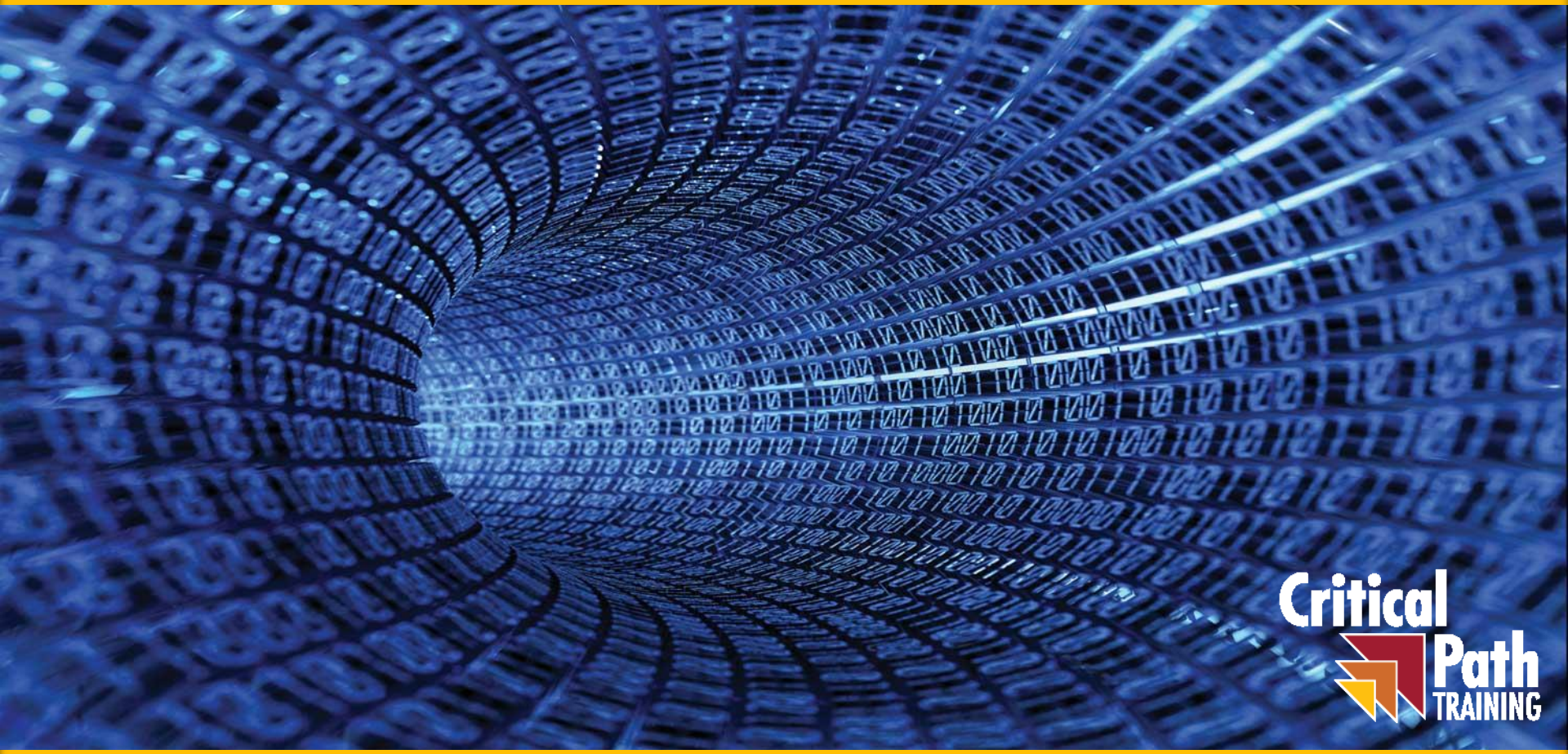
Packaging and Deploying SharePoint Framework Solutions



Agenda

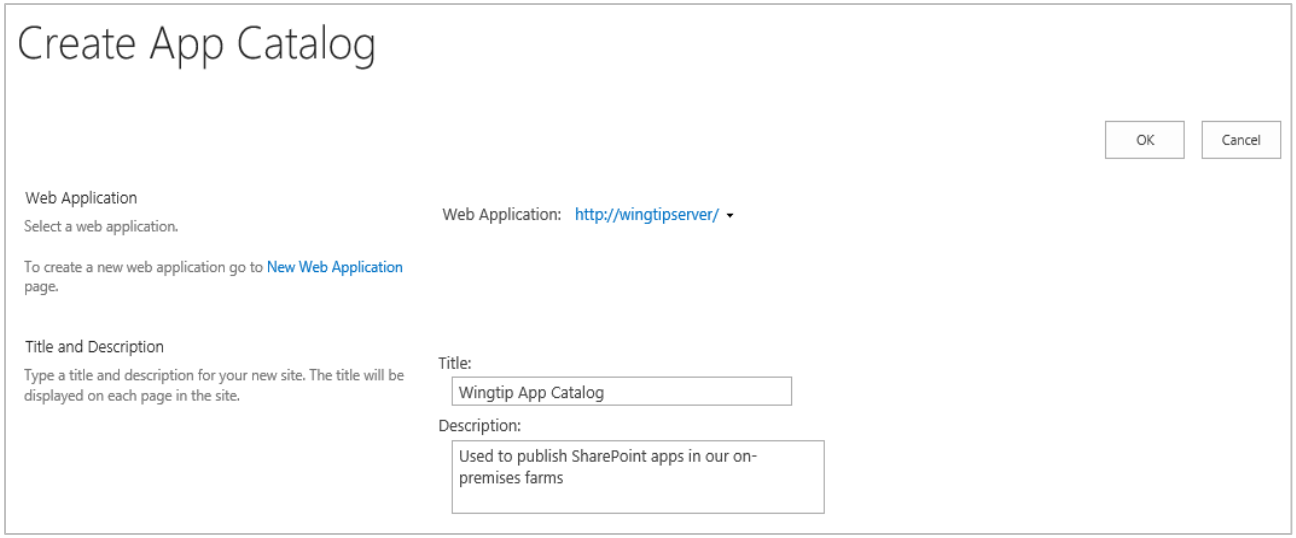
- Creating the App Catalog Site in SharePoint Online
- Packaging SPFX Solutions for Production
- Packaging JavaScript Libraries as External References
- Customizing SPFX Builds with Gulp and Webpack
- Publishing and Updating SPFX Solutions
- Installing an SPFX Solution in a SharePoint Site

Understanding the App Catalog

- App publishing scheme based on App Catalog
  - App Catalog is site collection with special doc library
  - App packages are published (uploaded) to app catalog
  - Provides better app discovery, installation and upgrade
- App Catalog in on-premises farms
  - One App Catalog site required for each web application
  - End users often play role of App Catalog administrator
- App Catalog in Office 365 & SharePoint Online
  - One App Catalog site used to manage entire tenancy

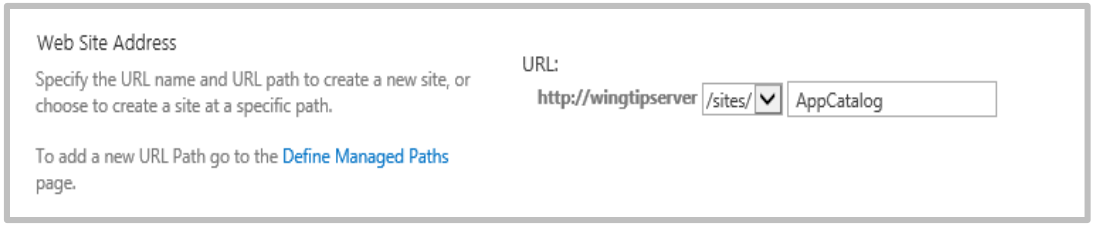
Creating the App Catalog Site Collection

- You must create the App Catalog site collection
  - You can create it using a PowerShell script
  - You can create it using Central Administration
  - App Catalog site associated with one web application



