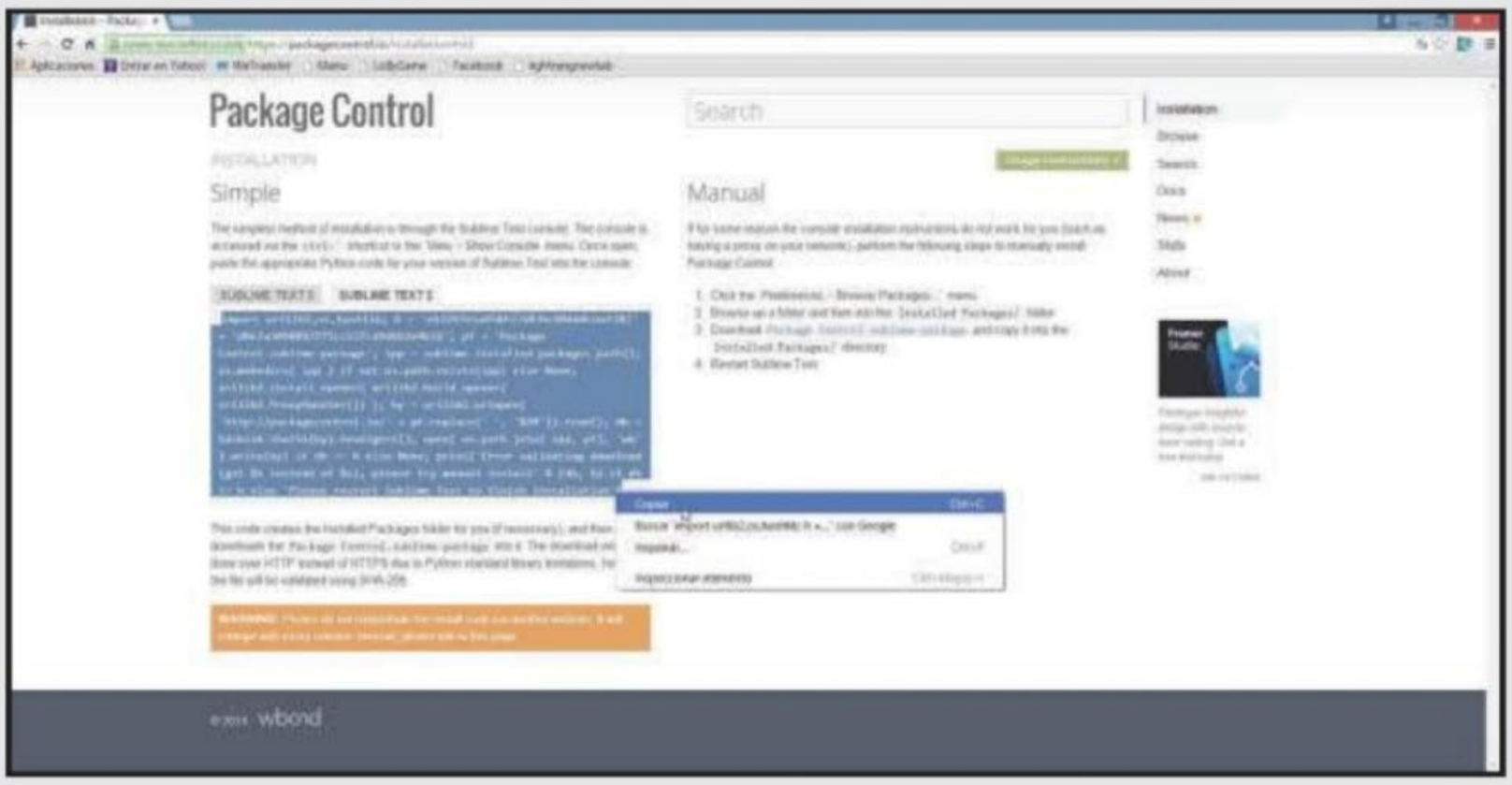
Instalar Emmet en Sublime Text 2

Para instalar Emmet en Sublime Text 2, debemos tener instalado el controlador de paquetes (en inglés, *Package Control*), cuyo sitio web es **https://packagecontrol.io/installation#st2**.

Este controlador se instala desde la consola del editor que utilicemos. Para hacerlo, debemos seguir las indicaciones que se muestran en el siguiente **Paso a paso**.

PAP: INSTALAR EMMET EN SUBLIME TEXT 2

01 Entre al sitio web **https://packagecontrol.io/installation#st2**. Luego copie el código correspondiente a la versión 2 de Sublime Text.

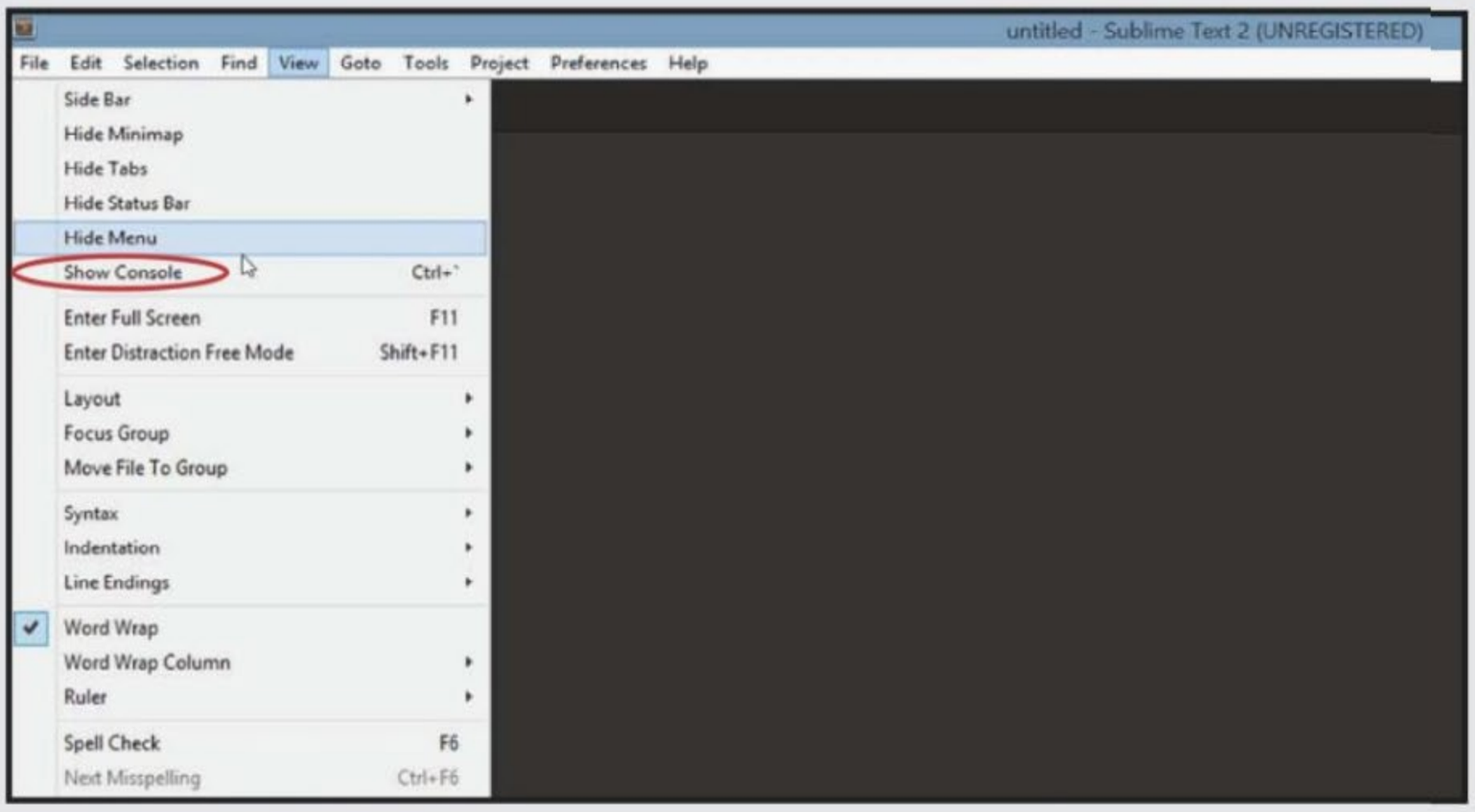


VISUALIZAR EL PROYECTO EN BRACKETS

Hacemos clic en el menú **Archivo**, luego seleccionamos **Vista previa en vivo**. Nos va a mostrar, en el navegador Google Chrome (solo admite este navegador), una vista previa en vivo del documento. Si modificamos algo, veremos cómo va cambiando la vista en vivo. Esto resulta muy útil a la hora de programar.

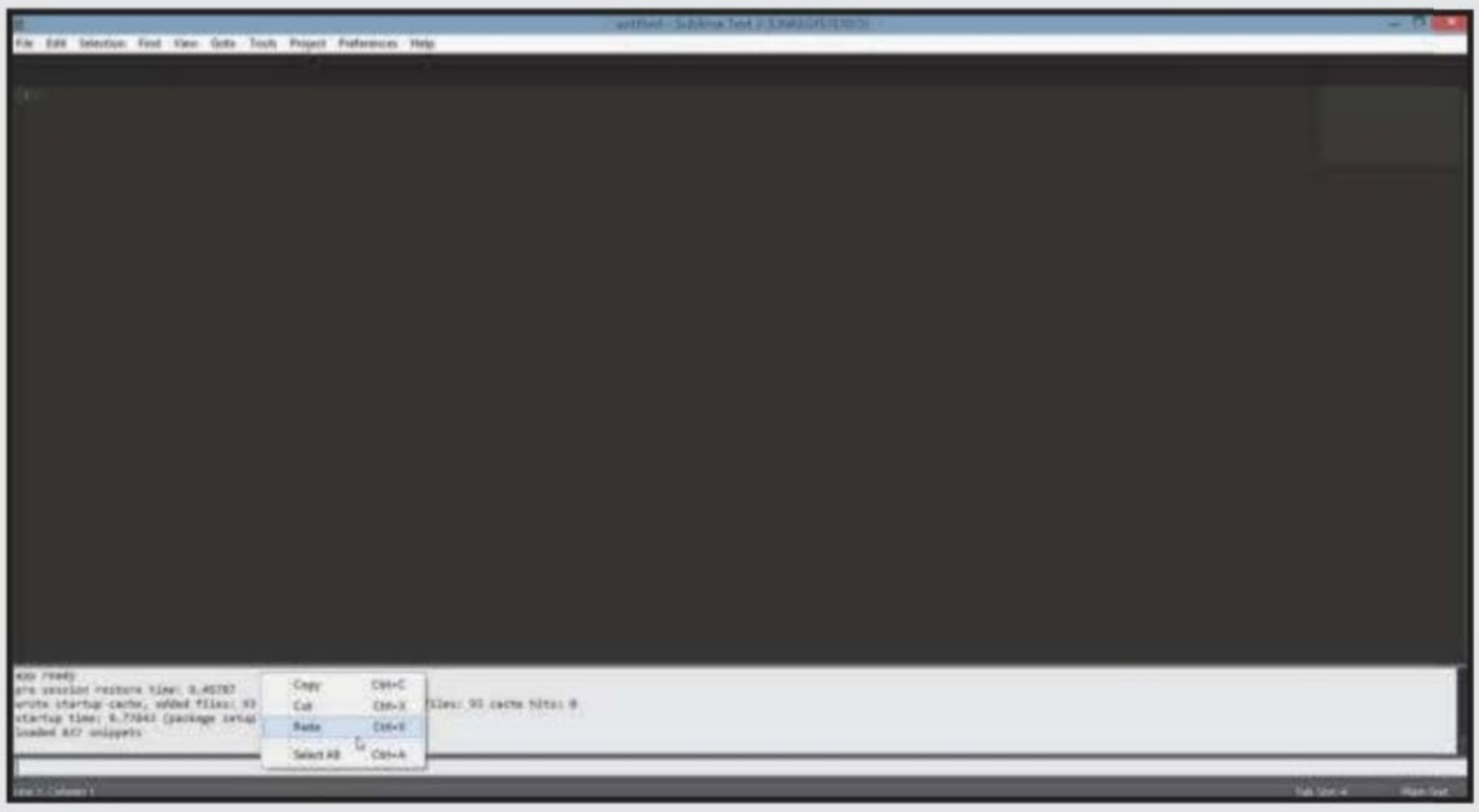
02

En la barra de menú del editor, haga clic en `View` y en el menú que se despliega seleccione Show `Console`.

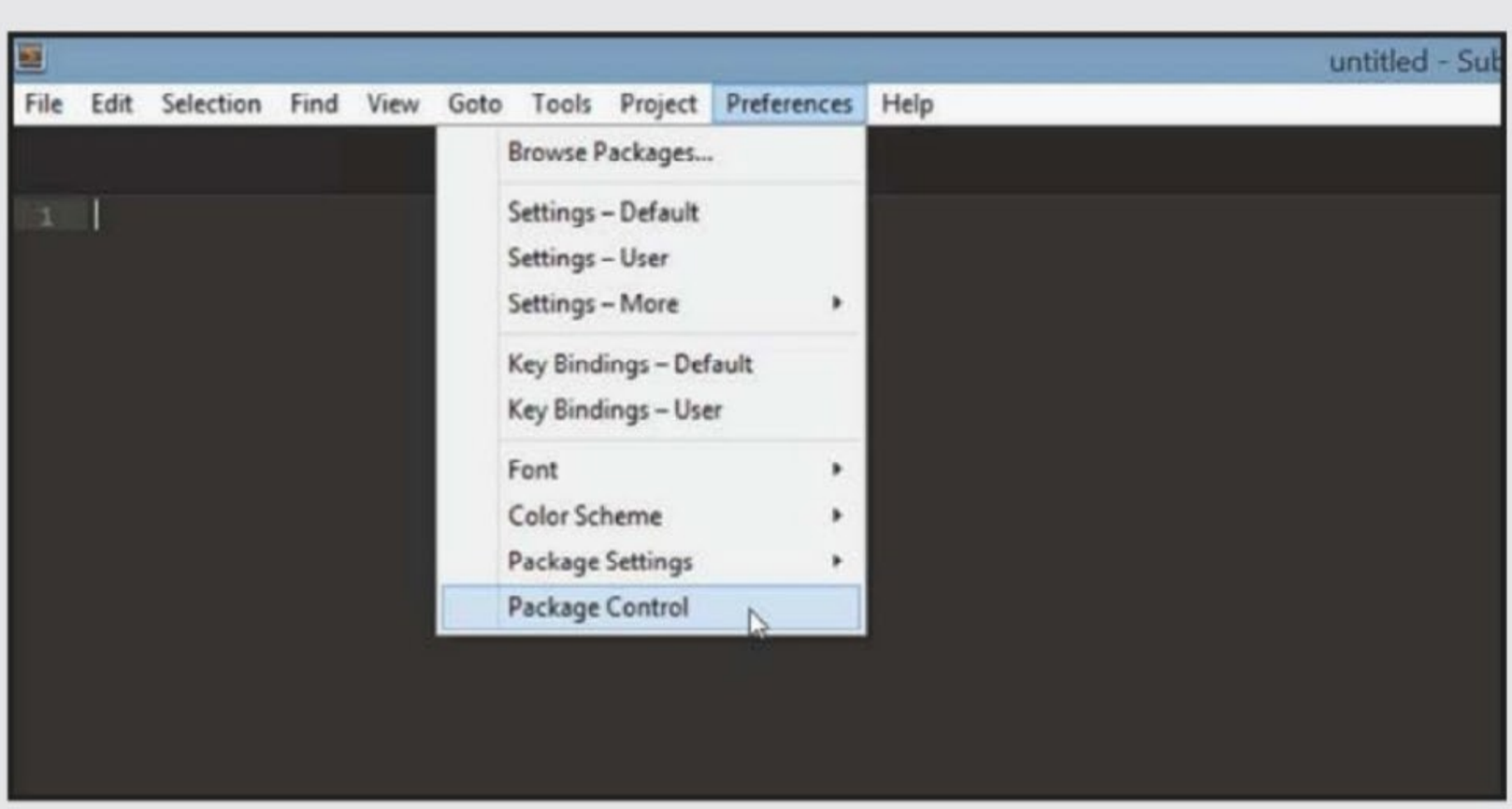


03

En la consola del editor, pegue el código copiado en el Paso `1`. Presione `Enter`. Luego reinicie Sublime Text.

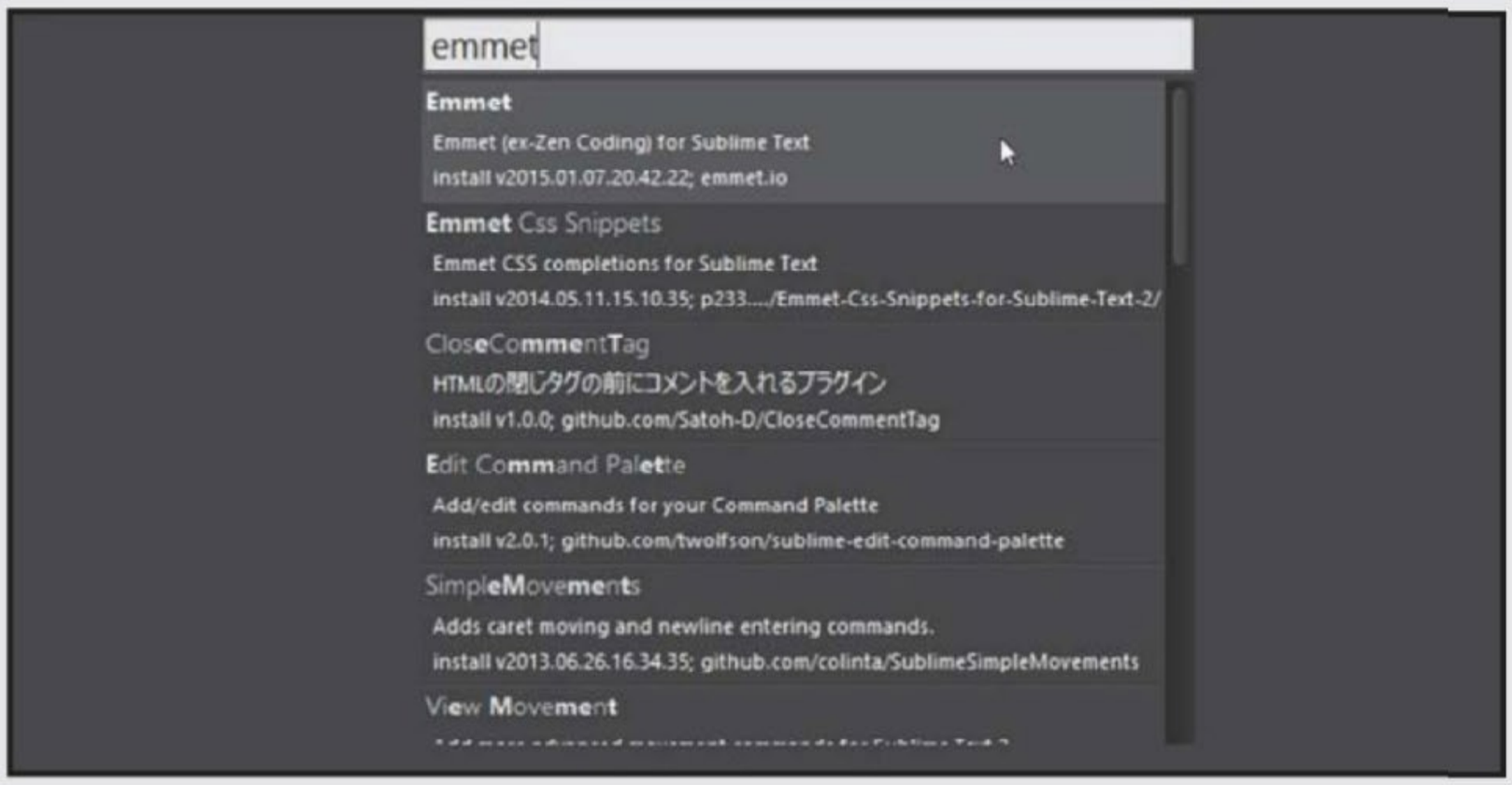


04 En la barra de menú del editor haga clic en `Preferences` y luego seleccione `Package Control`.



05 Se abrirá una ventana con una lista desplegable. Seleccione la opción `Package Control: Install Package`.

06 En la lista que se abre, escriba o busque la opción `Emmet (ex-Zen Coding) for Sublime Text`.



07 Para finalizar la instalación, reinicie Sublime Text.

Instalar Emmet en Brackets

Instalar Emmet en Brackets es mucho más sencillo que en el editor Sublime Text: simplemente abrimos el editor y en el panel derecho de la ventana, presionamos el botón **Gestor de Extensiones**. En el cuadro de diálogo que se abre, en la ficha **Disponibles**, escribimos en el buscador Emmet y, al encontrar la extensión, presionamos el botón **Instalar**.

Con estos pocos pasos ya tenemos instalada en Brackets la extensión de Emmet.

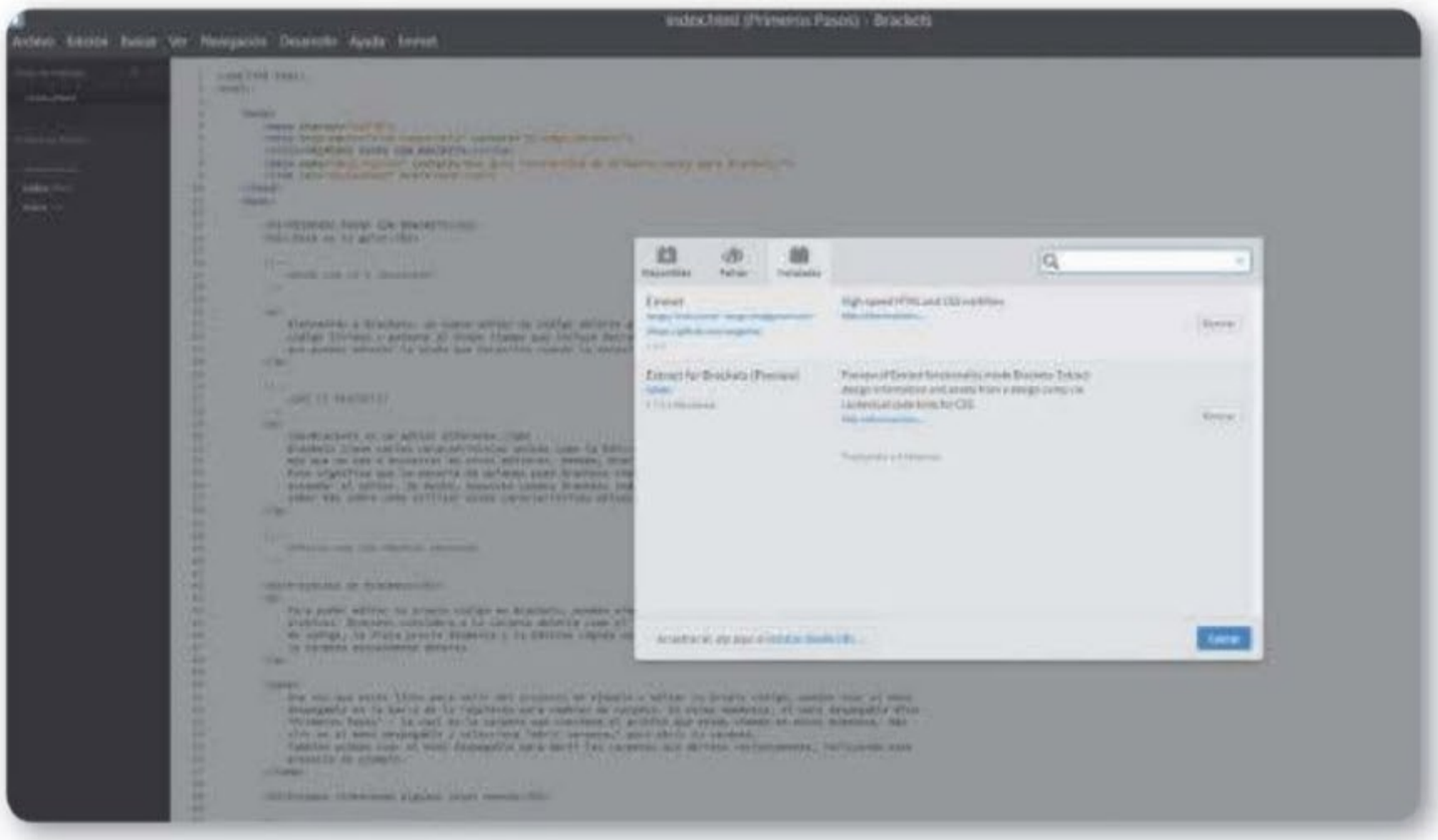


Figura 11. En la ficha **`Instaladas`** de la ventana **`Gestor de Extensiones`**, podemos ver las extensiones que hemos instalado en el editor.

Primera plantilla

Para probar el perfecto funcionamiento de Bootstrap, el editor y nuestro servidor local, vamos a realizar una **plantilla de muestra**.

En nuestra primera plantilla haremos un ejemplo sencillo que contendrá el texto "Este es un ejemplo básico de plantilla", en un título **<h1></h1>**, y una etiqueta resaltada en azul con la misma leyenda, "Este es un ejemplo básico de plantilla", como se muestra en el siguiente código:

```
<!DOCTYPE html>
<html lang="es">
```

```
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Bootstrap Plantilla ejemplo</title>

    <!-- Bootstrap -->

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap
/3.3.4/css/bootstrap.min.css">
      <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/boot-
strap/3.3.4/css/bootstrap-theme.min.css">
      <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.
min.js"></script>

  </head>

 <body>
    <h1> Este es un ejemplo básico de plantilla </h1>
 <span class="label label-primary"> Este es un ejemplo básico de plantilla </
span>

  </body>
 </html>
```

Figura 12. Primera plantilla creada mediante el framework de Bootstrap por CDN, utilizando un título y un label con la clase **`label-primary`**.

Analizando el código anterior, podemos observar que con la etiqueta **<!DOCTYPE html>** estamos definiendo que el tipo de archivo que creamos es un documento HTML y, con el atributo **lang**, que el lenguaje predeterminado del documento será español.

Posteriormente, dentro del **<head></head>** incluimos una serie de etiquetas HTML con las cuales estamos indicando la codificación de caracteres utilizada y la compatibilidad del documento con Internet Explorer:

```
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
```

Luego incluimos la etiqueta **viewport** para que el navegador sepa que tiene que adaptar la página a los diferentes dispositivos:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

A continuación, agregamos las librerías necesarias para trabajar con Bootstrap; en este ejemplo, estamos enlazando por medio de CDN los archivos **bootstrap.min.css**, **bootstrap-theme.min.css** y **bootstrap.min.js**.

```
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/
css/bootstrap.min.css">
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/boot-
strap/3.3.4/css/bootstrap-theme.min.css">
        <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.
min.js"></script>
```

Finalmente, dentro del **<body></body>** se incluye el contenido de la página, como veremos a partir del **Capítulo 3**. A modo de ejemplo, solamente agregamos un título y una etiqueta resaltada:

```
<h1> Este es un ejemplo básico de plantilla </h1>
<span class="label label-primary"> Este es un ejemplo básico de plantilla </span>
```

Otras plantillas

En la página oficial de Bootstrap, en el apartado **Examples** (en español, "ejemplos"), encontramos algunos de los ejemplos que podemos realizar con el framework y que más adelante explicaremos en profundidad, como ser botones, grillas, menús, páginas de login, entre otros.

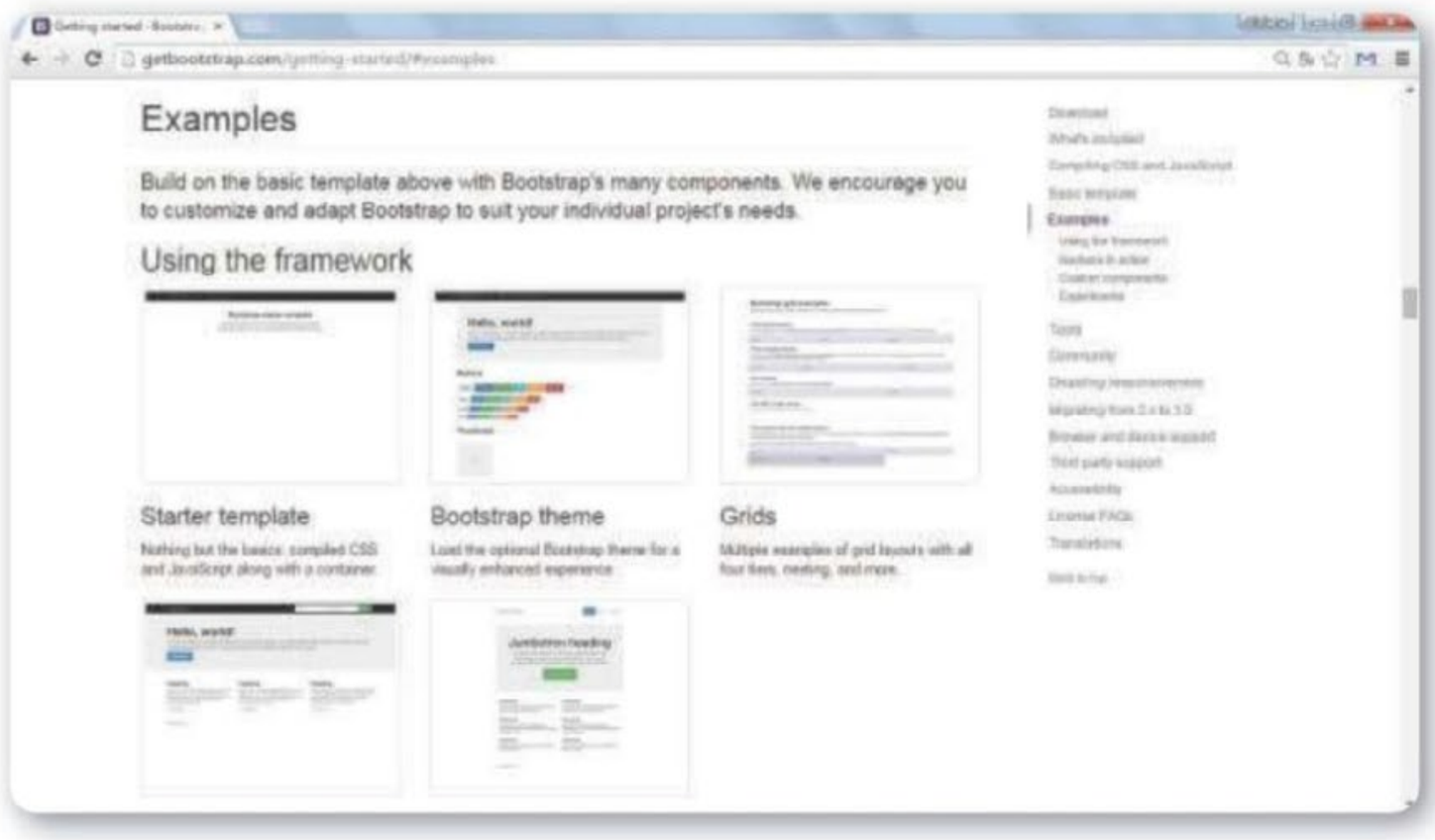


Figura 13. Podemos encontrar plantillas de Bootstrap para usar y editar en **http://getbootstrap.com/getting-started/#examples**

RESUMEN

En este segundo capítulo, conocimos cómo funciona la página web oficial del proyecto Bootstrap y cómo podemos aprovechar sus recursos. Aprendimos a instalar XAMPP en nuestra computadora y hacer de esta un servidor web que nos permita probar nuestros proyectos. Vimos, también, cómo instalar Bootstrap y cómo enlazar las librerías para usarlas en forma local o remota. Por otro lado, conocimos las características de dos editores de texto: Sublime Text y Brackets. Aprendimos a instalar la extensión Emmet en estos editores para extender su funcionalidad. Finalmente, creamos un primer modelo de plantilla en Bootstrap.

Actividades

TEST DE AUTOEVALUACIÓN

1. ¿Qué es un servidor web?
2. ¿Cómo podemos instalar Bootstrap?
3. ¿Cuál es el código que hay que escribir en el **<head>** del HTML para enlazar Bootstrap por medio de CDN?
4. ¿Qué nos permite hacer XAMMP?
5. Mencione alguna de las ventajas que ofrece instalar Emmet en un editor.

EJERCICIOS PRÁCTICOS

1. Instale XAMMP.
2. Reinicie el servicio de Apache y MySQL del servidor web.
3. Instale Sublime Text o Brackets y adicione la extensión Emmet.
4. Programe una lista de 5 ítems utilizando los atajos de Sublime Text con Emmet.
5. Realice una plantilla de Bootstrap con el clásico "Hola mundo!".

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

*03

Maquetación

En este capítulo, conocemos la maquetación web; aprendemos a utilizar Bootstrap a partir de un diseño y a configurar las filas y las columnas del framework. También vemos cómo se comporta Bootstrap en diferentes resoluciones de pantalla, aprendemos a ocultar contenido y conocemos qué son las *media queries*. Por último, realizamos un ejemplo a partir del contenido visto en el capítulo.

↗↗↗

¿Qué es la maquetación?

La maquetación web, o también llamada **diagramación web**, consiste en la distribución y organización de los distintos elementos de nuestra página web a partir de un diseño. Por ejemplo: dónde y qué contenido va en las cabeceras o pie de página, qué artículo poner, dónde y qué enlaces va a tener el menú principal, cómo van a estar distribuidas las imágenes, cuántos documentos HTML vamos a necesitar, entre otras cosas.

Tenemos que considerar que todo empieza por el diseño. Este es realizado por los diseñadores gráficos mediante programas como **Adobe Photoshop** o **Adobe Illustrator**.

Los programadores web antiguamente empleaban las etiquetas HTML **<table></table>** para maquetar; hoy, se utilizan las etiquetas ** ** conjuntamente con hojas de estilo (CSS). Todo proyecto web que nos pidan desarrollar podemos realizarlo también mediante Bootstrap, utilizando el concepto de filas y columnas que maneja muy bien este framework. Con Bootstrap obtendremos un resultado final idéntico al que nos puede llegar a pasar un diseñador gráfico y, a su vez, podremos hacerlo adaptable a cada dispositivo, si es que así nos lo piden.

Por último, una vez que tenemos el diseño y luego la maquetación del sitio web, solo nos resta programarlo para crear la interactividad entre el usuario y el servidor.

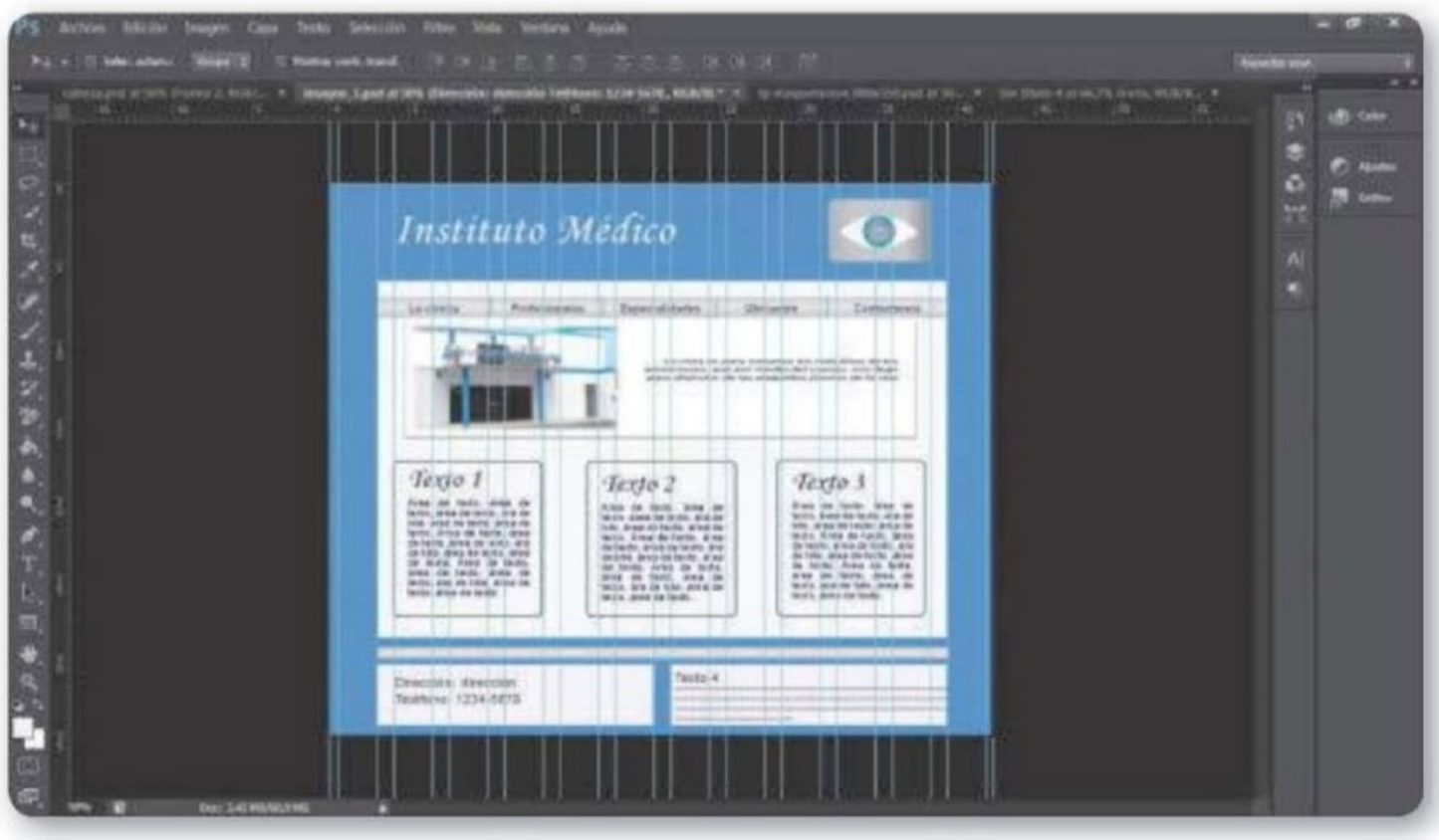


Figura 1. Ejemplo de sitio web maquetado con Adobe Photoshop.

Configuración de filas y columnas

Para realizar una buena maquetación de un sitio web es necesario entender cómo funciona en Bootstrap el concepto de filas y columnas.

Toda fila y columna debe ir siempre en un **contenedor** que se adapta al ancho de la pantalla, que puede tener un **ancho fijo** o un **ancho variable**. Para eso, debemos utilizar la clase **container** o **container-fluid**, respectivamente.

Un contenedor no es más que una etiqueta ** ** que nos va a servir para que podamos agregar nuestras filas y columnas al diseño.

