Technical Note

Extending the REP++ Metadata with User-Defined Attributes

Author: R&D Department

Publication date: December 14, 2006

Revised date: May 2010

Extending the REP++ Metadata with User-Defined Attributes

Overview

There are times when the metadata associated to a number of standard REP++ objects (e.g. Field, Rowset, RowsetTree, etc.) is simply insufficient to describe a particular kind of data. For instance, there have been a number of projects where:

- For each field, we had to maintain a URL pointing to online documentation for that field. This URL was later used by a custom report generator to produce hyperlinks to the online documentation.
- For each field, we had to add a flag that indicated the background color for the field column in a grid.
- For certain fields, we had to maintain the name of a cache item that stored a list of choices available for the field, in order to avoid re-reading the list every time.
- For each Rowset, we had to specify the location of a custom report template that was used by a custom report generator.

To implement these capabilities, we used user-defined attributes (UDAs). A UDA is an element of information added by the developer to extend the metadata of a number of standard REP++ objects in order to store additional information. UDAs can be associated to Fields (attached or not to a Rowset), Rowsets, RowsetTrees, RowsetTreeDefNodes, etc. They are created in a special editor that lets you define their type, the object to which they are associated, their size, and other related aspects.

This article describes how to create user-defined attributes using REP++*studio*, how to associate them to REP++ objects and how to access UDA values programmatically.

Creating user-defined attributes

User-defined attributes are created using the REP++ **User Attributes Editor**, which can be launched from REP++*studio* either from the start-up toolbar, from the main menu or from the main toolbar.

