

A decorative graphic on the left side of the slide. It consists of a blue parallelogram and a light green parallelogram, both tilted at an angle. The blue shape is in the foreground, and the green shape is partially behind it. They are set against a dark blue background with faint, lighter blue diagonal stripes.

Event handling med State



Agenda

- Stateful komponents
- Event Handling
 - knappar
 - text input
 - checkbox
 - radio button
 - submit
- Övningar + Läxa



Stateful Komponent

Vi hanterar state med useState funktion

Testa...

- counter: state
- setCounter: mekanism för att modifiera

```
import { useState } from 'react'

const App = () => {
  const [ counter, setCounter ] = useState(0)

  setTimeout(
    () => setCounter(counter + 1),
    1000
  )

  return (
    <div>{counter}</div>
  )
}

export default App
```



Stateful Komponent

Istället för att låta timer event trigga
kan vi låta “user event” trigga förändring

- Testa...

```
const App = () => {  
  const [ counter, setCounter ] = useState(0)  
  
  const handleClick = () => {  
    console.log('clicked')  
  }  
  
  return (  
    <div>  
      <div>{counter}</div>  
      <button onClick={handleClick}>  
        plus  
      </button>  
    </div>  
  )  
}
```



Stateful komponent

Vi kan börja skriva om
funktion till inline

- Hur kan vi ändra counter?

```
const App = () => {  
  const [ counter, setCounter ] = useState(0)  
  
  return (  
    <div>  
      <div>{counter}</div>  
      <button onClick={() => console.log('clicked')}>  
        plus  
      </button>  
    </div>  
  )  
}
```



Stateful Komponent

Vi kan helt enkelt låta den funktionen anrop `setCounter` istället för att `console.log`

```
<button onClick={() => setCounter(counter + 1)}>  
  plus  
</button>
```

Försök att lägga till en ny knapp som nollstället counter...



Stateful Komponent

```
const App = () => {  
  const [ counter, setCounter ] = useState(0)  
  
  return (  
    <div>  
      <div>{counter}</div>  
      <button onClick={() => setCounter(counter + 1)}>  
        plus  
      </button>  
      <button onClick={() => setCounter(0)}>  
        zero  
      </button>  
    </div>  
  )  
}
```



Event Handling

Varje JSX element har olika event handlers

- onClick är en sådan

```
<button onClick={() => setCounter(counter + 1)}>  
  plus  
</button>
```




Event Handling

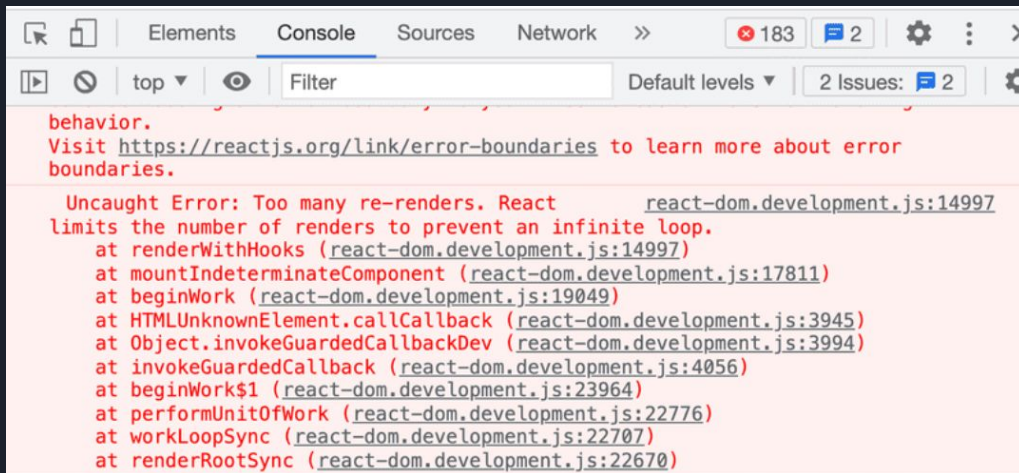
Vad skulle hända om vi skrev så här istället?

```
<button onClick={setCounter(counter + 1)}>  
  plus  
</button>
```

Event Handling

En event handler ska vara en funktion inte funktionsanrop!

