



BORJA FERNÁNDEZ

P I L D O R A

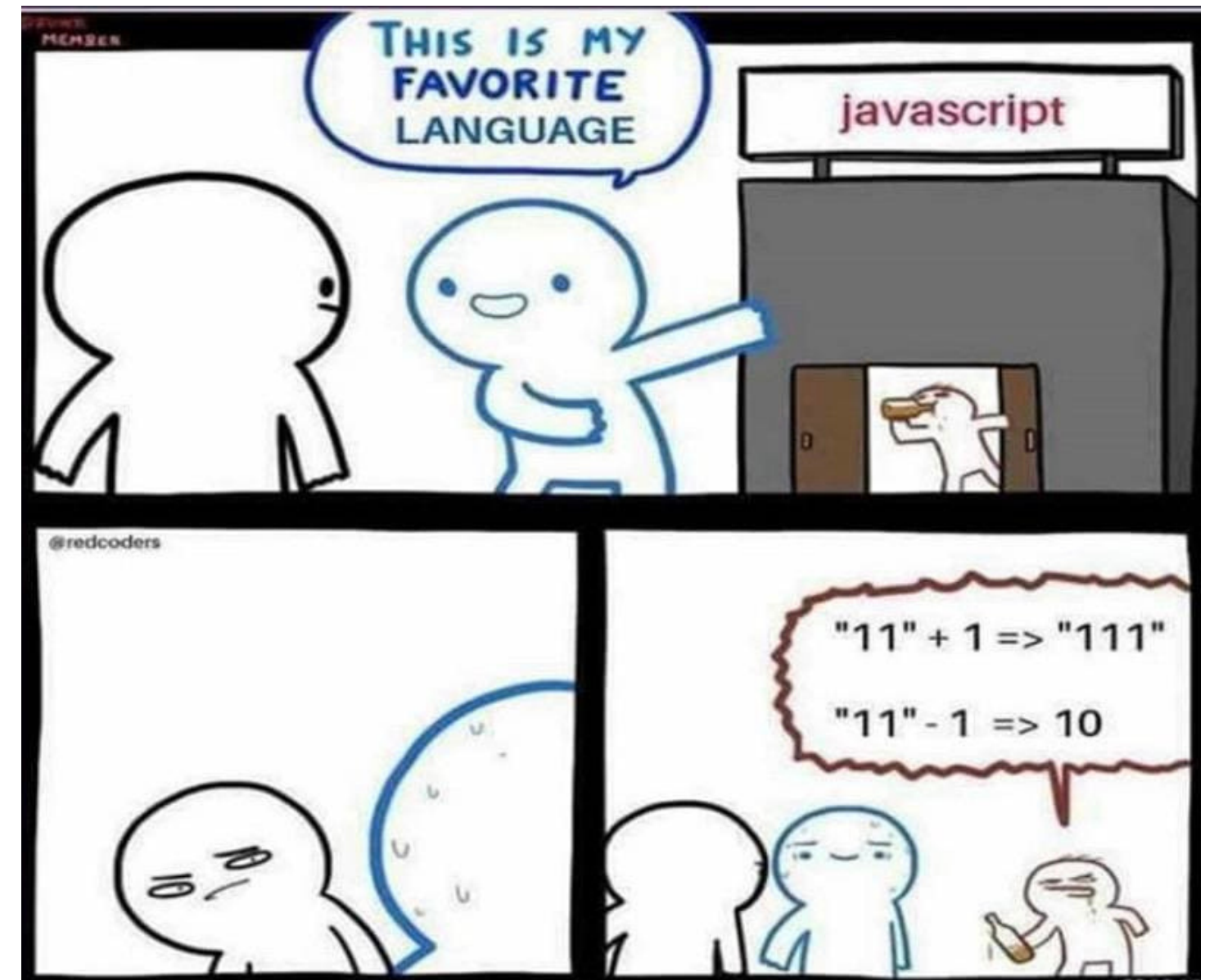
TYPESCRIPT

# ANTES DE NADA

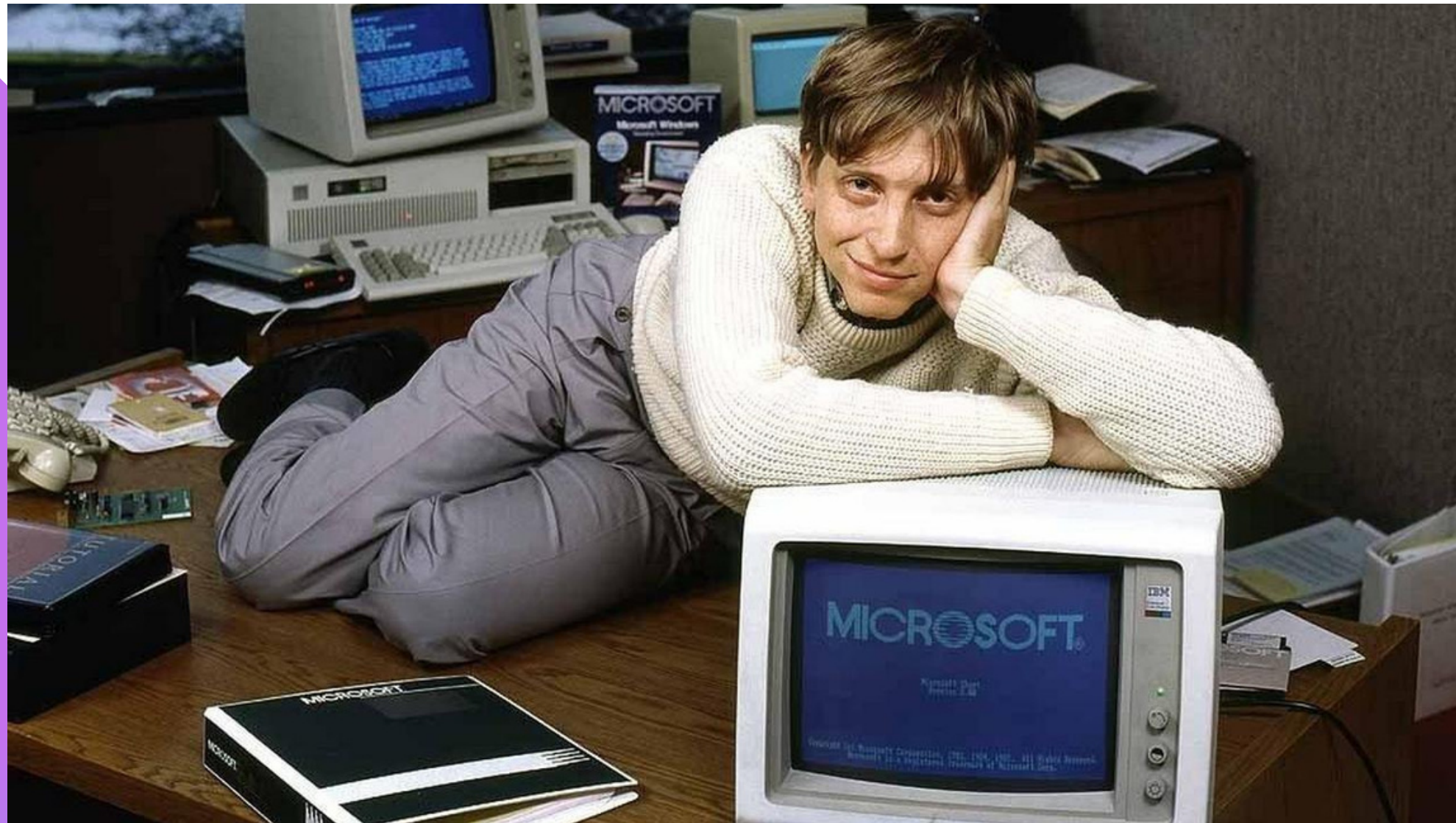
## ¿QUE ES JAVASCRIPT?

JavaScript fue creado por Brendan Eich por encargo de Mozilla como un lenguaje de Scripting para el navegador.

En función que la web crecía con aplicaciones más complejas como por ejemplo google maps facebook etc. JavaScript va evolucionando y poco a poco pasa de ser un lenguaje de Scripting a un lenguaje de propósito general y en consecuencia la popularidad de JS crece a ser el número 1 más usado en el mundo.



# HISTORIA TYPESCRIPT



En el año 2010 los ingenieros que trabajaban en Microsoft estaban desarrollando Outlook y sufrían de los problemas que tenía JS por lo que crearon un lenguaje llamado Type# lo que hacia era poder programar en C# y este código se transpilaba JS

Este equipo de desarrolladores le propusieron a Anders Hejlsberg creador de C# usar Type# para crear un lenguaje de programación que derrocaria a JS y ser el principal lenguaje de la web.



# NI DE BROMA

Anders Hejlsberg dijo que no, intentar derrotar a JS era una batalla perdida y propuso extender a JS. Es decir partiendo de JS añadir nuevas funcionalidades y mejorar las cosas donde es débil. Así no obligamos a la gente a aprender un nuevo lenguaje si no que los que ya saben JS solo tendrían que aprender las funcionalidades nuevas consiguiendo más oportunidades de que la gente escogiera TypeScript.

