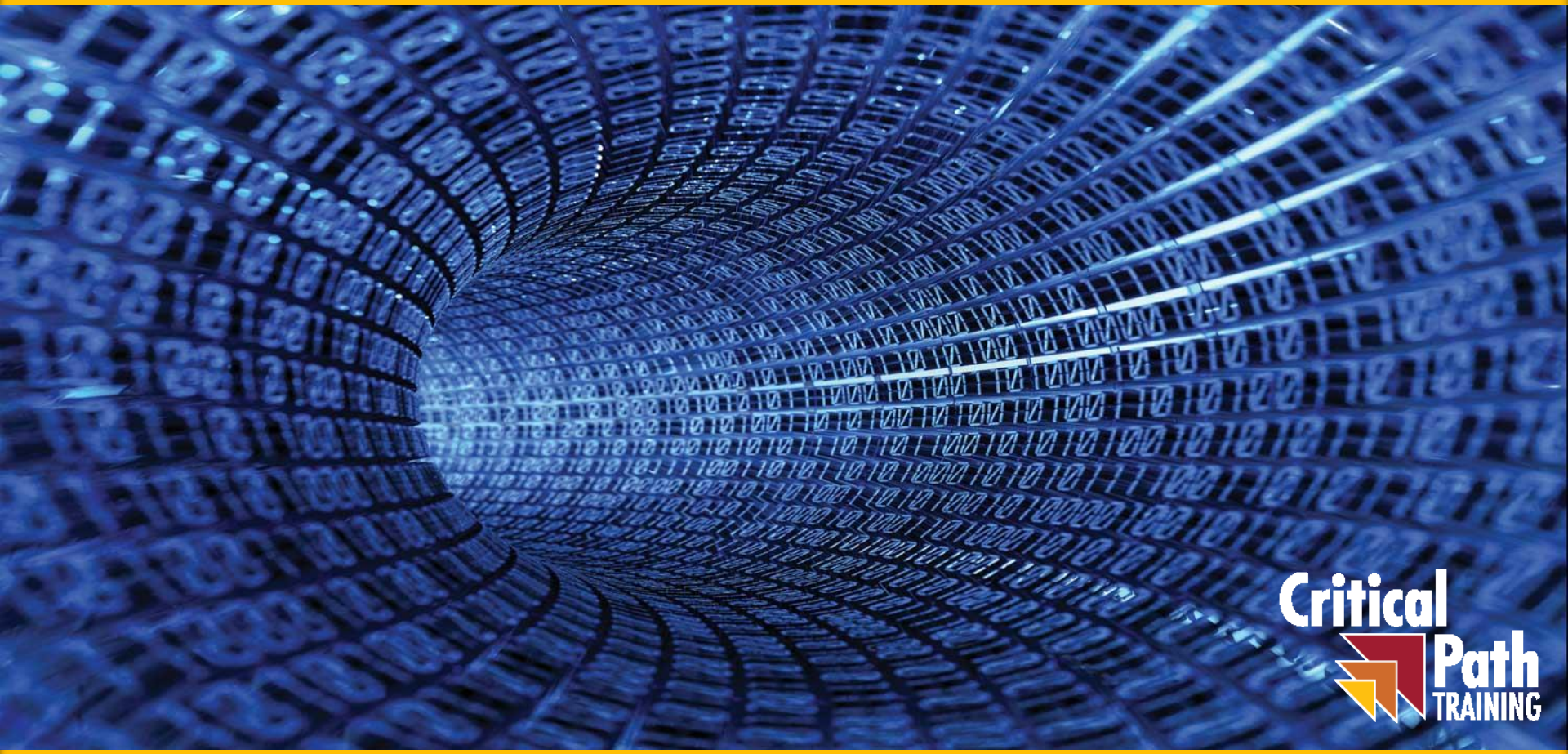
Publishing, Installing and Upgrading Add-ins



Agenda

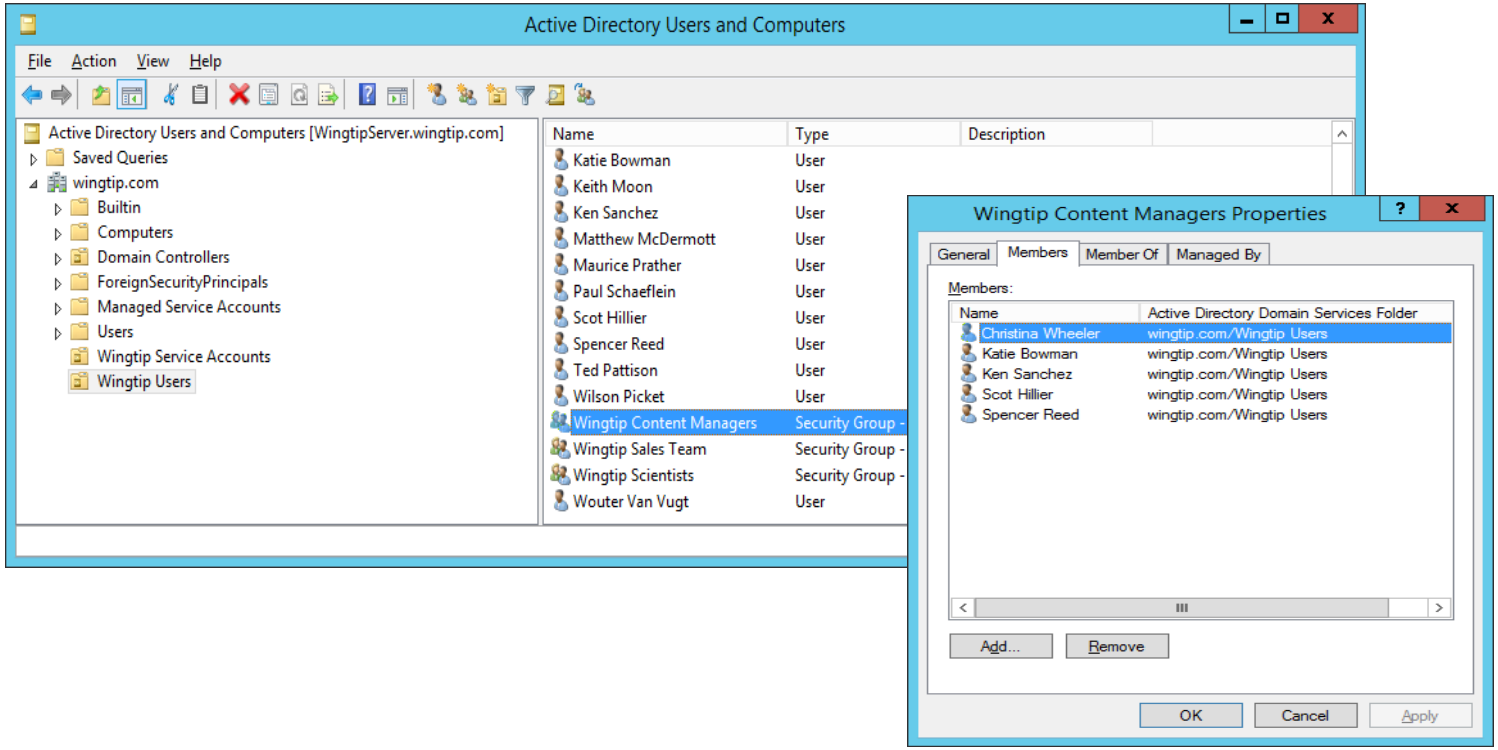
- Creating the App Catalog
- Publishing Apps in the App Catalog
- Installing and Upgrading Apps
- Deploying Provider-hosted Apps

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

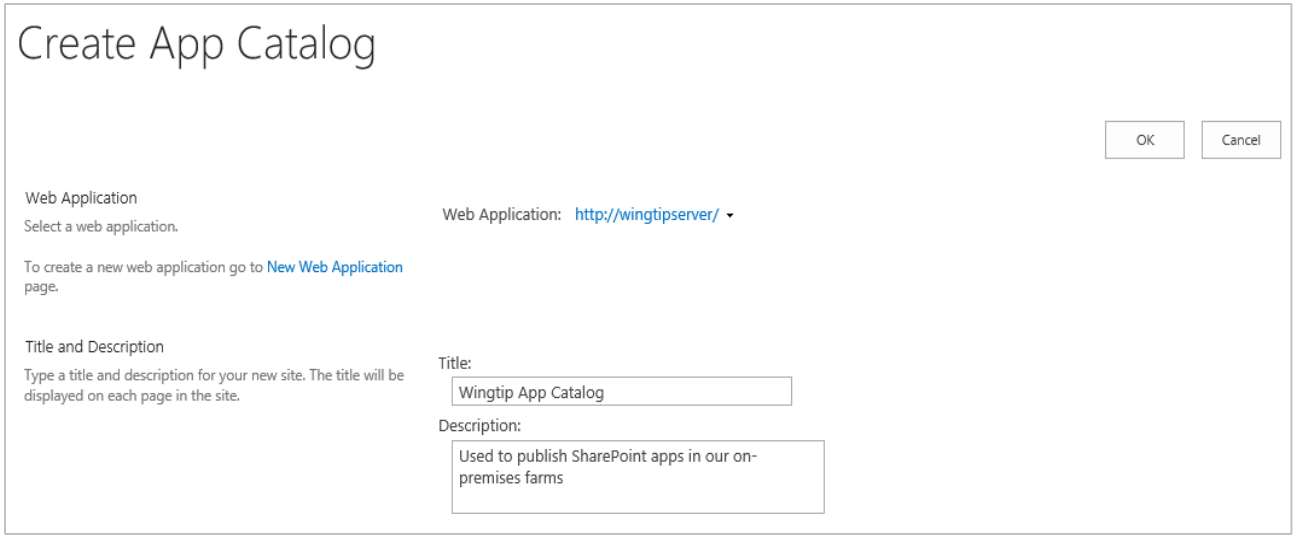
Who Manages the App Catalog?

