

# Curso Angularn Tema 01b

---

---

Introducción Typescript

Instalación y ejecución del compilador

Tipos básicos

Funciones

Clases e interfaces

Escritura de código reutilizable con genéricos

Angular y Typescript

---

# Introducción a typescript

---

---

Tipos dinámicos contra tipos estáticos

Introducción a Typescript

Beneficios de aprender y utilizar Typescript

# Diferencia entre tipos dinámicos y estáticos

---

Dinámico	Estático
Tipo determinado en la declaración	Tipo determinado en la declaración
Puede cambiar a lo largo de todo el programa libre de errores	No puede cambiar en tiempo de ejecución sin causar un error

# Introducción a Typescript

---

JavaScript fuertemente tipado

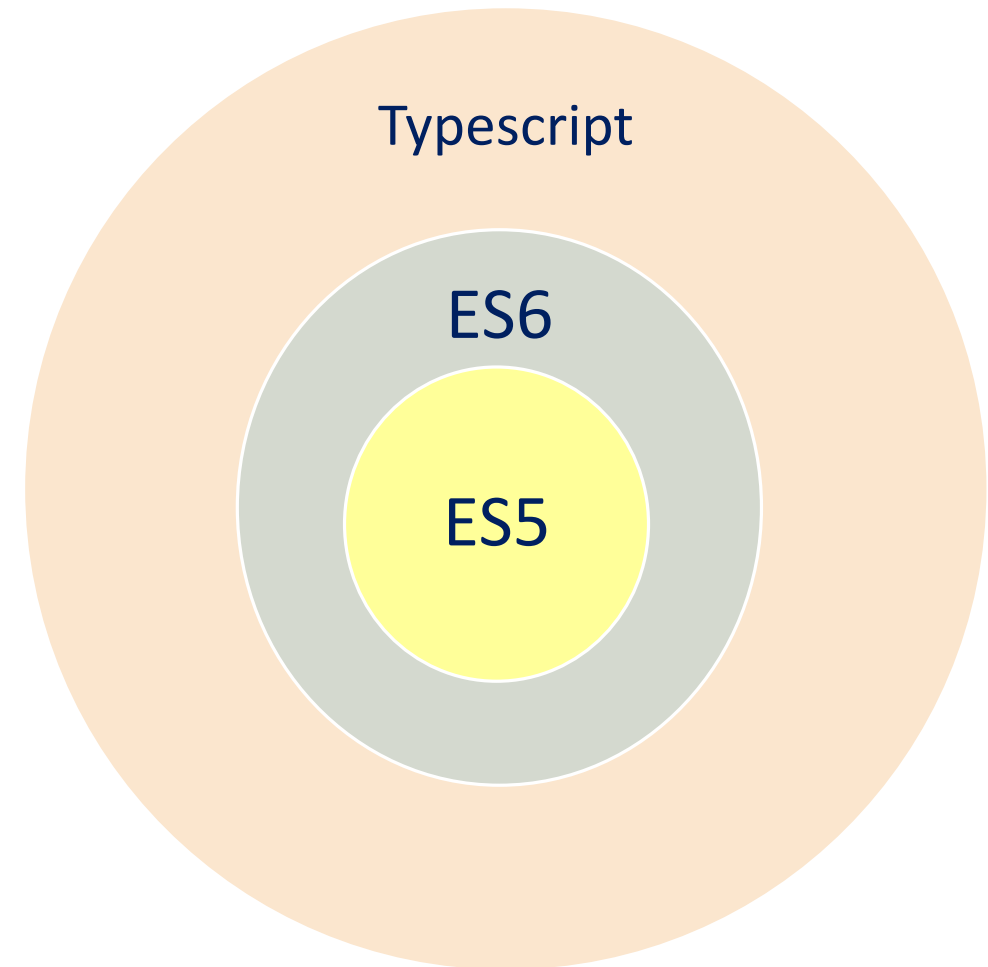
Compilador Javascript OpenSource

Compila a Javascript

Desarrollado por Microsoft

Estilo de programación más declarativo

Totalmente soportado por Angular 2/4/5



# ¿Qué ofrece TypeScript

---

Validación de tipos estático

Características ES6

Objetos basados en clase

Modularidad

# Beneficios

---

## Editores



Visual Studio 2017



Sublime  
Text



Emacs



Visual Studio  
Code



Atom



WebStorm



Visual Studio 2015



Eclipse



Vim



---

## Amigos de TypeScript



# Instalación y Ejecución del compilador TypeScript

---

---

Instalación de TypeScript

Compilación de un fichero de ejemplo TypeScript

# Descargar e Instalar Node

## Descarga de node



Node.js® es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome. Node.js usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes de Node.js, npm, es el ecosistema mas grande de librerías de código abierto en el mundo.

Important June 2018 security upgrades now available

### Descargar para Windows (x64)

**8.11.3 LTS**

Recomendado para la mayoría

**10.5.0 Actual**

Últimas características

[Otras Descargas](#) | [Cambios](#) | [Documentación del API](#)

[Otras Descargas](#) | [Cambios](#) | [Documentación del API](#)

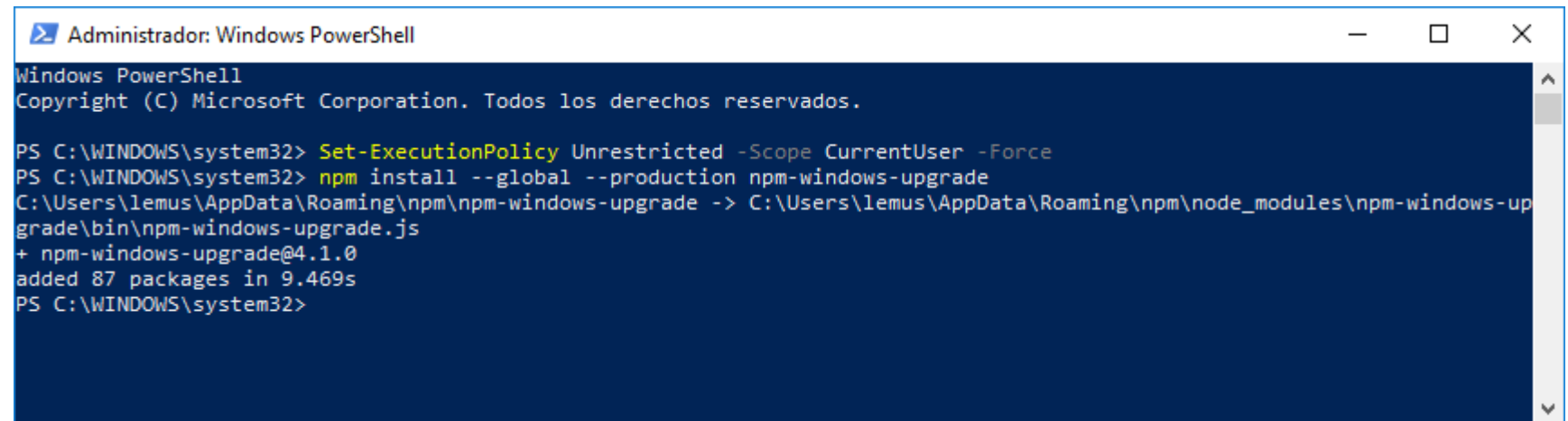
# Actualizar node

---

Ejecutar powerShell como administrador:

