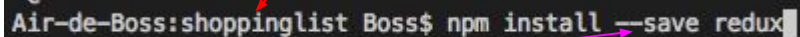


Installation de redux via le terminal

Se mettre au niveau de package.json dans le dossier du programme et installer redux via le terminal :



```
Air-de-Boss:shoppinglist Boss$ npm install --save redux
```

A screenshot of a terminal window with a black background and white text. The prompt is 'Air-de-Boss:shoppinglist Boss\$'. The command entered is 'npm install --save redux'. A red arrow points from the text 'Se mettre au niveau de package.json' above to the 'shoppinglist' part of the prompt. A pink arrow points from the text 'le --save sert à mettre redux dans le package.json' below to the '--save' flag in the command.

le --save sert à mettre redux dans le package.json

On import en destructurant les méthodes createStore() et combineReducers() de Redux.

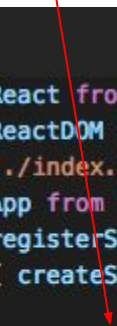
Les accolade servent à la syntaxe de destructuring de l'objet.

Un export simple requiert les accolades.

Un export default n'en requiert pas.

```
JS index.js x
1  import React from 'react';
2  import ReactDOM from 'react-dom';
3  import './index.css';
4  import App from './components/App';
5  import registerServiceWorker from './registerServiceWorker';
6  import { createStore, combineReducers } from 'redux';
7
8  ReactDOM.render(<App />, document.getElementById('root'));
9  registerServiceWorker();
10
```

On génère ensuite un store en appelant la méthode createStore().



```
JS index.js x
1  import React from 'react';
2  import ReactDOM from 'react-dom';
3  import './index.css';
4  import App from './components/App';
5  import registerServiceWorker from './registerServiceWorker';
6  import { createStore, combineReducers } from 'redux';
7
8  const store = createStore();
9
10 ReactDOM.render(<App />, document.getElementById('root'));
11 registerServiceWorker();
12
```

On créer un reducer qui sera le reducer qui gèrera le 'state' de la propriété: articles dans le 'state' de App.js

```
JS index.js •
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4 import App from './components/App';
5 import registerServiceWorker from './registerServiceWorker';
6 import { createStore, combineReducers } from 'redux';
7
8 const store = createStore();
9
10 const articlesReducer = (state = [], action) => {
11
12 }
13
14 ReactDOM.render(<App />, document.getElementById('root'));
15 registerServiceWorker();
16
```

utilisation du paramètre par défaut es6

```
JS App.js x
1 import React, { Component } from 'react';
2
3 import './App.css';
4 import Form from './Form';
5 import ItemList from './ItemList';
6
7 const uuidv1 = require('uuid/v1');
8 class App extends Component {
9
10   state = { articles : [] };
11
12   _addArticle = (article) => {
13     this.setState({ articles : [...this.state.articles, article] });
14     article.id = uuidv1();
15   }
16 }
```

Note importante : Quand un reducer est dispatcher tous les reducer() entre en action.

Ce qui veut dire que l'on va utiliser un switch, et que tous les reducers non concerné qui rentre quand même en action quand un des reducers du store est dispatché retournerons un state inchangé.

Il retournerons par le états inchangé par défaut.

```
const articlesReducer = (state = [], action) => {  
  switch(action.type){  
    default:  
      return state;  
  }  
}
```

Le state de App.js au lancement du programme.

```
class App extends Component {  
  state = { articles : [] };  
}
```

Notre fonction reducer 'articleReducer()' gère l'ajout d'article grâce à l'action 'ADD_ARTICLE'.

On donne au reducer donc 2 arguments (le state courant de l'action choisie, et l'action).

Quand la fonction articleReducer(currentState, action) est appelée l'action 'ADD_ARTICLE' est donc exécuté, et c'est dans ce corps de fonction que nous effectuons le changement de l'état de 'articles' qui nous retournera la part d'état dont la fonction articleReducer() est responsable, c'est à dire un nouvelle état et non une modification d'état (pense à l'immutabilité).

La fonction combineReducers() permet de stocker un reducer dans le store créer,

'articles' est la clé et 'articleReducer' notre fonction reducer.

state est le state de la propriété articles, qui est un morceau d'état également du composant App.

JS index.js x

```
1  import React from 'react';  
2  import ReactDOM from 'react-dom';  
3  import './index.css';  
4  import App from './components/App';  
5  import registerServiceWorker from './registerServiceWorker';  
6  import { createStore, combineReducers } from 'redux';  
7  
8  const articlesReducer = (state = [], action) => {  
9    switch(action.type) {  
10     case 'ADD_ARTICLE':  
11       console.log('ADD-ARTICLE');  
12       console.log('action', action);  
13       action.payload.id = Date.now();  
14       const newState = [...state, action.payload];  
15       return newState;  
16     default:  
17       return state;  
18     }  
19   }  
20  
21   const store = createStore(combineReducers({articles: articlesReducer }));  
22  
23   ReactDOM.render(<App />, document.getElementById('root'));  
24   registerServiceWorker();  
25
```

```
const store = createStore(combineReducers({articles: articleReducer }));
```

L'API de Redux comporte 4 méthode :

- **dispatch()** Permet de dispatcher une action.
- **getState()** Permet de récupérer l'état courant.
- **replaceReducer()** Permet de supporter le hotReloading.
- **subscribe()** Permet a un component container de s'abonner au store afin d'être mis au courant des changement d'état.

Note: Uniquement les smart container c'est à dire les containers component peuvent s'abonner au store, car les dumb components eux ne font que recevoir de la donnée de leur parent via les props et recrache du JSX pour une vue qu'il leur import pas.

Il faut brancher nos Smart Component au store de Redux avec le package “React Redux” ce qui permet de faire le lien entre une application React et un store Redux, pour permettre à nos Smart Component que Redux est disponible et que des action attende d’être utiliser.

Installation

React Redux requires **React 0.14 or later**.

```
npm install --save react-redux
```

Note: Penser a installer [ReduxDevtools](#) pour chrome.