Create, Configure And Deploy Provider Hosted Add-In For SharePoint Online - Part One

Priyaranjan K S   Sep 27 2016   32.2k   3   4

In this article you will learn how to create, configure and Deploy Provider Hosted Add-in for SharePoint Online.

SharePoint provides two types of add-ins (Previously Apps)

- *SharePoint Hosted Add-ins*
  Used to deploy SharePoint resources like pages, lists, workflows, custom content types, list templates, Web Parts and so on. It cannot contain custom server side object model code but supports REST and JSOM.

- *Provider hosted Add-ins*
  Used to deploy components outside the SharePoint Farm. It can include a web application, web service or database that is deployed outside SharePoint.

In this article series we will see how to deploy a provider hosted add-in to SharePoint Online in Office 365. Here the Website will be hosted in Azure and the App will be installed in SharePoint Online. Thus the core logic will be abstracted away from SharePoint farm. Upon opening the app in SharePoint Online, the redirection happens to the Website hosted in Azure. The article will be divided into 2 parts.

- Create, Configure and Deploy remote add-in web to Azure
- Deploy add-in to SharePoint online

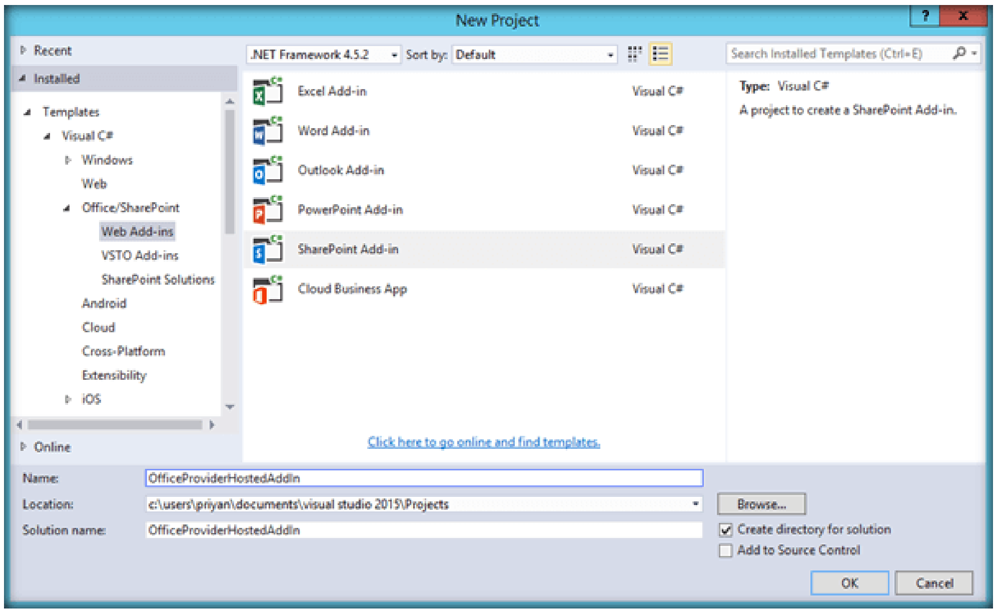
Prerequisites

Ensure that you have the below subscription to get started with the development

- *Azure Subscription* - The web site will be hosted in Azure. So ensure that we have an active Azure Subscription.
- *Office 365 SharePoint Online Subscription* - We will need a developer Site to which the app will be deployed

Create, Configure and Deploy remote add-in web to Azure

Spin up Visual Studio and Select 'SharePoint Add-in' template.


New Project

.NET Framework 4.5.2   Sort by: Default   Search Installed Templates (Ctrl+E)

Recent
Installed
Templates
Visual C#
Windows
Web
Office/SharePoint
Web Add-ins
VSTO Add-ins
SharePoint Solutions
Android
Cloud
Cross-Platform
Extensibility
iOS
Online

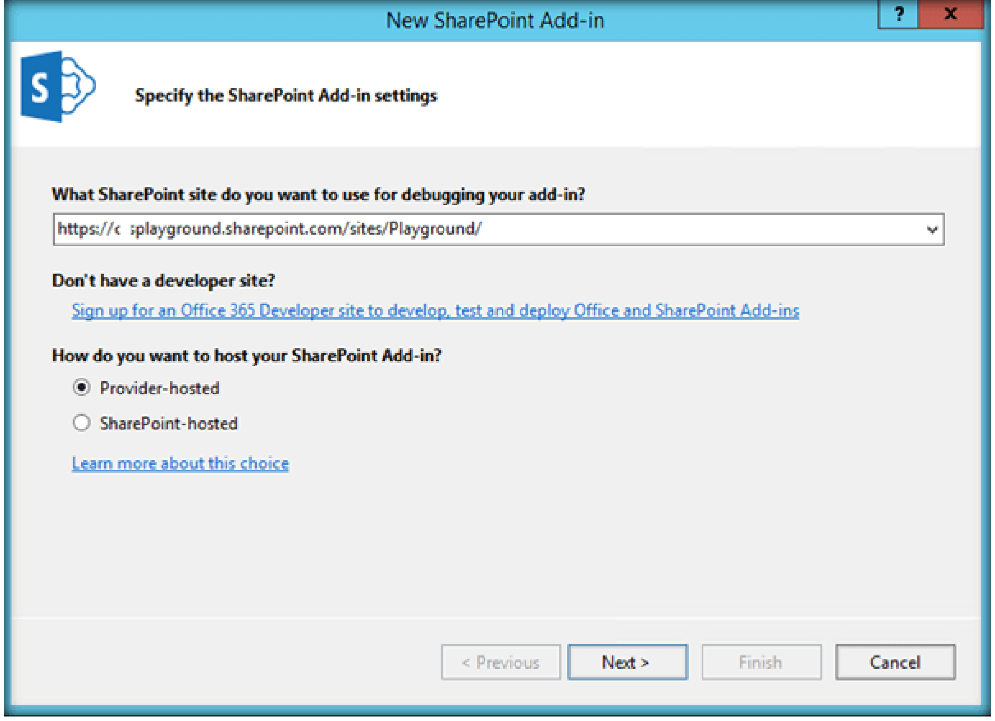
Excel Add-in	Visual C#
Word Add-in	Visual C#
Outlook Add-in	Visual C#
PowerPoint Add-in	Visual C#
SharePoint Add-in	Visual C#
Cloud Business App	Visual C#

Type: Visual C#
A project to create a SharePoint Add-in.

Click here to go online and find templates.

Name: OfficeProviderHostedAddIn
Location: c:\users\priyan\documents\visual studio 2015\Projects   Browse...
Solution name: OfficeProviderHostedAddIn
[✓] Create directory for solution
[ ] Add to Source Control
OK   Cancel

Specify the SharePoint Online Developer Site URL and select Provider-hosted option. Click on Next.


New SharePoint Add-in

Specify the SharePoint Add-in settings

What SharePoint site do you want to use for debugging your add-in?

https://c splayground.sharepoint.com/sites/Playground/

Don't have a developer site?

Sign up for an Office 365 Developer site to develop, test and deploy Office and SharePoint Add-ins

How do you want to host your SharePoint Add-in?

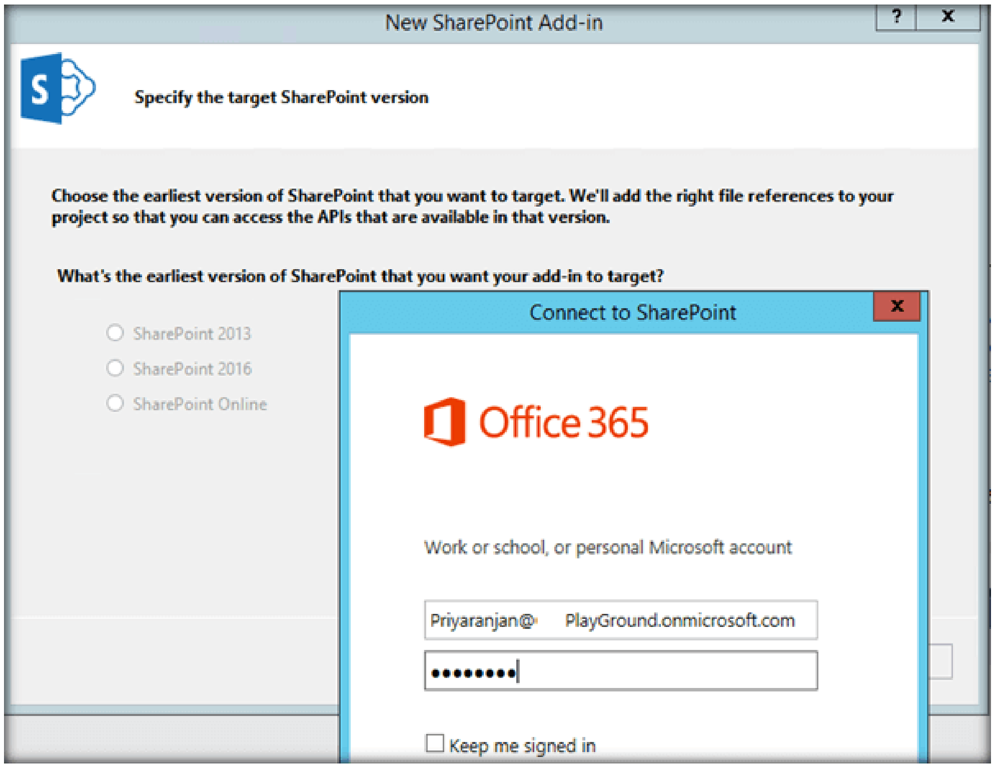
[✓ Provider-hosted]

