Building Smart .NET Microservice Clients

.NET FRINGE 2016

PRESENTERS: BEAU PALMQUIST AND JARED SCHAAB

Brief Introduction

- Beau Palmquist
  - IT Developer at the Home Depot QuoteCenter
  - https://github.com/beaupalmquist-hdqc
  - https://beaupalmquist.me/
  - @Beau_Palmquist

- Jared Schaab
  - Senior IT Developer at Home Depot QuoteCenter
  - http://stackoverflow.com/users/603520/jared
  - https://github.com/schaab
  - http://www.jaredschaab.com/
  - @JaredSchaab

What will be covered

- How to host a pure JS client in a NET Core 1.0 web application
- How to write a simple JS client that consumes .NET microservices
- How to build React JS web applications in a .NET Core world
- How to implement a simple SPA middleware for .NET core 1.0
- How to incorporate React-Router into a React application
- How to incorporate Redux into a React application

What will not be covered

- How to build microservices
  - Jesse Johnston has written a great collection of blogs on this topic: http://teamjohnston.net/blog/category/microservices/
- How to build Isomorphic JS applications
- How to become a Webpack expert
- How to take over the world

Tools of the Trade

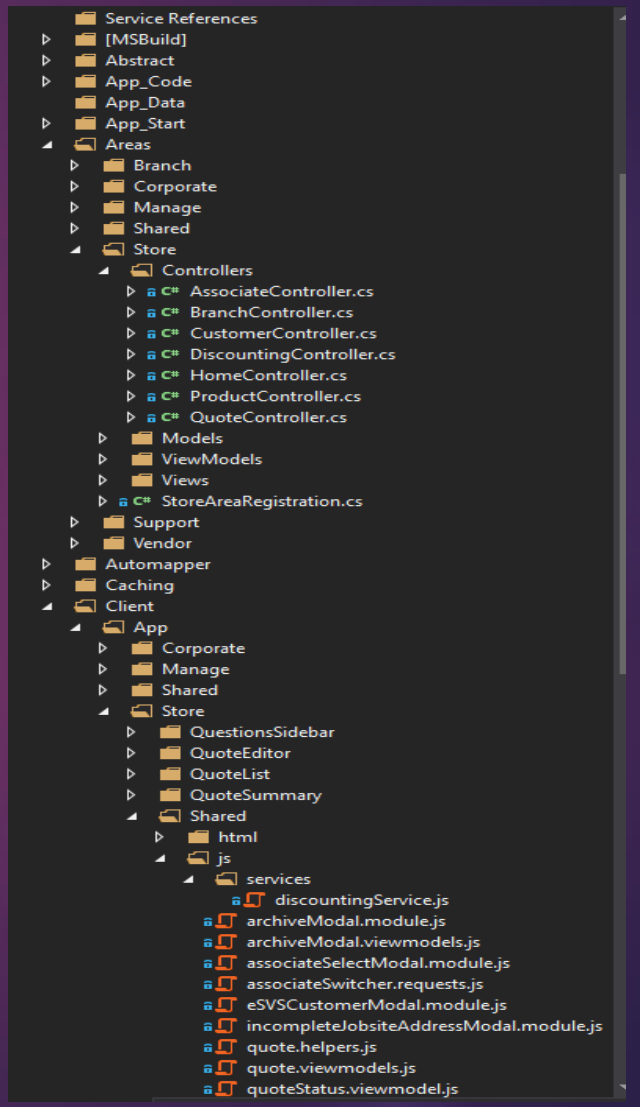
- Install Latest Node and NPM
  - https://nodejs.org/en/download/
  - https://docs.npmjs.com/getting-started/installing-node
- .NET Core 1.0 - https://go.microsoft.com/fwlink/?LinkID=809124 (mac)
- .NET Core 1.0 - https://go.microsoft.com/fwlink/?LinkID=809122 (win)
- https://dotnetfringeslack.herokuapp.com/
- Git Client
  - Git Kraken (Recommended) – https://www.gitkraken.com/
  - GitHub Desktop – https://desktop.github.com/
  - Git Bash (Installed with VS Community)
- Client IDEs
  - Atom - https://atom.io/
  - Brackets - http://brackets.io/
  - WebStorm ($$$) - https://www.jetbrains.com/webstorm/
  - Visual Studio Code - https://code.visualstudio.com/
- You can use Visual Studio but you will need
  - Visual Studio - https://www.visualstudio.com/en-us/products/visual-studio-community-vs.aspx
  - Visual Studio Update 3 - https://www.visualstudio.com/news/releasenotes/vs2015-update3-vs

GitHub Repo

- https://github.com/beaupalmquist-hdqc/microservice-client-workshop-2016
- The repo consists of four project categories
  - Vanilla JS
  - React
  - React-Router
  - Redux
- Each project category contains two projects, one that is the complete project and the other that is the starter project
- We will be working with the starter projects
- .NET Core class library that contains middleware for hosting a SPA in .NET Core

The Monolithic Web Application

- One project to rule them all (contains):
  - Business Logic
  - Views
  - Controllers
  - Data Models
  - View Models
  - Scripts



Why React and Microservices?

What are Microservices

- A logically connected group of web APIs
- Hosted independently from the consuming web app
- More granular and scalable than a monolithic web app back-end

Microservices in .NET

- ASP.NET provides a generic web API platform
- Oriented to building a monolithic web application

Forge Microservice Framework

- Built at Home Depot over the last year
- Forge team: Jesse Johnston, Beau Palmquist, Jared Schaab
- Based on ASP.NET Core 1.0 middleware
- Platform infrastructure is built on microservices
  - Deployment
  - Routing
  - Identity and Security
- Provider-based and technology-agnostic

Why React?

