
React

Formation AWS

Avant tout

Jeudi 28 mai - 2021

Qui suis-je ?
Description

Tech-Lead Fortuneo

AWS Solution Architect Certified

Co-Fondateur & CTO Azivko

Ingénieur diplomé de l'ISEN (2019)

Formateur Cloud, Devops et Frontend

Participation à des startups week-ends et conférences entreprenariat/tech

Format du cours

1. Interrompez-moi s'il y a quelque chose de pas clair
2. Il n'y a pas de question bête
3. N'hésitez pas à demander de clarifier un point ou d'approfondir un sujet
4. Travaillez à plusieurs si vous le souhaitez pour les TP
5. **Horaires:** 8h30-12h30, 14h-17h30.

   Jeudi matin: Cours

   Jeudi aprem: Cours et TP

   Vendredi matin: Cours

   Vendredi aprem: TP noté

Exemple d’exercice

Exercice: Titre

1. Instructions
2. …

Bonus: Pour ceux qui ont le temps ou la volonté

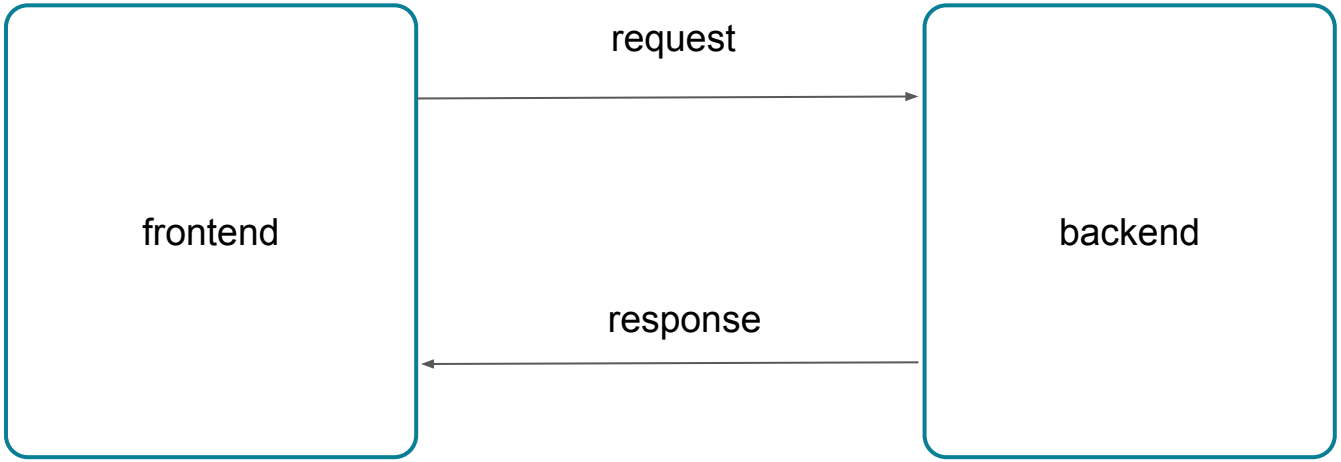
Formation REACT

Welcome

Setup

https://codesandbox.io/s/react-lessons-c0vx3c

Fondamentaux du web



Fondamentaux de JS: Ecmascript

C'est quoi ecmascript ?

ES6 a été pensé pour créer des applications web facilement maintenables, tout en restant compatibles avec le code existant. L'idée a été d'ajouter de nouvelles fonctionnalités au langage.

Fondamentaux de JS: falsy value

Fondamentaux de JS: arrow functions

```
const greetings = (name) => console.log('Hello, ' + name)
```

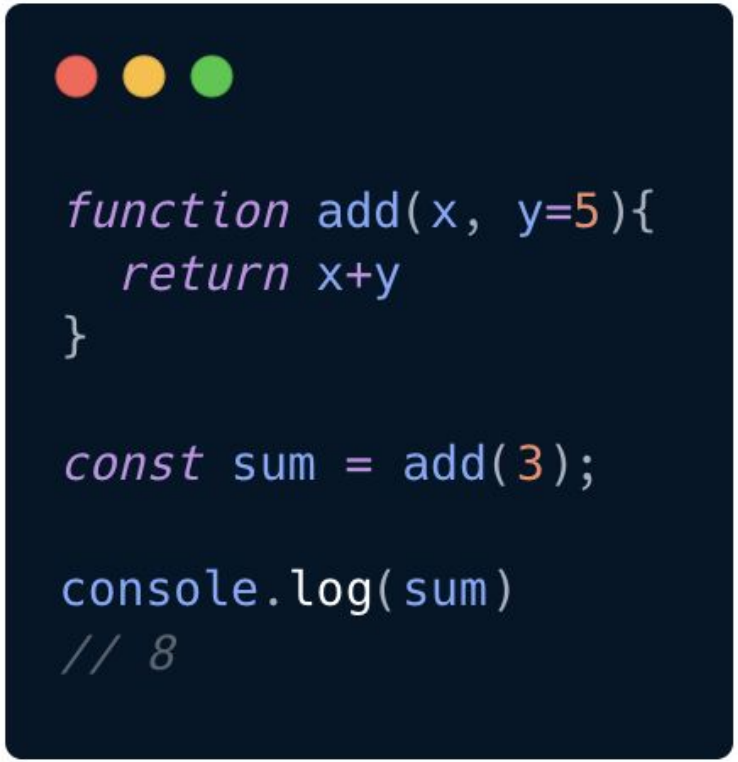
Fondamentaux de JS: déstructuration

```
const cadeaux = ["poupée","fromage"];
const nouveauxCadeaux = ["peigne"];

const tout = [...cadeaux, ...nouveauxCadeaux]

console.log(tout)
// ['poupée', 'fromage', 'peigne']
```

Fondamentaux de JS: déstructuration 2

```
const fromage = {
  croute: true
}

const newFromage = {
  ...fromage,
  color: "blue"
}

console.log(newFromage)
// {
//    croute: true,
//    color: "blue"
// }
```

