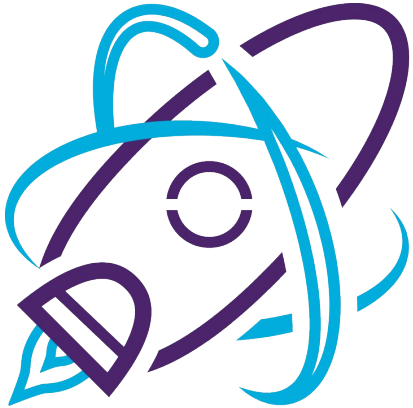


ReactIT

Formation & Recrutement
www.react-it.fr

Formation : React

ReactIT

Votre formateur

- **Loïc Guillois**
- Développeur Web full stack depuis plus de 10 ans
- Expérience en environnement
  - SSII
  - Grand comptes Banque / Assurance / La Poste
  - Startup (Gamific.TV, La Fourchette.com, Akeneo)
  - Indépendant / freelance
- Enseignant / formateur depuis 5 ans
  - Développement
  - No SQL
  - Devops

ReactIT

Programme AJC

- React (5 jours)
- Projet React (2 jours)

Objectifs pédagogiques

- A l'issue de cette formation, vous serez en mesure de :
  - Maîtriser les concepts et la syntaxe de React
  - Créer des applications web performantes
  - Organiser, modulariser et tester ses développements JavaScript

Introduction à ES6

- ES6 a.k.a. ECMA Script 2015
- Nouvelle politique de version
  - 2016: ES7
  - 2017: ES8
  - ...
  - 2020 : ES11
- ES6 a apporté énormément de changement
- Les versions suivantes sont plus incrémentale

Introduction à ES6

- ES6 a.k.a. ECMAScript 2015
- Nouvelle politique de version
  - 2016: ES7
  - 2017: ES8
  - ...
  - 2020 : ES11
- ES6 a apporté énormément de changement
- Les versions suivantes sont plus incrémentale

Introduction à ES6

- Support par les navigateurs (ou pas)

https://kangax.github.io/compat-table/esnext/

https://developer.mozilla.org/en-US/docs/Archive/Web/JavaScript/ECMAScript_Next_support_in_Mozilla

Introduction à ES6

Transcompilateur (exemple : Babel.js)

Permet de transformer un code source ES6/ES7/ES8 vers une version précédente.

Introduction à ES6

Polyfill a.k.a shim

« prothèse » d'émulation permettant d'implémenter un mot clef ou une syntaxe en javascript

Deux principales solutions:

1. Core JS : https://github.com/zloirock/core-js
2. Polyfill.io : https://polyfill.io/v3/

Introduction à ES6

Template string

- Aussi appelé en français « littéraux »
- Permet d'éviter l'usage de l'opérateur + pour la concaténation pour un gain en lisibilité

```
`texte ${expression} texte`
```

Introduction à ES6

Déclaration de variables

- On n'utiliser plus le mot clef `var`
- Le mot clef `let` permet de déclarer une variable visible uniquement dans le bloc courant

```
let x = 1
```

Introduction à ES6

Déclaration de constantes

- Permet de fger la référence
- Une erreur sera produite si on tente de modifier val

```
const val = 42
```

Attention, une constante n'est pas immuable. Si la constante contient un objet, celui ci pourra voir ses propriétés modifiées...

Introduction à ES6

Arrow function (fonction fléchées)

- Syntaxe plus courte
- Ne possède pas de scope (pas de this, arguments, super...)
- Utilisées souvent pour les callbacks

```
setInterval(() => {

    console.log("coucou")

}, 1000);
```

Introduction à ES6

Programmation fonctionnelle

- On évite d'utiliser les boucles (`for, while`) pour parcourir des tableaux
- Nombreuses possibilité : map, forEach, filter, sort...

```
map.forEach((value, key) => {

    console.log(key, value);

})
```

Introduction à ES6

Object.assign

- Permet de copier toutes les valeurs d'un objet dans un autre
- Pratique pour créer une copie et l'enrichir sans modifier l'objet d'origine
- Facilite l'implémentation de fonctions pures

```
const object1 = {
    a: 1,
    b: 2,
    c: 3
};
const object2 = Object.assign({}, object1)
```

Introduction à ES6

Parmi les nouveautés « intéressantes » on retrouve aussi

- Possibilité de créer des classe et héritage
- Paramètres par défaut pour les fonctions
- Paramètres illimités pour les appels de fonctions
- Déstructuration d'objet
- For..of
- Générateurs

Introduction à ES6

Des outils

Pour vous aider à apprendre le langage et à le respecter. La plupart des environnement de développement propose un “Linter”

ESLint vous permettra de vérifier la qualité de votre code.

