TypeScript

Jeremie Amsellem < classes@lp1.eu >

Sommaire

- Qu’est-ce que TypeScript ?
- Les spécifités de TypeScript
  - Les annotations de type
  - Les types primitifs
  - Les tableaux
  - Les interfaces
  - Les classes
  - Les modificateurs d’accès
  - Les generics
  - Les Enum

Qu'est-ce que TypeScript ?

TypeScript est un langage Libre (sous licence GNU/GPL) sorti en 2012

- Développé par Microsoft
- Orienté Objet
- Transcompilé (ou "transpilé") en JavaScript avec l'aide de babel
- Accompagné d’un IDE : Visual Studio Code
- Utilisé par le framework Angular depuis Angular 2

Les annotations de type

TypeScript est un langage plus fortement typé que JavaScript.
Il est possible de spécifier pour nos paramètres de fonctions, valeurs de retour et variables des types, par exemple :

```
const myVar: number = 123;

function square(param: number) : number {
        return param * param;
}
```

Les types primitifs

TypeScript comprend 5 types primitifs :

- Les nombres (number)
- Les chaînes de caractères (string)
- Les booléens (boolean)
- void
- null
- undefined

```
let num: number;
let str: string;
let bool: boolean;
```

Les tableaux

Ils sont similaires aux tableaux en JavaScript, à la différence qu'ils peuvent être également sujets aux annotations de type

```
let numberArray: number[];
let numberArray: Array<number>;

numberArray = [1,2,3]; //OK
numberArray = [1,'2',3]; //KO
```

Les interfaces

Les interfaces permettent de définir la structure d'un objet, cette structure est plus ou moins stricte et doit être respectée par tous les tiers l'implémentant.

```
interface Person {
        name: string;
        age: number;
        children?: Object[];
}

let person: Person;

person = {
        name: 'John',
        age: '25'
}
```

Les classes

Les classes de TypeScript sont globalement similaires à celles d'ES6. En revanche TypeScript permet de définir des propriétés en dehors du constructeur d'une classe :

```
class Point {
        x: number;
        y: number;

        constructor(x: number, y: number) {
                this.x = x;
                this.y = y;
        }

}
```

Les modificateurs d'accès

TypeScript permet également d'ajouter des modificateurs d'accès sur les propriétés de nos classes :

- public
- private
- protected

```
class Person {
  private nickname: string;
  protected name: string;
  public age: number;
}
```

Les generics

Tout comme certains autres langages haut-niveau il est possible de définir des generics relatifs aux types de données utilisées dans une fonction :

```
function add<T>(a: T, b: T): number {
        return parseInt(a) + parseInt(b);
}

add<string>('1', '2') // 3;
add<number>(1,2) // 3;
```

Les Enum

Autre nouveauté appréciable en comparaison avec JavaScript, nous pouvons ici définir des enum :

```
enum fd {
        STDIN,
        STDOUT,
        STDERR
}

enum returnCodes {
        ERROR=1,
        OK=0
}
```