Fondamentaux de JS



```
function add(x, y=5){
  return x+y
}

const sum = add(3);

console.log(sum)
// 8
```

Fondamentaux de JS: map

```
const list = [1,2,3,4];

const newList = list.map(num => num * 2);

console.log(newList);
// [2, 4, 6, 8]
```

Fondamentaux de JS: filter

```
const list = [1,2,3,4];

const newList = list.filter(num => num % 2 === 0);

console.log(newList);
// [2, 4]
```

Fondamentaux de JS: reduce

```
const list = [1,2,3,4];

const newList = list.reduce((total, value) => total + value);

console.log(newList);
// 10
```

Fondamentaux de JS: every

```
const list = ["fromage","fromage","fromage"];

const containsOnlyCheese = list.every(cheese => cheese === "fromage");

console.log(containsOnlyCheese);
// true
```

Fondamentaux de JS: some

```
const list = ["fromage","patate","fromage"];

const containsPatate = list.some(item => item === "patate");

console.log(containsPatate);
// true
```

Formation REACT

Les fondamentaux de React

Les bases de React

1. Pourquoi React
2. JS Basique
3. API react (react, reactDOM)
4. JSX
5. Créer un composant (function)
6. Styling (classname, style)
7. Formulaire (ref)
8. Array
9. Lifting state

https://codesandbox.io/s/react-lessons-c0vx3c

Les bases de React: Point culture

2011

Facebook Ads connaît une très forte croissance, entraînant des demandes de nouvelles features de plus en plus fréquentes et l'augmentation de la taille de l'équipe travaillant sur le projet. Jordane Walke, salarié chez Facebook, crée FaxJS.

2012

Facebook rachète Instagram. Voulant profiter des dernières avancées technologiques de Facebook pour améliorer ses produits, Instagram commence alors à faire pression pour que FaxJS devienne une librairie open-source.

2013

FaxJS devient officiellement "ReactJS" et passe open-source. La communauté n'est cependant pas encore celle qu'elle est aujourd'hui, et beaucoup regardent React d'un œil sceptique.

2014

React attire de plus en plus l'attention et prend une ampleur considérable. De plus, de nombreux outils annexes commencent à voir le jour, comme React Dev Tools ou encore React Hot Loader

2015

React devient stable et de plus en plus utilisé. Netflix en parle, ce qui propulsera la librairie sur les devants de la scène. AirBnB adopte également la librairie. Et la même année, sortie de React Native pour iOS et Android sur Github.

Les bases de React: Point culture

Si React a su se démarquer de ses concurrents à l'époque et séduire la communauté, c'est essentiellement grâce à ses nombreux points forts et sa DX remarquable :

- Code propre
  - orienté composants
  - avec des cycles de vie
  - JSX
  - Hooks
- Performant (Virtual DOM)
- Communauté importante et très active

Aujourd'hui, React est utilisé par de nombreuses sociétés partout dans le monde : Netflix, Facebook, Instagram, Airbnb, Reddit, Dropbox, WhatsApp, Uber, Yahoo, Atlassian, The New York Times, etc.

En 2022 : https://2021.stateofjs.com/en-US/libraries/front-end-frameworks

Les bases de React: Javascript, déclaration impérative

Exercice:

- Créer une div en html qui contient un id valant “root”
- Créer une div qui contient du texte en javascript et l’ajouter à la div root

Temps: 20 minutes

Bonus: Créer la div root en javascript

https://codesandbox.io/s/**react-lessons-c0vx3c**

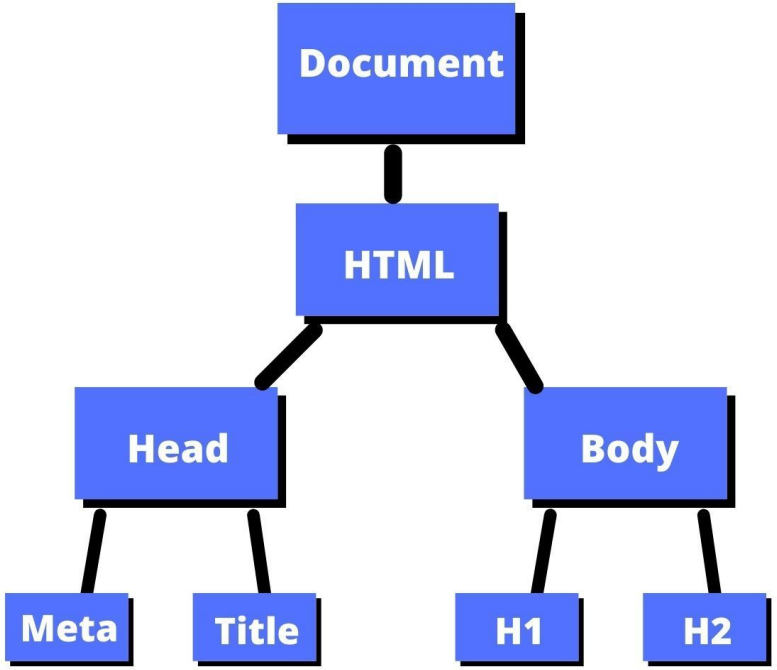
Les bases de React: Javascript, déclaration impérative

```
<body>
  <div id="root"></div>
  <script type="module">
    const rootElement = document.getElementById('root');
    const element = document.createElement('div');
    element.textContent = 'Hello World';
    element.className = 'container';
    rootElement.append(element);
  </script>
</body>
```

Les bases de React: Javascript, déclaration impérative

```
<body>
  <script type="module">
    const rootElement = document.createElement('div');
    rootElement.setAttribute('id', 'root');
    document.body.append(rootElement);
    const element = document.createElement('div');
    element.textContent = 'Hello World';
    element.className = 'container';
    rootElement.append(element);
  </script>
</body>
```

Les bases de React: Javascript, déclaration impérative



Les bases de React: Javascript, déclaration impérative

La programmation impérative est un paradigme de programmation qui décrit les opérations en termes d'états du programme et de séquences d'instructions exécutées par l'ordinateur pour modifier l'état du programme.

