# Office AMS: Page MODIFICATION

|  |  |
| --- | --- |
| Summary: | Applies to: |
| This sample demonstrates two basic techniques for provisioning wiki pages: creating a wiki page in a wiki library, and then manipulating it. | * Office 365 Multi-Tenant (MT) * Office 365 Dedicated (D) * SharePoint 2013 on-premises |
| Solution: | Contoso.Core.ModifyPages, 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.  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* | |

## Version log

Change log between different releases.

|  |  |
| --- | --- |
| Version | Notes |
| 1.0 | Initial |

# General comments

The sample uses the SitePages library which is default available in non-publishing sites, but the same sample can be easily adapted to create pages in a custom wiki page library.

In the last step, the empty wiki page is filled with HTML content. In the sample you can specify your HTML content and then click Run scenario 1 to create the wiki page and add the HTML

# SCENARIo: Create a wiki page

The following code adds a wiki page

using (var ctx = spContext.CreateUserClientContextForSPHost())

{

string scenarioPage = String.Format("scenario1-{0}.aspx", DateTime.Now.Ticks);

string scenarioPageUrl = AddWikiPage(ctx, ctx.Web, "Site Pages", scenarioPage);

AddHtmlToWikiPage(ctx, ctx.Web, "SitePages", htmlEntry.Text, scenarioPage);

hplPage.NavigateUrl = string.Format("{0}/{1}", Request.QueryString["SPHostUrl"], scenarioPageUrl);

}

The Following code will add HTML to the wiki page

Microsoft.SharePoint.Client.Folder pagesLib = web.GetFolderByServerRelativeUrl(folder);

ctx.Load(pagesLib.Files);

ctx.ExecuteQuery();

Microsoft.SharePoint.Client.File wikiPage = null;

foreach (Microsoft.SharePoint.Client.File aspxFile in pagesLib.Files)

{

if (aspxFile.Name.Equals(page, StringComparison.InvariantCultureIgnoreCase))

{

wikiPage = aspxFile;

break;

}

}

if (wikiPage == null)

{

return;

}

ctx.Load(wikiPage);

ctx.Load(wikiPage.ListItemAllFields);

ctx.ExecuteQuery();

string wikiField = (string)wikiPage.ListItemAllFields["WikiField"];

Microsoft.SharePoint.Client.ListItem listItem = wikiPage.ListItemAllFields;

listItem["WikiField"] = html;

listItem.Update();

ctx.ExecuteQuery();

# SCENARIo: Advanced WIKI PAGE MODIFICATION

This scenario extends the first scenario, providing the option to create wiki pages that use other layouts.

This sample reuses existing out-of-the-box layouts, but it's possible to use custom layouts as well. Once the layout is created, a table with rows and columns and, optionally, headers and footers are created. Next, the sample shows how to place a web part or HTML content into a table cell. When you click Run scenario 2, a wiki page will be created with the layout you've selected and:

Place an XsltListViewWebPart in the first content row on column 1. If there's no header, then the first row is row1, column 1. If there's a layout with a header, then the first row is row 2, column 1. The XsltListView web part is created for a promoted links list that will be automatically created if it doesn't already exist.

If there are two or more columns, the HTML content listed for scenario 1 will be inserted in the column next to the web part.

After the page is created, you can use the link to show the created page.

Once a page has been created the button Remove webpart from the last page created during scenario 2 run is enabled. Clicking on this button removes the XsltListViewWebPart web part from the page.

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

using (var ctx = spContext.CreateUserClientContextForSPHost())

{

if (new LabHelper().AddList(ctx, ctx.Web, 170, new Guid("192efa95-e50c-475e-87ab-361cede5dd7f"), "Links", false))

{

new LabHelper().AddPromotedSiteLink(ctx, ctx.Web, "Links", "OfficeAMS on CodePlex", "http://officeams.codeplex.com");

new LabHelper().AddPromotedSiteLink(ctx, ctx.Web, "Links", "Bing", "http://www.bing.com");

}

string scenario2Page = String.Format("scenario2-{0}.aspx", DateTime.Now.Ticks);

string scenario2PageUrl = AddWikiPage(ctx, ctx.Web, "Site Pages", scenario2Page);

AddHtmlToWikiPage(ctx, ctx.Web, "SitePages", LabHelper.WikiPage\_ThreeColumnsHeaderFooter, scenario2Page);

Guid linksID = new LabHelper().GetListID(ctx, ctx.Web, "Links");

WebPartEntity wp2 = new WebPartEntity();

wp2.WebPartXml = new LabHelper().WpPromotedLinks(linksID, string.Format("{0}/Lists/{1}",

Request.QueryString["SPHostUrl"], "Links"),

string.Format("{0}/{1}", Request.QueryString["SPHostUrl"],

scenario2PageUrl),"$Resources:core,linksList");

wp2.WebPartIndex = 1;

wp2.WebPartTitle = "Links";

new LabHelper().AddWebPartToWikiPage(ctx, ctx.Web, "SitePages", wp2, scenario2Page, 2, 1, false);

new LabHelper().AddHtmlToWikiPage(ctx, ctx.Web, "SitePages", htmlEntry.Text, scenario2Page, 2, 2);

this.hplPage2.NavigateUrl = string.Format("{0}/{1}", Request.QueryString["SPHostUrl"], scenario2PageUrl);

}

# Dependencies

* Microsoft.SharePoint.Client.dll
* Microsoft.SharePoint.Client.Runtime.dll