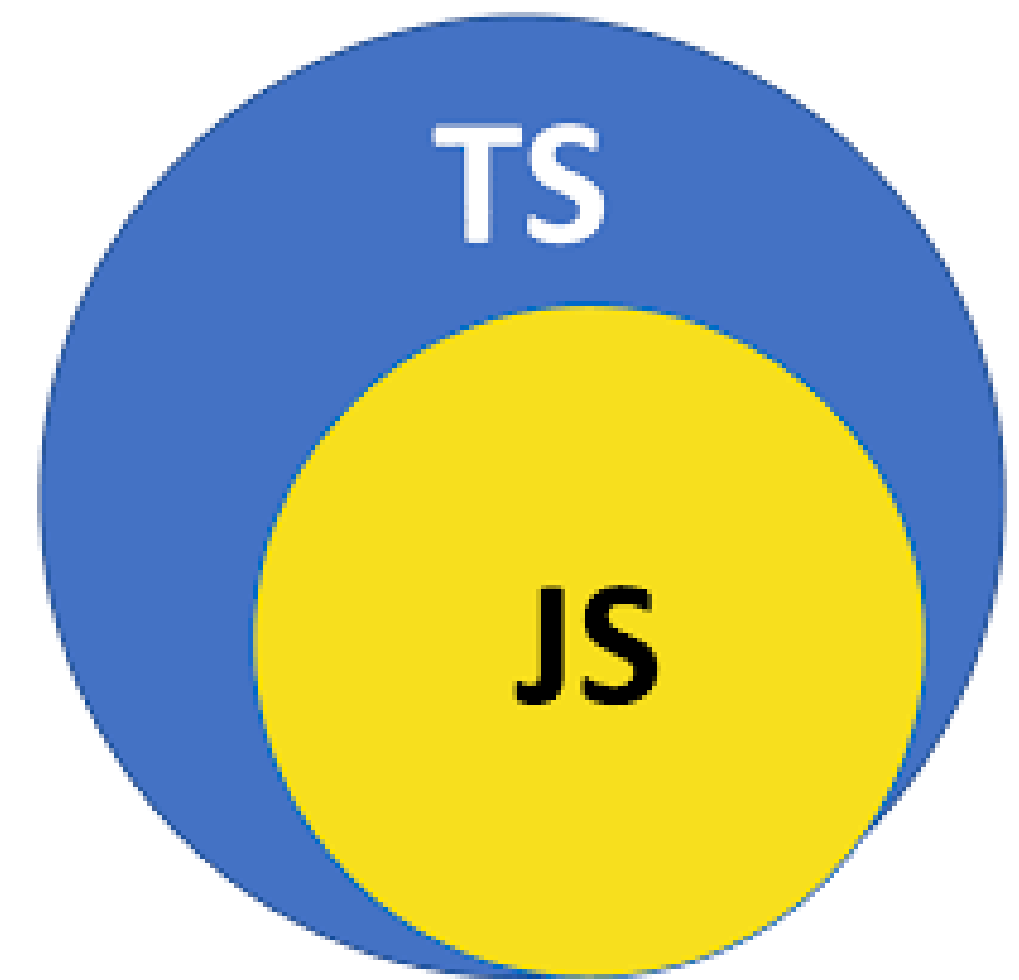


**ANDERS HEJLSBERG**

**CREADOR DE:  
TURBO PASCAL C# Y TYPESCRIPT**

# NACIMIENTO DE TYPESCRIPT

Y así nace TypeScript un “super conjunto” de JavaScript añadiendo funcionalidades nuevas y no un lenguaje de borrón y cuenta nueva olvidando todo lo que aprendiste de JavaScript