SharePoint-hosted

Learn more about this choice

< Previous | Next > | Finish | Cancel

This will open up the Office 365 authentication pop up. Enter the Office 365 credentials.


New SharePoint Add-in

Specify the target SharePoint version

Choose the earliest version of SharePoint that you want to target. We'll add the right file references to your project so that you can access the APIs that are available in that version.

What's the earliest version of SharePoint that you want your add-in to target?

SharePoint 2013

SharePoint 2016

SharePoint Online

Connect to SharePoint

Office 365

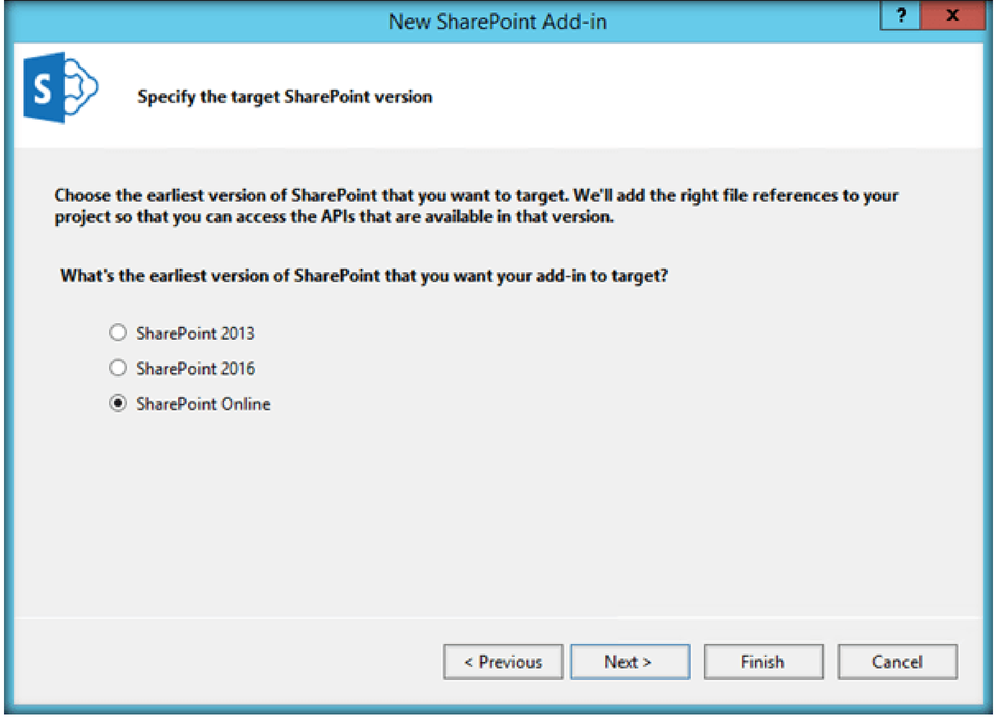
Work or school, or personal Microsoft account

Priyaranjan@ PlayGround.onmicrosoft.com

••••••••

Keep me signed in

It will automatically select 'SharePoint Online' version. If not select it manually.


New SharePoint Add-in

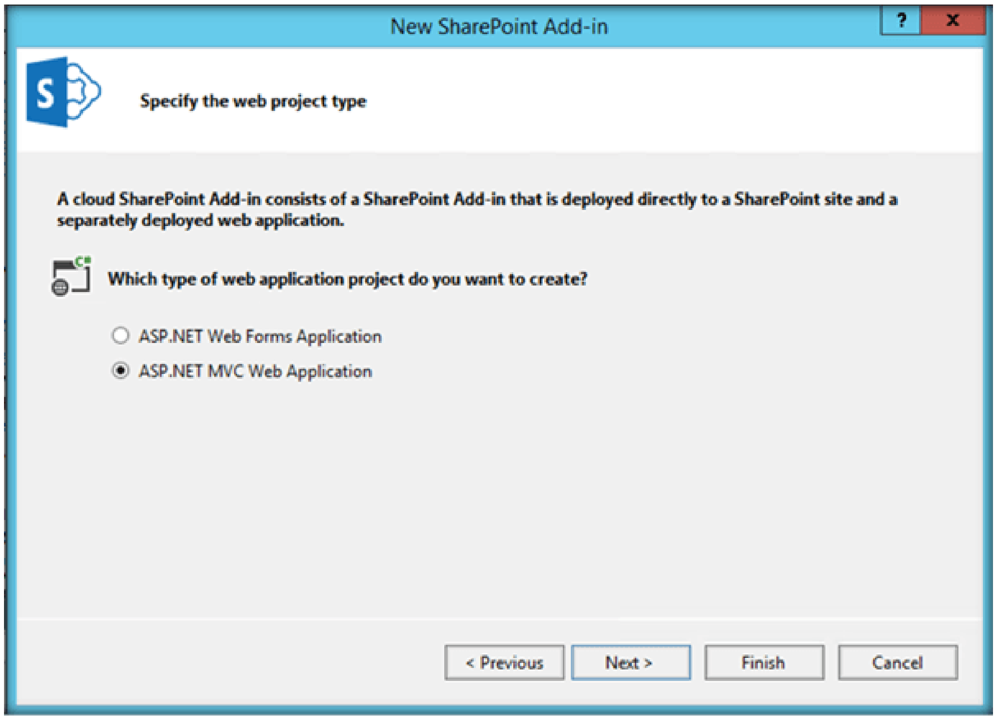
Specify the target SharePoint version

Choose the earliest version of SharePoint that you want to target. We'll add the right file references to your project so that you can access the APIs that are available in that version.

What's the earliest version of SharePoint that you want your add-in to target?

- ○ SharePoint 2013
- ○ SharePoint 2016
- ● SharePoint Online

You can either proceed with web Forms or MVC Web Application. Here I am moving ahead with MVC.


New SharePoint Add-in

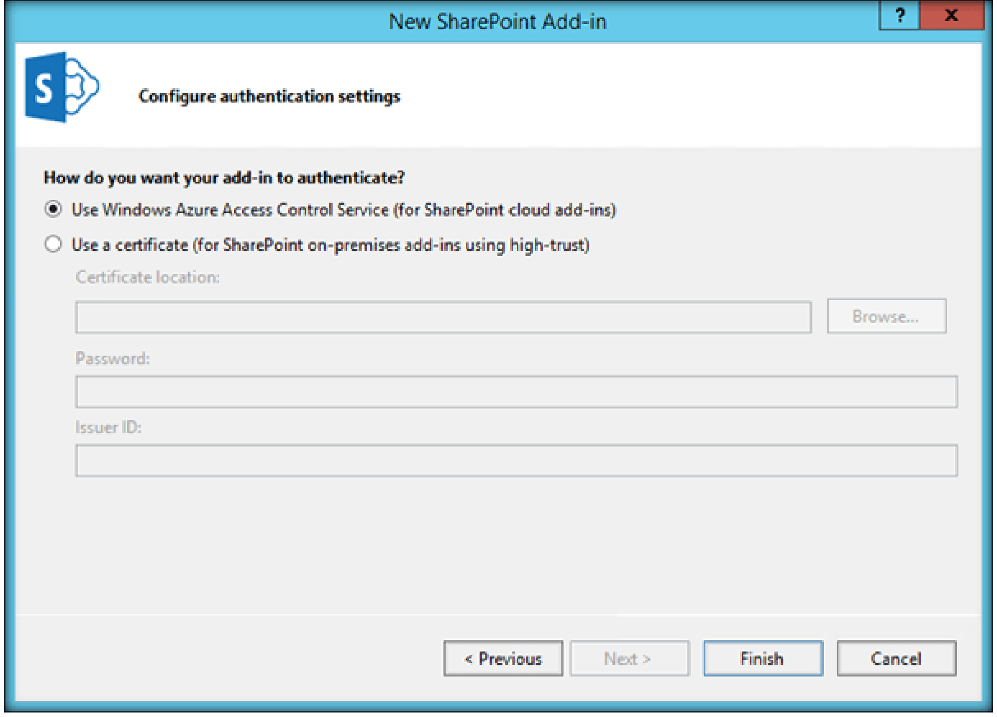
Specify the web project type

A cloud SharePoint Add-in consists of a SharePoint Add-in that is deployed directly to a SharePoint site and a separately deployed web application.

