
Corrigé Atelier N°3 Formulaires, routage

Dans cet atelier, nous allons comprendre le fonctionnement des formulaires. En HTML, les éléments de Form tels que <input>, <textarea>, <select> gèrent leurs statuts eux-mêmes. Leur statut peut être modifié par l'impaction de l'utilisateur.

Si on souhaite contrôler le comportement et les données de Form par React, on doit créer une relation bilatérale entre les valeurs des éléments de Form et le statut du React.

Objectifs :

- Comprendre le comportement des formulaires avec React.
- Mettre en place des champs de formulaire et récupérer les valeurs saisies.
- Ajouter des contrôles pour les valeurs saisies.
- Comprendre le fonctionnement du routage.

Exercices à faire :

Exercice 1 :

1. On voudrait écrire un script pour **le composant de type classe** intitulé « SimpleForm » qui permet de saisir le nom de l'utilisateur. Le state correspondant est initialisé à vide. Une fois on clique sur le bouton submit, la valeur saisie est affichée dans une alerte, comme le montre la capture ci-dessous.

Solution :

SimpleForm.js :

```
import React from 'react';

class SimpleForm extends React.Component {
  constructor(props) {
    super(props);

    this.state = {
      fullName: ""
    };
  }

  handleSubmitForm = (event) => {
    alert("Full Name: " + this.state.fullName);
    event.preventDefault();
  }

  handleChange = (event) => {
    var value = event.target.value;
```

```

    this.setState({
      fullName: value
    });
  }

  render() {
    return (
      <form onSubmit={event => this.handleSubmitForm(event)}>
        <label>
          Full Name:
          <input
            type="text"
            value={this.state.fullName}
            onChange={event => this.handleChange(event)}
          />
        </label>
        <input type="submit" value="Submit" />
        <p>{this.state.fullName}</p>
      </form>
    );
  }
}
export default SimpleForm;

```

App.js :

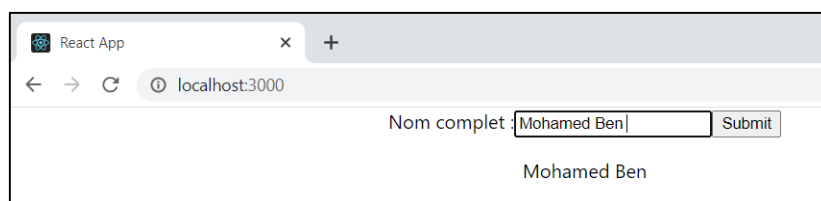
```

import SimpleForm from './SimpleForm';
import './App.css';
function App() {
  return (
    <div className="App">
      <SimpleForm/>
    </div>
  );
}
export default App;

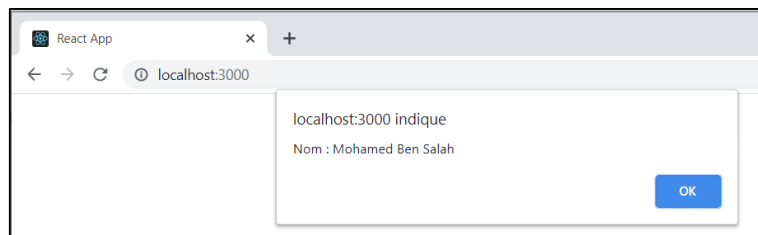
```

Résultat obtenu :

Saisi dans le champ :



Clic sur Submit :



2. Remplacer le champ par un élément textarea dans le formulaire pour ajouter un champ appelé description. Penser à élaborer le state correspondant.

