TP React 8
Formulaires contrôlés

Objectifs pédagogiques

À la fin de ce TP, vous serez capable de :
- comprendre le principe des formulaires contrôlés
- lier des champs de formulaire au state
- gérer `onChange` et `onSubmit`
- récupérer les données saisies par l'utilisateur

Introduction

Dans React, un formulaire est dit **contrôlé** lorsque les valeurs des champs sont pilotées par le state du composant.

> Le state devient la source unique de vérité des données saisies.

1 Champ de saisie simple

Exemple avec un champ texte :

```
import { useState } from "react";

function App() {
  const [nom, setNom] = useState("");

  return (
    <div>
      <input
        type="text"
        value={nom}
        onChange={(e) => setNom(e.target.value)}
      />
      <p>Nom saisi : {nom}</p>
    </div>
  );
}

export default App;
```

Principe :
- `value` est liée au state
- `onChange` met à jour le state

2 Formulaire avec soumission

Ajoutons un bouton de validation :

```
import { useState } from "react";

function App() {
  const [email, setEmail] = useState("");

  const handleSubmit = (e) => {
    e.preventDefault();
    console.log("Email envoyé :", email);
  };

  return (
    <form onSubmit={handleSubmit}>
      <input
        type="email"
        value={email}
        onChange={(e) => setEmail(e.target.value)}
      />
      <button type="submit">Envoyer</button>
    </form>
  );
}

export default App;
```

`preventDefault()` empêche le rechargement de la page au moment de l'envoi du formulaire.

3 Plusieurs champs avec un objet

Il est courant de gérer plusieurs champs dans un seul state objet.

```
import { useState } from "react";

function App() {
  const [formData, setFormData] = useState({
    nom: "",
    age: ""
  });

  const handleChange = (e) => {
    setFormData({
      ...formData,
      [e.target.name]: e.target.value
    });
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    console.log(formData);
  };

  return (
    <form onSubmit={handleSubmit}>
      <input
        name="nom"
        value={formData.nom}
        onChange={handleChange}
        placeholder="Nom"
      />

      <input
        name="age"
        value={formData.age}
        onChange={handleChange}
        placeholder="Âge"
      />

      <button type="submit">Valider</button>
    </form>
  );
}

export default App;
```

4 Validation simple

Exemple : afficher un message si le champ email est vide.

```
{email === "" && <p>Le champ email est obligatoire</p>}
```

Toujours valider les données avant de les envoyer à une API.

Conclusion

Dans ce TP, vous avez appris à :
— créer des formulaires contrôlés
— récupérer et traiter les données utilisateur
— mettre en place une validation simple

Les formulaires contrôlés sont essentiels pour toute application interactive.