

# CURSO DE JAVASCRIPT

**Corporación Ecuatoriana para el Desarrollo de la  
Investigación y la Academia**

**Ing. María Fernanda Granda J., PhD**

**11-11-2025**

**“Conectando ideas,  
transformando  
sociedades”**

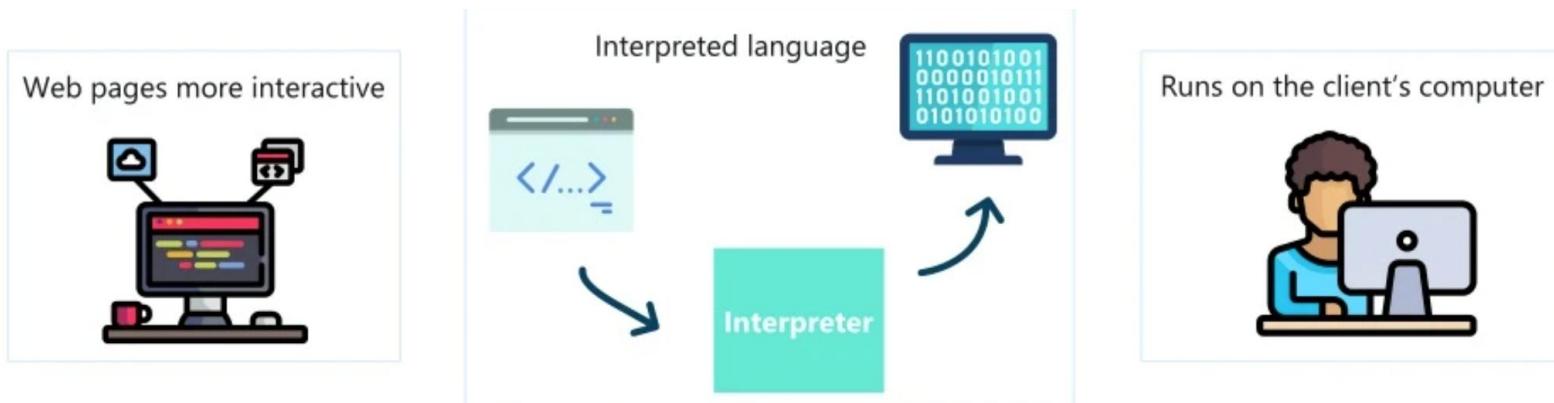
# Evaluación Diagnóstica

## Módulo 1: Introducción a JavaScript

- ¿Qué es Java Script?
- Integración con HTML y CSS
- Principios de la programación interpretada.
- Beneficios de JavaScript
- Características del Lenguaje
- ¿Qué es ECMASCIPT?
- Evolución de ECMASCIPT
- Ámbito de aplicación de la programación JavaScript
- ¿Por qué usar JavaScript?
- Sitios Web construidos usando JavaScript
- Conceptos generales de programación

# ¿Qué es JavaScript?

- Es un **lenguaje de programación** que se utiliza principalmente para aportar **dinamismo** a **sitios y aplicaciones web**.
- **Funciona en complemento** con los lenguajes web **HTML** y **CSS**, permitiendo crear nuevas funcionalidad e interacciones avanzadas con los usuarios.



# Integración con HTML y CSS

HTML



CSS

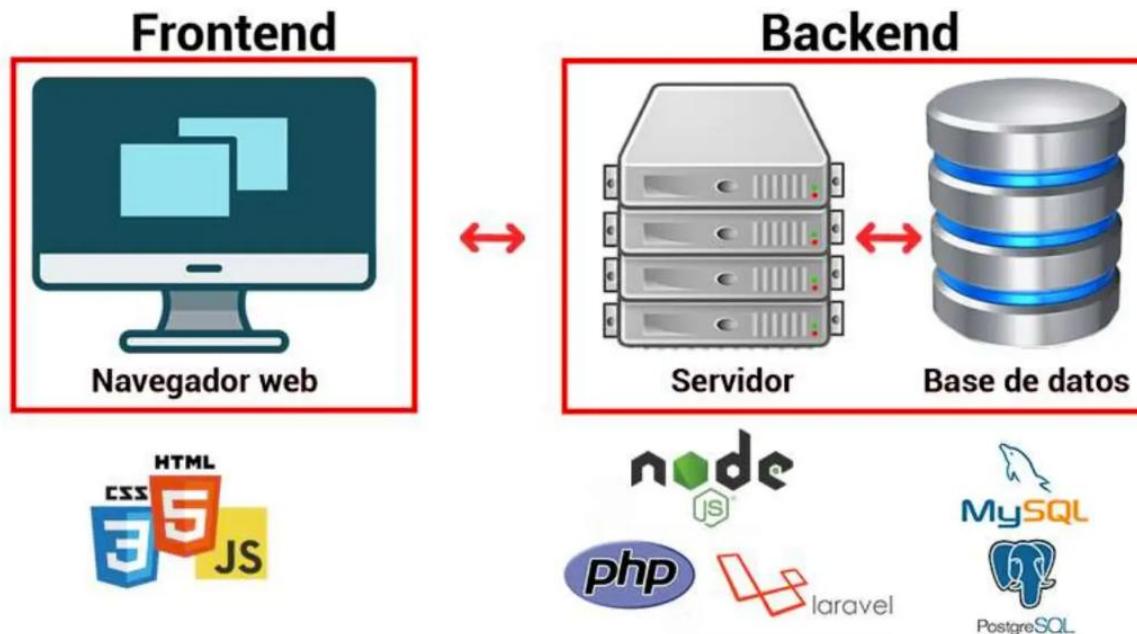


JavaScript



# Front-End y el Back-End

- Permite interactuar con otras aplicaciones o servicios de backend.