Set-ExecutionPolicy Unrestricted -Scope CurrentUser -Force

npm install --global --production npm-windows-upgrade



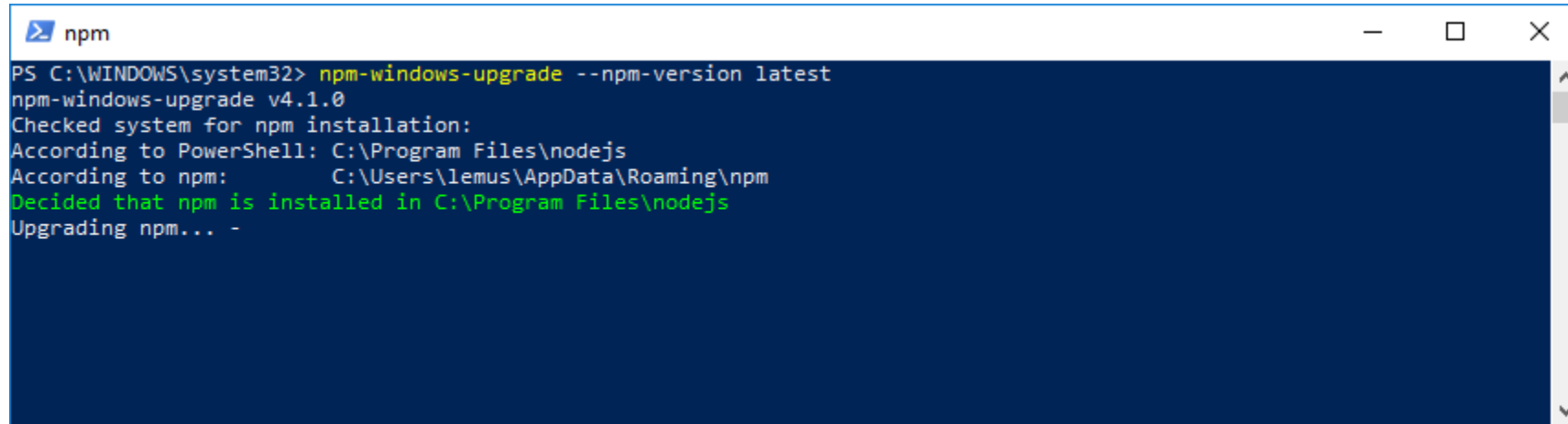
```
Administrador: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

PS C:\WINDOWS\system32> Set-ExecutionPolicy Unrestricted -Scope CurrentUser -Force
PS C:\WINDOWS\system32> npm install --global --production npm-windows-upgrade
C:\Users\lemus\AppData\Roaming\npm\npm-windows-upgrade -> C:\Users\lemus\AppData\Roaming\npm\node_modules\npm-windows-upgrade\bin\npm-windows-upgrade.js
+ npm-windows-upgrade@4.1.0
added 87 packages in 9.469s
PS C:\WINDOWS\system32>
```

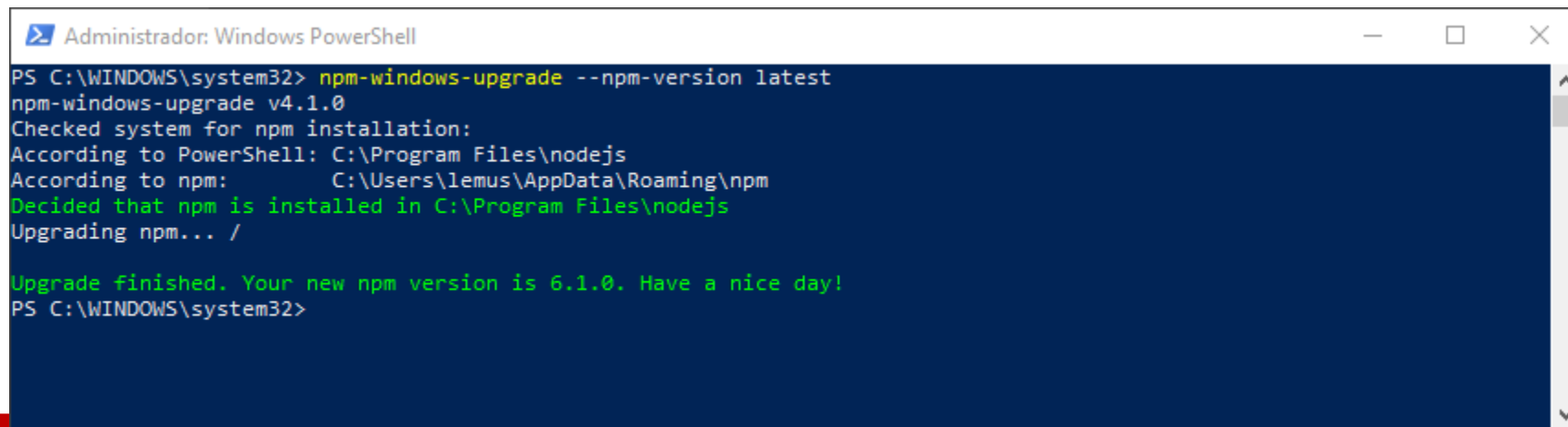
Ejecutar:

- ✓ npm-windows-upgrade
- ✓ npm-windows-upgrade --npm-version latest

# Actualizar node




```
PS C:\WINDOWS\system32> npm-windows-upgrade --npm-version latest
npm-windows-upgrade v4.1.0
Checked system for npm installation:
According to PowerShell: C:\Program Files\nodejs
According to npm:        C:\Users\lemus\AppData\Roaming\npm
Decided that npm is installed in C:\Program Files\nodejs
Upgrading npm... -
```



```
PS C:\WINDOWS\system32> npm-windows-upgrade --npm-version latest
npm-windows-upgrade v4.1.0
Checked system for npm installation:
According to PowerShell: C:\Program Files\nodejs
According to npm:        C:\Users\lemus\AppData\Roaming\npm
Decided that npm is installed in C:\Program Files\nodejs
Upgrading npm... /

Upgrade finished. Your new npm version is 6.1.0. Have a nice day!
PS C:\WINDOWS\system32>
```

## Visual Studio Code

 **Visual Studio Code** [Docs](#) [Updates](#) [Blog](#) [Community](#) [Extensions](#) [FAQ](#) [Search Docs](#) [Download](#)

Version 1.24 is now available! Read about the new features and fixes from May.

## Download Visual Studio Code

Free and open source. Integrated Git, debugging and extensions.



