
Array, CSV and Data Binding

Contents

| | |
|---|----|
| Import CSV File..... | 2 |
| CSV File..... | 2 |
| Import into EasyGrid | 2 |
| Set Column Visual Styles | 5 |
| Export to CSV File | 7 |
| Import and Export Columns..... | 11 |
| Export to column arrays..... | 12 |
| Import column arrays | 14 |
| Test..... | 15 |
| Import and Export Rows | 17 |
| Export row arrays..... | 18 |
| Import row arrays | 20 |
| Test..... | 22 |
| Use single column of data..... | 24 |
| Export one column of data | 24 |
| Import one column of data | 26 |
| Test..... | 28 |
| Use single row of data | 30 |
| Export one row of data | 31 |
| Import one row of data..... | 33 |
| Test..... | 35 |
| Use single elements | 37 |
| Get Element | 37 |
| Set single element..... | 40 |
| Test..... | 43 |
| Sample: List all days between two dates | 44 |
| Feedback | 59 |

Import CSV File

CSV File

Each line in a CSV (Comma Separated Values) file is one row of data. In a row, columns are separated by commas. Optionally the first line of a CSV file can be a list of column names.

Below is a CSV example.

```
ColumnX,ColumnY,ColumnZ
```

```
Some text,123,true
```

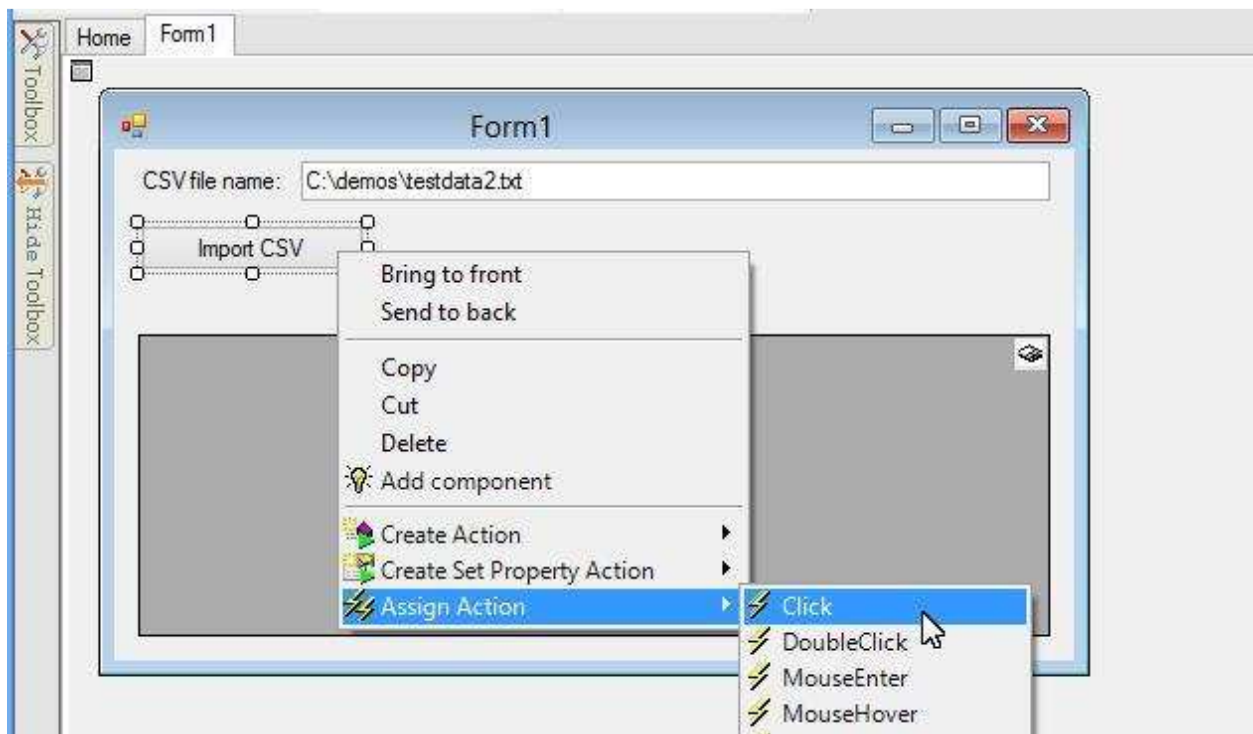
```
Hello World,100,false
```

```
"one, two, three",200,true
```

In this CSV file, the first line is a list of column names. The first column contains text data. The second column contains numbers. The third column contains Boolean data. You can see that commas are allowed in text data as long as the text data are double quoted. You may also use `&!2C` to represent a comma in text data.

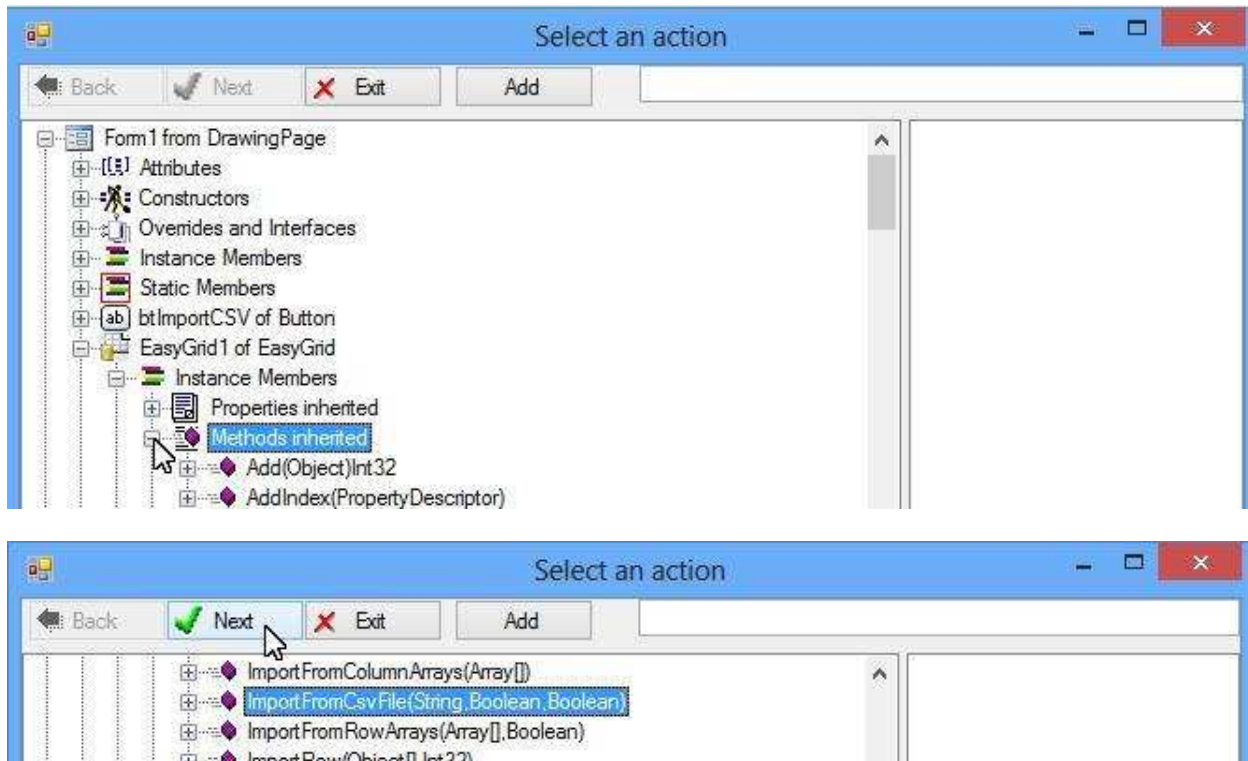
Import into EasyGrid

Let's import this CSV file into an EasyGrid:

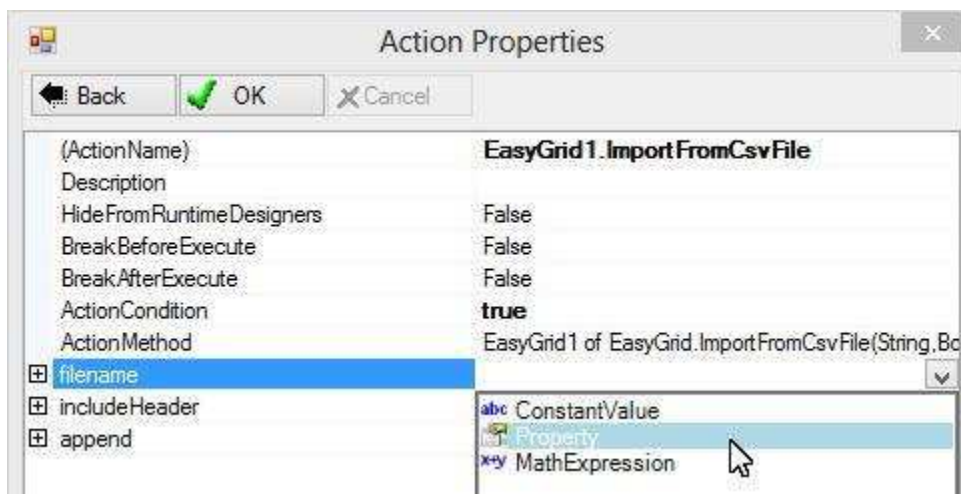


Array, CSV and Data Binding

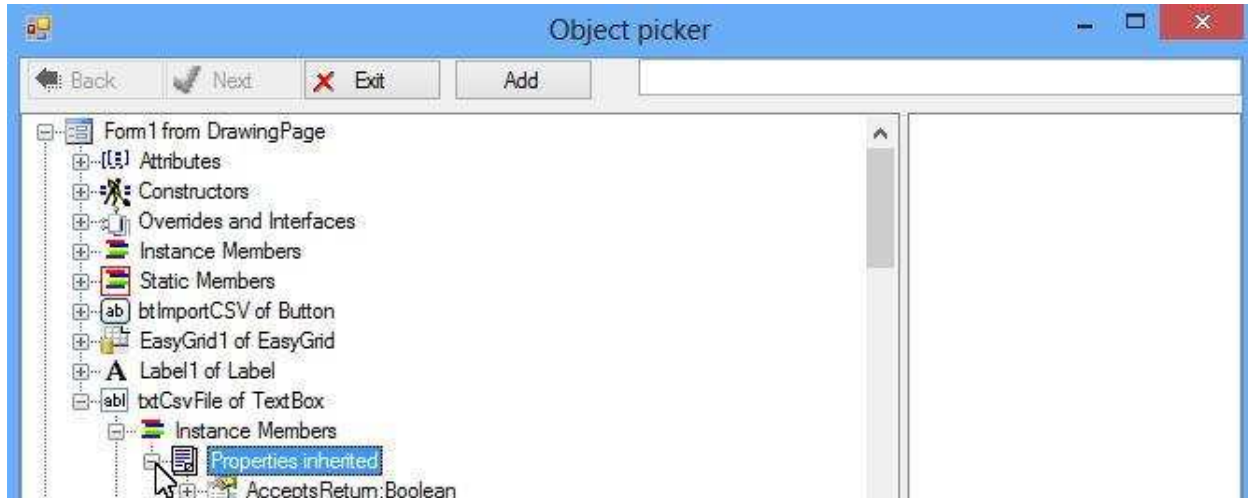
Select ImportFromCsvFile method of the EasyGrid:



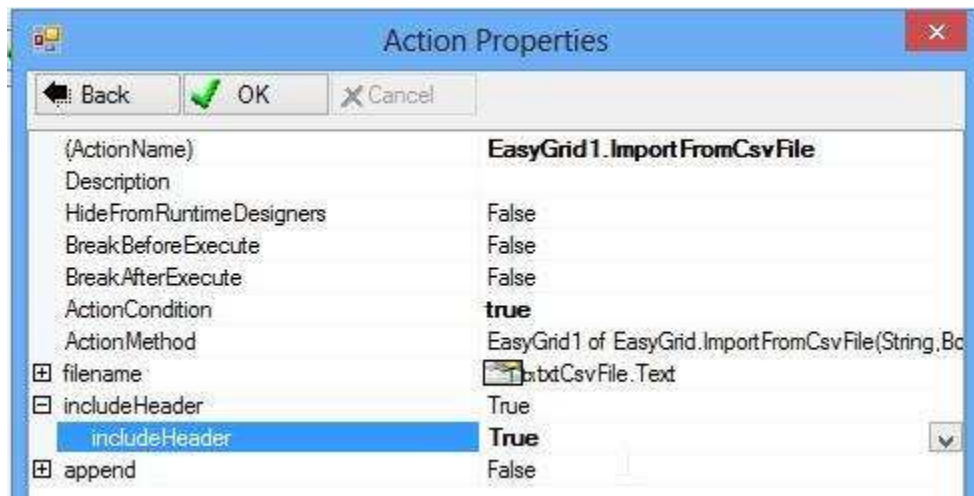
Select the Text property of the text box for the file name:



Array, CSV and Data Binding

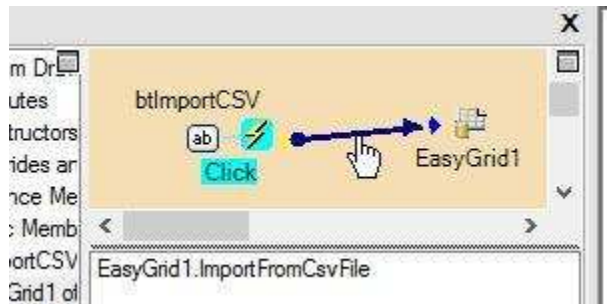


Set includeHeader to True because the first line of our sample CSV is column names:

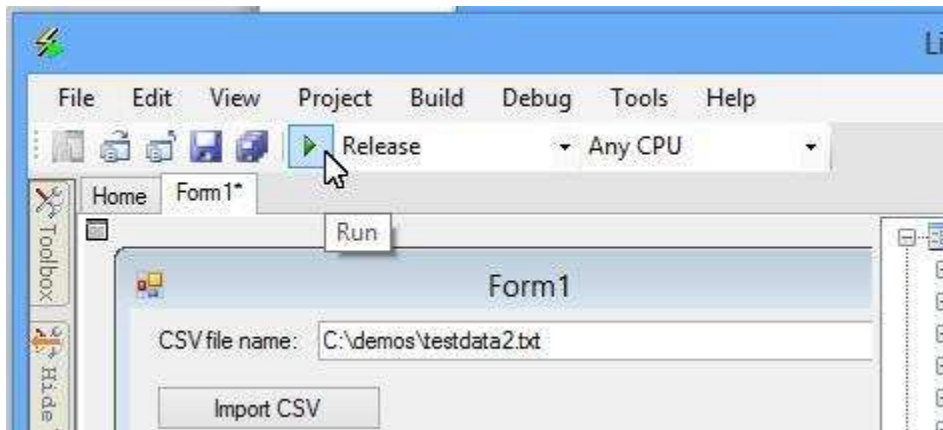


Click OK. The action is created and assigned to the button:

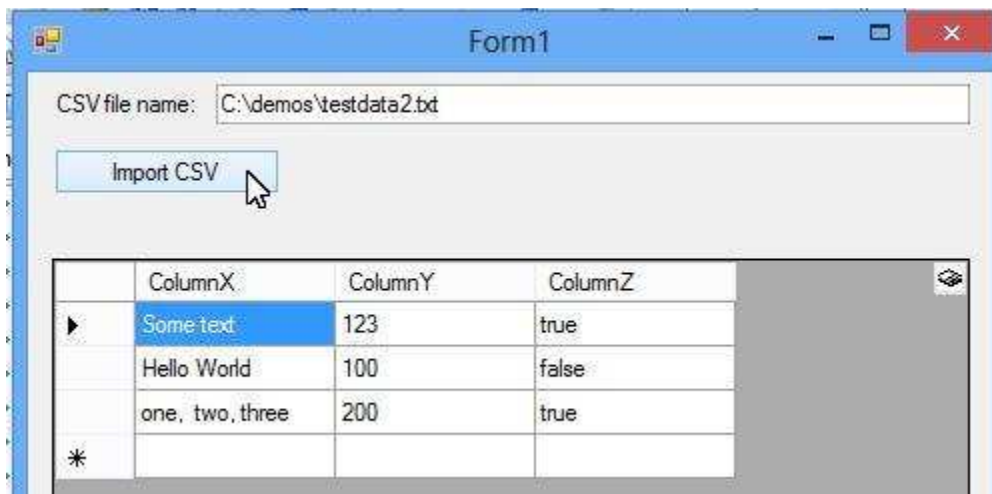
Array, CSV and Data Binding



Run the application:



Click the button. The data from the CSV file appear in the EasyGrid:



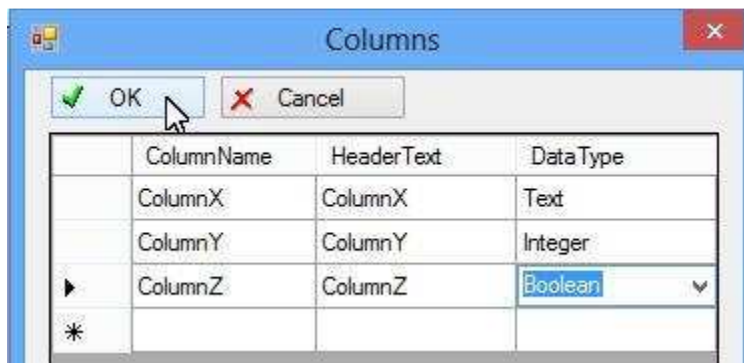
Set Column Visual Styles

We may set Columns property of the EasyGrid to set column display properties.

