

Chapitre 4 : Les formulaires



Introduction

- La gestion des formulaires concerne la façon dont on gère les données lorsqu'elles changent de valeur ou sont soumises.
- En HTML, les données du formulaire sont généralement gérées par le DOM,
- Dans React, les données de formulaire sont généralement gérées par les composants.
- Lorsque les données sont gérées par les composants, toutes les données sont stockées dans le composant state.
- On peut contrôler les modifications en ajoutant des gestionnaires d'événements dans l'attribut onChange.

Mécanisme d'un formulaire React 1/2

- ▶ La soumission de formulaire HTML fonctionne différemment lors de son implémentation dans un composant React.js.
- ▶ Normalement, le navigateur rendrait le HTML et, en fonction de l'action, soumettrait automatiquement les données du formulaire en fonction de l'attribut de nom de chaque élément.
- ▶ Bien que ce comportement par défaut fonctionne toujours dans React.js, il est fortement conseillé de soumettre par programme un formulaire en fournissant vos propres contrôles personnalisés sur la façon dont les données sont traitées par un composant.

Mécanisme d'un formulaire React 2/2

- ▶ La soumission de formulaire dans React.js est assez différente des soumissions de formulaire habituelles en HTML.
- ▶ React.js donne un contrôle total des valeurs qu'on passe à l'élément exploitable suivant, permettant de formater des structures de données plus complexes.
- ▶ Toutes les valeurs sont conservées par l'objet d'état d'un composant et sont propagées dans tous les éléments rendus, tels que ceux d'une entrée.
- ▶ Des gestionnaires d'événements sont également nécessaires pour indiquer à l'application comment réagir à certains changements, tels que la modification d'une entrée ou le clic sur un bouton.

Ajouter un formulaire avec React

```
import React,{ Component } from "react";
class MyForms extends Component{
  render()
  {
    return(
      <form>
        <h1>Donner Votre nom:</h1>
        <input type="text"/>

      </form>
    );
  }
}
export default MyForms;
```

Ajouter un formulaire avec React

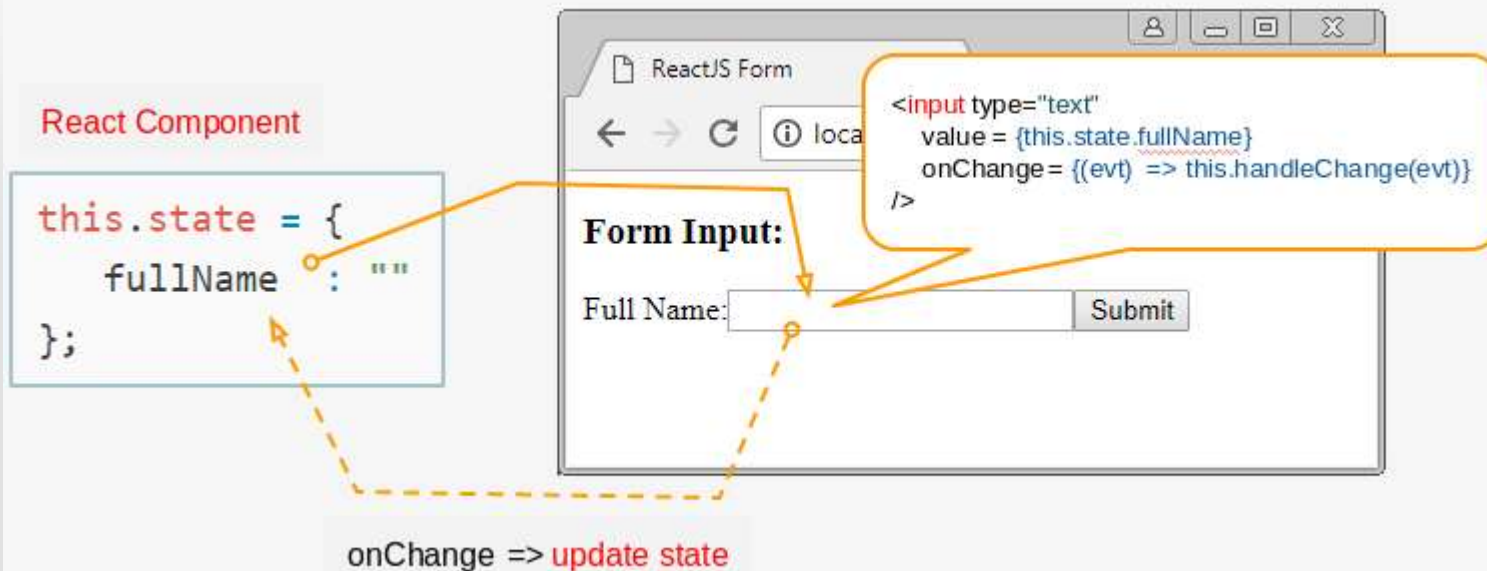
```
import './App.css';
import IdentForms from './Components/MyForms'
function App() {
  return (
    <div className="App">
      <IdentForms/>
    </div>
  );
}

export default App;
```



