

## React

### 1. Rappels

## 2. Syntaxes ES6+

- 3. Premiers pas avec React
- 4. Les composants
- 5. Formulaires et AJAX
- 6. Hooks & React Router

## 2. SYNTAXES ES6+

- Les bases
- POO
- Modules

## ES6: LET

- remplace "var"
- scopée

```
let i = 0;
for ( let i = 1; i < tableau.length; i++ ) {
    console.log( i ); // 1, 2, 3, ...
}
console.log( i ); // 0</pre>
```

#### Notes:

Fonctionne exactement comme le var traditionnellement employé. Cependant, le let a la particularité d'être scopé au niveau du bloc de code dans lequel il est déclaré, et pas forcément dans toute la fonction.

## ES6: CONST

- déclaration de constante
- scopée

```
const hello; // Erreur : une constante doit avoir une valeur !
const hello = 'Hello';

if (true) {
   const hello = 'Bonjour';
   hello = 'Hola'; // Erreur : une constante ne peut être modifiée !
   console.log(hello); // "Bonjour"
}

console.log(hello); // "Hello"
```

#### Notes:

const permet de déclarer une constante. De plus, comme le let, une variable const est scopée au niveau du bloc de code et non pas de la fonction comme le var.

## ES6: CONST?

## ES6: TEMPLATES STRINGS

- Nouveau délimiteur de chaîne : `
- Injection d'expressions (variable ou appel de fonction)
- multiline

```
const value = `danger`;
let phrase = `I am the ${ value } !`; // I am the danger !
function myFunction(valeur) {
    return valeur;
}
phrase = `I am the ${ myFunction('danger') } !`; // I am the danger !
```

```
function renderLink( target, cssClass, href ) {
    return `<a href="${ href.toLowerCase() }"</pre>
               class="${ cssClass }"
               target="${ target }">
                     Cliquez ici !
            </a>`; // multiline !
```

## ES6: DESTRUCTURING

- déclarer des variables au nom d'une propriété d'objet
- ... ou qui correspondent aux valeurs d'un tableau

```
const personnage = { prenom: 'Skyler', nom: 'White' };
// const prenom = personnage.prenom, nom = personnage.nom;
const { prenom, nom } = personnage;
console.log(`Salut ${prenom} ${nom} !`);

function maFonction( {prenom, nom} ) {
    return `Salut ${prenom} ${nom} !`;
}
maFonction( personnage );

const tableau = [ 'Walter Jr', 'White' ];
// const prenom = tableau[0], nom = tableau[1];
const [ prenom, nom ] = tableau;
```

#### Notes:

Le destructuring permet de faciliter la création et l'assignation de variables à partir issues d'objets ou de tableaux.

http://codepen.io/kumquats/pen/dXWxzW?editors=0011

## **ES6: ARROW FUNCTIONS**

- déclaration de fonction simplifiée
- return implicite
- scope préservé

```
// Fonction anonyme en ES5
var add = function( a, b ) {
   return a + b;
}

// et en ES6 (opérateur "fat arrow")
const add = ( a, b ) => a + b;
const square = x => x * x;
   // si un seul paramètre, pas besoin de parenthèses
```

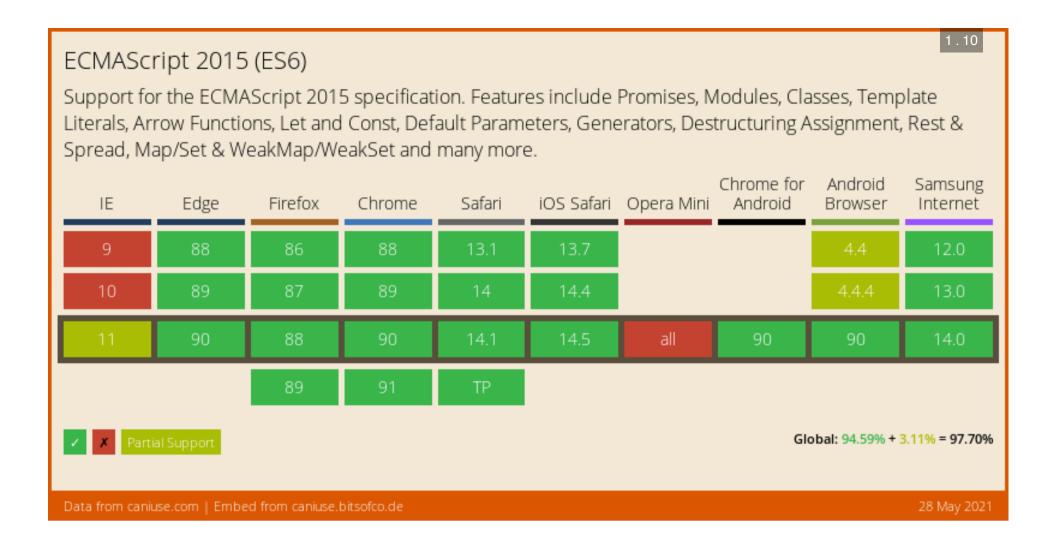
#### Notes:

La valeur de this dans une arrow function est toujours celui dans lequel elle est déclarée :

cf. https://codepen.io/uidlt/pen/wvGLwmL?editors=1001

#### 1.9

# PROBLÈME: SUPPORT NAVIGATEUR

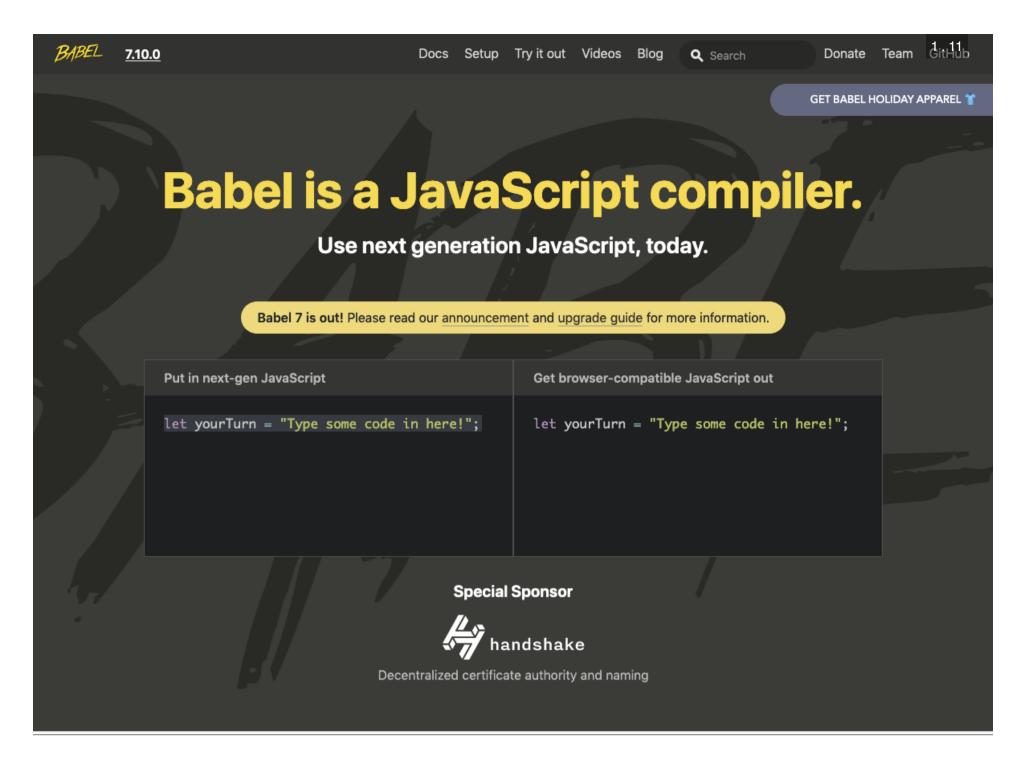


#### Notes:

Le support navigateur est très partiel, notamment sur IE. Il faut noter que chaque feature ES6 dispose de son propre support navigateur de manière indépendante. http://caniuse.com/#search=es6

http://kangax.github.io/compat-table/es6/ Autre tableau détaillé, navigateur par navigateur du support des différentes fonctionnalités ES6.

Pour ES2016 et suivantes voir http://kangax.github.io/compat-table/es2016plus/ et pour les features encore en cours de normalisation : http://kangax.github.io/compat-table/esnext/



#### Notes:

BabelJS https://babeljs.io/

Babel est un compilateur de code ES6+ en ES5.

Cet outil permet de rendre notre application JS compatible avec les vieux navigateurs.

Le principe est le suivant :

- on code de manière élégante à l'aide des dernières features du langage (ES6+)
- Babel compile le tout en ES5 (compatible avec les anciens navigateurs),
- c'est ce fichier compilé qui sera mis en ligne et servi aux visiteurs

```
GitH2b
                                                         Docs Setup Try it out Videos Blog
                                                                                                                     Donate Team
                                                                                                 Q Search
2 let square = x \Rightarrow x * x;
                                                                        4 var square = function square(x) {
5 const i = square(42);
                                                                        5 return x * x;
6 //i = 12; //erreur de compilation (constante)
                                                                        6 }; // const
8 // let scopé
9 for ( let i = 0; i < 12 ; i++ ){
                                                                        9 var i = square(42); //i = 12; //erreur de compilation (constante)
   console.log(i);
    if (true){
                                                                       12 for (var _i = 0; _i < 12; _{i++}) {
      let i = 5;
                                                                       13 console.log(_i);
      console.log(i);
17 // destructuring
                                                                              console.log(_i2);
18 o = {
                                                                       19 } // destructuring
 let {a, b} = o;
24 // objet rest spread (stage 3)
25 const params = { lol: 'lol' };
                                                                       25 };
      pouet: 'pouet',
                                                                             a = _0.a,
                                                                              b = _o.b; // objet rest spread (stage 3)
      ...params
29 }
                                                                       30 \text{ var } params = {
                                                                       31 lol: 'lol'
                                                                       32 };
                                                                       33 var fusion = {
                                                                       34 pouet: 'pouet',
                                                                       35 ...params
                                                                       36 };
```

#### Notes:

Babel dispose d'un outil en ligne qui permet de tester la conversion de code ES6+ en ES5. Cet outil a surtout un but pédagogique : montrer comment les syntaxes ES6+ permettent d'avoir un code plus propre et plus lisible que son équivalent en ES5.

Outil de test en ligne du compilateur babel (repl)

babel repl pré-rempli avec des exemples de code ES6