# Office Dev PnP documentation

|  |  |
| --- | --- |
| Summary: | Applies to: |
| Office AMS is a collection of SharePoint and Office apps created for the field. These apps use an Office AMS Core library that bundles the various functionalities implemented using CSOM into a single reusable library. | * Office 365 Multi Tenant (MT) * Office 365 Dedicated (D) * SharePoint 2013 on-premises |
| Solution: | OfficeDevPnP.Core, version 1.0 |
| Author: | Office Developer PnP team (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

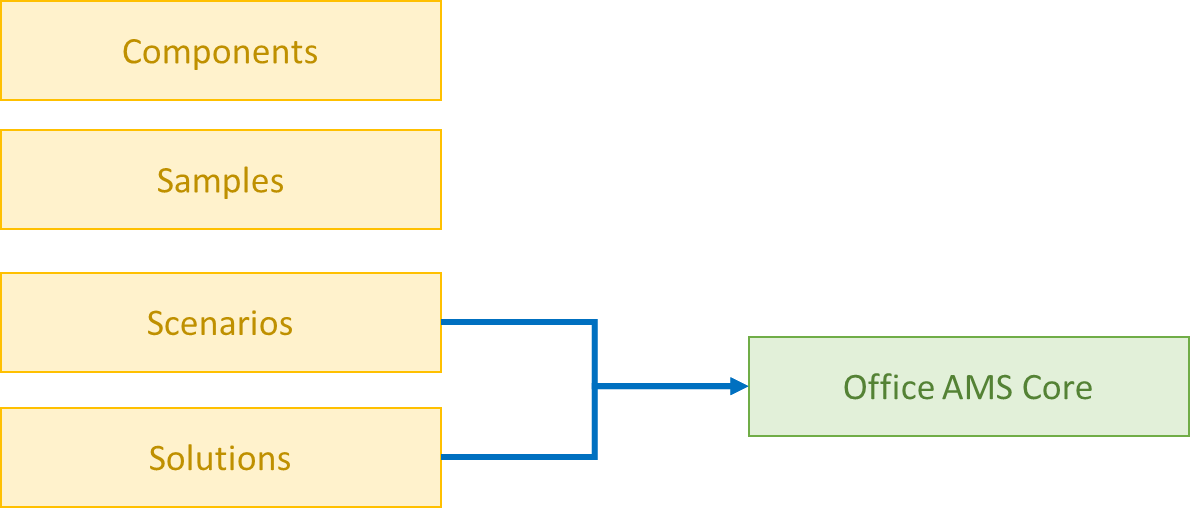
Office AMS initially started as a set of independent samples which we brought out publicly during the SharePoint Conference 2014 as we would like to provide these samples to field as soon as possible. If start looking at these samples then individually they all do their thing, but there are samples with overlapping pieces of code and for certain tasks like getting a valid client context multiple coding patterns where used. As of this release of Office AMS we’ve moved all reusable code into a single library called Office AMS Core and we’ve worked out proven patterns that will be reused by the samples in this Office AMS release. Even more important is the fact that you now can use this Office AMS Core library in your projects. This will make it easier for you to create great SharePoint Apps but will also make it easier to update your apps to newer releases of Office AMS as you only will need to update the Office AMS Core library.

## Office Dev PnP structure

Where the previous releases where just a list of samples we’ve now added some more structure to this Office AMS release. All the samples of the previous release and the newly added ones are placed under one of the following 4 categories:

* **Components**: this category holds all samples that implement a UI component (e.g. the people and taxonomy picker samples)
* **Samples**: this category contains small and simple samples that are meant to show one particular tasks in a simple, easy to understand manner. Typical samples would be how to create content type, how to use remote event receivers, etc.
* **Scenarios**: the scenarios category holds samples that show how you can do a number of related tasks. A sample here would be the provisioning of web pages as it shows how to create pages but also how to manipulate the web parts on those pages
* **Solutions**: this last category holds the more complex (enterprise scale) samples that show you how you can implement a particular process end-to-end. A sample here is site provisioning: this sample starts from a site directory, shows how to use Azure to deal with site provisioning tasks in background, shows how to do site templates using CSOM and finally also shows how to use Azure Service Bus to also deal with provisioning for your on-premises SharePoint environment

