

LE FRAMEWORK **REACT**



ANTHONY DI PERSIO

DECOUVERTE DE REACT

Comprendre le Framework React et
rappel des bases Javascript

01

INSTALLATION DE REACT

Outils nécessaires à l'utilisation
du Framework React

02

CRÉER UNE APPLICATION

Intégration de React dans un projet
et découverte de la CLI

03

STRUCTURE APPLICATION

Analyse d'une application React

04

TABLE DES MATIÈRES

05

COMPOSANTS

Les composants d'une application
React

06

CYCLES DE VIE

Cycles de vie dans le Framework
React

07

REACT-ROUTER

Les routes dans le Framework
React

08

STORE AVEC REDUX

Utilisation d'un store dans le
Framework React

01

DECOUVERTE DE REACT

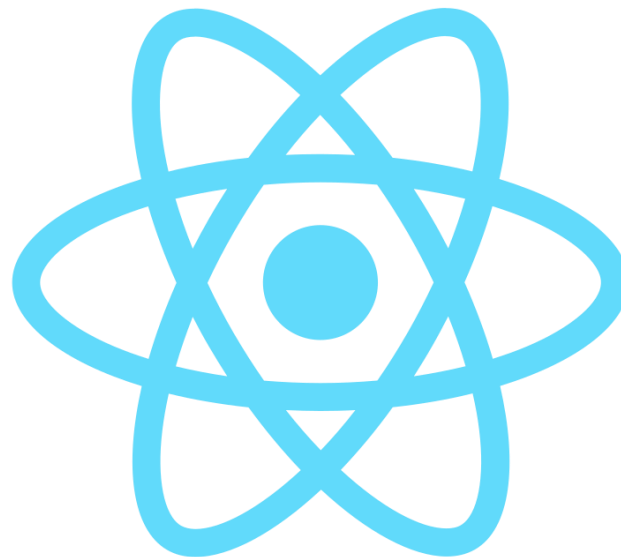
Comprendre le Framework React et rappel des bases Javascript

DECOUVERTE DE REACT

- React est un framework JavaScript
- Naissance en 2013
- Développé et maintenu par Facebook
- Le framework s'oriente très nettement sur la création d'interfaces utilisateurs

DECOUVERTE DE REACT

- Il s'impose aujourd'hui comme le framework de référence avec Angular
- Mais sans oublier Vue.js ...



DECOUVERTE DE REACT

- Trouver des sites ou des applications web qui utilise React est très simple aujourd'hui
- La réciproque n'est pas forcément évidente pour Angular ou Vue.js
- Vous utilisez au quotidien des sites ou applications qui utilisent React !
- Facebook bien entendu.. mais pas seulement

DECOUVERTE DE REACT



COMMENT SAVOIR SI UN SITE / APPLICATION UTILISE REACT ?



React Developer Tools

Proposé par : Facebook

★★★★★ 1122

[Outils de développement](#)

👤 1677 884 utilisateurs

DECOUVERTE DE REACT

- Contrairement à d'autres Framework il y a très peu de code spécifique à React
- Globalement que du JavaScript avec des notions et du vocabulaire propres à React
- Une bonne connaissance du JS vous aidera à appréhender rapidement le Framework React...

L'ESSENCE DE REACT

- Modularité
- Réutilisabilité du code
- Basé sur des « Component »
- Utilisable partout et par tous