- Backend agnostic
- Reusable Components
- Unit testable
- Other Frameworks were too heavy

Lab #1: Vanilla JS Web App

SIMPLEST CLIENT WE CAN WRITE

Vanilla JS Web App

- Project Details
  - Static HTML pages
  - Pure JS client that communicates with two microservice API endpoints
    - One endpoint returns a random quote
    - Second endpoint returns a list of all quotes
  - No additional JS frameworks required
  - Basic styling using Bootstrap css only, no Bootstrap JS
  - Microservice API proxy scripts referenced via script tags
    - Support proxy scripts
    - Service proxy scripts

Vanilla JS Web App - *Startup.cs*

```
public class Startup
{
    public void Configure(
        IApplicationBuilder app,
        IHostingEnvironment env,
        ILoggerFactory loggerFactory)
    {
        loggerFactory.AddConsole();

        app.UseDefaultFiles();
        app.UseStaticFiles();
    }
}
```

Vanilla JS Web App - *Program.cs*

```
public static void Main(string[] args)
{
    var host = new WebHostBuilder()
            .UseKestrel()
            .UseContentRoot(Directory.GetCurrentDirectory())
            .UseStartup<Startup>()
            .Build();

    host.Run();
}
```

Vanilla JS Web App – *Index.html*

Markup

```
<body>
<nav class="navbar navbar-inverse">
    <div class="container-fluid">
        <div class="navbar-header">
            <a class="navbar-brand" href="#">Vanilla JS</a>
        </div>
        <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
            <ul class="nav navbar-nav">
                <li class="active"><a href="#">Home</a></li>
                <li><a href="list.html">Quotes</a></li>
            </ul>
        </div>
    </div>
</nav>
<div class="container-fluid">
    <h2>Random Quote</h2>
    <h4 id="quote"></h4>
    <button class="btn btn-primary" onclick="getRandomQuote()">Get Random Quote</button>
</div>
// …
// …
// …
</body>
```

