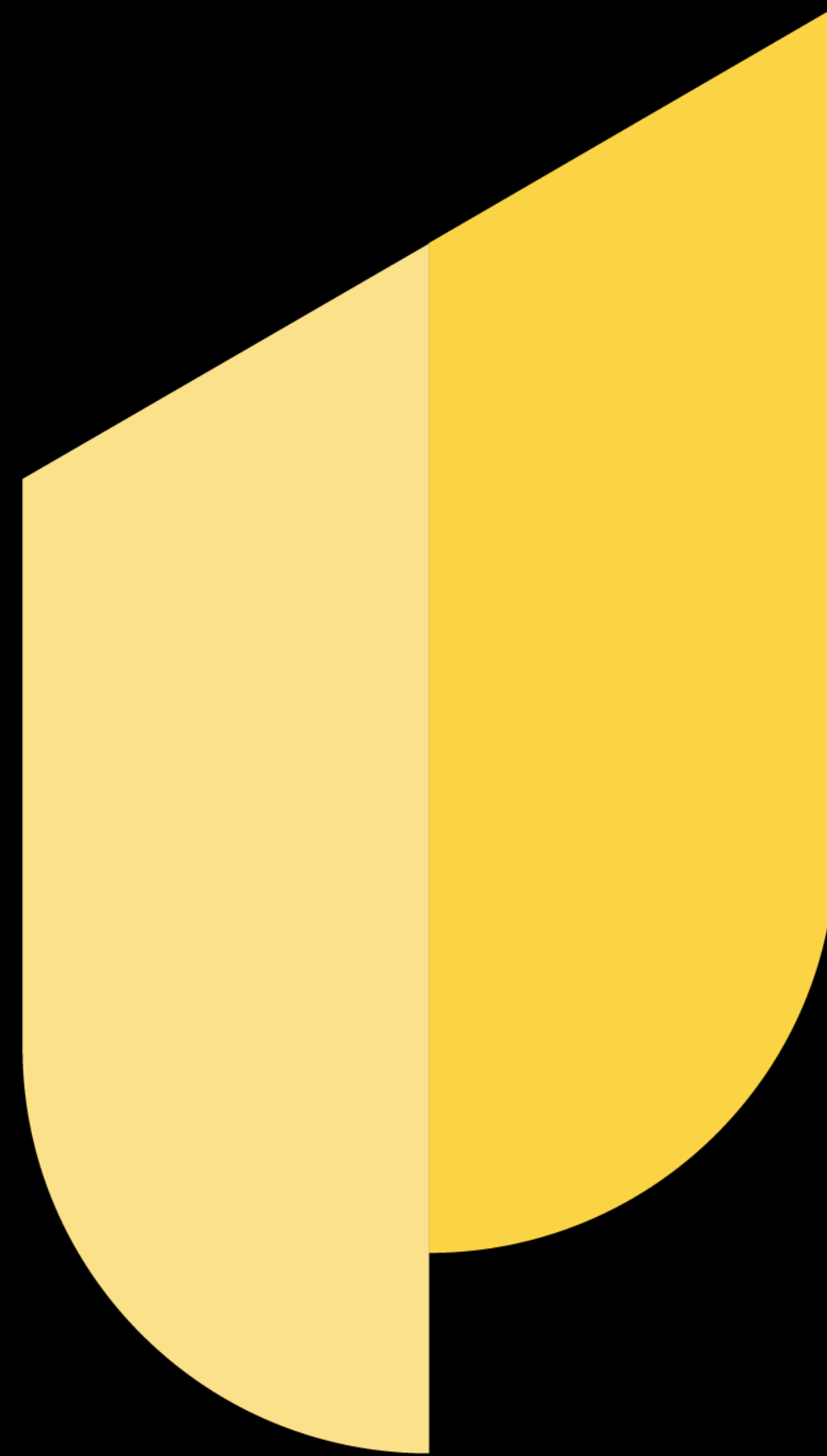


Ingeniería en Informática / Ingeniería en Sistemas de Información / Ingeniería de Software

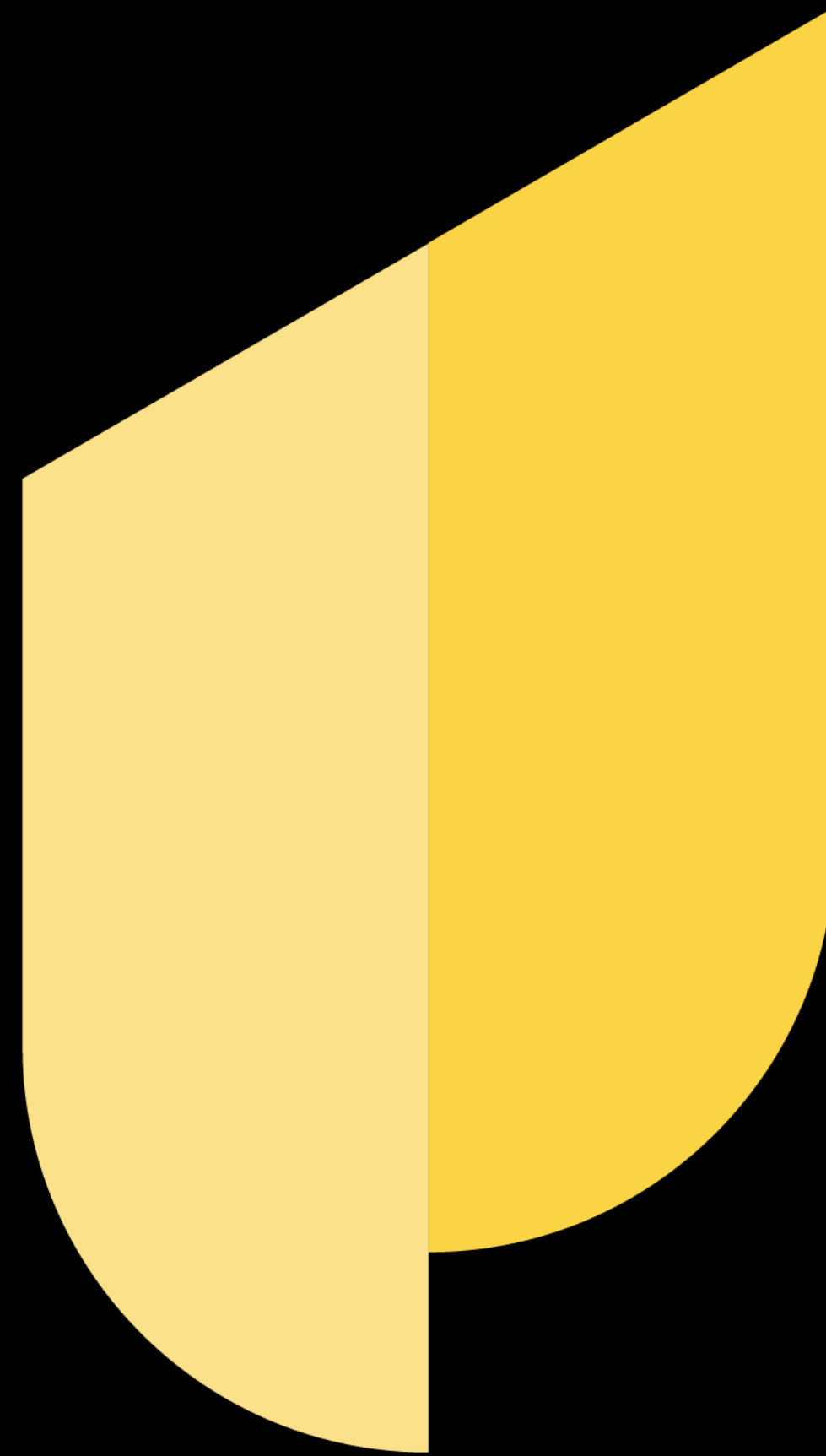
Programación III



JavaScript



DOM



¿Qué es el DOM?

- El modelo de objeto de documento (DOM) es una interfaz de programación para los documentos HTML y XML.
- Facilita una representación estructurada del documento y define de qué manera los programas pueden acceder, al fin de modificar, tanto su estructura, estilo y contenido.
- El DOM da una representación del documento como un grupo de nodos y objetos estructurados que tienen propiedades y métodos. Esencialmente, conecta las páginas web a scripts o lenguajes de programación.

- Una página web es un documento. Éste documento puede exhibirse en la ventana de un navegador o también como código fuente HTML. En ambos casos, es el mismo documento.
- El modelo de objeto de documento (DOM) proporciona otras formas de presentar, guardar y manipular este mismo documento.
- El DOM es una representación completamente orientada al objeto de la página web y puede ser modificado con un lenguaje de script como JavaScript.



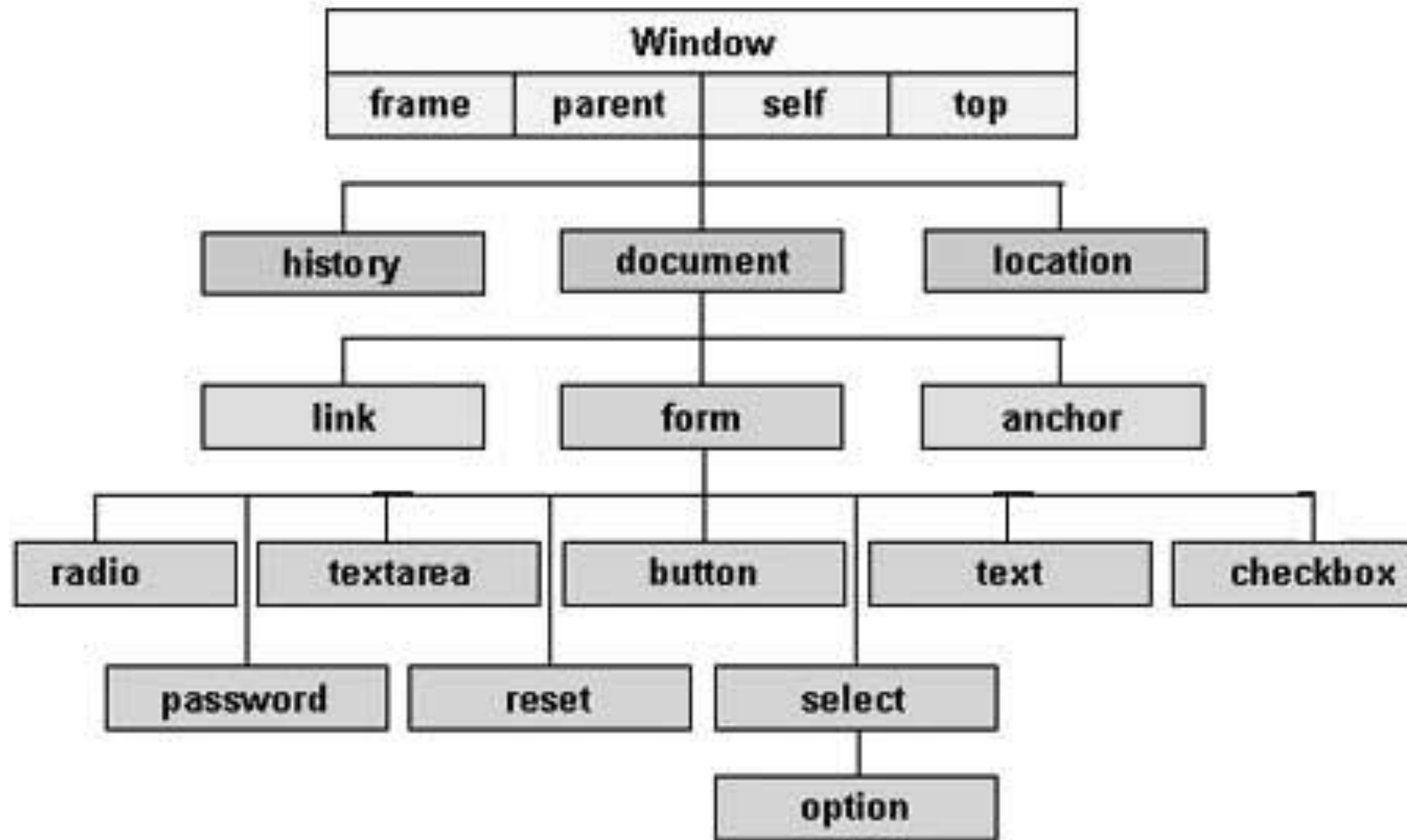
- El [W3C DOM](#) estándar forma la base del funcionamiento del DOM en muchos navegadores modernos.
- Por ejemplo, el DOM de W3C especifica que el método **getElementsByTagName** en el código de abajo debe devolver una lista de todos los elementos <p> del documento:

```
paragraphs = document.getElementsByTagName ("p");  
// paragraphs[0] es el primer elemento <p>  
// paragraphs[1] es el segundo elemento <p>, etc.  
alert (paragraphs [0].nodeName);
```