POURQUOI UTILISER REACT

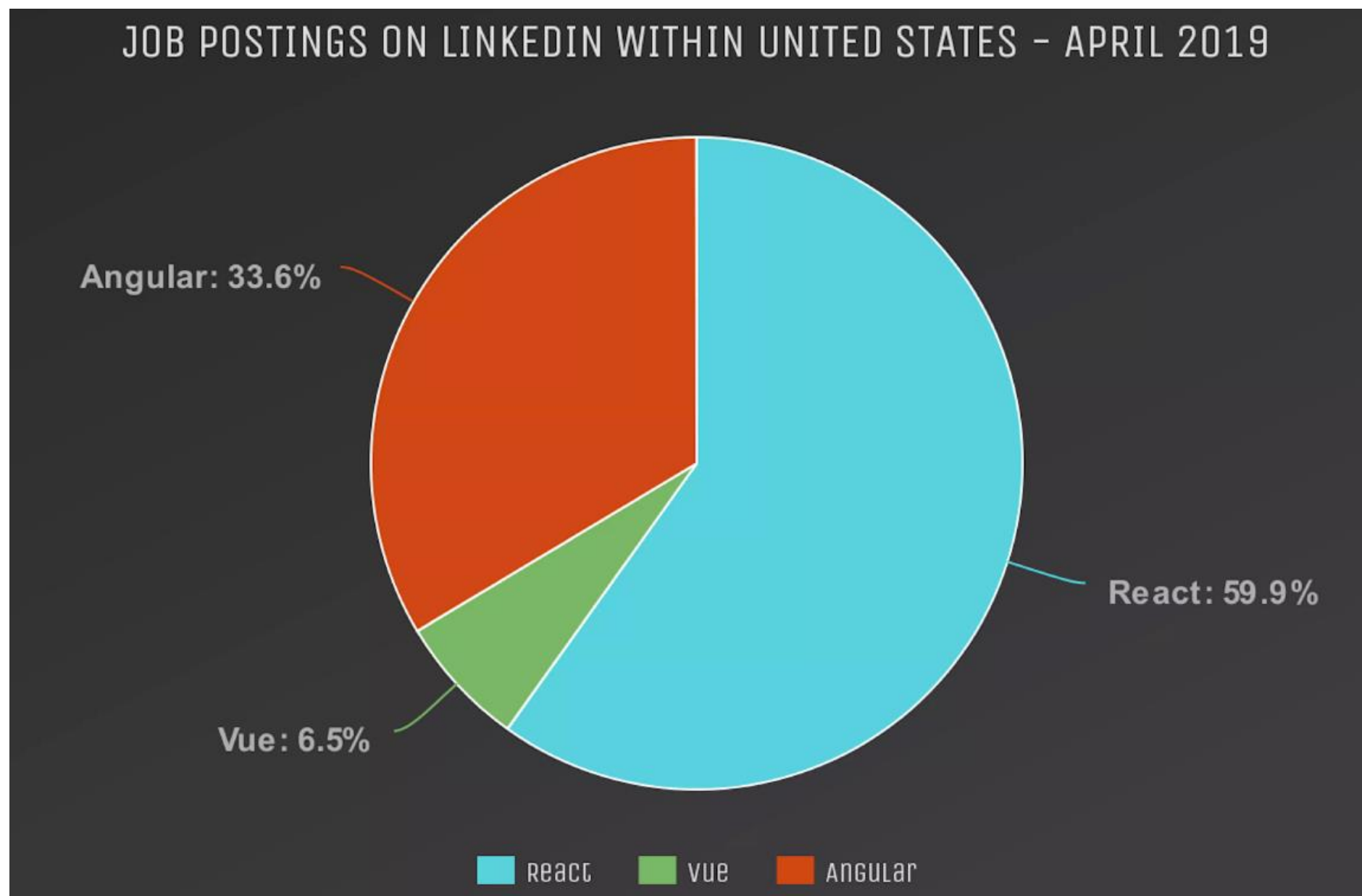
- Courbe d'apprentissage rapide
- Components réutilisable
- Rapidité d'interactivité (grâce au Virtual DOM)
- Bons outils de développement
- Grosse communauté
- React Native

POURQUOI UTILISER REACT

- Virtual DOM ? 🤖
 - La manipulation du DOM comme nous avons l'habitude de faire s'avère lourde et coûteuse
 - Le Virtual DOM ne va modifier dans le DOM uniquement ce qui a réellement été modifié
 - Le gain de performance et de réactivité est indéniable

POURQUOI UTILISER REACT

- Et aussi du boulot ..



RAPPELS EN JAVASCRIPT

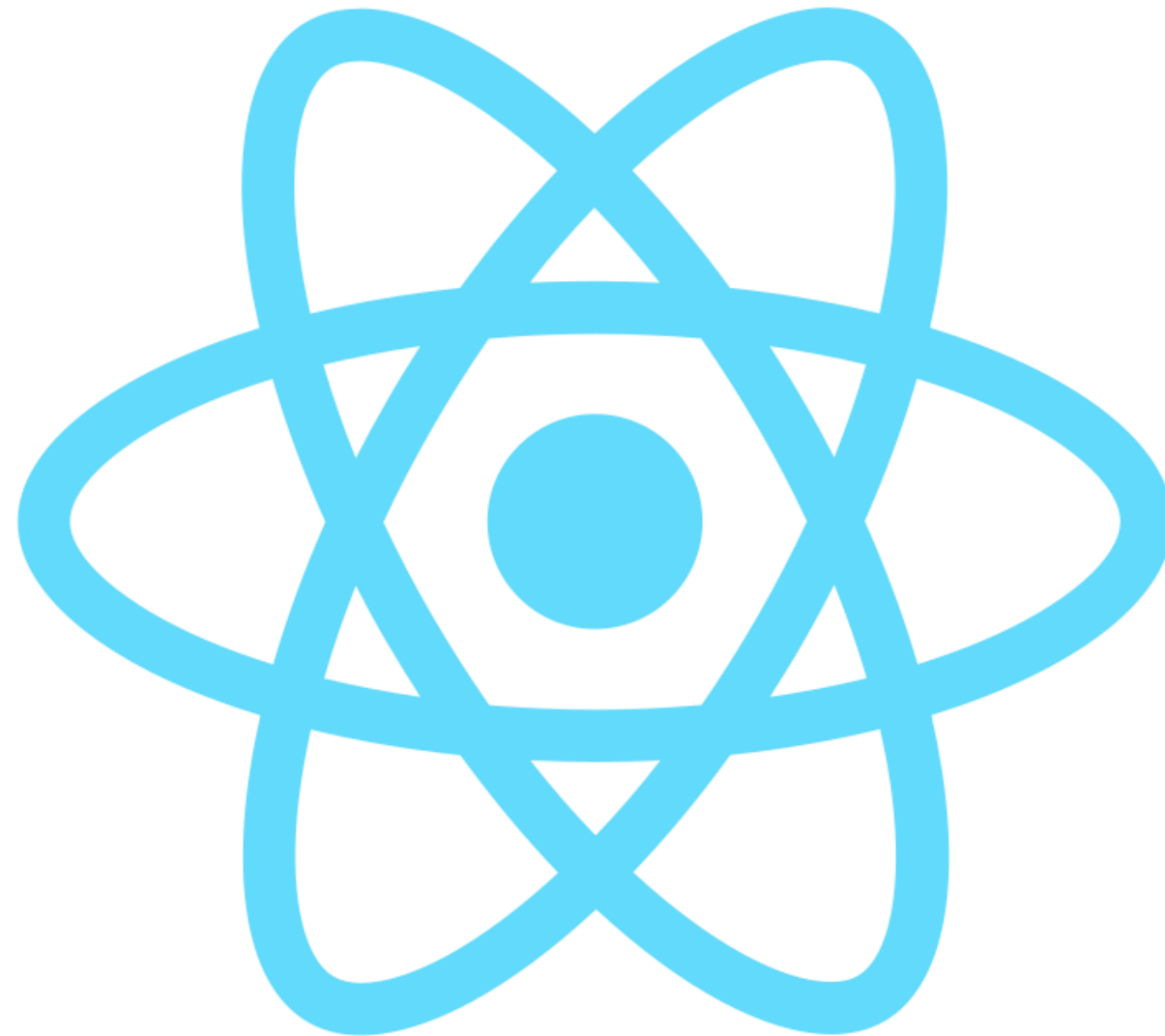
- ECMAScript ? ES5 ? ES2015 ? ES6 ?