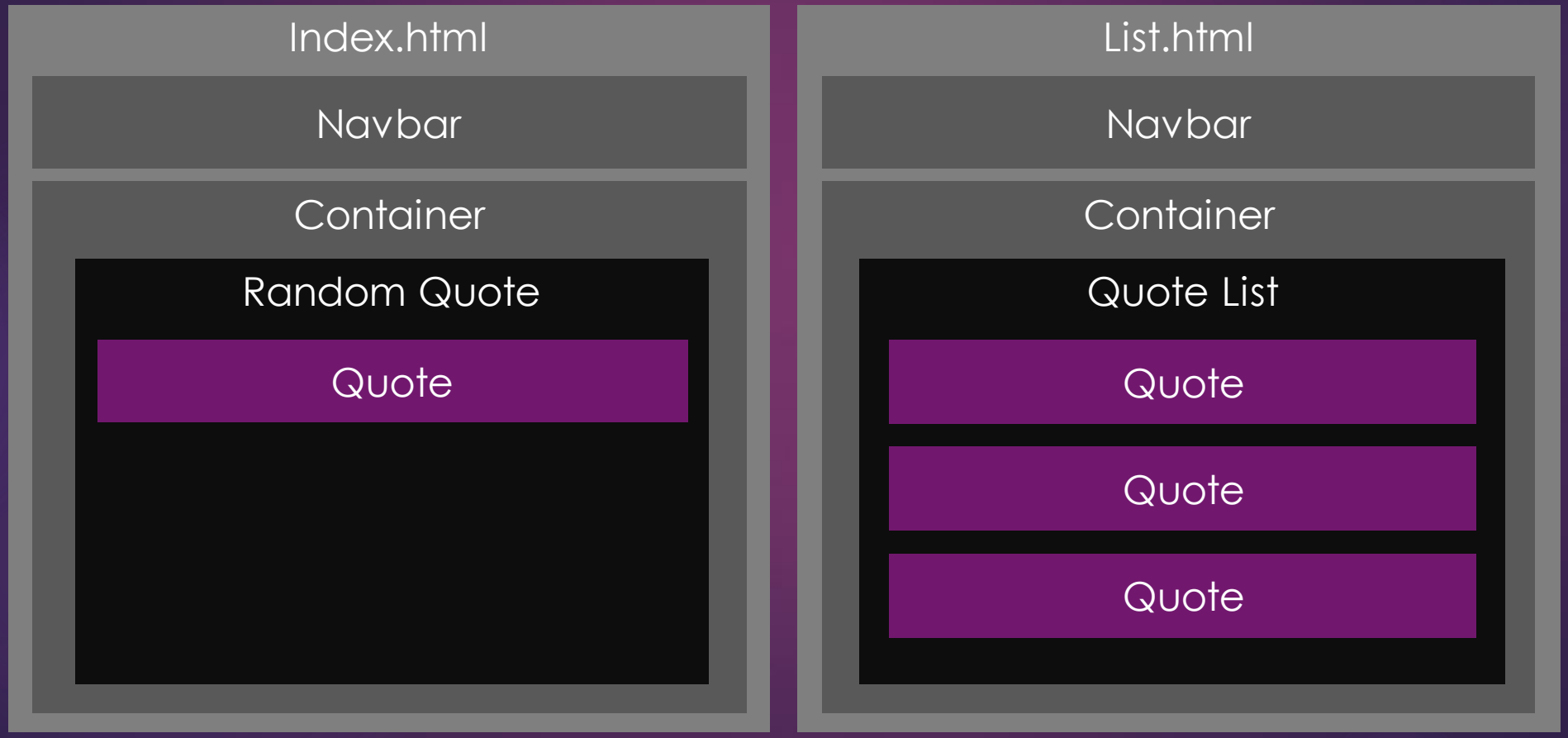
Vanilla JS Web App – *Index.html*

Support and Service proxy scripts

```
<body>
  // …
  // …
  // …
  <script src="http://dev-forge.api.hdquotecenter.com/serviceProxySupport/ajax.js"></script>
  <script src="http://dev-forge.api.hdquotecenter.com/serviceProxySupport/auth.js"></script>
  <script src="http://dev-forge.api.hdquotecenter.com/serviceProxySupport/notifications.js"></script>
  <script src="http://dev-forge.api.hdquotecenter.com/serviceProxySupport/ajaxOptions.js"></script>
  <script src="http://dev-forge.api.hdquotecenter.com/serviceProxySupport/promise.js"></script>
  <script src="http://dev-
forge.api.hdquotecenter.com/serviceProxies/HomeDepot.Platform.Samples.FunnyQuote.Microservice/1.0.1-build-
5/funnyQuote.js"></script>
  <script src="http://dev-forge.api.hdquotecenter.com/serviceProxies/HomeDepot.Platform.Identity.Microservice/1.0.1-build-
7/token.js"></script>
  <script>
      function getRandomQuote(){
          FunnyQuote.setOptions({ baseUri: 'http://dev-forge.api.hdquotecenter.com' });
          FunnyQuote.getRandomQuote().then(function(quote){
             var para = document.getElementById('quote');
              para.innerText=quote.Text;
          });
      };

      document.addEventListener('DOMContentLoaded', getRandomQuote());
  </script>
</body>
```

Vanilla JS Web App – *UI Diagram*



Vanilla JS Web App - Weaknesses

- Client is not modularized
- Redundant markup
- Navigation is clunky
- Data is not shared between views
- No reusable UI elements

Lab #2: React Web App

LETS GET A LITTLE FANCIER

React Fundamentals

- Declarative
  - Declare simple views for each state in your application
  - Results in better predictability and easier debugging
- Component-Based
  - Encapsulated components that manage their own state
  - Compose components into complex UIs
- Learn Once, Write Anywhere
  - React is agnostic to the rest of your tech stack
  - Develop new features in React without rewriting existing code

React JS SPA

- Project Details
  - React JS client that communicates with two microservice API endpoints
    - One endpoint returns a random quote
    - Second endpoint returns a list of all quotes
  - Utilizes React and React-DOM for rendering
  - UI elements component-ized
  - Microservice API proxy scripts downloaded and bundled into one bundle.js file that is imported into index.html
  - One entry point – index.html

React JS SPA - *Startup.cs*

```
public class Startup
{
    public void Configure(
        IApplicationBuilder app,
        IHostingEnvironment env,
        ILoggerFactory loggerFactory)
    {
        loggerFactory.AddConsole();

        app.UseDefaultFiles();
        app.UseStaticFiles();
    }
}
```

React JS SPA - *Program.cs*

```
public static void Main(string[] args)
{
    var host = new WebHostBuilder()
            .UseKestrel()
            .UseContentRoot(Directory.GetCurrentDirectory())
            .UseStartup<Startup>()
            .Build();

    host.Run();
}
```

React JS SPA – *Components*

App.js

```
import React, { Component } from 'react';
import Navbar from './Navbar';
import RandomQuote from './RandomQuote';
import QuoteList from './QuoteList';

export default class App extends Component {
    state = {
      activeLink: 'Home'
    };
    setActiveLink = (link) => {
        this.setState({activeLink: link});
    };
    render() {
        let content;

        switch(this.state.activeLink){
            case 'Home':{
                content = (<RandomQuote/>);
                break;
            }
            case 'Quotes': {
                content = (<QuoteList/>)
                break;
            }
            default:{
                content = (<h4>Unknown state</h4>);
                break;
            }
        }

        return (
            <div>
                {Navbar(this.setActiveLink, this.state.activeLink)}
                {content}
            </div>
        );
    }
}
```

React JS SPA – *Components*

Navbar.js

```
import React, { PropTypes } from 'react';

const Navbar = (setActiveLink, activeLink) =>
{
    return (<nav className="navbar navbar-inverse">
        <div className="container-fluid">
            <div className="navbar-header">
                <a className="navbar-brand" href="#">React Demo</a>
            </div>
            <div className="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
                <ul className="nav navbar-nav">
                     <li className={activeLink === 'Home' ? 'active' : ''}>
                            <a href="#" onClick={() => setActiveLink('Home')}>Home</a>
                     </li>
                     <li className={activeLink === 'Quotes' ? 'active' : ''}>
                            <a href="#" onClick={() => setActiveLink('Quotes')} >Quotes</a>
                    </li>
                </ul>
            </div>
        </div>
    </nav>);
}

Navbar.propTypes = {
    setActiveLink: PropTypes.func.isRequired,
    activeLink: PropTypes.string.isRequired
};

export default Navbar;
```

React JS SPA – *Components*

RandomQuote.js

```
import React, {Component} from 'react';
import FunnyQuote from '../common/js/forge/services/funnyQuote';
import Quote from './Quote';

export default class RandomQuote extends Component{
    state = {
        quote: undefined
    }
    componentWillMount(){
        if(!this.state.quote) {
            this.getRandomQuote();
        }
    }
    getRandomQuote = () => {
        FunnyQuote.setOptions({ baseUri: 'http://dev-forge.api.hdquotecenter.com' });
        FunnyQuote.getRandomQuote().then(quote => {
            this.setState({quote: quote});
        });
    };
    render(){
        return (
            <div className="container-fluid">
                <h2>Random Quote</h2>
                {Quote(this.state.quote)}
                <button className="btn btn-primary" onClick={this.getRandomQuote}>Get Random Quote</button>
            </div>
        );
    }
}
```

React JS SPA – *Components*

QuoteList.js

```
import React, { Component } from 'react';
import FunnyQuote from '../common/js/forge/services/funnyQuote';
import Quote from './Quote';

export default class QuoteList extends Component{
    state = {
        quotes: []
    };
    componentWillMount(){
        FunnyQuote.setOptions({ baseUri: 'http://dev-forge.api.hdquotecenter.com' });
        FunnyQuote.getAll().then((quotes) => {
            const items = [];
            quotes.forEach((quote) => {
                const key = `quote_item_${quote.Id}`;
                items.push(<li key={key} className="list-group-item">{Quote(quote)}</li>);
            });

            this.setState({quotes: items});
        });
    }
    render(){
        return (<div className="container-fluid">
            <h2>All Quotes</h2>
            <ul className="list-group">
                {this.state.quotes}
            </ul>
        </div>);
    }
}
```