Por ejemplo, si queremos utilizar un contenedor fijo —es decir, basado en pixeles—, empleamos el siguiente código:

```
<div class="container">

</div>
```

Si, en cambio, queremos que el contenido de nuestra página ocupe el 100 % del ancho del dispositivo en el que estamos visualizando la página, emplearemos el siguiente código:

```
<div class="container-fluid">

</div>
```

ETIQUETAS DIV

Las etiquetas de HTML ** ** se utilizan para definir una sección o contenido específico. Unida a las hojas de estilo (CSS), podemos darle a esta sección una ubicación dentro de nuestra página web, así como configurar su tamaño, color, tipo de letra, entre otras cosas. En HTML5, muchas secciones que se realizaban con esta etiqueta fueron reemplazadas por etiquetas nuevas, como **<section></section>** y **<footer></footer>**, entre otras. De esta manera, queda más organizado el contenido de nuestro sitio web.

Cada fila en Bootstrap ocupa **12 columnas** que representan el ancho total de la pantalla dividido en 12. Estas tienen un espacio de 15 px por lado, es decir, un total de 30 px, dado que cada columna tiene dos lados. El contenido de cada columna tiene un margen también de 15 px.

Para cada fila hay que agregar un ** ** con la clase **row**, como podemos observar en el siguiente ejemplo:

```
<div class="container">
    <div class="row">

    </div>
</div>
```

Dentro de cada fila tenemos que agregar las columnas, siempre de acuerdo a lo que necesitemos y teniendo en cuenta que el total de las mismas es siempre 12. A cada columna tenemos que aplicarle el prefijo de la clase, ya sea **.col-xs**, **.col-sm**, **.col-md** o **.col-lg**, de acuerdo al ancho de la pantalla del dispositivo, tal como veremos más adelante en este mismo capítulo.

Por ejemplo, si necesitamos poner en nuestro sitio web tres imágenes, tenemos que colocar en una fila solamente cuatro columnas, porque 4 es el resultado de dividir 12 sobre 3. Entonces, para la primera imagen, tenemos un ancho de cuatro columnas, al igual que para las imágenes restantes.

Figura 2. La suma de todas las columnas no puede superar el total de 12.

En cambio, si necesitamos ubicar cuatro imágenes en nuestra página web, el resultado de dividir 12 sobre 4 nos da 3, que es el número de columnas que vamos a utilizar. Para la primera imagen tenemos, entonces, un ancho de tres columnas, y lo mismo para las imágenes restantes.

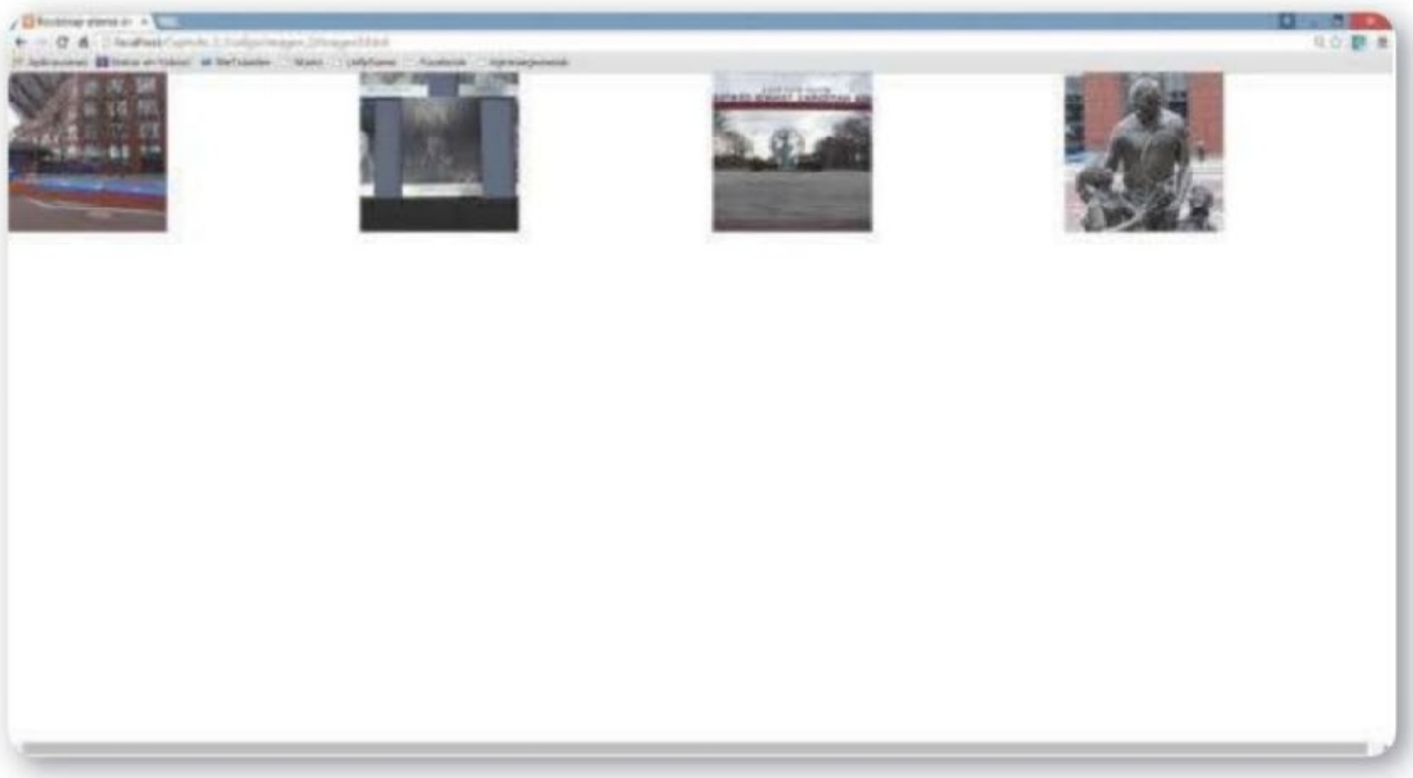


Figura 3. Ejemplo de cuatro columnas visualizado en una pantalla de resolución de 1200 px o superior.

Para codificar este ejemplo, una vez establecido el contenedor, creamos una fila con la clase **row** y, dentro de esta fila, las columnas con las clases **col-xs-***, **col-sm-***, **col-md-*** o **col-lg-***, donde el asterisco (*) representa la cantidad de columnas que queremos que ocupe el **div**. En este ejemplo, usamos la clase **lg** para una resolución de pantalla de 1200 px o más.

```
<body>

    <div class="content">
        <div class="row">
            <div class="col-lg-4" ><img src="img/1.jpg"></div>
            <div class="col-lg-4" ><img src="img/2.jpg"></div>
            <div class="col-lg-4" ><img src="img/3.jpg"></div>
    </div>
    </div>

</body>
```

Si, por ejemplo, queremos colocar en nuestro sitio web siete fotos, podemos utilizar dos filas: en la fila 1 tendríamos que disponer cuatro imágenes y en la fila 2, las tres restantes.

Para esto, tendríamos que dividir, en la fila 1, 12 sobre 4, que nos da 3, y, para la fila 2, dividir 12 sobre 3, que nos da 4. De esta forma, ubicaríamos cuatro imágenes en la primera fila de tres columnas y en la segunda fila, las tres imágenes restantes, que tendrían un ancho de cuatro columnas cada una, como podemos observar en la siguiente imagen.

Figura 4. Podemos observar cómo Bootstrap distribuye dos filas con siete imágenes (cuatro y tres imágenes por fila).

Con la clase **row** añadimos una fila y, dentro de cada una, agregamos las columnas con sus correspondientes imágenes, como se puede observar en el siguiente código:

```
<div class="row">
    <div class="col-md-3" ><img src="img/1.jpg"></div>
    <div class="col-md-3" ><img src="img/2.jpg"></div>
```

CREAR COLUMNAS CON SUBLIME TEXT

Para escribir rápidamente el código para crear tres columnas de un bloque de cuatro para una resolución extra small, en **Sublime Text** podemos escribir **div.col-xs-4*3** y luego presionar la tecla **TAB**. Automáticamente veremos cómo se crean tres **div** de tres columnas.

```
        <div class="col-md-3" ><img src="img/3.jpg"></div>
        <div class="col-md-3" ><img src="img/4.jpg"></div>
</div>
<div class="row">
        <div class="col-md-4" ><img src="img/5.jpg"></div>
        <div class="col-md-4" ><img src="img/6.jpg"></div>
        <div class="col-md-4" ><img src="img/7.jpg"></div>

</div>
```

Como podemos observar en todos estos ejemplos, siempre disponemos de un ancho de 12 columnas por cada fila, y, de esta forma, tenemos que utilizar esa medida para ubicar los distintos elementos en nuestra página web.

¿Cómo adaptar nuestro diseño?

Hasta hace unos pocos años atrás, la mayoría de los usuarios accedían a Internet mediante una computadora de escritorio o notebook. Hoy en día, con el advenimiento de las tablets y los teléfonos inteligentes nos encontramos con que más usuarios emplean estos dispositivos para navegar por los diferentes sitios. Es por esto que a la hora de diseñar un sitio web debemos pensar que puede ser visualizado en diversos dispositivos, los cuales pueden tener distintos tamaños y resoluciones de pantalla.

Por este motivo, debemos diseñar un solo sitio y que el contenido se organice adaptándose automáticamente a todos los dispositivos, ya sea un smartphone, una tablet o una computadora de escritorio.

Para lograr esto, seguiremos la filosofía de trabajo "**Mobile First**", que plantea que partamos por el diseño de la interfaz del sitio para una pantalla chica y posteriormente vayamos adaptándolo para un dispositivo de pantalla mediana y luego para uno de pantalla grande.

Para adaptar nuestro diseño, ya sea para una tablet, teléfono inteligente o computadora de escritorio, en Bootstrap contamos

con medidas que contemplan estas diferentes resoluciones de dispositivos. Por lo tanto, no solo tenemos que tener presente el ancho de 12 columnas para ubicar los distintos elementos en nuestro sitio web, sino que también tenemos que tener en cuenta el tipo de dispositivo en el cual se va a visualizar el sitio.

ADAPTAR LA WEB A CADA RESOLUCIÓN DE PANTALLA ES MÁS FÁCIL CON LAS CLASES DE BOOTSTRAP

Por ejemplo: un menú principal, en una resolución de pantalla de un ancho de 1200 px, no se va a ver igual en una resolución de un ancho de 768 px. Y si queremos hacer la página web para ambas resoluciones, tenemos que saber bien cuándo cambiará la forma en que se visualice el menú; es decir, programar que cuando la resolución de pantalla cambie de 1200 px a 768 px, el menú principal se visualice de otra forma acorde a esa resolución.

Para todo esto, el framework nos proporciona **clases** que nos facilitan la tarea cuando estamos haciendo una página web adaptable. Así, de acuerdo a la resolución del dispositivo con que se visualice la página web, esta tendrá un comportamiento distinto adaptándose a cada una de las resoluciones disponibles.

Como venimos mencionando desde el **Capítulo 1**, tenemos que tener presente con qué resolución vamos a trabajar y cuándo necesitamos que el contenido cambie si la resolución es menor o mayor a la que estemos utilizando.

Si trabajamos para una resolución mayor o igual a 1200 px, trabajamos con el prefijo de clase **.col-lg**. Y si, por ejemplo, tenemos 4 columnas, cuando baje de esa resolución —es decir, sea menor a 1200 px— se verán de forma vertical (una debajo de la otra) y no de forma horizontal (una al lado de la otra), como deberían verse normalmente.

MOBILE FIRST

Mobile First consiste en una filosofía de trabajo, desarrollada por **Luke Wroblewski**, que postula que se debe diseñar primero en la versión para teléfonos inteligentes antes que la versión para escritorio, de forma tal de jerarquizar los contenidos del sitio web según su importancia y ubicación.

```
    <div class="row">

        <div class="col-lg-3" style="background-color:#aaa" ><h3>Columna 1</
h3></div>
        <div class="col-lg-3"style="background-color:#bbb" ><h3>Columna 2</
h3></div>
        <div class="col-lg-3" style="background-color:#aaa"><h3>Columna 3</
h3></div>
        <div class="col-lg-3" style="background-color:#bbb" ><h3>Columna 4</
h3></div>

    </div>
```

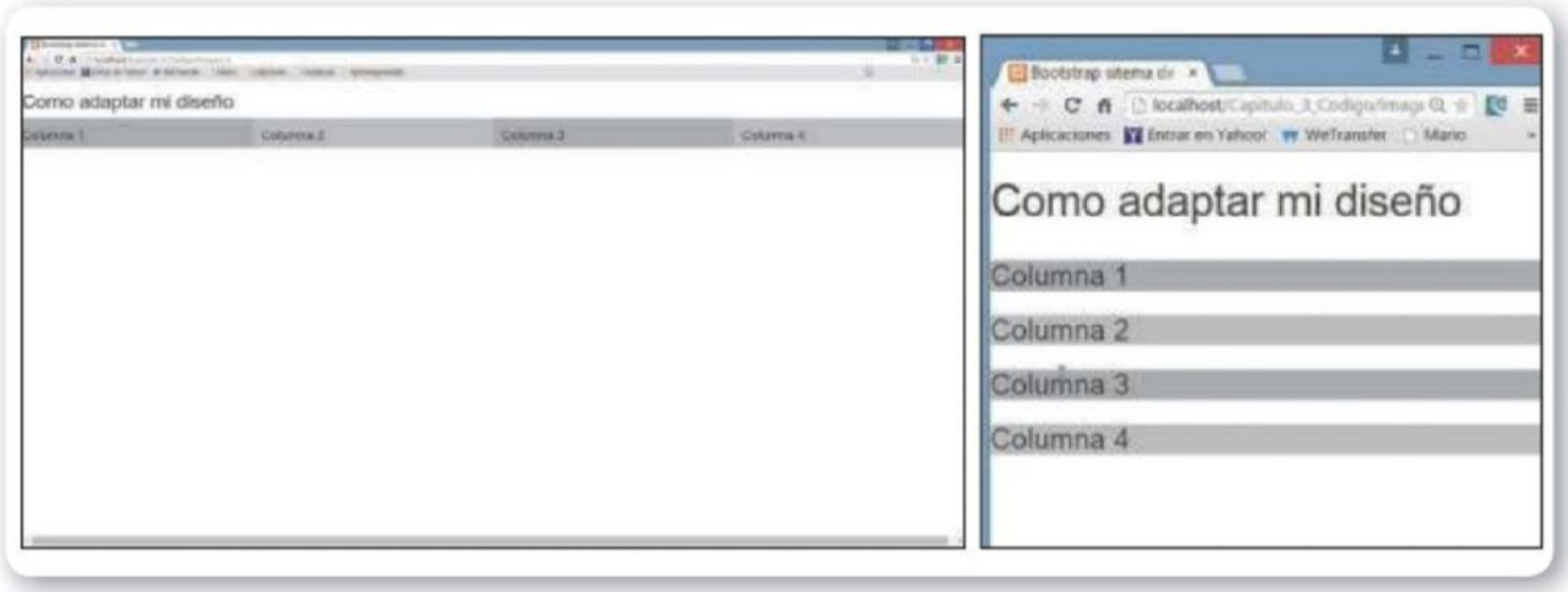


Figura 5. La primera imagen se ve en una resolución de 1200 px y la otra en una resolución menor a 1200 px.

Esto mismo ocurriría si siguiéramos la lógica de Bootstrap, donde, como dijimos anteriormente, si la resolución es menor a 768 px (como en un teléfono inteligente), el prefijo de la clase debe ser **.col-xs**. Para

LUKE WROBLEWSKI

En 2009 Luke Wroblewski fue quien introdujo por primera vez la expresión **"Mobile First"**. Es actualmente uno de los directores de producto de Google y autor de tres libros de diseño web: *Mobile First*, *Web Form Design* y *Site-Seeing: A Visual Approach to Web Usability*. Para mayor información, se puede visitar su sitio web: **www.lukew.com**

mayor o igual a 768 px, el prefijo de la clase es **.col-sm** y para mayores o iguales a 992 px, **.col-md**. Y como vimos en este último ejemplo, si son mayores o iguales a 1200 px, corresponde el prefijo de la clase **.col-lg**.

En la siguiente tabla se resumen los prefijos de clase que utiliza Bootstrap para establecer los puntos de cortes según el tipo de pantalla.

PREFIJOS DE CLASE EN BOOTSTRAP

TIPO DE DISPOSITIVO	RESOLUCIÓN	PREFIJO DE CLASE
Grande (*large devices*)	A partir de 1200 px	**.col-lg**
Medio (*medium devices*)	A partir de 992 px	**.col-md**
Tipo tablet (*small devices*)	A partir de 768 px	**.col-sm**
Dispositivos móviles (*extra small devices*)	Menos de 768 px	**.col-xs**

Tabla 1. Prefijos de clase que establecen puntos de corte según el tipo de pantalla del dispositivo.

Dos prefijos de clase

Otra opción también importante e interesante es que podemos incluir dos prefijos en una misma clase. De acuerdo al dispositivo en que se vea nuestra página web, podemos utilizar una cantidad de columnas para una resolución de pantalla y otra para otro tipo de resolución. Es decir, podemos hacer que el diseño no solamente se adapte a una determinada resolución, sino que, además, cambie, para hacer una experiencia de usuario todavía mejor.

GESTIONAR EL ORDEN DE LAS COLUMNAS

Podemos gestionar el orden de visualización de los elementos en Bootstrap de acuerdo al contexto en que se visualicen. Para ello, disponemos de las clases **push** y **pull**. Con la clase **push** desplazamos el elemento a la derecha el número de columnas que especifiquemos y con la clase **pull** hacia la izquierda, el número de columnas que especifiquemos.

Por ejemplo, tenemos dos columnas que en una resolución de pantalla media (**col-md**) abarcan 10 y 2 de ancho respectivamente, lo que da un total de 12, como es regla general en el framework. Pero queremos que en resoluciones chicas (dispositivos móviles, **col-xs**) tengan un ancho de 8 y 4; entonces tenemos que hacer lo siguiente:

```
<div class="row">
    <div class="col-xs-8 col-md-10" style="background-color:#bbb">8 xs o 10 md</div>
    <div class="col-xs-4 col-md-2" style="background-color:#aaa">4 xs o 2 md</div>
</div>
```

En un mismo **div** podemos poner más de una clase. En el ejemplo anterior, empleamos dos clases (**col-xs-8** y **col-md-10**) de forma tal que si la resolución de pantalla es mayor o igual a 992 px, las dos columnas ocuparían 10 y 2, y, si es una resolución menor o igual a 768 px, 8 y 4 de ancho.

Diseño fullwidth

En ciertas ocasiones, queremos ubicar en nuestra página un contenido en una columna y que esta ocupe el 100 % de la pantalla del dispositivo donde el sitio será visualizado.

Como mencionamos anteriormente, cuando creamos un contenedor en Bootstrap, este nos deja por defecto un margen izquierdo y otro derecho, que observaremos al visualizar la página en cualquier dispositivo, salvo en aquellos con resolución extra small (**.xs**).

Para lograr que el contenido ocupe el ancho total de la pantalla —es decir, que tenga un **diseño fullwidth**—, en lugar de crear un contenedor, usaremos la clase **full-width**, como vemos en el ejemplo:

```
<div class="full-width">
    <h1>Ejemplo de diseño fullwidth</h1>
    <p>Este es un ejemplo donde usamos la clase "full-width" en lugar de crear un contenedor con la clase "container". De esta forma lograremos tener un diseño fullwidth </p>
    </div>
```

En el ejemplo, creamos un **div** con la clase **full-width** y, dentro de este div, incluimos un título y un párrafo.

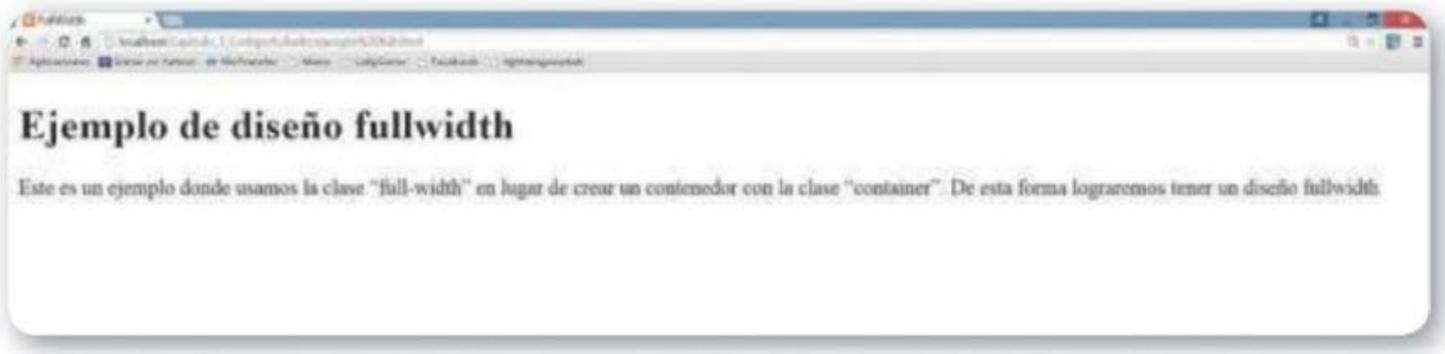


Figura 6. Podemos también hacer un **diseño fullwidth** sin dejar de usar la grilla de Bootstrap.

Viewport

Para asegurarnos de que nuestro sistema esté optimizado y se adapte a dispositivos móviles, es necesario agregar en las etiquetas **<head> </head>** de nuestra página HTML la etiqueta meta **viewport**.

Una **etiqueta meta** (en inglés, ***meta-tag***) es la que se utiliza para agregar información de nuestro sitio web a los motores de búsqueda o navegadores, y solo es visible para los programadores (se mantiene oculta para los usuarios comunes).

```
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
```

En este ejemplo de código le estamos indicando al navegador que adapte el contenido al ancho total (100 %) de la pantalla del dispositivo (**width=device-width**). Además, con **initial-scale** estamos controlando el nivel de zoom. Para deshabilitarlo, simplemente hay que agregar **user-scalable=no**; con esto, no se podrá hacer zoom en nuestra página web.

ROTACIÓN DE PANTALLA

Hoy en día nos encontramos con teléfonos inteligentes —como el iPhone 6 Plus— que, al rotar la pantalla de vertical a horizontal, produce un cambio de categoría, es decir que pasa de tener una resolución **xs** (*extra small*) a una categoría **sm** (*small*).

De esta forma, nuestro documento HTML listo para comenzar a trabajar en Bootstrap quedaría así:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Bootstrap </title>

    <!-- Bootstrap -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/
css/bootstrap.min.css">
      <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/boot-
strap/3.3.4/css/bootstrap-theme.min.css">
      <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.
min.js"></script>
  </head>
  <body>

  </body>
</html>
```

Ocultar contenido

Podemos mostrar u ocultar contenido en función del dispositivo desde el cual visualicemos nuestra página web: así, por ejemplo, si accedemos al sitio desde una computadora de escritorio podemos ver todo el contenido y cuando los hacemos desde un dispositivo móvil, el contenido menos relevante queda oculto. Para ocultar contenido o mostrarlo, Bootstrap utiliza las clases **show** y **hidden** respectivamente. Es equivalente a utilizar, en CSS, la propiedad **display** y pasarle por valor **block** o **hidden**.

LAS CLASES SHOW Y HIDDEN PERMITEN MOSTRAR U OCULTAR CONTENIDO DE NUESTRO SITIO WEB

En el siguiente ejemplo, vemos cómo ocultar, mediante la clase **hidden**, la primera columna: así, cuando estamos en una pantalla con una resolución de menos de 768 px, la misma no se hace visible:

```
<div class="content">
        <div class="col-xs-12">
          <h1>Ejemplo</h1>
        </div>
                    <div class="row">
                              <div class="col-xs-8 col-md-10 hidden-xs "
style="background-color:#bbb"><h3>Oculta en xs pero visible en md</h3></div>
                    <div class="col-xs-4 col-md-2 " style="background-
color:#aaa"><h3>Visible</h3></div>
                    </div>
          </div>
```

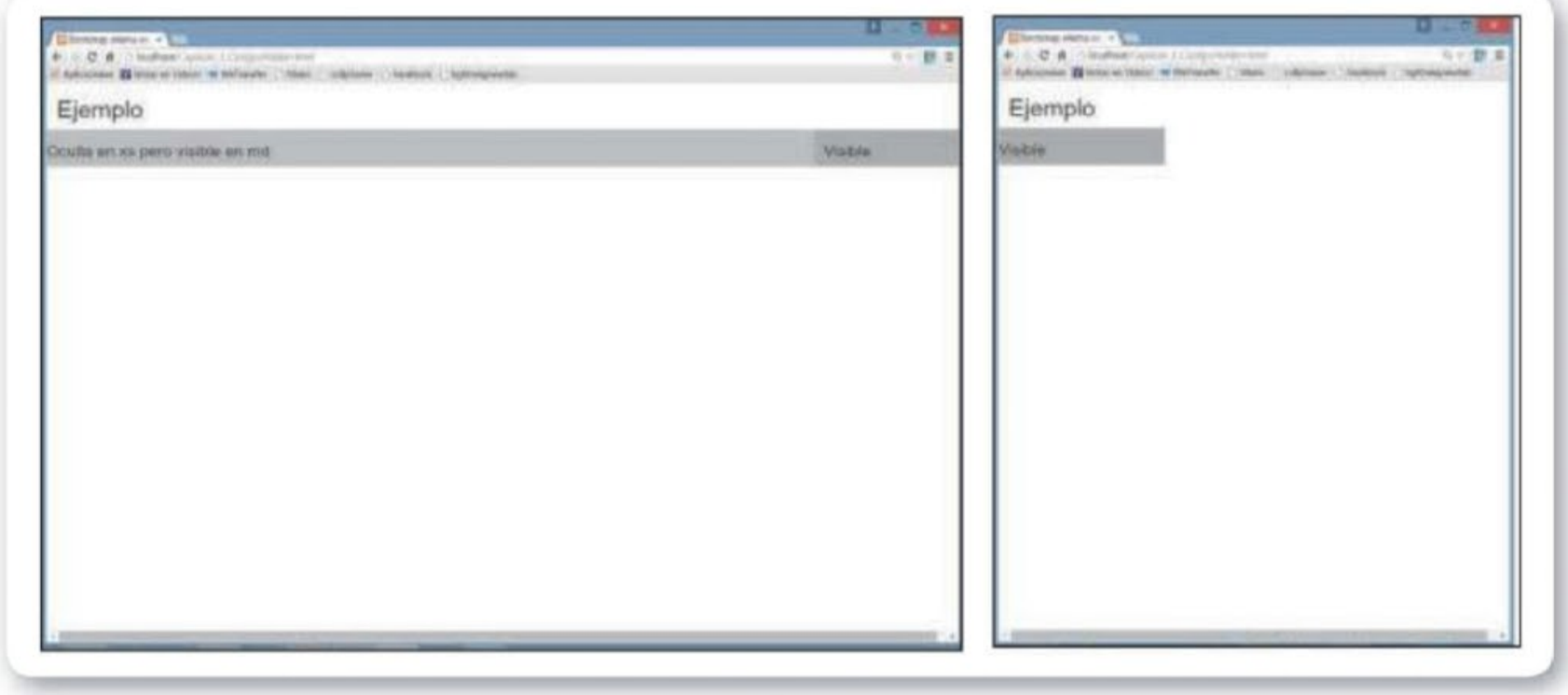


Figura 7. Al visualizar la página en una resolución menor, se oculta una columna.

También contamos con la clase **visible** para ocultar o mostrar contenido de acuerdo a una resolución de pantalla específica, como podemos observar en este ejemplo:

```
<div class="content">
        <div class="row">
         <div class="col-xs-3 visible-lg " style="background-color:#aaa" > <h3>Se
muestra en resoluciones mayores a 1200px </h3></div>
                    <div class="col-xs-3"style="background-color:#bbb" >
<h3>visible</h3></div>
```

```
                    <div class="col-xs-3" style="background-
color:#aaa"><h3>visible</h3></div>
                    <div class="col-xs-3" style="background-color:#bbb"
><h3>visible</h3></div>
          </div>
    </div>
```

Si la resolución de pantalla es mayor a 1200 px (prefijo de la clase **lg**) se va a mostrar dicha columna.

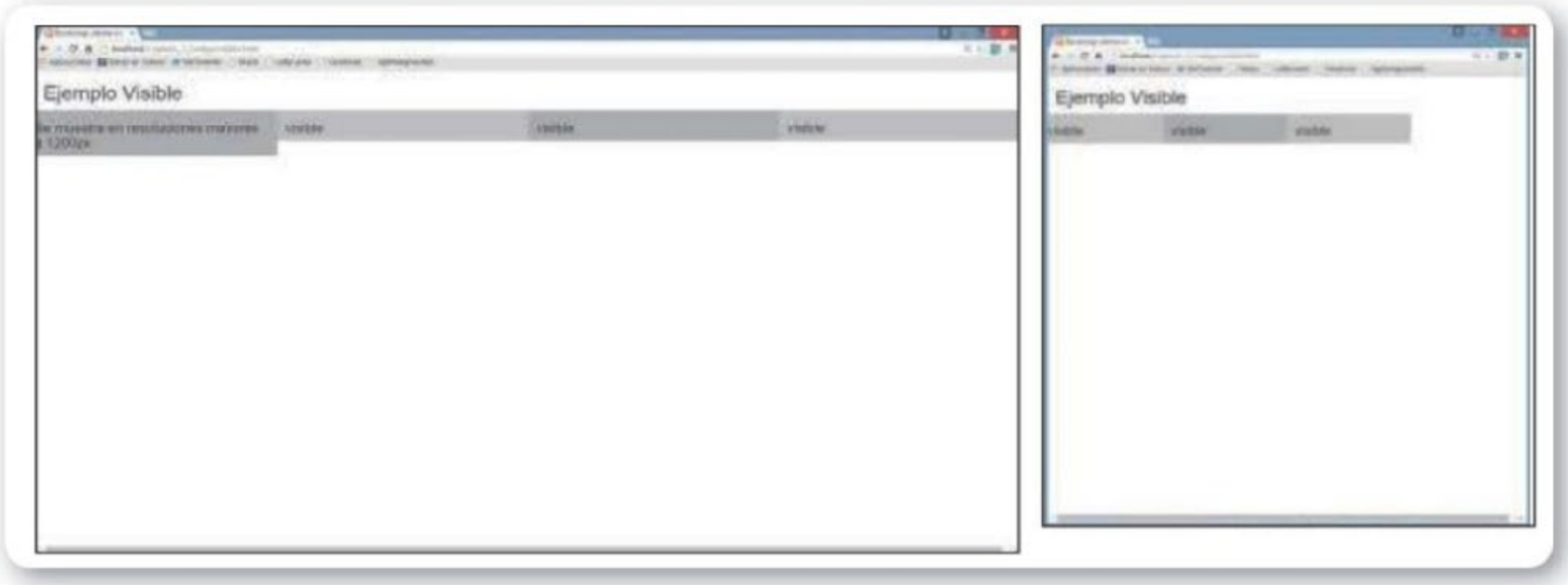


Figura 8. Solo cuando la resolución de pantalla supera los 1200 px se muestra la primera columna.

La clase clearfix

Suele suceder que no siempre tenemos el mismo contenido por columna. A veces, el contenido puede ser más largo en una columna que en otras, y al adaptar el diseño a un dispositivo móvil, por ejemplo, puede suceder que no se visualice de forma correcta. Para solucionar este inconveniente, contamos con la clase **clearfix**.

OFFSET

Por defecto, las columnas se agregan una a continuación de la otra. En ciertas ocasiones, necesitaremos agregar un espacio entre columnas, para lo que el framework nos ofrece la clase **offset**. Esta clase permite agregar un espacio entre columnas, equivalente al ancho de una o varias columnas. Por ejemplo, para añadir una columna en blanco por la izquierda usamos **col-xs-3 col xs-offset-1**.

En el siguiente ejemplo, tenemos cuatro columnas, de las cuales la primera presenta un contenido más extenso que las tres restantes.

```
<div class="content">
                    <div class="row">
            <div class="col-xs-6 col-sm-3" style="background-
color:#aaa"><h3>Columna A - Esta columna posee un contenido más extenso que la
columna que está a la derecha, por lo tanto tendrá, un alto diferente.</h3></div>
            <div class="col-xs-6 col-sm-3" style="background-
color:#bbb"><h3>Columna B</h3></div>

            <div class="clearfix visible-xs-block"></div>
            <div class="col-xs-6 col-sm-3" style="background-color:#aaa">
<h3>Columna C</h3></div>
            <div class="col-xs-6 col-sm-3" style="background-color:#bbb">
<h3>Columna D</h3></div>
          </div>
          </div>
```

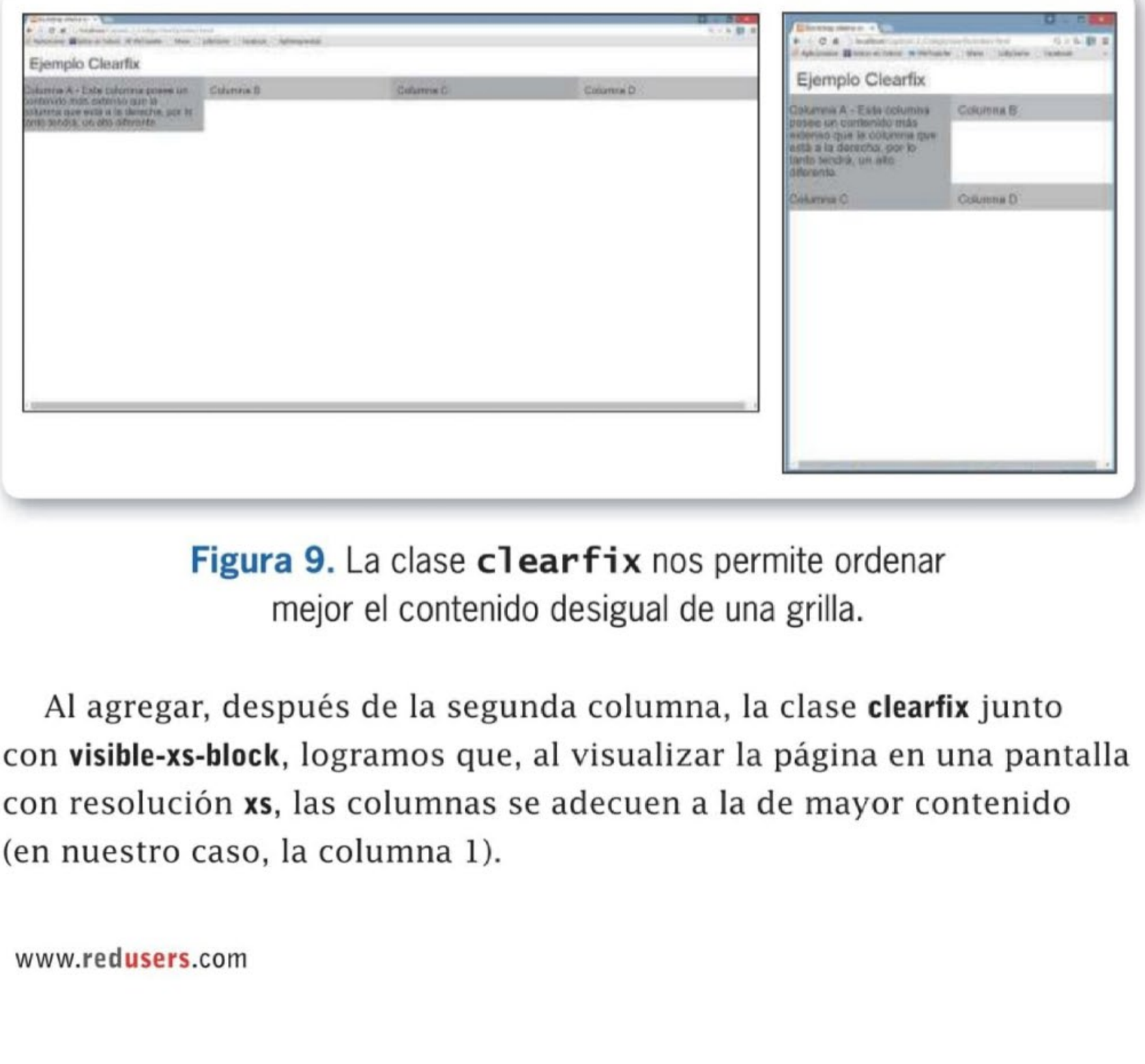


Figura 9. La clase **`clearfix`** nos permite ordenar mejor el contenido desigual de una grilla.

Al agregar, después de la segunda columna, la clase **clearfix** junto con **visible-xs-block**, logramos que, al visualizar la página en una pantalla con resolución **xs**, las columnas se adecuen a la de mayor contenido (en nuestro caso, la columna 1).

Utilizar media queries

Como dijimos anteriormente en este capítulo, Bootstrap está diseñado para cubrir todas las necesidades de los programadores y diseñadores, y, para esto, cuenta con 12 columnas adaptables a nuestra necesidad. Esas columnas adaptables están programadas con ***media queries***, respetando el ancho de cada dispositivo como vimos anteriormente.

Las *media queries* que utiliza el framework para adaptar nuestro diseño, ya sea una tablet, un teléfono inteligente o una PC de escritorio, son las siguientes:

- Para teléfonos inteligentes de 480 px:

```
@media (max-width: 480px)
{
 }
```

- Para tablets de 768 px:

```
@media (max-width: 768px)
{
 }
```

- Para tablets y PC de escritorio entre 768 px y 980 px:

```
@media (min-width: 768px) and (max-width: 980px)
{
 }
```

MEDIA QUERIES

Son un módulo de las hojas de estilo (CSS) que nos permiten adaptar nuestro contenido o comportamiento de acuerdo a la resolución de pantalla del dispositivo y de esta forma hacer una mejor experiencia de usuario. Bootstrap utiliza ***media queries*** para su sistema de 12 columnas, que puede ser modificado de acuerdo a nuestra necesidad. Sin embargo, el framework cuenta con muchísimas posibilidades para que no tengamos que modificar ni adaptar *media queries*.

- Para PC de escritorio de 1200 px o más:

```
@media (max-width: 1200px)
{
 }
```

Todas estas medidas en pixeles pueden ser cambiadas; sin embargo, no lo recomendamos, ya que el framework cuenta con suficientes recursos para que no haya necesidad de modificar las *media queries*.

Ejemplo práctico

Veremos a continuación un ejemplo para saber cómo implementar el sistema de columnas en Bootstrap a partir de un diseño.

Supongamos que tenemos el siguiente diseño que nos fue entregado por un diseñador gráfico:

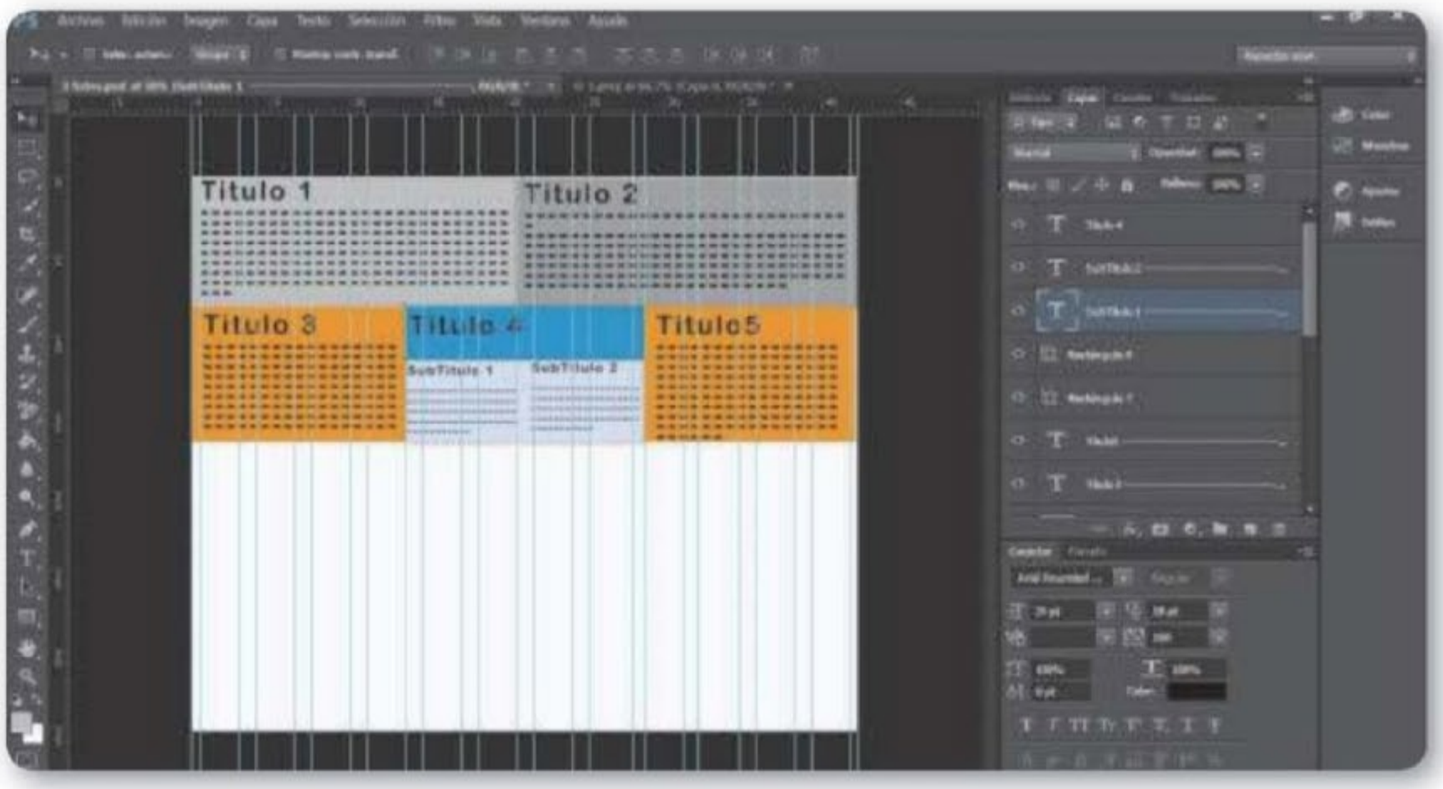


Figura 10. Diseño realizado en Adobe Photoshop para maquetar con Bootstrap.

Como podemos observar en el diseño, hay definidas claramente dos secciones: una parte superior, que podemos llamar **A**, y una parte inferior, que denominaremos **B**.

Estas dos secciones, a su vez, tienen varias columnas. La sección A está dividida en dos columnas; B, en 3 tres; y, a su vez, la columna del medio (Título 4) tiene dos columnas para los subtítulos.

Para realizar la maquetación, comenzamos con la sección A, que tiene solo una fila y dos columnas. Si las filas son 12 y necesitamos nada más que 2, el resultado de dividir 12 sobre 2 nos da **6**, que es la longitud que debe tener la clase **class="col-md-6"**. Veamos cómo queda:

```
<div class="row">
        <div class="col-md-6" style="background-color:#aaa" >
                <h3>Título 1</h3>
                <p>Lorem Ipsum es simplemente el texto de relleno de las im-
prentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno estándar de las
industrias desde el año 1500, cuando un impresor (N. del T. persona que se dedica a
la imprenta) desconocido usó una galería de textos y los mezcló de tal manera que
logró hacer un libro de textos especimen.</p>
        </div>
        <div class="col-md-6"style="background-color:#bbb" >
                <h3>Título 2</h3>
                <p>Lorem Ipsum es simplemente el texto de relleno de las im-
prentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno estándar de las
industrias desde el año 1500, cuando un impresor (N. del T. persona que se dedica a
la imprenta) desconocido usó una galería de textos y los mezcló de tal manera que
logró hacer un libro de textos especimen.</p>
        </div>
</div>
```

En este ejemplo estamos utilizando el prefijo de la clase **md** para resoluciones mayores o iguales a 992 px (computadoras de escritorio). También le estamos agregando un **estilo** para que se pueda apreciar mejor el ejemplo acorde al diseño. Más adelante, a partir del **Capítulo 7**, veremos cómo agregarle nuestro propio estilo en una hoja de estilo separada.

ADOBE PHOTOSHOP

Es un potente editor de imágenes, desarrollado por **Adobe Systems**, que entre sus funcionalidades nos provee de herramientas que nos permitirán maquetar nuestros sitios web. Está disponible tanto para la plataforma Windows como para Mac. Podemos descargar una versión de prueba por 30 días desde el sitio oficial: **www.adobe.com/la/products/photoshop.html**

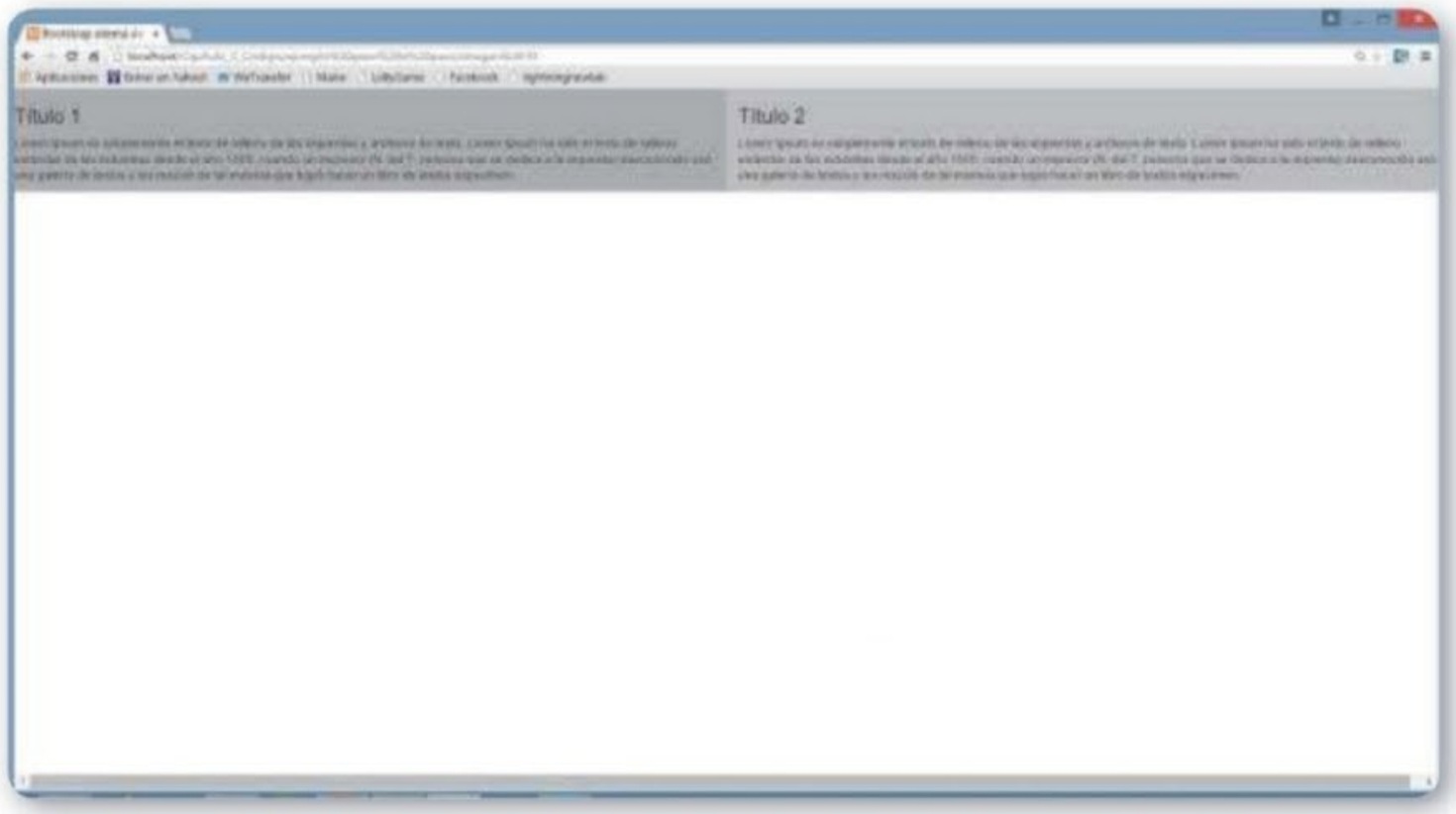


Figura 11. Primera parte del ejemplo (**Sección A**): diseño en Bootstrap.

Al ser la resolución mayor o igual a 992 px (**col-md-**), si achicamos el navegador y lo llevamos a una resolución menor a 992 px, veremos como el sistema de columnas en este ejemplo 2 colapsa; es decir, se ve de forma diferente en resoluciones menores (en este caso, una columna debajo de la otra).

Figura 12. En resoluciones **menores a 992 px**, cuando nuestro sistema de columnas es del tipo **`col-md-`**, colapsa para que pueda ser visualizado de otra forma en otras resoluciones.

Para la **sección B** de nuestro ejemplo, como ya dijimos, necesitamos tres columnas y una fila, pero una de las columnas, la del medio, en su interior contiene otras dos columnas más a modo de subtítulo. Entonces necesitamos que nuestras filas sean de cuatro, porque 12 sobre 3 nos da como resultado 4. Observemos el siguiente código:

```
<div class="row">
        <div class="col-md-4" style="background-color:#FFBF00" >
        </div>
        <div class="col-md-4"style="background-color:#00BFFF" >

        </div>
        <div class="col-md-4"style="background-color:#FFBF00" >
        </div>
</div>
```

Para terminar nuestro diseño, ahora solo nos resta agregarle las columnas necesarias a la columna del medio. La columna del medio en su interior tiene otras dos columnas que hacen de subtítulo, tal cual nos fue pedido desde el diseño. Entonces, para esto, vamos a necesitar dos columnas de 6. A esto se lo llama **anidar columnas**, cuando tenemos que incluir otra clase **row** dentro de una clase **row** ya definida.

```
<div class ="row">
        <div class="col-md-6" style="background-color:#E0F2F7">
                <h5>SubTítulo </h5>
                <p>Lorem Ipsum es simplemente el texto de relleno de las impren-
tas y archivos de texto. </p>
        </div>
        <div class="col-md-6" style="background-color:#E0F2F7">
                <h5>SubTítulo </h5>
                <p>Lorem Ipsum es simplemente el texto de relleno de las impren-
tas y archivos de texto. </p>
        </div>
    </div>
```

Y, por último, el código del ejemplo final:

```
<div class="content">
        <div class="row">
                <div class="col-md-6" style="background-color:#aaa" >
                        <h3>Título 1</h3>
                        <p>Lorem Ipsum es simplemente el texto de
```

```
relleno de las imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno
estándar de las industrias desde el año 1500, cuando un impresor (N. del T. persona
que se dedica a la imprenta) desconocido usó una galería de textos y los mezcló de
tal manera que logró hacer un libro de textos especimen.</p>
                    </div>
                    <div class="col-md-6"style="background-color:#bbb" >
                        <h3>Título 2</h3>
                        <p>Lorem Ipsum es simplemente el texto de
relleno de las imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno
estándar de las industrias desde el año 1500, cuando un impresor (N. del T. persona
que se dedica a la imprenta) desconocido usó una galería de textos y los mezcló de
tal manera que logró hacer un libro de textos especimen.</p>
                    </div>
                </div>

                <div class="row">
                    <div class="col-md-4" style="background-
color:#FFBF00" >
                        <h3>Título 3</h3>
                        <p>Lorem Ipsum es simplemente el texto de
relleno de las imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno
estándar de las industrias desde el año 1500, cuando un impresor (N. del T. persona
que se dedica a la imprenta) desconocido usó una galería de textos y los mezcló de
tal manera que logró hacer un libro de textos especimen.</p>
                    </div>
                    <div class="col-md-4"style="background-
color:#00BFFF" >
                        <h3>Título 4</h3>
                        <div class ="row">
                            <div class="col-md-6"
style="background-color:#E0F2F7">
                                <h5>SubTítulo </h5>
                                <p>Lorem Ipsum es simple-
mente el texto de relleno de las imprentas y archivos de texto. </p>
                            </div>
                            <div class="col-md-6"
style="background-color:#E0F2F7">
```

```
                                                    <h5>SubTítulo </h5>
                                                    <p>Lorem Ipsum es simple-
mente el texto de relleno de las imprentas y archivos de texto. </p>
                                                </div>
                                            </div>
                                        </div>
                                        <div class="col-md-4"style="background-
color:#FFBF00" >
                                                <h3>Título 5</h3>
                                                <p>Lorem Ipsum es simplemente el texto de
relleno de las imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno
estándar de las industrias desde el año 1500, cuando un impresor (N. del T. persona
que se dedica a la imprenta) desconocido usó una galería de textos y los mezcló de
tal manera que logró hacer un libro de textos especimen.</p>
                                        </div>

                                    </div>
    </div>
```

Figura 13. Ejemplo final terminado en el que podemos observar la maquetación realizada con Bootstrap a partir de un diseño en Photoshop.

RESUMEN

En este capítulo explicamos cómo utilizar el sistema de columnas que maneja Bootstrap para lograr un sitio *responsive*, es decir, que se adapte a todas las resoluciones de pantalla. Aprendimos a emplear las clases que nos ofrece el framework para crear filas y columnas. Posteriormente vimos cómo podemos anidar varias clases de Bootstrap para adaptar el diseño del sitio a diferentes dispositivos. Conocimos cómo gestionar la visibilidad de determinados elementos según el dispositivo desde el que accedamos a la página.

Actividades

TEST DE AUTOEVALUACIÓN

1. ¿Qué es la maquetación web?
2. ¿Para qué se utiliza la maquetación web?
3. ¿Cuántas columnas asigna Bootstrap al ancho de nuestra página web?
4. ¿Con qué clase se crea una fila?
5. ¿Qué clase se utiliza para mostrar contenido?

EJERCICIOS PRÁCTICOS

1. Diagrame una estructura para una página web que tenga tres columnas.
2. Al ejercicio 1 agréguele una fila más con cuatro columnas.
3. Al ejercicio anterior agréguele dos filas más con tres columnas.
4. Pruebe el código del ejemplo de la clase **clearfix**.
5. Pruebe el código del ejemplo práctico presentado en este capítulo.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

*04 =

Componentes gráficos

En este capítulo, conocemos los principales componentes gráficos que nos ofrece el framework y sus diversas formas de utilización: tipografía, tipos de encabezados, formato de párrafos, tablas, botones, etiquetas, badges, imágenes e iconos.

Tipografía

Bootstrap cuenta con una gran variedad de tipografía para los componentes que forman el framework, pero también tenemos la opción de agregarla nosotros mismos, como veremos a continuación.

Agregar tipografía propia

En caso de que por algún motivo quisiéramos cambiar la tipografía y utilizar alguna que no incluya el framework, lo que debemos hacer es crear nuestro propio estilo y enlazarlo en el **<head></head>** de la página HTML, siempre por debajo del llamado al estilo propio de Bootstrap:

```
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.2.0/
css/bootstrap.min.css">
<link rel="stylesheet" href="css/miEstilo.css" type="text/css" />
```

Además, en la hoja de estilo **miEstilo.css** tenemos que agregar la fuente y el vínculo a esta:

```
@font-face {
font-family: "miFuente";
src: url(/miFuente.ttf) format("truetype");
}
```

Al final, para utilizarla solo tenemos que hacer un llamado a nuestra fuente; si fuera un título **<h1></h1>**, quedaría de la siguiente forma:

```
h1 { font-family: "miFuente", sans-serif !important;}
```

DECLARACIÓN ¡IMPORTANT

La palabra reservada "**important**" se utiliza en hojas de estilo (CSS) para darle más importancia a algunas propiedades y, de esta forma, ignorar el estilo general de nuestra página web para considerar uno propio. Se debe colocar en el valor de la propiedad en la hoja de estilo, antes del punto y coma. Por ejemplo: **font-size: 20px !important;**.

Encabezados

Como bien sabemos, HTML dispone de **seis encabezados**, que se organizan por tamaño e importancia del 1 al 6. Así, por ejemplo, **<h1></h1>** es un título de tamaño grande y de mucha importancia, que nos servirá para un encabezado principal.

```
<h1> Título h1</h1>
<h2> Título h2</h2>
<h3> Título h3</h3>
<h4> Título h4</h4>
<h5> Título h5</h5>
<h6> Título h6</h6>
```

Si quisiéramos agregar un **subtexto** a ese mismo título, debemos utilizar la etiqueta **<small></small>**, y el código quedaría de este modo:

```
<h1>Título h1 <small>Este es un subtexto del Título h1</small></h1>
```

El subtexto tendrá el 80 % del tamaño de la etiqueta que lo contiene.

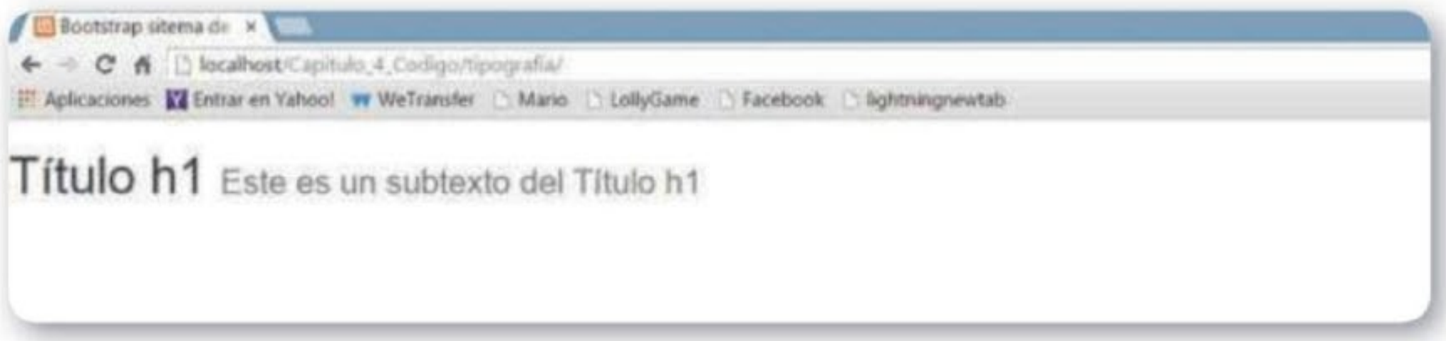


Figura 1. En Bootstrap se pueden utilizar **subtextos** dentro de la misma etiqueta HTML **<h1></h1>** con la etiqueta **<small></small>**.

¿DÓNDE ENCONTRAR EL ARCHIVO .LESS?

Si quisiéramos modificar algún archivo con extensión **.less**, como cualquier otro archivo del proyecto que no tenemos en la descarga del framework, debemos ir a la página principal de Bootstrap en GitHub: **https://github.com/twbs/bootstrap**. Allí podemos descargar el proyecto con todos los archivos sin compilar para modificar a nuestro gusto.

Cómo enriquecer el texto

El tamaño del texto en Bootstrap es de 14 px, tanto para párrafos ** ** como para todo el contenido en general. El interlineado (*line-height*) es de 1.428 con un margen (*margin-bottom*) de 10 px. Las variables que manejan el tamaño de la tipografía son dos y están incluidas en el directorio **variables.less**; una para el tamaño de la letra base (**@font-size-base**) y otra para el interlineado (**@font-height-base**).

Contamos con un conjunto de **etiquetas** que podemos emplear para enriquecer nuestros textos. A continuación veremos algunas de ellas.

Si queremos destacar algún párrafo, el framework cuenta con la clase **lead**, que puede ser aplicada en el párrafo, como podemos observar en el siguiente ejemplo:

```
<p>Párrafo normal sin aplicar ninguna clase</p>
<p class="lead">Párrafo con la clase lead</p>
```

Para destacar algún texto utilizando **negrita**, debemos utilizar la etiqueta ****:

```
<p>Párrafo normal <strong>texto resaltado con la etiqueta strong </strong></p>
```

Con la etiqueta **** obtenemos texto en **cursiva**:

```
<p>Párrafo normal <em>texto en cursiva con la etiqueta em </em></p>
```

Para texto **subrayado** debemos utilizar la etiqueta **<u></u>**:

```
<p>Párrafo normal <u>texto subrayado utilizando la etiqueta u </u></p>
```

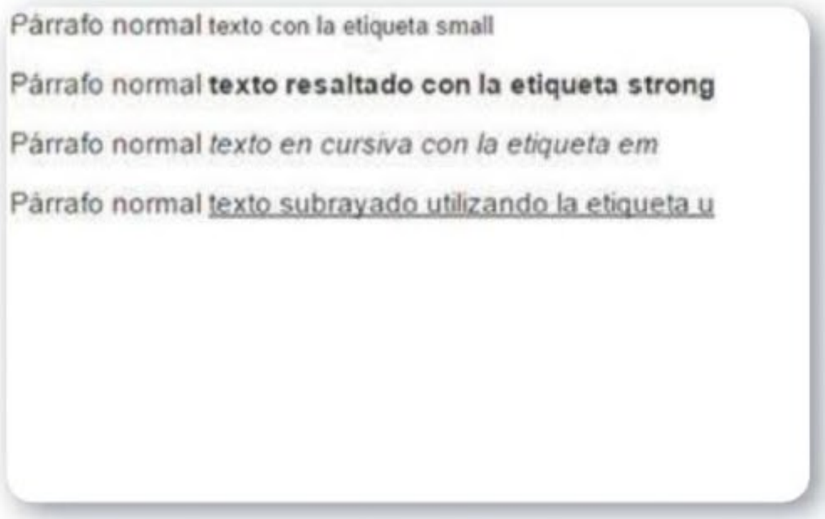


Figura 2. Ejemplos de estilos que pueden ser aplicados en nuestros textos.

Transformar texto utilizando clases

Existen en Bootstrap varias clases que nos permiten transformar texto como si estuviéramos editando una hoja de estilo CSS, con la diferencia que podemos hacerlo directamente desde el documento HTML.

Para **alinear** texto podemos utilizar varias clases, de acuerdo a la alineación que le demos al texto, por ejemplo:

```
<p class="text-left">Texto alineado a la izquierda.</p>
<p class="text-center"> Texto alineado en el centro.</p>
<p class="text-right"> Texto alineado a la derecha.</p>
<p class="text-justify"> Texto justificado.</p>
<p class="text-nowrap">Texto no ajustado.</p>
```

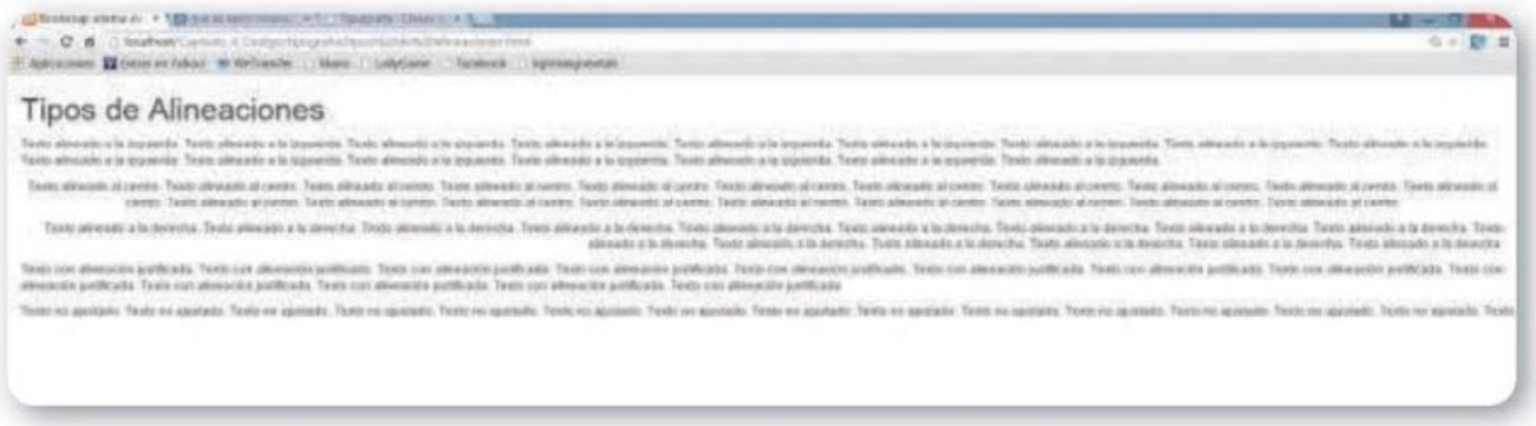


Figura 3. Párrafo con diferentes tipos de alineación, utilizando las variedades de clases que nos ofrece el framework.

Si queremos pasar el texto a **minúscula** o a **mayúscula**, o que todas las palabras **comiencen con mayúscula** debemos utilizar las clases **text-lowercase**, **text-appercase** y **text-capitalize** respectivamente:

```
<p class="text-lowercase">TEXTO EN MINÚSCULA.</p>
<p class="text-uppercase">texto en mayúscula.</p>
<p class="text-capitalize">texto que comienza siempre con una mayúscula.</p>
```

Figura 4. Párrafo que modifica la entrada de texto utilizando clases para mayúsculas, minúsculas o para que comiencen con mayúsculas.

Para darle formato a texto que va a contener una **dirección de contacto**, podemos utilizar la etiqueta **<address></address>** conjuntamente con la etiqueta HTML **
** para hacer un **salto de línea**:

```
<address>
  <strong>Editorial Users</strong><br>
  Arroyo 1357<br>
  CABA, (1416)<br>
  +54 (011) 4110-8700 <br>
  <a href="mailto:#">usershop.redusers.com</a>
</address>
```

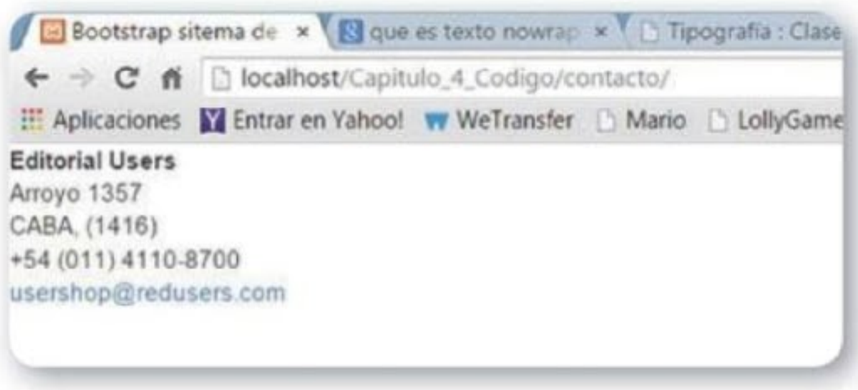


Figura 5. Con la etiqueta **<address></address>** podemos ingresar texto de contacto con formato alineado.

Otra opción interesante son las **abreviaturas** que se utilizan empleando la etiqueta **<abbr></abbr>**. Al pasar el cursor por una abreviatura se abre un **cartel de ayuda** (*tool-tip*) con un signo de interrogación que nos muestra la palabra completa sin abreviar. Estas etiquetas pueden ir acompañadas del atributo **title**, que nos sirve para pasarle como valor el significado de la etiqueta.

```
<abbr title="Señor">Sr.</abbr><br>
<abbr title="Señores">Sres.</abbr><br>
<abbr title="Doctor">Dr.</abbr><br>
<abbr title="Licenciado">Lic.</abbr><br>
```

Figura 6. La abreviatura se visualiza con una línea punteada y al pasar el mouse sobre la palabra nos muestra su significado.

Para darle formato a texto que va a contener una **dirección de contacto**, podemos utilizar la etiqueta **<address></address>** conjuntamente con la etiqueta HTML **
** para hacer un **salto de línea**:

```
<address>
  <strong>Editorial Users</strong><br>
  Arroyo 1357<br>
  CABA, (1416)<br>
  +54 (011) 4110-8700 <br>
  <a href="mailto:#">usershop.redusers.com</a>
</address>
```

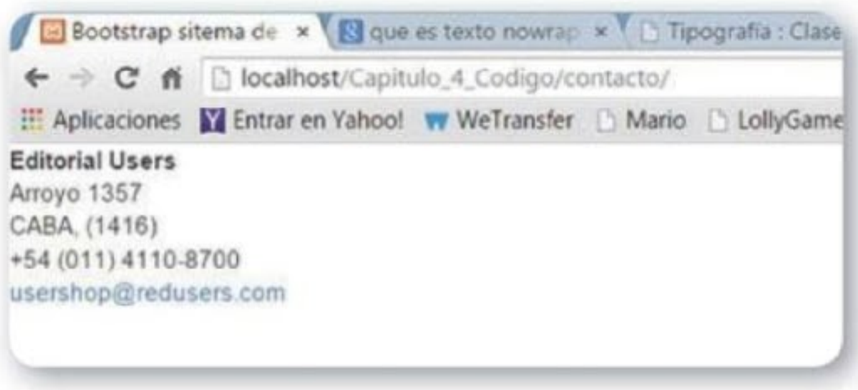


Figura 5. Con la etiqueta **<address></address>** podemos ingresar texto de contacto con formato alineado.

Otra opción interesante son las **abreviaturas** que se utilizan empleando la etiqueta **<abbr></abbr>**. Al pasar el cursor por una abreviatura se abre un **cartel de ayuda** (*tool-tip*) con un signo de interrogación que nos muestra la palabra completa sin abreviar. Estas etiquetas pueden ir acompañadas del atributo **title**, que nos sirve para pasarle como valor el significado de la etiqueta.

```
<abbr title="Señor">Sr.</abbr><br>
<abbr title="Señores">Sres.</abbr><br>
<abbr title="Doctor">Dr.</abbr><br>
<abbr title="Licenciado">Lic.</abbr><br>
```

Figura 6. La abreviatura se visualiza con una línea punteada y al pasar el mouse sobre la palabra nos muestra su significado.

Blockquotes

Si queremos diferenciar un párrafo para mostrar una **cita**, es decir, un texto perteneciente a otro autor, podemos usar la etiqueta **<blockquote></blockquote>**.

Esta etiqueta se utiliza para **agrupar contenido** y dentro de ella se incluye la etiqueta ** ** para agrupar el texto.

```
<p>Párrafo normal</p>
<blockquote>
    <p>Párrafo con blockquote</p>
</blockquote>
```

Para hacerlo aún más interesante, podemos agregarle una etiqueta **<footer></footer>** que el framework reconoce como un **pie de cita** del texto:

```
<blockquote>
  <p>Esto es un párrafo dentro de un blockquote.</p>
  <footer>Esto es el pie de la cita</footer>
</blockquote>
```

También podemos hacer que el blockquote se vea en el margen derecho agregándole la clase **blockquote-reverse**, como podemos ver a continuación:

```
<blockquote class= "blockquote-reverse">
  <p>Esto es un párrafo dentro de un blockquote.</p>
  <footer>Esto es el pie de la cita</footer>
</blockquote>
```

Figura 7. La etiqueta **<blockquote></blockquote>** crea sangría a la izquierda y derecha del párrafo para indicar que se trata de una cita.

Cambiar el color del texto según el tipo de contenido

Bootstrap utiliza **seis colores** a través de clases CSS para mostrar texto de acuerdo al contenido:

```
<p class="text-muted">Esto es un texto apagado.</p>
<p class="text-primary"> Esto es un texto primario.</p>
<p class="text-success"> Esto es un texto de éxito.</p>
<p class="text-info"> Esto es un texto para información.</p>
<p class="text-warning"> Esto es un texto de advertencia.</p>
<p class="text-danger"> Esto es un texto peligroso.</p>
```

Labels

Un *label* (en español, **etiqueta**) nos sirve para escribir texto que necesitamos mostrarle al usuario. Para agregarlo, debemos utilizar la clase **label** dentro de la etiqueta HTML **** y, como opción, podemos agregarle las distintas clases para mostrar el label en diferentes colores, tal como se puede ver en el siguiente ejemplo:

```
<span class="label label-default">Común</span>
<span class="label label-primary">Primario</span>
<span class="label label-success">Éxito</span>
<span class="label label-info">Información</span>
<span class="label label-warning">Cuidado</span>
<span class="label label-danger">Peligro</span>
```

Badges

Los *badges* (en español, *insignias*) son pequeños círculos que nos permiten saber la cantidad de elementos que hay dentro de un determinado componente. Así, por ejemplo, en una bandeja de entrada de correo electrónico, con un círculo y un número adentro del mismo

Cambiar el color del texto según el tipo de contenido

Bootstrap utiliza **seis colores** a través de clases CSS para mostrar texto de acuerdo al contenido:

```
<p class="text-muted">Esto es un texto apagado.</p>
<p class="text-primary"> Esto es un texto primario.</p>
<p class="text-success"> Esto es un texto de éxito.</p>
<p class="text-info"> Esto es un texto para información.</p>
<p class="text-warning"> Esto es un texto de advertencia.</p>
<p class="text-danger"> Esto es un texto peligroso.</p>
```

Labels

Un *label* (en español, **etiqueta**) nos sirve para escribir texto que necesitamos mostrarle al usuario. Para agregarlo, debemos utilizar la clase **label** dentro de la etiqueta HTML **** y, como opción, podemos agregarle las distintas clases para mostrar el label en diferentes colores, tal como se puede ver en el siguiente ejemplo:

```
<span class="label label-default">Común</span>
<span class="label label-primary">Primario</span>
<span class="label label-success">Éxito</span>
<span class="label label-info">Información</span>
<span class="label label-warning">Cuidado</span>
<span class="label label-danger">Peligro</span>
```

Badges

Los *badges* (en español, *insignias*) son pequeños círculos que nos permiten saber la cantidad de elementos que hay dentro de un determinado componente. Así, por ejemplo, en una bandeja de entrada de correo electrónico, con un círculo y un número adentro del mismo

podemos hacerle saber al usuario cuántos mensajes tiene sin leer. Para aplicarlo en el framework, debemos utilizar la clase **badge**.

```
<a href="#">Correo sin leer <span class="badge">42</span></a>
```

Figura 8. Podemos utilizar la clase **badge** para mostrarle al usuario la cantidad de elementos disponibles en un determinado componente.

Tablas

Las tablas, también llamadas **grillas** o **datagrid**, sirven para mostrar datos de manera ordenada y prolija. En Bootstrap contamos con la clase **table** que tiene que agregarse a la etiqueta de HTML:

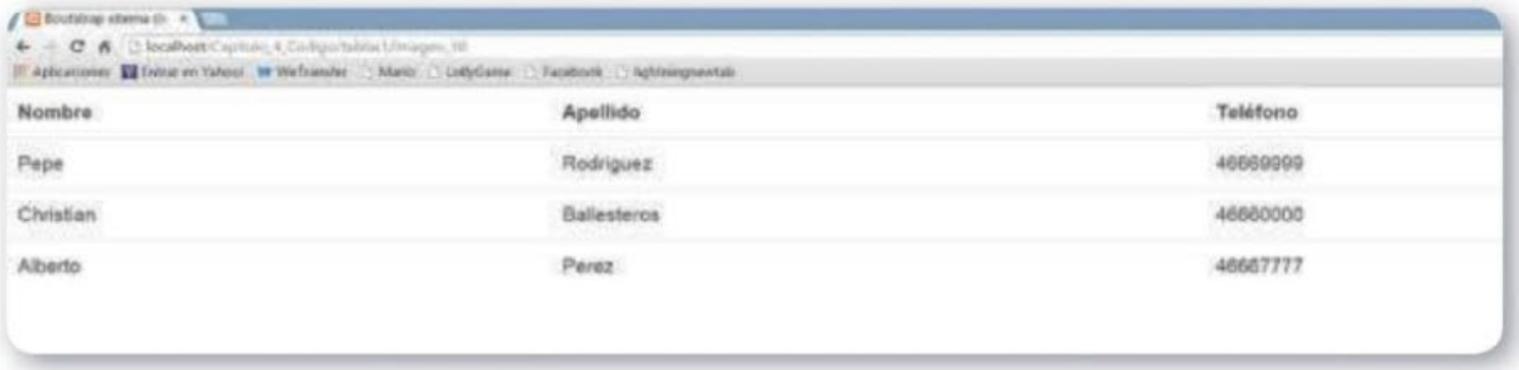
<table></table>.

```
<table class="table" >
                    <th>Nombre</th>
                    <th>Apellido</th>
                    <th>Teléfono</th>
                    <tr>
```

FONDO COLOREADO

Si queremos destacar texto, también podemos aplicarle un fondo coloreado. Para ello podemos utilizar las siguientes clases: **"bg-primary"** (azul), **"bg-success"** (verde), **"bg-info"** (celeste), **"bg-warning"** (amarillo) y **"bg-danger"** (rojo). Por ejemplo, si queremos que un texto tenga fondo azul, escribimos la siguiente sintaxis: ** Este texto tendrá un fondo azul **.

```
                <td>Christian</td>
                <td>Ballesteros</td>
                <td>46660000</td>
            </tr>
            <tr>
                <td>José</td>
                <td>Vazquez</td>
                <td>46660000</td>
            </tr>
            <tr>
                <td>Alberto</td>
                <td>Perez</td>
                <td>46667777</td>
            </tr>
</table>
```



Nombre	Apellido	Teléfono
Pepe	Rodriguez	46669999
Christian	Ballesteros	46660000
Alberto	Perez	46667777

Figura 9. Las tablas en Bootstrap tienen un aspecto minimalista que nos permite visualizar los datos de manera clara.

Opciones de estilo

Podemos darle un tono gris y blanco a las filas para resaltar más los datos. Para esto, tenemos que agregarle la clase **table-striped** a **table**.

ETIQUETA HTML SPAN

Las etiquetas HTML **** se utilizan para agrupar contenido o darle un estilo especifico al texto. En sí, esta etiqueta no agrega nada al documento HTML, pero es muy útil si a la misma le agregamos estilo CSS, para darle una visualización distinta a cualquier parte del texto. Es muy parecida a la etiqueta ** ** con la diferencia de que es utilizada para darle un formato especial al texto.