➤ CLASSES ? FONCTIONS FLÉCHÉES ?

➤ CONSTANTES ? LET ? FOR...OF ?

➤ MODULES ? DESTRUCTURING ? ...

PASSONS AU REACT...



02

INSTALLATION DE REACT

Outils nécessaires à l'utilisation du Framework React

OUTILS NÉCESSAIRES


- Node.JS (afin de disposer du node package manager)
- React Developer Tools (extension Chrome / Firefox)
- Visual Studio Code (ou autre IDE équivalent)
- Un terminal pour taper des lignes de commande

03

CRÉER UNE APPLICATION

Intégration de React dans un projet et découverte de la CLI

CRÉER UNE APPLICATION AVEC REACT

- Première option :
 - Tout configurer sois même
- Deuxième option :
 - Utiliser Create React App 

COMMENT CRÉER UNE APPLICATION REACT

- Ajouter React à un projet via le CDN

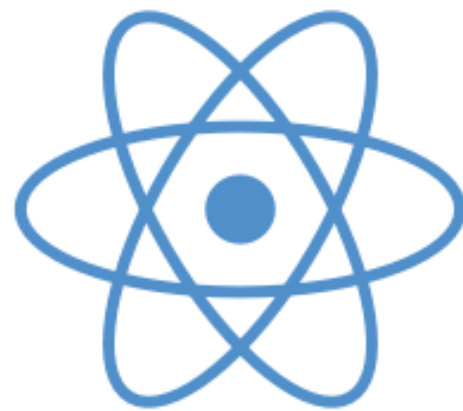
```
<html>
  <body>
    [...]
    <script src="https://cdnjs.cloudflare.com/ajax/libs/
react/16.8.3/umd/react.development.js" crossorigin>
  </script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/
reactdom/16.8.3/umd/reactdom.production.min.js"
  crossorigin></script>
  </body>
</html>
```


COMMENT CRÉER UNE APPLICATION REACT

- Pour utiliser JSX, il faut ajouter *BABEL*
 - `<script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>`
- Charger les scripts avec le MIME text/babelType :
 - `<script src="app.js" type="text/babel"></script>`

CRÉER UN PROJET AVEC LA CRA

Create React App




Official. No Setup. Minimal.

CRÉER UN PROJET AVEC LA CRA

- Création d'une nouvelle application via la commande
 - `$ npx create-react-app hello-react`
 - En utilisant npx vous savez que vous utiliserez toujours la dernière version de create-react-app
- Lancement de l'application via la commande (se positionner dans le dossier hello-react auparavant)
 - `$ npm start`

CRÉER UN PROJET AVEC LA CRA

- Attention , l'installation globale de create-react-app n'est plus conseillé par React et précononise l'utilisation de npx
 - `$ npm install -g create-react-app` 
- Si vous utilisiez cette façon de faire , utiliser la commande ci-dessous afin de pouvoir privilégier ensuite la commande npx
 - `$ npm uninstall -g create-react-app`

04

STRUCTURE APPLICATION

Analyse d'une application React

STRUCTURE APPLICATION REACT

- Analysons trois fichiers
 - `/public/index.html`
 - `/src/index.js`
 - `/src/App.js`

