



# Curso de Front-End

## Clase 07



# Agenda de la clase



# Agenda

- Feedback del curso.
- Juego – Kahoot.
- JavaScript.
  - Sentencias.
  - Tipos de datos.
  - Variables.
  - ¿Dónde colocar nuestro código JS?
- Ejercicios.



# Feedback del curso



# Feedback del curso

Aprovechemos para tomarnos unos minutos y charlar sobre cómo venimos el curso. Por ejemplo:

- ¿Qué es lo que más les ha gustado hasta el momento?
- ¿Hay algún tema que haya sido muy difícil?
- ¿Hay algún tema que les gustaría repasar?
- ¿Tienen alguna sugerencia para que mejoremos algún aspecto del curso?
- ¿Les gusta nuestra [selección musical](#)? (esto aplica para los cursos [presenciales](#)).



# Kahoot



# ¡Juguemos un rato!

Entren a <https://kahoot.it>

(Llegado el momento les daremos un PIN para entrar).

👉 Pero antes de jugar, ver ejemplos en las siguientes *slides*.



## En la pantalla del docente:

The interface consists of a central dark gray box with the text: **Bootstrap** sirve para construir sitios **responsive**.

At the top right is a blue button labeled "Skip".

On the left, a purple circle contains the number "8".

On the right, the text "0 Answers" is displayed.

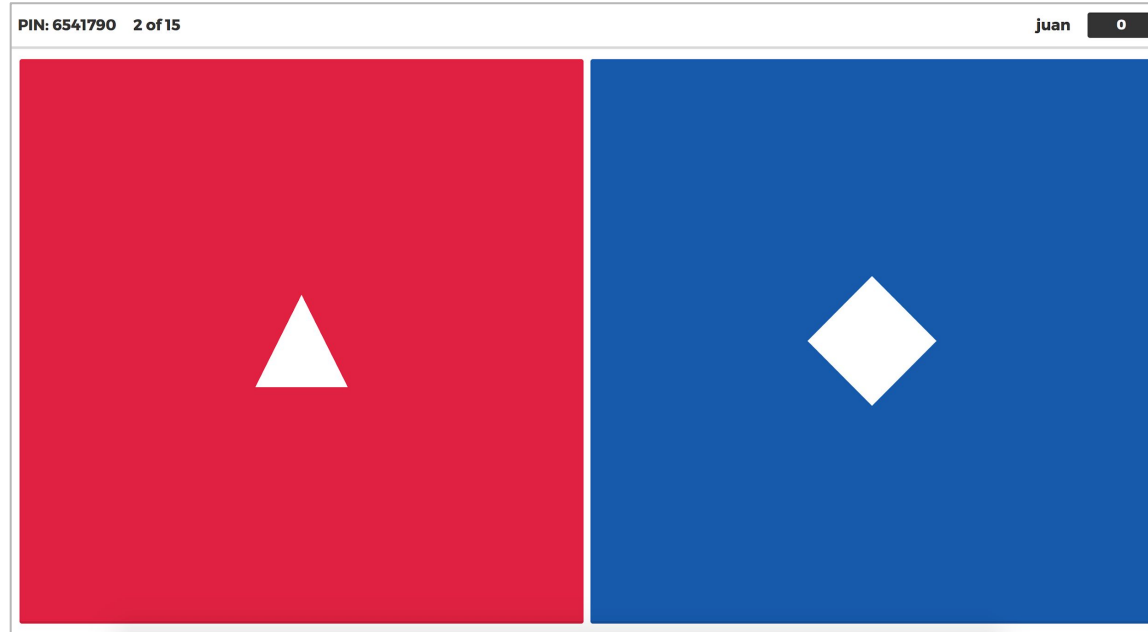
At the bottom, there are two large buttons: a red one labeled "Falso" with a white upward-pointing triangle icon, and a blue one labeled "Verdadero" with a white diamond icon.

Cantidad de segundos restantes para responder.

Cantidad de participantes que ya respondieron.