```
<table class="table table-striped" >
                    <th>Nombre</th>
                    <th>Apellido</th>
                    <th>Teléfono</th>
                    <tr>
                        <td>Christian</td>
                        <td>Ballesteros</td>
                        <td>46660000</td>
                    </tr>
                    <tr>
                        <td>José</td>
                        <td>Vazquez</td>
                        <td>46660000</td>
                    </tr>
                    <tr>
                        <td>Alberto</td>
                        <td>Perez</td>
                        <td>46667777</td>
                    </tr>
</table>
```

Figura 10. Para agregarle a la tabla un tono gris y blanco debemos utilizar la clase **`table-striped`**.

Si además queremos agregarle un **borde** fino a toda nuestra tabla solo tenemos que usar la clase **table-bordered**.

```
<table class="table table-striped table-bordered" >
                    <th>Nombre</th>
                    <th>Apellido</th>
                    <th>Teléfono</th>
                    <tr>
```

```
                    <td>Christian</td>
                    <td>Ballesteros</td>
                    <td>46660000</td>
                </tr>
                <tr>
                    <td>José</td>
                    <td>Vazquez</td>
                    <td>46660000</td>
                </tr>
                <tr>
                    <td>Alberto</td>
                    <td>Perez</td>
                    <td>46667777</td>
                </tr>
    </table>
```

Otra opción interesante que podemos agregar a nuestra tabla son los **efectos**: por ejemplo, que al pasar el cursor sobre una fila, se ponga de color gris claro. Eso se logra agregando la clase **table-hover**. ***Hover*** (en español, "estar suspendido" o "flotar") es un efecto, muy utilizado en programación web y de escritorio, que consiste en producir una alteración en un elemento cuando situamos el cursor sobre él (en este caso, al pasar sobre una fila o registro de la tabla). Esto puede ser útil para destacar una parte de texto, por ejemplo, cambiando su color.

```
<table class="table table-hover" >
                    <th>Nombre</th>
                    <th>Apellido</th>
                    <th>Teléfono</th>
                    <tr>
```


ESTRUCTURA DE UNA TABLA

Para definir la estructura de una tabla, usamos las siguientes etiquetas: **<tr></tr>** para indicar el comienzo y fin de una fila; **<td></td>** para indicar el comienzo y fin de una celda; **<th></th>** para indicar las celdas que serán consideradas como encabezado de la tabla.

```
                <td>Christian</td>
                <td>Ballesteros</td>
                <td>46660000</td>
            </tr>
            <tr>
                <td>José</td>
                <td>Vazquez</td>
                <td>46660000</td>
            </tr>
            <tr>
                <td>Alberto</td>
                <td>Perez</td>
                <td>46667777</td>
            </tr>
</table>
```

Figura 11. Al pasar el cursor del mouse sobre una fila, cambia de color a gris, para así poder visualizar mejor los datos.

Para reducir el ancho de las tablas utilizamos la clase **table-condensed**, que hace que se reduzca el ***padding*** (espacio interior) a la mitad.

```
<table class="table table-condensed" >
                <th>Nombre</th>
                <th>Apellido</th>
                <th>Teléfono</th>
                <tr>
                    <td>Christian</td>
                    <td>Ballesteros</td>
                    <td>46660000</td>
                </tr>
```

```
                <tr>
                        <td>José</td>
                        <td>Vazquez</td>
                        <td>46660000</td>
                </tr>
                <tr>
                        <td>Alberto</td>
                        <td>Perez</td>
                        <td>46667777</td>
                </tr>
    </table>
```

Anteriormente vimos que podíamos cambiar el color del texto de acuerdo al contenido. Esto mismo lo podemos hacer con las tablas: es válido tanto para **filas** (**<tr></tr>**) como para **columnas** (**<td></td>**). De acuerdo a su contenido, existen cinco clases que podemos aplicar, como podemos ver a continuación:

```
<table class="table table-bordered" >
        <tr >
                <th >Nombre</th>
                <th>Apellido</th>
                <th>Teléfono</th>
        </tr>
        <tr class="active">
                <td>Christian</td>
                <td>Ballesteros</td>
                <td>46660000</td>
        </tr>
        <tr class="success">
                <td>José</td>
                <td>Vazquez</td>
                <td>46669999</td>
        </tr>
        <tr class="warning">
                <td>Alberto</td>
                <td>Perez</td>
                <td>46669999</td>
```

```
        </tr>
        <tr class="danger">
                <td>Roberto</td>
                <td>Alvarez</td>
                <td>46669999</td>
        </tr>
        <tr class="info">
                <td>Héctor</td>
                <td>Gonzalez</td>
                <td>46669999</td>
        </tr>
</table>
```

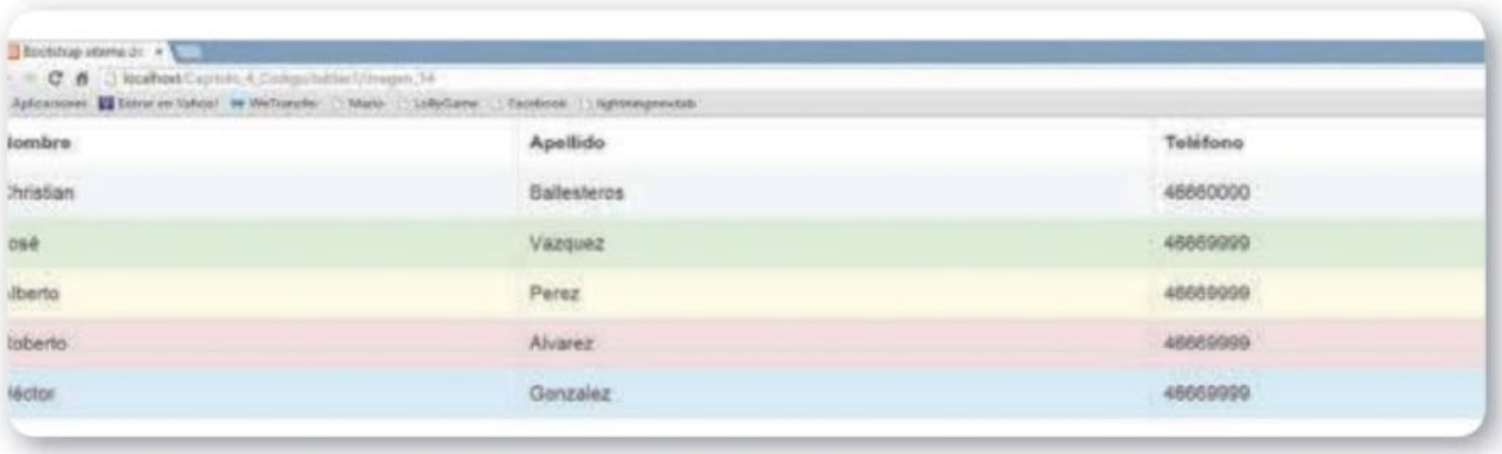


Figura 12. Podemos cambiar el color de filas o columnas de acuerdo a su contenido.

Por último, el framework nos ofrece la clase **table-responsive** para adaptar las tablas a pantallas más pequeñas (menores a 768 px). Lo que hace esta clase es agregarle un **barra de desplazamiento** (*scroll*) horizontal para que pueda visualizarse mejor el contenido.

```
<div class="table-responsive">
<table class="table table- bordered" >
                        <th>Nombre</th>
                        <th>Apellido</th>
                        <th>Teléfono</th>
                <tr>
                        <td>Pepe</td>
                        <td>Rodriguez</td>
                        <td>46669999</td>
                </tr>
```

```
                <tr>
                    <td>José</td>
                    <td>Vazquez</td>
                    <td>46660000</td>
                </tr>
                <tr>
                    <td>Alberto</td>
                    <td>Perez</td>
                    <td>46667777</td>
                </tr>
    </table>
    </div>
```

Botones

Para utilizar botones tenemos que usar la clase **btn btn-default** que puede utilizarse en las etiquetas HTML **<a>**, **<button>** o **<input>**, como podemos observar en el siguiente ejemplo:

```
<a class="btn btn-default" href="#" role="button">Botón Link</a>
<button class="btn btn-default" type="submit">Botón común</button>
<input class="btn btn-default" type="button" value="Botón Input">
<input class="btn btn-default" type="submit" value="Botón Enviar">
```

Figura 13. Bootstrap acepta los mismos botones que en HTML agregando la clase **`btn btn-default`**.

Como vimos anteriormente, de acuerdo al contenido, Bootstrap ofrece una **variedad de colores** también para los botones. Solo tenemos que agregarle la clase correspondiente:

```
<button type="button" class="btn btn-default">Botón común o default</button>
<button type="button" class="btn btn-primary">Botón primary</button>
<button type="button" class="btn btn-success">Botón success</button>
<button type="button" class="btn btn-info">Botón info</button>
<button type="button" class="btn btn-warning">Botón warning</button>
<button type="button" class="btn btn-danger">Botón danger</button>
<button type="button" class="btn btn-link">Botón para link</button>
```

Figura 14. De acuerdo al contenido, podemos agregarles color a nuestros botones.

Tamaño

Contamos con diferentes tamaños para los botones que pueden servirnos de acuerdo a la resolución de pantalla para la que estemos programando.

Agregándole la clase **btn-lg**, **btn-sm** o **btn-xs**, obtenemos botones grandes, chicos y muy chicos respectivamente. Si no agregamos ninguna de estas clases, por defecto tendremos un botón de tamaño estándar, cuyas medidas están entre los tamaños grande y chico.

```
<button type="button" class="btn btn-default btn-lg">Botón grande</button>
<button type="button" class="btn btn-default">Botón común</button>
<button type="button" class="btn btn-default btn-sm">Botón chico</button>
<button type="button" class="btn btn-default btn-xs">Botón muy chico</button>
```

BOTONES DE NIVEL DE BLOQUE

Mediante la clase **btn-block** podemos crear un botón que tenga el ancho del contenedor dentro del cual el botón está contendido. Por ejemplo, para crear un botón de nivel de bloque, escribiremos la siguiente sentencia: **<button type="button" class="btn btn-primary btn-block">Button 1</button>**

Figura 15. En Bootstrap contamos con diferentes tamaños de botones para adaptarlos a las distintas resoluciones de pantalla.

Agrupar botones

Bootstrap nos permite agrupar botones en una misma línea utilizando la clase **btn-group**, que debemos incluirla en un ** ** antes de crear los botones.

```
<div class="btn-group">
  <button type="button" class="btn btn-default">Botón A</button>
  <button type="button" class="btn btn-default">Botón B</button>
  <button type="button" class="btn btn-default">Botón C</button>
</div>
```

Figura 16. Podemos agrupar varios botones para crear un menú.

Si combinamos varios grupos de botones, podemos fácilmente crear una **barra de herramientas**.

```
<div class="btn-group">
  <button type="button" class="btn btn-default">Botón A</button>
  <button type="button" class="btn btn-default">Botón B</button>
  <button type="button" class="btn btn-default">Botón C</button>
</div>
<div class="btn-group">
  <button type="button" class="btn btn-default">Botón D</button>
```

```
  <button type="button" class="btn btn-default">Botón E</button>
  <button type="button" class="btn btn-default">Botón F</button>
</div>
<div class="btn-group">
  <button type="button" class="btn btn-default">Botón G</button>
  <button type="button" class="btn btn-default">Botón H</button>
  <button type="button" class="btn btn-default">Botón I</button>
  <button type="button" class="btn btn-default">Botón J</button>
</div>
```

Figura 17. Podemos crear fácilmente una barra de herramientas combinando grupos de botones.

Si quisiéramos cambiar su tamaño, tenemos que incluir la clase **lg** para botones grandes, **sm** para pequeños y **xs** para botones muy pequeños.

```
<div class="btn-group btn-group-lg">
  <button type="button" class="btn btn-default">Botón A</button>
  <button type="button" class="btn btn-default">Botón B</button>
  <button type="button" class="btn btn-default">Botón C</button>
</div>
<div class="btn-group btn-group-sm">
  <button type="button" class="btn btn-default">Botón D</button>
  <button type="button" class="btn btn-default">Botón E</button>
  <button type="button" class="btn btn-default">Botón F</button>
</div>
<div class="btn-group btn-group-xs">
  <button type="button" class="btn btn-default">Botón G</button>
  <button type="button" class="btn btn-default">Botón H</button>
  <button type="button" class="btn btn-default">Botón I</button>
  <button type="button" class="btn btn-default">Botón J</button>
</div>
```

Otra opción es ubicarlos en forma **vertical**, con la apariencia de un menú. Para eso, añadimos la clase **btn-group-vertical**:

```
<div class="btn-group-vertical">
  <button type="button" class="btn btn-default">Botón A</button>
  <button type="button" class="btn btn-default">Botón B</button>
  <button type="button" class="btn btn-default">Botón C</button>
  <button type="button" class="btn btn-default">Botón D</button>
</div>
```

Figura 18. Podemos crear un menú vertical agrupando botones.

Botones desplegables

Los botones desplegables (en inglés, ***dropdowns***) nos permiten crear un menú desplegable desde el propio botón. Siempre que usemos un botón desplegable debemos tener en cuenta que tiene que contener opciones en **formato de lista HTML** con las etiquetas **
** y **
** en su interior.

Además, recordemos que debemos tener enlazada la librería de **jQuery** entre las etiquetas **<head></head>** del archivo HTML; de lo contrario, no funcionará el despliegue del botón. El enlace por **CDN** es el siguiente: **<script src="http://code.jquery.com/jquery-2.1.3.min.js"></script>**

```
<div class="btn-group">
  <button type="button" class="btn btn-default dropdown-toggle"
    data-toggle="dropdown">
    Default <span class="caret"></span>
  </button>
  <ul class="dropdown-menu" role="menu">
```

```
      <li><a href="#">Link 1</a></li>
      <li><a href="#">Link 2</a></li>
      <li><a href="#">Link 3</a></li>
      <li class="divider"></li>
      <li><a href="#">Link 4 separado</a></li>
    </ul>
</div>
```

Figura 19. La clase **divider** nos permite separar mediante una línea un grupo de ítems dentro del menú.

Analizando el código anterior, empleamos la clase **dropdown-toggle** para indicar el comportamiento de menú desplegable. La clase **caret** nos muestra la pequeña flecha que aparece en el botón. Mediante la etiqueta **<li>** agregamos los elementos del menú. Con la clase **divider** agregamos una línea que separa los elementos del menú.

Podemos también cambiar el tamaño de los botones agregando la clase **btn-lg** para botones grandes, **btn-sm** para chicos y **btn-xs** para botones pequeños.

En el siguiente código de ejemplo, podemos observar el funcionamiento de la clase **btn-xs** para que el botón se vea pequeño:

```
<div class="btn-group">
  <button type="button" class="btn btn-default btn-xs dropdown-toggle"
```

DESACTIVAR UN BOTÓN

Para desactivar un botón, podemos agregarle el atributo **disabled="disabled"**, o añadirlo como clase **class="btn btn-default disabled"**. De las dos maneras, el botón aparecerá como desactivado (sin la opción de ser presionado). Esta opción puede ser útil para evitar el envío reiterado de un formulario: una vez que el usuario presiona el botón para enviar un formulario, lo desactivamos y con eso evitamos que se vuelva a enviar.

```
    data-toggle="dropdown">
    Default <span class="caret"></span>
  </button>
  <ul class="dropdown-menu" role="menu">
    <li><a href="#">Link 1</a></li>
    <li><a href="#">Link 2</a></li>
    <li><a href="#">Link 3</a></li>
    <li class="divider"></li>
    <li><a href="#">Link 4 separado</a></li>
  </ul>
</div>
```

En lugar de desplegar el menú hacia abajo, podemos crear un botón que despliegue el menú hacia arriba. Para ello, usaremos la clase **dropup**:

```
<div class="btn-group dropup" >
  <button type="button" class="btn btn-default btn-xs dropdown-toggle"
    data-toggle="dropdown" >
    Default <span class="caret"></span>
  </button>
  <ul class="dropdown-menu" role="menu">
    <li><a href="#">Link 1</a></li>
    <li><a href="#">Link 2</a></li>
    <li><a href="#">Link 3</a></li>
    <li class="divider"></li>
    <li><a href="#">Link 4 separado</a></li>
  </ul>
</div>
```

SVG

Los **Scalable Vector Graphics** (en español, "gráficos vectoriales escalables") están basados en XML y son gráficos con un lenguaje de programación de marcado (tags). Entre sus características, nos permiten escalar una imagen, achicarla o ampliarla sin perder calidad (algo que con otro tipo de imágenes, como JPG, no es posible). Fue recomendado por la W3C en 2001.

Figura 20. La clase **dropup**, además, cambiará el sentido de la flecha del botón.

Iconos

Bootstrap cuenta con 250 iconos, que, según lo indica su sitio web, son pagos. Sin embargo, el creador los dejó sin costo para el framework, siempre y cuando se lo mencione cada vez que los utilizamos.

Los iconos que utiliza Bootstrap no son imágenes, es decir, están basados en un formato de fuente. Están ubicados en el directorio **fonts** del framework, ya compilados en sus hojas de estilo. Por eso, si deseamos cambiarlos, tenemos que actualizar las hojas de estilo.

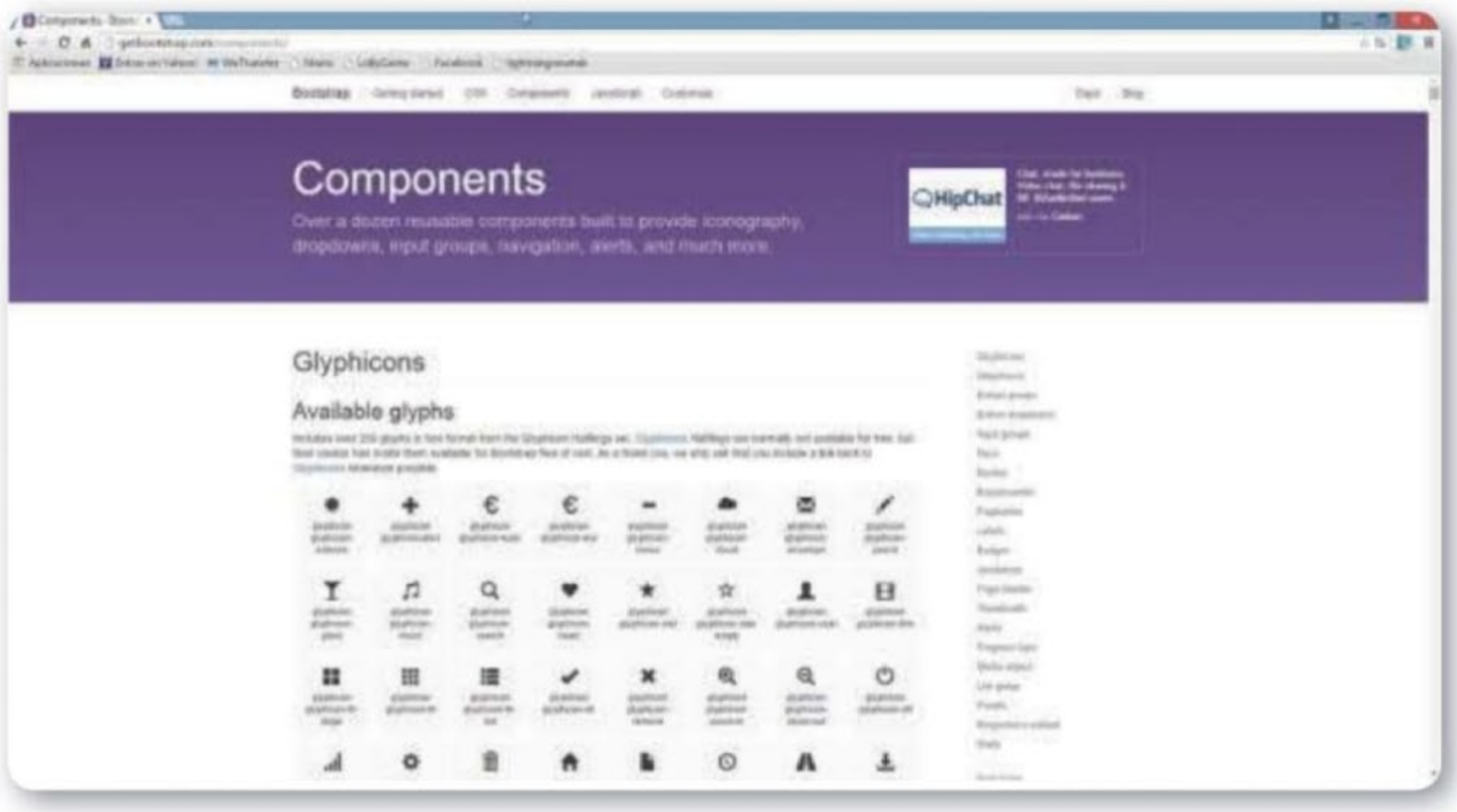


Figura 21. En el sitio oficial de Bootstrap (**http://getbootstrap.com/components**) encontramos la lista completa de iconos.

Para poder utilizar los iconos con sus respectivas clases, lo único que tenemos que hacer es agregar la clase correspondiente con el nombre del icono:

```
<button type="button" class="btn btn-default btn-lg">
  <span class="glyphicon glyphicon-phone-alt" ></span> Teléfono
</button>
```

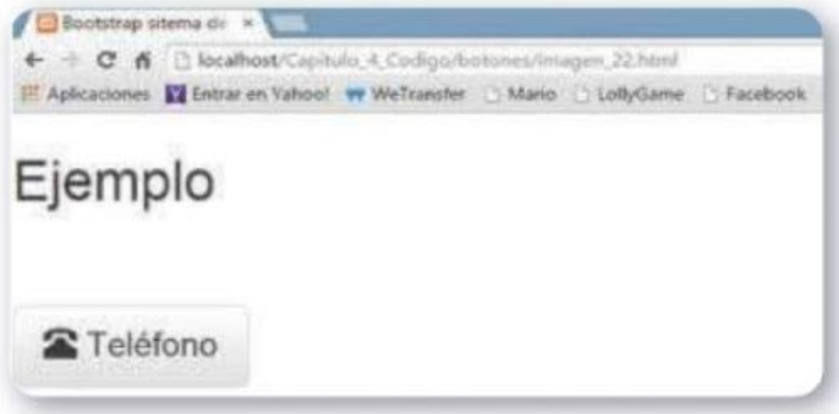


Figura 22. Los iconos se pueden integrar con cualquier elemento HTML.

Imágenes

Bootstrap permite hacer uso de las imágenes de acuerdo a nuestra necesidad, por medio de clases. Estas permiten que las imágenes se adapten a cada resolución disponible o se vean de una determinada forma.

Para hacer que nuestra imagen se pueda adaptar a las distintas resoluciones que utiliza el framework debemos usar la clase **img-responsive**. Cabe aclarar que las imágenes **SVG** en Internet Explorer (desde la versión 8 a la 10) se ven desproporcionadas, motivo por el cual los autores del framework recomiendan darle un ancho (*width*) de 100 %.

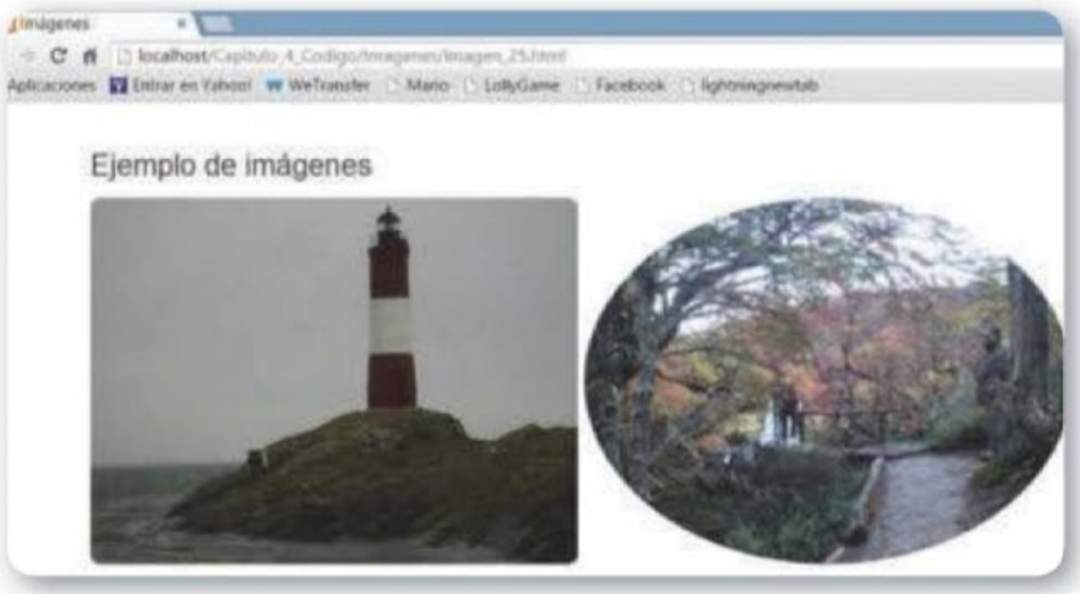


Figura 23. Podemos crear imágenes con bordes redondeados o imágenes circulares utilizando las clases que nos proporciona el framework.

Contamos también con clases que nos permiten cambiar el aspecto de la imagen: por ejemplo, darle un borde o un entorno redondeado, entre otras opciones. Para darle un **borde redondeado** tenemos la clase **img-rounded**. Y si queremos darle una **forma circular** podemos utilizar la clase **img-circle**:

```
<img src="imagen_1.JPG" class="img-rounded" >
<img src="imagen_2.JPG" class="img-circle" >
```

Thumbnails

Los *thumbnails* (**imágenes en miniatura**, en español) generalmente se utilizan para mostrar una escena de un video o una visualización en miniatura de una imagen de mayor tamaño (como en una galería de fotos). Para usar este tipo de imágenes, el framework cuenta con una clase llamada **thumbnail**. Con solo agregarla a la imagen, la reduce y nos proporciona la miniatura.

```
<img src="imagen_a.JPG" class="img-thumbnail">
```

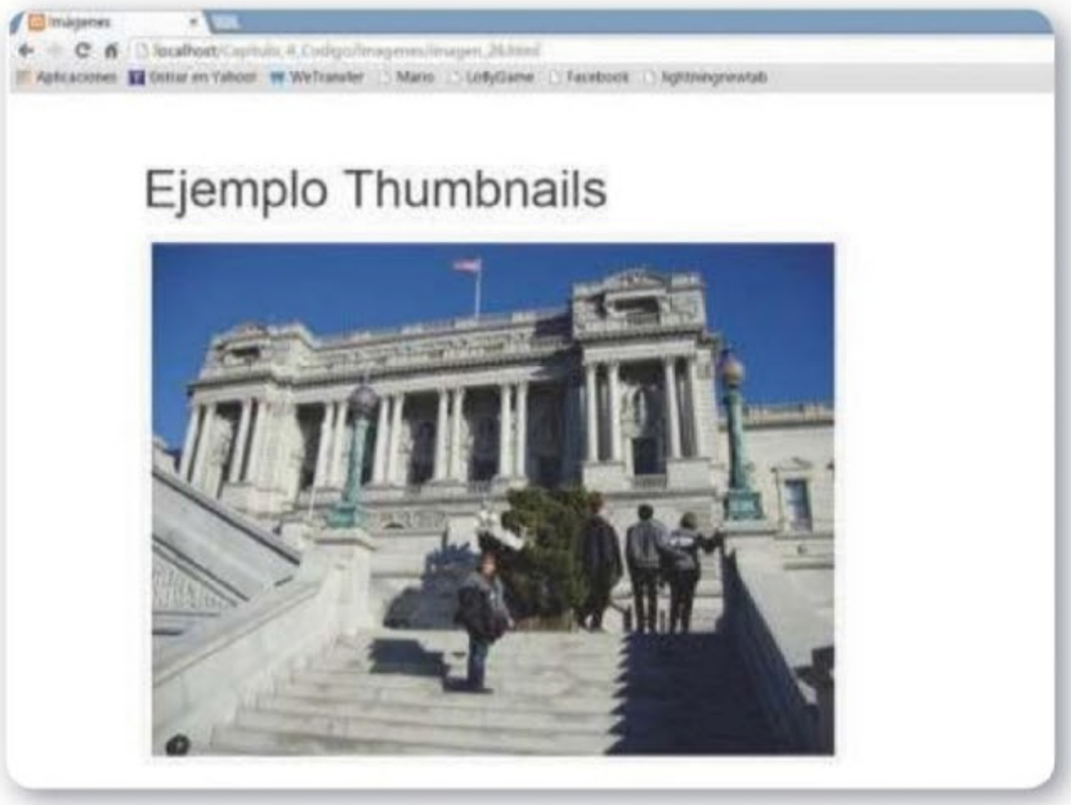


Figura 24. Los **thumbnails** pueden, a su vez, usar las mismas clases de las imágenes comunes, como ser **`img-rounded`** o **`img-circle`** para tener un mejor aspecto.

RESUMEN

En este capítulo explicamos cómo utilizar tipografía, encabezados y distintas variantes de texto utilizando clases. También conocimos los *blockquotes* y las tablas, así como las distintas clases para cambiar su color, forma y tamaño. Aprendimos a agregar botones de distintos colores y tamaños, cómo desactivarlos y agruparlos para formar menús y desplegables. A continuación, vimos los distintos tipos de iconos e imágenes que nos ofrece el framework. Por último, dedicamos una sección a los distintos tipos de etiquetas (*labels*) y a los *badges*, que nos permiten mostrar la cantidad de elementos disponibles en un determinado componente.

Actividades

TEST DE AUTOEVALUACIÓN

1. ¿Cómo hacemos para agregar tipografía en Bootstrap?
2. ¿Qué clase se utiliza para destacar un párrafo?
3. ¿Para qué sirven los *blockquotes*?
4. ¿Para qué se utiliza la clase **table-striped**?
5. ¿Se puede cambiar el tamaño de los botones? ¿Con qué clase?
6. ¿Qué clase debemos utilizar para hacer el contorno de una imagen circular?

EJERCICIOS PRÁCTICOS

1. A un documento HTML agréguele un encabezado que tenga un subtexto.
2. Luego añada un *blockquote* que se visualice en el margen derecho de la pantalla y que tenga una cita.
3. Agregue también un grupo de botones que se visualice verticalmente y que tenga tres botones de color azul.
4. Al mismo documento HTML agréguele un botón con un icono.
5. Por último, agregue también una imagen que tenga un contorno circular.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

Actividades

TEST DE AUTOEVALUACIÓN

1. ¿Cómo hacemos para agregar tipografía en Bootstrap?
2. ¿Qué clase se utiliza para destacar un párrafo?
3. ¿Para qué sirven los *blockquotes*?
4. ¿Para qué se utiliza la clase **table-striped**?
5. ¿Se puede cambiar el tamaño de los botones? ¿Con qué clase?
6. ¿Qué clase debemos utilizar para hacer el contorno de una imagen circular?

EJERCICIOS PRÁCTICOS

1. A un documento HTML agréguele un encabezado que tenga un subtexto.
2. Luego añada un *blockquote* que se visualice en el margen derecho de la pantalla y que tenga una cita.
3. Agregue también un grupo de botones que se visualice verticalmente y que tenga tres botones de color azul.
4. Al mismo documento HTML agréguele un botón con un icono.
5. Por último, agregue también una imagen que tenga un contorno circular.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

*05

Componentes generales

En este capítulo aprendemos cómo crear un menú de navegación utilizando botones y herramientas disponibles en el framework. Vemos qué es una paginación y cómo armarla. También explicamos cómo utilizar migas de pan y cómo configurar mensajes de alerta. Por último, creamos barras de progreso y configuramos listas y paneles.

↗↗↗

Menú

Un menú consiste en un conjunto de ítems a partir de los cuales los usuarios pueden acceder a una página web o realizar una determinada tarea previamente programada por el sistema.

Para realizar un menú desplegable en Bootstrap, tenemos que utilizar la clase de JavaScript **dropdown**. Para ello, es conveniente tener enlazadas las librerías correspondientes de JavaScript, como ya vimos en el **Capítulo 2** y subsiguientes. En este ejemplo, el enlace es por CDN:

```
<script src="http://code.jquery.com/jquery-2.1.3.min.js"></script>
```

Recordemos que las librerías deben ser incluidas antes del llamado a la librería de Bootstrap, para hacer uso de las clases que nos proporciona JavaScript.

Navegadores (navs)

Una variedad de menús son los **navegadores** (en inglés, *navs*), que nos permiten ir de una página a otra. Para esto, debemos utilizar la clase **.nav**, combinada con algún otro prefijo de clase de acuerdo al resultado que queramos mostrar, como pueden ser pestañas o píldoras.

Pestañas de navegación

Las pestañas de navegación, también conocidas como **solapas** (en inglés, *tabs*), le permiten al usuario navegar y ver contenido sin tener que cambiar de ventana.

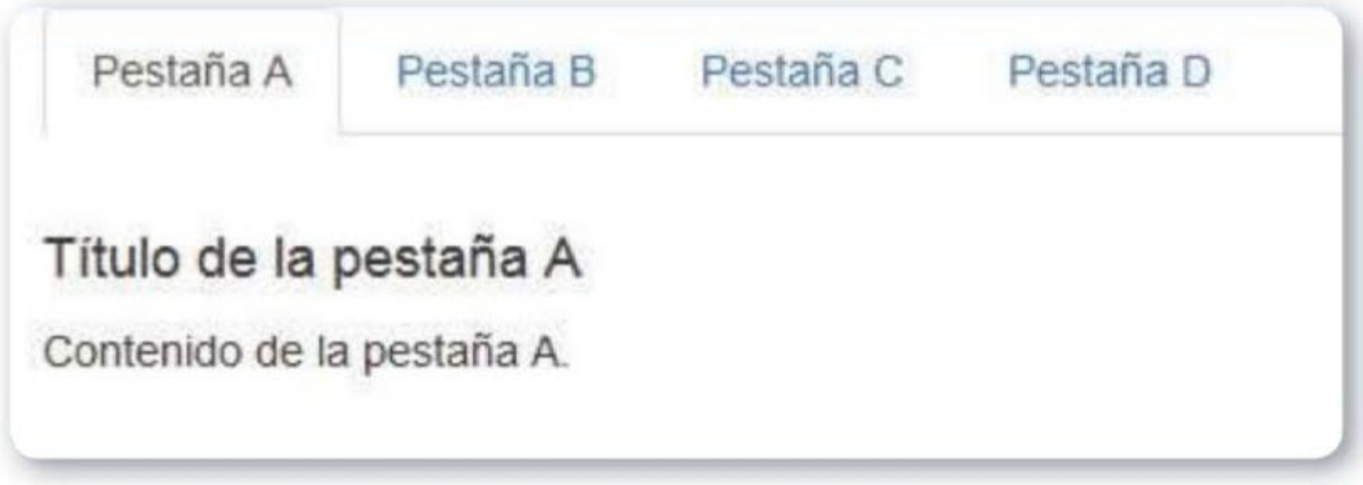


Figura 1. La navegación por pestañas nos permite agregar contenido más organizado a nuestro sitio web sin tener que cambiar de página.

Para poder realizar una **pestaña de navegación** en Bootstrap, a la clase **nav** tenemos que agregarle el prefijo de clase **nav-tabs**, dentro de una lista HTML **
**. En las etiquetas HTML de la lista **
** vamos a escribir el título y enlace de las diferentes pestañas de nuestro navegador. El siguiente código muestra un ejemplo de pestañas de navegación:

```
<ul class="nav nav-tabs">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#">Trabajos</a></li>
  <li><a href="#">Contacto</a></li>
</ul>
```

También podemos aprovechar la opción para que una de las pestañas esté activa con la clase **active** dentro de las etiquetas **
**, como lo hicimos en el ejemplo anterior.

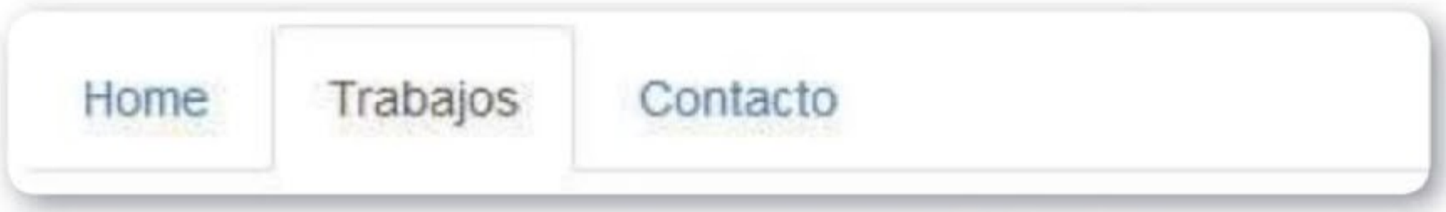


Figura 2. Tres pestañas con Bootstrap, en las que se puede apreciar la pestaña **Trabajos** que se encuentra activa mostrando su contenido gracias a la clase **active**.

Píldoras

El menú **píldoras** (en inglés, *pills*) es muy parecido al menú de pestañas de navegación. La diferencia radica en que este tipo de menú muestra cada opción como si fuera un botón (o píldora), en lugar de mostrar el recuadro de pestañas.

PARA DISPOSITIVOS MÓVILES

Cuando realizamos una barra de navegación para dispositivos móviles, generalmente esta se encuentra minimizada y, al tocarla, se despliega y muestra las opciones de nuestro menú. Si queremos cambiar el punto de ruptura para que se vea minimizada a partir de un determinado ancho de nuestra página y no en el estándar que nos ofrece el framework, tenemos que modificar el valor de la variable del archivo **Less** **@grid-float-breackpoint**.

Para crear un menú de navegación de píldoras tenemos que agregar la clase .**nav-pills**. De esta forma obtendremos un menú muy parecido a los menús tradicionales realizado con botones.

```
<ul class="nav nav-pills">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#">Trabajos</a></li>
  <li><a href="#">Contacto</a></li>
</ul>
```

Agregando la clase **active** veremos cómo ese enlace de nuestro menú aparece como activo o seleccionado.

```
<li class="active"><a href="#">Home</a></li>
```

Figura 3. El menú **píldoras** se obtiene agregando la clase **nav-pills**.

Opciones para configurar nuestro menú

Para dejar nuestro menú de navegación fijo en la parte superior de la página, debemos utilizar la clase **navbar-static-top**.

```
<nav class="navbar navbar-static-top" role="navigation">
</nav>
```

Esto hará que nuestro menú esté ubicado en la parte superior, pero si hacemos scroll, desaparecerá. En cambio, si queremos que quede fijo y no desaparezca, debemos utilizar la clase **navbar-fixed**.

Si queremos que los enlaces de la barra de navegación se muestren a la derecha de la pantalla, debemos utilizar la clase **pull-right**; si queremos que aparezcan a la izquierda, la clase **pull-left**.