App Catalog URL and Permissions

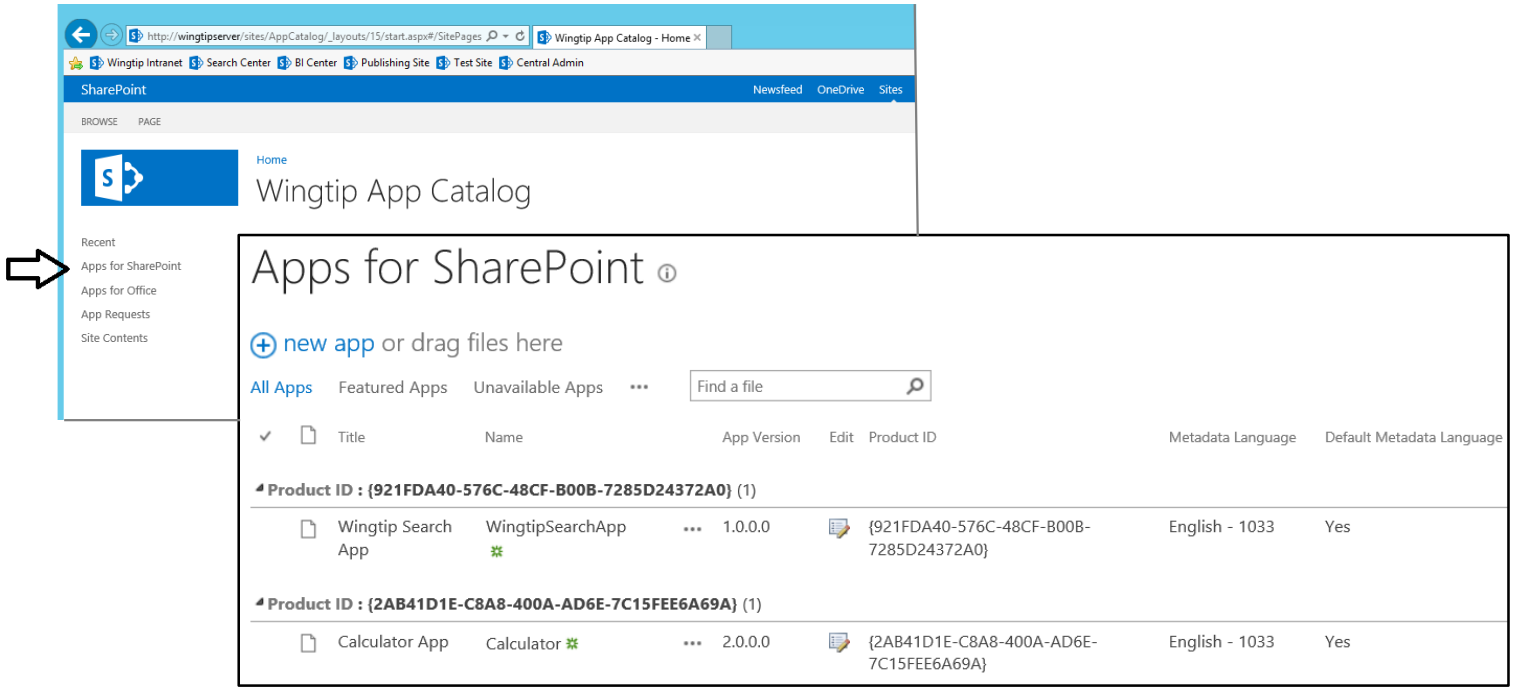
- App catalog site created at a specific URL
  - Creating App Catalog site with PowerShell is more flexible
    you can create site as top-level domain using host-named site collections (HNSCs)



- Setting App Catalog permissions
  - Site collection administrator becomes App Catalog administrator
  - End user permissions allows user to discover and install apps

Apps for SharePoint Document Library

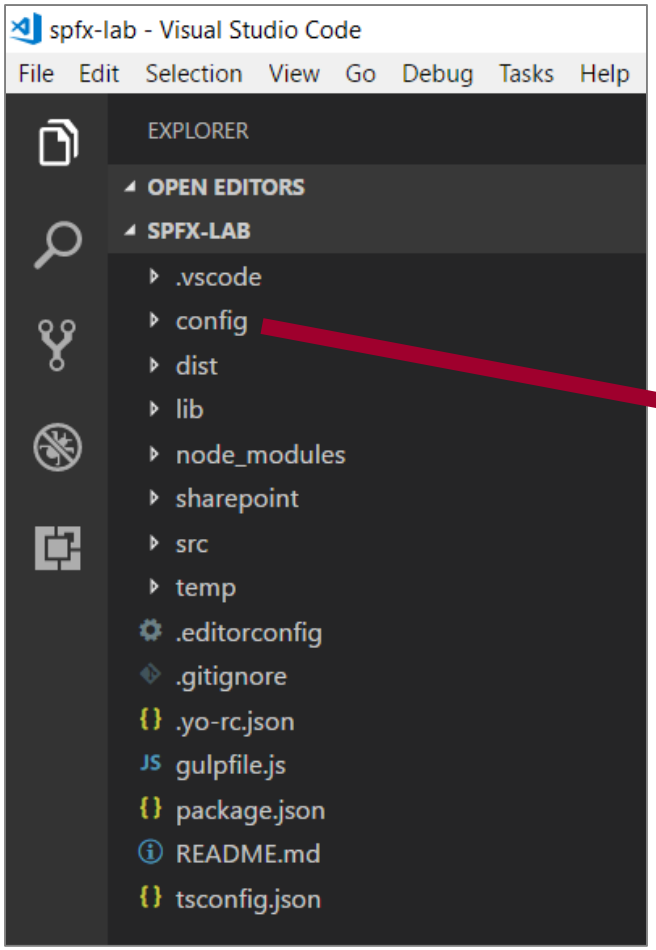
- Apps for SharePoint is special document library
  - It's the place where you publish SharePoint apps
  - You upload app package and enter the related metadata

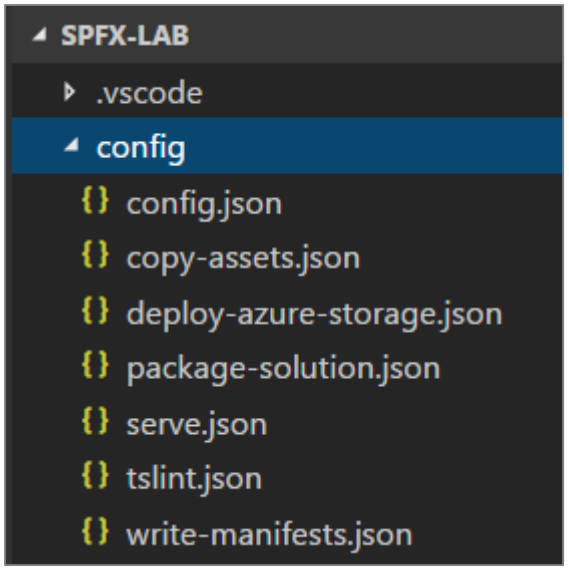


DEMO

Creating an App Catalog Site

SPFx Project Configuration Files





package-solution.json

- package-solution.json
  - Contains top-level project properties (`id, name, version`)
  - `includeClientSideAssets`
  - `skipFeatureDeployment`
  - `zippedPackage`

package-solution.json

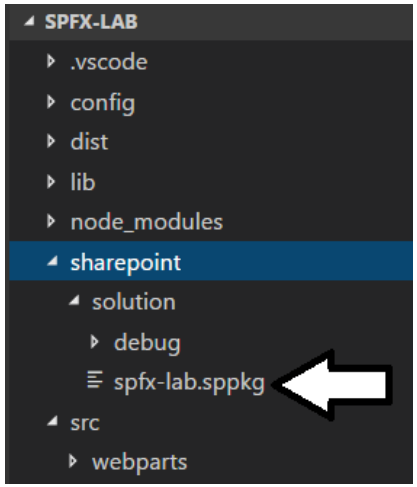
```
{
  "$schema": "https://dev.office.com/json-schemas/spfx-build/package-solution.schema.json",
  "solution": {
    "name": "SPFX Lab",
    "id": "aa5d865d-3cfb-4fd6-a4d4-6fc5d21dd9d7",
    "version": "1.0.0.0",
    "includeClientSideAssets": true,
    "skipFeatureDeployment": true
  },
  "paths": {
    "zippedPackage": "solution/spfx-lab.sppkg"
  }
}
```