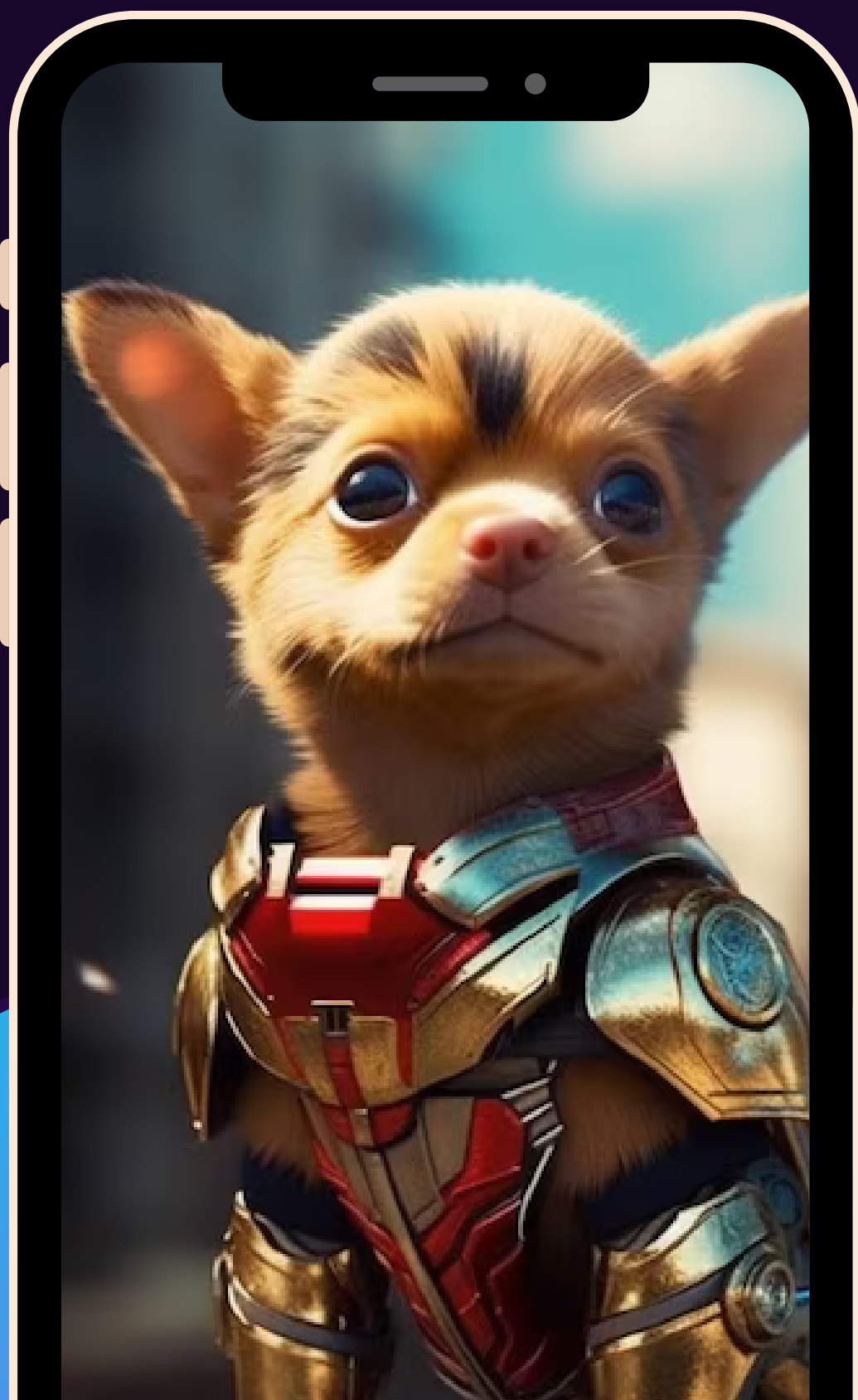


# ¿QUE ES UN SUPER CONJUNTO?

Imaginaros un super conjunto como si caes en un cubo radiactivo y os salen super poderes.

Eso es TypeScript simplemente un JavaScript con superpoderes dando las siguientes habilidades.