```
<ul class="nav nav-pills pull-right">
    <li class="active"><a href="#">Home</a></li>
    <li><a href="#">Trabajos</a></li>
```

```
        <li><a href="#">Contacto</a></li>
    </ul>
```

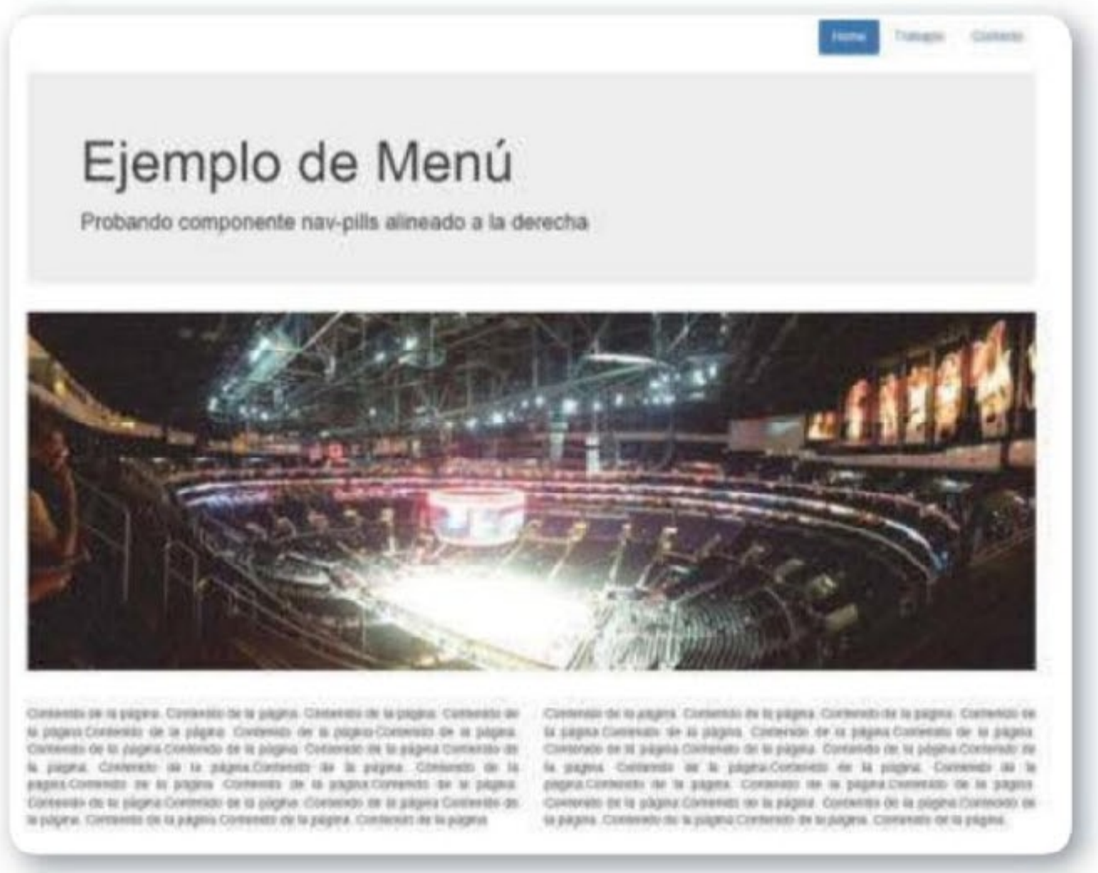


Figura 4. Para alinear el menú a la derecha, utilizamos la clase **`pull-right`**. Lo mismo ocurre con las clases **`navbar-left`** (izquierda) y **`navbar-right`** (derecha).

Otra opción interesante que nos puede servir —más que nada para dispositivos móviles o tablets— es ubicar el menú debajo de nuestra página web. Para esto debemos utilizar la clase **navbar-fixed-bottom**.

```
<ul class="nav nav-pills navbar-fixed-bottom ">
        <li class="active"><a href="#">Home</a></li>
        <li><a href="#">Trabajos</a></li>
        <li><a href="#">Contacto</a></li>
</ul>
```

TABS Y PILLS CON ICONOS

Para darle un impacto visual mayor a nuestras barras de navegación podemos agregar iconos, ya sea a las pestañas o a las píldoras. Para ello, le agregamos al elemento de la lista la clase **glyphicon** con el nombre del icono, como se muestra en el siguiente ejemplo: ** Contáctenos .**

Para deshabilitar un enlace y que no se pueda acceder a él, debemos utilizar la clase **disabled**. Esto hará que quede deshabilitado: el enlace se verá de un color gris claro y, al querer pulsarlo, aparecerá un círculo rojo cruzado que indica que no se puede acceder. De todas formas, si lo presionamos, accederemos al enlace; por eso, a esto debemos sumarle nuestra programación para asegurarnos de que no se pueda acceder a este link. Esto lo podemos realizar por medio de, por ejemplo, JavaScript:

```
<ul class="nav nav-pills ">
<li class="active"><a href="#">Home</a></li>
    <li><a href="#">Trabajos</a></li>
    <li class="disabled"><a href="#">Contacto</a></li>
</ul>
```

Figura 5. Con la clase **disabled** podemos desactivar en forma visual un ítem.

Hacer un **menú vertical** es sencillo, por medio de la clase **nav-stacked**. Esta clase hace que nuestro menú se vea de forma vertical y no horizontal, como sucede con la mayoría de los menús.

```
<ul class="nav nav-pills nav-stacked ">
<li class="active"><a href="#">Home</a></li>
    <li><a href="#">Trabajos</a></li>
    <li><a href="#">Contacto</a></li>
</ul>
```

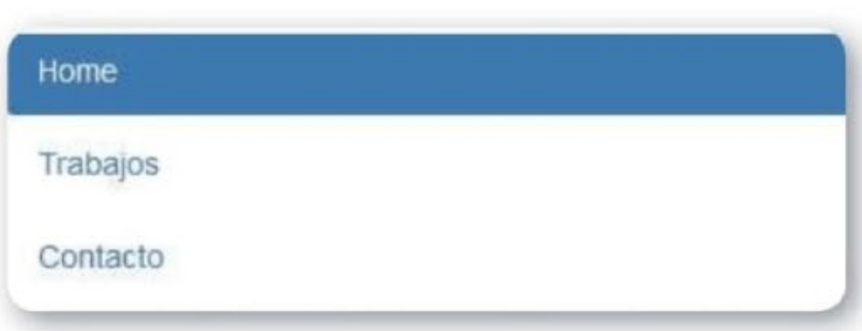


Figura 6. Podemos crear fácilmente un menú vertical con la clase **nav-stacked**.

Si queremos que nuestro menú ocupe todo el ancho del contenedor y que el ancho sea el mismo para todos sus elementos, podemos lograrlo utilizando la clase **nav-justified**.

```
<ul class="nav nav-pills nav-justified">
  <li class="active"><a href="#">Home</a></li>
  <li><a href="#">Trabajos</a></li>
  <li><a href="#">Contacto</a></li>
</ul>
```

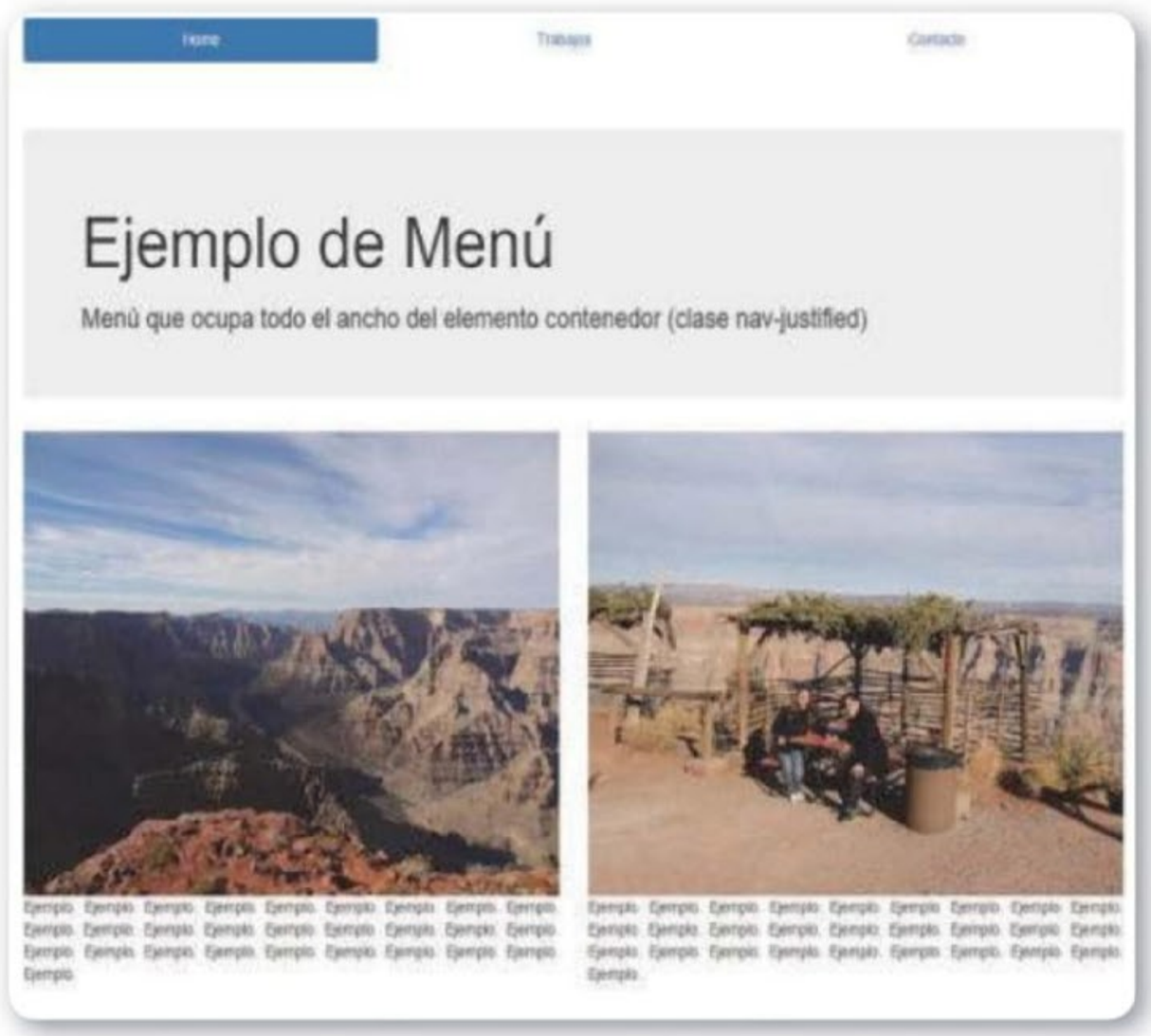


Figura 7. Menú justificado que ocupa el ancho del elemento de mayor tamaño dentro de los enlaces.

Paginación

La paginación (en inglés, *pagination*) consiste en numerar las páginas o documentos hasta el final. Esta numeración puede ser de acuerdo al contenido (nuevo o viejo), a determinadas categorías o al orden de importancia, entre otras.

Para hacer la paginación en Bootstrap contamos con dos tipos de numeración: puede ser con **números** (1, 2, 3, 4...) o con **enlaces** del tipo **anterior / siguiente**, o **antiguas / recientes.**

Por defecto, la paginación en Bootstrap es con números. Para crear una paginación en una lista HTML no ordenada **
, tenemos que agregar la clase **pagination, como podemos observar en el siguiente ejemplo:

```
<nav>
 <ul class="pagination">
  <li>
   <a href="#" aria-label="Anterior">
       <span aria-hidden="true">&laquo;</span>
   </a>
  </li>
  <li><a href="#">1</a></li>
  <li><a href="#">2</a></li>
  <li><a href="#">3</a></li>
  <li><a href="#">4</a></li>
  <li><a href="#">5</a></li>
<li><a href="#">6</a></li>
<li><a href="#">7</a></li>
  <li>
 <a href="#" aria-label="Siguiente">
     <span aria-hidden="true">&raquo;</span>
   </a>
  </li>
 </ul>
</nav>
```

Figura 8. Paginación clásica, utilizada, por ejemplo, en blogs.

Otra opción de paginación para sitios simples, como se recomienda en la página oficial de Bootstrap, es mediante la utilización de solamente dos botones: **anterior** o **siguiente**. Para esto, utilizamos la clase **pager**, como podemos ver en la siguiente sintaxis:

```
<nav>
 <ul class="pager">
  <li ><a href="#">Anterior</a></li>
  <li ><a href="#">Siguiente </a></li>
 </ul>
</nav>
```

Si queremos que los botones se alineen a ambos lados de la página, usamos, junto a **pager**, las clases **previous** (anterior) y **next** (siguiente):

```
<nav>
  <ul class="pager">
    <li class="previous"><a href="#"><span aria-hidden="true">&larr;</span>
Anterior</a></li>
    <li class="next"><a href="#">Siguiente <span aria-hidden="true">&rarr;</
span></a></li>
  </ul>
</nav>
```

Figura 9. Paginación simple con dos botones con el título **Anterior** y **Siguiente**. Podemos ponerles el nombre que creamos conveniente.

Migas de pan

Las migas de pan (en inglés, *breadcrumbs*) nos indican en dónde estamos ubicados con respecto a la navegación de nuestro sitio. En otras palabras, las **migas de pan** nos señalan el recorrido que hicimos y la forma de regresar mediante los enlaces por los cuales fuimos navegando. Para utilizarlas, debemos agregar la clase **breadcrumb** dentro de una lista HTML, como podemos observar en el siguiente ejemplo:

```
<ol class="breadcrumb">
  <li><a href="#">Home</a></li>
  <li><a href="#">Trabajos</a></li>
  <li class="active">Contacto</li>
</ol>
```

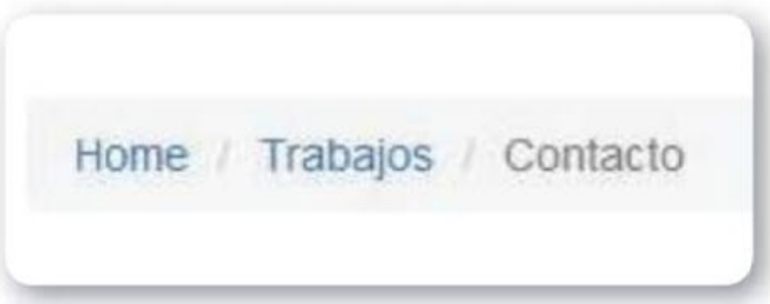


Figura 10. Las **migas de pan** se utilizan en programación web para orientarnos acerca de dónde estamos parados dentro del sitio.

Alertas

Los **mensajes de alerta** o **notificaciones** sirven para darle al usuario información sobre alguna acción que él mismo o el sistema ha generado. Para esto, debemos utilizar la clase **alert**.

Como vimos en otros componentes, estos mensajes de alerta también pueden cambiar de color utilizando las clases que vimos anteriormente: **alert-success**, **alert-info**, **alert-warning**, **alert-danger**.

```
<div class="alert alert-success">Mensaje Verde</div>
<div class="alert alert-info">Mensaje Azul</div>
<div class="alert alert-warning">Mensaje Amarillo</div>
<div class="alert alert-danger">Mensaje Rojo</div>
```

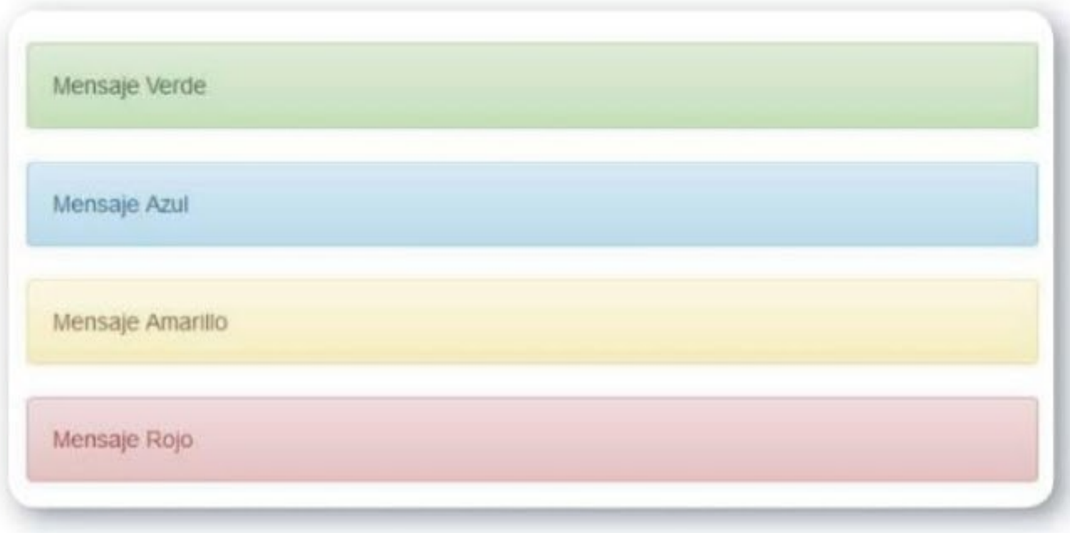


Figura 11. De acuerdo al contenido, podemos crear mensajes de alerta de distintos colores.

También podemos agregarle un botón de cierre del mensaje con la clase **close**:

```
<div class="alert alert-info">
<button type="button" class="close" data-dismiss="alert">&times;</button>
Mensaje con botón de cerrar
</div>
```

MENSAJES DE ALERTA ANIMADOS

Podemos crear un mensaje de alerta animado, incluyendo la clase **fade in** junto a la clase **alert**, como se muestra a continuación: ** **. Con esta clase crearemos un efecto de transición de desvanecimiento al cerrar el cuadro de mensaje de alerta.

Para que el botón con la cruz de cerrar funcione correctamente en todos los dispositivos, debemos agregarle el atributo **data-dismiss="alert"**. Y si queremos agregar un enlace en el texto, debemos utilizar la clase **alert-link**:

```
<div class="alert alert-info">
<button type="button" class="close" data-dismiss="alert">&times;</button>
Mensaje con botón de cerrar <a href="#" class="alert-link">con un enlace</a>
</div>
```

Figura 12. Mensaje de alerta con un botón de cerrar y un enlace.

Barra de progreso

Las barras de progreso o **barras de estado** (en inglés, *progress bar*) sirven para mostrarnos gráficamente el estado de una tarea. Por lo general, van acompañadas de una etiqueta que nos indica el porcentaje de progreso de la tarea, desde 0 % hasta 100 %. Para realizar una barra de progreso en Bootstrap debemos utilizar el siguiente código:

```
<div class="progress">
  <div class="progress-bar" role="progressbar" aria-valuenow="80" aria-value-
min="0" aria-valuemax="100" style="width: 80%;">
    <span class="sr-only">80% Completado</span>
  </div>
</div>
```

Como se ve en el ejemplo anterior, para crear la barra de progreso tenemos que usar la clase **progress**, que, a su vez, debe tener en su interior otro ** ** con la clase **progress-bar**. Allí cuenta con las propiedades **aria-valuenow** (valor actual), **aria-valuemin** (donde tenemos que darle el valor mínimo de la barra de progreso, generalmente 0) y **aria-valuemax** (donde indicamos el valor máximo que va a tener nuestra barra, generalmente 100).

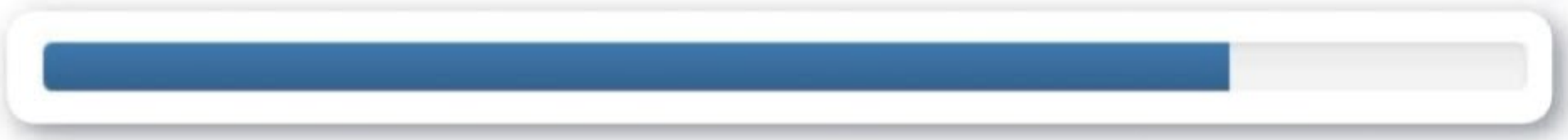


Figura 13. Barra de progreso en un 80 % de la tarea completada.

Si quitamos la etiqueta HTML ****, podemos ver el estado de la barra de progreso con el porcentaje recorrido hasta el momento.

```
<div class="progress">
  <div class="progress-bar" role="progressbar" aria-valuenow="80" aria-value-
min="0" aria-valuemax="100" style="width: 80%;">
    80% completado
  </div>
</div>
```

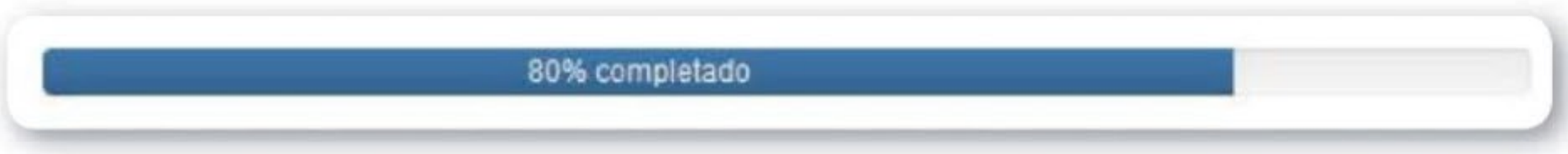


Figura 14. Podemos agregar el porcentaje completado a nuestra barra de progreso para darle al usuario el valor estimado que falta para completar la tarea.

También, como estuvimos viendo en componentes anteriores, se le puede agregar un color de acuerdo al estado o situación en que se encuentra nuestra barra de progreso.

```
<div class="progress">
  <div class="progress-bar progress-bar-success" role="progressbar"
      aria-valuenow="50" aria-valuemin="0" aria-valuemax="100"
      style="width: 50%">
    50% completado
  </div>
</div>
 <br>
 <div class="progress">
  <div class="progress-bar progress-bar-info" role="progressbar"
      aria-valuenow="50" aria-valuemin="0" aria-valuemax="100"
      style="width: 50%">
    50% completado
```

```
  </div>
</div>
<br>
 <div class="progress">
  <div class="progress-bar progress-bar-warning" role="progressbar"
      aria-valuenow="50" aria-valuemin="0" aria-valuemax="100"
      style="width: 50%">
    50% completado
  </div>
</div>

<br>
 <div class="progress">
  <div class="progress-bar progress-bar-danger" role="progressbar"
      aria-valuenow="50" aria-valuemin="0" aria-valuemax="100"
      style="width: 50%">
    50% completado
  </div>
 </div>
```

Otra opción que hace al diseño es agregarle rayas a la barra de progreso, lo que le da una forma acebrada. Para esto, debemos agregarle la clase **progress-bar-striped**.

```
    <div class="progress">
      <div class="progress-bar progress-bar-success progress-bar-striped"
role="progressbar"
      aria-valuenow="90" aria-valuemin="0" aria-valuemax="100"
style="width:90%">
       90% Completado
      </div>
    </div>
```

Figura 15. Con la clase **`striped`** le damos otro diseño a nuestra barra de progreso; en este caso, a rayas.

Agregando la clase **active** podemos crear una animación, lo que genera el efecto de que la barra se está moviendo en su interior.

```
<div class="progress">
  <div class="progress-bar progress-bar-success progress-bar-striped active"
role="progressbar"
    aria-valuenow="70" aria-valuemin="0" aria-valuemax="100"
    style="width: 70%">
   70% completado
  </div>
</div>
```

También podemos crear una **barra de progreso apilada** (*stacked*). Para esto, debemos incluir varias barras de progreso en el mismo ** **. El resultado va a ser una barra de progreso dividida en segmentos, cada uno de los cuales va a tener un color de acuerdo al porcentaje que se vaya completando.

Todas estas opciones que hemos mencionado nos permiten adaptar el sitio web a nuestras necesidades y darle un diseño específico.

```
<div class="progress">
  <div class="progress-bar progress-bar-success" style="width: 25%">
   25% completado
  </div>
  <div class="progress-bar progress-bar-warning progress-bar-striped"
style="width: 30%">
   30% completado
  </div>
  <div class="progress-bar progress-bar-danger" style="width: 20%">
   20% completado
  </div>
</div>
```

Figura 16. Podemos concatenar nuestra barra de progreso con varios colores. Para esto siempre se debe utilizar la clase **progress**.

Listas

Una lista en programación nos sirve para contener y mostrar datos del mismo tipo. En Bootstrap las listas deben estar contenidas por la clase **list-group** dentro de una etiqueta HTML **
** o ** **. Y a cada ítem de la lista deberemos agregarle la clase **list-group-item**, como vemos en el siguiente ejemplo de código:

```
<ul class="list-group">
  <li class="list-group-item">Argentina</li>
  <li class="list-group-item">Brasil</li>
  <li class="list-group-item">Chile</li>
  <li class="list-group-item">Ecuador</li>
  <li class="list-group-item">México</li>
  <li class="list-group-item">Uruguay</li>
  <li class="list-group-item">Venezuela</li>
</ul>
```

Figura 17. Ejemplo de lista de países utilizando la clase **`list-group`**.

Podemos agregar un **badge** a nuestras listas. Este componente nos sirve para mostrarle cierta información al usuario, como por ejemplo el número de mensajes de correo electrónico recibidos. Esta información se representa dentro de una burbuja o círculo en el extremo derecho de la lista y nos puede resultar muy útil en determinadas ocasiones.

Para que el elemento de la lista nos muestre un badge, debemos utilizar la clase **badge** dentro de una etiqueta HTML ****, como se muestra en el siguiente ejemplo de código:

```
<ul class="list-group">
  <li class="list-group-item">
    <span class="badge">22</span>
    Bandeja de entrada
  </li>
  <li class="list-group-item">
    <span class="badge">28</span>
    Correo sin leer
  </li>
  <li class="list-group-item">
    <span class="badge">1</span>
    Elementos enviados
  </li>
</ul>
```

Figura 18. Podemos agregarle una placa a la lista con la clase **badge**, como se observa en este ejemplo de un **servidor de correo**.

Con la clase **active** podemos dejar seleccionado un elemento de la lista, y con la etiqueta HTML **<a>** podemos crear un enlace a algún elemento de la lista.

Si queremos desactivar un elemento de la lista debemos utilizar la clase **disabled**. Esto hará que el elemento se vea de un color gris y al querer pulsarlo aparecerá un círculo rojo que avisa al usuario que no se puede hacer clic sobre el mismo.

```
<div class="list-group">
  <a href="#" class="list-group-item">Argentina</a>
  <a href="#" class="list-group-item disabled">Brasil</a>
  <a href="#" class="list-group-item">Chile</a>
  <a href="#" class="list-group-item">Ecuador</a>
```

```
  <a href="#" class="list-group-item">México</a>
  <a href="#" class="list-group-item">Uruguay</a>
   <a href="#" class="list-group-item">Venezuela</a>
</div>
```

Figura 19. Los elementos de una lista pueden contener enlaces y algunos de ellos pueden estar inhabilitados.

También podemos cambiar los colores de los elementos de la lista de acuerdo a su contenido, como podemos ver en el siguiente ejemplo:

```
<ul class="list-group">
  <li class="list-group-item list-group-item-success">Argentina</li>
  <li class="list-group-item list-group-item-info">Brasil</li>
  <li class="list-group-item list-group-item-warning">Chile</li>
```

LISTA EN LÍNEA

En lugar de visualizar los elementos de una lista uno debajo del otro, podemos representarlos uno al lado del otro, aplicando la clase **list-inline**. Esta clase se puede aplicar tanto para una lista ordenada como para una lista desordenada, por ejemplo:

```
<ul class="list-inline">
 <li > Argentina</li>
 <li >Brasil</li>
</ul>
```

```
  <li class="list-group-item list-group-item-danger">Ecuador</li>
  <li class="list-group-item list-group-item-success">México</li>
  <li class="list-group-item list-group-item-info">Uruguay</li>
  <li class="list-group-item list-group-item-danger">Venezuela</li>
</ul>
```

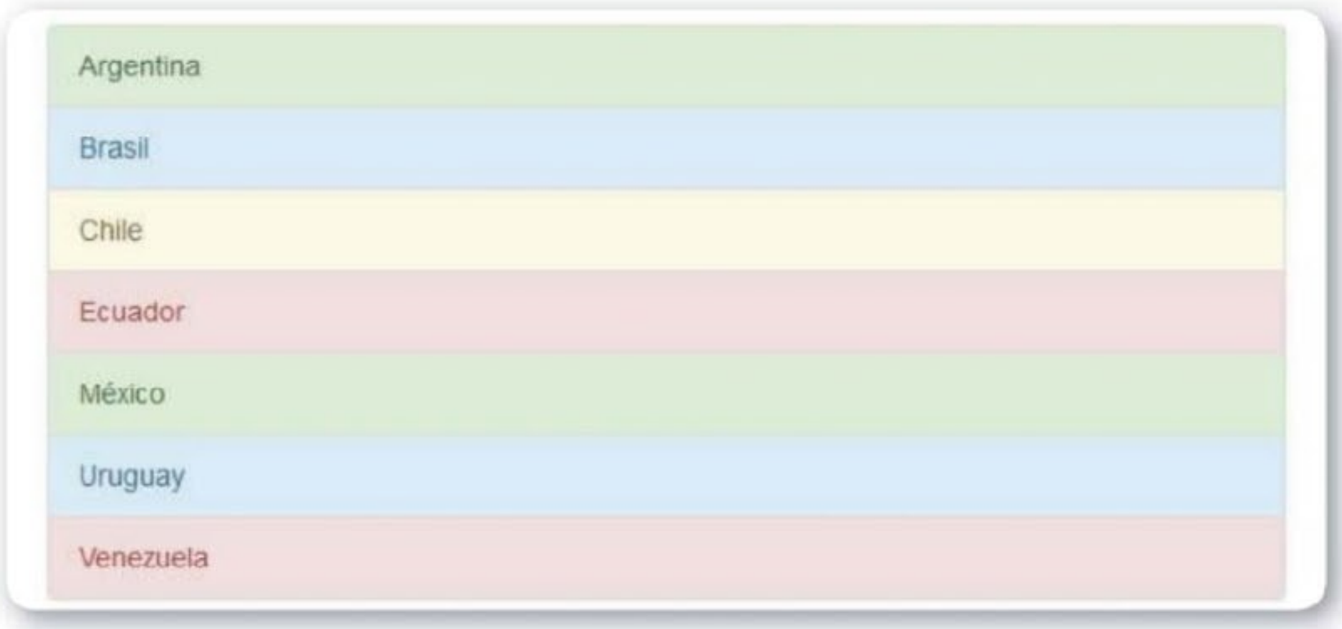


Figura 20. Utilizando diferentes clases podemos cambiar el color de los ítems de la lista.

También podemos agregar elementos HTML a nuestra lista, como cabeceras, párrafos y títulos, para darle un diseño aún más atractivo y adaptable a nuestra necesidad.

```
<div class="list-group">
  <a href="#" class="list-group-item active">
    <h4 class="list-group-item-heading">Argentina</h4>
    <p class="list-group-item-text">La República Argentina, conocida simplemente como Argentina, es un país de Sudamérica, ubicado en el extremo sur y sudeste de dicho subcontinente. (Wikipedia)</p>
  </a>

  <a href="#" class="list-group-item">
    <h4 class="list-group-item-heading">Brasil</h4>
    <p class="list-group-item-text">Brasil, oficialmente República Federativa del Brasil, es un país soberano de América del Sur que comprende la mitad oriental del subcontinente y algunos grupos de pequeñas islas en el océano Atlántico. (Wikipedia)</p>
```

```
    </a>

    <a href="#" class="list-group-item">
      <h4 class="list-group-item-heading">Chile</h4>
      <p class="list-group-item-text">Chile es un país de América ubicado en el
extremo sudoeste de América del Sur. Su nombre oficial es República de Chile y su
capital es la ciudad de Santiago. Chile se describe normalmente constituido por tres
zonas geográficas. (Wikipedia)</p>
    </a>
  </div>
```

Figura 21. Todas las posibilidades que nos brinda HTML podemos utilizarlas en una lista.

Paneles

Los paneles (en inglés, *panels*) nos sirven para mostrar contenido del mismo tipo o para resaltarlo dentro de un recuadro. En Bootstrap, contamos con muchas maneras de configurar los paneles para obtener un resultado acorde a nuestras necesidades y propósitos.

Con la clase **panel** obtendremos un **panel básico**, como podemos observar en el siguiente código de ejemplo:

```
<div class="panel panel-default">
  <div class="panel-body">
    Esto es un panel básico en Bootstrap.
  </div>
</div>
```

Como se observa, con la clase **panel panel-default** indicamos que nuestro panel es básico. Dentro de unas etiquetas ** ** con la clase **panel-body**, estamos marcando que eso es el cuerpo del panel, donde se va a escribir el contenido (en este ejemplo, escribimos **Esto es un panel básico en Bootstrap**).

Figura 22. Panel básico en Bootstrap con la clase **panel-default**.

Generalmente todo panel lleva un **encabezado** o **título**. Para realizarlo, tenemos dos opciones: con la clase **panel-heading** podemos crear un encabezado para nuestro panel y, también, como alternativa, agregarle un título HTML dentro de la clase **panel-title**. Es decir que podemos crear un encabezado o un título utilizando las etiquetas que nos proporciona HTML.

```
<div class="panel panel-default">
  <div class="panel-heading">Encabezado del panel</div>
  <div class="panel-body">
    Contenido del panel.
  </div>
</div>

<div class="panel panel-default">
  <div class="panel-heading">
    <h3 class="panel-title">Título del panel</h3>
  </div>
  <div class="panel-body">
    Contenido del panel.
  </div>
</div>
```

Podemos observar en el código del ejemplo que, a diferencia del encabezado, el título debe ir en una etiqueta HTML de título. En este ejemplo, usamos la etiqueta **<h3></h3>**.

Figura 23. Podemos agregarle a un panel un encabezado o un título con las clases **panel-heading** o **panel-title**.

Así como podemos agregar un título, podemos agregar además un **pie de panel** (en inglés, *footer*): con una etiqueta ** ** agregamos la clase **panel-footer**. De esta forma, quedará configurado un pie en la parte inferior de nuestro panel.

```
<div class="panel panel-default">
  <div class="panel-heading">Encabezado del panel</div>
  <div class="panel-body">
    Contenido del panel.
  </div>
  <div class="panel-footer">Pie del panel</div>
</div>
```

Como vimos anteriormente con otros componentes, podemos cambiarle el color a los paneles utilizando las diferentes clases que nos brinda el framework:

```
<div class="panel panel-default panel-primary">
  <div class="panel-heading">Encabezado del panel</div>
  <div class="panel-body">
    Contenido del panel.
  </div>
  <div class="panel-footer">Pie del panel</div>
</div>

<div class="panel panel-default panel-success">
  <div class="panel-heading">Encabezado del panel</div>
  <div class="panel-body">
    Contenido del panel.
  </div>
  <div class="panel-footer">Pie del panel</div>
```

```
</div>

<div class="panel panel-default panel-info">
  <div class="panel-heading">Encabezado del panel</div>
  <div class="panel-body">
    Contenido del panel.
  </div>
  <div class="panel-footer">Pie del panel</div>
</div>

<div class="panel panel-default panel-warning">
  <div class="panel-heading">Encabezado del panel</div>
  <div class="panel-body">
    Contenido del panel.
  </div>
  <div class="panel-footer">Pie del panel</div>
</div>

<div class="panel panel-default panel-danger">
  <div class="panel-heading">Encabezado del panel</div>
  <div class="panel-body">
    Contenido del panel.
  </div>
  <div class="panel-footer">Pie del panel</div>
</div>
```

Otra opción interesante es encerrar mediante un panel una tabla o una lista para mostrar datos. En el caso de usar una tabla, podemos agregar un encabezado y el contenido del panel seguido de la tabla, o bien hacer que el contenido del panel sea la propia tabla:

```
<div class="panel panel-default panel-primary">
 <div class="panel-heading">Fútbol Sudamericano</div>
 <div class="panel-body">
   <p>Equipos de Fútbol</p>
 </div>

 <!-- Tabla -->
```

```
<table class="table">
  <th>Argentina</th>
  <th>Chile</th>
  <th>Paraguay</th>
  <tr>
    <td>River Plate</td>
    <td>Colo-Colo</td>
    <td>Olimpia</td>
  </tr>
  <tr>
    <td>Boca Juniors</td>
    <td>Unversidad de Chile</td>
    <td>Cerro Porteño</td>
  </tr>
 </table>
</div>

<div class="panel panel-default panel-primary">
  <div class="panel-heading">Fútbol Sudamericano</div>

  <!-- Tabla -->
  <table class="table">
   <th>Argentina</th>
   <th>Chile</th>
   <th>Paraguay</th>
```

JUMBOTRON

Podemos resaltar un contenido para llamar la atención del usuario empleando la clase **jumbotron**. Esta clase hará que la información se muestre sobre un cuadro de fondo gris y amplía el tamaño de fuente del texto que contenga en su interior. Si incluimos la clase **jumbotron** dentro de un elemento contendor (**container**), ocupará el ancho del mismo.

```
<div class="jumbotron">
  <h1>Título</h1>
  <p>Párrafos para d</p>
 </div>
```

```
      <tr>
        <td>River Plate</td>
        <td>Colo-Colo</td>
        <td>Olimpia</td>
      </tr>
      <tr>
        <td>Boca Juniors</td>
        <td>Unversidad de Chile</td>
        <td>Cerro Porteño</td>
      </tr>
    </table>
  </div>
```

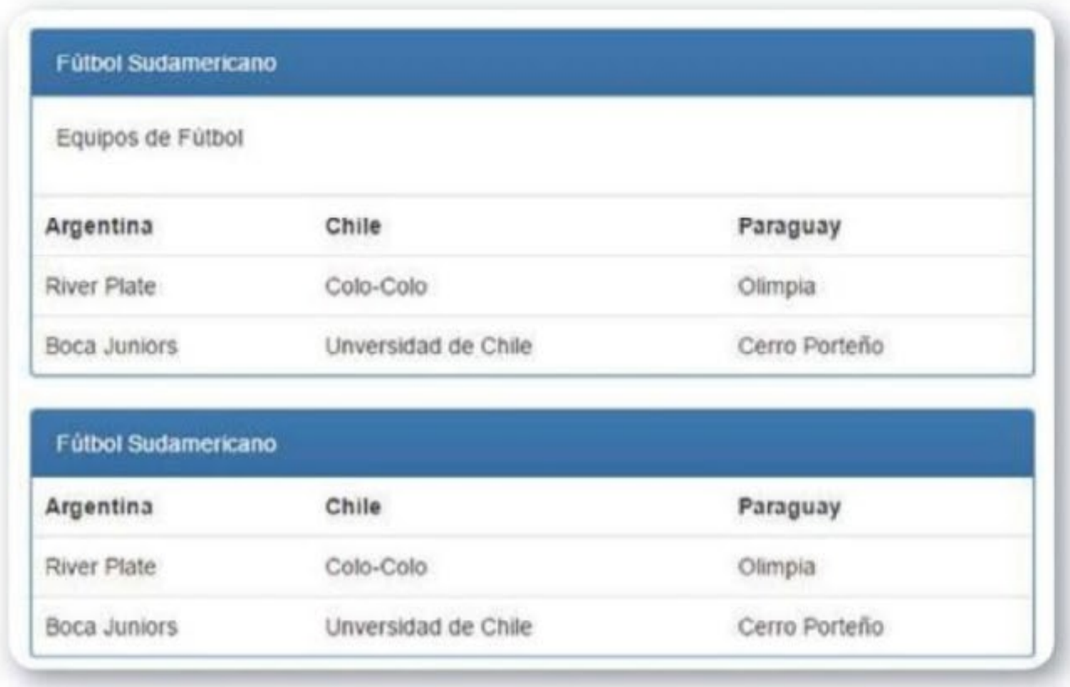


Figura 24. Se pueden utilizar diferentes formas de mostrar tablas usando un panel.

Para mostrar una lista dentro de un panel tenemos que hacer exactamente lo mismo que con las tablas, mostrando la misma debajo del contenido:

```
<div class="panel panel-default panel-primary">
  <div class="panel-heading">Encabezado del panel</div>
```

LISTAS EN HTML

La **lista no ordenada** consiste en un conjunto de elementos relacionados entre sí en los que no se indica un orden o secuencia determinados. HTML nos provee de las etiquetas **<ul>** para definir la lista y la etiqueta **<li>** para definir cada uno de los elementos de la lista.

