ECMAScript 2015

Conclusion

Ce qu’on a vu…

SOAT

Les changements qui vont impacter votre code :

Modules

```
export function myFunction()
import {myFunction} from 'module'
```

Classes

```
class MyClass extends Parent {}
```

Promesses

```
var prom = new Promise(function(resolve, reject)) {}
prom.then(otherPromise);
```

Générateurs

```
function* myGenerator {
    yield 'value';
}
```

Les changements qui vont vous simplifier la vie :

String templates

```
`hello ${world}`
```

Destructuring

```
var [val1,val2] = ['value1','value2'];
var {prop1,prop2} = {prop1:'value1', prop2: 'value2'};
```

Paramètres par défaut

```
function myFunction(param1, param2 = 'value') {}
```

Fat arrow functions

```
(param) => { expr; }
```

SOAT

Les changements qui vont vous simplifier la vie :

Block scoping

```
let scopedVar = "value";
const scopedConst = "value";
```

Paramètres Rest

```
function myfunction(param1, ...params) {}
```

Spreading

```
...['value1','value2']
```

Le futur : les propositions les + en populaires

Async/Await

- Equivalent aux coroutines directement dans le langage

```
async function asyncFun () {
  var value = await Promise
    .resolve(1)
    .then(x => x * 3)
    .then(x => x + 5)
    .then(x => x / 2);
  return value;
}
asyncFun().then(x => console.log(`x: ${x}`));
```

Classes

- Attributes
- Private attributes

```
class Point {
  #x = 0;
  #y = 0;

  doSomething() {
    this.#x++; // 1
    this.#y--; // -1
  }
}
```

Décorateurs

- Représenté sous forme d'une annotation
- Un décorateur est une fonction prenant en paramètre une cible et ses méta-données (rappelez vous la meta programmation avec les proxies)
- Fonctionne sur les classes et ses propriétés (attributs/méthodes)

```
function myDecorator(target) {
  target.decorated = true;
}

@myDecorator
class Decorated {}

let obj = new Decorated();
obj.decorated; // true
```

Stratégies d’adoption

Faites le progressivement

- Ajouter Babel/webpack à votre projet
- Utilisez es2015 dans le nouveau code
- Lorsque vous modifiez le code existant introduisez les patterns ES2015 que nous avons vu

- Commencez pas des petits projets pour vous faire la main
- Migrez vos librairies internes progressivement et utilisez des outils pour les exposer en UMD
- Faites tous vos nouveaux projets en ES2015
- Ajoutez des questions sur le sujet dans votre recrutement

Aller plus loin

Les liens utiles

- Le livre indispensable : http://exploringjs.com/
  - Gratuit en ligne, payant en epub/pdf
- Le blog http://www.2ality.com/
- Suivre le standard : https://ecmascript-daily.github.io/
- Les features et leur stage : https://github.com/tc39/ecma262

Feedback et ROTI