# Principios de la programación interpretada

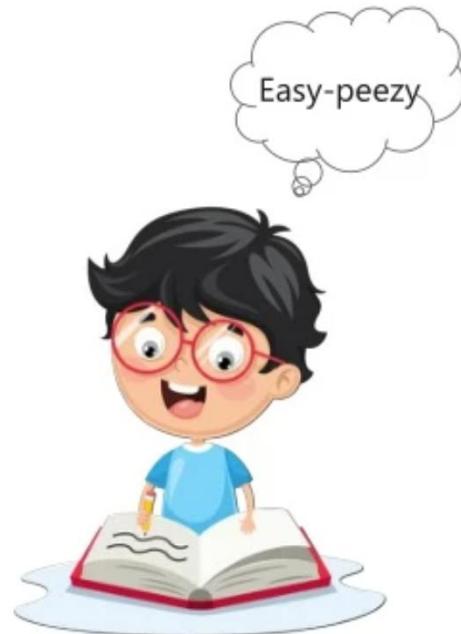


A diferencia de un lenguaje de programación compilado, Javascript es un lenguaje **interpretado**, lo que significa que se ejecuta por medio de un programa intérprete.

En nuestro caso, los **navegadores** son los programas que se encargan de interpretar y ejecutar el código Javascript que creemos.

Cuando accedemos a alguna página a través del navegador, este se encarga de leer y ejecutar todos los archivos que conforman la aplicación (HTML, CSS, JS).

# Beneficios de JavaScript



Fácil de aprender



Velocidad

# Beneficios de JavaScript



Provee un amplio Marco de trabajo



Hace las páginas más interactivas

# Beneficios de JavaScript



No necesita compilar



Independiente de Plataforma

## Características del Lenguaje

- Lenguaje de Programación
- Interpretado
- Orientado a Objetos
- Imperativo
- Case sensitive (diferencia mayúsculas y minúsculas)
- Basado en prototipos /instancias /classless
- Tipado débil
- Lenguaje dinámico

## ¿Qué es ECMASCIPT?

- ECMASCIPT 5.1, 6, 7, 8, NEXT
- ECMAScript es la norma que estandariza el lenguaje de scripting JavaScript
- Garantiza su interoperabilidad en diferentes navegadores y entornos de desarrollo.
- Mantenida por Ecma International, y su especificación se documenta en ECMA-262.

# Evolución de ECMASCIPT

Edición	Fecha de publicación	Cambios desde la edición anterior	Editor
1	Junio de 1997	Primera edición	Guy L. Steele, Jr.
2	Junio de 1998	Cambios editoriales para mantener la especificación completa alineada con el estándar internacional ISO/IEC 16262	Mike Cowlishaw
3	Diciembre de 1999	Se agregaron expresiones regulares, mejor manejo de strings, nuevo control de declaraciones, manejo de excepciones con try/catch, definición más estricta de errores, formato para la salida numérica y otras mejoras.	Mike Cowlishaw
4	Abandonado	La cuarta edición fue abandonada debido a diferencias políticas respecto a la complejidad del lenguaje. Muchas características propuestas para la cuarta edición fueron completamente abandonadas; algunas fueron propuestas para la edición ECMAScript Harmony.	
5	Diciembre de 2009	Agrega el modo estricto ("strict mode"), un subconjunto destinado a proporcionar una mejor comprobación de errores y evitar constructores propensos a errores. Aclara varias ambigüedades de la tercera edición, y afina el comportamiento de las implementaciones del "mundo real" que difieren consistentemente desde esa especificación. Agrega algunas nuevas características, como getters y setters, librería para el soporte de JSON, y una más completa reflexión sobre las propiedades de los objetos. <sup>11</sup>	Pratap Lakshman, Allen Wirfs-Brock

# Evolución de ECMASCIPT

6	Junio de 2015 <sup>12</sup>	<p>La sexta edición agrega cambios significativos en la sintaxis para escribir aplicaciones complejas, incluyendo clases y módulos, definiéndolos sémanticamente en los mismos términos del modo estricto de la edición ECMAScript 5. Otras nuevas características incluyen iteradores <code>for/of</code> loops, generadores y generador de expresiones estilo <a href="#">Python</a>, funciones de dirección, datos binarios, colecciones (mapas, sets, mapas débiles), y proxies (?) metaprogramación para objetos virtuales y wrappers). Al ser la primera especificación "ECMAScript Harmony", es también conocida como "ES6 Harmony".</p>	Allen Wirfs-Brock
7	Junio de 2016	<p>La séptima edición está en una etapa muy temprana de desarrollo, pero está orientada a continuar con la reforma del lenguaje, aislamiento de códigos, control de efectos y librerías/herramientas habilitadas desde ES6. Nuevas características propuestas incluyen promesas/concurrencia, matemáticas y datos numéricos mejorados, guards y trademarks (una alternativa al tipado estático), sobrecarga de operadores, value types (first-class number-like objects), nuevas estructuras de registro (registros, tuples y vectores tipados), pattern matching, y <a href="#">traits</a>.<sup>13</sup></p>	Brian Terlson
8	Junio de 2017	<p>La 8<sup>a</sup> edición, oficialmente conocida como ECMAScript 2017, fue finalizada en junio de 2017.<sup>[11]</sup> Incluye constructores <code>async/await</code>, los cuales funcionan usando generadores y promesas.</p>	Brian Terlson
9	Junio de 2018	<p>La 9.<sup>a</sup> edición, oficialmente conocida como ECMAScript 2018, incluye operadores rest/spread para variables (tres puntos: <code>...identificador</code>), iteracion asincronica, <code>Promise.prototype.finally()</code></p>	Brian Terlson
10	Enero de 2019	<p>La 10.<sup>a</sup> edición, oficialmente conocida como ECMAScript 2019, incorporó <code>Array.flat()</code>, <code>Array.flatMap()</code>, <code>String.trimStart()</code>, <code>String.trimEnd()</code>, errores opcionales en el bloque <code>catch</code>, <code>Object.fromEntries()</code>, <code>Symbol.description</code></p>	Mathías Bynens

# Ámbito de aplicación de la programación JavaScript

- **Desarrollo web front-end:** crear interfaces de usuario interactivas y dinámicas. Permite agregar funcionalidades como validación de formularios, menús desplegables, animaciones y efectos visuales, y mucho más.
- **Aplicaciones web de una sola página (SPA):** junto con frameworks como Angular, React o Vue.js, permite desarrollar aplicaciones web que cargan su contenido dinámicamente sin recargar toda la página.