Building a SPFx Solution

- Done by executing gulp package-solution
  - Can be done with or without `--ship` parameter

```
PS C:\Demos\spfx-lab> gulp package-solution
Build target: DEBUG
[20:02:30] Using gulpfile C:\Demos\spfx-lab\gulpfile.js
[20:02:30] Starting gulp
[20:02:30] Starting 'package-solution'...
[20:02:30] Starting subtask 'configure-sp-build-rig'...
[20:02:30] Finished subtask 'configure-sp-build-rig' after 4.95 ms
[20:02:30] Starting subtask 'package-solution'...
[20:02:30] Warning - [package-solution] This is not a production build (--ship or --production),
```

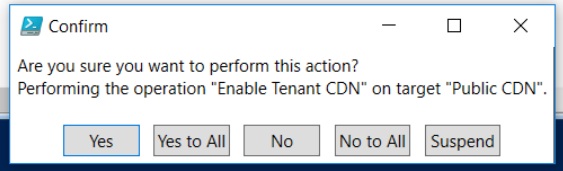
  - Generates zip archive with `.sppkg` extension



Enabling the Office 365 CDN

- Office 365 provides CDN for SPFx solution deployment
  - Office 365 CDN must be enabled using PowerShell

```
Set-SPOTenantCdnEnabled -CdnType Public
```




```
PS C:\Student_New> C:\Student_New\SetupOfficeCDN.ps1

WARNING: This is a feature built on a 3rd party application with privacy and compliance standar
ds that differ from the commitments outlined by the Microsoft Office365 Trust Center. Any data
cached through this service does not conform to the Microsoft Data Processing Terms (DPT) and i
s outside of the Microsoft Office365 Trust Center compliance boundaries.
```

  - Enabling CDN creates `*/CLIENTSIDEASSETS` origin

```
Get-SPOTenantCdnOrigins -CdnType Public

Public CDN enabled locations:
*/MASTERPAGE (configuration pending)
*/STYLE LIBRARY (configuration pending)
*/CLIENTSIDEASSETS (configuration pending)
```

  - CDN supports these file type extensions
    `CSS, EOT, GIF, ICO, JPEG, JPG, JS, MAP, PNG, SVG, TTF, WOFF`

includeClientSideAssets = True

- If Office 365 CDN is enabled…
  - it will be used automatically with default settings.
- If Office 365 CDN is not enabled…
  - assets will be served from app catalog site collection.

package-solution.json

```
{
  "$schema": "https://dev.office.com/json-schemas/spfx-build/package-solution.schema.json",
  "solution": {
    "name": "SPFX Lab",
    "id": "aa5d865d-3cfb-4fd6-a4d4-6fc5d21dd9d7",
    "version": "1.0.0.0",
    "includeClientSideAssets": true,
    "skipFeatureDeployment": true
  },
  "paths": {
    "zippedPackage": "solution/spfx-lab.sppkg"
  }
}
```

Packaging a SPFx Solution for Distribution

- gulp bundle --ship
  - Uses dynamic URL for assets based on tenant CDN settings

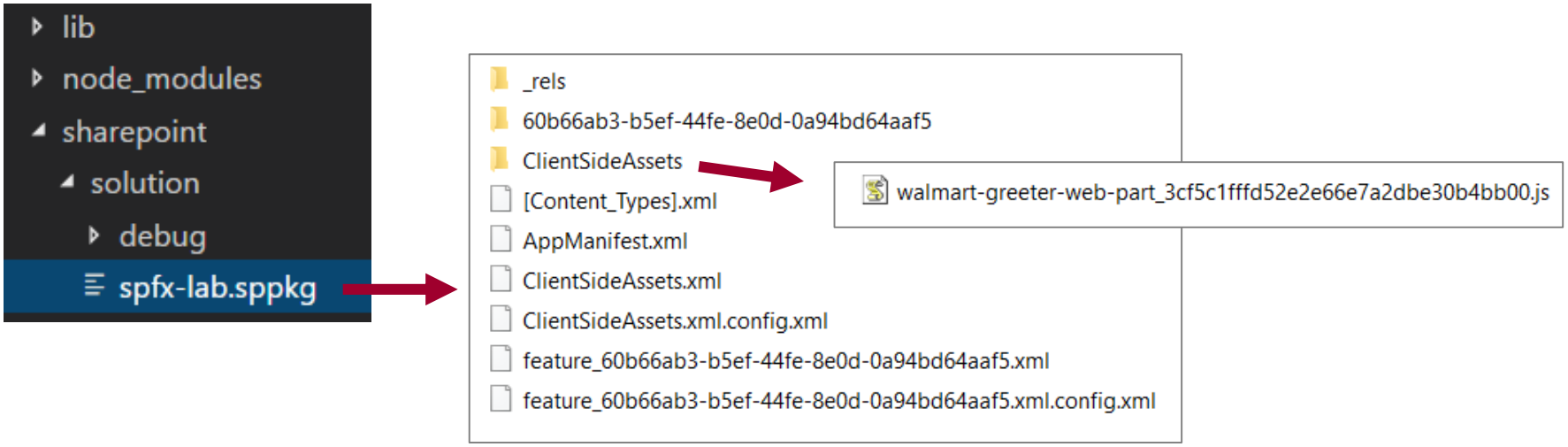
```
PS C:\Demos\spfx-lab> gulp bundle --ship
Build target: SHIP
[20:51:24] Using gulpfile C:\Demos\spfx-lab\gulpfile.js
[20:51:24] Starting gulp
[20:51:24] Starting 'bundle'...
[20:51:24] Starting subtask 'configure-sp-build-rig'...
[20:51:24] Finished subtask 'configure-sp-build-rig' after 4.45 ms
```

- gulp package-solution --ship
  - Uses dynamic URL for assets based on tenant CDN settings

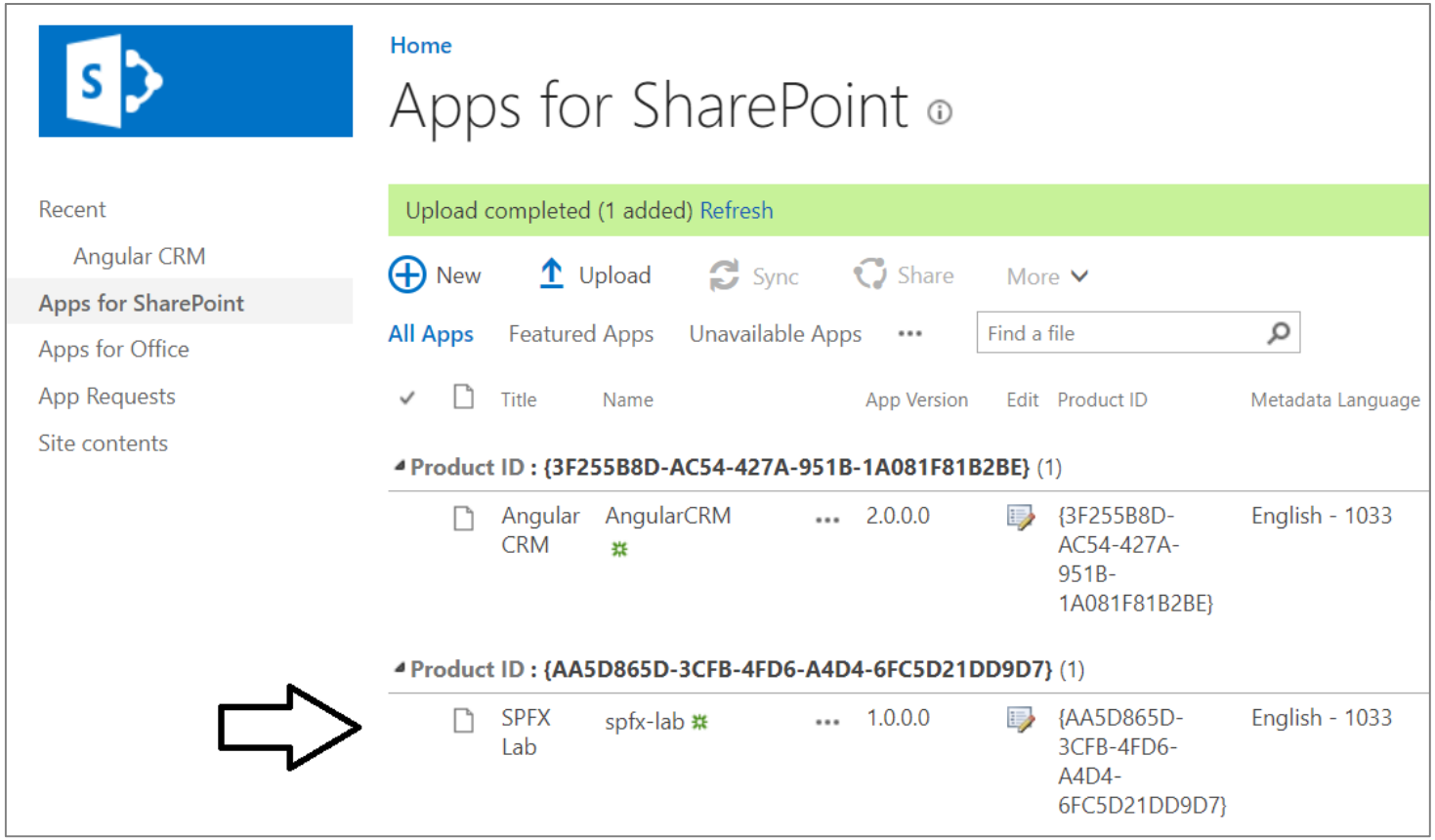
```
PS C:\Demos\spfx-lab> gulp package-solution --ship
Build target: SHIP
[20:53:42] Using gulpfile C:\Demos\spfx-lab\gulpfile.js
[20:53:42] Starting gulp
[20:53:42] Starting 'package-solution'...
[20:53:42] Starting subtask 'configure-sp-build-rig'...
[20:53:42] Finished subtask 'configure-sp-build-rig' after 5.02 ms
[20:53:42] Starting subtask 'package-solution'...
```

