«Talento Tech»

Automation Testing

Clase 5

Buenos Aires Ciudad

Clase N° 5: Introducción a HTML y Estructura de Páginas Web

Temario

- Anatomía de un documento HTML
- Elementos HTML comunes (div, form, input, button, etc.)
- Atributos esenciales para automatización (id, class, name, value)
- Introducción a CSS
    - Estilos en línea ↔ bloque <style> ↔ archivo externo
    - Selectores básicos (por etiqueta, id, clase)
    - Especificidad y jerarquía de selectores
- DevTools a fondo
    - Inspección de elementos y edición en vivo
    - Copia de selectores y validación en consola
    - Modo dispositivo y otras utilidades
- Buenas prácticas de selectores

Objetivos de la clase

En esta clase exploraremos los fundamentos del desarrollo web con foco en su aplicación para la automatización de pruebas. Comenzaremos reconociendo la estructura mínima de un documento HTML y comprendiendo el propósito de sus principales secciones como `<head>` y `<body>`. Aprenderemos a identificar los elementos más frecuentes en una página y su uso habitual, así como la importancia de los atributos `id`, `class` y `name`, claves para interactuar con elementos desde herramientas como Selenium. También abordaremos las distintas formas de aplicar CSS y por qué optar por un archivo externo es la mejor práctica en proyectos reales. Finalmente, utilizaremos DevTools para inspeccionar, copiar y probar selectores, y generaremos un mini-documento de referencia que nos servirá como base para automatizar pruebas de interfaz.

Antes de automatizar, entendamos cómo está hecha la web

Antes de poder automatizar una interfaz, necesitamos comprender cómo está construida. **HTML** define la estructura de la página —es como el esqueleto del sitio—, mientras que **CSS** se encarga del diseño visual: colores, tamaños, tipografías y disposición.

En esta clase vamos a explorar ambos lenguajes para poder "leer" correctamente una página web. Además, aprenderemos a usar las **DevTools del navegador**, una herramienta fundamental que actúa como tu lupa digital: te permite inspeccionar el código fuente, probar estilos en vivo y entender cómo se comportan los elementos en distintas resoluciones.

Anatomía de un documento HTML

HTML (HyperText Markup Language) es el lenguaje que define la estructura de una página web.

Cada página tiene un "esqueleto" hecho con etiquetas que indican qué es cada parte: un título, un párrafo, un formulario, etc.

Ejemplo mínimo:

```
<!DOCTYPE html>
<html lang="es">
 <head>
   <meta charset="UTF-8" />
   <title>Página de ejemplo</title>
 </head>
 <body>
   <h1>¡Hola TalentoLab!</h1>
   <p>Esta es nuestra primera página.</p>
 </body>
</html>
```

- **`<!DOCTYPE html>`**: indica al navegador que usará HTML5.
- **`<head>`**: contiene metadatos, hojas de estilo y scripts.
- **`<body>`**: todo lo que el usuario ve e interactúa.

Elementos HTML comunes

Etiqueta	Uso principal	Mini-ejemplo
`<div>`	Contenedor genérico	`<div class="tarjeta">…`
`<form>`	Agrupa campos de entrada	`<form action="/alta">…`
`<input>`	Campo de texto, checkbox, etc.	`<input type="email" name="correo" />`
`<button>`	Botón de acción	`<button>Enviar</button>`
`<a>`	Enlace a otra URL	`<a href="/inicio">Inicio</a>`
`<ul> / <li>`	Listas	`<ul><li>Item</li></ul>`

💡**Nota**: Aunque existen etiquetas semánticas (header, nav, section), muchas apps aún utilizan como ladrillo base. Por eso verás muchos selectores apuntando a ellos.

Para más conocer mas información, etiquetas y sus usos ingresá a:
👉 MDN Developer Mozilla

Atributos esenciales

Atributo	¿Para qué sirve?	Ejemplo
`id`	Identificador único en la página. Perfecto para selectores robustos.	`<input id="email" />`
`class`	Agrupa elementos que comparten estilo o función. Un elemento puede tener varias clases.	`<div class="alert error">`
`name`	Nombre que recibe el backend al enviar el formulario; Selenium también lo usa.	`<input name="password" />`
`value`	Valor inicial o etiqueta de un control.	`<button value="guardar">Guardar</button>`

💡**Consejo: elige `id` siempre que sea estable**. Evitarás selectores frágiles que se rompan con cambios de diseño.

