UTN FACULTAD REGIONAL CÓRDOBA
INGENIERÍA EN SISTEMAS DE INFORMACIÓN

Desarrollo de Software
Notas de Clase

Apunte 05 - Maquetación Web y Bootstrap

Maquetación Web

La maquetación Web es el proceso de estructurar y organizar el contenido de una página web utilizando elementos y etiquetas HTML. Este proceso implica dividir el contenido en secciones lógicas y aplicar una estructura coherente y semántica que facilite la comprensión y la navegación por parte de los usuarios y los motores de búsqueda.

En la maquetación HTML 5, se utilizan elementos como `<header>`, `<nav>`, `<main>`, `<section>`, `<article>`, `<aside>`, `<footer>`, entre otros, para definir la estructura y el flujo del contenido de la página. Estos elementos proporcionan un significado semántico al contenido y ayudan a los desarrolladores a entender la función y el propósito de cada sección de la página.

Además de definir la estructura del contenido, la maquetación HTML también implica la aplicación de estilos CSS para controlar el diseño visual de la página, como el tamaño y el posicionamiento de los elementos, los colores y las fuentes utilizadas, los márgenes y rellenos, entre otros aspectos.

En resumen, la maquetación HTML es el proceso de organizar y estructurar el contenido de una página web utilizando etiquetas HTML y estilos CSS para crear una experiencia de usuario coherente y atractiva.

Estructura de la maquetación

Se podría decir que una página web tiene una estructura básica, la cual está conformada por las siguientes etiquetas:

- Cabecera (`<header>`)
- Menú principal (`<nav>`)
- Cuerpo (`<body>`)
- Pie de página (`<footer>`)

Ahora, si hay que hablar específicamente de una estructura de maquetación web HTML5, se le agregan algunas secciones como:

- Títulos y subtítulos (`<hgroup>`)
- Contenido principal del documento (`<main>`)
- Contenido (`<article>`)
- Sección del documento (`<section>`)
- Menús secundarios (`<aside>`)
- Foto y leyenda (`<figure>` y `<figcaption>`)
- Detalles adicionales y sumario (`<details>` y `<summary>`)

También, existe una estructura de maquetación web tradicional, la cual estaba delimitada por etiquetas `<div>`. Es posible combinar ambas estructuras debido a que algunos navegadores podrían no reconocer las nuevas etiquetas. Si ponemos un ejemplo, la etiqueta `<header>`, con una estructura combinada, sería `<div id="header">`. En los comienzos del desarrollo web toda maquetación se reducía simplemente al uso de etiquetas `<div>`anidadas, combinadas en ocasiones con tablas, y reglas CSS bien definidas.

Modelo de Cajas

Todo en CSS tiene una caja alrededor, y comprender estas cajas es clave para poder crear diseños con CSS o para alinear elementos con otros elementos.

Cajas en bloque y en línea

En CSS, en general, hay dos tipos de cajas: cajas en bloque y cajas en línea. Estas características se refieren al modo como se comporta la caja en términos de flujo de página y en relación con otras cajas de la página.

Si una caja se define como un bloque, se comportará de las maneras siguientes:

- La caja fuerza un salto de línea al llegar al final de la línea.
- La caja se extenderá en la dirección de la línea para llenar todo el espacio disponible que haya en su contenedor. En la mayoría de los casos, esto significa que la caja será tan ancha como su contenedor, y llenará el 100% del espacio disponible.
- Se respetan las propiedades width y height.
- El relleno, el margen y el borde mantienen a los otros elementos alejados de la caja.

A menos que decidamos cambiar el tipo de visualización a en línea, elementos como los encabezados (por ejemplo, `<h1>`) y todos los elementos `<p>` usan por defecto block como tipo de visualización externa.

Si una caja tiene una visualización externa de tipo inline, entonces:

- La caja no fuerza ningún salto de línea al llegar al final de la línea.
- Las propiedades width y height no se aplican. Se aplican relleno, margen y bordes verticales, pero no mantienen alejadas otras cajas en línea.
- Se aplican relleno, margen y bordes horizontales, y mantienen alejadas otras cajas en línea.

El elemento `<a>`, que se utiliza para los enlaces, y los elementos `<span>`, `<em>` y `<strong>` son ejemplos de elementos que se muestran en línea por defecto.

El tipo de caja que se aplica a un elemento está definido por los valores de propiedad display, como block y inline, y se relaciona con el valor externo (outer) de visualización (display).

