IIC2026
Ayudantía 3 - DOM, eventos y SVG

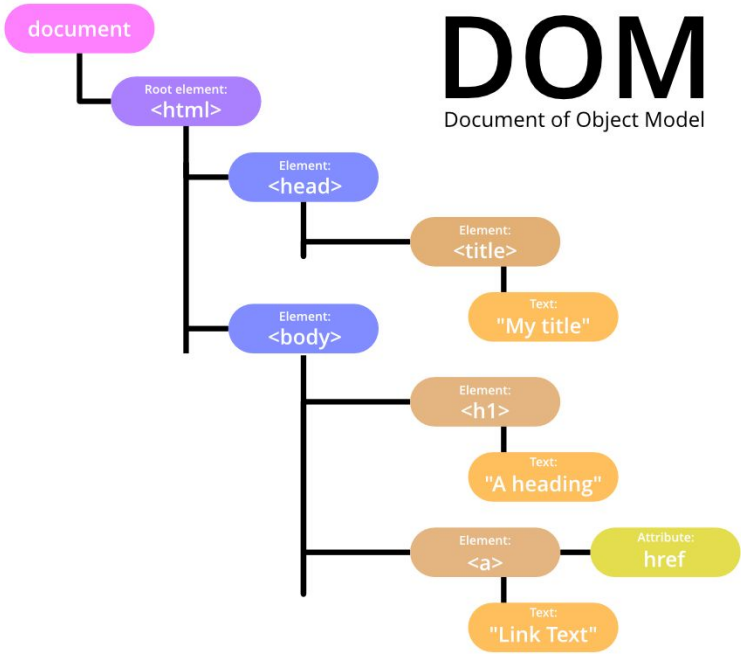
Javiera Azócar y Camila Basulto

Document Object Model (DOM)

¿Qué es el DOM?

Forma de representar el contenido de un documento HTML como objeto de programación.

```
<!DOCTYPE html>
<html>
 <head>
   <title>My Title</title>

</head>

 <body>
   <h1>A heading</h1>
   <a href="...">Link text</a>


 </body>
</html>
```



Variable document

Con la variable document podremos acceder a todo tipo información del HTML:

```
// Imprimimos el link de donde está el documento
console.log(document.URL);

// Obtenemos la lista de hijos del documento
document.children; // [<html>]

// Obtenemos elemento HTML con id="uwu"
const elemento = document.getElementById("hola");


// Creamos un elemento cuyo tag es <p>
const parrafo = document.createElement("p");
```

¿Cómo creamos un elementos?

```
// Creamos un elemento cuyo tag es <p>
const parrafo = document.createElement("p");


// Creamos un elemento tipo texto
const texto = document.createTextNode("¡Soy un texto >.<!");


// Agregamos el texto al párrafo
parrafo.appendChild(texto);
```

Clases, id y atributos

```
// Creamos un elemento con tag <img>
const elemento = document.createElement("img");

// Definimos la clase del elemento
elemento.className = "importante";

// Definimos un único id al elemento
elemento.id = "principal";


// Definimos un atributo al elemento
elemento.setAttribute("width", 200)
```

