Typescript + ES6

Knowledge round-up

Typescript

+ What is TypeScript and Why do we need it?

TypeScript is an open source and developed by Microsoft. It is a superset of Javascript that compiles to clean Javascript output. In addition, it adds optional types, OOP classes and modules to JavaScript.

+ We need it because:

- It is an open source -> free
- Almost framework encourage using typescript to develop project
- With OOP and newest technical, it useful to develop large project

+ How can you get TypeScript and install it?

Requirement: install npm and nodejs (newest version was encouraged)
Install: Run command below in CLI:
`npm install -g typescript`

+ How do you compile TypeScript files?

Run command below in CLI:
`tsc <path/filename.ts>`

+ Which Object Oriented terms are supported by TypeScript?

- Class
  Allow to use object-oriented class-based approach and compile them down to Javascript
  Example:

```
class ATer {
    level: string = "Undefine";
    fullName: string;
    constructor(fullName: string) {
        this.fullName = fullName;
    }

    greeting() {
        return "Hello, I'm " + this.fullName + ", and I'm working at Asian Tech as " + this.level + "!
    }
}

class Intern extends ATer {
```

```
    super(level);
    level = "Internship";
    constructor(fullName: string) {
        super(fullName);
    }
}

class ASE extends ATer {
    super(level);
    level = "ASE";
    constructor(fullName: string) {
        super(fullName);
    }
}

var intern = new Intern("Nguyen Van A");
var ase = new ASE("Kimono");
console.log(intern.greeting());
console.log(ase.greeting());
```

- Interface
  Allow to use Interface describes object
  Example

```
interface Boss {
    name: string;
    position: string;
}

function sayHello(boss: Boss) {
    var helloSentence = "Hello, I'm " + boss.position + ", " + boss.name;
    return helloSentence;
}

var myBoss: Boss = {
    name: "Duong Qua",
    position: "CEO"
}

console.log(sayHello(myBoss));
```

+ How do you implement inheritance in TypeScript?

Use extends to implement inheritance.
Use super() to call the base class constructor and update property in base class
Example: *I use interface and sub class with overloading, extends i have used above*

```
interface Boss {
    name: string;
    position: string;
    sayBye():void;
    sayBye():string;
}
```

```
class subBoss implements Boss {
    constructor() {
        console.log('inside subBoss rightnow');
    }

    sayBye():void {
        console.log('subBoss say Goodbye!');
    }

    sayBye():string {
        return 'subBoss say Goodbye!';
    }
}

var subBoss = new subBoss();
var sayBye:string = subBoss.sayBye();
console.log(sayBye);
```

ES6

Block scope variable

```
//this function will show undefine
function whoAmI(me) {
    var myName = "undefine";
    if ( me == "hihi" ) {
        let myName = "Ronaldo";
    }
    {
        let myName = "Messi";
        {
            let myName = "Rooney";
        }
    }
    return myName;
}
```

Template Literals

```
var innerHTML = whoAmI("test");
var $pTag = document.getElementById('p-tag');
$pTag.innerHTML = `Result: ${innerHTML}`;
```

Multi-line strings

```
var multiLine = `1 One
                 2 Two
                 3 Three
                 4 Four
                 5 Five`;
```

Arrow functions

```
var showMyName = name => {
    return name;
}

var $pTag = document.getElementById('p-tag');
$pTag.innerHTML = `Show My Name: ${myName}`;
```

for..of

```
var randomArray = ["hung", 22, "quang tri"];
var innerHTMLSub = '';
for(var item of randomArray) {
    innerHTMLSub += `${item} <br>`;
}

$pTag.innerHTML = innerHTMLSub;
```

Default parameters

```
function showMyAge(age:number = 22) {
    return age;
}

showMyAge();
//call this function will return 22
```