

Keeping Front-end Development Simple with React

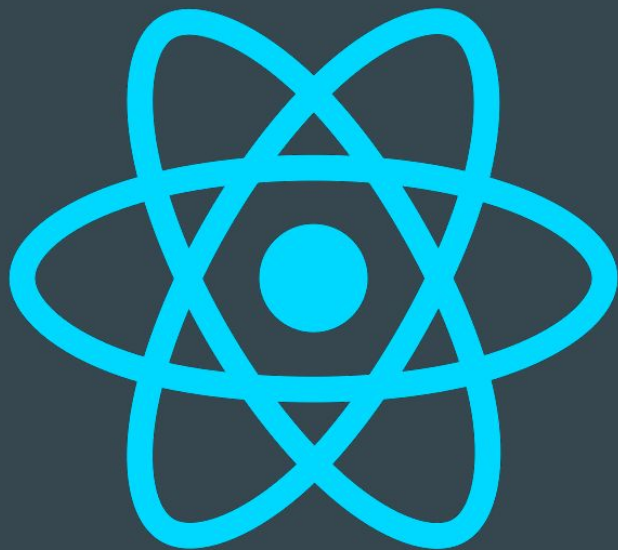
...

Tony Tsui



 @tony_tsui

React



KEEP IT SIMPLE

simple /'sɪmp(ə)l/

easily understood or done



<https://flic.kr/p/uYhasX>

KEEP IT SIMPLE

simple /'sɪmp(ə)l/
easily understood or done

1. Composable Components



KEEP IT SIMPLE

simple /'sɪmp(ə)l/

easily understood or done

1. Composable Components
2. Predictable Data Flow



KEEP IT SIMPLE

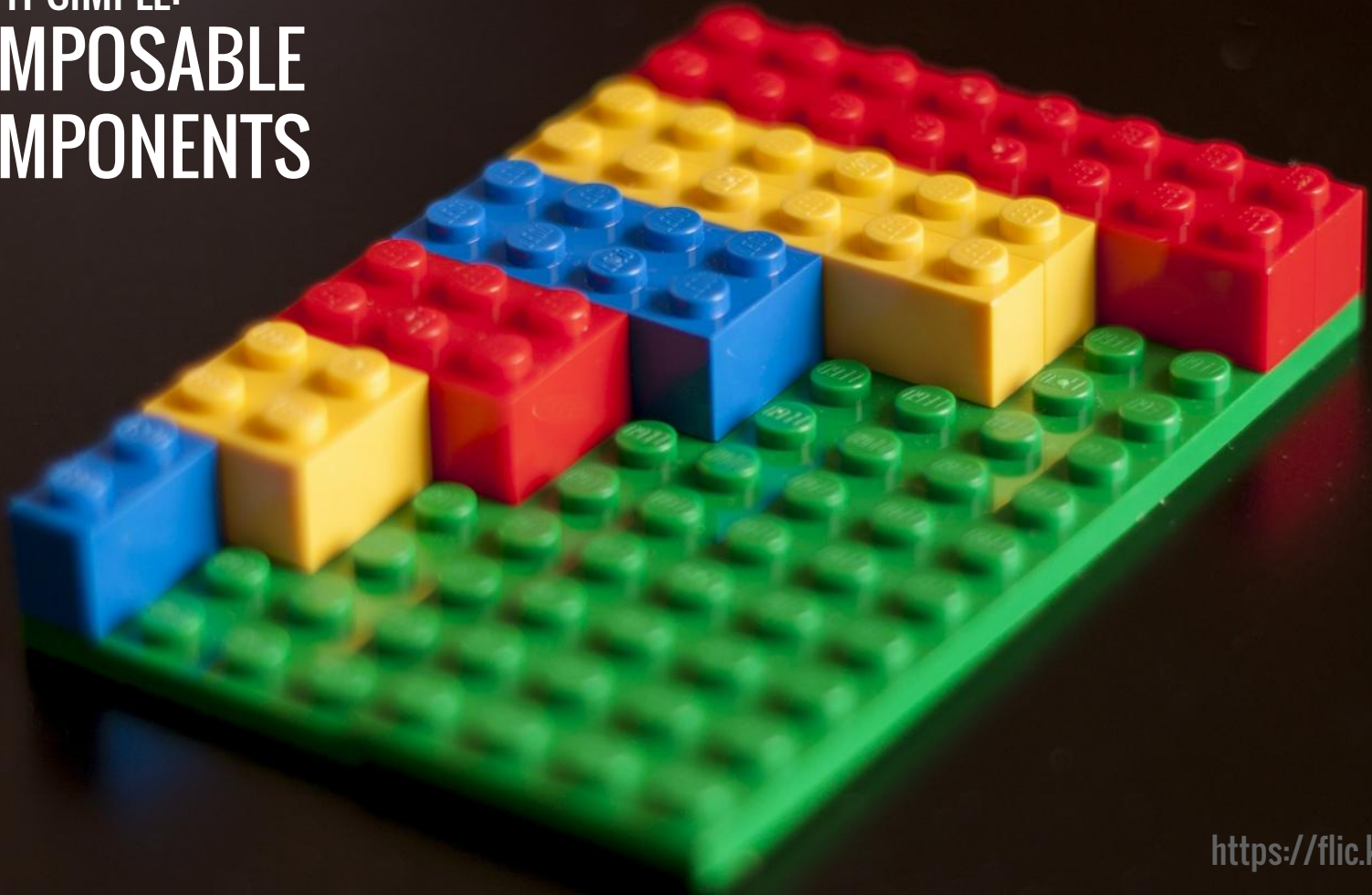
simple /'sɪmp(ə)l/

easily understood or done

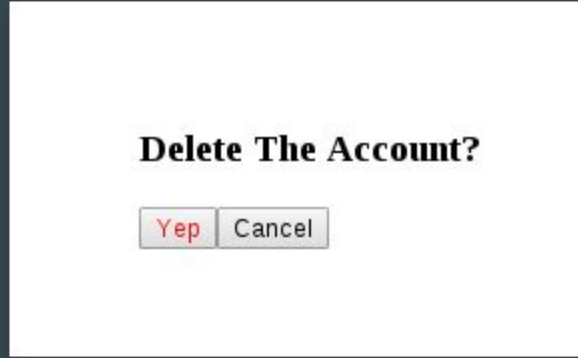
1. Composable Components
2. Predictable Data Flow
3. Carefree Rendering



