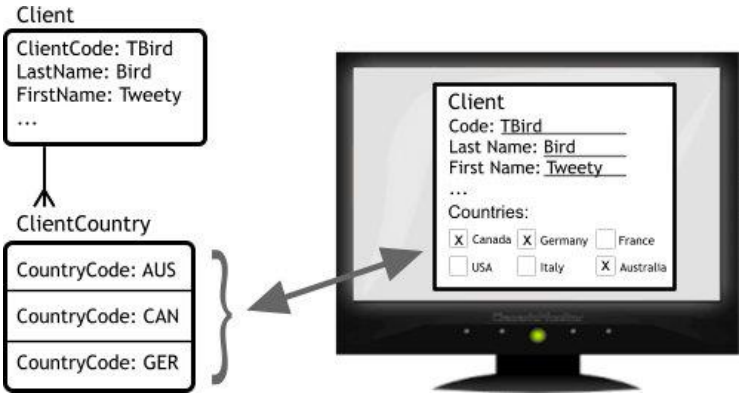
Technical Note

Implementing a choice list attached to a Rowset

Overview

There are situations when you may consider child rowsets as a set of selected items among a list. However there is no visual control in the user interface to display and change the selected items.

For instance, your clients distribute your products in different countries. The Client root rowset has any number of countries represented by lines in the ClientCountry rowset, each of which indicating a country where distribution is done for the client. You want to be able to select the countries among the entire list of countries, even though you cannot edit the ClientCountry rowset itself.



We defined methods that load and save the values of a field found in the lines of a child Rowset so you can use them in the parent Rowset. The list of items is built dynamically, and the selected items are retrieved from the lines of the child Rowset. Conversely, changing the selection in your choice list modifies the lines of the child Rowset.

This article describes how to implement this technique.

Implementing a choice list attached to a Rowset

Context

There are situations when you want to view and edit the content of a child rowset:

- that is not displayed in the application
- that includes a field whose value can be considered as an item of a list
- that contains several lines.

In the example described in the overview, we have a Client owset in a 0-n relation with its child ClientCountry rowset: each client can be associated with 0 or several countries, from a global list of countries saved in a database table.

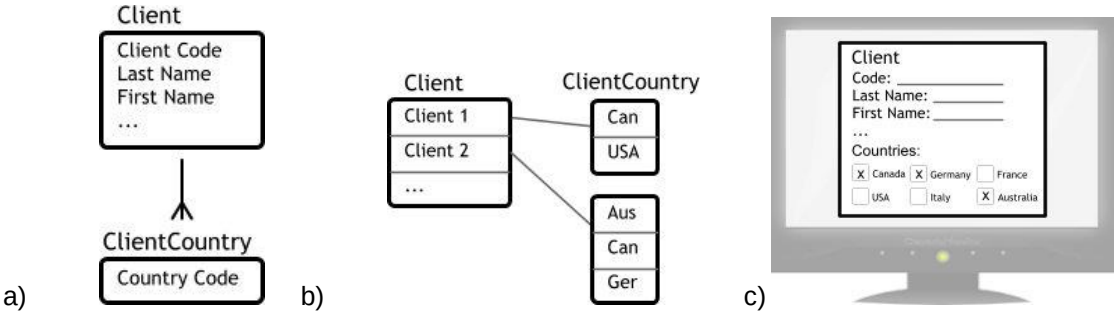


Figure 1. a) The Client – ClientCountry relation. Each client can be associated with none or several countries through the Client – ClientCountry relation. The ClientCountry rowset may contain other fields. The list of all possible countries for a client in the application is stored in the database. b) Each client may be associated with a set of countries. c) In the application, the countries associated with a client are checked.

The goal is to extract the value of the ClientCountry field from each line of the child rowset, display them as selected choices in a list, and save back the new selection in the child rowset after an edit.