# Ámbito de aplicación de la programación JavaScript

- **Desarrollo web back-end:** Con Node.js, JavaScript se puede usar para desarrollar el lado servidor de aplicaciones web, manejando conexiones a bases de datos, lógica de negocio y más.
- **Desarrollo de aplicaciones móviles:** Con frameworks como React Native, JavaScript se puede utilizar para desarrollar aplicaciones móviles multiplataforma (iOS y Android).
- **Juegos en línea:** es utilizado para crear juegos simples y complejos, tanto en el navegador como aplicaciones de escritorio.

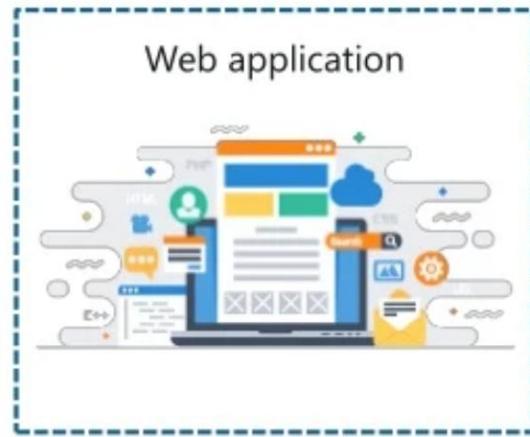
# Ámbito de aplicación de la programación JavaScript

- **Animaciones y gráficos:** en combinación con herramientas como Canvas y WebGL, se usa para crear animaciones, efectos visuales y juegos 2D y 3D.
- **Internet de las cosas (IoT):** se usa para programar dispositivos conectados a través de plataformas como Johnny-Five o Node-RED.
- **Aprendizaje automático:** Bibliotecas como TensorFlow permiten utilizar JavaScript para crear modelos de aprendizaje automático.

# Ámbito de aplicación de la programación JavaScript

- **Interacción con el usuario:** permite crear ventanas emergentes, mensajes de alerta, notificaciones y otros elementos interactivos que mejoran la experiencia del usuario.
- **Validación de datos:** se utiliza para validar la información ingresada por el usuario en formularios antes de enviarla al servidor.
- **Manipulación del DOM:** permite acceder, modificar y manipular elementos del HTML (Document Object Model) de una página web.

# Ámbito de aplicación de la programación JavaScript



# ¿Por qué usar JavaScript?



# Sitios Web construidos usando JavaScript



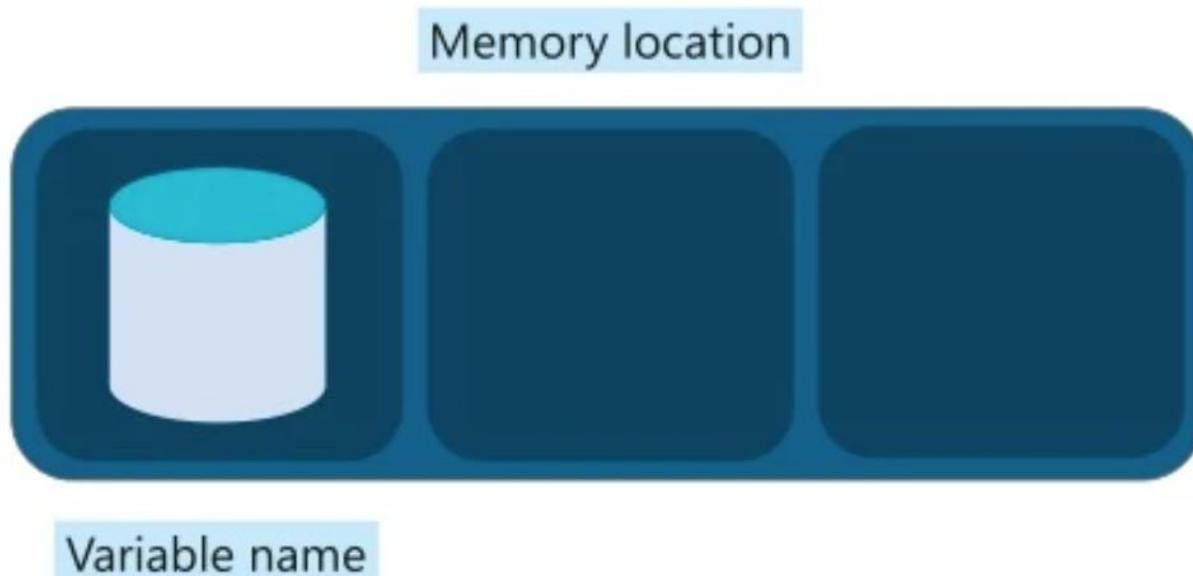
# Conceptos generales de Programación

# Variables

- Es un nombre dado a un espacio de memoria, el cual actúa como un contenedor para almacenar datos.

Syntax:

```
1 | let age;  
2 | age = 22;
```

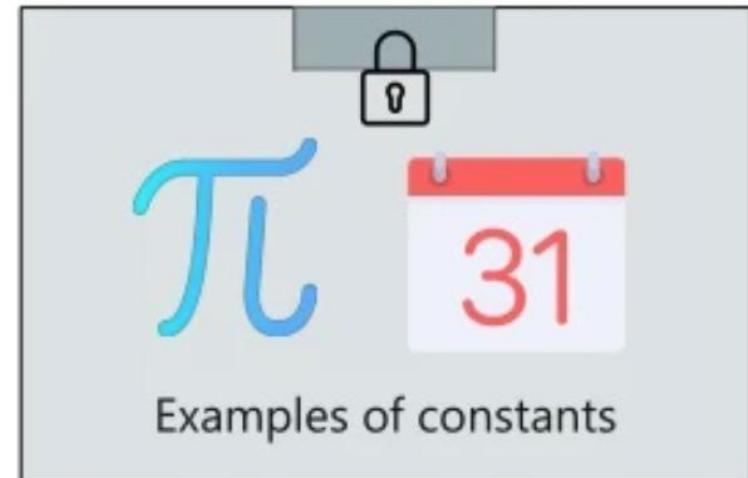


# Constantes

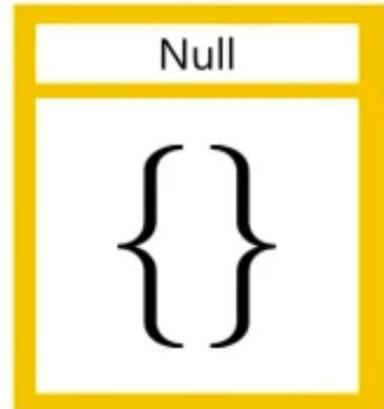
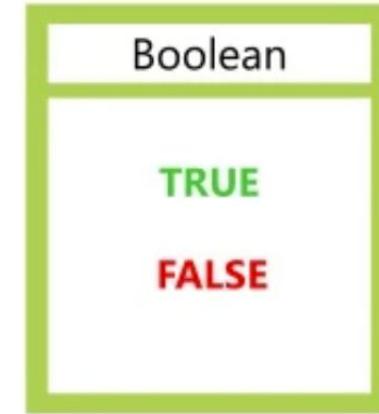
- Son valores fijos que no pueden ser cambiados durante el tiempo de ejecución.

