# Office AMS: Modify OneDrive for business site branding

|  |  |
| --- | --- |
| Summary: | Applies to: |
| This sample demonstrates how to provide custom branding to personal my site or OneDrive for Business side automatically using Social CSOM API. | * Office 365 Multi Tenant (MT) * Works also with small adjustments   + Office 365 Dedicated (D)   + SharePoint 2013 on-premises |
| Solution: | Provisioning.OneDrive, 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.  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* | |

# General comments

This scenario shows the pattern on how to access end user’s own OneDrive for Business and to apply custom branding to it automatically. Getting access to the personal my site will happen using Social CSOM which provides read access to user profile properties and also access to the Site object of the personal OneDrive for Business.

Actual branding is applied by uploading custom theme to the Site by using file upload mechanisms in the client side OM and then applying theme to the site. Notice that themes are not visible in the UI of the personal OneDrive for Business sites, but you can still use them.

In general it’s recommended to perform my site branding using themes and to avoid custom master page usage. If you’d start using custom master pages, you would have to ensure that any new changes on the oob master pages are reflected on custom master pages as well. On top of the themes, you can also inject custom CSS to the site to modify layout slightly without the need of changing actual master page. These would be preferred options with the branding.

Actual branding is applied from app part, which can be placed anywhere in the tenant, since it operates cross the site collections as needed. End user will only see gif animation indicating operations when app part is accessing the personal OneDrive for Business site. Typical locations for this customizer would be following locations.

* Intranet front page – When users arrive to Intranet, branding in OneDrive for Business is checked and applied if needed
* Public side of the my site – for example on the news feed page

Code also stores the version of the used branding, so that changes are only applied as needed, which will avoid performance issues with constantly deploying files without clear advantages.

# Scenario: Modify personal OneDrive site

This scenario shows how to access user profile, create personal OneDrive for Business site and accessing created site for modification purposes

## Accessing user profile

Using social CSOM API you can easily access the user profile of particular user.

// Get user profile

ProfileLoader loader = Microsoft.SharePoint.Client.UserProfiles.ProfileLoader.GetProfileLoader(clientContext);

UserProfile profile = loader.GetUserProfile();

User profile then gives access to additional options, like accessing stored information in the user profile properties or to get access to the personal OneDrive for Business site, like in this case.

## Starting OneDrive for Businessvisioning if it doesn’t exist

In this scenario we also schedule the OneDrive for Business creation if it has not yet been created for the particular user. This is done by using following lines of code.

Microsoft.SharePoint.Client.Site personalSite = profile.PersonalSite;

clientContext.Load(personalSite);

clientContext.ExecuteQuery();

// Let's check if the site already exists

if (personalSite.ServerObjectIsNull.Value)

{

// Let's queue the personal site creation using oob timer job based approach

// Using async mode, since end user could go away from browser, you could do this using oob web part as well

profile.CreatePersonalSiteEnque(true);

clientContext.ExecuteQuery();

WriteDebugInformationIfNeeded("My site was not present, will be provisioned.");

}

This will schedule creation of the personal OneDrive for Business site using timer job based approach, like with the oob behavior.

## Accessing OneDrive for Business site cross site collections

If personal OneDrive for Business site has been already created, we can access that simply using standard CSOM after getting instance to the site collection object from the user profile.

Microsoft.SharePoint.Client.Site personalSite = profile.PersonalSite;

clientContext.Load(personalSite);

clientContext.ExecuteQuery();  
  
Web rootWeb = personalSite.RootWeb;

clientContext.Load(rootWeb);

clientContext.ExecuteQuery();

# Scenario: Deploy theme and apply that to site

This scenario shows how to deploy and apply custom theme to site.

## Deployign theme

Deploying of the theme can be achieve just by deploying theme files to right locations in the site using FileCreationInformation object. In this example case, we deploy three different files, which are then applied as “theme” to the site. You could actually deploy these files anywhere in the site, but for consistency sake, they are added to the same location as the oob files using following pattern.

