REACT.JS SPA'S ON SHAREPOINT USING REST

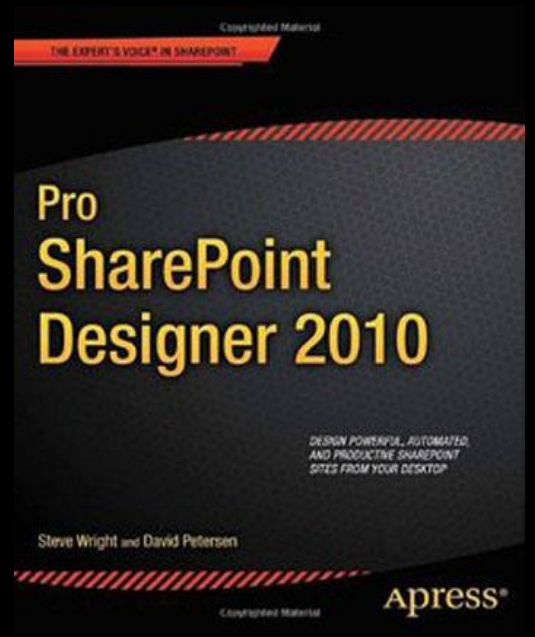
DEMO CODE

- The code I used in my demo can be viewed and copied at:

https://github.com/dipetersen/ReactOnSharePoint

DAVID PETERSEN

- Independent SharePoint Consultant
- Started developing on SharePoint in 2001 (Tahoe Server)
- Omaha SharePoint User Group Leader
- Organizer of SharePoint Saturday – Omaha
  - 5th Annual SharePoint Saturday – Omaha on April 22, 2017



david@dipetersen.com
@dipetersen
http://linkedin.com/in/dpetersen

WHAT IS REACT?

React (sometimes styled React.js or **ReactJS**) is an open-source JavaScript library providing a view for data rendered as HTML. React views are typically rendered using components that contain additional components specified as custom HTML tags.

ANOTHER DEFINITION

React is a library for building user interfaces, which comprise only one part of an app. Deciding on all the other parts — styles, routers, npm modules, ES6 code, bundling and more — and then figuring out how to use them is a drain on developers. This has become known as JavaScript fatigue. Despite this complexity, usage of React continues to grow.

REACTJS

Declarative

React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes.

Declarative views make your code more predictable and easier to debug.

Component-Based

Build encapsulated components that manage their own state, then compose them to make complex UIs.

Since component logic is written in JavaScript instead of templates, you can easily pass rich data through your app and keep state out of the DOM.

Learn Once, Write Anywhere

We don't make assumptions about the rest of your technology stack, so you can develop new features in React without rewriting existing code.

React can also render on the server using Node and power mobile apps using React Native.

REACTJS

- Created by Facebook
- Open Source
- Both Facebook and Twitter are written in ReactJS
- It is only concerned with one thing – the UI.
- While you can write everything using JavaScript, it does have JSX which makes the code easier to read.

EXAMPLE – JAVASCRIPT

```
"use strict";
var HelloMessage = React.createClass({
  displayName: "HelloMessage",
  render: function render() {
    return React.createElement(
      "div",
      null,
      "Hello ",
      this.props.name
    );
  }
});
ReactDOM.render(React.createElement(HelloMess
age, { name: "John" }), mountNode);
```

EXAMPLE - JSX

```
var HelloMessage = React.createClass({
  render: function() {
    return <div>Hello {this.props.name}</div>;
  }
});

ReactDOM.render(<HelloMessage name="John" />, mountNode);
```

Thinking in React

Edit on GitHub

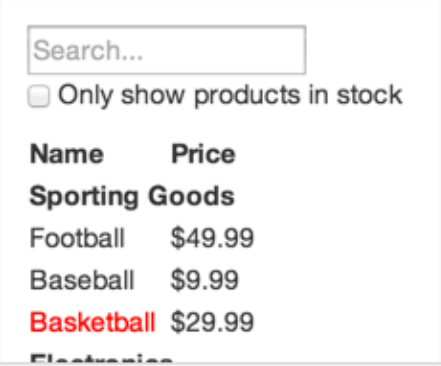
by Pete Hunt

React is, in my opinion, the premier way to build big, fast Web apps with JavaScript. It has scaled very well for us at Facebook and Instagram.