Les bases de React: Javascript, déclaration impérative

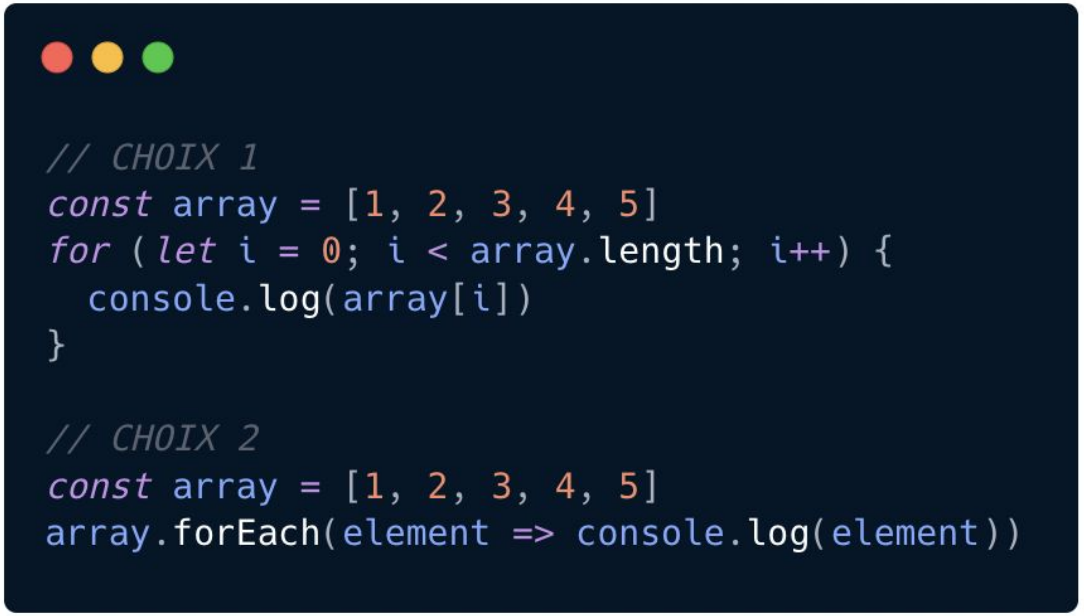
La programmation **déclarative** décrit **ce que le programme doit accomplir**, alors que la programmation **impérative** se concentre sur **la manière de l'accomplir.**

Les bases de React: Javascript, déclaration impérative

```
<body>
  <script type="text/javascript">
    // Créé une div pour contenir le paragraphe
    const rootDiv = document.createElement('div')
    // Attribue l'ID "root" à la div
    rootDiv.setAttribute('id', 'root')
    // Rajoute la div à la page
    document.body.appendChild(rootDiv)

    // Créé un paragraphe
    const paragraph = document.createElement('p')
    // ...lui attribue le texte "Hello World"
    paragraph.textContent = 'Hello World'
    // ...et l'ajoute à la div créée précedemment
    rootDiv.append(paragraph)
  </script>
</body>
```

Les bases de React: Javascript, déclaration impérative

Les bases de React: impératif / déclaratif ?



```
// CHOIX 1
const array = [1, 2, 3, 4, 5]
for (let i = 0; i < array.length; i++) {
  console.log(array[i])
}

// CHOIX 2
const array = [1, 2, 3, 4, 5]
array.forEach(element => console.log(element))
```

Les bases de React: Les API React: React, ReactDOM

```
React.createElement(type, [props], [...children])
```

- **type:** chaîne de caractères désignant un tag HTML *('div', 'span', 'p', etc.)*, ou un composant React
- **props:** attributs qui seront ajoutés au tag (comme un id ou un className, par exemple).
- **children:** *"enfants"* de l'élément, autrement dit ses descendants directs dans le DOM HTML généré.

Les bases de React: Les API React: React, ReactDOM

```
// Pas de props, un seul enfant
React.createElement('p', null, 'Hello World')
// Pas de props, plusieurs enfants
React.createElement('p', null, 'Hello', ' ', 'World')
// On précise les enfants avec la prop "children"
React.createElement('p', {
  children: ['Hello', ' ', 'World'],
})
```

Les bases de React: Les API React: React, ReactDOM

```
// Création de l'élément root
const rootElement = document.getElementById('root')
// Création de la racine React
const root = ReactDOM.createRoot(rootElement)
// Création d'un element
const paragraph = React.createElement('div', null, "coucou");

// Rendu de l'élément
root.render(element)
```

Les bases de React: Les API React

Exercice:
Avec les APIs React, migrer l'application précédente

Bonus:
Avec les APIs React, créer :

- un paragraphe avec la classe "content-text" et le texte "Liste de courses :"
- une liste non-ordonnée ul avec l'ID "shopping-list" contenant les éléments li suivants : Pommes, Poires,Fraises,Pâtes

