Reactjs

week 4

Margit Tennosaar

Last session

- Immutable JS
- Unidirectional flow
- Forms and controlled components

This session

- Destructuring
- Managing lists and keys
- Conditional rendering techniques

Forms and controlled components

```
import { useState } from 'react';
import Header from './Header';
import Footer from './Footer';
import Form from './Form';
import View from './View';

function App() {
const [formData, setFormData] = useState({});

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

return (
    <>
      <Header />
      <Form changeHandler={changeFormHandler} />
      <View {...formData} />
      <Footer />
    </>
  );
}

export default App;
```

```
function SingleInputForm() {
  const [inputValue, setInputValue] = useState('');

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

  const handleSubmit = (event) => {
    event.preventDefault();
    alert('A name was submitted: ' + inputValue); };

  return (
    <form onSubmit={handleSubmit}>
      <label>
        Name:
        <input type="text" value={inputValue} onChange={handleChange} />
      </label>
      <button type="submit">Submit</button>
    </form>
  );
}

export default SingleInputForm;
```

```
function MultipleInputForm() {
  const [formState, setFormState] = useState({
    name: '',
    age: ''
  });

  const handleChange = (event) => {
    const { name, value } = event.target;
    setFormState(prevState => ({
      ...prevState,
      [name]: value
    }));
  };

  const handleSubmit = (event) => {
    event.preventDefault();
    alert(`Name: ${formState.name},
           Age: ${formState.age}`);
  };
```

```
  return (
    <form onSubmit={handleSubmit}>
      <label>
        Name:
        <input
          type="text"
          name="name"
          value={formState.name}
          onChange={handleChange}
        />
      </label>
      <label>
        Age:
        <input
          type="number"
          name="age"
          value={formState.age}
          onChange={handleChange}
        />
      </label>
      <button type="submit">Submit</button>
    </form>
  );
}
```

Destructuring

The destructuring **assignment** syntax is a JavaScript expression that makes it possible to unpack values from arrays, or properties from objects, into distinct variables.

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Destructuring_assignment

Destructuring

Instead of assigning the entire props object into a variable called *props* and then assigning its properties into the variables handleClick and text, we assign the values of the properties directly to variables by destructuring the props object that is passed to the component function as a parameter.

Destructuring

```
props = {
 name: ‘Maria’;
 age: 29,
}
```

```
const Card = (props) => {
  return (
    <>
      <p>Name: {props.name}</p>
      <p>Age: {props.age}</p>
    </>
  );
};
```

```
const Card = (props) => {
  const { name, age } = props
  return (
    <>
      <p>Name: {name}</p>
      <p>Age: {age}</p>
    </>
  );
};
```

```
 const [persons, setPersons] = useState([
    { id: 1, name: 'Margit', title: 'CEO', age: 29 },
    { id: 2, name: 'Kati', title: 'developer', age: 25 },
    { id: 3, name: 'Karin', title: 'designer', age: 45 },
  ])
```

```
const Card = ({name, age}) => {
  return (
    <div className="card">
      <p>Name: {name}</p>
      <p>Age: {age}</p>
    </div>
  );
};

export default Card;
```

```
{persons.map((person) => (
   <Card key={person.id} {...person} />
))}
```

Images in React

Local image in React project

```
import image from './assets/react.svg’

export default function App() {
  return (
    <>
      <img src={image} alt="React logo" />
    </>
  );
}
```

```
.my-image {
  background-image: url('./assets/react.svg');
  width: 100px;
  height: 100px;
}
```

Styling in react

Styling in react

external (and inline styling)

```
return (
   <div className="card primary">
     <p>Name: {props.name}</p>
     <p style={{color: "red"}}>Age: {props.age}</p>
   </div>
 );
```

Passing parameters to event handlers

```
const clickHandler = (id) => {
  console.log(id);
};
```

```
<button onClick={() => clickHandler(id)}> Like </button>
```

```
<button onClick={clickHandler.bind(this, id)}> Like </button>
```

https://reactjs.org/docs/handling-events.html#passing-arguments-to-event-handlers