- Todas las propiedades, métodos y eventos disponibles para la manipulación y la creación de páginas web está organizado dentro de objetos.
- Un ejemplo: el objeto document representa al documento mismo, el objeto table hace funcionar la interfaz especial HTMLTableElement del DOM para acceder a tablas HTML, y así sucesivamente.



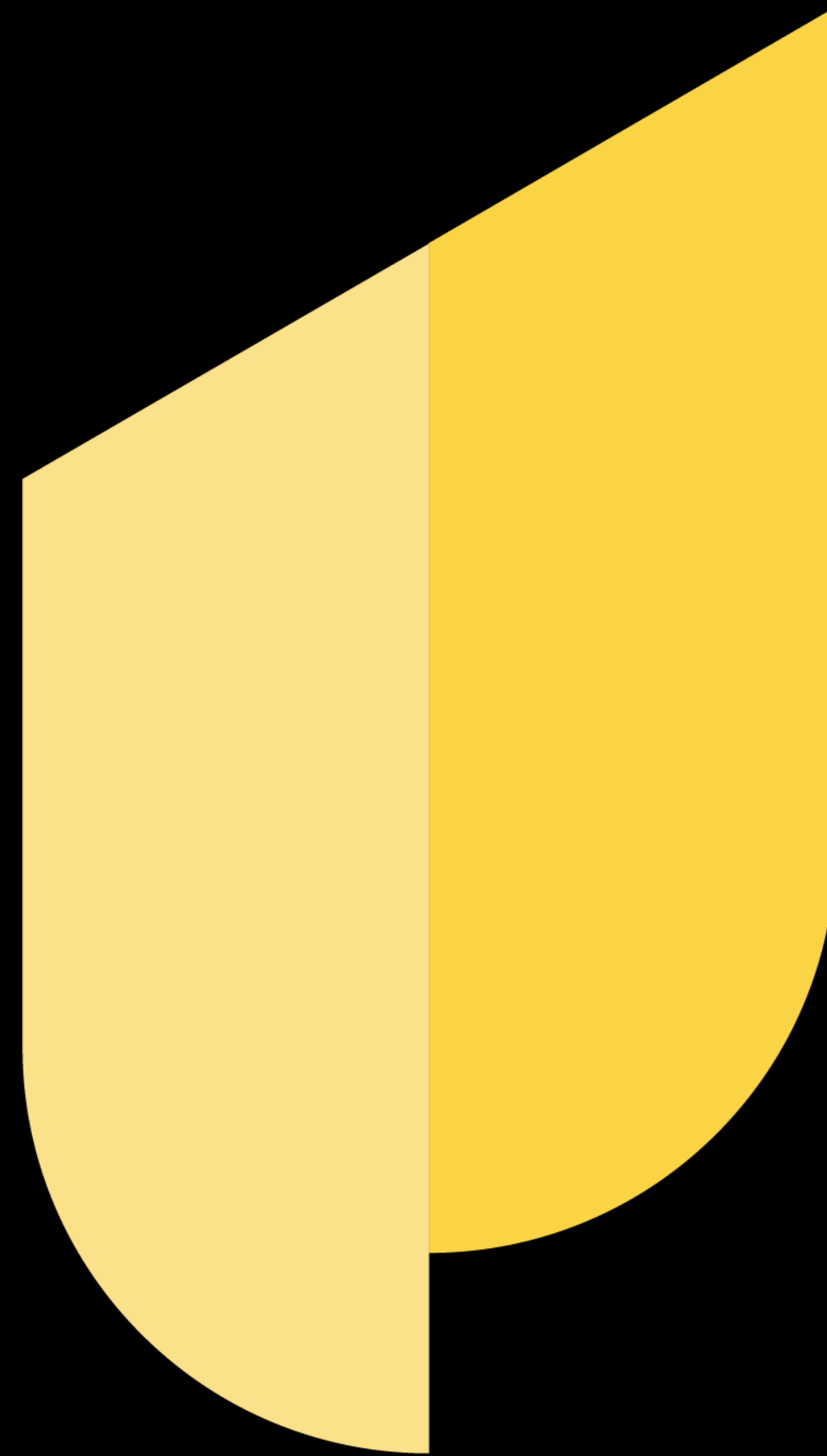
Representación del DOM



JavaScript y DOM

- El DOM no es un lenguaje de programación pero sin él, el lenguaje JavaScript no tiene ningún modelo o noción de las páginas web, de la páginas XML ni de los elementos con los cuales es usualmente relacionado.

JavaScript



Seleccionar
nodos

Métodos actuales

- `document.querySelector('css-selectors')`
- `document.querySelectorAll('css-selectors', ...)`



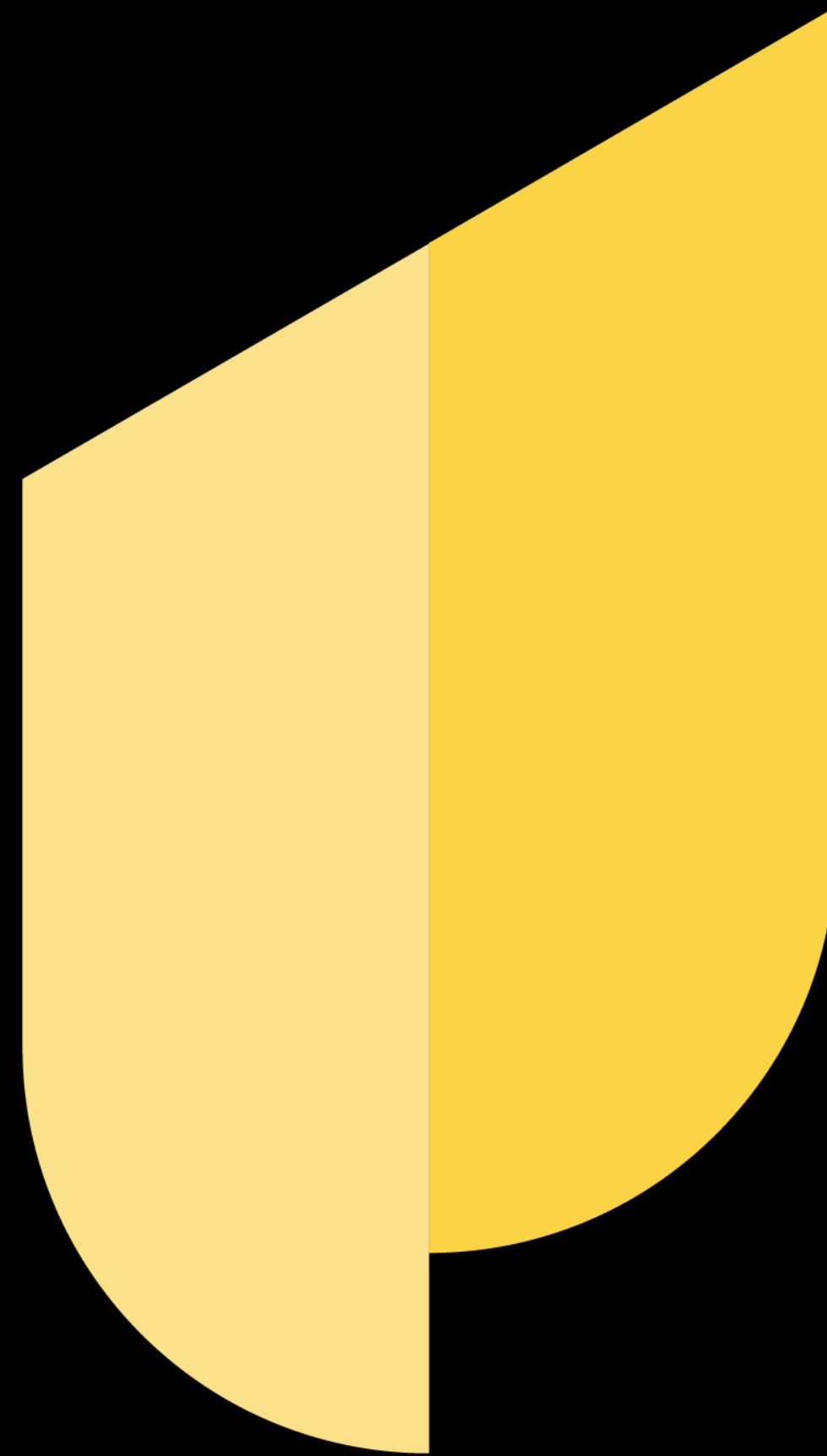
Métodos anteriores

- `document.getElementsByTagName('element-name')`
- `document.getElementsByClassName('class-name')`
- `document.getElementById('id')`

Obtener detalles del elemento

- `node.nextSibling`
- `node.firstChild`
- `node.lastChild`
- `node.parentNode`
- `node.childNodes`
- `node.children`