Visualización interna y externa

Como se mencionó anteriormente, las cajas en CSS tienen un tipo de visualización externa, que define si se trata de una caja en bloque o en línea.

Sin embargo, las cajas también tienen un tipo de visualización interna, que determina cómo se disponen los elementos dentro de esa caja. De forma predeterminada, los elementos dentro de una caja se presentan en **flujo normal**, lo que significa que se comportan como otros elementos de tipo en bloque o en línea (como se explicó anteriormente).

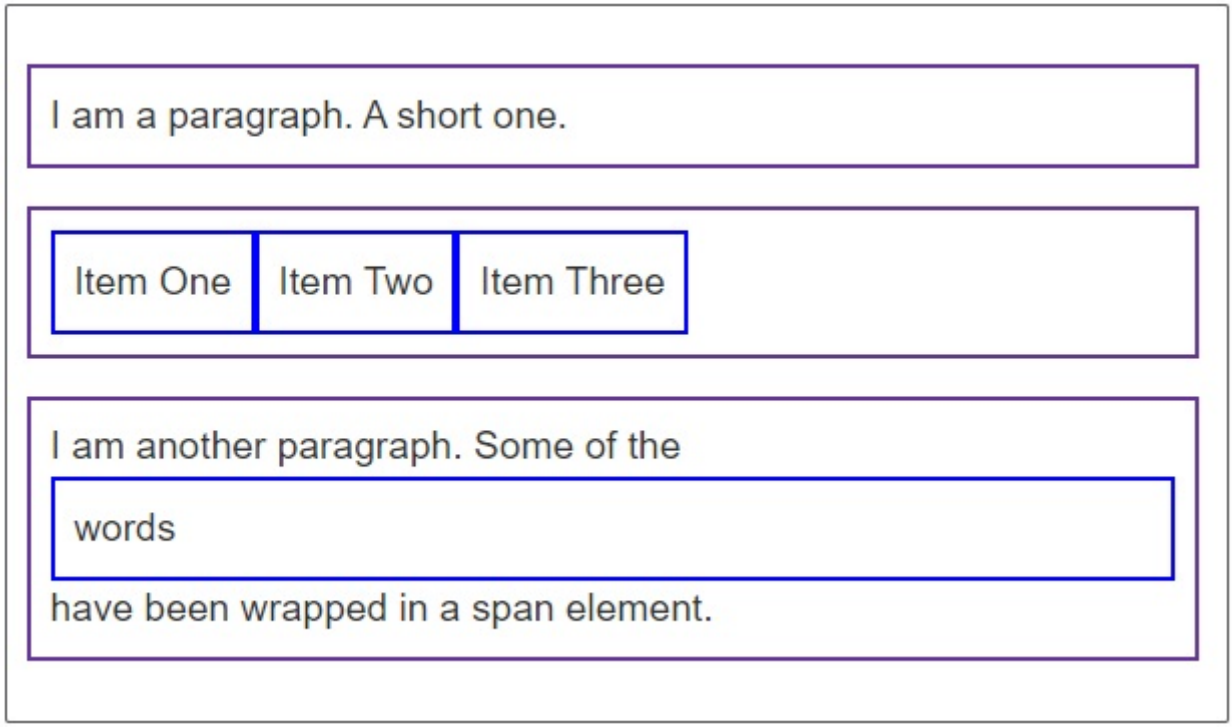
Sin embargo, podemos cambiar el tipo de visualización interna utilizando valores de display, como flex. Si en un elemento establecemos display: flex;, el tipo de visualización externa es de tipo bloque (block), pero el tipo de visualización interna cambia a flexible (flex). Cualquier elemento que sea hijo directo de esta caja pasará a comportarse como un elemento de tipo flex, de acuerdo con las reglas que se establecen en la especificación de Flexbox, tema que veremos más adelante.

Un ejemplo sencillo

A continuación se muestran tres elementos HTML diferentes, todos con visualización externa de tipo block. El primero es un párrafo, que tiene un borde añadido con CSS. El navegador representa esto como una caja en bloque, por lo que el párrafo comienza en una línea nueva y se expande por todo el ancho disponible.

El segundo es una lista, que se presenta usando display: flex. Esto establece una disposición flexible para los elementos que están dentro del contenedor; sin embargo, la lista en sí misma es una caja que se comporta en bloque y, como el párrafo, se expande por todo el ancho del contenedor y fuerza un salto de línea al llegar al final de línea.