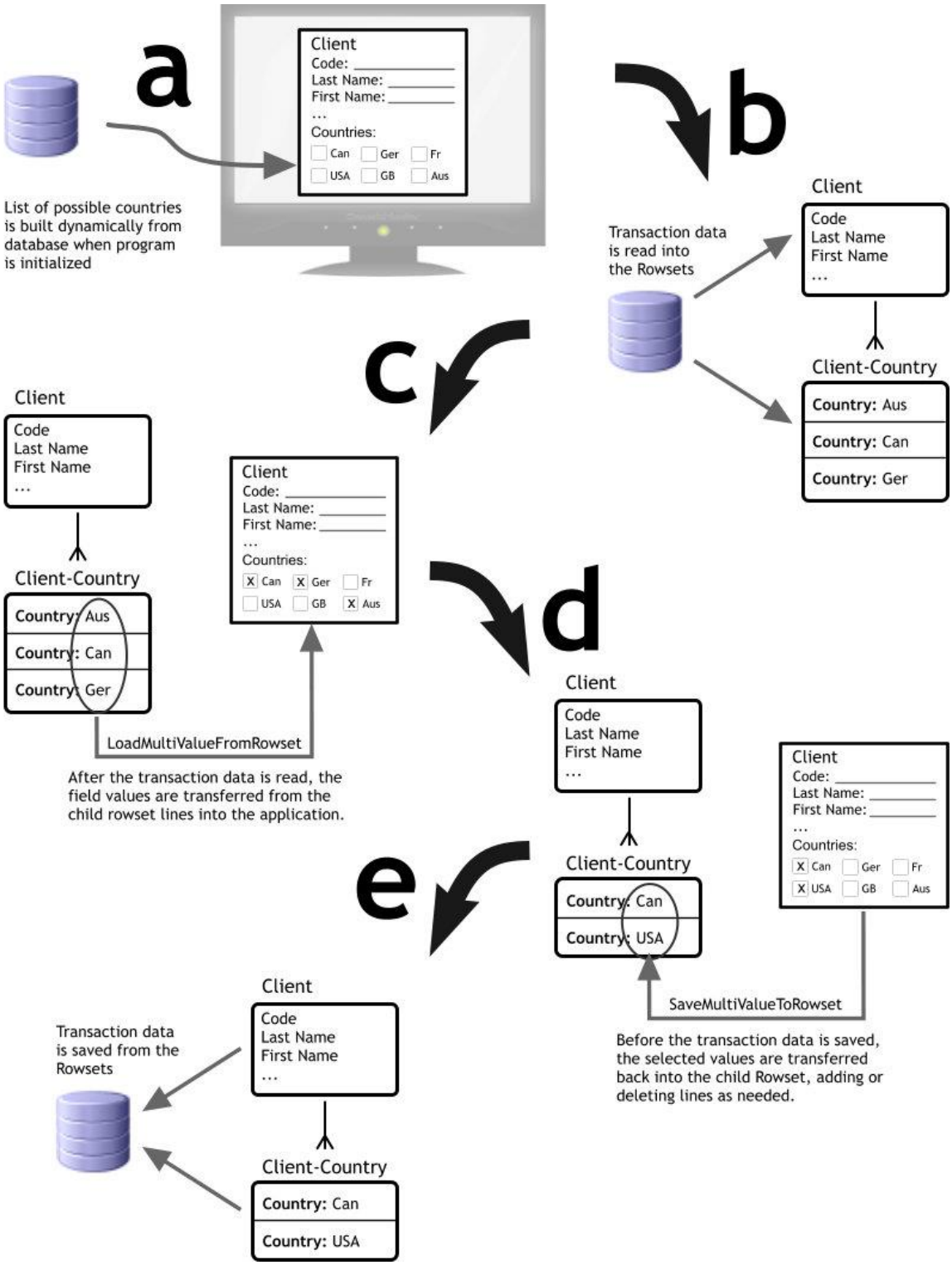
Two methods are defined to that end:

- FieldValue.LoadMultiValueFromRowset: retrieves field values from the child rowset lines as a choice list selection.
- FieldValue.SaveMultiValueToRowset: modifies the lines of the child rowset to reflect the end user selection.

To edit a list of choices extracted from a child rowset, the overall procedure is as follows:

a. List the possible country values in the form using, for instance, a set of check box controls.
b. Read the transaction data.
c. Retrieve the value of the country field in each line of the child rowset and select the corresponding countries in the interface.

