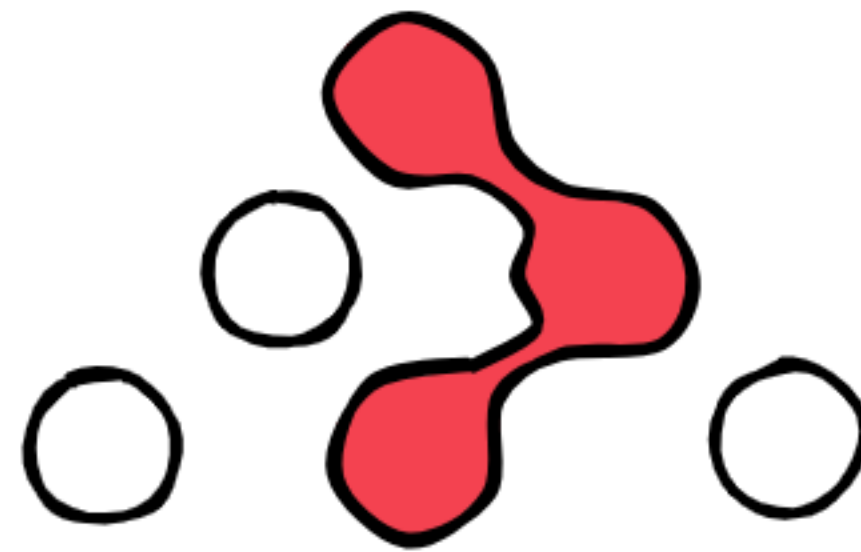
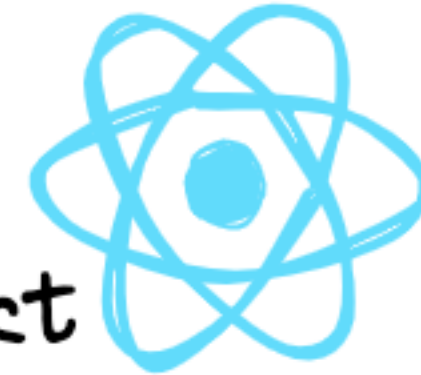


Pierre KRAEMER - kraemer@unistra.fr

Objectifs :

- connaître les bases du fonctionnement de React
- développer des applications web qui dialoguent avec des API
- utiliser une bibliothèque dédiée pour :
 - la gestion de l'interface via des routes client



React
Router

Étapes :

1- bases (composants, données, hooks, forms, ...)

2- application d'exemple (liste de courses)