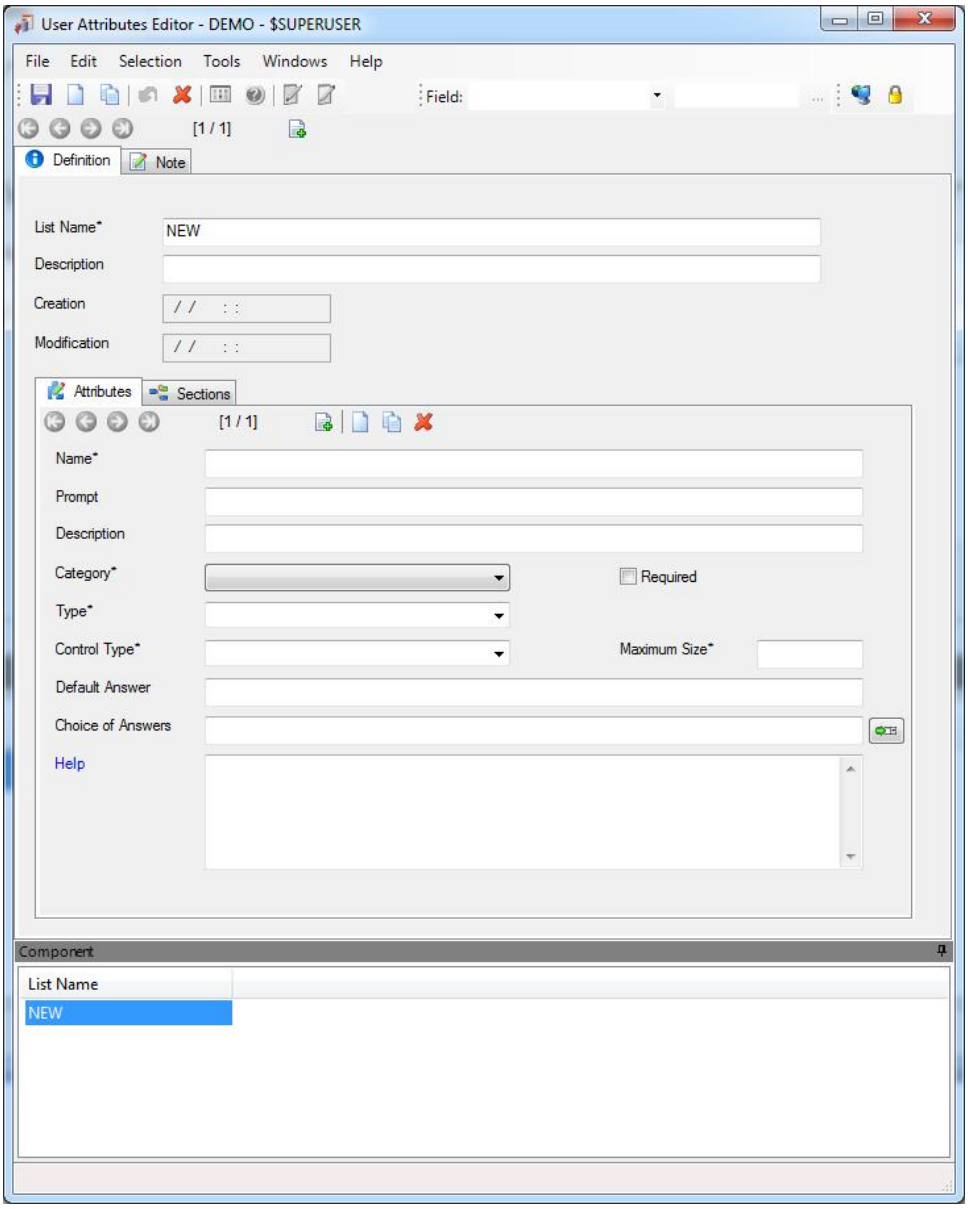


*Figure 1. The **User Attributes Editor**.*

When you create a UDA, you will enter the following information in the editor.

Table 1. Description of the fields of a UDA.

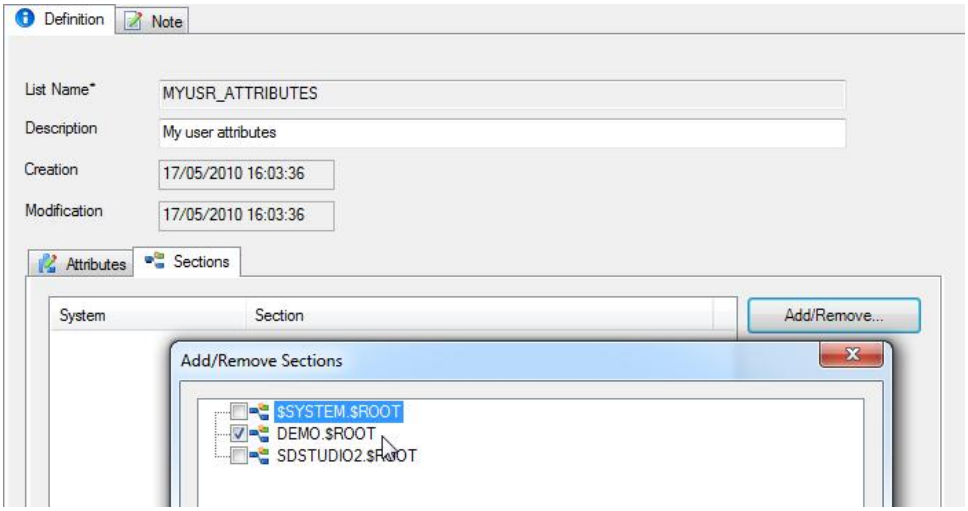
Field	Description
Name	ID of the UDA. Used mainly to access the UDA programmatically.
Prompt	Label of the UDA in REP++*studio.*
Description	Description of the UDA (for documentation purposes).
Category	REP++ object to which the UDA is associated (Field, Rowset, RowsetTreeDefNode, RowsetTree, etc.)
Type	Data type of the UDA value.
Maximum Size	Maximum size of the UDA value.

Field	Description
Required	Indicates if the UDA value must be specified. This is enforced by REP++*studio* when an item with an associated UDA is about to be saved.
Control Type	Type of the control that will be used to display and edit the UDA value.
Default Answer	Default UDA value.
Choice of Answers	List of choices for the UDA value.
Help	Help text to be displayed in REP++*studio* when the user presses F1 while the UDA control is the active control.

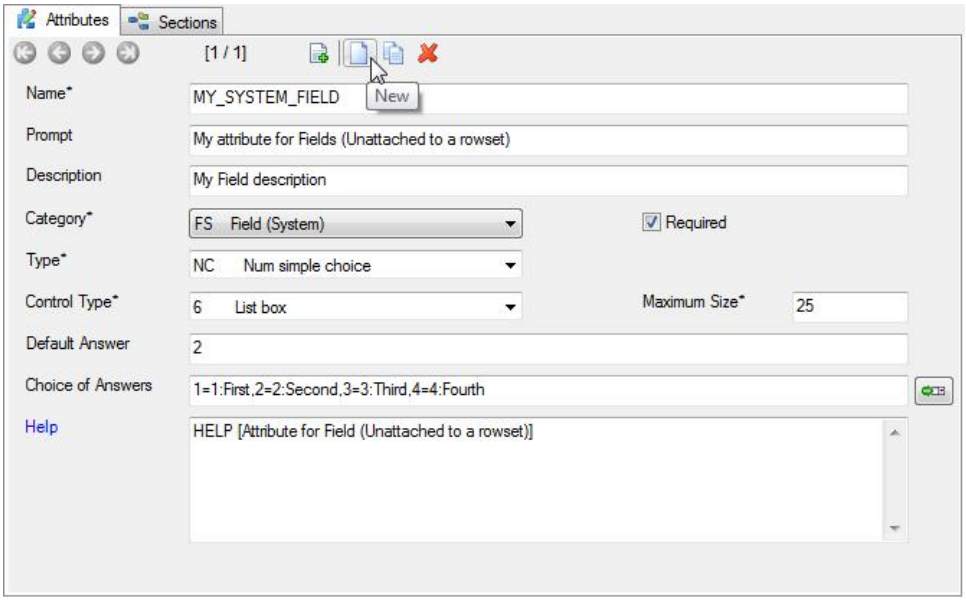
To illustrate the concept, you will create several UDAs in the contact management application.