Temps: 30 minutes

https://codesandbox.io/s/react-lessons-c0vx3c?file=/exercice-2.html

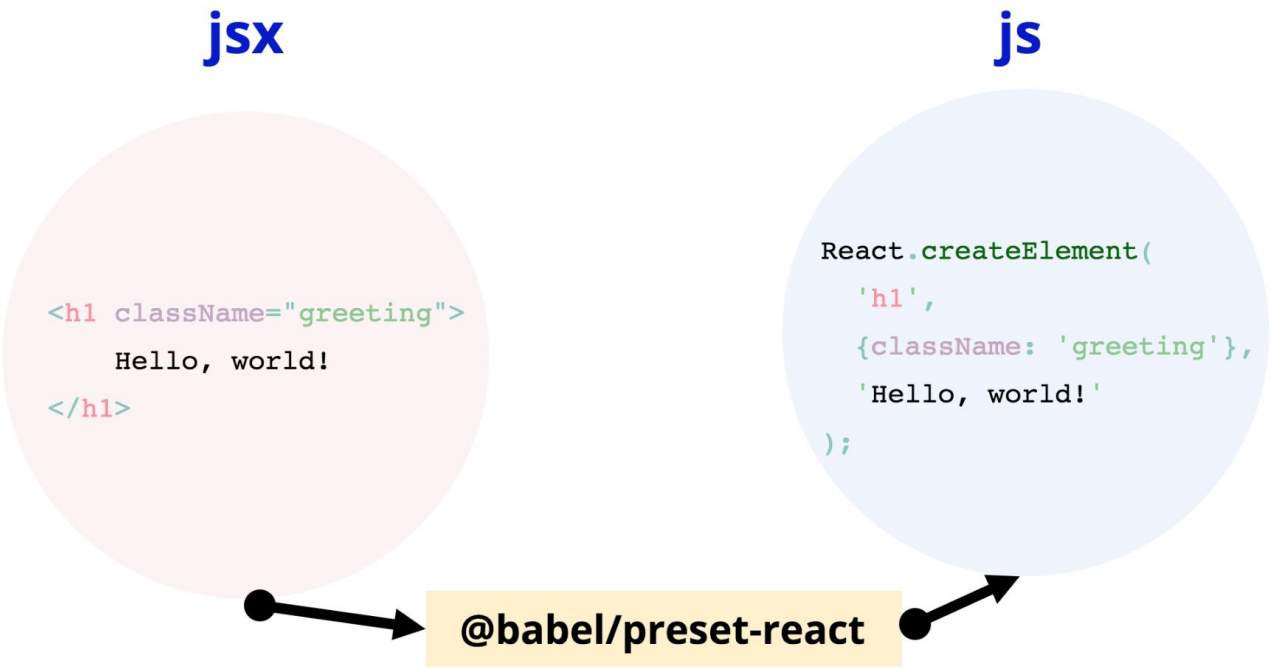
Les bases de React: Les API React

```
<script type="module">
 const rootElement = document.getElementById('root')
 const element = React.createElement('div', {
   className: 'container',
   children: 'Hello World',
 })
 ReactDOM.createRoot(rootElement).render(element)
</script>
```

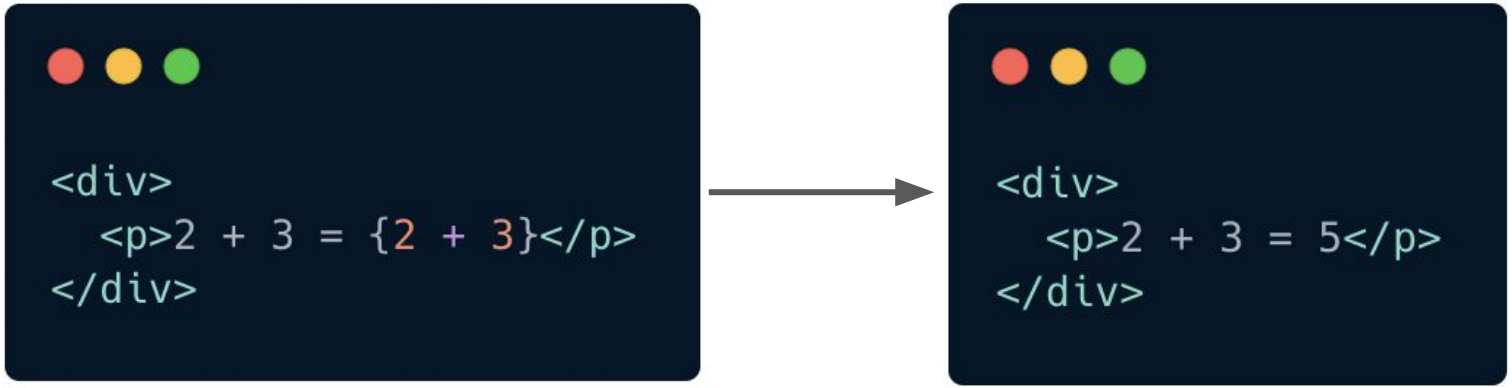
Les bases de React: Les API React

```
<script type="text/javascript">
  const root = ReactDOM.createRoot(document.getElementById('root'))

  const paragraph = React.createElement('p', {
    className: 'content-text'
  }, 'Liste de courses :')
  const shoppingList = React.createElement('ul', {
    id: 'shopping-list',
    children: [
      React.createElement('li', null, 'Pommes'),
      React.createElement('li', null, 'Poires'),
      React.createElement('li', null, 'Fraises'),
      React.createElement('li', null, 'Pâtes'),
    ]
  })
  const mainDiv = React.createElement('div', {
    className: 'main',
  }, paragraph, shoppingList)

  root.render(mainDiv)
</script>
```

Les bases de React: JSX



Les bases de React: JSX



Les bases de React: JSX & Ecmascript

```
const greetings = (name) =>
console.log('Hello, ' + name)
```



```
function greetings(name) {
  console.log('Hello, ' + name)
}
```

Les bases de React: JSX

```jsx
// Interpolation
const href = 'https://google.fr'
const link = <a href={href}>Google</a>

// Gérer des états
const user = {
  profile: "test.html",
  avatarUrl: "image.jpeg"
}
<a href={user.profile}>
  <img src={user.avatarUrl} />
</a>

// convention camelCase
<button onClick={() => console.log("coucou")>Bouton</button>
```

Les bases de React: JSX

Exercice: Migrer l'application précédente en jsx

Temps: 20 minutes

Aide:

```
<body>
  <div id="root"></div>
  <script src="https://unpkg.com/react@18.1.0/umd/react.development.js"></script>
  <script src="https://unpkg.com/react-dom@18.1.0/umd/react-dom.development.js"></script>
  <script src="https://unpkg.com/@babel/standalone@7.12.4/babel.js"></script>
  <script type="text/babel">
    const element = ???
    ReactDOM.createRoot(document.getElementById('root')).render(element);
  </script>
</body>
```

Les bases de React: JSX

```
<body>
  <div id="root"></div>
  <script src="https://unpkg.com/react@18.1.0/umd/react.development.js"></script>
  <script src="https://unpkg.com/react-dom@18.1.0/umd/react-dom.development.js"></script>
  <script src="https://unpkg.com/@babel/standalone@7.12.4/babel.js"></script>
  <script type="text/babel">
    const element = <div className='container'>Hello World</div>;
    ReactDOM.createRoot(document.getElementById('root')).render(element);
  </script>
</body>
```