Debajo hay un párrafo a nivel de bloque, dentro del cual hay dos elementos `<span>`. Estos elementos normalmente serían de tipo inline; sin embargo, uno de los elementos tiene una clase de bloque, y lo hemos establecido como display: block.



Que en código HTML resulta:

```
<p>I am a paragraph. A short one.</p>
<ul>
  <li>Item One</li>
  <li>Item Two</li>
  <li>Item Three</li>
</ul>
<p>I am another paragraph. Some of the <span class="block">words</span> have been
wrapped in a <span>span element</span>.</p>
```

y en hoja de estilos (CSS):

```
p,
ul {
  border: 2px solid rebeccapurple;
  padding: .5em;
}

.block,
li {
  border: 2px solid blue;
  padding: .5em;
}

ul {
  display: flex;
  list-style: none;
}

.block {
  display: block;
}
```

Se puede ver cómo se comportan los elementos inline en el ejemplo siguiente. Los elementos `<span>` del primer párrafo están en línea de manera predeterminada y, por lo tanto, no fuerzan ningún salto de línea.

También hay un elemento `<ul>` que se establece como display: inline-flex, que crea una caja con un comportamiento de tipo en línea alrededor de algunos elementos de tipo flex.

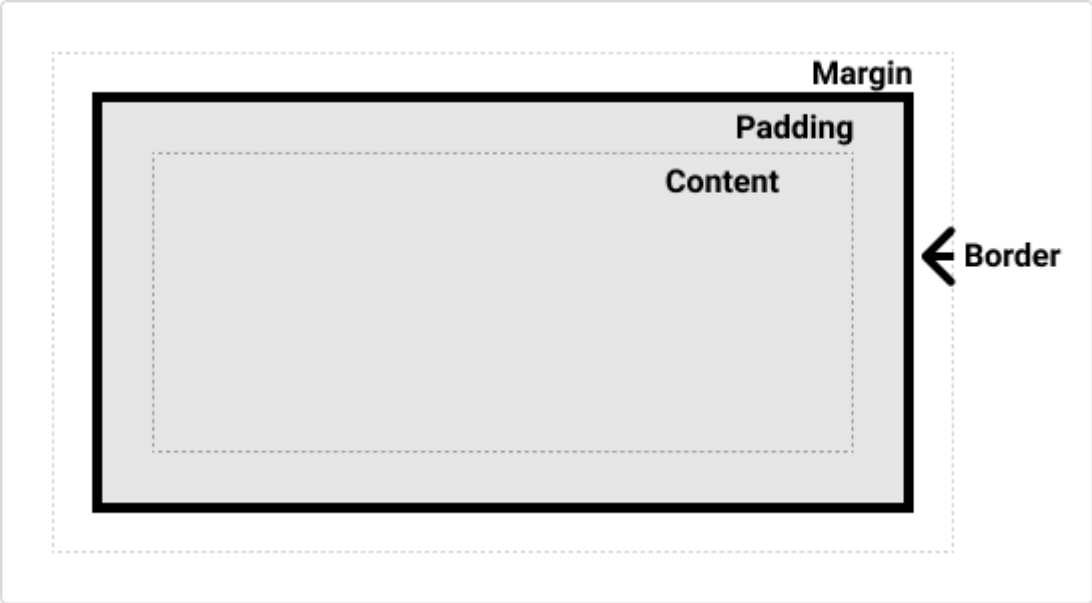
Finalmente, hay dos párrafos configurados con display: inline. El contenedor flexible en línea y los párrafos fluyen todos juntos en línea, en lugar de dividirse en líneas nuevas como lo harían si se mostraran como elementos de bloque.

Partes de una caja

Al hacer una caja de tipo bloque en CSS tenemos los elementos siguientes:

- El contenido de la caja **(o content box)**: El área donde se muestra el contenido, cuyo tamaño puede cambiarse utilizando propiedades como width y height.
- El relleno de la caja **(o padding box)**: El relleno es espacio en blanco alrededor del contenido; es posible controlar su tamaño usando la propiedad padding y otras propiedades relacionadas.
- El borde de la caja **(o border box)**: El borde de la caja envuelve el contenido y el de relleno. Es posible controlar su tamaño y estilo utilizando la propiedad border y otras propiedades relacionadas.
- El margen de la caja **(o margin box)**: El margen es la capa más externa. Envuelve el contenido, el relleno y el borde como espacio en blanco entre la caja y otros elementos. Es posible controlar su tamaño usando la propiedad margin y otras propiedades relacionadas.