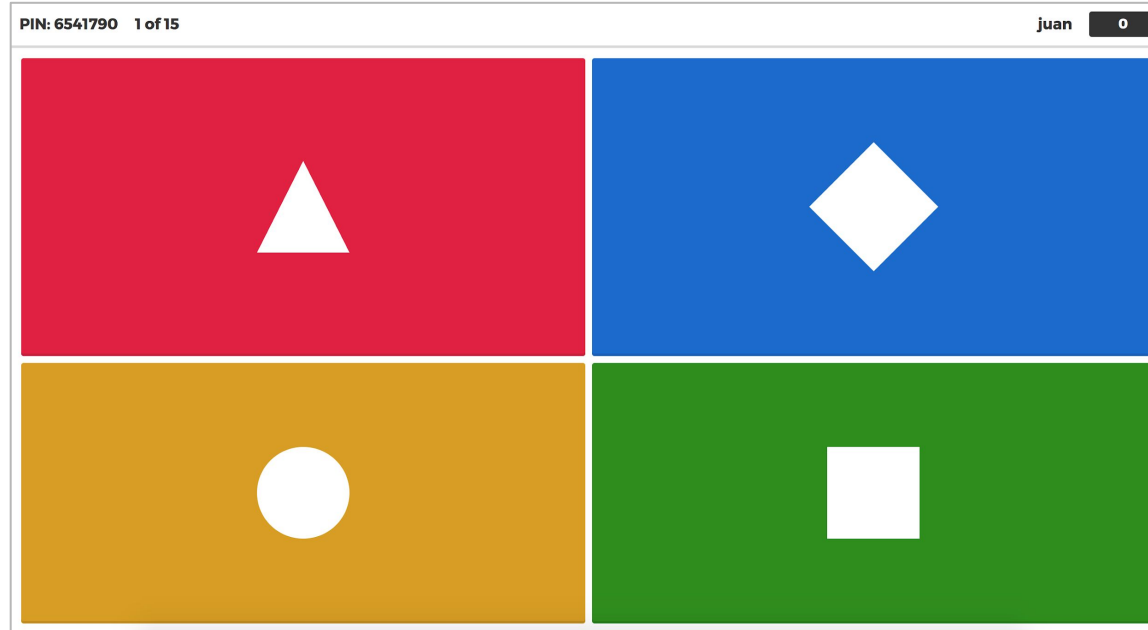


## En las pantallas de los alumnos:



⚠ Notar que en sus pantallas no aparecerán los textos de las preguntas ni de las respuestas. Eso lo deben leer en la pantalla del docente.

## En las pantallas de los alumnos:



⚠ Notar que en sus pantallas no aparecerán los textos de las preguntas ni de las respuestas. Eso lo deben leer en la pantalla del docente.



# JavaScript (JS)



*¡Hoy empezamos a programar!*



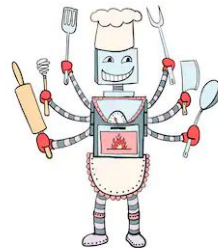
# ¿Qué es programar? (1/2)

Programar es “*crear un conjunto de instrucciones que serán ejecutadas por una computadora*”.

Por el momento, las computadoras son máquinas bastante “tontas”. Hay que decirles, con lujo de detalles (y **sin ambigüedades**), lo que queremos que hagan.

---

**Analogía:** Escribir un programa es como escribir una **receta de cocina** para una persona que nunca cocinó algo en su vida. Imagínense que en lugar de una persona es un robot. No alcanza con especificar los ingredientes y las cantidades. Es necesario explicar con precisión cada uno de los pasos que hay que realizar. Ejemplo: “*Girar la cuchara 90 veces en sentido horario a una velocidad de 300 rpm. Si quedan grumos, girar en sentido anti-horario a 120 rpm hasta que desaparezcan, etc*”. A la persona (o al robot) no le deben quedar dudas sobre cómo proceder.





# ¿Qué es programar? (2/2)

## Tipos de lenguajes de programación:

- **Código de máquina / Código binario**

- Es el menor nivel de abstracción. Consiste en en 1's y 0's.

```
0110001100
1011010110
1111011110
```

- **Lenguaje Assembly**

- Representación simbólica de código de máquina.

- **Lenguajes compilados**

- Lenguajes de alto nivel que necesitan ser compilados a código de máquina.
- Ej: C, C++, Go, Fortran, Pascal, Java.