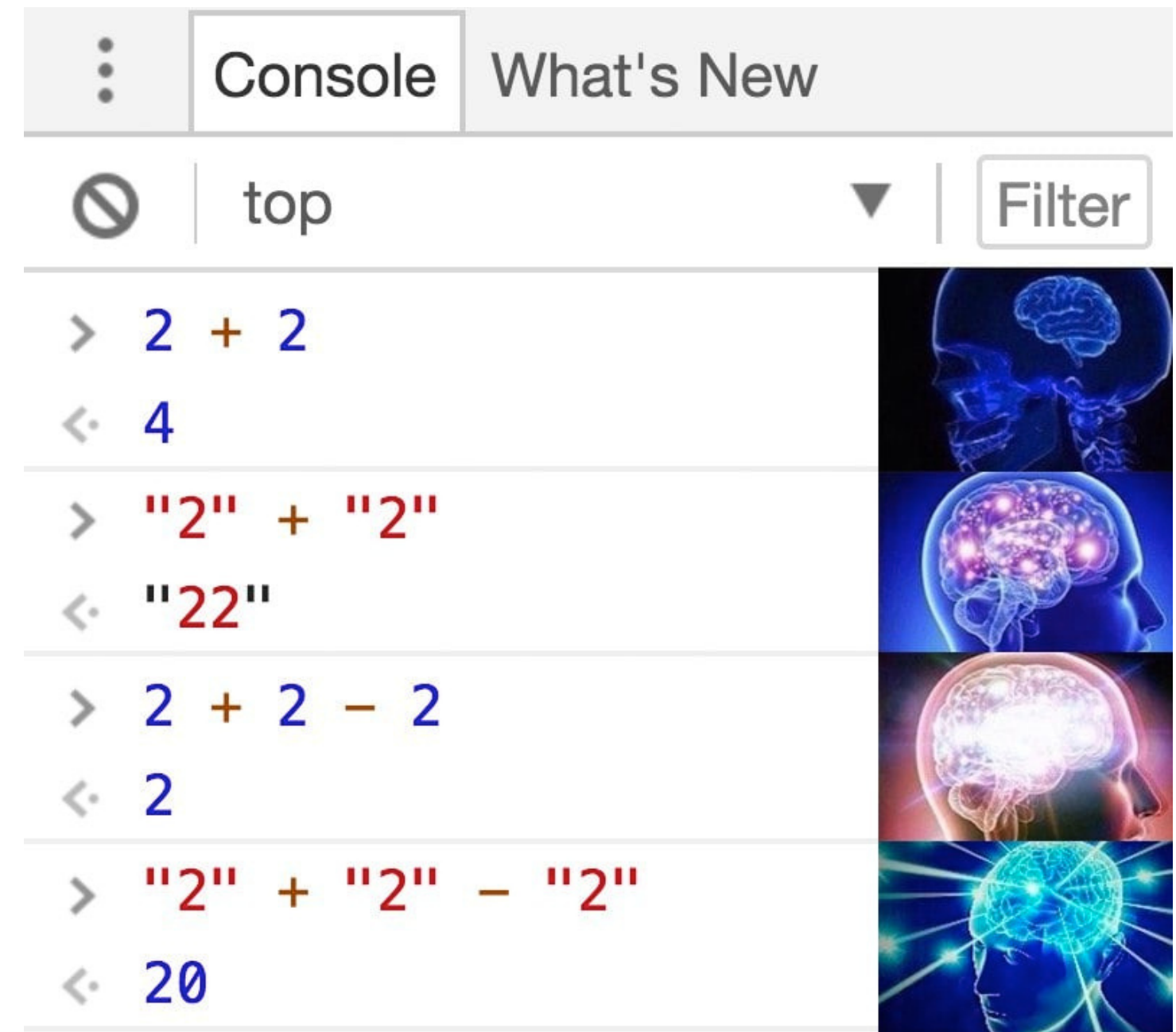
- Tipado estático.
- POO
- Interfaces
- Compatibilidad con JavaScript.
- Mejora la productividad.
- Y mucho más.