↓ **Windows**

Windows 7, 8, 10

.zip | 32 bit versions



↓ **.deb**

Debian, Ubuntu

↓ **.rpm**

Red Hat, Fedora, SUSE

.tar.gz | 32 bit versions



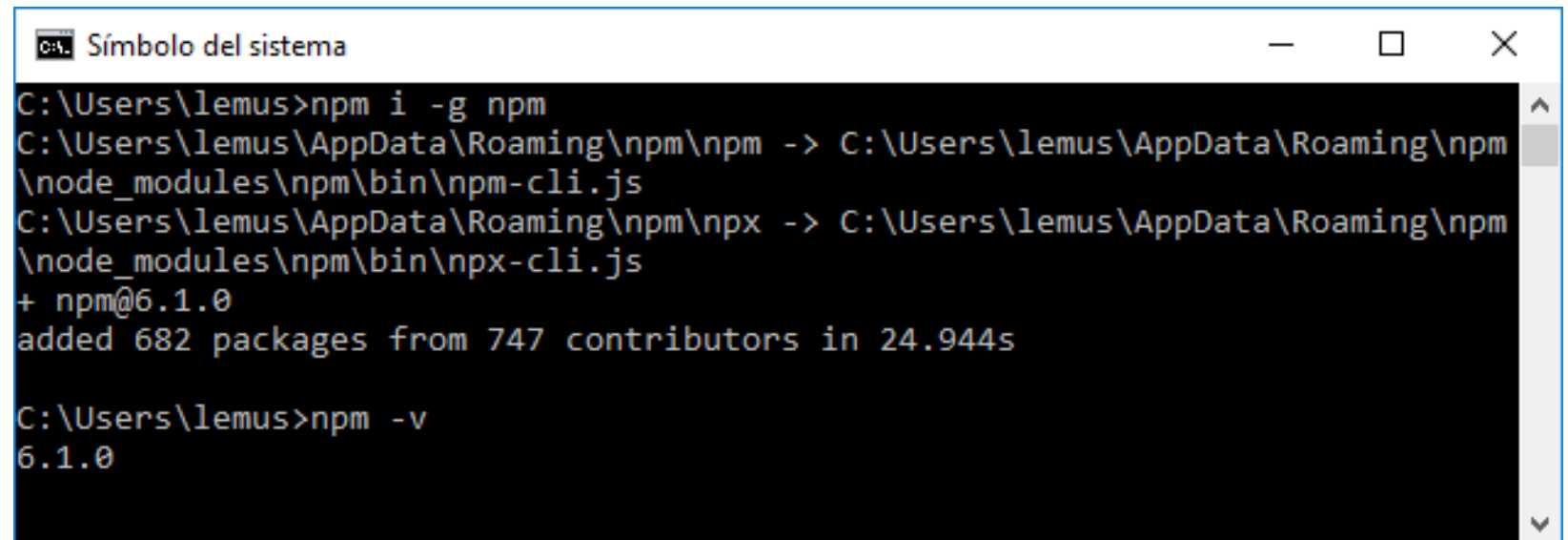
↓ **Mac**

macOS 10.9+

# Actualizar npm

---

En una terminal ejecutar `npm i -g npm`



```
C:\Users\lemus>npm i -g npm
C:\Users\lemus\AppData\Roaming\npm\npm -> C:\Users\lemus\AppData\Roaming\npm
\node_modules\npm\bin\npm-cli.js
C:\Users\lemus\AppData\Roaming\npm\npx -> C:\Users\lemus\AppData\Roaming\npm
\node_modules\npm\bin\npx-cli.js
+ npm@6.1.0
added 682 packages from 747 contributors in 24.944s

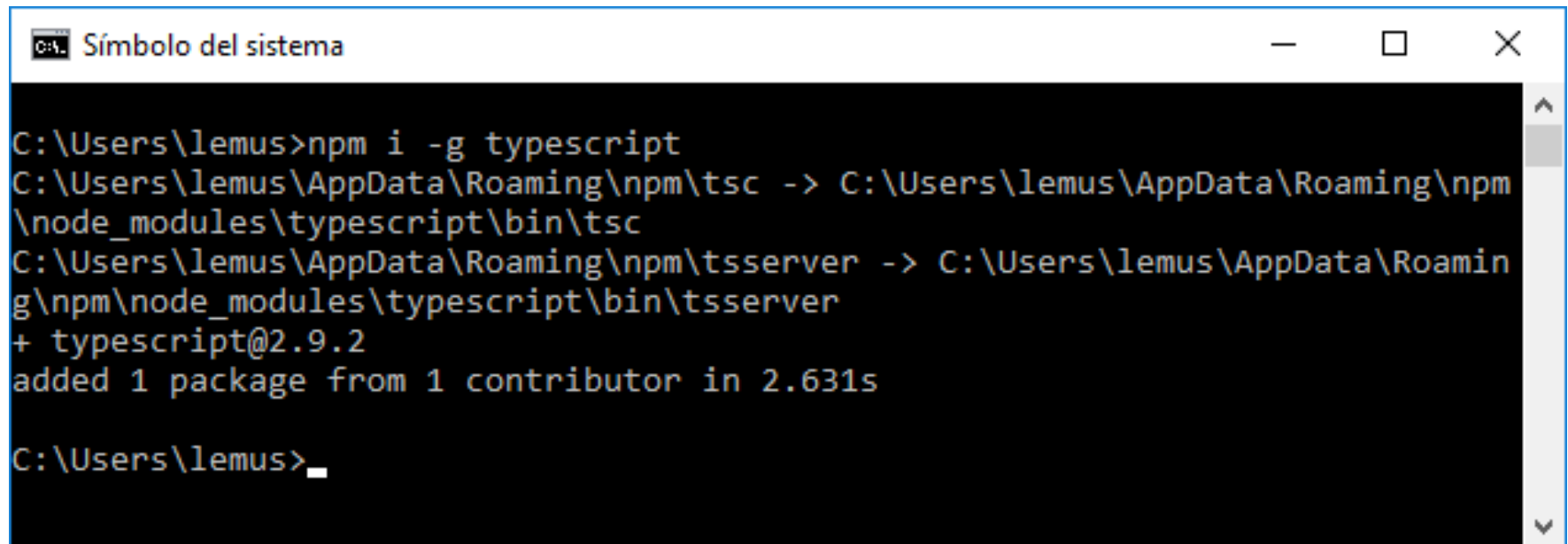
C:\Users\lemus>npm -v
6.1.0
```



# Instalar typescript

---

`npm i -g typescript`



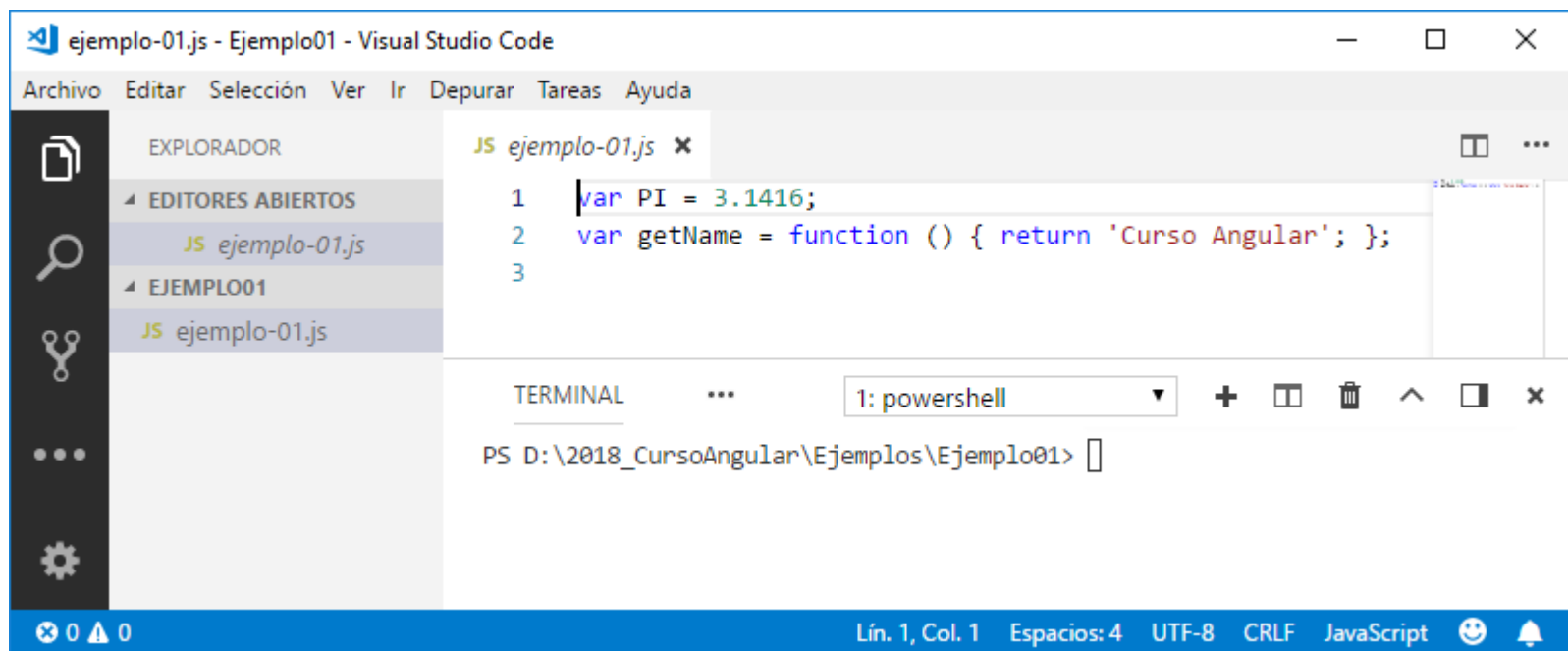
```
Símbolo del sistema

C:\Users\lemus>npm i -g typescript
C:\Users\lemus\AppData\Roaming\npm\tsc -> C:\Users\lemus\AppData\Roaming\npm\node_modules\typescript\bin\tsc
C:\Users\lemus\AppData\Roaming\npm\tsserver -> C:\Users\lemus\AppData\Roaming\npm\node_modules\typescript\bin\tsserver
+ typescript@2.9.2
added 1 package from 1 contributor in 2.631s

C:\Users\lemus>
```