To create a UDA

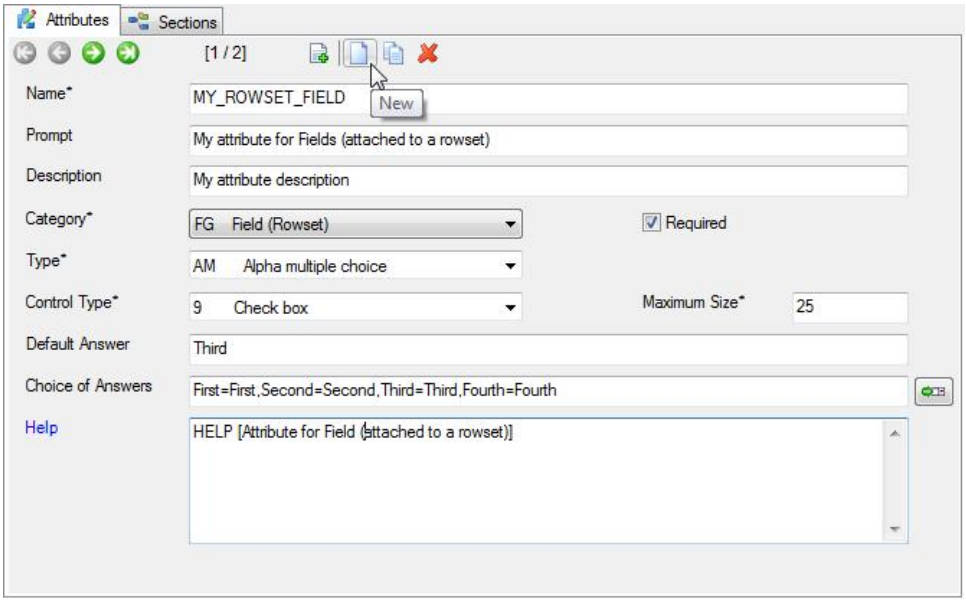
1. Connect to the repository of the contact management application.
2. Launch the **User Attributes Editor**.
3. Create a new list named MYUSR_ATTRIBUTES.
4. Click the **Sections** tab. UDA definitions are grouped in lists that can be associated with one or more systems and sections.
5. Select the DEMO.$ROOT section. Click **OK**.



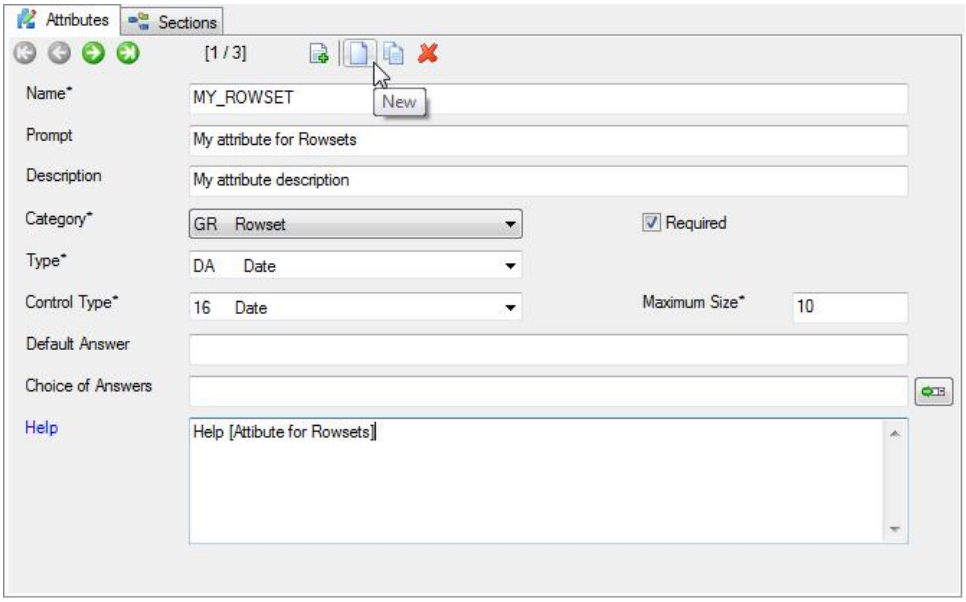
6. Click the **Attributes** tab.
7. Click the **New** button on the **Attributes** toolbar. Enter the information as illustrated in the following figures.
   - Create a UDA for a field not attached to a Rowset.