React JS SPA – *Components*

Quote.js

```
import React, { PropTypes } from 'react';

const Quote = (quote) =>
{
  const quoteText = `"${quote ? quote.Text : ""}"`
  return (<div className="well well-sm"><p>{quoteText}</p></div>);
}

Quote.propTypes = {
  quote: PropTypes.object.isRequired
};

export default Quote;
```

React JS SPA – *index.js*

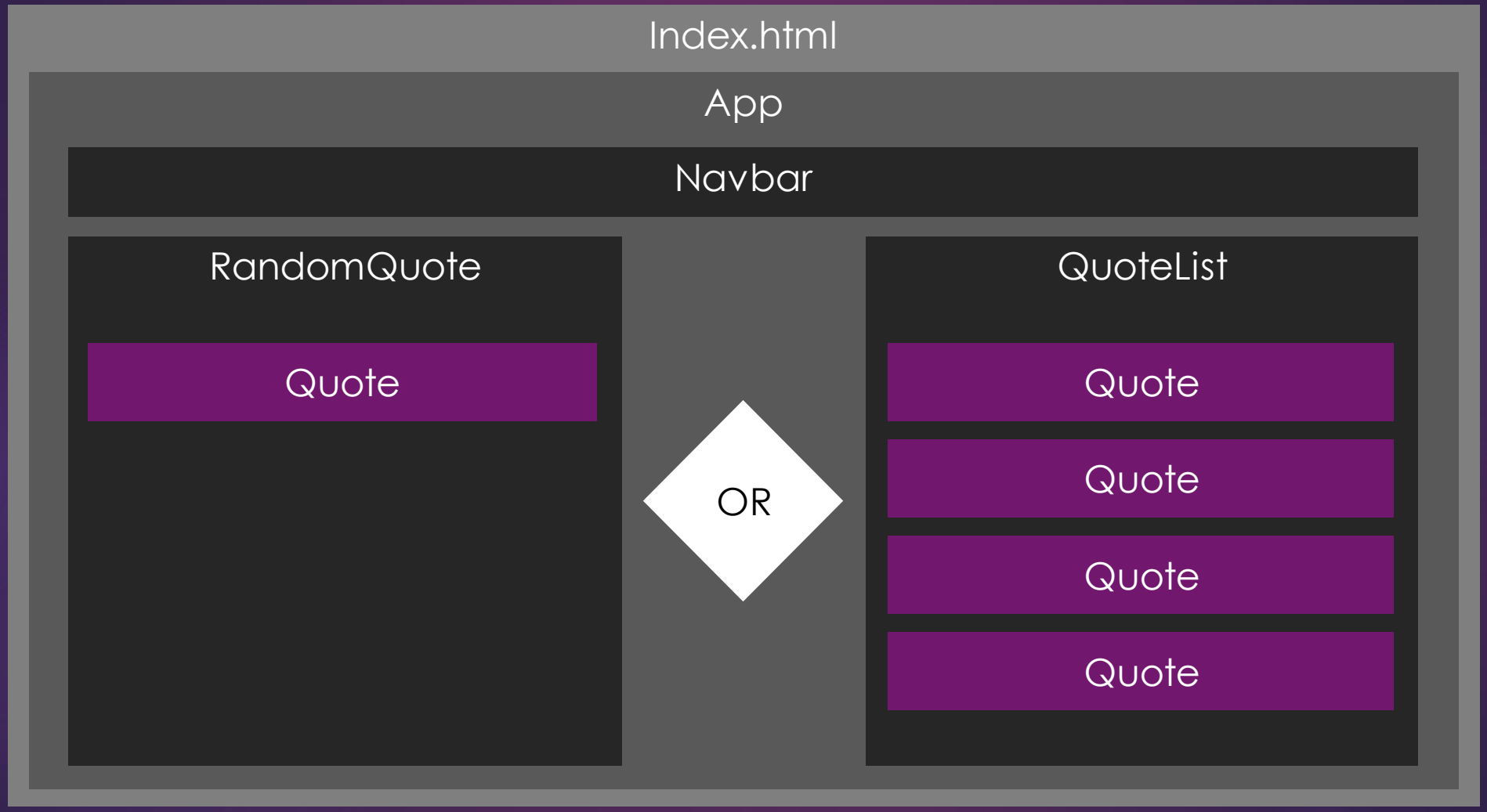
```
import React from 'react';
import {render} from 'react-dom';
import App from './components/App';

export default (() => {
    render((<App/>), document.getElementById('root'))
})();
```