Solution :

L'élément textarea dans React est légèrement différent du HTML ordinaire. En HTML, la valeur d'une zone de texte était le texte entre la balise de début `<textarea>` et la balise de fin `</textarea>`, dans React la valeur d'une zone de texte est placée dans un attribut de value :

```
<textarea value={this.state.description} />
```

3. Remplacer le champ par une liste déroulante (une zone de sélection) qui proposera une liste de villes.

Solution :

Une liste déroulante, ou une zone de sélection, dans React est également un peu différente du HTML. Dans React, la valeur sélectionnée est définie avec un attribut value dans la balise select :

```
<select value={this.state.mytown}>
  <option value="Sfax">Sfax</option>
  <option value="Tunis">Tunis</option>
  <option value="Sousse">Sousse</option>
</select>
```

Exercice 2 :

1. On voudrait écrire un script pour le **composant de type classe** intitulé « SimpleForm » qui permet de saisir le nom de l'utilisateur ainsi que son âge. On veut contrôler les valeurs de plusieurs champs de saisi en ajoutant un attribut « name » à chaque élément. Lorsqu'on initialise l'état dans le constructeur, on utilise les noms des champs.
 - a. Accéder aux champs du gestionnaire d'événements, en utilisant la syntaxe **event.target.name** en plus de **event.target.value**.
 - b. Mettre à jour l'état dans la méthode **this.setState**, en utilisant des **crochets** [notation entre crochets] autour du nom de la propriété ([nam]: val).

Solution :

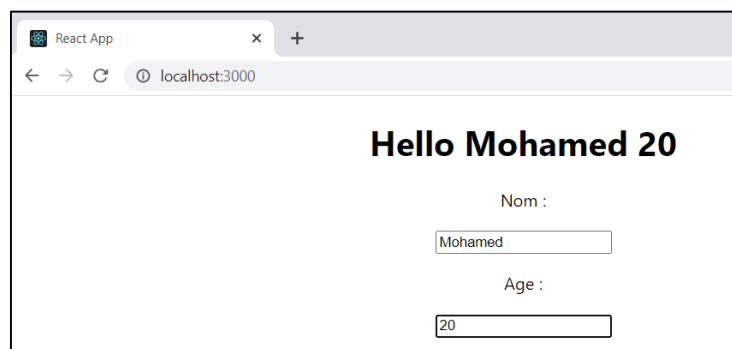
```
import React from 'react';
class SimpleForm extends React.Component {
  constructor(props) {
```

```

    super(props);
    this.state = {
      username: '',
      age: null,
    };
  }
  myChangeHandler = (event) => {
    let nam = event.target.name;
    let val = event.target.value;
    this.setState({[nam]: val});
  }
  render() {
    return (
      <form>
        <h1>Hello {this.state.username} {this.state.age}</h1>
        <p>Nom :</p>
        <input
          type='text'
          name='username'
          onChange={this.myChangeHandler}
        />
        <p>Age :</p>
        <input
          type='text'
          name='age'
          onChange={this.myChangeHandler}
        />
      </form>
    );
  }
}
export default SimpleForm;

```

Résultat obtenu :



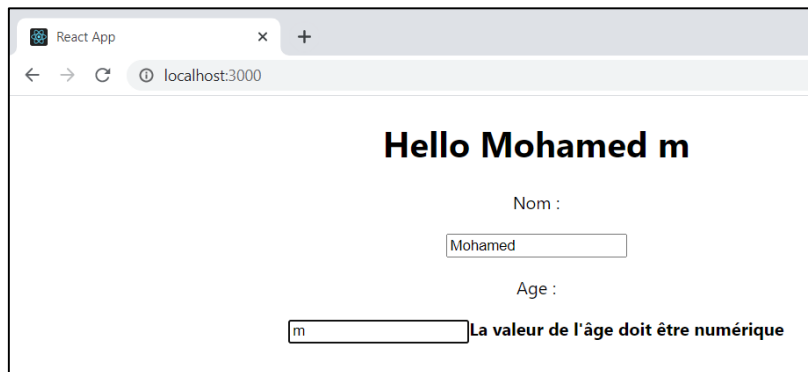
2. On veut créer un message d'erreur vide par défaut, mais qui affiche l'erreur lorsque l'utilisateur entre des valeurs erronées. A ce propos, il faut créer un state « errormessage » initialisé à vide. Cet état change lorsqu'on rencontre une erreur de saisie. Cette dernière pourrait être la valeur non numérique de l'âge.

```
import React from 'react';

class SimpleForm extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      username: '',
      age: null,
      errormessage: ''
    };
  }
  myChangeHandler = (event) => {
    let nam = event.target.name;
    let val = event.target.value;
    let err = '';
    if (nam === "age") {
      if (val !== "" && !Number(val)) {
        err = <strong>La valeur de l'âge doit être numérique</strong>;
      }
    }
    this.setState({errormessage: err});
    this.setState({[nam]: val});
  }
  render() {
    return (
      <form>
        <h1>Hello {this.state.username} {this.state.age}</h1>
        <p>Nom :</p>
        <input
          type='text'
          name='username'
          onChange={this.myChangeHandler}
        />
        <p>Age :</p>
        <input
          type='text'
          name='age'
          onChange={this.myChangeHandler}
        />
        {this.state.errormessage}
      </form>
    );
  }
}
```

```
}  
export default SimpleForm;
```