Inside a SPFx Solution Package

- Your .js code is included in ClientSideAssets



Deploying Solution to Office 365 Tenancy



Agenda

- ✓ Understanding the SharePoint Online App Catalog
- ✓ Publishing and Installing SharePoint Add-ins
- ✓ Packaging SharePoint Framework Projects
- ➢ Deploying Provider-hosted Add-ins

Package the web part

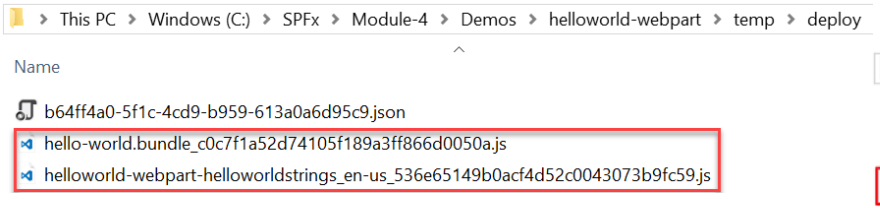
Use the **bundle** gulp task to build, localize, and bundle the project

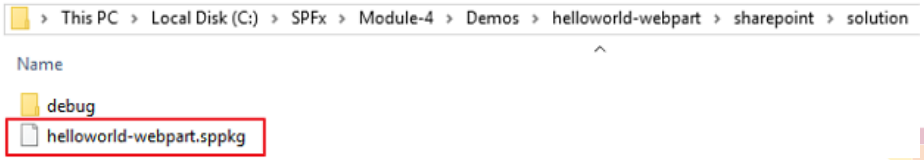
```
> gulp bundle --ship
```

Use the **package-solution** gulp task to package the project into a .sppkg file

```
> gulp package-solution --ship
```

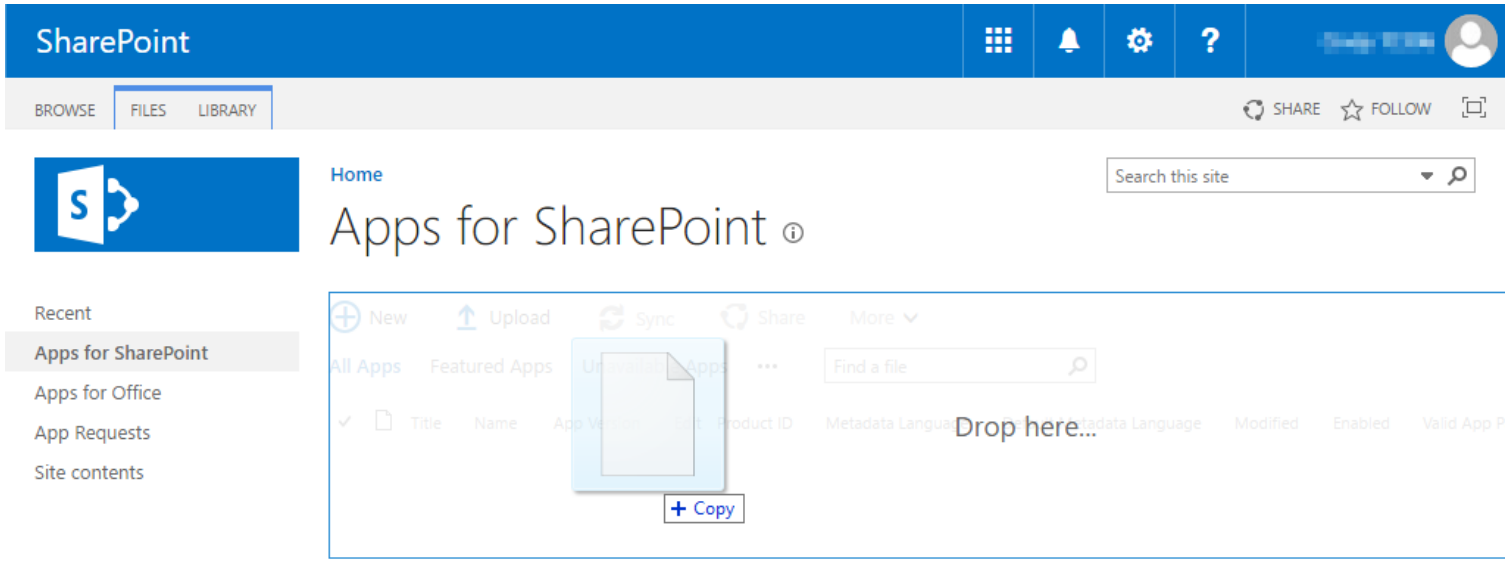
The **ship parameter** build task creates a minified version of the bundle and copies all of the web part assets, including the web part bundle, into the temp\deploy folder. The .sppkg file is generated in the sharepoint\solution folder.





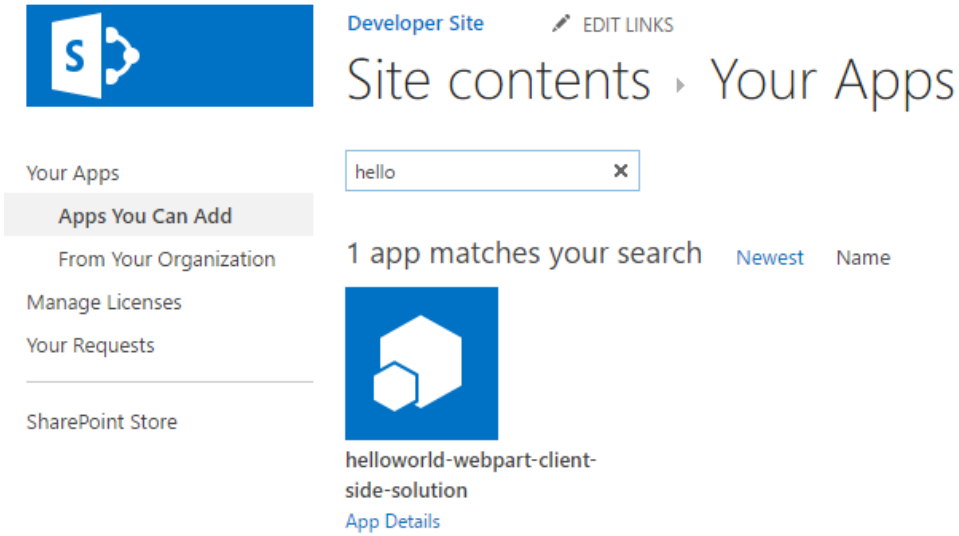
Deploy App to the SharePoint App Catalog

- Go to your Office 365 App Catalog site
- In the left sidebar, choose Apps for SharePoint
- Upload the package (.sppkg file) for the web part



Install the App

- Go to your Office 365 site
- Add the App you just deployed to the SharePoint App Catalog