# Ejemplo 1

Crear el fichero ejemplo-01.ts



The screenshot shows the Visual Studio Code interface with the file explorer on the left. The 'EXPLORADOR' pane shows a project named 'EJEMPLO01' containing a file 'ejemplo-01.ts'. The editor displays the TypeScript code for 'ejemplo-01.ts':

```
1 const PI: number = 3.1416;  
2 const getCourseName = () => 'Curso Angular 5';
```

The terminal at the bottom shows the command to compile the file:

```
PS D:\2018_CursoAngular\Ejemplos\Ejemplo01> tsc ejemplo-01.ts
```

A callout box with the text 'Compilar el fichero' points to the 'tsc' command in the terminal.

Compilar el fichero

The screenshot shows the Visual Studio Code interface after compilation. The file explorer on the left now shows a new file 'ejemplo-01.js' under the 'EJEMPLO01' project. The editor displays the generated JavaScript code for 'ejemplo-01.js':

```
1 var PI = 3.1416;  
2 var getCourseName = function () { return 'Curso Angular 5';  
3
```

The terminal at the bottom shows the command to compile the file:

```
PS D:\2018_CursoAngular\Ejemplos\Ejemplo01> tsc ejemplo-01.ts  
PS D:\2018_CursoAngular\Ejemplos\Ejemplo01>
```

A callout box with the text 'Código javascript generado' points to the JavaScript code in the editor.

