# Colors & Elements

# Chalk Arrows

# Titles

## SubTitles

# Title

### SubLevel

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book

```
export class TokenService {

  constructor() { }

  saveToken(token: string) {
    // your token
  }

  getToken(token: string) {
    // your token
  }

  clearToken() {
    // your token
}
```

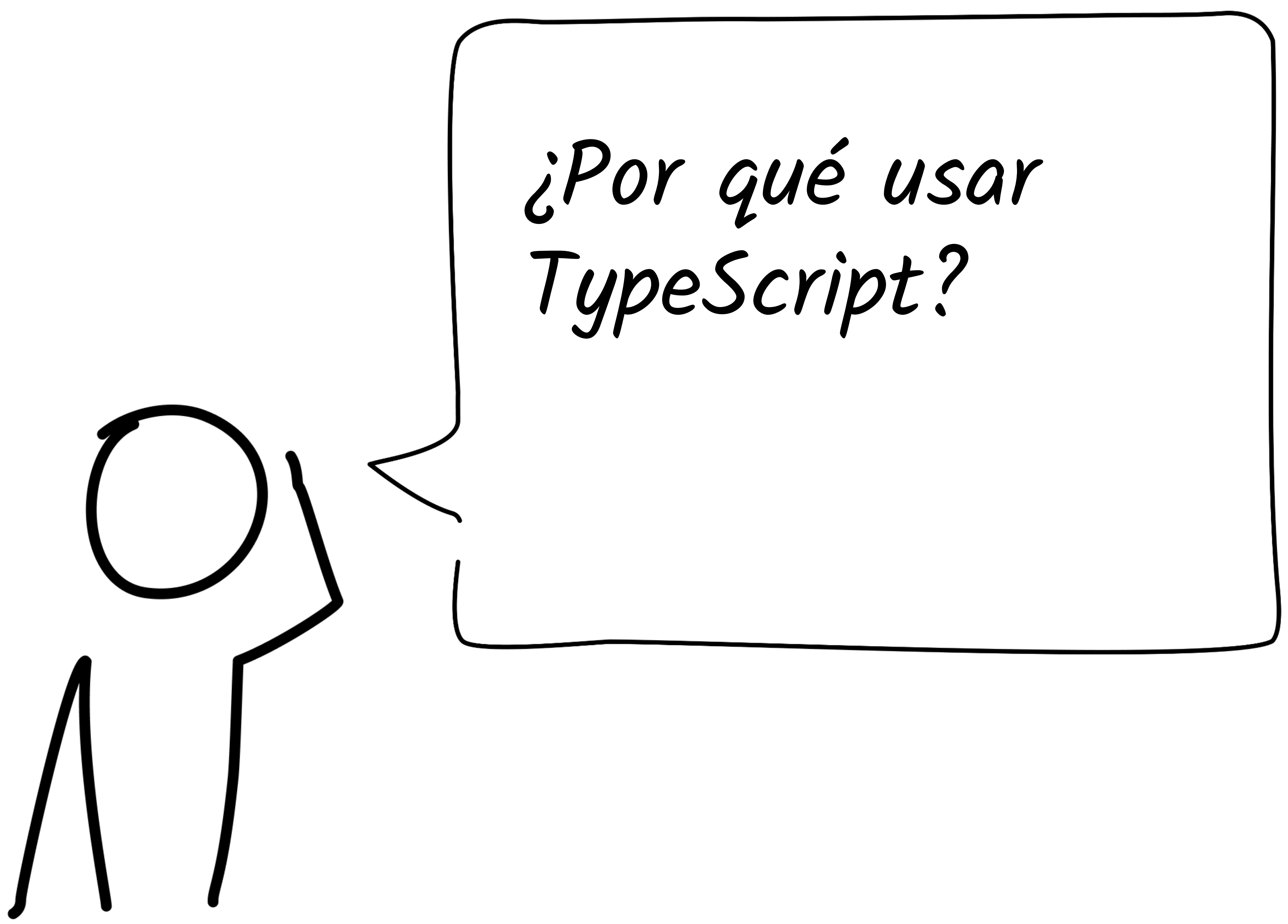
```
export class TokenService {

  constructor() { }

  saveToken(token: string) {
    // your token
  }

  getToken(token: string) {
    // your token
  }

  clearToken() {
    // your token
}
```


¿Por qué usar TypeScript?