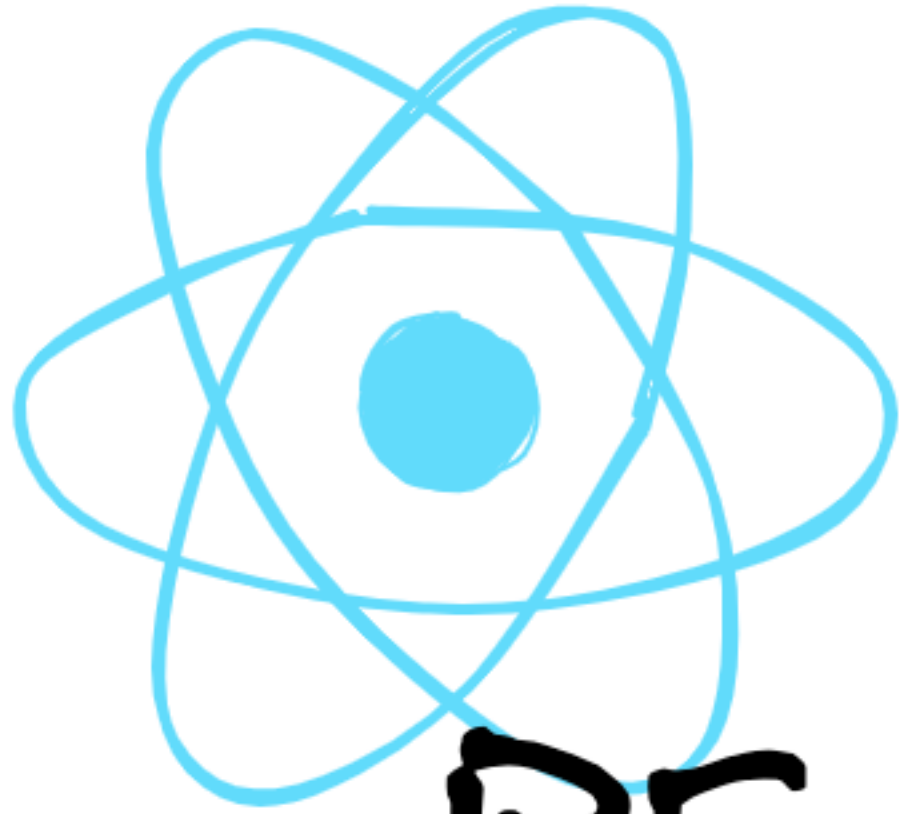
3- react-router 

4- échanges avec une API HTTP
-> interface pour la gestion de livres

Pas objectifs :

- faire des interfaces "jolies"
- maîtriser des bibliothèques de composants
(shadcn, Material-UI, React Bootstrap, Chakra UI, ...)

ou des bibliothèques CSS
(TailwindCSS, ...)



REACT

=

library (≠ framework)


declarative (vs. imperative)

component-based (JS functions)

"thinking about how the UI should look at any given moment,
rather than how to change it over time"

Component = function : data \rightarrow description of UI

$$\text{data} = \begin{cases} \text{parameters} \rightarrow \text{props} \\ \text{local} \rightarrow \text{state} \end{cases}$$


React elements

```
const Message = (props) => {  
  return <p>{props.msg}</p>;  
};
```


not HTML !


JSX !

```
 $\Leftrightarrow$  return React.createElement("p", null, props.msg);  
 $\Leftrightarrow$  return {  
  type: "p",  
  props: {  
    children: " ... "  
  }  
}
```

Component = function : data -> description of UI

data = $\begin{cases} \text{parameters} \rightarrow \text{props} \\ \text{local} \rightarrow \text{state} \end{cases}$

o stocké où ?
o par qui ?
o comment récupérer
la valeur courante
d'une évaluation à
l'autre ?



```
const Toggle = () => {  
  const [value, setValue] = useState(true);  
  const handleChange = () => {  
    setValue(v => !v);  
  };  
  return <div>  
    <p>{value ? "ON" : "OFF"}</p>  
    <button onChange={handleChange}>Change</button>  
  </div>;  
};
```