```
<div class="panel-body">
  <p>Contenido del panel</p>
</div>

<!-- Lista-->
<ul class="list-group">
  <li class="list-group-item">Argentina</li>
  <li class="list-group-item">México</li>
  <li class="list-group-item">España</li>
  <li class="list-group-item">Italia</li>
  <li class="list-group-item">Uruguay</li>
 </ul>
</div>
```

Figura 25. Podemos crear listas dentro de un panel para resaltar aún más el diseño.

RESUMEN

En este capítulo vimos qué es un menú y cómo realizarlo. Aprendimos a crear navegadores, pestañas y menú de píldoras con sus distintas configuraciones. Explicamos qué es la paginación y sus dos opciones para realizarla. Posteriormente, vimos qué son las migas de pan y cómo crear mensajes o alertas y las distintas opciones que existen para crear barras de progreso. También explicamos cómo crear listas y paneles, cómo agregarle un título, un encabezado y un pie; y, por último, cómo agrupar una tabla o lista dentro de un panel.

Actividades

TEST DE AUTOEVALUACIÓN

1. ¿Qué es un menú de navegación y qué tipos de menús nos ofrece Bootstrap?
2. ¿Cuál es la diferencia entre un menú de píldoras y un menú de pestañas de navegación?
3. ¿Qué es la paginación y para qué sirve?
4. ¿Para qué se utilizan las migas de pan o breadcrumbs?
5. ¿Cuál es la clase que se utiliza para crear un mensaje de alerta?
6. ¿Para qué se utilizan las barras de progreso?
7. ¿Se puede incluir dentro de un panel una lista o tabla?

EJERCICIOS PRÁCTICOS

1. Realice un menú de navegación del tipo pestañas.
2. Realice un menú de navegación del tipo píldoras.
3. Realice un mensaje de alerta con la posibilidad de que el usuario pueda cerrarlo, informándole que su sesión ha expirado.
4. Cree una lista con cinco elementos.
5. Programe un panel con un título, un contenido y un pie.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

*06

Formularios

Aprendemos en este apartado qué son los formularios y cómo utilizarlos en Bootstrap. Conocemos sus distintos componentes y las distintas maneras de configurarlos. Por último, vemos un ejemplo práctico de cómo armar un formulario de ingreso de usuarios, utilizando iconos y el sistema de columnas del framework.

↗↗↗

Formularios

Un formulario se utiliza para recibir datos del usuario. Estos luego son enviados y procesados en el servidor con un lenguaje como, por ejemplo, PHP.

Todo formulario debe tener un **método de envío** (**method**) y una **ruta al archivo** en el servidor en donde se van a procesar esos datos (**action**). En este libro hablaremos del envío de datos a un servidor utilizando el lenguaje de programación **PHP**, aunque el concepto nos va a servir para todos los lenguajes que procesan datos del lado del servidor.

Un **formulario básico** en HTML tiene la siguiente estructura:

```
<form action="archivo.php" method="POST">
    <!--Contenido del formulario   -->
    <label for="nombre">Ingrese su nombre: </label>
    <input type="text" name="nombre" placeholder="ingrese su nombre">

    <!--Botón para el envío de datos   -->
    <input type="submit" value="Enviar">

</form>
```

En este ejemplo, el formulario cuenta con un control HTML **input** del tipo **text**, en donde el usuario debe ingresar su nombre y después hacer clic en el botón **Enviar**, para mandar esos datos que ingresó al servidor para ser procesados por medio del lenguaje de programación PHP.

Figura 1. Formulario simple en HTML sin Bootstrap.

En Bootstrap, contamos con tres tipos de diseño de formulario: vertical o básico, en línea y horizontal.

Formulario vertical o básico

Para crear un formulario de este tipo debemos utilizar la clase **form-control** que nos va a permitir agrupar los distintos componentes del formulario. Por defecto, estos toman el estilo de Bootstrap, pero

agregándoles algunas clases podemos modificar el comportamiento de los componentes del formulario. A continuación, podemos observar el mismo formulario de la **Figura 1**, pero esta vez utilizando el framework:

```
<form action="archivo.php" method="POST">
<!--Contenido del formulario  -->
    <div class="form-group">
      <label for="nombre">Ingrese su nombre: </label>
      <input type="text" name="nombre" class="form-control"
placeholder="ingrese su nombre">
    </div>
    <div class="form-group">
      <label for="apellido">Ingrese su Apellido: </label>
      <input type="text" name="apellido" class="form-control"
placeholder="ingrese su apellido">
    </div>
    <!--Botón para el envío de datos  -->
    <input type="submit" class="btn btn-default" value="Enviar">
</form>
```

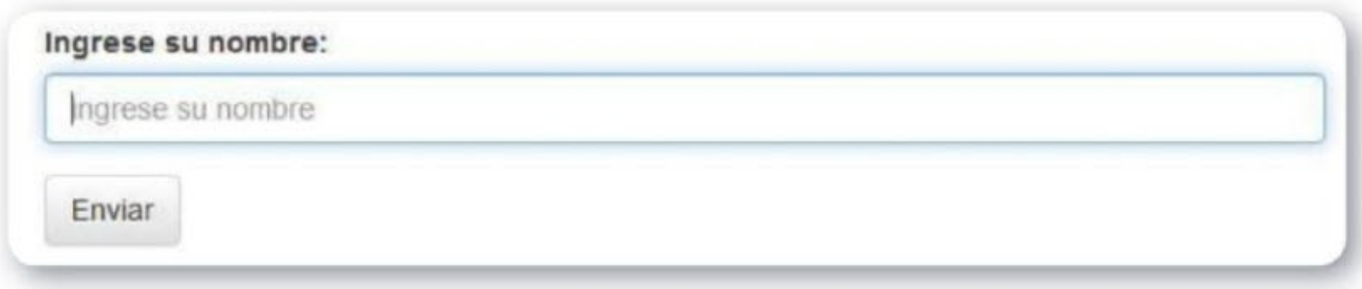


Figura 2. Formulario básico creado mediante Bootstrap.

Observando el código anterior, vemos que la clase **form-group** nos permite agrupar cada elemento **label** con su correspondiente **input**. Si agregamos la clase **from-control** a un componente —como, en nuestro ejemplo, a los componentes **input**— le estamos proporcionando una anchura del 100 % a dicho componente.

Formulario en línea

Los formularios en línea (en inglés, *inline form*) tienen la particularidad de que todos sus componentes se encuentran en una línea. Este tipo de formulario se utiliza cuando tenemos

que trabajar para espacios reducidos, como tablets o teléfonos inteligentes, generalmente en medidas con un ancho inferior a 768 px.

Para crear un formulario en línea debemos utilizar la clase **form-inline**, directamente en la etiqueta de creación del formulario HTML **<form> </form>**. Si también le añadimos a cada **label** la clase **sr-only**, ocultamos las etiquetas y de esta forma ocupamos menos espacio en pantalla. Al emplear esta clase debemos utilizar la propiedad **placeholder**.

```
<form action="archivo.php" method="POST" class="form-inline">
<!--Contenido del formulario  -->
   <div class="form-group">
      <label class="sr-only" for="nombre">Ingrese su nombre: </label>
      <input type="text" name="nombre" class="form-control"
placeholder="ingrese su nombre">
   </div>
   <div class="form-group">
      <label class="sr-only" for="apellido">Ingrese su Apellido: </label>
      <input type="text" name="apellido" class="form-control"
placeholder="ingrese su apellido">
   </div>
   <!--Botón para el envío de datos  -->
   <input type="submit" class="btn btn-default" value="Enviar">
</form>
```

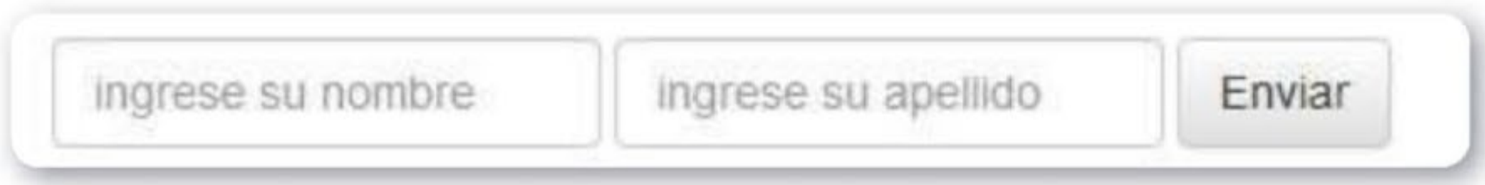


Figura 3. Los formularios en línea nos sirven cuando estamos programando para resoluciones con espacios reducidos.

PLACEHOLDER

Placeholder se puede traducir al español como "marcador de posición". La propiedad **placeholder** nos sirve para darle al usuario una sugerencia de qué es lo que queremos que escriba en un campo de texto del formulario. Esa sugerencia o ayuda aparecerá dentro del campo de texto como marca de agua. De esta forma, nos permite ahorrar espacio, por lo que se recomienda su uso en teléfonos inteligentes.

Formulario horizontal

Podemos también utilizar la clase **form-horizontal** para que el formulario se alinee utilizando la cuadrícula de Bootstrap sin la necesidad de utilizar la clase **.row**.

En el siguiente ejemplo podemos observar cómo dividimos en columnas del tipo **–md** cada grupo del formulario (**.form-group**): los **label** en **2** (**col-md-2**) y los **input** en **10** (**col-md-10**), lo que nos da un total de **12**.

```
<form action="archivo.php" method="POST" class="form-horizontal">
<!--Contenido del formulario  -->
   <div class="form-group">
     <label  for="nombre" class="col-md-2">Ingrese su nombre: </label>
     <div class="col-md-10">
       <input type="text" name="nombre" class="form-control"
placeholder="ingrese su nombre">
     </div>
   </div>
   <div class="form-group">
     <label for="apellido" class="col-md-2">Ingrese su Apellido: </label>
     <div class="col-md-10">
       <input type="text" name="apellido" class="form-control"
placeholder="ingrese su apellido">
     </div>
   </div>
   <!--Botón para el envío de datos  -->
   <div class="form-group">
     <input type="submit" class="btn btn-primary" value="Enviar">
   </div>
</form>
```

Figura 4. Podemos dividir los elementos de nuestro formulario como si fuera una grilla de Bootstrap, sin utilizar la clase **.row**.

Componentes de un formulario

El framework nos permite adecuar los distintos componentes de HTML5 pertenecientes a un formulario para hacer aún más llevadera la experiencia de usuario.

Cuadros de texto

El cambio generado en HTML5 con respecto a las etiquetas **input** ha sido muy importante, ya que se han agregado varios valores nuevos a la propiedad **type**. El framework nos da la posibilidad de adaptar el estilo de los mismos a nuestro formulario.

El campo de texto más utilizado es el que tiene la propiedad **input** por valor **text**, que presenta un aspecto similar al que se muestra en la **Figura 5**.

Figura 5. Cuadro de texto **input** del tipo **text**, el más usado en formularios.

También podemos incluir la clase **input-group**, que nos proporciona la opción de agregar un botón o imagen al cuadro de texto. Esto puede ser al comienzo o al final del **text** y se puede utilizar en varios componentes, como checkboxes, radios, botones y listas desplegables, entre otros.

```
<div class="input-group">
  <span class="input-group-addon">$</span>
     <input type="text" name="nombre" class="form-control"
placeholder="Ingrese el importe">
     <span class="input-group-addon">.00</span>
</div>
```

Figura 6. Agregando la clase **input-group-addon** le podemos insertar una imagen en forma de botón al componente.

Existen distintos tipos de **text input**, como contraseña, teléfono, e-mail y otros. Para crearlos, solamente tenemos que agregar al atributo **type** el valor que necesitemos para cada uno. En la **Tabla 1** vemos los posibles valores que podemos asignarle a este atributo:

VALORES DEL ATRIBUTO TYPE

VALOR	DESCRIPCIÓN	SINTAXIS
password	Permite el ingreso de texto, pero oculta visualmente los caracteres ingresados reemplazándolos por puntos o asteriscos.	<input type="password" name="pass">
number	Se utiliza para ingresar valores numéricos.	<input type="number" name="numero">
email	Permite ingresar direcciones de correo electrónico.	<input type="email" name="email">
url	Se utiliza para ingresar una dirección web.	<input type="url" name="url">
time	Permite ingresar una hora.	<input type="time" name="hora">
tel	Se emplea para ingresar números de teléfono.	<input type="tel" name="tel">
date	Permite ingresar una fecha. Dependiendo del navegador, puede aparecer o no el selector de fecha en el campo.	<input type="date" name="fecha">
month	Permite seleccionar un mes y un año.	<input type="month" name="mes">
week	Permite seleccionar una semana del año.	<input type="Week" name="semana">
datetime	Permite seleccionar una fecha y hora.	<input type="datetime" name="fechayhora">
color	Se utiliza para seleccionar un color.	<input type="color" name="color">

Tabla 1. Valores que podemos asignarle al atributo **type**.

Textarea

Este componente se utiliza para crear texto de más de una línea. Con la propiedad **rows** podemos configurar la cantidad inicial de estas líneas; en este ejemplo, el valor es **5**:

```
<textarea class="form-control" rows="5"></textarea>
```

Figura 7. A diferencia del componente **text**, el **texarea** nos permite escribir más de una línea de texto.

Checkboxes y radios

Los *checkboxes* (**casillas de verificación**) y *radios* (**botones de opciones**) nos permiten la selección de uno o varios ítems dentro de un conjunto de opciones. Si utilizamos un checkbox, podremos seleccionar más de una opción y si, en cambio, utilizamos un radio, podremos elegir solamente una.

```
<h3>Varias opciones</h3>
<div class="checkbox">
                <label>
                        <input type="checkbox" value="">
            Me gusta el helado de chocolate
      </label>
      </div>
    <div class="checkbox">
                <label>
                        <input type="checkbox" value="">
            Me gusta el helado de vainilla
                </label>
      </div>
    <div class="checkbox">
                <label>
              <input type="checkbox" value="">
                Me gusta el helado de limón
      </label>
      </div>
    <div class="checkbox disabled">
                <label>
                        <input type="checkbox" value="" disabled>
```

```
            Me gusta el helado de crema (opción deshabilitada)
    </label>
    </div>
<h3>Sólo una opción</h3>
<div class="radio">
            <label>
        <input type="radio" name="Radios" id="optionsRadios1"
value="option1" checked>
            Masculino
        </label>
        </div>
    <div class="radio">
            <label>
                <input type="radio" name="Radios"
id="optionsRadios2" value="option2">
            Femenino
            </label>
    </div>
```

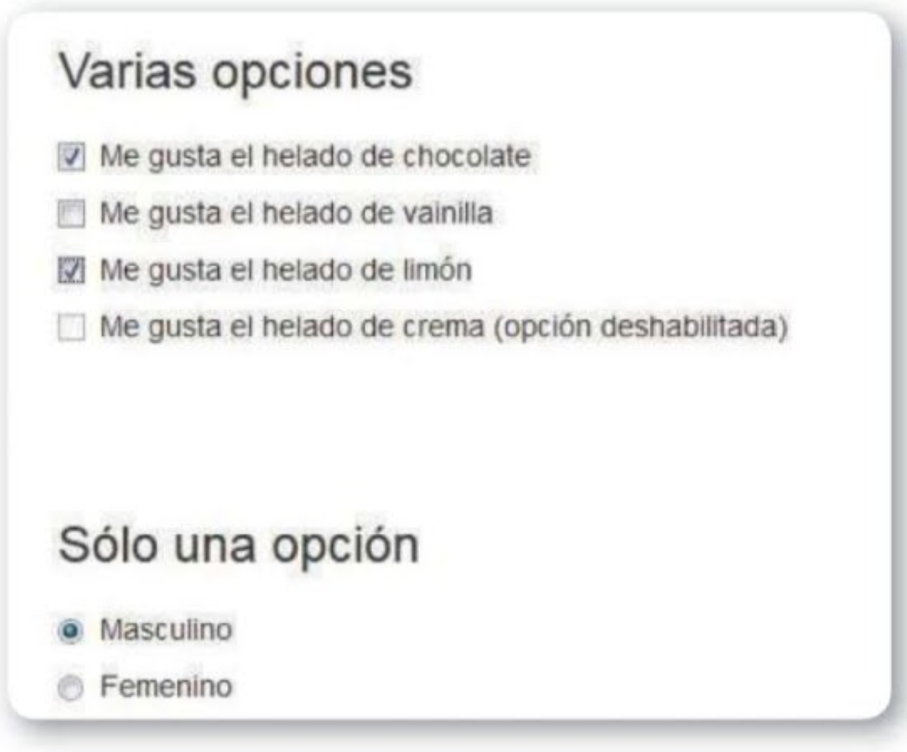


Figura 8. Checkboxes y radios en Bootstrap que nos permiten seleccionar varias opciones o solo una, respectivamente.

Para deshabilitar un checkbox o un radio, como en el ejemplo anterior, simplemente tenemos que agregar la clase **disabled**.

A diferencia de los checkboxes, en el caso de los radios, estos deben tener siempre el mismo **name**, ya que de esta forma se agrupan, y pueden tener **solo una opción**. Si no estuvieran agrupados, cumplirían el mismo rol que los checkboxes, ya que se comportarían de manera individual y podríamos entonces seleccionar más de una opción.

La propiedad **checked**, tanto para checkboxes como para radios, nos permite tener seleccionada una opción al momento de cargar la página, como se puede apreciar, en el ejemplo anterior, en los radios que tienen la opción **Masculino** seleccionada al momento de cargar la página web.

Podemos usar también la clase **inline** con checkboxes y radios, para que ocupen menos espacio en nuestro formulario.

```
<h3>Varias opciones</h3>
<div class="checkbox-inline">
                <label>
                        <input type="checkbox" value="">
            Me gusta el helado de chocolate
      </label>
      </div>
    <div class="checkbox-inline">
                <label>
                        <input type="checkbox" value="">
            Me gusta el helado de vainilla
                </label>
      </div>

<div class="checkbox-inline">
                <label>
            <input type="checkbox" value="">
             Me gusta el helado de limón
      </label>
      </div>
   <div class="checkbox-inline">
                <label>
                        <input type="checkbox" value="">
            Me gusta el helado de crema
      </label>
      </div>

<h3>Sólo una opción</h3>
    <div class="radio-inline">
                <label>
            <input type="radio" name="Radios" id="optionsRadios1"
```

```
value="option1" checked>
                Masculino
            </label>
            </div>
        <div class="radio-inline">
                    <label>
                        <input type="radio" name="Radios"
id="optionsRadios2" value="option2">
                Femenino
                    </label>
    </div>
```

Figura 9. Agregando la clase **`inline`** podemos ahorrar espacio en nuestro formulario cuando optamos por resoluciones de tablets o smartphones.

Listas desplegables (comboboxes)

Los *comboboxes* o listas desplegables (en inglés, *select*) cumplen las mismas funciones que los checkboxes o radios, con la diferencia de que las opciones no están visibles sino hasta que se hace clic sobre el componente. Recién en ese momento se despliegan en forma de lista. Estas opciones pueden ser seleccionadas de a una, como en el caso de los radios, o varias en simultáneo, como los checkboxes. Para este último caso, debemos agregarle a la etiqueta **select** la opción **multiple**, como se observa en el siguiente ejemplo:

```
<h3>Varias opciones</h3>
<select multiple class="form-control">
                <option>Chocolate</option>
    <option>Vainilla</option>
    <option>Limón</option>
    <option>Crema</option>
```

```
        </select>
<h3>Sólo una opción</h3>
<select class="form-control">
                <option>Masculino</option>
        <option>Femenino</option>
        <option>Empresa</option>
</select>
```

Figura 10. Con las listas desplegables podemos seleccionar una o varias opciones, como con los checkboxes y radios.

Configuraciones generales

Existen configuraciones generales que son comunes a casi todos los componentes de un formulario y que son utilizados generalmente en los campos de texto, como la validación, altura y ancho, o la ayuda visual para los usuarios.

Validación de errores

De acuerdo al estado de los componentes, podemos cambiar el color para indicar, por ejemplo, un error o un ingreso exitoso en el formulario. De esta manera, indicamos con la clase **.has-success** un ingreso de datos exitoso; con **.has-warning**, una advertencia; y con **.has-error**, un error en el formulario.

```
<div class="form-group has-success">
<label class="control-label" for="exito">Dato ingresado con éxito</label>
   <input type="text" class="form-control" id="exito">
</div>
<div class="form-group has-warning">
<label class="control-label" for="advertencia">Advertencia sobre el dato in-
gresado</label>
   <input type="text" class="form-control" id="advertencia">
</div>
<div class="form-group has-error">
<label class="control-label" for="error">Error al ingresar el dato</label>
   <input type="text" class="form-control" id="error">
</div>
```

También podemos agregarle un icono para hacer notar aún más el estado del campo. Esto se logra agregando las clases **has-feedback** y **glyphicon**:

```
<div class="form-group has-success has-feedback">
<label class="control-label" for="exito">Dato ingresado con éxito</label>
   <input type="text" class="form-control" id="exito">
   <span class="glyphicon glyphicon-ok form-control-feedback" aria-
hidden="true"></span>
</div>
<div class="form-group has-warning has-feedback ">
<label class="control-label" for="advertencia">Advertencia sobre el dato in-
gresado</label>
   <input type="text" class="form-control" id="advertencia">
   <span class="glyphicon glyphicon-warning-sign form-control-feedback" aria-
hidden="true"></span>
</div>
<div class="form-group has-error has-feedback">
<label class="control-label" for="error">Error al ingresar el dato</label>
   <input type="text" class="form-control" id="error">
   <span class="glyphicon glyphicon-remove form-control-feedback" aria-
hidden="true"></span>
</div>
```

Figura 11. También podemos insertar un icono para hacer más claro el mensaje que enviamos al usuario acerca del ingreso de los datos.

Control estático

En muchas ocasiones necesitamos mostrar directamente el valor de un campo, sin ser necesario que el usuario lo agregue. Por ejemplo: si en un formulario anterior o inicio de sesión ya se le pidió al usuario que ingresara su nombre, podemos directamente asignarlo al formulario para que no tenga que volver a ingresar esa información. Para esto, debemos utilizar la clase **.form-control-static** unida a la etiqueta HTML ** ** (párrafo).

```
<div class="form-group ">
<label class="control-label" for="nombre">Nombre: </label>
   <p class="form-control-static">Pepe Rodriguez</p>
</div>
<div class="form-group ">
<label class="control-label" for="mail">E-mail: </label>
   <input type="text" class="form-control" id="mail">
</div>
```

Figura 12. Con la clase **control-static** podemos ingresar automáticamente algunos datos en el formulario.

Desactivar campo

Para desactivar un campo del formulario debemos utilizar la propiedad **disabled**, que toma por valor **true** o **false** (verdadero o falso).

```
<div class="form-group ">
<label class="control-label" for="mail">E-mail: </label>
   <input type="text" class="form-control" id="mail" disabled>
</div>
```

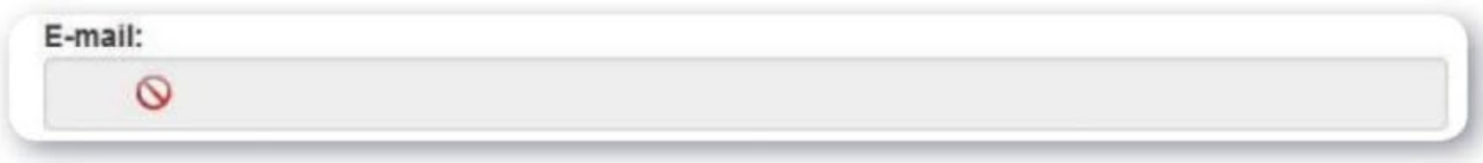


Figura 13. Utilizando el atributo **`disabled`** podemos deshabilitar la edición de un control.

También contamos con la opción de deshabilitar un grupo de componentes, siempre y cuando estén dentro de la etiqueta HTML **<fieldset></fieldset>**.

```
<fieldset disabled>
<div class="form-group ">
                <label class="control-label" for="mail">E-mail: </label>
    <input type="text" class="form-control" id="mail" disabled>
    </div>
  <div class="form-group ">
                <label class="control-label" for="telefono">Teléfono: </label>
    <input type="text" class="form-control" id="telefono" disabled>
    </div>
  <div class="form-group ">
                <label class="control-label" for="celular">Celular: </label>
    <input type="text" class="form-control" id="celular" disabled>
    </div>
</fieldset>
```

Tamaño de los campos

Podemos cambiar el tamaño de los campos, utilizando la clase **input-lg** para agrandarlos, o **input-sm** para achicarlos.

```
    <div class="form-group ">
        <input type="text" class="form-control " id="normal"
placeholder="Tamaño normal" >
    </div>
    <div class="form-group ">
        <input type="text" class="form-control input-lg" id="grande"
placeholder="Tamaño grande" >
    </div>
    <div class="form-group ">
        <input type="text" class="form-control input-sm" id="chico"
placeholder="Tamaño chico" >
    </div>
```

Texto de ayuda

Agregando la etiqueta **** de HTML y la clase **help-block**, podemos agregar un texto de ayuda para el usuario, que lo oriente en el ingreso de datos en el formulario.

```
    <div class="form-group ">
        <label for="apellido" class="col-md-2">Apellido: </label>
      <input type="text" class="form-control " id="apellido" >
      <span id="bloqueAyuda" class="help-block">En este campo debe ingresar su
Apellido</span>
    </div>
```

Figura 14. Agregar un texto de ayuda hace más fácil el ingreso de datos.

SELECCIÓN DE UN CAMPO

Según el estado del campo, al estar seleccionado el estilo del framework se genera un remarcado, de color azul, alrededor del mismo (en los bordes) para hacernos notar que fue seleccionado. Esto se debe a que Bootstrap utiliza la propiedad **box-shadow** para darle esa sombra azul, que se aplica a la clase **:focus** de estilos CSS.

Ejemplo práctico: formulario de ingreso

Para crear un formulario de ingreso —comúnmente llamado **formulario de login**— vamos a necesitar un formulario y tres etiquetas HTML **input**. Como en este ejemplo no vamos a crear un estilo CSS propio sumado al de Bootstrap, trataremos de acomodarlo utilizando algún recurso de HTML, como pueden ser las etiquetas **</br>**. También lo centraremos utilizando el sistema de rejillas que utiliza el framework. El resultado final que obtendremos se asemeja al que podemos observar en la **Figura 15**.

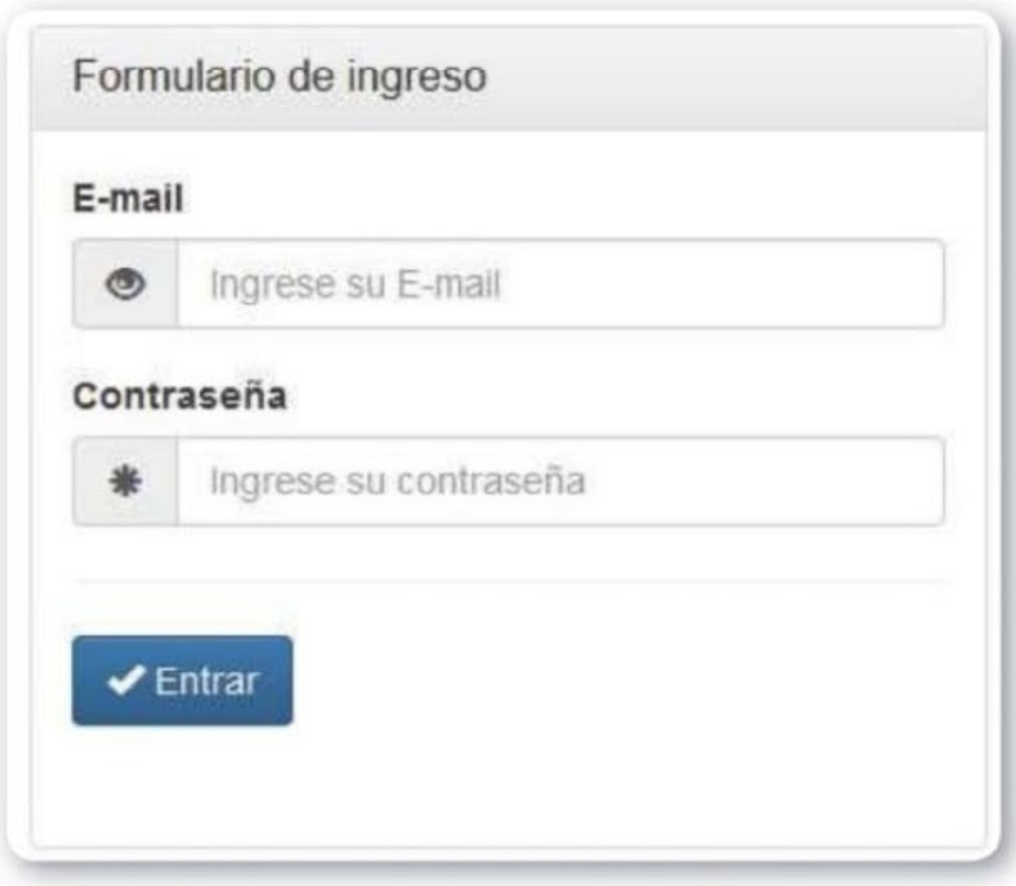


Figura 15. Formulario de ingreso de usuario a un sitio web, desarrollado mediante Bootstrap.

Para comenzar, primero tenemos que armar nuestro sistema de rejillas. Debemos tener en cuenta que si lo queremos centralizar en la página, debemos dividir el sistema en tres columnas; de esta forma, dejamos las primeras cuatro de la izquierda libres; en las cuatro del medio armamos el login, y dejamos las últimas cuatro de la derecha también libres. Así, el formulario nos quedará centrado en el medio de la página web. Vamos a usar una resolución **md** para PC de escritorio.

```
<div class="row">
<div class="col-md-4">
      <!-- LIBRE -->
</div>
```

```
<div class="col-md-4">
    <!-- FORMULARIO -->
</div>
<div class="col-md-4">
    <!-- LIBRE -->
</div>
</div>
```

El formulario, como vimos en la imagen del ejemplo, va a tener dos **input** del tipo **text**. En este caso, mediante HTML5, podemos facilitar la programación cambiando el **type="text"** por, en el primer caso, **type="email"** y, en el segundo, por **type="password"**, ya que para el inicio de sesión requerimos que el usuario ingrese el e-mail y luego la contraseña.

AL UTILIZAR HTML5 FACILITAMOS MUCHAS TAREAS DE PROGRAMACIÓN EN BOOTSTRAP

Con estos dos nuevos valores que utiliza la propiedad **type**, en el caso del e-mail, si el usuario ingresa algo diferente, la validación del mismo se hace automáticamente sin necesidad de programarla. Por ejemplo, si ingresa algún texto sin el @, al presionar el botón **Entrar**, le aparecerá un mensaje de error para que ingrese una dirección de correo válida. Y en el caso de la contraseña, el ingreso se hace mediante puntos, así no se puede ver lo que escribió el usuario. Todo esto lo hace el componente de forma automática, sin tener que programar absolutamente nada. También le agregamos iconos a nuestro formulario con las clases **glyphicon glyphicon-eye-open** y **glyphicon glyphicon-asterisk** respectivamente.

LECTORES DE PANTALLA

El lector de pantalla (*screen reader*) es una aplicación que interpreta lo que mostramos en la pantalla y, en combinación con un sintetizador de voz, lo reproduce en forma de audio. Este software es de gran ayuda para las personas que tienen problemas de visión o para los no videntes. Actualmente muchos sistemas operativos cuentan con este tipo de aplicación.

```
<div class="form-group">
<label for="mail">E-mail</label>
   <div class="input-group">
                <span class="input-group-addon">
                <span class="glyphicon glyphicon-eye-open">
                </span>
      </span>
         <input type="email" class="form-control" id="mail" placeholder="Ingrese
su E-mail">
   </div>
   </div>
   <div class="form-group">
   <label for="pass">Contraseña</label>
      <div class="input-group">
                <span class="input-group-addon">
                <span class="glyphicon glyphicon-asterisk">
                </span>
         </span>
         <input type="password" class="form-control" id="pass"
placeholder="Ingrese su contraseña">
   </div>
   </div>
```

Al botón de **Entrar** le agregamos la clase **btn btn-primary** para que quede de color azul y con las etiquetas HTML ****

le agregamos la clase **glyphicon glyphicon-ok**, para que nos muestre el icono sobre el botón. Finalmente, el código completo del ejemplo:

```
<div class="container">
<br/><br/>
    <div class="row">
        <div class="col-md-4"></div>
        <div class="col-md-4">
            <div class="panel panel-default">
                <div class="panel-heading">
                    <h3 class="panel-title">Formulario
de ingreso</h3>
                </div>
            <div class="panel-body">
              <form action="#" method="POST" >
            <div class="form-group">
                <label for="mail">E-mail</label>
                <div class="input-group">
                    <span class="input-group-addon">
                        <span
class="glyphicon glyphicon-eye-open">
                        </span>
                    </span>
                <input type="email" class="form-
control" id="mail" placeholder="Ingrese su E-mail">
                </div>
            </div>
            <div class="form-group">
                <label for="pass">Contraseña</label>
                <div class="input-group">
```


ACCESIBILIDAD

Con el propósito de mejorar la accesibilidad, para aquellas personas que presentan dificultades en la visión y emplean un software lector de pantallas, es conveniente que al crear un formulario utilicemos, en la etiqueta para crearlo, la propiedad **role** con el valor **form**, de la siguiente forma: **<form-role="form">-.**

```
                                    <span class="input-group-addon">
                                        <span class="glyphicon
glyphicon-asterisk">
                                        </span>
                                    </span>
                                    <input type="password" class="form-control"
id="pass" placeholder="Ingrese su contraseña">
                                    </div>
                                </div>
                                <hr/>
                                <button type="submit" class="btn btn-primary"><span
class="glyphicon glyphicon-ok"></span> Entrar</button>
                                <p><br/></p>
                                </form>
                            </div>
                        </div>
                    </div>
                    <div class="col-md-4">
                    </div>
                </div>
            </div>
```

RESUMEN

En este capítulo vimos qué es un formulario y qué tipos existen, según su ubicación: vertical, en línea u horizontal. También analizamos los distintos componentes de un formulario y para qué se utiliza cada uno. Para terminar, desarrollamos un ejemplo completo de un formulario de ingreso o login, analizando qué componentes utilizar y cómo configurar el diseño de 12 columnas de Bootstrap.

Actividades

TEST DE AUTOEVALUACIÓN

1. ¿Qué es un formulario y para qué se utiliza?
2. Enumere distintos tipos de cuadros de texto.
3. ¿Cuál es la diferencia entre un **checkbox** y un **radio**?
4. ¿Cuál es la propiedad que se utiliza para deshabilitar un componente?
5. Aparte del tamaño estándar, ¿qué otros dos tamaños existen para los componentes **text**?

EJERCICIOS PRÁCTICOS

1. Cree un formulario y en el interior agregue dos campos de texto.
2. A los campos de texto del ejercicio anterior agrégueles un texto de ayuda.
3. Al formulario del ejercicio 1 agréguele un botón del tipo **submit** para poder enviar los datos.
4. Al formulario anterior agréguele un checkbox con 4 opciones y un radio con solamente 2 opciones.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

*07

Ejemplo práctico: diseño de un portfolio

A través de un ejemplo sencillo, vemos cómo diseñar y armar un sitio web partiendo de un diseño recibido. Aprendemos las distintas formas de diagramar el sitio web, así como el armado de las diferentes páginas. Explicamos cómo realizar los distintos enlaces entre las páginas del sitio y finalmente aprendemos a realizar una página de contacto utilizando un formulario en Bootstrap.

Diseño

El diseño y armado de nuestra página web de ejemplo consistirá en un **portfolio**, esto es, un sitio en el que un diseñador gráfico muestra todos sus trabajos en fotografía, dibujo y editorial. El sitio tendrá también una página de contacto para realizar consultas o pedidos de presupuesto. Para poder realizar el portfolio, tenemos que tener en cuenta que todo comienza con el diseño. Para esto, partiremos de un diseño que, si bien está incompleto, ya que es solo a modo de ejemplo, nos servirá para ver cómo podemos trabajar utilizando el framework. Una vez que terminemos este ejemplo, tendremos la posibilidad de armar nuestra propia web o bien, a partir de un diseño, diagramar la estructura HTML utilizando Bootstrap.

Figura 1. Diseño de un sitio web de portfolio realizado en Adobe Photoshop.

PORTFOLIO

Un portfolio (o **portafolio**, en español) no es más que la muestra de trabajos (ya sea en fotos o con enlaces a páginas web) realizados por un diseñador gráfico, diseñador web o fotógrafo, entre otros. Es como si fuera un *curriculum vitae* en donde se puede acceder a los trabajos más importantes realizados por el profesional. Por esta razón, es lo más recomendado para darse a conocer en la Web o conseguir clientes.

Organización

Para comenzar a trabajar, lo primero que tenemos que hacer es organizarnos. La organización consiste en, simplemente, armar los directorios del proyecto para que, una vez terminado, se pueda entender con facilidad en dónde están los archivos que lo componen. También es importante tener organizado el proyecto para trabajar en equipo, ya que no todos trabajamos de la misma manera. Por eso es mejor estandarizar la forma de trabajo, para que sea entendible en futuras versiones o por diferentes desarrolladores.

Al directorio principal de nuestro proyecto lo vamos a llamar **portfolio**, y es ahí en donde vamos a ubicar todos los archivos y carpetas que lo van a componer. También debemos crear una carpeta **css** para agregar nuestros propios estilos CSS, más allá del estilo principal que utilicemos con las librerías de Bootstrap —que llamaremos por medio de CDN (red de entrega de contenidos)—; un directorio para incluir los archivos PHP, que podemos llamar **inc**; y, por último, un directorio de imágenes compuesto por varias carpetas de acuerdo a la categoría de imágenes que necesitemos, al que llamaremos **img**. En caso de que trabajemos con los archivos que componen el framework de Bootstrap, podemos utilizar una carpeta llamada **js**, pero, como dijimos anteriormente, el enlace lo haremos por medio de CDN.

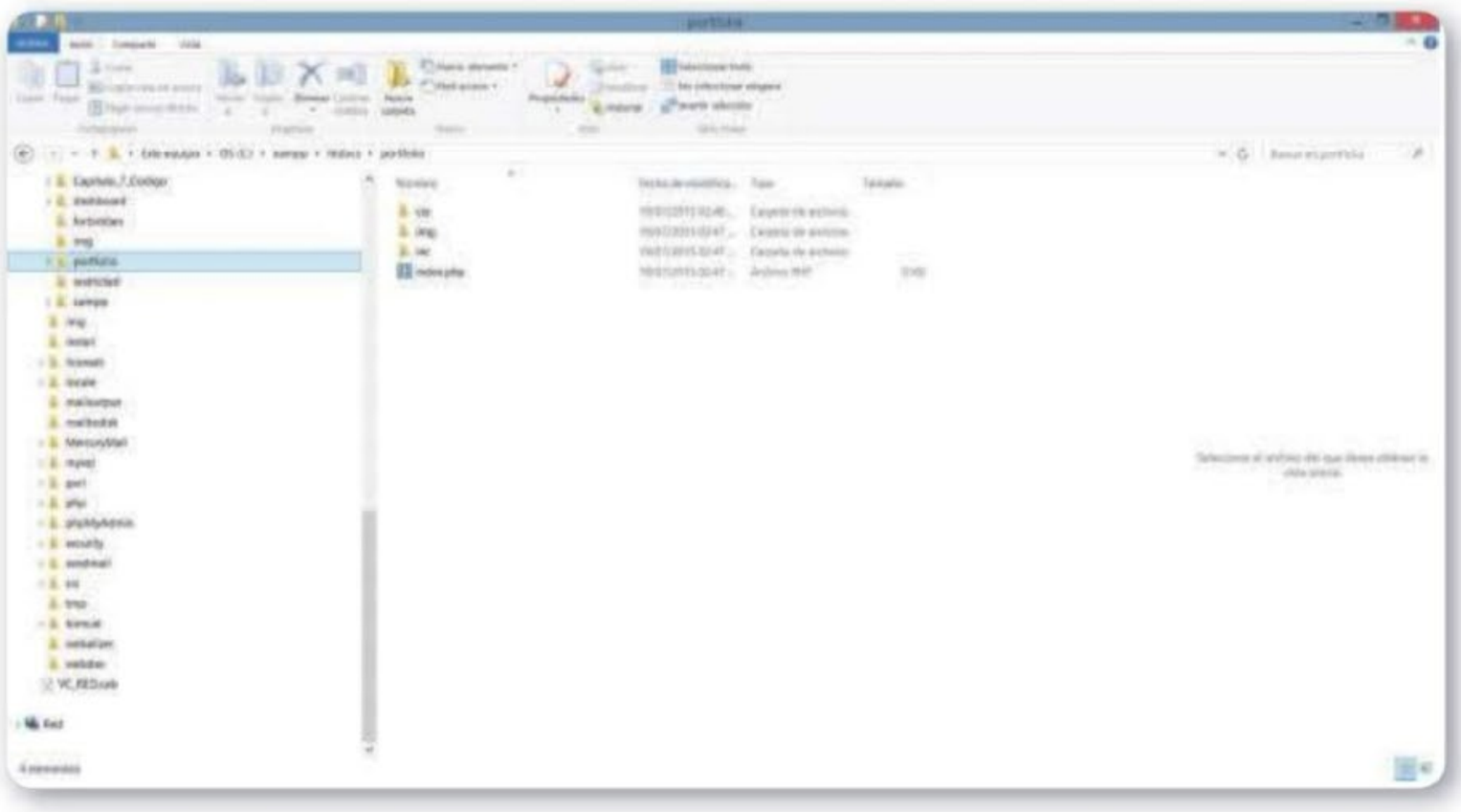


Figura 2. La organización del directorio de nuestro proyecto nos permite trabajar más fácil y cómodamente.

La manera en la que trabajaremos, teniendo en cuenta la estructura de los directorios que hemos creado, va a ser la siguiente: en el directorio principal crearemos un archivo llamado **index.php** que será el encargado de contener la información principal del sitio con los diferentes enlaces a otros directorios:

```
<!DOCTYPE html>
<html lang="es">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <title>Mi portfolio</title>

        <!-- Bootstrap -->
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/boot-
strap/3.3.4/css/bootstrap.min.css">
            <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/
bootstrap/3.3.4/css/bootstrap-theme.min.css">
            <link rel="stylesheet" href="css/estilo.css" type="text/css">

        <!-- jQuery -->
            <script src="http://code.jquery.com/jquery-2.1.3.min.js"></script>
            <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/js/
bootstrap.min.js"></script>
    </head>
    <body>
            <?php
                include 'inc/cabezal.php';

                include 'inc/detalle.php';

                include 'inc/pie.php'
            ?>
    </body>
</html>
```

Como podemos observar en el código, toda la estructura dentro de la etiqueta HTML **<head></head>** es igual a lo que estuvimos viendo

en ejemplos anteriores. Dentro de estas etiquetas se incluyen los enlaces a las librerías tanto de Bootstrap como de jQuery y un enlace al directorio del proyecto llamado **css**. En este directorio es donde vamos a incluir un archivo llamado **estilo.css**. Este enlace lo escribimos después del llamado a la librería propia del framework **css**.

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

A continuación, dentro de las etiquetas HTML **<body></body>**, hacemos las llamadas a los archivos PHP que necesitamos para armar la estructura de la página principal. Estos archivos se llaman:

- **cabezal.php**: será el encargado de contener el menú y el logo.
- **detalle.php**: incluirá toda la estructura principal de la página.
- **pie.php**: contendrá el pie de página.

Los archivos **cabezal.php** y **detalle.php** son los que se van a repetir en todas las páginas del sitio. Es por eso que armamos una estructura incluyendo estos archivos con la función PHP **include()**. Con solo invocarla y con la ayuda de estilos, vamos a tener una estructura fija para todas las páginas sin la necesidad de repetir varias veces el código del pie de página y del cabezal.

Menú principal

Como pudimos observar en la **Figura 1**, el sitio cuenta con un menú principal y, como explicamos en el **Capítulo 5**, el mismo sirve para tener un acceso directo a otras partes del sitio. El menú principal está situado en la parte superior, centrado en el medio de la página. Cuenta con un lugar reservado para el logo en el sector izquierdo y con los diferentes enlaces en el sector derecho. Estos enlaces son:

- **Inicio**: comúnmente llamado, en inglés, *home*. Es obligatorio en todos los sitios ya que le permite al usuario volver a la portada.
- **Biografía**: tendrá una breve descripción del trabajo y antecedentes o estudios del autor.

- **Trabajos**: a su vez, cuenta con un menú desplegable con la descripción de las distintas categorías de trabajo: **Fotografía**, **Dibujo** y **Editorial**.
- **Contacto**: se incluirá allí un formulario que nos servirá para comunicar a los usuarios con el dueño del sitio.

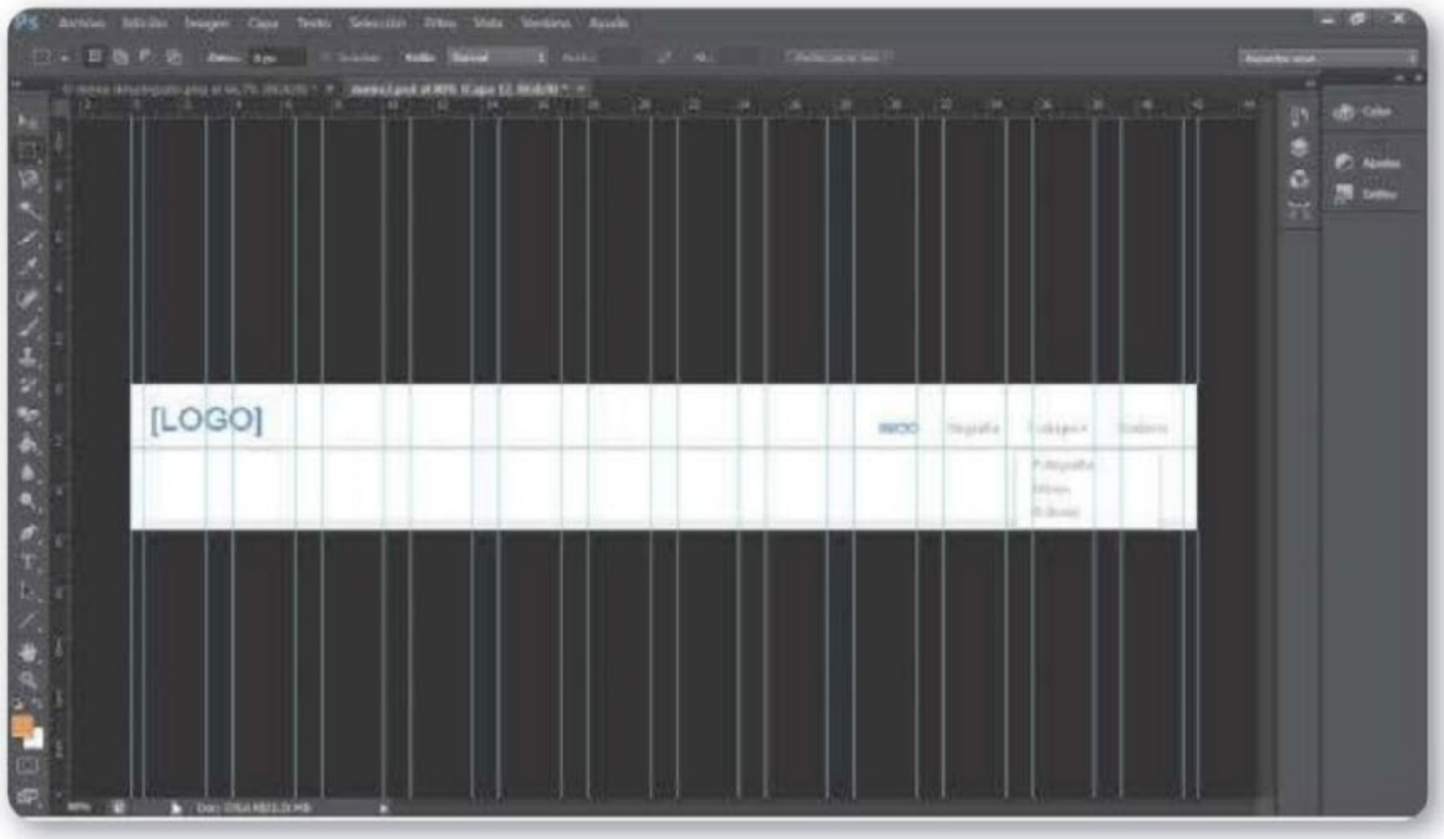


Figura 3. Menú de nuestro sitio web realizado en Adobe Photoshop CC para adaptar en Bootstrap.

Para diseñar el menú, creamos un archivo llamado **cabezal.php**, que es el que incluimos desde el archivo **index.php**. Luego lo guardamos dentro de la carpeta **inc**.

```
<header id="main-header">
    <div class="container">
        <div class="row">
            <div class="col-md-3">
                <h1 id="main-logo"><a href="index.
php"><span>[LOGO]</span></a></h1>
            </div>
            <div class="col-md-9">
                <ul id="main-menu" class="nav nav-pills">
                    <li class="active"><a href="index.
php">INICIO</a></li>
                    <li><a href="biografia.
php">Biografía</a></li>
```

```
                                        <li class="dropdown">
                                            <a href="#"
class="dropdown-toglle" data-toggle="dropdown">Trabajos<span class="caret"></
span></a>
                                            <ul class="dropdown-menu"
role="menu">
                                                <li><a
href="fotografia.php">Fotografía</a></li>
                                                <li><a
href="dibujo.php">Dibujo</a></li>
                                                <li><a
href="editorial.php">Editorial</a></li>
                                            </ul>
                                        </li>
                                        <li><a href="contacto.
php">Contacto</a></li>
                    </ul>
                        </div>
                </div>
        </div>
    </header>
```

Como estamos definiendo la cabecera de nuestra página, vamos a utilizar las etiquetas HTML **<header></header>**. Y como la idea para el proyecto es trabajar para plataformas de escritorio mayores a 992 px, utilizamos la clase **col-md**. En este caso, el espacio ocupado por el menú es mayor que el del logo, por lo que vamos a utilizar **col-md-9** para el primero y **col-md-3** para el segundo. Esto nos dará un total de 12 columnas, como corresponde en el sistema de grillas de Bootstrap.

> **PARA EL PORTFOLIO NO UTILIZAREMOS LOS ESTILOS QUE NOS PROVEE BOOTSTRAP, SINO UNO PROPIO**

Para crear el menú utilizamos la clase **nav nav-pills**, y para que la opción **Trabajos** sea desplegable con sus tres opciones (**Fotografía**, **Dibujo** y **Editorial**) utilizamos la clase **dropdown-toglle**. En lo que se refiere al estilo, utilizamos un estilo propio, más allá del que nos provee el framework. Este estilo lo guardamos en la carpeta **css**, siendo este el código:

```
* HEADER */
#main-header{
        position: fixed;
        top:0;
        width: 100%;

        background-color: rgb(255, 255, 255);
        background-color: rgba(255, 255, 255, 0.95);
        -moz-box-shadow: 0 1px 5px #999;
        -webkit-box-shadow: 0 1px 5px #999;
        box-shadow: 0 1px 5px #999;
}
#main-menu{
        float: right;
        margin-top: 30px;
}

#main-menu a{
        color:#aaa;
}

#main-menu a:hover{
        color: #0071da;
}

#main-menu.nav-pills>li.active>a,
#main-menu.nav-pills>li.active>a:hover,
#main-menu.nav-pills>li.active>a:focus,
#main-menu.nav-pills>li.active>a:hover{
        background: none;
}

#main-menu.nav-pills>li.active>a,
#main-menu.nav-pills>li.active>a:hover,
#main-menu.nav-pills>li.active>a:focus{
        color: #0071da;
}
```

```
#main-menu.nav .open>a,
#main-menu.nav .open>a:hover,
#main-menu.nav .open>a:focus{
        background: none;
}
```

Analizando el código anterior, vemos que lo primero que hacemos es darle una posición fija (**position: fixed;**) que no tenga espacio entre el margen superior y nuestro cabezal con **top: 0;**. También le damos un ancho que ocupe todo el espacio de la grilla que diagramamos con sus 12 columnas en total, divididas en tres para el logo y nueve para el menú **width: 100%;**. Con esto estaría listo el diseño del cabezal. Solo nos falta agregarle el color: utilizamos la propiedad **background-color** y, mediante la propiedad **box-shadow**, le damos un poco de sombra, tal como está en el diseño.

Con respecto al menú, con la propiedad **float** y dándole por valor **right** haremos que nuestro ** ** contenedor del menú se corra lo que más pueda hacia la derecha dentro de las nueve columnas que le establecimos por el sistema de Bootstrap. Por último, le asignamos un color a los enlaces **<a>** y otro color diferente para cuando el usuario pase el cursor por los mismos. Esto lo realizamos gracias al evento **hover**.

En lo que respecta al logo, solamente con el máximo de tres columnas que le dimos (**col-md-3**) quedará perfectamente ubicado a la izquierda de la pantalla.

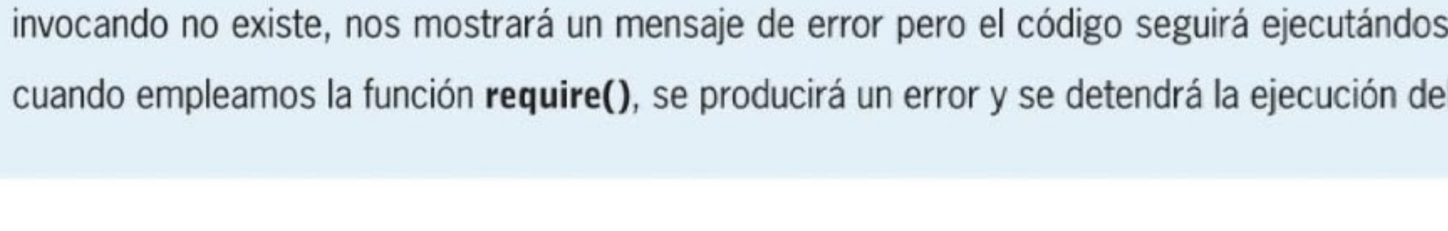


Figura 4. Menú de nuestro sitio web (portfolio) con varios enlaces a las distintas páginas que lo componen.

DIFERENCIA ENTRE INCLUDE() Y REQUIRE()

Si bien estas funciones de PHP nos permiten incluir código de otro archivo en un archivo tantas veces como queramos, la diferencia radica en que cuando usamos la función **include()**, si el archivo que estamos invocando no existe, nos mostrará un mensaje de error pero el código seguirá ejecutándose. En cambio, cuando empleamos la función **require()**, se producirá un error y se detendrá la ejecución del código.

Pie de página

Otra parte importante en nuestro proyecto y que se va a repetir en las diferentes secciones de nuestra web es el **pie de página** o *footer*. Para crear el pie de página, crearemos un archivo en la carpeta **inc** llamado **pie.php**. Este archivo es el que se va a incluir en las diferentes páginas que componen nuestra web. En él contamos nuevamente con una sección de enlaces a las diferentes partes de la página web, y con otra, en el sector derecho de la pantalla, con un e-mail de consulta y distintos iconos de redes sociales. Como el pie de página está dividido en dos secciones, dividimos las columnas de Bootstrap en dos, lo que nos da un total de seis columnas por lado (**col-md-6**). El código resultante es el siguiente:

```
<footer id="main-footer">
    <div class="container">
        <div class="row">
            <div class="col-md-6">
                <ul>
                    <li><a href="biografia.
php">BIOGRAFÍA</a></li>
                    <li>TRABAJOS</li>
                        <ul>
                            <li><a
href="fotografia.php">Fotografía</a></li>
                            <li><a
href="dibujo.php">Dibujo</a></li>
                            <li><a
href="editorial.php">Editorial</a></li>
                        </ul>
                    <li><a href="contacto.
php">CONTACTO</a></li>
                </ul>

                <p>©copyright 2015 - Derechos reservados</p>
            </div>
            <div class="col-md-6">
                <h3>Consultas: </h3>
```

```
                                        <a href="mailto:mail@miempresa.com.
ar?Subject=Consulta" target="_top">mail@miempresa.com.ar</a>
                                        <h3>Redes sociales</h3>
                                        <a href="#"><img src="img/social/facebook.
png" alt="" ></a>
                                        <a href="#"><img src="img/social/youtube.
png" alt="" ></a>
                                        <a href="#"><img src="img/social/twitter.png"
alt="" ></a>
                                        <a href="#"><img src="img/social/google.png"
alt="" ></a>
                            </div>
                    </div>
            </div>
    </footer>
```

Esta sección está establecida con las etiquetas HTML **<footer> </footer>**, para indicarle al navegador que lo que vamos a incluir dentro de esas dos etiquetas es el pie de página. Después, dentro de las dos secciones de seis columnas cada una, agregamos en la primera los enlaces a las distintas secciones del sitio que crearemos más adelante, y, en la otra, los enlaces e imágenes de las distintas redes sociales.

En lo que respecta al estilo, le aplicamos un color de fondo negro (**background-color: #000000;**) y, además, definimos varios colores para los enlaces y títulos, como también su ubicación.

```
#main-footer{
        margin: 30px 0;
        background-color: #000000;
}
#main-footer a, footer a:hover{
        color:#fff;
}

#main-footer p{
        color: #ddd;
}
```

```
#main-footer h3{
        margin-top: 30px;
        color:#fff;
        font-size: 16px;

}
#main-footer li{
        color: #fff;
}
```

Figura 5. Pie de página terminado con los iconos y colores que incluimos en el diseño.

Página principal

La página principal (en inglés, *home*) está compuesta por tres secciones, dos que ya tenemos terminadas (la cabecera y el pie de página), y una tercera sección que llamamos **detalle.php** y que incluimos en la carpeta **inc** de nuestro proyecto. De esta forma, nuestro archivo principal **index.php** en el cuerpo (*body*) del HTML quedaría formado de la siguiente forma:

PRIMERA RED SOCIAL

La primera red social fue **Six Degrees**. Fue creada en el año 1997 por el abogado Andrew Winreich, en la ciudad de Nueva York. Esta red social se basaba en la teoría de que una persona puede llegar a conocer a cualquier otra persona del mundo mediante la conexión de seis conocidos. Six Degrees estuvo activa hasta 2001 y llegó a tener alrededor de un millón de miembros registrados.

```
<body>
<?php
                include 'inc/cabezal.php';
                include 'inc/detalle.php';
                include 'inc/pie.php'
      ?>
</body>
```

Como dijimos anteriormente, los dos archivos **cabezal.php** y **pie.php** se van a repetir por cada página de nuestra web. Cada vez que los necesitemos simplemente tenemos que incluirlos como hicimos en el código del ejemplo anterior, mediante la función de PHP **include()**.

El archivo **detalle.php** estará compuesto por el contenido principal de nuestra página, que, como podemos observar claramente en el diseño, está compuesto por dos filas. Una de ellas se extiende a lo ancho, por lo que podemos darle directamente el total de 12 columnas de Bootstrap (**col-md-12**), y la otra fila tiene tres imágenes, por lo que podemos dividir las columnas en tres de cuatro (**col-md-4**).

```
<section>
<article id="contenido1">
      <div class="container">
            <div class="row">
            <div class="col-md-12">
                  <div class="jumbotron">
                   <h1>Bienvenidos a mi sitio!</h1>
                   <p>Lorem ipsum dolor sit amet, consectetur adipiscing
elit. Vivamus ligula diam, pretium ac nulla eget, egestas egestas ipsum. Vestibulum in
orci metus.  </p>
                   <p><a class="btn btn-success btn-lg" href="#"
role="button">Leer más...</a></p>
                  </div>
                  </div>
            </div>
      </div>
</article>

<article id="contenido2">
```

```
        <div class="container">
                <div class="page-header">
                        <h1>Mis mejores trabajos <small>Estos son sólo algunos
ejemplos</small></h1>
                </div>
                <div class="row">
                  <div class="col-md-4">
                    <div class="thumbnail">
                      <img src="img/principal/guitar.jpg" >
                      <div class="caption">
                        <h3>Fotografía</h3>
                        <p>Lorem Ipsum es simplemente el texto de relleno de las
imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno estándar de
las industrias desde el año 1500, cuando un impresor</p>
                        <p><a href="#" class="btn btn-primary" role="button">ver
detalles</a> </p>
                      </div>
                    </div>
                  </div>
                  <div class="col-md-4">
                    <div class="thumbnail">
                      <img src="img/principal/dibujo.jpg" >
                      <div class="caption">
                        <h3>Dibujo</h3>
                        <p>Lorem Ipsum es simplemente el texto de relleno de las
imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno estándar de
las industrias desde el año 1500, cuando un impresor</p>
                        <p><a href="#" class="btn btn-primary" role="button">ver
detalles</a> </p>
                      </div>
                    </div>
                  </div>
                  <div class="col-md-4">
                    <div class="thumbnail">
                      <img src="img/principal/editorial.jpg" >
                      <div class="caption">
                        <h3>Editorial</h3>
```

```
                    <p>Lorem Ipsum es simplemente el texto de relleno de las
imprentas y archivos de texto. Lorem Ipsum ha sido el texto de relleno estándar de
las industrias desde el año 1500, cuando un impresor</p>
                    <p><a href="#" class="btn btn-primary" role="button">ver
detalles</a> </p>
                </div>
              </div>
            </div>
          </div>
        </article>
        </section>
```

Utilizamos las etiquetas HTML **<section></section>** y dentro de estas, **<article></article>** para que el navegador reconozca que lo que va a leer es una sección y que dentro de ella puede contener uno o varios artículos. En este caso, como son dos filas que creamos anteriormente, vamos a utilizar dos artículos. Para la primera fila, utilizamos la clase **jumbotron** que nos va a dar el recuadro de color gris y en su interior el título con las etiquetas **<h1></h1>**, un párrafo y un botón de color verde. La segunda fila está compuesta por un título y tres fotos que están ubicadas en tres columnas mediante la clase **col-md-4**.

MEDIANTE LA CLASE MARGIN-TOP LE AGREGAMOS UN MARGEN A LAS FILAS Y LAS CENTRAMOS

En cuanto al estilo de estas dos filas, simplemente le damos un **margin-top**, para darle un margen y centrarlo dentro de la página:

```
#contenido1 {
        margin-top: 150px;
        /*background-image: url('../img/video.png'); */

}

#contenido2 {
        margin-top: 50px;
}
```

Figura 6. Página principal donde podemos observar cómo queda el archivo **detalle.php**.

Biografía

La página **Biografía** es una de las más simples: como podemos observar en el diseño, solamente posee una fotografía a la izquierda de la pantalla y un texto al costado izquierdo con un título. Al haber dos elementos en la página, solo tenemos que dividir las 12 columnas por 2. Como el mayor tamaño es el del título y el texto que le sigue a continuación, vamos a darle a esta parte un tamaño de 10 (**col-md-10**), y a la fotografía, como tiene un menor ancho, solamente le vamos a dar dos columnas (**col-md-2**). Estos valores, sumados, nos darán un total de 12 columnas, tal cual lo requiere el framework. El código es el siguiente:

```
<section>
        <article id="contenidoBio">
                <div class="container">

                        <div class="row">
                                <div class="col-md-2">
                                        <img src="img/biografia/foto.png"
class="img-rounded" >
                                </div>
```

```
                                        <div class="col-md-10">
                                            <div class="page-header">

                                                <h1>Todo sobre mí
<small>(Biografía)</small></h1>
                                            </div>
                                            <p>Maecenas sit amet interdum
turpis, et tristique ex. In at libero in felis posuere aliquam vitae eget ipsum. Nam
vel justo vitae lorem luctus scelerisque. Nulla sit amet augue ac dui consecte-
tur porta eget ac nunc. Vivamus quis tortor leo. In ultricies pulvinar metus sed
varius. Vivamus a rutrum nisi. Morbi consequat, mauris nec efficitur hendrerit,
orci urna iaculis risus, ac sodales est nisi et augue. Donec dapibus neque nulla, ut
lobortis lorem placerat dapibus. Sed metus libero, pulvinar et cursus vel, ultrices
nec quam. Ut luctus id elit pellentesque malesuada. Aliquam eget mi fermentum,
interdum odio et, faucibus purus. Cras in ex elit. Phasellus posuere ante convallis
tincidunt sodales. <br>
                                                Maecenas sit amet interdum turpis,
et tristique ex. In at libero in felis posuere aliquam vitae eget ipsum. Nam vel
justo vitae lorem luctus scelerisque. Nulla sit amet augue ac dui consectetur
porta eget ac nunc. Vivamus quis tortor leo. In ultricies pulvinar metus sed
varius. Vivamus a rutrum nisi. Morbi consequat, mauris nec efficitur hendrerit,
orci urna iaculis risus, ac sodales est nisi et augue. Donec dapibus neque nulla, ut
lobortis lorem placerat dapibus. Sed metus libero, pulvinar et cursus vel, ultrices
nec quam. Ut luctus id elit pellentesque malesuada. Aliquam eget mi fermentum,
interdum odio et, faucibus purus. Cras in ex elit. Phasellus posuere ante convallis
tincidunt sodales. <br>
                                                Maecenas sit amet interdum turpis,
et tristique ex. In at libero in felis posuere aliquam vitae eget ipsum. Nam vel
justo vitae lorem luctus scelerisque. Nulla sit amet augue ac dui consectetur
```

PROBAR LOS ARCHIVOS CON XAMPP

XAMM nos permite probar nuestros archivos PHP y Bootstrap sin necesidad de contratar un hosting. Una vez que lo hemos instalado, tenemos que guardar nuestros archivos dentro de la carpeta **c:\xampp\htdocs**. Luego, para visualizarlo, escribimos en la barra de dirección de nuestro navegador la siguiente dirección: **http://localhost/index.php**

```
porta eget ac nunc. Vivamus quis tortor leo. In ultricies pulvinar metus sed
varius. Vivamus a rutrum nisi. Morbi consequat, mauris nec efficitur hendrerit,
orci urna iaculis risus, ac sodales est nisi et augue. Donec dapibus neque nulla, ut
lobortis lorem placerat dapibus. Sed metus libero, pulvinar et cursus vel, ultrices
nec quam. Ut luctus id elit pellentesque malesuada. Aliquam eget mi fermentum,
interdum odio et, faucibus purus. Cras in ex elit. Phasellus posuere ante convallis
tincidunt sodales. <br>
                                        Maecenas sit amet interdum turpis,
et tristique ex. In at libero in felis posuere aliquam vitae eget ipsum. Nam vel
justo vitae lorem luctus scelerisque. Nulla sit amet augue ac dui consectetur
porta eget ac nunc. Vivamus quis tortor leo. In ultricies pulvinar metus sed
varius. Vivamus a rutrum nisi. Morbi consequat, mauris nec efficitur hendrerit,
orci urna iaculis risus, ac sodales est nisi et augue. Donec dapibus neque nulla, ut
lobortis lorem placerat dapibus. Sed metus libero, pulvinar et cursus vel, ultrices
nec quam. Ut luctus id elit pellentesque malesuada. Aliquam eget mi fermentum,
interdum odio et, faucibus purus. Cras in ex elit. Phasellus posuere ante convallis
tincidunt sodales. </p>
                                    </div>
                                </div>
                            </div>
                        </article>
                    </section>
```

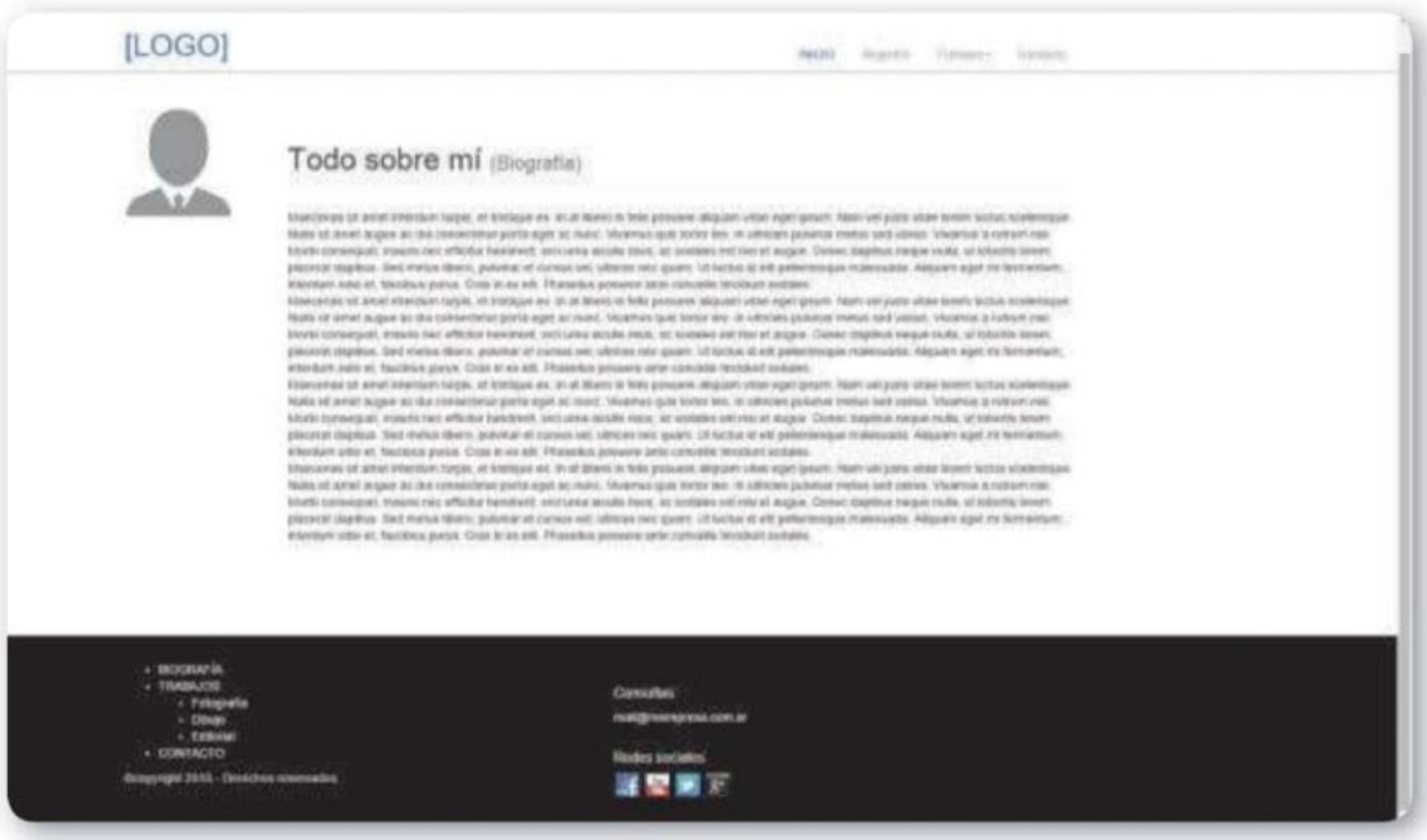


Figura 7. Página **Biografía**, que se encuentra ubicada en la carpeta **inc** (**`detalle_biografia-php`**).

Trabajos

La página **Trabajos** contiene tres enlaces que son Fotografía, Dibujo y Editorial, es decir que vamos a tener que hacer una página por cada enlace.

Las páginas de **Fotografía** y **Editorial** son similares. La primera contiene tres fotos y un enlace para ver el contenido de la categoría. Y la página **Editorial** contiene seis fotos ubicadas en dos filas. Para la página de **Dibujo** utilizamos una lista con la clase **list-group-item** junto con la clase **badge**. La idea es que al hacer clic sobre un elemento de la lista, se abra otra página mostrando todos los trabajos de esa categoría.

Figura 8.
Tres páginas que componen el enlace Trabajos: **detalle_dibujo.php**, **detalle_fotografia.php** y **detalle_editorial.php**.

El código de las estas tres páginas es el siguiente:

Fotografía (**detalle_fotografia.php**):

```
<section>
    <article id="contenidoFoto">
        <div class="container">

            <div class="row">
              <div class="col-md-4">
                <div class="thumbnail">
```

```
                    <img src="img/trabajos/guitar.jpg" alt="...">
                    <div class="caption">
                      <h3>Guitarras</h3>
                      <p>Lorem ipsum dolor sit amet, consectetur adipi-
scing elit. Vivamus ligula diam, pretium ac nulla eget, egestas egestas ipsum. Ves-
tibulum in orci metus. Vivamus tincidunt leo in leo consequat vehicula. Aenean justo
metus, facilisis eget finibus at, pellentesque in lectus.</p>
                        <p><a href="#" class="btn btn-primary"
role="button">ver trabajos</a></p>
                      </div>
                    </div>
                  </div>
                  <div class="col-md-4">
                    <div class="thumbnail">
                      <img src="img/trabajos/calle.jpg" alt="...">
                      <div class="caption">
                        <h3>Ciudad</h3>
                        <p>Lorem ipsum dolor sit amet, consectetur adipi-
scing elit. Vivamus ligula diam, pretium ac nulla eget, egestas egestas ipsum. Ves-
tibulum in orci metus. Vivamus tincidunt leo in leo consequat vehicula. Aenean justo
metus, facilisis eget finibus at, pellentesque in lectus.</p>
                        <p><a href="#" class="btn btn-primary"
role="button">ver trabajos</a></p>
                      </div>
                    </div>
                  </div>
                  <div class="col-md-4">
                    <div class="thumbnail">
                      <img src="img/trabajos/iglesia.jpg" alt="...">
                      <div class="caption">
```

INCLUDE_ONCE() Y REQUIRE_ONCE()

Las funciones **include_once()** y **require_once()** permiten incluir un archivo solo una vez, es decir, evalúan si el archivo ya fue incluido y si es así no lo vuelven a incluir, a diferencia de las funciones **include()** y **require()** que nos permiten incluir el archivo tantas veces como queramos.

```
                    <h3>Iglesias</h3>
                    <p>Lorem ipsum dolor sit amet, consectetur adipi-
scing elit. Vivamus ligula diam, pretium ac nulla eget, egestas egestas ipsum. Ves-
tibulum in orci metus. Vivamus tincidunt leo in leo consequat vehicula. Aenean justo
metus, facilisis eget finibus at, pellentesque in lectus.</p>
                    <p><a href="#" class="btn btn-primary"
role="button">ver trabajos</a></p>
                  </div>
                </div>
              </div>
        </div>
            </div>
        </article>
    </section>
```

Dibujo (**detalle_dibujo.php**):

```
    <section>
        <article id="contenidoDibujo">
            <div class="container">

                <div class="row">
                    <div class="col-md-12">
                    <div class="page-header">

                        <h1>Estos son mis dibujos
<small>(Dibujos)</small></h1>
                    </div>
                    <div class="list-group">
                        <a href="#" class="list-group-item
active">

                            <h4 class="list-group-item-
heading">Retratos</h4>
                            <span class="badge">15</span>
                            <p class="list-group-item-
text">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus ligula diam,
pretium ac nulla eget, egestas egestas ipsum. Vestibulum in orci metus. Vivamus
```

```
tincidunt leo in leo consequat vehicula. Aenean justo metus, facilisis eget finibus at,
pellentesque in lectus.</p>
                                                    </a>

                                                    <a href="#" class="list-group-
item">
                                                        <h4 class="list-group-item-
heading">Paisajes</h4>
                                                        <span class="badge">38</span>
                                                        <p class="list-group-item-
text">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus ligula diam,
pretium ac nulla eget, egestas egestas ipsum. Vestibulum in orci metus. Vivamus
tincidunt leo in leo consequat vehicula. Aenean justo metus, facilisis eget finibus at,
pellentesque in lectus.</p>
                                                    </a>
                                                    </a>
                                                    <a href="#" class="list-group-
item">
                                                        <h4 class="list-group-item-
heading">Animales</h4>
                                                        <span class="badge">9</span>
                                                        <p class="list-group-item-
text">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus ligula diam,
pretium ac nulla eget, egestas egestas ipsum. Vestibulum in orci metus. Vivamus
tincidunt leo in leo consequat vehicula. Aenean justo metus, facilisis eget finibus at,
pellentesque in lectus.</p>
                                                    </a>
                                                    </a>
                                                    <a href="#" class="list-group-
```

FAVICON

Un **favicon** consiste en un pequeño icono que puede ser realizado por cualquier programa de diseño, como por ejemplo Photoshop o Illustrator. Por lo general tienen un tamaño de 16x16, 24x24, 32x32, 64x64 pixeles. Esta imagen se visualiza en la barra de dirección antes de la URL, en la lista de favoritos o en las pestañas de los navegadores.

