# Office AMS: Create sub site or site collection

|  |  |
| --- | --- |
| Summary: | Applies to: |
| This sample shows how to use OfficeAMS core component to create sub sites or new site collections | * Office 365 Multi Tenant (MT) |
| Solution: | Provisioning.CreateSite, version 1.0 |
| Author: | Vesa Juvonen (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.  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* | |

# Scenario: Create sub sites or Site collectiosn usign core component

This sample demonstrates how to create sub sites or site collections using the extensions methods from the Office AMS core component. Extensions are available from normal client side object model objects after you have referenced the Office AMS core component.

## Sub site creation

Sub site creation is actually a single line of code. Following calls are for applying small modifications to the newly created sub site.

// Create the sub site

Web newWeb = ctx.Web.CreateSite(txtName.Text, txtUrl.Text, "Description", drpContentTypes.SelectedValue, 1033);

// Let's add two document libraries to the site

newWeb.CreateDocumentLibrary("Specifications");

newWeb.CreateDocumentLibrary("Presentations");

// Let's also apply theme to the site to demonstrate how easy this is

newWeb.SetThemeToSubWeb(ctx.Web, "Characters");

## Site collection creation

To be able to create site collections, you’ll need to associate to the admin site of the Office365 tenant and in this example we are also using the app only token so that end user does not have to have high permission to the tenant. In following lines we resolve the access token and then create site collection using extension methods.

User currUser = ResolveCurrentUser();

//get the base tenant admin urls

var tenantStr = Page.Request["SPHostUrl"].ToLower().Replace("-my", "").Substring(8);

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

// Let's resolve the admin URL and wanted new site URL

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

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

// Creating new app only context for the operation

string accessToken = TokenHelper.GetAppOnlyAccessToken(

TokenHelper.SharePointPrincipal,

tenantAdminUri.Authority,

TokenHelper.GetRealmFromTargetUrl(tenantAdminUri)).AccessToken;

using (var ctx = TokenHelper.GetClientContextWithAccessToken(tenantAdminUri.ToString(), accessToken))

{

if (ctx.Web.SiteExistsInTenant(webUrl))

{

lblStatus1.Text = string.Format("Site already existed. Used URL - {0}", webUrl);

}

else

{

// Create new site collection with some storage limts and English locale

ctx.Web.CreateSiteCollectionTenant(webUrl, txtName.Text, currUser.Email, drpContentTypes.SelectedValue, 500, 400, 7, 7, 1, 1033);

// Let's get instance to the newly added site collection using URLs

var siteUri = new Uri(webUrl);

string token = TokenHelper.GetAppOnlyAccessToken(TokenHelper.SharePointPrincipal, siteUri.Authority, TokenHelper.GetRealmFromTargetUrl(new Uri(webUrl))).AccessToken;

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

{

// Let's modify the web slightly

var newWeb = newWebContext.Web;

newWebContext.Load(newWeb);

newWebContext.ExecuteQuery();

// Let's add two document libraries to the site

newWeb.CreateDocumentLibrary("Specifications");

newWeb.CreateDocumentLibrary("Presentations");

// Let's also apply theme to the site to demonstrate how easy this is

newWeb.SetThemeToWeb("Characters");

}

lblStatus1.Text = string.Format("Created a new site collection to address <a href='{0}'>{1}</a>", webUrl, webUrl);

}

}