Customization Plug-In Guide

**There are two ways of creating a Customization Plug-In file:**

1) Using the customization Editor

2) Copying the code into a cs file and including it in your project.

Graphical user interface, application

Description automatically generated

**The Customization Plug-In code works the same as Acumatica Framework code.**

1) Instantiate an instance of a graph – while “using” the associated namespace for that object

2) Create a matching DAC object and fill it with data

3) Update the graph with the created object

4) Save the now filled graph instance.

**\*See next Page for Simple examples and explanation\***

Graphical user interface, application

Description automatically generated

**// RoleAccess graph = Business Logic name**

RoleAccess graph = PXGraph.CreateInstance<RoleAccess>();  
**//graph.Roles.Insert = Roles view name, .Insert is pressing the insert “+” button**     var exRoles= graph.Roles.Insert();  
      **//Rolename = Data Field name**  
     exRoles.Rolename="TestRole";  
    exRoles.Descr="Test Role";

**//to insert into the sub graph, you need to do the same, UsersByRole = the viewname of the sub graph, and we need to press Insert**  
     var usersInRoles= graph.UsersByRole.Insert();  
     usersInRoles.Username= "Andrews";  
**//graph.Persist is pressing the Save button on the main graph.**     graph.Persist();

**Here are some more basic examples:**