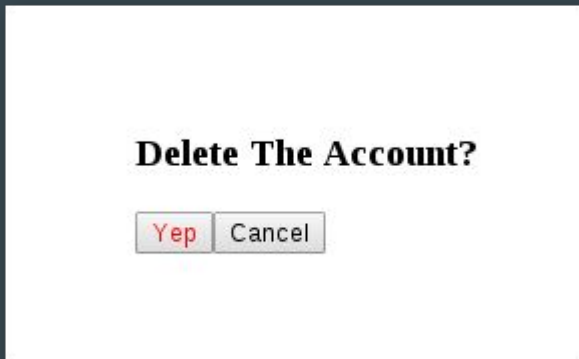
KEEP IT SIMPLE:
**COMPOSABLE
COMPONENTS**



ANATOMY OF A REACT COMPONENT

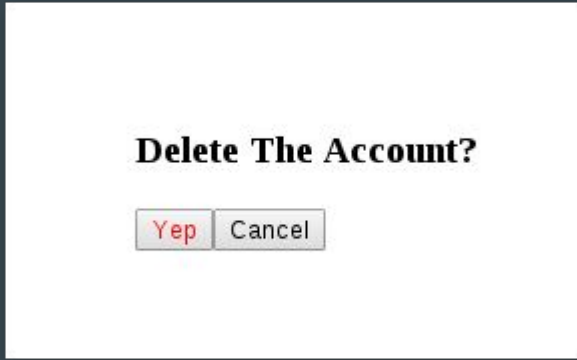


ANATOMY OF A REACT COMPONENT



```
<div>  
  <h3>Delete The Account?</h3>  
  <DangerButton>Yep</DangerButton>  
  <button>Cancel</button>  
</div>
```

ANATOMY OF A REACT COMPONENT



```
const DeleteAccount = () => (  
  <div>  
    <h3>Delete The Account?</h3>  
    <DangerButton>Yep</DangerButton>  
    <button>Cancel</button>  
  </div>  
);
```

ANATOMY OF A REACT COMPONENT



```
const DeleteAccount = () => (  
  <div>  
    <h3>Delete The Account?</h3>  
    <DangerButton>Yep</DangerButton>  
    <button>Cancel</button>  
  </div>  
);
```

ANATOMY OF A REACT COMPONENT



```
const DeleteAccount = () => (
```

```
  <div>
```

```
    <h3>Delete The Account?</h3>
```

```
    <DangerButton>Yep</DangerButton>
```


```
    <button>Cancel</button>
```

```
  </div>
```

```
);
```

ANATOMY OF A REACT COMPONENT

ES2015 Arrow
Function Syntax



```
const DeleteAccount = () => (  
  <div>  
    <h3>Delete The Account?</h3>  
    <DangerButton>Yep</DangerButton>  
    <button>Cancel</button>  
  </div>  
);
```

ANATOMY OF A REACT COMPONENT

JSX



```
const DeleteAccount = () => (  
  <div>  
    <h3>Delete The Account?</h3>  
    <DangerButton>Yep</DangerButton>  
    <button>Cancel</button>  
  </div>  
);
```

JSX: BEHIND THE CURTAINS

```
const DeleteAccount = () => (  
  <div>  
    <h3>Delete The Account?</h3>  
    <DangerButton>Yep</DangerButton>  
    <button>Cancel</button>  
  </div>  
);
```

JSX: BEHIND THE CURTAINS

```
const DeleteAccount = () => (  
  <div>  
    <h3>Delete The Account?</h3>  
    <DangerButton>Yep</DangerButton>  
    <button>Cancel</button>  
  </div>  
)
```



```
const DeleteAccount = () => (  
  React.createElement('div', null,  
    React.createElement('h3', null, 'Delete The Account?'),  
    React.createElement(DangerButton, null, 'Yep' ),  
    React.createElement('button', null, 'Cancel')  
  )  
)
```


EMBRACE JAVASCRIPT

Handlebars

```
<ul>  
  {{#each accounts}}  
    <li>{{this}}</li>  
  {{/each}}  
</ul>
```

EMBRACE JAVASCRIPT

Handlebars

```
<ul>
  {{#each accounts}}
    <li>{{this}}</li>
  {{/each}}
</ul>
```



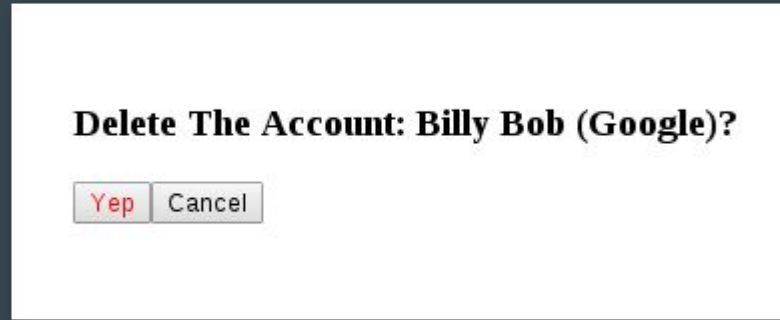
React

```
<ul>
  { accounts.map(account => <li>{account}</li> ) }
</ul>
```

COMPONENT INTERFACE

Delete The Account: Billy Bob (Google)?

