Building Applications for Office 365 and SharePoint with the SharePoint Framework

Scot Hillier
scot@scothillier.net
@ScotHillier

SharePoint *intersection*   Office 365 *intersection*

Agenda

- My first SPFX web part
- ECMAScript and TypeScript
  - Overview
  - Modules
- React framework
  - Fundamentals
  - Lifecycle and async operations
- Developing with SPFX
  - Web parts and extensions
  - Deployment

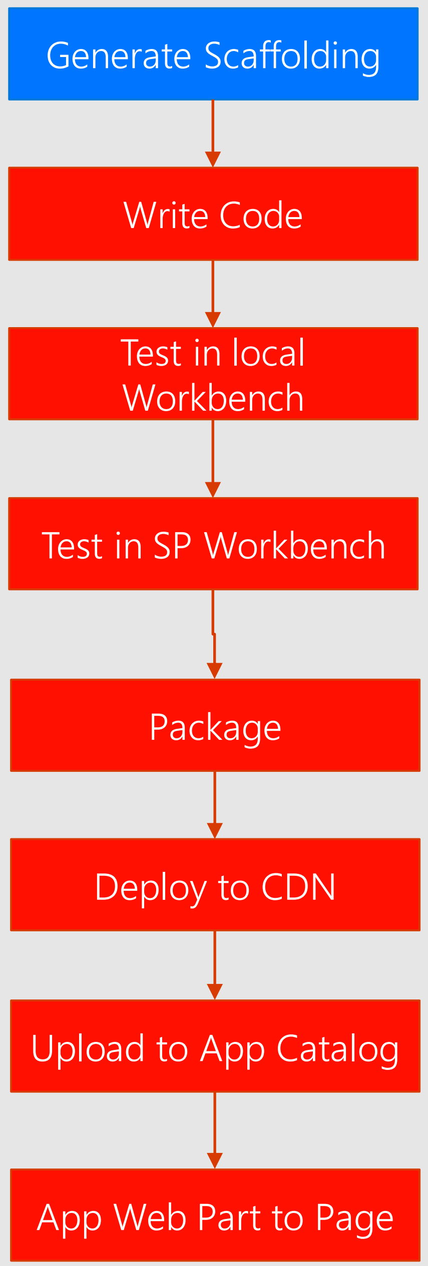
My first SPFX web part

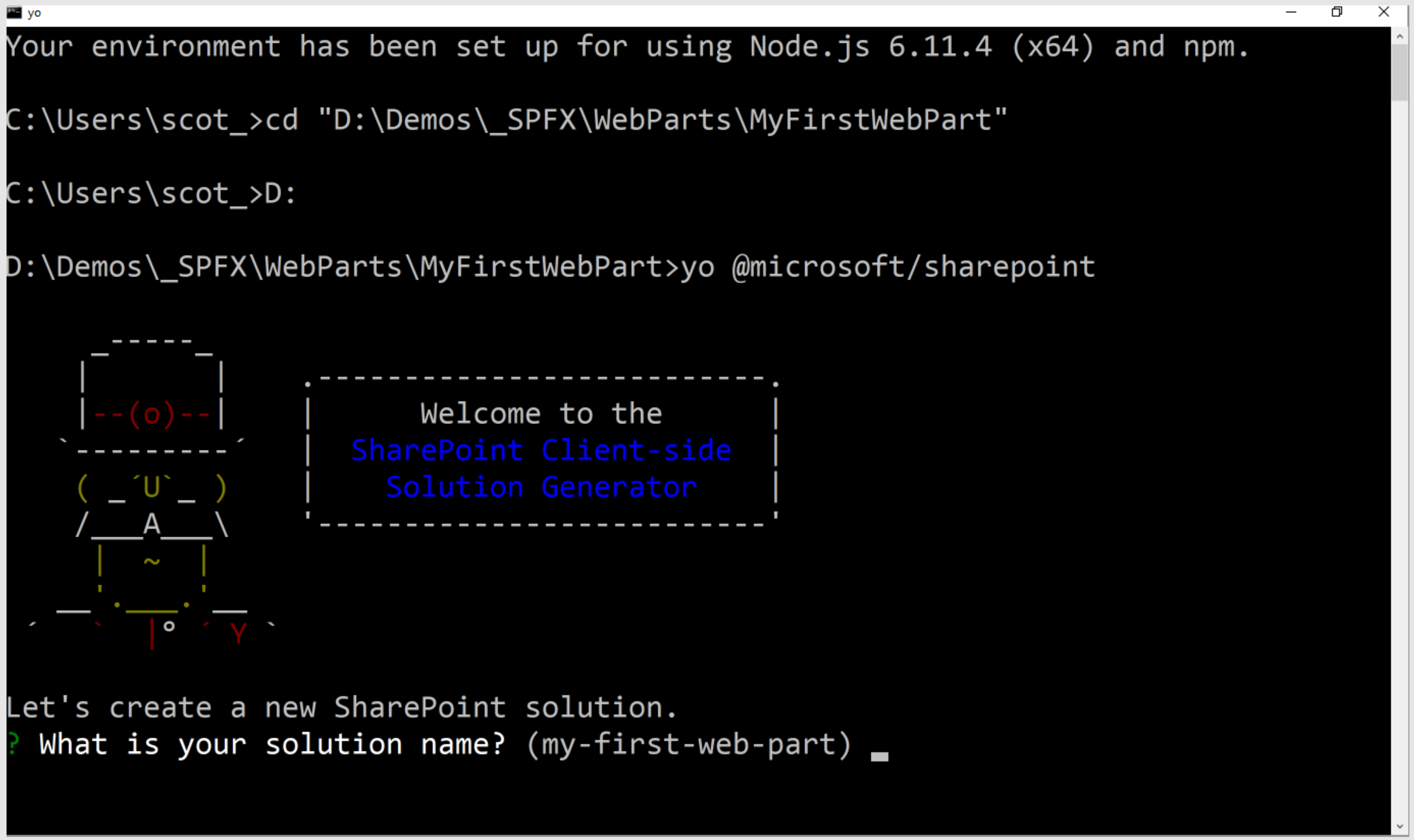
Set up your SharePoint environment

- Cloud or on-premises
  - Office 365 tenant
  - SharePoint 2016, Feature Pack 2
- App catalog
- Developer site collection
- SharePoint workbench
  - Local
  - Tenant
  - On-premises

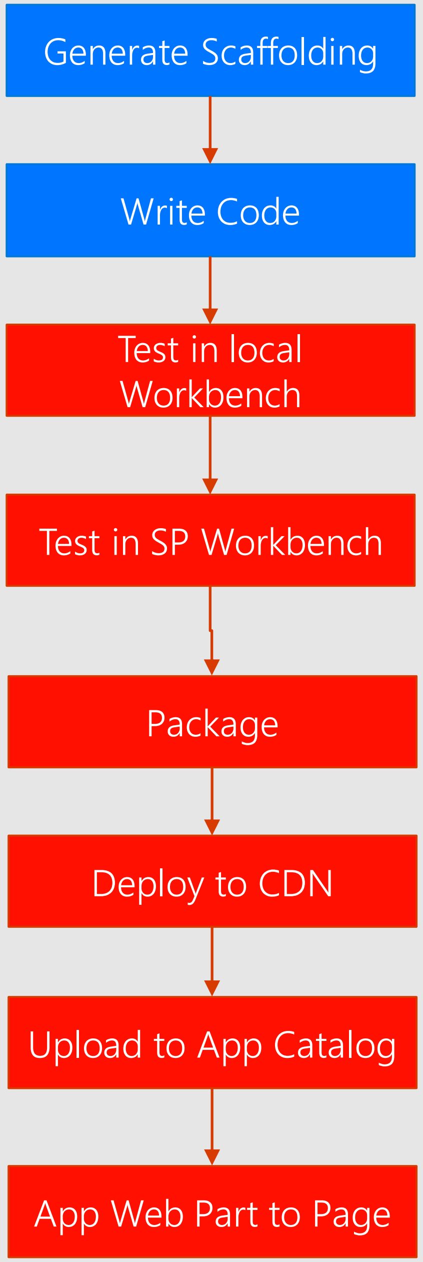
Set up your development environment

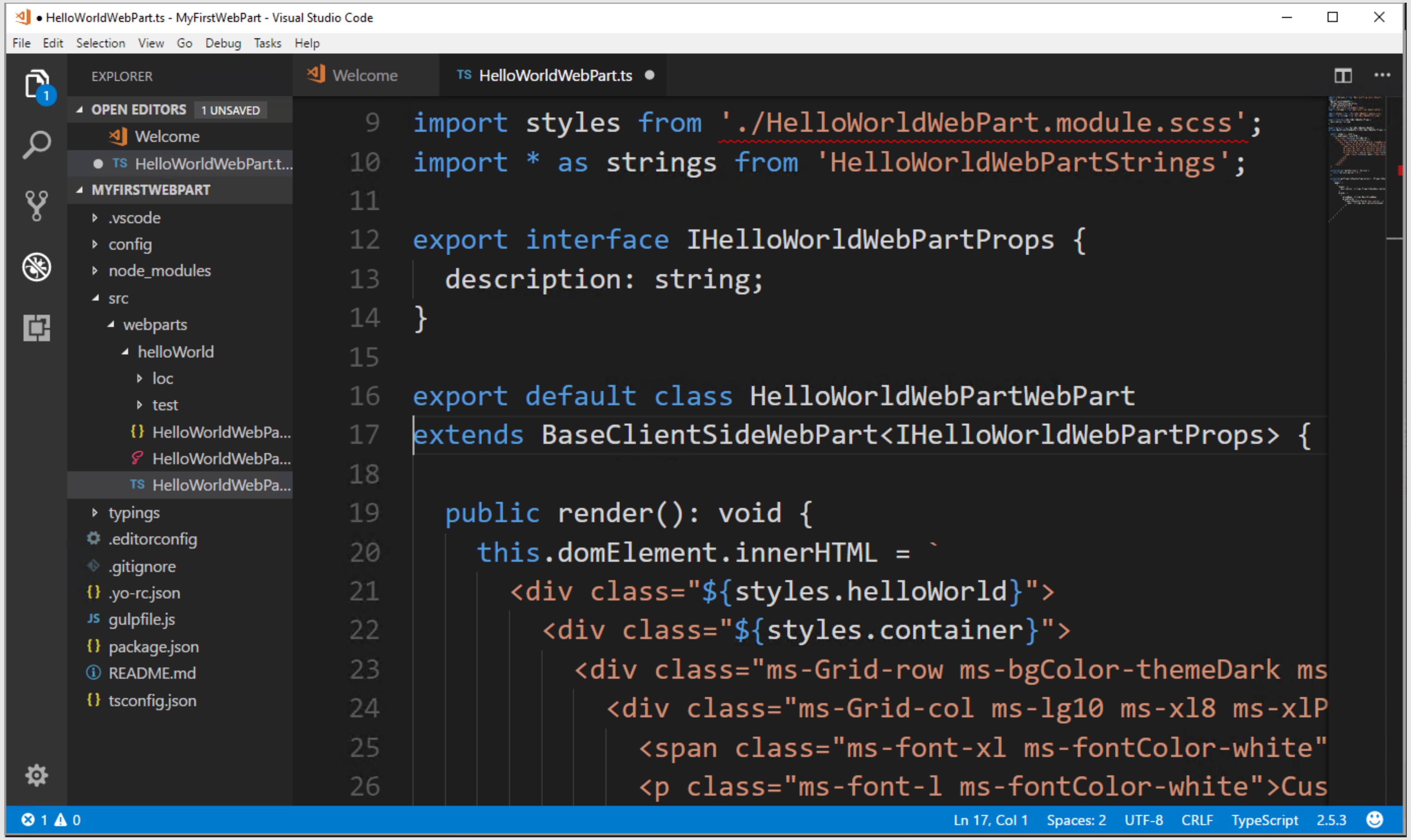
- NodeJS LTS v6.11.4 (https://nodejs.org/en)
  - Node package manager (npm) v3.10.10 automatically installed
  - NodeJS command prompt used to interact with npm
- Visual Studio Code
- Gulp
  - `npm install -g gulp`
- Yeoman
  - `npm install -g yo`
- SharePoint Generator
  - `npm install -g @Microsoft/generator-sharepoint`