Introducción a CSS

CSS “pinta” el HTML. Sin él todo sería texto negro. En otras palabras define el aspecto visual de una página web, como colores, fuentes, márgenes, espaciados y otros elementos de diseño. Tres modos de aplicarlo:

Estilos en línea (pruebas rápidas)

```
<h1 style="color: steelblue; font-size: 24px;">Título en línea</h1>
```

✅ Bueno para pruebas rápidas

❌ Malo para mantenimiento

Bloque `<style>` incrustado

```
<head>
        <style>
          .boton {
            background: #0275d8;
            color: #fff;
            padding: 8px 16px;
            border: none;
          }
        </style>
</head>
```

✅ Útil en demos o prototipos

❌ No recomendado en proyectos reales

Archivo externo (mejor práctica)

Separar CSS permite cachear estilos, reutilizarlos en varias páginas y mantener el HTML limpio.

Acá vamos a tener dos archivos, el index.html y el estilos.css. Desde el index vamos a referenciar al estilos.css por medio de esta linea de codigo: `<link rel="stylesheet" href="estilos.css" />`

✅ Separar estructura (HTML) de estilo (CSS) mejora el mantenimiento, trabajo en equipo y rendimiento.

Ejemplo completo:

index.html

```
<!DOCTYPE html>
<html lang="es">
 <head>
   <meta charset="UTF-8" />
   <title>Página de ejemplo</title>
   <head>
     <link rel="stylesheet" href="estilos.css" />
   </head>
 </head>
 <body>
   <h1>¡Hola TalentoLab!</h1>
   <p>Esta es nuestra primera página.</p>
 </body>
</html>
```

estilos.css

```
#principal {
   color: steelblue;
   font-family: 'Segoe UI', Arial, sans-serif;
 }
  .boton {
   background: #0275d8;
   color: #fff;
   padding: 8px 16px;
   border: none;
   border-radius: 4px;
```

```
    cursor: pointer;
}
```

Selectores básicos en CSS

Cuando escribes CSS, necesitas “decirle” al navegador **qué** elemento(s) quieres estilizar. Para eso usas *selectores*. Los tres más usados —y los primeros que debes dominar— son:

1. Selector por etiqueta

estilos.css

```
h1 {
    color: midnightblue;
    font-size: 2rem;
    margin-bottom: 12px;
}
```

- **¿Qué hace?** Afecta **a todos los elementos `<h1>`** que existan en la página.
- **Cuándo usarlo:** cuando deseas que todos tus títulos principales compartan la misma apariencia sin tener que asignarles ninguna clase o id.

html

```
<h1>Primer título</h1>
<h1>Segundo título</h1>
<!-- Ambos h1 aparecerán en azul oscuro y con el mismo tamaño -->
```

2. Selector por id

css

```
#principal {
    color: tomato;
    text-transform: uppercase;
}
```

- **¿Qué hace?** Estiliza **solo al elemento** que tenga `id="principal"`.
- **Regla importante:** un `id` debe ser único en todo el documento; usarlo dos veces rompe esa unicidad.

html

```
<h1 id="principal">Título destacado</h1>

    <!-- Este h1 se pintará de rojo y en mayúsculas; ningún otro elemento se verá afectado -->
```

3. Selector por clase

css

```
.boton {
   background: #0275d8;
   color: #fff;
   padding: 8px 16px;
   border: none;
   border-radius: 4px;
   cursor: pointer;
 }
```

- **¿Qué hace?** Aplica el estilo **a todos los elementos** que tengan `class="boton"`.
- **Ventaja:** puedes reutilizar la misma clase en decenas de botones, enlaces o divs.

```
<button class="boton">Guardar</button>
<button class="boton">Cancelar</button>
<!-- Ambos botones comparten el mismo look →
```

💡**Tip:** los elementos pueden tener **varias clases** (por ejemplo `class="btn btn-primary"`), lo que te permite mezclar estilos como bloques de Lego.