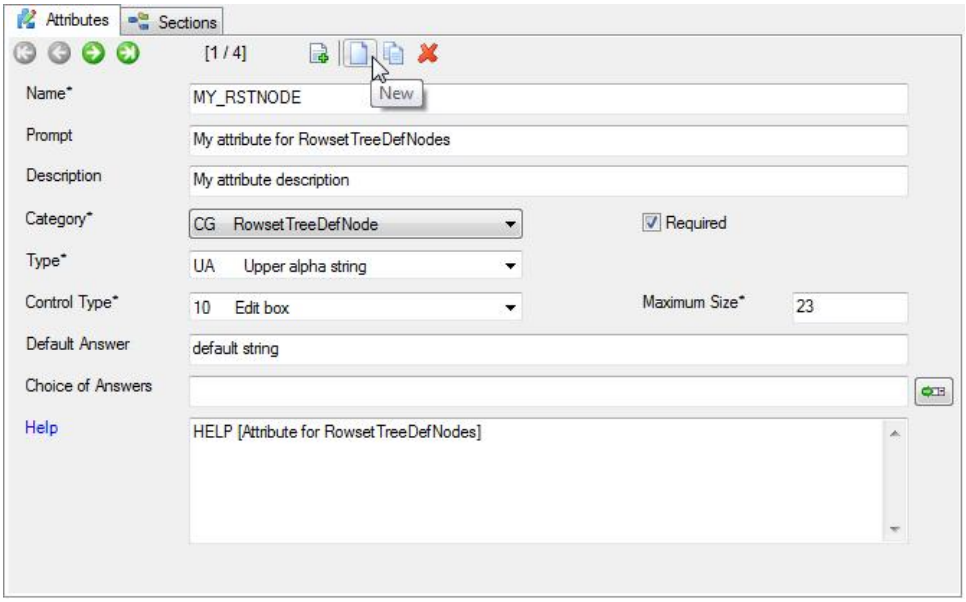
- Create a UDA for a field attached to a Rowset.



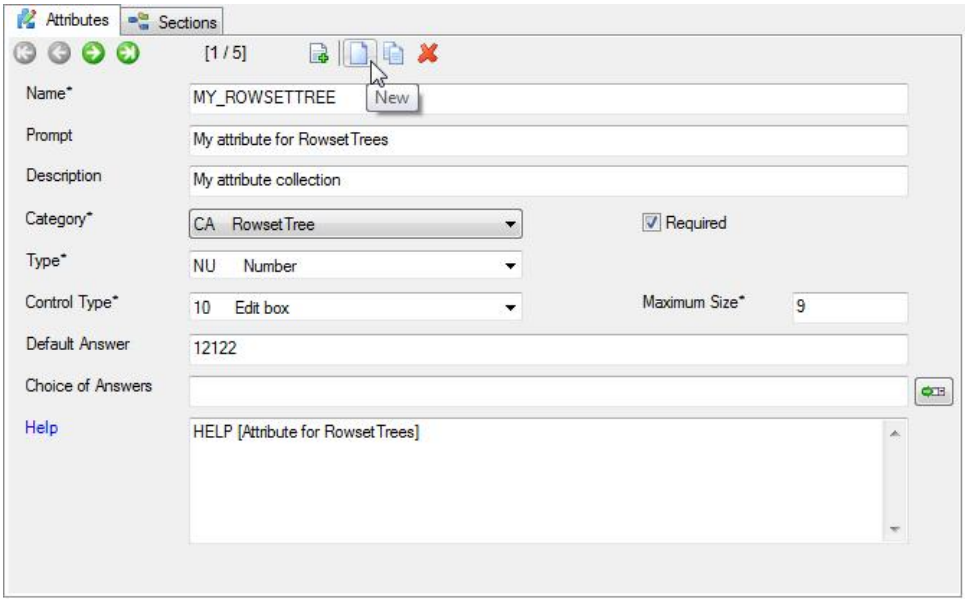
- Create a UDA for a Rowset.