- App Catalog can be managed by business users
  - IT staff doesn't have to install and manage apps
  - Security can be configured using Active Directory group



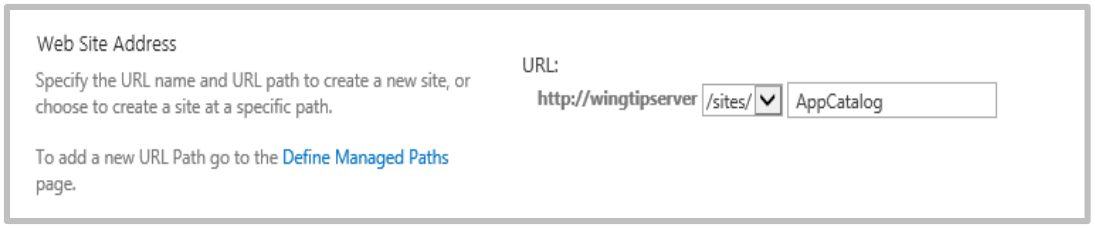
Creating the App Catalog Site Collection

- 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

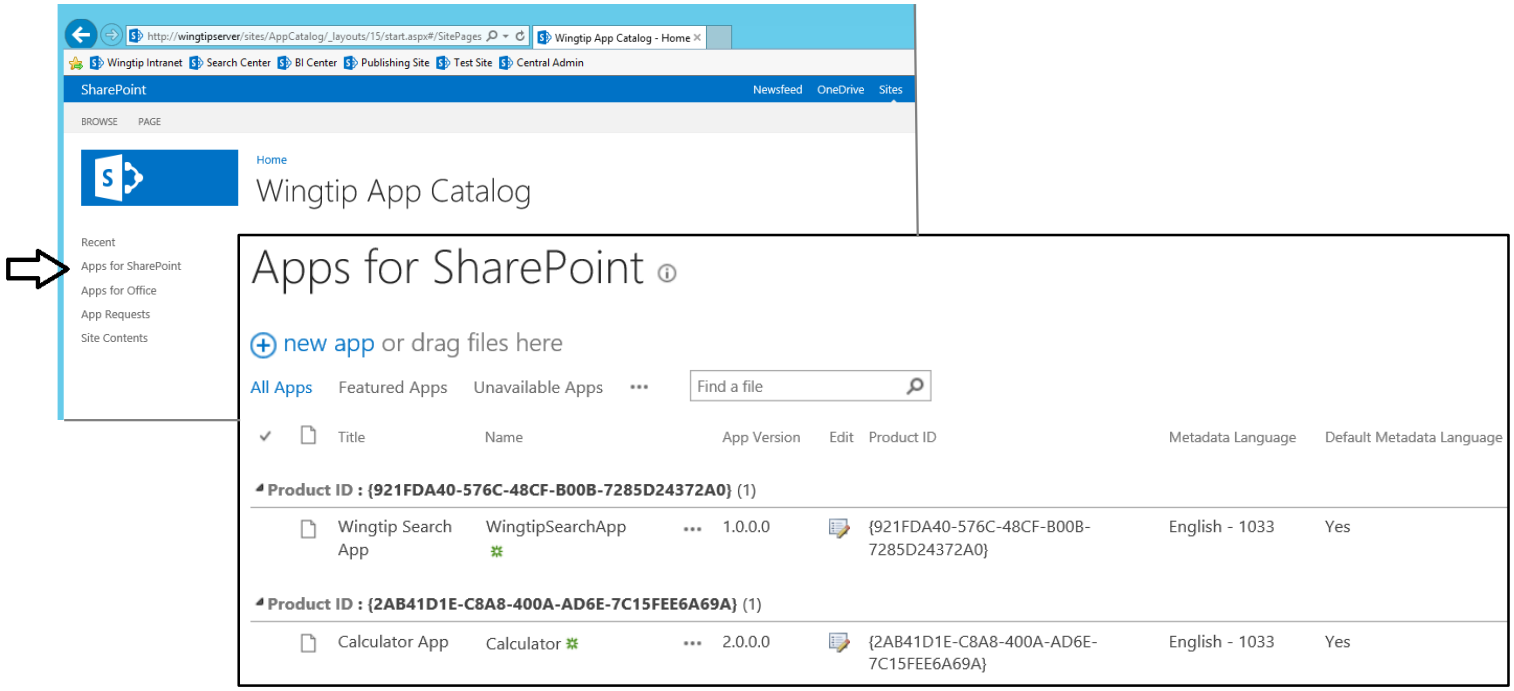
- App catalog site created at a specific URL
  - Creating App Catalog site with PowerShell is more flexible
    you can create site as top-level domain using host-named site collections (HNSCs)



- 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

Agenda

- ✓ Creating the App Catalog
- ➢ Publishing Apps in the App Catalog
- Installing and Upgrading Apps
- Deploying Provider-hosted Apps

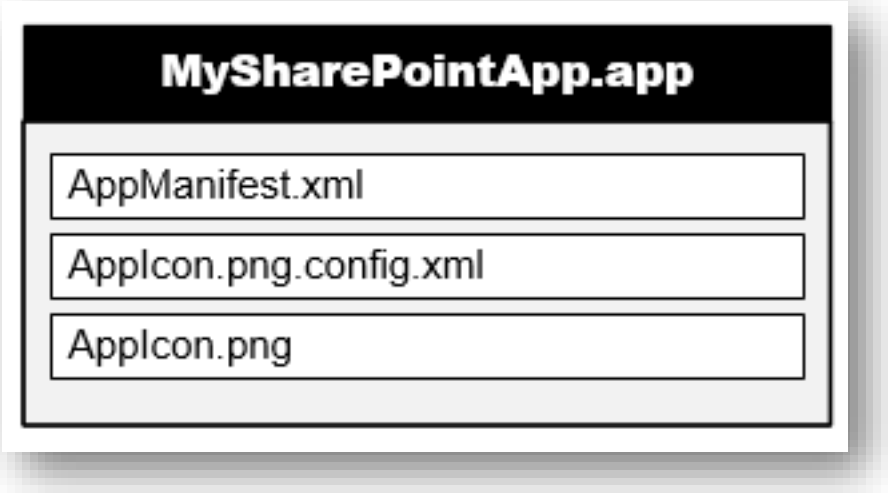
Add-in Deployment Overview

for a SharePoint-hosted Add-in

1. Package the add-in
   - Package add-in resources in distributable file
2. Publish the add-in
   - Make add-in available for installation and upgrade
3. Install the add-in
   - Make add-in available for use
4. Upgrade the add-in
   - Replace current version of add-in with newer version

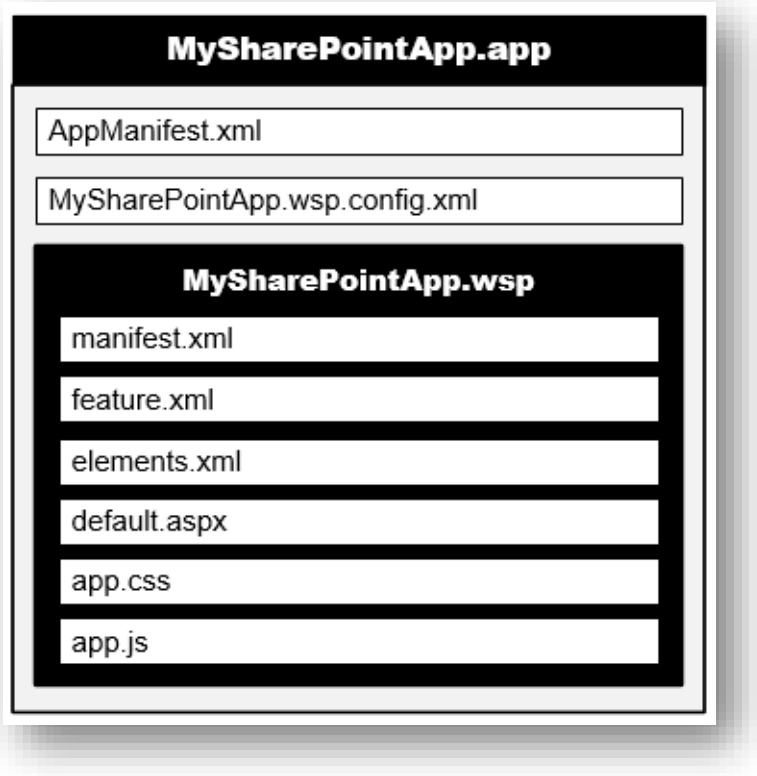
App Package

- SharePoint apps distributed using app packages
  - App package is ZIP archive file with `*.app` extension
  - Built according to Open Package Convention (OPC)
  - Same packaging format used in Apps for Office
  - App package must contain `AppManifest.xml`
  - App package will often contain file for app icon



App Web Solution Package

- App package contains inner WSP for app web
  - Elements deployed to app web using solution package
  - Solution package built into app package as inner WSP

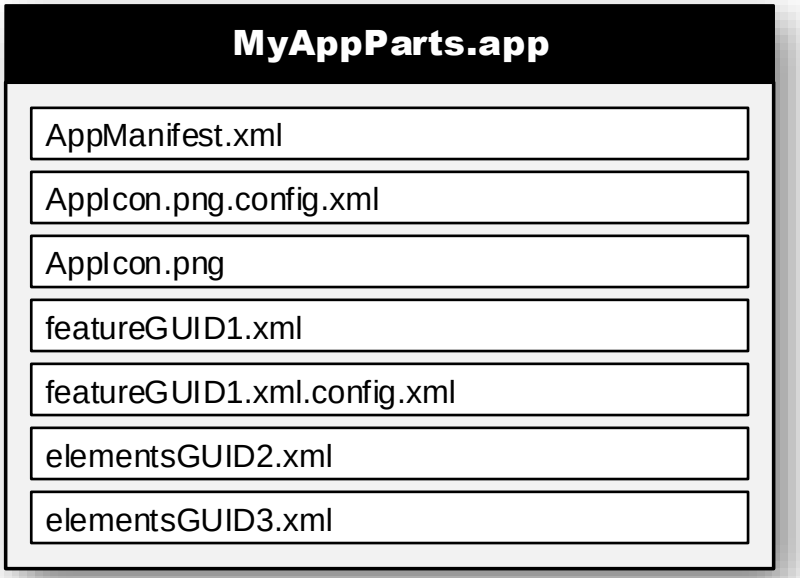


All SharePoint-hosted apps will have an inner WSP in their app package

Cloud-hosted apps will not have an inner WSP in their app package unless they have been implemented to create an app web

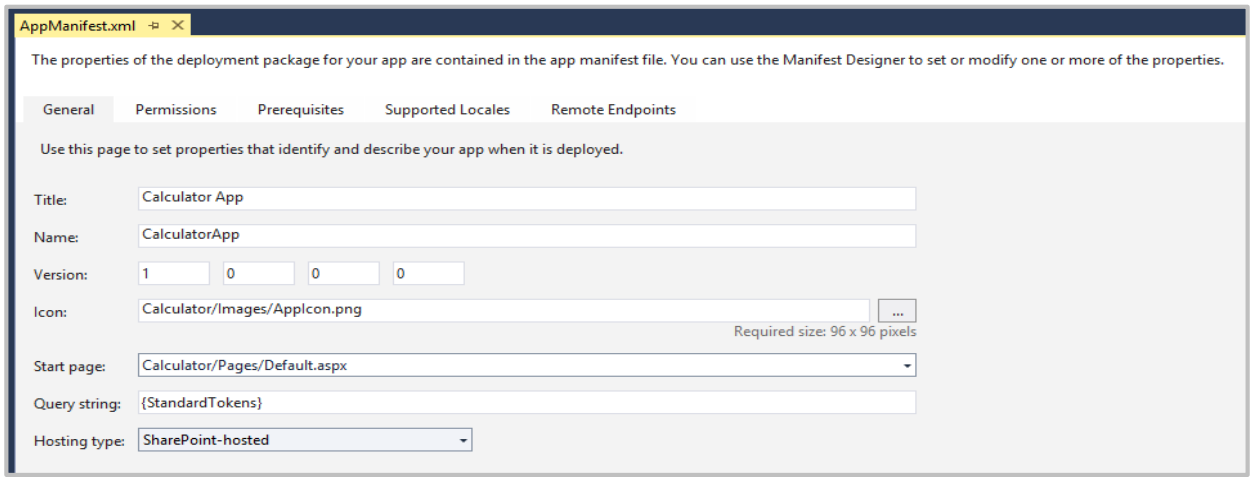
Packaging Host Web Features

- Host web feature elements added at top level
  - `elements.xml` file added for each app part
  - `elements.xml` file added for each UI custom action
  - `features.xml` file added for host web feature
  - Visual Studio adds GUID to file names