STRUCTURE APPLICATION REACT

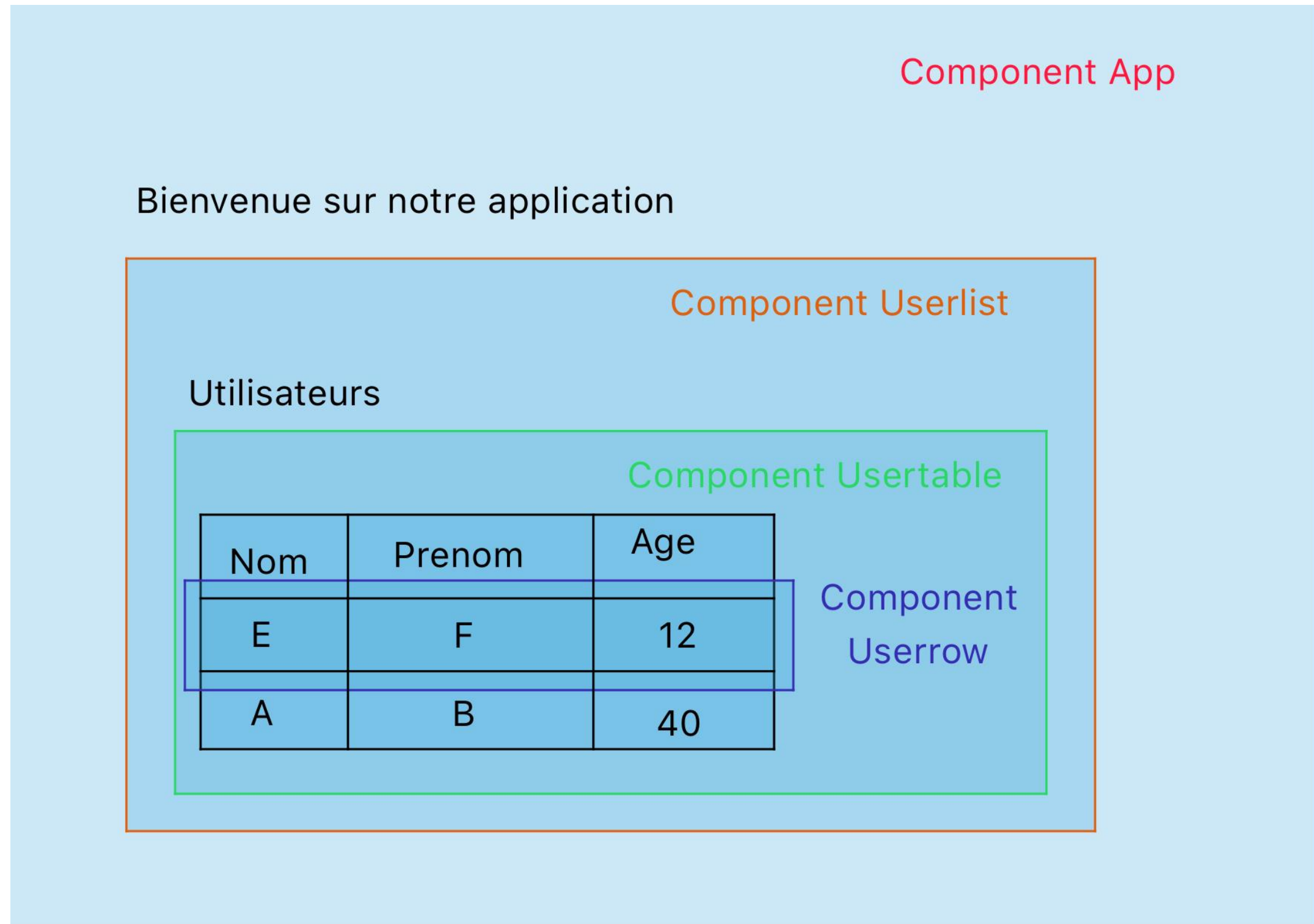
- App est le premier composant React que vous rencontrerez
- En React on appelle ça un « Component »
- Par défaut la syntaxe ES5 est utilisée mais utilisons la syntaxe des classes introduit en ES6
- Lorsque l'application sera compilée un transpileur (en l'occurrence *BABEL*) sera utilisé pour transformer le code ES6 en ES5

05

COMPOSANTS

Les composants d'une application React

COMPOSANTS D'UNE APPLI REACT



COMPOSANTS D'UNE APPLI REACT

- Création d'un composant
 - En créant une classe ES6
 - Attention, votre composant doit hériter de la classe Component (ou React.Component)
 - Ou en créant une simple fonction

COMPOSANTS D'UNE APPLI REACT

- Deux types de composants :
 - stateful (sous forme de classe)
 - stateless (sous forme de fonction)
- La différence : la présence ou non d'un « state »
- Nous verrons plus tard qu'il est possible d'avoir des composants stateless avec un state avec la notion de Hooks ⓘ

COMPOSANTS D'UNE APPLI REACT

- Les éléments qui composent un composant stateful :
 - La méthode render (avec la découverte du JSX)
 - Les propriétés d'un composant appelées « props »
 - Un objet appelé state pour stocker les propriétés qui appartiennent au composant (uniquement pour un composant stateful)

COMPOSANTS D'UNE APPLI REACT

- Qu'est ce que le « JSX » ?
 - Il s'agit d'une extension syntaxique du JavaScript
 - C'est le JSX qui va vous permettre de définir simplement la structure de notre interface utilisateur
 - Ce n'est pas obligatoire de l'utiliser mais je vous incite fortement à l'utiliser

COMPOSANTS D'UNE APPLI REACT

- Le « JSX » ?
 - Pourquoi l'utiliser ?
 - Car le JSX va vous aider à représenter le rendu qui va être généré par le composant
 - A quoi ça ressemble ?
 - `const testJSX = <h1>Bonjour</h1>;`

COMPOSANTS D'UNE APPLI REACT

- Le « JSX » ?
 - Mais c'est du HTML ? Non car derrière une simple ressemblance le JSX vous offre toute la richesse du JS

```
const firstname = "Jean";  
const testJSX = <h1>Bonjour {firstname}</h1>;
```

- Ainsi que les accolades du JSX vous pouvez mettre n'importe quelle expression JS

COMPOSANTS D'UNE APPLI REACT

- Le « JSX » ?