- Varje gång appen renderas om så anropas eventHandlern och triggas åter en ny omrendering i infinitum..

A screenshot of a web browser's developer console. The 'Console' tab is selected, showing a red error message. The message states: 'Uncaught Error: Too many re-renders. React limits the number of renders to prevent an infinite loop.' It includes a stack trace with the following frames: 'at renderWithHooks (react-dom.development.js:14997)', 'at mountIndeterminateComponent (react-dom.development.js:17811)', 'at beginWork (react-dom.development.js:19049)', 'at HTMLUnknownElement.callCallback (react-dom.development.js:3945)', 'at Object.invokeGuardedCallbackDev (react-dom.development.js:3994)', 'at invokeGuardedCallback (react-dom.development.js:4056)', 'at beginWork\$1 (react-dom.development.js:23964)', 'at performUnitOfWork (react-dom.development.js:22776)', 'at workLoopSync (react-dom.development.js:22707)', and 'at renderRootSync (react-dom.development.js:22670)'. Above the error message, there is a link to 'https://reactjs.org/link/error-boundaries' with the text 'Visit' and 'to learn more about error boundaries.' The console also shows a 'behavior.' label at the top of the error block.

```
behavior.  
Visit https://reactjs.org/link/error-boundaries to learn more about error  
boundaries.  
  
Uncaught Error: Too many re-renders. React react-dom.development.js:14997  
limits the number of renders to prevent an infinite loop.  
    at renderWithHooks (react-dom.development.js:14997)  
    at mountIndeterminateComponent (react-dom.development.js:17811)  
    at beginWork (react-dom.development.js:19049)  
    at HTMLUnknownElement.callCallback (react-dom.development.js:3945)  
    at Object.invokeGuardedCallbackDev (react-dom.development.js:3994)  
    at invokeGuardedCallback (react-dom.development.js:4056)  
    at beginWork$1 (react-dom.development.js:23964)  
    at performUnitOfWork (react-dom.development.js:22776)  
    at workLoopSync (react-dom.development.js:22707)  
    at renderRootSync (react-dom.development.js:22670)
```



Event Handling

Övning: lägg till knapp som minskar
counter

```
const App = () => {  
  const [ counter, setCounter ] = useState(0)  
  
  const increaseByOne = () => setCounter(counter + 1)  
  
  const setToZero = () => setCounter(0)  
  
  return (  
    <div>  
      <div>{counter}</div>  
      <button onClick={increaseByOne}>  
        plus  
      </button>  
      <button onClick={setToZero}>  
        zero  
      </button>  
    </div>  
  )  
}
```



Event Handling

Övning:

- Skapa nytt react projekt
- Ta bort alla filer i src förutom main.jsx och App.jsx (och korrigera importer)
- Skapa en toggle knapp som togglar mellan “jag fattar” och “jag fattar inte”
 - När man öppnar sidan ska man se en div med “Jag fattar inte” och knapp
 - När man klickar på knappen ska man se div med “Jag fattar”
 - När man klickar på knappen igen ska man se “Jag fattar inte” osv...

Event Handling

onChange event

- Testa...

```
1  import React, { useState } from "react";
2  import ReactDOM from "react-dom/client";
3
4  const App = () => {
5    const [text, setText] = useState("");
6    return (
7      <form>
8        <input
9          type="text"
10         value={text}
11         onChange={(e) => setText(e.target.value)}
12        />
13        <p>{text}</p>
14      </form>
15    );
16  };
17
18  ReactDOM.createRoot(document.getElementById("root")).render(<App />);
```

Event Handling

onChange event (text)