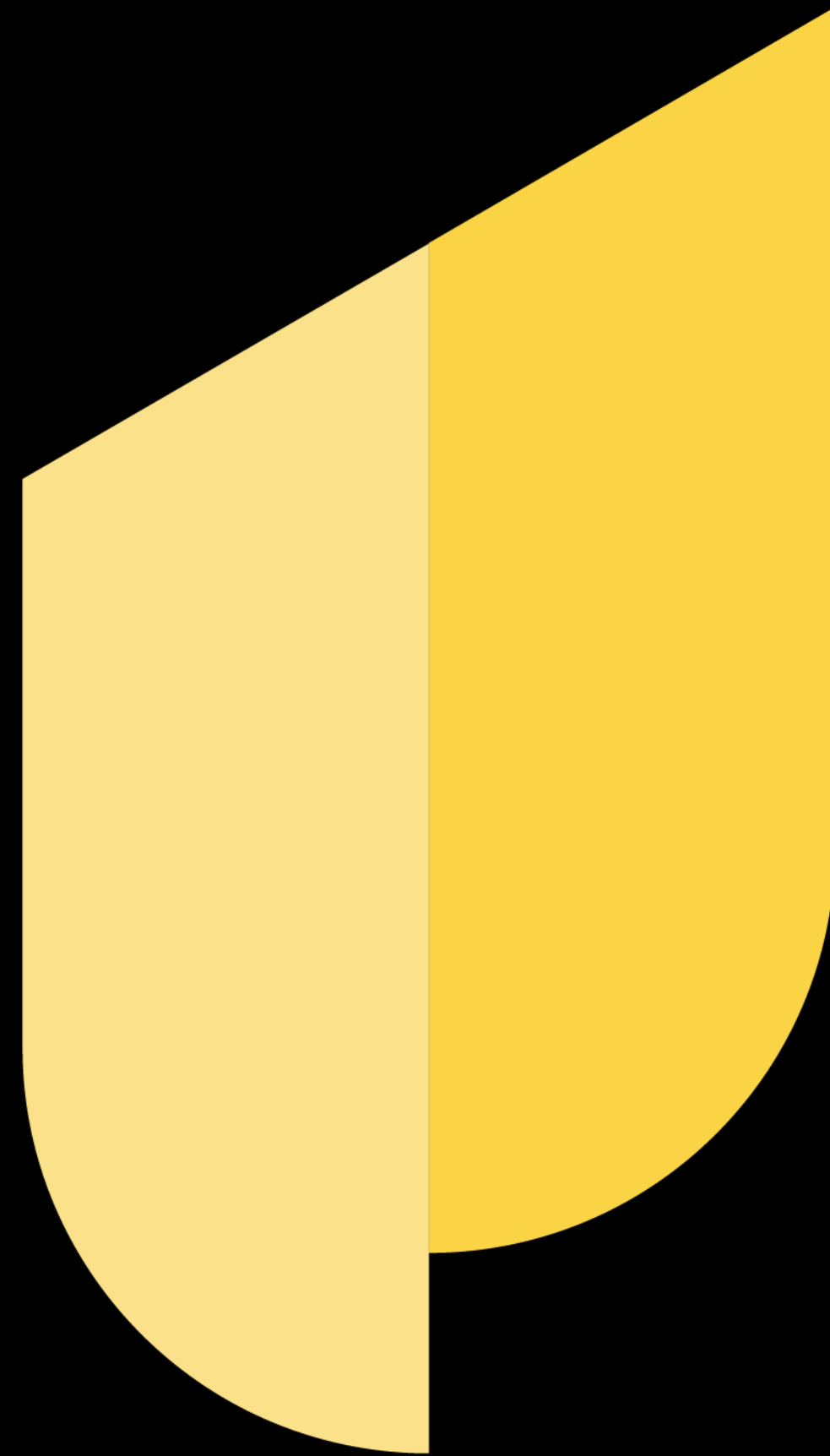
JavaScript



Modificar contenido
de un nodo

- `node.innerHTML = "valor";`

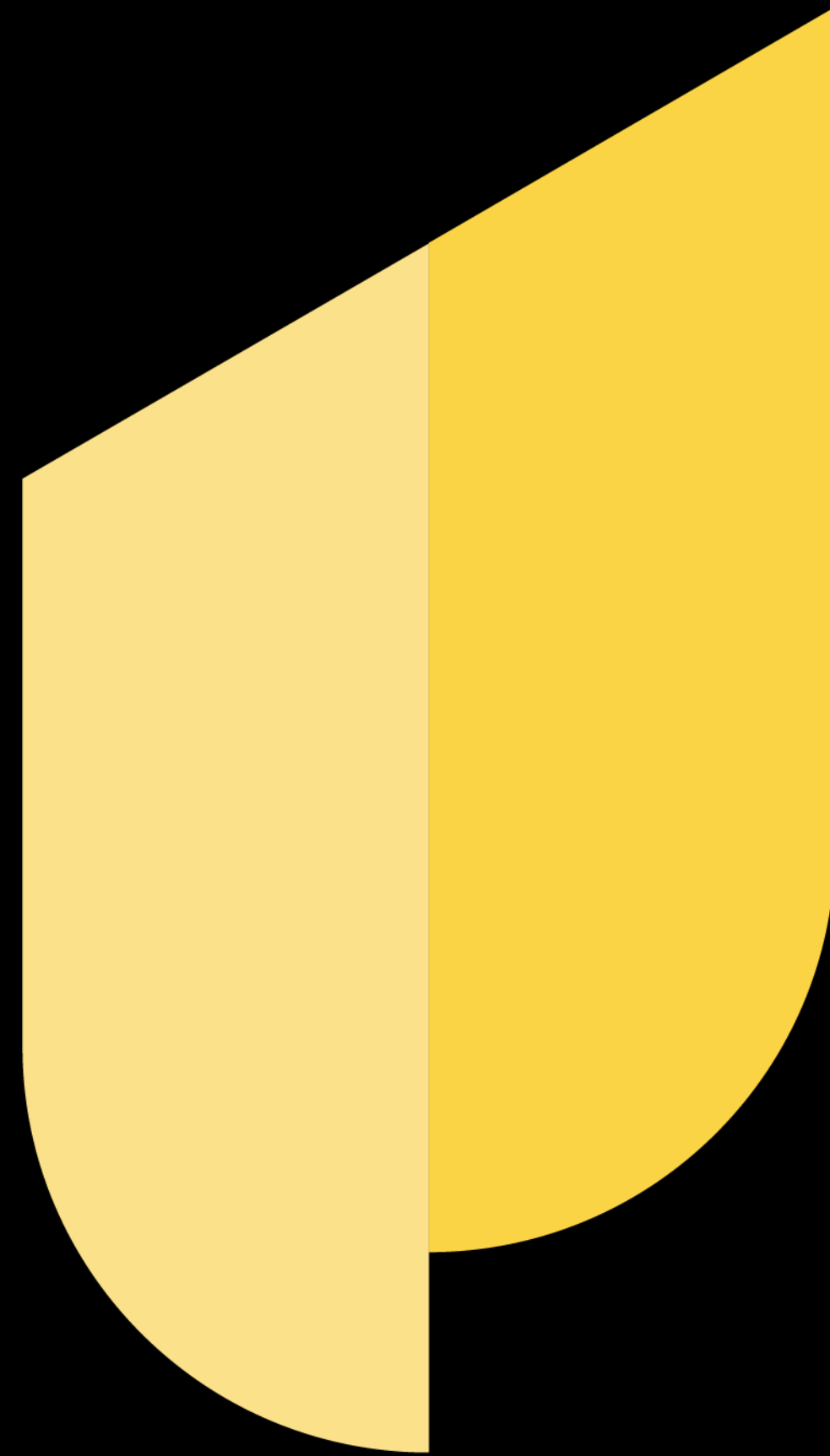
JavaScript



Modificar atributos
de un nodo

- `Node.nombreAtributo=valor;`
- Ejemplos:
 - `Node.src="img/imagen2.png"`
 - `Node.value="Error";`
- `node.setAttribute('attr-name', 'attr-value')`
- `node.removeAttribute('attr-name')`

JavaScript

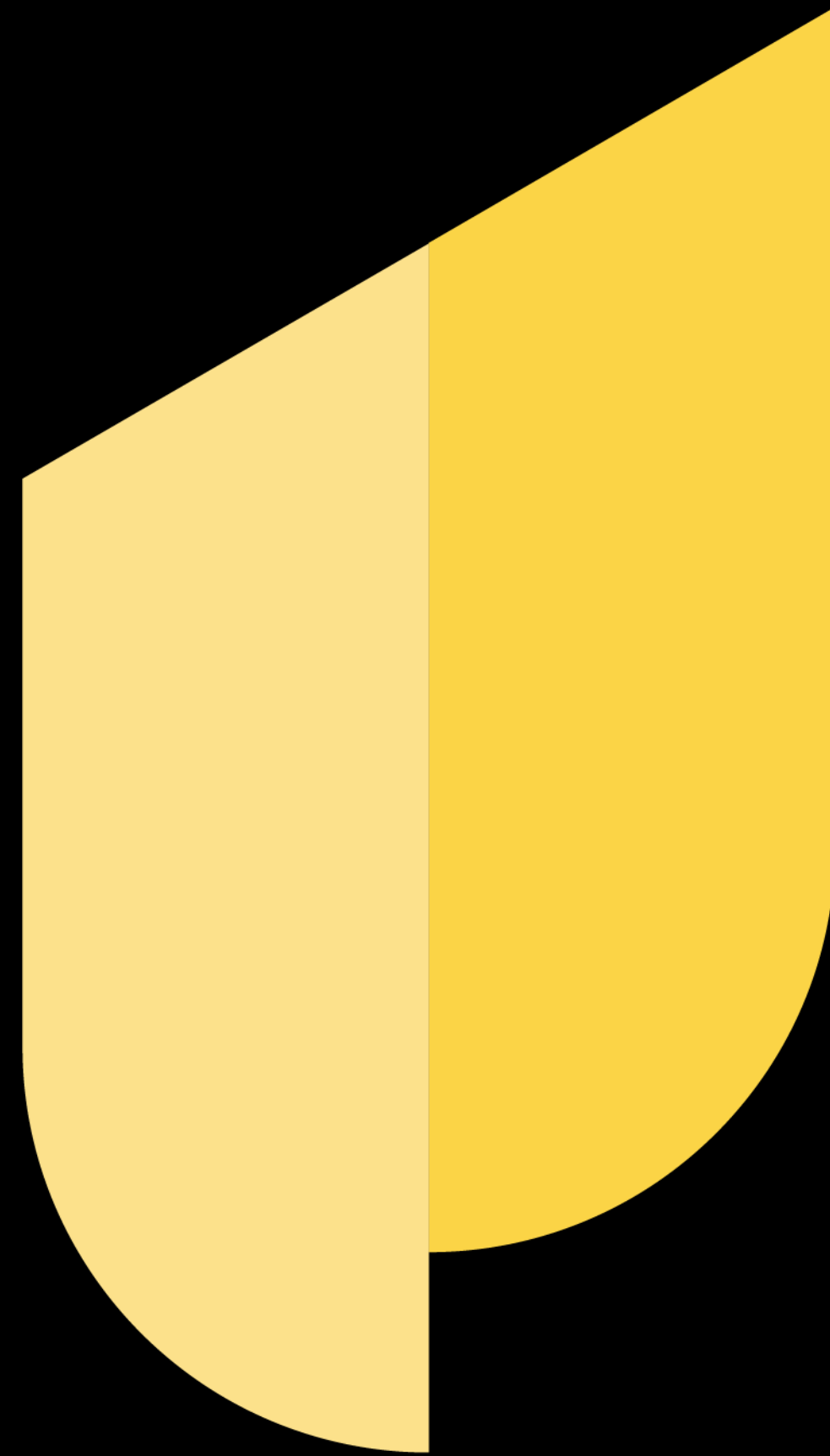


Modificar estilos
de un nodo

- `node.style.color='red'`
- `node.style.padding='10px',`
- `node.style.fontSize='200%'`