React JS SPA – *index.html*

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Complete React Demo</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css"
integrity="sha384-1q8mTJOASx8j1Au+a5WDVnPi2lkFfwwEAa8hDDdjZlpLegxhjVME1fgjWPGmkzs7" crossorigin="anonymous">
</head>
<body>
    <div id="root"></div>
    <script src="src/js/bundle.js"></script>
</body>
</html>
```

React JS SPA – *UI Diagram*



React JS SPA - Weaknesses

- No browser history
- No route or deep link support
- All data is stored in component state
- Data cannot be shared between components

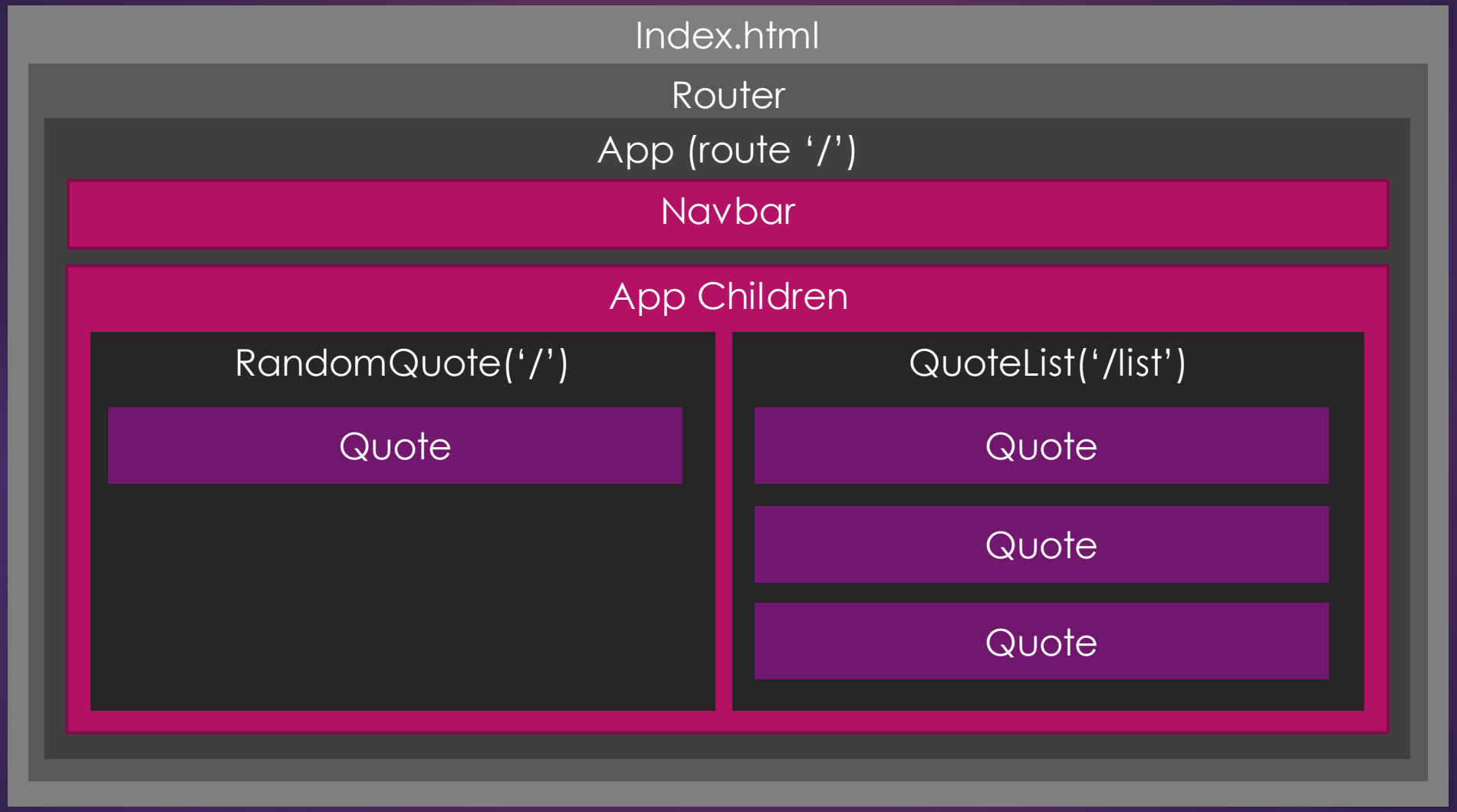
React-Router Fundamentals

- Routes
  - The composition of a React component and a specific application path
- History
  - Maintains the browser history on the client
- Links
  - Special React components that provide navigational elements that are compliant with react-router and do not result in a request to the server
- Router
  - Provides a mechanism for implementing programmatic navigation
- Location
  - Global variable that contains relevant information about the current path

React-Router SPA

- Project Details
  - React-Router client that communicates with two microservice API endpoints
  - Utilizes React and React-DOM for rendering
  - UI elements component-ized
  - Microservice API proxy scripts downloaded and bundled into one bundle.js file that is imported into index.html
  - One entry point – index.html
  - Support for HTML5 browser history
  - Support for routing and deep links

React-Router SPA – *UI Diagram*



React-Router SPA- *Startup.cs*

```
public class Startup
{
    public void Configure(
        IApplicationBuilder app,
        IHostingEnvironment env,
        ILoggerFactory loggerFactory)
    {
        loggerFactory.AddConsole();

        app.UseSpaMode();
        app.UseDefaultFiles();
        app.UseStaticFiles();
    }
}
```

React-Router SPA- *Program.cs*

```
public static void Main(string[] args)
{
    var host = new WebHostBuilder()
            .UseKestrel()
            .UseContentRoot(Directory.GetCurrentDirectory())
            .UseStartup<Startup>()
            .Build();

    host.Run();
}
```

React-Router SPA – *Components*

App.js

```
import React, { Component } from 'react';
import Navbar from './Navbar';

export default class App extends Component {
    render() {
        return (
            <div>
                {Navbar()}
                {this.props.children}
            </div>
        );
    }
}
```

React-Router SPA – *Components*

Navbar.js