Deploy assets to a CDN and configure web part

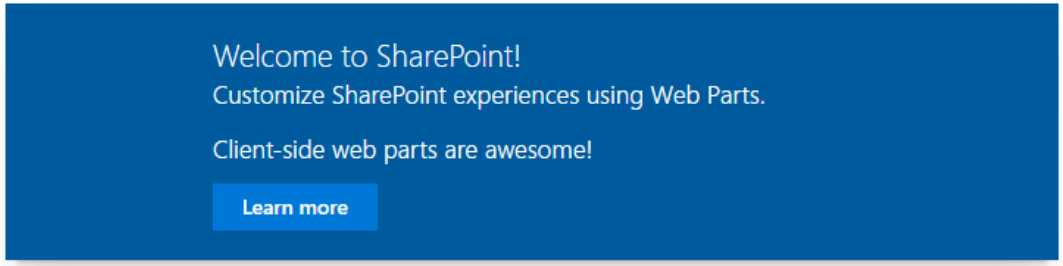
- At this point the web part will not run
- You must first
  - deploy the web part assets to a CDN location
  - update the cdnBasePath in the write-manifests.json file

```
{
  "cdnBasePath": "<!-- PATH TO CDN -->"
}
```

- See the other sections in this module that describe how to
  - deploy to a SharePoint CDN and an Azure Storage CDN
  - update the cdnBasePath in the write-manifests.json file

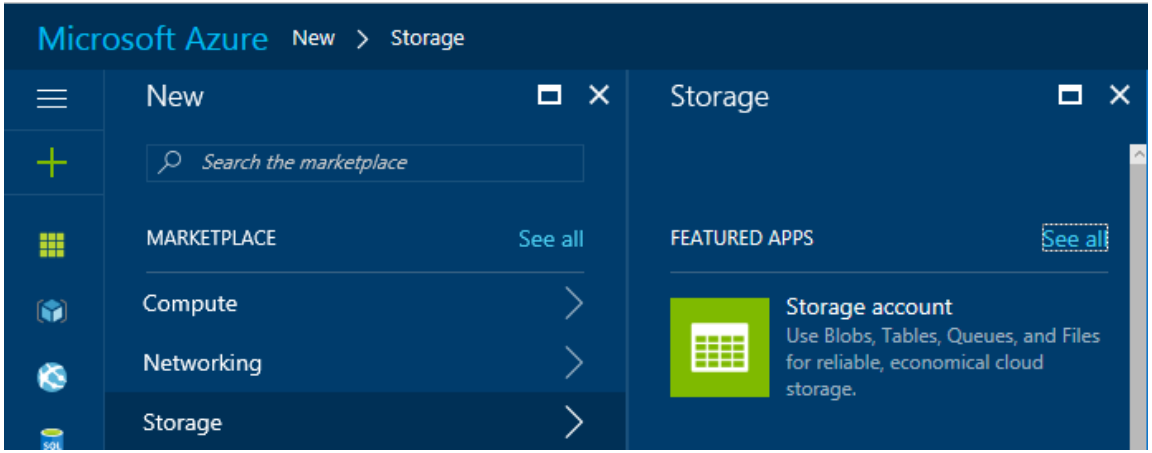
Add the web part to a SharePoint page

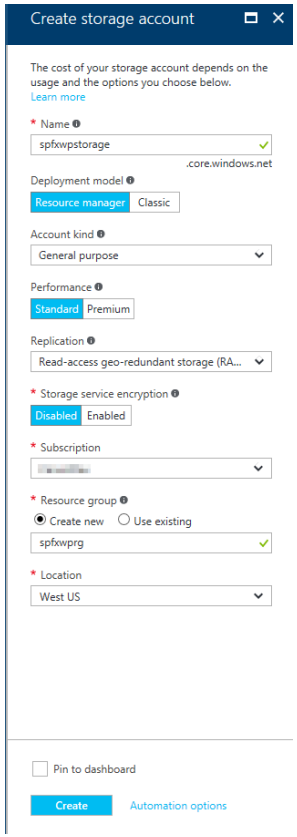
- After the web part assets are deployed to a CDN location you can preview the web part
- Create a page on your Office 365 site
- Add the web part to the page



Create Azure Storage Account

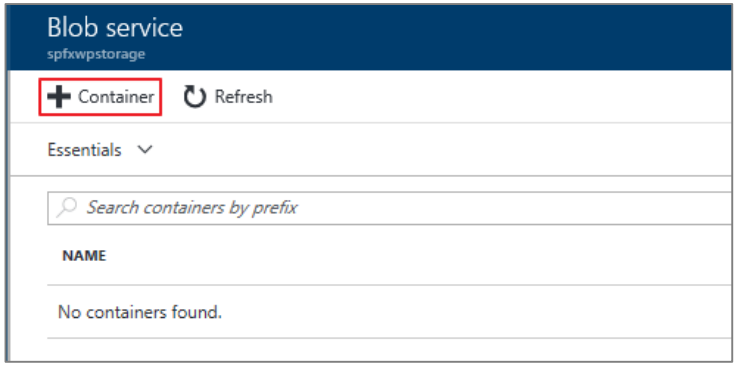
- Go to the Azure Management Portal
- Create a new Storage Account

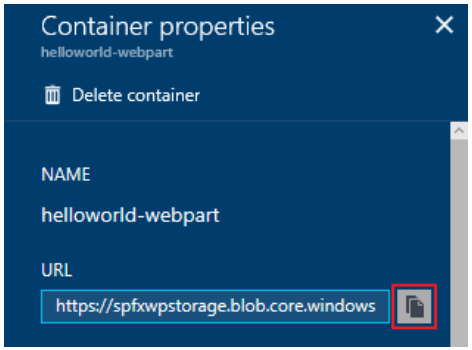




Create Blob Container in Storage Account

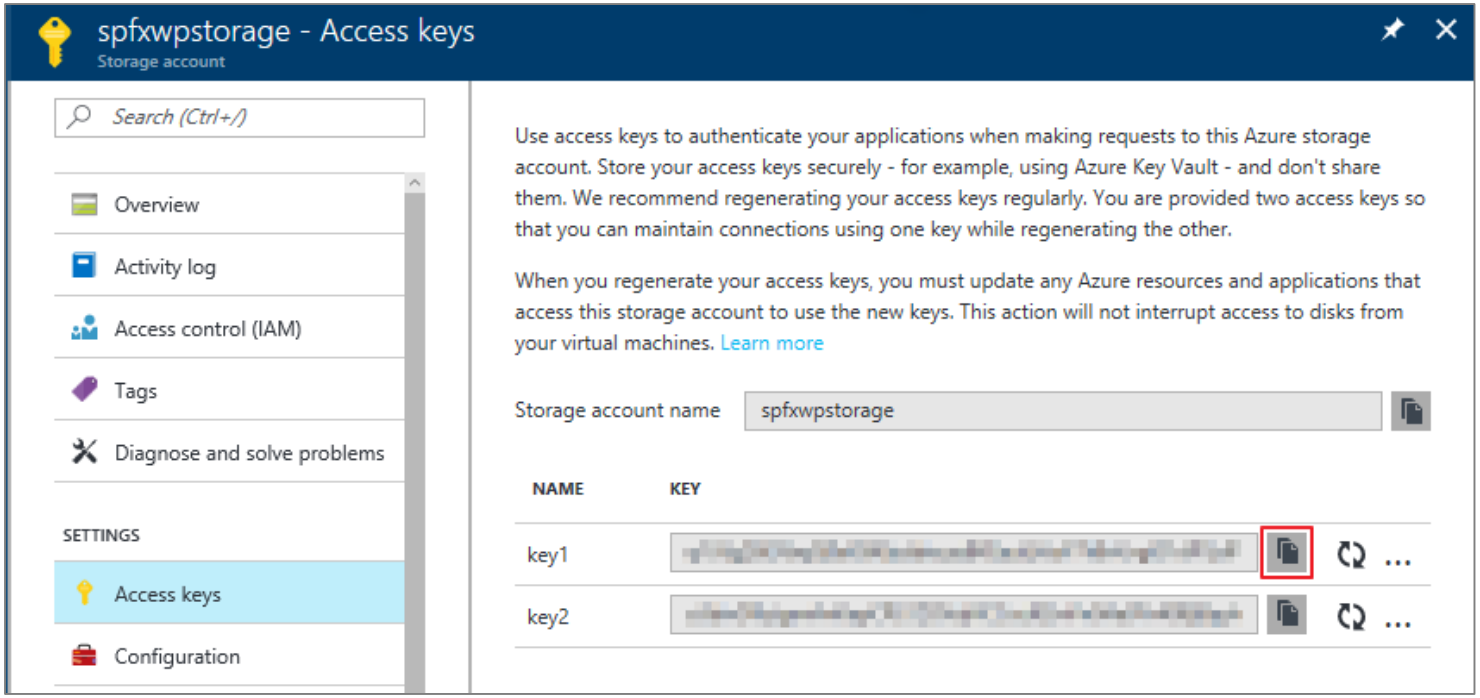
- Create a Blob Container and copy the URL





