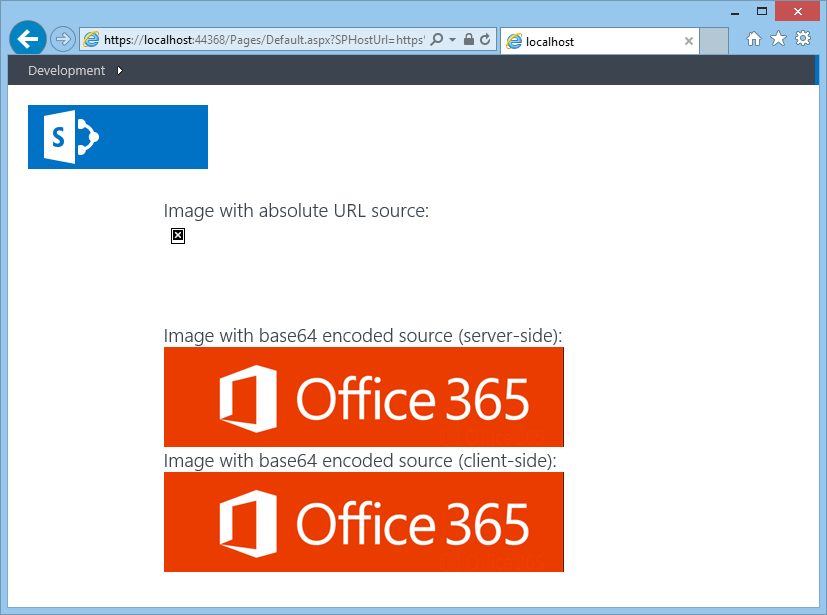
# Office AMS: Cross-Domain Images

|  |  |
| --- | --- |
| Summary: | Applies to: |
| This sample shows how display secured/cross-domain images in a provider-hosted app by using a service to convert them as base64 encoded strings | * Office 365 Multi Tenant (MT) * Office 365 Dedicated (D) * SharePoint 2013 on-premises |
| Solution: | Core.CrossDomainImages, version 1.0 |
| Author: | Richard diZerega (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

The sample shows a technique for displaying secured images in provider-hosted app that live in a separate domain (ex: AppWeb or MySite). This can be largely problematic when the two domains are in separate security zones as the domains will not be able to share browser tokens. The technique outlined in the sample leverages a REST Service in the provider-hosted app to “proxy” the image delivery as a base64 encoded string instead of an absolute URL. All you need is a SharePoint access token.

The following image shows an expected outcome of running the app. Notice that the first images fails to load, but the base64 encoded images display as expected:



# Scenario 1: GetTing Image Server-Side

This scenario shows how to get the encoded image server-side:

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

using (var clientContext = spContext.CreateUserClientContextForSPAppWeb())

{

//set access token in hidden field for client calls

hdnAccessToken.Value = spContext.UserAccessTokenForSPAppWeb;

//Set image source using traditional techniques (absolute URL)

Image1.ImageUrl = spContext.SPAppWebUrl + "AppImages/O365.png";

//Set image source as Base64 string by using our ImgService Proxy

Services.ImgService svc = new Services.ImgService();

Image2.ImageUrl = svc.GetImage(spContext.UserAccessTokenForSPAppWeb, spContext.SPAppWebUrl.ToString(), "AppImages", "O365.png");

}

# Scenario 2: GetTing Image Client-Side

This scenario shows how to get the encoded image client-side:

<script type="text/javascript">

var context;

//Wait for the page to load

$(document).ready(function () {

var appWebUrl = decodeURIComponent(getQueryStringParameter('SPAppWebUrl'));

//make client-side call for the base64 image

$.ajax({

url: '../Services/ImgService.svc/GetImage?accessToken=' + $('#hdnAccessToken').val() + '&site=' + encodeURIComponent(appWebUrl + '/') + '&folder=AppImages&file=O365.png',

dataType: 'json',

success: function (data) {

$('#Image3').attr('src', data.d);

},

error: function (err) {

alert('error occurred');

}

});

});</script>