```
import React from 'react';
import { Link } from 'react-router';

const Navbar = () => {
    return (<nav className="navbar navbar-inverse">
        <div className="container-fluid">
            <div className="navbar-header">
             <Link className="navbar-brand" to="/">
                    React Router Demo
             </Link>
            </div>
            <div className="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
                <ul className="nav navbar-nav">
                    <li className={location.pathname === '/' ? 'active' : ''}><Link to="/">Home</Link></li>
                    <li className={location.pathname === '/list' ? 'active' : ''}><Link to="/list">Quotes</Link></li>
                </ul>
            </div>
        </div>
    </nav>);
};

export default Navbar;
```

React-Router SPA – *Components*

RandomQuote.js

```
import React, {Component} from 'react';
import FunnyQuote from '../common/js/forge/services/funnyQuote';
import Quote from './Quote';

export default class RandomQuote extends Component{
    state = {
        quote: undefined
    }
    componentWillMount(){
        if(!this.state.quote) {
            this.getRandomQuote();
        }
    }
    getRandomQuote = () => {
        FunnyQuote.setOptions({ baseUri: 'http://dev-forge.api.hdquotecenter.com' });
        FunnyQuote.getRandomQuote().then(quote => {
            this.setState({quote: quote});
        });
    };
    render(){
        return (
            <div className="container-fluid">
                <h2>Random Quote</h2>
                {Quote(this.state.quote)}
                <button className="btn btn-primary" onClick={this.getRandomQuote}>Get Random Quote</button>
            </div>
        );
    }
}
```

React-Router SPA – *Components*

QuoteList.js

```
import React, { Component } from 'react';
import FunnyQuote from '../common/js/forge/services/funnyQuote';
import Quote from './Quote';

export default class QuoteList extends Component{
    state = {
        quotes: []
    };
    componentWillMount(){
        FunnyQuote.setOptions({ baseUri: 'http://dev-forge.api.hdquotecenter.com' });
        FunnyQuote.getAll().then((quotes) => {
            const items = [];
            quotes.forEach((quote) => {
                const key = `quote_item_${quote.Id}`;
                items.push(<li key={key} className="list-group-item">{Quote(quote)}</li>);
            });

            this.setState({quotes: items});
        });
    }
    render(){
        return (<div className="container-fluid">
            <h2>All Quotes</h2>
            <ul className="list-group">
                {this.state.quotes}
            </ul>
        </div>);
    }
}
```

React-Router SPA – *Components*

Quote.js

```
import React, { PropTypes } from 'react';

const Quote = (quote) =>
{
  const quoteText = `"${quote ? quote.Text : ""}"`
  return (<div className="well well-sm"><p>{quoteText}</p></div>);
}

Quote.propTypes = {
  quote: PropTypes.object.isRequired
};

export default Quote;
```

React-Router SPA – *index.js*

```
import React from 'react';
import {render} from 'react-dom';
import App from './components/App';
import RandomQuote from './components/RandomQuote';
import QuoteList from './components/QuoteList';
import { Router, Route, IndexRoute, browserHistory } from 'react-router';

export default (() => {
    render((
        <Router history={browserHistory}>
            <Route path="/" component={App}>
                <IndexRoute component={RandomQuote}/>
                <Route path="/list" component={QuoteList}/>
            </Route>
        </Router>
        ), document.getElementById('root'))
})();
```

React-Router SPA – *index.html*

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Complete React Demo</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css"
integrity="sha384-1q8mTJOASx8j1Au+a5WDVnPi2lkFfwwEAa8hDDdjZlpLegxhjVME1fgjWPGmkzs7" crossorigin="anonymous">
</head>
<body>
    <div id="root"></div>
    <script src="src/js/bundle.js"></script>
</body>
</html>
```

React-Router SPA - Weaknesses

- All data is stored at the component state level
- Data is not kept in sync between components