- Le JSX est ensuite transformé en code JS via Babel

```
const testJSX = React.createElement("h1", "Bonjour");
```

- Tout ce qui se trouve dans la méthode render est ensuite injecté dans le document HTML

COMPOSANTS D'UNE APPLI REACT

- Dans **VSCode** il est possible de rajouter le code suivant dans le fichier **settings.json**

```
"emmet.includeLanguages": {  
    "javascript": "javascriptreact"  
}
```

- Très pratique pour avoir les abréviations emmet même avec du JSX

COMPOSANTS D'UNE APPLI REACT

- A noter que la déclaration des propriétés d'une classe dans le constructeur peut se simplifier depuis ES10 (ES2019)

```
constructor() {  
  this.firstname = "Jean";  
  this.lastname = "Dupond";  
  this.age = 34;  
}
```



```
firstname = "Jean";  
lastname = "Dupond";
```

COMPOSANTS D'UNE APPLI REACT

- En React nous pouvons donc simplifier la déclaration des propriétés d'un composant stateful

```
constructor(props) {  
  super(props);  
  this.state = {...}  
}
```



```
state = {...};
```

COMPOSANTS D'UNE APPLI REACT

- Il va maintenant être intéressant de savoir comment on peut interagir avec ces composants avec des événements
 - Au clic
 - Au changement de valeur
 - etc...
- Et apprendre également à manipuler le state pour ajouter ou retirer des éléments par exemple

COMPOSANTS D'UNE APPLI REACT

- Se souvenir de trois choses :
 - Ne pas faire appel directement à la méthode de l'événement mais passer par une "arrow function" sinon la méthode s'exécute tout de suite
 - Ne pas manipuler le state directement mais passer par la méthode "setState"
 - Utilisation de "rest operator" pour faire les copies et éviter les mutations de state

06

CYCLES DE VIE

Cycles de vie dans le framework React

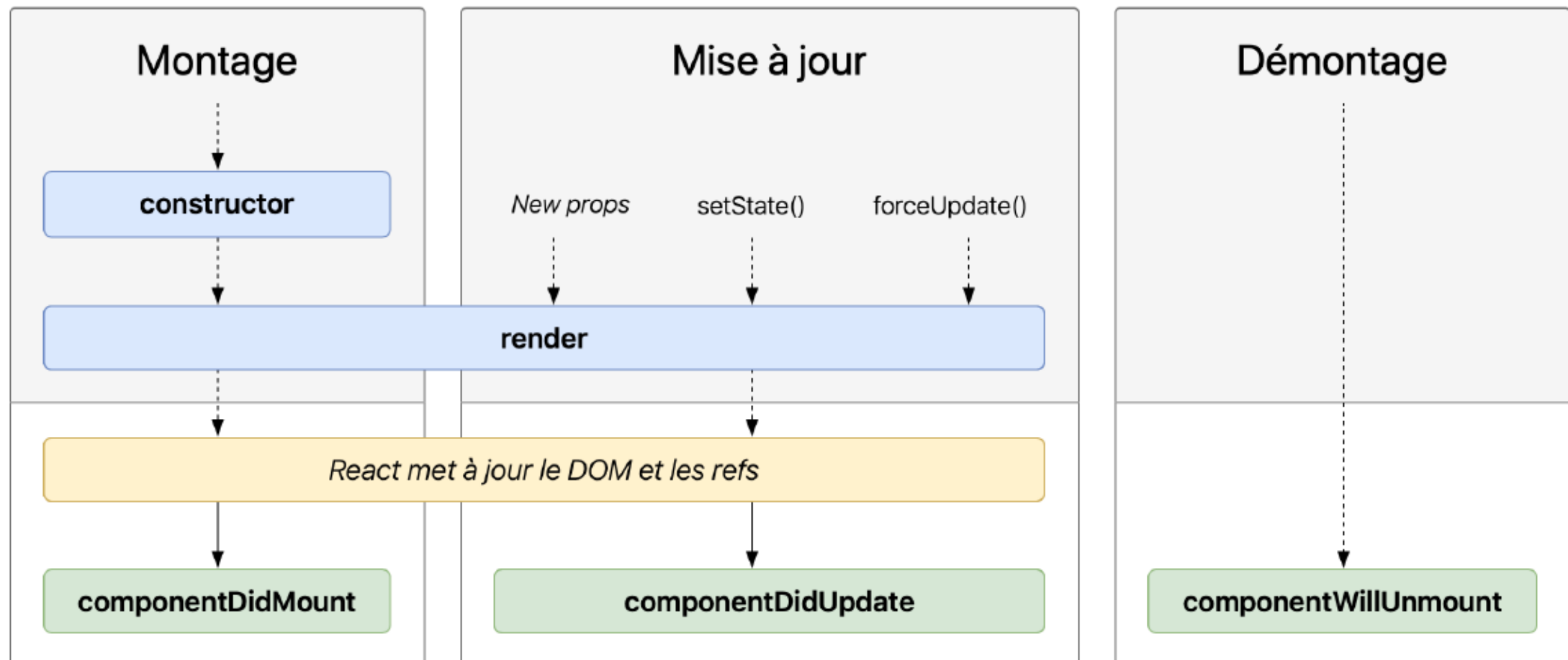
CYCLES DE VIE

"Phase de Render"

Méthodes pures, sans effets secondaires. Peuvent être interrompues, annulées ou redémarrées par React.