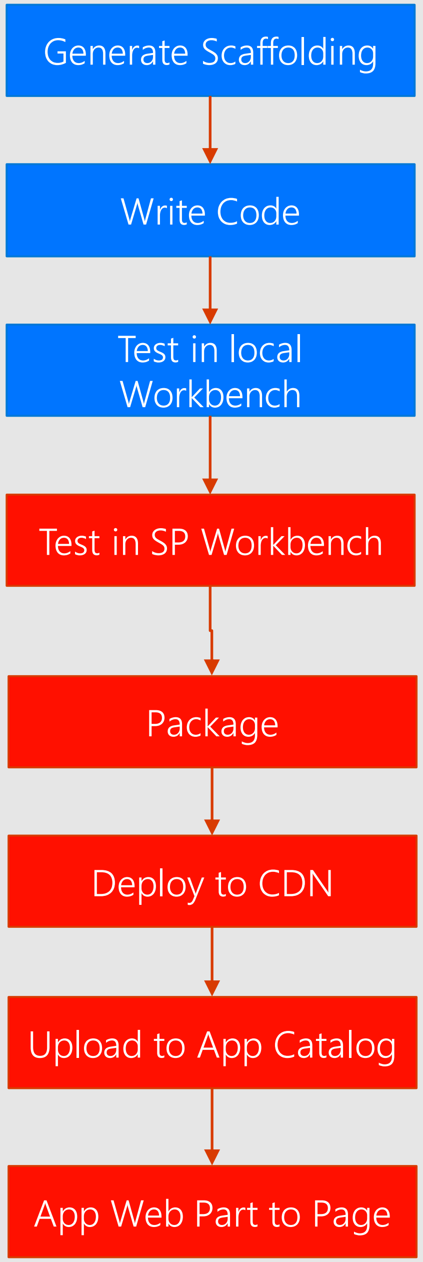


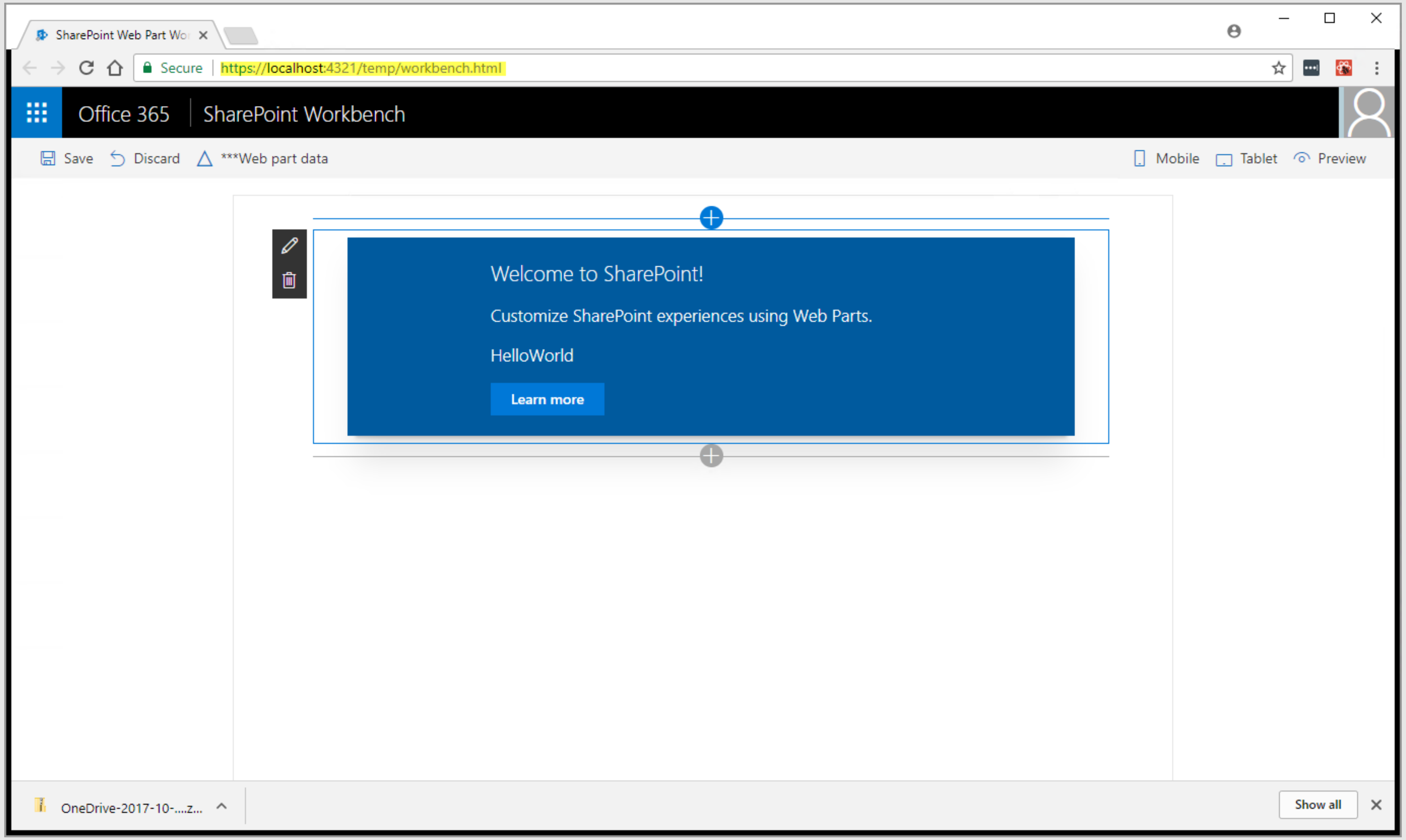
```
Your environment has been set up for using Node.js 6.11.4 (x64) and npm.

C:\Users\scot_>cd "D:\Demos\_SPFX\WebParts\MyFirstWebPart"

C:\Users\scot_>D:

D:\Demos\_SPFX\WebParts\MyFirstWebPart>yo @microsoft/sharepoint

     _-----_
    |       |
    |--(o)--|     .--------------------------.
   `---------´    |      Welcome to the      |
    ( _´U`_ )     |   SharePoint Client-side  |
    /___A___\     |     Solution Generator    |
     |  ~  |      '--------------------------'
   __'.___.'__
 ´   `  |° ´ Y `

Let's create a new SharePoint solution.
? What is your solution name? (my-first-web-part)
```





```
import styles from './HelloWorldWebPart.module.scss';
import * as strings from 'HelloWorldWebPartStrings';

export interface IHelloWorldWebPartProps {
  description: string;
}

export default class HelloWorldWebPartWebPart
extends BaseClientSideWebPart<IHelloWorldWebPartProps> {

  public render(): void {
    this.domElement.innerHTML = `
      <div class="${styles.helloWorld}">
        <div class="${styles.container}">
          <div class="ms-Grid-row ms-bgColor-themeDark ms
            <div class="ms-Grid-col ms-lg10 ms-xl8 ms-xlP
              <span class="ms-font-xl ms-fontColor-white"
              <p class="ms-font-l ms-fontColor-white">Cus
```