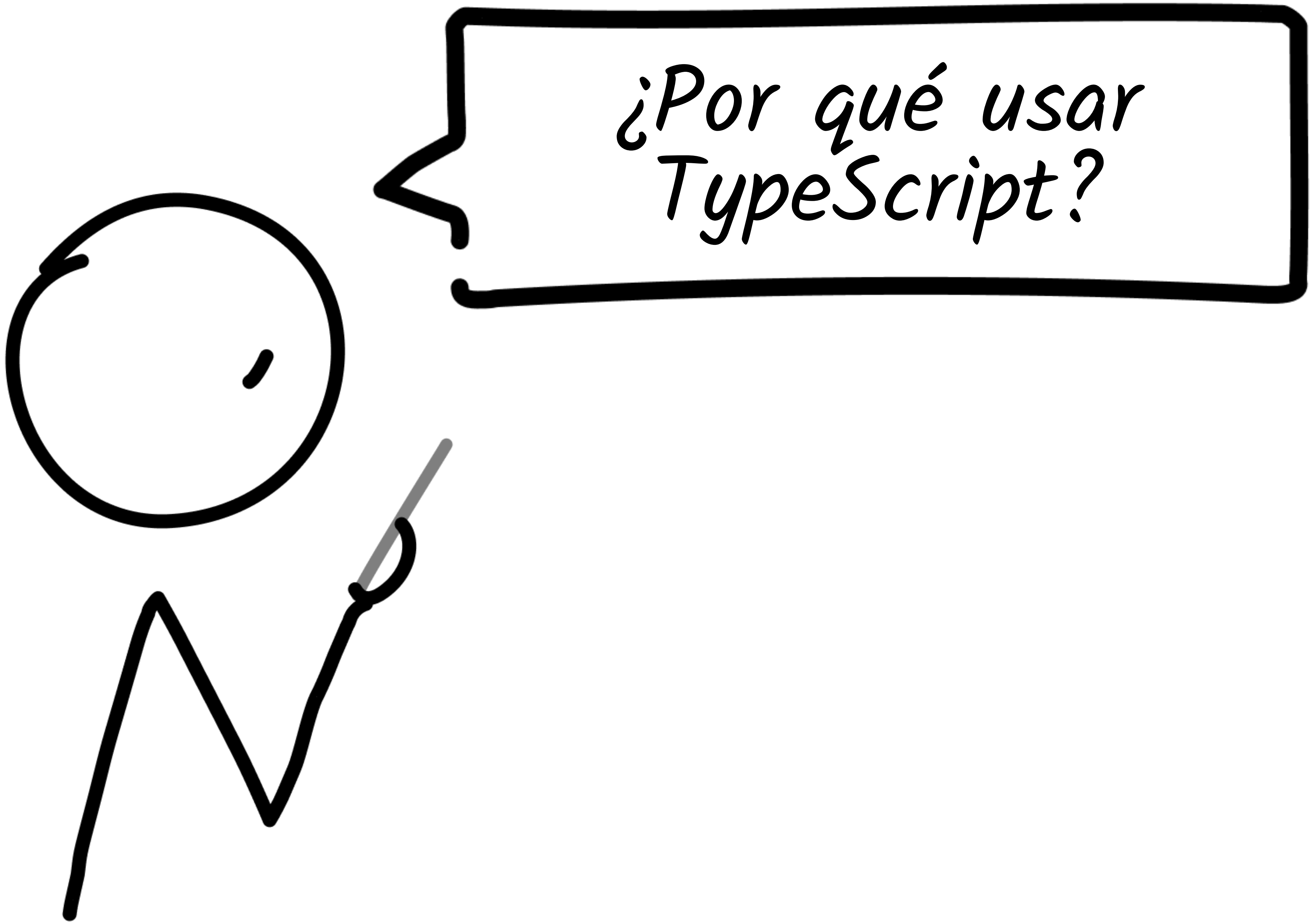
# Fundamentos de TypeScript

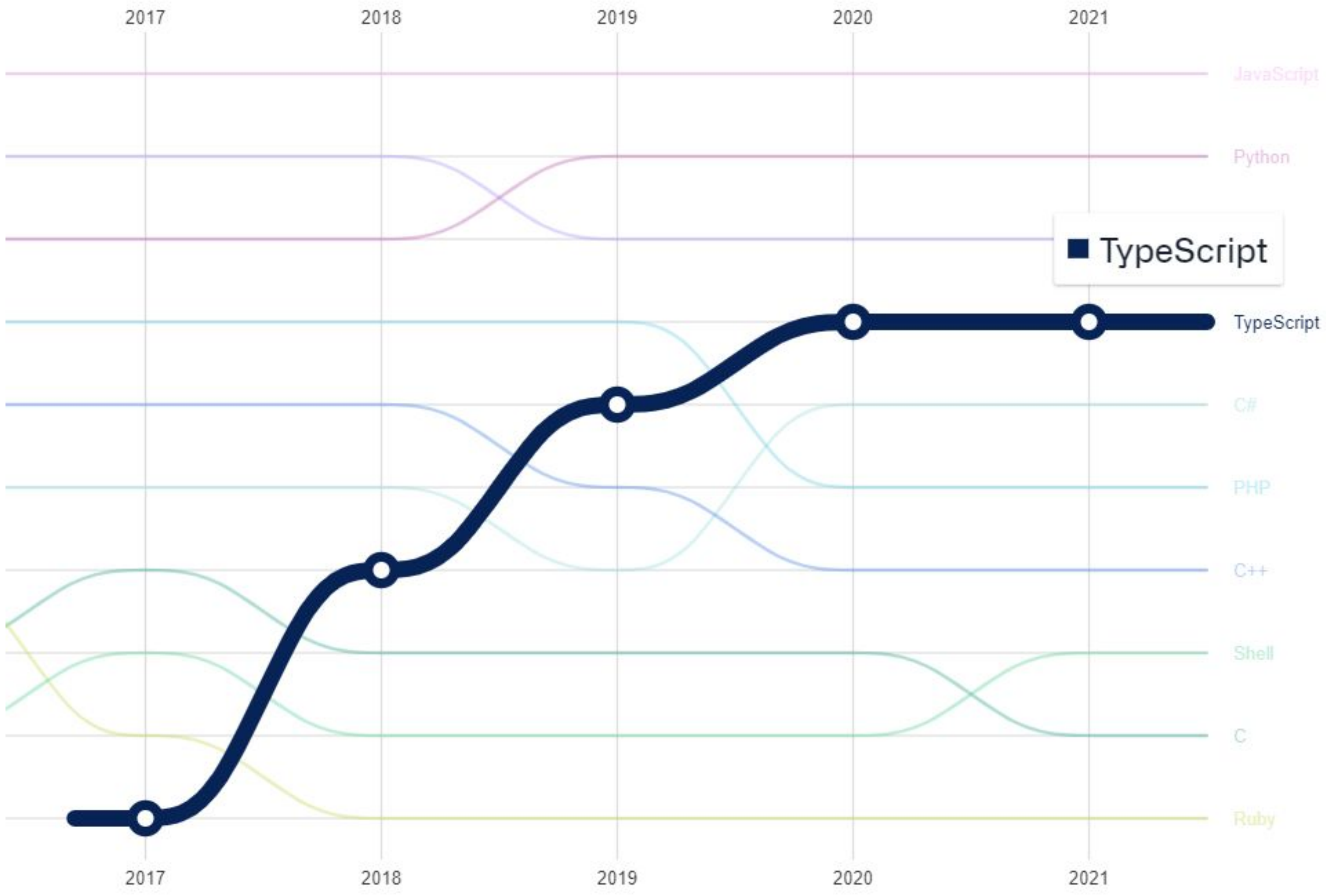
by @nicobytes

# Nicolas Molina

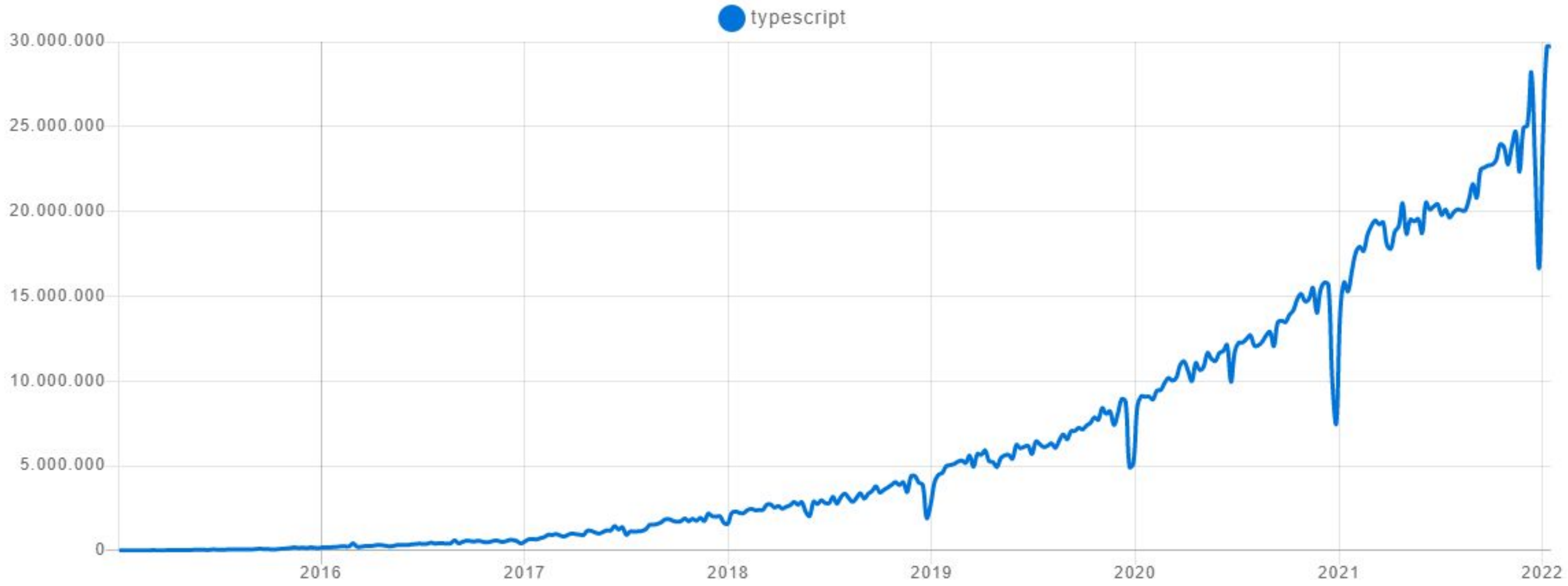
**@nicobytes**

**Google** Developer Expert
Dev and Teacher at **Platzi**


¿Por qué usar TypeScript?



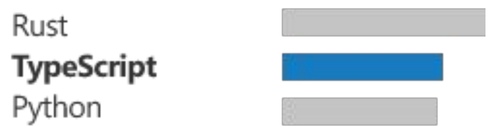
Octoverse 2021


typescript
30.000.000
25.000.000
20.000.000
15.000.000
10.000.000
5.000.000
0
2016
2017
2018
2019
2020
2021
2022

Npm Trends

*“Voted 2nd* ***most loved programming language*** *in the Stack Overflow 2020 Developer survey”*

2020 Developer Survey



*“A static type system can help prevent many* ***potential runtime errors****, especially as applications grow”.*

VueJS

*“Static type checkers like Flow and TypeScript identify certain types of* ***problems before*** *you even run your code”.*

*ReactJS*

Jest

IONIC

nest

SVELTE

Vue.js

TYPEORM

GitHub

*“First, we were surprised by the number of* ***small bugs*** *we found when converting our code”.*

