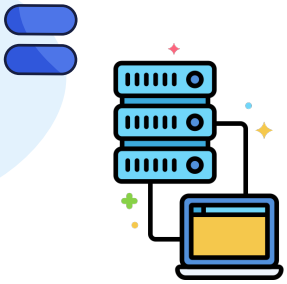
SMK Telkom Makassar

CLIENT SIDE

# PROGRAMMING





# TypeScript

TypeScript merupakan salah satu bahasa pemrograman tingkat tinggi yang dikembangkan oleh Microsoft Corporation pada tahun 2012. Bahasa pemrograman secara penggunaan mirip dengan JavaScript, tetapi memiliki beberapa fitur tambahan. TypeScript dirancang untuk pengembangan aplikasi skala besar dan dapat dikompilasi menjadi JavaScript.
Bahasa pemrograman ini merupakan superset dari JavaScript, ini menyebabkan secara sintaks, semua program JavaScript adalah TypeScript juga. Walaupun mirip, tapi bisa jadi juga bisa terjadi ketidakcocokan ketika penulisan dalam pemeriksaan pengetikan karena alasan keamanan.

## Deteksi eror lebih awal

Umumnya ketika menulis kode menggunakan JavaScript ataupun bahasa lainnya, jika terjadi eror maka program tidak akan bisa dijalankan. TypeScript merupakan bahasa tipe statis, dimana pengecekan bug atau eror dilakukan sebelum program dieksekusi dan memberikan saran terkait eror yang terjadi.

## Superset Javascript

Seperti yang sudah disinggung di awal, TypeScript merupakan superset JavaScript sehingga sintaks Javascript dianggap valid oleh TypeScript. Sintaks mengacu pada cara kita menulis kode untuk membentuk suatu program. Ini berarti pengguna dapat mengambil kode JavaScript apa pun yang berfungsi dan memasukkannya ke dalam TypeScript tanpa mengkhawatirkan cara penulisannya.

## Fleksibilitas lebih tinggi

Sebagai superset dari JavaScript, TypeScript dapat diintegrasikan dengan bahasa pemrograman lain yang mendukung JavaScript, seperti Node.js, React, dan Angular. Ini menyebabkan bahasa ini lebih fleksibel.

## Installasi

untuk menggunakan typescript pastikan di komputer anda telah terinstall node js, npm/yarn versi yang lebih baru. buatlah folder proyek dan install typescript:

```
npm install -g typescript
```

untuk mengecek versi dari typescript:

```
tsc -v
```

selanjutnya buat file konfigurasi dengan:

```
tsc --init
```

## tsconfig.json

lakukan beberapa file konfigurasi tsconfig.json dengan cara menghilangkan komentar:

```
“noImplicitAny”: true,
“outDir”: “./build”,
“rootDir”: “./src”,
```

buatlah floder src pada folder proyek anda:


buatlah file dengan extensi .ts untuk memulai typescript di dalam folder src:

```
let sesuatu:string=”hello world”;
console.log(sesuatu);
```

kemudian compile dan jalankan dengan:

```
tsc && node build/latihan.js
//hello world
```

## Lets Code

1. type data string

```
let saya:string=”saya adalah string”;
//let saya:string
//saya=”saya adalah string”
```

2. type data number

```
let angka:number=1234;
//let angka:number
//number=1234
```

3. type data boolean

```
let kondisi:boolean=true;
//let kondisi:boolean
//kondisi=true
```

4. type data any

```
let hero:any=”siderman”;
hero=5678
```

4. type data array

```
let data:string[]
data=[“aku”,”suka”,”kamu”]
let data2:nmber[]
data=[1,2,3]
```

```
let campuran:any[];
campuran=[1,'belajar',true,{jk:'laki-laki'}]
```

2. type data array tuples

```
let biodata:[string,number]
biodata=['baco',18]
```

3. type data enum

```
//number
enum bulan{
    jan=1,
    feb,
    mar
}
//string
enum bulan{
    jan='januari',
    feb='februari',
    mar='maret'
}
```

4. fungsi

```
function halo():string{
  return "it work"
}

//menggunakan void
function halo():void{
  console.log('hello')
}
halo()
```

```
//dengan argumen/parameter
  function perkalian (x:number,y:number)
    :number{
        return x*y
    }
     let hasil=perkalian(9,2)
     console.log(hasil)

//function as type
type tambah=(val1:number,val2:
number)=>number;
const add:tambah=(x:number,y:number)
 :number=>{
  return x+y
 }
console.log(add(3,3))
```

5. Object

```
type User={
    username:string,
    age:number
}
const akun:User={
    username:"admin",
    age:23
}
```

6. union type

Tipe data Union adalah fitur yang memungkinkan sebuah variabel atau parameter memiliki lebih dari satu tipe data. Ini berarti variabel tersebut dapat menampung beberapa tipe yang berbeda, dan TypeScript akan memastikan bahwa variabel tersebut hanya menerima salah satu dari tipe-tipe yang dideklarasikan.

```
let hp:number | string
 hp=6281288236771;
 hp="081288236771";
```

7. fungsi dengan default parameter

```
const namaLengkap=(depan:string,
  belakang:string="alfian"):string=>{
  return depan+ ' '+ belakang
}
  console.log(namaLengkap('baso'))
```

3. fungsi dengan optional parameter

TypeScript memungkinkan kamu mendefinisikan parameter opsional dengan menambahkan tanda tanya ***(?)*** setelah nama parameter. Jika parameter tersebut tidak diberikan saat fungsi dipanggil, secara default nilainya akan menjadi undefined

```
  const namaLengkap=(depan:string,
   belakang?:string):string=>{
   return depan+ ' '+ belakang
}
  console.log(namaLengkap('baso'))
```

4. class

```
export class User{
  public nama: string
  constructor(nama:string){
      this.nama=nama
  }
}
let user=new User("andi baso");
console.log(user.nama)
```

```
export class User{
    constructor(public nama:string){
    }
}
let user=new User("andi baso");
console.log(user.nama)
```