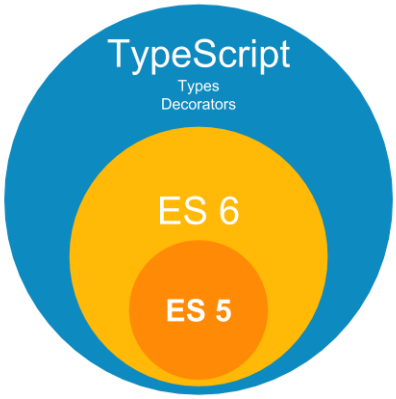


TypeScript is a javascript superset

Que es Typescript?

Es un lenguaje super conjunto de javascript desarrollado por Microsoft

TypeScript convierte su código en Javascript común (Transpile).

Lenguaje orientado a objetos.

Lenguaje

Syntax , type annotations (como tipado estatico)

Es6 syntax (evolución de javascript)

Compilador

Conversión a código js

Su compilador está escrito en TS

Compila .ts a .js

Instalación:

```
npm install -g typescript
```

compilación:

```
tsc mi-codigo.ts
```

archivo compilado: mi-codigo.js

Sintaxis basica:

number (integers y floating point numbers)

string

boolean

Array<T> y T[].

number[] - array de numbers

Array<string> - array of strings

[boolean, string] - tuple

[number, number, number] - tuple de 3 numbers

{} – object

{nombre: string, edad: number} - object

{[key: string]: number}

enum - { Red = 0, Blue, Green }

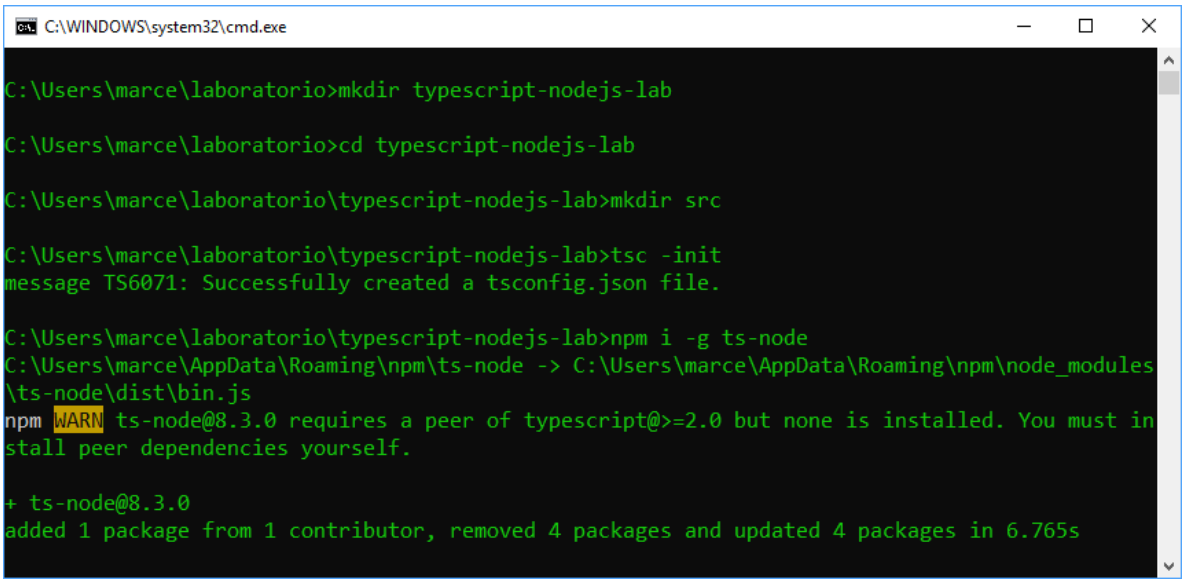
any

void

null

Corriendo TypeScript usando ts-node

npm install -g ts-node



ts-node src/index.ts

```
class Saludo {
  texto: string;

  constructor(mensaje: string) {
    this.texto = mensaje;
  }

  saludar(): string {
    return this.texto;
  }

}

let saludo = new Saludo("Hola Mundo!");
console.log(saludo.saludar());
```

Funciones:

```
const factorial = (num : number): number => {
    if (num <= 0 ) return 1
    else return (num * factorial(num-1))
 }

 console.log(factorial(5));
```

Interfaces:

```
interface Persona {
      nombres: string,
      apellido : string,
      edad : number,
      estado : boolean
  };

  let alumno : Persona;

  alumno = { nombres : 'ana', apellido: 'gomez', edad: 25 , estado : true};

  console.log('alumno:',alumno);

  let alumnos : Persona[] = [];

  alumnos.push(alumno);
  alumnos.push({ nombres : 'jorge', apellido: 'chavez', edad: 30 , estado : true});
  alumnos.push({ nombres : 'abel', apellido: 'lopez', edad: 27 , estado : true});

  alumnos.forEach(alumno => {
      console.log('alumnos:',alumno);
  })
```