Syntax:

```
1 | const mybirthday;  
2 | mybirthday = '03.08.1996';
```



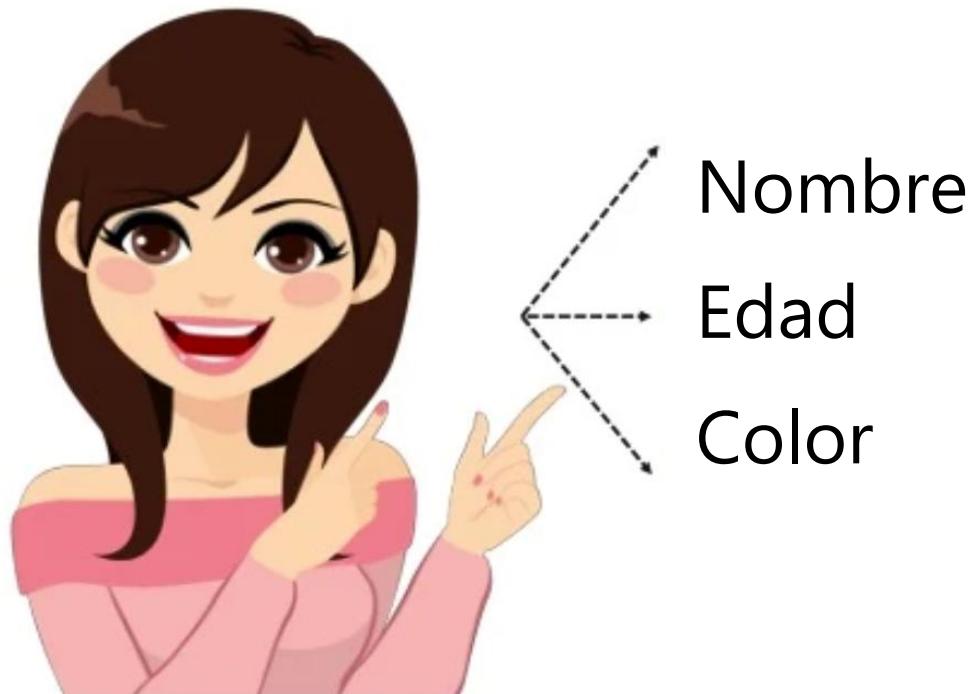
# Datos Básico o Primitivos



# Tipos de datos Referenciados - Objetos

Syntax:

```
1 | let object1 = {};
```

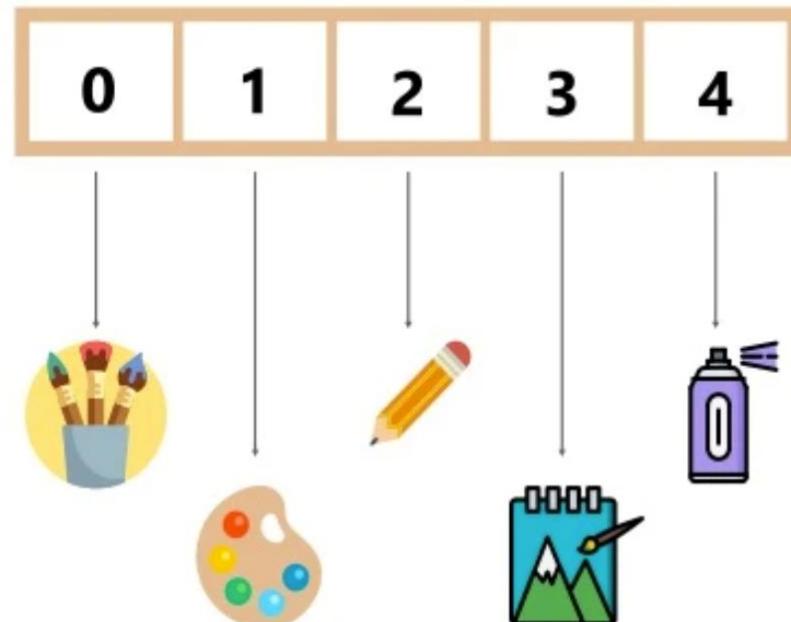


# Tipos de Datos Referenciados - Arreglos

- ❑ Es una estructura que contiene una lista de elementos.
- ❑ Son todos del mismo tipo como integer o string.