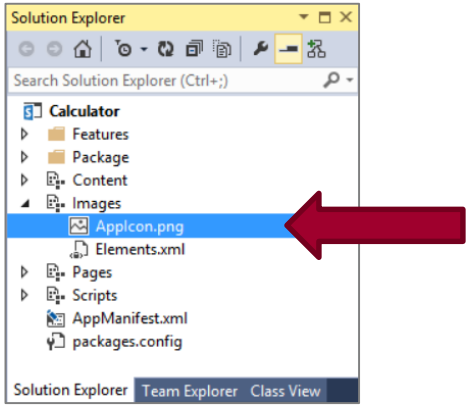


Preparing a Project for Packaging

- Review **AppManifest.xml** and make any required changes

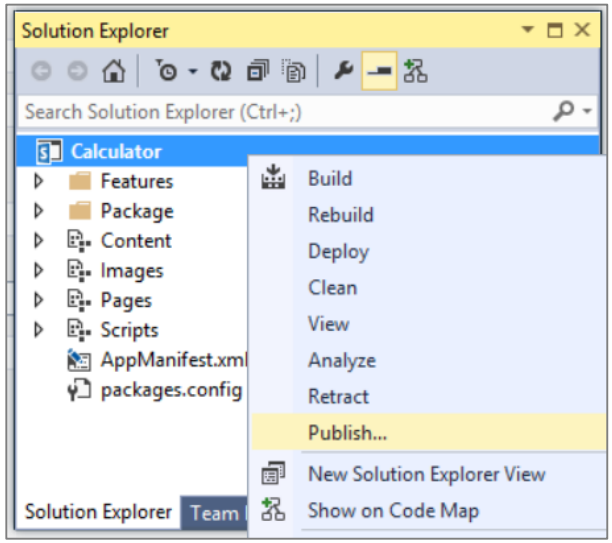


- Add a custom App Icon

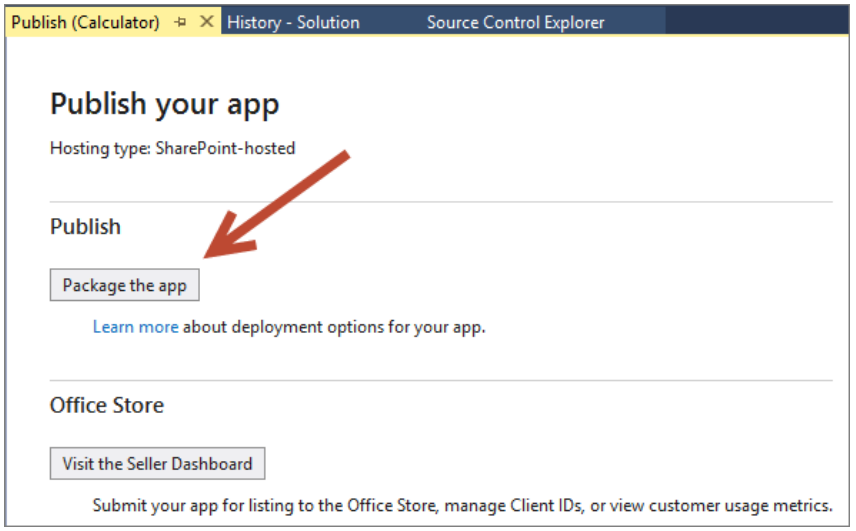


Creating an App Package

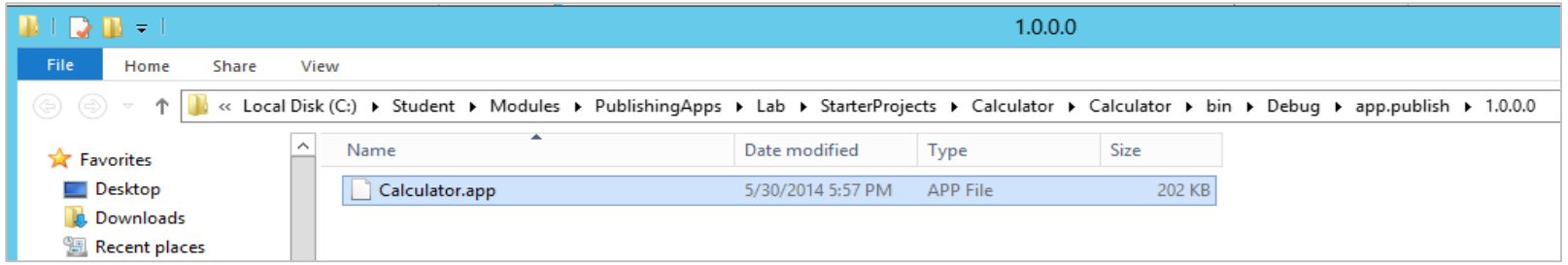
1



2

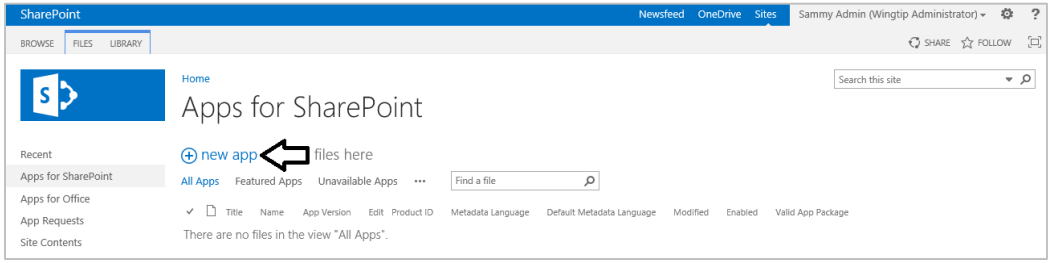


3

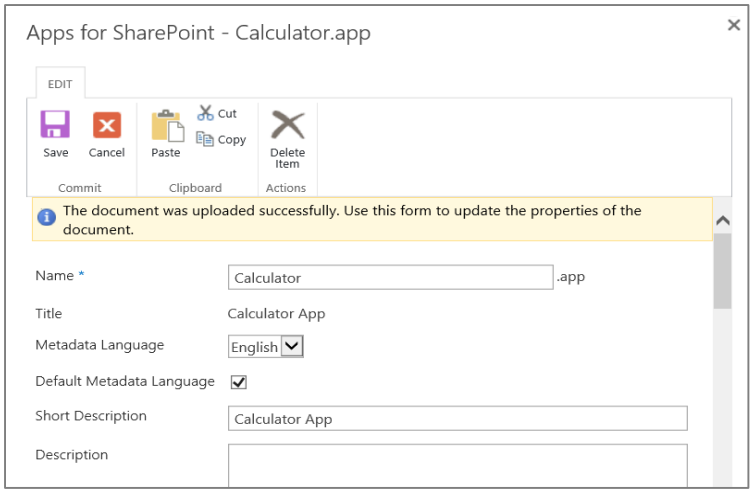


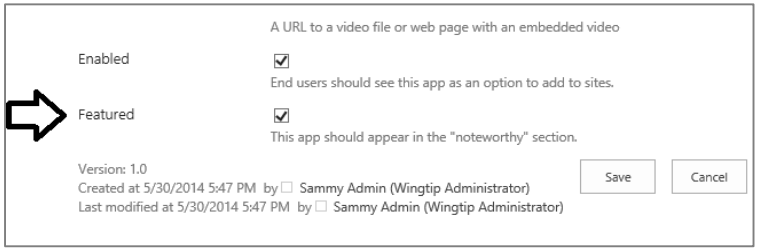
Publishing an App

- Upload app package to Apps for SharePoint library



- Enter app metadata to complete publishing process



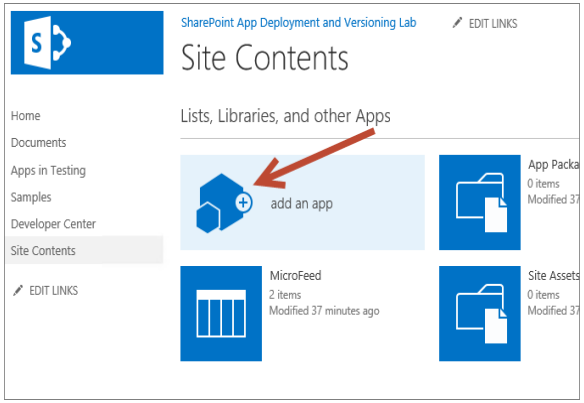


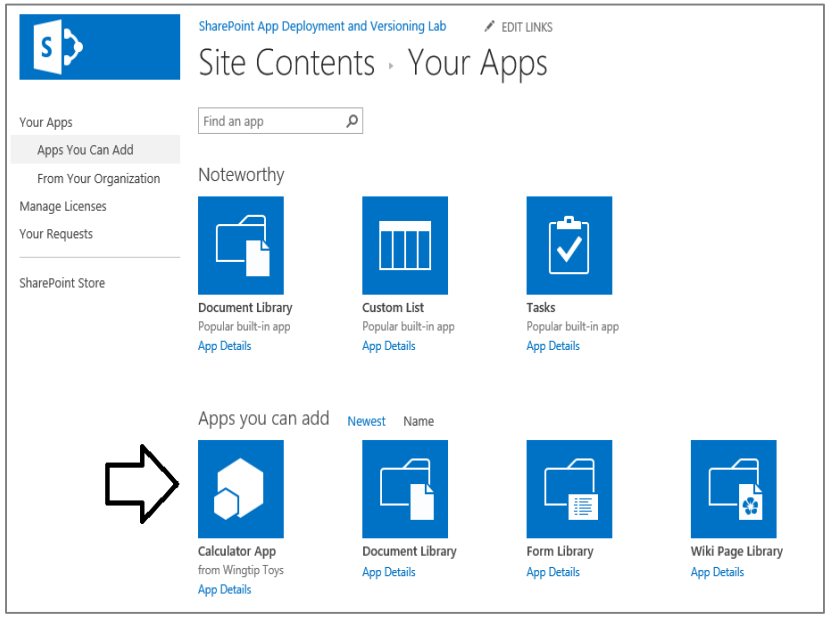
Agenda

- ✓ Creating the App Catalog
- ✓ Publishing Apps in the App Catalog
- ➢ Installing and Upgrading Apps
- • Deploying Provider-hosted Apps

