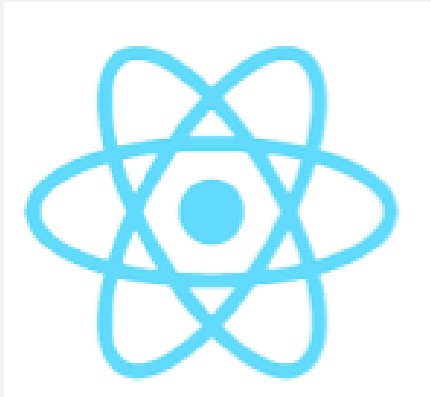


JSX/ES6



LE JSX

EXPRESSION JSX

React utilise JSX La syntaxe de JSX est la combinaison entre Javascript et HTML, permettant une écriture simple et flexible au développeur et intégrateur web. Les fichiers JSX a l'extension telle que jsx.

- Exemple 1

```
// index.js

import React from 'react';
import ReactDOM from 'react-dom';

const element1 = <h1>I love JSX</h1>;

ReactDOM.render(element1, document.getElementById('root'));
```

- Le JSX est une extension syntaxique de JavaScript
- la méthode render() renvoie la vue du composant

EXPRESSION JSX

- Exemple 2

```
// index.js
import React from 'react';
import ReactDOM from 'react-dom';

const element2 = React.createElement('h1', {}, 'No JSX');

ReactDOM.render(element2, document.getElementById('root'));
```

On peut utiliser n'importe quelle expression js dans les accolades jsx

Exemple : {2+2} {user.nom} {f(user)}

```
// index.js
import React from 'react';
import ReactDOM from 'react-dom';
const name = 'iano';
const element2 = <h1> Hello {name} </h1>;

ReactDOM.render(element2, document.getElementById('root'));
```

EXPRESSION JSX

- Exemple 3

```
// index.js
import React from 'react';
import ReactDOM from 'react-dom';

const element3 = (
  <ul>
    <li>Orange</li>
    <li>Banane</li>
    <li>Fraise</li>
  </ul>
);

ReactDOM.render(element3, document.getElementById('root'));
```

EXPRESSION JSX

- Exemple 4 : deux h1 utilise div

```
import React from 'react';
import ReactDOM from 'react-dom';

const element4 = (
  <div>
    <h1>Header 1</h1>
    <h1>Header 2</h1>
  </div>
);

ReactDOM.render(element4, document.getElementById('root'));
```

EXPRESSION JSX

En JSX, l'attribut `class='x'` deviendra `className='x'`

- Exemple 4 : `element6`, `element7` et `element8` sont identiques

```
import React from 'react';
import ReactDOM from 'react-dom';

const element5 = <img src={img.imgurl} />;
const element6 = (
  <h1 className='salut'> Bonjour </h1>
);

const element7 = React.createElement('h1', {className='salut'}, 'Bonjour');

const element8 = {type = 'h1',
  props: {className='salut', children: 'Bonjour'}};
```

EXEMPLE APPEL D'UNE FONCTION

Dans index.js ou app.js

```
function formatName(user) {  
  return user.firstName + ' ' + user.lastName;  
}  
  
const user = {  
  firstName: 'Kylia',  
  lastName: 'Mbappé'  
};  
  
const element = (  
  <h1>  
    Bonjour, {formatName(user)} !  
  </h1>  
)  
);  
  
ReactDOM.render(  
  element,  
  document.getElementById('root')  
);
```


MISE A JOUR D'UN ELEMENT

- Une fois crée, les éléments React sont immuables
- Jusqu'ici, cet exemple met à jour l'interface utilisateur en créant un nouvel élément et de le passer à ReactDOM.render()

```
function tic() {  
  const elem = (  
    <div>  
      <h2>Il est {new Date().toLocaleTimeString()}.</h2>  
    </div>  
  );  
  ReactDOM.render(elem, document.getElementById('root'));  
}  
  
setInterval(tic, 1000);
```

- setInterval() est une fonction de rappel a chaque seconde

RAPPEL MAP()

- La méthode `map()` passe sur chaque élément d'un tableau. Elle lui applique une fonction, et renvoie un nouveau tableau contenant les résultats de cette fonction appliquée sur chaque élément.
- Exemple : une fonction qui doublerait la valeur d'un élément

```
const numbers = [1, 2, 3, 4];  
const doubles = numbers.map(x => x * 2); // [2, 4, 6, 8]
```

- Exemple : renvoie un élément `` pour chaque entrée du tableau et affiche une liste à puces de nombres entre 1 et 4

```
const numbers = [1, 2, 3, 4];  
const listNumbers = numbers.map((number) =>  
  <li>{number}</li>  
);
```

```
ReactDOM.render(  
  <ul>{listNumbers}</ul>,  
  document.getElementById('root')  
);
```