- **Lenguajes interpretados**

- Lenguajes de alto nivel que no necesitan ser compilados. Programas residen en memoria.
- Ej: JavaScript, Python, PHP, Ruby.



Más nivel de abstracción





# Estructura de una página web

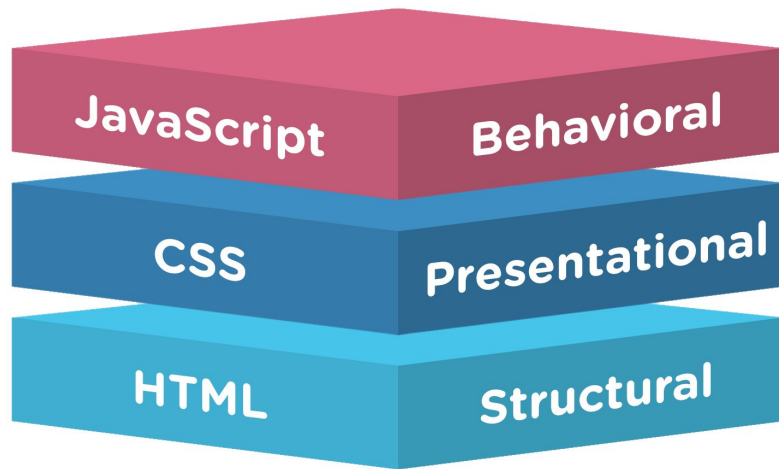
Contenido: Texto, imágenes, videos.

+ **HTML**: Estructura + Semántica

+ **CSS**: Presentación + Diseño

+ **JS**: Interacción



= Página web



Nota: Esta es la estructura básica de una página desde el punto de vista de un desarrollador Front-End. Un desarrollador Back-End también consideraría, por ejemplo, el guardado (persistencia) de los datos.



# ¿Qué es JavaScript?

- Es el lenguaje de programación de la web.
- Nació en 1995 (desarrollado por Netscape).  Netscape®
- Interactuamos con JavaScript todos los días  
En Gmail, Facebook, Google Search, etc. Es difícil encontrar un sitio que no use JS.
- JavaScript   $\neq$  Java 
- Documentación:
  - MDN: <https://developer.mozilla.org/es/docs/Web/JavaScript>
  - W3Schools: <http://www.w3schools.com/js/default.asp>





# *Lenguaje de programación más popular en 2021.*

Según encuesta de Stack Overflow: <https://insights.stackoverflow.com/survey/2021#technology-most-popular-technologies>.



# ¿Qué se puede hacer con JS?

- Operaciones matemáticas como sumar, restar, multiplicar y dividir.
- Cosas simples como mostrar un mensaje cuando el usuario hace click en un botón.
- Mostrar gráficos: [GapMinder](#).
- Aplicaciones web como Gmail, Google Docs, Google Maps, WhatsApp Web, etc.
- Scroll infinito (como el News Feed de Facebook).
- Trackeo de usuarios (Google Analytics).
- Dropdowns, modales, tabs, sliders (carousels), accordions, etc.
- Cosas “cool” como estas: <http://paperjs.org/examples/>.



## Nota sobre las versiones de JavaScript

Por un tema de **simplificación** a la hora del **aprendizaje**, en este curso se usará (mayoritariamente) la versión **ES5** de JavaScript.

Por este motivo, para declarar variables, se usará la *keyword* `var` en lugar de `const` y `let`. A la hora de declarar funciones, no se usará la notación *arrow* (flecha). Tampoco se usará *spread operator* ni desestructuración de objetos, por nombrar algunos ejemplos.

### Notas:

- Los navegadores más modernos soportan nuevas funcionalidades de JavaScript que fueron incorporadas en las versiones ES6 y posteriores (ES6+).
- La mayoría de los desarrolladores en la industria IT trabajan con ES6+ y es lo que usamos en [cursos más avanzados](#) en HA.
- Todo el código que aprendan sobre ES5 sigue siendo compatible con ES6+.
- Ver más información sobre [ECMAScript](#).

