Typescript

Lasse Haverinen

Type System over JavaScript

JavaScript

```
let cityObject = {
  name: "London",
  population: 1000000,
  isCapital: true
}

function printCityPopulation(city) {
  console.log(city.name +
  " has a population of "
  + city.population);
}
```

TypeScript

```
interface City {
  name: string;
  population: number;
  isCapital: boolean;
}


let cityTsObject: City = {
  name: "London",
  population: 1000000,
  isCapital: true
}

function printCityPopulationTs(city: City) {
  console.log(city.name +
  " has a population of "
  + city.population);
}
```

Overview

- Adds types for JS
  - Declare variable, declare also the type of data what it is going to use
- TS is on top of JS -> superset of JS -> Everything valid in JS is valid in TS
- Detect error in the code without running it - Static Type Checking

Basic Variable Declaration

```
2   /* Inferring types */
3   let hello = "Hello World";
4   console.log(hello);
5
6   hello = 5;
7   console.log(hello);
8
9   /* Explicit types */
10  let hello2: string = "Hello World";
11  console.log(hello2);
12
13  hello2 = 5;
14  console.log(hello2);
15
16  let hello3: any = "Hello World";
17  console.log(hello3);
18  hello3 = 7;
19
```

- string
  - represents string values like "Hello, world"
- number
  - is for numbers like 42. JavaScript does not have a special runtime value for integers, so there’s no equivalent to int or float - everything is simply number
- boolean
  - is for the two values true and false

Functions

```
1  function calculateSum(a: number, b: number): number {
2      return a + b;
3  }
4
5  let result: number = calculateSum(5, 8);
6  console.log(result);
```

- Parameter declarations similar to variables
- Function return type declaration

Objects

```
let person = {
    name: "John",
    familyName: "Doe",
    age: 30,
    isMarried: false
}

function printPersonAge(person : { name: string, familyName: string, age: number, isMarried: boolean }) {
    console.log(person.name + " " + person.familyName + " is " + person.age + " years old");
}

```

- Above is so called anonymous object type

Objects

```
12  interface Person {
13      name: string;
14      familyName: string;
15      age: number;
16      isMarried: boolean;
17      occupation?: string;
18  }
19
20  function getFullPersonName(person: Person) {
21      return person.name + " " + person.familyName;
22  }
23
```

- Object definition with interface
  - Notice the optional occupation property – annotated with ? Sign
- Function parameter declares

Arrays

```
1   let numbers = [1, 2, 3, 4, 5];
2   let words = ["Hello", "World"];
3
4   function getSecondElement(array : number[]) : number {
5       return array[1];
6   }
7
8   console.log(getSecondElement(numbers));
9   console.log(getSecondElement(words));
```

- Type of data in an array is defined by using the type information and adding [] after it

TypeScript for Node.js

- ts-node
- https://www.npmjs.com/package/ts-node