# SUPER PODER 1

## LENGUAJE TIPADO

La esencia de programar es manipular datos.  
 Lo que sucede en JavaScript es un lenguaje dinámicamente tipado lo que significa que según las circunstancias permite que un tipo de dato se convierta a otro tipo (auto casting) por lo que da muchos problemas a los programadores ya que JavaScript acepta todo lo que le digas y dichos errores se van a producción (horas extras y noches en vela)



```

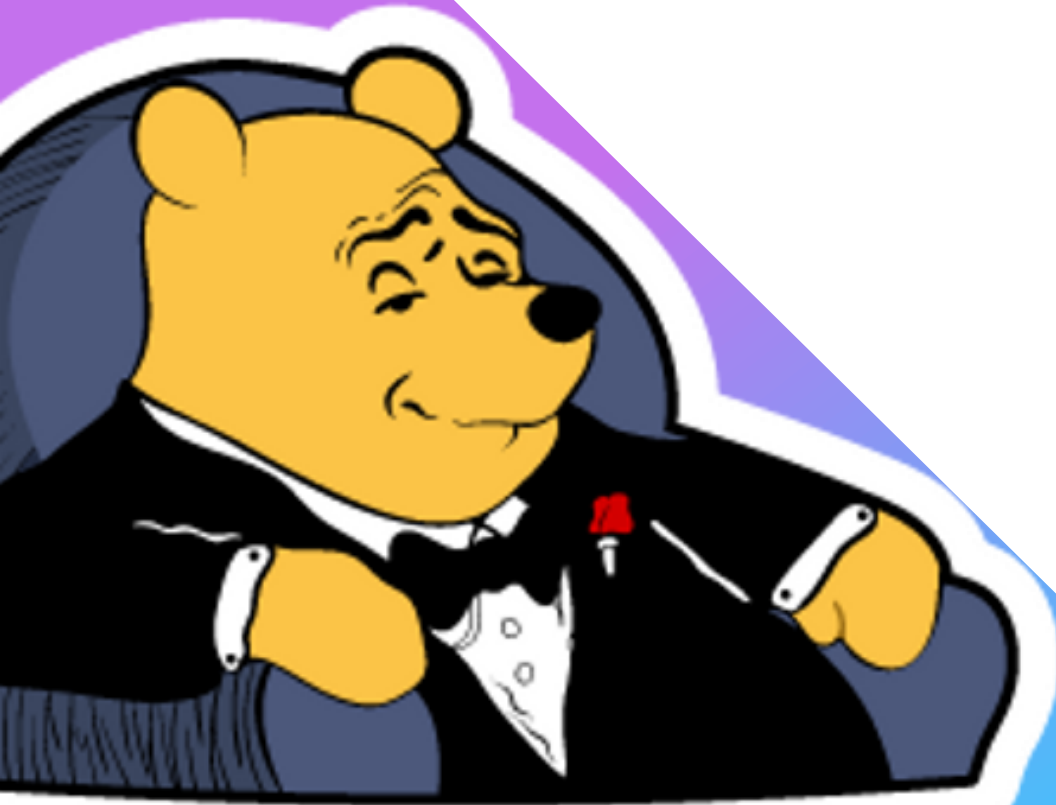
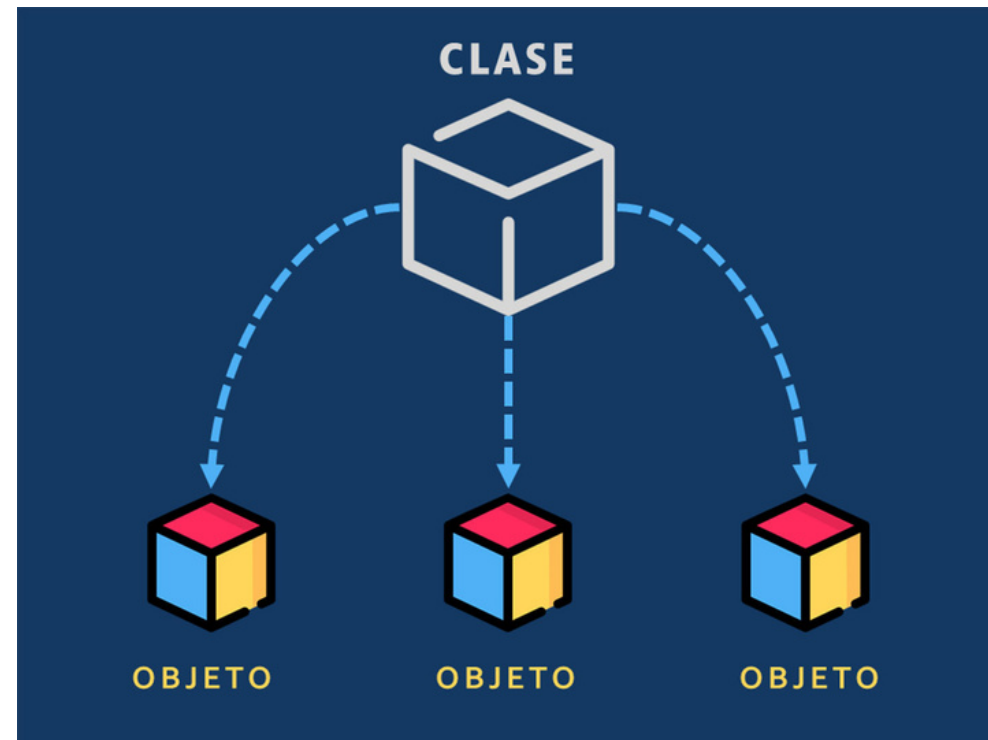
    > 2 + 2
    < 4

    > "2" + "2"
    < "22"

    > 2 + 2 - 2
    < 2

    > "2" + "2" - "2"
    < 20
  
```





## SUPER PODER 2 POO

JavaScript no tiene POO como tal y se basa en prototipos o comúnmente llamada POO basada en prototipos.

TypeScript añade clases y interfaces mejorando la programación en este paradigma y gracias al tipado estático mejora su comprensión y mantenimiento



# POO BASADA EN CLASES VS POO EN PROTOTIPOS

## CLASS

La POO basada en clases utiliza clases como plantilla para crear objetos con su estructura y comportamiento.