Which type of web application project do you want to create?

- ○ ASP.NET Web Forms Application
- ● ASP.NET MVC Web Application

Since we are deploying the Web to Azure, we will be making use of Azure Access Control Services (Known as Low Trust Authentication). If we were deploying the web to an on premise IIS server we could have used a signed certificate (known as High Trust).


New SharePoint Add-in

Configure authentication settings

How do you want your add-in to authenticate?

[✓ Use Windows Azure Access Control Service (for SharePoint cloud add-ins)]

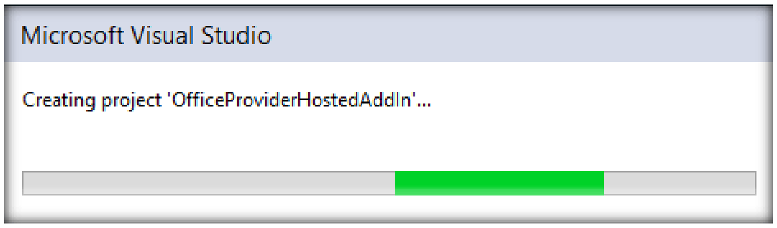
Use a certificate (for SharePoint on-premises add-ins using high-trust)

Certificate location:

Password:

Issuer ID:

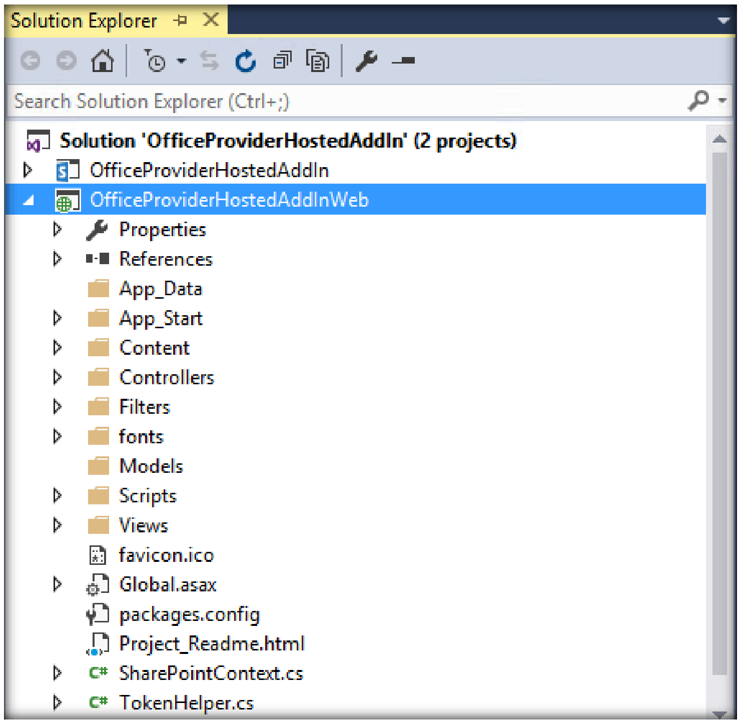
Click on Finish. This will create the Project structure for us.


Microsoft Visual Studio

Creating project 'OfficeProviderHostedAddIn'...

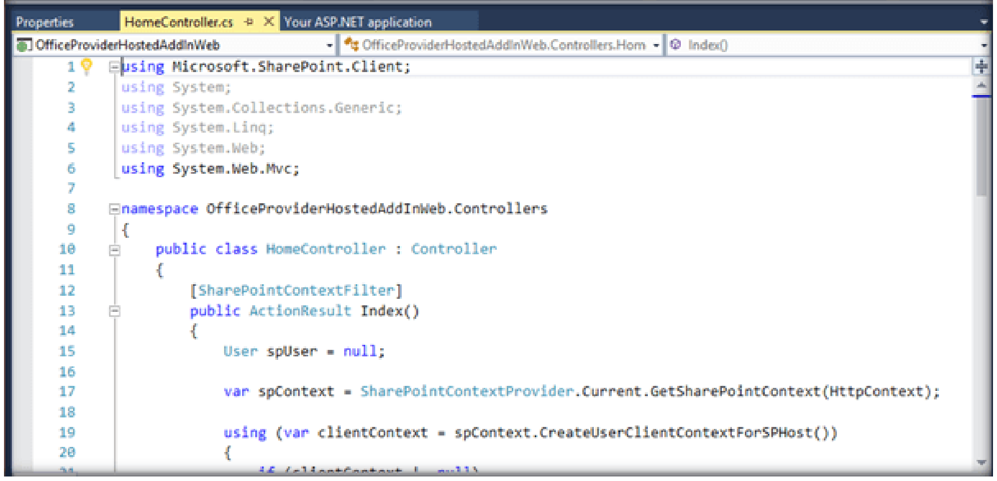
Thus you can see that there are two sections in the created solution structure below,

- *OfficeProviderHostedAdd-in* - it will be deployed to the SharePoint Online Site.
- *OfficeProviderHostedAdd-inWeb* - it will be deployed to Azure.



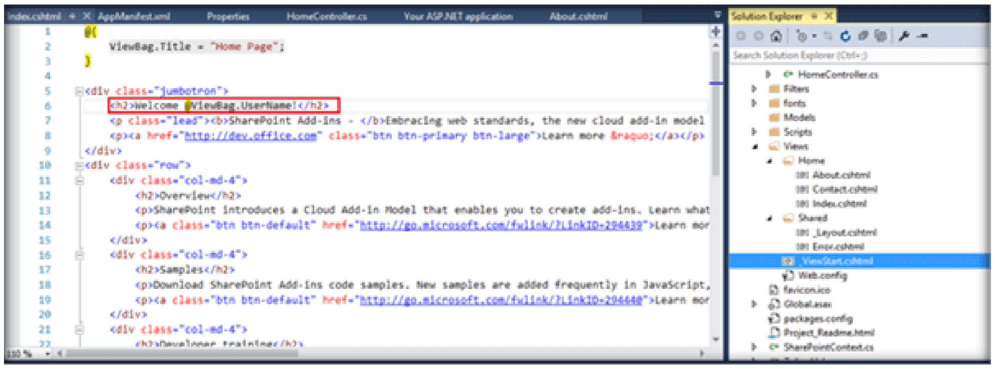
```
Solution 'OfficeProviderHostedAddIn' (2 projects)
  OfficeProviderHostedAddIn
  OfficeProviderHostedAddInWeb
    Properties
    References
    App_Data
    App_Start
    Content
    Controllers
    Filters
    fonts
    Models
    Scripts
    Views
    favicon.ico
    Global.asax
    packages.config
    Project_Readme.html
    SharePointContext.cs
    TokenHelper.cs
```

If we expand the Controller folder, we can see the controller files.