El diagrama siguiente muestra estas capas:



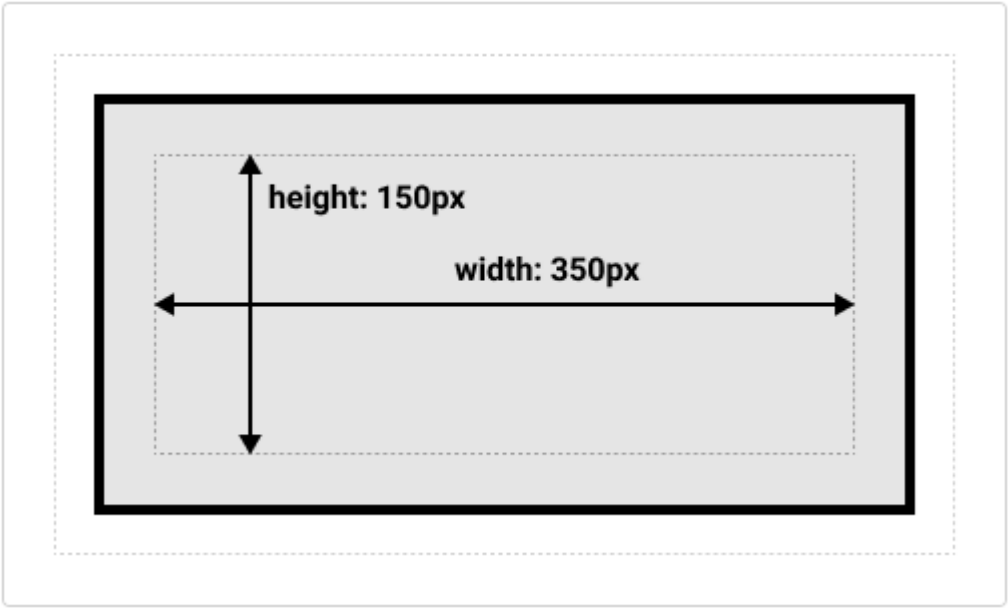
El modelo de cajas CSS estándar

En el modelo de cajas estándar, cuando estableces los atributos width y height para una caja, defines el ancho y el alto del contenido de la caja. Cualquier área de relleno y borde se añade a ese ancho y alto para obtener el tamaño total que ocupa la caja. Esto se muestra en la imagen que se muestra a continuación.

Si se supone que la caja tiene el CSS siguiente, que establece los valores para las propiedades width, height, margin, border, y padding:

```
.box {
  width: 350px;
  height: 150px;
  margin: 10px;
  padding: 25px;
  border: 5px solid black;
}
```

El espacio que ocupa nuestra caja usando el modelo de cajas estándar será en realidad de 410 px (350 + 25 + 25 + 5 + 5); y su altura, de 210 px (150 + 25 + 25 + 5 + 5), porque el área de relleno y el borde se añaden al ancho que se utiliza para el contenido de la caja.



> **Nota:** El margen no se cuenta para el tamaño real de la caja; por supuesto, afecta al espacio total que la caja ocupa en la página, pero solo al espacio de fuera de la caja. El área de la caja se termina en el borde, no se extiende hasta el margen.

Bootstrap: diseño **responsive**

Todo lo anterior es válido para pensar cómo organizar y estructurar el contenido de una página HTML, pero falta un detalle, si se quiere mostrar ese diseño en diferentes dispositivos haciendo que el contenido se ajuste automáticamente a ciertas dimensiones (resoluciones), el esfuerzo es sin dudas mucho mayor.

Bootstrap

Bootstrap es un framework de desarrollo web de código abierto que proporciona una colección de herramientas y estilos CSS predefinidos para facilitar la creación de sitios web modernos y receptivos. La necesidad de un diseño responsivo se ha vuelto cada vez más crucial en el mundo del desarrollo web debido a la creciente diversidad de dispositivos y pantallas con diferentes tamaños y resoluciones.

Con la proliferación de dispositivos móviles, tablets y otros dispositivos con acceso a Internet, es fundamental que los sitios web se adapten y se vean bien en todas las plataformas. Un diseño responsivo garantiza una experiencia de usuario consistente y óptima, independientemente del dispositivo que se esté utilizando para acceder al sitio web.