SVG

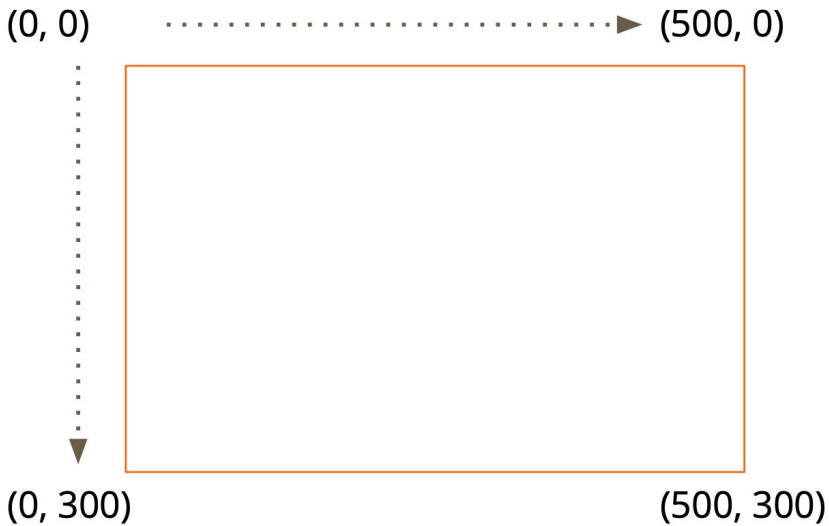
SVG

Scalable **V**ector **G**raphics

- Formato de imágenes bidimensionales **vectorial**.
    - Como no son pixeles, la calidad es mucho mayor.
- Define su contenido a partir de objetos con propiedades.
- Muy compatible con HTML.

SVG - Formato

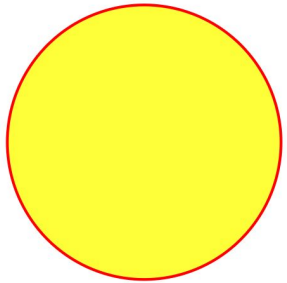
```
<svg width="500" height="300">

... </svg>
```



SVG - Círculo

```
<svg width="500" height="300">
   <circle cx="100" cy="100" r="50" fill="#FFFF00" color="red" radius="10000" />
</svg>
```

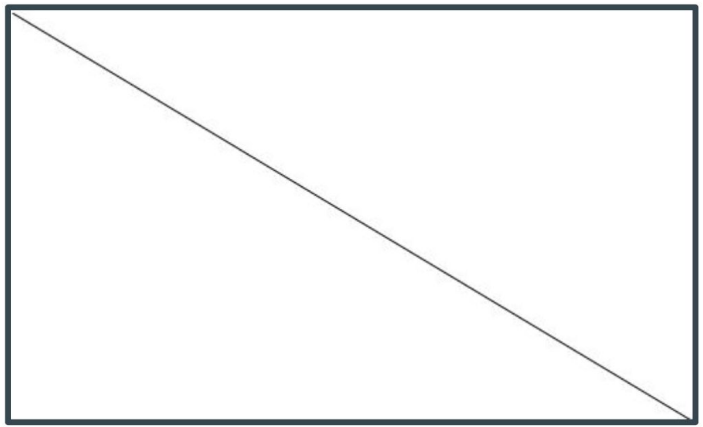
- Tag <circle/> para crear un **círculo**.
- cx="100" → Indica el centro en el eje X.
- cy="100" → Indica el centro en el eje Y.
- r="50" → Indica radio del círculo.
- fill="#0000FF" → Indica color de relleno (azul).
- color="magenta" → **No existe**.
- radius="10000" → **No existe**.



SVG - Línea

```
<svg width="500" height="300">
    <line x1="0" y1="0" x2="500" y2="300" stroke="black"/>
</svg>
```

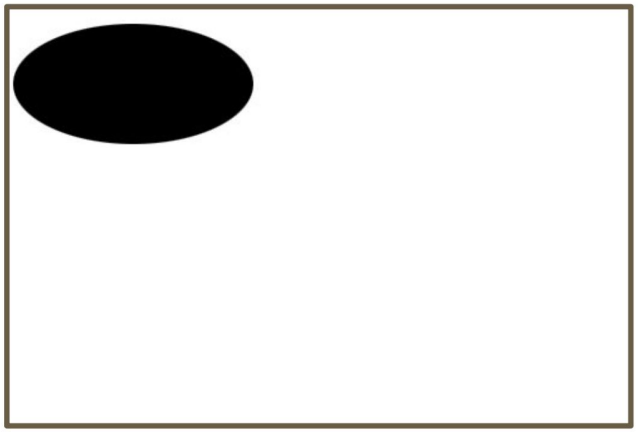
- Tag <line/> para crear una **línea**.
- x1="0" → Indica el inicio de una recta en el eje X.
- y1="0" → Indica el inicio de una recta en el eje Y.
- x2="y2" → Indica fin inicio de una recta en el eje X.
- y2="300" → Indica el fin de una recta en el eje Y.
- stroke="black" → Color de la recta.