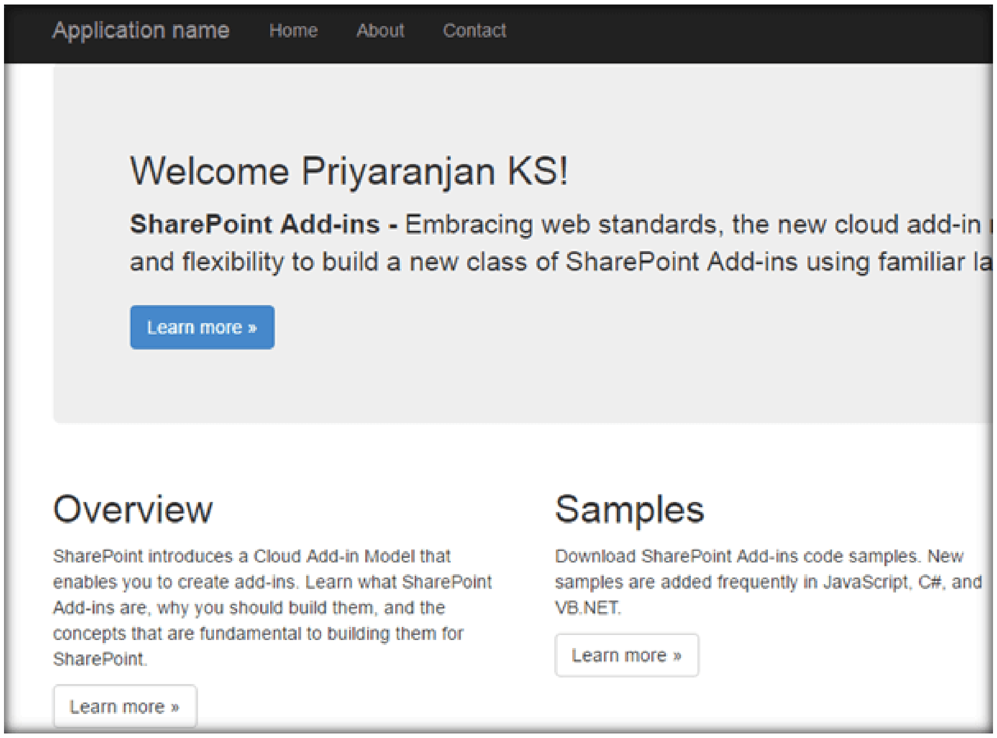


```
using Microsoft.SharePoint.Client;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace OfficeProviderHostedAddInWeb.Controllers
{
    public class HomeController : Controller
    {
        [SharePointContextFilter]
        public ActionResult Index()
        {
            User spUser = null;

            var spContext = SharePointContextProvider.Current.GetSharePointContext(HttpContext);

            using (var clientContext = spContext.CreateUserClientContextForSPHost())
            {
```

Within the Views Folder we can see the View file. We will be getting the current user name in the controller and will be displaying it using the view as shown below.



```
@{
    ViewBag.Title = "Home Page";
}

<div class="jumbotron">
    <h2>Welcome @ViewBag.UserName!</h2>
    <p class="lead"><b>SharePoint Add-ins - </b>Embracing web standards, the new cloud add-in model
    <p><a href="http://dev.office.com" class="btn btn-primary btn-large">Learn more &raquo;</a></p>
</div>
<div class="row">
    <div class="col-md-4">
        <h2>Overview</h2>
        <p>SharePoint introduces a Cloud Add-in Model that enables you to create add-ins. Learn what
        <p><a class="btn btn-default" href="http://go.microsoft.com/fwlink/?LinkID=294439">Learn mor
    </div>
    <div class="col-md-4">
        <h2>Samples</h2>
        <p>Download SharePoint Add-ins code samples. New samples are added frequently in JavaScript,
        <p><a class="btn btn-default" href="http://go.microsoft.com/fwlink/?LinkID=294440">Learn mor
    </div>
    <div class="col-md-4">
        <h2>Developer training</h2>
```



Welcome Priyaranjan KS!

SharePoint Add-ins - Embracing web standards, the new cloud add-in
and flexibility to build a new class of SharePoint Add-ins using familiar la

Learn more »

Overview

SharePoint introduces a Cloud Add-in Model that enables you to create add-ins. Learn what SharePoint Add-ins are, why you should build them, and the concepts that are fundamental to building them for SharePoint.

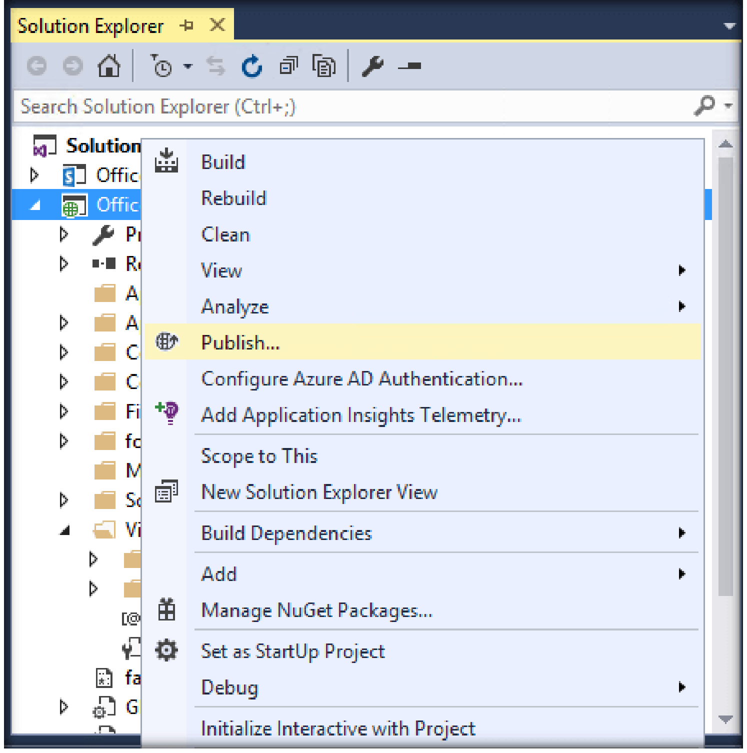
Learn more »

Samples

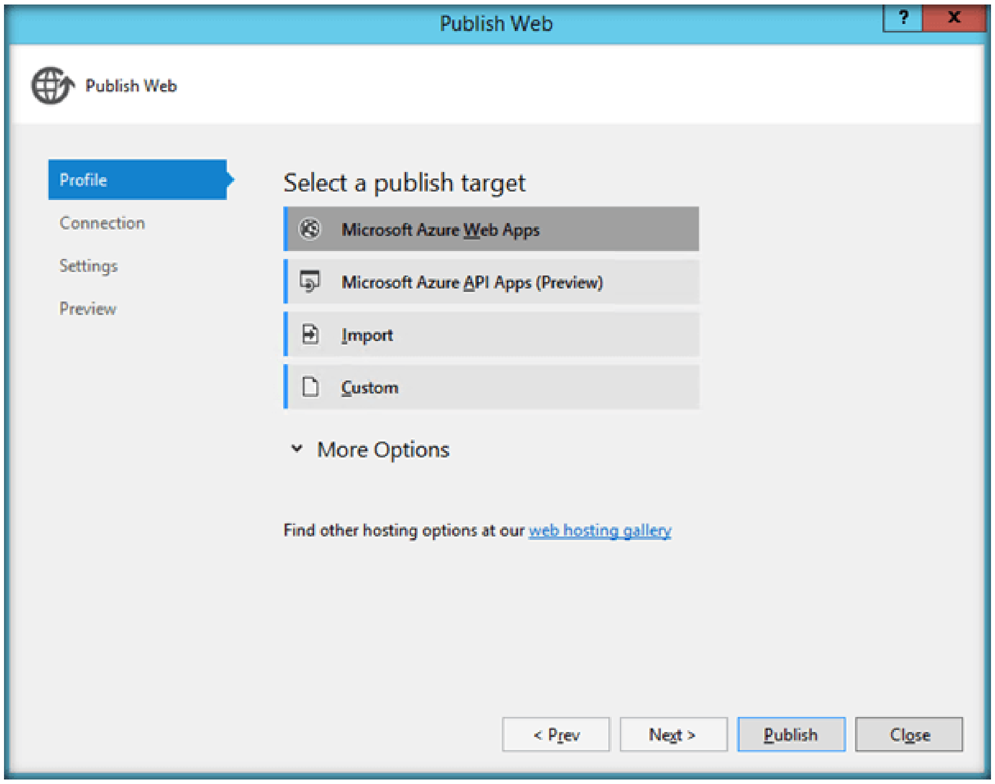
Download SharePoint Add-ins code samples. New samples are added frequently in JavaScript, C#, and VB.NET.