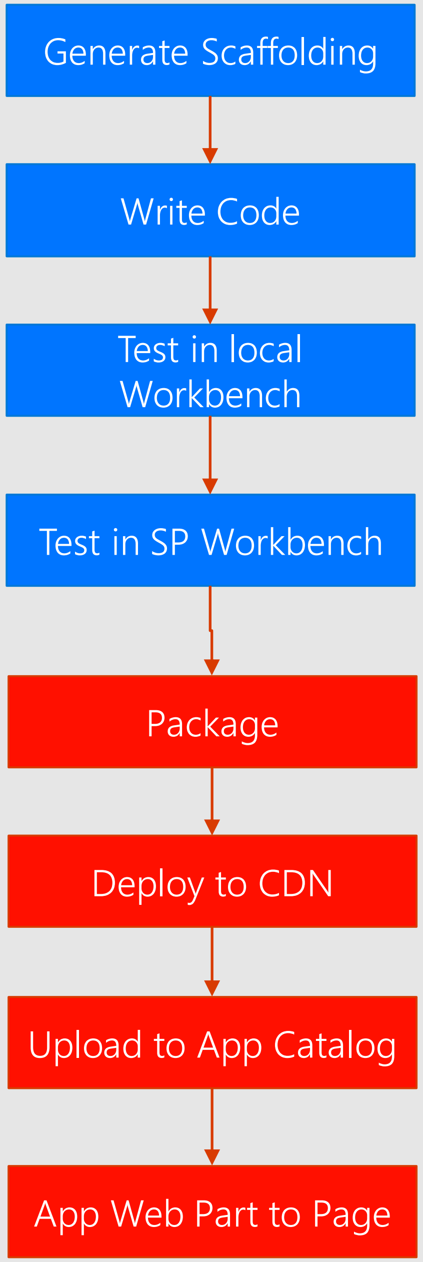


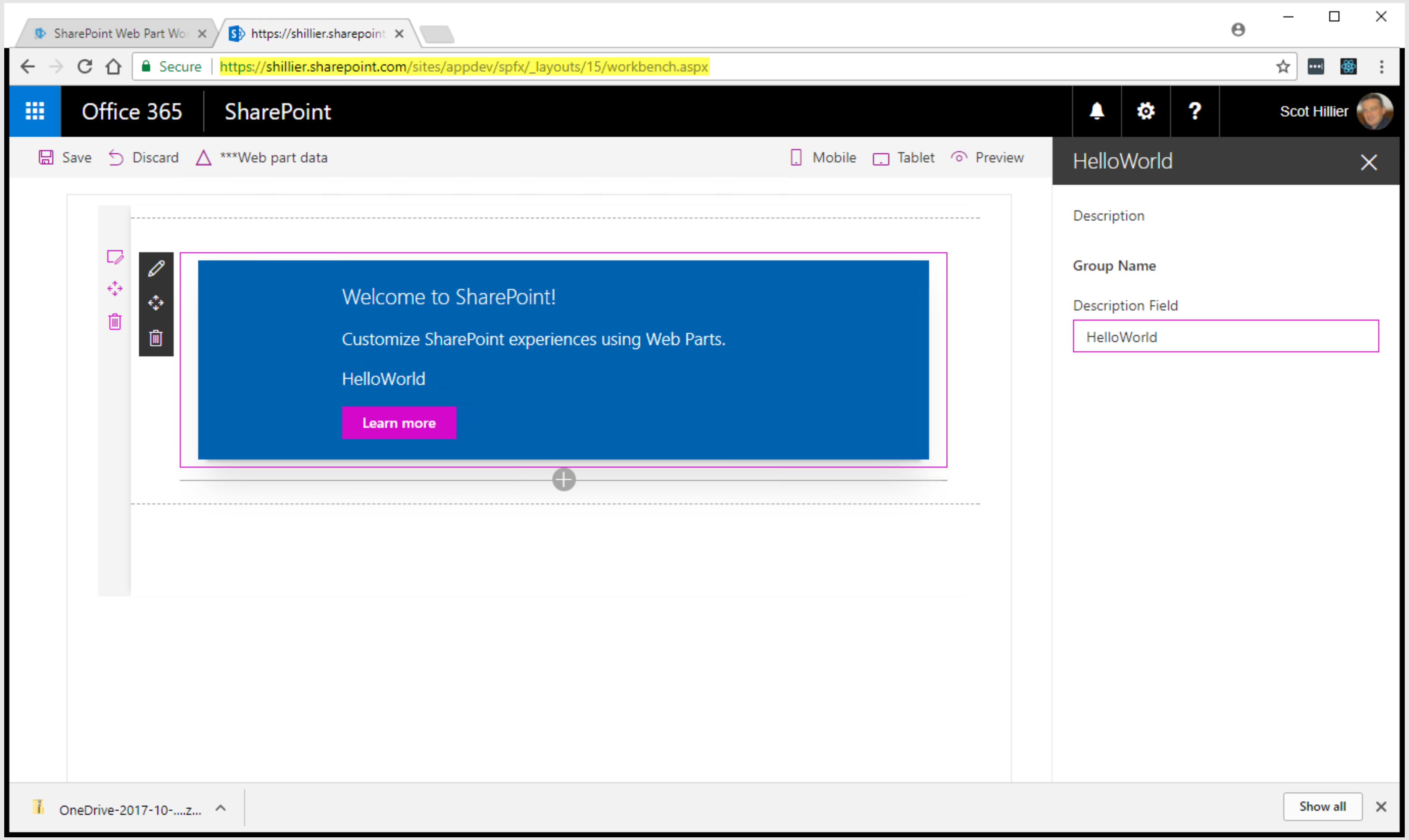


Welcome to SharePoint!

Customize SharePoint experiences using Web Parts.

HelloWorld





https://shillier.sharepoint.com/sites/appdev/spfx/_layouts/15/workbench.aspx

***Web part data

Welcome to SharePoint!

Customize SharePoint experiences using Web Parts.

HelloWorld

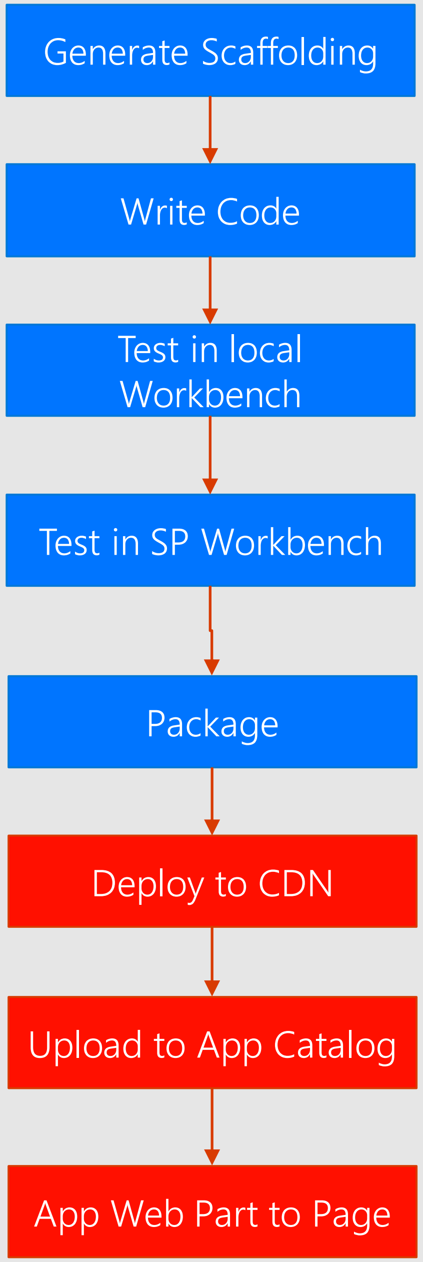
HelloWorld

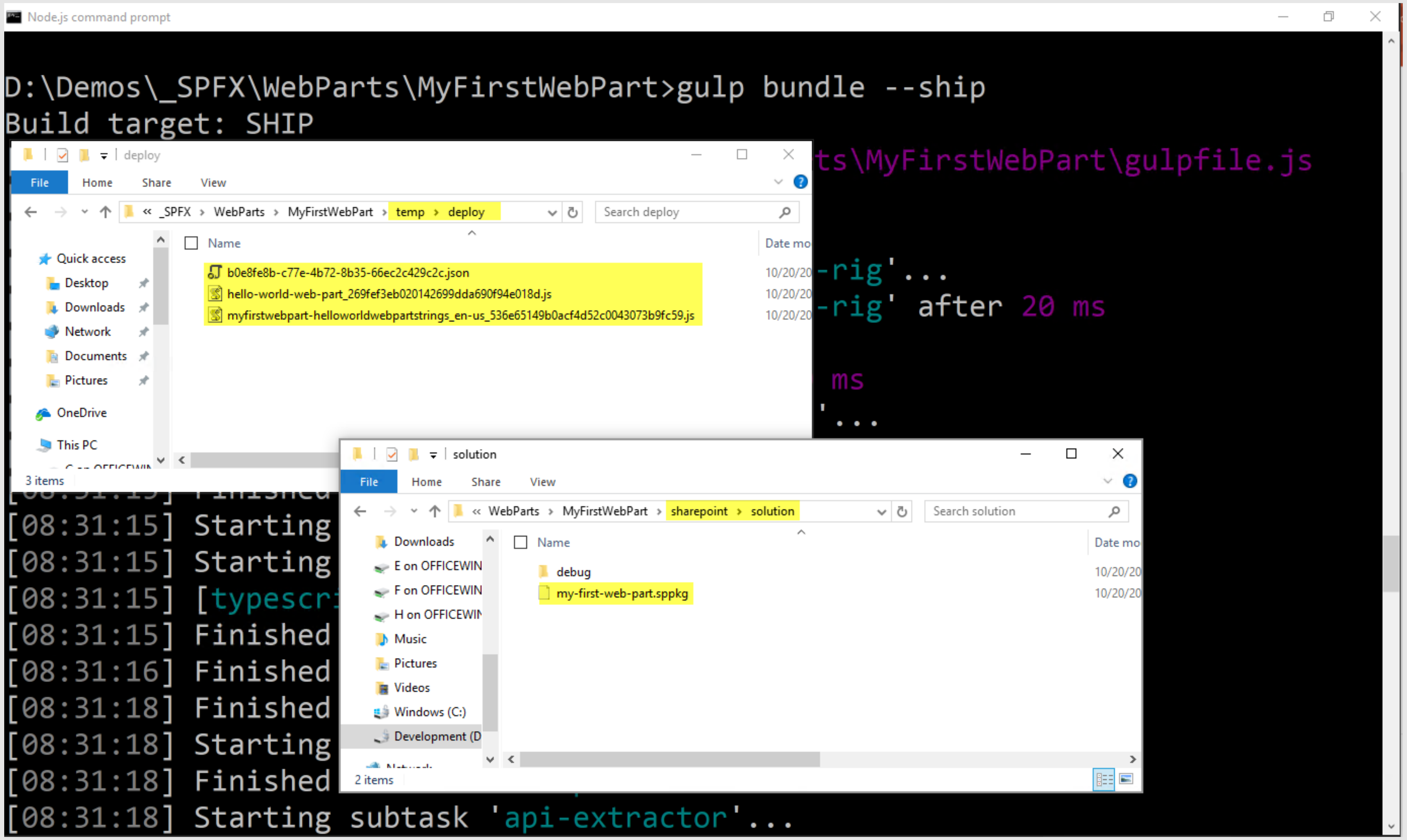
Description

Group Name

Description Field: HelloWorld

OneDrive-2017-10-....z...





```
D:\Demos\_SPFX\WebParts\MyFirstWebPart>gulp bundle --ship
Build target: SHIP
ts\MyFirstWebPart\gulpfile.js

-rig'...
-rig' after 20 ms

ms
'...
[08:31:15] Finished
[08:31:15] Starting
[08:31:15] Starting
[08:31:15] [typescr
[08:31:15] Finished
[08:31:16] Finished
[08:31:18] Finished
[08:31:18] Starting
[08:31:18] Finished
[08:31:18] Starting subtask 'api-extractor'...
```

« _SPFX › WebParts › MyFirstWebPart › temp › deploy

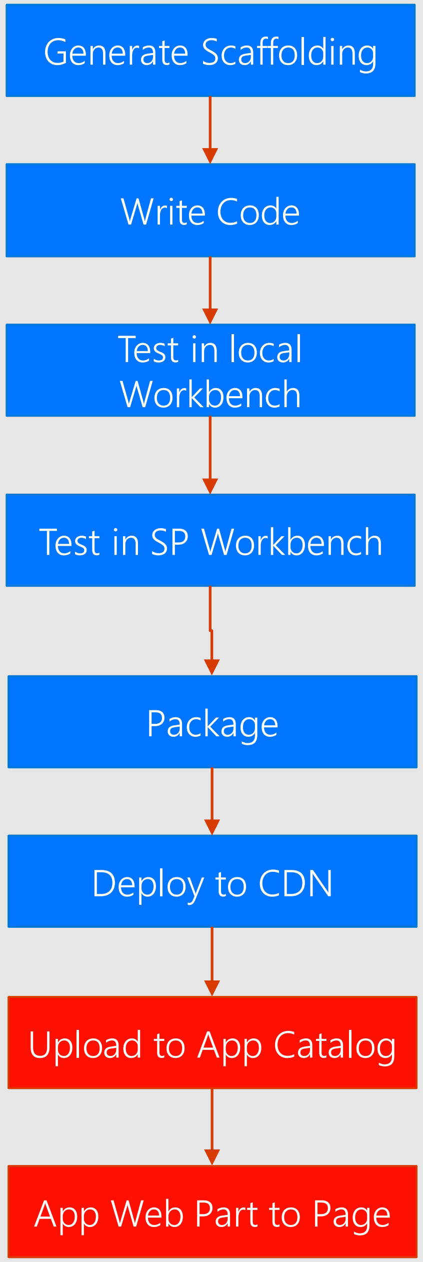
Name	Date mo
b0e8fe8b-c77e-4b72-8b35-66ec2c429c2c.json	10/20/20
hello-world-web-part_269fef3eb020142699dda690f94e018d.js	10/20/20
myfirstwebpart-helloworldwebpartstrings_en-us_536e65149b0acf4d52c0043073b9fc59.js	10/20/20

