

Développer avec React

Create-React-App

Développer avec React

Create-React-App... C'est quoi?

Facebook fournit l'outil « Create-React-App » qui permet de générer un projet React sans devoir effectuer de configuration.

Celui-ci est disponible sur github : <https://github.com/facebook/create-react-app>

Quels éléments intéressants installé via cet outil :

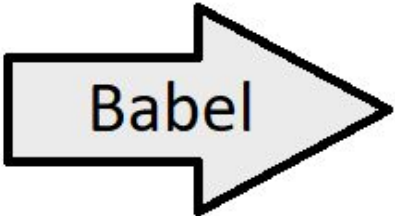
- Le support de la syntaxe ES6, React, JSX.
- Préfixe CSS automatique
- Un serveur de développement en direct
- Script pour générer la "build" de production

Bibliothèques intéressantes incluses

★ Babel

Babel est un **transcompilateur** qui traduit notre code en ES6+ et JSX vers du javascript ES5 pour s'assurer une rétro-compatibilité avec tous les navigateurs Web.

```
[1, 2, 3].map(n => n ** 2);
```



```
[1, 2, 3].map(function(n) {  
  return Math.pow(n, 2);  
});
```

Bibliothèques intéressantes incluses

★ Webpack

Permet de combiner les modules ainsi que leurs dépendances en un seul pack de ressources statiques.

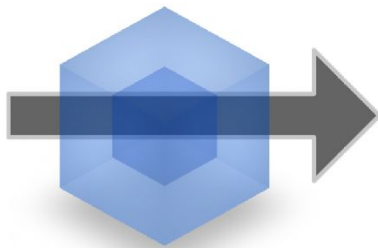
cats.js

```
var cats = ['dave', 'henry', 'martha'];  
module.exports = cats;
```

app.js

```
var cats = require('./cats.js');  
console.log(cats);
```

webpack
MODULE BUNDLER



app.bundle.js

```
!function(r){function t(o){if(n[o])return n[o].exports;  
var e=n[o]={i:o,l:!1,exports:{}};return r[o].call(  
e.exports,e,e.exports,t),e.l=!0,e.exports}var n={};  
return t.m=r,t.c=n,t.p="",t(t.s=1)}([function(r,t){  
var n=["dave","henry","martha"];r.exports=n,function(  
r,t,n){cats=n(0),console.log(cats)}}]);
```

Bibliothèques intéressantes incluses

★ Webpack-dev-server

Créer un serveur de développement basé sur la bibliothèque « Webpack ».

Il nous permettra de :

- Tester notre application en direct.
- Voir les erreurs de syntaxe de notre code.

Création d'un projet React

Développer avec React


Création de la base du projet

Ouvrir le terminal dans un répertoire.

Générer le projet à l'aide de « Create-React-App » avec la commande :

“ npm init react-app <project-name> ”

 Le nom du projet doit être en minuscule et il ne doit pas contenir d'espace !



```
ca: Administrateur : Invite de commandes
Microsoft Windows [version 10.0.16299.611]
(c) 2017 Microsoft Corporation. Tous droits réservés.

C:\WINDOWS\system32>cd c:\React

c:\React>npm init react-app firstapp_
```

Remarque : Le nom peut être remplacé par un point, pour créer le projet dans le répertoire actuel.

Création de la base du projet

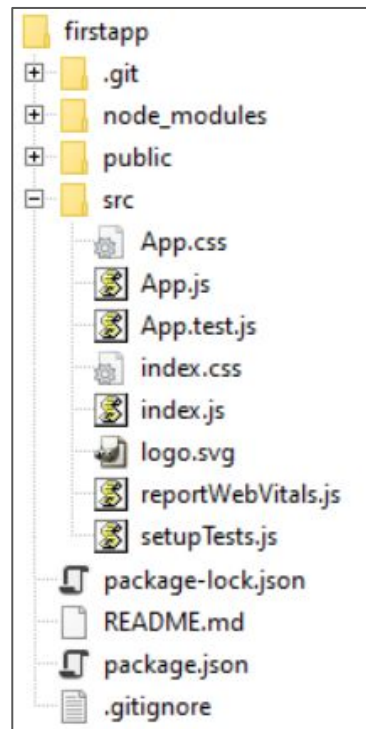
Une fois la génération de code terminée, nous obtenons cette structure.

Pour tester le projet, nous allons démarrer le serveur de développement.

Pour cela, utiliser les commandes :

" cd <my-project> "

" npm start "



Explication des éléments générés

Quelques explications sur les éléments créés par « Create-React-App »

- Le dossier « Public »
Contient le **template HTML** dans lequel l'application React va être insérée.
- Les fichiers « App »
Composant React principal de l'application.
- Les fichiers « Index »
Utilisation du **ReactDOM** pour injecter le composant « App » dans le code HTML.

Structure de fichier

React n'impose pas de structure pour vous ordonnez vos fichiers.

Il est conseillé d'en mettre une en place pour faciliter le développement et la maintenance des projets.

Des solutions populaires existent pour ordonner les fichiers d'un projet.

Par exemple :

- Par types d'éléments (components, store, api, page, ...)
- Par fonctionnalités

Structure de fichier – Choix pour le support

La structure utilisée dans le support est « Par types d'éléments », sous la forme :

- Un dossier par éléments.
- Les éléments seront séparé en catégories :
 - **Components** Composant “*simple*” et réutilisable.
Exemple → Une barre de recherche.
 - **Containers** Composant reprenant les différentes features.
Exemple → Le menu de navigation.

D'autres catégories de fichiers seront ajoutés par la suite (*page, store, api, ...*).

Le but des composants

Développer avec React

Les Composants React

Les composants React vont nous permettre de créer nos interfaces en blocs de code indépendant et réutilisable. Ceux-ci ont un paramètre d'entrée et un rendu.

Il existe deux familles de composant React :

- Les fonctions
 - Basique : limité à la génération de rendu via le JSX
 - Avancé : utilisation du mécanisme des Hooks
- Les classes héritant de « `React.Component` »

⚠ Lors du développement, nous allons créer un fichier par composant React. Pour cela, nous utiliserons les instructions d'import et d'export du JavaScript.

Découpe d'un rendu sous forme de composant

Pour concevoir une application Web en React.

Il est nécessaire de découper le rendu sous forme de différents composants.

Exemple de découpe :

- 1) **FilterableProductTable** : Le composant principal.
- 2) **SearchBar** : Interaction avec la saisie de l'utilisateur.
- 3) **ProductTable** : Affiche et filtre les données.
- 4) **ProductCategory** : Affiche le titre d'une catégorie.
- 5) **ProductRow** : Affiche de détail d'un produit.