Learn more »

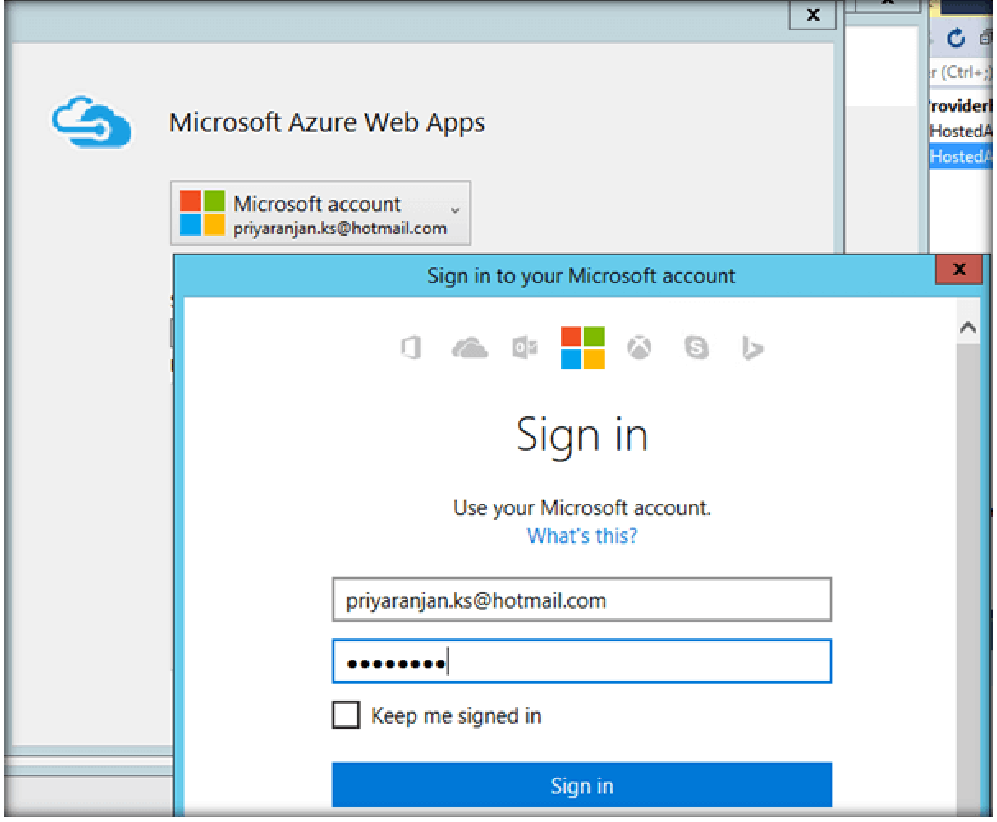
We can update the MVC logic as required in the Controller and View files. MVC implementation is similar to the .NET counterpart. Let’s see what we have to do to get the web site deployed into Azure. Right click the add-in web and select ‘Publish’.


Solution Explorer
Search Solution Explorer (Ctrl+;)
Build
Rebuild
Clean
View
Analyze
Publish...
Configure Azure AD Authentication...
Add Application Insights Telemetry...
Scope to This
New Solution Explorer View
Build Dependencies
Add
Manage NuGet Packages...
Set as StartUp Project
Debug
Initialize Interactive with Project

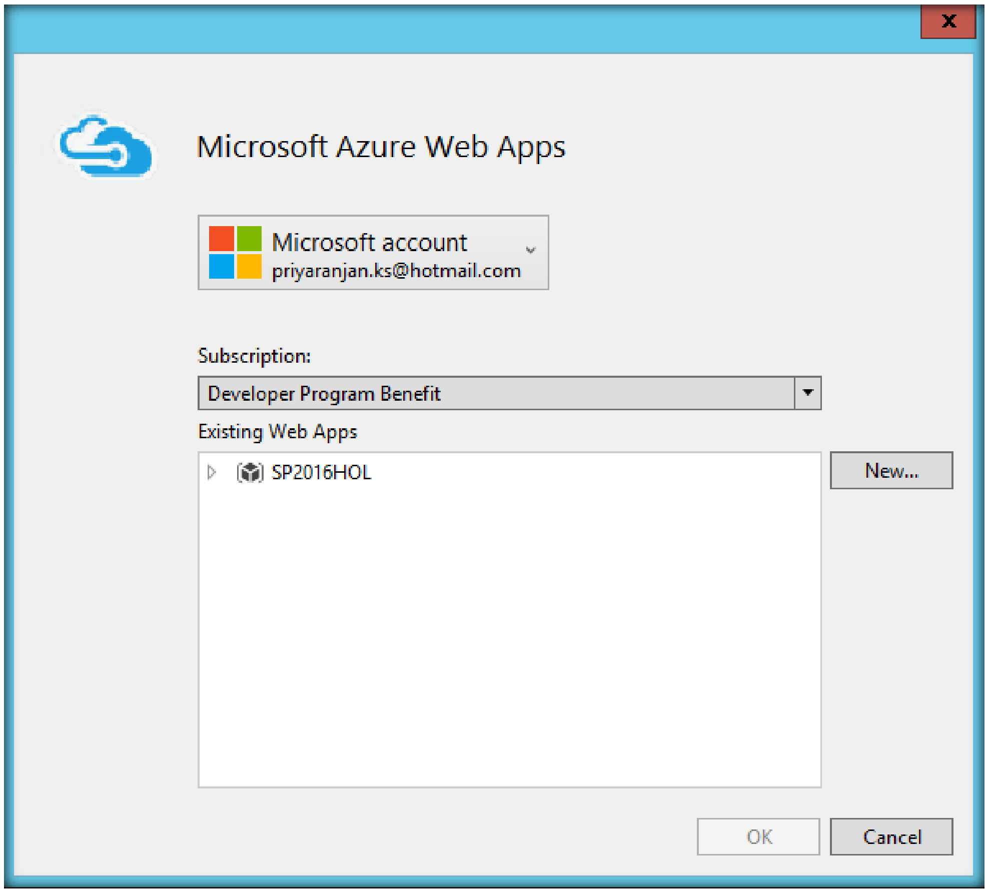
Select Azure Web Apps and click on Next.



This will open up the Azure Subscription Authentication page. Enter the credentials and click on Sign In.


Microsoft Azure Web Apps
Microsoft account
priyaranjan.ks@hotmail.com
Sign in to your Microsoft account
Sign in
Use your Microsoft account.
What's this?
priyaranjan.ks@hotmail.com
Keep me signed in
Sign in

Upon Successful Authentication, It will list out exiting web apps. Let’s create a new one.

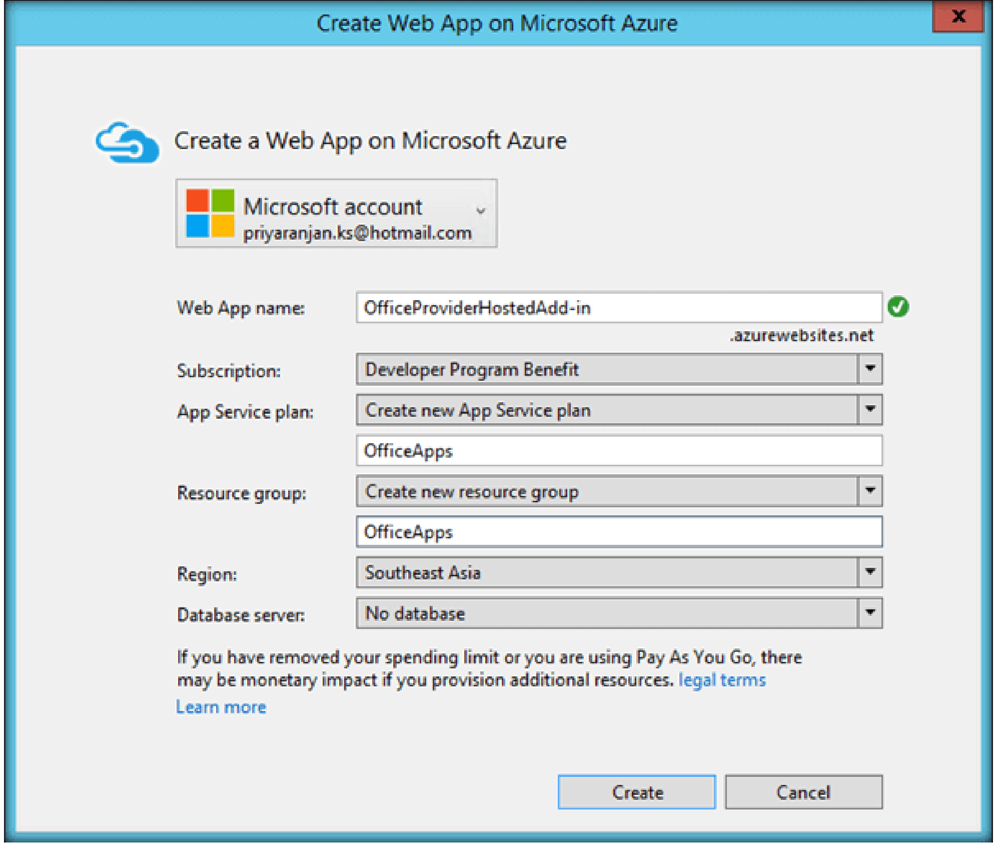


Subscription: Developer Program Benefit

Existing Web Apps

SP2016HOL

Add the details for the new web app and click on Create.


