Frontend Frameworks
Part 3

SWE 432, Fall 2017

Design and Implementation of Software for the Web

Where were are now

- HW9 is due next **Thurs**

- Two more lectures on front end development (today and next Tues)

- Starting Part 3 on Usability next Thurs

Today

- Q&A session on React
- React design patterns

- In class activity

- Front end routing w/ react-router

Questions about React

Organization of a React project

Todo in React

```
class TodoApp extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [], text: '' };
    this.handleChange = this.handleChange.bind(this);
    this.handleSubmit = this.handleSubmit.bind(this);
  }

  render() {
    return (
      <div>
        <h3>TODO</h3>
        <TodoList items={this.state.items} />
        <form onSubmit={this.handleSubmit}>
          <input
          onChange={this.handleChange}
          value={this.state.text}
        />
        <button>
          Add #{this.state.items.length + 1}
        </button>
      </form>
    </div>
  );
}
```

```
handleChange(e) {
    this.setState({ text: e.target.value });
 }

 handleSubmit(e) {
    e.preventDefault();
    if (!this.state.text.length) {
      return;
    }
    const newItem = {
      text: this.state.text,
      id: Date.now()
    };
    this.setState(prevState => ({
      items: prevState.items.concat(newItem),
      text: ''
    }));
  }
}

class TodoList extends React.Component {
  render() {
    return (
      <ul>
        {this.props.items.map(item => (
          <li key={item.id}>{item.text}</li>
        ))}
      </ul>
    );
  }
}

ReactDOM.render(<TodoApp />, mountNode);
```

Handling events from collections of controls

```
class Board extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      squares: Array(9).fill(null),
    };
  }

  handleClick(i) {
    const squares = this.state.squares.slice()
    squares[i] = 'X';
    this.setState({squares: squares});
  }

  renderSquare(i) {
    return (
      <Square
        value={this.state.squares[i]}
        onClick={() => this.handleClick(i)}
      />
    );
  }
```

```
  render() {
      const status = 'Next player: X';

      return (
        <div>
          <div className="status">{status}</div>
          <div className="board-row">
            {this.renderSquare(0)}
            {this.renderSquare(1)}
            {this.renderSquare(2)}
          </div>
          <div className="board-row">
            {this.renderSquare(3)}
            {this.renderSquare(4)}
            {this.renderSquare(5)}
          </div>
          <div className="board-row">
            {this.renderSquare(6)}
            {this.renderSquare(7)}
            {this.renderSquare(8)}
          </div>
        </div>
      );
    }
}
```

fetch

- Where's the best place to put a fetch request?
- When the page loads, all of the components will immediately be asked to render.
  - Don't want to block the render from happenning....
- But want the component to update itself after the data gets back.
- Solution: componentDidMount
  - Make the fetch request from componentDidMount
  - After data comes back, update state with new data
  - This will trigger component to re-render