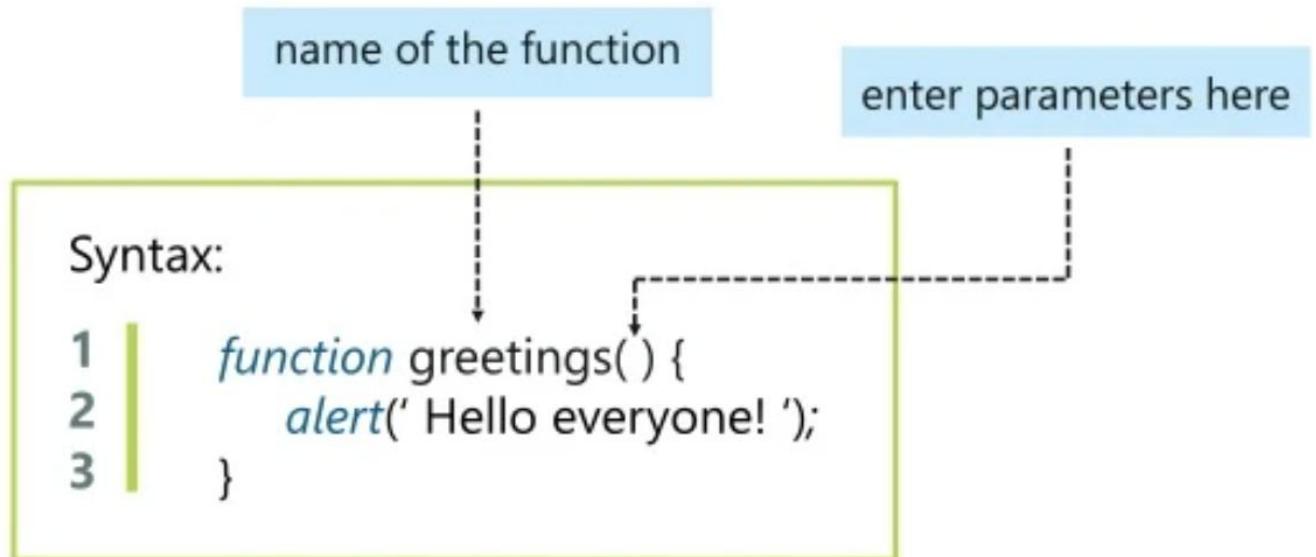
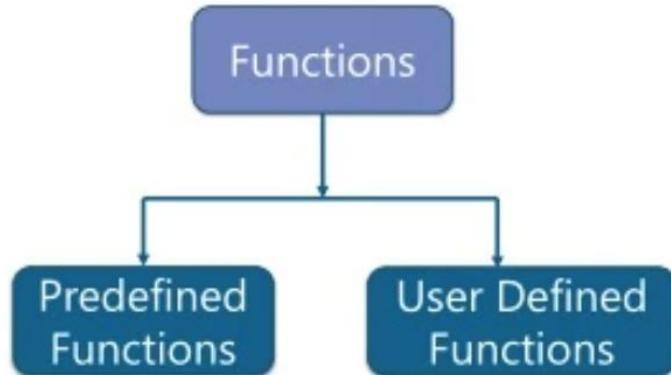
Syntax:

```
1 let arr[ ];  
2 let arr = new Array();
```



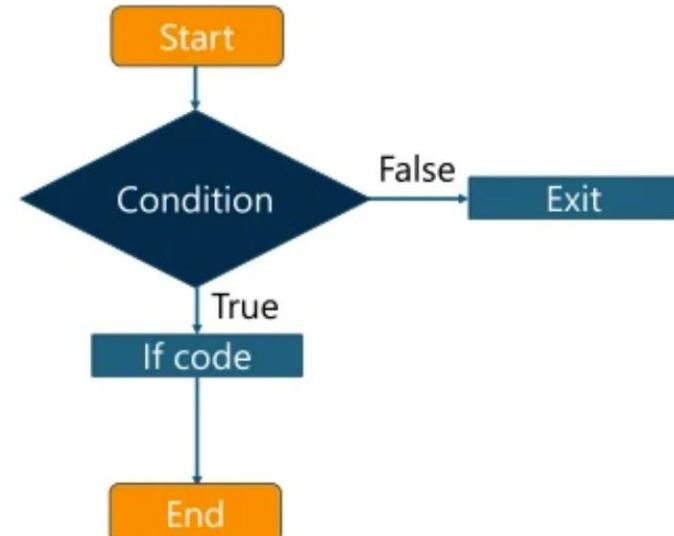
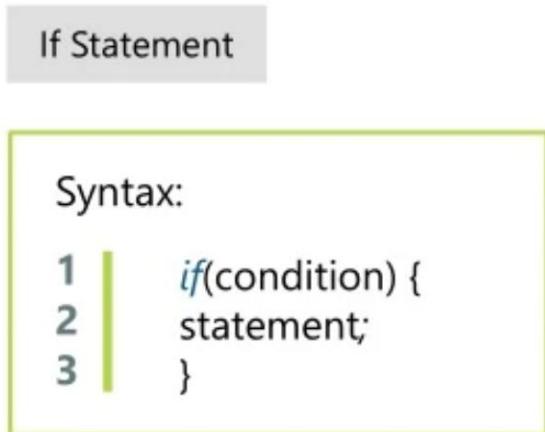
# Tipos de Datos Referenciados - Funciones

- Es un bloque organizado de código, que se puede reusar para ejecutar acciones.



# Sentencias Condicionales

- ❑ Es un conjunto de reglas ejecutadas si una condición es cumplida.
- ❑ Es una sentencia if-then