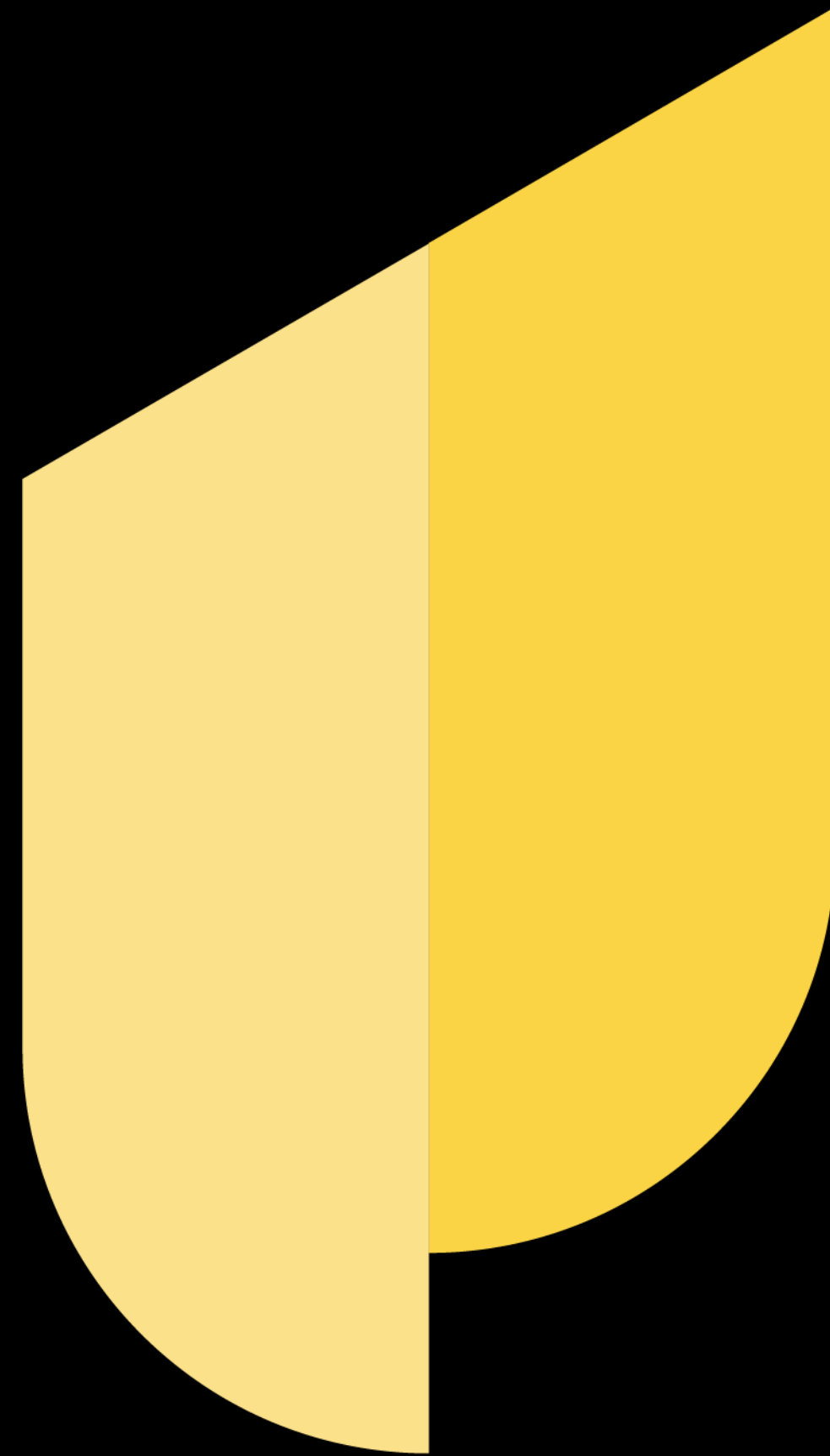
JavaScript



manipular
classes

- `node.classList.add('class-name', ...)`
- `node.classList.remove('class-name', ...)`
- `node.classList.toggle('class-name')`
- `node.classList.contains('class-name')`
- `node.classList.replace('old', 'new')`

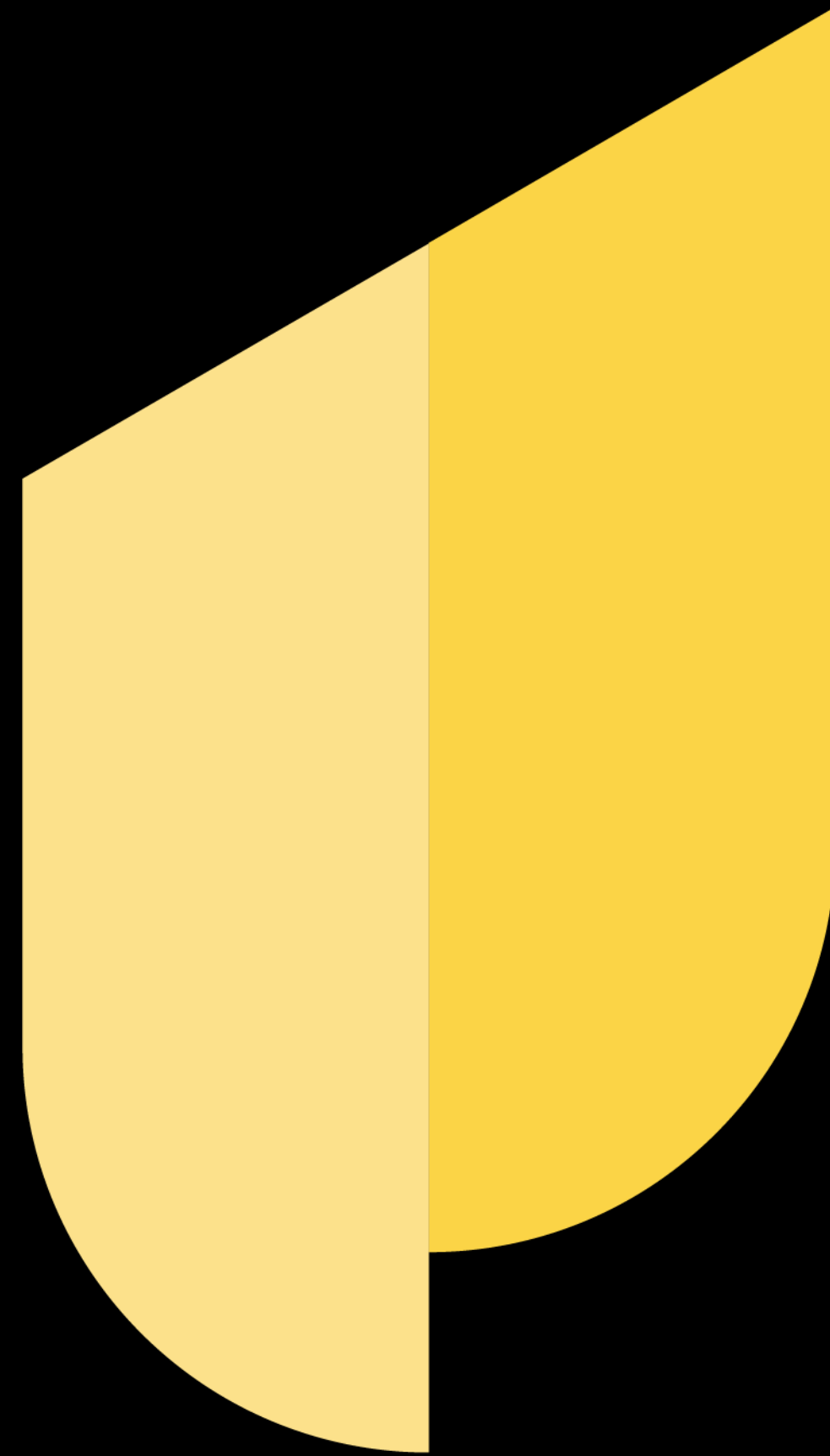
JavaScript



Eliminar nodos

- `parentNode.removeChild(nodeToRemove)`
- `nodeToRemove.parentNode.removeChild(nodeToRemove)`
//para que se elimine un mismo nodo

JavaScript



Variables



Tipo	Ejemplo
String	“Texto” ‘Texto’ `Texto`
Number	123.45 123
Boolean	True false
Null	Null
Undefined	Undefined
Object	Array Json Object



Declaración de variables y constantes

- **var** identificadorA="valor"; // variable global
- **let** identificadorB="valor"; // variable local
- **const** identificadorC="valor"; // constante

Modo estricto

- Obliga a definir una variable antes de ser utilizada.
- Instrucción: **'use strict'**; activa este modo antes de cualquier expresión.



