TYPESCRIPT

LE JAVASCRIPT STATIQUEMENT TYPÉ

DEVOXX™ France

QUI SUIS-JE ?

- Benoit Lemoine
- Développeur full-stack chez Xebia
- @benoit_lemoine
- plus de 2000m sur les pattes avant à Goat Simulator

JAVASCRIPT C'EST BIEN, MAIS...

HISTORIQUE DE TYPESCRIPT

- Made in Microsoft en 2012...
- ... mais open-source et libre (License Apache 2) : https://github.com/Microsoft/TypeScript
- Super-ensemble d'ES5
- Typage statique
- Polyfill pour ES6

LES TYPES - DÉCLARER UN TYPE

```
var name:string = 'Dolan';
var nbLegs:number = 2;

var isMamal = false;

//doesn't compile
//number is not assignable to type boolean
isMamal = 3;
```

LES TYPES - LES GÉNÉRIQUES

```
var featherColors:Array<string> = ['green', 'red', 'grey'];


var lengthOfClors: Array<number> = featherColors.map(function(colo
 return color.length;
}); // [5, 3, 4]
```

LES TYPES - LES CLASSES

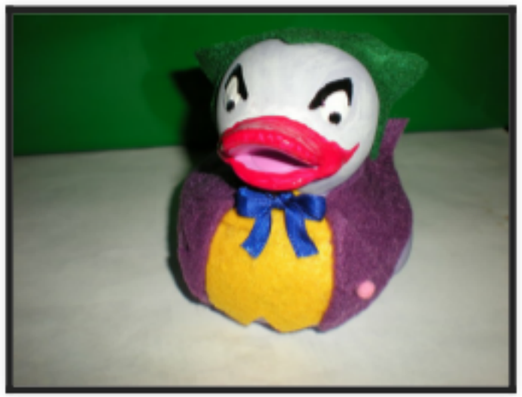
```
class Animal {
  constructor(public name) { }
}

class Duck extends Animal {
  quack() {
    return "quack";
  }
}

class Platypus extends Animal {}

var dolan:Duck = new Duck("Dolan");
console.log(dolan.quack());
```

LES TYPES - TYPAGE GRADUEL



```
var scrooge = new Duck('Scrooge');
var perry = new Platypus('Perry');
//Doesn't compile Platypus is not assignable to type Duck
scrooge = perry;

//we can assign anything to any
var jokerDuck:any = perry;

//We can assign any to everything
scrooge = jokerDuck;
```

LES TYPES - LES INTERFACES

```
interface Quacker {
  name:string
  quack():string;
}

class Goose extends Animal implements Quacker {
  quack() {
    return "honk";
  }
}
var daffie:Quacker = new Goose("daffie");
```

LES TYPES - LE TYPAGE STRUCTUREL

```
interface Quacker {
  name:string
  quack():string;
}

var chicken:Quacker = {
  name:'Chicken',
  quack: function() {
    return 'cluck cluck';
  }
};
```

LES TYPES - UNION TYPE

```
var perry = new Platypus('Perry');
var donald = new Duck('Donald');

var animals:Array<Animal> = [perry, donald];
var duckOrPlatypus:Array<Duck|Platypus> = [perry, donald];
```

ECMASCRIPT 2015 - ECMASCRIPT 6

```
//TypeScript
let lordify = function(names = []) {


  return names
    .map(name => `sir ${name}`);
}
```

```
//JavaScript
let lordify = function(names) {
  if (names === void 0) {
    names = [];
  }
  return names
    .map(function (name) {
      return ("sir " + name);
    });
}
```

LES MODULES INTERNES

```
//Fichier Animal.ts
module Animal {
 var privateVar = "test";
}


//Fichier Duck.ts
/// <reference path="Animal.ts" />
module Animal {
    export class Duck {}
}




new Animal.Duck();
```

```
//Fichier Animal.ts compilé
var Animal;
(function (Animal) {
    var privateVar = "test";
})(Animal || (Animal = {}));

//Fichier Duck.ts compilé
var Animal;
(function (Animal) {
    var Duck = (function () {
        function Duck() { }
        return Duck;
    })();
    Animal.Duck = Duck;
})(Animal || (Animal = {}));
new Animal.Duck();
```