Introduction à ES6

Conclusion

- Ce n’est pas un nouveau langage
- ES6 est rétrocompatible
- Les polyfills permettent le support des dernières évolutions du langage dans tous les navigateurs

Mise en pratique ES6

https://github.com/kentcdodds/es6-workshop

Faciliter la création d'application single page

- Caractéristiques :
  - ReactJS permet de fabriquer des composants Web
  - Un composant ReactJS génère du code HTML à chaque changement d'état
  - ReactJS ne gère que la partie interface de l'application web (Vue)
  - ReactJS peut être utilisé avec une autres bibliothèque ou un framework (AngularJS)
  - ReactJS s'appui beaucoup sur ES6 et désormais ES7

Langage JSX

- Les composants sont écrits avec le langage JSX :
  - Un composant possède un état (state)
  - Un composant défini la méthode render()
  - Un composant accepte des paramètres appelés propriétés (props)
    - une propriété peut être un objet javascript, un tableau, un type primitif
    - une propriété peut être une fonction (utile pour communiquer entre composant : événements : onClick, onValidate, onSubmit…)
  - Un composant est monté sur un noeud présent dans le virtual DOM

Exemple

```
import React from 'react';
import PropTypes from 'prop-types';

export default class Stars extends React.PureComponent {

  state = {
    starArray: []
  }

  static propTypes = {
    value: PropTypes.number.isRequired,
    style: PropTypes.any
  }

  constructor(props) {
...
  }

  render() {
...
      );
  }
}
```

Exemple

```
constructor(props) {
    super(props);
    if (props.value !== null) {
        this.state.starArray = new Array(MAX_STARS)
        .fill('star', 0, props.value)
        .fill('star_empty', props.value, MAX_STARS);
    }
}
```

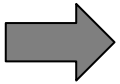
Exemple

```
render() {
    return (
        <span className="Stars" style={this.props.style}>
            {
                this.state.starArray.map((star, index) =>
                    <span
                        key={index}
                        className={star === 'star_empty' ? 'far fa-star' : 'fas fa-star'}
                        style={{ width: '18px' }}
                    />
                )
            }
        </span>
    );
}
```

Babel

- Les scripts JSX doivent être compilés en JavaScript

```
const HelloMessage = React.createClass({
  render: function () {
    return <div>Hello {this.props.name}</div>;
  }
});
ReactDOM.render(<HelloMessage name="John" />, app);
```



```
const HelloMessage = React.createClass({
  displayName: "HelloMessage",
  render: function render() {
    return React.createElement("div", null, "Hello ", this.props.name);
  }
});
ReactDOM.render(React.createElement(HelloMessage, { name: "John" }), app);
```

create react app

- Create react app est un outil de compilation React

```
npm install -g create-react-app
```

```
create-react-app --version
```

create react app

- Créer un projet

```
create-react-app hello-world
```

La structure de votre projet sera automatiquement généré avec tests unitaires et toutes les dépendances nécessaires et à jour.

create react app

- Lancer un projet

```
npm start
```

La commande start lance un serveur Web local avec les fichiers JSX compilés et le hot reload activé. Par défaut, sur le port 3000.

create react app

- Déployer une application

```
npm run test
```

```
npm run build
```

Après avoir configuré le nom de domaine dans le fichier package.json, il n'y a plus qu'à lancer le build. Celui ci s'occupera de la minification des fichiers javascripts et CSS.