KEY

- Une « clé » (key) est un attribut spécial qu'on doit inclure quand on crée une liste d'éléments.
- Les clés aident React à identifier quels éléments d'une liste ont changé, ont été ajoutés ou supprimés.
- On doit donner une clé à chaque élément dans un tableau afin d'apporter aux éléments une identité stable
- Exemple : Le meilleur moyen de choisir une clé est d'utiliser l'ID de notre donnée comme clé

```
const tabItems = tab.map((e) =>  
  <li key={e.id}>  
    {e.text}  
  </li>  
);
```

UNE FONCTION QUI RETOURNE JSX

- L'intérêt de JSX est qu'il permet de créer ses propres éléments HTML, qui seront vus comme des éléments React
- On va alors créer l'élément `<ListeElement>` qui représentera la liste `` contenant les éléments ``.

```
var elems = ["Element1","Element2","Element3","Element4","Element5"];

var ListeElements = function(){
  return <ul>
    {
      elems.map(function(elem,index){
        return <li key={index}>{elem}</li>;
      })
    }
    </ul>
  }
  ReactDOM.render(<ListeElements/>,document.getElementById("root"));
```

TRANSMISSION DES ATTRIBUTS DANS UN ELEMENT JSX

- le tableau `elems` pourrait être transmis dans l'attribut `elems` de l'élément JSX

```
var elems = ["Element1","Element2","Element3","Element4","Element5"];

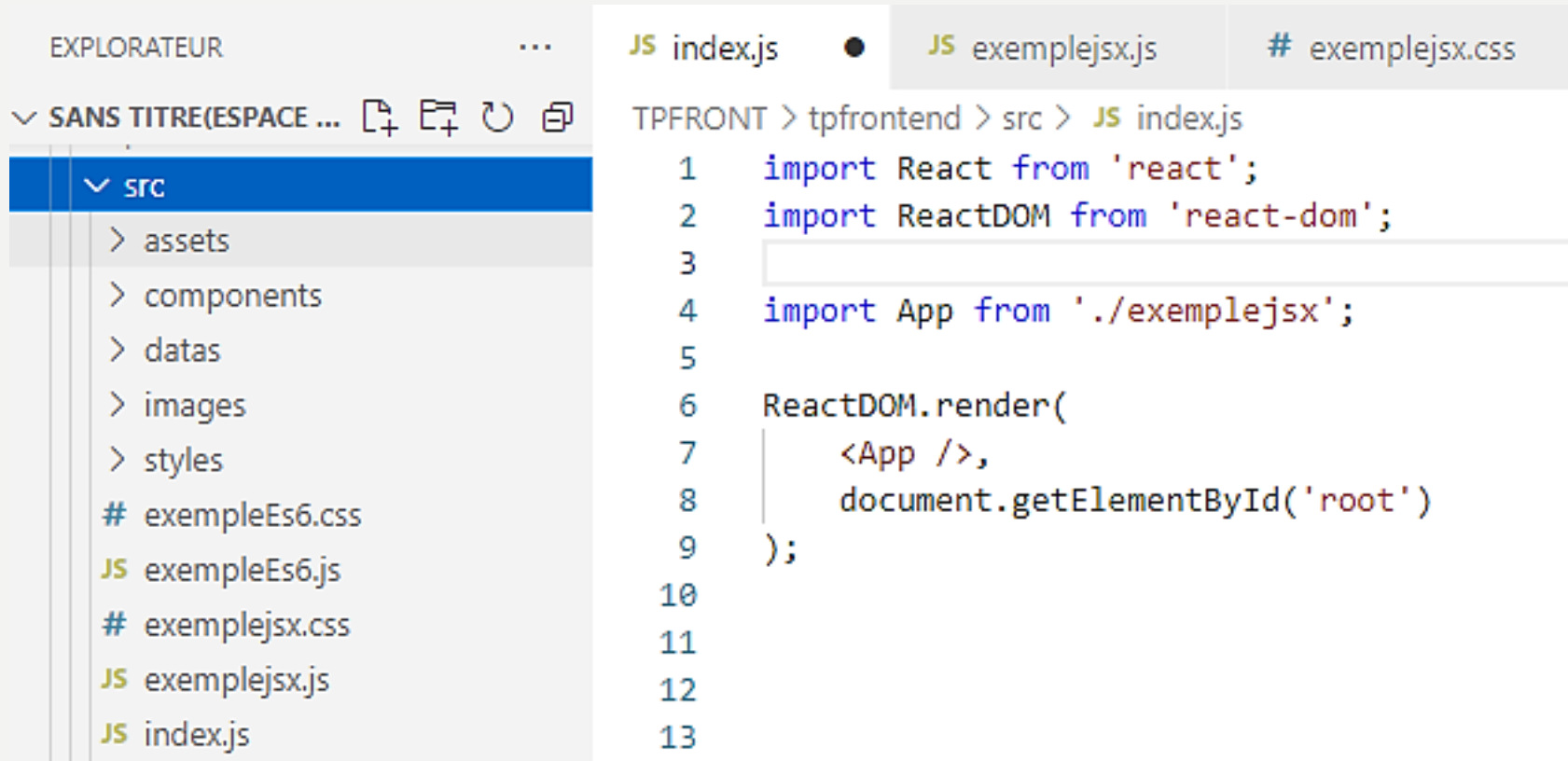
var ListeElements = function(props){
  return <ul>
    {
      elems.map(function(elem,index){
        return <li key={index}>{elem}</li>;
      })
    }
  </ul>
}

ReactDOM.render(<ListeElements elems={elems} />,document.getElementById("root"));
```

