

Clase 2

HTML y CSS

IIC2513 - Tecnologías y Aplicaciones Web

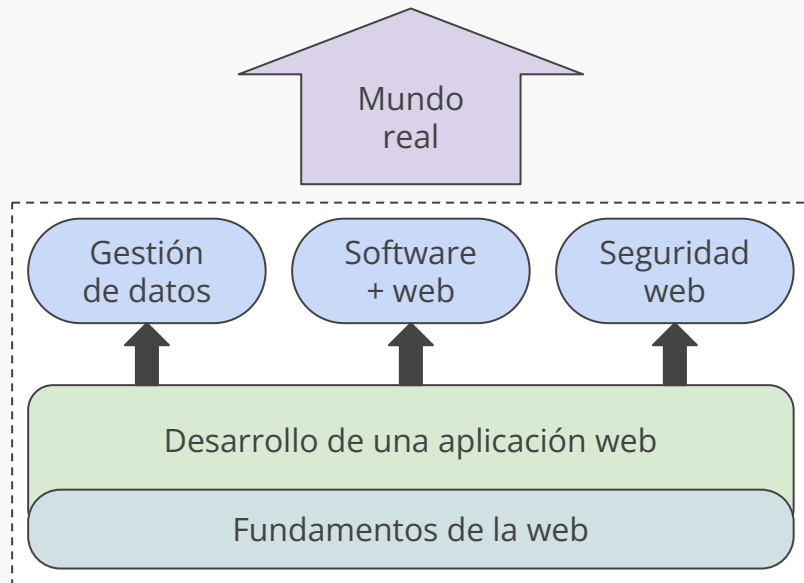
Antonio Ossa Guerra
aaossa@ing.puc.cl

1. ECA se aplicará en el curso
2. Hoy será la primera actividad

Anuncios

Contenidos del curso

- (1) Fundamentos de la web
- (2) Desarrollo de una app web
- (3) Gestión de datos
- (4) Software + web
- (5) Seguridad web
- (6) El mundo real

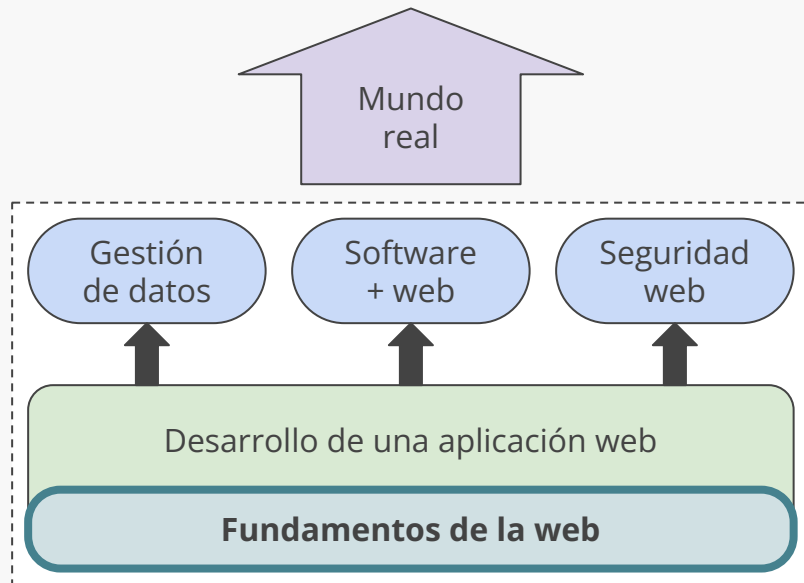


Contenidos del curso

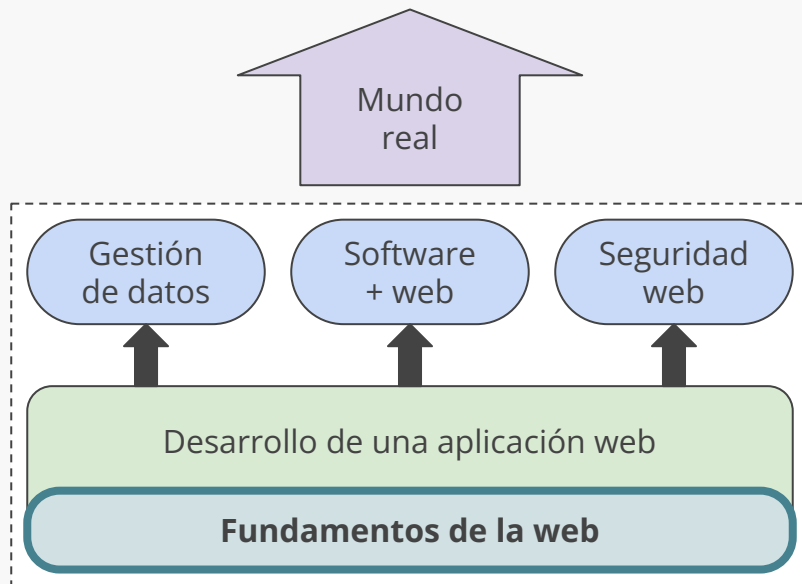
Fundamentos de la web

Veremos definiciones relacionadas con internet, protocolos y qué son **HTML**, **CSS** y JavaScript...

... para comprender la web como una **plataforma de desarrollo**, su funcionamiento básico, y protocolos y **tecnologías** detrás de esta