React evaluation

```
ReactDOM.render(  
  <Toggle />  
);
```

React

Elements tree

Toggle

```
{  
  type: Toggle,  
  props: null  
}
```

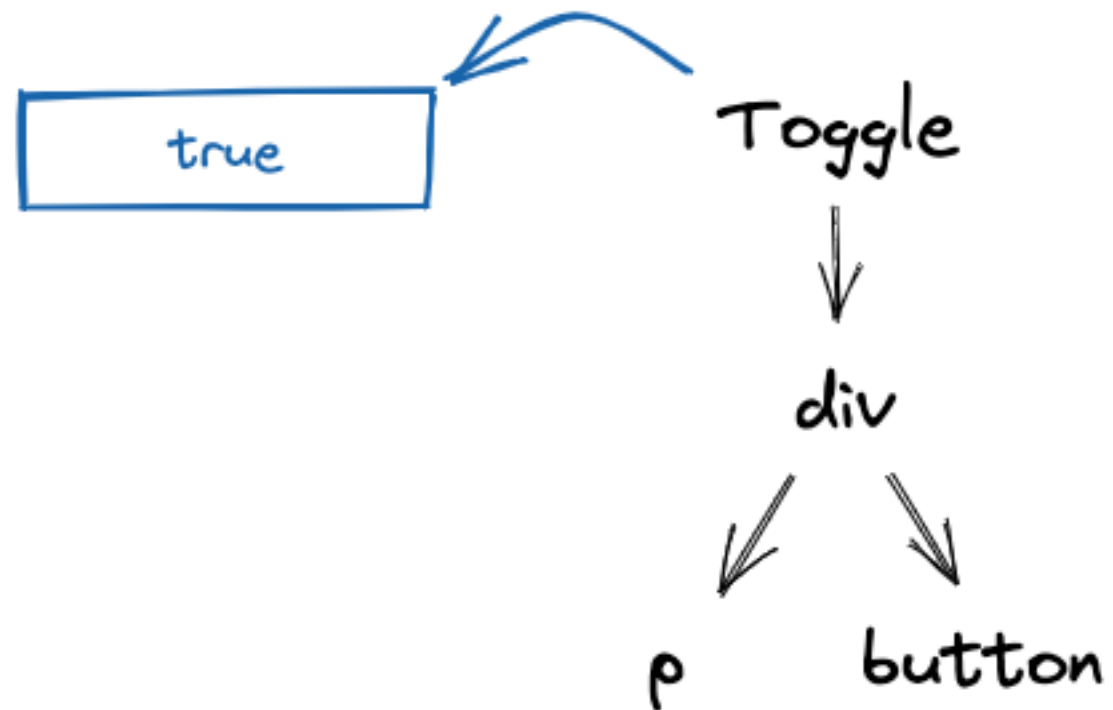
function (component)
-> must be called

React evaluation

```
ReactDOM.render(  
  <Toggle />  
)
```

React

Elements tree



```
{  
  type: 'div',  
  props: {  
    children: [  
      { type: 'p', props: { children: "ON" } },  
      { type: 'button', props: { onChange: ..., children: "Change" } }  
    ]  
  }  
}
```

string (known by the renderer)

React evaluation

```
ReactDOM.render(  
  <Toggle />  
);
```

React

Elements tree

ReactDOM

DOM

```
<div>  
  <p>ON</p>  
  <button>Change</button>  
</div>
```

```
{  
  type: 'div',  
  props: {  
    children: [  
      { type: 'p', props: { children: "ON" } },  
      { type: 'button', props: { onChange: ..., children: "Change" } }  
    ]  
  }  
}
```