Installing Apps

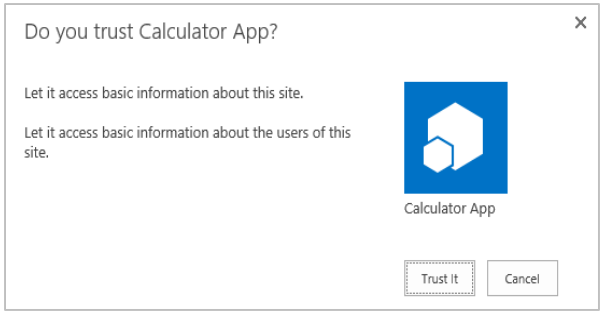
- App installer must be site administrator
  - Click **add an app** link on **Site Contents** page
  - **add an app** link takes you to app discovery page (`addanapp.aspx`)
  - On **Your apps** page, click on app tile to install that app



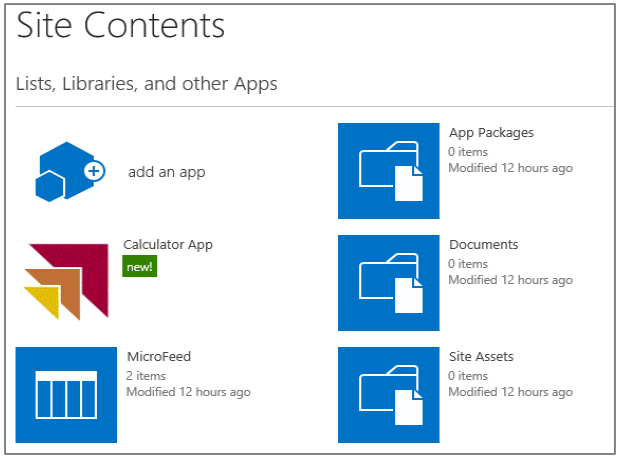


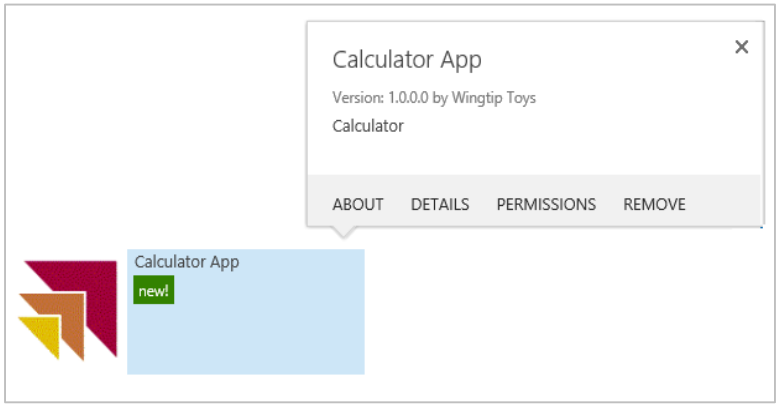
Completing the App installation Process

- App installer is Prompted to Explicitly Trust the app During Installation



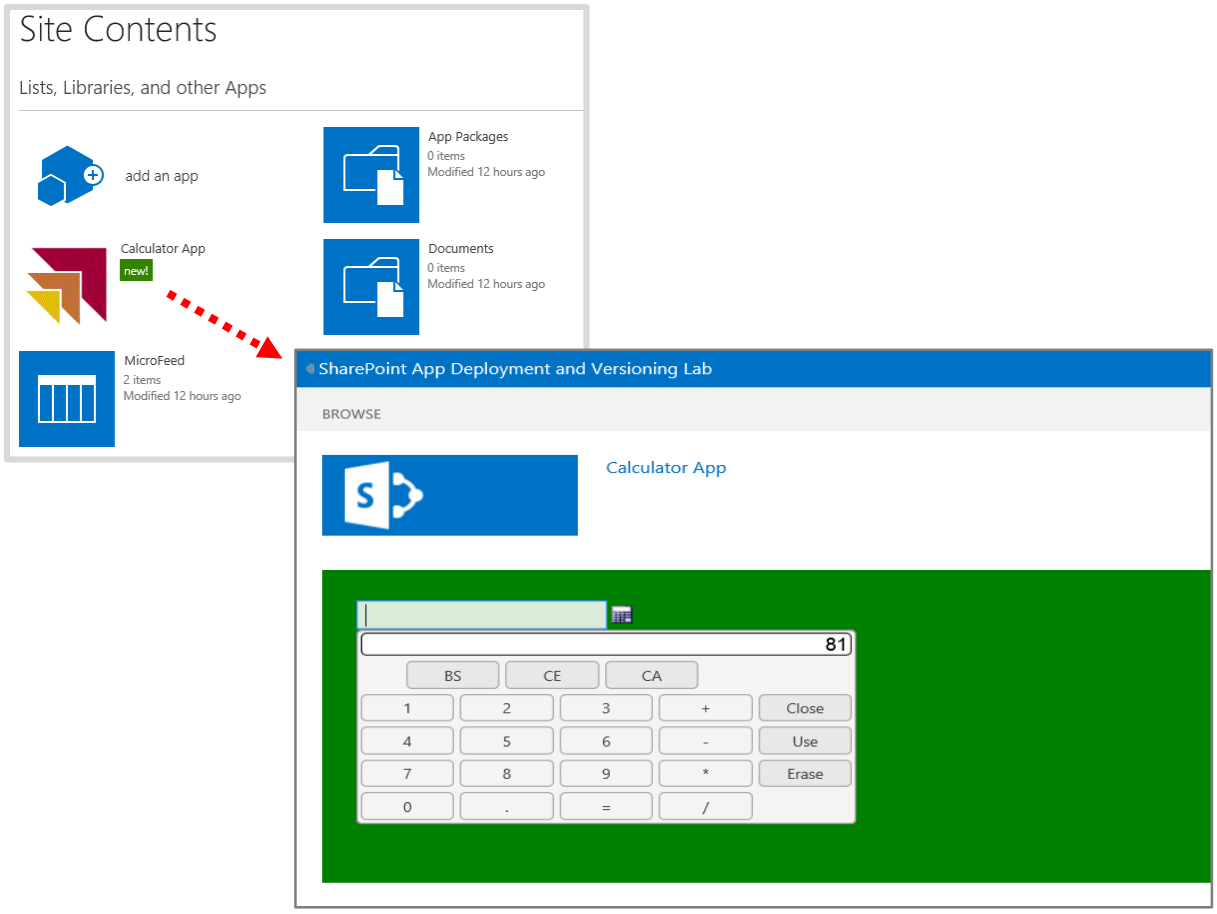
- After installation, app tile added to **Site Contents** page
  app tile provides fly-out menu to assist with app management





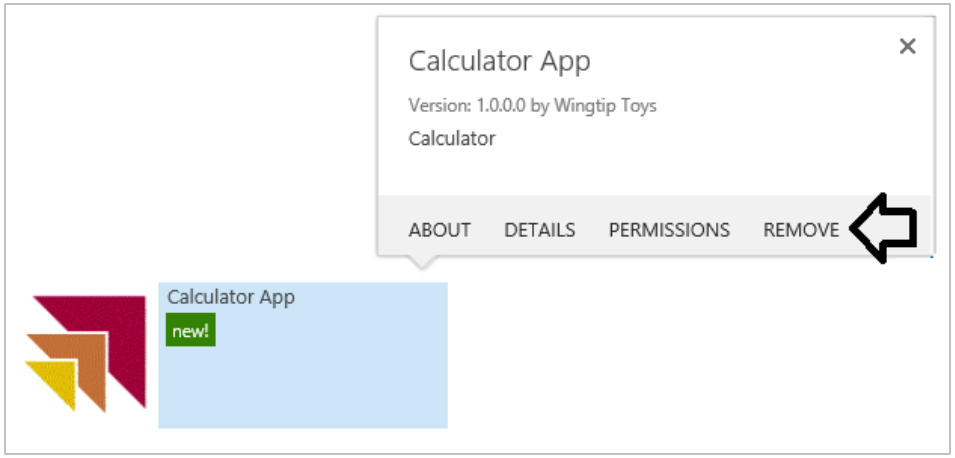
Launching the App After Installation

- Clicking app tile on Site Contents page launches app
  - User is redirected to app start page



Uninstalling an App

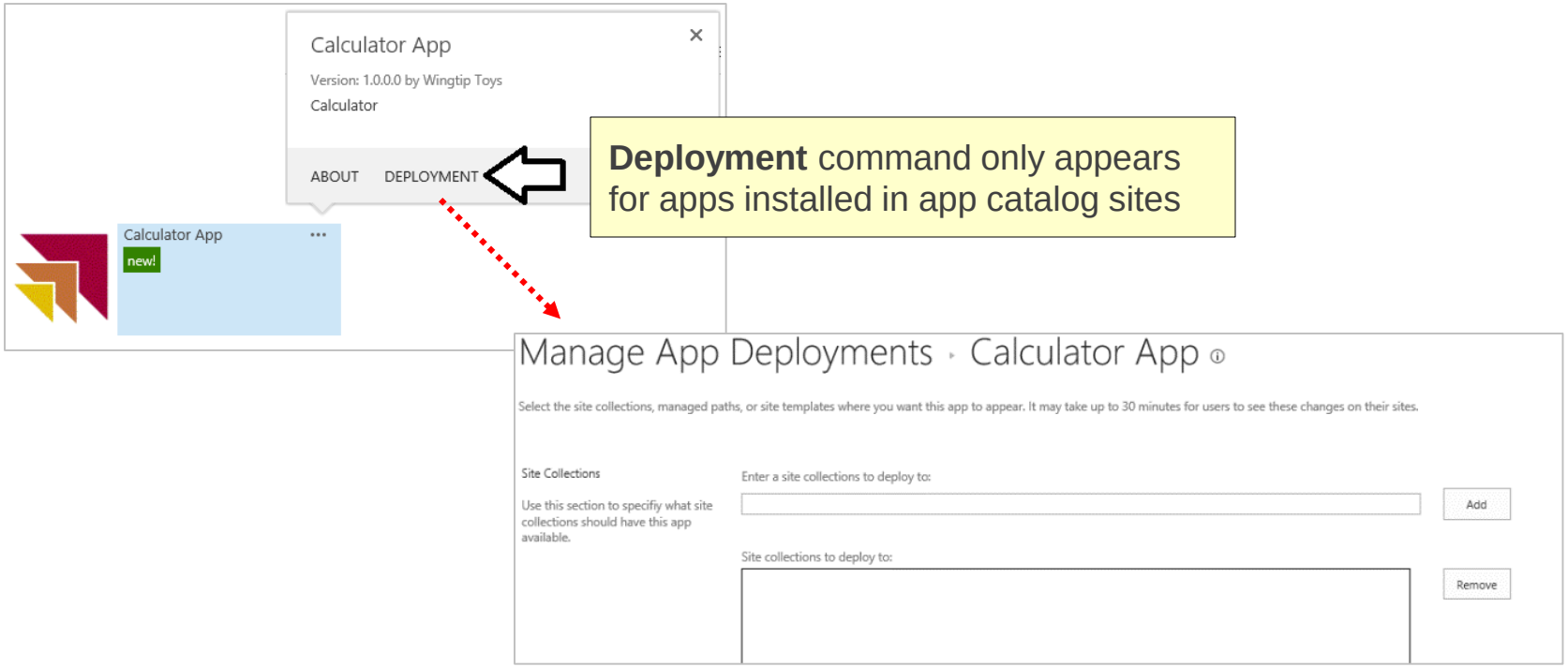
- Use Remove command in app tile flyout menu
  - SharePoint deletes App Web if there is one