Contenidos del curso

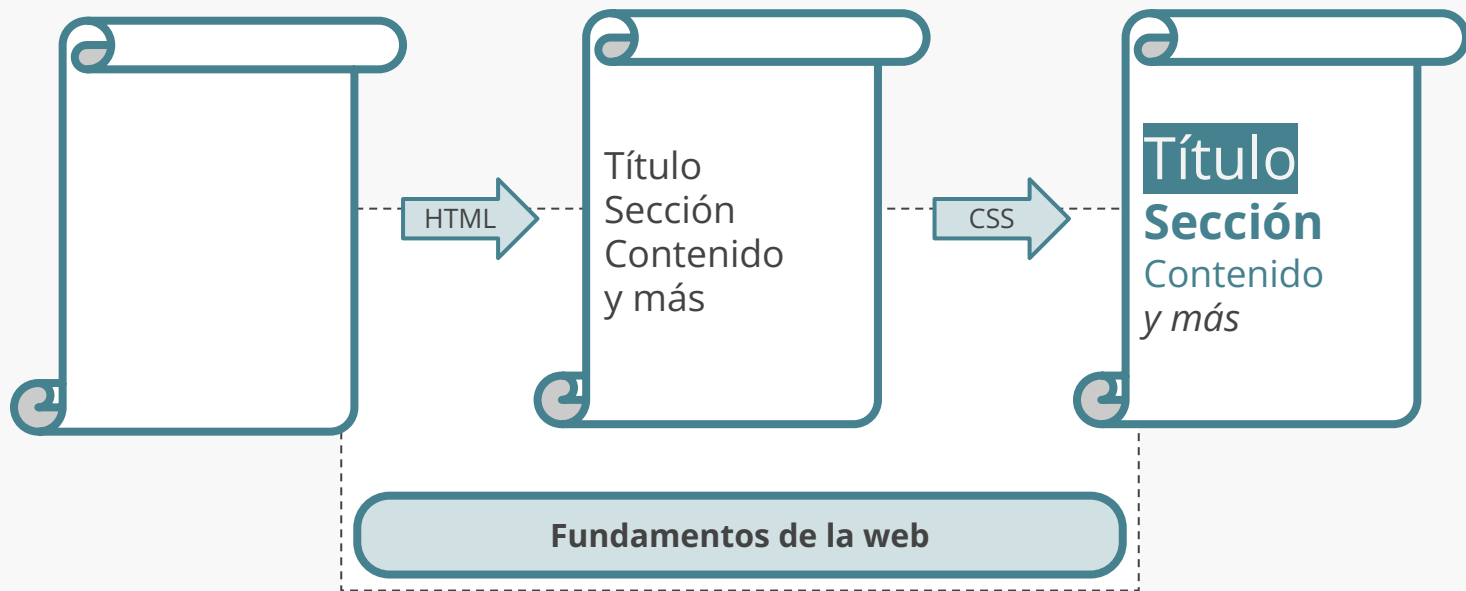


Contenidos del curso



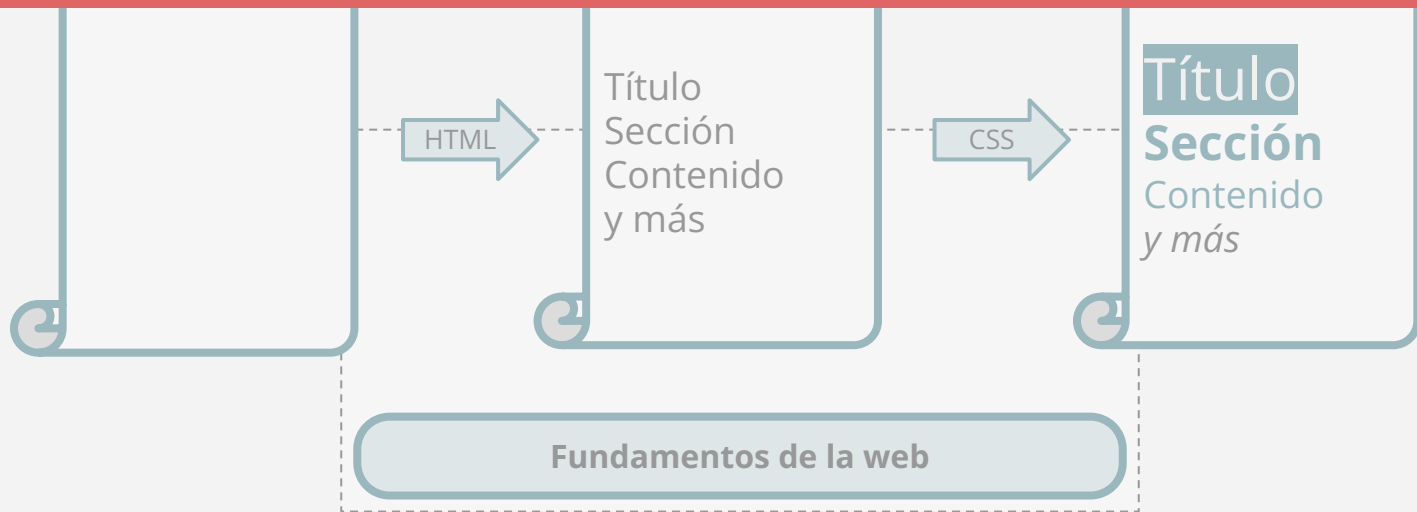
Fundamentos de la web

Contenidos del curso

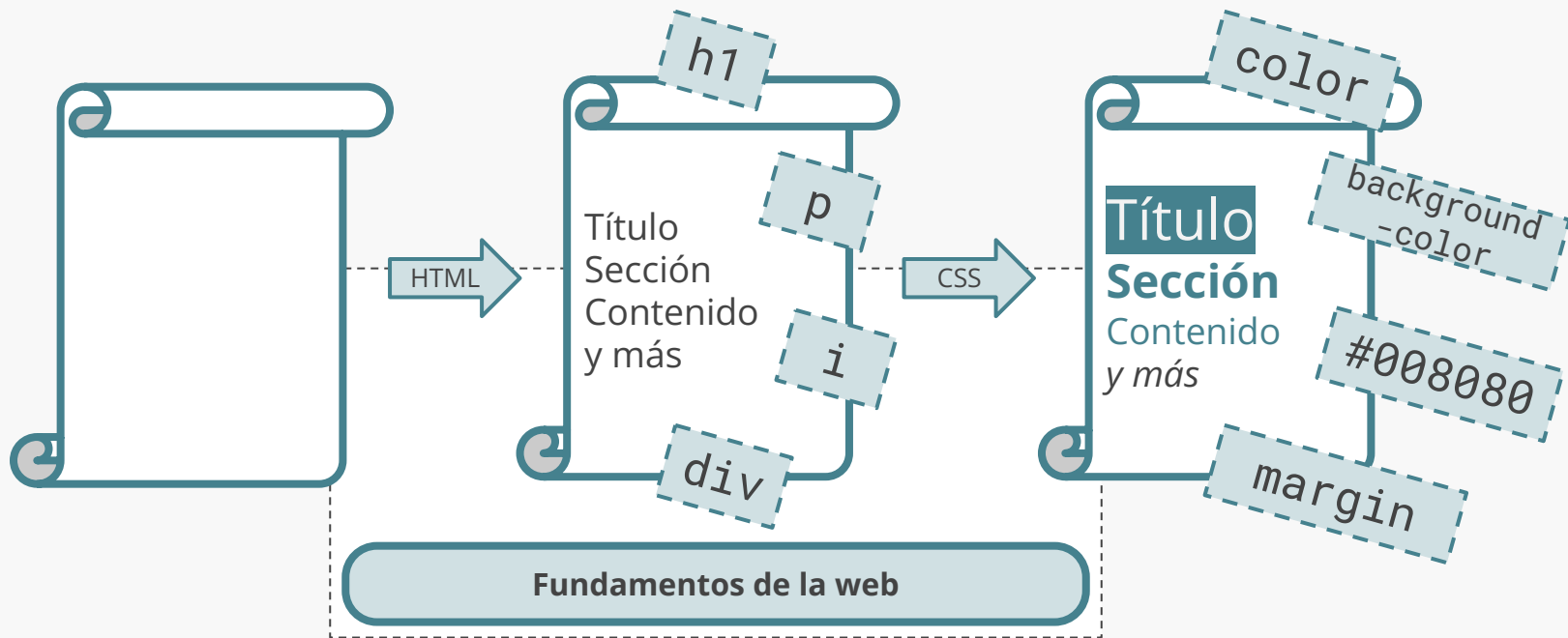


Contenidos del curso

Pregunta# ¿Cómo funcionan HTML y CSS en conjunto para desplegar contenido web en el navegador?



Contenidos del curso



Menti*#Empecemos a hablar de esto*

¿Cuánto crees que sabes de...