Les bases de React: Créer un composant

```
<div class="container">
  <h1>Hello World</h1>
  <p class="text-body">Some text</p>
  <p class="text-body">Another paragraph</p>
  <p class="text-body">Yet another one</p>
</div>
```

Les bases de React: Créer un composant

Exercice: Créer une fonction appelé message, prenant pour argument children, qui retourne du JSX (depuis l'exercice précédent)

Temps: 10 minutes

Aide:

- Partir du même code que l'exercice précédent
- Doit être appelée de la manière suivante :

```
{paragraph('Some text')}
```

Les bases de React: Créer un composant

```
function paragraph(text) {
  return <p className="text-body">{text}</p>
}

const ui = (
  <div className="container">
    <h1>Hello World</h1>
    {paragraph('Some text')}
    {paragraph('Another paragraph')}
    {paragraph('Yet another one')}
  </div>
)
```

Les bases de React: Créer un composant

Exercice: Utiliser React.createElement() et lui passer la fonction paragraph.

Temps: 10 minutes

Les bases de React: Créer un composant

```
<body>
  <div id="root"></div>
  <script src="https://unpkg.com/react@18.1.0/umd/react.development.js"></script>
  <script src="https://unpkg.com/react-dom@18.1.0/umd/react-dom.development.js"></script>
  <script src="https://unpkg.com/@babel/standalone@7.12.4/babel.js"></script>
  <script type="text/babel">
    function message({ children }) {
      return <div className='message'>{children}</div>;
    }

    const element = (
      <div className='container'>
        {React.createElement(message, { children: 'Hello World' })}
        {React.createElement(message, { children: 'Goodbye World' })}
      </div>
    );

    ReactDOM.createRoot(document.getElementById('root')).render(element);
  </script>
</body>
```

Les bases de React: Créer un composant

```
function paragraph(props) {
  return <p className="text-body">{props.children}</p>
}

const ui = (
  <div className="container">
    <h1>Hello World</h1>
    {React.createElement(paragraph, { children: 'Some text' })}
    {React.createElement(paragraph, { children: 'Another paragraph' })}
    {React.createElement(paragraph, { children: 'Yet another one' })}
  </div>
)
```

Les bases de React: Créer un composant

Exercice: Remplacer le createElement par du JSX

Temps: 10 minutes

Les bases de React: Créer un composant

```jsx
function paragraph(props) {
  return <p className="text-body">{props.children}</p>
}

const ui = (
  <div className="container">
    <h1>Hello World</h1>
    {React.createElement(paragraph, { children: 'Some text' })}
    {React.createElement(paragraph, { children: 'Another paragraph' })}
    {React.createElement(paragraph, { children: 'Yet another one' })}
  </div>
)
```



```jsx
<div className="container">
   <h1>Hello World</h1>
   <paragraph>Some text</paragraph>
   <paragraph>Another paragraph</paragraph>
   <paragraph>Yet another one</paragraph>
 </div>
```

Les bases de React: Créer un composant

```
function paragraph(props) {
  return <p className="text-body">{props.children}</p>
}

const ui = (
  <div className="container">
    <h1>Hello World</h1>
    <paragraph>Some text</paragraph>
    <paragraph>Another paragraph</paragraph>
    <paragraph>Yet another one</paragraph>
  </div>
)
```

```
function Paragraph(props) {
  return <p className="text-body">{props.children}</p>
}

const ui = (
  <div className="container">
    <h1>Hello World</h1>
    <Paragraph>Some text</Paragraph>
    <Paragraph>Another paragraph</Paragraph>
    <Paragraph>Yet another one</Paragraph>
  </div>
)
```

Les bases de React: Comment avoir du style

1. La propriété style

   HTML : `<div style="margin-top: 20px; background-color: blue;"></div>`
   React: `<div style={{marginTop: 20, backgroundColor: 'blue'}} />`

   la notation {{ }} est une combinaison de JSX et d'un objet, on peut le décomposer en :
   const myStyles = {marginTop: 20, backgroundColor: 'blue'}
   * *

2. La propriété className

   HTML : `<div class="test"></div>`
   React: `<div className="test" />`

   le mot class est réservé en javascript et ne peut pas être utilisé dans un autre contexte

Les bases de React: Comment avoir du style

Exercice: Ajouter du style à un composant

Bonus:

1. Créer un composant custom
2. Accepter une propriété size

https://codesandbox.io/s/react-lessons-c0vx3c?file=/exercice-4.html

Les bases de React: Formulaire

```
function submit(event){
    console.log(event.target.elements.username.value);
}

<form onSubmit={submit}>
    <input name="username"/>
</form>
```

Les bases de React: Formulaire

Exercice: Gérer un formulaire

https://codesandbox.io/s/react-lessons-c0vx3c

Les bases de React: Array

```
const myList = ['Pommes', 'Poires', 'Fraises']
const ui = (
  <ul>
    {myList.map(item => (
      <li>{item}</li>
    ))}
  </ul>
)
```

Les bases de React: Array

Failed to compile

```
./src/components/project/label/index.js
  Line 111:17:  Missing "key" prop for element in iterator  react/jsx-key

Search for the keywords to learn more about each error.
```

This error occurred during the build time and cannot be dismissed.

Les bases de React: Array, règle #1

```
const ui = (
  <ul>
    {students.map((student) => (
      <li key={student.id}>
        <strong>{student.name}</strong>
        <i>{student.grade}</i>
      </li>
    )}
  </ul>
```

La key doit être spécifiée sur les descendants directs de la liste, autrement dit sur l'élément JSX parent retourné dans le .map

Les bases de React: Array, règle #2

```
const ui = (
  <ul>
    {students.map((student) => (
      <li key={student.id}>
        <strong>{student.name}</strong>
        <i>{student.grade}</i>
      </li>
    )}
  </ul>
```

La clef unique doit vraiment l'être : unique. Si deux enfants d'une liste ont la même key, vous risquez fort d'avoir de mauvaises surprises.