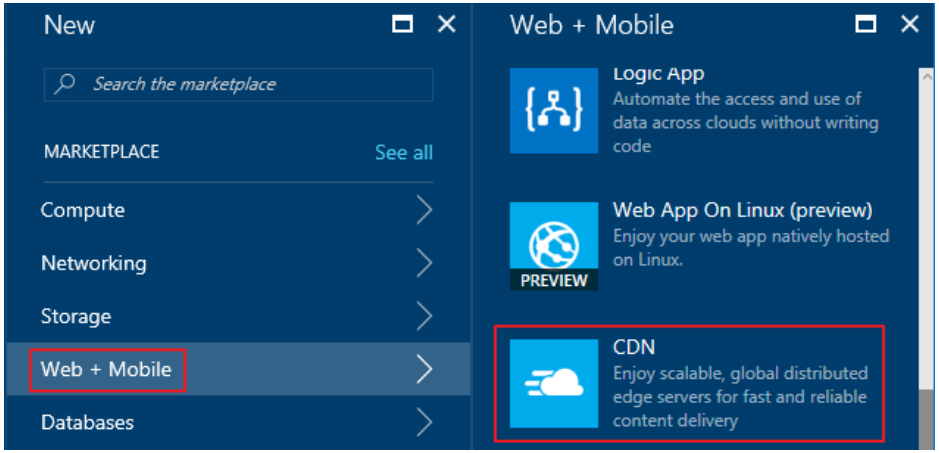
Obtain Storage Account Access Key

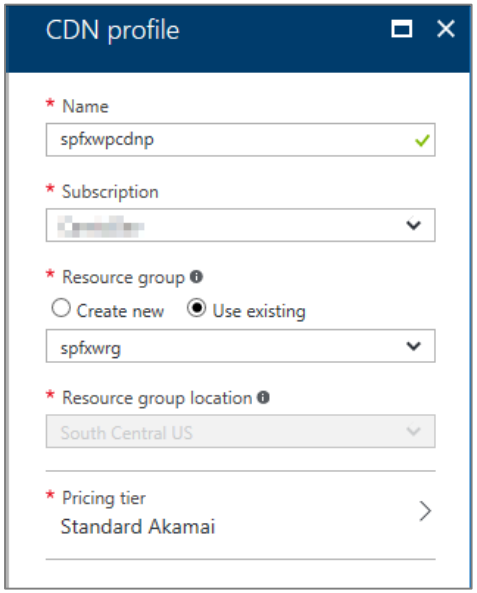
- The Storage Account Access Key is used to automate deployments to the Azure Storage CDN, copy one of the keys



Create CDN Profile

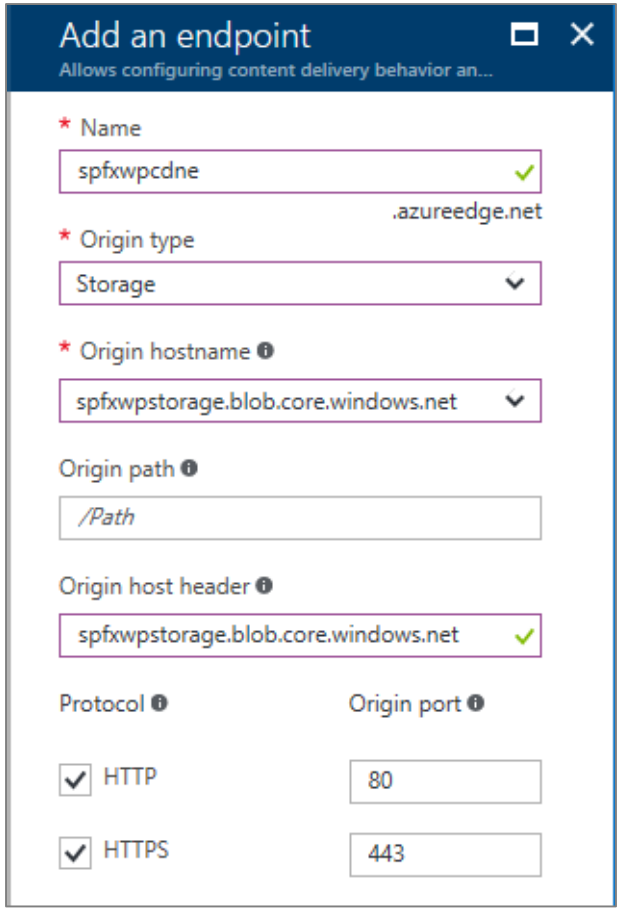
- In the Azure Management Portal, create a new CDN Profile





Create CDN Endpoint

- In the CDN Profile, create a new CDN endpoint



Configure Webpart to Deploy Assets to CDN

- Update the account, container, and accessKey in the deploy-azure-storage.json file
  - account – Storage account name
  - container – Name of the container you wish to use for the web part
  - accessKey – Storage Account Access Key

```
{
  "workingDir": "./temp/deploy/",
  "account": "spfxwpstorage",
  "container": "helloworld-webpart",
  "accessKey": "hL2503 … "
}
```

Configure the web part to use the CDN

- Create CDN base path

  https://<Storage Account Name>.blob.core.windows.net/<Container Name>

- Update the cdnBasePath in the write-manifests.json file

```
{
  "cdnBasePath": "https://spfxwpstorage.blob.core.windows.net/helloworld-webpart"
}
```

-

Deploy web part assets to the Azure Storage Account

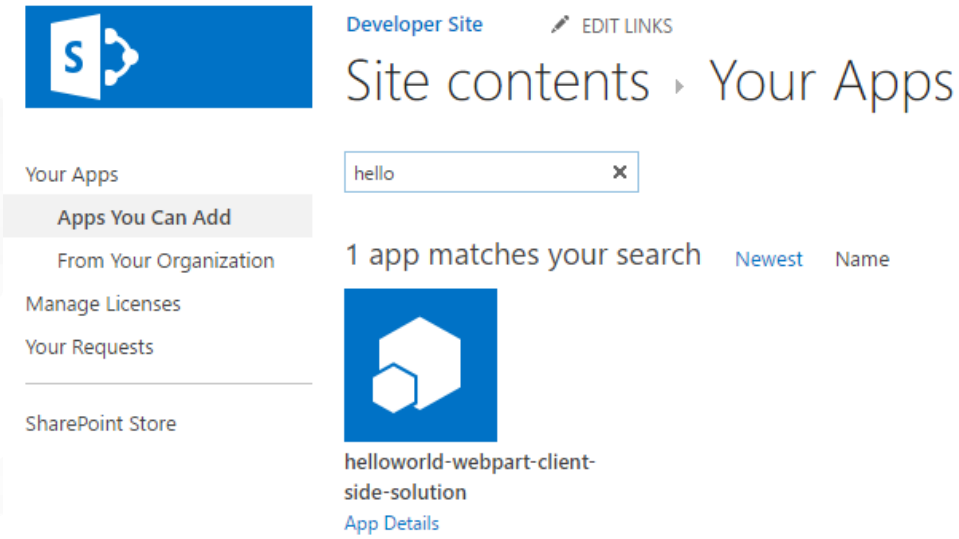
- Use the deploy-azure-storage gulp task to deploy the assets to the Azure Storage Account

```
> gulp deploy-azure-storage
```

```
> gulp deploy-azure-storage
Build target: DEBUG
[11:56:24] Using gulpfile C:\SPFx\helloworld-webpart\gulpfile.js
[11:56:24] Starting gulp
[11:56:24] Starting 'deploy-azure-storage'...
[11:56:24] Starting subtask 'deploy-azure-storage'...
[11:56:24] [deploy-azure-storage] Uploading files '**/*.*' from directory './temp/deploy/' to Azure
[11:56:25] [deploy-azure-storage] Created container: helloworld-webpart
[11:56:25] [deploy-azure-storage] Uploading [3] files...
[11:56:26] [deploy-azure-storage] Uploaded file: helloworld-webpart-helloworldstrings_en-us_536e65149b0acf4d52c0043073b9fc59.js
[11:56:26] [deploy-azure-storage] Uploaded file: hello-world.bundle_b8a80975dedeb31de300b580fab61182.js
[11:56:26] [deploy-azure-storage] Uploaded file: dd331d09-a9cd-448d-a687-7e43060191e2.json
[11:56:26] [deploy-azure-storage] Upload complete!
[11:56:26] [deploy-azure-storage] Access your files at: https://spfxwpstorage.blob.core.windows.net/helloworld-webpart
[11:56:26] Finished subtask 'deploy-azure-storage' after 1.92 s
[11:56:26] Finished 'deploy-azure-storage' after 1.93 s
[11:56:26] =================[ Finished ]=================
[11:56:27] Project helloworld-webpart version: 0.0.1
[11:56:27] Build tools version: 2.4.0
[11:56:27] Node version: v6.10.0
[11:56:27] Total duration: 5.94 s
```