HTML 🏢

3.0

CSS 🎨

2.5

JS 🤖

1.8

Poquito 🐢

Soy crack 😎

18



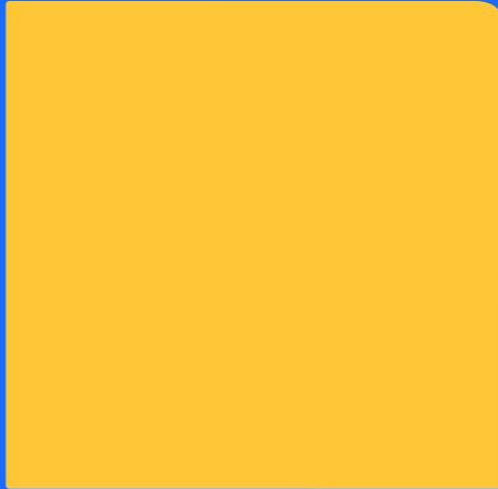
65



¿Sabes cómo se llama esta planta? 🌿



33



Sipis

26



Nopis 🦴

13

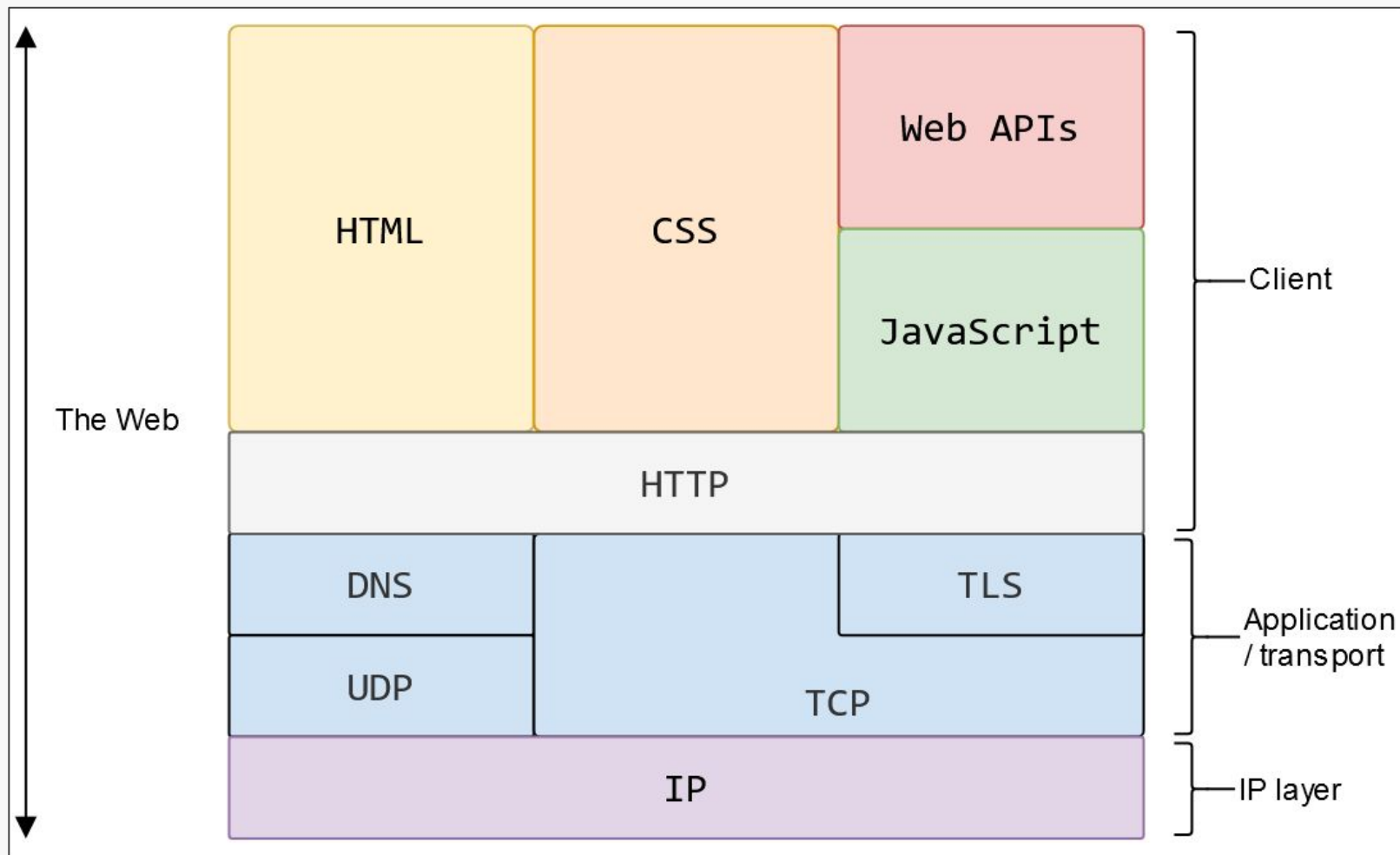


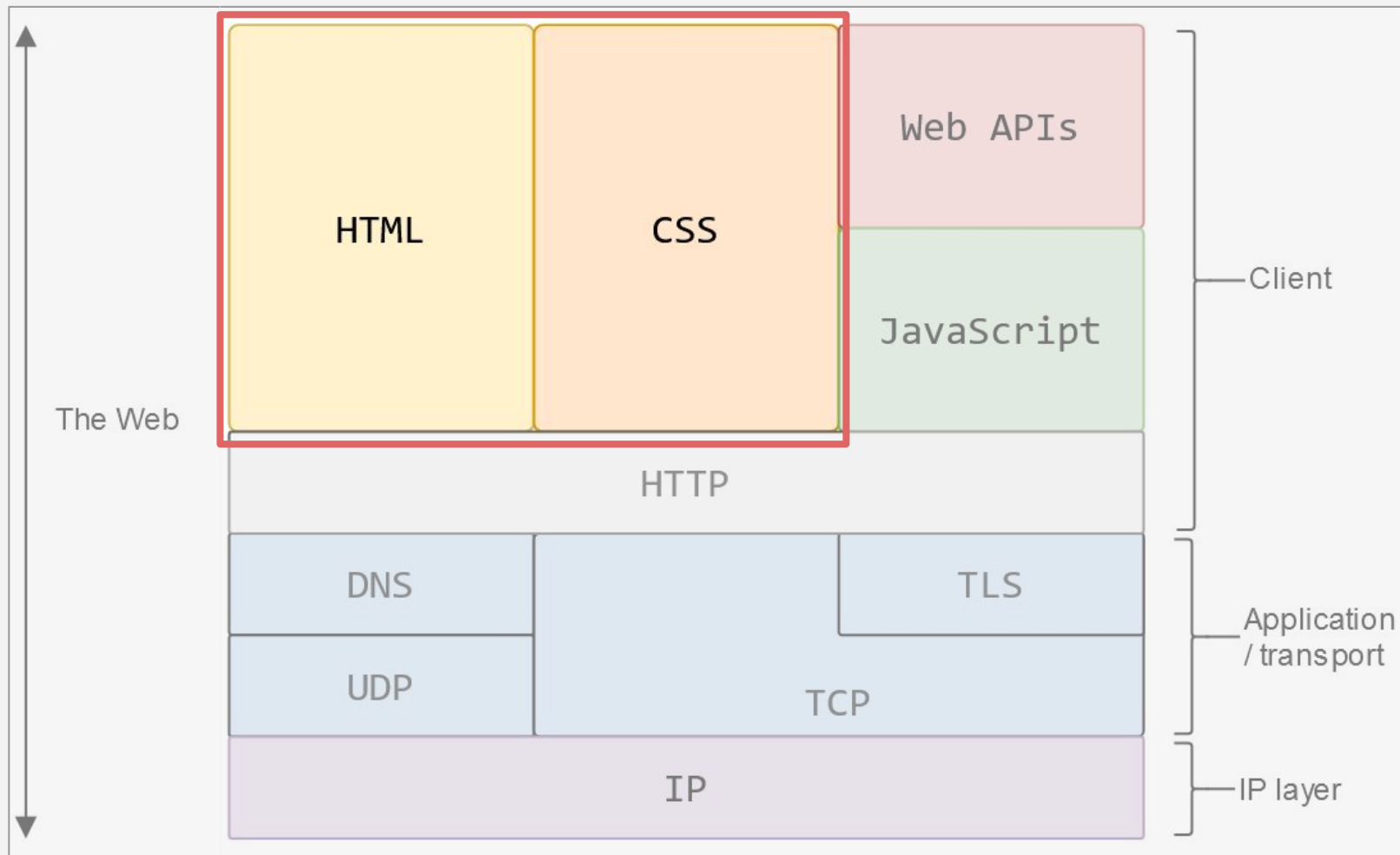
59



¡Gracias!

Empecemos con la clase de este tema 🧐





Agenda

✨ **HTML para estructura**

CSS para presentación

Qué hace el navegador

HTML (estructura)

HyperText Markup Language (HTML) es un **lenguaje de marcado**, no de programación, que define el significado y la estructura del contenido web

Este etiquetado se entrega en forma de etiquetas (*tags*) que describen de qué se trata un texto e información adicional de un elemento:

```
<p class="mi-planta">Mi planta es muy verde y tiene 5 años</p>  

```

Mi planta es muy verde y tiene 5 años



Brevísima historia de HTML

- 1991** ● **HTML "1"** Primera versión por Tim Berners-Lee listando 18 *tags*
- 1994** ● Creación de The World Wide Web Consortium ([W3C](#))
- 1995-1997** ● **HTML 2** Estandarización de *features* existentes para "equilibrar terreno"
- 1995-2001** ● Primera "browser war" (Netscape, Mosaic / Internet Explorer)
- 1997** ● **HTML 3.2** Primera versión desarrollada y estandarizada por la W3C
- 1997-2014** ● **HTML 4.0** Soporte para CSS y JavaScript
- 2004-2007** ● Segunda "browser war" (Netscape / Mozilla, IE / Edge, Chrome)
- 2014-...** ● **HTML5** Versión más accesible y semántica, usada hoy

