Lexpin

¿Qué es TypeScript?

TypeScript es un lenguaje de programación de código abierto desarrollado por Microsoft que se basa en JavaScript. Es un superset de JavaScript que agrega constructores que no se encuentran en JavaScript.

¿TypeScript es lo mismo que JavaScript?

Se podría decir que en gran parte si, sin embargo, no es exactamente lo mismo, se compila a JavaScript y se puede utilizar en lugar de JavaScript en aplicaciones web, aplicaciones de servidor, aplicaciones de escritorio y otros tipos de proyectos de software. Cabe recalcar que la sintaxis básica de TypeScript es muy similar a la de JavaScript, por lo que es relativamente fácil para los desarrolladores de JavaScript aprenderlo.

Sin embargo, a diferencia de JavaScript, TypeScript permite la declaración de tipos de datos, lo que puede mejorar la legibilidad y mantenibilidad del código. Por ejemplo, en TypeScript, se puede declarar el tipo de una variable como un número, una cadena, un booleano, un objeto, una función, etc.

¿Cómo instalar TypeScript?

Para instalar TypeScript en tu computadora, sigue estos pasos:

1. **Instalar Node.js**: TypeScript se ejecuta en Node.js, por lo que debes instalar Node.js en tu computadora. Puedes descargar el instalador de Node.js desde la página web oficial de Node.js: https://nodejs.org/en/
2. **Instalar TypeScript**: Una vez que tienes Node.js instalado en tu computadora, puedes instalar TypeScript utilizando el siguiente comando en la línea de comandos: **npm install -g typescript**. Este comando instalará TypeScript de manera global en tu sistema.
3. **Verificar la instalación**: Para verificar que TypeScript se haya instalado correctamente, puedes ejecutar el siguiente comando en la línea de comandos: **tsc –v** o **-version.** Este comando imprimirá la versión de TypeScript que se encuentra instalada en tu computadora.
4. Personaliza a que versión de JavaScript se va a traducir el código escrito: abre la consola en la carpeta de tu proyecto y escribe **stc -init**, esto creara un archivo llamado **tsconfig.json**, en el puedes elegir una versión de JavaScript (Las versiones con mayor compatibilidad son ES5 y ES6)

¿Qué es el tipado de datos?

El tipado de datos se refiere a la práctica de declarar explícitamente el tipo de datos que se almacenan en una variable, parámetro de función, propiedad de objeto o cualquier otro elemento del código.

TypeScript es un lenguaje de programación tipados estáticamente (ver Glosario), a diferencia de JavaScript, donde el tipado es dinámico (ver Glosario).

Tipos de datos:

Con TypeScript se pueden utilizar varios tipos de datos que se dividen en dos categorías:

- **Tipos de datos primitivos**:
  1. **number**: para números enteros o decimales. Ejemplo: let edad: number = 20;
  2. **string**: para cadenas de texto. Ejemplo: let nombre: string = "Diego";
  3. **boolean**: para valores verdaderos o falsos (true o false).
     Ejemplo: let active: boolean = true;
  4. **null**: para valores nulos. Ejemplo: let miVariable: null = null;
  5. **undefined**: para valores indefinidos.
     Ejemplo: let miVariable: undefined = undefined;

Tipos de datos complejos:

- **object**: para cualquier objeto que no sea de tipo primitivo.
  Ejemplo:
  ```
  let persona: {nombre: string, edad: number} = {nombre: "Daniel", edad: 27};
  ```

- **array**: para listas de valores, puede ser de tipo primitivo o de objeto.
  Ejemplo:
  ```
  let numeros: number[] = [1, 2, 3, 4];
  ```

- **tuple**: para una secuencia fija de tipos de datos conocidos.
  Ejemplo:
  ```
  let tuple: [string, number] = ["Marcos", 25];
  ```

- **enum**: se utiliza para definir un conjunto de valores con nombre y asignarles un valor numérico por defecto.
  Ejemplo:
  ```
  enum Color {Red,  Green,  Blue}
  let miColor: Color = Color.Green;
  ```

- **any**: para cualquier tipo de valor, incluyendo tipos de datos desconocidos o dinámicos.
  Ejemplo:
  ```
  let variable: any = "Esto puede ser cualquier valor";
  ```

- **void**: para funciones que no devuelven ningún valor.
  Ejemplo:

```
function miFuncion(): void {
console.log("Aquí se retorna cualquier valor");
}
```

Glosario

Superset: Un superset es un lenguaje de programación que incluye todas las características y funcionalidades de otro lenguaje de programación, pero que también tiene características adicionales y funcionalidades que no están disponibles en el lenguaje de origen.

Tipado estático: En un lenguaje de programación con tipado estático, se verifica la corrección del tipo de datos en tiempo de compilación. Es decir, antes de ejecutar el programa, el compilador verifica que todas las variables y expresiones tengan el tipo de datos correcto. El compilador puede encontrar errores de tipo antes de que se ejecute el programa, lo que puede prevenir errores en tiempo de ejecución.

Tipado dinámico: En un lenguaje de programación con tipado dinámico, los tipos de datos se verifican en tiempo de ejecución. Es decir, el programa se ejecuta primero y durante la ejecución se verifica si las variables y expresiones tienen el tipo de datos correcto. Esto significa que los tipos de datos no se detectan hasta que el programa se ejecute, dando lugar a errores durante la ejecución.