- Les attributs d'un élément défini par une fonction sont transmis dans l'objet `props` en paramètres de la fonction. Ainsi, pour accéder à l'attribut `elems` dans la fonction, on utilise `props.elems`.

IMPORTATION ÉLÉMENTS, IMAGE ET CSS

- Dans le même répertoire src, on aura alors trois fichiers : exemplejsx.js , exemplejsx.css et index.js



IMPORTATION ÉLÉMENTS, IMAGE ET CSS

-

JS index.js

JS exemplejsx.js ×

exemplejsx.css

TPFRONT > tpfrontend > src > JS exemplejsx.js > App > contenu

```
1 import React from 'react'
2 import iano from './images/iano.png';
3
4 import './exemplejsx.css'
5
6 function App() {
7   const coordonnee = (
8 >     <div>...
16   </div>
17 );
18
19   const entete = (
20     <div className="entete">
21       <img src={iano} />
22 >     <div className='coordonnee'> ...
24     </div>
25   </div>
26 );
27
```

JS index.js

JS exemplejsx.js ×

exemplejsx.css

TPFRONT > tpfrontend > src > JS exemplejsx.js > default

```
27
28 const contenu = (
29 >   <div className='contenu'>...
46   </div>
47 );
48
49 const corps= (
50 >   <div>...
53   </div>
54 );
55
56 return (
57   <div className='app'>{corps}</div>
58 );
59 }
60
61 export default App;
```

EXERCICE :

• RÉALISER VOTRE CV

localhost:3001/%7Bcontenu2%7D

46 Gmail Maps Traduire YouTube Boot-Loader v2.0 Mon Drive - Google Co



CURRICULUM VITAE

- Nom : RANDRIANASOA
- Prenoms : Lalaridimby Iano
- Telephone : 034 05 299 00
- email : iano.anas@gmail.com

DIPLOMES

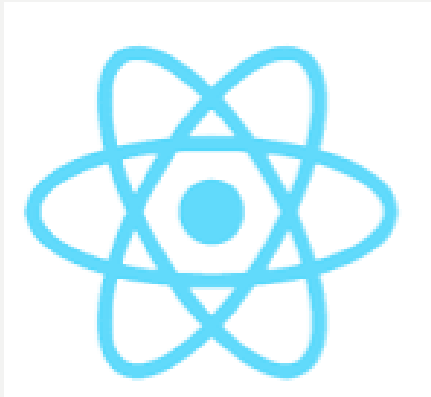
- Doctorant en Electronique-Informatique Université Antananarivo
- 2016 : Ingenieur

EXPERIENCES

- Formateur
- Developpeur
- Consultant

COMPETENCES

- Programmation et Réseau
- Circuits Electroniques



LA CLASSE ES6

EXPRESSION ES6

- Exemple :

super() se réfère à la classe parent.

```
class Car {
  constructor(name) {
    |   this.b = name;
  }

  present() {
    |   return 'Ma Voiture : ' + this.b;
  }
}

class Model extends Car {
  constructor(name, mod) {
    |   super(name); // appel le constructeur parent
    |   this.model = mod;
  }
  show() {
    |   return this.present() + ', modele : ' + this.model;
  }
}

mcar = new Model("Ford", "Mustang");
mcar.show();
```

EXPRESSION ES6

- ***Un composant React peut être défini comme une classe ES6 qui étend la classe `React.Component` base.***
- Sous sa forme minimale, un composant doit définir une méthode de render spécifiant le rendu du composant dans le DOM.
- **Exemple :** `src/exemple1.js`

```
import React from 'react'

class HelloWorld extends React.Component {
  render() {
    return <h1>Hello World</h1>
  }
}

export default HelloWorld
```