COMPONENT INTERFACE



```
<DeleteAccount accountName="Billy Bob (Google)" />
```

COMPONENT INTERFACE

```
<DeleteAccount accountName="Billy Bob (Google)" />
```



```
const DeleteAccount = (props) => (  
  <div>  
    <h3>Delete The Account: {props.accountName}?</h3>  
    <DangerButton>Yep</DangerButton>  
    <button>Cancel</button>  
  </div>  
);
```

COMPONENT INTERFACE

```
<DeleteAccount accountName="Billy Bob (Google)"  
    onConfirm={ _handleDeleteAccountConfirmation }  
    onCancel={ _handleDeleteAccountCancellation }/>
```

COMPONENT INTERFACE

```
<DeleteAccount accountName="Billy Bob (Google)"
  onConfirm={ _handleDeleteAccountConfirmation }
  onCancel={ _handleDeleteAccountCancellation }/>
```



```
const DeleteAccount = (props) => (
  <div>
    <h3>Delete The Account: {props.accountName}?</h3>
    <DangerButton onClick={() => props.onConfirm(props.accountName)}
      >Yep</DangerButton>
    <button onClick={props.onCancel}>Cancel</button>
  </div>
);
```

REACT COMPONENT

```
const DeleteAccount = (props) => (  
  <div>  
    <h3>Delete The Account: {props.accountName}?</h3>  
    <DangerButton onClick={() => props.onConfirm(props.accountName)}  
      >Yep</DangerButton>  
    <button onClick={props.onCancel}>Cancel</button>  
  </div>  
);
```