JavaScript



Operadores

Operadores matemáticos

```
a = b + c - d;           // addition, subtraction
a = b * (c / d);         // multiplication, division
x = 100 % 48;            // modulo. 100 / 48 remainder = 4
```

Operadores incremento/decremento

```
a++; b--; // postfix increment and decrement
```

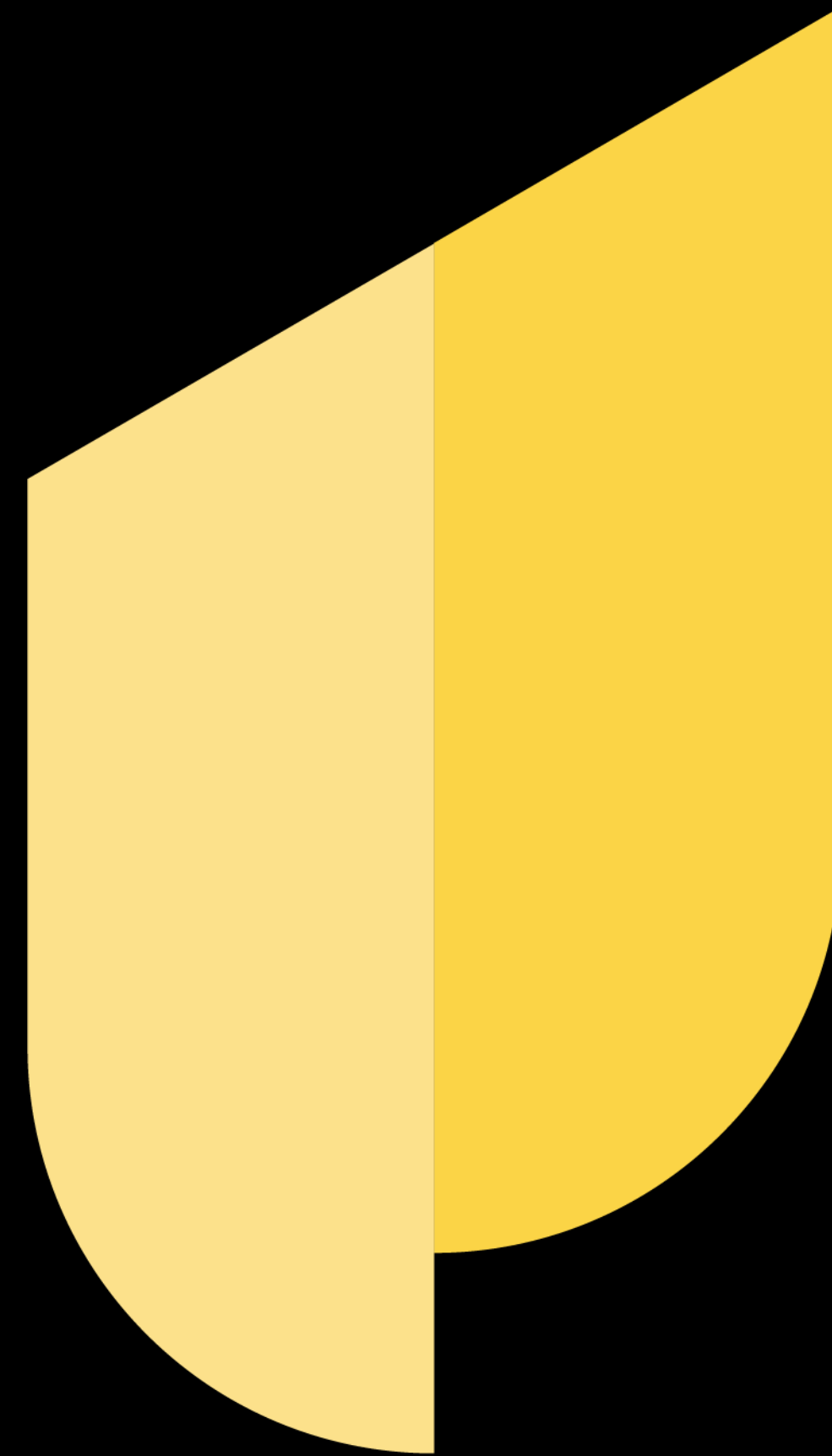
Operadores asignación

- .. = .. Assign value
- .. += .. Add then assign
- .. -= .. Subtract then assign
- .. *= .. Multiply then assign

Operadores Lógicos

```
a == b           // equals
a != b           // unequal
a === b          // strict equal
a !== b          // strict unequal
a < b    a > b    // less and greater than
a <= b   a >= b   // less or equal, greater or eq
a += b    // a = a + b (works with - * %...)
a && b     // logical and
a || b     // logical or
```

JavaScript



Funciones

Funciones con Nombre

```
function name(parameter1, parameter2, parameter3) {  
    // code to be executed  
}
```

Funciones con valor retorno

```
function myFunction(a, b) {  
  return a * b;  
}
```

Funciones como variables

- `let mifuncion = function covertirKg(kg){`
 - `Return kg*1000;`
- `}`
- `let msgKilos = "Kilos a Gramos es: " + x(2);`

Funciones anónimas

- `let cuadrado = function(numero) { return numero * numero }`

Funciones de flecha

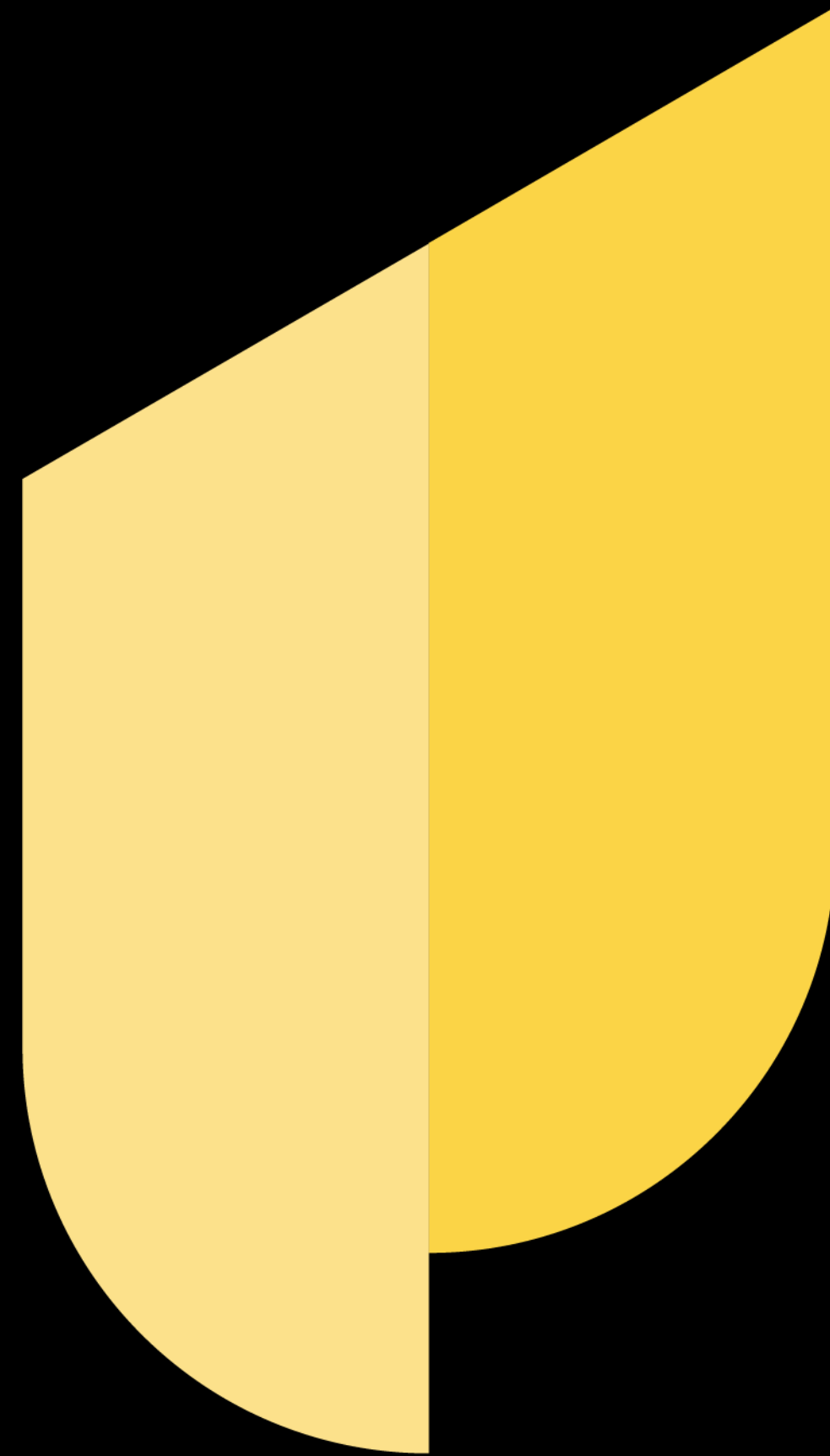
```
(param1, paramN) => expression
```

```
param => {  
  let a = 1;  
  return a + b;  
}
```

Funciones de tiempo

- `setTimeout(function () {//instrucciones}, Tiempo);`
- `setInterval(function () {//instrucciones}, Tiempo);`
- Eliminar:
 - `clearTimeout(myTimeout);`
 - `clearInterval(myInterval);`

JavaScript



Condicionales

If

```
if (condition) {  
    // block of code to be executed if the condition is true  
}
```

```
if (hour < 18) {  
    greeting = "Good day";  
}
```

If .. else

```
if (condition) {  
    // block of code to be executed if the condition is true  
} else {  
    // block of code to be executed if the condition is false  
}
```

```
if ((age >= 14) && (age < 19)) {  
    status = "Eligible.";   
} else {  
    status = "Not eligible.";   
}
```

If anidado

```
if (condition1) {  
    // block of code to be executed if condition1 is true  
} else if (condition2) {  
    // block of code to be executed if the condition1 is false and condition2 is true  
} else {  
    // block of code to be executed if the condition1 is false and condition2 is false  
}
```