DEMO
Publishing and Installing a SharePoint-hosted App

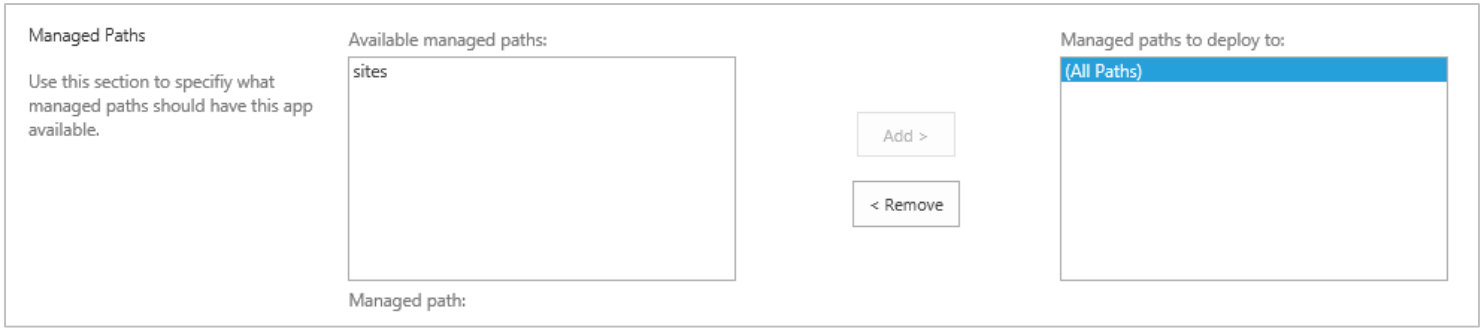
Installing Apps at Tenancy Scope

- App catalog site used for tenancy-scoped install
  - Step 1: Publish app to app catalog site
  - Step 2: Install app in same app catalog site
  - Step 3: Configure app to make it available in other sites

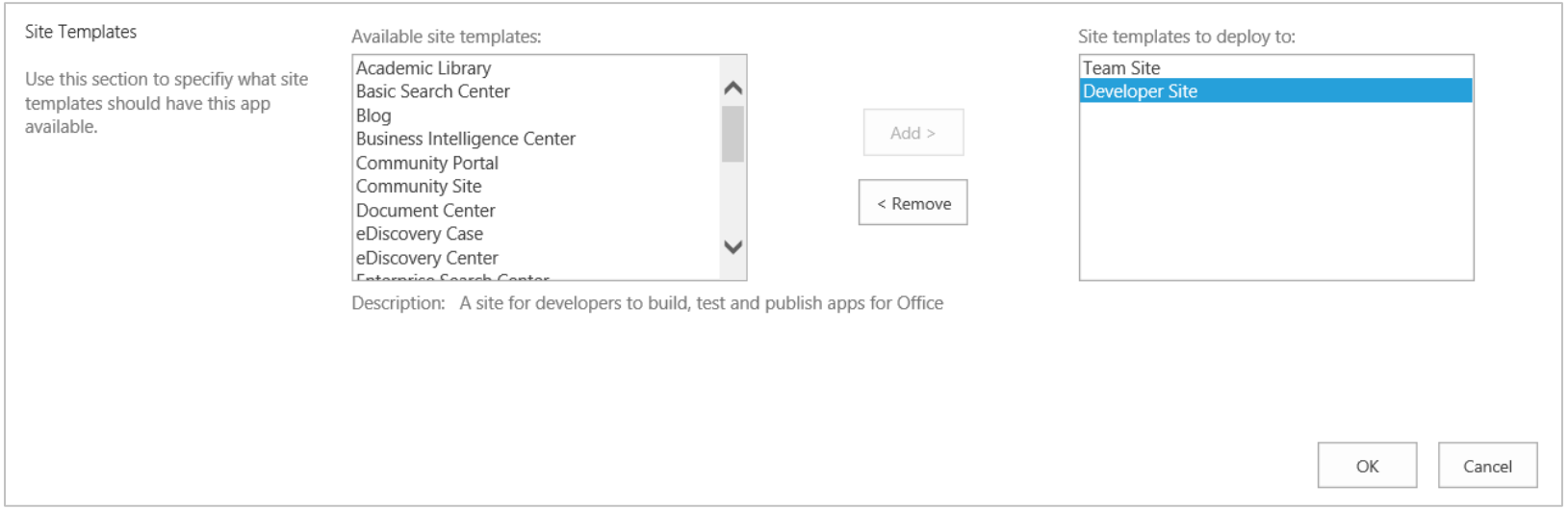


Configuring an App Deployment

- App Deployment can be configured in terms of managed paths…

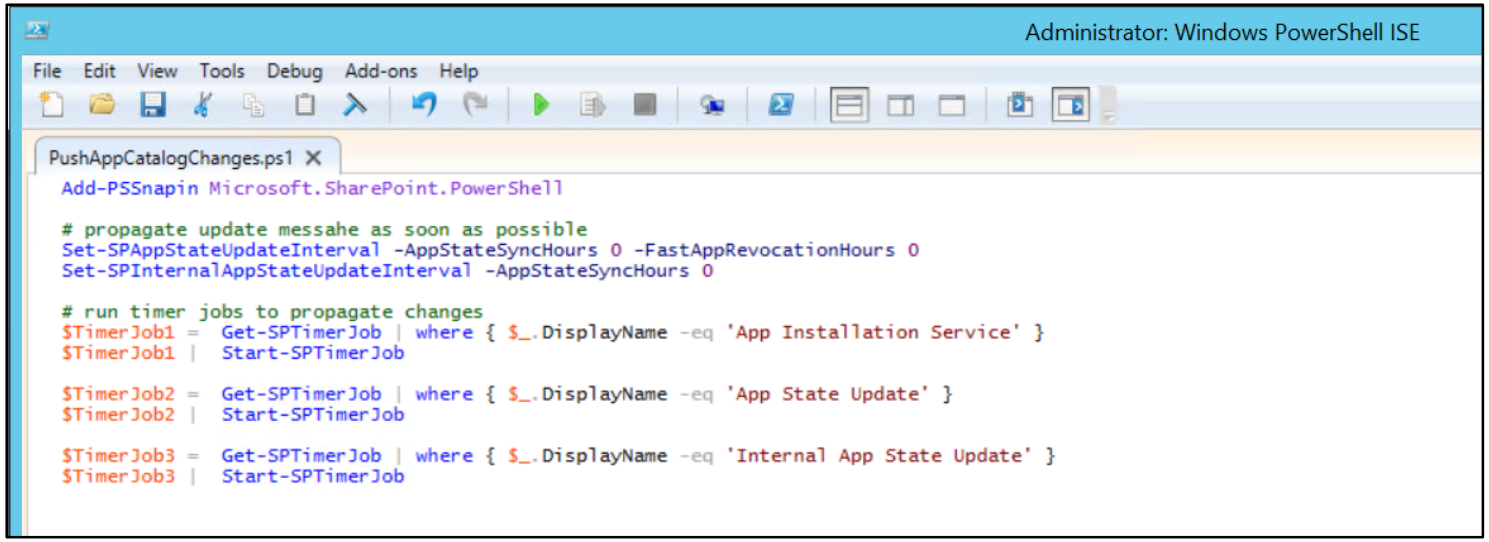


- App Deployment can be configured in terms of Site templates…



Pushing out App Deployment Changes

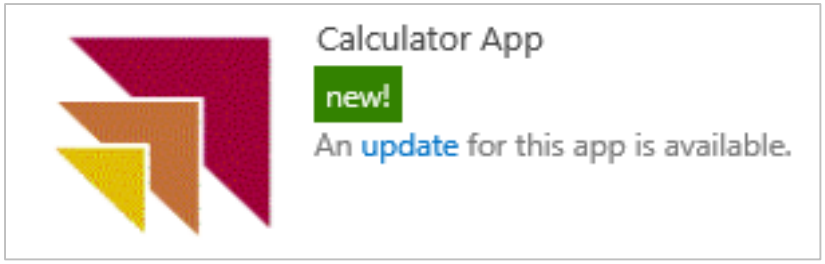
- Deployment changes are not instant
  - Usually requires time job to push changes to all sites
  - Updates can take up to 24 hours to propagate
  - PowerShell script can be used to speed things up for testing