Les bases de React: Array, règle #3

```
const ui = (
  <ul>
    {items.map((item, index) => (
      <li key={index}>{item}</li>
    ))}
  </ul>
)
```

Ne pas utiliser l'index comme clé, mais utiliser une clé de l'objet ou en fabriquer une.

Les bases de React: Array

Exercice: Le mot clé “key” en React

Temps: 10 minutes

Les bases de React: Lifting & collocating state

Exercice: Remonter le state

Temps: 20 minutes

Les bases de React: Affichage conditionnel

```
isFromage ? <p>chouette</p> : <p>Oh non ...</p>
```

```
isFromage && <p>chouette</p>
```

Les bases de React: Fragments

```
const ui = (
    <h1>Hello world</h1>
    <p>Click on the button bellow !</p>
    <button>Click me !</button>
)
```

Les bases de React: Fragments

```jsx
const ui = (
  <>
    <h1>Hello world</h1>
    <p>Click on the button bellow !</p>
    <button>Click me !</button>
  </>
)
```

Formation REACT

Les hooks

Les hooks React

1. useState (formulaire)
2. useEffect (side effect)
3. useEffect (HTTP)
4. useRef
5. useReducer

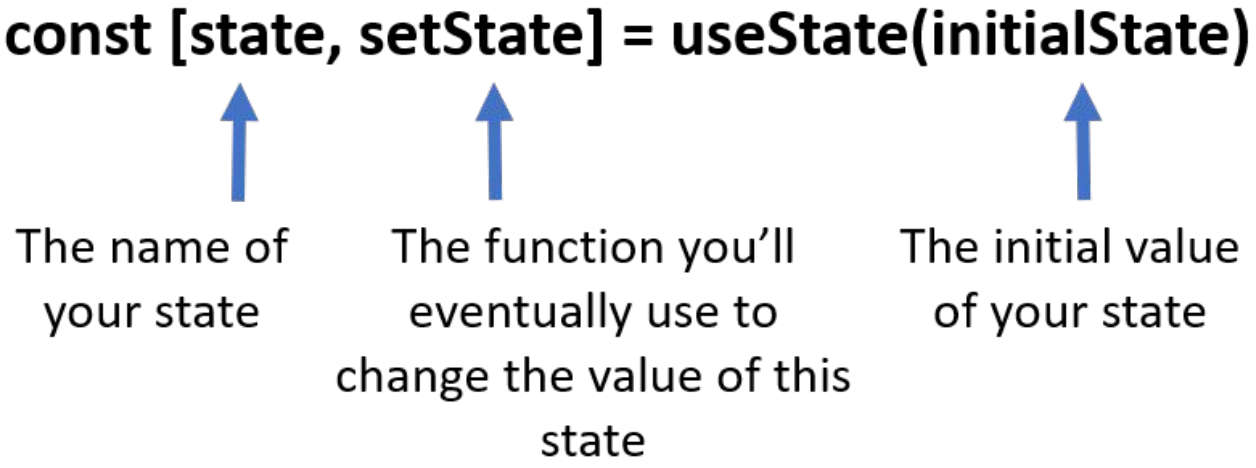
https://codesandbox.io/s/**je3jf4**

Les hooks React: Contexte

1. Compliqué de séparer en plus petits composants
2. Améliorer la lisibilité du code (fin des classes et des méthodes lifecycle)
3. Améliorer le partage de code entre les composants

Références: Introducing Hooks – React, Hooks at a Glance – React

Les hooks React: State Hook



Les hooks React: State Hook

Exercice: Gérer un “état”

Bonus:

- forcer le nom en minuscule
- afficher une erreur quand le nom n’est pas valide

Temps: 10 minutes

Les hooks React: Effect Hook

```
function Example() {
  const [count, setCount] = useState(0);

 useEffect(() => {
   document.title = `Vous avez cliqué ${count} fois`;
 });


  return (
    <div>
      <p>Vous avez cliqué {count} fois</p>
      <button onClick={() => setCount(count + 1)}>
        Cliquez ici
      </button>
    </div>
  );
}
```

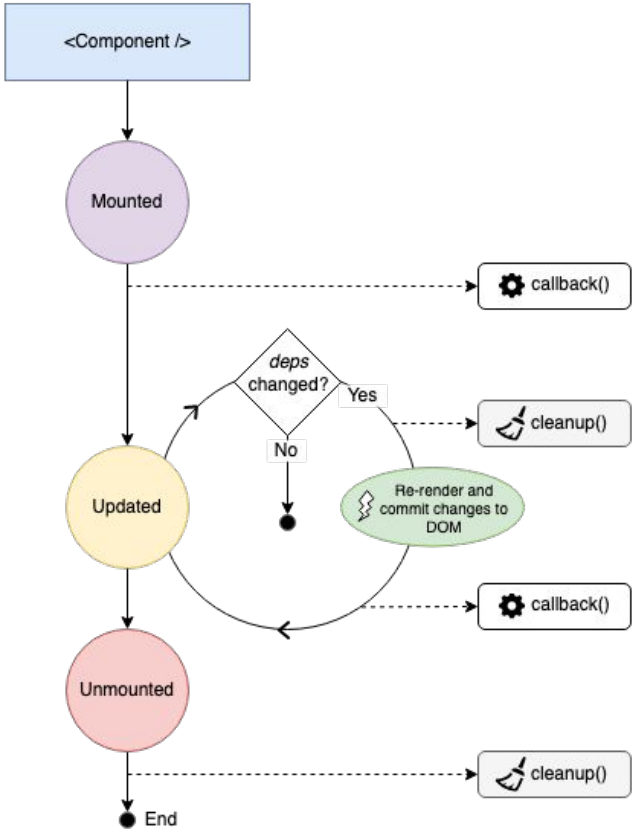
Les hooks React: Effect Hook

```
function Example() {
  const [count, setCount] = useState(0);

 useEffect(() => {
   document.title = `Vous avez cliqué ${count} fois`;
 },[count]);


  return (
    <div>
      <p>Vous avez cliqué {count} fois</p>
      <button onClick={() => setCount(count + 1)}>
        Cliquez ici
      </button>
    </div>
  );
}
```