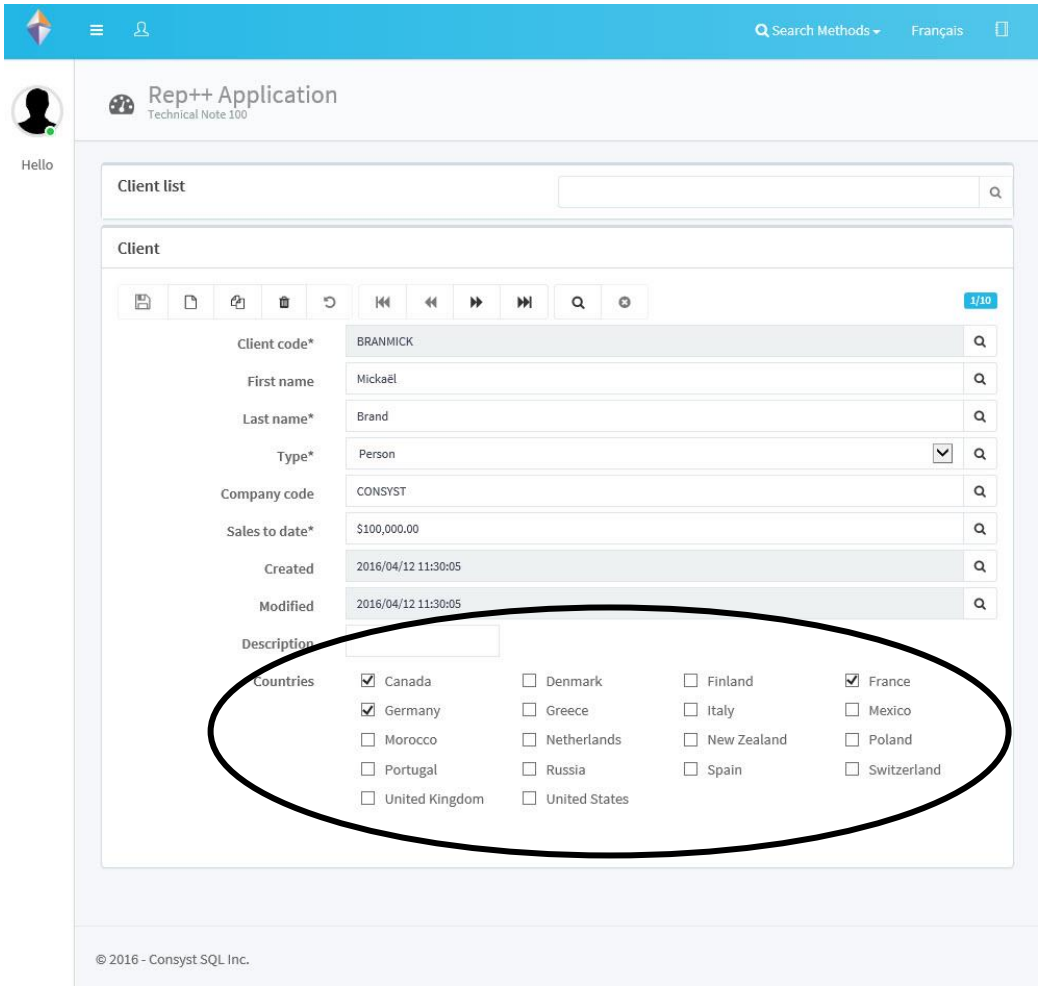
d. After an edit, save the selected set of countries back in the child rowset.
e. Save the transaction data.



To illustrate this feature, you will create a field to display the list of countries and add it to your main Client form. The list is defined as a user-managed field, built dynamically but not linked to

the database. Its only purpose is to display the list of available countries and to reflect the country values found in the lines of the child ClientCountry Rowset in the following way:

- If you check a country in the list, a line will be inserted in the ClientCountry Rowset with the corresponding country value.
- If you clear a country from the list, the corresponding line in the ClientCountry Rowset will be removed.