Ajouter un formulaire avec React

Ci-dessous, l'exemple simple avec l'élément `<input>`. La valeur de cet élément est assignée de `this.state.fullName`. Lorsque les gens changent la valeur de `<input>`, cette valeur a besoin d'être mise à jour pour `this.state.fullName` via la méthode `setState()`.



Formulaires et gestionnaires d'événements ReactJS

```
import React, { Component } from "react";
class MyForms extends Component {
  constructor(props) {
    super(props);
    this.state =
      { nom: '',
        prenom: ''
      };
  }
  changerStateNom = (event) => {
    this.setState({nom: event.target.value});
  }
  changerStatePrenom = (event) => {
    this.setState({prenom: event.target.value});
  }
}
```


Formulaires et gestionnaires d'événements ReactJS

```
render()  
{  
  return(  
    <form>  
      nom:  
      <input type="text"  
        onChange={this.changerStateNom}/>  
      prénom:  
      <input type="text"  
        onChange={this.changerStatePrenom}/>  
      <h1>Bonjour {this.state.nom} {this.state.prenom} </h1>  
    </form>  
  );  
}  
}  
export default MyForms;
```

Soumettre un formulaire avec ReactJS

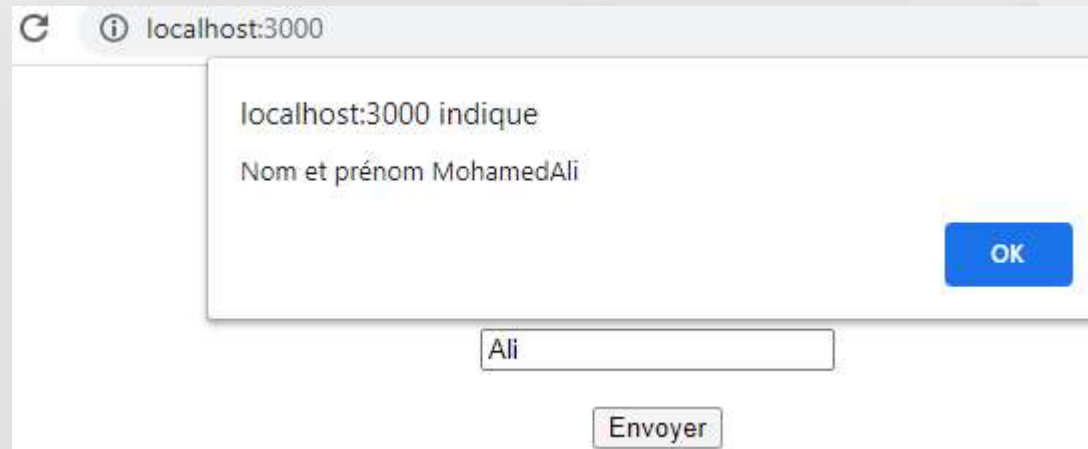
```
import React, { Component } from "react";
class MyForms extends Component {
  constructor(props) {
    super(props);
    this.state =
      { nom: '',
        prenom: ''
      };
  }
  changerStateNom = (event) => {
    this.setState({nom: event.target.value});
  }
  changerStatePrenom = (event) => {
    this.setState({prenom: event.target.value});
  }
}
```