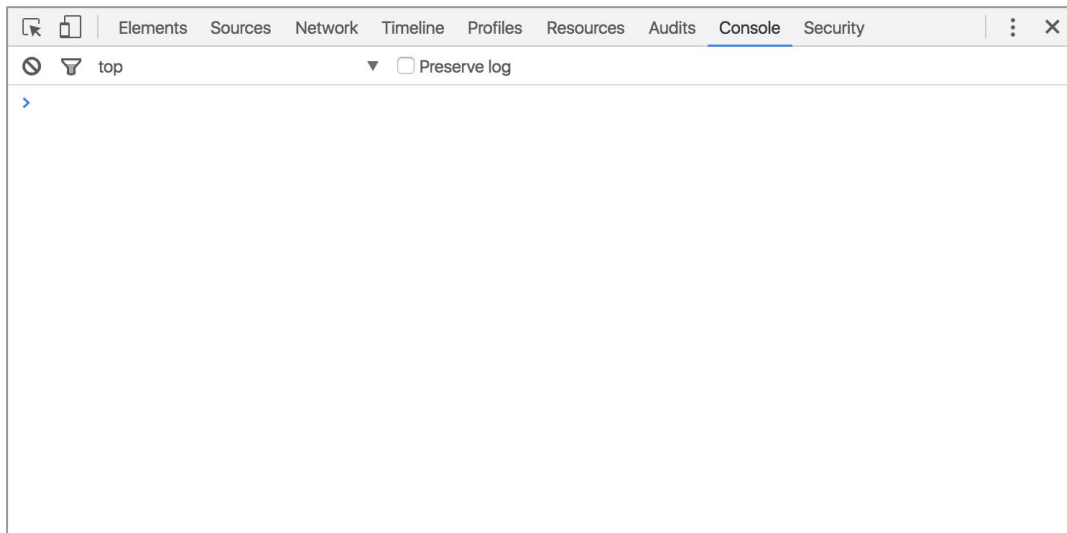


Empecemos a probar JS

# Empecemos a probar JS



- Abrir Google Chrome.
- Ir a View > Developer > **JavaScript Console** (Windows y Linux: Ctrl + Shift + J– Mac: Cmd + Option + J).



Prueben de escribir:

> 5+4

> 15 / 4

División

> 15 % 4

Módulo

> 16 % 4

> "Hola mundo!"



# Módulo de un número (1/2): $21 \div 4 = 1$

Es el resto de la “división entera”.

$$\begin{array}{r} 21 \\ 4 \overline{) 21} \\ \underline{4} \phantom{0} \\ 5 \end{array}$$

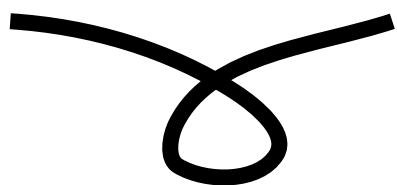
The diagram illustrates the division of 21 by 4. The number 21 is on the left, and 4 is on the right. A horizontal line is drawn under the 4. Below the line, the number 5 is written. To the left of the 5, the number 1 is written in orange. A large, stylized black 'X' is drawn over the 1 and the 5.

La operación módulo es útil para consultar si un número es múltiplo de otro. Por ejemplo, es común usarla para conocer si un número es par o impar.



# Módulo de un número (2/2): $20 \% 4 = 0$

Es el resto de la “división entera”.

$$\begin{array}{r} 20 \\ 4 \overline{) 20} \\ \underline{20} \\ 0 \end{array}$$


La operación módulo es útil para consultar si un número es múltiplo de otro. Por ejemplo, es común usarla para conocer si un número es par o impar.



# Sentencias





# Sentencias en JS

- Una **sentencia** (statement) es una instrucción que se le da a la computadora.
- Puede estar formada por una o varias líneas de código.
- No es una palabra clave, sino un grupo de palabras clave.
- Es usual (y se recomienda) separar las sentencias con un punto y coma ";".

Prueben escribir en la consola:

```
> console.log("Estamos arrancando con JS!");  
> alert("Estamos arrancando con JS!");
```



# Tipos de datos



# Tipos de datos en JS > String y Number (1/2)

Hay varios tipos de datos en JS, hoy vamos a ver String y Number.