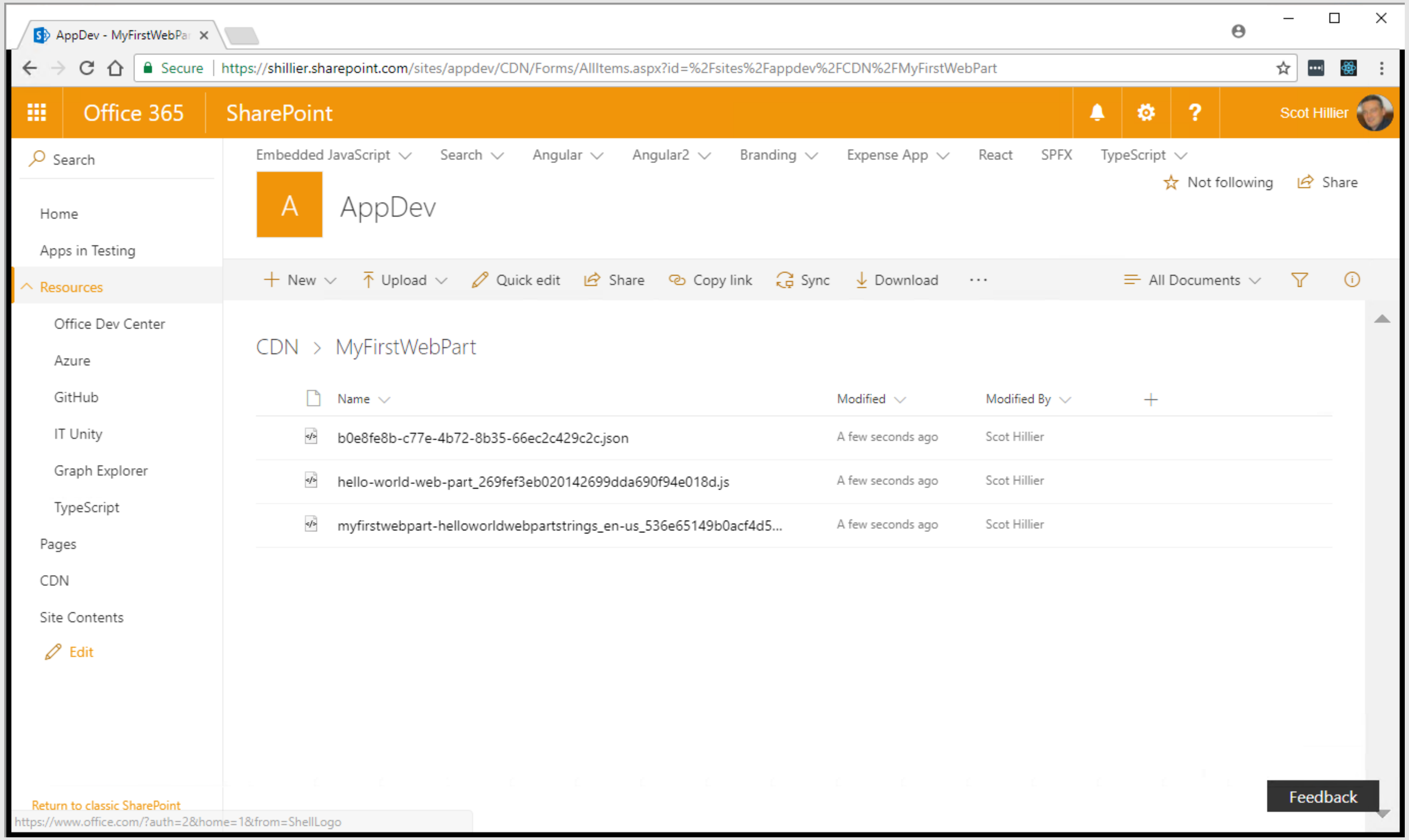
3 items

« WebParts › MyFirstWebPart › sharepoint › solution

Name	Date mo
debug	10/20/20
my-first-web-part.sppkg	10/20/20

2 items



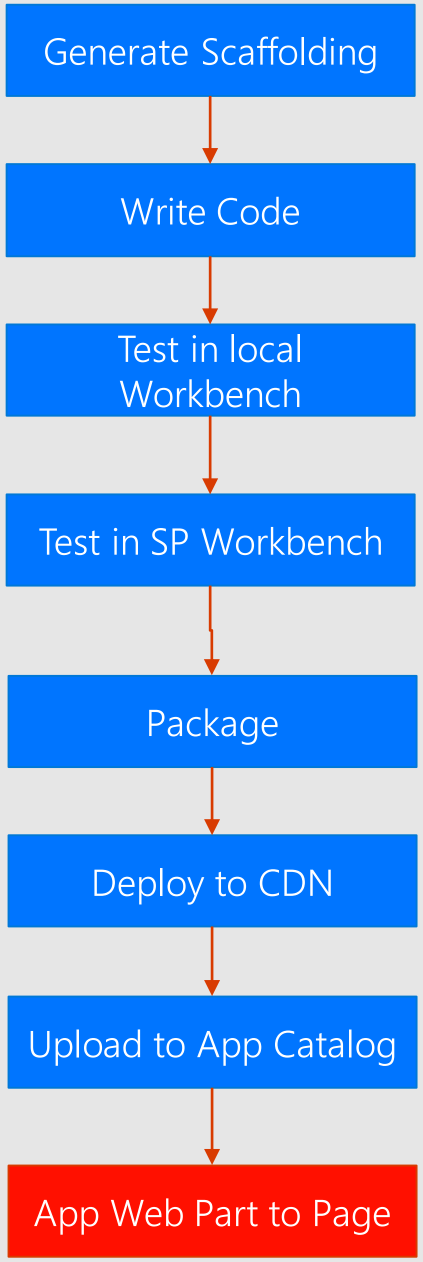


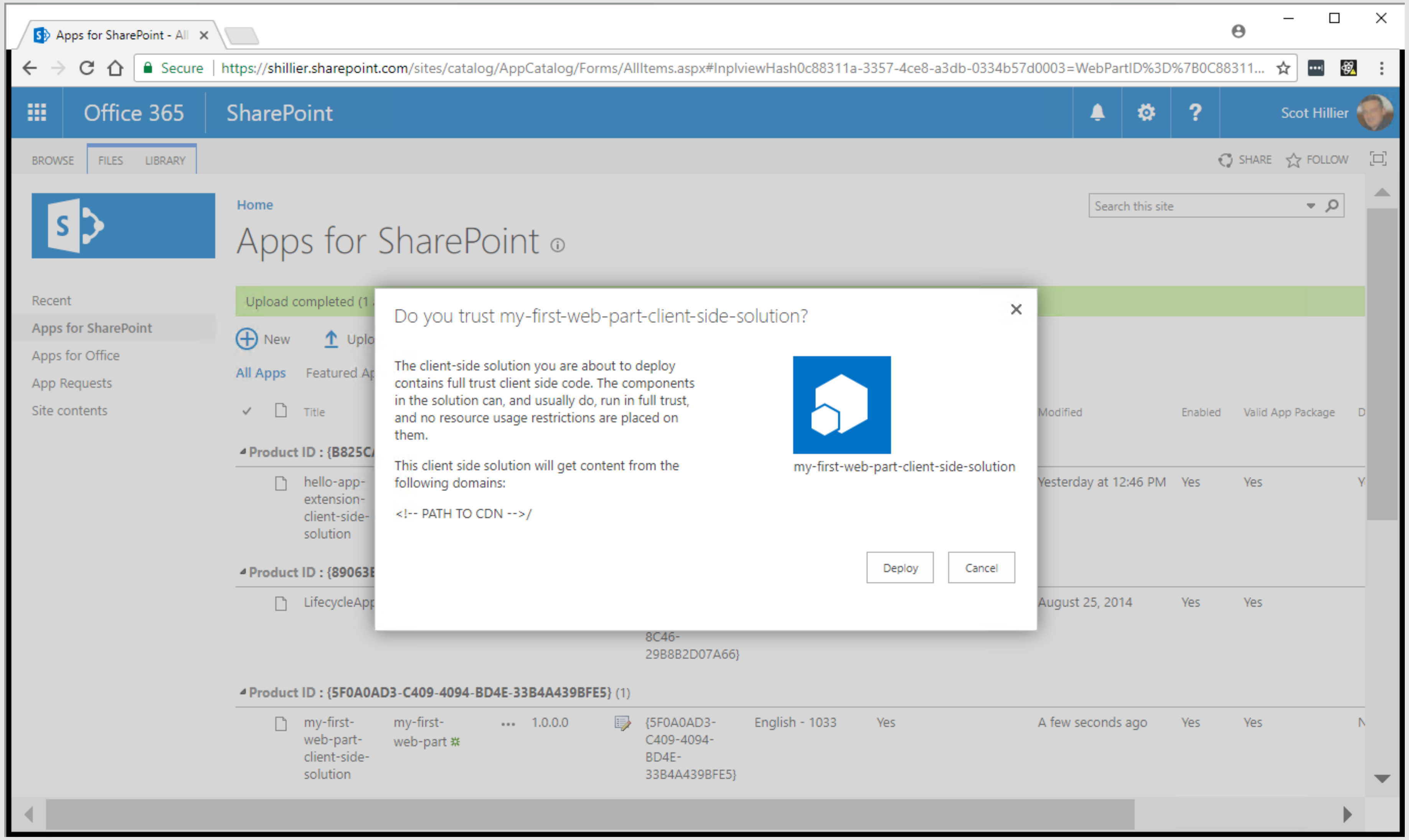
https://shillier.sharepoint.com/sites/appdev/CDN/Forms/AllItems.aspx?id=%2Fsites%2Fappdev%2FCDN%2FMyFirstWebPart

CDN > MyFirstWebPart

Name	Modified	Modified By
b0e8fe8b-c77e-4b72-8b35-66ec2c429c2c.json	A few seconds ago	Scot Hillier
hello-world-web-part_269fef3eb020142699dda690f94e018d.js	A few seconds ago	Scot Hillier
myfirstwebpart-helloworldwebpartstrings_en-us_536e65149b0acf4d5...	A few seconds ago	Scot Hillier

https://www.office.com/?auth=2&home=1&from=ShellLogo





https://shillier.sharepoint.com/sites/catalog/AppCatalog/Forms/AllItems.aspx#InplviewHash0c88311a-3357-4ce8-a3db-0334b57d0003=WebPartID%3D%7B0C88311...

Home

Apps for SharePoint

Upload completed (1

Do you trust my-first-web-part-client-side-solution?