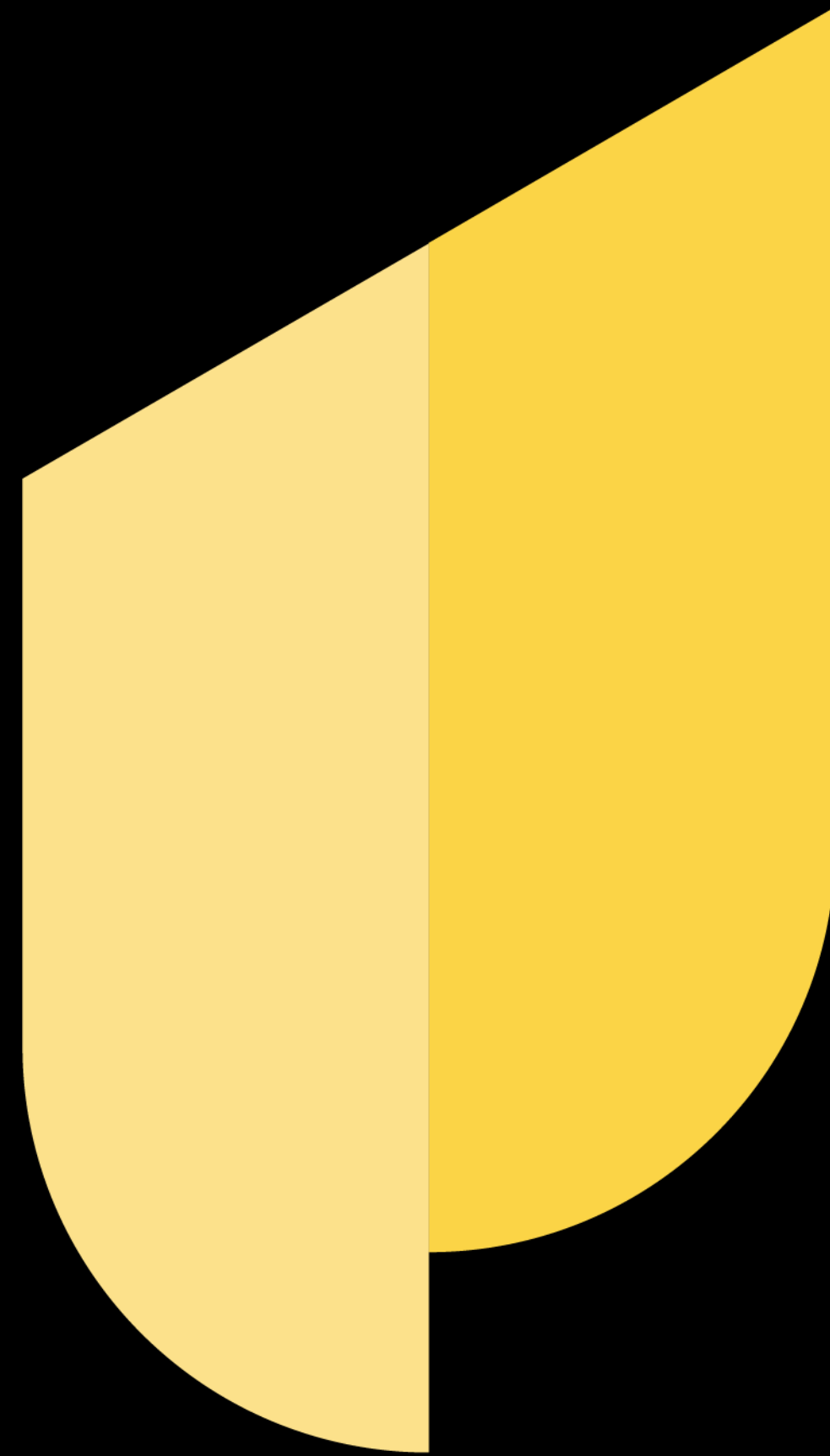
```
if (time < 10) {  
    greeting = "Good morning";  
} else if (time < 20) {  
    greeting = "Good day";  
} else {  
    greeting = "Good evening";  
}
```

Switch

```
switch(expression) {  
  case x:  
    // code block  
    break;  
  case y:  
    // code block  
    break;  
  default:  
    // code block  
}
```

```
switch (new Date().getDay()) {  
  case 6:  
    text = "Saturday";  
    break;  
  case 0:  
    text = "Sunday";  
    break;  
  default:  
    text = "Whatever";  
}
```

JavaScript



Bucles

for

```
for (statement 1; statement 2; statement 3) {  
    // code block to be executed  
}
```

```
for (let i = 0, len = cars.length, text = ""; i < len; i++) {  
    text += cars[i] + "<br>";  
}
```

for..of

```
for (const x of letters) {  
  // code block to be executed  
}  
  
// Create aa Array  
const letters = ["a","b","c"];  
  
// List all Elements  
let text = "";  
for (const x of letters) {  
  text += x + "<br>";  
}
```

Array.foreach

array.forEach(function(currentValue, index, arr), thisValue)

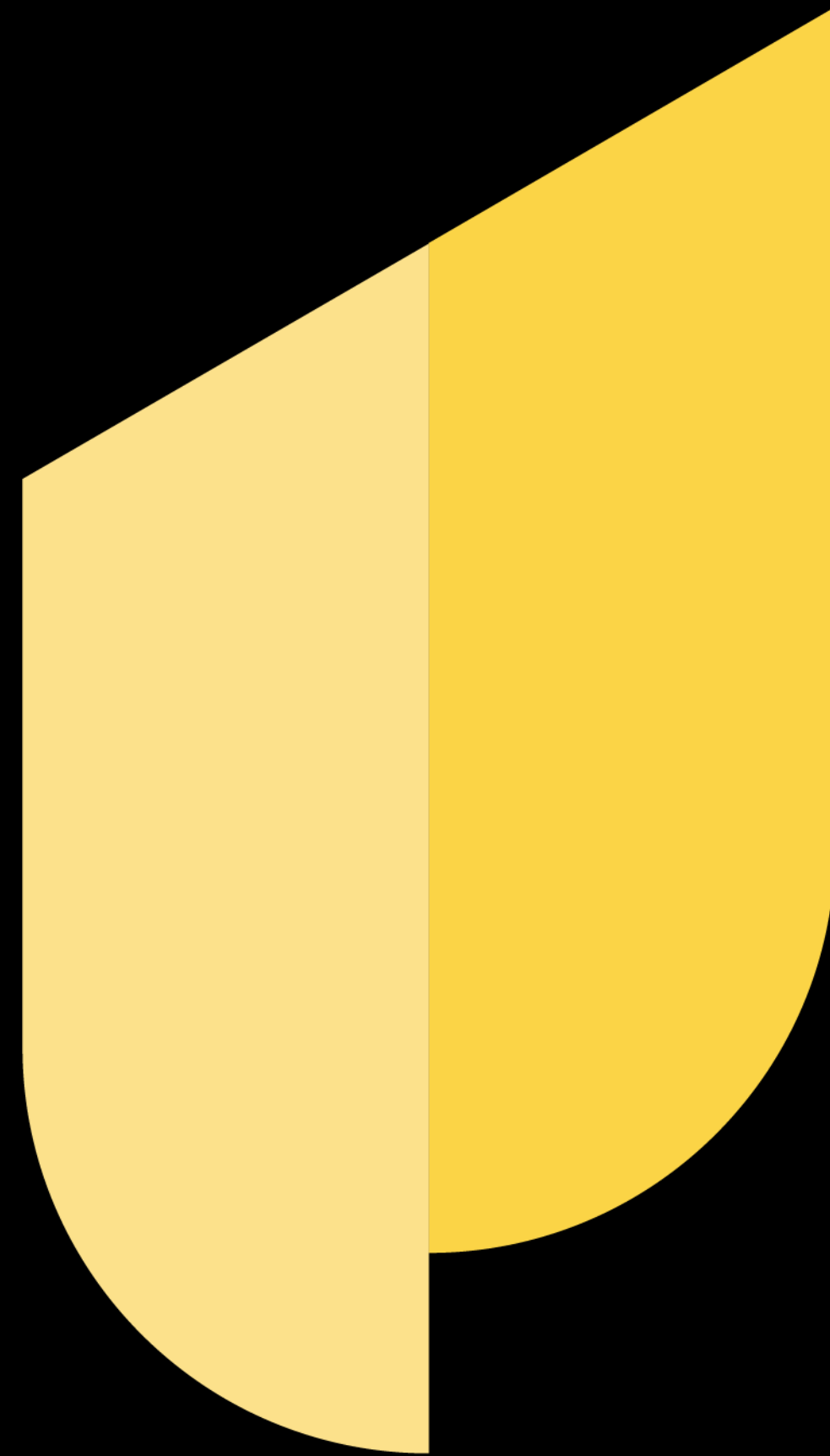
```
let sum = 0;
const numbers = [65, 44, 12, 4];
numbers.forEach(myFunction);

function myFunction(item) {
  sum += item;
}
```

while

```
while (condition) {  
    // code block to be executed  
}  
  
while (i < 10) {  
    text += "The number is " + i;  
    i++;  
}
```

JavaScript

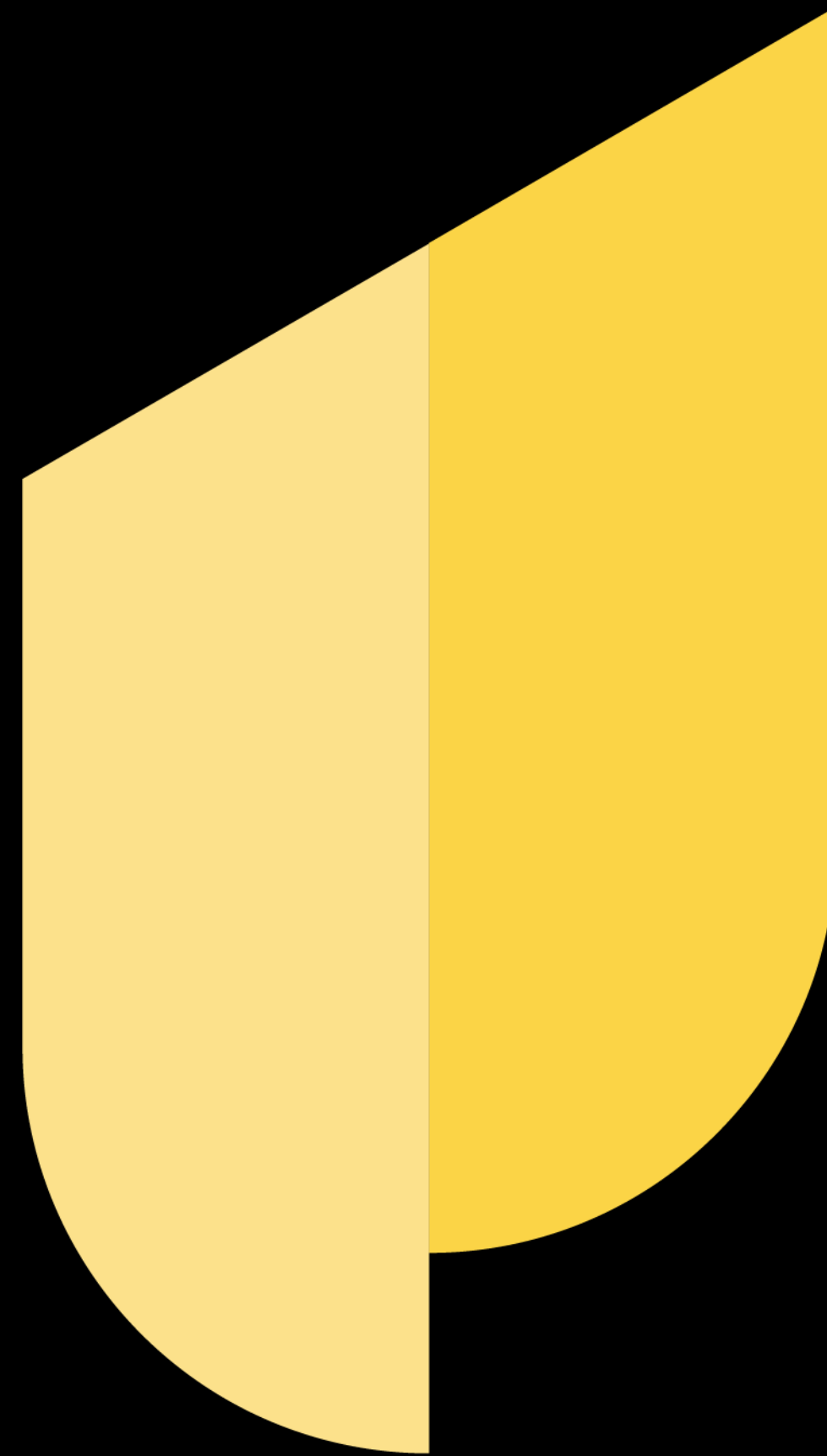


Objeto String

var abc = "texto"	
var esc = 'Hola \n Mundo'	\n carácter escape salto de línea
var len = abc.length	tamaño de la cadena
abc.indexOf("lmno")	busca la subcadena, retorno el número de posición, de lo contrario -1
abc.lastIndexOf("lmno")	última conciencia
abc.slice(3, 6)	corta "def", los valores negativos cuentan desde atrás
abc.replace("abc","123")	encuentra y reemplaza la cadena
abc.toUpperCase()	convierte a upper case
abc.toLowerCase()	convierte a lower case
abc.concat(" ", str2)	abc + " " + str2
abc.charAt(2)	Carácter en índice 2: "c"
abc.charCodeAt(2)	Código carácter índice 2: "c" -> 99
abc.split(",")	dividir una cadena en comas (cadena comodina), retorna un arreglo
abc.split("")	Separa en caracteres retorna un arreglo