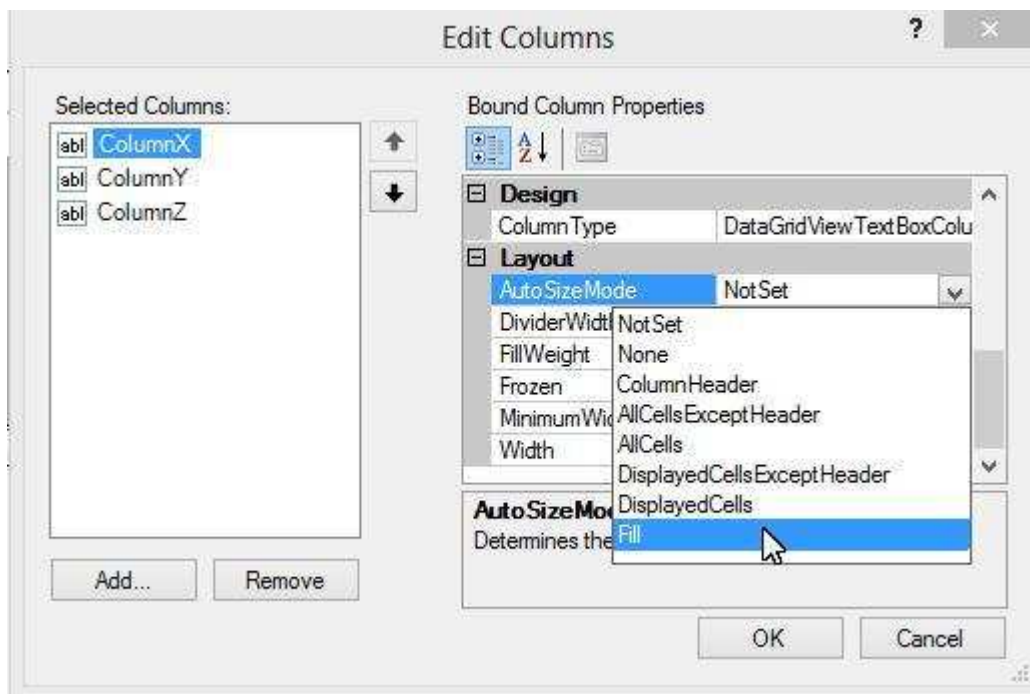
Array, CSV and Data Binding



A dialogue box appears letting you add columns. In this dialogue, the column names must match the names from the CSV file, if the CSV file contains column names.



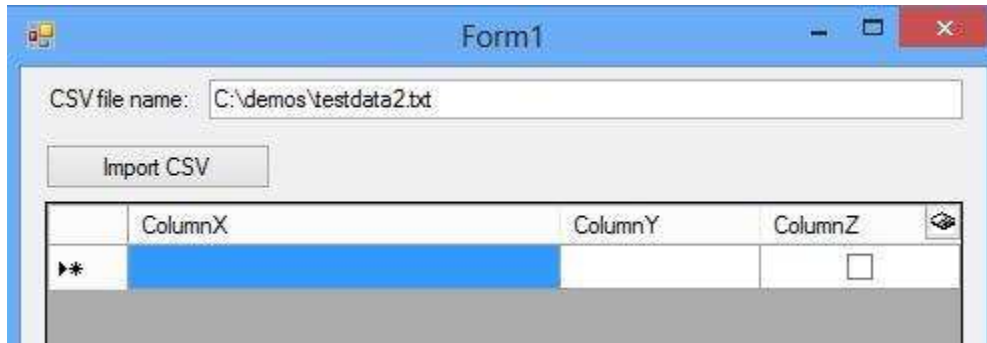
Note the data types we specified. Click OK; it allows us to specify more properties for each column. For example, we may let the first column to automatically fill the grid:



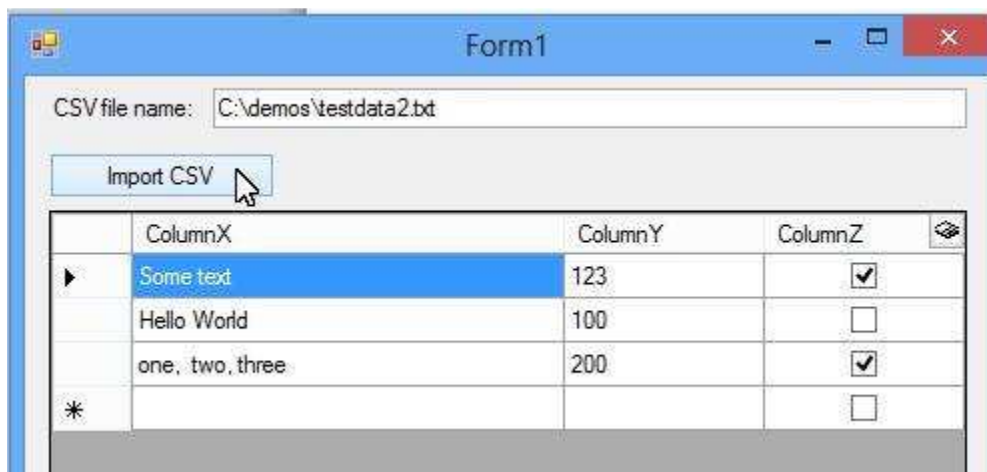
Array, CSV and Data Binding

The above dialogue is from .Net Framework. It has a bug: clicking “Add...” button will show a dialogue box for adding new columns; but that dialogue will not work. So, to add new columns, you must use the dialogue box we showed before.

Let’s run it again to click the import button.



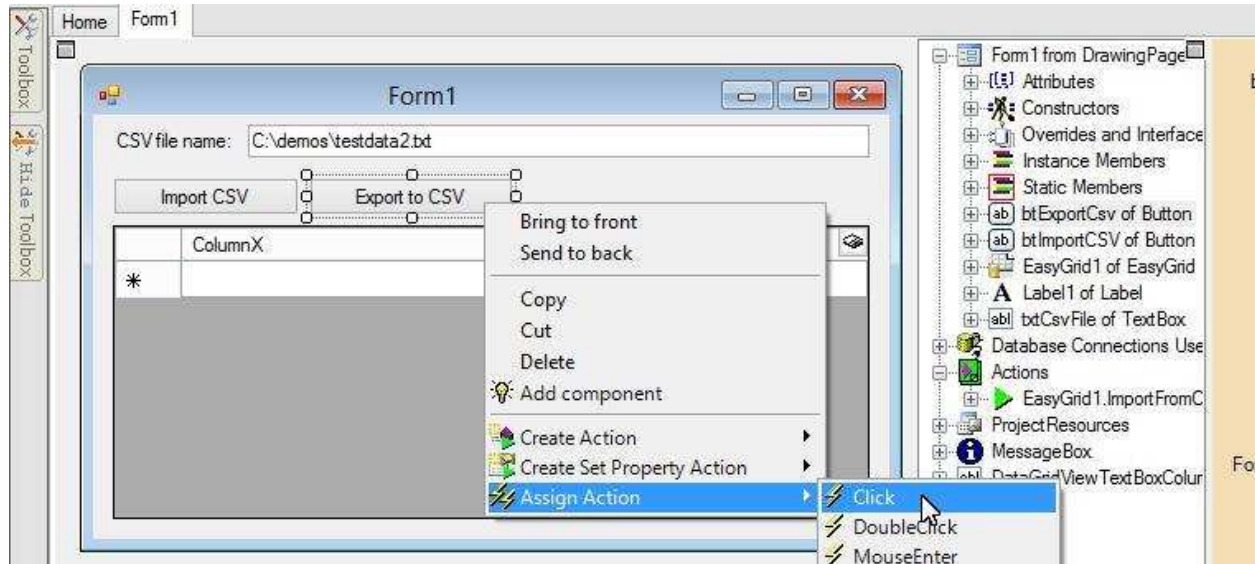
Click the button to import the CSV file. You may compare the display result with previous sample.