KEEP IT SIMPLE:
PREDICTABLE DATA FLOW

KEEP IT SIMPLE: PREDICTABLE DATA FLOW



<http://www.wikihow.com/Image:Make-Spaghetti-Step-1.jpg>

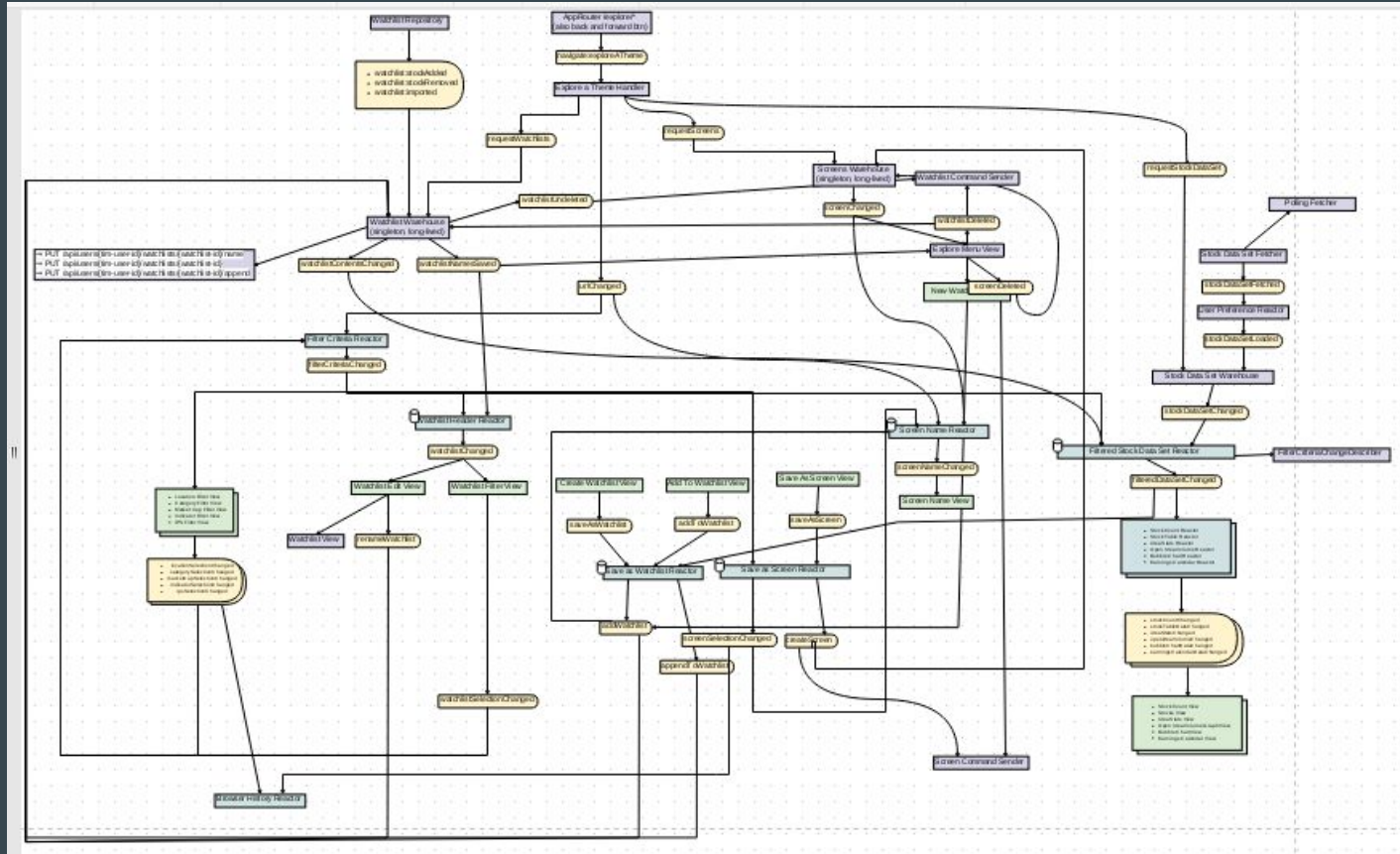
KEEP IT SIMPLE: PREDICTABLE DATA FLOW



<http://www.wikihow.com/Image:Make-Spaghetti-Step-1.jpg>

<https://flic.kr/p/7eRnDo>

PUB-SUB SPAGHETTI



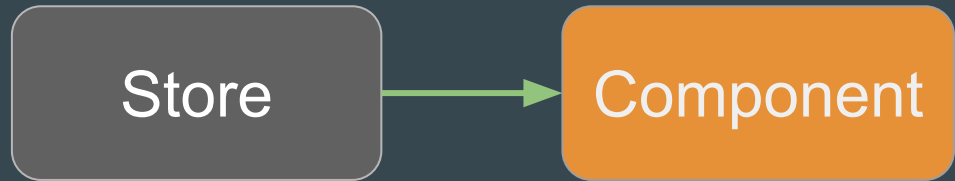
UNIDIRECTIONAL DATA FLOW