Les hooks React: Effect Hook

```
function FriendStatus(props) {
  const [isOnline, setIsOnline] = useState(null);

 useEffect(() => {
   function handleStatusChange(status) {
     setIsOnline(status.isOnline);
   }
   ChatAPI.subscribeToFriendStatus(props.friend.id, handleStatusChange);
   return function cleanup() {
     ChatAPI.unsubscribeFromFriendStatus(props.friend.id, handleStatusChange);
   };
 });

  if (isOnline === null) {
    return 'Chargement...';
  }
  return isOnline ? 'En ligne' : 'Hors-ligne';
}
```

Les hooks React: Effect Hook



Les hooks React: Effect Hook

Exercice: Gérer un “effet”

Bonus:

- “Optimiser” le localStorage get

Temps: 10 minutes

Formation REACT

Projet

Projet Todo

Pré-requis: Nécessite une application React

Exercice: Créer une application todo

1. Input pour ajouter un élément
2. Bouton pour enlever un élément
3. Ajouter un peu de style

Bonus:

- garder l'état en mémoire localstorage

Temps: 1h / 4h

Projet Todo

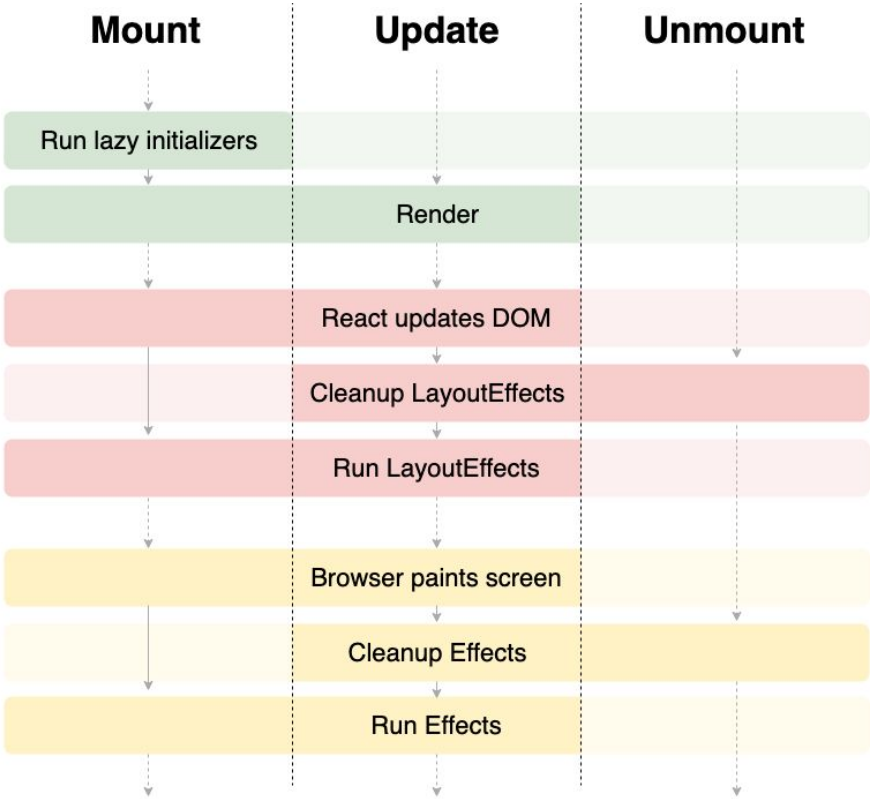
1. Finir l’exercice
2. On push sur github
3. On regarde des projets ensemble
4. On regarde comment faire

Les hooks React: Effect Hook

Exercice: Gérer un “effet” réseau

Temps: 10 minutes

Les hooks React: Petit récap



Les hooks React: Reference Hook

```jsx
function TextInputWithFocusButton() {
  const inputEl = useRef(null);

  const onButtonClick = () => {
    // `current` fait référence au champ textuel monté dans le DOM
    inputEl.current.focus();
  };
  return (
    <>
      <input ref={inputEl} type="text" />
      <button onClick={onButtonClick}>Donner le focus au champ</button>
    </>
  );
}
```

Les hooks React: Reference Hook

Exercice: Gérer une référence

Temps: 10 minutes

Les hooks React: Reducer Hook

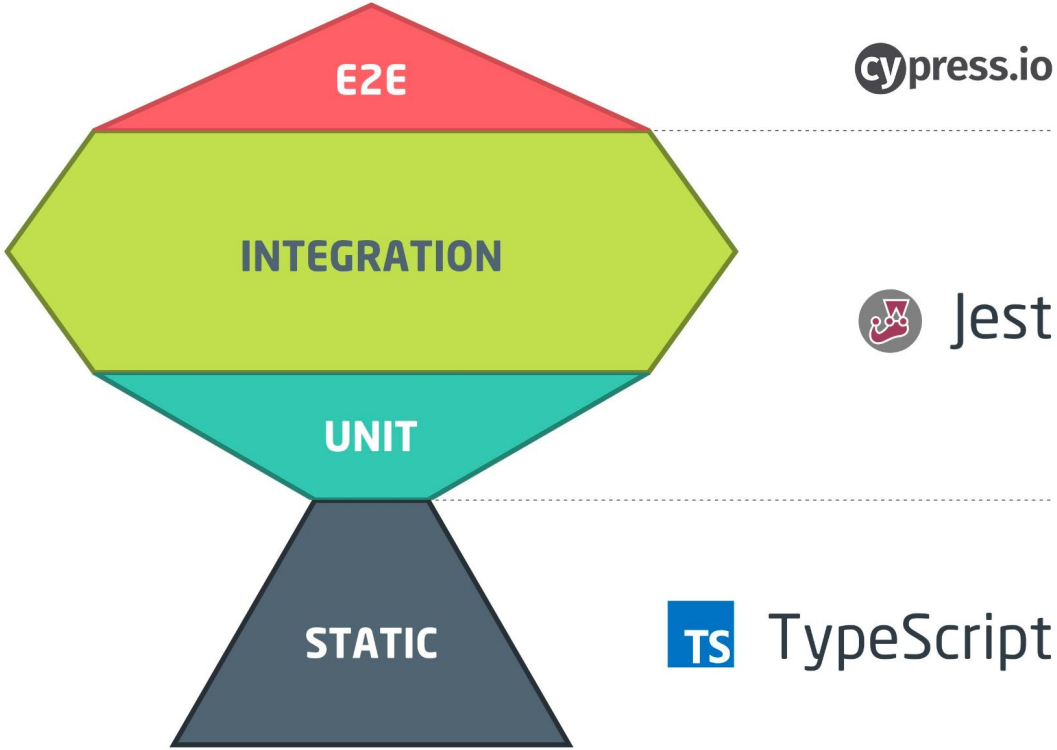
```
const initialState = {count: 0};

function reducer(state, action) {
  switch (action.type) {
    case 'increment':
      return {count: state.count + 1};
    case 'decrement':
      return {count: state.count - 1};
    default:
      throw new Error();
  }
}

function Counter() {
  const [state, dispatch] = useReducer(reducer, initialState);
  return (
    <>
      Total : {state.count}
      <button onClick={() => dispatch({type: 'decrement'})}>-</button>
      <button onClick={() => dispatch({type: 'increment'})}>+</button>
    </>
  );
}
```