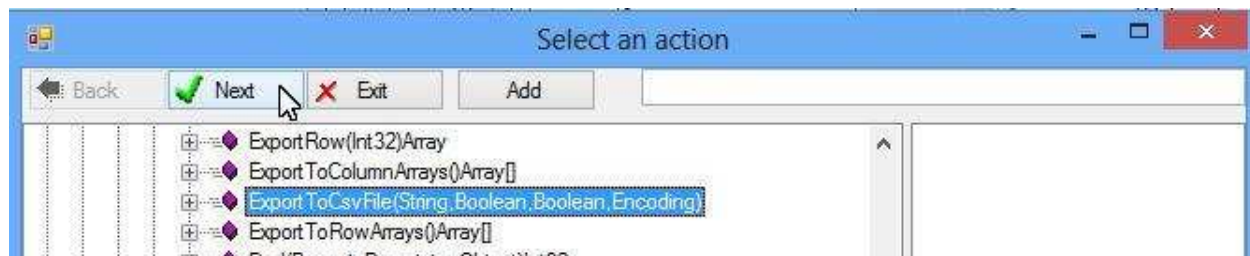
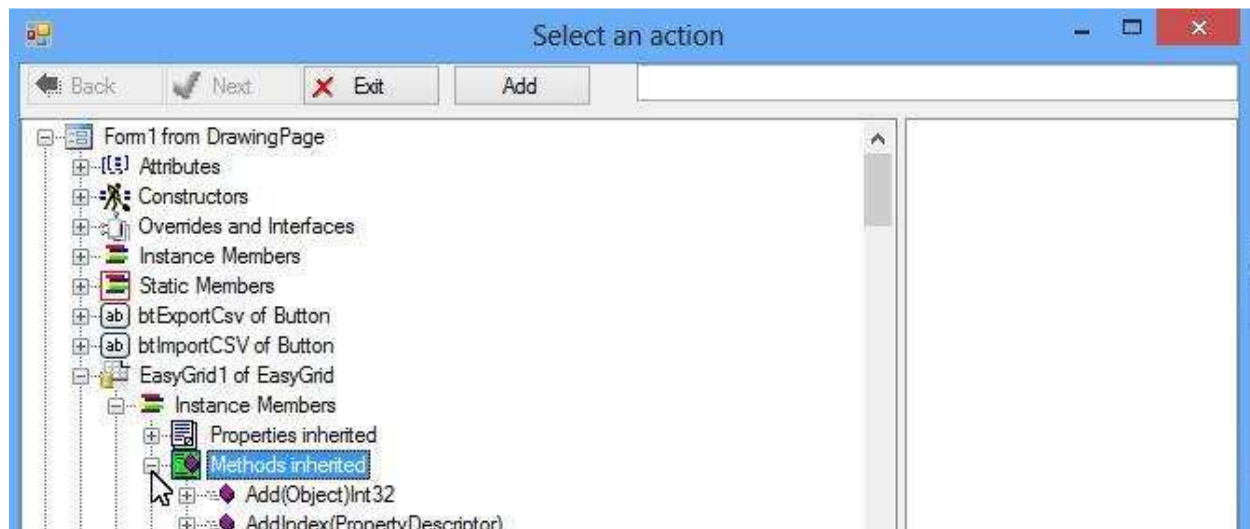
Export to CSV File

Let’s use a button for exporting data in the EasyGrid to a CSV file.

Array, CSV and Data Binding

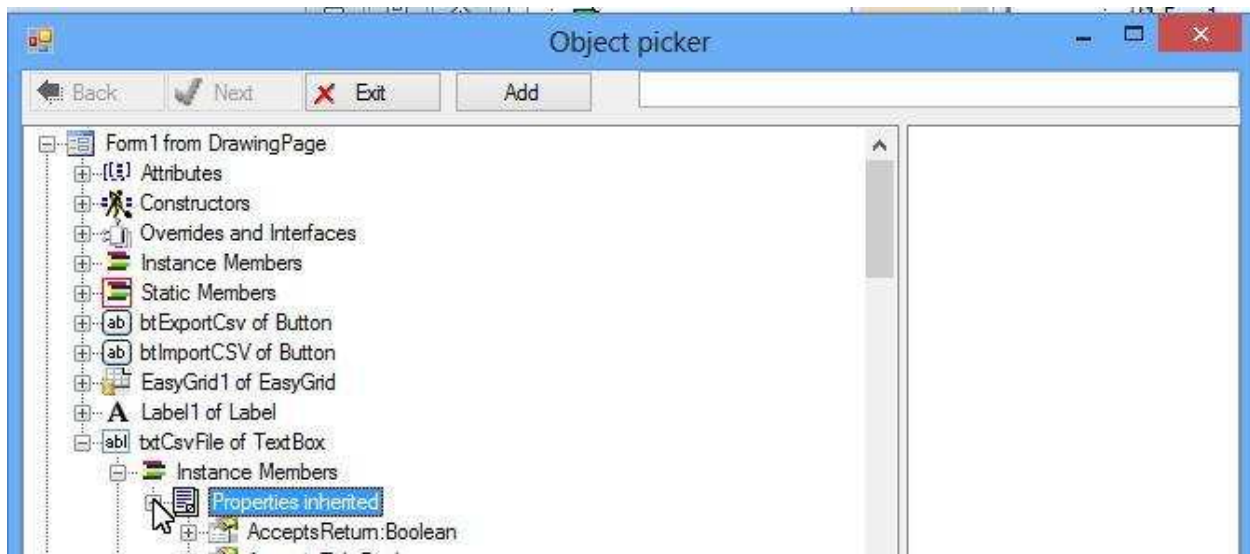
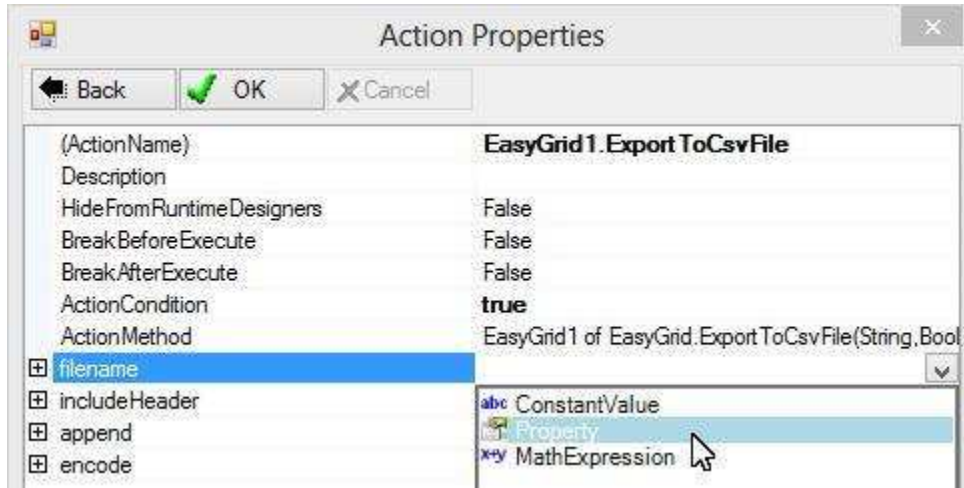


Select ExportToCsvFile method of the EasyGrid:



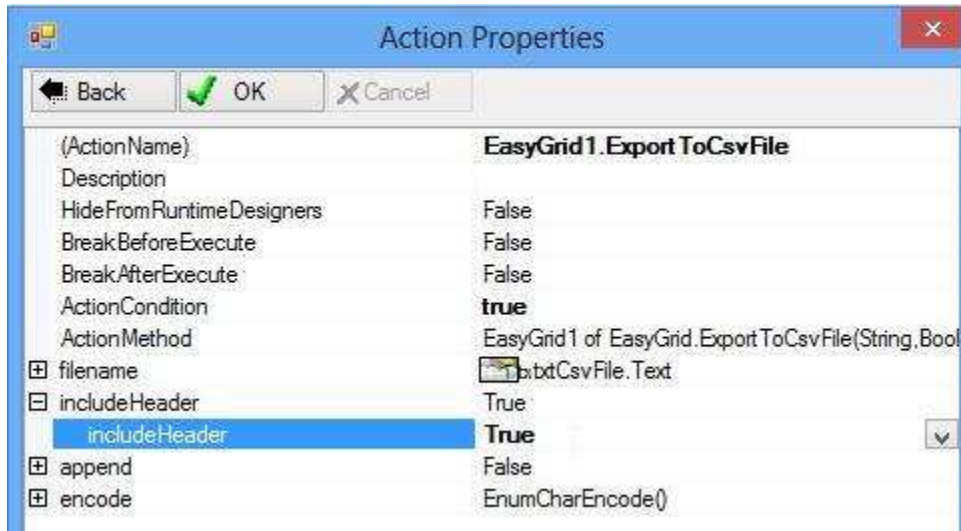
Select the Text property of the text box as the CSV file name:

Array, CSV and Data Binding

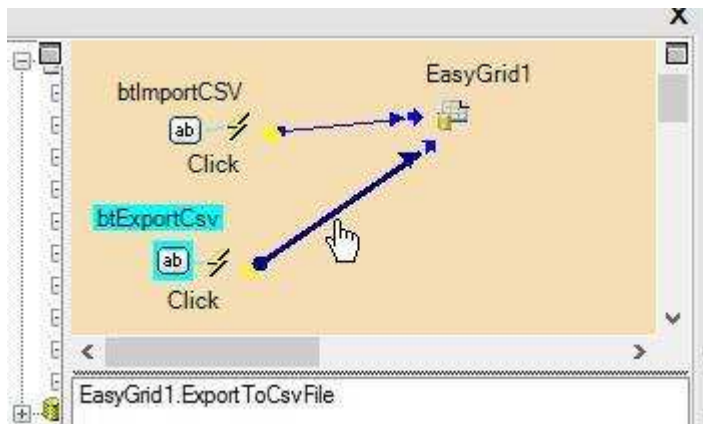


Include column names:

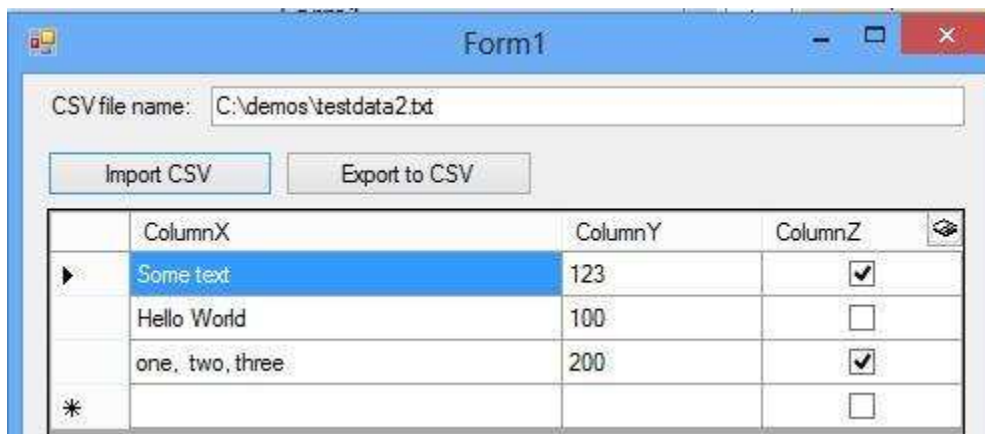
Array, CSV and Data Binding



Click OK. The action is created and assigned to the button:

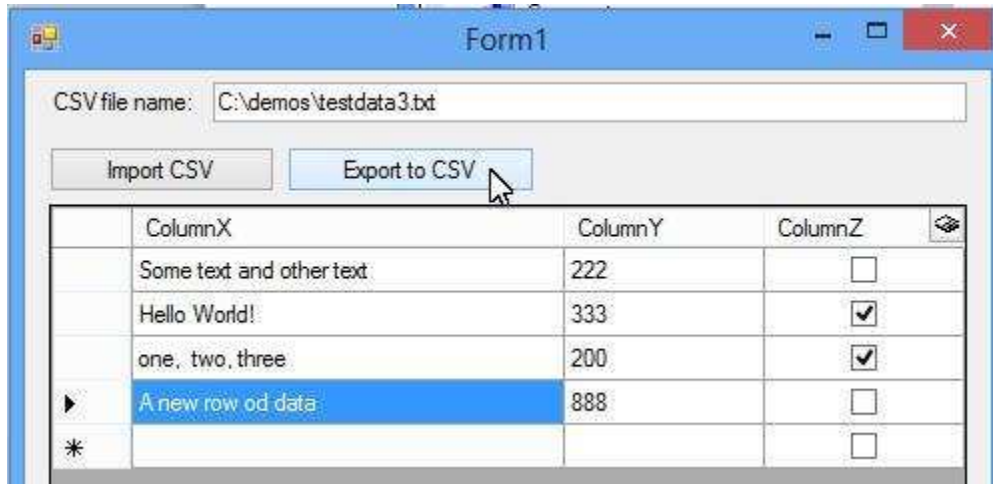


Run the application. Load data from a CSV file.

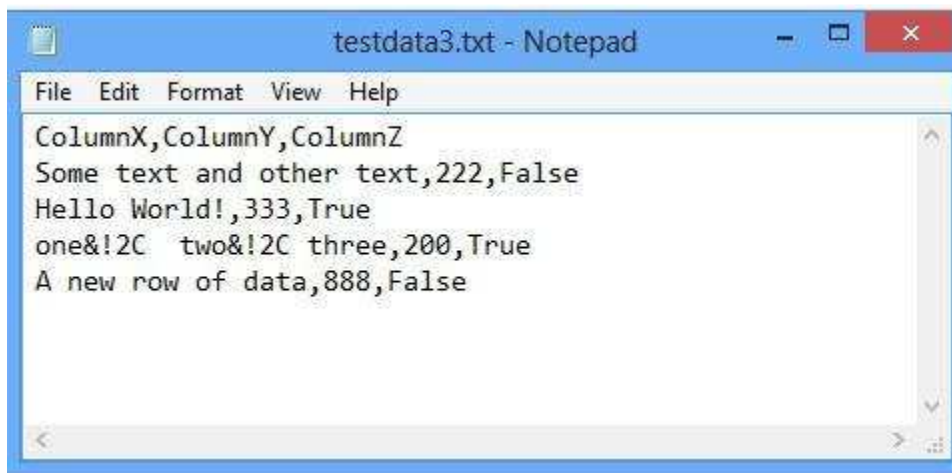


Modify data in the EasyGrid. Set the file name to testdata3.txt. Click "Export to CSV".

Array, CSV and Data Binding



A new text file is created. We may open it to view its contents:



Import and Export Columns

Method `ExportToColumnArrays` of `EasyGrid` returns an array of arrays. The first array contains data for the first column; the second array contains data for the second column; and so on.

Method `ImportFromColumnArrays` of `EasyGrid` imports data from an array of arrays. The first array contains data for the first column; the second array contains data for the second column; and so on.

