# ES6++

Jéremie Amsellem <lp1.eu>

# I - La déstructuration

Il est possible, d'assigner d'un seul coup plusieurs variables à partir de valeurs d'un tableau ou d'un objet.

On appelle cette opération de la déstructuration.

La syntaxe est la suivante avec un tableau :

```
let [firstName, lastName, , location] = [ 'Bob', 'TestMan',
```

Et avec un objet :

```
let {firstName, lastName} = {firstName: 'Bob', lastName: 'Te
```

# II - Les Classes

Une des plus grandes nouveautés d'ES6 est la possibilité d'instancier des Classes dans le langage.

On utilise le mot-clé class pour en créer une nouvelle :

```
class Person {
}
```

Ainsi que le mot-clé new pour l'instancier

```
const p = new Person()
```

# Les Classes ES6 - Le constructeur

Pour modifier le constructeur de notre classe pour y ajouter de la logique ou des paramètres on utilise le nom de méthode **constructor** :

```
class Person {
  constructor(name) {
    this.name = name
  }
}
```

Notez que le mot-clé *this* s'utilise comme dans n'importe quel autre langage en ES6.

# Les Classes ES6 - Les méthodes et variables de classe

On ne définit pas de variables de classe dans le corps de notre classe.

Elles sont définies dans le constructeur en utilisant la syntaxe :

```
class Person {
  constructor(name, age) {
    this.name = name
    this.age = age
  }
  getBirthYear() {
    return 2019 - this.age
  }
}
```

De plus, l'ajout de méthodes se fait sans les séparer par une virgule dans une classe JavaScript !

# Les Classes ES6 - get et set

On peut également définir des get et set dans des classes ES6, par exemple :

```
class Person {
  constructor(name, age) {
    this.name = name
    this.age = age
  }
  get birthYear() {
    return 2019 - this.age
  }
  set name(newName) {
    this.name = newName.toUpperCase()
  }
}

const person = new Person('Bob', 42)
person.name = 'Robert'
console.log(p.birthYear) /* 1977 */
console.log(p.name) /* ROBERT */
```

# Les Classes ES6 - L'héritage

En ES6 il n'y a pas d'héritage multiple, on hérite d'une seule classe parente.

La syntaxe pour l'héritage est la même que dans beaucoup d'autres langages, on utilise le mot clé 'extends' suivi du nom de la classe parente :

```
class Person {
  constructor(name, age) {
    this.name = name
    this.age = age
  }
}

class Dev extends Person {
}

const dev = new Dev('Bob', 42)
```

# Les Classes ES6 - Super

Lorsqu'on hérite d'une classe il est possible d'appeler dans la méthode d'une classe fille,
la méthode correspondant dans la classe parent.

Pour ce faire on utilise le mot-clé **super** :

## Par exemple :

```
class Person {
  constructor(name, age) {
    this.name = name
    this.age = age
  }
}

class Dev extends Person {
  constructor(name, age, language) {
    super(name, age)
    this.language = language
  }
}
```