Línea de tiempo de HTML

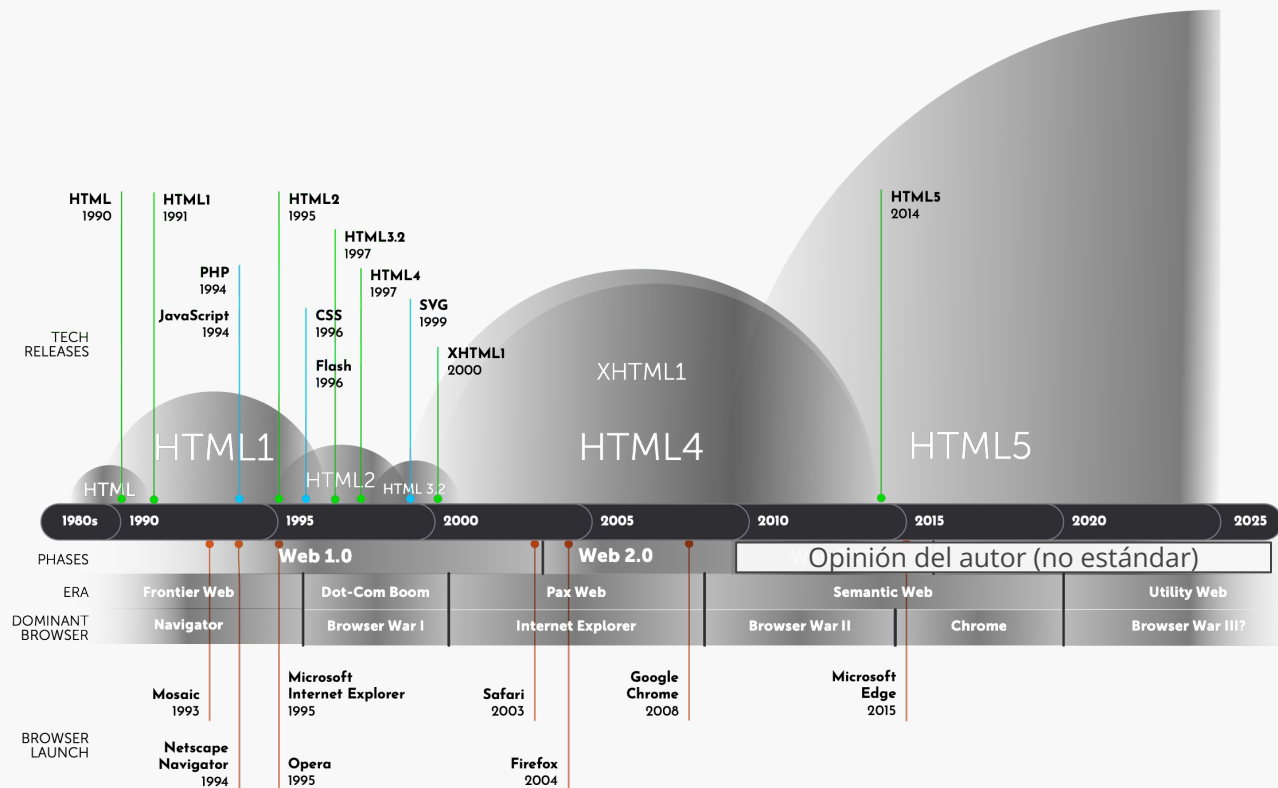


Figura: [The Hellish History of HTML: An incomplete and personal account - HTMHeHl](#)



Intro a HTML

Anatomía de un elemento

Por medio de *tags* (que se abren y cierran) y atributos (propiedades con un valor fijo) podemos definir e indicar un **elemento HTML**:

```
<tag-apertura atributo="valor">Contenido</tag-cierre>
```

Los elementos HTML son **anidables**, lo que permite estructurar una página web utilizando una sintaxis clara, que además es “parseable”:

```
<p>Mi planta es <strong>muy</strong> verde</p>
```

Aunque también existen **elementos vacíos**, que no poseen contenido:

```

```

Un documento HTML

Algunas partes importantes de un documento:

- **<!DOCTYPE html>** indica al navegador cómo parsear el documento ([w3c](#))
- **html** es el nodo raíz del documento HTML
- **head** contiene el título, links a documentos auxiliares, metadata, etc.
- **<meta charset="utf-8">** una forma de declarar el *encoding* del documento
- **title** es el título de la página (pestaña)
- **body** contiene el cuerpo del documento, todo lo que se desea mostrar

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Título</title>
</head>
<body>
  <p>Contenido</p>
</body>
</html>
```

Un documento HTML

Algunas partes importantes de un documento:

- `<!DOCTYPE html>` indica al navegador cómo parsear el documento ([w3c](#))
- `html` es el nodo raíz del documento HTML
- `head` contiene el título, links a documentos auxiliares, metadata, etc.
- `<meta charset="utf-8">` una forma de declarar el *encoding* del documento
- `title` es el título de la página (pestaña)
- `body` contiene el cuerpo del documento, todo lo que se desea mostrar

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>Título</title>
</head>
<body>
    <p>Contenido</p>
</body>
</html>
```

Un documento HTML

Algunas partes importantes de un documento:

- **<!DOCTYPE html>** indica al navegador cómo parsear el documento ([w3c](#))
- **html** es el nodo raíz del documento HTML
- **head** contiene el título, links a documentos auxiliares, metadata, etc.
- **<meta charset="utf-8">** una forma de declarar el *encoding* del documento
- **title** es el título de la página (pestaña)
- **body** contiene el cuerpo del documento, todo lo que se desea mostrar

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Título</title>
  </head>
  <body>
    <p>Contenido</p>
  </body>
</html>
```


Un documento HTML

Algunas partes importantes de un documento:

- **<!DOCTYPE html>** indica al navegador cómo parsear el documento ([w3c](#))
- **html** es el nodo raíz del documento HTML
- **head** contiene el título, links a documentos auxiliares, metadata, etc.
- **<meta charset="utf-8">** una forma de declarar el *encoding* del documento
- **title** es el título de la página (pestaña)
- **body** contiene el cuerpo del documento, todo lo que se desea mostrar

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Título</title>
  </head>
  <body>
    <p>Contenido</p>
  </body>
</html>
```

Un documento HTML

Algunas partes importantes de un documento:

- **<!DOCTYPE html>** indica al navegador cómo parsear el documento ([w3c](#))
- **html** es el nodo raíz del documento HTML
- **head** contiene el título, links a documentos auxiliares, metadata, etc.
- **<meta charset="utf-8">** una forma de declarar el *encoding* del documento
- **title** es el título de la página (pestaña)
- **body** contiene el cuerpo del documento, todo lo que se desea mostrar

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Título</title>
</head>
<body>
  <p>Contenido</p>
</body>
</html>
```

Un documento HTML

Algunas partes importantes de un documento:

- **<!DOCTYPE html>** indica al navegador cómo parsear el documento ([w3c](#))
- **html** es el nodo raíz del documento HTML
- **head** contiene el título, links a documentos auxiliares, metadata, etc.
- **<meta charset="utf-8">** una forma de declarar el *encoding* del documento
- **title** es el título de la página (pestaña)
- **body** contiene el cuerpo del documento, todo lo que se desea mostrar

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Título</title>
</head>
<body>
  <p>Contenido</p>
</body>
</html>
```

Un documento HTML

Algunas partes importantes de un documento:

- **<!DOCTYPE html>** indica al navegador cómo parsear el documento ([w3c](#))
- **html** es el nodo raíz del documento HTML
- **head** contiene el título, links a documentos auxiliares, metadata, etc.
- **<meta charset="utf-8">** una forma de declarar el *encoding* del documento
- **title** es el título de la página (pestaña)
- **body** contiene el cuerpo del documento, todo lo que se desea mostrar

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Título</title>
</head>
<body>
  <p>Contenido</p>
</body>
</html>
```

Marcado de texto

Mi título principal

Título de nivel superior

Subtítulo

Sub-subtítulo

Y así...

... hasta llegar a h6

```
<h1>Mi título principal</h1>  
<h2>Título de nivel superior</h2>  
<h3>Subtítulo</h3>  
<h4>Sub-subtítulo</h4>  
<h5>Y así...</h5>  
<h6>... hasta llegar a h6</h6>
```

Marcado de texto

- **h1**, ..., **h6** son encabezados (*headings*) o títulos de secciones
- **div** es un contenedor genérico de contenido (más usado en CSS)
- **p** define un párrafo de texto en la página
- **em**, **b** y **i** indican énfasis, **bold** e *italics*
- **ol** y **ul** indican una lista ordenada (numerada) y no ordenada, mientras que **li** es cada elemento listado