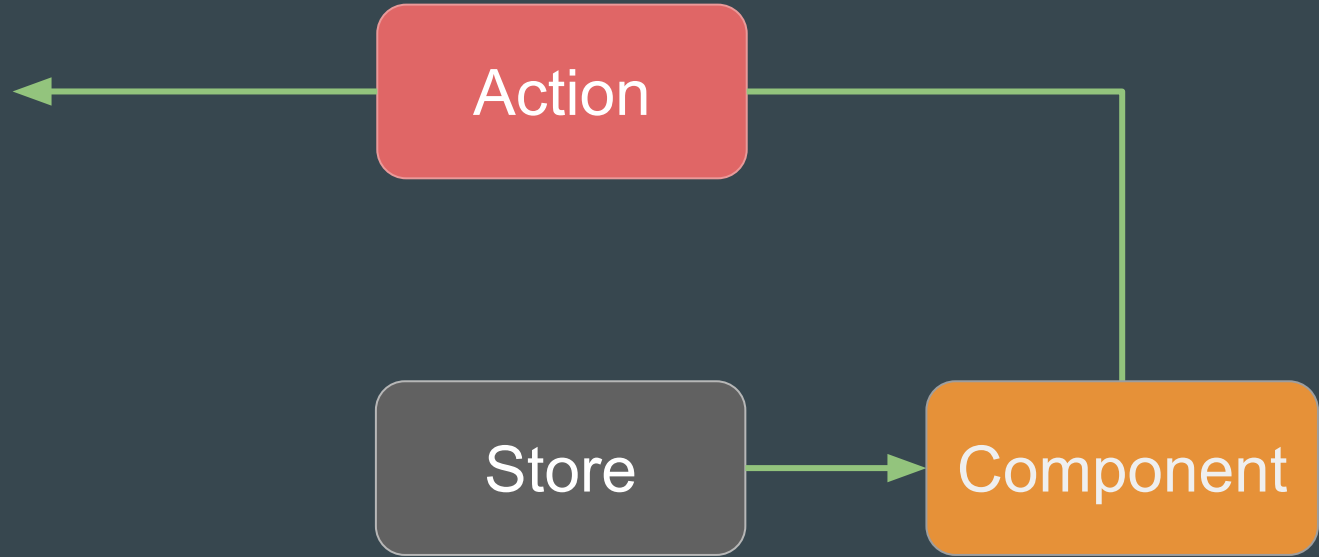
UNIDIRECTIONAL DATA FLOW

Component

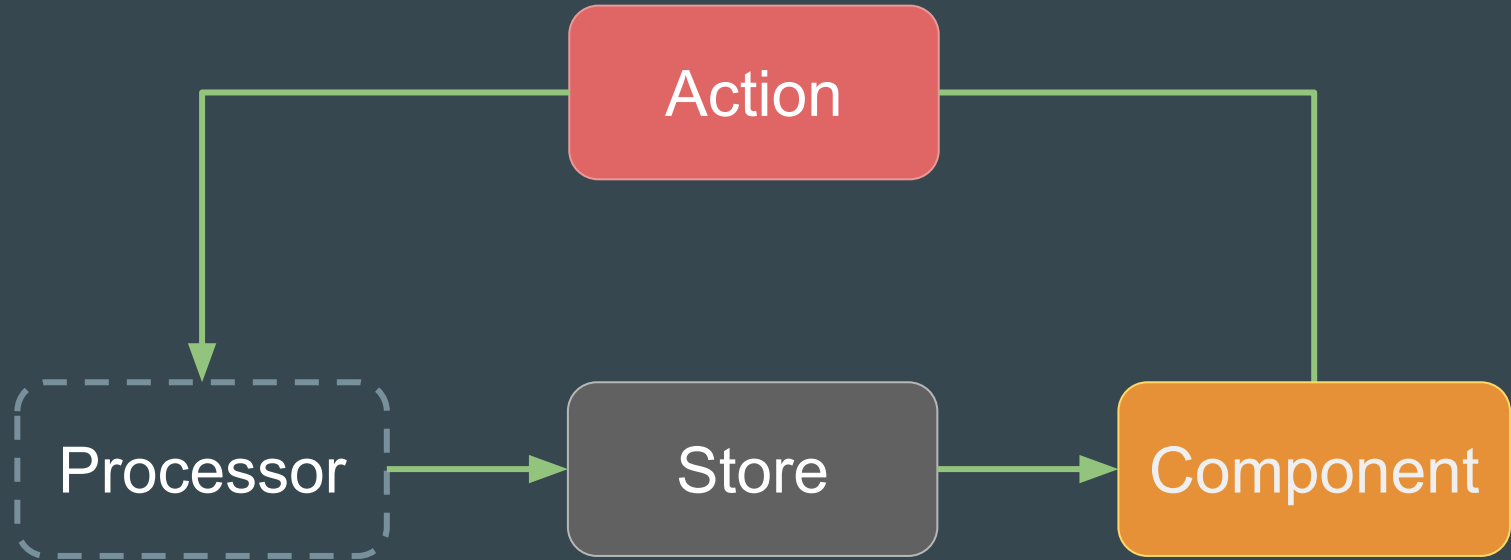
UNIDIRECTIONAL DATA FLOW



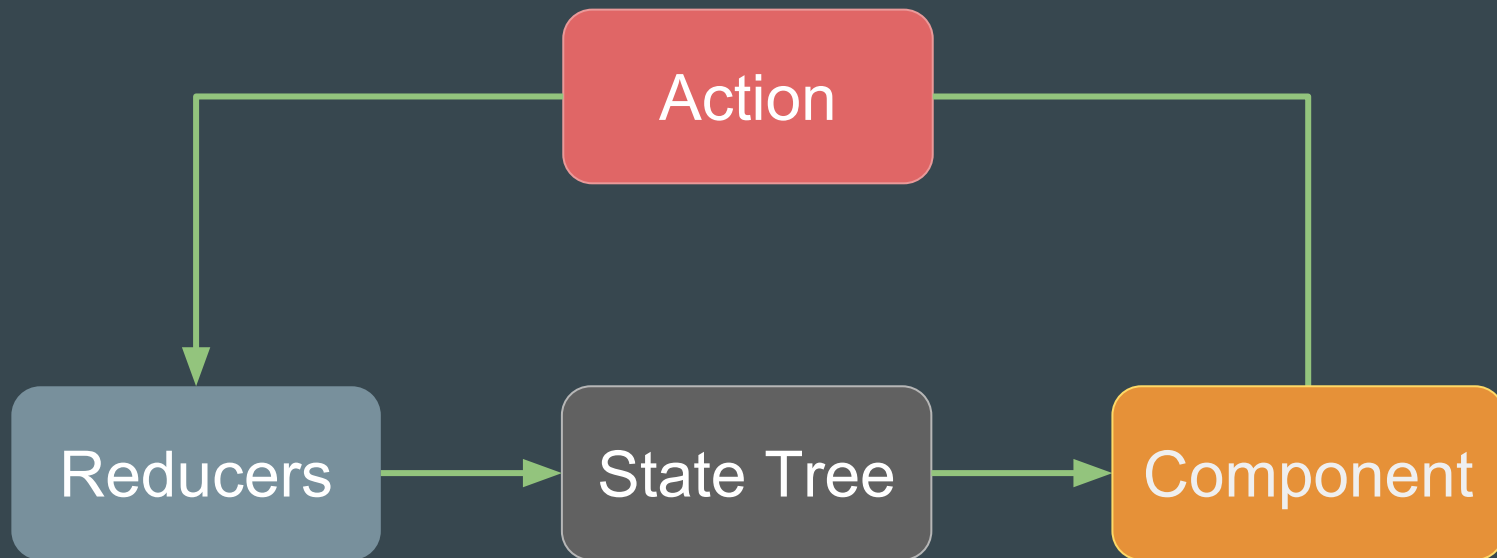
UNIDIRECTIONAL DATA FLOW



UNIDIRECTIONAL DATA FLOW



REDUX



REDUX

Action

Reducers

State Tree

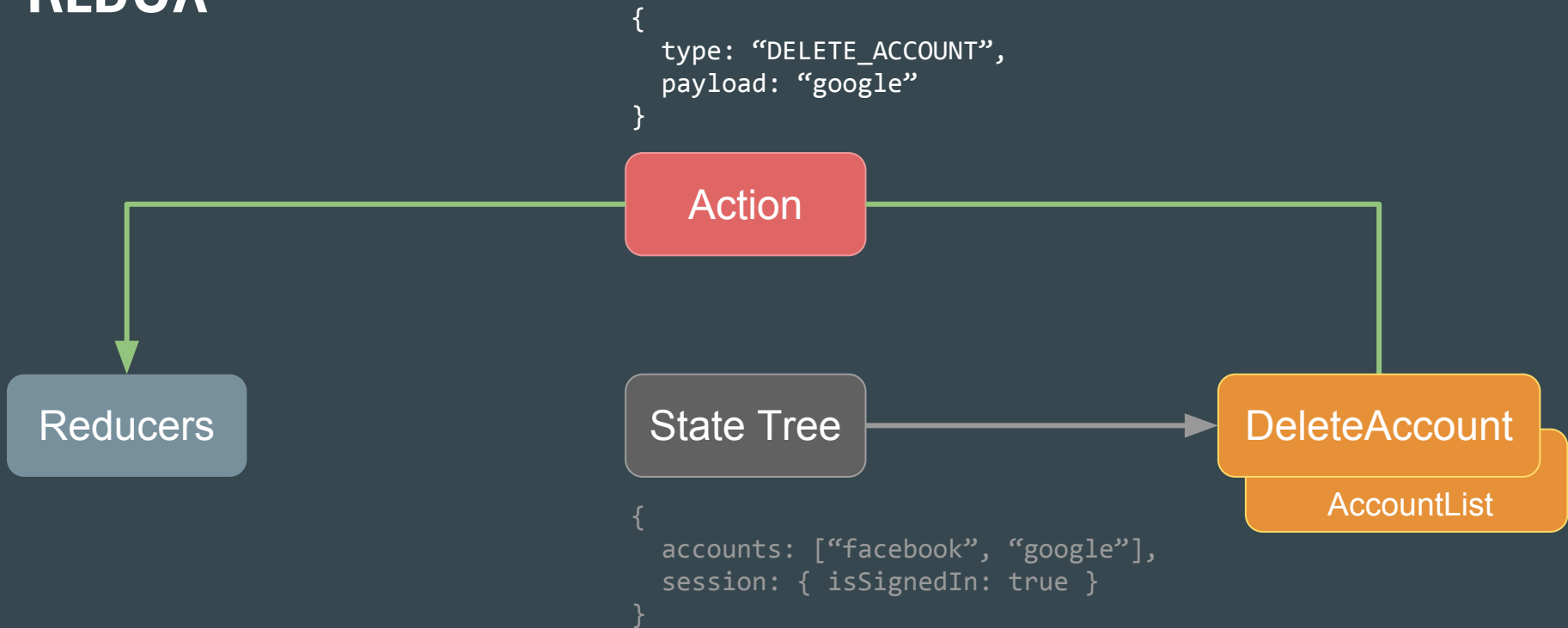
DeleteAccount

AccountList

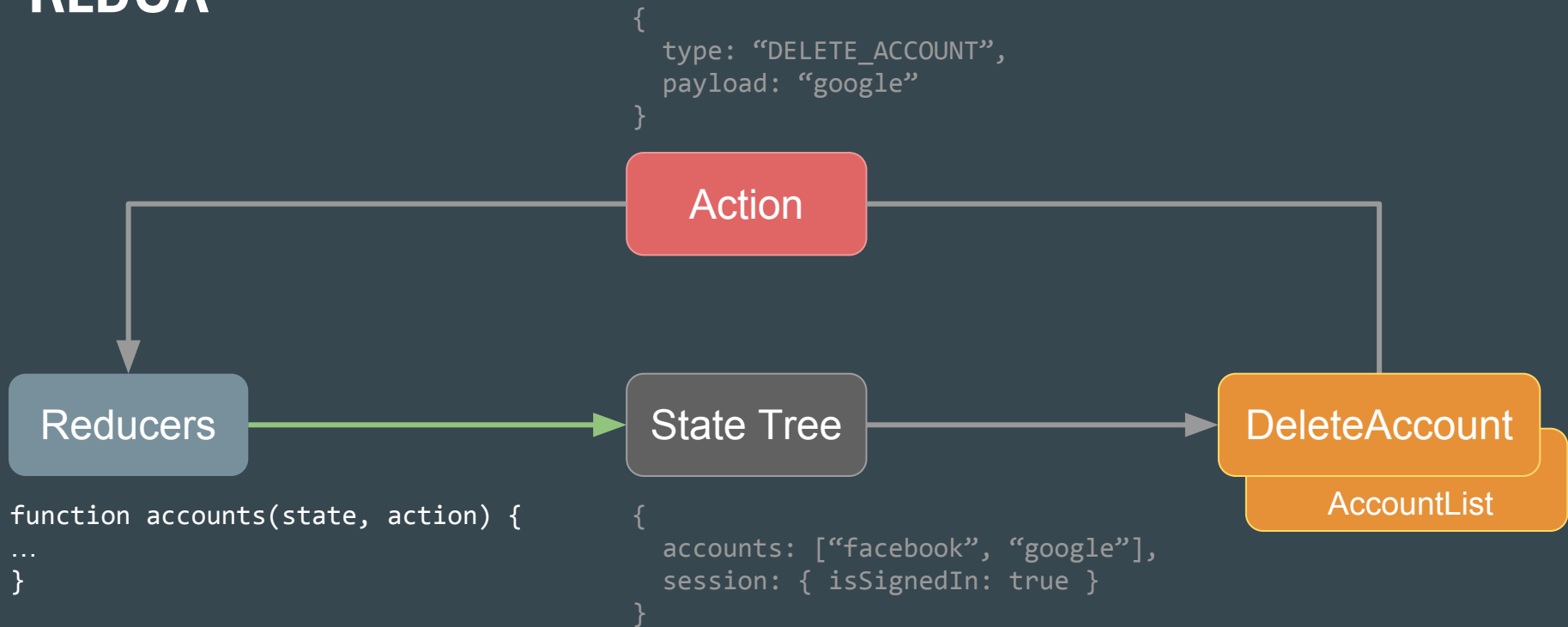
REDUX



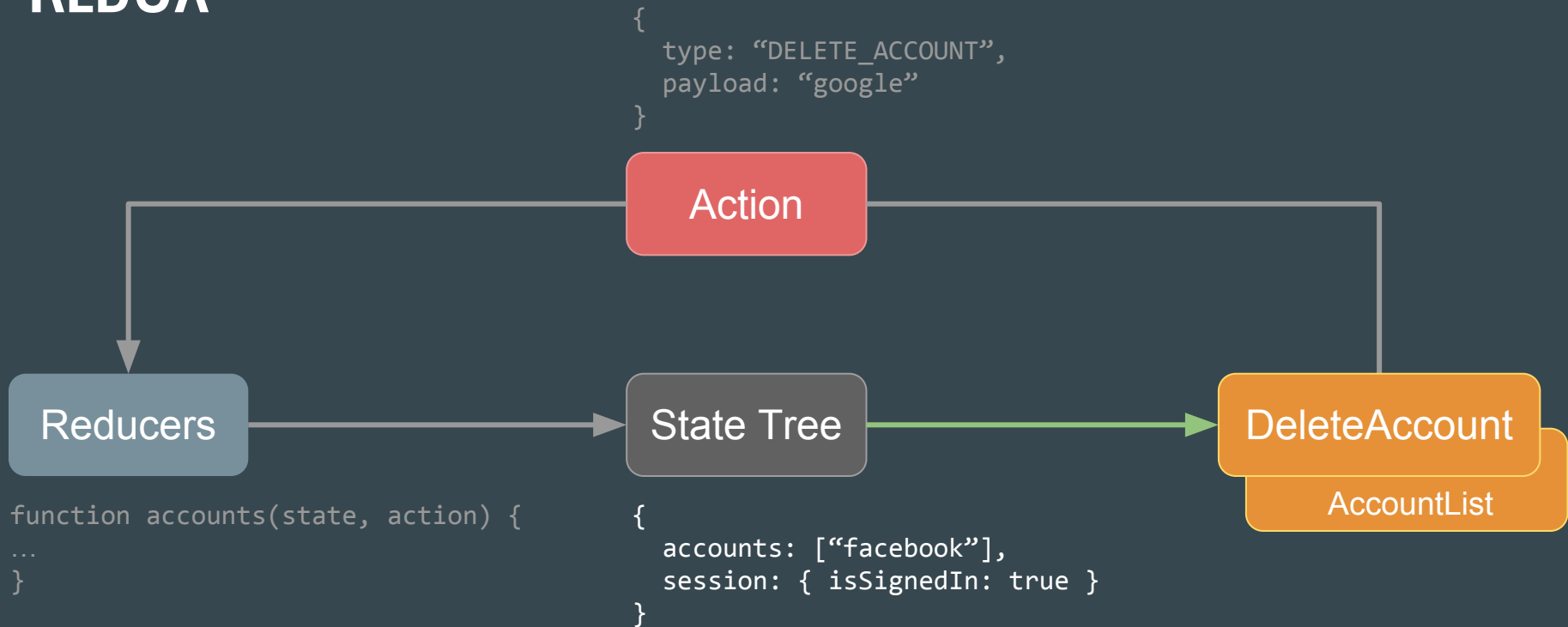
REDUX



REDUX



REDUX



REDUCER

```
function accounts(state, action) {  
  switch (action.type) {  
    case DELETE_ACCOUNT:  
      const accountToDelete = action.payload;  
      const accounts  
        = state.accounts  
          .filter(account => account !== accountToDelete);  
  
      return Object.assign({}, state, { accounts });  
    default:  
      return state  
  }  
}
```