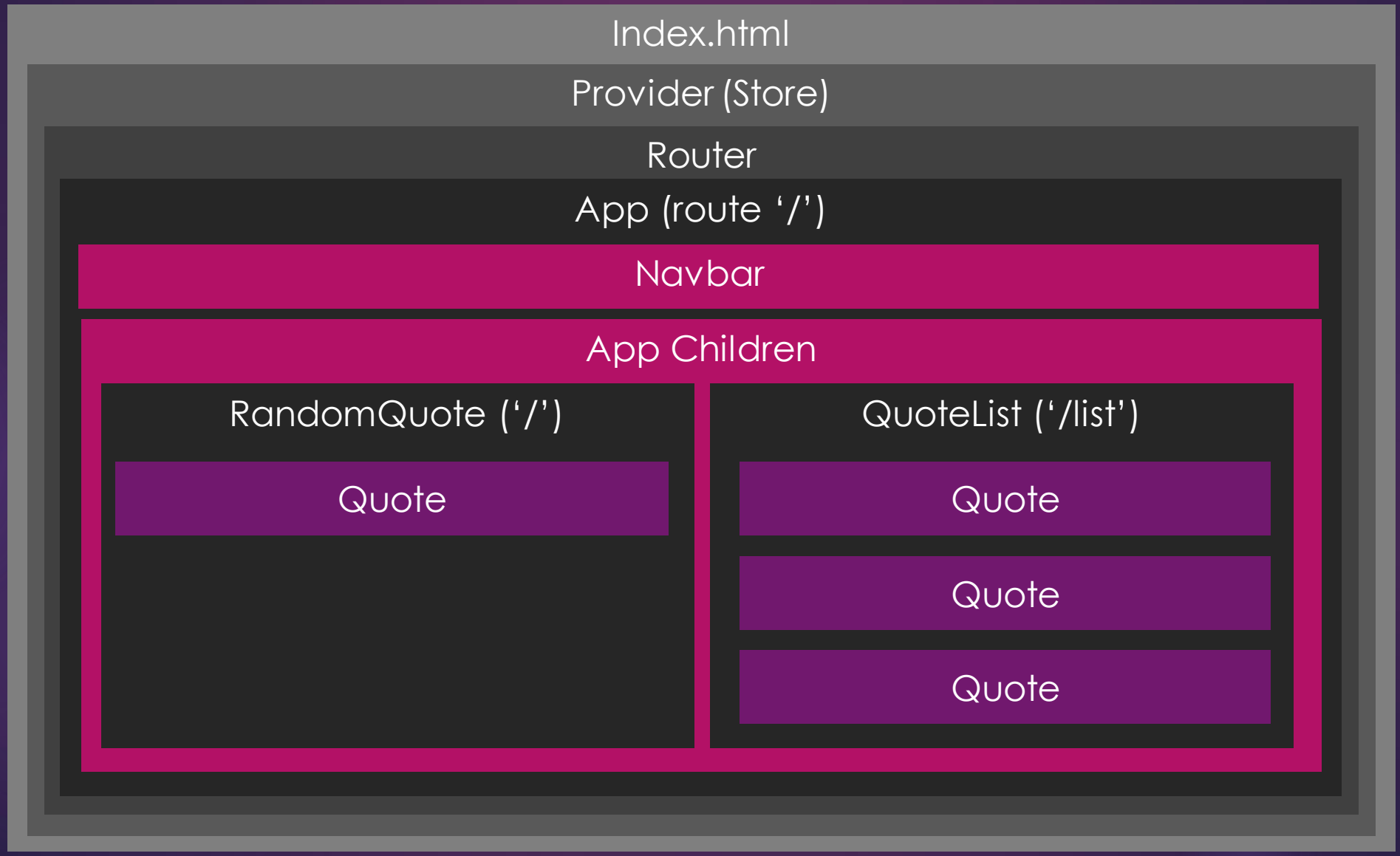
Redux Fundamentals

- Single source of truth for the client
  - State of entire application stored in an object tree within a single store
- App State is read-only (immutable)
  - Only way to mutate state is to emit an ***action***
  - ***Actions*** describe what happened
- Changes to state are made with ***reducers***
  - ***Reducers*** are pure functions that take previous state and an action and return the next state
  - An important thing to remember with reducers is to return a new state object instead of mutating the previous state
  - Reducers represent a way to split your data store into more specific parts of the state tree

React/Redux SPA

- Project Details
  - React-Redux client that communicates with a microservice
  - Rendering provided by React
  - UI elements component-ized
  - Microservice API proxy scripts downloaded and bundled into one bundle.js file that is imported into index.html
  - One entry point – index.html
  - Routing and browser history provided by React-Router
  - Client side data store provided by Redux
  - One way data flow provided by React-Redux

React/Redux SPA – *UI Diagram*



React/Redux SPA- *Startup.cs*

```
public class Startup
{
    public void Configure(
        IApplicationBuilder app,
        IHostingEnvironment env,
        ILoggerFactory loggerFactory)
    {
        loggerFactory.AddConsole();

        app.UseSpaMode();
        app.UseDefaultFiles();
        app.UseStaticFiles();
    }
}
```

React/Redux SPA- *Program.cs*

```
public static void Main(string[] args)
{
    var host = new WebHostBuilder()
            .UseKestrel()
            .UseContentRoot(Directory.GetCurrentDirectory())
            .UseStartup<Startup>()
            .Build();

    host.Run();
}
```

React/Redux SPA – *Containers*

AppRoot.js

```
import React, {Component, PropTypes } from 'react';
import { bindActionCreators } from 'redux';
import { connect } from 'react-redux';
import * as AppActionCreators from '../actions/App';
import Navbar from '../components/Navbar';

class AppRoot extends Component {
    static propTypes = {
        appData: PropTypes.object.isRequired,
        children: PropTypes.object,
        dispatch: PropTypes.func.isRequired
    };

    render() {
        const { children, appData, dispatch } = this.props;
        const appActions = bindActionCreators(AppActionCreators, dispatch);

        return (
            <div>
                {Navbar()}
                {children && React.cloneElement(children, { appData, appActions })}
            </div>
        );
    }
}

const mapStateToProps = (state) => {
    return ({
        appData: state.appData
    });
}

export default connect(mapStateToProps)(AppRoot);
```

React/Redux SPA – *Reducers*

App.js

```
import * as AppActionTypes from '../actionTypes/App';
import {combineReducers} from 'redux';

const initialize = () => ({
    quotes: [],
    randomQuote: null,
    loading: false
});

const App = (state = initialize(), action) => {
    switch(action.type) {
        case AppActionTypes.LOADING_QUOTES:
            return {
                ...state,
                loading: true
            };
        case AppActionTypes.LOADED_QUOTES:
            return {
                ...state,
                quotes: action.data,
                loading: false
            };
        case AppActionTypes.LOADING_RANDOM_QUOTE:
            return {
                ...state,
                loading: true
            };
        case AppActionTypes.LOADED_RANDOM_QUOTE:
            return {
                ...state,
                randomQuote: action.data,
                loading: false
            };
        default:
            return state;
    }
}

const reducer = combineReducers({
    appData: App
});

export default reducer;
```

React/Redux SPA – *Actions*

App.js

```
import * as AppActionTypes from '../actionTypes/App';
import FunnyQuote from '../common/js/forge/services/funnyQuote';

export function loadQuotes() {
    return (dispatch) => {
        dispatch({type: AppActionTypes.LOADING_QUOTES});
        FunnyQuote.setOptions({ baseUri: 'http://dev-forge.api.hdquotecenter.com' });
        FunnyQuote.getAll().then(quotes => {
           dispatch({type: AppActionTypes.LOADED_QUOTES, data: quotes});
        });
    };
}

export function getRandomQuote() {
    return (dispatch) => {
        dispatch({type: AppActionTypes.LOADING_RANDOM_QUOTE});
        FunnyQuote.setOptions({ baseUri: 'http://dev-forge.api.hdquotecenter.com' });
        FunnyQuote.getRandomQuote()
            .then(quote => dispatch({type: AppActionTypes.LOADED_RANDOM_QUOTE, data: quote}));
    };
}
```

React/Redux SPA – *ActionTypes*

App.js

```
export const LOADING_QUOTES = 'App/LOADING_QUOTES';
export const LOADED_QUOTES = 'App/LOADED_QUOTES';
export const LOADING_RANDOM_QUOTE = 'App/LOADING_RANDOM_QUOTE';
export const LOADED_RANDOM_QUOTE = 'App/LOADED_RANDOM_QUOTE';
```