¿Cuál gana si varios coinciden? — Especificidad

Si un mismo elemento coincide con varios selectores, el navegador elige el más "específico":

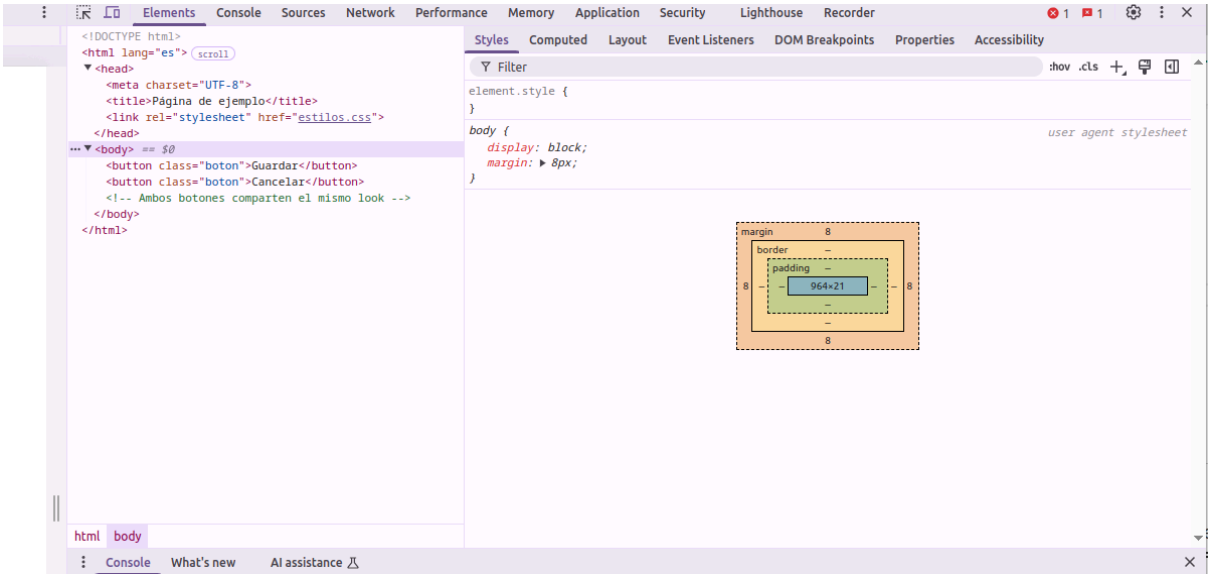
`Etiqueta <  Clase .  <  Id #`

Es decir, un estilo definido con `#id` sobreescribirá al de `.clase`, y ambos sobreescribirán al de `h1` (o cualquier otra etiqueta). Conocer esta jerarquía te evitará sorpresas cuando un cambio parezca "no hacer efecto".

Con estos tres selectores ya puedes comenzar a dar estilo a tus páginas y, lo que es más importante para nosotros como testers, a **identificar** elementos de forma estable cuando escribamos scripts de automatización.

Herramientas de desarrollo del navegador (DevTools)

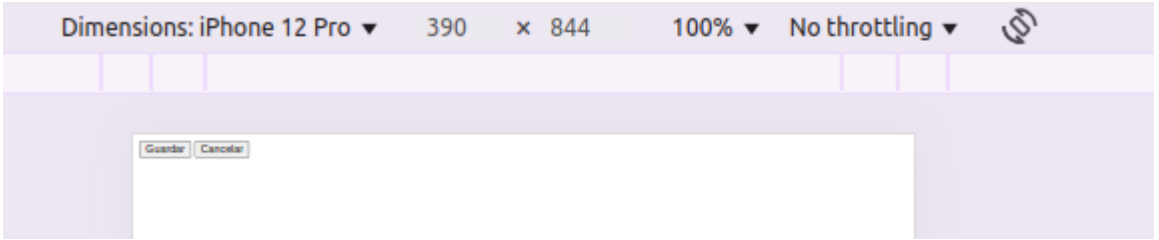
¡Tu microscopio para ver el DOM en vivo! Ábrelo con **F12** o clic derecho → *Inspeccionar*.



Pestañas principales

- **Elements**: árbol del DOM y estilos aplicados.
- **Styles**: reglas CSS; desmarca para ver cambios inmediatos.
- **Console**: ejecuta JS y revisa errores.
- **Network**: solicitudes HTTP (excelente para API debugging).
- **Performance** y **Lighthouse**: métricas de carga y accesibilidad.

Modo dispositivo



- Haz clic en el icono 📱 para emular móvil / tablet.
- Cambia entre presets (`iPhone 12 Pro`, `Pixel 6`) o define tu tamaño.
- Gira la pantalla y ajusta DPR (zoom real) con un clic.

