**TYPESCRIPT**

**[2025]**

|  |
| --- |
|  |

# : Introduction:

* What is Typescript?
* Why Typescript?
* We can define data types here sometime we can call static typing.
* Improved code maintainability.
* Code quality.
* Advanced features.
* Important Typescript features?
* Define data types.
* Interfaces.
* Decorators.
* Namespaces.
* Type interface.
* Advanced features.
* Code quality.
* Installation setup?
* Locally installation means only one projects inside typescript install.

npm install typescript --save-dev

* Globally installation means we can use typescript entire our system (laptop).

npm install -g typescript

* How to run Typescript File?
* Create, app.ts file.
* Run this command to generate Js file because browser understand only JS file not TS (compilation every time).

npx tsc app.ts

* Compilation every time.

npx tsc app.ts --watch

* Interview Questions?
* **tsc full form** → TypeScript Compiler.
* **--watch command use** → Recompiles automatically on file changes.
* **npm full form** → Node Package Manager
* **Can we create name, name variable in TypeScript?**

No, because name, name is already defined variable in window object that is why we cannot create variable name, name.

# : Data Type in TypeScript:

* What are Data Types?

Data types specify what kind of data can be stored in a variable.

* Data Types in Typescript?
* Primitive types.
* Objects types.
* Special types.
* Advanced types.
* Function types.
* What are primitive data types?

Primitive data types are inbuild data types.

Ex: number, string, Boolean, bigInt, null, undefined, symbol.

* Object Data types?

These included Arrays, Objects, Tuples.

Ex: Array, Object, Tuple.

* Array: array is a collection of elements same types, because it is number types of arrays.

Ex: var num: number [] = [1,2,3,4,5,6,7];

* Tuples: Tuples is an array here we can include numbers, string, object, Boolean, all types of data.

Ex: var person: [number, string] = [25, “Ashu”];

* Object: Object is a key and values pair which is separate defined data.

Ex: var obj: {id: number, name: string, isActive: boolean} = {id:1, name: “Ashu", isActive: true};

console.log(obj);

* Special Data Types?

These are extra types provided by Typescript,

Ex: Any, unknown, void, never.

* Any: any can hold any types of data.
* Unknown: unknown like any but must be typed checked before use.
* Void: Void used for functions that don’t return anything.
* Never: We can use Never with the function which we do not want to return.
* Advanced Data Types?

Ex: union, intersection, Type alias, Enum, Literal types.

* Union: union allows a variable to hold multiple types.
* Intersection: intersection is a combination of multiple types into one.
* Type alias: Type alias, you can create a custom name for a type.
* Enum: Enum defines a set of named constants.
* Literal types: Literal types, restrict variable to specific values.
* Function Data Types?

Defines a type of a function.

# : Primitive Data Types & TypeScript Configuration:

* **What are primitive data types?**

Primitive data types are inbuild data types.

Ex: number, string, Boolean, bigInt, null, undefined, symbol.

**Syntax**:

var variable\_name: datatypes = Value;

* **Number Data Types**: Represents both integer and floating-point numeric values.
* **String Data Types**: Represents textual data enclosed in single, double, or backticks.
* **Boolean Data Types**: Represents only two values: true or false.
* **BigInt Data Types**: Represents integers of arbitrary length, larger than Number can hold. Maximum Integer Number, MAX\_SAFE\_INTEGER = 2^53 -1 (TWO KI POWER 53 MINUS ONE)
* **Null Data Types**: Represents an intentional absence of any object value.
* **Undefined Data Types**: Represents a variable that has been declared but not assigned a value.
* **Symbol Data Types**: Represents a unique and immutable identifier.
* **TypeScript Config file:**
* How to generate config file:

tsc --init

* Use of config file:

We can change according to my use case.

* Converts all TS file into JS file:

tsc

* Common use of config:

# 