JavaScript

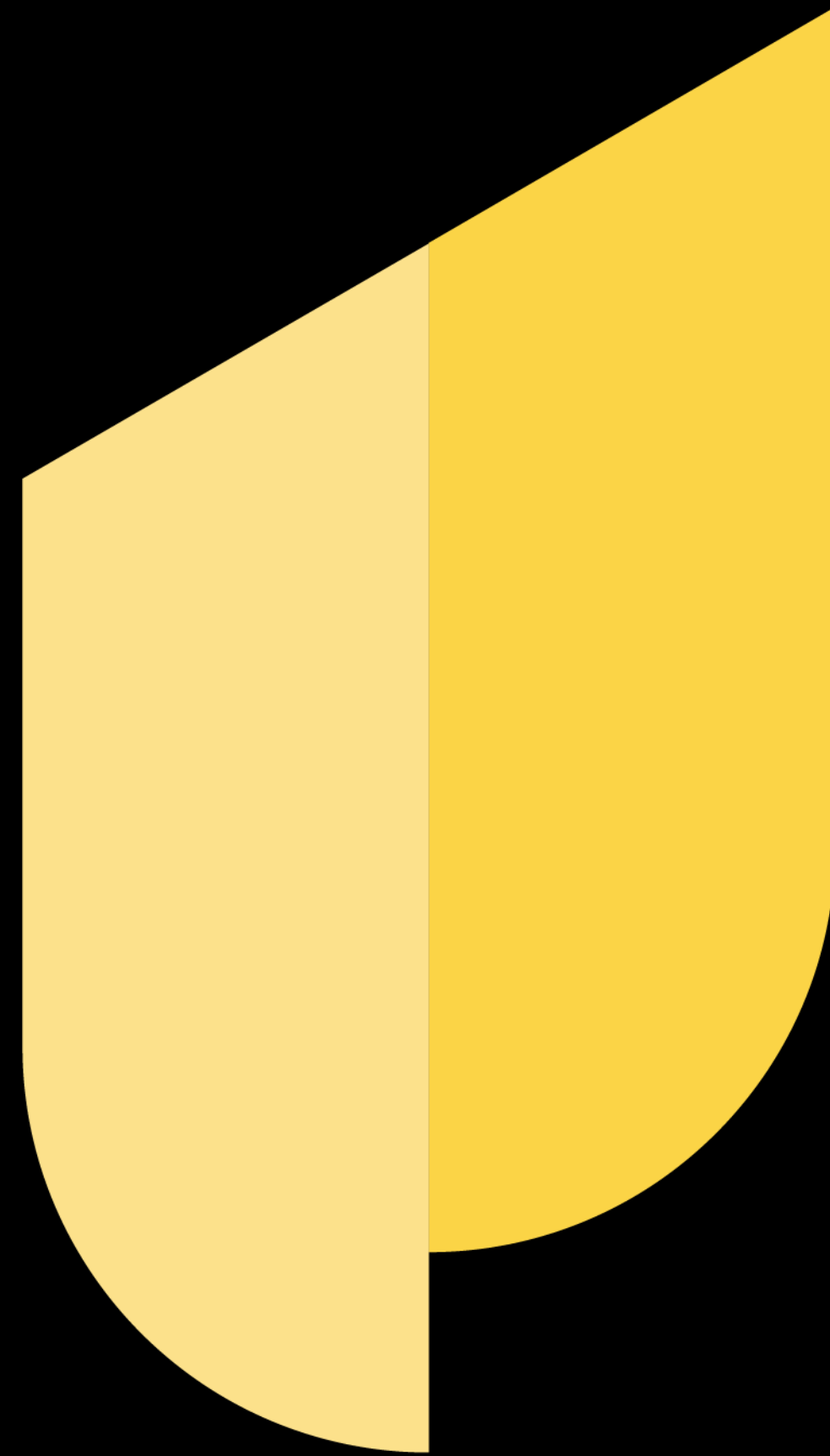


Objeto Number

var pi = 3.141;	
pi.toFixed(0);	// returns 3
pi.toFixed(2);	// regresa 3.14 – para trabajar con dinero
pi.toPrecision(2)	// regresa 3.1
pi.valueOf();	// regresa number
Number(true);	// convierte a number
Number(new Date());	// numero de milisgundos desde 1970
parseInt("3");	// regresa 3 en tipo number
parseFloat("3.5");	// regresa 3.5
Number.MAX_VALUE	// Numero mas grande posible en JS
Number.MIN_VALUE	// Numero mas pequeño posible en JS
Number.NEGATIVE_INFINITY	// -Infinito
Number.POSITIVE_INFINITY	// Infinito



JavaScript

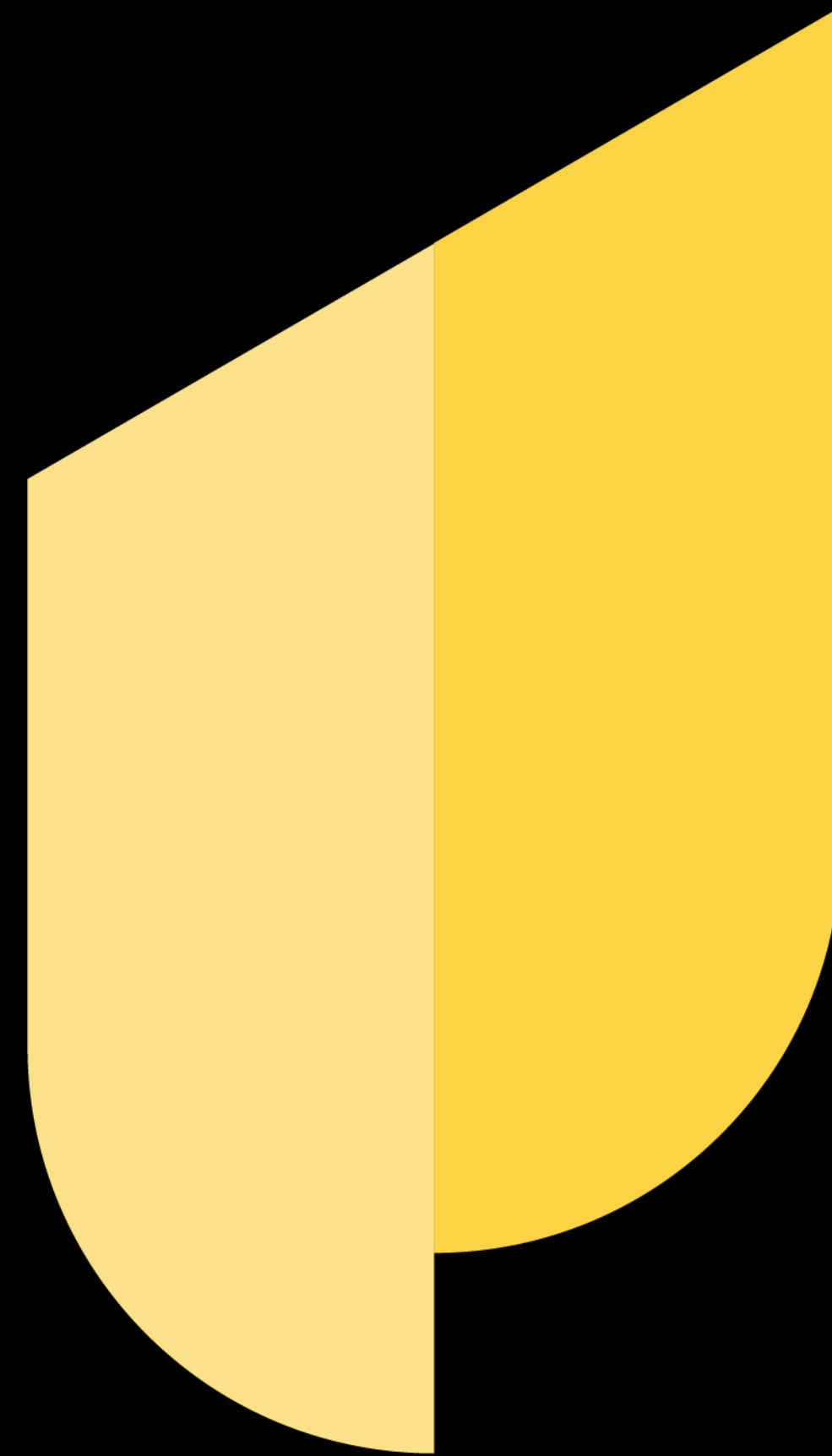


Objeto Math

var pi = Math.PI	3.141592653589793
Math.round(4.4)	4 - rounded
Math.round(4.5)	= 5
Math.pow(2,8)	= 256 - 2 to the power of 8
Math.sqrt(49)	= 7 - square root
Math.abs(-3.14)	= 3.14 - absolute, positive value
Math.ceil(3.14)	= 4 - rounded up
Math.floor(3.99)	= 3 - rounded down
Math.sin(0)	= 0 - sine
Math.cos(Math.PI)	OTHERS: tan,atan,asin,acos,
Math.min(0, 3, -2, 2)	= -2 - the lowest value
Math.max(0, 3, -2, 2)	= 3 - the highest value
Math.log(1)	= 0 natural logarithm