slack

*“Second, we underestimated how powerful the* ***editor integration*** *is”.*

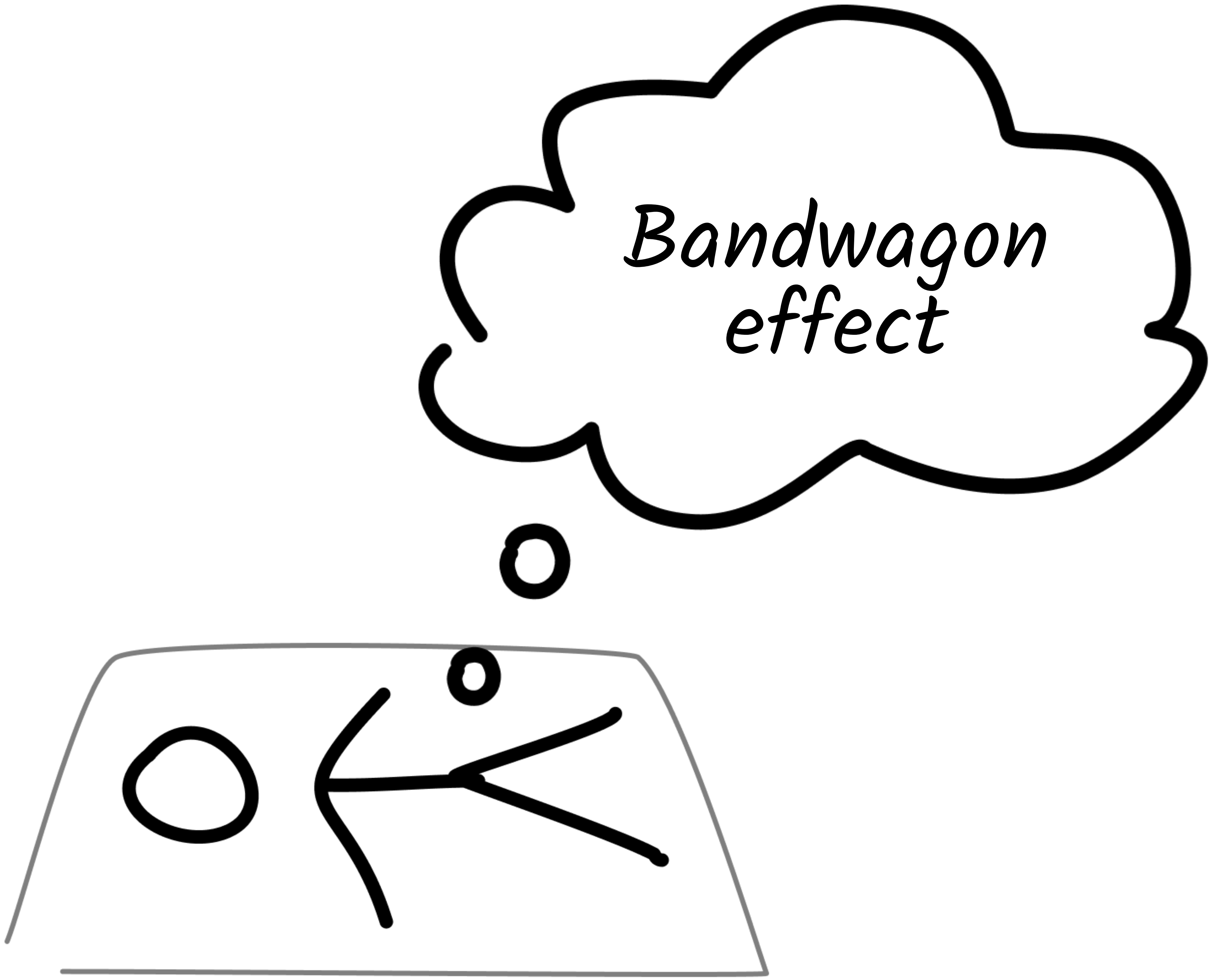
slack

“***38% bugs preventable*** *with TypeScript according to postmortem analysis”.*

airbnb

*“With TypeScript, engineers can* ***move faster more safely****”.*

airbnb

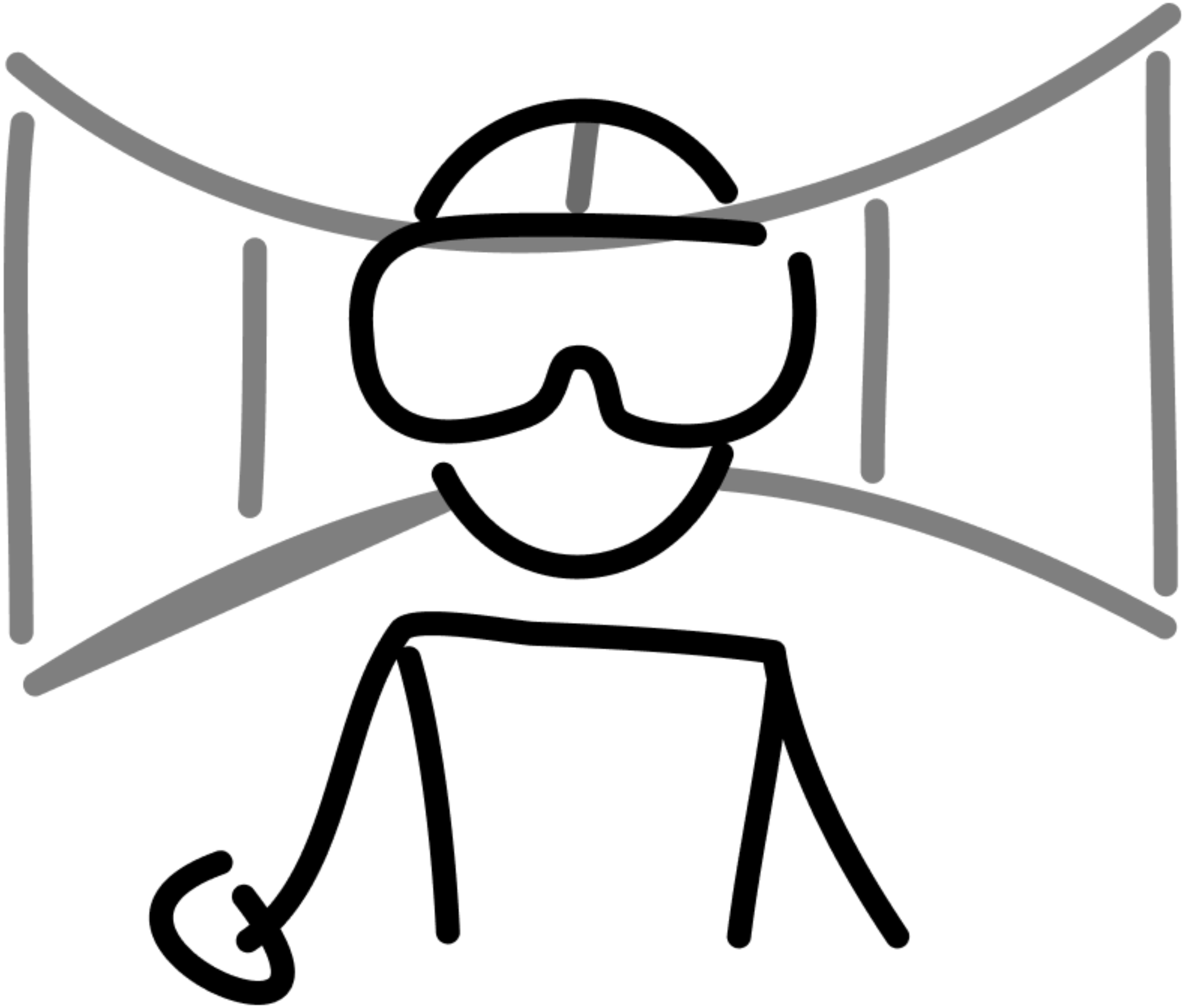

Bandwagon
effect

BANDWAGON EFFECT
JUST DO IT.
... EVERYONE ELSE DID!
TDL

*“Puede prevenir hasta un* **15% de Bugs** *en tus proyectos”.*

*"TypeScript **analyses my code constantly**. And can give great information on my code without me needing to do anything".*