Résultat obtenu :



React App

localhost:3000

Hello Mohamed m

Nom :

Age :

 La valeur de l'âge doit être numérique

Exercice 3 :

1. On se propose de créer un composant fonctionnel intitulé « departements ». Ce composant contient un champ de formulaire pour saisir un nouveau département permettant d'alimenter ce tableau :

```
const [departs, setDeparts]=useState([  
  {"id":1, "nom": "Techniques d'Informatique"},  
  {"id":2, "nom": "Génie Mécanique"},  
  {"id":3, "nom": "Génie Civill"},  
]);
```

Solution :

App.js

```
import Departements from './Departements';  
  
export default function App() {  
  return (  
    <>  
      <Departements />  
    </>  
  )  
}
```

Departements.js

```
import {useState } from 'react';  
const Departements=()=> {  
  const [depart, setSepar] = useState('');
```

```

const [departs, setDeparts]=useState([
  {"id":1,"nom":"Techniques d'Informatique"},
  {"id":2,"nom":"Génie Mécanique"},
  {"id":3,"nom":"Génie Civill"},
]);

const handleSubmitForm = (event) => {
  event.preventDefault();
  let newDepart={
    id:[...departs].pop().id+1,
    nom:depart
  }
  setDeparts([...departs,newDepart])
}

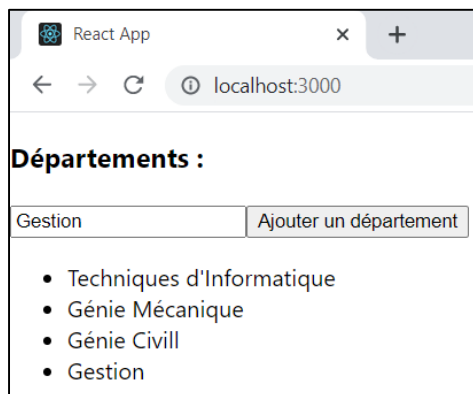
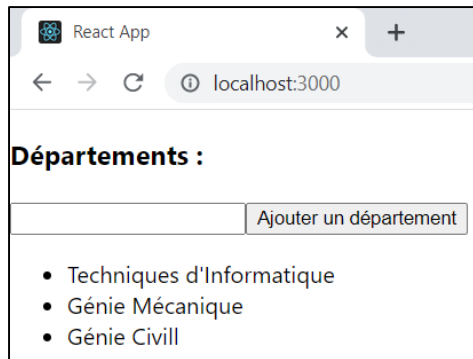
const handleChange = (event) => {
  var value = event.target.value;
  setSepart(value);
}

return (
  <>
  <form onSubmit={event => handleSubmitForm(event)}>
    <h3>Départements :</h3>

    <input
      type='text'
      name='depart'
      value={depart}
      onChange={event => handleChange(event)}
    />
    <input type="submit" value="Ajouter un département" />
  </form>
  <ul>
    { departs.map((val,ind)=>{
      return(<li key={ind}>{val.nom}</li>)
    })}
  </ul>
</>
)
}
export default Departements;

```

Résultat obtenu :



2. Ajouter devant chaque département le bouton (X) : en cliquant dessus, on pourrait supprimer le département concerné.

Solution :