Create Web App on Microsoft Azure

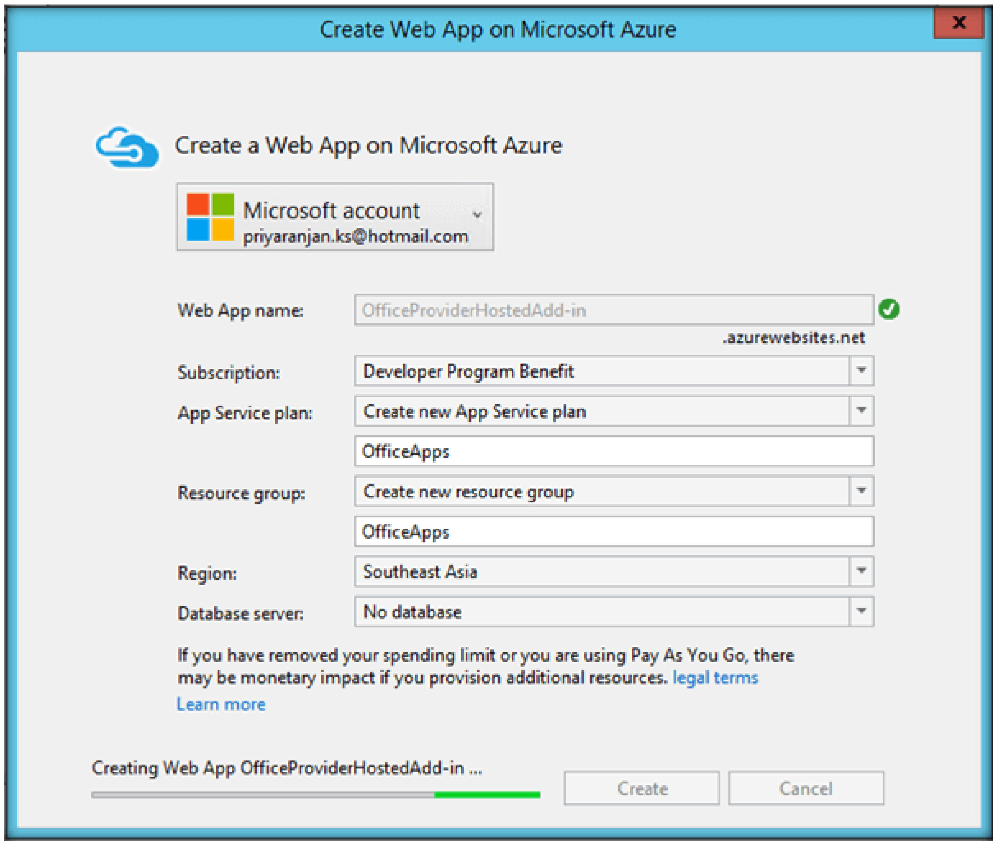
Create a Web App on Microsoft Azure

Microsoft account
priyaranjan.ks@hotmail.com

Web App name:	OfficeProviderHostedAdd-in .azurewebsites.net
Subscription:	Developer Program Benefit
App Service plan:	Create new App Service plan
	OfficeApps
Resource group:	Create new resource group
	OfficeApps
Region:	Southeast Asia
Database server:	No database

If you have removed your spending limit or you are using Pay As You Go, there may be monetary impact if you provision additional resources. legal terms
Learn more

It will start provisioning the Web App.


Create Web App on Microsoft Azure

Create a Web App on Microsoft Azure

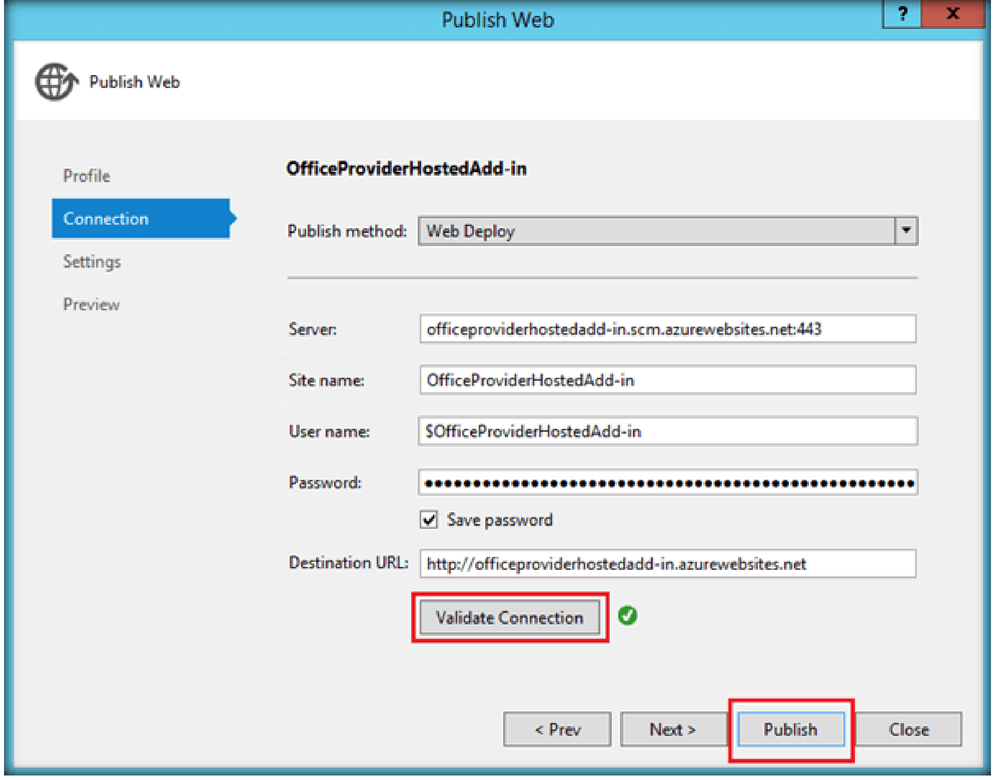
Microsoft account
priyaranjan.ks@hotmail.com

Web App name:	OfficeProviderHostedAdd-in .azurewebsites.net
Subscription:	Developer Program Benefit
App Service plan:	Create new App Service plan
	OfficeApps
Resource group:	Create new resource group
	OfficeApps
Region:	Southeast Asia
Database server:	No database

If you have removed your spending limit or you are using Pay As You Go, there may be monetary impact if you provision additional resources. legal terms
Learn more

Creating Web App OfficeProviderHostedAdd-in ...

Click on validate connection to test the connection. Once it succeeds click on Publish.