```
<h1>IIC2513 - Sección 1</h1>
```

```
<p><b>Profesor:</b> Antonio Ossa  
Guerra</p>
```

```
<div>
```

```
    <p>Curso de <i>web</i></p>
```

```
    <ul>
```

```
        <li>Martes módulo 2</li>
```

```
        <li>Jueves módulo 2</li>
```

```
        <li>Viernes módulo 4</li>
```

```
    </ul>
```

```
</div>
```

Algunos otros *tags* HTML

- **select** indica que hay un menú de opciones desplegable, cada una contenida por un elemento **option**
- **a** es *anchor* (ancla), referencia otra página
- **br** es un salto de línea, un *break*
- **img** permite incrustar una imagen
- **input** define distintos tipos de *input* interactivos (por ejemplo, *text*, *password*, *button*, *checkbox*, *radio*, etc.)
- **<!-- Esto es un comentario -->**

```
<select>
  <option>Primero</option>
  <option>Segundo</option>
  <option>Tercero</option>
</select>
```

```
<p><a href="mailto:noreply@uc.cl">Enviar
un correo</a></p>
```

```
<p>Una imagen:</p>
```

```
</img>
```

```
<form>
  <input type="text"><br><br>
  <input type="submit">
</form>
```

Referencia de Elementos HTML | MDN: Documentación de MDN sobre elementos HTML

Filter

HTML: Lenguaje de etiquetas de hipertexto

► Beginner's tutorials

► Guides

References

► HTML elements

► Global attributes

► Attributes

► <input> types

Referencia de Elementos HTML

Esta página lista todos los [elementos HTML](#). Están agrupados por funciones para ayudarte a encontrar lo que tienes en mente con facilidad. Aunque esta guía está escrita para aquellos que son nuevos escribiendo código, se pretende que sea una referencia útil para cualquiera.

Nota: Para más información básica acerca de los elementos y atributos HTML, vea [la sección sobre elementos en el artículo 'Introducción a HTML'](#) (inglés).

Raíz principal

Elemento	Descripción
<code><html></code>	El elemento HTML <code><html></code> (o <i>elemento HTML raíz</i>) representa la raíz de un documento HTML. El resto de elementos descienden de este elemento.

Metadatos del documento

Los metadatos contienen información sobre la página. Esto incluye información sobre estilos, *scripts* y datos que ayudan al *software* ([search engines](#) (inglés), [browsers](#), etc.) a usar y generar la página. Los metadatos de estilos y *scripts* pueden estar definidos en la página o estar enlazados a otro fichero que contiene la información.

En este artículo

Raíz principal

Metadatos del documento

Seccionamiento básico

Seccionamiento del contenido

Contenido del texto

Semántica del texto en línea

Imagen y multimedia

Contenido incrustado

SVG and MathML

Scripting

Ediciones demarcadas

Tablas

Formularios

Elementos Interactivos

Componentes Web

Elementos obsoletos y en desuso

Vamos al navegador:

1. Abramos *DevTools* en el navegador
2. Busquemos una *request* que entregue HTML

También podemos ver el código fuente de la página actual desplegada por nuestro navegador (Ctrl+U)

Escribir HTML a partir del contenido: un ejemplo

Introducción

Razones para tomar IIC2513:

Aprender sobre tecnologías web

Aprender sobre aplicaciones web

Hay Mentis

Escribir HTML a partir del contenido: un ejemplo

```
<h2>Introducción</h2>
```

```
<p>Razones para tomar IIC2513:</p>
```

```
<ul><li>Aprender sobre tecnologías web</li>
```

```
<li>Aprender sobre aplicaciones web</li>
```

```
<li>Hay Mentis</li></ul>
```

Escribir HTML a partir del contenido: un ejemplo

```
<h2>Introducción</h2>
```

```
<p>Razones para tomar IIC2513:</p>
```

```
<ul>
```

```
  <li>Aprender sobre tecnologías web</li>
```

```
  <li>Aprender sobre aplicaciones web</li>
```

```
  <li>Hay Mentis</li>
```

```
</ul>
```


Más features de HTML5

Además, con HTML5 llegaron nuevas funcionalidades. Algunas de ellas son:

- Nuevos tags con significado semántico en base a usos comunes, como `<nav>`, `<header>` y `<footer>`
- Soporte para audio y video con los tags `<audio>` y `<video>`
- Soporte para gráficos vectoriales tipo SVG de manera nativa
- Nuevos almacenamientos de información del lado del cliente (`localStorage` y `sessionStorage`)
- Introducción de la API para WebSockets

```
<!DOCTYPE html>
<html>

<head>
  <title>Título de la página</title>
</head>

<body>
  <article>
    <header>
      <h1>Título del artículo.</h1>
      <h4>Un sub-heading.</h4>
      <p>Descripción o algo así.</p>
    </header>
  </article>
</body>

</html>
```

Más features de HTML5

Además, con HTML5 llegaron nuevas funcionalidades. Algunas de ellas son:

- Nuevos tags con significado semántico en base a usos comunes, como `<nav>`, `<header>` y `<footer>`
- Soporte para audio y video con los tags `<audio>` y `<video>`
- Soporte para gráficos vectoriales tipo SVG de manera nativa
- Nuevos almacenamientos de información del lado del cliente (`localStorage` y `sessionStorage`)
- Introducción de la API para WebSockets

```
<!DOCTYPE html>
<html>

<head>
  <title>Título de la página</title>
</head>

<body>
  <article>
    <header>
      <h1>Título del artículo.</h1>
      <h4>Un sub-heading.</h4>
      <p>Descripción o algo así.</p>
    </header>
  </article>
</body>

</html>
```

Más features de HTML5

Además, con HTML5 llegaron nuevas funcionalidades. Algunas de ellas son:

- Nuevos tags con significado semántico en base a usos comunes, como `<nav>`, `<header>` y `<footer>`
- Soporte para audio y video con los tags `<audio>` y `<video>`
- Soporte para gráficos vectoriales tipo SVG de manera nativa
- Nuevos almacenamientos de información del lado del cliente (`localStorage` y `sessionStorage`)
- Introducción de la API para WebSockets

Fuente: [Top 10 New Features of HTML5 - GeeksforGeeks](#)

```
<!DOCTYPE html>
<html>

<head>
  <title>Título de la página</title>
</head>

<body>
  <h2>Ejemplo de video y audio</h2>

  <video controls autoplay>
    <source src="x.ogg" type="video/ogg" />
    <source src="x.mp4" type="video/mp4" />
    Navegador no soporta elemento video.
  </video>

  <audio controls autoplay>
    <source src="a.ogg" type="audio/ogg" />
    <source src="a.wav" type="audio/wav" />
    Navegador no soporta elemento audio.
  </audio>
</body>

</html>
```


Más features de HTML5

Además, con HTML5 llegaron nuevas funcionalidades. Algunas de ellas son:

- Nuevos tags con significado semántico en base a usos comunes, como `<nav>`, `<header>` y `<footer>`
- Soporte para audio y video con los tags `<audio>` y `<video>`
- Soporte para gráficos vectoriales tipo SVG de manera nativa
- Nuevos almacenamientos de información del lado del cliente (`localStorage` y `sessionStorage`)
- Introducción de la API para `WebSockets`

```
<!DOCTYPE html>
<html>

<head>
  <title>Título de la página</title>
</head>

<body>
<h2>Ejemplo de soporte para SVG</h2>

<svg width="100" height="100">
  <circle cx="50" cy="50"
    r="40"
    stroke="black"
    stroke-width="3"
    fill="red" />
</svg>
</body>

</html>
```

Agenda

HTML para estructura

✨ **CSS para presentación**

Qué hace el navegador

CSS (presentación)

CSS (*Cascading Style Sheets*) es un **lenguaje especializado para diseño gráfico** (tampoco es lenguaje de programación) que define el “estilo”: indica cómo debe verse un elemento particular

Es un complemento a HTML, para implementar la presentación, visualización y diseño de páginas web. Para esto, ocupamos reglas CSS:

```
h1 {  
  color: green;  
}
```

IIC2513

Profesor: Antonio Ossa Guerra

Clases

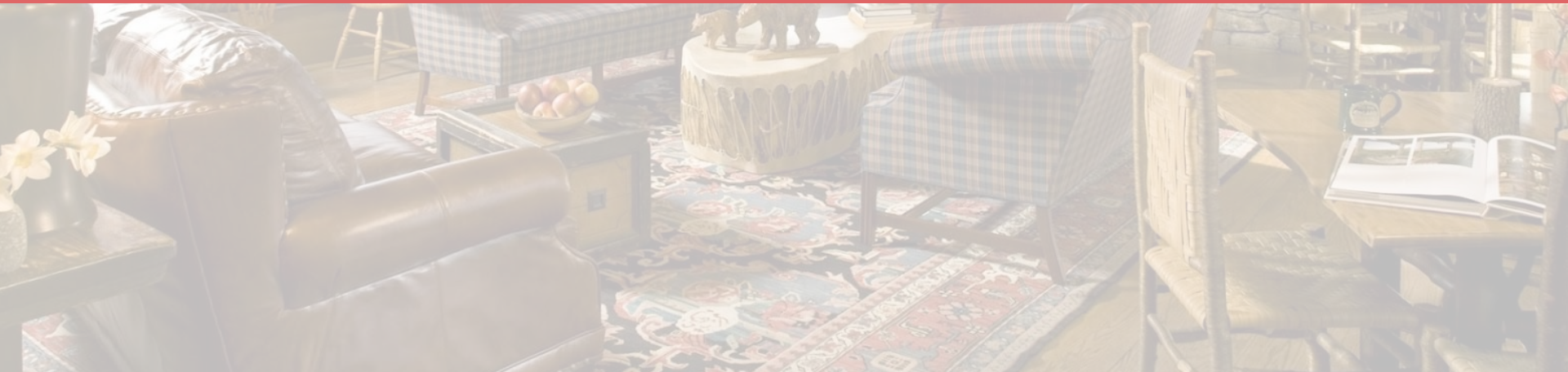
1. Bienvenida al curso
2. De la URL al servidor
3. HTML y CSS
4. JavaScript
5. ...

Brevísima historia de CSS

- 1994 ● Creación de The World Wide Web Consortium ([W3C](#))
- 1996 ● **CSS1** Primera especificación de la W3C, incluyendo propiedades sobre *fonts*, colores, atributos de texto, y alineamiento y posicionamiento
- 1998 ● **CSS2** Superset de CSS1 agregando distintos tipos de posicionamiento, los conceptos de z-index y media types, y más
- 1999+ ● **CSS3** Se divide en “módulos” que agregan o extienden funcionalidad definida en CSS2. Estos siguen en trabajo y desarrollo al día de hoy
- 2007+ ● **CSS4** No hay una sola versión porque el *spec* está dividido en varios módulos con distintos niveles (ej. Flexbox)
- 2012 ● **CSS Flexbox** estable y soportado por navegadores
- 2016 ● **CSS Grid** estable y soportado por navegadores
- ... ● Nuevos módulos y *features* publicados constantemente



Intro a CSS



Anatomía de un elemento

En una estructura llamada **regla**, podemos determinar cómo se deberán mostrar en el navegador los elementos HTML seleccionados

```
selector {  
    propiedad: valor;  
}
```

En una misma regla podemos utilizar **múltiples declaraciones**, modificando varios valores de propiedad a la vez

```
p {  
    width: 500px;  
    border: 1px solid black;  
}
```

Y también podemos **utilizar varios selectores a la vez**, aplicando una misma regla a todos los elementos que coincidan con alguno de ellos

```
p, li, h1 {  
    color: red;  
}
```

Selectores CSS

Con selectores podemos referenciar uno o varios elementos HTML y, aunque hay varios tipos de selectores, los más comunes son:

- elemento
- #id
- .clase
- [atributo]
- :pseudoclase

```
h1 {  
    color: red;  
}
```

```
#identificador {  
    background-color: red;  
}
```

```
.planta {  
    color: green;  
}
```

```
li[class^="a"] {  
    font-size: 200%;  
}
```

```
a:hover {  
    color: red;  
}
```

Cómo insertar CSS en un documento HTML

Existen 3 maneras distintas de insertar CSS en un documento HTML para indicar las reglas a aplicar sobre dichos elementos:

- **CSS externo:** Con el elemento `<link>` (dentro de la sección *head*) se puede referenciar una hoja de estilos externa (un archivo) con los estilos a aplicar
- **CSS interno:** Con el elemento `<style>` se pueden indicar las reglas de CSS que aplican para toda la página actual
- **CSS inline:** Permite indicar el estilo a aplicar para un elemento particular en el documento HTML como parte de uno de sus atributos (el atributo `style=`)

CSS externo

Aplicar CSS de manera externa es bastante común para aplicar estilos compartidos por varias páginas, ya que solo referenciamos un archivo

Con el elemento `<link>` (dentro de la sección *head*) se puede referenciar una hoja de estilos externa (un archivo) con los estilos a aplicar

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="style.css">
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

```
<link rel="stylesheet" href="mi-estilo.css"/>
```

CSS interno

Puede ser útil a la hora de aplicar estilo a una página particular, aunque en general solo se usa si no podemos modificar directamente los archivos CSS externos

Con el elemento **<style>** se pueden indicar las reglas de CSS que aplican para toda la página actual

```
<style>
h1 {
    color: red;
}
</style>
```

```
<!DOCTYPE html>
<html>
  <head>
    <title>Mi experimento CSS</title>
    <style>
      h1 {
        color: blue;
        background-color: yellow;
        border: 1px solid black;
      }

      p {
        color: red;
      }
    </style>
  </head>
  <body>
    <h1>¡Hola, mundo!</h1>
    <p>Este es un ejemplo de CSS</p>
  </body>
</html>
```

CSS inline

En general es mala práctica desarrollar así, ya que dificulta la mantenibilidad, obliga a duplicar/repetir código y mezcla HTML (estructura) con CSS (presentación)

Permite indicar el estilo a aplicar para un elemento particular en el documento HTML como parte de uno de sus atributos (el atributo **style=**)

```
<p style="color:red;">Texto rojo</p>
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Mi experimento CSS</title>
  </head>
  <body>
    <h1 style="color:
blue;background-color: yellow;border: 1px
solid black;">
      ¡Hola mundo!
    </h1>
    <p style="color:red;">Este es mi
primer ejemplo de CSS</p>
  </body>
</html>
```

¿En qué orden se aplican los estilos?

Nuestro navegador tomará todas reglas de estilo que definamos y las va a aplicar, en estilo cascada (la C en CSS) con distintos niveles de prioridad:

1. Las reglas más prioritarias son las de **CSS *inline***
2. Luego se aplican las de **CSS externo e interno** (todo lo que está en head)
3. Como última prioridad, se usa un **estilo por defecto** definido por el navegador

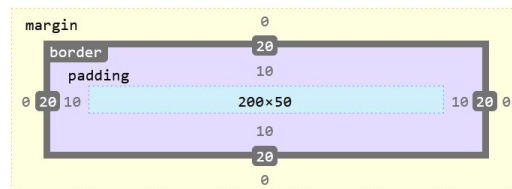
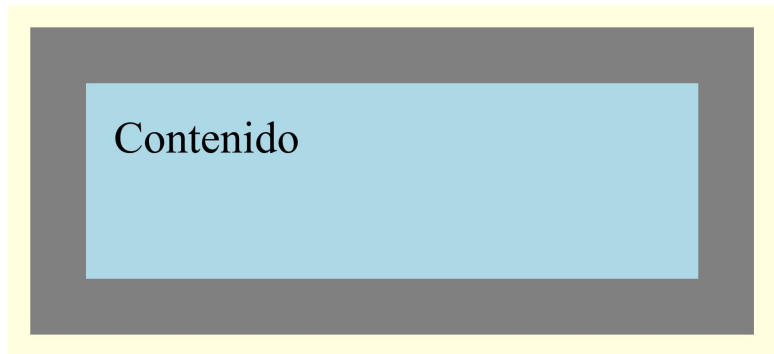
Cualquier regla *inline* sobrescribe reglas de CSS externo e interno, las que a su vez sobrescriben cualquier valor *default* del navegador

Modelo de caja

Este modelo es una **caja que envuelve todo elemento HTML**. Desde afuera hacia adentro, consiste en: margen, borde, relleno, y el contenido...

... aunque en CSS usamos inglés:

- *margin*: distancia entre exterior y el borde del elemento
- *border*: la frontera visible del elemento
- *padding*: distancia entre contenido y borde



260×110

static

▼ Box Model Properties

box-sizing

content-box

line-height

normal

display

block

position

static

float

none

z-index

auto

Vamos al navegador:

1. Abramos *DevTools* en el navegador
2. Modifiquemos los estilos de una página apagando, prendiendo y editando propiedades

¿Cómo probar cosas en HTML y CSS?

- a. Usando un archivo local
- b. Usando un servidor local como el de Python (viene por defecto):
`python -m http.server`
- c. [Codepen](#)

Agenda

HTML para estructura

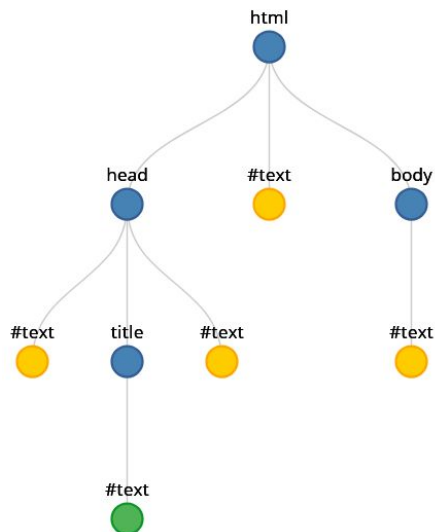
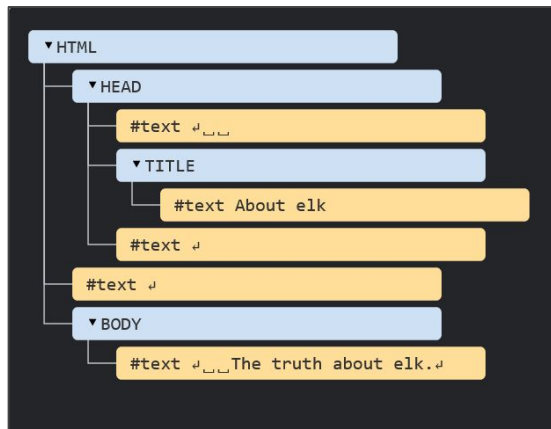
CSS para presentación

✨ **Qué hace el navegador**

HTML DOM

La representación interna de un documento HTML se llama **DOM (document object model)**. El DOM es un modelo jerárquico, basado en un árbol, donde *tags* y todo elemento en el HTML son representados como nodos, construyéndose según la jerarquía de la página desde el *tag* html. Estandarizado por la W3C ([ver estándar](#))