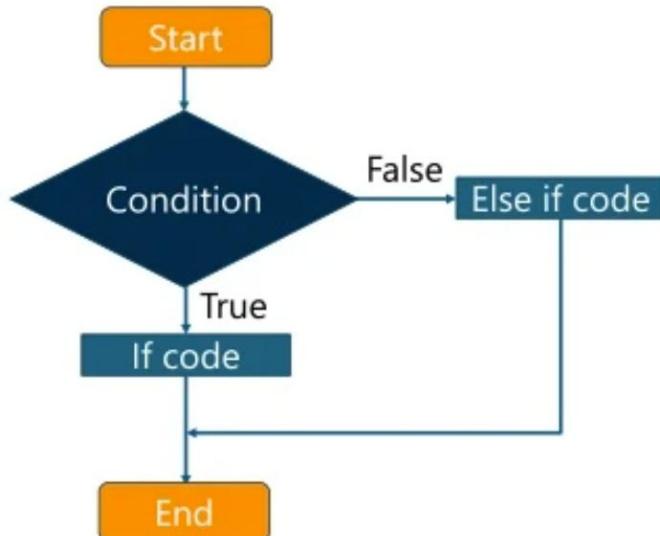
# Sentencias Condicionales

## □ if-then-else

### Else If Statement

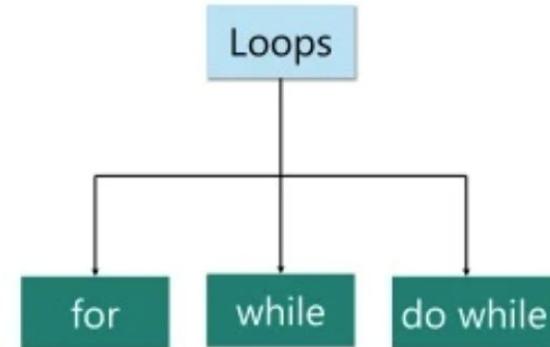
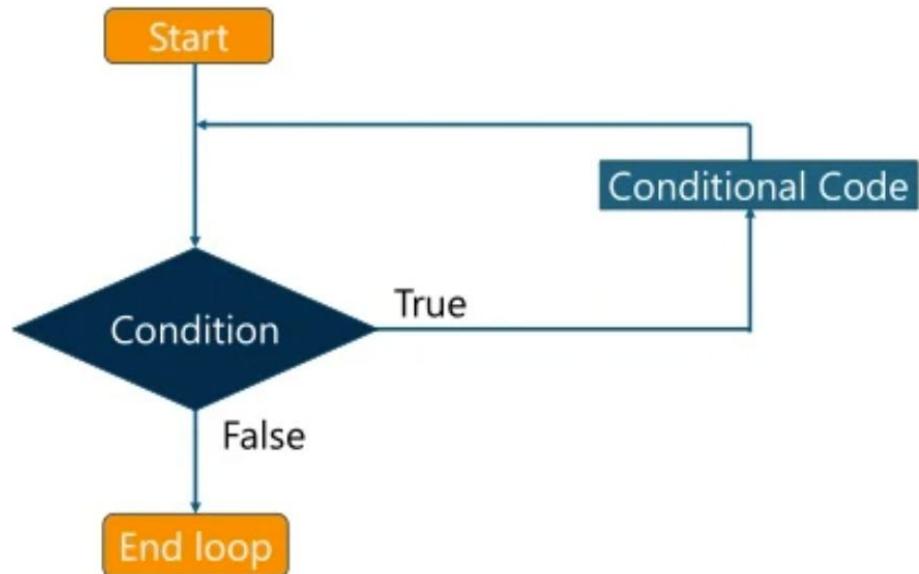
#### Syntax:

```
1  if(condition) {  
2      statement a;  
3  }  
4  else (condition) {  
5      statement b;  
6  }
```



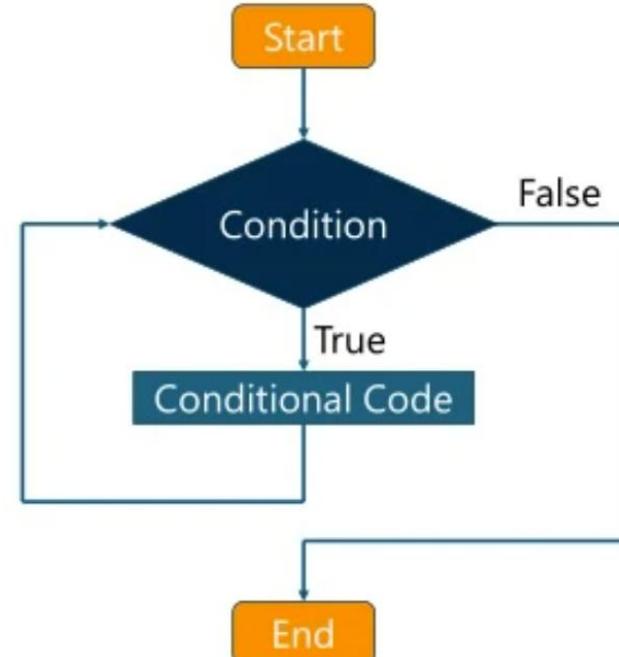
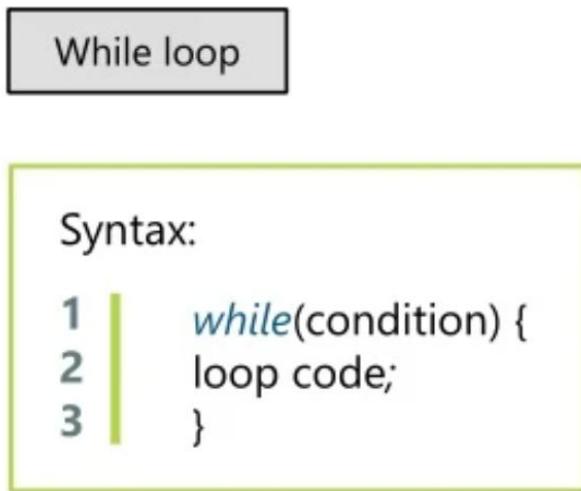
# Bucles

- Son usados para repetir un bloque específico de código hasta que la condición sea cumplida.



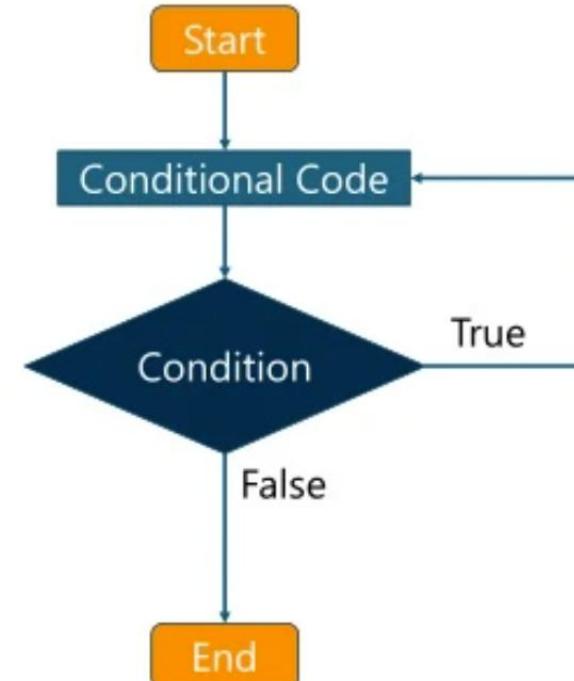
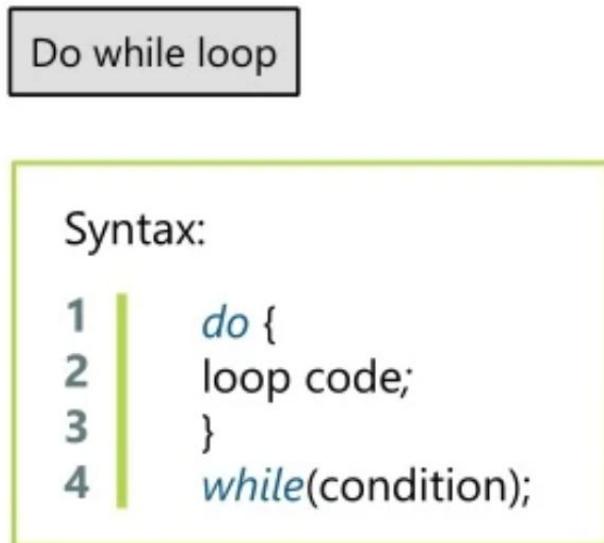
# Bucle While

- Mientras la condición sea verdadera, el código dentro del bucle es ejecutado.



