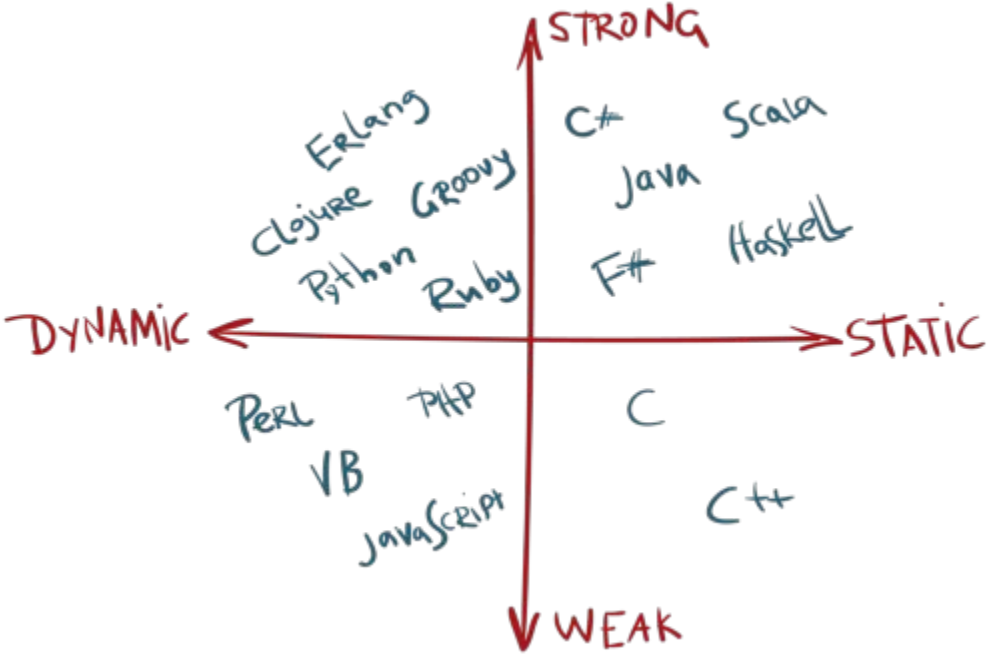
TypeScript

Maktab Sharif Coding Bootcamp

Ali MohammadAliRajab

Types overview



What's the difference between strong and weak types?

```
→  ~ python3
Python 3.8.5 (default, Jul 28 2020, 12:59:40)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> "ali"+17
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can only concatenate str (not "int") to str
>>>
```

```
→  ~ node
Welcome to Node.js v14.17.0.
Type ".help" for more information.
> "mohammad"+300
'mohammad300'
>
```

What’s the difference between strong and weak types?

```cpp
#include <iostream>
#include <string>
using namespace std;
int main() {
  int num = "123456";
  // string num = "123456"; correct type notation
  cout << num;
  return 0;
}
```

```
python3

→  ~ python3
Python 3.8.5 (default, Jul
12:59:40)
[GCC 9.3.0] on linux
Type "help", "copyright", "
or "license" for more infor
>>> var = 123456
>>> var
123456
>>> var = "dynamic type"
>>> var
'dynamic type'
>>>
```

Static type JavaScript alternatives

Let’s started! - Basic Types


```typescript
const isHappy: boolean = true // boolean
let myAge: number = 20 // number
var myName: string = 'Ali' // string
let introduceMyself: string = `I'm ${myAge} and my name is ${myName}.` // string with backtick notation

let cars: string[] = ['Ford Mustang', 'Corvette', 'Cadillac Eldorado'] // Array
let cars2: Array<string> = ['Hummer', 'Ford GT40', 'Tesla Model S'] // Array
let company: object = {
  name: 'MaktabSharif',
} // Object

let chert: undefined = undefined // Undefined
let pert: null = null // Null
```

TypeScript Types

```
1 let price: [string, number]; // Tuple Declaration
2 price = ["Pizza", 1.2]; // Tuple Initialization
3
4 // enum
5 enum Size { Small = 0.7, Medium = 1.2, Large = 2}
6 let pizzaSizeName: string = Size[2];
7 let pizzaSizeNum: number = Size.Large;
```

TypeScript Types

```typescript
// any (danger zone !)
let ts_dynamic_var_fun: any = ":D";
ts_dynamic_var_fun = Size.Small;

//  void
let sing = (): void => console.log("lalalalalala la la la laaaaaa");

// never (void + will never occur)
let error = (): never => {
  throw new Error(":|");
};
```

Be care about type ‘Any’

TypeScript Types & Interfaces

```typescript
// interface
interface User {
  id: number;
  username: string;
  password: string;
}
var sayWelcome = (user: User): void => {
  console.log(`Welcome ${user.username}`);
};
```

https://medium.com/@martin_hotell/interface-vs-type-alias-in-typescript-2-7-2a8f1777af4c

TypeScript Types & Interfaces

```
1 var sayWelcome = (user: {
2   id: number;
3   username: string;
4   password: string;
5 }): void => {
6   console.log(`Welcome ${user.username}`);
7 };
8 // type
9 type User2 = {
10   id: number;
11   username: string;
12   password: string;
13 };
```

TypeScript Interfaces

```
1 interface Admin extends User {
2   Nationality?: string;
3   readonly nationalCode: string;
4 }
5
6 let ali = {} as Admin;
7 ali.password = "password";
```

TypeScript Classes

```typescript
class Animal {
  static defaultSing: string = "lalaalalaaaaaa";
  public type: string = "Animal";
  private sing: string = "lalallalaaa";

  constructor(sound: string) {
    this.sing = sound;
  }

  greet(): string {
    return `Hello ${this.sing}`;
  }
}

let lion = new Animal("RAAAWWWR");
console.log(lion.greet());
console.log(Animal.defaultSing);
```

TypeScript Union types

```
1 // Union
2 let confused: string | number | boolean = "ali";
3 confused = true;
4 confused = 123456;
5
6 // Union types are solution of this problem:
7 let y = "I'm y !";
8 y = 123; // you can't do that because initial
  value type was string!
```

Ali MohammadAliRajab

mohammadalirajab.a@gmail.com