Código javascript  
generado

Fichero generado

# Tipos básicos

---

---

Tipos básicos

Variables

Vectores y Matrices

# Tipos básicos

---

String

Boolean

Number

Enum

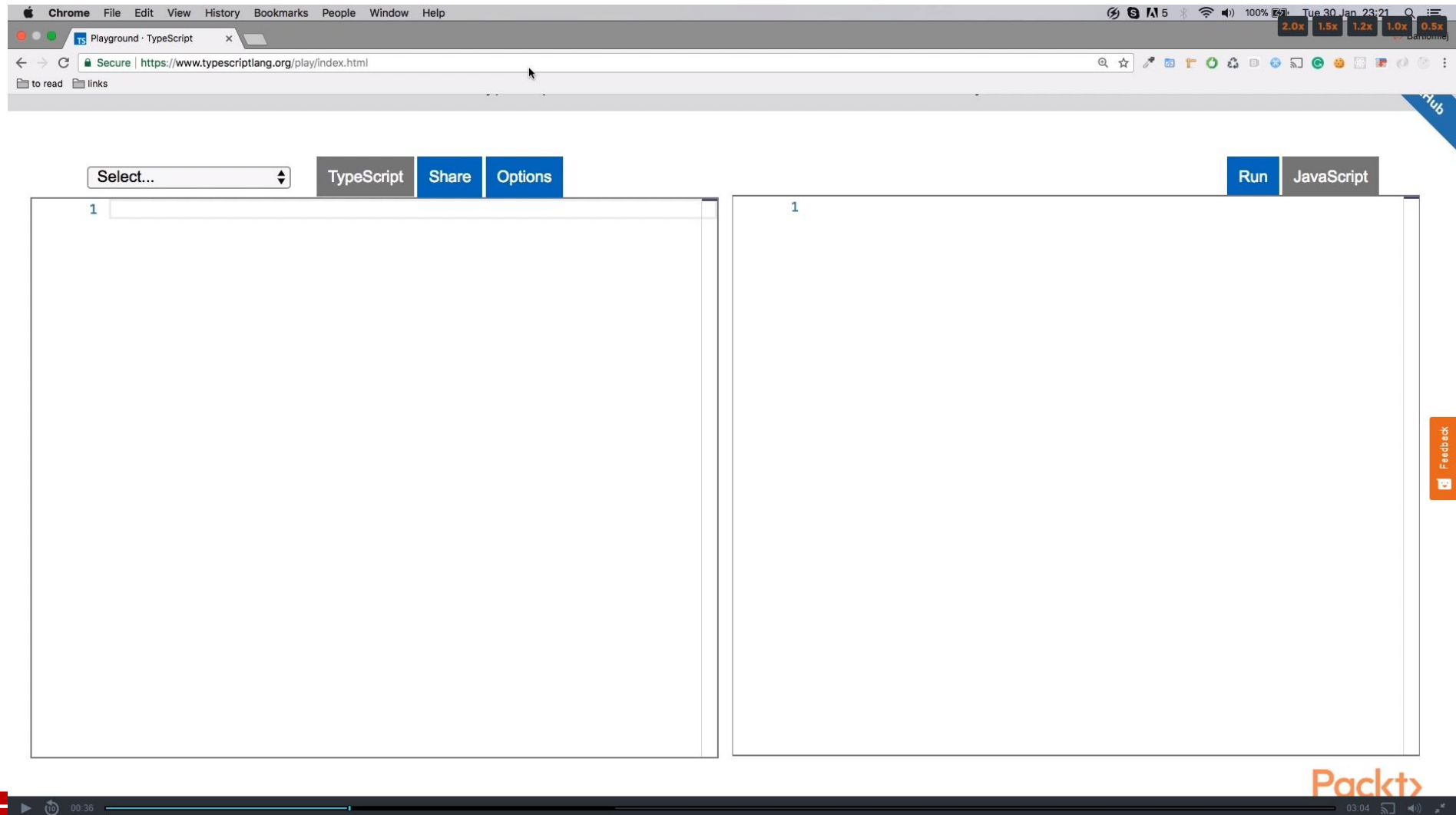
Any

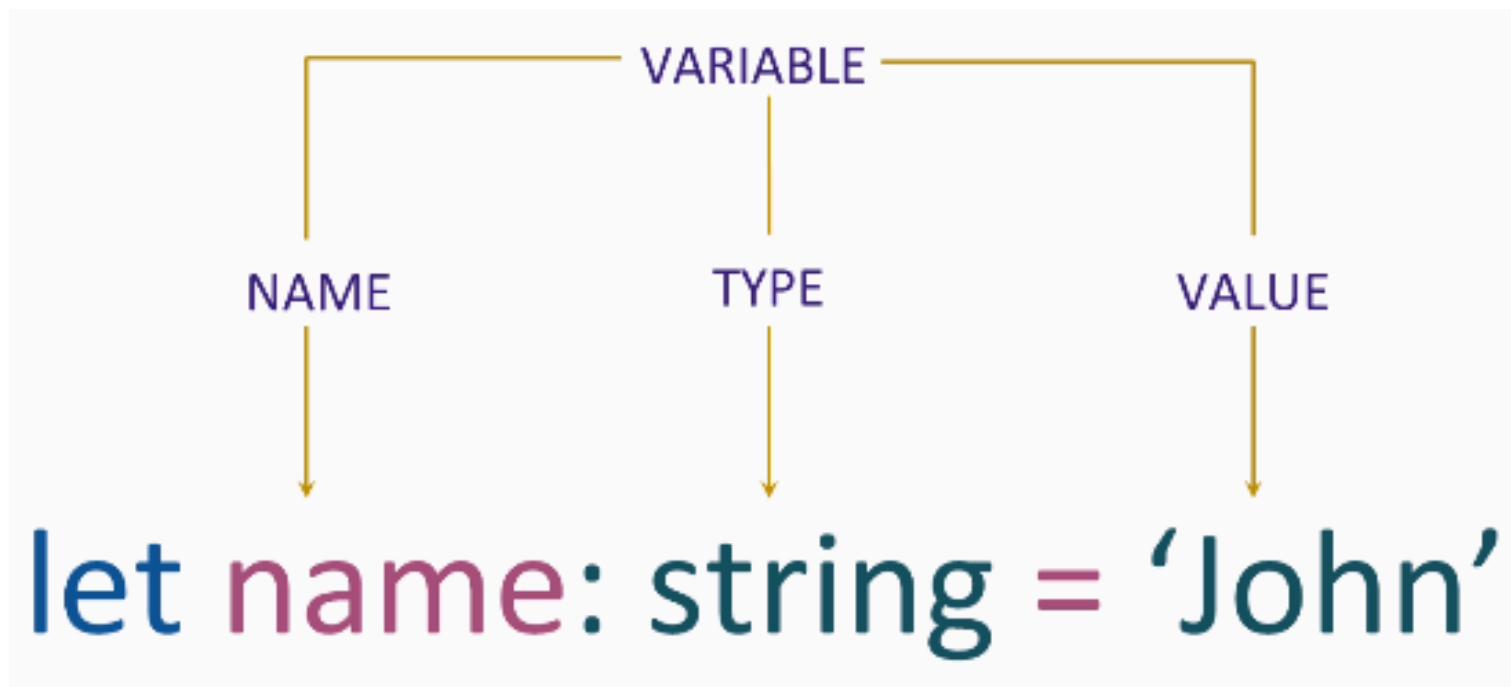
Void

Null

Undefined

# playground TypeScript







Playground · TypeScript

Secure | https://www.typescriptlang.org/play/index.html

Apps GFI Small Business Firew Instalando Moodle e Spreading Excellence Knowledge Alliances

Other bookmarks

TypeScriptQuick StartDocumentationDownloadConnectPlayground

TypeScript 2.9 is now available. Download our latest version today!

Using ClassesTypeScriptShareOptionsRunJavaScript

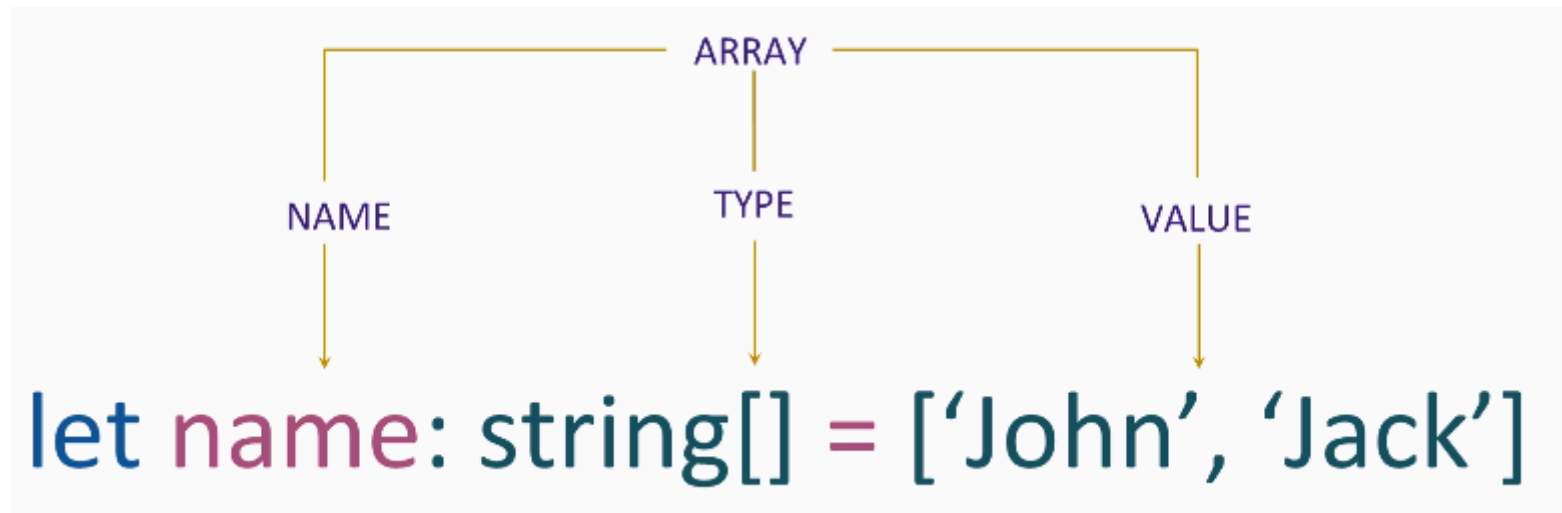
```
1 const PI: number = 3.1416;
2
3 let edad: number = 32;
4 let miNombre: string = 'Lenin';
5
6 let esProgramador: boolean = true;
7
8
9
```

```
1 var PI = 3.1416;
2 var edad = 32;
3 var miNombre = 'Lenin';
4 var esProgramador = true;
5
```

9781789342161-B....pdfShow all

# Vectores y Matrices

---



Playground · TypeScript

Secure | https://www.typescriptlang.org/play/index.html

Apps GFI Small Business Firew Instalando Moodle e Spreading Excellence Knowledge Alliances Other bookmarks

TypeScriptQuick StartDocumentationDownloadConnectPlayground

TypeScript 2.9 is now available. Download our latest version today!

Using ClassesTypeScriptShareOptionsRunJavaScript

```
1 const PI: number = 3.1416;
2
3 let edad: number = 32;
4 let miNombre: string = 'Lenin';
5
6 let esProgramador: boolean = true;
7
8 let jugadores: string[] = ['juan', 'pedro',
9
10 jugadores.push('Gerardo');
11
12
```

```
1 var PI = 3.1416;
2 var edad = 32;
3 var miNombre = 'Lenin';
4 var esProgramador = true;
5 var jugadores = ['juan', 'pedro', 'elena'];
6 jugadores.push('Gerardo');
7
```