# Bucle Do While

- Primero ejecuta el código, luego verifica la condición y mientras se mantenga verdadera, se sigue ejecutando.



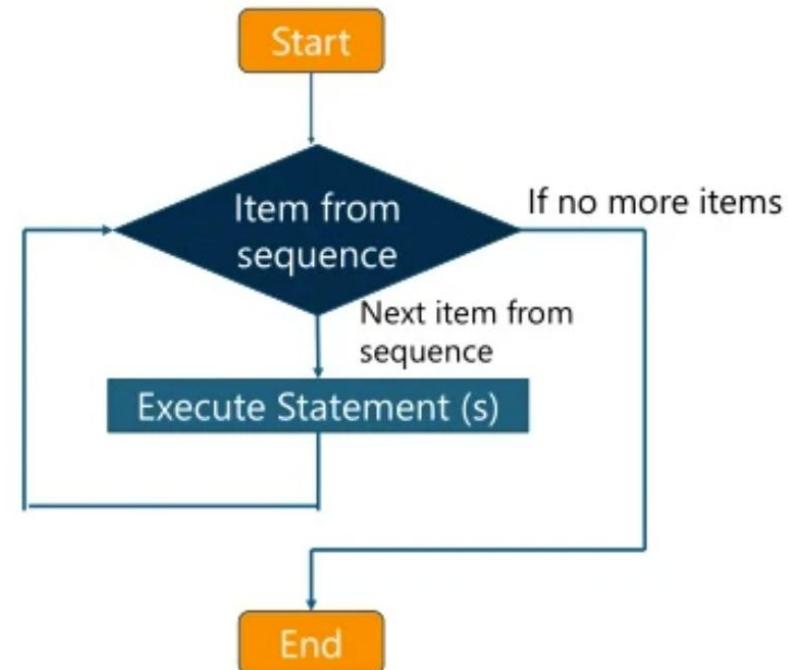
# Bucle For

- Repetidamente ejecuta el código mientras la condición se mantenga verdadera. Verifica la condición antes de ingresar al bucle

For loop

Syntax:

```
1 for(begin; condition; step) {  
2     loop code;  
3 }
```

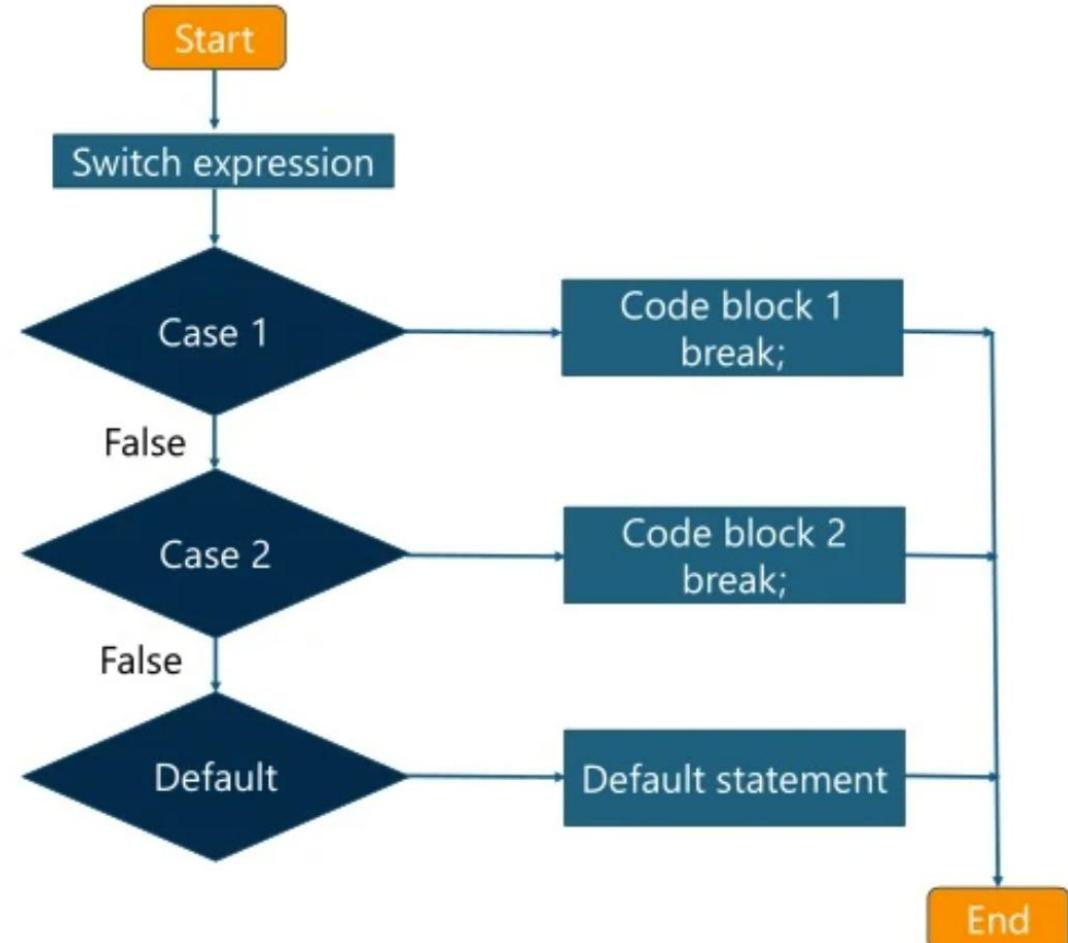


# Switch Case

- ❑ Ejecuta acciones basado en diferentes condiciones

Syntax:

```
1   switch(expression) {  
2       case 1 :  
3           code block 1  
4           break;  
5       case 2 :  
6           code block 2  
7           break;  
8       default :  
9           code block 3  
10      }
```