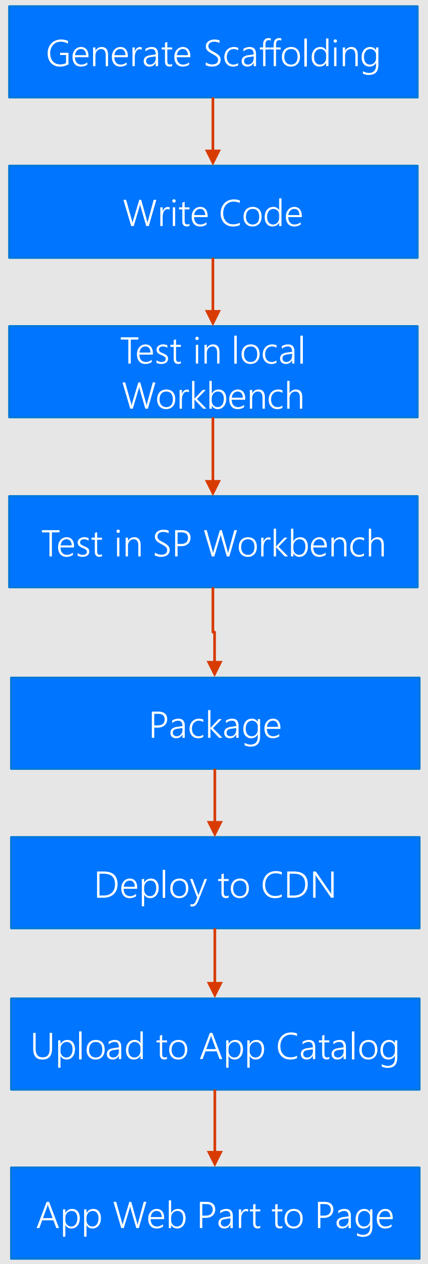
The client-side solution you are about to deploy contains full trust client side code. The components in the solution can, and usually do, run in full trust, and no resource usage restrictions are placed on them.

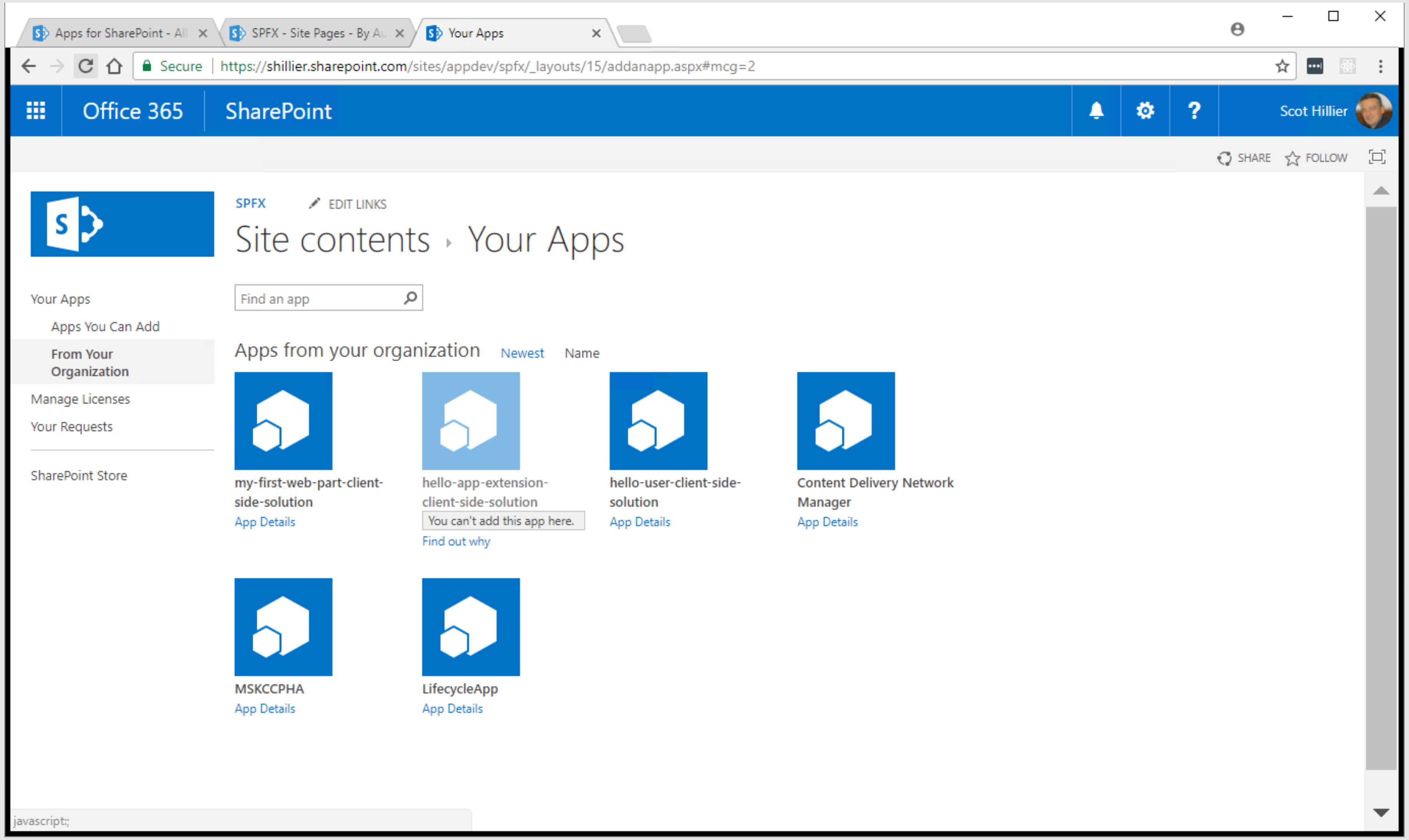
This client side solution will get content from the following domains:

<!-- PATH TO CDN -->/

my-first-web-part-client-side-solution

✓	Title							Modified	Enabled	Valid App Package	D
Product ID : {B825C											
	hello-app-extension-client-side-solution							Yesterday at 12:46 PM	Yes	Yes	Y
Product ID : {89063E											
	LifecycleApp				8C46-29B8B2D07A66}			August 25, 2014	Yes	Yes	
Product ID : {5F0A0AD3-C409-4094-BD4E-33B4A439BFE5} (1)											
	my-first-web-part-client-side-solution	my-first-web-part	...	1.0.0.0	{5F0A0AD3-C409-4094-BD4E-33B4A439BFE5}	English - 1033	Yes	A few seconds ago	Yes	Yes	N


Generate Scaffolding
Write Code
Test in local Workbench
Test in SP Workbench
Package
Deploy to CDN
Upload to App Catalog
App Web Part to Page


Apps for SharePoint - All
SPFX - Site Pages - By Au
Your Apps
Secure
https://shillier.sharepoint.com/sites/appdev/spfx/_layouts/15/addanapp.aspx#mcg=2
Office 365
SharePoint
Scot Hillier
SHARE
FOLLOW
SPFX
EDIT LINKS
Site contents › Your Apps
Your Apps
Apps You Can Add
From Your Organization
Manage Licenses
Your Requests
SharePoint Store
Find an app
Apps from your organization
Newest
Name
my-first-web-part-client-side-solution
App Details
hello-app-extension-client-side-solution
You can't add this app here.
Find out why
hello-user-client-side-solution
App Details
Content Delivery Network Manager
App Details
MSKCCPHA
App Details
LifecycleApp
App Details
javascript:;

Demo

Key resources

ECMAScript and TypeScript

JavaScript keeps getting better and better...

- ECMAScript 2015 (ES6)

classes	modules	
default parameters	string interpolation	multi-line strings
block scoped let	promises	arrow functions

- ECMAScript 2016 (ES7)

· array includes	exponent operator **

- ECMAScript 2017 (ES8)

· array key,value,entry	string padding	trailing commas
· Property descriptors	async functions	

ECMAScript 2015 Classes

```
class person {

    constructor(firstName, lastName) {
        this.firstName = firstName;
        this.lastName = lastName;
    }

    speak() {
        return `My name is ${this.firstName} ${this.lastName}.`;
    }
}

var p = new person('Scot', 'Hillier');
alert(p.speak());
```

ECMAScript 2015 Inheritance

```
class customer extends person {
    constructor(firstName, lastName, title, company, email) {
        super(firstName, lastName);
        this.title = title;
        this.company = company;
        this.email = email;
    }
    talk() {
        return `${super.speak()} I work for ${this.company}.`;
    }
}

var c = new customer('Scot', 'Hillier', 'MVP',
        'Scot Hillier Technical Solutions', 'scot@scothillier.net');
alert(c.talk());
```

ECMAScript 2017 async/await

```
function findOpenTable(partySize) {
    return new Promise(resolve => {
        setTimeout(() => {
            let tableId = 5;
            resolve(tableId);
        }, 5000);
    });
}

async function makeReservations() {
    let tableId1 = await findOpenTable(2);
    let tableId2 = await findOpenTable(2);
});
```

Unfortunately, JavaScript is vulnerable to typos

```
(function () {
    function loadMe() {
        var greetings = ["Happy", "Have a good", "It's"];
        var weekdays = ["Sunday", "Monday", "Tuesday", "Wednesday",
                        "Thursday", "Friday", "Saturday"];
        var greeting =
            greetings[Math.floor(Math.random * greetings.length)];
        var todayIs = weekdays[Date().getDay()];
        document.getElementByID('elt1').innerHTML =
        greeting + " " + todayIs + "!";
    }
    window.onload = loadMe();
})()
```

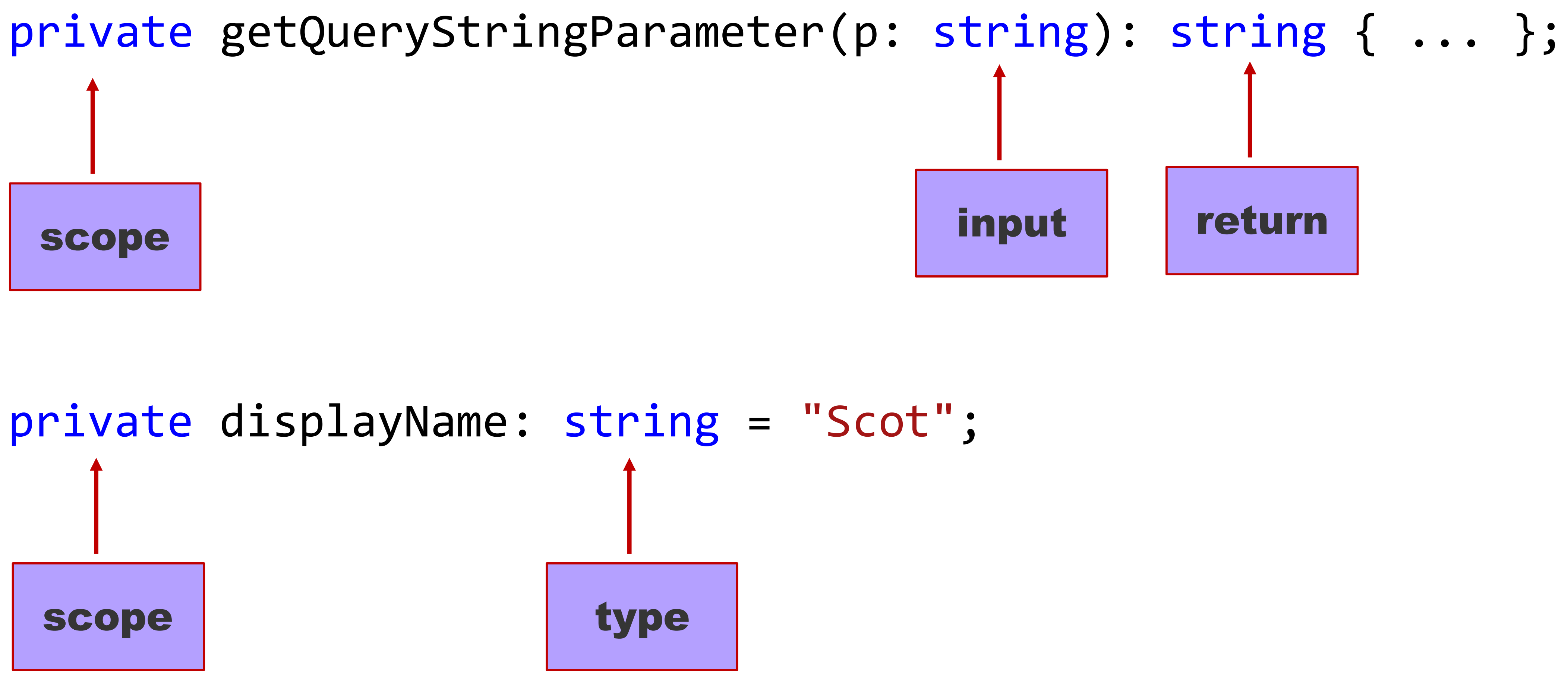
How many did you find?

```
(function () {
    function loadMe() {
        var greetings = ["Happy", "Have a good", "It's"];
        var weekdays = ["Sunday", "Monday", "Tuesday", "Wednesday",
                        "Thursday", "Friday", "Saturday"];
        var greeting =
            greetings[Math.floor(Math.random() * greetings.length)];
        var todayIs = weekdays[new Date().getDay()];
        document.getElementByI~~D~~d('elt1').innerHTML =
        greeting + " " + todayIs + "!";
    }
    window.onload = loadMe~~()~~;
})()
```