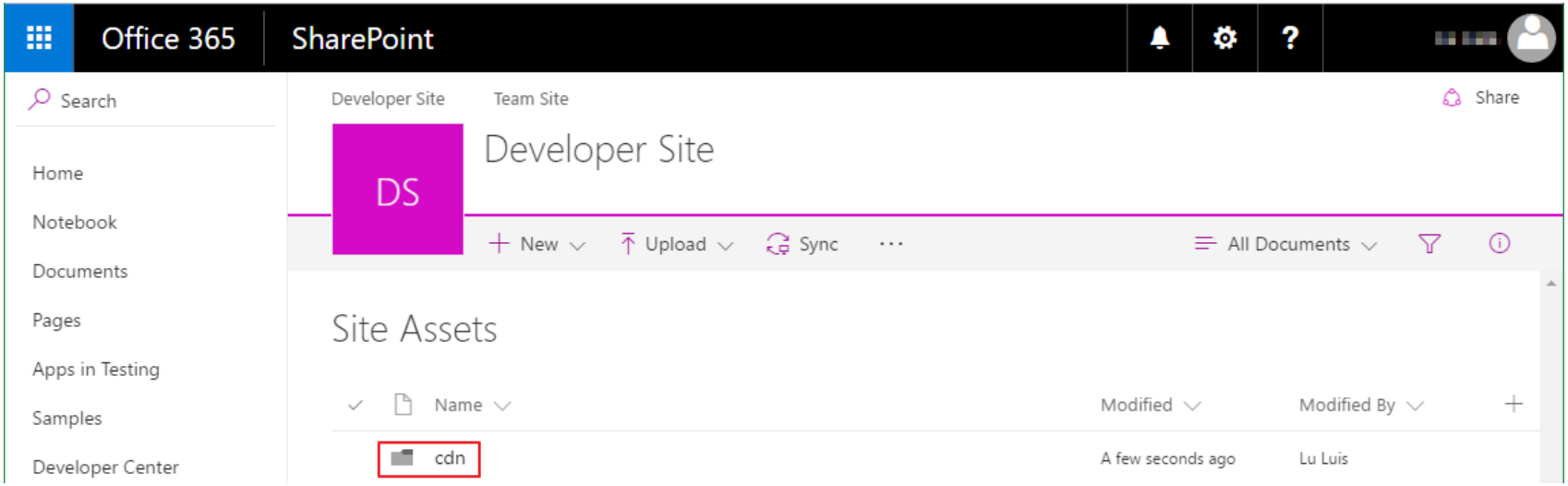
Install the App

- **Go to your Office 365 site**
- **Add the App you just deployed to the SharePoint App Catalog**



Create SharePoint CDN origin

- Go to the Site Assets Document Library
- Create a folder for the CDN origin



Enable the SharePoint CDN origin

- Open the SharePoint Online Management Shell
- Execute the following Powershell commands to enable the CDN, set the CDN origin, and return the CDN Origin ID

```
> $creds = Get-Credential
> Connect-SPOService -Url https://<TENANCY>-admin.sharepoint.com/ -Credential $creds
> Set-SPOTenant -PublicCdnEnabled $true
> Set-SPOTenant -PublicCdnAllowedFileTypes "CSS,EOT,GIF,ICO,JPEG,JPG,JS,MAP,PNG,SVG,TTF,WOFF,TXT"
> New-SPOPublicCdnOrigin -Url "https://<TENANCY>.sharepoint.com/sites/<SITE>/siteassets/cdn"
> Get-SPOPublicCdnOrigins
```

```
PS C:\WINDOWS\System32> Get-SPOPublicCdnOrigins

Id                                                                  Url
--                                                                  ---
19690057cb318c9369ed0bdea07d8ac5326be[redacted]ace05438 HTTPS://[redacted].SHAREPOINT.COM/SITES/DEV/...
```

Configure the web part to use the CDN

Create CDN base path

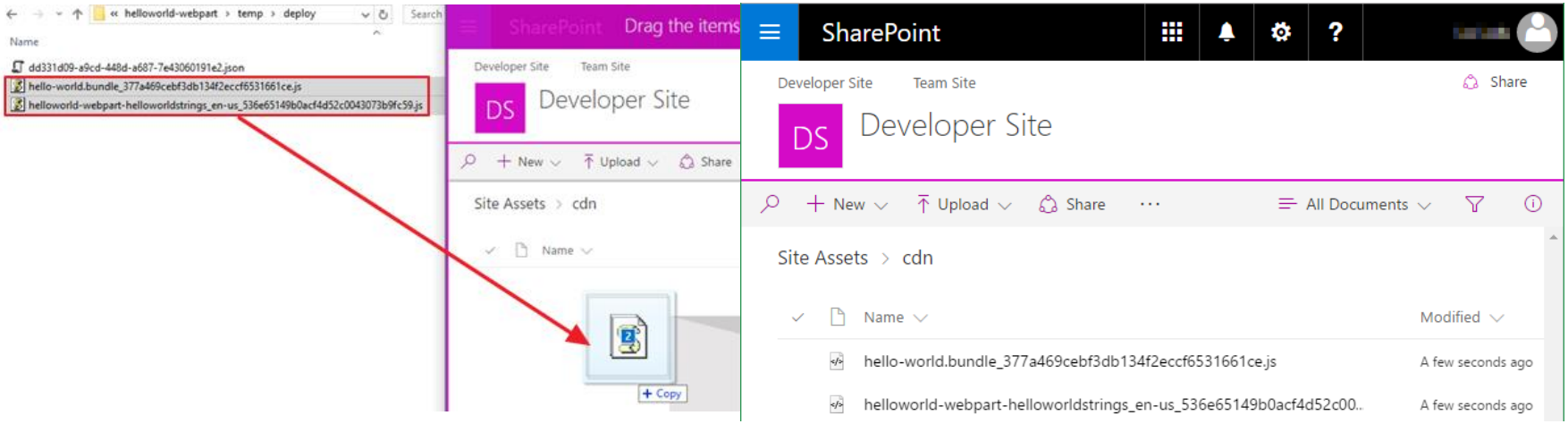
https://publiccdn.sharepointonline.com/<TENANCY>.sharepoint.com/<your-CDN-origin-Id>

Update the cdnBasePath in the write-manifests.json file

```
{
  "cdnBasePath": "https://publiccdn.sharepointonline.com/<TENANCY>.sharepoint.com/<your-CDN-origin-Id>"
}
```