DEMO
Installing an App at Tenancy Scope

Upgrading Apps

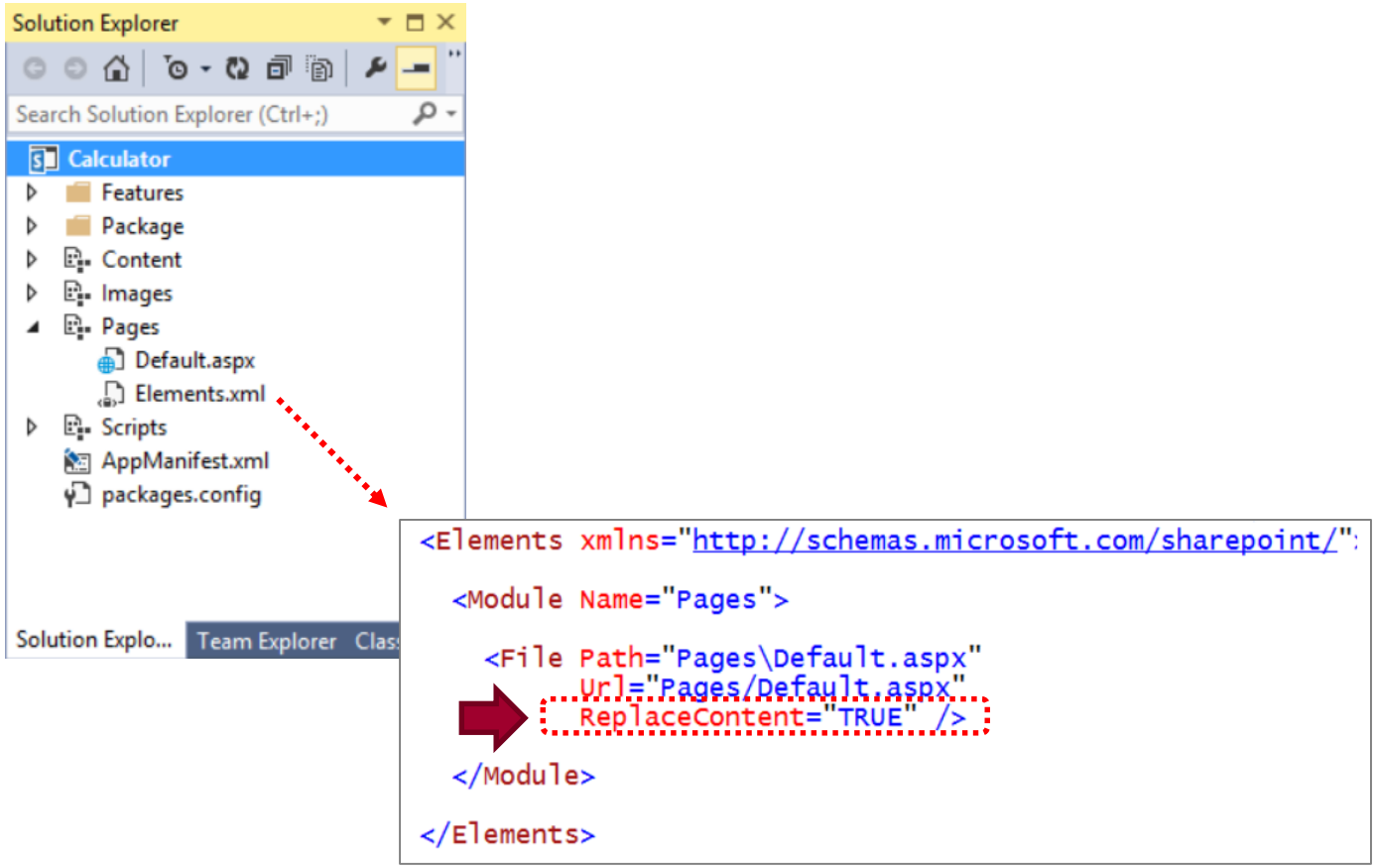
- Overview of SharePoint app upgrade process
  - App catalog tracks current version number of app
  - SharePoint host records app version at install time
  - Updated version of app can be uploaded to app catalog
  - SharePoint host notifies user when there's new version



- User makes choice whether upgrade or not
  - User (app installer) is never forced to upgrade
  - If user decides to upgrade, upgrade involves clicking button

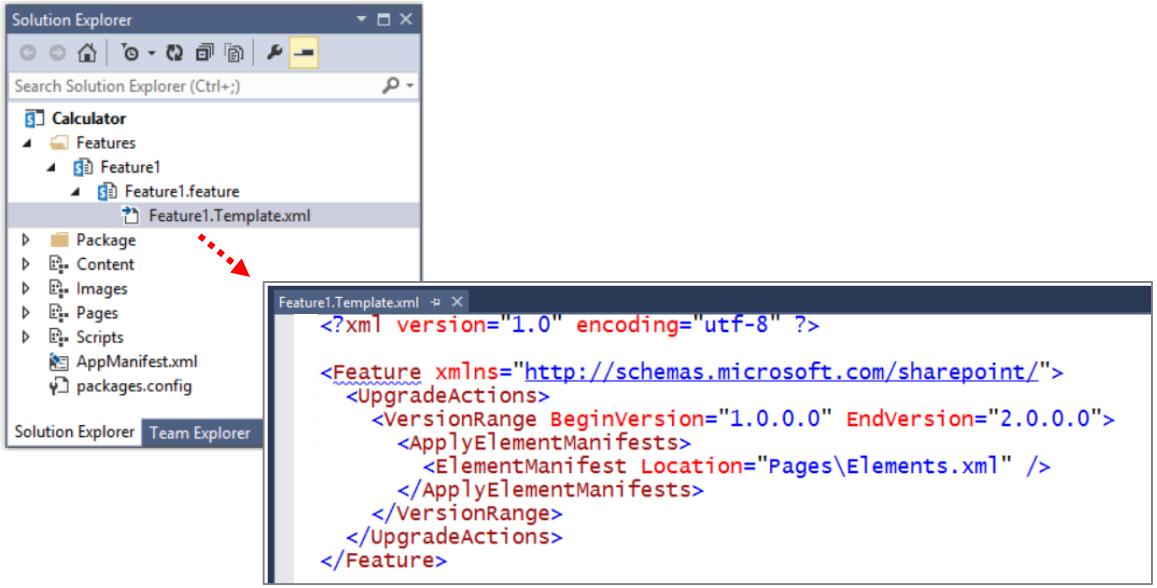
Overwriting Existing Files During Upgrade

- Files to be upgrade required **ReplaceContent** attribute
  - Edit must be made by hand to **Elements.xml** file

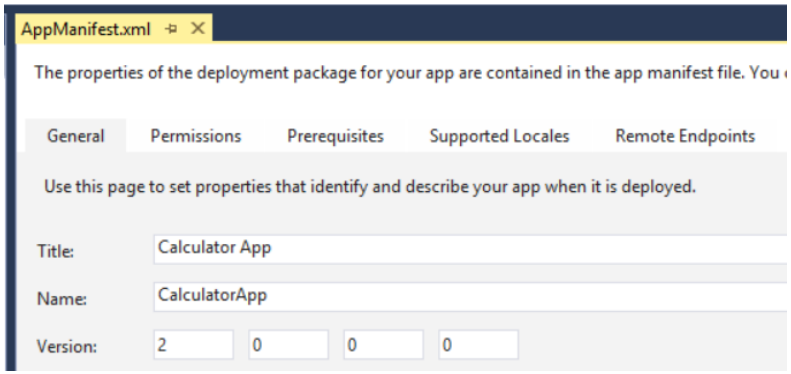


Configuring Feature Upgrade Actions

- Upgrade actions must be entered by hand



- Make sure to update version number in AppManifest.xml




DEMO
Upgrading the Start Page of a SharePoint-hosted App

Agenda

- ✓ Creating the App Catalog
- ✓ Publishing Apps in the App Catalog
- ✓ Installing and Upgrading Apps
- ➢ Deploying Provider-hosted Apps

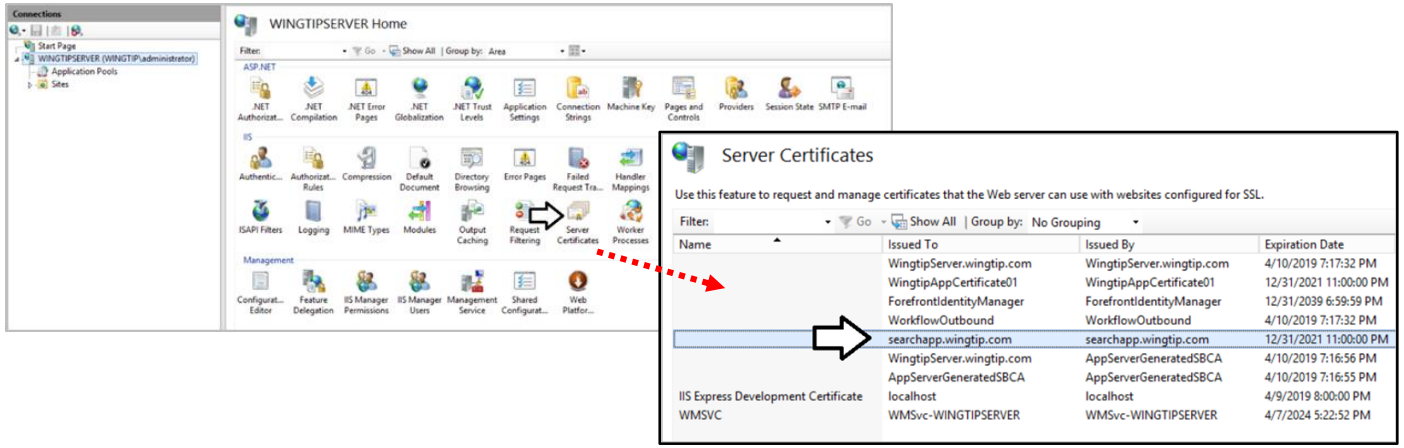
Creating an SSL Certificate

- Test SSL certificate can be created & installed using PowerShell

CreateCertificateForWingtipSearchApp.ps1

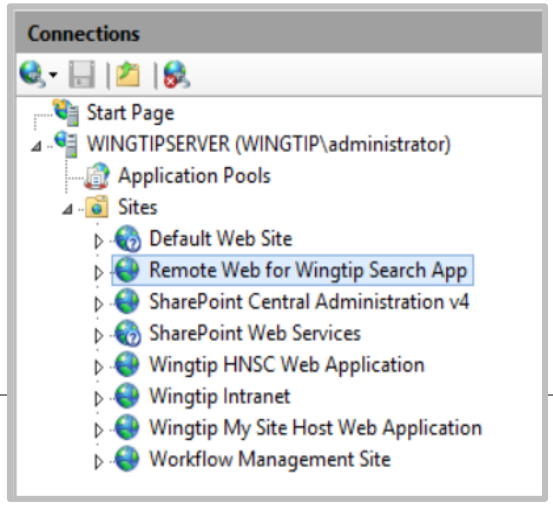
```
$makecert = "C:\Program Files\Microsoft Office Servers\15.0\Tools\makecert.exe"
$certmgr = "C:\Program Files\Microsoft Office Servers\15.0\Tools\certmgr.exe"

# specify subject for SSL certificate
$subject = "searchapp.wingtip.com"

# create output directory to create SSL certificate file
$outputDirectory = "c:\Certs\"
New-Item $outputDirectory -ItemType Directory -Force -Confirm:$false | Out-Null

# create file name for SSL certificate files
$publicCertificatePath  =  $outputDirectory + $subject + ".cer"

Write-Host
Write-Host "Creating .cer certificate file..."
& $makecert -r -pe -n "CN=$subject" -b 01/01/2012 -e 01/01/2022 -eku 1.3.6.1.5.5.7.3.1 -ss my -sr localMachi

Write-Host
Write-Host "Registering certificate with IIS..."
& $certmgr /add $publicCertificatePath /s /r localMachine root
```