String metodos

Método	Descripción
charAt()	Devuelve el carácter en el índice dado
concat()	Devuelve una combinación de las dos o más cadenas especificadas
indexOf()	Devuelve un índice de la primera aparición de la subcadena especificada de una cadena (-1 si no se encuentra)
replace()	Reemplaza la subcadena coincidente con una nueva subcadena
split()	Divide la cadena en subcadenas y devuelve una matriz
toUpperCase()	Convierte todos los caracteres de la cadena en mayúsculas
toLowerCase()	Convierte todos los caracteres de la cadena en minúsculas
charCodeAt()	Devuelve un número que es el valor de la unidad de código UTF-16 en el índice dado
codePointAt()	Devuelve un número entero no negativo que es el valor del punto de código del punto de código codificado UTF-16 que comienza en el índice especificado
includes()	Comprueba si una cadena incluye otra cadena
endsWith()	Comprueba si una cadena termina con otra cadena
LastIndexOf()	Devuelve el índice de la última aparición de valor en la cadena
localeCompare()	Comprueba si una cadena viene antes, después o es igual a la cadena dada
match()	Coincide con una expresión regular contra la cadena dada
normalize()	Devuelve el formulario de normalización Unicode de la cadena dada.
padEnd()	Rellena el final de la cadena actual con la cadena dada

Método	Descripción
padStart()	Rellena el comienzo de la cadena actual con la cadena dada
repeat()	Devuelve una cadena que consta de los elementos del objeto repetidos en los tiempos dados.
search()	Busca una coincidencia entre una expresión regular y una cadena
slice()	Devuelve una sección de una cadena
startsWith()	Comprueba si una cadena comienza con otra cadena
substr()	Devuelve una cadena que comienza en la ubicación especificada y de los caracteres dados
substring()	Devuelve una cadena entre los dos índices dados
toLocaleLowerCase()	Devuelve una cadena en minúsculas respetando la configuración regional actual
toLocaleUpperCase()	Devuelve una cadena en mayúscula respetando la configuración regional actual
trim()	Recorta el espacio en blanco desde el principio y el final de la cadena.
trimLeft()	Recorta el espacio en blanco del lado izquierdo de la cuerda
trimRight()	Recorta el espacio en blanco desde el lado derecho de la cuerda

Clases:

/src/Empleado.ts

```
export default class Empleado {
    codigo: number;
    nombres: string;

    constructor(codigo: number,nombres: string) {
        this.codigo = codigo;
        this.nombres = nombres;
    }

    getSalario() : number {
        return 2500;
    }

}
```

/src/index.ts

```
import Empleado from './Empleado';

let empleado = new Empleado(1,'Ana Gomez');

console.log('---[ clases ]---')
console.log(`Empleado : ${empleado.nombres} ${empleado.getSalario() }
`);
```

/src/Alumno.ts

```
class Persona {
    codigo: number;
    nombres: string;

    constructor(codigo: number,nombres: string){
        this.codigo = codigo;
        this.nombres = nombres;
    }
}

class Alumno extends Persona {
    matricula : number;

    constructor(matricula : number,codigo : number, nombres: string){
        super(codigo,nombres)
        this.matricula = matricula;
    }

    verRegistro(): void {
        console.log(` ${this.nombres}, matricula : ${this.matricula} `);
    }
}

export default Alumno;
```

/src/index.js

```
console.log('---[ clases herencia ]---')
let alum = new Alumno(22222,123,'Jose Vargas');

alum.verRegistro();
```

/src/Curso.ts

```
class Curso {

    public codigo: number;
    public nombre: string;
    private tipo : string;

    constructor(){
        this.tipo = 'PRESENCIAL';
        this.codigo = 0;
        this.nombre = '';
    }

    info(): string {
        return ` codigo : ${this.codigo} - ${this.nombre} - ${this.tipo} `;
    }

}

export default Curso;
```

/src/index.ts

```
import Curso from './Curso';

console.log('---[ clases public, private ]---');

let curso = new Curso();

curso.codigo = 123;
curso.nombre = 'Jorge Chavez';

console.log(curso.info());
```

/src/Util.ts

```
class Util {

    static monedaNacional = 'BOB';

    static monedaDolarAmericano = 'USD';

    static fechaActual(): string {
        const fecha = new Date();
        const dd = fecha.getDate();
        const mm = fecha.getMonth() + 1;
        const yyyy = fecha.getFullYear();
        let dia : string = '';
        let mes : string = '';
        if (dd < 10) {
            dia = '0' + dd;
        } else { dia = dd.toString() }
        if (mm < 10) {
            mes = '0' + mm;
        }  else { mes = mm.toString() }
        const hoy = dia + '/' + mes + '/' + yyyy;
        return hoy;
    }

}

export default Util;
```

/src/index.js

```
console.log('---[ clases static ]---');

import Util from './Util';

console.log(Util.fechaActual());
console.log(Util.monedaNacional);
```