SVG - Elipse

```
<svg width="500" height="300">
    <ellipse cx="100" cy="50" rx="100" ry="50"/>
</svg>
```

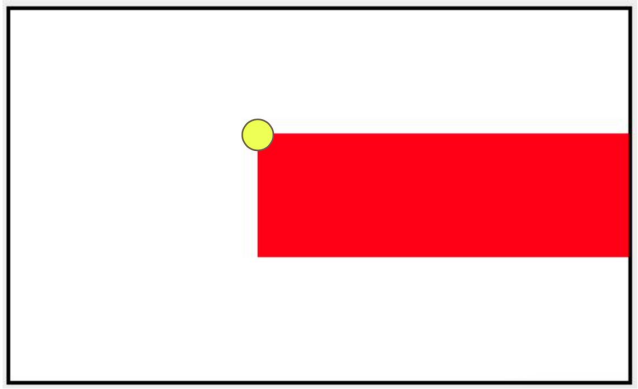
- Tag <ellipse/> para crear una **elipse**.
- cx="100" → Indica el centro en el eje X.
- cy="50" → Indica el centro en el eje Y.
- rx="100" → Indica el radio de la elipse en el eje X.
- ry="50" → Indica el radio de la elipse en el eje Y.



SVG - Rectángulo

```
<svg width="500" height="300">
    <rect x="200" y="100" width="300" height="100" fill="red" color="magenta" />
</svg>
```

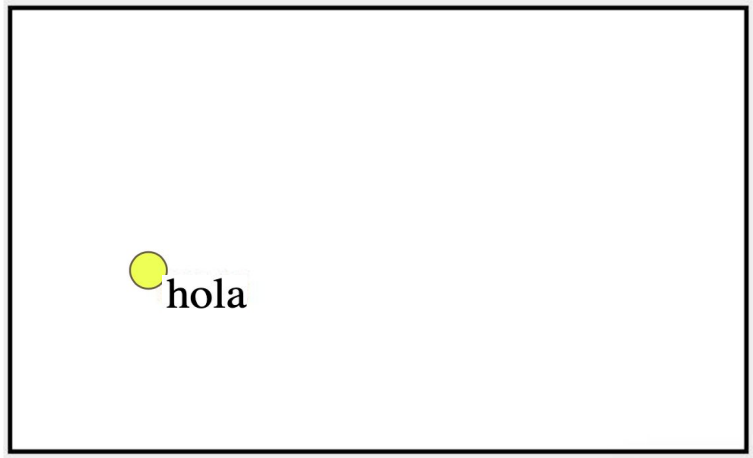
- Tag <rect/> para crear un **rectángulo**.
- x="200" → Indica donde estará la esquina izquierda superior en el eje X (O).
- 7="300" → Indica donde estará la esquina izquierda superior en el eje Y (O).
- width="300" → Indica el ancho.
- height="100" → Indica el alto.
- fill="red" → Indica el color de relleno.



SVG - Texto

```
<svg width="500" height="300">
  <text x="100" y="200" font-size="34" size="1">hola </text>
</svg>
```

- Tag <text/> para crear un **texto**.
- Dentro del tag se pone el texto a mostrar.
- x="100" → Indica donde estará la esquina izquierda superior en el eje X (0).
- y="200" → Indica donde estará la esquina izquierda superior en el eje Y (0).
- font-size="34" → Indica el tamaño del texto.
- size="1" → No existe.



Eventos

Posibles eventos de HTML

↳ de mouse:

`onclick`: Se dispara cuando el usuario hace click en un elemento.

`ondblclick`: Se dispara cuando el usuario hace doble click en un elemento.