## String

- Conjunto de caracteres encerrados por comillas (simples, dobles o backticks).
- Ejemplos: *"Hola Mundo"* , *'Hola Mundo'* y *`Hola Mundo`*.

## Number

- Pueden ser números enteros o decimales.
- Ejemplos: *5* y *5.2349659*.



# Tipos de datos en JS > String y Number (2/2)

Prueben en la consola escribir lo siguiente:

```
> typeof("Hola");           // También funciona: typeof "Hola"
> typeof(Hola);             // También funciona: typeof Hola
> typeof(2);                // También funciona: typeof 2
> typeof("2");              // También funciona: typeof "2"
```

`typeof` es una funcionalidad que brinda JS para consultar el tipo de “algo”.



# Variables



# Variables (1/5)

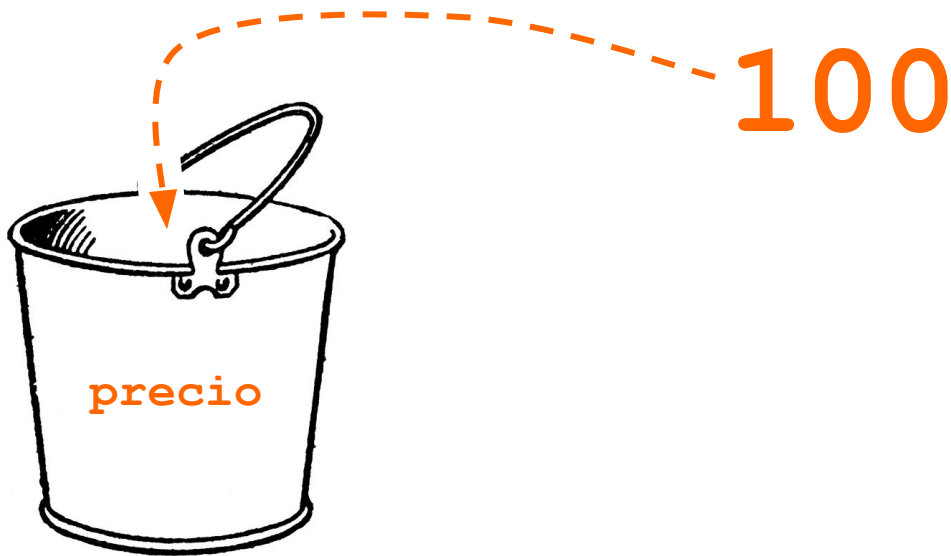
Las variables son **contenedores que permiten guardar información** (valores) y poder usarla más tarde.





## Variables (2/5)

Para definir una variable en JavaScript y asignarle un valor, se escribe el siguiente código: `var precio = 100;`





# Variables (3/5)

Prueben en la consola escribir lo siguiente:

```
> var precioA = 100;  
> var precioB = 50.5;  
> precioA = precioA + precioB;  
> var precioC;
```

Línea 1: Sentencia que declara una variable llamada `precioA` y le asigna el valor 100.

Línea 2: Sentencia que declara una variable llamada `precioB` y le asigna el valor 50.5.

Línea 3: Sentencia que asigna la suma de `precioA` + `precioB` a la variable `precioA`.

Línea 4: Sentencia que declara una variable llamada `precioC`.





## Variables (4/5)

Las variables pueden guardar el resultado de **expresiones**.

```
> var unNumero = 5 + 7;  
  
> var unString = "Hola" + " " + "Mundo";
```

Importante: no se guardan las expresiones en sí mismas, lo que se guarda es el resultado.



# Variables (5/5) – ¿Cómo se nombran?

- Comenzar los nombres con letras.  
(Después verán casos en lo que los nombres pueden empezar con \$ o \_).
- Sólo usar letras, números, \$ y \_.
- NO usar espacios.
- Recordar que los nombres son **case sensitive** (*hola* y *hoLa* son distintos).
- **Evitar palabras reservadas** (palabras que ya son usadas por JS).  
Ej: una variable no puede ser llamada `var` ni `typeof`. Y si bien se puede, por favor evitar nombres como `vAr` o `typeOF`.
- Usar nombres mnemotécnicos (**descriptivos**) (Ej: es mejor usar `precioTotal` que `x1`).
- Es común usar **camelCase** (Ej: `precioVariableTotal` en lugar de `preciovariabletotal`).
- Es común (y recomendable) usar nombres en **inglés**.