OfficeProviderHostedAdd-in

Publish method: Web Deploy

Server: officeproviderhostedadd-in.scm.azurewebsites.net:443

Site name: OfficeProviderHostedAdd-in

User name: $OfficeProviderHostedAdd-in

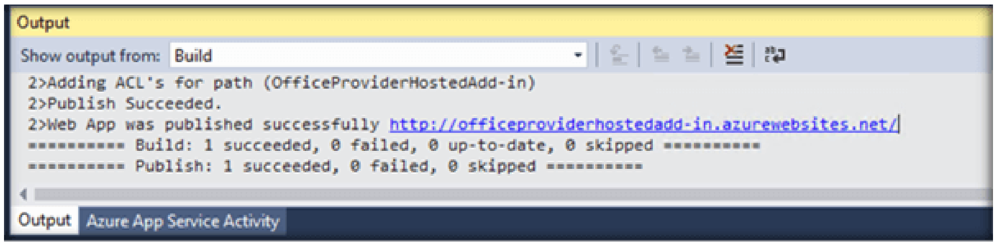
Password: ••

☑ Save password

Destination URL: http://officeproviderhostedadd-in.azurewebsites.net

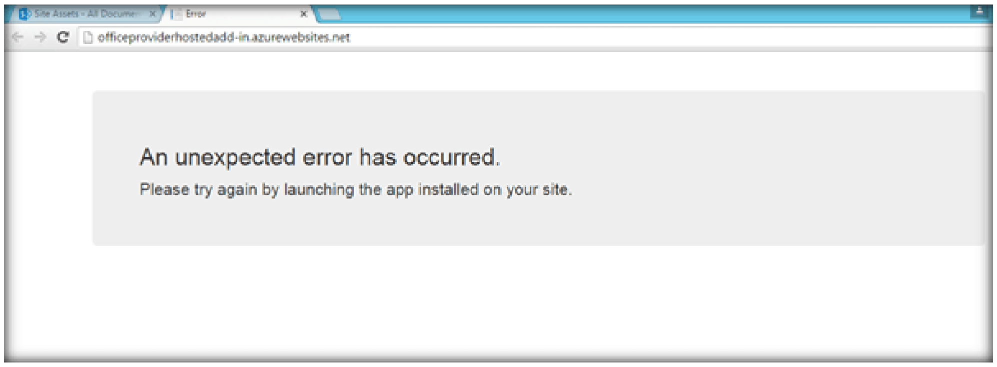
Validate Connection ✓

This will publish the web add-in to Azure.



```
2>Adding ACL's for path (OfficeProviderHostedAdd-in)
2>Publish Succeeded.
2>Web App was published successfully http://officeproviderhostedadd-in.azurewebsites.net/
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
========== Publish: 1 succeeded, 0 failed, 0 skipped ==========
```

It will open up the new web app but will show the below error. This is because we are only half way through and have to update the add-in settings as well. Make a note of the below URL as we will be using it to register the add-in.

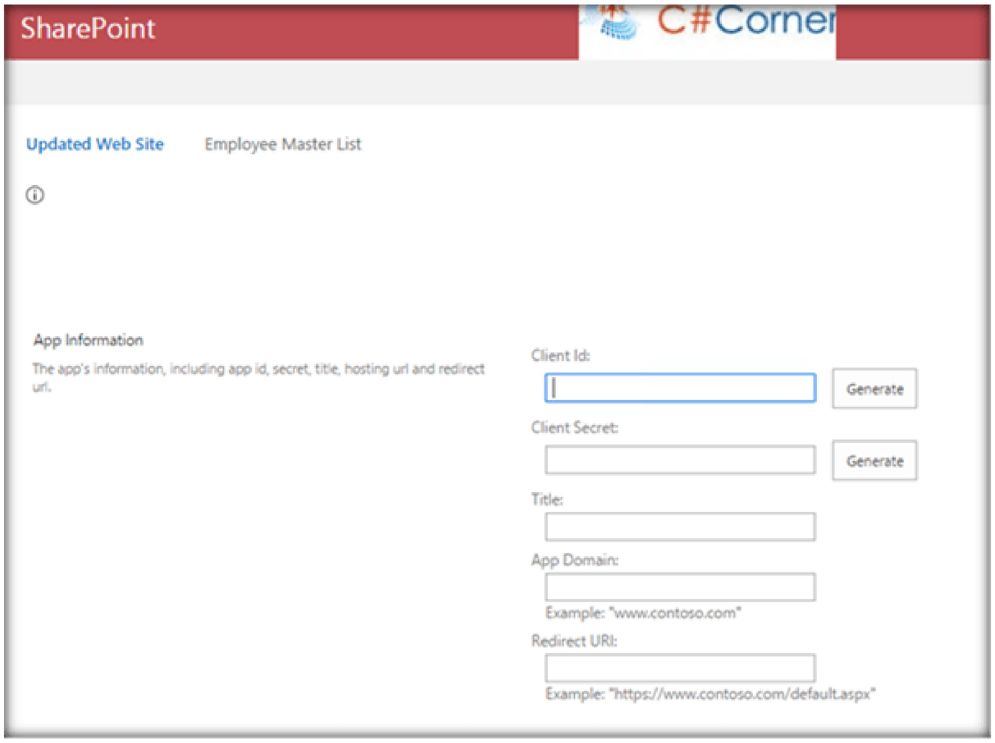

Site Assets - All Documen
Error
officeproviderhostedadd-in.azurewebsites.net

An unexpected error has occurred.

Please try again by launching the app installed on your site.

Register the Add-in in SharePoint Online

Head over to the SharePoint Developer Site and append '/_layouts/15/appregnew.aspx' to the end of the URL. In my case it is: 'https:// playground.sharepoint.com/sites/Playground/_layouts/15/appregnew.aspx'. This will open up the add-in registration page.



App Information

The app's information, including app id, secret, title, hosting url and redirect url.

Client Id:

Client Secret:

Title:

App Domain:

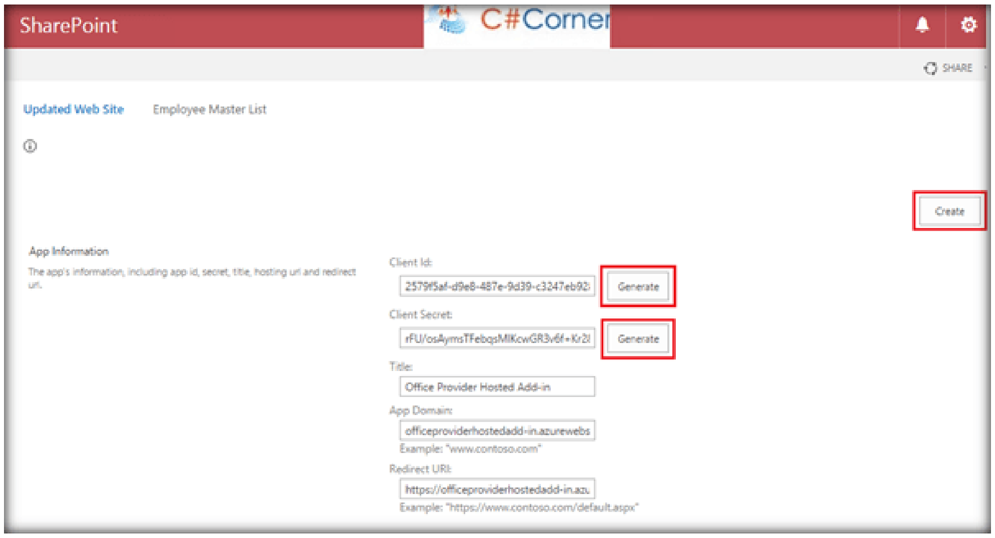
Example: "www.contoso.com"

Redirect URI:

Example: "https://www.contoso.com/default.aspx"

- Click on generate next to auto generate the Client Id and Client Secret. This will generate the Client Id and Client Secret which we will have to update in the add-in solution. Note them down.
- Specify the title as 'Office Provider Hosted Add-in'.
- Specify the app domain of the recently created azure web site 'officeproviderhostedadd-in.azurewebsites.net'
- Enter the redirect URL as 'https://officeproviderhostedadd-in.azurewebsites.net/' . Make sure that you add 'https' to the URL.

Click on Create button.



App Information

The app's information, including app id, secret, title, hosting url and redirect url.

Client Id: 2579f5af-d9e8-487e-9d39-c3247eb92

Client Secret: rFU/osAymsTFebqsMlKcwGR3v6f+Kr2l

Title: Office Provider Hosted Add-in

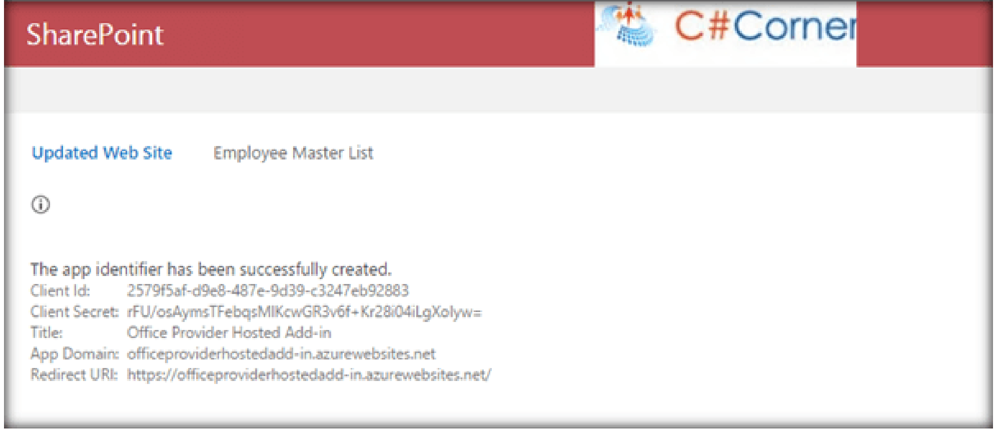
App Domain: officeproviderhostedadd-in.azurewebs

Example: "www.contoso.com"

Redirect URI: https://officeproviderhostedadd-in.azu

Example: "https://www.contoso.com/default.aspx"

This will create the app identifier. Note the entire app identifier details as it will be used later.