true

Toggle

div

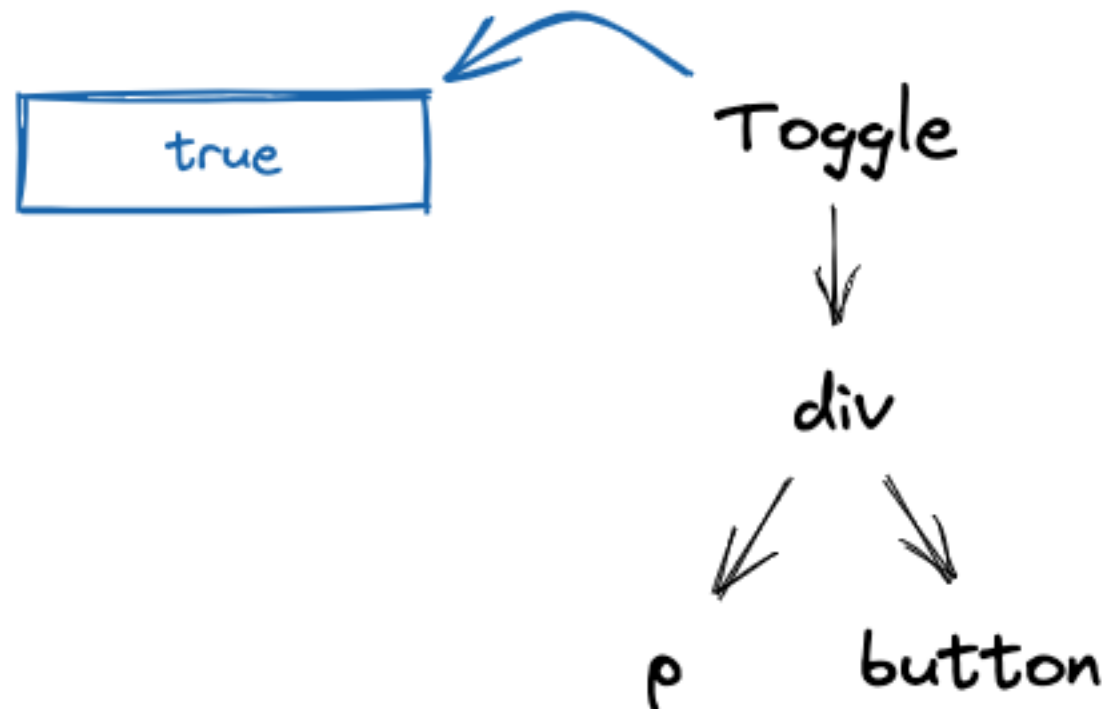
p

button

React evaluation

```
ReactDOM.render(  
  <Toggle />  
)
```

Elements tree



```
{  
  type: 'div',  
  props: {  
    children: [  
      { type: 'p', props: { children: "ON" } },  
      { type: 'button', props: { onChange: ..., children: "Change" } }  
    ]  
  }  
}
```