*TypeScript in 50 Lessons*



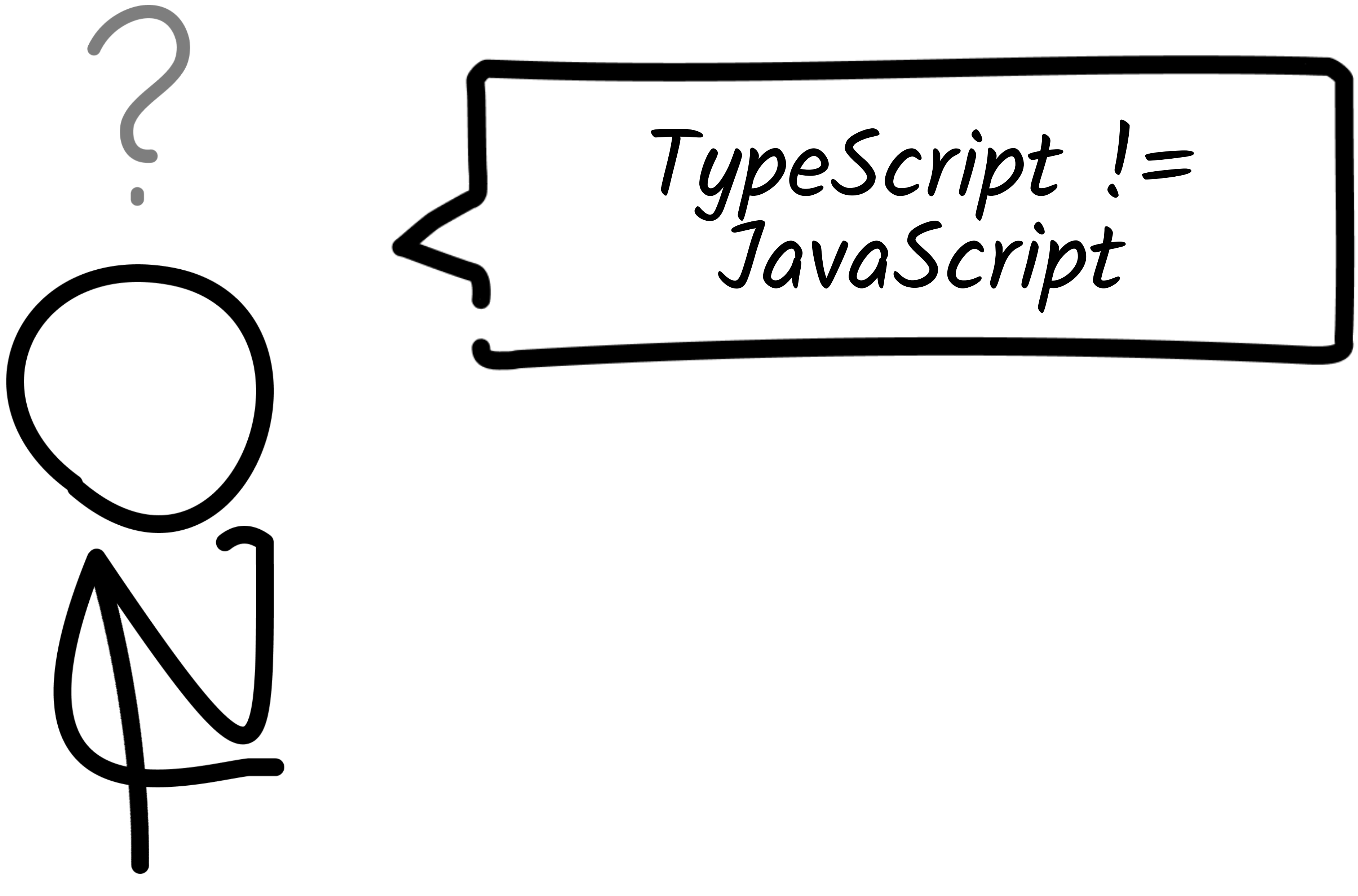
Visual Studio Code

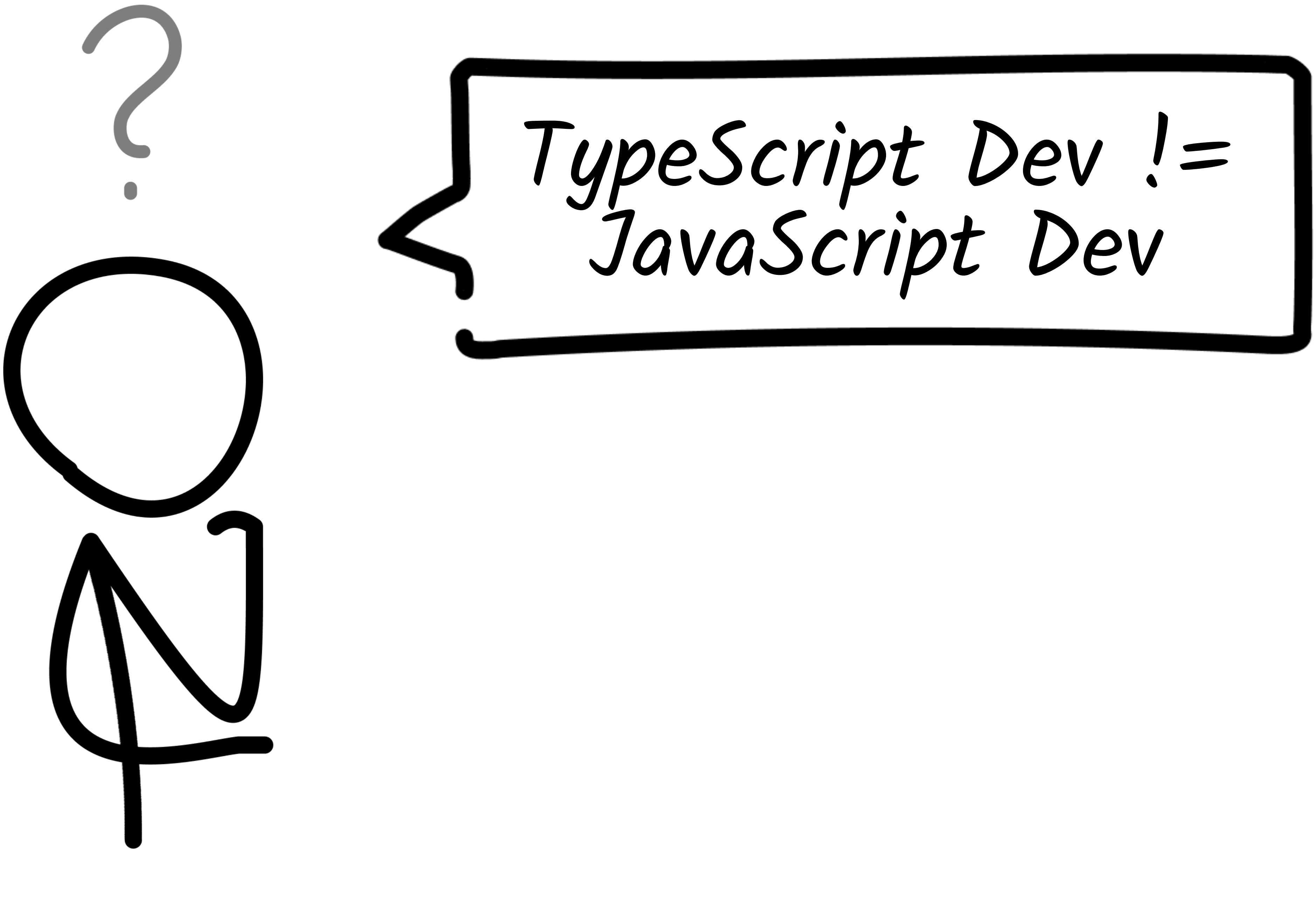
node

JS

# JS vs. TS

¿Qué debo tener en cuenta?

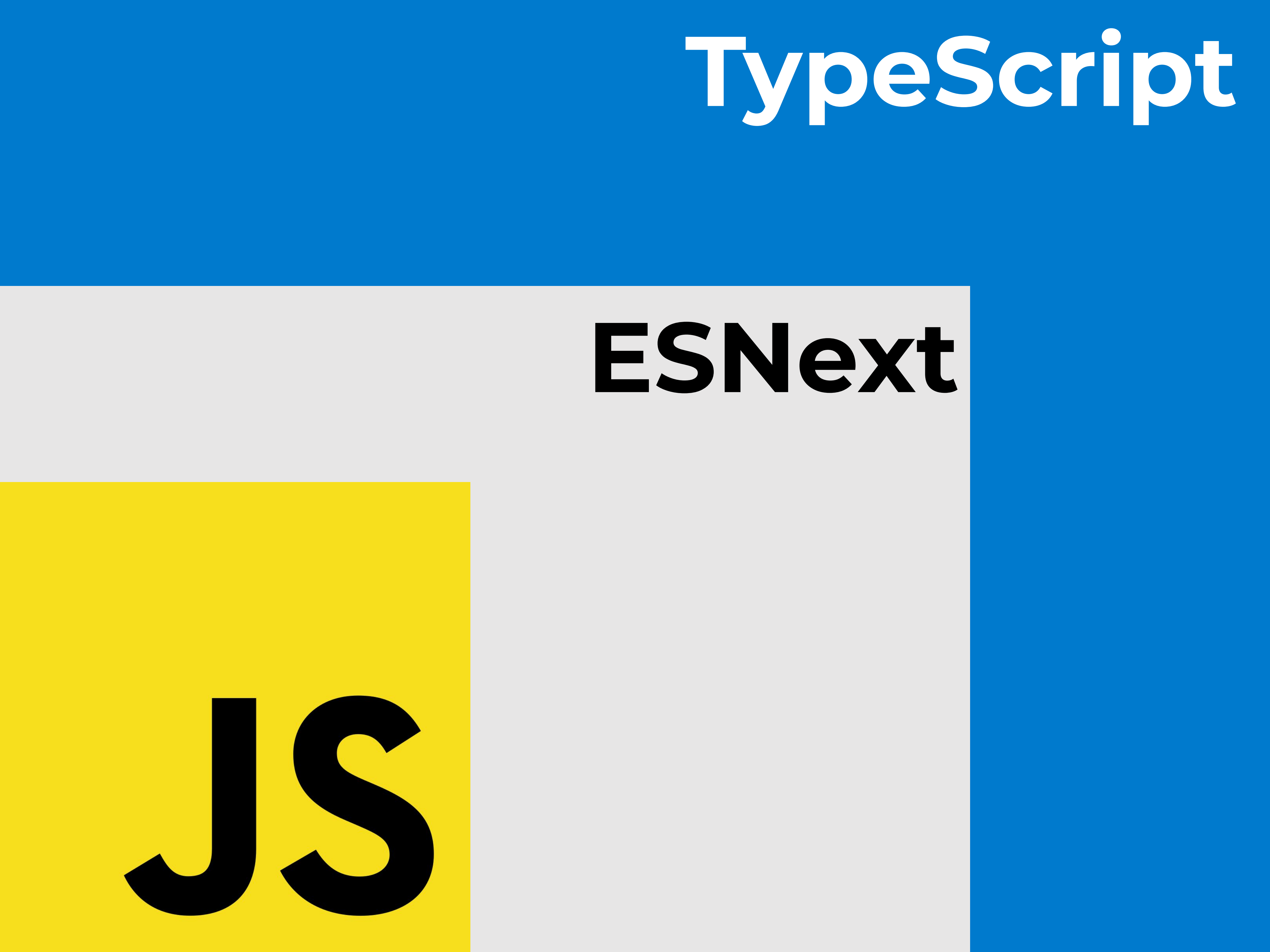

?
TypeScript !=
JavaScript


?
TypeScript Dev !=
JavaScript Dev

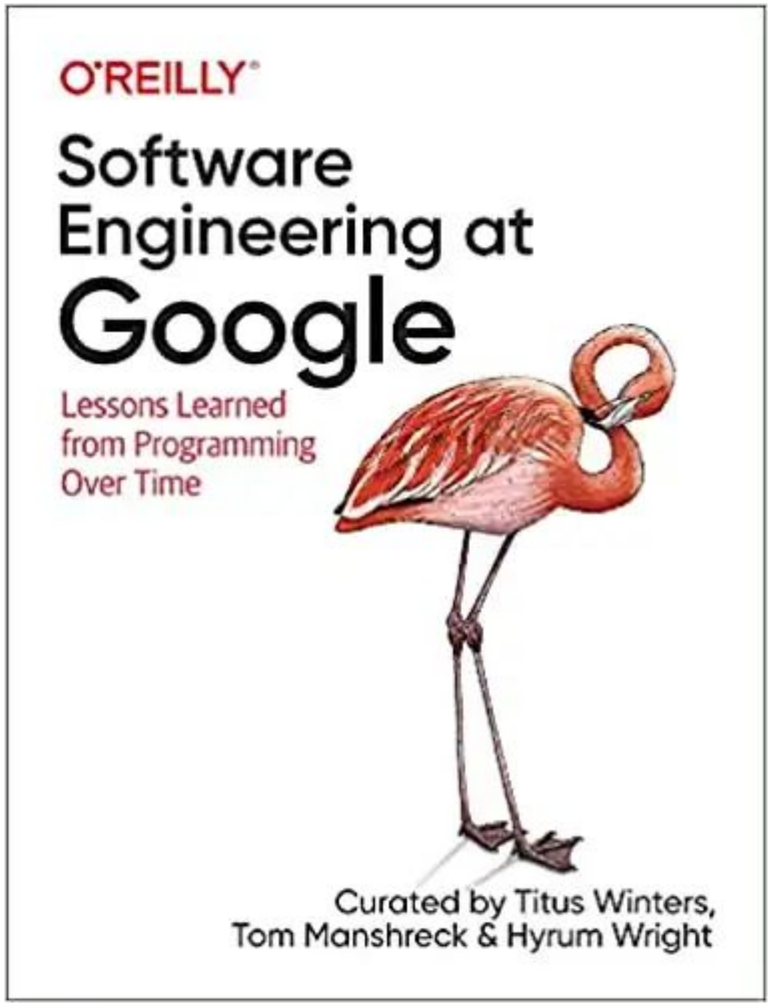
JS

*“Solo te das cuenta hasta que el código* ***está en ejecución****”.*

*"El objetivo es tener **menos errores** en producción y **feedback rápido** en desarrollo".*