React/Redux SPA – *Store*

CreateStore.js

```
import { createStore, applyMiddleware } from 'redux';
import thunk from 'redux-thunk';
import createLogger from 'redux-logger';
import appReducer from './reducers/App';

export default function() {
    const logger = createLogger();
    return createStore(appReducer, applyMiddleware(thunk, logger));
}
```

React/Redux SPA – *index.js*

```
import React from 'react';
import {render} from 'react-dom';
import { Provider } from 'react-redux';
import AppRoot from './containers/AppRoot';
import RandomQuote from './components/RandomQuote';
import QuoteList from './components/QuoteList';
import { Router, Route, IndexRoute, browserHistory } from 'react-router';
import ConfigureStore from './CreateStore';

export default (() => {
      const store = ConfigureStore();
      render((<Provider store = {store} >
                <Router history={browserHistory} >
                    <Route path="/" component={AppRoot}>
                        <IndexRoute component={RandomQuote} />
                        <Route path="/list" component={QuoteList} />
                    </Route>
                </Router>
            </Provider>),
        document.getElementById('root'));
    }
)();
```

React/Redux SPA – *Components*

Navbar.js

```
import React from 'react';
import { Link } from 'react-router';

const Navbar = () => {
    return (<nav className="navbar navbar-inverse">
        <div className="container-fluid">
            <div className="navbar-header">
             <Link className="navbar-brand" to="/">
                    React Router Demo
             </Link>
            </div>
            <div className="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
                <ul className="nav navbar-nav">
                    <li className={location.pathname === '/' ? 'active' : ''}><Link to="/">Home</Link></li>
                    <li className={location.pathname === '/list' ? 'active' : ''}><Link to="/list">Quotes</Link></li>
                </ul>
            </div>
        </div>
    </nav>);
};

export default Navbar;
```

React/Redux SPA – *Components*

RandomQuote.js

```
import React, {Component} from 'react';
import FunnyQuote from '../common/js/forge/services/funnyQuote';
import Quote from './Quote';

export default class RandomQuote extends Component{
    state = {
        quote: undefined
    }
    componentWillMount(){
        if(!this.state.quote) {
            this.getRandomQuote();
        }
    }
    getRandomQuote = () => {
        FunnyQuote.setOptions({ baseUri: 'http://dev-forge.api.hdquotecenter.com' });
        FunnyQuote.getRandomQuote().then(quote => {
            this.setState({quote: quote});
        });
    };
    render(){
        return (
            <div className="container-fluid">
                <h2>Random Quote</h2>
                {Quote(this.state.quote)}
                <button className="btn btn-primary" onClick={this.getRandomQuote}>Get Random Quote</button>
            </div>
        );
    }
}
```

React/Redux SPA – *Components*

QuoteList.js

```
import React, { Component } from 'react';
import FunnyQuote from '../common/js/forge/services/funnyQuote';
import Quote from './Quote';

export default class QuoteList extends Component{
    state = {
        quotes: []
    };
    componentWillMount(){
        FunnyQuote.setOptions({ baseUri: 'http://dev-forge.api.hdquotecenter.com' });
        FunnyQuote.getAll().then((quotes) => {
            const items = [];
            quotes.forEach((quote) => {
                const key = `quote_item_${quote.Id}`;
                items.push(<li key={key} className="list-group-item">{Quote(quote)}</li>);
            });

            this.setState({quotes: items});
        });
    }
    render(){
        return (<div className="container-fluid">
            <h2>All Quotes</h2>
            <ul className="list-group">
                {this.state.quotes}
            </ul>
        </div>);
    }
}
```

React/Redux SPA – *Components*

Quote.js

```
import React, { PropTypes } from 'react';

const Quote = (quote) =>
{
  const quoteText = `"${quote ? quote.Text : ""}"`
  return (<div className="well well-sm"><p>{quoteText}</p></div>);
}

Quote.propTypes = {
  quote: PropTypes.object.isRequired
};

export default Quote;
```

React/Redux SPA – *index.html*

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Complete React Demo</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css"
integrity="sha384-1q8mTJOASx8j1Au+a5WDVnPi2lkFfwwEAa8hDDdjZlpLegxhjVME1fgjWPGmkzs7" crossorigin="anonymous">
</head>
<body>
    <div id="root"></div>
    <script src="src/js/bundle.js"></script>
</body>
</html>
```

React/Redux SPA - Weaknesses

- Nothing – Redux is perfect and is the answer to all my dreams!!
- Well, not quite…
  - There is a decent amount of overhead to setting up the entire Redux pipeline
  - The Redux concepts are not always easy to grok
    - Dispatch, Actions and Reducers… oh my!
  - Redux really shines when your application begins to grow, so using it comes down to how much your application will need to scale
  - I encourage you to wire up the Redux pipeline earlier in the development life cycle rather than go back and retrofit your application

Recap

- What did we learn?
  - It is easy to consume Forge .NET microservices from a pure JS client
  - React allows you to organize your views into discrete components and reduces UI redundancy
  - React-Router provides you with the tools required to make a responsive SPA with complete deep-link support
  - Redux provides a powerful data flow pipeline that makes it easier to provide data to the right parts of your application

Client Tools and Technologies

- I encourage you all to learn more about some of the client tools and technologies that went into this workshop:
  - NPM - https://docs.npmjs.com/getting-started/what-is-npm
  - Webpack – http://webpack.github.io/docs/what-is-webpack.html
  - Babel – https://babeljs.io/
  - Eslint - http://eslint.org/docs/about/
  - ECMAScript 2015
    - Starter - https://babeljs.io/docs/learn-es2015/
    - Complete Reference - http://www.ecma-international.org/ecma-262/6.0/
- Webpack is by far the most powerful, but confusing weapon in this arsenal, so if you feel overwhelmed, ask questions often

What now…?

- Learn more about the Forge .NET Microservice platform
- Start building up your React component toolbox
- Incorporate Unit Tests into your clients