React

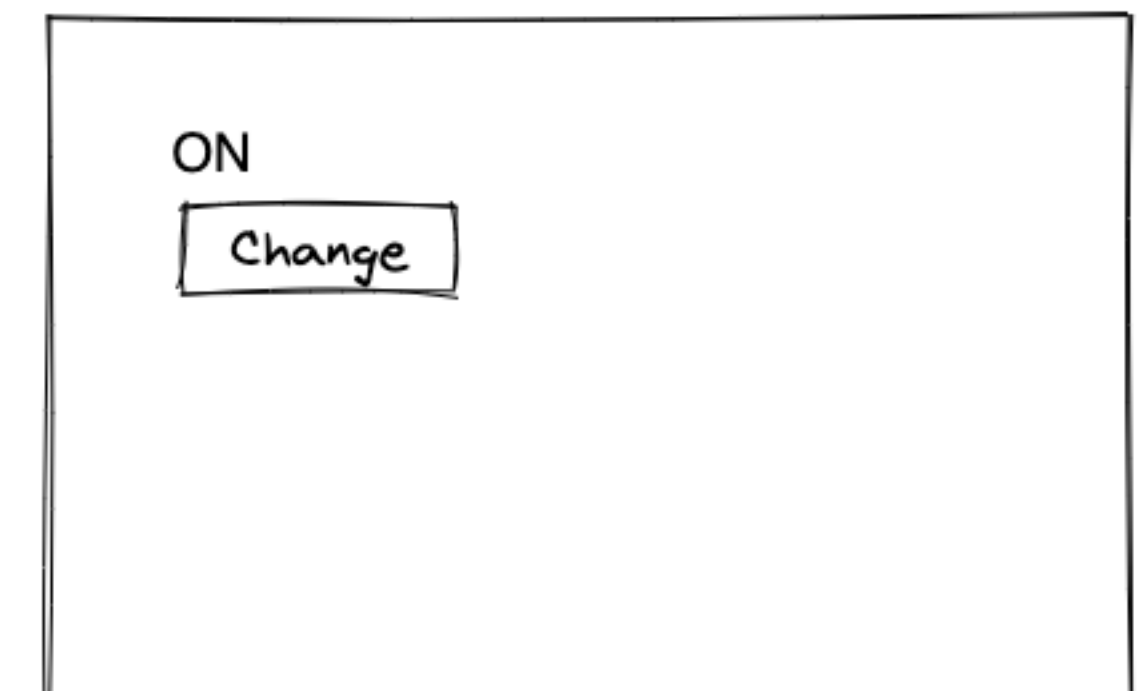
ReactDOM

DOM

```
<div>  
  <p>ON</p>  
  <button>Change</button>  
</div>
```

Web browser

UI



React evaluation

```
ReactDOM.render(  
  <Toggle />  
)
```

Elements tree

React

ReactDOM

Web browser

UI



Toggle

div

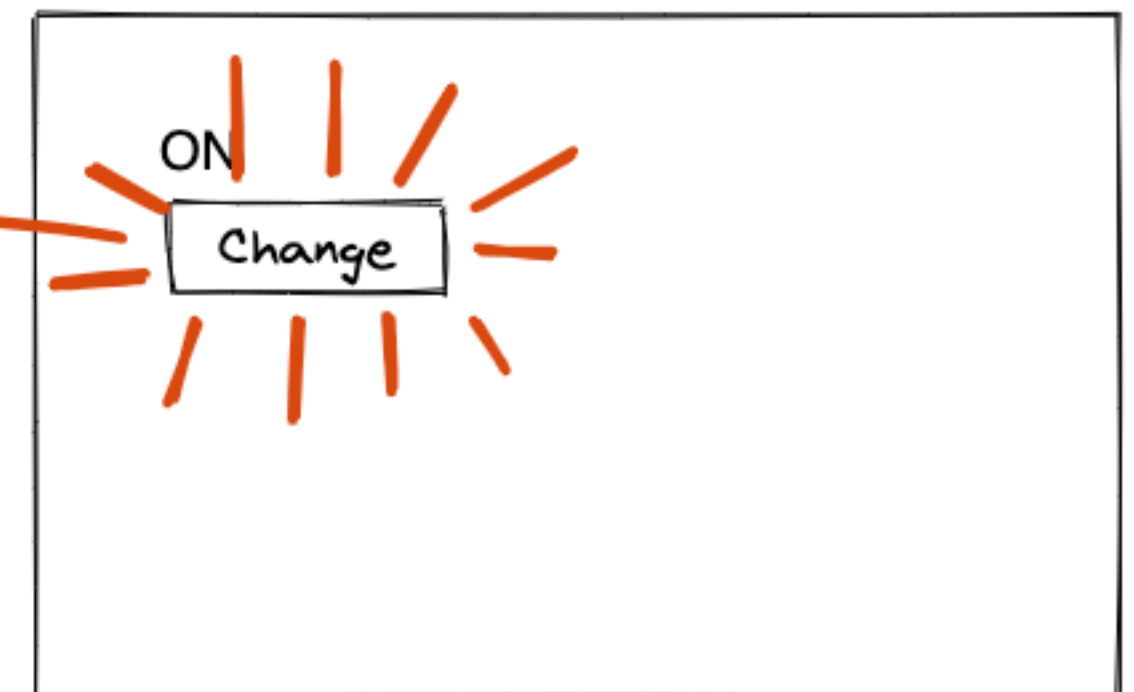
p

button

DOM

```
<div>  
  <p>ON</p>  
  <button>Change</button>  
</div>
```

```
{  
  type: 'div',  
  props: {  
    children: [  
      { type: 'p', props: { children: "ON" } },  
      { type: 'button', props: { onChange: ..., children: "Change" } }  
    ]  
  }  
}
```



React evaluation

```
ReactDOM.render(  
  <Toggle />  
)
```

Elements tree

ReactDOM



Toggle

div

p

button

DOM

```
<div>  
  <p>OFF</p>  
  <button>Change</button>  
</div>
```

Web browser

UI

```
{  
  type: 'div',  
  props: {  
    children: [  
      { type: 'p', props: { children: "ON" } },  
      { type: 'button', props: { onChange: ..., children: "Change" } }  
    ]  
  }  
}
```



npm create vite@latest

npm run dev

DÉMO

useState

localStorage

useEffect

map(), filter(), ...