- Create a UDA for a RowsetTreeDefNode.

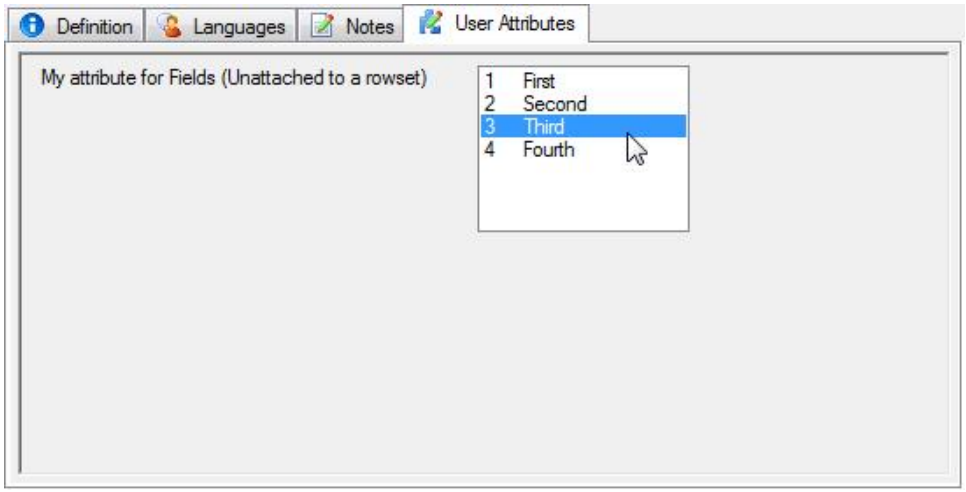


- Create a UDA for a RowsetTree.

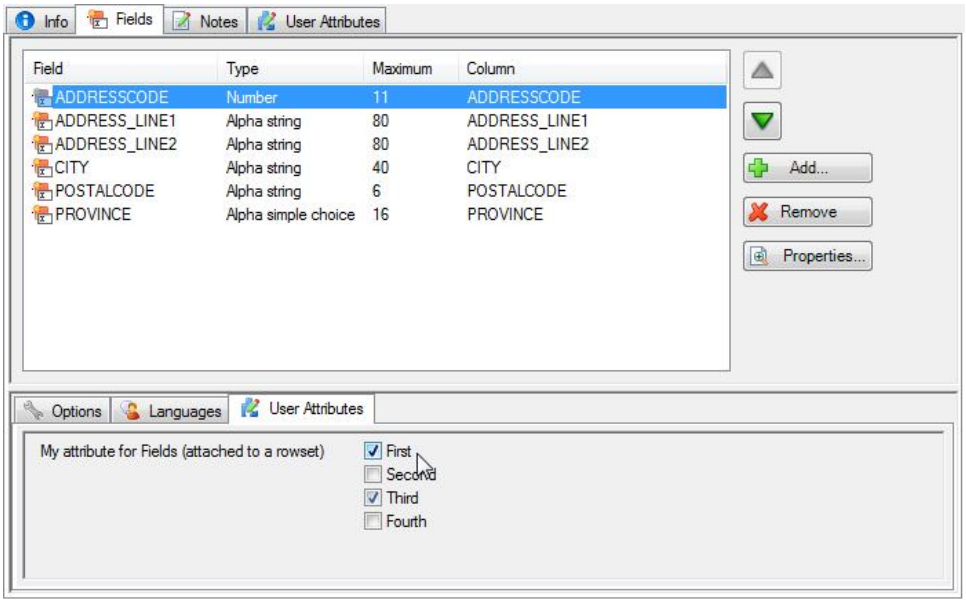


Once all the UDAs are created, save everything, close all REP++*studio* editors. Then start REP++*studio* again. Your UDAs now appear as if they were standard metadata of their associated REP++ objects.