9781789342161-B....pdfShow all

# Trabajando con funciones

---

---

Declaración de funciones y expresiones

Tipos de función

Parámetros opcionales

Parámetros por defecto

# Declaración de funciones y expresiones

---

Select... ▼

TypeScript

Share

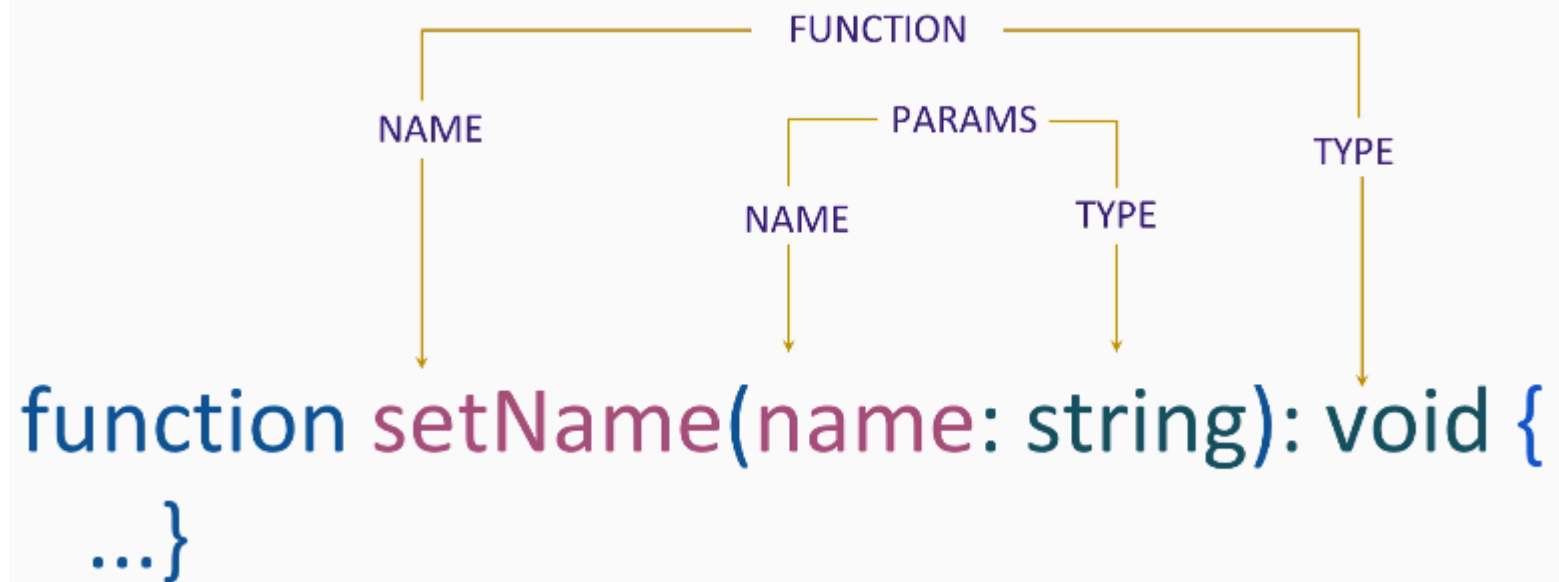
Options

Run

JavaScript

```
1 function getNombre() {  
2     return 'Guillermo';  
3 }  
4 const getApellido = function () {  
5     return 'Lemus';  
6 }
```

```
1 function getNombre() {  
2     return 'Guillermo';  
3 }  
4 var getApellido = function () {  
5     return 'Lemus';  
6 };  
7
```



Select... ▼

TypeScriptShareOptions

```
1
2 function getNombre() {
3     return 'Guillermo';
4 }
5 const getApellido = function () {
6     return 'Lemus';
7 }
8 let names: string[] = [];
9 function addNombre(nombre: string): string[] {
10     names.push(nombre);
11     return names;
12 }
13
14 console.log(addNombre('Laura'))
```

RunJavaScript

```
1 function getNombre() {
2     return 'Guillermo';
3 }
4 var getApellido = function () {
5     return 'Lemus';
6 };
7 var names = [];
8 function addNombre(nombre) {
9     names.push(nombre);
10    return names;
11 }
12 console.log(addNombre('Laura'));
13
```

ElementsConsoleSources

▶ top ▼ Filter

▼ Array(1) 1

0: "Laura"

length: 1

▶ \_\_proto\_\_: Array(0)

> |



# Parámetros opcionales

Select...

TypeScriptShareOptionsRunJavaScript

```
1
2 function getNombre() {
3     return 'Guillermo';
4 }
5 const getApellido = function () {
6     return 'Lemus';
7 }
8 let names: string[] = [];
9 function addNombre(nombre?: string): string[] {
10     names.push(nombre);
11     return names;
12 }
13
14 console.log(addNombre('Laura'))
```

```
1 function getNombre() {
2     return 'Guillermo';
3 }
4 var getApellido = function () {
5     return 'Lemus';
6 };
7 var names = [];
8 function addNombre(nombre) {
9     names.push(nombre);
10    return names;
11 }
12 console.log(addNombre('Laura'));
13
```

ElementsConsole

top

▼ Array(1) 

0: "Laura"

length: 1

\_\_proto\_\_: Array(0)

> |

# Parámetros por defecto en una función

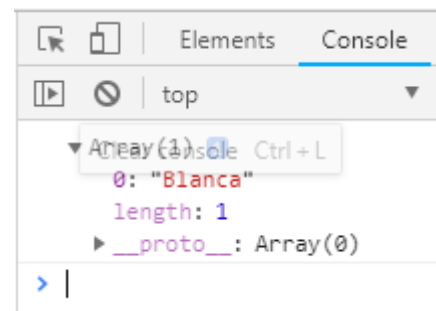
Select...

TypeScriptShareOptions

```
1 function getNombre() {
2     return 'Guillermo';
3 }
4 const getApellido = function () {
5     return 'Lemus';
6 }
7 let names: string[] = [];
8 function addNombre(
9     nombre: string = 'Blanca'
10 ): string[] {
11     names.push(nombre);
12     return names;
13 }
14
15 console.log(addNombre());
16
```

RunJavaScript

```
1 function getNombre() {
2     return 'Guillermo';
3 }
4 var getApellido = function () {
5     return 'Lemus';
6 };
7 var names = [];
8 function addNombre(nombre) {
9     if (nombre === void 0) { nombre = 'Blanca'; }
10     names.push(nombre);
11     return names;
12 }
13 console.log(addNombre());
14
```



# Nº indefinido de parámetros en una función

Select...

TypeScriptShareOptions

```
1 function getNombre() {
2   return 'Guillermo';
3 }
4 const getApellido = function () {
5   return 'Lemus';
6 }
7 let names: string[] = [];
8 function addNombre(
9   nombre: string = 'Blanca'
10 ): string[] {
11   names.push(nombre);
12   return names;
13 }
14 function addNombres(
15   ...nameStr: string[]): string[]
16 {
17   nameStr.forEach(function (name) {
18     this.names.push(name)
19   });
20   return this.names;
21 }
22
23 console.log(addNombres('Gretchen', 'Hansel'));
24
```

RunJavaScript

```
1 function getNombre() {
2   return 'Guillermo';
3 }
4 var getApellido = function () {
5   return 'Lemus';
6 };
7 var names = [];
8 function addNombre(nombre) {
9   if (nombre === void 0) { nombre = 'Blanca'; }
10  names.push(nombre);
11  return names;
12 }
13 function addNombres() {
14   var nameStr = [];
15   for (var _i = 0; _i < arguments.length; _i++)
16     nameStr[_i] = arguments[_i];
17 }
18 nameStr.forEach(function (name) {
19   this.names.push(name);
20 });
21 return this.names;
22 }
23 console.log(addNombres('Gretchen', 'Hansel'));
24
```

ElementsConsole

top

▼ Array(2) ⓘ  
0: "Gretchen"  
1: "Hansel"  
length: 2  
\_\_proto\_\_: Array(0)

> |

# Classes e Interfaces

---

---

Importancia de tener clases

Definición de una clase

Herencia

Definición de una interfaz

# Clases

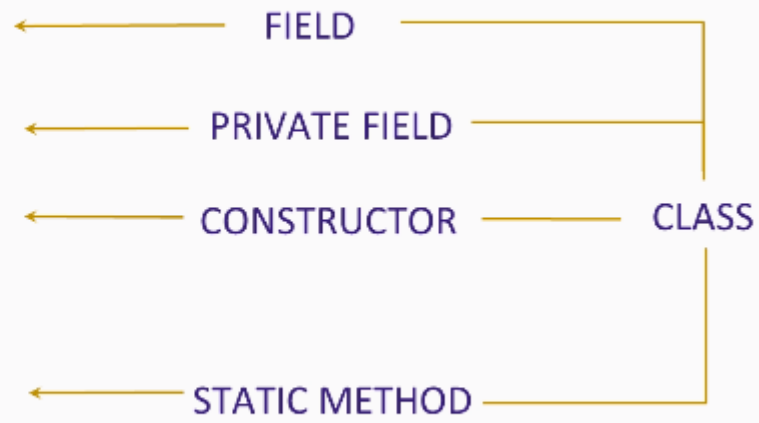
---

Ofrece una abstracción estructural

Forma consistente para los desarrolladores

Desarrollo orientado a objetos utiliza concepto de clase

```
class Person {  
    name  
    private surname  
    constructor  
  
    static calcBMI  
}
```



# Ejemplo de una clase

Select...