TypeScript
ESNext
JS

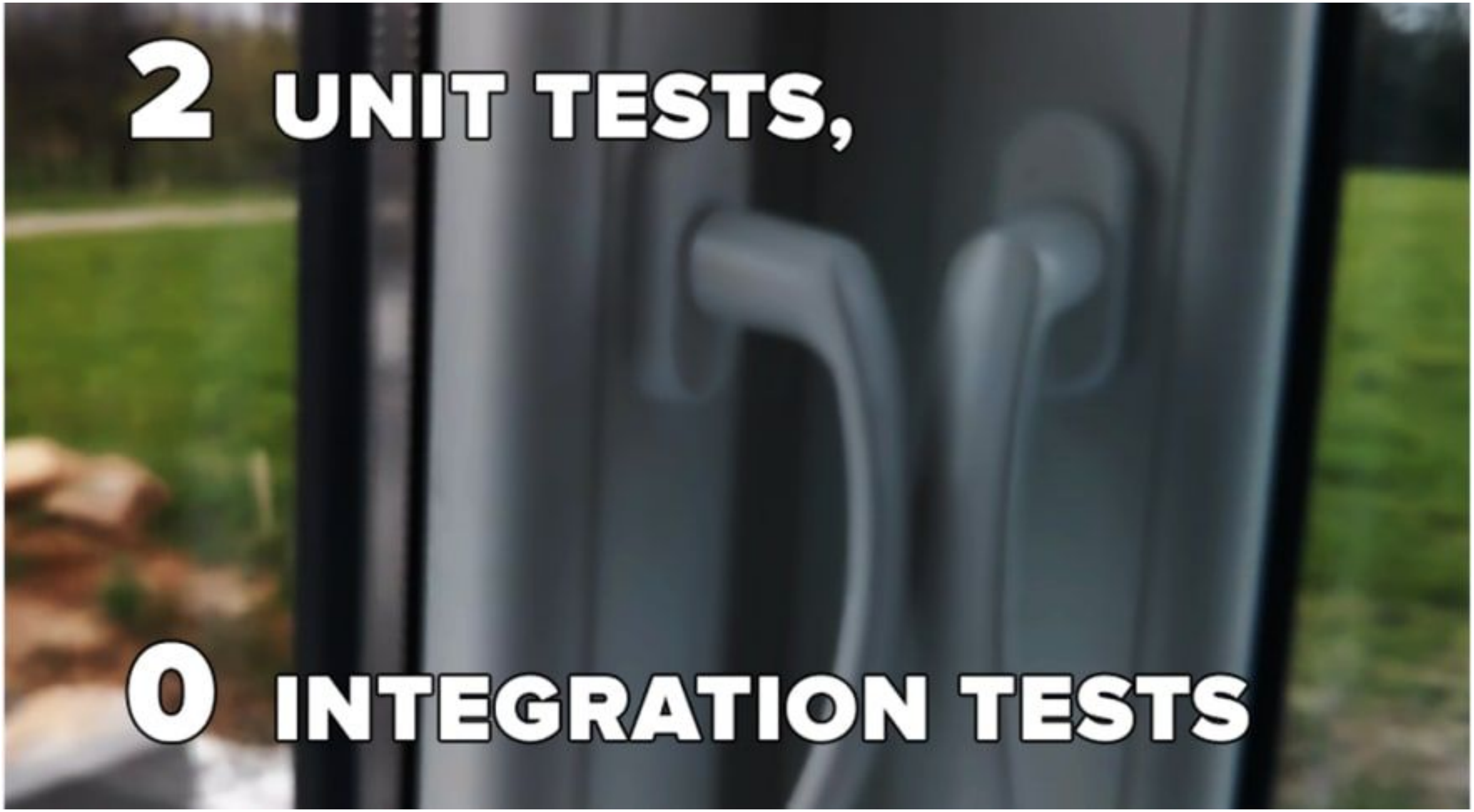
*"The earlier you find a mistake,* ***the easier it is to fix***"*.*


O'REILLY
Software Engineering at Google
Lessons Learned from Programming Over Time
Curated by Titus Winters, Tom Manshreck & Hyrum Wright

*“**Static analysis** runs in your editor. Finds typos, incorrect function calls, autocompletes code”.*

“**Unit tests** *take a few seconds to verify your code does what you think it does*”.

“**Integration tests** *take a few minutes to validate your system works. May catch fun edge cases”.*


2 UNIT TESTS,
0 INTEGRATION TESTS

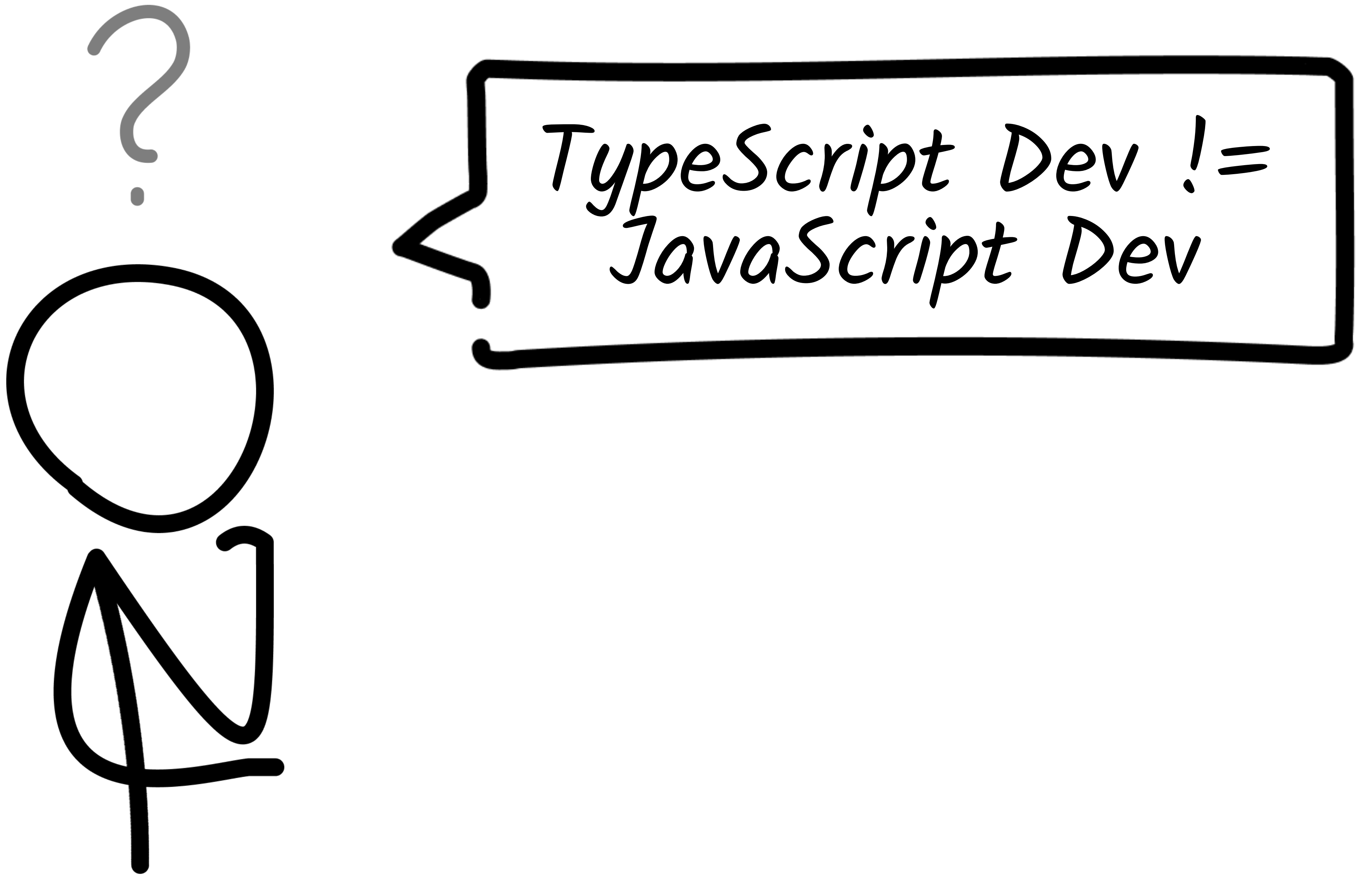
“***Code review*** *takes a few hours to validate you're following standard norms and practices of your team”.*

“**QA** *takes a few hours or days to ensure everything works together as expected”.*

*“**Static analysis** runs in your editor. Finds typos, incorrect function calls, autocompletes code”.*