```
<!DOCTYPE HTML>
<html>
<head>
  <title>About elk</title>
</head>
<body>
  The truth about elk.
</body>
</html>
```



HTML DOM

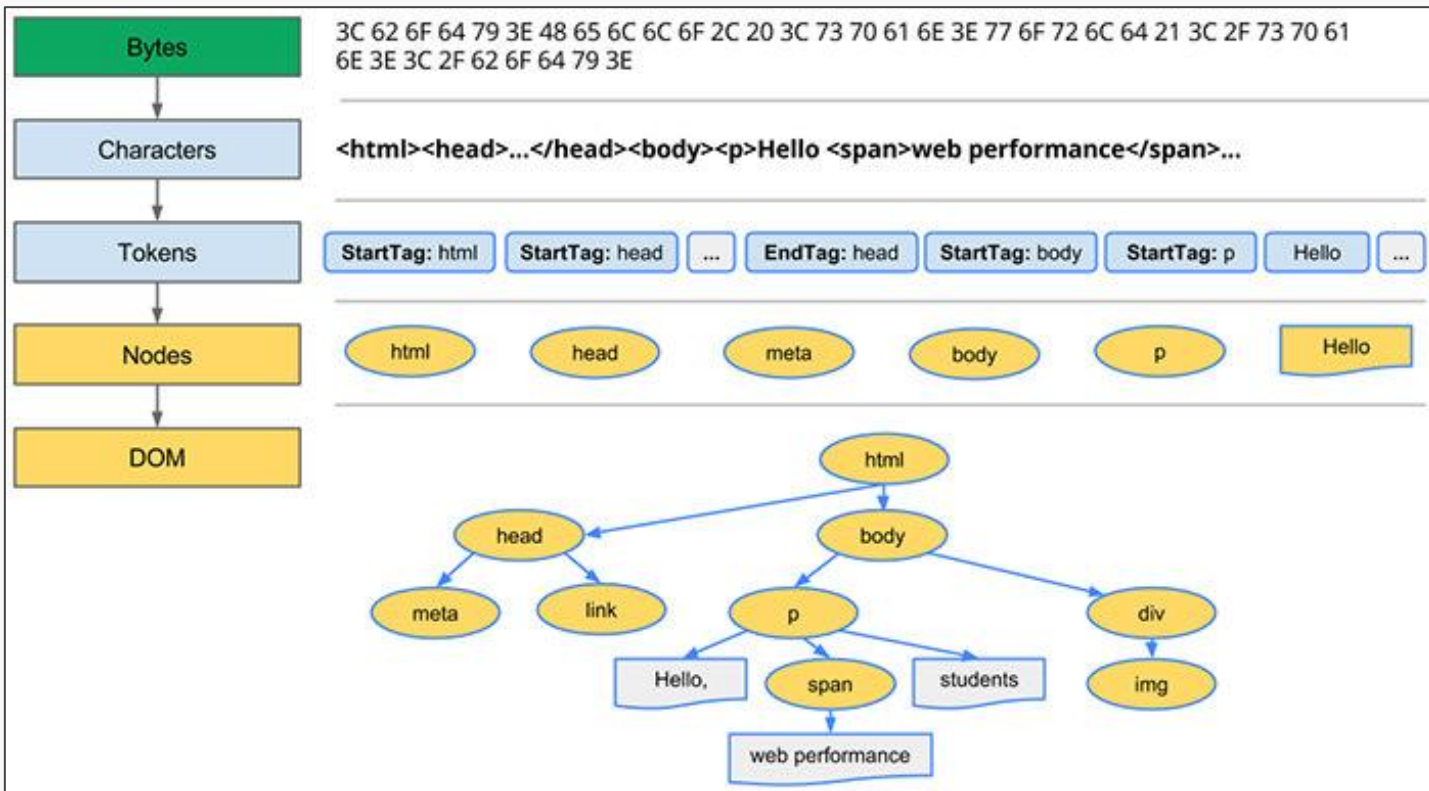
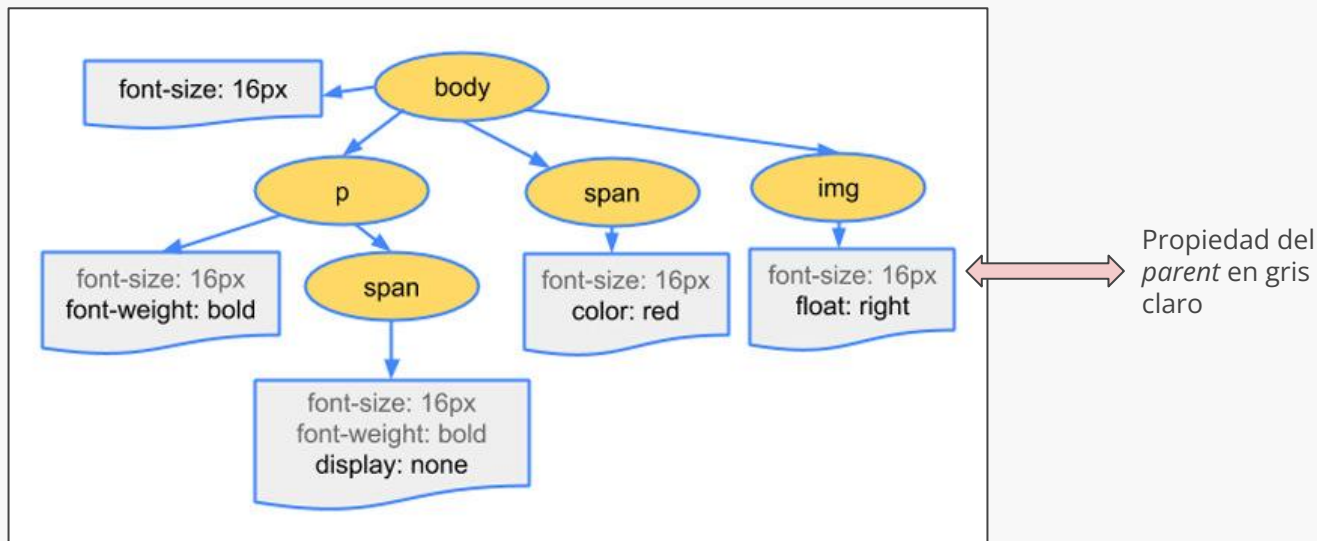


Figura: [The Beginners Guide to CSS Object Model \(CSSOM\)](#) | Hongkiat

CSSOM

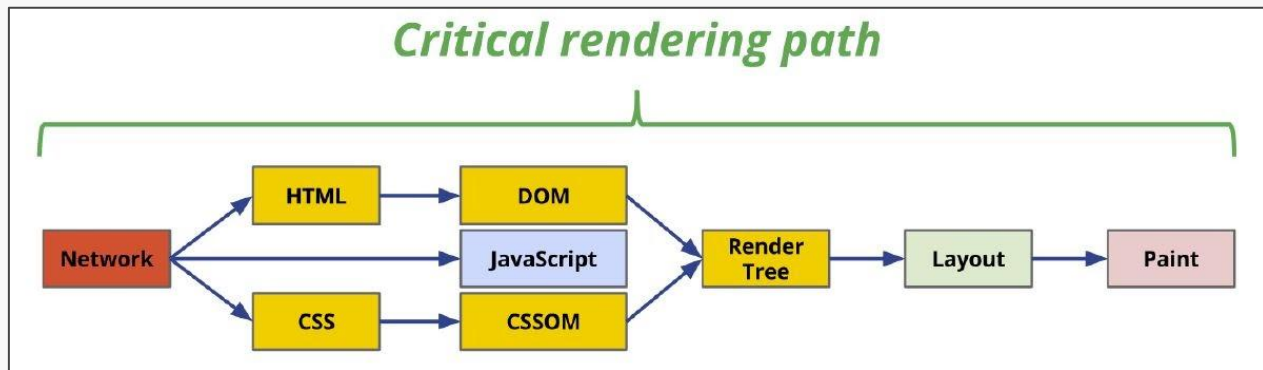
Así como existe el DOM para HTML, existe **CSSOM (CSS Object Model)** para CSS. Este modelo toma todo el código CSS y arma un árbol (nuevamente) con todos los selectores que definamos, permitiendo el funcionamiento de “cascada” de CSS y que se pueda interactuar con CSS desde JavaScript. También estandarizado por la W3C ([ver spec](#))



Critical rendering path

Después de todas las *requests* y calcular el DOM y CSSOM, ocurren más cosas:

- **Render tree:** reconciliación de DOM y CSSOM, solo incluyendo partes visibles
- **Layout:** calcula las posiciones y tamaños exactos de cada elemento (aquí es donde entra en acción el “box model” que vimos)
- **Paint:** el verdadero dibujado de los elementos, a nivel de píxeles, para ser mostrados en pantalla



Sobre navegadores...

Al igual que con los editores de código, un navegador es una herramienta para lograr un objetivo. Si bien el funcionamiento de mecanismo a utilizar está estandarizado, los navegadores implementan y soportan funcionalidades de manera levemente distinta

Lo más importante es **elegir un navegador con el que sientan comodidad y familiaridad** para su uso diario (incluyendo su rol de desarrolladores). Esto incluye desde el uso básico hasta *keyboard shortcuts* y personalizaciones



Google Chrome



Microsoft Edge



Mozilla Firefox



Apple Safari

HTML y CSS: Recursos recomendados

Para comenzar...

- [Conceptos básicos de HTML | MDN](#): Documentación de MDN para entender y conocer los conceptos relacionados a HTML por medio de ejemplos
- [HTML Basic Examples | W3Schools](#): Descripción y ejemplos de código para múltiples elementos y *tags* HTML
- [CSS básico - Aprende desarrollo web | MDN](#): Documentación de MDN para entender y conocer los conceptos básicos relacionados con CSS

