# OfFICE PNP: Site Creation RER Sample

|  |  |
| --- | --- |
| Summary: | Applies to: |
| This sample shows how to attach remote event receivers to a list in the host web. | * Office 365 Multi-Tenant (MT) |
| Solution: | Provisioning.SiteCollectionReR, version 1.0 |
| Author: | Vesa Juvonen ,Bert Jansen & Frank Marasco (Microsoft) |
| //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  // THIS CODE IS PROVIDED \*AS IS\* WITHOUT WARRANTY OF  // ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING ANY  // IMPLIED WARRANTIES OF FITNESS FOR A PARTICULAR  // PURPOSE, MERCHANTABILITY, OR NON-INFRINGEMENT.  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* | |

# Introduction

The APIs for creating site collections, subsites, and OneDrive for Business sites are different. Only the on-demand pattern applies to OneDrive for Business sites, see [Frank’s](http://blogs.msdn.com/b/frank_marasco/archive/2014/03/25/so-you-want-to-programmatically-provision-personal-sites-one-drive-for-business-in-office-365.aspx) blog for additional information. You can apply the other two patterns to creation of all other types of SharePoint sites**.** The [Self-Service Site Provisioning using Apps for SharePoint 2013](http://blogs.msdn.com/b/richard_dizeregas_blog/archive/2013/04/04/self-service-site-provisioning-using-apps-for-sharepoint-2013.aspx)sample by Richard diZerega demonstrates this by enabling creation of both subsites and site collections through a [customization form](http://blogs.msdn.com/b/richard_dizeregas_blog/archive/2013/04/04/self-service-site-provisioning-using-apps-for-sharepoint-2013.aspx). You may also visit Vesa "vesku" Juvonen blog for addition information [SharePoint 2013 site provisioning](http://blogs.msdn.com/b/vesku/archive/2014/03/02/sharepoint-online-solution-pack-for-branding-and-provisioning-released.aspx) techniques presentation video recording.

This samples demonstrates

* Association of remote event receiver with custom list on the host web during the app install event
* Removal of the remote event receiver with the app is uninstalled
* Creation of a Site Request List on the host web
* Remote site collection provisioning in a remote event receiver using the app-only policy.

This sample assumes that you have a workflow deployed to the Host web.

This code only works on an Office 365 Multi-Tenant (MT) SharePoint site. With slight modifications this sample will work in an on-premises installation of SharePoint or the current version of SharePoint Online Dedicated. See [Vesa’s](http://blogs.msdn.com/b/vesku/archive/2014/06/09/provisioning-site-collections-using-sp-app-model-in-on-premises-with-just-csom.aspx) Blog for additional information.

public static string CreateSiteCollection(ClientContext ctx, string hostWebUrl, string template, string title, string description, string userEmail)

{

//get the base tenant admin urls

var tenantStr = hostWebUrl.ToLower().Replace("-my", "").Substring(8);

tenantStr = tenantStr.Substring(0, tenantStr.IndexOf("."));

//create site collection using the Tenant object

var webUrl = String.Format("https://{0}.sharepoint.com/{1}/{2}", tenantStr, "sites", title);

var tenantAdminUri = new Uri(String.Format("https://{0}-admin.sharepoint.com", tenantStr));

string realm = TokenHelper.GetRealmFromTargetUrl(tenantAdminUri);

var token = TokenHelper.GetAppOnlyAccessToken(TokenHelper.SharePointPrincipal, tenantAdminUri.Authority, realm).AccessToken;

using (var adminContext = TokenHelper.GetClientContextWithAccessToken(tenantAdminUri.ToString(), token))

{

var tenant = new Tenant(adminContext);

var properties = new SiteCreationProperties()

{

Url = webUrl,

Owner = userEmail,

Title = title,

Template = template,

StorageMaximumLevel = 100,

UserCodeMaximumLevel = 100

};

//start the SPO operation to create the site

SpoOperation op = tenant.CreateSite(properties);

adminContext.Load(tenant);

adminContext.Load(op, i => i.IsComplete);

adminContext.ExecuteQuery();

//check if site creation operation is complete

while (!op.IsComplete)

{

//wait 30seconds and try again

System.Threading.Thread.Sleep(30000);

op.RefreshLoad();

adminContext.ExecuteQuery();

}

}

//get the new site collection

var siteUri = new Uri(webUrl);

token = TokenHelper.GetAppOnlyAccessToken(TokenHelper.SharePointPrincipal, siteUri.Authority, realm).AccessToken;

using (var newWebContext = TokenHelper.GetClientContextWithAccessToken(siteUri.ToString(), token))

{

var newWeb = newWebContext.Web;

newWebContext.Load(newWeb);

newWebContext.ExecuteQuery();

new LabHelper().SetThemeBasedOnName(newWebContext, newWeb, newWeb, "Orange");