*“Or, how I learned to* ***stop worrying*** *& trust the compiler”.*

slack


?
TypeScript Dev !=
JavaScript Dev

## Proyecto

Preparando nuestro entorno de trabajo

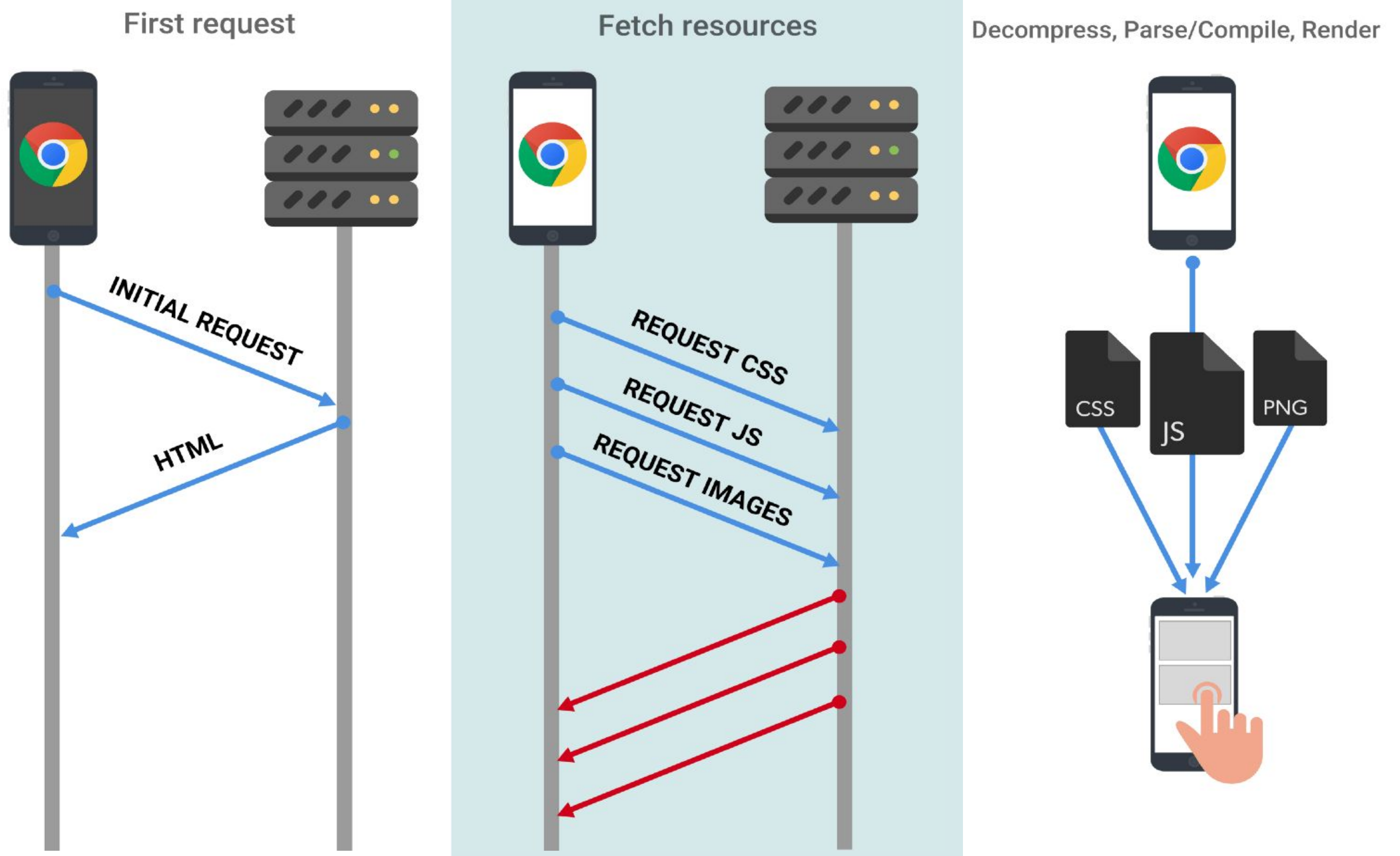
```
npm install typescript --save-dev
npx tsc --version
```

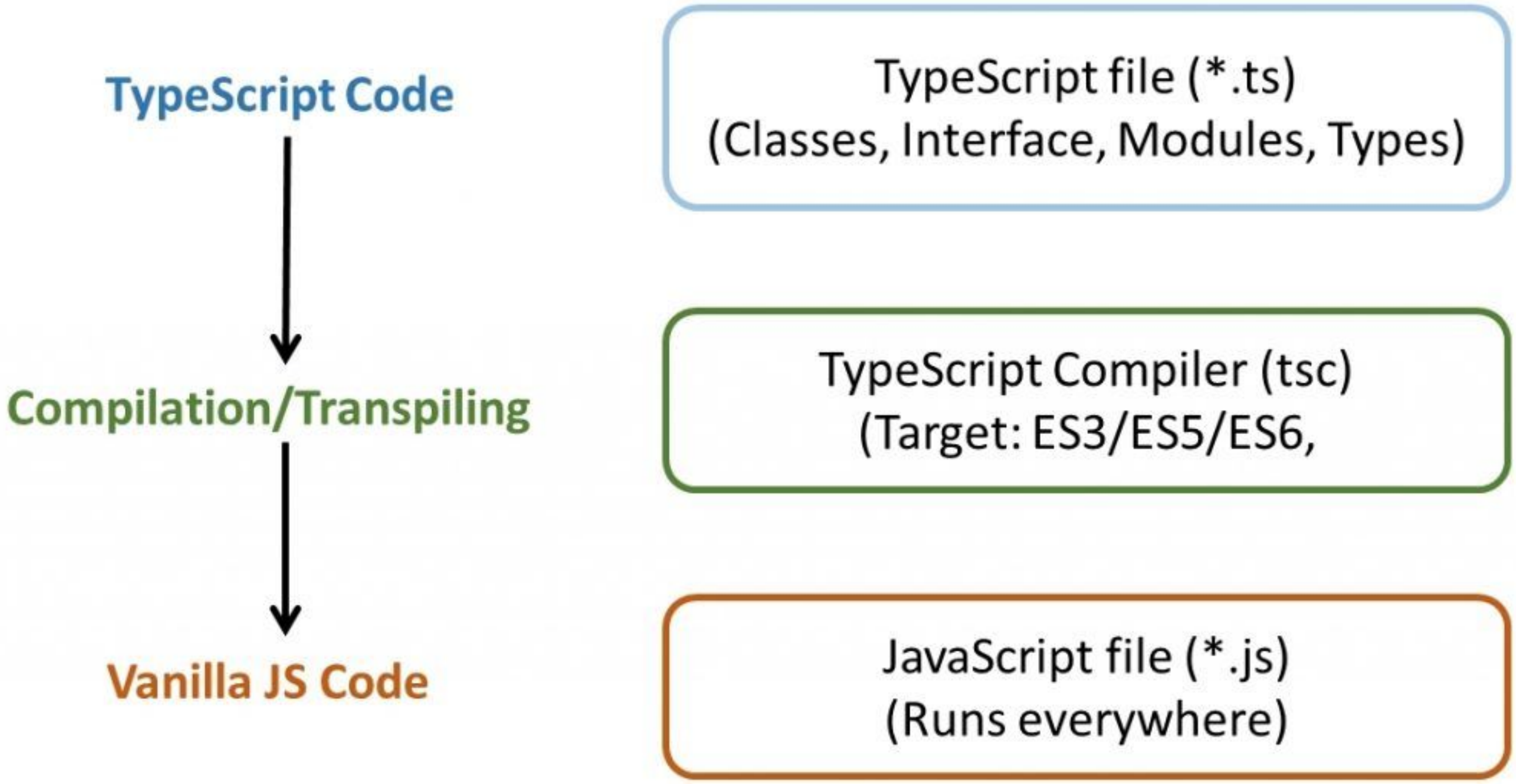
# Atrapando errores

Análisis estático de código

# El compilador

Transpila TS y genera JS


First request
INITIAL REQUEST
HTML
Fetch resources
REQUEST CSS
REQUEST JS
REQUEST IMAGES
Decompress, Parse/Compile, Render
CSS
JS
PNG

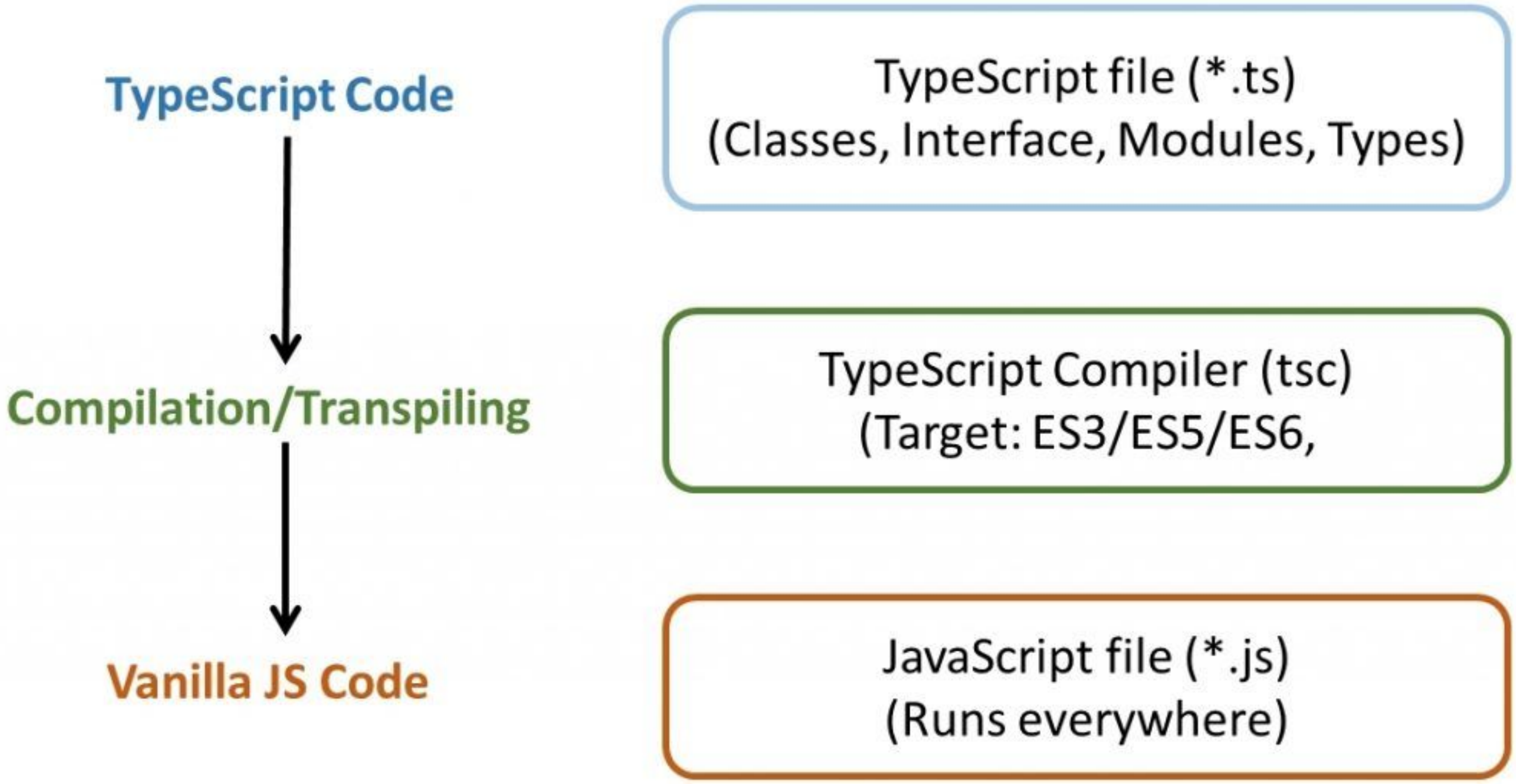

TypeScript Code
Compilation/Transpiling
Vanilla JS Code
TypeScript file (*.ts)
(Classes, Interface, Modules, Types)
TypeScript Compiler (tsc)
(Target: ES3/ES5/ES6,
JavaScript file (*.js)
(Runs everywhere)

```
npx tsc src/hello.ts
node src/hello.js
# Node y browser just run JS
npx tsc src/hello.ts --outDir dist
npx tsc src/cart.ts --outDir dist
# By default target is ES3
npx tsc src/cart.ts --outDir dist --target es6
```