One of the many great parts of React is how it makes you think about apps as you build them. In this post, I'll walk you through the thought process of building a searchable product data table using React.

Start with a mock

Imagine that we already have a JSON API and a mock from our designer. Our designer apparently isn't very good because the mock looks like this:



TUTORIAL

https://facebook.github.io/react/docs/thinking-in-react.html

WHY DO WE CARE?

- With O365 Farm solutions don't work.
- Existing development models for SharePoint encounter constraints of the Iframe, and a reduced set of APIs and inability to integrate across Office 365 workloads.
- More and more solutions are developed using JavaScript and REST
- Or JSOM

MICROSOFT'S ANSWER

- The SharePoint Framework
- http://dev.office.com/sharepoint/docs/spfx/sharepoint-framework-overview

- The SharePoint Framework (SPFx) is a page and web part model that provides full support for client-side SharePoint development, easy integration with SharePoint data, and support for open source tooling.

KEY FEATURES

- Runs in the context of the current user and connection in the browser. There are no iFrames.
- The controls are rendered in the normal page DOM.
- The controls are responsive and accessible by nature.
- Enables the developer to access the lifecycle - including, in addition to render - load, serialize and deserialize, configuration changes, and more.
- It's framework agnostic. You can use any browser framework that you like: React, Handlebars, Knockout, Angular, and more.
- The toolchain is based on common open source client development tools like npm, TypeScript, Yeoman, webpack, and gulp.
- Performance is reliable.
- End users can use SPFx client-side solutions that are approved by the tenant administrators (or their delegates) on all sites, including self-service team, group, or personal sites.
- Solutions can be deployed in both classic web part and publishing pages and modern pages.

TOOLS AND LIBRARIES

- TypeScript
- UI Libraries/Frameworks
  - React
  - AngularJS 1.x
  - Angular 2 for TypeScript 2.x
  - Handlebars

- Node Package Manager (npm)
- Node.js
- Gulp task runner
- Webpack
- Yeoman generators
- SharePoint REST APIs

EXAMPLE BUILDING SPX WEBPART

https://blog.mastykarz.nl/sharepoint-framework-client-side-web-parts-jquery-plugins-cdn/

WHAT ELSE CAN WE DO?

We can use React to build SPA's using the SharePoint Rest Interfaces.

HOW TO DEAL WITH THE COMPLEXITY?

- Roll your own.
- or
- Use pre-built application scaffolding
  - One is build by Facebook developers
  - https://github.com/facebookincubator/create-react-app

CREATE-REACT-APP

Create React apps with no build configuration.

Getting Started – How to create a new app.

User Guide – How to develop apps bootstrapped with Create React App.

tl;dr

```
npm install -g create-react-app

create-react-app my-app
cd my-app/
npm start
```

Then open http://localhost:3000/ to see your app.

When you're ready to deploy to production, create a minified bundle with npm run build.

CREATE REST MODULE

```
function GetData(webUrl, listName, filter, selectFields, sortField)

function DeleteListItem(webUrl, listName, id)

function UpdateListItem(webUrl, listName, id, listdata)

function CreateListItem(webUrl, listName, listdata)
```

IMPORT INTO APP

```
import React, {Component} from 'react';
import ContactForm from './ContactForm';
import ContactFormApi from '../api/ContactApi';
import toastr from 'toastr';
```

DEMO

LINKS

- Call REST Services with AJAX
- https://social.technet.microsoft.com/Forums/sharepoint/en-US/c5c11462-56a8-4d2d-b770-02c6a00a2d63/sharepoint-rest-api-with-javascript?forum=sharepointgeneral

- SharePoint Hosted Add-in with ReactJS
- http://vintaurus.weebly.com/blog/sharepoint-hosted-add-in-with-reactjs

- ReactJS Tutorial on calling AJAX
- http://facebook.github.io/react/docs/tutorial.html
-
- From Facebook.
- https://github.com/facebookincubator/create-react-app

- D3 and React
- https://www.codementor.io/reactjs/tutorial/3-steps-scalable-data-visualization-react-js-d3-js