- **`src/app.js` ou `index.js`**

```
import React from 'react';
import ReactDOM from 'react-dom';
import HelloWorld from './exemple1.js';

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

FONCTION FLECHEE

-

```
hello = function() {  
  return "Bonjour";  
}
```



```
hello = () => {  
  return "Bonjour";  
}
```

```
hello = function(val){  
  return "Bonjour " + val;  
}
```



```
hello = (val) => {  
  return "Bonjour " + val;  
}
```

IMPORTATION ÉLÉMENTS, IMAGE ET CSS

JS index.js

JS exemplejsx.js

exemplejsx.css

JS exempleEs6.js

TPFRONT > tpfrontend > src > JS exempleEs6.js > App > render > coordonnee

```
1 import React,{Component} from 'react'
2 import iano from './images/iano.png';
3
4 import './exempleEs6.css'
5
6 class App extends Component{
7   constructor(props){
8     super();
9     this.data = {  nom : "RANDRIANASOA",
10                   prenoms : "Lalaridimby Iano",
11                   tel : "034 05 299 00",
12                   email : "iano.anas@gmail.com",
13                   photo : iano };
14   }
15   render(){
16     const coordonnee = (
17 >     <div>...
25     </div>
26   );
27
```

JS index.js

JS exemplejsx.js

exemplejsx.css

JS ex

TPFRONT > tpfrontend > src > JS exempleEs6.js > App > render >


```
27
28   const entete = (
29     <div className="entete">
30       <img src={this.data.photo} />
31       <div className='coordonnee'>
32         {coordonnee}
33       </div>
34     </div>
35   );
36   const contenu = (
37 >   <div className='contenu'>...
54   </div>
55   );
56
57   const corps = (
58 >   <div>...
61   </div>
62   );
63
64   return (
65     <div className='app'>{corps}</div>
66   );
67 }
68 }
69 export default App;
```

APPEL D'UNE FONCTION

En JSX, l'attribut onclick='x' deviendra **onClick='x'**

- ***Exemple dans la fonction rendre() d'une classe***


```
const menu = (  
  <div className='menu'>  
    <button className="sstitre" onClick={this.fct} value={1}>DIPLOMES</button>  
    <button className="sstitre" onClick={this.fct} value={2}>EXPERIENCES</button>  
    <button className="sstitre" onClick={this.fct} value={3}>COMPETENCES</button>  
  </div>  
);  
  
return (  
  <div className='app'>  
    <div>{entete}</div>  
    <div>{menu}</div>  
    <div id='demo'></div>  
  </div>  
);
```



APPEL D'UNE FONCTION

- *Exemple d'une méthode fct*

```
fct = (param) => {  
  console.log(param.target.value)  
  this.diplomes = "<div><ul><li>" + this.diplome.d1  
    + "</li><li>" + this.diplome.d2  
    + "</li></ul></div>";  
  this.experiences = "<div><ul><li> " + this.experience.e1  
    + " </li><li>" + this.experience.e2  
    + " </li><li>" + this.experience.e3  
    + " </li></ul></div>";  
  this.competences = "<div><ul><li>" + this.competence.c1  
    + " </li><li>" + this.competence.c2  
    + " </li></ul></div>";  
  if(param.target.value == 1)  
    document.getElementById('demo').innerHTML = this.diplomes;  
  else if(param.target.value == 2)  
    document.getElementById('demo').innerHTML = this.experiences;  
  else  
    document.getElementById('demo').innerHTML = this.competences;  
}
```




APPEL D'UNE FONCTION

- *Exemple d'un constructeur*

```
constructor(props){  
  super();  
  this.data = {  
    nom : "RANDRIANASOA",  
    prenom : "Lalaridimby Iano",  
    tel : "034 05 299 00",  
    email : "iano.anas@gmail.com",  
    photo : iano  
  }  
  this.diplome = {  
    d1 : "Doctorant en Electronique-Informatique Université Antananarivo",  
    d2 : "2016 : Ingenieur"  
  }  
  this.experience = {  
    e1 : "Formateur" ,  
    e2 : "Developpeur",  
    e3 : "Consultant"  
  }  
  this.competence = {  
    c1 : "Programmation et Réseau ",  
    c2 : "Circuits Electroniques"  
  }  
}
```


EXERCICE

		<h2>CURRICULUM VITAE</h2> <ul style="list-style-type: none">• Nom : RANDRIANASOA• Prenoms : Lalaridimby Iano• Telephone : 034 05 299 00• email : iano.anas@gmail.com
DIPLOMES	EXPERIENCES	COMPETENCES

- Les diplômes s'affichent en cliquant sur diplômes
- De même pour les autres