Introducing TypeScript

- Typed superset that transpiles to plain JavaScript
  - You write ts files, it produces js files
  - Produces cross-browser-compatible code
  - Support for the latest ECMAScript features now
- Fully integrated into Visual Studio 2017/Code
  - Type Annotations
  - Interfaces
  - Compilation, Intellisense, and error checking

Type annotations



```
private getQueryStringParameter(p: string): string { ... };
```

```
private displayName: string = "Scot";
```

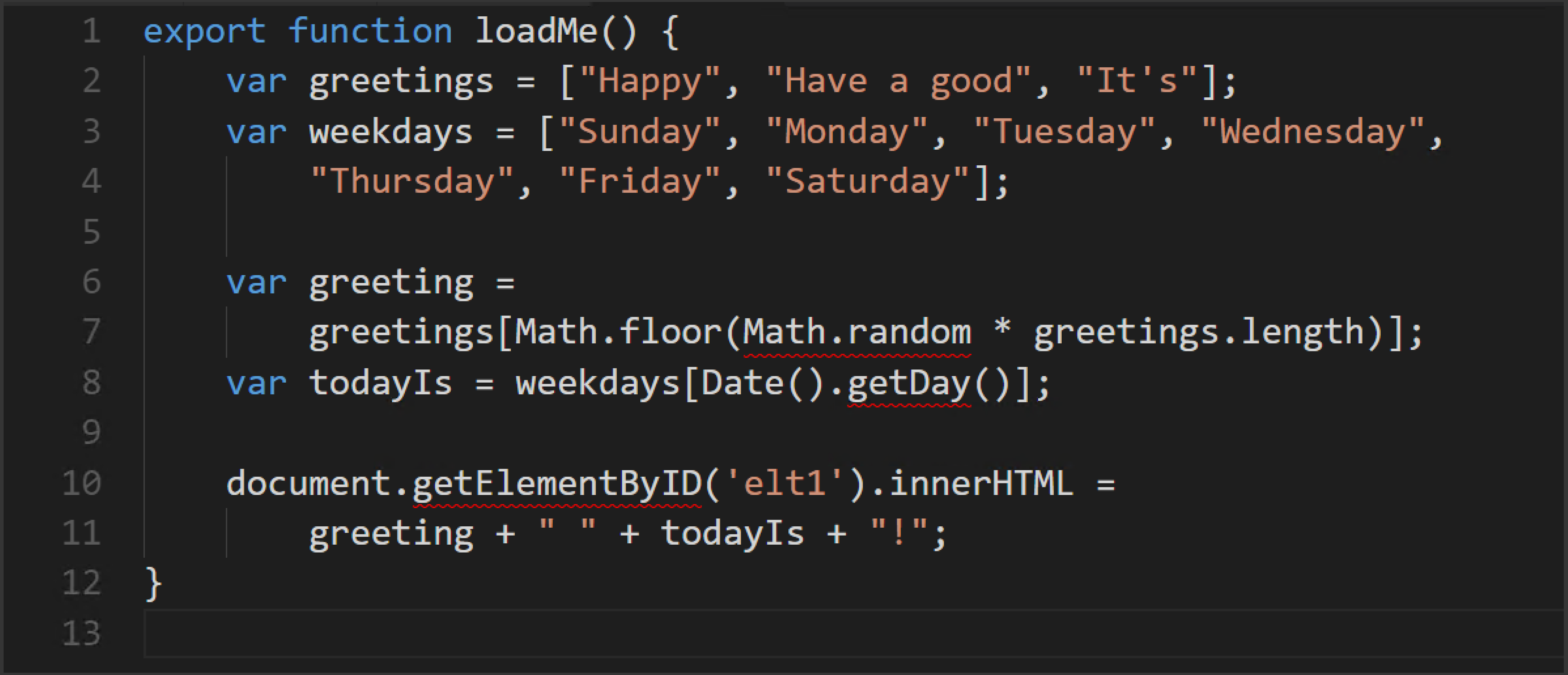
Interfaces

```
interface WelcomeData {
   pictureUrl: string;
   displayName: string;
 }


class Welcome {
   public get_viewModel(): WelcomeData {
       return {
           "pictureUrl": this.pictureUrl,
           "displayName": this.displayName
       };
   }
}
```

Define Interface

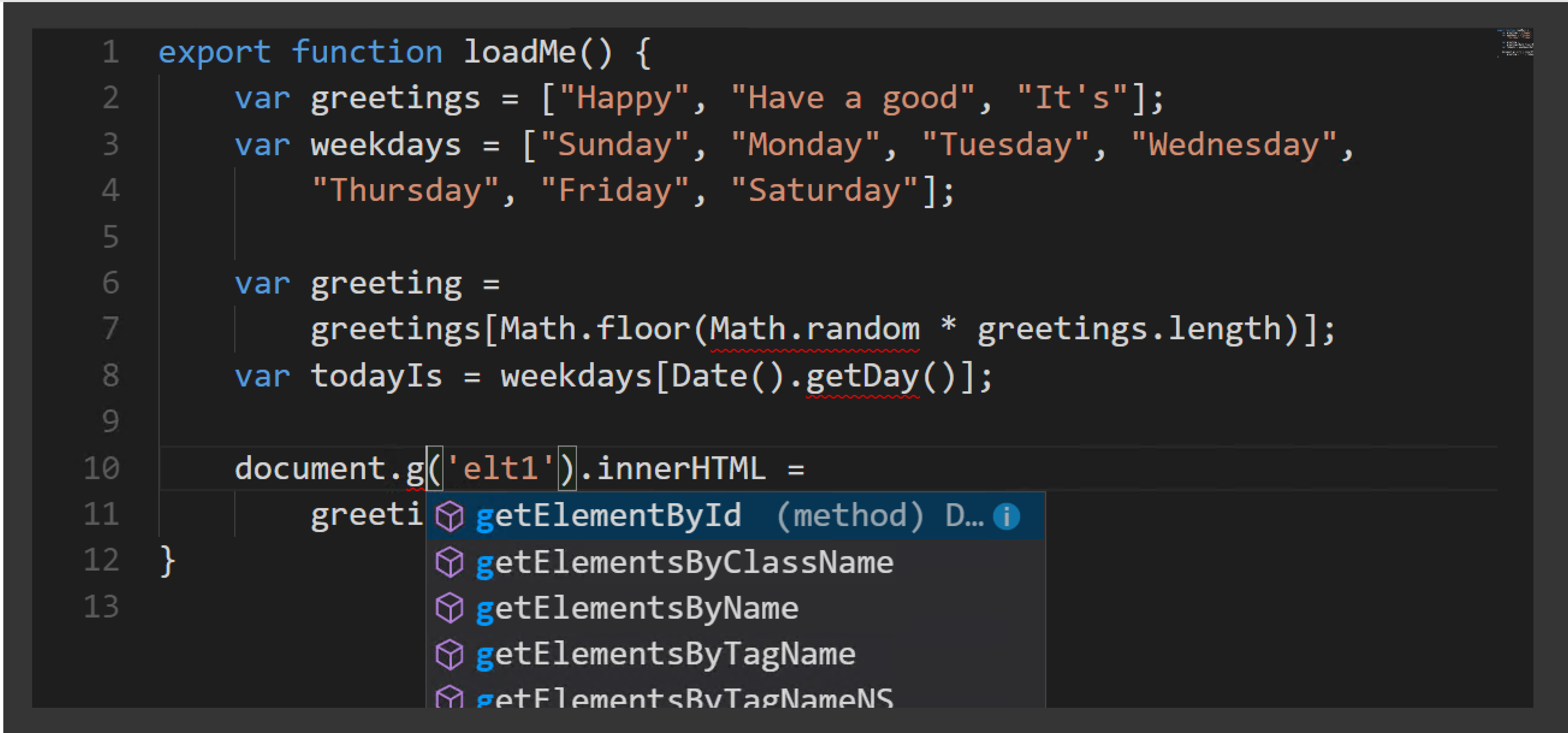
Implement Interface

Error checking



```
export function loadMe() {
    var greetings = ["Happy", "Have a good", "It's"];
    var weekdays = ["Sunday", "Monday", "Tuesday", "Wednesday",
        "Thursday", "Friday", "Saturday"];

    var greeting =
        greetings[Math.floor(Math.random * greetings.length)];
    var todayIs = weekdays[Date().getDay()];

    document.getElementByID('elt1').innerHTML =
        greeting + " " + todayIs + "!";
}

```

Intellisense



```
export function loadMe() {
    var greetings = ["Happy", "Have a good", "It's"];
    var weekdays = ["Sunday", "Monday", "Tuesday", "Wednesday",
        "Thursday", "Friday", "Saturday"];

    var greeting =
        greetings[Math.floor(Math.random * greetings.length)];
    var todayIs = weekdays[Date().getDay()];

    document.g('elt1').innerHTML =
        greeti
}

```

Type definitions

- Supports the use of external libraries
  - Takes the form of a ***.d.ts file**
  - Supports use of other Node packages with TypeScript
- Provides intellisense in TypeScript environment
- Download from npm through **@types** packages
  - `npm install @types/sharepoint -dev --save`
  - Can also simply write your own

Compilation

```
{
    "compilerOptions": {
    "target": "es5",
    "module": "system",
    "moduleResolution": "node",
    "sourceMap": true,
    "emitDecoratorMetadata": true,
    "experimentalDecorators": true,
    "removeComments": false,
    "noImplicitAny": false
},
    "exclude": [
        "node_modules"
    ]
}
```

Demo

ECMAScript and TypeScript

Modules

Understanding modules

- ECMAScript modules
  - Supports export/import without polluting global namespace
  - Development experience similar to `using` in .NET
- Module loading standards
  - No true standards – 3rd-party loaders still required
  - **CommonJS** format used with node.js server
  - **AMD** format with require.js in browser
  - **UMD** format is compatible with AMD, CommonJS and no loader at all