"Phase de Commit"

Peuvent opérer sur le DOM, engendrer des effets secondaires, programmer des mise à jour.



Source : Wojciech Maj

07

REACT ROUTER

Les routes dans le framework React

REACT ROUTER

- Les routes permettent de naviguer d'un composant à un autre
- Nous allons pour cela utiliser React Router
- Autrement dit nous allons gérer la navigation dans notre application

REACT ROUTER

- Les routes permettent de naviguer d'un composant à un autre

➤ npm install `react-router-dom`

- Importation des composants nécessaires dans index.js

```
import {  
  BrowserRouter as Router,  
  Route,  
  Switch  
} from "react-router-dom";
```

REACT ROUTER

- Création des routes
- Faire le rendu du composant Root à la place du composant App

```
ReactDOM.render(<App />,  
document.getElementById('root'));
```



```
ReactDOM.render(<Root />,  
document.getElementById('root'));
```

REACT ROUTER

- Se souvenir de trois choses :
 - « path » permet d'indiquer le modèle du chemin de l'URL
 - « exact » permet d'indiquer que l'URL ne sera pas variable et respectera strictement ce modèle
 - « :param » dans l'attribut path pour indiquer un param qui sera véhiculé dans l'URL

REACT ROUTER

- Comment changer l'URL de la page et appeler un nouveau composant lié à notre route ?
 - `this.props.history.push()`
 - En paramètre de cette méthode il suffit tout simplement de passer l'URL souhaitée
 - Exemple: `this.props.history.push('/login')`

REACT ROUTER

- Comment changer l'URL de la page et appeler un nouveau composant lié à notre route
 - Une autre façon de faire consiste à utiliser le composant Link de React Router
 - Import du composant

```
import {Link} from "react-routerdom"
```

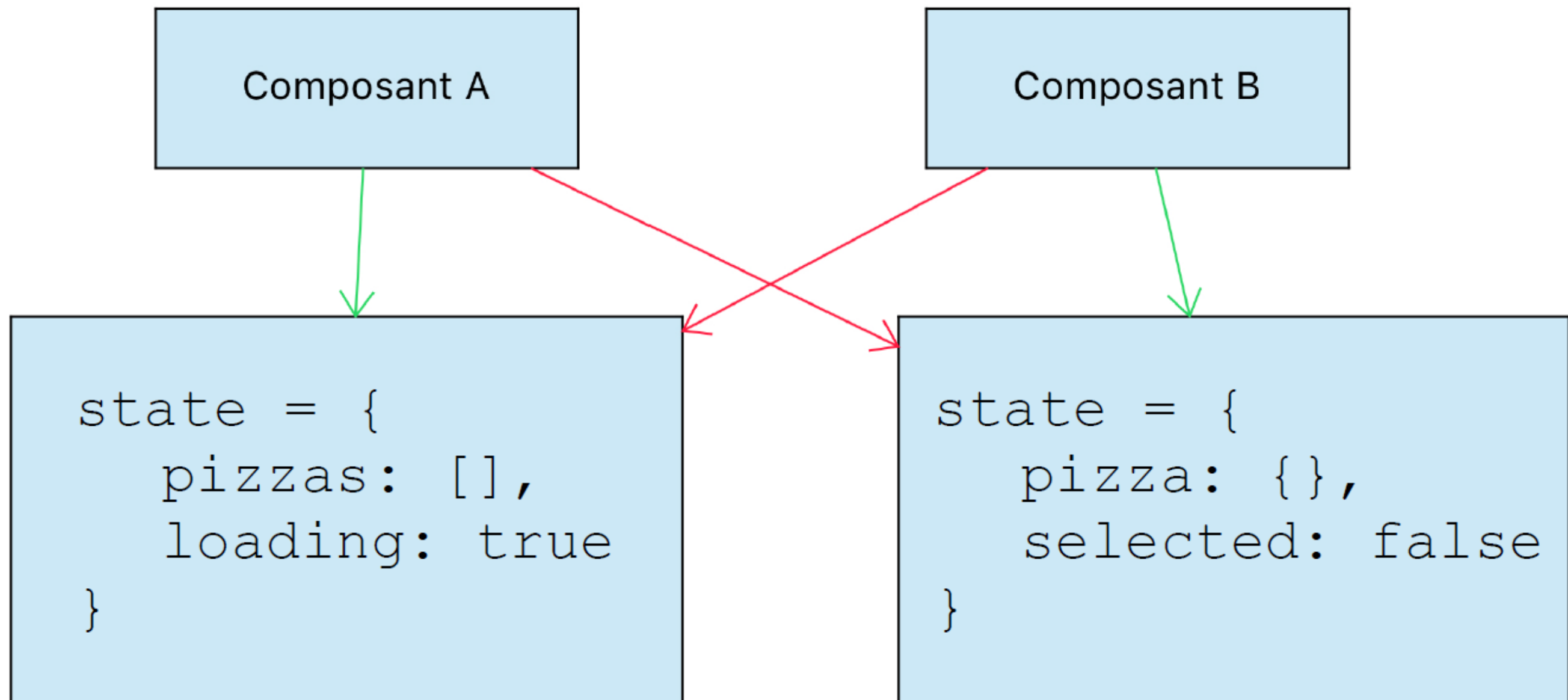
- Utilisation du composant
- `<Link to={`/login`} >Connexion</ Link>`