TypeScript

Share

Options

```
1 class Persona {
2     public hobby: string;
3     constructor(
4         private nombre: string,
5         private edad: number) { }
6     info(): void {
7         console.log(this.nombre + ' ' +
8             this.edad);
9     }
10 }
11 const pedro = new Persona('Pedro', 20);
12 pedro.info();
```

Run

JavaScript

```
1 var Persona = /** @class */ (function () {
2     function Persona(nombre, edad) {
3         this.nombre = nombre;
4         this.edad = edad;
5     }
6     Persona.prototype.info = function () {
7         console.log(this.nombre + ' ' +
8             this.edad);
9     };
10     return Persona;
11 }());
12 var pedro = new Persona('Pedro', 20);
13 pedro.info();
14
```

Elements Console

top

Pedro 20

> |



# Herencia

```
Select... TypeScript Share Options
1 class Persona {
2     public hobby: string;
3     constructor(
4         private nombre: string,
5         private edad: number) { }
6     info(): void {
7         console.log(this.nombre + ' ' +
8             this.edad);
9     }
10 }
11 const pedro = new Persona('Pedro', 20);
12 pedro.info();
13 class Jugador extends Persona {
14     constructor(nombre: string, edad: number) {
15         super(nombre, edad);
16     }
17 }
18 const alejandro = new Jugador('alejandro', 35);
19 alejandro.info();
```

```
Run JavaScript
1 var __extends = (this && this.__extends) || (function () {
2     var extendStatics = Object.setPrototypeOf ||
3         ({ __proto__: [] } instanceof Array && function (d, b) { for (var p in b) if (b.hasOwnProperty(p)) d[p] = b[p]; });
4     function __extends(d, b) {
5         __extends(d, b);
6         return function (d, b) {
7             extendStatics(d, b);
8             function __() { this.constructor = d; }
9             d.prototype = b === null ? Object.create(b.prototype) : b.prototype;
10         }();
11     }
12     var Persona = /** @class */ (function () {
13         function Persona(nombre, edad) {
14             this.nombre = nombre;
15             this.edad = edad;
16         }
17         Persona.prototype.info = function () {
18             console.log(this.nombre + ' ' + this.edad);
19         };
20         return Persona;
21     })();
22     var pedro = new Persona('Pedro', 20);
23     pedro.info();
24     var Jugador = /** @class */ (function (_super) {
25         __extends(Jugador, _super);
26         function Jugador(nombre, edad) {
27             return _super.call(this, nombre, edad) || this;
28         }
29         return Jugador;
30     })(Persona);
31     var alejandro = new Jugador('alejandro', 35);
32     alejandro.info();
33 }
```

Elements Console

top

Pedro 20

Alejandro 35

> |

Select... ▼

TypeScript

Share

Options

Run

JavaScript

```
1 class Persona {
2     public hobby: string;
3     constructor(
4         private nombre: string,
5         private edad: number) { }
6     info(): void {
7         console.log(this.nombre + ' ' +
8             this.edad);
9     }
10 }
11 const pedro = new Persona('Pedro', 20);
12 pedro.info();
13 class Jugador extends Persona {
14     constructor(nombre: string, edad: number) {
15         super(nombre, edad);
16     }
17 }
18 const alejandro = new Jugador('Alejandro', 35);
19 alejandro.info();
20 console.log(pedro);
21 console.log(alejandro);
22
```

```
1 var __extends = (this && this.__extends) || (function () {
2     var extendStatics = Object.setPrototypeOf ||
3         ({ __proto__: [] } instanceof Array && function (d, b) { for (var p in b) if (b.hasOwnProperty(p)) d[p] = b[p]; }) ||
4         function (d, b) {
5             extendStatics(d, b);
6             function __() { this.constructor = d; }
7             d.prototype = b === null ? Object.create(b) : (__.prototype = b.prototype, __);
8         };
9     return __extends;
10 })();
11 var Persona = /** @class */ (function () {
12     function Persona(nombre, edad) {
13         this.nombre = nombre;
14         this.edad = edad;
15     }
16     Persona.prototype.info = function () {
17         console.log(this.nombre + ' ' +
18             this.edad);
19     };
20     return Persona;
21 })();
22 var pedro = new Persona('Pedro', 20);
23 pedro.info();
24 var Jugador = /** @class */ (function (_super) {
25     __extends(Jugador, _super);
26     function Jugador(nombre, edad) {
27         return _super.call(this, nombre, edad) || this;
28     }
29     return Jugador;
30 })(Persona);
31 var alejandro = new Jugador('Alejandro', 35);
32 alejandro.info();
33 console.log(pedro);
34 console.log(alejandro);
```

Elements Console

top ▼

Pedro 20

Alejandro 35

▼ Persona 1

edad: 20

nombre: "Pedro"

▶ \_\_proto\_\_: Object

▼ Jugador 1

edad: 35

nombre: "Alejandro"

▶ \_\_proto\_\_: Persona

> |

---

```
class Persona {
  public hobby: string;
  constructor(
    private nombre: string,
    private edad: number) { }
  info(): void {
    console.log(this.nombre + ' ' +
      this.edad);
  }
}
const pedro = new Persona('Pedro', 20);
pedro.info();
class Jugador extends Persona {
  constructor(nombre: string, edad: number) {
    super(nombre, edad);
  }
}
const alejandro = new Jugador('Alejandro', 35);
alejandro.info();
console.log(pedro);
console.log(alejandro);
```

# Interfaz

---

```
interface Person {  
    name: string,  
    calcBMI()  
}
```



Contienen solo la decoración de los miembros

Es un contrato

Se usa para describir un tipo complejo

Select...

TypeScript

Share

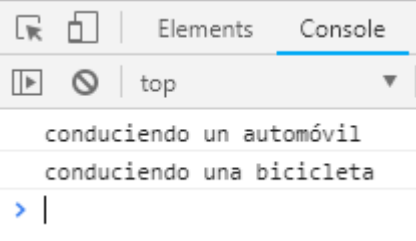
Options

Run

JavaScript

```
1
2 interface Persona {
3     nombre: string,
4     edad: number
5 }
6
7 const juan: Persona = {
8     nombre: 'Juan',
9     edad: 50
10 }
11
12 interface Vehiculo{
13     drive(): any;
14 }
15
16 class Automovil implements Vehiculo{
17     drive(): void {
18         console.log('conduciendo un automóvil');
19     }
20 }
21 class Bicicleta implements Vehiculo{
22     drive(): void {
23         console.log('conduciendo una bicicleta');
24     }
25 }
26
27 const auto = new Automovil();
28 const bici = new Bicicleta();
29
30 auto.drive();
31 bici.drive();
32
```

```
1 var juan = {
2     nombre: 'Juan',
3     edad: 50
4 };
5 var Automovil = /** @class */ (function () {
6     function Automovil() {
7     }
8     Automovil.prototype.drive = function () {
9         console.log('conduciendo un automóvil');
10    };
11    return Automovil;
12 }());
13 var Bicicleta = /** @class */ (function () {
14     function Bicicleta() {
15     }
16     Bicicleta.prototype.drive = function () {
17         console.log('conduciendo una bicicleta');
18     };
19    return Bicicleta;
20 }());
21 var auto = new Automovil();
22 var bici = new Bicicleta();
23 auto.drive();
24 bici.drive();
25
```



---

```
interface Persona {
  nombre: string,
  edad: number
}
const juan: Persona = {
  nombre: 'Juan',
  edad: 50
}
interface Vehiculo{
  drive(): any;
}
class Automovil implements Vehiculo{
  drive(): void {
    console.log('conduciendo un automóvil');
  }
}
class Bicicleta implements Vehiculo{
  drive(): void {
    console.log('conduciendo una bicicleta');
  }
}

const auto = new Automovil();
const bici = new Bicicleta();

auto.drive();
bici.drive();
```

