# Office AMS: Embedded editing

|  |  |
| --- | --- |
| Summary: | Applies to: |
| This sample shows how manage app part properties using embedded edit controls instead of the controls provided by the OOTB app part editor. | * Office 365 Multi Tenant (MT) * Office 365 Dedicated (D) * SharePoint 2013 on-premises |
| Solution: | Core.EmbeddedEditing, version 1.0 |
| Author: | Matt Mazzola (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 1: How to manage app part properties using embedded controls

## Problem

The out of the box (OOTB) app part editor controls may not offer all the flexibility you need for your project. For instance, when developing advanced apps, you may have properties that are dynamic or have custom validation. The OOTB properties are restricted to only allow the following basic types: string, int, enum, boolean.

## Background

When researching how to solve this problem there are two main questions we must answer:

1. How can the app part know if it is being edited?
2. How can we store properties specific to this instance of the app part?

Fortunately, SharePoint provides two properties to the app part via the url querystring when loading the iFrame into the page. Those two properties are **editMode** and **wpId**.

These properties are self-explanatory.  
**editMode**: Indicates if the app part is being edited with a value of 1 for edit and 0 for non-edit.  
**wpId:** Is the ID of the app part / web part which is a GUID generated when the a new instance of the app is added to a page.

## References

* [Use the SharePoint hidden list to store configuration data for an app part instance](http://blogs.msdn.com/b/officeapps/archive/2013/09/19/use-the-sharepoint-hidden-list-to-store-configuration-data-for-an-app-part-instance.aspx)
* [Adding Custom Properties to an App Part](http://msdn.microsoft.com/en-us/library/office/fp179921.aspx)

## Solution

Now that we know about these extra properties available to app parts we can easily provide a solution to the problem and the two questions listed in the background section above:

1. **How can the app part know if it is being edited?**
   1. When the app part is loaded check the status of the **editMode** property, if edit mode is enabled, expose the controls to edit the app part, otherwise behave normally.
2. **How can we store properties specific to this instance of the app part?**
   1. When the user saves the edit form, save the property names and values along with the web part id (**wpId** value) into a local list (list located on the app web) so that we can retrieve them later.
      1. We must have logic to know when to create new items in the list for properties that do not have values yet, or to update list items with new property values.

\* The downside of this approach is that there is no way to know if an app part is removed from a page in order to remove/clean-up its information from the configuration list. For more information review the first link in the references section.

## Scenarios To Handle

Ideally there is one entry with the app part id per property the app part needs; however, there might be no entries, partial entries, or more entries that needed. Here is a one way to handle these cases:

|  |  |
| --- | --- |
| Case | Action |
| Configuration List Not Found | Use app property defaults for this instance. |
| No entries found for this app part in config list. | Use defaults from app, save to config list, display results in form. |
| Partial Entries found for this app part in config list. | Fill in missing data with defaults, save to config list, display results in form. |
| More Entries than expected for this app part in config list. | Last parsed item overwrites previous value, display results in form. \* Don't delete duplicates since we're not sure why they occurred and we don't really know which entry is correct. |

## Objectives

* Recreate the OOTB functionality with basic properties such as string, integer, boolean, and enum.
* Extend with advanced functionality such as host list selection.

## Description

We will assume a real world scenario of a basic app which displays data from a list on the host web which needs the following properties:

* **Title**: Internal Title. This would be to allow styling other than what chrome control provides. (Basic Property Objective)
* **RowLimit**: Maximum number of items to request from the host web list. (Basic Property Objective)
* **List GUID**: The guid for the host web list so we can request data. (Advanced Objective)