KEEP IT SIMPLE:
CAREFREE RENDERING

The Problem

KEEP IT SIMPLE: CAREFREE RENDERING

```
<ul>  
  <li>Billy Bob (Google)</li>  
  <li>Mary Sue (Facebook)</li>  
</ul>
```

KEEP IT SIMPLE: CAREFREE RENDERING

```
<li>Jon Smith (Twitter)</li> —————> <ul>  
                                         <li>Billy Bob (Google)</li>  
                                         <li>Mary Sue (Facebook)</li>  
                                         </ul>
```

KEEP IT SIMPLE: CAREFREE RENDERING

```
const jon = document.createElement("li");  
jon.innerText = "Jon Smith (Twitter)";
```

KEEP IT SIMPLE:

CAREFREE RENDERING

```
const jon = document.createElement("li");  
jon.innerText = "Jon Smith (Twitter)";
```

```
const mary = document.getElementsByTagName("li")[1];
```

KEEP IT SIMPLE:

CAREFREE RENDERING

```
const jon = document.createElement("li");  
jon.innerText = "Jon Smith (Twitter)";
```

```
const mary = document.getElementsByTagName("li")[1];
```

```
const list = mary.parentNode;  
list.insertBefore(jon, mary);
```


KEEP IT SIMPLE:

CAREFREE RENDERING

```
const jon = document.createElement("li");  
jon.innerText = "Jon Smith (Twitter)";
```

```
const mary = document.getElementsByTagName("li")[1];
```

```
const list = mary.parentNode;  
list.insertBefore(jon, mary);
```

```
const billy = document.getElementsByTagName("li")[0];  
billy.remove();
```

KEEP IT SIMPLE:
CAREFREE RENDERING

The Ideal:
Just Render Everything All The Time

KEEP IT SIMPLE:
CAREFREE RENDERING

React's Solution:
Abstract Away The DOM

KEEP IT SIMPLE: CAREFREE RENDERING

Application State

```
{  
  accounts: [  
    "John Smith (Twitter)",  
    "Mary Sue (Facebook)"  
  ]  
}
```



Render Component

Virtual DOM

```
<ul>  
  <li>John Smith (Twitter)</li>  
  <li>Mary Sue (Facebook)</li>  
</ul>
```

Browser (Real) DOM

```
<ul>  
  <li>Billy Bob (Google)</li>  
  <li>Mary Sue (Facebook)</li>  
</ul>
```

KEEP IT SIMPLE: CAREFREE RENDERING

Application State

```
{  
  accounts: [  
    "John Smith (Twitter)",  
    "Mary Sue (Facebook)"  
  ]  
}
```



Virtual DOM

```
<ul>  
  <li>John Smith (Twitter)</li>  
  <li>Mary Sue (Facebook)</li>  
</ul>
```

Calculate Diff



Browser (Real) DOM

```
<ul>  
  <li>Billy Bob (Google)</li>  
  <li>Mary Sue (Facebook)</li>  
</ul>
```

KEEP IT SIMPLE: CAREFREE RENDERING

Application State

```
{  
  accounts: [  
    "John Smith (Twitter)",  
    "Mary Sue (Facebook)"  
  ]  
}
```



Virtual DOM

```
<ul>  
  <li>John Smith (Twitter)</li>  
  <li>Mary Sue (Facebook)</li>  
</ul>
```

Apply Patch



```
replaceAttribute  
textContent  
'Jon Smith (Twitter)'
```

Browser (Real) DOM

```
<ul>  
  <li>Billy Bob (Google)</li>  
  <li>Mary Sue (Facebook)</li>  
</ul>
```

KEEP IT SIMPLE: CAREFREE RENDERING

Application State

```
{  
  accounts: [  
    "John Smith (Twitter)",  
    "Mary Sue (Facebook)"  
  ]  
}
```



Virtual DOM

```
<ul>  
  <li>John Smith (Twitter)</li>  
  <li>Mary Sue (Facebook)</li>  
</ul>
```



Browser (Real) DOM

```
<ul>  
  <li>John Smith (Twitter)</li>  
  <li>Mary Sue (Facebook)</li>  
</ul>
```

Developer Tools



HOT RELOADING

The image shows a side-by-side comparison of a code editor and a web browser, demonstrating hot reloading. On the left, the IDE (JetBrains IDEA) displays the source code for a React application. The code defines an `App` component that renders a list of accounts and an `AddAccount` form. The `accounts` array contains "Billy Bob (Google)" and "Mary Sue (Facebook)". The `AddAccount` component has a text input and an `Add` button. The browser on the right shows the rendered output of this code. The page title is "Redux shopping cart example - Google Chrome". The URL is `localhost:3000/#/sign-in?_k=w1o9r8`. The page content includes a heading "Accounts", a horizontal line, a bulleted list of the two accounts, and a form with the label "Add account:" followed by a text input field and an "Add" button. The browser's address bar shows the current URL, and the IDE's status bar at the bottom indicates "8:1 UTF-8: LF:".

```
1 import React, { Component } from 'react'
2
3 const accounts = [
4   "Billy Bob (Google)",
5   "Mary Sue (Facebook)"
6 ];
7
8 class App extends Component {
9   render() {
10    return (
11      <div>
12        <Accounts />
13        <AddAccount />
14      </div>
15    );
16  }
17 }
18
19 const Accounts = () => (
20   <div>
21     <h1>Accounts</h1>
22     <hr />
23     <ul>
24       { accounts.map(account => <li key={account}>{account}</li> ) }
25     </ul>
26   </div>
27 );
28
29 class AddAccount extends Component {
30   render() {
31     return (
32       <div>
33         <label>Add account:</label>
34         <input type="text"
35
```

Redux shopping cart example - Google Chrome
localhost:3000/#/sign-in?_k=w1o9r8

Accounts

- Billy Bob (Google)
- Mary Sue (Facebook)

Add account:

HOT RELOADING

The image shows a development environment with two windows. The left window is an IDE (JetBrains IDEA) displaying the source code for a React application. The right window is a web browser (Google Chrome) showing the rendered output of the application.