Departements.js

```
import {useState } from 'react';
const Departements=()=> {
  const [depart,setDepart]=useState('');
  const [departs,setDeparts]=useState([
    {"id":1,"nom":"Comptabilité"},
    {"id":2,"nom":"Finance"},
    {"id":3,"nom":"Commercial"},
  ]);

  const handleSubmitForm = (event) => {
    event.preventDefault();
    let newDepart={
      id:[...departs].pop().id+1,
      nom:depart
    }
    setDeparts([...departs,newDepart])
  }

  const handleChange = (event) => {
```



```

    var value = event.target.value;
    setDepart(value);
  }

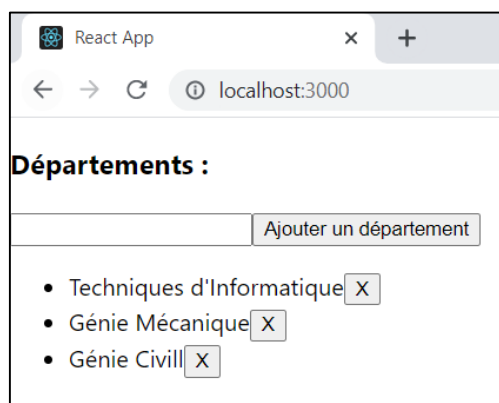
  const deleteDepart=(d)=>{
    let index = departs.indexOf(d);
    let listDeparts=[...departs];
    listDeparts.splice(index,1);
    setDeparts(listDeparts);
  }

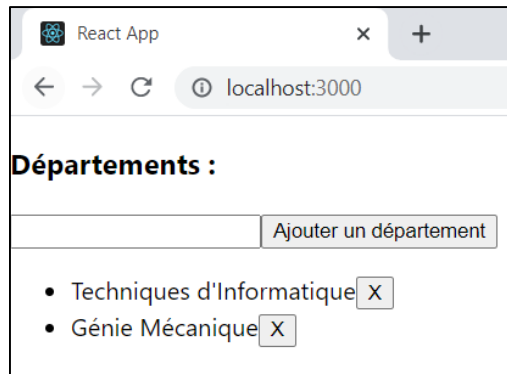
  return (
    <>
    <form onSubmit={event => handleSubmitForm(event)}>
      <h3>Départements :</h3>

      <input
        type='text'
        name='depart'
        value={depart}
        onChange={event => handleChange(event)}
      />
      <input type="submit" value="Ajouter un département" />
    </form>
    <ul>
      { departs.map((val,ind)=>{
        return(<li key={ind}>>{val.nom}<button onClick={()=>deleteDepart
(val.id)}>X</button></li>)
      })}
    </ul>
  </>
)
}
export default Departements;

```

Résultat obtenu :





3. Restructurer maintenant votre code en subdivisant le composant principal (App.js) en plusieurs sous **composants fonctionnels** indépendants comme suit :

- Accueil.js : affiche le message de bienvenue et a photo de l'ISET.
- A propos.js : affiche les informations générales sur l'ISET.
- Departements : formulaire d'ajout du nouveau département et affichage des différents départements avec possibilité de suppression.

Ajouter un menu (barre de navigation) incluant trois liens afin de pouvoir naviguer entre les différents composants.

Penser à améliorer l'affichage en intégrant Bootstrap : `npm install --save bootstrap`

Solution :

App.js

```
import {Route, Switch, Link, BrowserRouter as Router} from 'react-router-dom';
import 'bootstrap/dist/css/bootstrap.min.css';
import Accueil from './Accueil';
import A propos from './Apropos';
import Departements from './Departements';

export default function App() {
  return (
    <Router>
    <nav className="navbar navbar-expand-lg navbar-dark bg-dark">
      <ul className="navbar-nav">
        <li><Link className="nav-link" to="/">Accueil</Link></li>
        <li><Link className="nav-link" to="/apropos">A propos</Link></li>
        <li><Link className="nav-
link" to="/departements">Départements</Link></li>
      </ul>
    </nav>
    <div className="m-4">
      <Switch>
        <Route exact path="/" component={Accueil}></Route>
        <Route path="/apropos" component={Apropos}></Route>
        <Route path="/departements" component={Departements}></Route>
      </Switch>
    </div>
  )
}
```

```
    </Switch>
  </div>
</Router>
)
}
```