- Use IIS Administrative tool to further configure SSL certificate



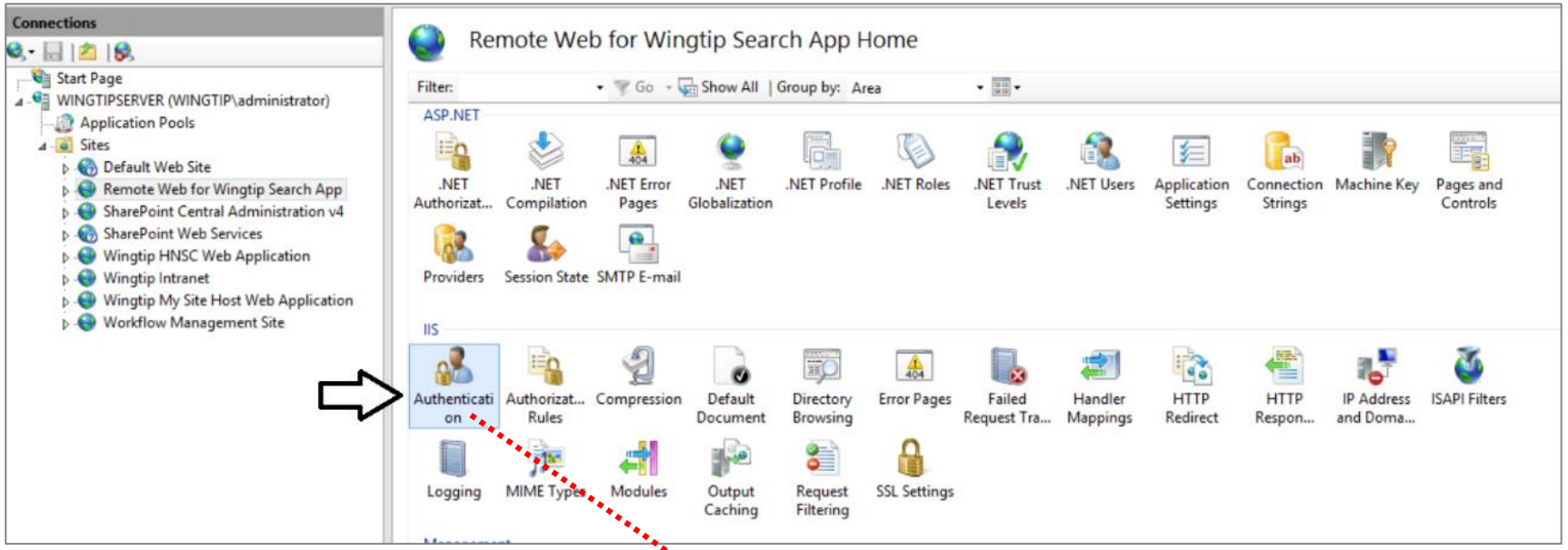
Creating the IIS Website

- Creating an IIS Website
  - Can be done manually
  - Can be done using a PowerShell script




```
$IisWebSiteName = "Remote Web for Wingtip Search App"
$IisWebSitePath = "C:\inetpub\WingtipSearchApp"
$hostHeader     = "searchapp.wingtip.com"
$bindingPort    = 80
$appPoolName    = "DefaultAppPool"
$IisWebSite = Get-Website | Where-Object {$_.Name -eq $IisWebSiteName}

if ($IisWebSite -ne $null) {  Write-Host "    IIS Web site named $IisWebSiteName has already created" -ForegroundColor Gray }
 else {
  #check for existing physical path
  if (Test-Path $IisWebSitePath) {}
  else{
    $silence = New-Item -Path $IisWebSitePath -ItemType Directory
    Write-Host "    Path $IisWebSitePath created" -ForegroundColor Gray
  }
 Write-Host "    .. creating IIS web site: $IisWebSiteName..." -ForegroundColor Gray
 $IisWebSite = New-Website -Name $IisWebSiteName -PhysicalPath $IisWebSitePath -ApplicationPool $appPoolName -HostHeader $hostHeader -Port $bindingPort
 Write-Host "    $IisWebSiteName site created" -ForegroundColor Gray
}

# add entry to HOST file to fix Visual Studio bug
$hostsFilePath = "c:\Windows\System32\Drivers\etc\hosts"
$hostFileEntry = "127.0.0.1     $hostHeader"
Add-Content -Path $hostsFilePath -Value "`r`n$hostFileEntry"
Write-Host "HOST file entry added: $hostFileEntry" -ForegroundColor Gray


Write-Host
Write-Host "Finished! You can now import web applications." -ForegroundColor Green
```

Configuring Windows Authentication

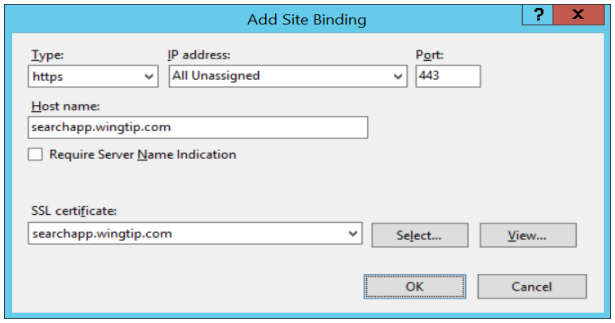


Name	Status	Response Type
Anonymous Authentication	Enabled	
ASP.NET Impersonation	Disabled	
Basic Authentication	Disabled	HTTP 401 Challenge
Digest Authentication	Disabled	HTTP 401 Challenge
Forms Authentication	Disabled	HTTP 302 Login/Redirect
Windows Authentication	Enabled	HTTP 401 Challenge

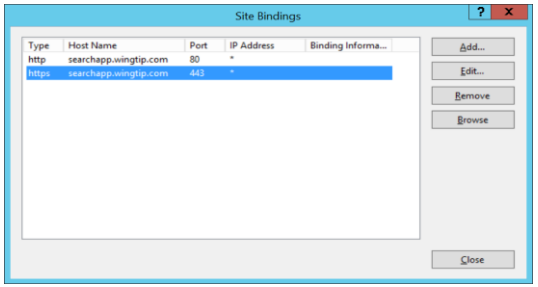
Configuring SSL for the Remote Web

- Editing bindings to the associated IIS Website

- Create a new HTTPS binding and associate with SSL certificate



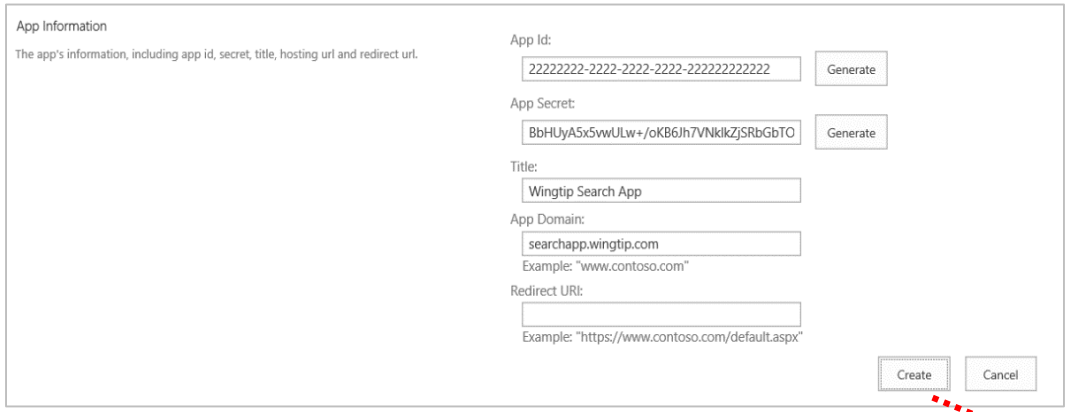
- Binding listens for requests targeting **http://searchapp.wingtip.com**

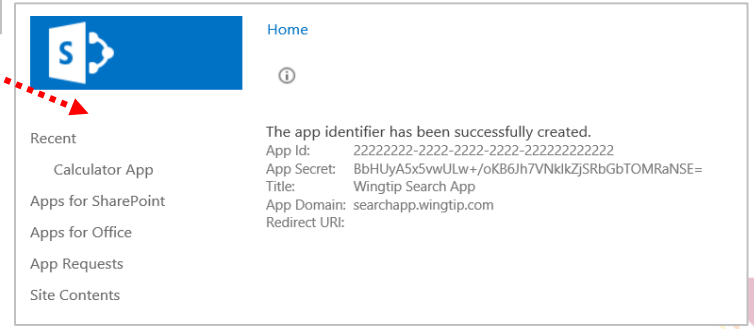



DEMO
Creating and Configuring an IIS Website to Host the Remote Web

Registering the App Security Principal

- Can be done manually using AppRegNew.aspx page
  - Enter the **App ID** (aka Client ID)
  - Enter an **App Secret** (even in S2S scenario when it is not used)
  - Enter a **Title**
  - Enter an **App Domain**





Registering App Principal with PowerShell

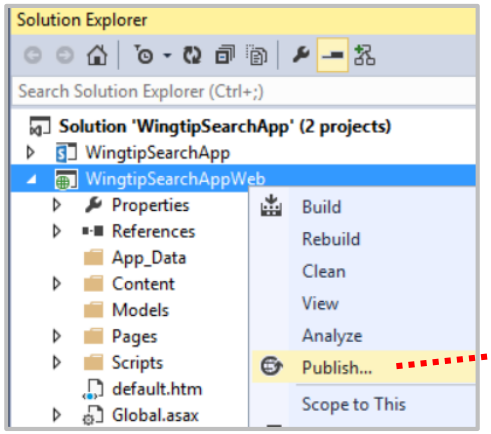
- Tow ways to do this
  - The easy way using Register-SPAppPrincipal
  - The more flexible way using SPAppPrincipalManager

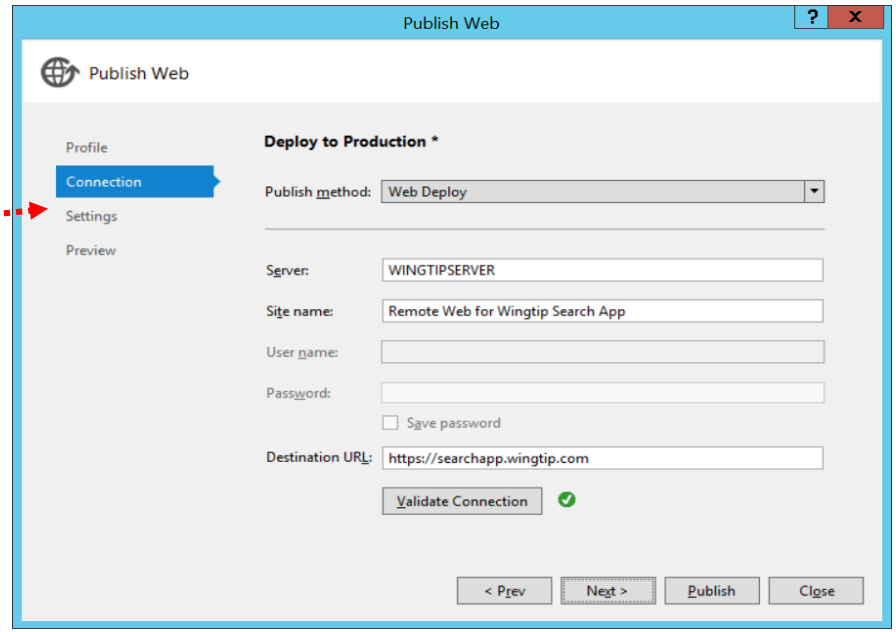
```
Add-PSSnapin "Microsoft.SharePoint.PowerShell"

# set intialization values for new app principal
$appDisplayName = "My First S2S App"
$clientID = "22222222-2222-2222-2222-222222222222"

$targetSiteUrl = "http://wingtipserver"
$targetSite = Get-SPSite $targetSiteUrl
$realm = Get-SPAuthenticationRealm -ServiceContext $targetSite
$fullAppPrincipalIdentifier = $clientID + '@' + $realm


Write-Host "Register new app principal"
$registeredAppPrincipal = Register-SPAppPrincipal -NameIdentifier $fullAppPrincipalIdentifier
                                                  -Site $targetSite.RootWeb
                                                  -DisplayName $AppDisplayName

$registeredAppPrincipal | select * | Format-List
$registeredAppPrincipal | select * | Format-List | Out-File -FilePath "SecurityPrincipal01.txt"
```