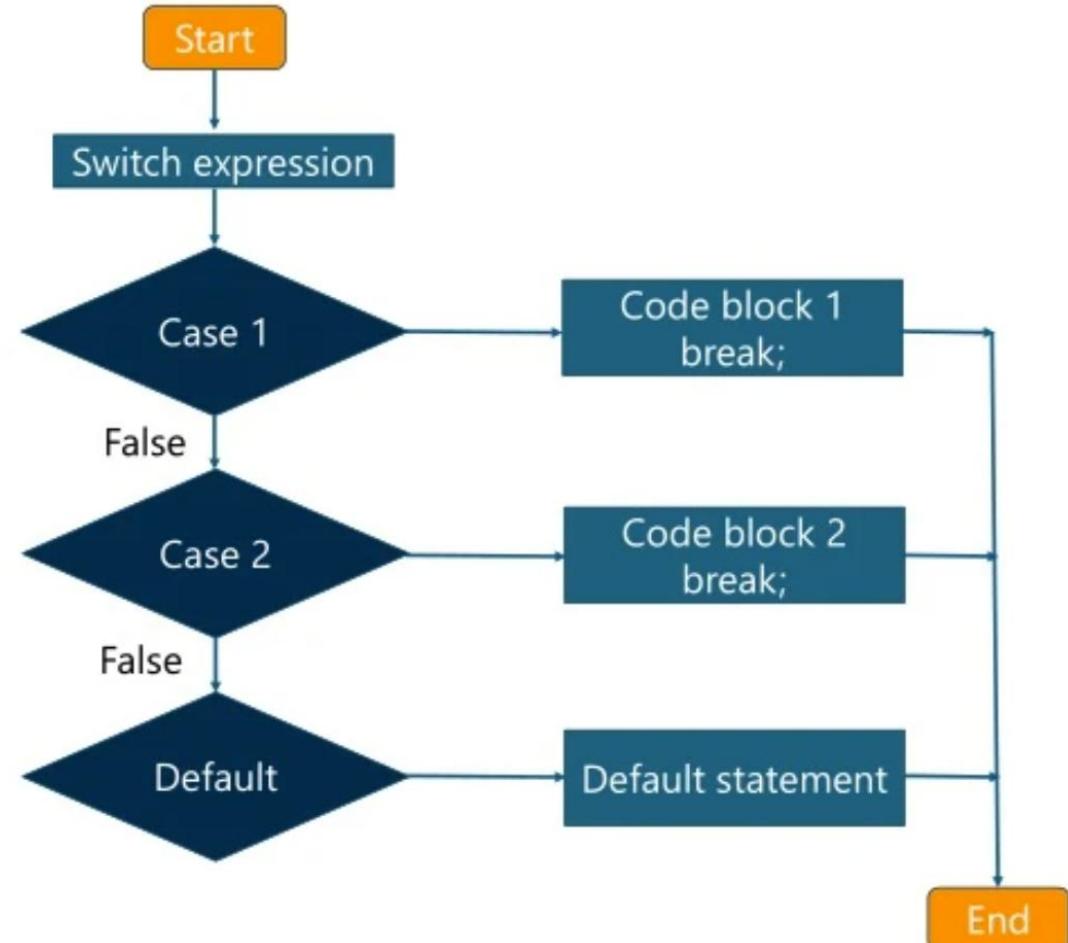


# Switch Case

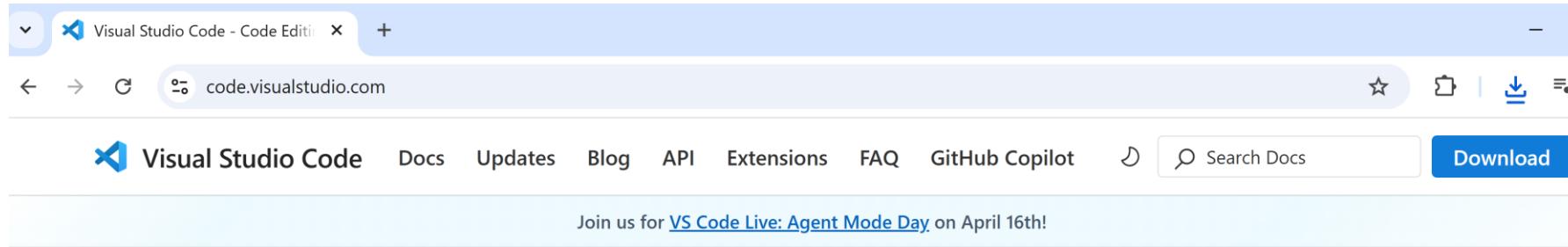
- ❑ Ejecuta acciones basado en diferentes condiciones

Syntax:

```
1   switch(expression) {  
2       case 1 :  
3           code block 1  
4           break;  
5       case 2 :  
6           code block 2  
7           break;  
8       default :  
9           code block 3  
10      }
```



# Editor - Visual Studio Code



**Your code editor.  
Redefined with AI.**

[Download for Windows](#)

[Try agent mode](#)

[Web](#), [Insiders edition](#), or [other platforms](#)



# Editor - Sublime Text

→ C sublimetext.com/3

## Sublime Text

Download

## Download

Sublime Text 4 has been released, and contains significant improvements over this version.

**Version:** Build 3211

- [OS X](#) (10.7 or later is required)
- [Windows](#) - also available as a [portable version](#)
- [Windows 64 bit](#) - also available as a [portable version](#)
- [Linux repos](#) - also available as:
  - [64 bit .deb](#) - [sig](#), [key](#)
  - [64 bit .rpm](#) - signed, [key](#)
  - [64 bit .pkg.tar.xz](#) - [sig](#), [key](#)
  - [64 bit .tar.bz2](#) - [sig](#), [key](#)
  - [32 bit .tar.bz2](#) - [sig](#), [key](#)

# GRACIAS