08

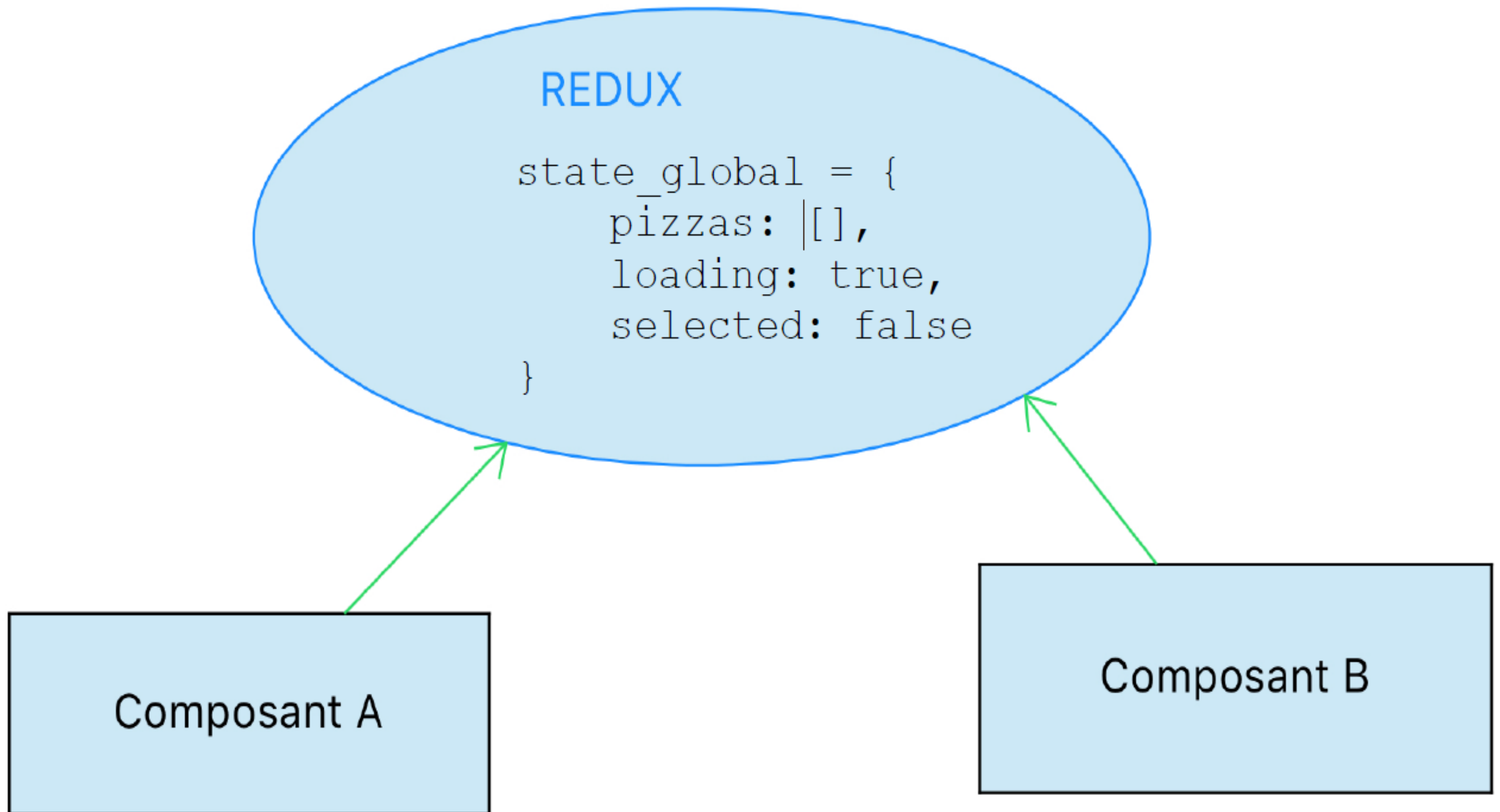
STORE AVEC REDUX

Utilisation d'un store dans le Framework React

LIMITATION POSSIBLE SANS REDUX



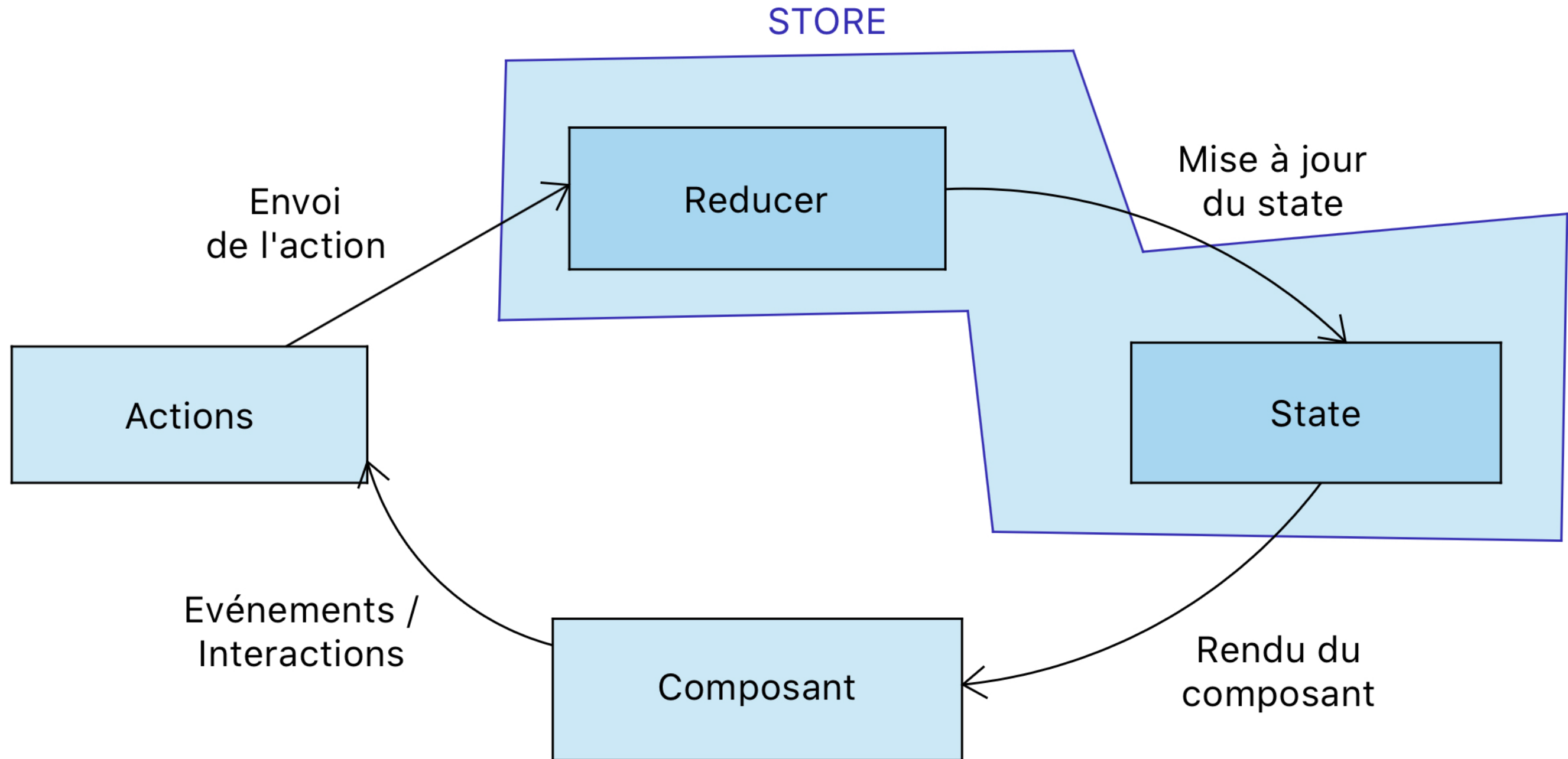
CE QUE PROPOSE REDUX



STORE AVEC REDUX

- Nous allons utiliser Redux pour la gestion d'un state global
- Avec Redux le state ne va plus se limiter à un composant mais va être partagé avec l'ensemble de l'application
- Redux est utile dès lors qu'une application devient importante ou difficile à faire évoluer ou une application avec une grande imbrication des composants

COMMENT FONCTIONNE REDUX



ACTION

- **Object** JavaScript qui dit au **Reducer** comment modifier les données du **State** global
- Une **Action** a toujours une propriété **type** pour donner un nom/identifiant à cette action
- Une **Action** a souvent des données à faire passer au **Reducer** via une propriété appelée **payload** afin de mettre à jour le **State** global

ACTION

- Exemple d'une action :

```
{  
  type: "ADD_ORDER",  
  payload: {  
    id: "CMD-1234",  
    pizzas: [],  
    total: 0,  
    paid: false  
  }  
}
```


REDUCER

- Le **Reducer** est une **fonction** qui retourne des données (le nouveau **state**)
- Cette fonction permet deux paramètres :
 - Le précédent state (souvent intitulé initialState)
 - Une action
- Par défaut la fonction retourne le précédent state

REDUCER

- Exemple d'un Reducer :

```
function rootReducer(state, action) {  
  switch (action.type) {  