Bootstrap aborda esta necesidad ofreciendo un sistema de cuadrícula flexible y componentes receptivos que se adaptan automáticamente al tamaño de la pantalla del dispositivo. Esto permite a los desarrolladores crear fácilmente sitios web que se ajusten y se vean bien en una amplia variedad de dispositivos, desde teléfonos inteligentes y tabletas hasta computadoras de escritorio y televisores inteligentes.

Al utilizar Bootstrap, los desarrolladores pueden ahorrar tiempo y esfuerzo en el proceso de desarrollo, ya que el framework proporciona una base sólida y consistente para construir sitios web receptivos y atractivos. Además, Bootstrap ofrece una amplia gama de componentes y utilidades listas para usar, lo que facilita la creación de diseños complejos y funcionalidades interactivas sin tener que escribir código CSS personalizado desde cero.

¿Cómo funciona Bootstrap?

Bootstrap está constituido por una serie de archivos CSS y JavaScript responsables de asignar características específicas a los elementos de la página.

Hay un archivo principal llamado **bootstrap.css**, que contiene una definición para todos los estilos utilizados. Básicamente, la estructura del framework se compone de dos directorios:

- css: contiene los archivos necesarios para la estilización de los elementos y una alternativa al tema original;
- js: contiene la parte posterior del archivo bootstrap.js (original y minificado), responsable de la ejecución de aplicaciones de estilo que requieren manipulación interactiva.

Bootstrap se puede instalar de varias maneras, pero una de las formas más comunes es a través de CDN (Content Delivery Network) o mediante descarga y configuración manual.

Bootstrap se puede instalar de varias maneras, pero una de las formas más comunes es a través de CDN (Content Delivery Network) o mediante descarga y configuración manual. Aquí tienes una descripción general de ambas opciones:

1. **Instalación a través de CDN:**
    - Una forma rápida y sencilla de comenzar a usar Bootstrap es enlazando los archivos CSS y JavaScript de Bootstrap directamente desde un CDN.
    - En tu documento HTML, puedes agregar los enlaces a los archivos de Bootstrap proporcionados por un CDN, como por ejemplo los enlaces a los archivos CSS y JavaScript de Bootstrap en tu sección `<head>`:

```
<!-- CSS de Bootstrap -->
<link
href="https://stackpath.bootstrapcdn.com/bootstrap/5.0.0/css/bootstrap.min.c
ss" rel="stylesheet">

<!-- JavaScript de Bootstrap y dependencias -->
<script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js">
</script>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/5.0.0/js/bootstrap.bundle.
min.js"></script>
```

    Esto enlazará tu página web con los archivos CSS y JavaScript de Bootstrap alojados en el CDN correspondiente, lo que te permite comenzar a utilizar Bootstrap de inmediato.

2. Descarga y configuración manual:
    - Otra opción es descargar los archivos CSS y JavaScript de Bootstrap desde el sitio web oficial de Bootstrap (https://getbootstrap.com/) y agregarlos manualmente a tu proyecto.
    - Una vez descargados, puedes enlazar los archivos CSS y JavaScript de Bootstrap en tu documento HTML de la misma manera que lo harías con los enlaces de CDN, pero esta vez utilizando rutas locales a los archivos descargados en tu proyecto.

```
<!-- CSS de Bootstrap -->
<link href="ruta/a/bootstrap.min.css" rel="stylesheet">

<!-- JavaScript de Bootstrap y dependencias -->
<script src="ruta/a/jquery.min.js"></script>
<script src="ruta/a/bootstrap.bundle.min.js"></script>
```

Es importante ajustar las rutas de los archivos CSS y JavaScript para que coincidan con la estructura de directorios de tu proyecto.

Principales estilos (clases)

1. Grid System (Sistema de Cuadrícula):

   `.container`: Contenedor principal que envuelve el contenido y proporciona márgenes laterales. `.row`: Fila que contiene columnas dentro de un contenedor. `.col-*`: Columnas que se colocan dentro de filas para crear diseños responsivos. El asterisco * puede ser un número del 1 al 12, indicando el ancho de la columna en diferentes tamaños de pantalla.

2. Typography (Tipografía):

   `.h1` a `.h6`: Títulos de encabezado con diferentes tamaños. `.lead`: Párrafo con estilo de plomo para destacar texto. `.text-*`: Clases para cambiar el color del texto. El asterisco * puede ser primary, secondary, success, info, warning, danger, light o dark.