`onmousedown`: Se dispara cuando el usuario presiona un botón del ratón sobre un elemento.

`onmouseup`: Se dispara cuando el usuario suelta un botón del ratón sobre un elemento.

`onmouseover`: Se dispara cuando el cursor del ratón se coloca sobre un elemento o cualquiera de sus hijos.

`onmouseout`: Se dispara cuando el cursor del ratón se mueve fuera de un elemento específico, incluso a un hijo.

`onmousemove`: Se dispara cuando el usuario mueve el ratón mientras está sobre un elemento.

`onmouseenter`: Se dispara cuando el cursor del ratón entra en un elemento.

`onmouseleave`: Se dispara cuando el cursor del ratón sale de un elemento completamente.

↳ de teclado/keyboard:

`onkeydown` : Se dispara cuando se presiona una tecla.

`onkeyup` : Se dispara cuando se suelta una tecla.

`onkeypress` : Se dispara cuando el usuario presiona una tecla que produce un carácter. (deprecated/obsoleto)

Posibles eventos de HTML

↳ de formulario/form:

`onfocus`: Se dispara cuando un elemento (por ejemplo, un campo de entrada) gana el foco.

`onblur`: Se dispara cuando un elemento pierde el foco.

`onchange`: Se dispara cuando el valor de un elemento de formulario cambia.

`oninput`: Se dispara cada vez que cambia el valor de un campo de entrada.

`onsubmit`: Se dispara cuando se envía un formulario.

`onreset`: Se dispara cuando se restablece un formulario.

↳ de ventana/page

`onload`: Se dispara cuando la página ha terminado de cargar.

`onunload`: Se dispara cuando el usuario abandona la página.

`onresize`: Se dispara cuando se cambia el tamaño de la ventana del navegador.

`onscroll`: Se dispara cuando el usuario desplaza la página o un elemento.

`onbeforeunload`: Se dispara antes de que el documento sea descargado (cerrado o recargado).

`onhashchange`: Se dispara cuando cambia la parte del hash en la URL.

Posibles eventos de HTML

↳ de portapapeles/clipboard:

`oncopy`: Se dispara cuando el usuario copia contenido.
`oncut`: Se dispara cuando el usuario corta contenido.
`onpaste`: Se dispara cuando el usuario pega contenido.

↳ de teclado/keyboard:

`ondrag`: Se dispara cuando un elemento es arrastrado.
`ondragstart`: Se dispara al iniciar el arrastre de un elemento.
`ondragend`: Se dispara al terminar el arrastre de un elemento.
`ondragenter`: Se dispara cuando un elemento arrastrado entra en una zona de destino.
`ondragleave`: Se dispara cuando un elemento arrastrado sale de una zona de destino.
`ondragover`: Se dispara cuando un elemento arrastrado está sobre una zona de destino.
`ondrop`: Se dispara cuando un elemento arrastrado se suelta en una zona de destino.

Posibles eventos de HTML

↳ otros eventos

`onerror`: Se dispara cuando ocurre un error durante la carga de un archivo multimedia.
`ontouchstart`: Se dispara cuando un dedo toca la pantalla (para dispositivos táctiles).
`ontouchmove`: Se dispara cuando un dedo se mueve sobre la pantalla (para dispositivos táctiles).
`ontouchend`: Se dispara cuando un dedo deja de tocar la pantalla (para dispositivos táctiles).
`oncontextmenu`: Se dispara cuando se abre un menú contextual.

Hay muchos eventos para hacer todo tipo de cosas!!

Pero... en el curso nos centraremos en algunos pocos relacionados a interactuar con las visualizaciones, como los eventos de mouse y formulario.

Vamos a ver un ejemplo de cómo utilizar estos eventos. Los archivos **eventos.html** y **eventos.js** ya están subidos en Canvas.

IIC2026
Ayudantía 3 - DOM, eventos y SVG

Javiera Azócar y Camila Basulto