    case "ADD_ORDER":  
      return {  
        // Nouveau state :)  
      };  
    default:  
      return state;  
  }  
}
```

STORE

- Le **Store** est un **objet** qui contient les données de notre application , c'est à dire notre state global
- Pour créer un **Store**:
 - `Redux.createStore(rootReducer);`
- Pour rendre le store accessible depuis l'ensemble des composants , utilisation d'un composant **Provider**

STORE

- A noter que pour pouvoir utiliser l'extension Redux DevTools (usage recommandé durant le développement) il faut ajouter l'information suivante au moment de la création du store

```
const store = createStore(  
  rootReducer,  
  window.__REDUX_DEVTOOLS_EXTENSION__  
&& window.__REDUX_DEVTOOLS_EXTENSION__()  
);
```

CONNECT

- Afin de connecter nos composants à notre **Store** nous avons besoin d'une fonction **connect** de React Redux
 - `connect(A, B)(Composant à connecter)`
 - A est une fonction généralement appelée `mapStateToProps`
 - B est une fonction généralement appelée `mapDispatchToProps`

CONNECT

- `mapStateToProps` est une fonction qui prend le state global en paramètre et retourne un objet
- Cet objet permet d'accéder au state global via les props d'où le nom de la fonction

CONNECT

- `mapDispatchToProps` est une fonction qui prend la fonction `dispatch` en paramètre et retourne un objet
- Cet objet permet de déclencher des actions via les props

REDUX

- Installation via npm
 - `npm install redux react-redux`