- Lägg till fält
 - name
 - tel
- Kapitalisera (name)
- Kontrollera nr (tel)
- Visa name och tel

```
1 import React, { useState } from "react";
2 import ReactDOM from "react-dom/client";
3
4 const App = () => {
5   const [text, setText] = useState("");
6   const changeText = (event) => {
7     setText(event.target.value)
8   }
9   return (
10     <form>
11       <input
12         type="text"
13         value={text}
14         onChange={changeText}
15       />
16       <p>{text}</p>
17     </form>
18   );
19 };
20
21 ReactDOM.createRoot(document.getElementById("root")).render(<App />);
--
```



Event Handling

onChange event (checkbox)

```
1  import React, { useState } from "react";
2  import ReactDOM from "react-dom/client";
3
4  const App = () => {
5    const [isChecked, setIsChecked] = useState(false);
6    const changeCheckBox = () => {
7      setIsChecked(!isChecked);
8    };
9    return (
10     <form>
11       <input checked={isChecked} type="checkbox" onChange={changeCheckBox} />
12       <p>{isChecked ? "checked" : "not"} </p>
13     </form>
14   );
15 };
16
17 ReactDOM.createRoot(document.getElementById("root")).render(<App />);
```

Event Handling

Övning

- Lägg till i name och tel formulär
 - Checkbox: Male/female
 - Checkbox: present/not here

```
1 import React, { useState } from "react";
2 import ReactDOM from "react-dom/client";
3
4 const App = () => {
5   const [isChecked, setIsChecked] = useState(false);
6   const changeCheckBox = () => {
7     setIsChecked(!isChecked);
8   };
9   return (
10     <form>
11       <input checked={isChecked} type="checkbox" onChange={changeCheckBox} />
12       <p>{isChecked ? "checked" : "not"}</p>
13     </form>
14   );
15 };
16
17 ReactDOM.createRoot(document.getElementById("root")).render(<App />);
```




Event Handling

Övning

- Fixa så att vi kan ändra värde
- Lägg till ett alternativ "C"

```
const App = () => {  
  const [radioVal, setRadioVal] = useState("A");  
  const changeRadioButton = (event) => {  
    console.log(event.target.value);  
  };  
  return (  
    <form>  
      <input  
        type="radio"  
        checked={radioVal == "A"}  
        value="A"  
        onChange={changeRadioButton}/>  
      <input  
        type="radio"  
        checked={radioVal == "B"}  
        value="B"  
        onChange={changeRadioButton}/>  
      <p>{radioVal}</p>  
    </form>  
  );  
};
```



Event Handling

onSubmit event (formulär)

```
const App = () => {  
  const [name, setName] = useState("");  
  const changeName = (event) => {  
    setName(event.target.value);  
  };  
  const sayHello = (event) => {  
    event.preventDefault();  
    alert("Hello, " + name);  
  };  
  return (  
    <form onSubmit={sayHello}>  
      <input type="text" value={name} onChange={changeName} />  
      <input type="submit" value="save"></input>  
    </form>  
  );  
};
```

Event Handling

Övning:

- Lägg till i name och tel formulär submit knapp
 - Låt submit callback alerta all information som fyllts i

```
const App = () => {  
  const [name, setName] = useState("");  
  const changeName = (event) => {  
    setName(event.target.value);  
  };  
  const sayHello = (event) => {  
    event.preventDefault();  
    alert("Hello, " + name);  
  };  
  return (  
    <form onSubmit={sayHello}>  
      <input type="text" value={name} onChange={changeName} />  
      <input type="submit" value="save"></input>  
    </form>  
  );  
};
```



Event Handling

Övning: Skapa ett nytt formulär

- username (text)
- female/male (radio)
- interests: “coding”, “sports”, “games” (checkbox)
- submit knapp som alert all info



Event Handling

Övning: Skapa ett formulär

- En text input (onChange)
- En spara knapp (onSubmit)
- När man skrivit något och tryckt “spara” läggs text i lista
- Visa listan i div eller konsoll