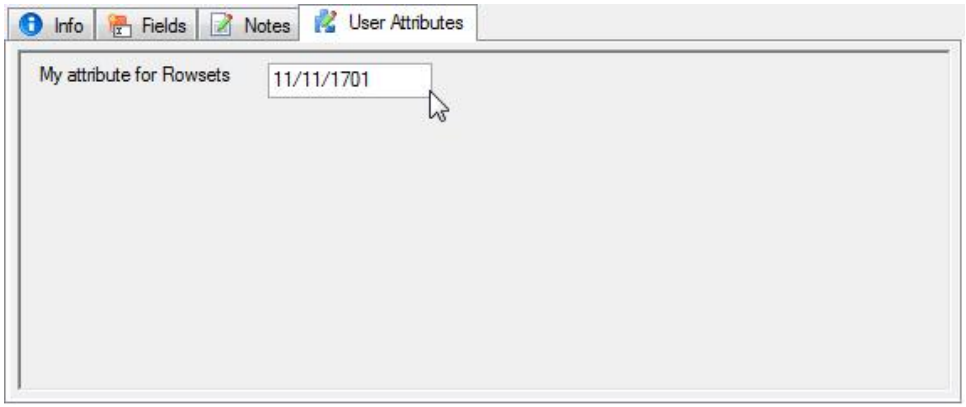
- User attributes for fields not attached to a Rowset.



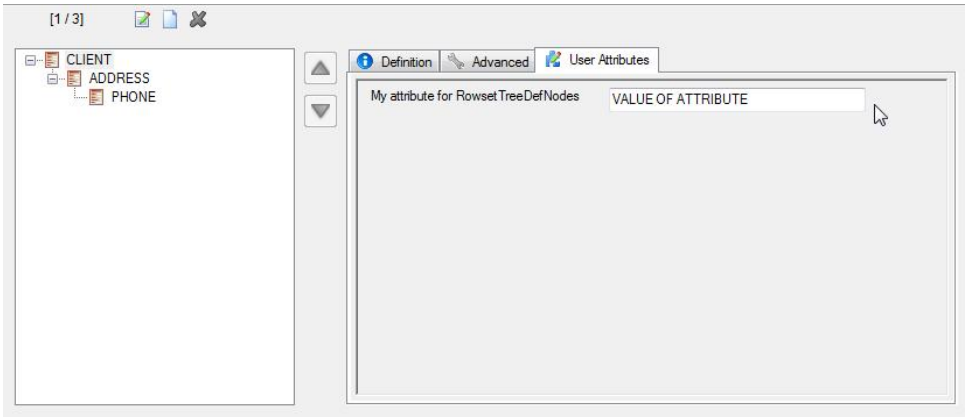
- User attributes for fields attached to a Rowset.



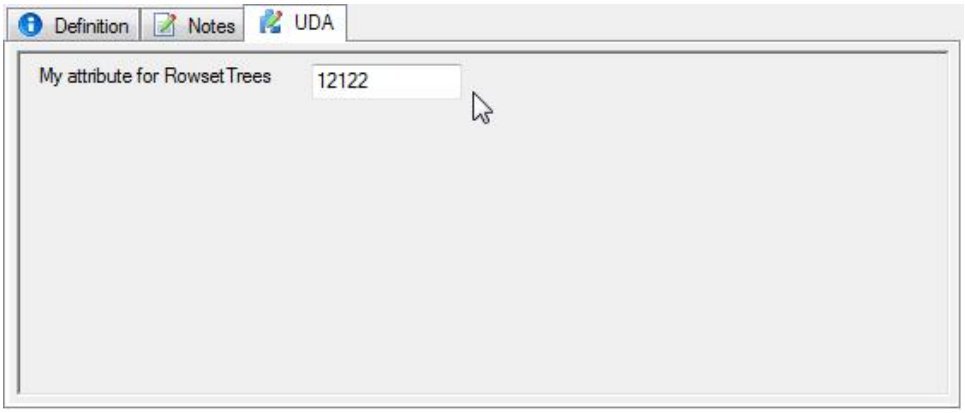
- User attributes for Rowsets.



- User attributes for RowsetTreeDefNodes.