HTML y CSS: Recursos adicionales

Temas que veremos más adelante...

- [Conceptos básicos de flexbox | MDN](#): Documentación sobre Flexbox
- [CSS Grid Layout | MDN](#): Documentación de MDN sobre Grid Layout
- [Bootstrap](#): Librería con estilos por defecto para incorporar en HTML/CSS
- [Awwwards](#): Sitio para conocer y explorar tendencias en diseño web

Agenda

HTML para estructura


CSS para presentación

Qué hace el navegador

✨ **Actividad 1**

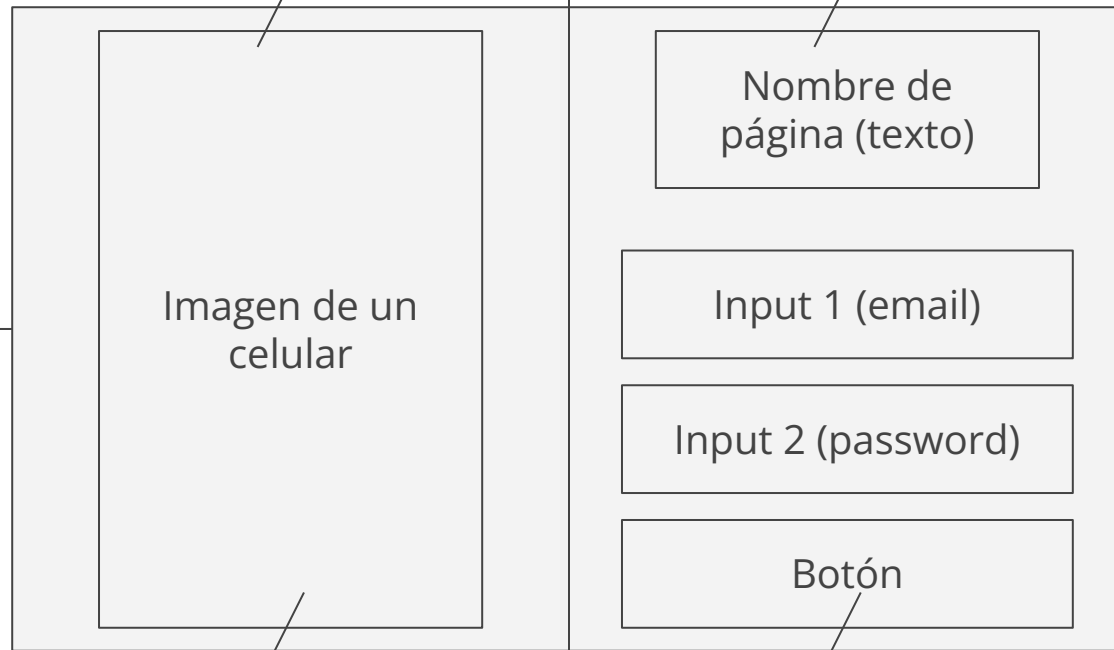
Desafío (hasta las 23:59 de hoy en Canvas)

1. Hacer página visualmente similar al *log in* de Instagram:

- a. Imagen de celular (más alto que ancho: )
- b. Nombre de la página
- c. Dos entradas de texto y botón de *log in* (usar tag form e *input types* apropiados)
- d. Todo centrado apropiadamente (en vertical y horizontal)
- e. Solo usar HTML y CSS puro (no JavaScript, no dependencias externas)

2. Recuerden sus opciones:

- a. Usar un archivo local
- b. Usar el servidor de Python: `python -m http.server`
- c. [Codepen](#)



Agenda

HTML para estructura

CSS para presentación

Qué hace el navegador

✨ **¿Qué se viene pronto?**

Próximas fechas importantes: clases y ayudantías

	Clases y ayudantías						
	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado	Domingo
Semana 1 (03/3 - 09/3)				Presentación del curso			
Semana 2 (10/3 - 16/3)		Desde el navegador a mi sitio web		HOY			
Semana 3 (17/3 - 23/3)		JavaScript pt. 1 (Intro)		JavaScript pt. 2 (DOM y async)	Ayudantía: JavaScript		
Semana 4 (24/3 - 30/3)		React		Exploración y consumo de APIs	Ayudantía: React		
Semana 5 (31/3 - 06/4)		Node.js		Desarrollando servidores	Ayudantía: Consumo de APIs		

Próximas fechas importantes: evaluaciones

	Evaluaciones						
	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado	Domingo
Semana 1 (03/3 - 09/3)							
Semana 2 (10/3 - 16/3)				HOY			
Semana 3 (17/3 - 23/3)							
Semana 4 (24/3 - 30/3)	Control 1				Inicio T1 React		
Semana 5 (31/3 - 06/4)					Fin T1 (React)		

Clase 2

HTML y CSS

IIC2513 - Tecnologías y Aplicaciones Web

Antonio Ossa Guerra
aaossa@ing.puc.cl