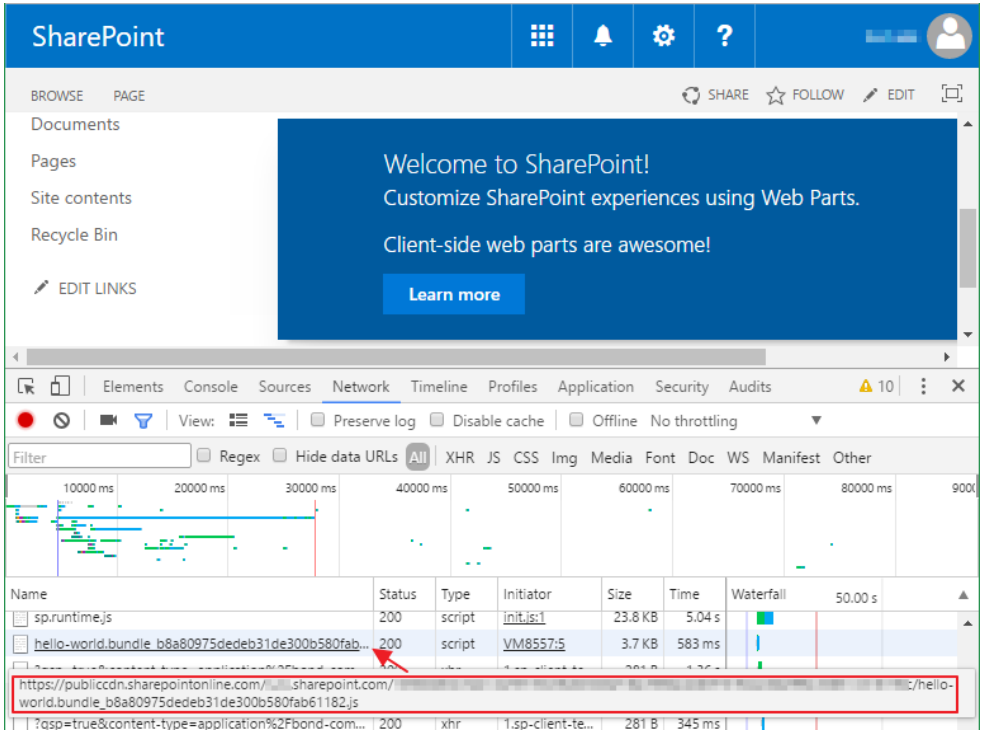
Deploy web part assets to the SharePoint CDN

- Go to the Site Assets Document Library
- Upload the assets for your web part to the CDN enabled folder



Add the web part to a SharePoint page

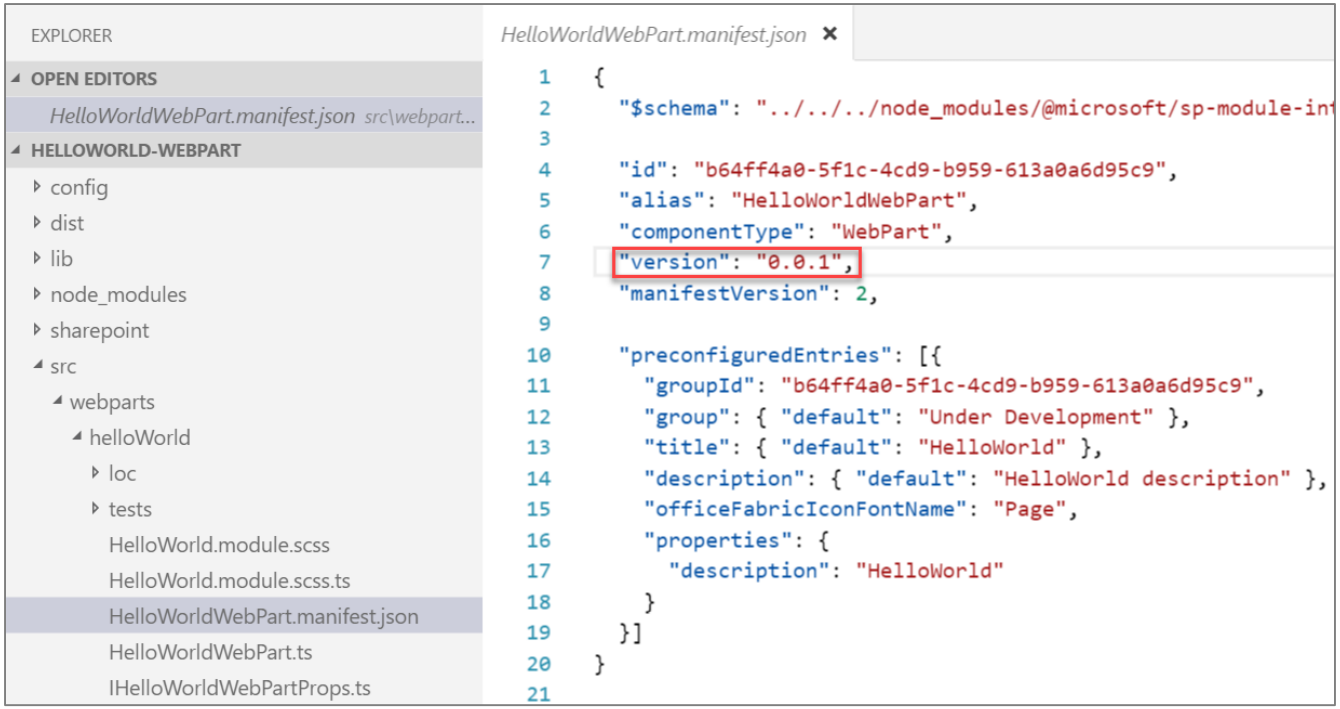
- After the web part assets are deployed to a CDN location you can preview the web part
- Create page in your SPO site and add the web part to the page
- In the developer console verify the assets are served from the CDN



Update Webparts and Increment Version

Update the code or configuration for your web part

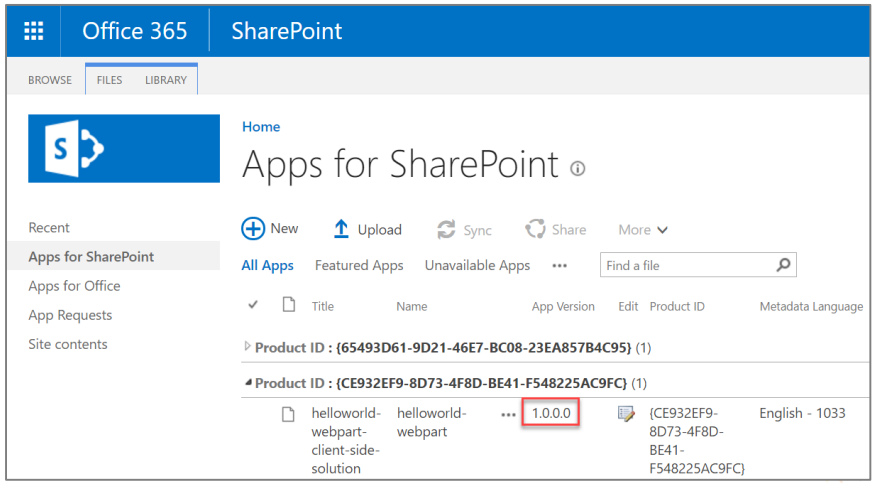
- Increment the version in the <web part name>.manifest.json file
- This sets the version for the web part
- Multiple web parts in the same solution may be versioned independently



Update Version in package-solution.json

This sets the version for the .sppkg Add-in

This version is displayed in the app catalog



Update the version in the package.json file

- Change the version every time the package changes

EXPLORER

OPEN EDITORS
- package.json

HELLOWORLD-WEBPART
- config
- dist
- lib
- node_modules
- sharepoint
- src
- temp
- typings
- .editorconfig
- .gitattributes
- .gitignore
- .npmignore
- .yo-rc.json
- gulpfile.js
- package.json
- README.md
- tsconfig.json

package.json

```
{
  "name": "helloworld-webpart",
  "version": "0.0.1",
  "private": true,
  "engines": {
    "node": ">=0.10.0"
  },
  "dependencies": {
    "@microsoft/sp-client-base": "~0.7.0",
    "@microsoft/sp-core-library": "~0.1.2",
    "@microsoft/sp-webpart-base": "~0.4.0",
    "@microsoft/sp-client-preview": "~0.9.0",
    "@types/webpack-env": ">=1.12.1 <1.14.0"
  },
  "devDependencies": {
    "@microsoft/sp-build-web": "~0.9.0",
    "@microsoft/sp-module-interfaces": "~0.7.0",
    "@microsoft/sp-webpart-workbench": "~0.8.0",
    "gulp": "~3.9.1",
    "@types/chai": ">=3.4.34 <3.6.0",
    "@types/mocha": ">=2.2.33 <2.6.0"
  },
  "scripts": {
```