Formation REACT

Tester

M2_nantes_GL@isen-ouest.yncrea.fr

Tester: La pyramide du contrôle



Tester: Static

Tutoriels recommandés :

https://www.totaltypescript.com/tutorials/beginners-typescript

Tester: Integration

Tutoriels recommandés :

https://github.com/kentcdodds/testing-react-apps/

Tester: E2E

Formation REACT

Outils

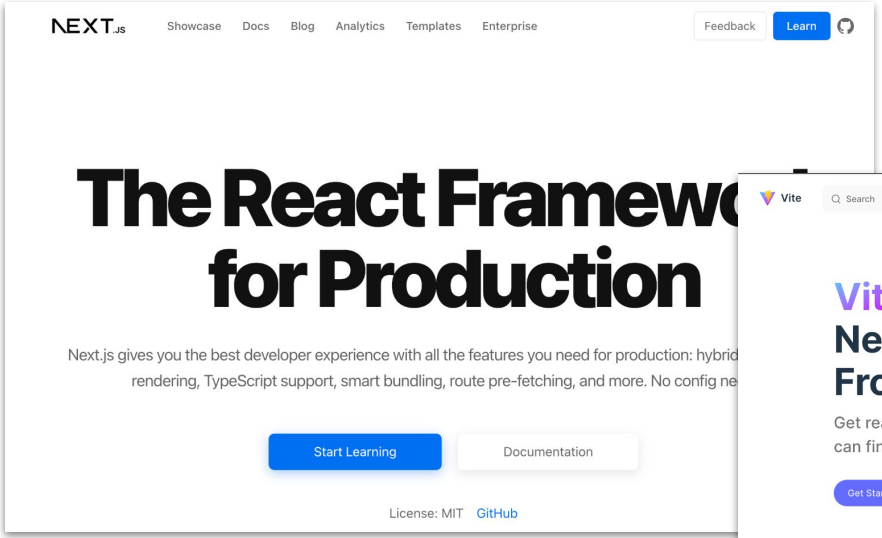
Outils: MSW

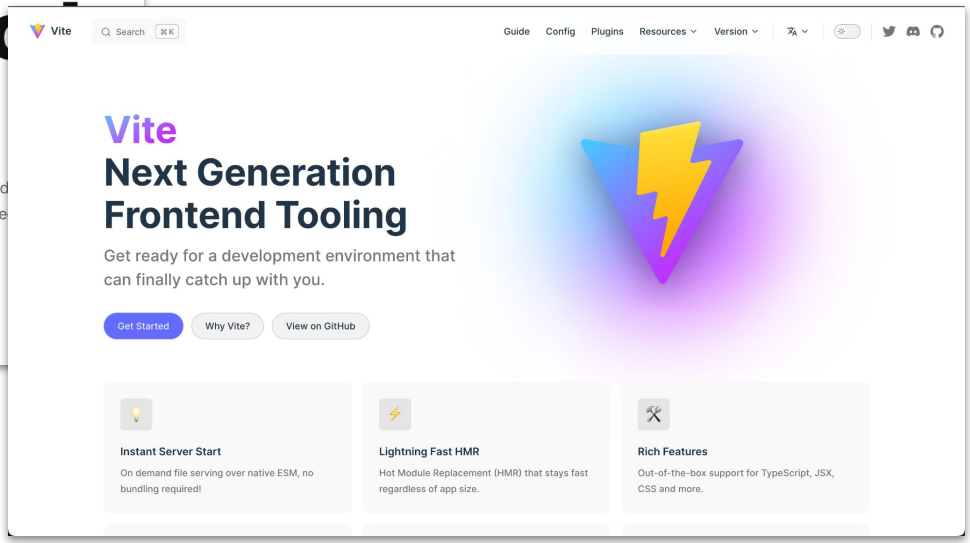
Outils: Tailwind

Formation REACT

Framework

Et après ?


NEXT.js
Showcase
Docs
Blog
Analytics
Templates
Enterprise
Feedback
Learn
The React Framewo
for Production
Next.js gives you the best developer experience with all the features you need for production: hybrid
rendering, TypeScript support, smart bundling, route pre-fetching, and more. No config ne
Start Learning
Documentation
License: MIT
GitHub


Vite
Search
⌘K
Guide
Config
Plugins
Resources
Version
Vite
Next Generation
Frontend Tooling
Get ready for a development environment that
can finally catch up with you.
Get Started
Why Vite?
View on GitHub
Instant Server Start
On demand file serving over native ESM, no
bundling required!
Lightning Fast HMR
Hot Module Replacement (HMR) that stays fast
regardless of app size.
Rich Features
Out-of-the-box support for TypeScript, JSX,
CSS and more.

Formation REACT

Partir en prod !

Partir en prod