> *Esto te ayuda a validar que los selectores y estilos sean consistentes en vista móvil.*

Inspección y copia de selectores

Ejemplo breve

1. En `https://example.com`, inspecciona el `<h1>`.
2. Copia selector CSS → suele ser `body > div > h1`.

Cambia su color en vivo:

```
document.querySelector('body > div > h1').style.color = 'red';
```

3. Si se vuelve rojo, tu selector es correcto.

Buenas prácticas de selectores

- **Usa `id`** cuando exista.
- Prefiere clases semánticas (`.form-login .btn-primary`).
- Evita rutas largas y `nth-child` a menos que no haya alternativa.

🗨️**Storytelling**

El equipo de Talento Lab va a mostrar la nueva calculadora web en la demo interna de la semana próxima.
Silvia necesita que el formulario esté listo para ser automatizado con Selenium y te pide:

"Quiero un HTML limpio, sin JavaScript, con todos los atributos `id` y `name` únicos. Además, dejame un documento con los selectores que usarás en tus scripts."

Matías añade:

"Olvidate de la lógica por ahora; ya la validamos con Pytest.
Solo maquetá el front y anotá buenos selectores. Después integraremos Selenium."

Ejercicio Práctico

Objetivos:

- Construir la vista estática de la calculadora (HTML + CSS externo).
- Etiquetar cada campo con identificadores estables (`id`, `name`, clases semánticas).
- Practicar DevTools: inspeccionar elementos, copiar selectores y comprobar que apunten a unívocos.
- Crear `selectors.md` — tu futura hoja de ruta para los tests UI.

Entregables y estructura de carpetas

```
Clase05/
|  index.html        ← estructura HTML 100 % estática
|  estilos.css       ← hoja de estilos externa
|  selectors.md      ← tabla de selectores (Markdown)
└─ README.md         ← breve guía de ejecución y verificación
```

Requisitos funcionales

Sección	Detalles
Encabezado	`<h1 id="titulo-principal">Calculadora Modular</h1>` + un párrafo introductorio.
Formulario	`<form id="form-calculadora" action="#" method="post">` con: • Input número 1 id="num1" name="num1" • Input número 2 id="num2" name="num2" • Radio group operación (sumar, restar, multiplicar, dividir) todos con name="operacion" y valores claros. • Botón enviar id="btn-calcular".
Estilos mínimos	– Centrar el bloque en la pantalla. – Tipografía sistema. – Botón azul (#0275d8) con :hover más oscuro. – Bordes redondeados y ligera sombra al contenedor principal.
Accesibilidad	Cada <input> con su <label> asociado vía for.
Sin JS	El botón puede enviar el formulario o no hacer nada (action="#").

Pasos sugeridos:

1. **Maqueta rápida:**

   Crea un archivo `index.html` con la estructura mínima de una página web:

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Formulario de prueba</title>
    <link rel="stylesheet" href="estilos.css">
</head>

<body>
    <!-- Aquí irá tu formulario -->
</body>
</html>
```

2. **Hoja de estilos:**

Agregá un archivo `estilos.css` con estilos básicos (reset, colores, clases utilitarias, etc.). Podés usar una plantilla base o una que hayas trabajado previamente.

3. **Abrir con DevTools:**
   - Desde el navegador, presioná `Ctrl + O` y abrí el archivo `index.html`.
   - Luego, usá `F12` o clic derecho → *Inspeccionar* para acceder a las DevTools.

4. **Verificación de controles:**
   - Inspeccioná cada input, botón o campo del formulario.
   - Asegurate de que cada uno tenga un atributo `id` **único**.
   - Si un elemento no tiene `id`, agregalo y guardá los cambios.

5. **Test rápido de selectores CSS:**
   Copiá el selector CSS de cada elemento y probalo en la consola del navegador con este comando:

```
document.querySelector('#num1').style.border = '2px solid red';
```

> 🔴 Si el borde se pinta de rojo, el selector es correcto.

6. **Documentación de selectores (`selectors.md`):**

Generá un archivo Markdown con una tabla que registre los selectores relevantes:

Elemento	Atributo usado	Selector CSS
Campo Número 1	`id`	`#num1`
Campo Email	`name`	`input[name="email"]`
Botón Enviar	`class`	`.btn.enviar`
Checkbox Términos	`id`	`#acepta-condiciones`
Error debajo de email	`class`	`.error-msg.email`

Buenos Aires Ciudad