Soumettre un formulaire ReactJS

```
mySubmitHandler = (event) => {  
  event.preventDefault();  
  alert("Nom et prénom " + this.state.nom + this.state.prenom);  
}  
render()  
{  
  return(  
    <form onSubmit={this.mySubmitHandler}>  
      <p>nom:</p>  
      <input type="text"  
        onChange={this.changerStateNom}/>  
      <p>prénom:</p>  
      <input type="text"  
        onChange={this.changerStatePrenom}/>  
      <p> <input type='submit' /></p>  
    </form>  
  )  
}
```

Soumettre un formulaire ReactJS

```
    );  
  }  
}  
export default MyForms;
```



The screenshot shows a web browser window with the address bar displaying 'localhost:3000'. A modal dialog box is open, displaying the message: 'localhost:3000 indique' followed by 'Nom et prénom MohamedAli'. Below the dialog, there is a text input field containing the text 'Ali' and a button labeled 'Envoyer'.

Plusieurs champs dans un formulaire React 1/4

- Vous pouvez contrôler les valeurs de plusieurs champs d'entrée en ajoutant un attribut de nom à chaque élément.

Lorsque vous initialisez l'état dans le constructeur, utilisez les «names» des champs.

- Pour accéder aux champs du gestionnaire d'événements, utilisez la syntaxe `event.target.name` et `event.target.value`.

- Pour mettre à jour l'état dans la méthode `this.setState`, utilisez des crochets [notation entre crochets] autour du nom de la propriété.

Plusieurs champs dans un formulaire React 2/4

```
import React,{ Component } from "react";
class MyForms extends Component{
  constructor(props){
    super(props);
    this.state =
      { nom: '',
        age: ''
      };
  }
  changerState = (event) => {
    let namefields=event.target.name;
    let valfields=event.target.value;
    this.setState({[namefields]: valfields});
  }
}
```

Plusieurs champs dans un formulaire React 3/4

```
render()
{
  return(
    <form >
      <p>nom:</p>
      <input type="text"
        name="nom"
        onChange={this.changerState}/>
      <p>age:</p>
      <input type="text"
        name="age"
        onChange={this.changerState}/>
      <h1>Bonjour {this.state.nom} age: {this.state.age}</h1>
    </form>
  )
}
```

Plusieurs champs dans un formulaire React 4/4

```
    );  
  }  
}  
export default MyForms;
```

nom:

age:

Bonjour Turki age: 28

TextArea

Une zone de texte simple avec du contenu initialisé dans le constructeur:

```
import React, { Component } from "react";
class MyForms extends Component {
  constructor(props) {
    super(props);
    this.state = {
      description: 'le type de ce réseaux est 5G'
    };
  }
}
```

TextArea

```
render()    {  
    return(  
        <form >  
            <textarea value={this.state.description} />  
        </form>  
    );  
}  
}  
export default MyForms;
```

Select

Sélectionner Une liste déroulante, ou une zone de sélection, dans React est également un peu différente du HTML.

Exemple: Une boîte de sélection simple, où la valeur sélectionnée "Volvo" est initialisée dans le constructeur:

```
import React, { Component } from "react";
class MyForms extends Component {
  constructor(props) {
    super(props);
    this.state = {
      mavoiture: 'Volvo'
    };
  }
}
```

Select

```
render() {  
  return(  
    <form >  
      <onChange={this.handleChange} value={this.state.mavoiture}>  
        <option value="Ford">Ford</option>  
        <option value="Volvo">Volvo</option>  
        <option value="Fiat">Fiat</option>  
      </select>  
    </form>  
  );  
}  
}  
export default MyForms;
```