---

# Genericos

---

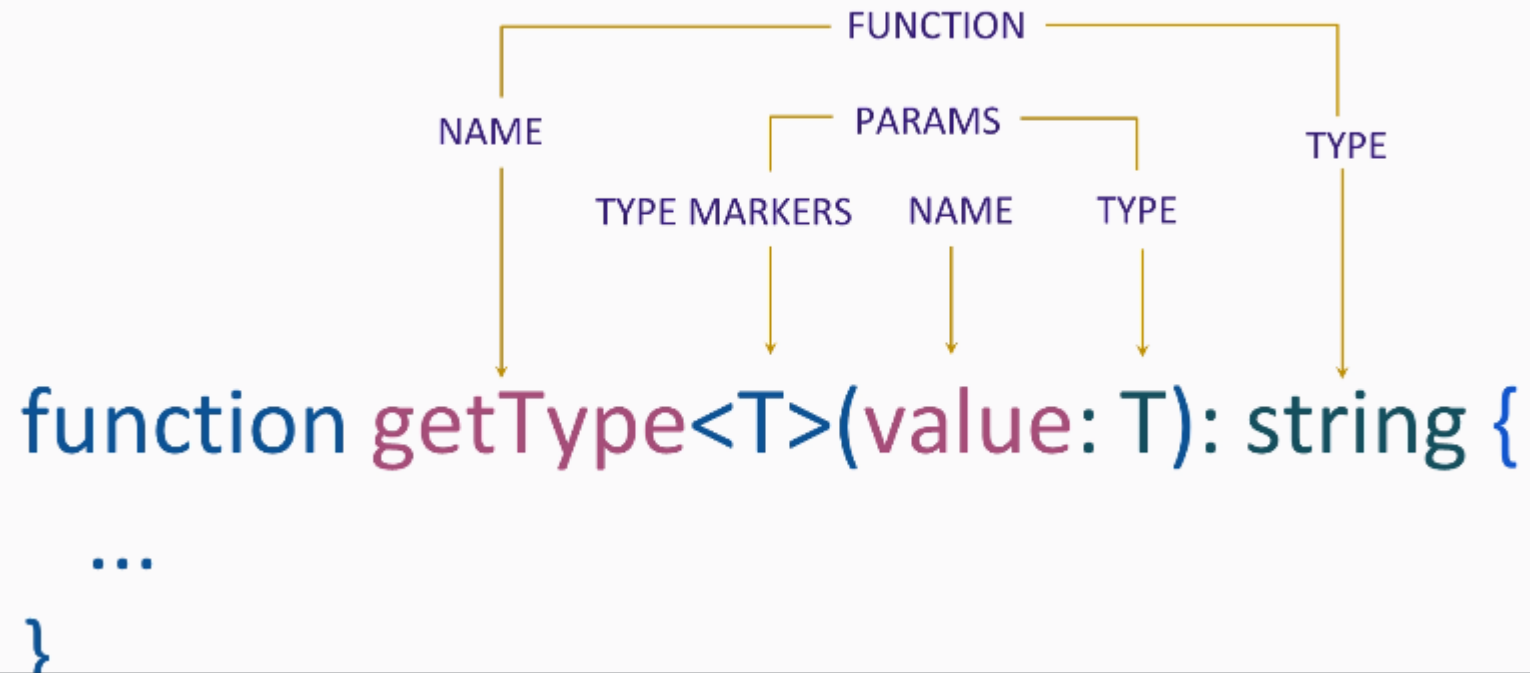
---

Función genérica

Problema de cola

Solución al problema de cola





Select...

TypeScript

Share

Options

```
1 Number.toString();
2 String.toString();
3 Boolean.toString();
4
5 function primitivetoString<T>(type: T): string {
6     return type.toString();
7 }
8
9 [Number, Boolean, String].forEach(function (type)
10     console.log(primitivetoString(type));
11 })
```

Run

JavaScript

```
1 Number.toString();
2 String.toString();
3 Boolean.toString();
4 function primitivetoString(type) {
5     return type.toString();
6 }
7 [Number, Boolean, String].forEach(function (type)
8     console.log(primitivetoString(type));
9 });
10
```

Elements Console Sources

top Filter

function Number() { [native code] }

function Boolean() { [native code] }

function String() { [native code] }

> |

---

```
Number.toString();  
String.toString();  
Boolean.toString();
```

```
function primitivetoString<T>(type: T): string {  
    return type.toString();  
}
```

```
[Number, Boolean, String].forEach(function (type) {  
    console.log(primitivetoString(type));  
})
```

Select...

TypeScript

Share

Options

```
1 class ColaDeNumeros {
2   private datos = [];
3   push(item: number) {
4     this.datos.push(item);
5   }
6   pop() {
7     this.datos.shift();
8   }
9 }
10 const cola = new ColaDeNumeros();
11 cola.push(1);
12 cola.push(2);
13 cola.push(3);
14 console.log(cola);
```

Run

JavaScript

```
1 var ColaDeNumeros = /** @class */ (function () {
2   function ColaDeNumeros() {
3     this.datos = [];
4   }
5   ColaDeNumeros.prototype.push = function (item) {
6     this.datos.push(item);
7   };
8   ColaDeNumeros.prototype.pop = function () {
9     this.datos.shift();
10  };
11  return ColaDeNumeros;
12 }());
13 var cola = new ColaDeNumeros();
14 cola.push(1);
15 cola.push(2);
16 cola.push(3);
17 console.log(cola);
18
```

Elements Console

▼ ColaDeNumeros 1  
▶ datos: (3) [1, 2, 3]  
▶ \_\_proto\_\_: Object

> |

---

```
class ColaDeNumeros {  
  private datos = [];  
  push(item: number) {  
    this.datos.push(item);  
  }  
  pop() {  
    this.datos.shift();  
  }  
}  
const cola = new ColaDeNumeros();  
cola.push(1);  
cola.push(2);  
cola.push(3);  
console.log(cola);
```

---

Select...

TypeScript

Share

Options

```
1 class Cola<T> {
2     private datos = [];
3     push(item: T) {
4         this.datos.push(item);
5     }
6     pop() {
7         this.datos.shift();
8     }
9 }
10 const colaDeNumeros = new Cola<number>();
11 colaDeNumeros.push(10);
12 colaDeNumeros.push(20);
13 console.log(colaDeNumeros);
14
15 const colaDeCadenaCaracteres = new Cola<string>();
16 colaDeCadenaCaracteres.push('Curso');
17 colaDeCadenaCaracteres.push('TypeScript');
18 console.log(colaDeCadenaCaracteres);
```



Run

JavaScript

```
1 var Cola = /** @class */ (function () {
2     function Cola() {
3         this.datos = [];
4     }
5     Cola.prototype.push = function (item) {
6         this.datos.push(item);
7     };
8     Cola.prototype.pop = function () {
9         this.datos.shift();
10    };
11    return Cola;
12 }());
13 var colaDeNumeros = new Cola();
14 colaDeNumeros.push(10);
15 colaDeNumeros.push(20);
16 console.log(colaDeNumeros);
17 var colaDeCadenaCaracteres = new Cola();
18 colaDeCadenaCaracteres.push('Curso');
19 colaDeCadenaCaracteres.push('TypeScript');
20 console.log(colaDeCadenaCaracteres);
21
```

Elements Console

top

- Cola 
  - datos: Array(2)
    - 0: 10
    - 1: 20
  - length: 2
  - \_\_proto\_\_: Array(0)
- Cola 
  - datos: Array(2)
    - 0: "Curso"
    - 1: "TypeScript"
  - length: 2
  - \_\_proto\_\_: Array(0)

> |

---

```
class Cola<T> {  
  private datos = [];  
  push(item: T) {  
    this.datos.push(item);  
  }  
  pop() {  
    this.datos.shift();  
  }  
}  
  
const colaDeNumeros = new Cola<number>();  
colaDeNumeros.push(10);  
colaDeNumeros.push(20);  
console.log(colaDeNumeros);  
  
const colaDeCadenaCaracteres = new Cola<string>();  
colaDeCadenaCaracteres.push('Curso');  
colaDeCadenaCaracteres.push('TypeScript');  
console.log(colaDeCadenaCaracteres);
```