If you look a bit deeper into the samples than you’ll notice that all the samples under the scenarios and solutions categories are using the Office AMS Core library whereas as the samples under the components and samples category do not use Office AMS Core. This is a deliberate choice that the Office AMS team made as we wanted the samples to be very simple, without references to reusable library, as their main goal is to teach how a particular task can be done using Cloud Application Model.

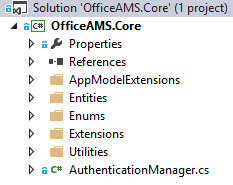


The samples, regardless of which of the 4 categories they belong to, all do have a prefix which indicates their main purpose. For this release of Office AMS we’ve foreseen the following prefixes:

* **Branding**: prefix for all samples that deal with branding related work such as working with themes and applying CSS
* **Provisioning**: prefix for all samples that deal with site provisioning. The more complex site provisioning samples also do branding, but since the main goal is provisioning the call under this prefix
* **Core**: prefix for all other core SharePoint purposes.

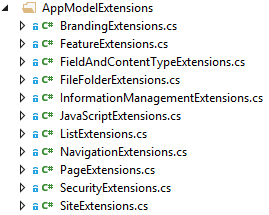
# Office AMS Core

In this chapter we’ll explain a bit more about the Office AMS core library. The Office AMS Core library contains the following elements which will be described in more detail:



## AppModelExtensions

This folder contains only static classes with [extension methods](http://msdn.microsoft.com/en-us/library/bb383977.aspx) that extend the SharePoint Web, Site and List objects. Below screenshot shows the currently available extension classes:



A typical extension method in these classes looks like this:

public static ContentType CreateContentType(this Web web, string name, string id, string group)

{

// Load the current collection of content types

ContentTypeCollection contentTypes = web.ContentTypes;

web.Context.Load(contentTypes);

web.Context.ExecuteQuery();

ContentTypeCreationInformation newCt = new ContentTypeCreationInformation();

// Set the properties for the content type

newCt.Name = name;

newCt.Id = id;

newCt.Group = group;

ContentType myContentType = contentTypes.Add(newCt);

web.Context.ExecuteQuery();

//Return the content type object

return myContentType;

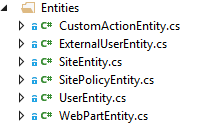
}

To use this extension method you can do this:

cc.Web.CreateContentType("My CT","0x0101009189AB5D3D2647B580F011DA2F356FB2","My custom content types group");

## Entities

Entities are simple classes used to provide and retrieve more complex objects from the extensions methods in AppModelExtensions. Currently following entities are defined:



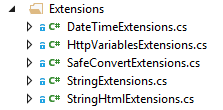
## Enums

Extension methods in AppModelExtensions can use enums and if so these enum classes are created in this folder:



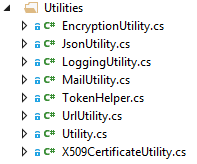
## Extensions

This folder contains [extension methods](http://msdn.microsoft.com/en-us/library/bb383977.aspx) that are not SharePoint related such as extension methods to help with string manipulations.



## Utilities

Utility classes (helper classes) are created in this folder.



## AuthenticationManager.cs

AuthenticationManager is the class that you can use to obtain a client context in case you’re not having one available as part of the SharePoint App (e.g. in console projects) or when you want to create a client context using different credentials or using an AppOnly app. Following methods are available for this class:

public ClientContext GetSharePointOnlineAuthenticatedContextTenant(string siteUrl, string tenantUser, string tenantUserPassword)

public ClientContext GetAppOnlyAuthenticatedContext(string siteUrl, string realm, string appId, string appSecret)

public ClientContext GetNetworkCredentialAuthenticatedContext(string siteUrl, string user, string password, string domain)