Deno

## TSConfig.json

Ahorra trabajo


TypeScript Code
Compilation/Transpiling
Vanilla JS Code
TypeScript file (*.ts)
(Classes, Interface, Modules, Types)
TypeScript Compiler (tsc)
(Target: ES3/ES5/ES6,
JavaScript file (*.js)
(Runs everywhere)

```
npx tsc --init
npx tsc
npx tsc --watch
```

Jest

IONIC

nest

SVELTE

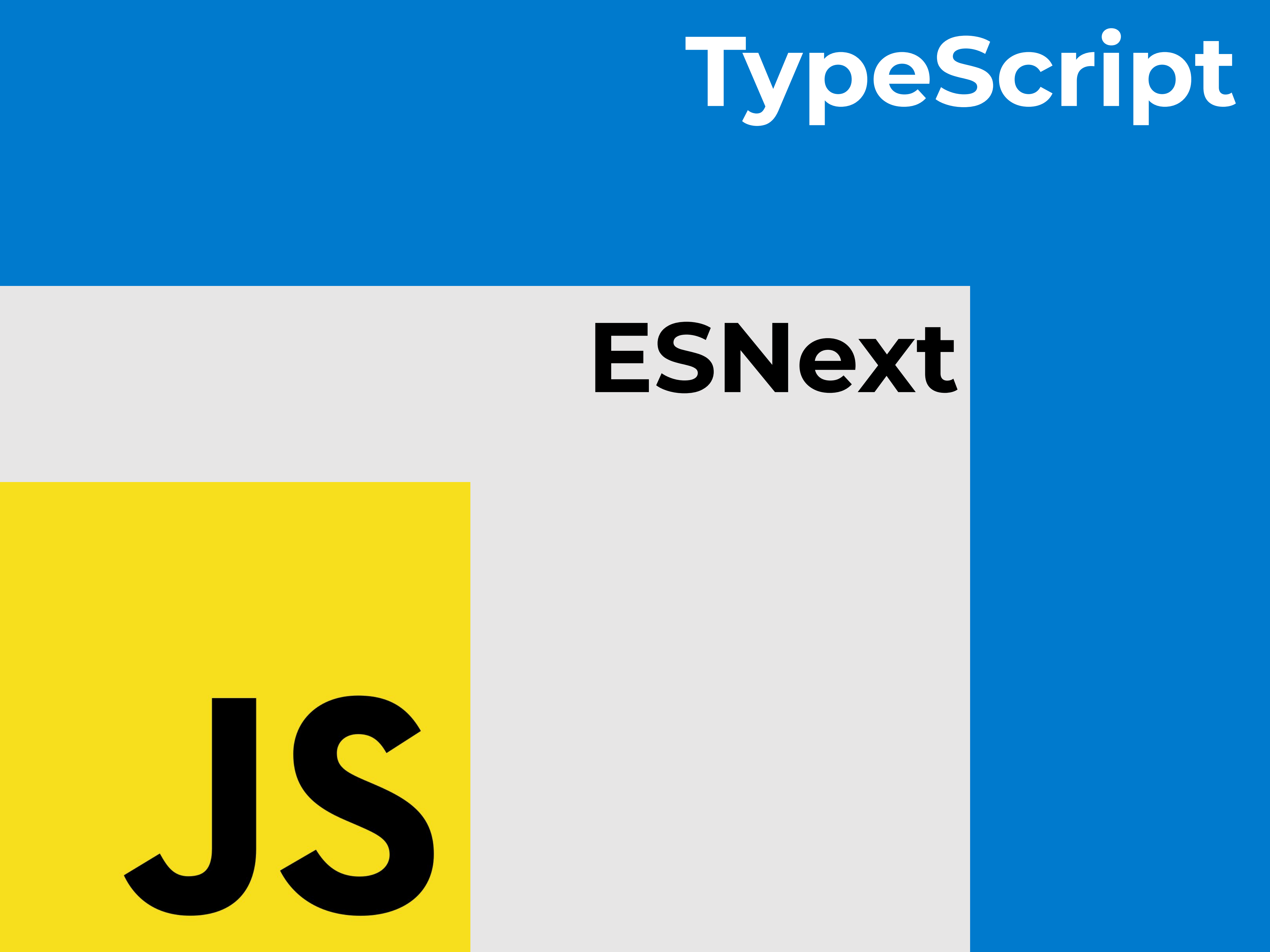
Vue.js

TYPEORM

GitHub

# El tipado

Ayuda a TypeScript


TypeScript
ESNext
JS

```javascript
let example = null; // null
example = 'string'; // string
example = 3.14; // number
example = true; // boolean
example = undefined; // undefined
example = []; // array
example = Symbol("abc") // Symbol

example = { // objs
  name: 'Nicolas',
  lastName: 'Molina'
}

example = function (a) { // functions
  return a;
}
```

```javascript
class Rectangulo {
  constructor(alto, ancho) {
    this.alto = alto;
    this.ancho = ancho;
  }
}

const p = new Rectangle();
```

```javascript
class Punto {
  constructor ( x , y ){
    this.x = x;
    this.y = y;
  }

  static distancia ( a , b) {
    const dx = a.x - b.x;
    const dy = a.y - b.y;

    return Math.sqrt ( dx * dx + dy * dy );
  }
}
```

```
class Punto {
  constructor ( x , y ){
    this.x = x;
    this.y = y;
  }

  static distancia ( a , b) {
    const dx = a.x - b.x;
    const dy = a.y - b.y;

    return Math.sqrt ( dx * dx + dy * dy );
  }
}
```

```javascript
const productPrice = 12;
```

```typescript
const productPrice: number = 12;
```

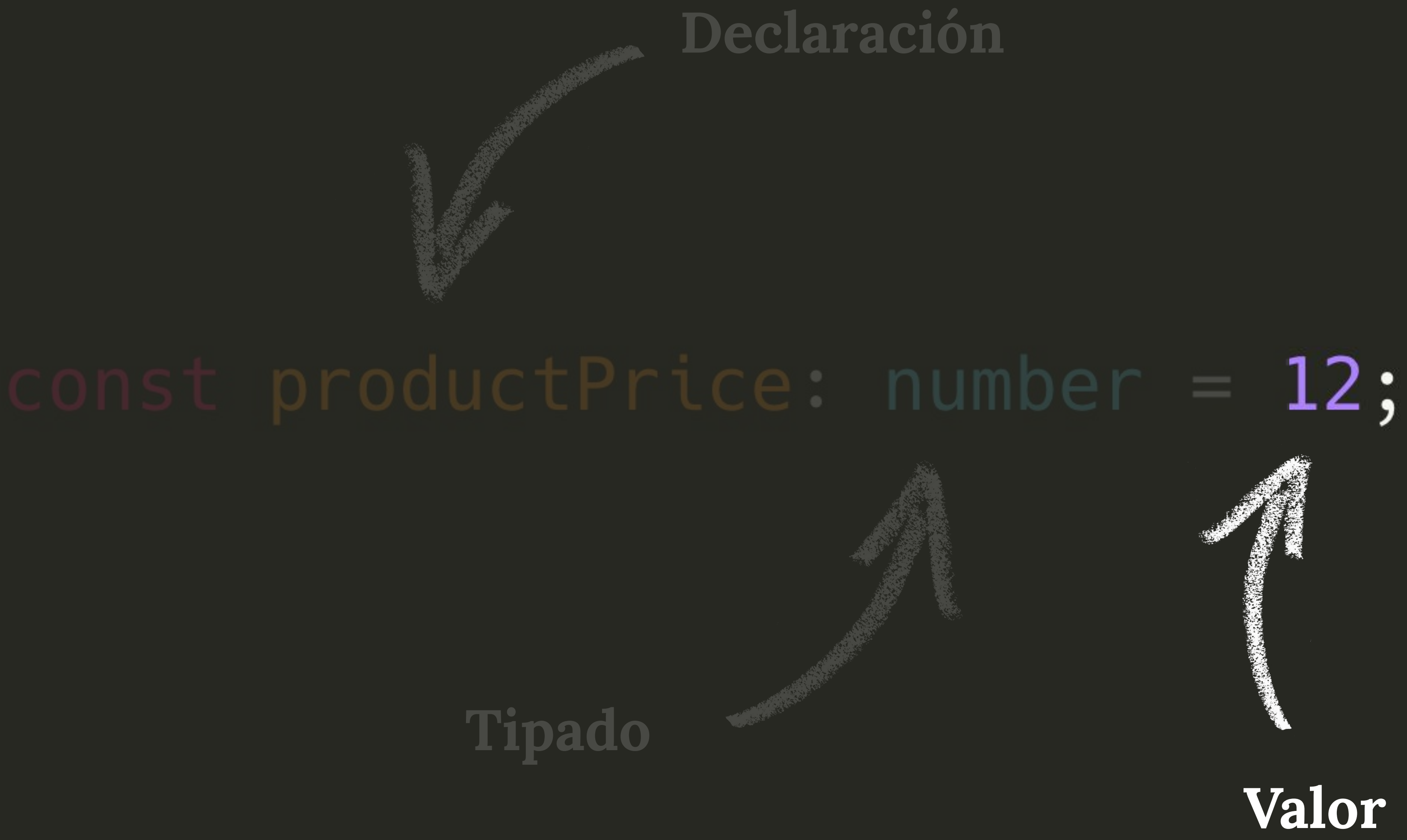
**Declaración**

```
const productPrice: number = 12;
```