Sample system

A sample system designed to demonstrate this capability, TECHNOTE100, is included with the Rep++ installation. The information is found in three tables, TN100_CLIENT, TN100_CLIENTCOUNTRY and TN100_COUNTRY. As described above, the RowsetTree consists of the CLIENT root Rowset and the CLIENTCOUNTRY child Rowset.

a)

b)

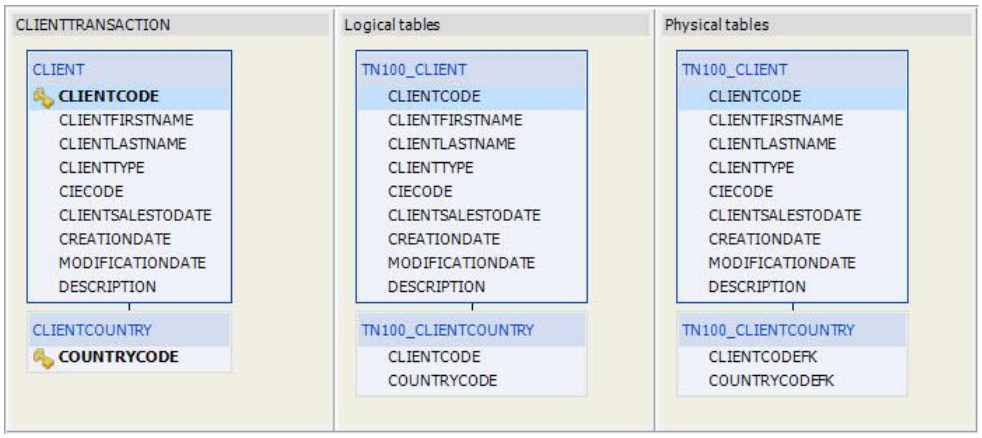


Figure 2. a) The ClientTransaction RowsetTree in Rep++ studio consists of the Client and ClientCountry Rowsets. b) The RowsetTree Viewer depicts the tables for the RowsetTree.

The functionality will be tested in a single-page application (SPA), where you will insert the code to load and save your country selections from/to the ClientCountry Rowset.

In summary, you will:

1. Create the tables in the repository and import the TECHNOTE100 system.
2. Create a choice list field, in Rep++ studio, to hold the country values and include it in your root Rowset.
3. Create a new SPA application in Visual Studio® using the Rep++ wizard.
4. Insert the code to load from and save to the ClientCountry Rowset.

Prerequisites

Rep++ installed (includes Rep++ studio and SD Tools). Working knowledge of Rep++. SQL Server as database.

Create tables for the TECHNOTE100 system

The tables for this system will be added to the repository for the chosen connection.

To create the tables

1. Open SD Tools.
2. Open the connection where you will import your system.
3. On the **File** menu, click **Open**.
4. Choose the Demo/Technotes/Technote100_Srv.sql file under the Rep++ installation folder.
5. On the toolbar, click **Execute**.

Your tables have been added. You can now close SD Tools.

Import the demonstration system

To import the Technote100 system

1. Open Rep++ studio with the connection where the tables were created.
2. Import the Demo/Technotes/Technote100.sys system under the Rep++ installation folder.

Create the field to display the list of countries

The CountryList field is a user-managed field whose content is built dynamically using a SQL command from the content of the TN100_COUNTRY table in the database. You need to define the SQL command to load the list of countries, create the CountryList field, add it to your Client Rowset, and associate the field to the SQL command.