Accueil.js

```
const Accueil = () => {

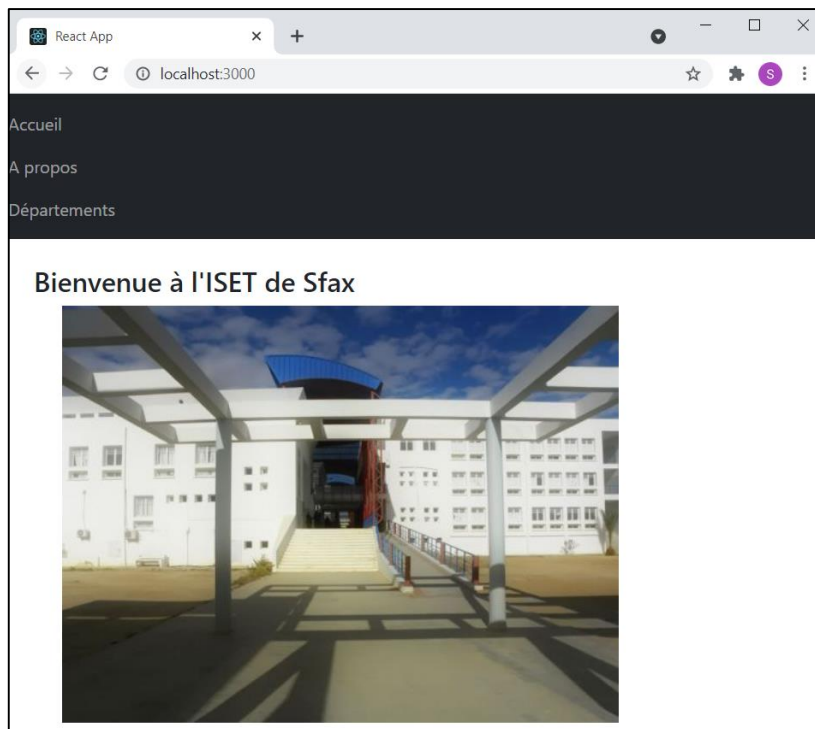
  return (
    <>
      <h3>Bienvenue à l'ISSET de Sfax</h3>
      
    </>
  )
}
export default Accueil;
```

Apropos.js

```
const Apropos = () => {

  return (
    <>
      <h3>L'Institut Supérieur des Etudes Technologiques de Sfax est
      un établissement public doté de la personnalité civile
      et de l'autonomie financière.</h3>
    </>
  )
}
export default Apropos;
```

Résultat obtenu :



4. Modifier le code pour les pages « App.js » et « Apropas.js » de telle sorte qu'on pourrait envoyer le nom de la ville (Sfax) comme paramètre vers « Apropas ».

Solution :

App.js

```
import {Route, Switch, Link, BrowserRouter as Router} from 'react-router-dom';
import 'bootstrap/dist/css/bootstrap.min.css';
import Accueil from './Accueil';
import Apropas from './Apropas';
import Departements from './Departements';

export default function App() {
  const ville="Sfax";
  return (
    <Router>
    <nav className="navbar navbar-expand-lg navbar-dark bg-dark">
      <ul className="navbar-nav">
        <li><Link className="nav-link" to="/">Accueil</Link></li>
        <li><Link className="nav-link" to={`/${apropas/${ville}}`}>A propos</Link></li>
        <li><Link className="nav-link" to="/departements">Départements</Link></li>
      </ul>
    </nav>
    <div className="m-4">
      <Switch>
        <Route exact path="/" component={Accueil}></Route>
```

```

    <Route path="/apropos/:ville" component={Apropos}></Route>
    <Route path="/departements" component={Departements}></Route>
  </Switch>
</div>
</Router>
)
}

```

Apropos.js

```

const Apropos = ({match}) => {

const ville=match.params.ville;

  return (
    <>
      <h2>ISET de {ville}</h2>
      <h3>L'Institut Supérieur des Etudes Technologiques de {ville} est
      un établissement public doté de la personnalité civile
      et de l'autonomie financière.</h3>
    </>
  )
}
export default Apropos;

```

Résultat obtenu :