La herencia utiliza el modelo de las clases para reutilizar propiedades y métodos de clases ya existentes



## PROTOTYPE

La POO basada en prototipos utiliza objetos directamente y estos se usan como prototipo para otros objetos.

La herencia se logra a través de la propiedad prototype y este puede heredar propiedades y métodos de otros objetos al establecer su prototipo

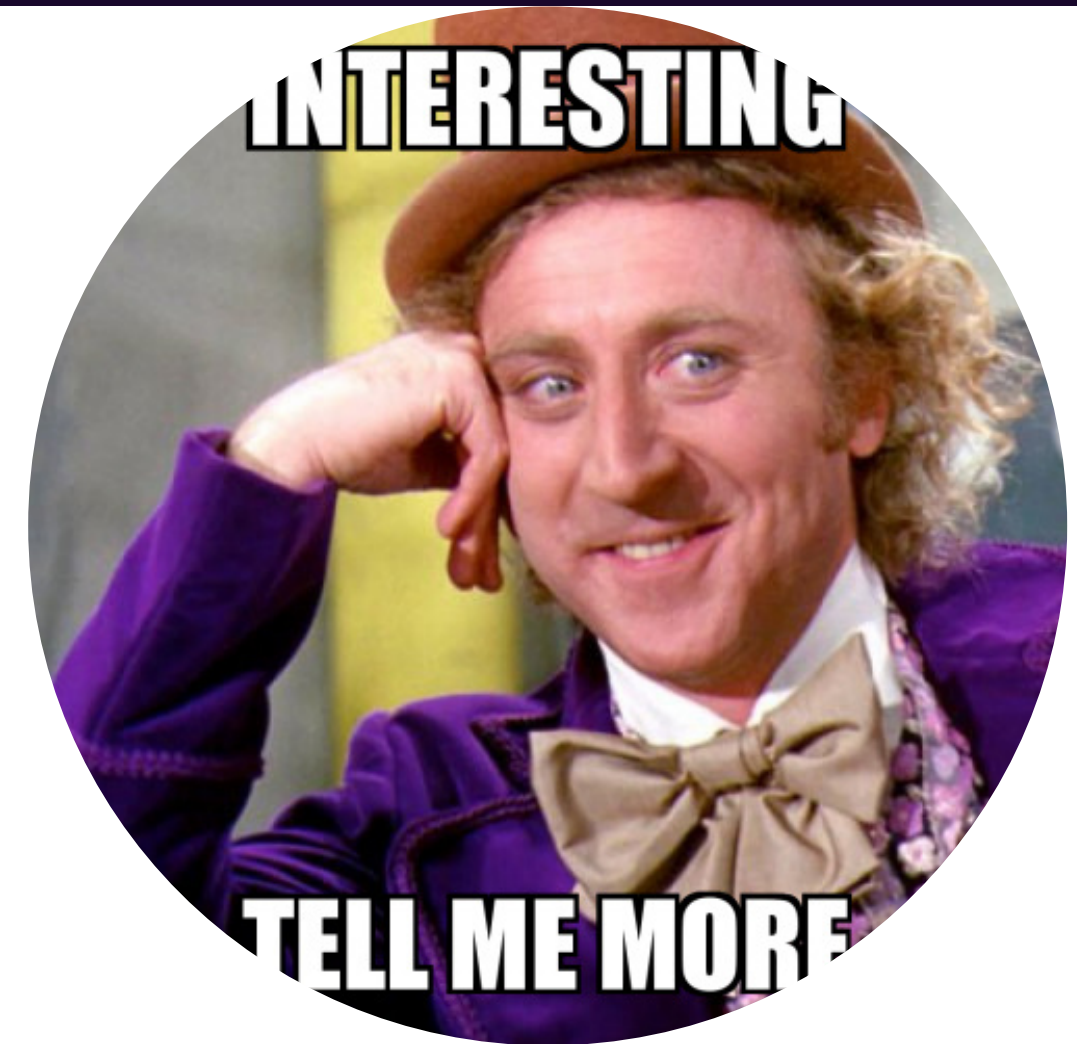
# SUPER PODER 3 INTERFACES

Las interfaces son como moldes que establecen las características que los objetos deben cumplir. Esto asegura que las clases que implementan una interfaz tengan las propiedades y métodos requeridos.