In Class Activity

- In groups of 2

- Build a calculator in React
  - Use

Front end routing

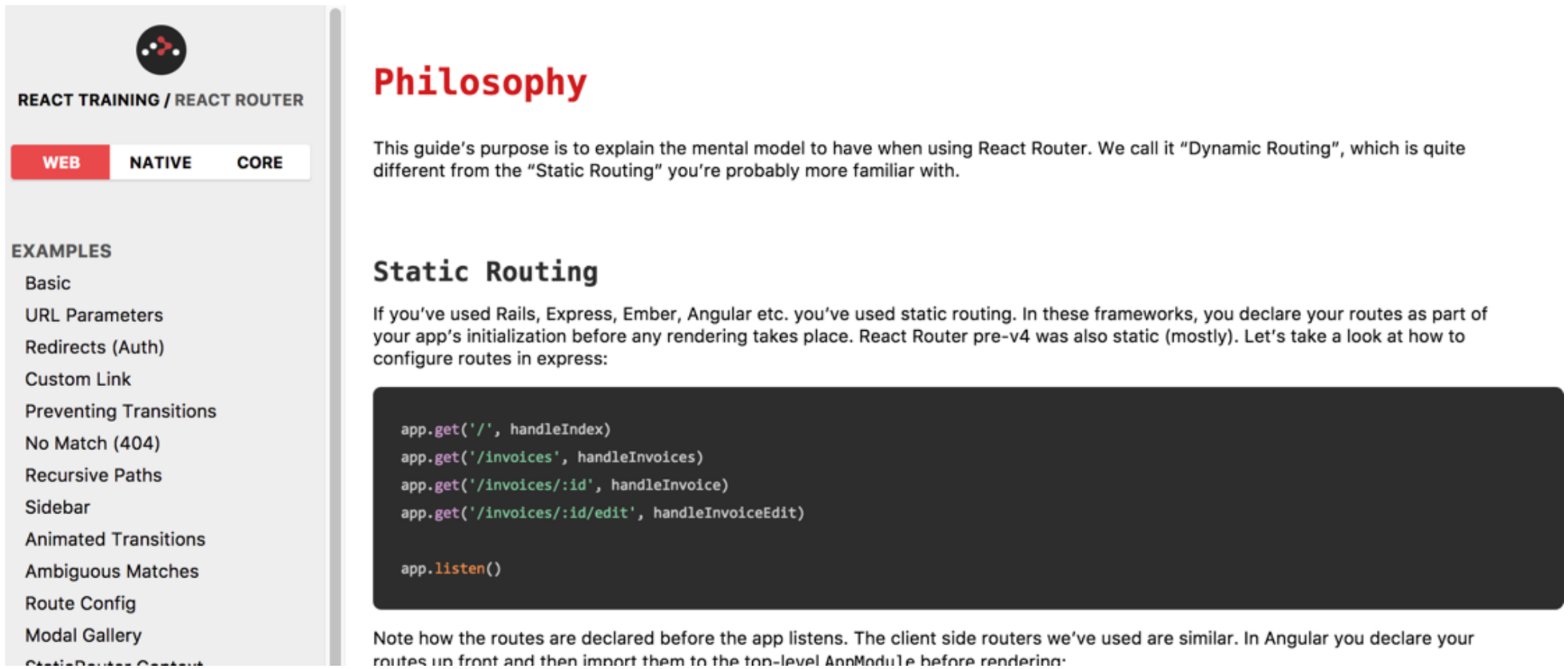
- Using state to represent views is great
- But....
  - Does not offer unique URL for each view
  - Breaks the back / forward buttons
  - Makes it harder to deep link to specific views

- Would be great to simply render a component based on the current URL
  - => front end routing

React-Router

- Install npm package in **client**

```
npm install react-router-dom
```



https://reacttraining.com/react-router/web/guides/philosophy

In App.js

```
import React from 'react'
import {
  BrowserRouter as Router,
  Route,
  Link
} from 'react-router-dom'

const Home = () => (
  <div>
    <h2>Home</h2>
  </div>
)

const About = () => (
  <div>
    <h2>About</h2>
  </div>
)

const Topic = ({ match }) => (
  <div>
    <h3>{match.params.topicId}</h3>
  </div>
)
```

```
const Topics = ({ match }) => (
  <div>
    <h2>Topics</h2>
    <ul>
      <li>
        <Link to={`${match.url}/rendering`}>
          Rendering with React
        </Link>
      </li>
      <li>
        <Link to={`${match.url}/components`}>
          Components
        </Link>
      </li>
      <li>
        <Link to={`${match.url}/props-v-state`}>
          Props v. State
        </Link>
      </li>
    </ul>

    <Route path={`${match.url}/:topicId`} component={Topic}/>
    <Route exact path={match.url} render={() => (
      <h3>Please select a topic.</h3>
    )}/>
  </div>
)

const BasicExample = () => (
  <Router>
    <div>
      <ul>
        <li><Link to="/">Home</Link></li>
        <li><Link to="/about">About</Link></li>
        <li><Link to="/topics">Topics</Link></li>
      </ul>

      <hr/>

      <Route exact path="/" component={Home}/>
      <Route path="/about" component={About}/>
      <Route path="/topics" component={Topics}/>
    </div>
  </Router>
)
export default BasicExample
```