AMD modules in widespread use today

```
//index.html
<script src="/Scripts/require.js" data-main="/Scripts/app"></script>

//app.js
define(["require", "exports", "customer"], function (customer) {
    alert(customer.speak());
});

//customer.js
define(["require", "exports"], function (require, exports) {
    var Customer = (function () {
        function Customer(fn, ln) {
            this.firstName = fn;
            this.lastName = ln;
        }
        Customer.prototype.speak = function () {
            return "My name is " + this.firstName + " " + this.lastName;
        };
        return Customer;
    })();
    exports.Customer = Customer;
});
```

ECMAScript declarative module loading

```
//requires a transpiler or polyfill today

//person.js
export class Person {

    constructor(firstName, lastName) {
        this.firstName = firstName;
        this.lastName = lastName;
    }
}

//app.js
import {Person} from 'person';
let p = new Person();
```

ECMAScript dynamic module loading

```
//requires a transpiler or polyfill today

//person.js
export class Person {
    constructor(firstName, lastName) {
        this.firstName = firstName;
        this.lastName = lastName;
    }
}

//app.js
System.import('person').then(Person => {
    let p = new Person();
}).catch(error => { alert(error); });
```

System JS dynamic module loading

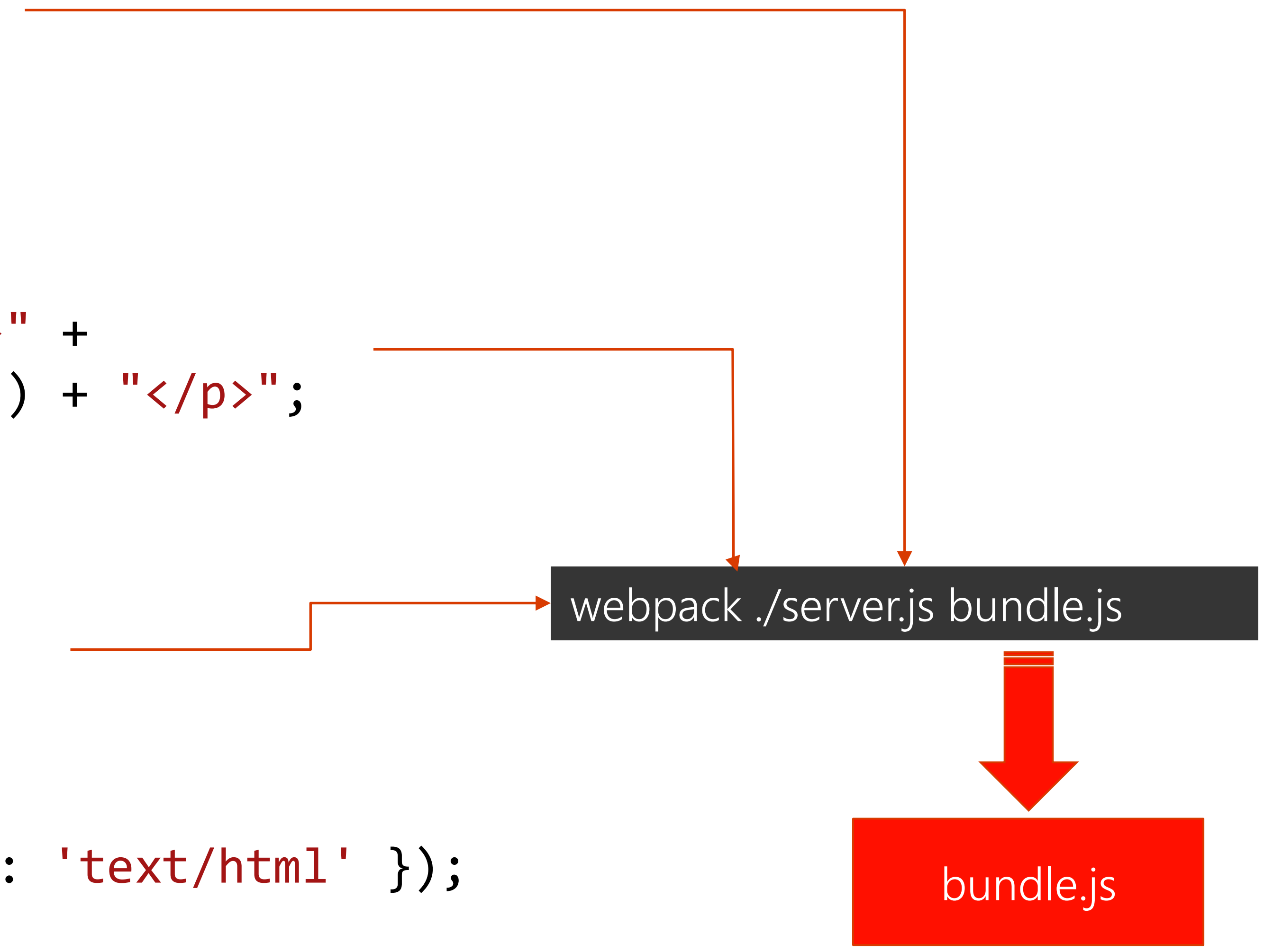
```
<script src="system.js"></script>

<!-- loads any module format, invokes transpiler if necessary -->

<script>
        System.config({
            packages: {
                'app': {
                    defaultExtension: 'js'
                }
            }
        });
        System.import('app/myModule’)
            .then(null, console.error.bind(console));
  </script>
```

WebPack module bundling

```
//message.js
module.exports = "Hello, webpack!";
```

```
//content.js
module.exports = "<p style='color:blue'>" +
                 require("./message.js") + "</p>";
```

```
//server.js
var http = require('http');
var port = process.env.port || 1337;

http.createServer(function (req, res) {
    res.writeHead(200, { 'Content-Type': 'text/html' });
    res.end(require('./content.js'));
}).listen(port);
```



Demo

Modules

React

Introducing React

- React is a framework for building user interfaces
- Emphasizes component-based development
- Lighter than other frameworks
- Ideal for SPFX

React Fundamentals

- Obtain the framework from a CDN or npm
  - https://cdnjs.cloudflare.com/ajax/libs/react/15.5.4/react.min.js
  - https://cdnjs.cloudflare.com/ajax/libs/react/15.5.4/react-dom.min.js
  - npm install react –save
  - npm install react-dom --save
- **`React`** object is the main entry point to APIs
- ReactDOM object is used to render visual elements
- React.DOM object wraps standard HTML elements

Hello, World

```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>React JavaScript Basics</title>
</head>
<body>

    <div id="app"></div>

    <!-- React Libraries -->
    <script src="https://cdnjs.cloudflare.com/ajax/libs/react/15.5.4/react.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/react/15.5.4/react-dom.min.js"></script>

    <script>
        ReactDOM.render(React.DOM.h1(null, "Hello, React!"), document.getElementById("app"));
    </script>

</body>
</html>
```

React.DOM contains HTML components

ReactDOM allows rendering of HTML

React components

- A custom class extending **`React.Component`**
- **`Render`** method returns a React component
- Immutable **`props`** for component configuration
- Changeable **`state`** used to render component

ECMAScript component

```
class Component extends React.Component {

    constructor(props) {
        super(props);
        this.state = { text: props.message };
        this.updateTextState = this.updateTextState.bind(this);
    }

    render() { return React.DOM.h1(
                    { onClick: this.updateTextState },
                    this.state.text); }

    updateTextState(newText) { this.setState({ text: "Thank you!" }); }

}
```

Event handling

```
constructor(props: IMyProps){
        super(props);
        this.state.value = props.value;
        this.changed = this.changed.bind(this);
   }

   public render(): React.ReactElement<any> {
           return (<div className={ this.className }>
                    <input onChange={this.changed}  type="text"
                           value={this.state.value} />
                  </div>);
   }

   public changed(event): void {
     var newValue: string = event.target.value; }
```

Be sure to bind 'this'

Designate handler

Implement handler

Utilizing JSX

- JSX is a preprocessor step that adds XML syntax to JavaScript
  - It is optional, but very useful for organizing components
  - It requires a transpiler like Traceur, Babel, or TypeScript
  - The following are equivalent:

```
ReactDOM.render(
        React.createElement(Component, { message: "My first component" }),
                document.getElementById("app"));


ReactDOM.render(
        <Component message="My first component" />,
                document.getElementById("app"));
```

Demo

React fundamentals

Component lifecycle

- `componentWillUpdate`
  - executed before component is rendered
- `componentDidUpdate`
  - executed after component is rendered
- `componentWillMount`
  - executed before node is added to the DOM
- `componentDidMount`
  - executed after node is added to the DOM
- `componentWillUnmount`
  - executed before node is removed from the DOM
- `shouldComponentUpdate(newProps, newState)`
  - executed before component is updated

Fetch

- The Fetch standard defines how to fetch all resources
  - https://fetch.spec.whatwg.org/
- Also defines the fetch() JavaScript API

Fetching

```
public componentDidMount(): void {
          fetch(
              '../../_api/web/currentuser',
              {
                  method: 'GET',
                  credentials: 'same-origin',
                  headers: {
                      'accept': 'application/json'
                  }
              }
          ).then(response => {
              return response.json();
          }).then(json => {
              this.setState({ data: json.Title, isValid: true });
          }).catch(e => {
              console.log(e);
          });
  }
```

Critical for SharePoint

Demo

React lifecycle

Developing with SPFX

Capabilities

- Common
  - Office UI Fabric integration
  - `HttpClient` and `GraphHttpClient` class
- Web parts
  - Property pane
- Extensions
  - Application customizer
  - Field customizer
  - Command set

Office UI Fabric

- Office UI fabric is the default front-end framework
  - Office UI Fabric
  - Office UI Fabric React

```
import {
    DocumentCard,
    DocumentCardPreview,
    DocumentCardTitle,
    DocumentCardActivity,
    IDocumentCardPreviewProps
} from 'office-ui-fabric-react/lib/DocumentCard';
```

HttpClient and GraphHttpClient

```
public getContacts(): Promise<Contact[]>{

let url = this.webAbsoluteUrl +
    "/_api/Lists/getByTitle('Contacts')/items?$select=Id,Title,FirstName,WorkPhone,Email";
    this.contacts = [];

return this.httpClient.get(url, SPHttpClient.configurations.v1)
    .then((response: SPHttpClientResponse) => {
        return response.json().then((data) => {
            data.value.forEach(c => {
                this.contacts.push(new Contact(c.Id,c.Title,c.FirstName,c.WorkPhone,c.Email));
            });
            return this.contacts;
        });
    });
}
```

Property pane

- Import field types from `@microsoft/sp-webpart-base`
- Define an interface to save property values
- Define static strings for property pane labels
- Define property pane pages, groups, and controls
- Use property values when rendering web part

Extensions

- New application customizer, field customizer, command
- Define new properties
- Define new functionality

Demo

Developing with SPFX

Deployment

Deployment steps

- Prerequisites
  - Create a CDN and app catalog
- Package
  - Update external references in `config.js`
  - Update `element.xml`
  - Update `write-manifests.json`
  - Bundle and package
- Deploy
  - CDN deployment
  - App catalog deployment
  - Tenant deployment

Create a CDN

- Enable the Office 365 CDN
  - `Set-SPOTenantCdnEnabled -CdnType Public`
  - Create a document library to act as the CDN endpoint
  - `Add-SPOTenantCdnOrigin -CdnType Public -OriginUrl */cdn`
- Utilize an Azure CDN
  - Storage account
  - Blob container
  - CDN profile
- Utilize an on-premises endpoint