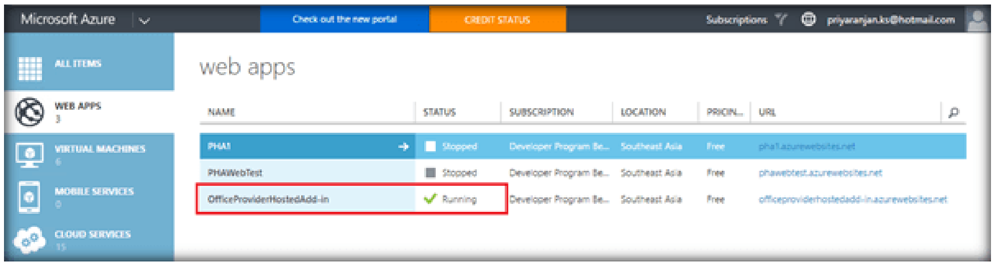


The app identifier has been successfully created.

Client Id:	2579f5af-d9e8-487e-9d39-c3247eb92883
Client Secret:	rFU/osAymsTFebqsMlKcwGR3v6f+Kr28i04iLgXolyw=
Title:	Office Provider Hosted Add-in
App Domain:	officeproviderhostedadd-in.azurewebsites.net
Redirect URI:	https://officeproviderhostedadd-in.azurewebsites.net/

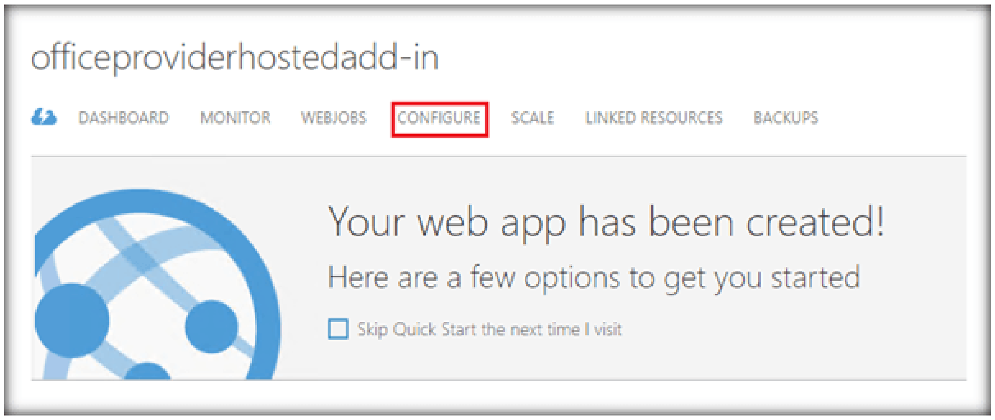
Setup Azure Web Site's App Settings

Now let's configure the app settings for the recently created Azure web site. In order to do that login to the azure subscription. In the web apps section we can see the previously created web app. In our case it is 'OfficeProviderHostedAdd-in'.

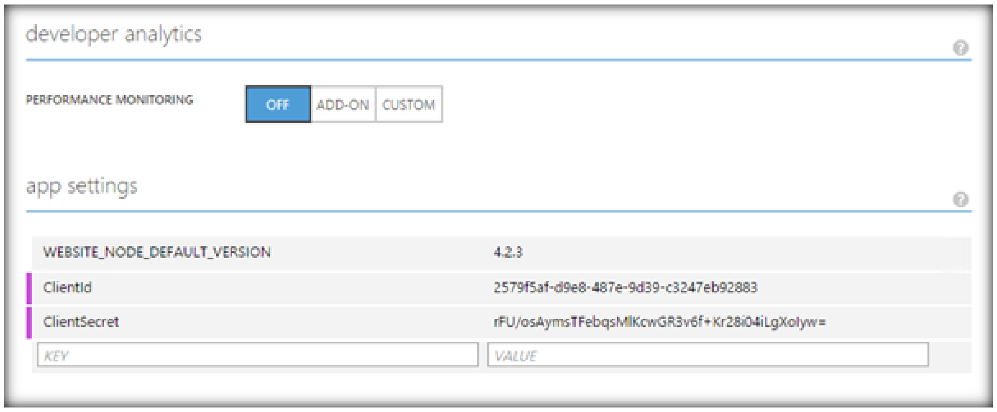


NAME	STATUS	SUBSCRIPTION	LOCATION	PRICIN...	URL
PHA1	Stopped	Developer Program Be...	Southeast Asia	Free	pha1.azurewebsites.net
PHAWebTest	Stopped	Developer Program Be...	Southeast Asia	Free	phawebtest.azurewebsites.net
OfficeProviderHostedAdd-in	✓ Running	Developer Program Be...	Southeast Asia	Free	officeproviderhostedadd-in.azurewebsites.net

Click on the Web App and select 'Configure' tab.


officeproviderhostedadd-in
DASHBOARD
MONITOR
WEBJOBS
CONFIGURE
SCALE
LINKED RESOURCES
BACKUPS
Your web app has been created!
Here are a few options to get you started
Skip Quick Start the next time I visit

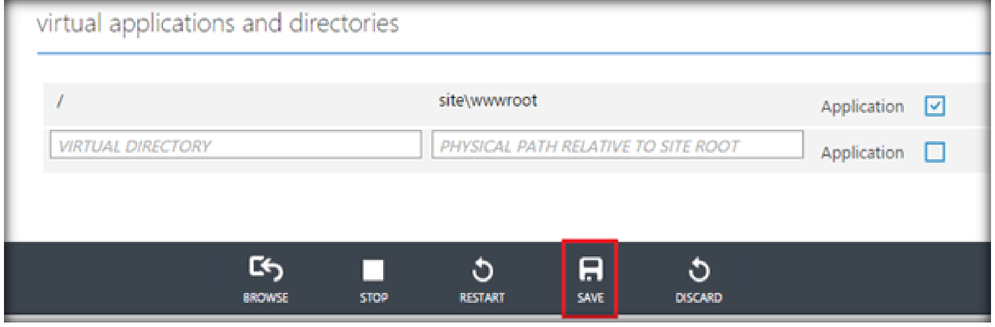
Go to the app settings section and add the 'ClientId' and 'ClientSecret' entries which were created in the SharePoint Online App registration page.



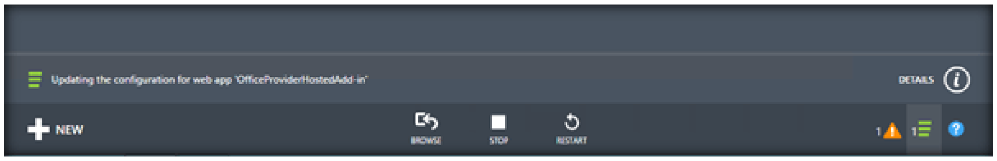
PERFORMANCE MONITORING: [✓ OFF] ADD-ON CUSTOM

WEBSITE_NODE_DEFAULT_VERSION	4.2.3
ClientId	2579f5af-d9e8-487e-9d39-c3247eb92883
ClientSecret	rFU/osAymsTFebqsMlKcwGR3v6f+Kr28i04iLgXolyw=

Click on Save to update the app settings entries.



This will update the Configuration for the Azure Web App where the App Web Site will be hosted.


Updating the configuration for web app 'OfficeProviderHostedAdd-in'

Summary - Thus we saw how to deploy the App Web to the Azure. In the next article we will see how to deploy the app to SharePoint Online in Office 365.

Next Recommended Article

Create And Deploy Provider Hosted Add-In For SharePoint Online - Part Two

In this article, you will learn how to create and deploy Provider Hosted Add-in for SharePoint Online.

Deploy Provider Hosted Add-in | Provider Hosted Add-In | SharePoint Online

Priyaranjan K S TOP 50

Priyaranjan is a Microsoft MVP and a SharePoint developer. He has worked on SharePoint iterations 2007, 2010, 2013 and 2016. He also has hands on experience with Asp.net, C#, Javascript, Jquery, Ajax, JSON, Web Service,... Read more

https://www.c-sharpcorner.com/members/priyaranjan-k-s