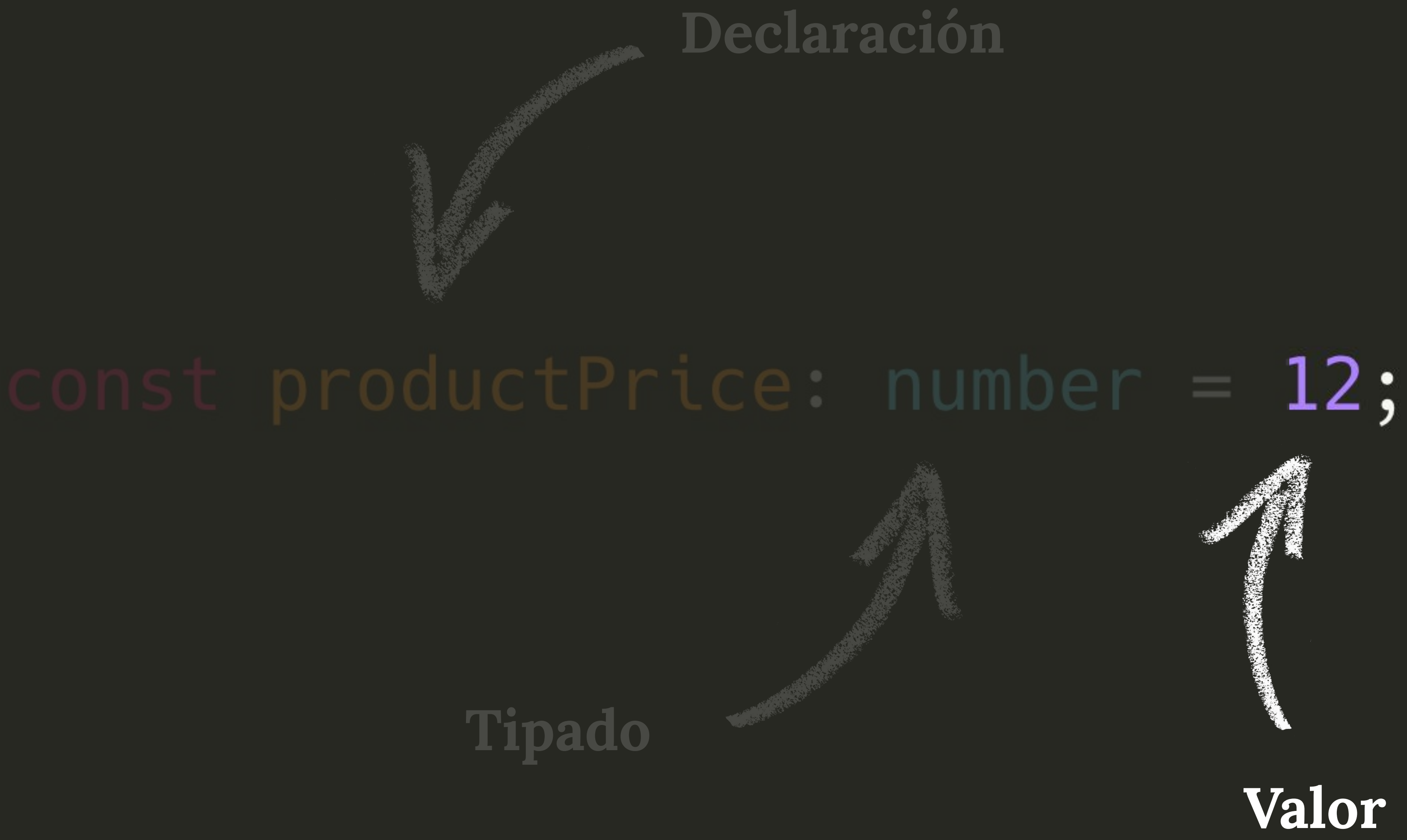
Declaración

```
const productPrice: number = 12;
```

Tipado



```typescript
const productPrice: number = 12;
```



```
const productPrice: number = 12;
```

```typescript
const productPrice: number = 12;
```

**Type Annotation**

# Tipos inferidos

Dejar que TypeScript nos ayude

```
const productPrice: number = 12;
```

TS

## Type: number

Trabajando con números

# Type: boolean

Trabajando con true y false

## Type: string

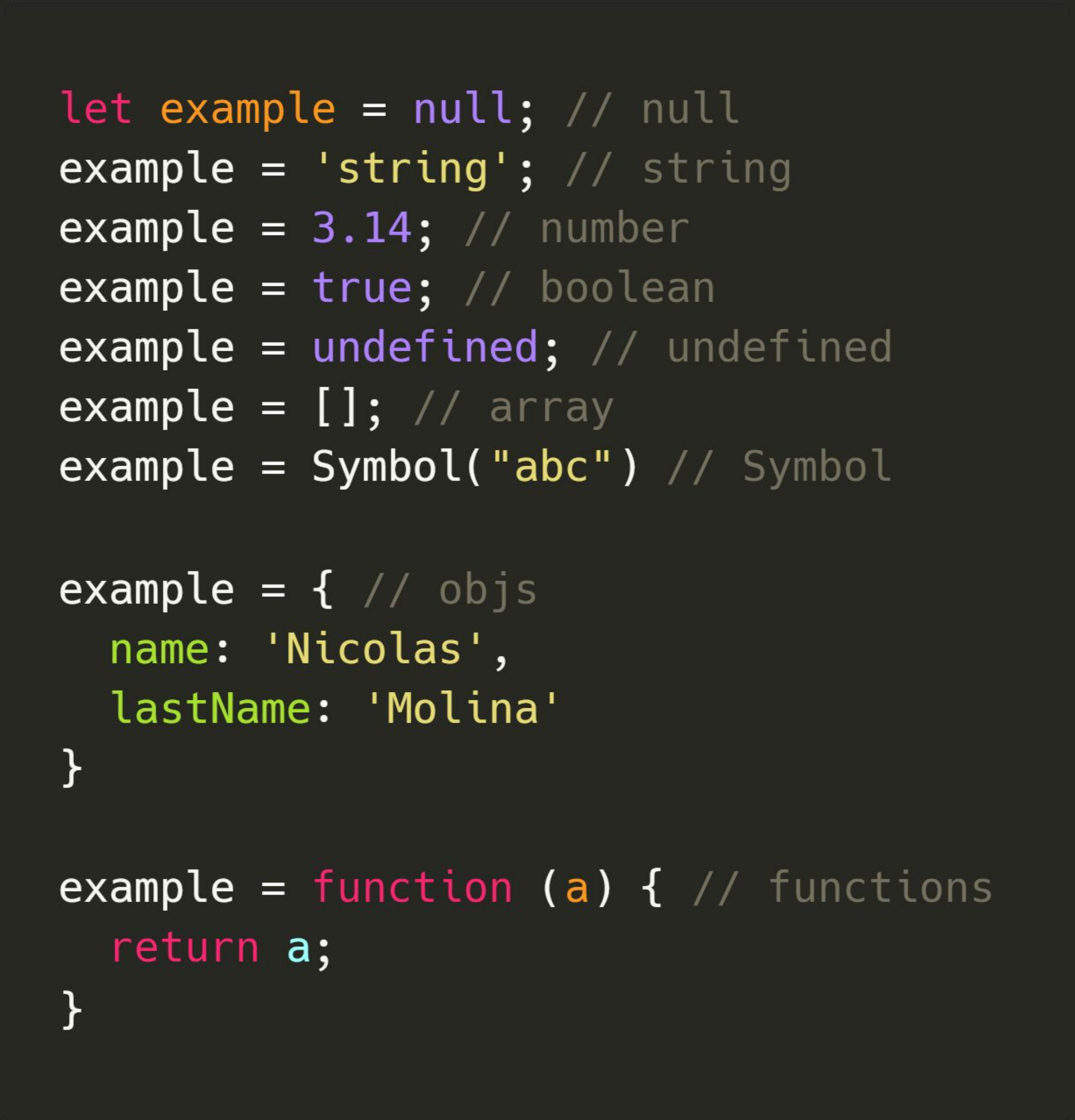
Trabajando con texto

## Type: arrays[]

Trabajando listas

## ¿Any está OK?

### Cómo nos podemos ayudar del Any



```javascript
let example = null; // null
example = 'string'; // string
example = 3.14; // number
example = true; // boolean
example = undefined; // undefined
example = []; // array
example = Symbol("abc") // Symbol

example = { // objs
  name: 'Nicolas',
  lastName: 'Molina'
}

example = function (a) { // functions
  return a;
}
```

## Union Types

Flexibilidad en TS

# Alias y tipos literales

Creando nuestros propios tipos

# Null & Undefined

Recuerda que son diferentes

# Funciones

Aprende sobre los tipos de funciones

# Void

## Retorno de funciones

## Objetos

En funciones

# Objetos como tipos

Crea tipos propios más complejos

## Módulos

Trabaja con export e import

TS

# Usando libs

Cómo usar librerías externas