Code in the IDE (App.js):

```
1 import React, { Component } from 'react'
2
3 const accounts = [
4   "Billy Bob (Google)",
5   "Mary Sue (Facebook)"
6 ];
7
8 class App extends Component {
9   render() {
10    return (
11      <div>
12        <Accounts />
13        <AddAccount />
14      </div>
15    );
16  }
17 }
18
19 const Accounts = () => (
20   <div>
21     <h1>Accounts</h1>
22     <hr />
23     <ul>
24       { accounts.map(account => <li key={account}>{account}</li> ) }
25     </ul>
26   </div>
27 );
28
29 class AddAccount extends Component {
30   render() {
31     return (
32       <div>
33         <Label>Add account:</Label>
34         <input type="text"
35
```

Browser Output:

Redux shopping cart example – Google Chrome
localhost:3000/#/sign-in?_k=w1o9r8

Accounts

- Billy Bob (Google)
- Mary Sue (Facebook)

Add account:

HOT RELOADING

The image shows a development environment with two windows. The left window is an IDE (JetBrains IDEA) displaying the source code for a React application. The right window is a web browser (Google Chrome) showing the rendered output of the application.

Code in containers/App.js:

```
1 import React, { Component } from 'react'
2
3 const accounts = [
4   "Billy Bob (Google)",
5   "Mary Sue (Facebook)"
6 ];
7
8 class App extends Component {...}
9
10 const Accounts = () => (
11   <div>
12     <h1>Your Accounts</h1>
13     <hr/>
14     <ul>
15       { accounts.map(account => <li key={account}>{account}</li> ) }
16     </ul>
17   </div>
18 );
19
20 class AddAccount extends Component {
21   render() {
22     return (
23       <div>
24         <label>Add account:</label>
25         <input type="text"
26           placeholder="account name"
27           style={{margin: "0 5px"}}/>
28         <button type="button">Add</button>
29       </div>
30     );
31   }
32 }
33
34 export default App;
```

Browser Output (localhost:3000/#/sign-in?_k=w1o9r8):

Your Accounts

- Billy Bob (Google)
- Mary Sue (Facebook)

Add account:

REDUX LOGGER

The screenshot shows a web browser window with the title "BOB Konf 2016 - Front-end Development with React - Google Chrome". The address bar shows "localhost:3000". The page content includes a heading "Your Accounts", a list of accounts, and an "Add account" form.

Your Accounts

- Billy Bob (Google) [Delete](#)
- Mary Sue (Facebook) [Delete](#)

Add account:

The Chrome DevTools console is open, showing a message: "[HMR] connected" from the source "client.js?3ac5:55".

REDUX LOGGER

The screenshot shows a web browser window with the title "BOB Konf 2016 - Front-end Development with React - Google Chrome". The address bar shows "localhost:3000". The main content area displays "Your Accounts" with a list of accounts:

- Billy Bob (Google) [Delete](#)
- Mary Sue (Facebook) [Delete](#)

Below the list is an "Add account:" label, a text input field containing "account name", and an "Add" button.

The Chrome DevTools console is open, showing a message: "[HMR] connected" from "client.js?3ac5:55". The console also shows "Elements", "Sources", and "Network" tabs, and a "Preserve log" checkbox.

REDUX DEBUGGER

The image shows a web browser window with the Redux DevTools debugger open. The browser tab is titled "BOB Konf 2016 - Front-end Development with React - Google Chrome". The address bar shows "localhost:3000". The page content includes a heading "Your Accounts", a list of accounts, and an "Add account" form.

Your Accounts

- Billy Bob (Google) [Delete](#)
- Mary Sue (Facebook) [Delete](#)

Add account:

The Redux DevTools debugger is open on the "Redux" tab, showing the state of the application. The state is an object with one key, "accounts", which is an empty array. The debugger also shows the "@@INIT" state.

```
@@@INIT
state: {} 1 key
  accounts: [] 2 items
```

REDUX DEBUGGER

The screenshot shows a web browser window with the URL `localhost:3000`. The page title is "Your Accounts". The page content includes:

- A heading "Your Accounts".
- A list of accounts: "Billy Bob (Google)" and "Mary Sue (Facebook)".
- Each account has a "Delete" link next to it.
- An "Add account:" form with an input field containing "account name" and an "Add" button.

The Redux DevTools debugger is open on the right side of the browser window. The "Redux" tab is selected, showing the state of the application. The state is a plain object with one key, "accounts", which is an empty array. The state is displayed as follows:

```
@@INIT
state: {} 1 key
  accounts: [] 2 items
```

REACT DEV TOOL

The screenshot shows a web browser window with the URL `localhost:3000`. The page displays a heading "Your Accounts" and a list of accounts: "Billy Bob (Google)" and "Mary Sue (Facebook)". Each account has a "Delete" link next to it. Below the list is a form labeled "Add account:" with an input field for "account name" and an "Add" button.

The React DevTools component inspector is open, showing the component tree on the left and the selected component's props and state on the right. The component tree shows the following structure:

```
Provider store={dispatch: fn(), subscribe: subscri...
  <Connect(App)>
    <App accounts=["Billy Bob (Google)", "Mary Sue (...
      <div>
        <Accounts accounts=["Billy Bob (Google)", "Me...
          <AddAccount onAdd=onAdd()>
            <div>
              <label>Add account:</label>
              <input type="text" placeholder="account r...
              <button type="button" onClick=bound proxi...
            </div>
          </AddAccount>
        </div>
      </App>
    </Connect(App)>
  </Provider>
```

The right pane shows the props for the selected `<Accounts>` component:

```
<Accounts>
  ($r in the console)
  Props read-only
  ▼ accounts: Array[2]
    0: "Billy Bob (Google)"
    1: "Mary Sue (Facebook)"
  ▶ onDelete: onDelete()
```

The breadcrumb at the bottom of the component inspector shows the path: `Provider > Connect(App) > App > div > Accounts`.

Thank you!

Tony Tsui



 @tony_tsui

tsui.tony@gmail.com

Assembly Required