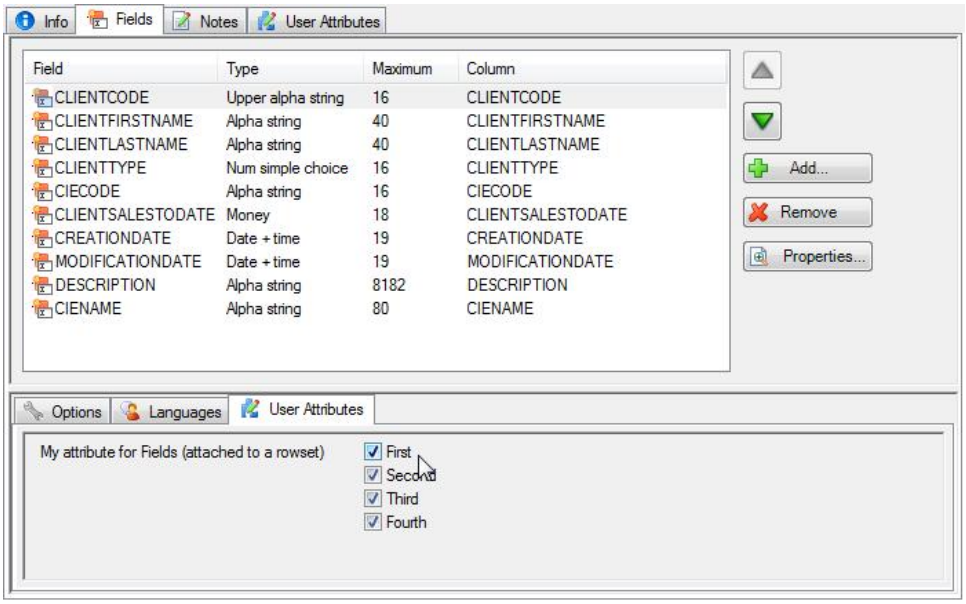


- User attributes for RowsetTrees.



Accessing the values of user-defined fields programmatically

Now that you have created our UDAs and set their values using REP++*studio*, you can access these UDAs programmatically and get or set the attribute values. First, set the UDA value of the CLIENTCODE field that is attached to the CLIENT Rowset by selecting all the attributes associated with the label *My attribute for Fields (attached to a Rowset)*. Remember, this attribute is of type *Alpha multiple choice*.



The value of the MY_ROWSET_FIELD UDA associated to the field CLIENT.CLIENTCODE should be "First,Second,Third,Fourth". To retrieve this value in your program, create a Windows® application project and add the following code to the generated project:

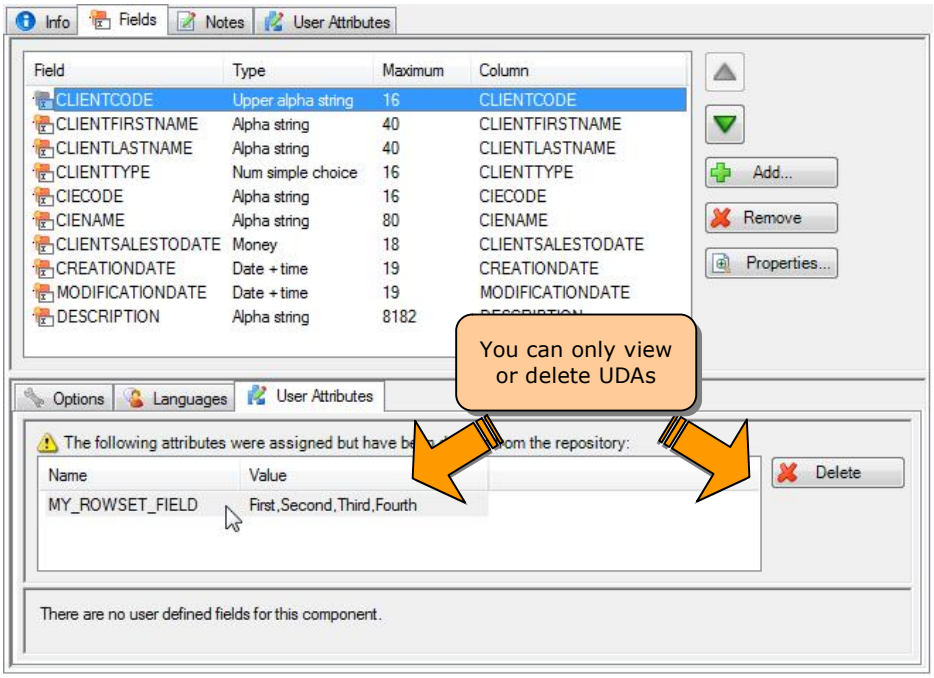
```
using System;
using System.Windows.Forms;

namespace Scrap {
    public partial class Form1 : Form {
        public Form1() {
            RepPP.Application   app;
            RepPP.FieldDef      fld;
            RepPP.Attribute     uda;
```

```
            InitializeComponent();
            app     = RepPP.Application.CreateFromRes();
            fld     = app.FieldDef("CLIENT", "CLIENTCODE");
            uda     = fld.Attributes["MY_ROWSET_FIELD"];
            if (uda != null) {
                MessageBox.Show(uda.Value);
            }
        }
    }
}
```

When you run your program, a message box displays the value of the MY_ROWSET_FIELD UDA associated with the CLIENTCLODE field within the CLIENT Rowset.