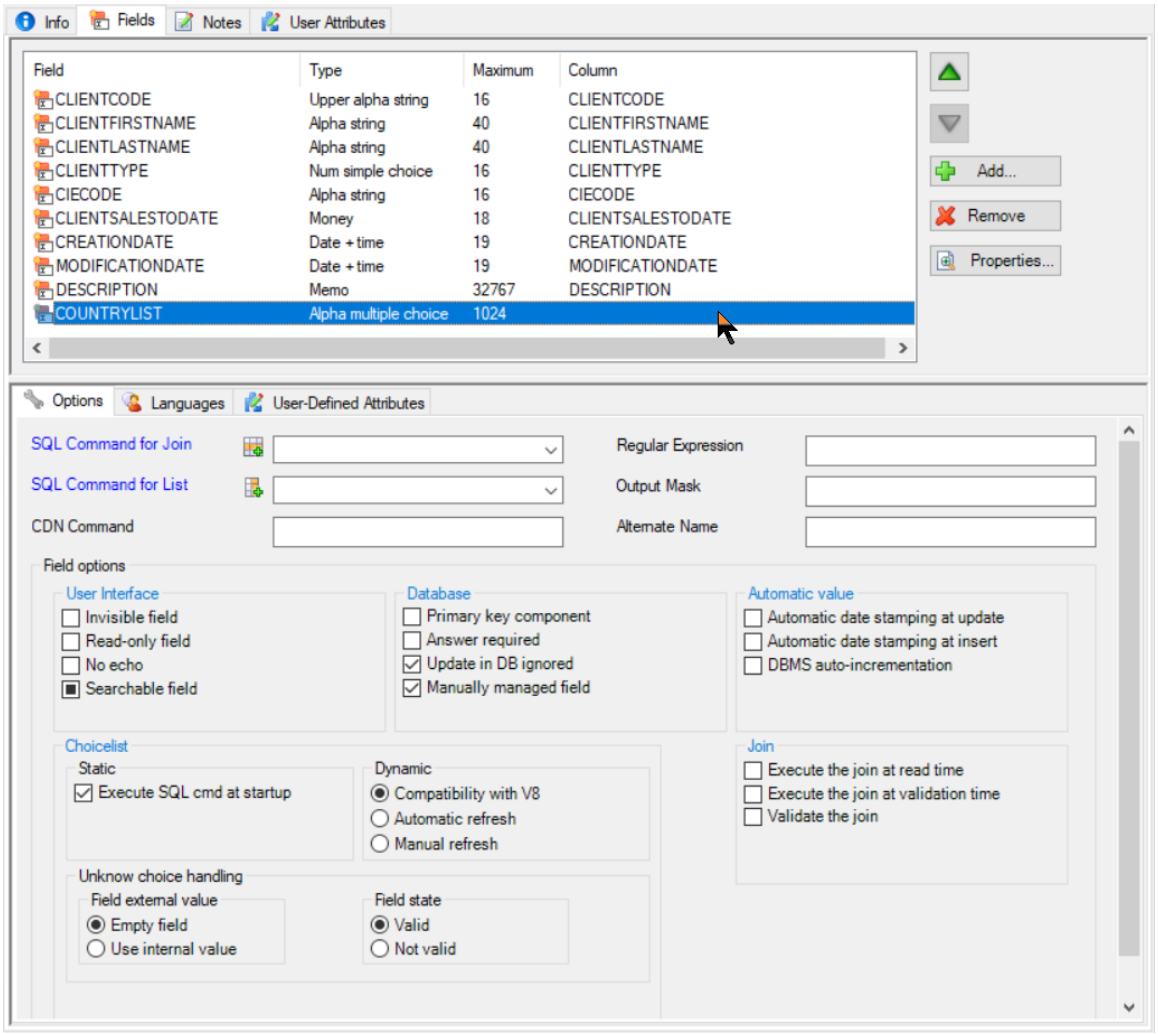
To create the list of countries choice list

1. In the SQL commands editor, add an SQL command atom named List_Countries that contains the following code:

```
SELECT TN100_COUNTRY.COUNTRYCODE, TN100_COUNTRY.COUNTRYNAME
INTO :$CODE, :$DESC
FROM TN100_COUNTRY
ORDER BY TN100_COUNTRY.COUNTRYNAME
```

2. Save your SQL command.
3. In the Fields editor, add a field named COUNTRYLIST with the following attributes:
    - **Type**: Alpha multiple choice
    - **Control Type**: Check box
    - **Min Size**: 0
    - **Max Size**: 1024
    - **Prompt** (in **Languages** pane, select *English*): Countries
    - **SQL command for list**: List_Countries
4. Save your field.
5. In the Rowsets editor, select the Client Rowset.
6. Add the COUNTRYLIST field to the list of fields:
    a. Click **Add**.
    b. In the *Add Fields* window, click the **All Fields** option.
    c. Select COUNTRYLIST and move it to the **Selected** column.
    d. Click **OK**.
7. In the **Options** pane of the Rowsets editor, check the following 3 options for the COUNTRYLIST field:
    - **Update in DB ignored**
    - **Manually managed field**
    - In the **Choice list** group, make sure that:
        - **Compatibility with V8** option is selected.
        - **Execute SQL cmd at startup** is checked.