// All done, let's return the newly created site

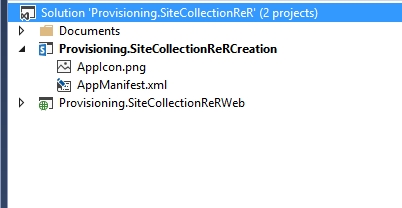
return newWeb.Url;

}

}

}

# Solution



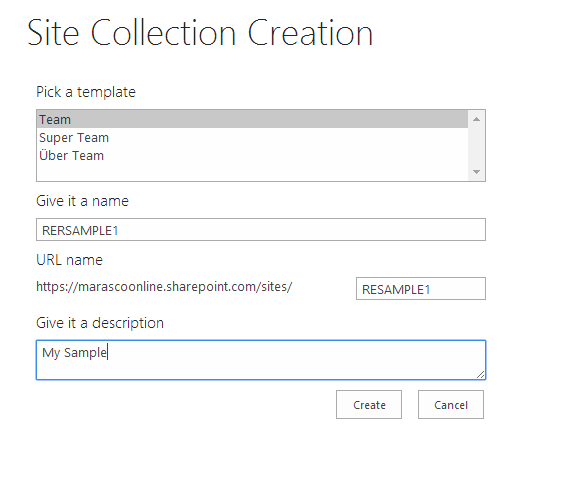
Provisioning.SiteCollectionReRWeb– SharePoint Provider Hosted Application

Because the app needs the ability to create site collections anywhere in the tenancy, it will need FullControl permission on the entire tenancy.  The app will also need to make app-only calls to SharePoint, so it can work with tenant objects or sites outside the context.  Both these settings can be configured in the Permissions tab of the AppManifest.xml.

***NOTE***: *You should typically avoid requesting tenancy permissions in your apps…especially with FullControl.  It is a best practice for apps to request the minimum permissions they need to function.  The “tenancy” permission scope is in place specifically for scenarios like provisioning.*

# Running the SAMPLE

Navigate to the Application and fill in the supplied form. The Site information will be saved to the Site Requests list in the host web.



Navigate to the Site Request List and you may change the State of the given item from New to “Approved”



Once the item has been set to approved, this will invoke the remote event receiver to handle the provisioning logical.

private void HandleItemUpdated(SPRemoteEventProperties properties)

{

using (ClientContext clientContext = TokenHelper.CreateRemoteEventReceiverClientContext(properties))

{

if(clientContext != null)

{

List requestList = clientContext.Web.Lists.GetById(properties.ItemEventProperties.ListId);

ListItem item = requestList.GetItemById(properties.ItemEventProperties.ListItemId);

clientContext.Load(item);

clientContext.ExecuteQuery();

if (String.Compare(item[SiteRequestFields.State].ToString(), "Approved", true) == 0)

{

try

{

string site\_title = item[SiteRequestFields.Title].ToString();

string site\_description = item[SiteRequestFields.Description].ToString();

string site\_template = item[SiteRequestFields.Template].ToString();

string site\_url = item[SiteRequestFields.Url].ToString();

SharePointUser site\_owner = LabHelper.BaseSetUser(clientContext, item, SiteRequestFields.Owner);

LabHelper.CreateSiteCollection(clientContext, site\_url, site\_template, site\_title, site\_description, site\_owner.Email);

item[SiteRequestFields.State] = "COMPLETED";

}

catch(Exception ex)

{

item[SiteRequestFields.State] = "ERROR";

item[SiteRequestFields.StatusMessage] = ex.Message;

}

item.Update();

clientContext.ExecuteQuery();

}

}

}

}

After the site is provisioned the site request will have its status updated to “COMPLETED” or “ERROR” based on the outcome.

# SharePOINT ONLINE SETUP

The first step to create the application principal. The app principal is an actual principal in SharePoint 2013 for the app that can be granted permissions. To register the app principal, we will use the “\_layouts/AppRegNew.aspx”.

Now we need to grant permissions to the app principal. You will have to navigate to another page in SharePoint which is the “\_layouts/AppInv.aspx”. This is where you will grant the application Tenant permissions, so that our Site Provisioning application may create site collections.

<AppPermissionRequests AllowAppOnlyPolicy="true">

<AppPermissionRequest Scope="http://sharepoint/content/tenant" Right="FullControl" />

</AppPermissionRequests>

# Dependencies

* Microsoft.Online.SharePoint.Client.Tenant
* Microsoft.SharePoint.Client.dll
* Microsoft.SharePoint.Client.Runtime.dll
* [Setting up provider hosted app to Windows Azure for Office365 tenant](http://blogs.msdn.com/b/vesku/archive/2013/11/25/setting-up-provider-hosted-app-to-windows-azure-for-office365-tenant.aspx)