DEMO
Registering the App Security Principal for a Provider-hosted App

Deploying the App's Remote Web Project

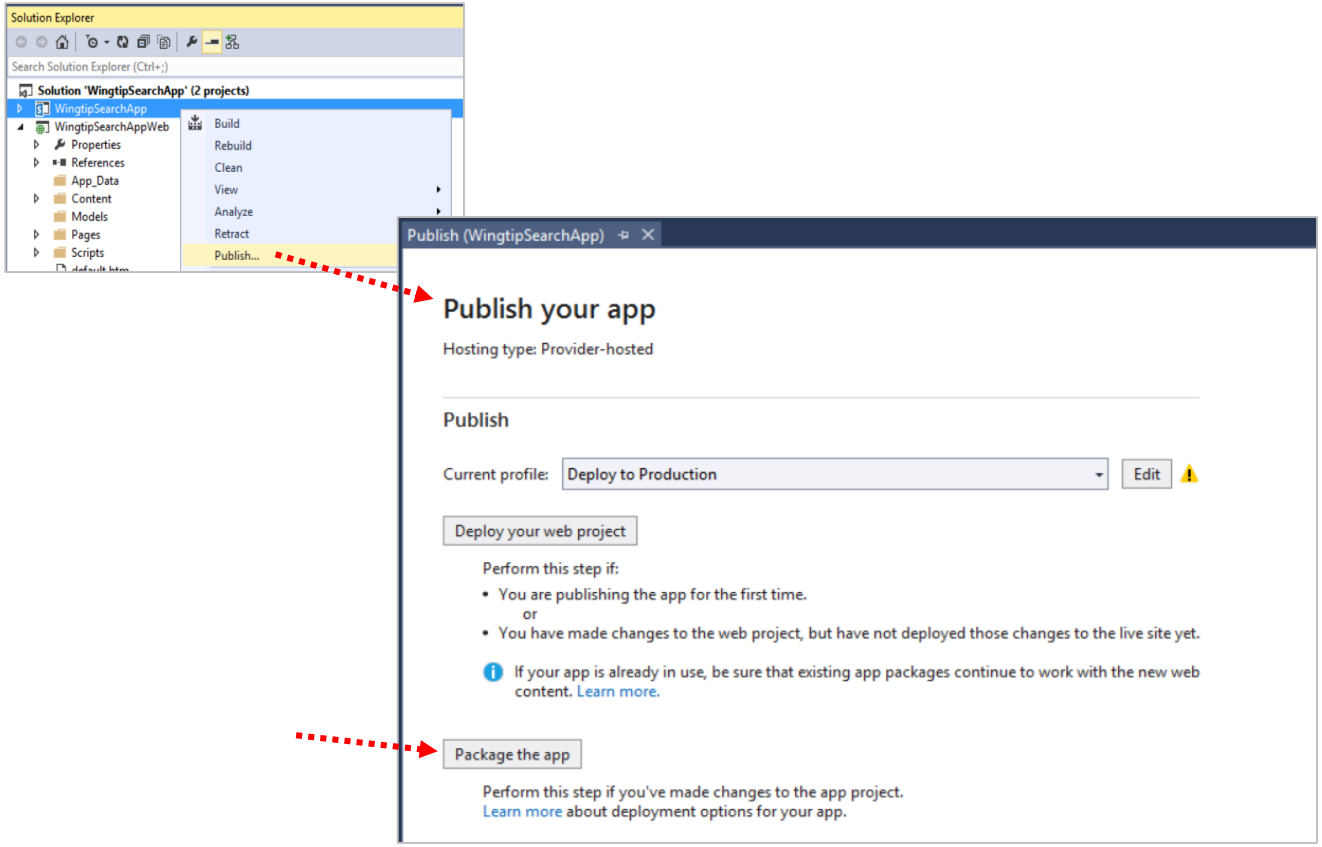
- Visual Studio provides Publish Web wizard
  - Allows you to create one or more deployment configurations
  - Automatically pushes Web Project files into IIS website





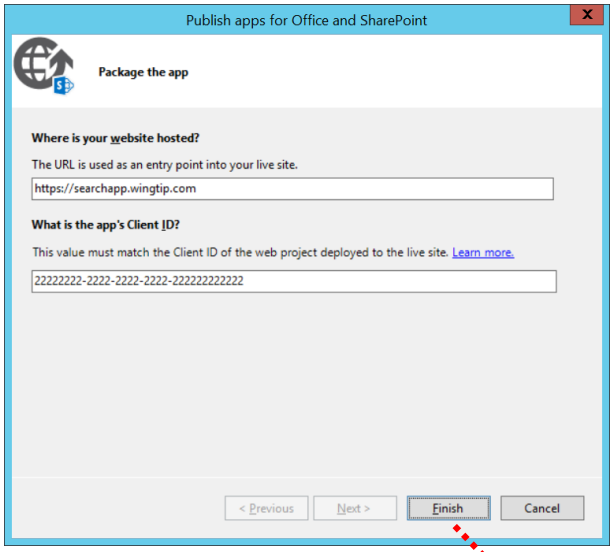
Creating the Provider-hosted App Package

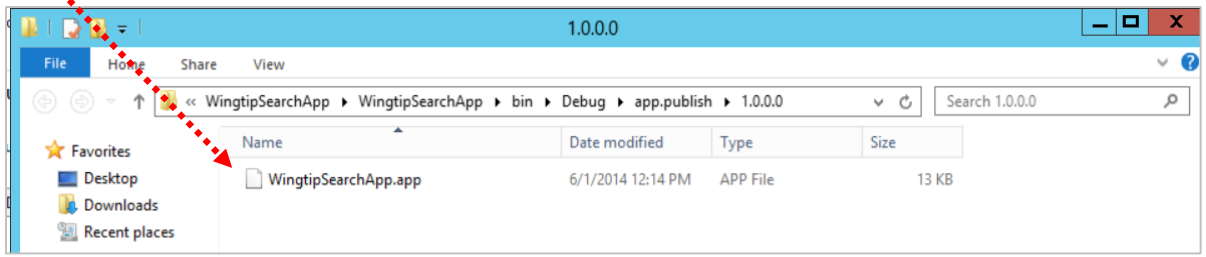
- Visual Studio provides Publish your app page
  - Click **Package the app** to begin packaging process



Enter Information for the App Package

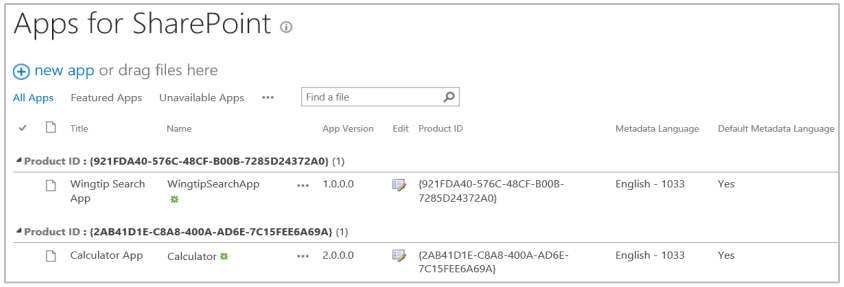
- Provide-hosted App must be published with…
  - SSL-based URL where the remote web has been deployed
  - The App ID (aka Client ID)



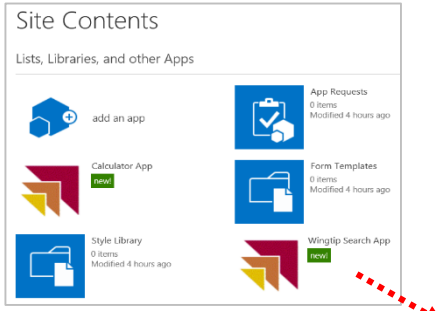


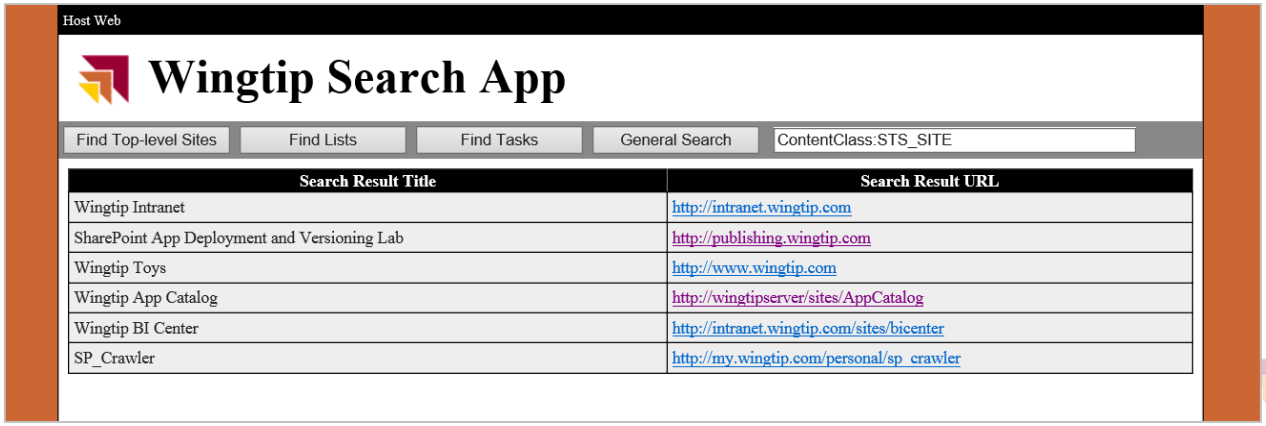
Publishing and Installing the App

- Publish the app package *(just like with a SharePoint-hosted app)*



- Install and launch






DEMO
Deploying a Provider-hosted App in an On-premises Farm

Summary

- ✓ Creating the App Catalog
- ✓ Publishing Apps in the App Catalog
- ✓ Installing and Upgrading Apps
- ✓ Deploying Provider-hosted Apps