```
item">
                                                    <h4 class="list-group-item-
heading">Objetos</h4>
                                                    <span class="badge">16</span>
                                                    <p class="list-group-item-
text">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus ligula diam,
pretium ac nulla eget, egestas egestas ipsum. Vestibulum in orci metus. Vivamus
tincidunt leo in leo consequat vehicula. Aenean justo metus, facilisis eget finibus at,
pellentesque in lectus.</p>
                                                </a>
                                                </a>
                                                <a href="#" class="list-group-
item">
                                                    <h4 class="list-group-item-
heading">Artísticos</h4>
                                                    <span class="badge">62</span>
                                                    <p class="list-group-item-
text">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus ligula diam,
pretium ac nulla eget, egestas egestas ipsum. Vestibulum in orci metus. Vivamus
tincidunt leo in leo consequat vehicula. Aenean justo metus, facilisis eget finibus at,
pellentesque in lectus.</p>
                                                </a>
                                                </a>
                                            </div>
                                        </div>


                                    </div>
                                </div>
                            </article>
                        </section>
```

Editorial (**detalle_editorial.php**):

```
<section>
    <article id="contenidoEditorial">
        <div class="container">
```

```
<div class="row">
 <div class="col-md-4">
  <div class="thumbnail">
   <img src="img/editorial/editorial1.jpg" alt="...">
   <div class="caption">
    <h3>Trabajo 1</h3>
    <p>Lorem ipsum dolor sit amet, consectetur adipi-
scing elit. </p>

   </div>
  </div>
 </div>
 <div class="col-md-4">
  <div class="thumbnail">
   <img src="img/editorial/editorial2.jpg" alt="...">
   <div class="caption">
    <h3>Trabajo 2</h3>
    <p>Lorem ipsum dolor sit amet, consectetur adipi-
scing elit. </p>

   </div>
  </div>
 </div>
 <div class="col-md-4">
  <div class="thumbnail">
   <img src="img/editorial/editorial3.jpg" alt="...">
   <div class="caption">
    <h3>Trabajo 3</h3>
    <p>Lorem ipsum dolor sit amet, consectetur adipi-
scing elit. </p>

   </div>
  </div>
 </div>
</div>
<div class="row">
 <div class="col-md-4">
  <div class="thumbnail">
```

```
                        <img src="img/editorial/editorial4.jpg" alt="...">
                        <div class="caption">
                          <h3>Trabajo 4</h3>
                          <p>Lorem ipsum dolor sit amet, consectetur adipi-
scing elit. </p>

                        </div>
                      </div>
                    </div>
                    <div class="col-md-4">
                      <div class="thumbnail">
                        <img src="img/editorial/editorial5.jpg" alt="...">
                        <div class="caption">
                          <h3>Trabajo 5</h3>
                          <p>Lorem ipsum dolor sit amet, consectetur adipi-
scing elit. </p>

                        </div>
                      </div>
                    </div>
                    <div class="col-md-4">
                      <div class="thumbnail">
                        <img src="img/editorial/editorial6.jpg" alt="...">
                        <div class="caption">
                          <h3>Trabajo 6</h3>
                          <p>Lorem ipsum dolor sit amet, consectetur adipi-
scing elit. </p>

                        </div>
                      </div>
                    </div>
        </div>
                  </div>
        </article>
    </section>
```

Contacto

La página de contacto está compuesta por un panel y en su interior un formulario con tres campos para solicitar información.

El código es el siguiente:

```
<section>
  <article id="contenidoContacto">
    <div class="container">

      <div class="row">
        <div class="col-md-12">
          <div class="panel panel-primary">
            <div class="panel-heading">
              <h3 class="panel-title">Formulario de consulta</h3>
            </div>
            <div class="panel-body">
             <form>
                  <div class="form-group">
                   <label for="mail">Email</label>
                   <input type="email" class="form-control" id="mail"
placeholder="Ingrese su Email">
                  </div>
                  <div class="form-group">
                   <label for="asunto">Asunto</label>
                   <input type="text" class="form-control" id="asunto"
placeholder="Ingrese el asunto">
                  </div>
                  <div class="form-group">
                  <label for="consulta">Consulta</label>
                  <textarea class="form-control" id="consulta" rows="3"
placeholder="Ingrese su consulta"></textarea>
                  </div>

                  <button type="submit" class="btn btn-success">Enviar consulta</
button>
                </form>
            </div>
```

```
        </div>
      </div>
    </div>
  </div>
 </article>
</section>
```

En relación al estilo, esta página solamente tiene un margen para centrar su contenido en la pantalla.

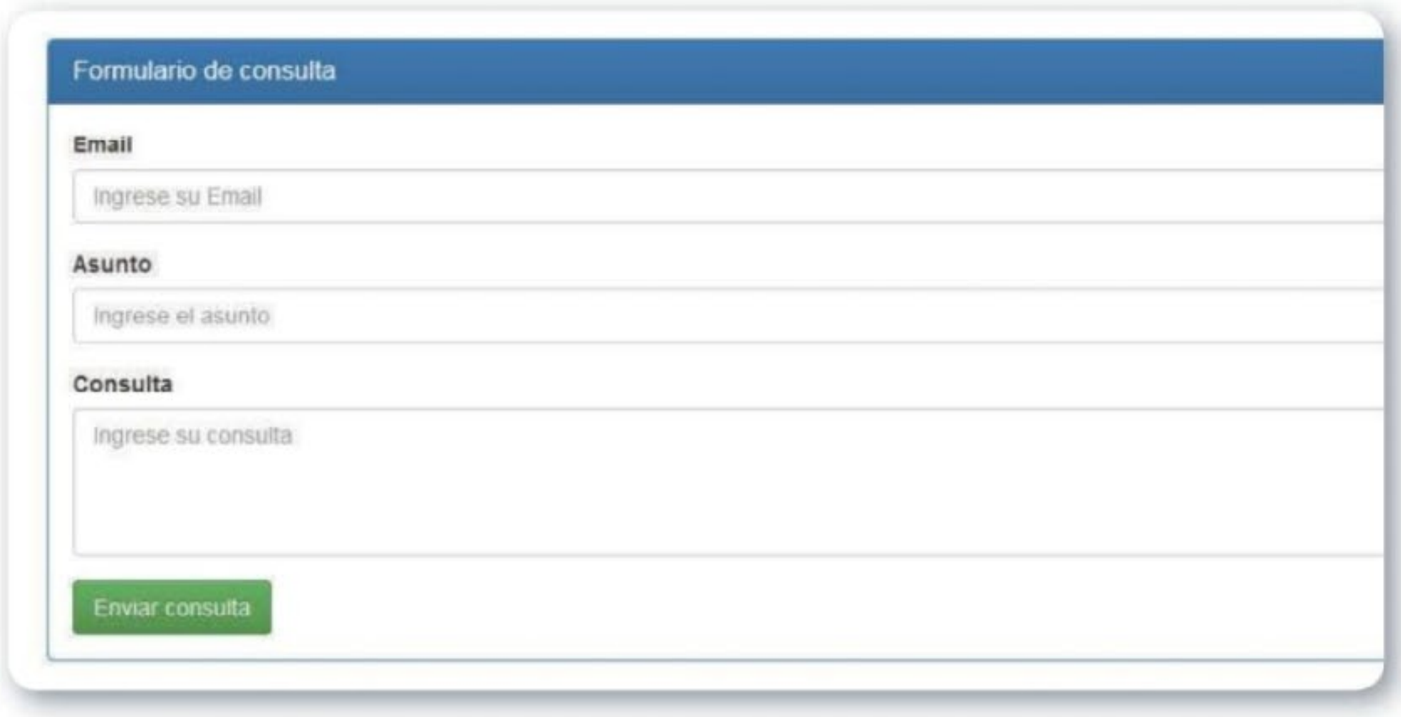


Figura 9. Página de contacto ubicada en la carpeta **inc**, llamada **`detalle_contacto.php`**.

RESUMEN

En este capítulo, mediante un sencillo ejemplo, vimos cómo diseñar y armar un portfolio partiendo de un diseño. En base a lo desarrollado en capítulos anteriores, armamos las diferentes páginas que componen el sitio, aplicando los conceptos de grillas de Bootstrap; empleamos también diferentes componentes gráficos, tales como botones e imágenes; y utilizamos, además, los componentes de menú y paneles. Finalmente creamos una página de contacto empleando un formulario de Bootstrap.

Actividades

TEST DE AUTOEVALUACIÓN

1. ¿Qué es un portfolio?
2. ¿Cuál es la ventaja de usar el menú principal?
3. ¿Por qué armamos una estructura de carpetas?
4. ¿Por qué optamos por utilizar la clase **.col-md** en la página principal?
5. ¿Cuál es la ventaja de utilizar la función **include()**?

EJERCICIOS PRÁCTICOS

1. Inicie el servidor web XAMPP. En el directorio **c:\xampp\htdocs** cree una carpeta llamada **portfolio** y dentro de ella las subcarpetas **css**, **img** e **inc**.
2. Tomando como base el código explicado en este capítulo, genere los archivos **index.php**, **cabezal.php**, **detalle.php** y **pie.php.**
3. En base al estilo explicado, modifique el color de fondo variando los valores RGB y RGBA.
4. Escriba el código para crear los archivos **biografia.php** y **trabajos.php**. A su vez, genere los archivos **fotografía.php**, **dibujos.php** y **editorial.php**.
5. Genere un formulario de contactos.
6. Visualice el sitio a través del servidor local.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

*08

Ejemplo práctico con Master Bootstrap

En este capítulo, explicamos cómo funciona el gestor de contenidos Joomla! con Bootstrap y, mediante un ejemplo, vemos cómo instalar la última versión y configurar su idioma. Luego seguimos algunos pasos para instalar la plantilla Master Bootstrap y comenzar a configurar nuestro proyecto. Aprendemos cómo crear el menú principal, el pie de página y el contenido principal de nuestro sitio web.

Bootstrap y CMS

Desde el lanzamiento de Bootstrap, casi todos los gestores de contenido o **CMS** (que veremos en detalle en el **Capítulo OnWeb**) se han adaptado y permiten utilizar este framework. Esto les brinda a los desarrolladores herramientas para poder utilizar y crear el sistema de rejillas del CMS, haciendo de esta forma que el sitio web se adapte a todos los dispositivos, utilizando sus clases, componentes y estilos.

Como la mayoría de los gestores de contenido, **Joomla!** se vio afectado con el lanzamiento de Bootstrap. Es por eso que desde la versión **2.x** nos permite incorporar el framework y, actualmente, con la última versión, **3.x**, lo hace de manera óptima, porque directamente lo incorpora en el núcleo, creando la librería **JUI** (*Joomla User Interface*; en español, 'interfaz para usuarios Joomla').

Instalación de Joomla!

Para comenzar a trabajar en este nuevo proyecto web, lo primero que necesitamos es descargar del sitio oficial de Joomla! su última versión disponible (al momento de realizar este libro, la **3.4.3**). Para descargarlo, ingresamos a: **www.joomla.org/download.html**

También, desde la misma web, vamos a necesitar instalar el idioma, que por defecto es el inglés. Para esto, tenemos que ir al menú principal y hacer clic en **Extend** y luego seleccionar la opción **Translation**. Dentro de esa página hacemos clic en **Joomla! 3.x Translation Packs** y luego buscamos el idioma que deseamos instalar; en nuestro caso, el español. Al hacer clic sobre **Spanish**, nos lleva a la página del proyecto, desde donde vamos a poder descargar la traducción.

Una vez que descargamos Joomla!, descomprimimos el archivo y lo ubicamos en el directorio local de nuestra computadora, donde tenemos instalado **XAMPP**. Vamos a denominar **ejemplo2** a la carpeta con el contenido de Joomla!. De esta forma, nos queda un directorio parecido o igual al siguiente: **C:\xampp\htdocs\ejemplo2**

Para ejecutar el programa, debemos escribir en nuestro navegador la URL del sitio; en este ejemplo: **http://localhost/ejemplo2**

Una vez que ingresamos a esa dirección, seguimos los pasos que nos indica el asistente y, al finalizar, tendremos instalado Joomla! 3.

Figura 1. Al comenzar la instalación, veremos una página como la que se puede apreciar aquí.

En la primera pantalla de la instalación, debemos configurar el nombre del sitio (en nuestro caso, **Ejemplo2**), una descripción general para los motores de búsqueda, el e-mail del administrador y una contraseña que siempre debemos recordar para administrar la web.

El segundo paso, de gran importancia, es la configuración de la base de datos. Seleccionamos primero qué tipo de base de datos vamos a utilizar (en nuestro caso, **MySQL**) y después nos pedirá su hospedaje: puede ser la dirección **IP**, o bien, como la base se encuentra en el mismo servidor, **LocalHost**. Luego debemos colocar nuestro usuario y contraseña. También completamos el nombre de la base de datos: en nuestro caso, **base_joomla**. Por último, el prefijo que tendrán las tablas: **amxuf_**.

Finalmente, en el paso 3 (finalización) de la instalación de Joomla! se nos pregunta si queremos cargar datos de ejemplo predeterminado o simplemente queremos la instalación vacía sin ningún dato. En nuestro caso, la vamos a dejar vacía; por lo tanto, hacemos clic en el ítem **Ninguno**.

Para finalizar, antes de comenzar la instalación, Joomla! nos mostrará un resumen de toda la configuración general de nuestra web. Chequeamos que esté todo correcto y hacemos clic en el botón **Instalar** que se encuentra situado arriba a la derecha de la pantalla.

DIRECCIÓN IP

La dirección **IP** (**IP address**) es un número único e irrepetible que se utiliza para identificar a una computadora que se encuentra conectada a una red de computadoras que utilizan el protocolo IP (*Internet Protocol*). Esta dirección consiste en un conjunto de cuatro números del 0 al 255, separados por puntos, como por ejemplo 200.32.124.130.

Antes de finalizar la instalación, y si tenemos conexión a Internet, podemos proceder a instalar el idioma para nuestro sitio (en nuestro caso, el español). Presionamos el botón **Paso extra: Instalar idiomas**, luego, en la ventana que se abre, seleccionamos la opción **Spanish (español)** y el botón **Siguiente**. En la ventana que se abre, donde aparecen las opciones para la administración y el idioma predeterminado para el sitio, seleccionamos nuevamente **Spanish (español)**.

Figura 2. Si la instalación se ha realizado correctamente, veremos esta pantalla.

En el caso de no poseer conexión a Internet, podemos configurar el idioma desde la administración del CMS, usando el archivo que bajamos del sitio oficial de Joomla! al comienzo de la instalación.

Finalmente, debemos borrar la carpeta de instalación: podemos hacerlo manualmente, en el directorio principal de nuestro sitio web, o bien haciendo clic en el botón **Eliminar carpeta de instalación**.

El administrador de Joomla!

El administrador de nuestro sitio web creado en Joomla! es la herramienta más importante, ya que nos permitirá configurar el sitio y agregarle contenido. Para acceder a él, debemos añadir a la dirección URL de nuestra web el directorio para referenciarlo (**administrator**). En nuestro ejemplo: **http://localhost/ejemplo2/administrator**

Al acceder vamos a encontrar una pantalla de logueo, en donde tendremos que completar nuestro usuario y contraseña de administradores, que son los que configuramos al instalar el sitio web.

Figura 3. Pantalla principal del administrador de Joomla!, donde podemos configurar y darle forma a nuestro sitio web.

En el administrador tenemos dividido todo lo que se puede hacer con él, por categoría. En la sección **Usuarios** podemos crear, eliminar y modificar usuarios, así como administrar sus permisos. Para agregar artículos, categorías y administrar archivos multimedia, tenemos la sección de **Contenido**. Por otro lado, en **Estructura** podemos gestionar y crear los menús, así como agregar módulos a nuestro sitio. Desde la sección **Configuración** podemos acceder a la configuración global del sitio, al gestor de plantillas, desde donde podemos agregarlas o modificarlas, y, por último, el gestor de idiomas, que nos permite cambiar el idioma predeterminado o sumar alguno nuevo. Si queremos instalar extensiones, propias o de terceros, debemos dirigirnos a **Extensiones**; y, por último, la sección **Mantenimiento** nos avisa si hay alguna actualización disponible de la versión de Joomla! que instalamos y nos permite vaciar la caché del sitio.

Instalación de Master Bootstrap

Master Bootstrap es una plantilla creada para Joomla! (versión 3 o superior) por **Gonzalo Suez**, de la República de Chile, quien gentilmente nos permitió incorporar su desarrollo en este libro. Es una excelente plantilla que se adapta a cualquier sitio que queramos realizar, sin importar la ubicación de los distintos componentes y haciendo muy adaptable el diseño, tanto para la realización del proyecto como para la aplicación terminada. En breve saldrán nuevas actualizaciones que sumarán novedosas opciones y una versión especial con varios estilos.

Para instalar esta plantilla, debemos ingresar al sitio web oficial del proyecto: **www.masterbootstrap.com/index.php/es**. Allí, hacemos clic en el botón **Descargar desde Github**. Una vez que accedimos a la página web de GitHub donde se aloja (**https://github.com/gsuez/master-bootstrap-3**), hacemos clic en el botón de la derecha, **Download ZIP**.

Figura 4. Sitio oficial de **Master Bootstrap**, una plantilla realizada para Joomla! con todas las ventajas de Bootstrap.

Para instalarlo en nuestra web de Joomla!, desde el administrador tenemos que ir al menú **Extensiones/Gestor de extensiones**. En la ficha **Subir paquete**, presionando el botón **Seleccionar el archivo**, buscamos el archivo que descargamos de la web de GitHub (**master-bootstrap-3-master.zip**). Finalmente, hacemos clic en el botón **Subir e instalar**. Con este procedimiento, ya tendremos instalado Master Bootstrap en nuestro sitio.

Lo que nos resta es habilitarlo para que se vea en nuestro sitio web y podamos configurarlo. Vamos al menú **Extensiones** y hacemos clic en **Gestor de plantillas**; una vez allí, buscamos la plantilla Master Bootstrap y, en la columna **Predeterminado**, hacemos clic en la estrella de color negro, que deberá ponerse amarilla.

Instalados Joomla! y la plantilla de Master Bootstrap, ya podemos comenzar a trabajar en nuestro proyecto.

Figura 5. En el **gestor de extensiones** del administrador de Joomla! podemos instalar el archivo comprimido de Master Bootstrap que descargamos.

Configuración del proyecto

El ejemplo práctico se basará en un blog sencillo dedicado a la programación, en el que nuestro cliente escribe artículos sobre diferentes lenguajes de programación. Nos basamos en el sitio web **www.murcielagoblanco.com.ar**, de nuestra autoría, que fue realizado completamente con Joomla! y Master Bootstrap, modificando algún estilo, más allá del que utiliza el framework, y siguiendo la documentación y ejemplos de Master Bootstrap del autor Gonzalo Suez.

Figura 6. Página web realizada en Joomla! con Master Bootstrap, dedicada a la programación en general.

Como podemos observar en la **Figura 6**, el sitio web cuenta con: un menú principal, donde se encuentra el logo; una sección que ocupa todo el ancho de la pantalla, con una imagen; unos cinco recuadros con información; otra sección con imágenes, con logos de diferentes lenguajes de programación; una publicidad, que, al hacer clic, nos lleva a otra página web; una sección con varias imágenes de iconos de redes sociales; y un pie de página. De esta forma, tenemos bien definido un total de **siete secciones** en la página principal (o *home*).

Armar una hoja de estilo propia

Como mencionamos anteriormente, vamos a necesitar armar una hoja de estilo propia para poder manejar algunos elementos de nuestra página web. Para esto, tenemos dos opciones: o bien creamos una

hoja de estilo nueva y la llamamos desde la página principal, o bien utilizamos alguna de las ya creadas por la plantilla. En este ejemplo, al no ser mucho lo que vamos a agregar, utilizaremos alguna hoja de estilo de Master Bootstrap.

Para modificar alguno de los estilos ya creados, contamos con dos posibilidades. Una es ir al directorio principal de nuestro sitio web, buscar la carpeta **templates** y, después, la carpeta con la plantilla que estemos utilizando (**masterbootstrap**, en nuestro ejemplo).

De esta forma, la ruta sería la siguiente: **C:\xampp\htdocs\ejemplo2\templates\masterbootstrap\css**. En la carpeta **css** nos encontramos con todos los estilos que utiliza la plantilla de Master Bootstrap. Más adelante, de acuerdo a nuestra necesidad, veremos cuál de todos ellos vamos a modificar.

La otra posibilidad es ir al administrador de Joomla! y, en el menú **Extensiones**, ingresar a **Gestor de plantillas**. Seleccionamos la plantilla que queremos modificar (en nuestro caso, **MasterBootstrap**) y hacemos clic en la columna **Plantilla** de la fila **MasterBootstrap**. Esto nos va a llevar al editor; allí, en la columna izquierda, contamos con todos los archivos que componen la plantilla. Seleccionamos la carpeta **CSS** y, a partir de ahí, cualquiera de los archivos, para luego modificarlos.

Cabecera de la página

Para configurar la cabecera de la página (*head*), tenemos que entrar directamente a la configuración de la plantilla de Master Bootstrap. Vamos al menú **Extensiones** dentro del administrador de Joomla!, y luego hacemos clic en **Gestor de Plantillas**. Ahí veremos que tenemos un listado con todas las plantillas disponibles e instaladas, y elegimos la nuestra, **Master Bootstrap – Predeterminado**. Al hacer clic sobre ella, se nos abrirá el editor, en donde tenemos tres solapas, **Detalles**, **Logo** y **Asignación a los menús**. Como lo primero que vamos a hacer es subir el logo de nuestra página web, entramos a la solapa **Logo**. Dentro de esta pestaña, contamos con la opción de seleccionar una imagen desde nuestro servidor —es decir, que ya hayamos subido—, o bien subir una nueva imagen desde el disco duro de nuestra PC.

También podemos configurar el ancho y largo de nuestra imagen (*width* y *height*, respectivamente). Como nuestro logo es muy chico, modificamos las medidas: 54 px de ancho por 46 px de largo.

El menú principal

El paso siguiente es configurar el menú principal, y con esto ya terminaríamos de armar la cabecera de nuestra página web. Tenemos que ubicar, dentro del administrador de Joomla!, el módulo correspondiente al menú principal; vamos al menú **Extensiones** y hacemos clic en la opción **Gestor de módulos**. Allí, buscamos el módulo **Main Menú** (en español, **menú principal**). Al hacer clic sobre esta opción, entramos a la configuración general del menú principal. Nos dirigimos a la primera solapa, **Módulo**, y allí, en la opción a la derecha (**Posición**), le asignamos la posición que va a tener nuestro menú principal: **Navigation** (en español, **navegación**). Esta posición —arriba, al comienzo de nuestra página web— ya está definida por el estilo que maneja la plantilla Master Bootstrap. Para ubicar el menú a la derecha, en la pestaña **Avanzado**, escribimos, dentro de la opción **Clase CSS**, lo siguiente:

```
nav menu navbar-nav navbar-right
```

Con estos pasos ya configuramos nuestro menú principal en la ubicación deseada: arriba de nuestra página web, junto al logo y a la derecha, como podemos observar en la **Figura 7**.

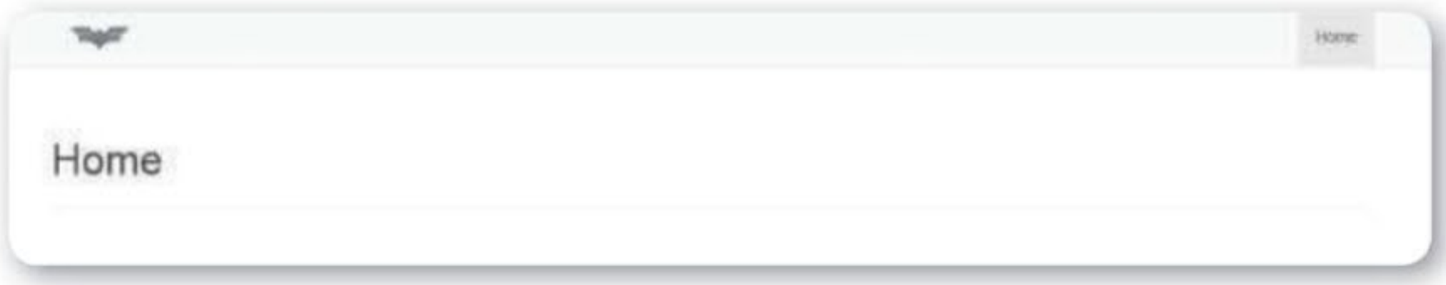


Figura 7. Menú y logo en la cabecera de nuestro sitio web, configurado con los estilos de Master Bootstrap.

QUITAR LA PALABRA "HOME" O "INICIO"

Para quitar la palabra **HOME** o **INICIO** que generalmente aparece en la página del comienzo de nuestro sitio web realizado en Joomla!, debemos ir al menú principal del administrador, al ítem **Menús**, y luego hacer clic en la opción **Gestor de menús**. Una vez ahí, hacemos clic sobre el ítem **Elementos del menú** y luego sobre la palabra **Home** o **Inicio**, dependiendo de cómo esté configurada nuestra página. Luego hacemos clic en la solapa **Visualización de la página** y marcamos con **No** la opción **Mostrar el encabezado de la página**.

Imagen principal

Para la imagen principal, debemos crear un módulo y configurarlo de acuerdo a las opciones que necesitemos. Para ello, entramos al menú principal del administrador de Joomla! y hacemos clic en **Extensiones**. Allí elegimos la opción **Gestor de módulos** y hacemos clic en el botón verde situado a la izquierda, **Nuevo**. Nos aparecerán varias opciones de módulos prediseñados, pero elegimos la opción **HTML personalizado**, para poder configurar uno de acuerdo a nuestra necesidad.

Sobre el editor, hacemos clic en el botón **Imagen** y seleccionamos alguna imagen para agregar al sitio. Puede ser una que ya tengamos cargada en el servidor o guardada en el disco duro de nuestra PC.

Sobre la derecha, hacemos clic en la opción **Ocultar** del menú **Mostrar el título**. En **Posición**, optamos por **fullwidth** porque lo que queremos es que nuestra imagen abarque el ancho total de la pantalla. Hacemos clic en **Guardar** y con esto ya tenemos configurada la imagen principal.

Módulos de servicios

Los módulos de servicios —como los llama el autor de la plantilla— nos servirán, en nuestro ejemplo, para dar una breve explicación de cada lenguaje de programación que forma parte del sitio web.

Vamos a necesitar cinco módulos, ya que, como vimos en la **Figura 6**, los necesitamos para Delphi, JS, VB, PHP y jQuery Mobile. Creamos un módulo nuevo **HTML personalizado**, le ponemos un título (en nuestro ejemplo, al primero lo llamamos **Delphi**) y, en **Posición**, selecionamos **content-bottom**. En el editor, escribimos la descripción del lenguaje de programación tal cual está en el ejemplo. En la pestaña **Avanzado** de **Clase CSS del módulo**, escribimos el siguiente código:

¿QUÉ ES UN MÓDULO EN JOOMLA!?

Los módulos son extensiones que nos permiten ampliar la funcionalidad de Joomla! y podemos agregar información para mostrarla en determinadas zonas de nuestra plantilla. Esta información se toma de una base de datos o de un formulario. Joomla! posee módulos preinstalados como, por ejemplo, el módulo de inicio de sesión (login), pero también podemos instalar módulos desarrollados por terceros.

```
col-md-2 height-bottom
```

Como son cinco cuadros, le daremos el espacio que necesitamos en las columnas de Bootstrap, de acuerdo a su contenido. En nuestro ejemplo, todos tienen un contenido similar, menos el cuadro de **PHP**. Entonces, le agregamos la clase **.col-md-2** a todas las columnas, menos a una (la que corresponde a PHP), a la que le agregaremos **.col-md-4**. Esto nos da el total de 12 columnas, como nos indica la regla básica de armado de columnas de Bootstrap. Por último, con la clase **.height-bottom** le indicamos que todos los cuadros tengan el mismo largo.

Imágenes en la parte central

Para ubicar las imágenes en la parte central de nuestro sitio web también utilizaremos un módulo personalizado HTML, siguiendo los mismos procedimientos que en el ejemplo anterior. La única diferencia es que aquí vamos a utilizar un módulo solo.

En el editor, insertamos las imágenes desde el menú **Insertar/Insertar imagen**. A la derecha, en **Mostrar Título**, seleccionamos **Ocultar**, y, en **Posición**, configuramos que sea **fullwidth**, ya que queremos que ocupe toda la pantalla. También podríamos agregarle una imagen de fondo, pero en nuestro ejemplo vamos a dejar únicamente las imágenes, sin fondo. Otra cosa que podemos agregarle a cada imagen es un enlace a otra página: simplemente tenemos que seleccionar la imagen y hacer clic en el menú **Insertar/Insertar enlace**, donde debemos escribir la dirección URL de la página a donde queramos dirigirla.

> **LOS MÓDULOS NOS PERMITEN AGREGAR INFORMACIÓN EN ALGUNAS ZONAS DE LA PLANTILLA**

XML

XML (en inglés, ***Extensible Markup Language***) es un formato flexible que permite la gestión y el intercambio de datos estructurados mediante simples archivos de texto. Al igual que HTML, es un lenguaje compuesto por etiquetas (nodos), pero con la diferencia de que en XML somos nosotros quienes definimos las etiquetas y sus atributos. Cada nodo es el equivalente al nombre de campo de una tabla.

Al guardar, podemos observar que nuestro nuevo módulo de imágenes se ubica por arriba del módulo de servicios que hicimos en el ejemplo anterior, y con fondo (*background*) de color blanco.

Figura 8. Al terminar el módulo de imágenes, este se ubica por encima del módulo de servicios.

Comparándolo con la imagen de muestra (**Figura 6**), vemos que el módulo que creamos no está ubicado debajo del módulo de servicios y que el fondo no es gris claro. Para solucionar esto, lo que debemos hacer es cambiar el estilo **CSS**, parte del código **PHP** y modificar un archivo **XML**. Lo podemos hacer desde la ruta en el servidor local (nuestra PC), que en nuestro caso es: **C:\xampp\htdocs\ejemplo2\templates\masterbootstrap** y allí buscar, dentro del directorio **css**, los archivos **template.css**, **templateDetails.xml** e **index.php** para modificarlos.

Otra forma más fácil de realizar esta edición de los archivos es directamente desde el administrador de Joomla!. Desde el menú principal, hacemos clic en **Extensiones** y luego en **Gestor de Plantillas**. Seleccionamos

MASTER BOOTSWATCH

Es una nueva versión de la plantilla Master Bootstrap, creada por Gonzalo Suez. Cuenta con 17 estilos de Bootswatch, plantilla gratuita diseñada en Bootstrap, que fue adaptada por el autor para utilizar en sitios web diseñados en Joomla!. Posee tipografías y colores que se pueden configurar directamente desde la creación de un módulo. Se puede descargar gratuitamente —al igual que Master Bootstrap— desde GitHub: **http://bootswatch.masterbootstrap.com**

MasterBootstrap – Defecto y en la columna **Plantilla** hacemos clic en **MasterBootstrap**. En la solapa **Editor**, podemos ver todo el directorio de archivos que componen la plantilla. Elegimos la carpeta **css** y ahí seleccionamos los archivos **template.css**, **templateDetails.xml** e **index.php**.

En la parte derecha de la pantalla podemos visualizar cada archivo abierto, listo para ser editado, como si fuera un editor de código de los que usamos habitualmente, como NotePad++ o Sublime Text.

El primer archivo que vamos a modificar va a ser **templateDetails.xml**. En este archivo están, entre otras cosas, las posiciones que va a tener el módulo, que es justo lo que debemos agregar:

```
<positions>
                <position>fullwidthbottom2</position>
                <position>top</position>
        <position>breadcrumbs</position>
                <position>navigation</position>
                <position>fullwidth</position>
        <position>showcase</position>
                <position>feature</position>
        <position>left</position>
        <position>right</position>
        <position>content-top</position>
        <position>content-bottom</position>
        <position>bottom</position>
        <position>footer</position>
                <position>debug</position>
</positions>
```

Al comienzo del nodo **<positions></positions>**, podemos observar la nueva posición que agregamos para poder utilizar en el armado de la parte central de nuestra web. Esa posición es: **fullwidthbottom2**. Luego, en el archivo **index.php**, debemos agregar estas líneas de código:

```
<?php if($this->countModules('fullwidthbottom2')) : ?>
<div id="fullwidthbottom2">
<div class="row">
<jdoc:include type="modules" name="fullwidthbottom2" style="block"/>
```

```
</div>
</div>
<?php endif; ?>
```

Debemos agregar este código arriba del comentario **<!-- bottom -->**, para darle una posición determinada dentro de la estructura de la página.

Y, finalmente, en el archivo **template.css** agregamos el siguiente código para cambiar su estilo: queremos que, en vez de tener un fondo color blanco, sea gris claro. Además, le daremos un margen de 20 px.

```
#fullwidthbottom2 .moduletable {
        background: none repeat scroll 0 0 #f3f3f3;
        border: 0;
        margin-bottom: 20px;
}
```

Una vez que realizamos todos estos pasos, volvemos a configurar el módulo de imágenes y, en **Posición**, vamos a encontrar, entre todas las posiciones, la que agregamos recientemente, **fullwidthbottom2**.
Al guardar, veremos que el módulo ya se encuentra ubicado donde queríamos que estuviese: justo debajo del módulo de servicios.
Solo nos resta un módulo de publicidad, donde tenemos la imagen de un libro y su correspondiente enlace a la web del editorial.
El procedimiento es exactamente el mismo: tenemos que agregar el módulo HTML y modificar los diferentes archivos que vimos en el ejemplo. También debemos cambiar el identificador (id) para diferenciarlo del que ya creamos, que es **fullwidthbottom2**.

Pie de página

Para el pie de página y la sección de redes sociales también debemos agregar un módulo HTML personalizado. En el caso de redes sociales, debemos crear una posición nueva en el archivo **templateDetails.xml**, agregar código en el archivo **index.php** y finalmente dentro de la carpeta **css** agregar el estilo en el archivo **template.css**, de la misma forma que vimos en este capítulo en el apartado **Imágenes de la parte central**.

En lo que respecta al pie de página, no necesitamos agregar una nueva posición, porque ya está definida por la plantilla Master

Bootstrap. En el editor, agregamos todos los datos (en nuestro ejemplo, el logo con los derechos de autor); en **Sufijo clase Módulo** agregamos **col-md-12** ya que, como es una sola columna, no necesitamos dividirla. Ocultamos **Mostrar Título** y en **Posición** seleccionamos **footer**.

Para definir el color (en nuestro ejemplo, color negro), tenemos que, nuevamente, abrir y agregar un estilo CSS en el archivo **template.css** dentro de la carpeta **css** de la plantilla.

```
#footer {
  height: 250px;
  background-color: #000000;
}
```

Le damos un alto de 250 px y color negro **(#000000)**. De esta manera, ya tenemos armado el pie de página de nuestro sitio web.

Figura 9. Al pie de página podemos modificarle el ancho, alto y color, entre otras propiedades, modificando el archivo **template.css**.

RESUMEN

En este capítulo, conocimos la adaptación de Joomla! con Bootstrap. Explicamos el procedimiento para instalar la versión 3 de este CMS y aprendimos cómo configurar su idioma. Posteriormente, analizamos la interfaz del administrador del sitio y vimos cómo funciona. Explicamos los pasos para instalar la plantilla Master Bootstrap y luego vimos cómo configurar un proyecto, creando el menú principal, el pie de página y el contenido principal de un sitio web adaptable a todas las resoluciones de pantalla.

Actividades

TEST DE AUTOEVALUACIÓN

1. ¿Cuáles son los pasos para instalar Joomla! 3?
2. ¿Cómo accedemos al administrador de Joomla!?
3. ¿Cuáles son los pasos para instalar Master Bootstrap?
4. ¿Qué utilidad tienen los módulos en Master Bootstrap?
5. ¿Cómo podemos quitar la palabra **Home** que generalmente aparece en la página de inicio de nuestro sitio realizado con Joomla!?

EJERCICIOS PRÁCTICOS

1. Cree la carpeta **ejemplo2**.
2. Descargue e instale el gestor de contenidos Joomla! 3.
3. Configure el español como idioma predeterminado para la administración y para el sitio.
4. Instale la plantilla Master Bootstrap.
5. Cree el menú principal, el pie de página y el contenido del ejemplo, siguiendo los pasos explicados.

PROFESOR EN LÍNEA

Si tiene alguna consulta técnica relacionada con el contenido, puede contactarse con nuestros expertos: profesor@redusers.com

CONÉCTESE CON LOS MEJORES LIBROS DE COMPUTACIÓN

Es posible crear una aplicación o sitio web para móviles sin la necesidad de programar en el lenguaje nativo de cada sistema operativo.

» DESARROLLO
» 192 PÁGINAS
» ISBN 978-987-734-004-4

LLEGAMOS A TODO EL MUNDO VÍA OCA* Y DHL**

MÁS INFORMACIÓN / CONTÁCTENOS

usershop.redusers.com +54 (011) 4110-8700 usershop@redusers.com

*SÓLO VÁLIDO EN LA REPÚBLICA ARGENTINA // **VÁLIDO EN TODO EL MUNDO EXCEPTO ARGENTINA

USERS

Bootstrap

Este libro está destinado a programadores, diseñadores y toda persona que posea conocimientos básicos sobre programación web (HTML, JavaScript y CSS) y quiera aprender a utilizar las herramientas que nos proporciona este framework para la creación de sitios web responsive. Con explicaciones sencillas y ejemplos prácticos, el lector aprenderá a utilizar la herramienta ideal para desarrollar, de manera simple y rápida, sitios web que se adapten a diferentes dispositivos y pantallas.

* EN ESTE LIBRO ENCONTRARÁ:

/ **Introducción:** diseño web adaptable (responsive web design) y el sistema de columnas. Características básicas de HTML5, JavaScript. Hojas de estilo CSS3 y PHP. / **Instalación:** crear un servidor con XAMPP. Editores de texto Sublime Text y Brackets. / **Maquetación:** configuración de las filas y las columnas. Uso de media queries. / **Componentes gráficos:** tipografía, tipos de encabezados, formato de párrafos, tablas, botones, etiquetas, badges, imágenes e iconos. / **Componentes generales:** Armado de la estructura del sitio. Creación de un menú de navegación. Configuración de la paginación. Migas de pan, mensajes de alerta, barras de progreso, listas y paneles. / **Formularios:** creación y configuración. / **Ejemplos prácticos:** gestores de contenido (CMS) y su integración con Bootstrap. Diseño de un portfolio. Instalación y configuración de Joomla!. Presentación de la plantilla Master Bootstrap de Gonzalo Suez.

CON DESARROLLO DE EJEMPLOS PRÁCTICOS

» SOBRE LOS AUTORES

Luis Hereter es analista de sistemas de computación, profesor de Sistemas y desarrollador de aplicaciones móviles y de escritorio.

Viviana Zanini es analista de sistemas de computación y profesora de Informática. Ha sido autora de varias publicaciones en esta misma editorial.

» **NIVEL DE USUARIO**
Intermedio / Avanzado

» **CATEGORÍA**
Desarrollo / Internet / Mobile

ISBN: 978-987-734-049-5

RU

REDUSERS.com
En nuestro sitio podrá encontrar noticias relacionadas y también participar de la comunidad de tecnología más importante de América Latina.

PROFESOR EN LÍNEA
Ante cualquier consulta técnica relacionada con el libro, puede contactarse con nuestros expertos: profesor@redusers.com.