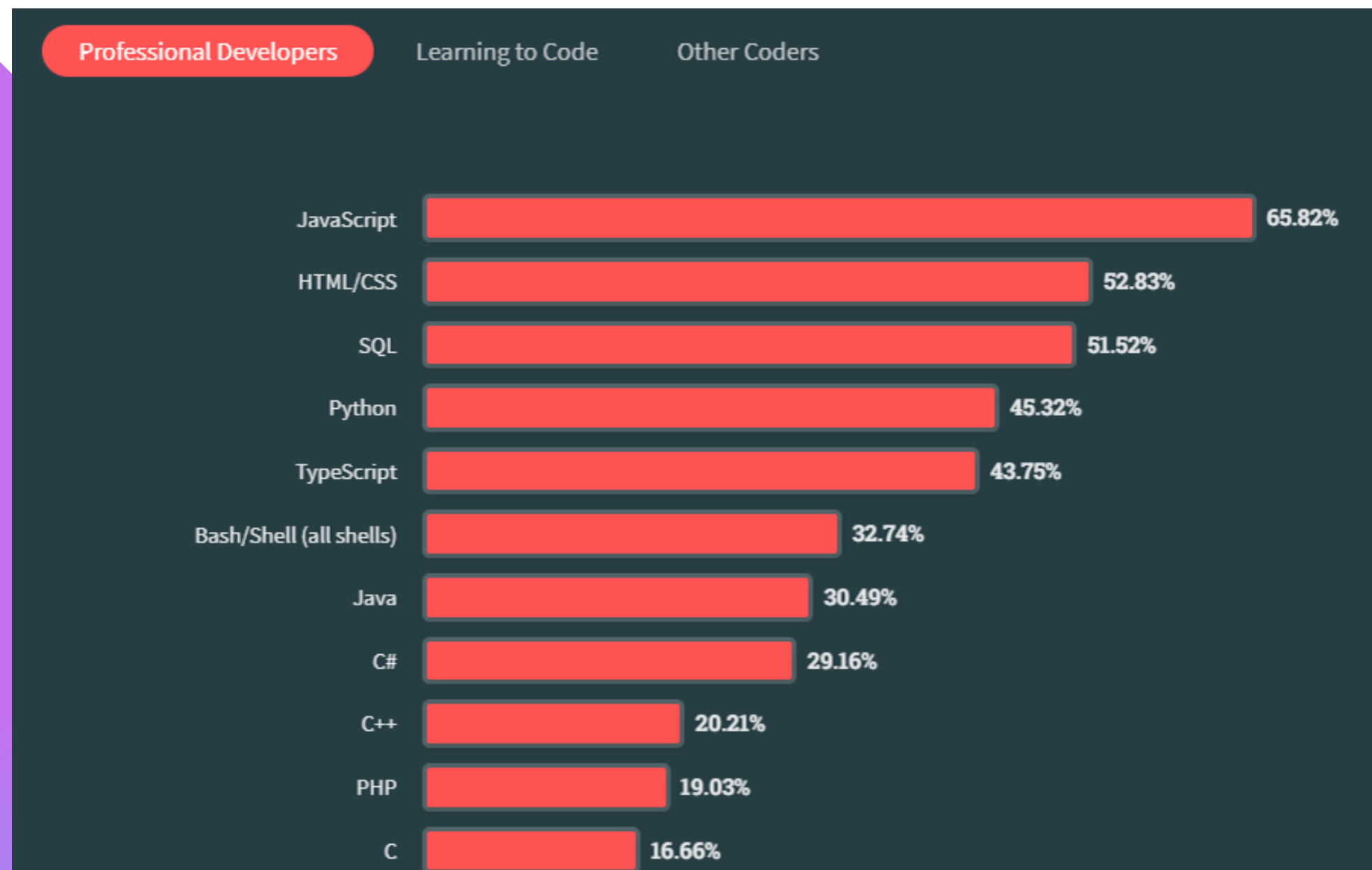
# TYPESCRIPT POPULARITY

Typescript's popularity increased significantly over the last year, moving to third place in Github's ranking of most used languages.





# SURVEY



In the latest Stack Overflow Developer Survey typescript was the fifth most used tool worldwide and the fourth most loved language

This popularity is due to the fact that this programming language facilitates auto-completion, is compatible with JavaScript and improves the readability and refactoring of our code.

# MOST POPULAR ALTERNATIVES TO TYPESCRIPT



**Dart**

*Open source by Google  
Syntax similar to Java and C#*



**CoffeeScript**

*Open source  
Syntax similar to Python and  
Ruby*



**Elm**

*Freeware source  
Functional and strongly typed  
language*

# IN CONCLUSION

Opting for TypeScript means equipping our projects with a set of advanced tools that not only improve code quality, but also simplify the development process. The transition to TypeScript is not only a natural evolution from JavaScript, but also an investment in the long-term reliability and scalability of our projects.



**TS**  
**TypeScript**