DEVOXX France

LES MODULES EXTERNES AMD OU COMMON JS

```
//Fichier Dolan.ts
import Duck = require('Duck');



export var dolan = new Duck('Dolan')
```

```
//Dolan.ts compilé en AMD
define(
 ["require", "exports", 'Duck'],
 function (require, exports, Duck) {
    exports.dolan = new Duck('Dolan'
 }
);
```

```
//Dolan.ts compilé en CommonJS
var Duck = require('Duck');

module.exports = new Duck('Dolan')
```

DEVOXX™ France

LES DÉCLARATIONS D'AMBIANCE

```
declare var _;

_.filter([1,2,3], (i) => i%2 === 0);
```

DEFINITELY TYPED

http://definitelytyped.org/

```
/// <reference path="lodash/lodash.d.ts" />

_.filter([1, 2, 3, 4], (i) => i%2 === 0) // [2,4]
_.filter([1, 2, 3, 4], (i:string) => i%2 === 0) //ne compile pas
```

TSD

http://definitelytyped.org/tsd/

- Gestionnaire de dépendances pour les fichiers de définition
- Pas de gestion de version (mais ça arrive...)

EXEMPLE ANGULAR - AVANT

```
angular.module('MyCtrl', ['myService','$scope',
  function(myService, $scope) {

    $scope.myValue = myService.maValue;

    $scope.changeValue = function() {
        $scope.myValue = Math.random();
    };
  }
])

//template
<div ng-controller="MyCtrl">
 {{myValue}}
</div>
```

EXEMPLE ANGULAR - APRÈS

```
class MyCtrl {
    myValue:number;

    static $inject = ['myService'];
    constructor(myService:MyService) {
     this.myValue = myService.maValue;
    }
    changeValue() {
        this.myValue = Math.random();
    }
}
angular.module(MyCtrl.name, MyCtrl);
//template
<div ng-controller="MyCtrl as myCtrl">
 {{myCtrl.myValue}}
</div>
```

UTILISER TYPESCRIPT

- Grunt, Gulp
- Plugin pour Play, Wro4j, Grails, etc.
- à la main : compilateur en JavaScript

TYPESCRIPT 1.5 (?) - RTTI

```typescript
//TypeScript


function firstChar(word : string) {
  return word.charAt(0);
}


var x : any = 3;
displayFirstCharacter(x);
```

```javascript
//JavaScript
import * as rtts from 'rtts';

function firstChar(word) {
  rtts.types(word, rtts.string);
  return word.charAt(0);
}

var x = 3;
displayFirstCharacter(x);
```

TYPESCRIPT 1.5 - DECORATORS

```
@annotation
class MyClass { }

function annotation(target) {
   // Add a property on target
   target.annotated = true;
}
```

```
var MyClass = annotation(function ()
    function MyClass() {
    }
    return MyClass;
})();

function annotation(target) {
    // Add a property on target
    target.annotated = true;
}
```

TYPESCRIPT 1.5 - ES6

```
var [m, d, y] = [3, 14, 1977];
console.log(d) // 14

for (let word of ["one", "two", "three"])
  console.log(word);  //affiche "one","two","three"

function* idMaker() {
    var index = 0;
    while(true)
        yield index++;
}

var gen = idMaker();

console.log(gen.next().value); // 0
console.log(gen.next().value); // 1
```

TYPESCRIPT 1.5 - MODULE ES6

```
// math.ts
export function add(x, y) { return x + y }
export function subtract(x, y) { return x – y }
export default function multiply(x, y) { return x * y }


// myFile.ts
import {add, subtract} from "math";
import times from "math";
var result = times(add(2, 3), subtract(5, 3));
```

TYPESCRIPT 2.0

- ES6 (template String++, etc.)
- ES7 (async/await)

LES LIMITES

- Un précompileur de plus
- La vitesse de compilation
- Les fichiers de définition
- Pas compatible avec JSX

CONCLUSION

- Migration Progressive
- Facilité d'apprentissage
- Compatibilité avec ES6
- Refactoring simplifié
- Assistance de l'IDE améliorée

DEVOXX™ France

QUESTIONS ?