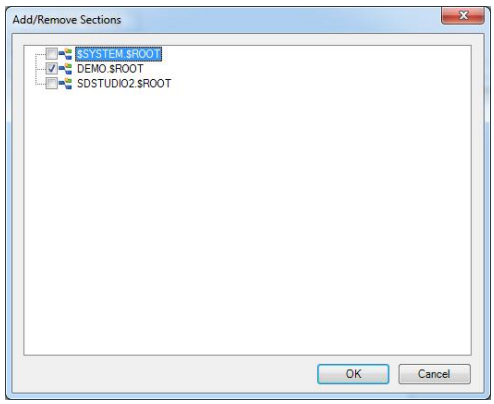
Deploying systems that include user-defined attributes

User-defined attributes are stored independently from the systems in which they are used in the repository. They are not automatically exported nor associated with them. That is why when you export a system that uses UDAs and then import it under a different name in the same repository or into a different repository, you need to explicitly re-associate your UDAs. Otherwise, you will only be able to view the current value of UDAs or delete the UDAs associated to the currently selected REP++ object, with no possibility to edit. REP++*studio* loads the definition of the UDAs associated to a section to create the user interface control used to edit its value. It then loads the value of the UDA in the control. If it cannot access the definition of the UDA in the section, REP++*studio* cannot create the control, and therefore no editing is possible.

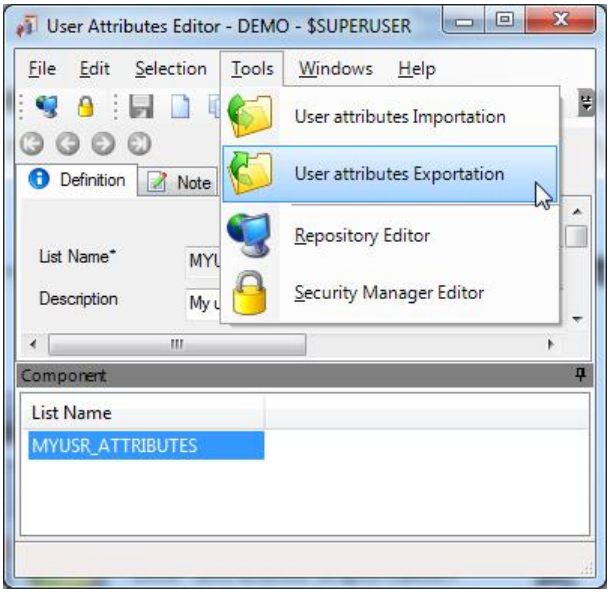


In order to be able to edit the values of the UDAs then make sure you do the following, according to your case:

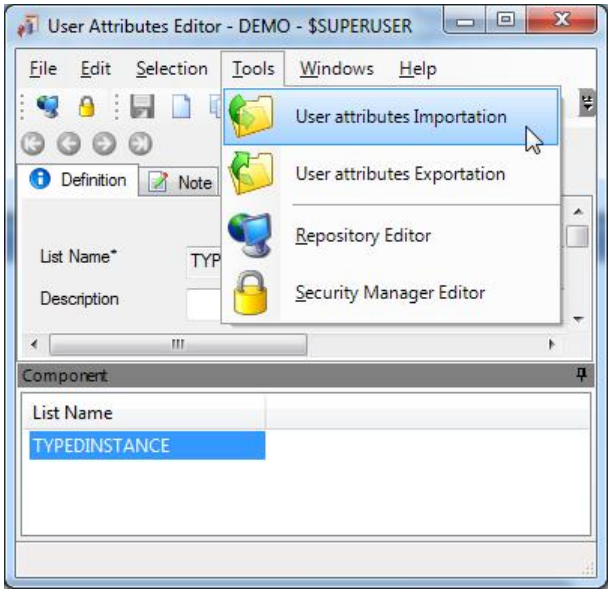
- **You imported the system under a different name in the same repository**. Since the UDAs are already stored in the repository, all you have to do is to associate the UDA list with new **Section** name.



- **You imported the system in a different repository**. The UDAs do not exist in the new repository, so they have to be exported from the original repository and imported in the new one.
    - Export your UDA list from the old repository using the **User attributes Exportation** command.



    - Import your UDA list into the new repository using the **User attributes Importation** command.



- Associate the UDA list with the sections that need the UDA.

Conclusion

You have seen how you can extend the metadata of the REP++ repository for certain REP++ objects. Most of the time, you will be able to accomplish your tasks simply using the standard metadata that is already available. There are no specific situations where UDAs should be used: it all depends on application requirements and your judgment as an application designer.