Create an App Catalog

- Office 365
  - Admin > Admin Centers > SharePoint
  - Apps > App Catalog
- On-Premises
  - Central Administration > App Management > Manage App Catalog

Update external references in config.js

```
{
    "entries": [
        {
            "entry": "./lib/webparts/ngBasics/NgBasicsWebPart.js",
            "manifest": "./src/webparts/ngBasics/NgBasicsWebPart.manifest.json",
            "outputPath": "./dist/ng-basics.bundle.js"
        }
    ],
    "externals": {
        "rxjs": "https://unpkg.com/rxjs"
    },
    "localizedResources": {
        "ngBasicsStrings": "webparts/ngBasics/loc/{locale}.js"
    }
}
```

Update elements.xml

```
<?xml version="1.0" encoding="utf-8"?>
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">

    <CustomAction
        Title="SPFxApplicationCustomizer"
        Location="ClientSideExtension.ApplicationCustomizer"
        ClientSideComponentId="46606aa6-5dd8-4792-b017-1555ec0a43a4"
        ClientSideComponentProperties="{&quot;Top&quot;:&quot;Top area of the
                        page&quot;,&quot;Bottom&quot;:&quot;Bottom area in the page&quot;}">

    </CustomAction>

</Elements>
```

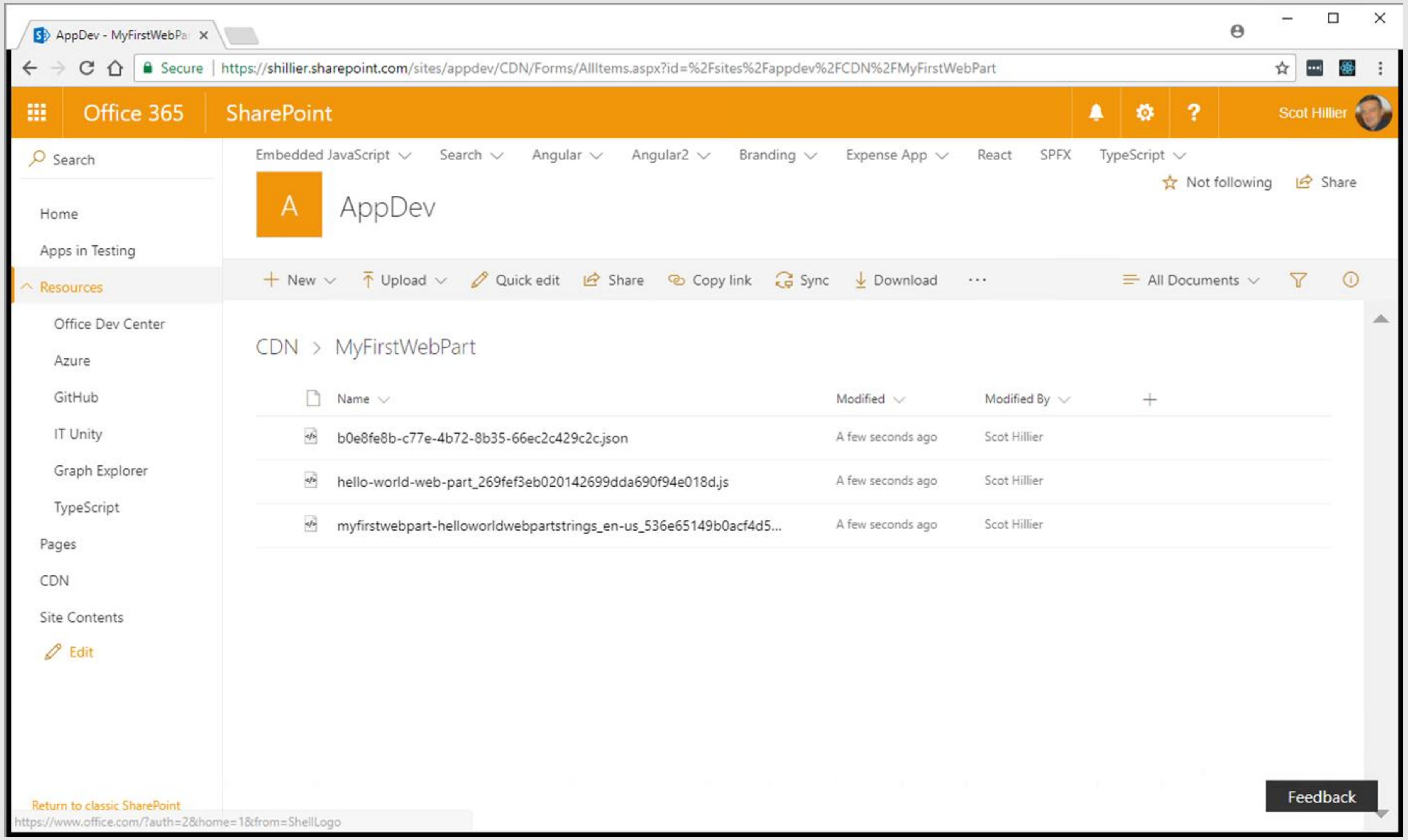
Update write-manifests.json

```
{
  "$schema":
  "https://dev.office.com/json-schemas/spfx-build/write-manifests.schema.json",
  "cdnBasePath":
  "https://publiccdn.sharepointonline.com/contoso.sharepoint.com/CDN/myextension"
}
```

Bundle and package

```
gulp bundle –ship

gulp package-solution --ship
```

Deploy to CDN

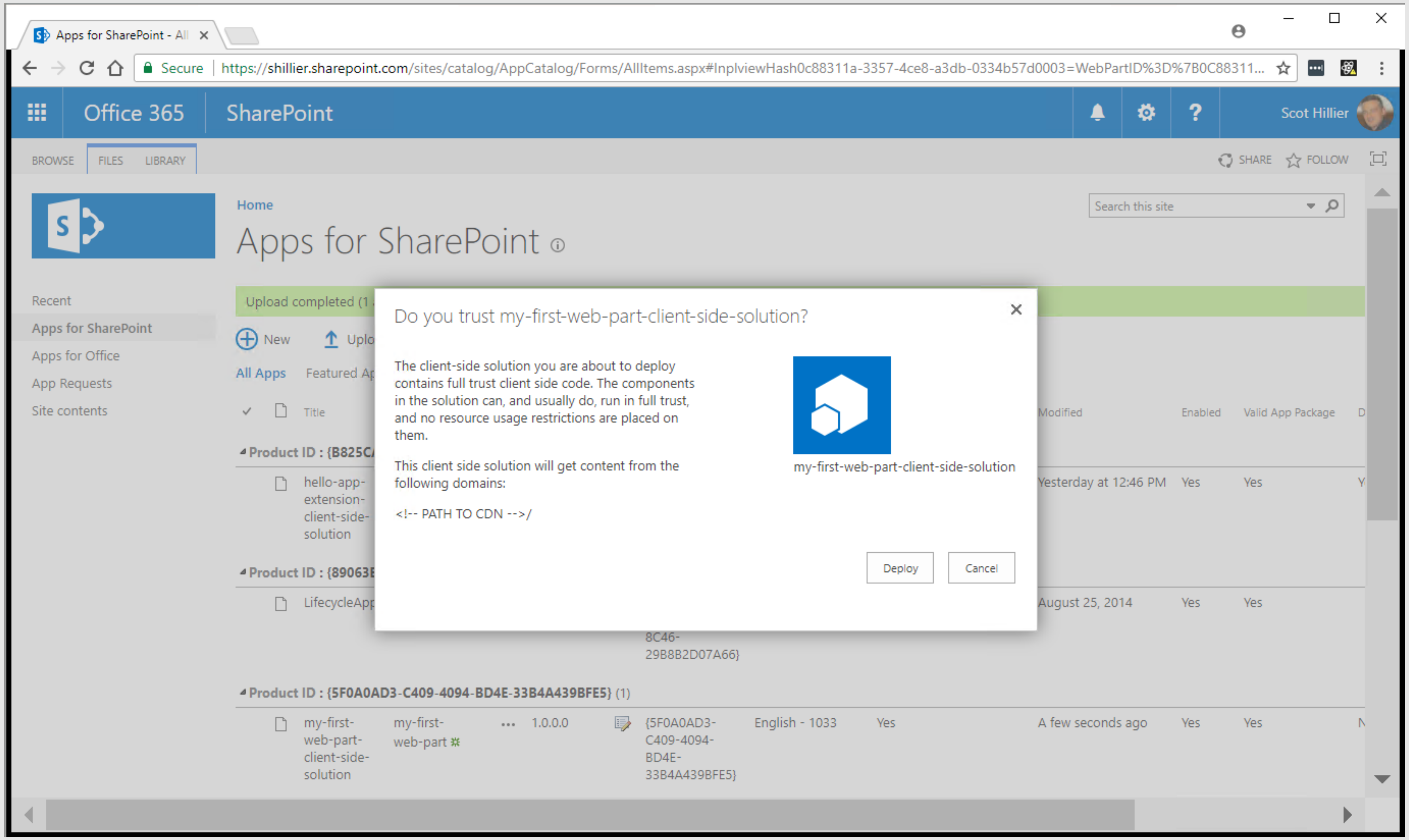


https://shillier.sharepoint.com/sites/appdev/CDN/Forms/AllItems.aspx?id=%2Fsites%2Fappdev%2FCDN%2FMyFirstWebPart

CDN > MyFirstWebPart

Name	Modified	Modified By
b0e8fe8b-c77e-4b72-8b35-66ec2c429c2c.json	A few seconds ago	Scot Hillier
hello-world-web-part_269fef3eb020142699dda690f94e018d.js	A few seconds ago	Scot Hillier
myfirstwebpart-helloworldwebpartstrings_en-us_536e65149b0acf4d5...	A few seconds ago	Scot Hillier

Upload to app catalog



Do you trust my-first-web-part-client-side-solution?

The client-side solution you are about to deploy contains full trust client side code. The components in the solution can, and usually do, run in full trust, and no resource usage restrictions are placed on them.

This client side solution will get content from the following domains:

<!-- PATH TO CDN -->/

my-first-web-part-client-side-solution

Product ID : {B825C

Title	Modified	Enabled	Valid App Package
hello-app-extension-client-side-solution	Yesterday at 12:46 PM	Yes	Yes

Product ID : {89063E

Title					Modified	Enabled	Valid App Package
LifecycleApp	8C46-29B8B2D07A66}				August 25, 2014	Yes	Yes

Product ID : {5F0A0AD3-C409-4094-BD4E-33B4A439BFE5} (1)

Title						Modified	Enabled	Valid App Package
my-first-web-part-client-side-solution	my-first-web-part	1.0.0.0	{5F0A0AD3-C409-4094-BD4E-33B4A439BFE5}	English - 1033	Yes	A few seconds ago	Yes	Yes

Tenant-scoped deployment

```
{
  "solution": {
    "name": "tenant-deploy-client-side-solution",
    "id": "dd4feca4-6f7e-47f1-a0e2-97de8890e3fa",
    "version": "1.0.0.0",
    "skipFeatureDeployment": true
  },
  "paths": {
    "zippedPackage": "solution/tenant-deploy-true.sppkg"
  }
}
```

Demo

Deployment