JavaScript



Objeto Date

<code>var d = new <i>Date</i>();</code>	Obtener fecha de sistema
<code>Date("2017-06-23");</code>	Declaración fecha en String
<code>Date("2017");</code>	Define fecha al 1 de enero 2017
<code>Date("2017-06-23T12:00:00-09:45");</code>	Fecha formato YYYY-MM-DDTHH:MM:SSZ
<code>Date("June 23 2017");</code>	Formato fecha larga
<code>a = d.getDay();</code>	Obtener día de la semana (0-6)
<code>getDate();</code>	Día del mes (1-31)
<code>getFullYear();</code>	Año de 4 dígitos
<code>getHours();</code>	Hora 0-23
<code>getMilliseconds();</code>	Milisegundos 0-999
<code>getMinutes();</code>	Minutos 0-59
<code>getMonth();</code>	Mes 0-11
<code>getSeconds();</code>	Segundos 0-59
<code>getTime();</code>	Milisegundos desde 1970



JavaScript

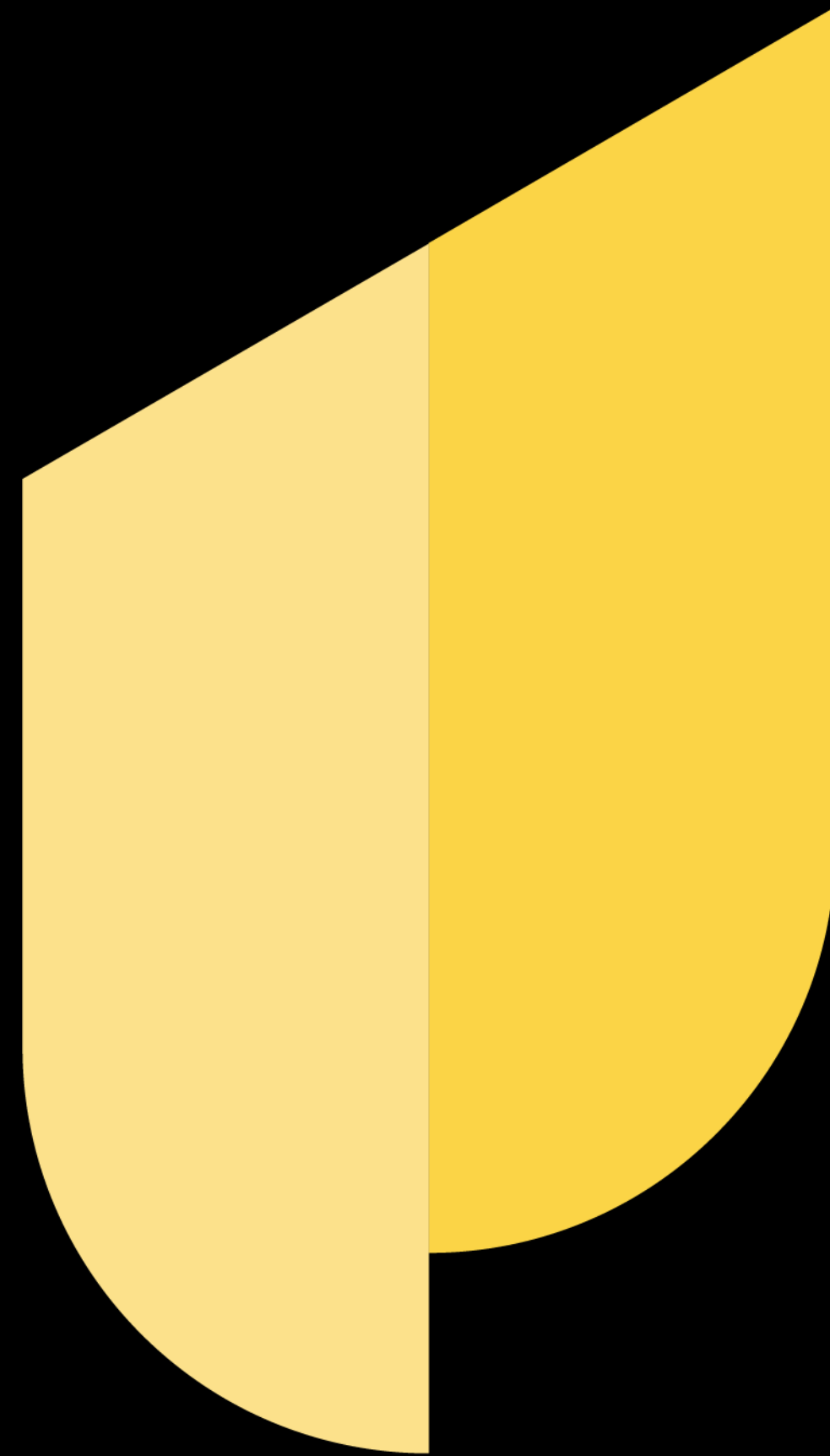


Objeto array

var dogs = ["Bulldog", "Beagle", "Labrador"];	
dogs.toString();	Convierte a String el arreglo
dogs.join(" * ");	Une el arreglo mediante el carácter *
dogs.pop();	Elimina el último elemento
dogs.push("Chihuahua");	Agregar un nuevo elemento l inicio
dogs.length];	Obtener el tamaño del arreglo
dogs.shift();	Elimina el primer elemento
dogs.unshift("Chihuahua");	Agrega un nuevo elemento al inicio
var animals = dogs.concat(cats,birds);	Une dos arreglos
dogs.sort();	Ordena alfabéticamente
dogs.reverse();	Ordena descendentemente
x.sort(function(a, b){return a - b});	Ordena números de forma descendentemente
x.sort(function(a, b){return 0.5 - Math.random()});	Ordena aleatoriamente el arreglo



JavaScript



Eventos

Tipos de evento

- Mouse
- Teclado
- Form
- Drag
- Clipboard
- Media
- Animation
- Misceláneos



Mouse

- onclick, oncontextmenu, ondblclick, onmousedown, onmouseenter, onmouseleave, onmousemove, onmouseover, onmouseout, onmouseup

Teclado

- onkeydown, onkeypress, onkeyup



Form

- onblur, onchange, onfocus, onfocusin, onfocusout, oninput, oninvalid, onreset, onsearch, onselect, onsubmit

Drag

- ondrag, ondragend, ondragenter, ondragleave, ondragover, ondragstart, ondrop



Clipboard

- oncopy, oncut, onpaste



Media

- onabort, oncanplay, oncanplaythrough, ondurationchange, onended, onerror, onloadeddata, onloadedmetadata, onloadstart, onpause, onplay, onplaying, onprogress, onratechange, onseeked, onseeking, onstalled, onsuspend, ontimeupdate, onvolumechange, onwaiting

Animation

- animationend, animationiteration, animationstart



Misceláneo

- transitionend, onmessage, onmousewheel, ononline, onoffline, onpopstate, onshow, onstorage, ontoggle, onwheel, ontouchcancel, ontouchend, ontouchmove, ontouchstart



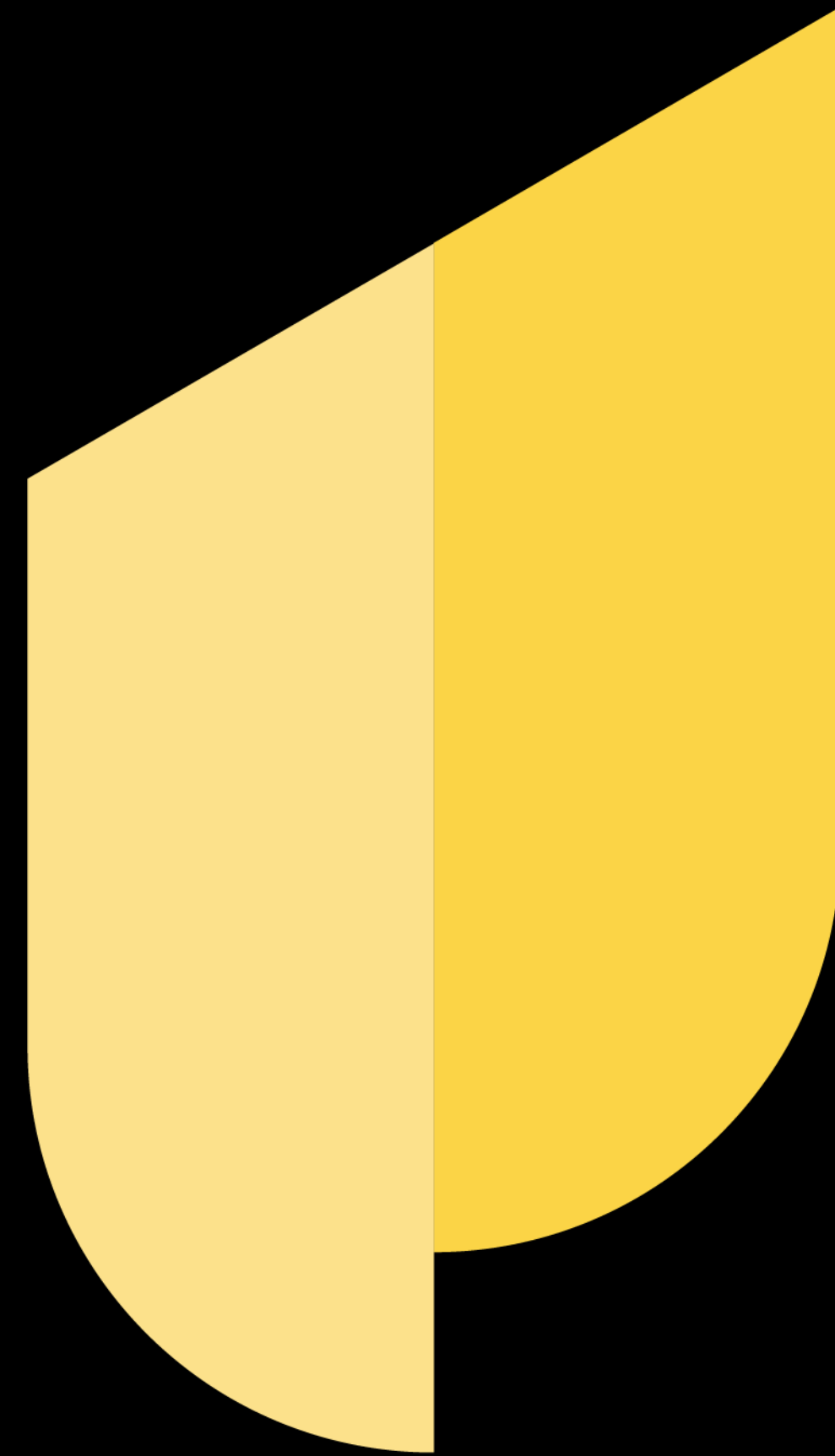
Sintaxis para crear un evento

- `Nodo.nombreEvento=función manejadora.`

Agregar eventos a nodos

- Mediante atributo: `node.evento=funcionManejadora`
- Mediante listener: `node.addEventListener('event-name',callback-function)` //este ultimo indispensable par asignar eventos a elementos dinámicos

MUCHAS



GRACIAS