Bibliothèques externe

- Il est facile d'utiliser les dépendances npm

```
npm install --save moment
```

```
import moment from 'moment';
```

CSS par composant

- Chaque composant peut avoir sa propre feuille de style, sans effet de bord :

```
import './dashboard.css';
```

Composant complexe

- Un composant complexe peut s'appuyer sur plusieurs composant

```
import Graph from './Graph';
import Table from './Table';

export default class Dashboard extends React.Component {
...
}
```

Composant complexe

- Un composant complexe peut s'appuyer sur plusieurs composant

```
<div className="row">
  <div className="col-md-6">
    <Table codeFinanceur={this.props.codeFinanceur} codeRegion={this.props.codeRegion} changeType={this.onChangeType}
type={this.state.type} year={this.state.year} />
  </div>
  <div className="col-md-6">
    <Graph codeFinanceur={this.props.codeFinanceur} codeRegion={this.props.codeRegion} type={this.state.type}
year={this.state.year} />
  </div>
</div>
```

Les événements

- Les événements Javascript sont faciles à exploiter

```
const changeType = type => {
...
}
```

```
<td onClick={this.changeType.bind(this, content.type)}>{content.label}</td>
```

Les requêtes Ajax

- Les requêtes sont possibles par exemple avec fetch ou axios (hors ReactJS)

```
const baseUrl = process.env.REACT_APP_API_URL;

export const login = (user) => {
  return fetch(baseUrl+'/login', {
      method: 'POST',
      headers: {
         'Accept' : 'application/json',
         'Content-Type': 'application/json'
      },
      body: JSON.stringify(user)
  }).then(res => res.json())
}
```

React Router

Bibliothèque de routage populaire et complète pour React.js qui maintient l'interface utilisateur synchronisée avec l'URL.

Cela permet l’utilisation des favoris et le partage d’URL entre utilisateurs mais également l’accès à l’historique de navigation (précédent, suivant).

React Router

Pour l'utiliser nous installerons le module suivant :

npm install --save react-router-dom

React Router

Pour la mise en place :

import { BrowserRouter as Router, Redirect, Route, Switch } from 'react-router-dom';

et on englobe l'application dans le composant <Router>

React Router

Pour la mise en place :

import { BrowserRouter as Router, Redirect, Route, Switch } from 'react-router-dom';

et on englobe l'application dans le composant <Router>

React Router

On peut mettre en place une redirection :

```
<Switch>
    <Redirect exact from="/" to="/admin" />
</Switch>
```

React Router

Associer une route à un composant :

```
<Route
  path="/mon-compte"
  render={props => (<MyAccount {...props} />)} />
```

React Router

Associer une route à un composant :

```
<Route
        path="/admin/moderation/stagiaires/:filter/:page?"
        render={props => {
            let parameters = {
                filter: props.match.params.filter,
                page: props.match.params.page,
            };
```

React Router

Associer une route à un composant :

```
            return <StagiairesPanel

                                onChange={params => {

                                    let stagiaire = params.stagiaire ?
`?stagiaire=${params.stagiaire}` : '';


props.history.push(`/admin/moderation/stagiaires/${params.filter}/${params.page ||
1}${stagiaire}`);

                                }} />;

                    }} />
```

Prise en main de React

Objectifs

- Mise en place de votre environnement (ESLint, extensions React dans votre IDE et votre navigateur)
- Installation de create-react-app
- Création de votre première application
- Créer vos premiers composants:
  - Login
  - Homepage
- Système de router (/, /home)

Restons en contact

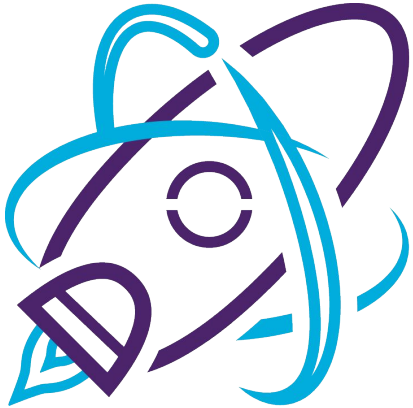
Votre contact:

- React IT
  - Loïc Guillois, président
  - hello@react-it.fr

www.react-it.fr

JUNIOR JOBS

Découvrez aussi http://junior.jobs



ReactIT

Formation & Recrutement
www.react-it.fr

Formation : React