¿Dónde colocar nuestro código JS?



# ¿Dónde ponemos nuestro código JS? (1/3) – Método 1

Similar a como sucede con CSS, hay varias maneras de agregar código JS a nuestras páginas HTML.

```
<body>

...

<script>

    // Aquí se coloca el código JS.

    // En general se agrega al final del documento, antes de cerrar el body.

</script>

</body>
```

Esto es similar a cuando  
agregábamos CSS con las  
etiquetas `<style>`



## ¿Dónde ponemos nuestro código JS? (2/3) – Método 2

Similar a como sucede con CSS, hay varias maneras de agregar código JS a nuestras páginas HTML.

```
<body>
```

```
...
```

```
...
```

```
<script src="js/app.js"></script>
```

```
</body>
```

Esto es similar a cuando  
agregábamos CSS con la  
etiqueta `<link>`.



# ¿Dónde ponemos nuestro código JS? (3/3)

Si pusiésemos el código JS al inicio del documento HTML, el HTML no se cargaría (y no se mostraría) hasta que primero se haya cargado el JS.

Dependiendo del tamaño del JS, el usuario podría ver la página en blanco durante algunos segundos.

Además, en caso de que el código JS tuviese la intención de modificar algo de la página, si el JS se ejecutase al principio, no tendría nada que modificar. Es decir, el JS no puede modificar algo que aún no existe.

Por eso, en general, lo recomendable es **colocar el JS al final** del documento HTML (justo antes del tag `</body>` que cierra).



# Ejercicios



# Ejercicio 1

1. Crear una carpeta en el Escritorio (o donde prefieran) con el nombre `Clase07_Ejercicio1`.
2. Abrir dicha carpeta en **Visual Studio Code**.  
Esto se puede hacer yendo al menú: `File > Open Folder` en Windows o `File > Open` en Mac.
3. Desde VSC, crear un archivo llamado `index.html` dentro de la carpeta.
4. Desde VSC, crear una carpeta `js` y dentro de la misma el archivo `app.js`.
5. Linkear el archivo HTML con el archivo JS usando el “Método 2” visto anteriormente.





## Ejercicio 1 (cont)

6. Guardar dentro de una variable llamada `result` el resultado de la siguiente expresión:

$$\left(\frac{154 \times 56}{98}\right) + \left(\frac{867 - 56}{13}\right) + 43$$

Ver ayuda en la siguiente slide.

7. Mostrar el contenido de la variable `result` dentro de un `alert` cada vez que se cargue la página (en el load), con el texto: *“El resultado es...”*.

El resultado debería darles `193.3846153846154`.



# Ejercicio 1 (cont)

Por un tema de comodidad a la hora de escribir la ecuación, tal vez les puede ser útil crear dos variables auxiliares con el contenido de los primeros dos sumandos: `sumando1` y `sumando2`.

$$\underbrace{\left(\frac{154 \times 56}{98}\right)}_{\text{sumando1}} + \underbrace{\left(\frac{867 - 56}{13}\right)}_{\text{sumando2}} + 43$$

$\underbrace{\hspace{15em}}_{\text{result}}$



## Ejercicio 2

Dado el código de abajo, ¿cuál es el tipo y contenido de cada variable?

```
var varA = 2021;  
var varB = "Hack";  
var varC = varA + varB;  
var varD = varA / 0;  
var varE = varE + 0;  
var varF;
```



## Ejercicio 3

¿Funciona el siguiente código?

```
var varA = "Hack ";  
  
var varC = varA + varB;  
  
var varB = "Academy";
```



## Ejercicio 4

Escribir en la consola una sentencia que despliegue un `alert` en la página con el siguiente texto:

**Bienvenido a "nuestro" 'sitio' web.**

Nota: la palabra "nuestro" debe aparecer entre comillas dobles y 'sitio' con comillas simples.