3. Buttons (Botones):

   `.btn`: Clase base para estilizar botones. `.btn-*`: Variaciones de estilo para diferentes tipos de botones, como btn-primary, btn-secondary, btn-success, etc. `.btn-lg`, `.btn-sm`, `.btn-block`: Clases para cambiar el tamaño y el ancho de los botones.

4. Forms (Formularios):

   `.form-control`: Estilo para inputs, selects y textareas. `.form-check`: Estilo para checkboxes y radio buttons. `.form-group`: Contenedor de elementos de formulario para agruparlos y aplicar estilos de manera uniforme. `.form-inline`, `.form-horizontal`: Clases para cambiar el diseño de los formularios.

5. Navbar (Barra de Navegación):

   `.navbar`: Contenedor principal de la barra de navegación. `.navbar-brand`: Estilo para el logotipo o nombre de la marca. `.navbar-nav`: Estilo para el menú de navegación. `.navbar-toggler`: Botón de alternancia para dispositivos móviles.

6. Utilities (Utilidades):

   `.d-*`: Clases para controlar la visualización de elementos en diferentes tamaños de pantalla (por ejemplo, d-none, d-block, d-flex). `.m-*`, `.p-*`: Clases para agregar márgenes y rellenos a elementos (por ejemplo, m-2, p-3). `.text-*`, `.bg-*`: Clases para cambiar el color del texto y el fondo respectivamente (por ejemplo, text-muted, bg-light).

Componentes

Los componentes Bootstrap son elementos y conjuntos de elementos predefinidos que facilitan la construcción de interfaces de usuario modernas y receptivas en sitios web. Estos componentes están diseñados para ser fáciles de usar y altamente personalizables, lo que permite a los desarrolladores crear rápidamente diseños atractivos y funcionales sin necesidad de escribir código CSS o JavaScript personalizado desde cero.

Algunos ejemplos de componentes Bootstrap incluyen:

- **Navbar** (Barra de Navegación): Una barra de navegación responsiva que puede contener enlaces, botones y otros elementos de navegación.
- **Cards** (Tarjetas): Contenedores flexibles que se utilizan para mostrar contenido como imágenes, texto y botones de una manera organizada y visualmente atractiva.
- **Buttons** (Botones): Estilos predefinidos para botones que se utilizan para realizar acciones, como enviar formularios, iniciar sesión o descargar archivos.
- **Forms** (Formularios): Estilos predefinidos para inputs, selects, textareas, checkboxes y radio buttons que facilitan la creación de formularios bien diseñados y receptivos.
- **Carousel** (Carrusel): Componentes interactivos que permiten mostrar múltiples elementos de contenido, como imágenes o videos, en un espacio limitado de la página.
- **Modal** (Ventana Modal): Ventanas emergentes superpuestas que se utilizan para mostrar información adicional o solicitar acciones del usuario, como confirmaciones o ingreso de datos.
- **Dropdown** (Menú Desplegable): Menús interactivos que permiten al usuario seleccionar una opción de una lista desplegable de elementos.

Ejemplo de aplicación

Diseñar una pagína sencilla con la siguiente estructura:

- Presenta una barra de navegación con dos enlaces para desplazarse a las secciones "Bienvenida" y "Contacto". La sección de "Bienvenida" muestra un encabezado y un párrafo de bienvenida, mientras que la sección de "Contacto" incluye un formulario de contacto con campos para nombre, correo electrónico y mensaje.
- Finalmente, hay un pie de página básico con información de copyright.

Utilizar la CDN de Bootstrap para cargar los estilos CSS y los scripts JavaScript necesarios (según la última versión disponible).

Ver código del ejemplo aquí

Práctica para el estudiante:

Tomando como base el ejemplo anterior se pide como desafío para el estudiante implementar las siguientes variantes:

- **Personalización del diseño**: cambiar el color de fondo de las secciones, los estilos de fuente y los tamaños de los elementos.
- **Agregar más secciones**: agregar una sección de servicios con una galería de imágenes (carrusel) y una sección de testimonios.
- **Validación de formularios**: implementar la validación del formulario utilizando JavaScript y las clases de validación de formularios de Bootstrap.
- **Integración de fuentes y iconos personalizados**: integrar fuentes personalizadas y iconos de fuentes web en la página utilizando herramientas como Google Fonts y Font Awesome.
- **Responsive Design**: asegurar que la página se vea bien en diferentes dispositivos y tamaños de pantalla. Utilizar las herramientas para desarrolladores integrada con el navegador.

Bibliografía o Referencia

- El modelo de caja
- Comience con Bootstrap