Individual files are handled one-by-one by calling same method.

// Deploy files one by one to proper location

DeployFileToThemeFolderSite(clientContext, web, "DeploymentFiles/Theme/Contoso.spcolor");

DeployFileToThemeFolderSite(clientContext, web, "DeploymentFiles/Theme/Contoso.spfont");

DeployFileToThemeFolderSite(clientContext, web, "DeploymentFiles/Theme/contosobg.jpg");

Actual deployment is done in the method as follows.

private void DeployFileToThemeFolderSite(ClientContext clientContext, Web web, string sourceAddress)

{

// Get the path to the file which we are about to deploy

string file = HostingEnvironment.MapPath(string.Format("~/{0}", sourceAddress));

List themesList = web.GetCatalog(123);

// get the theme list

clientContext.Load(themesList);

clientContext.ExecuteQuery();

Folder rootfolder = themesList.RootFolder;

clientContext.Load(rootfolder);

clientContext.Load(rootfolder.Folders);

clientContext.ExecuteQuery();

Folder folder15 = rootfolder;

foreach (Folder folder in rootfolder.Folders)

{

if (folder.Name == "15")

{

folder15 = folder;

break;

}

}

// Use CSOM to uplaod the file in

FileCreationInformation newFile = new FileCreationInformation();

newFile.Content = System.IO.File.ReadAllBytes(file);

newFile.Url = folder15.ServerRelativeUrl + "/" + Path.GetFileName(sourceAddress);

newFile.Overwrite = true;

Microsoft.SharePoint.Client.File uploadFile = folder15.Files.Add(newFile);

clientContext.Load(uploadFile);

clientContext.ExecuteQuery();

}

Code adds also new theme option to the theme item list, which would not actually be needed and it not visible in the my sites, but you can use the same pattern when deploying theme to team sites and this would result new option in theme selection.

private void AddNewThemeOptionToSite(ClientContext clientContext, Web web)

{

// Let's get instance to the composite look gallery

List themesOverviewList = web.GetCatalog(124);

clientContext.Load(themesOverviewList);

clientContext.ExecuteQuery();

// Is the item already in the list?

if (!ContosoThemeEntryExists(clientContext, web, themesOverviewList))

{

// Let's create new theme entry. Notice that theme selection is not available from UI in personal sites, so this is just for consistency sake

ListItemCreationInformation itemInfo = new ListItemCreationInformation();

Microsoft.SharePoint.Client.ListItem item = themesOverviewList.AddItem(itemInfo);

item["Name"] = "Contoso";

item["Title"] = "Contoso";

item["ThemeUrl"] = URLCombine(web.ServerRelativeUrl, "/\_catalogs/theme/15/contoso.spcolor"); ;

item["FontSchemeUrl"] = URLCombine(web.ServerRelativeUrl, "/\_catalogs/theme/15/contoso.spfont"); ;

item["ImageUrl"] = URLCombine(web.ServerRelativeUrl, "/\_catalogs/theme/15/contosobg.jpg");

// Notice that we use oob master, but just as well you vould upload and use custom one

item["MasterPageUrl"] = URLCombine(web.ServerRelativeUrl, "/\_catalogs/masterpage/seattle.master");

item["DisplayOrder"] = 0;

item.Update();

clientContext.ExecuteQuery();

}

}

## Applying theme

Actual applying of the theme is done with single line of code as long as the URLs to the file are properly created.

//Set the properties for applying custom theme which was just uploaded

string spColorURL = URLCombine(rootWeb.ServerRelativeUrl, "/\_catalogs/theme/15/contoso.spcolor");

string spFontURL = URLCombine(rootWeb.ServerRelativeUrl, "/\_catalogs/theme/15/contoso.spfont");

string backGroundImage = URLCombine(rootWeb.ServerRelativeUrl, "/\_catalogs/theme/15/contosobg.jpg");

// Use the Red theme for demonstration

rootWeb.ApplyTheme(spColorURL,

spFontURL,

backGroundImage,

false);

clientContext.ExecuteQuery();