8. Save your Rowset.



Create Rep++ SPA application to test your functionality

Using the Rep++ wizards, you will create a Rep++ SPA application that will include your new proxy choice list field.

To create the SPA program

1. In Visual Studio®, create a standard ASP.NET Web Application (not Rep++).
2. In the *New ASP.NET Project* window, select the **Rep++ MVC V8** template.
3. In the Rep++ Application wizard's *Connect to the Rep++ repository* page, select the connection, TECHNOTE100 system and TECHNOTE100 program.
4. In the Select *Rep++ repository components* page, select the *ClientTransaction* and *ClientSelectTransaction* RowsetTrees for the transaction and selection buffer, respectively.
5. In the *Select n-tier options* page, leave the default values.
6. In the *Specify generation information for typed instances* page, leave the default values.
7. In the *Specify generation information for POCO entities* page, clear **Generate POCO entities**.
8. In the *Select template* page, select *MVC SPA using Angle Template*.

9. In the *Select layout options* page, do:
    a. Choose the ClientCountry node.
    b. Under **Include node**, click **Do not generate node**.
10. In the *Select replacement mode* page, click **Finish** to create the application.
11. In the *Search for TypeScript Typings* message box, click **No**.

If you now build and run your program, you'll see the list of countries displayed under the client information, retrieved using your SQL command attached to the COUNTRYLIST field. However no countries are selected: the information is located in the ClientCountry Rowset, which is not visible. Furthermore, if you check countries and save, nothing is saved. The next section shows you how to connect the COUNTRYLIST field with the information from the ClientCountry Rowset.

Incorporate the logic for using the choice list field

Each line of the ClientCountry Rowset contains the country code associated with the current client. For each country code found, the goal is to check the appropriate country in the COUNTRYLIST field in your main client form. Conversely, the set of checked countries should be reflected in the lines of the ClientCountry Rowset if the end user changes it.

To display the selected countries, you will override the CCDbAfterReadRowsetTree custom method and use it to call the Field.LoadMultiValueFromRowset method right after loading the Client data. The field values from the lines of the ClientCountry Rowset will be displayed in the proxy field COUNTRYLIST. The SQL command for choice list that you associated with the proxy field will translate the country code found in the ClientCountry Rowset into the country name.

To load the selected countries from the ClientCountry Rowset

1. In the Solution Explorer, expand the **Models** node, then open the RTCClientTransaction.cs file.
2. Expand the Business Model region.
3. Override the CCDbAfterReadRowsetTree custom method as follows:

```
public override void CCDbAfterReadRowsetTree(RowsetTreeEventArgs e) {

    rsClient.fldCOUNTRYLIST.LoadMultiValueFromRowset(rsClientCountry.fldCOUNTRYCODE,
                                                     false);
    base.CCDbAfterReadRowsetTree(e);
 }
```

The Field.LoadMultiValueFromRowset method loads the combined values of the field fldCOUNTRYCODE from the lines of the ClientCountry Rowset, and displays them in the COUNTRYLIST multiple choice list field.

When you make changes to the list, you need to record the new selection in the lines of the ClientCountry Rowset. The SaveMultiValueToRowset method is used to that end.

To save the selected countries to the ClientCountry Rowset

1. In the RTCClientTransaction.cs file, override the CCDbBeforeUpdRowsetTree as follows:

```
public override void CCDbBeforeUpdRowsetTree(RowsetTreeEventArgs e) {

    rsClient.fldCOUNTRYLIST.SaveMultiValueToRowset(rsClientCountry.fldCOUNTRYCODE);
    base.CCDbBeforeUpdRowsetTree(e);
 }
```

2. Rebuild your solution.

If you run your application, you will now be able to see, modify and save the different country selections.