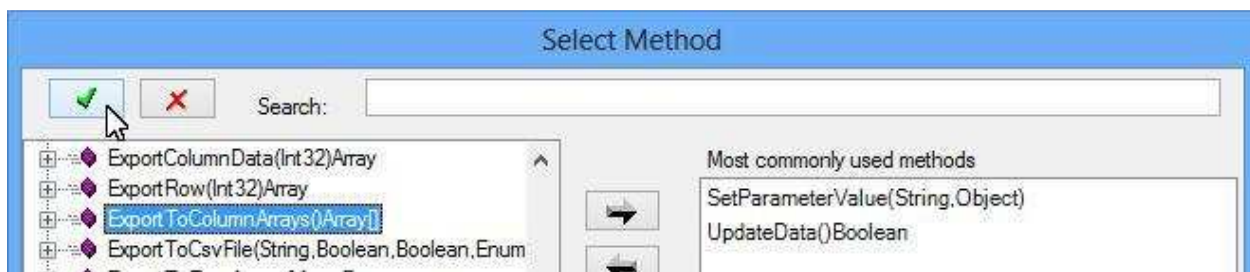
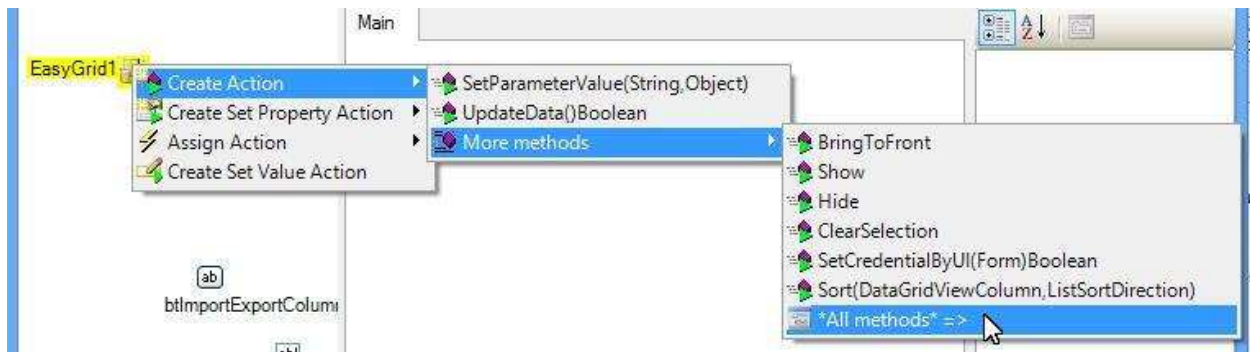
We create a new method to demonstrate the use of `ExportToColumnArrays` and `ImportFromColumnArrays`.



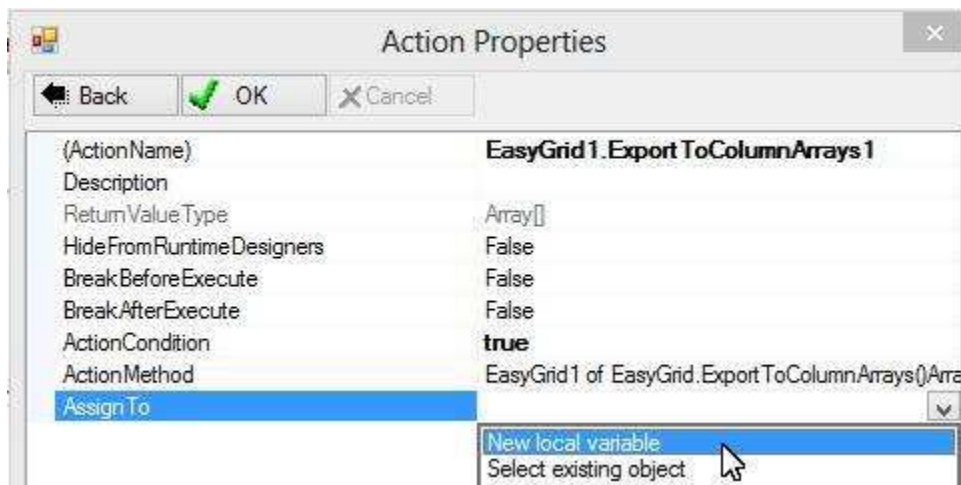
Array, CSV and Data Binding

Export to column arrays

Create an action to export data to column arrays:

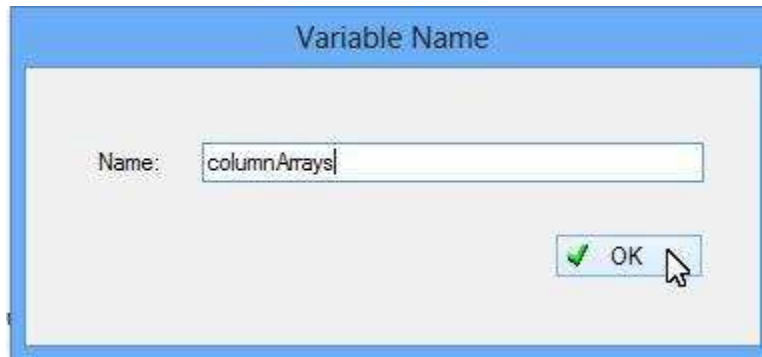


Choose "New local variable" to use a new variable for the action result:

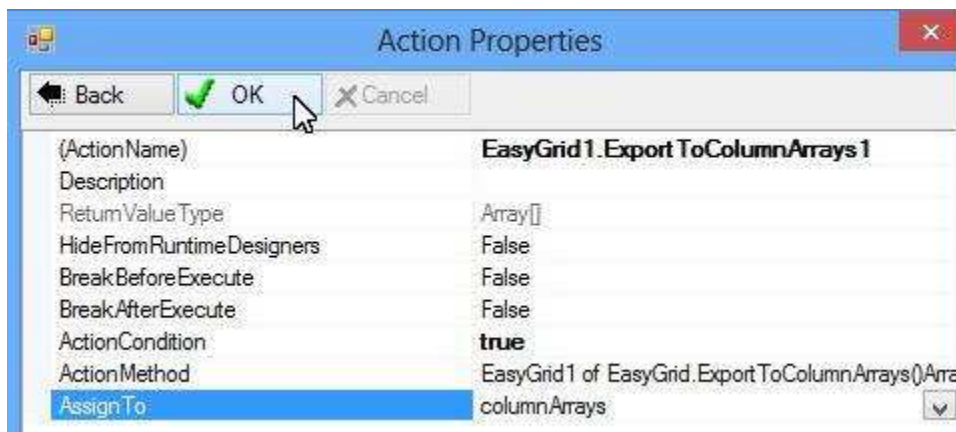


Name the new variable "columnArrays":

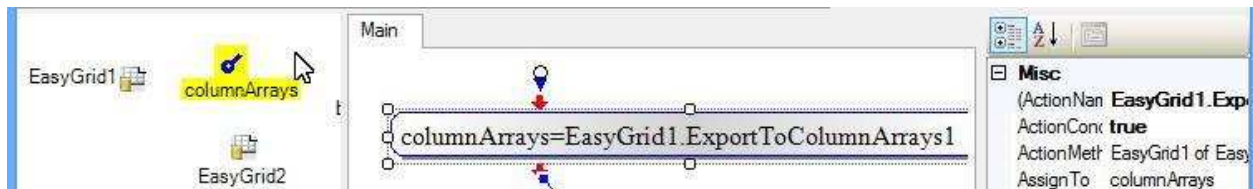
Array, CSV and Data Binding



Click OK to create the action:



An action and a variable are created:



The variable, columnArrays, contains one array for each column. If our sample CSV file, testdata2.txt, is imported then columnArrays will contain 3 arrays.

The first array:

```
Some text  
Hello World  
one, two, three
```

The second array:

```
123  
100  
200
```

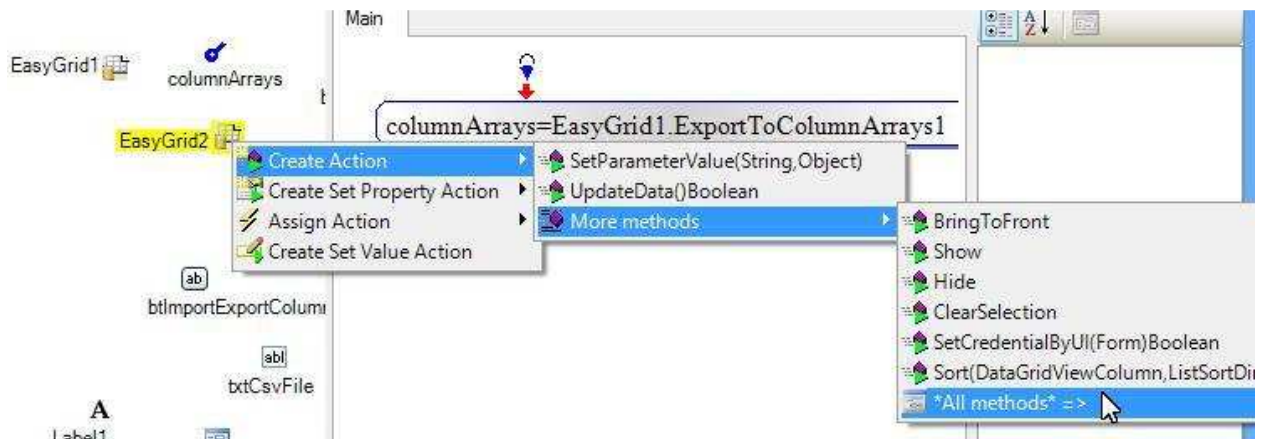

Array, CSV and Data Binding

The 3rd array:

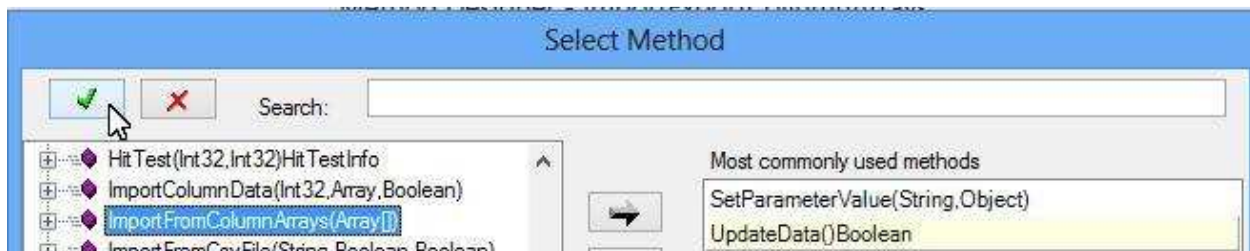
True
False
True

Import column arrays

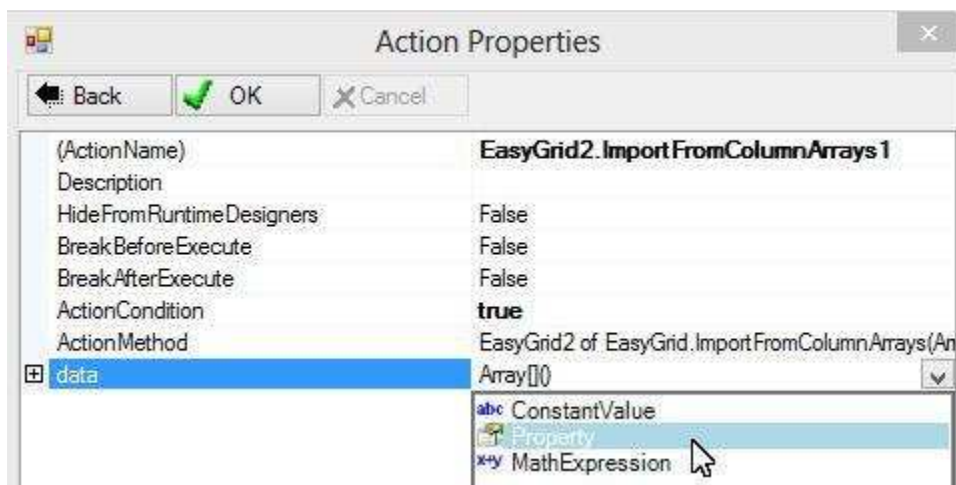
You may use variable, columnArrays, in your programming logics. In this sample, we simply import it into another EasyGrid, EasyGrid2.



Select `ImportFromColumnArrays`:



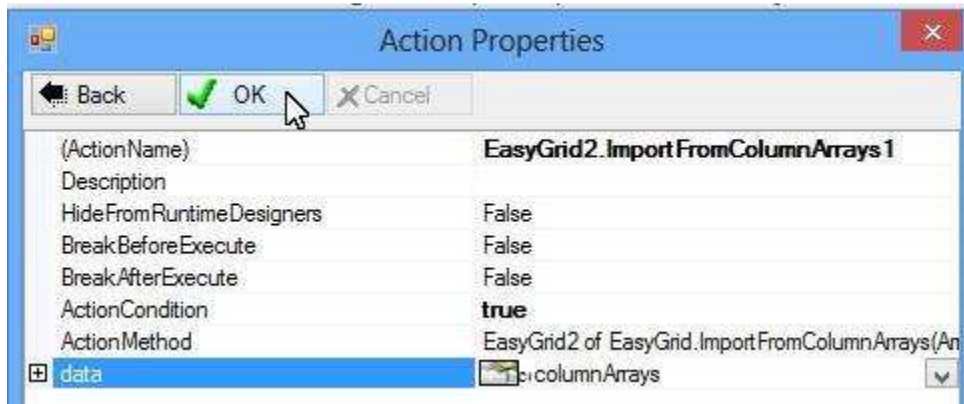
Choose variable, `columnArrays`, for the “data” in the action:



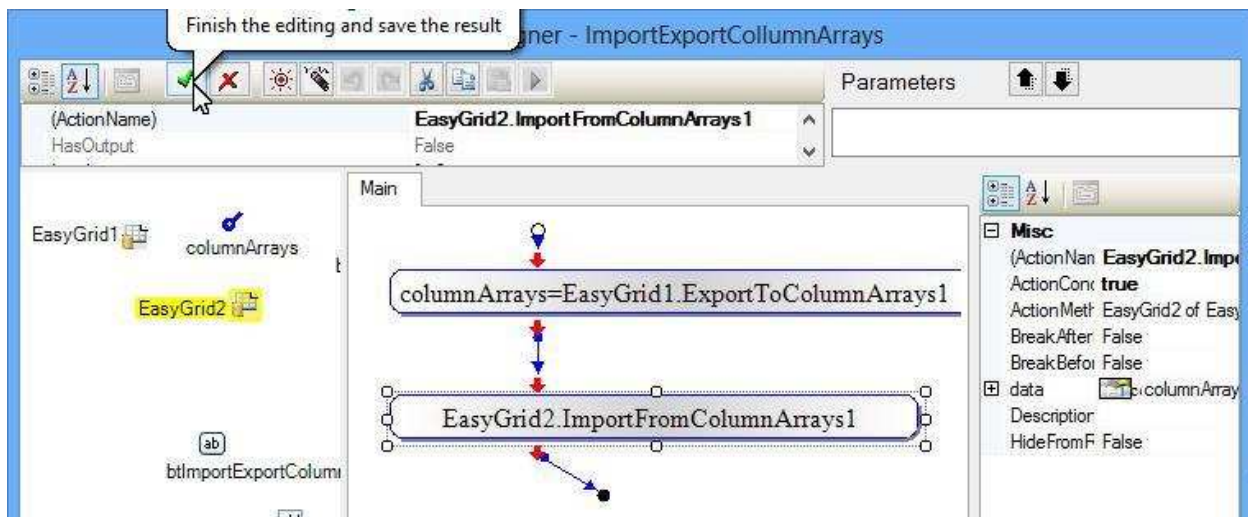
Array, CSV and Data Binding



Click OK to create the action:



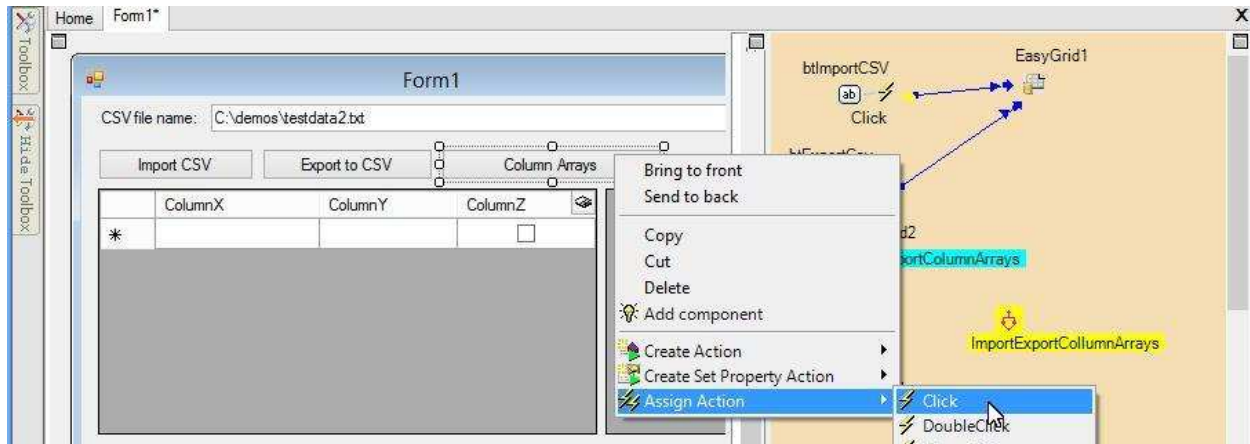
Link the new action to the last action:



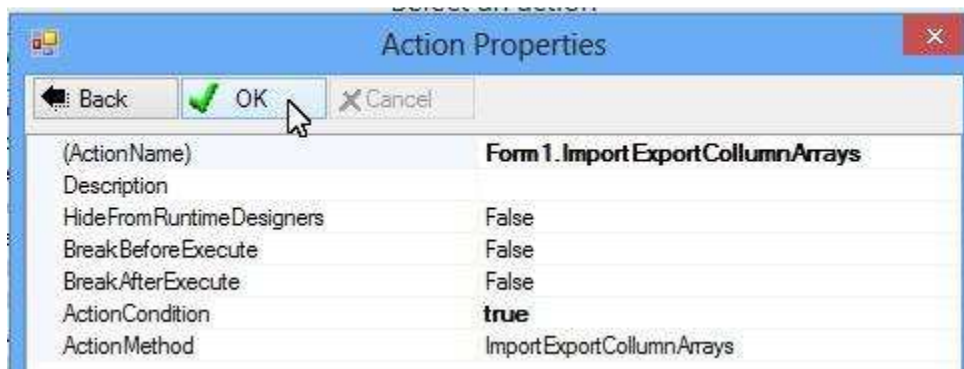
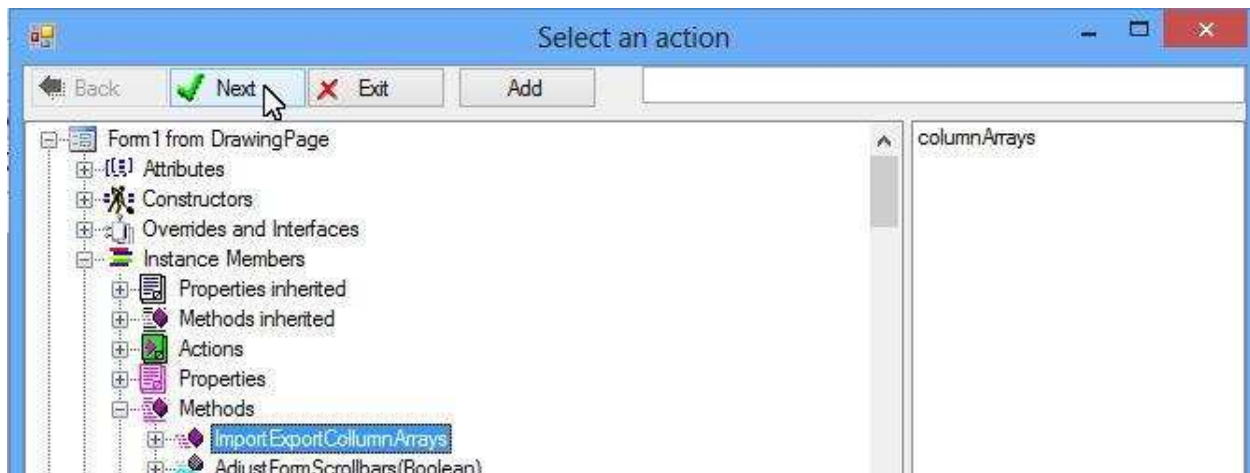
Test

Let's use a button to execute this new method:

Array, CSV and Data Binding

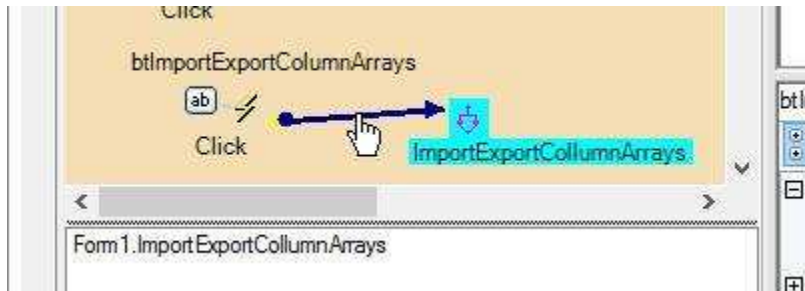


Choose the method:



A new action is created and assigned to the button:

Array, CSV and Data Binding



We may run the application. Click the import button to load CSV file:

The screenshot shows the application window titled 'Form1'. It contains a text box for 'CSV file name' with the value 'C:\demos\testdata2.txt'. Below the text box are three buttons: 'Import CSV', 'Export to CSV', and 'Column Arrays'. The 'Import CSV' button is highlighted. Below the buttons is a table with three columns: 'ColumnX', 'ColumnY', and 'ColumnZ'. The table contains three rows of data.

| | ColumnX | ColumnY | ColumnZ |
|---|-----------------|---------|-------------------------------------|
| ▶ | Some text | 123 | <input checked="" type="checkbox"/> |
| | Hello World | 100 | <input type="checkbox"/> |
| | one, two, three | 200 | <input checked="" type="checkbox"/> |
| * | | | <input type="checkbox"/> |

Click button "Column Arrays". We can see that the data appear in the second EasyGrid:

The screenshot shows the application window titled 'Form1' with the 'Column Arrays' button highlighted. Below the buttons are two tables. The first table is the same as the one in the previous screenshot. The second table, titled 'Column1', 'Column2', and 'Column3', contains the same data as the first table.

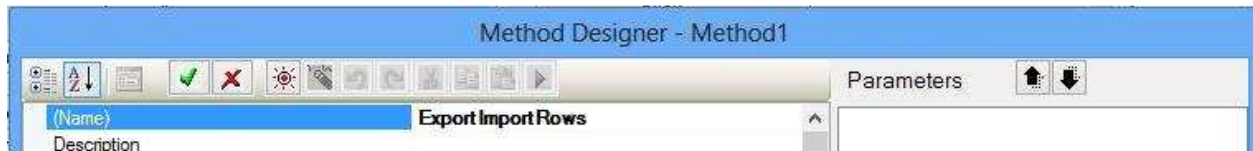
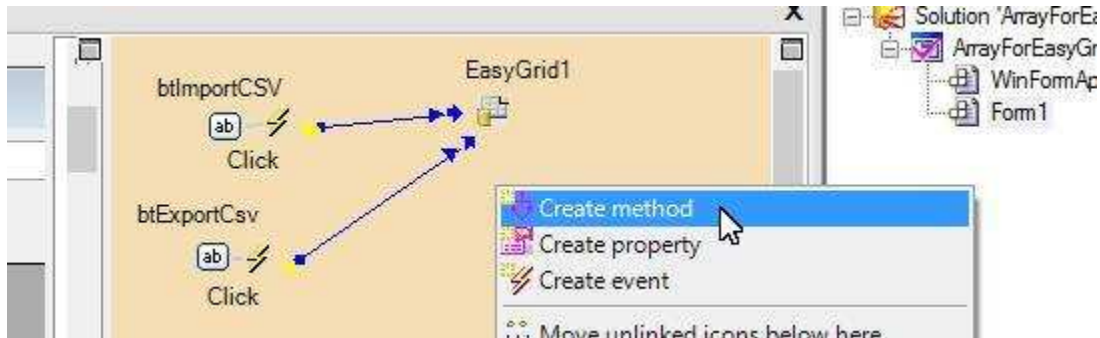
| | ColumnX | ColumnY | ColumnZ |
|---|-----------------|---------|-------------------------------------|
| ▶ | Some text | 123 | <input checked="" type="checkbox"/> |
| | Hello World | 100 | <input type="checkbox"/> |
| | one, two, three | 200 | <input checked="" type="checkbox"/> |
| * | | | <input type="checkbox"/> |

| | Column1 | Column2 | Column3 |
|---|-----------------|---------|---------|
| ▶ | Some text | 123 | True |
| | Hello World | 100 | False |
| | one, two, three | 200 | True |
| * | | | |

Import and Export Rows

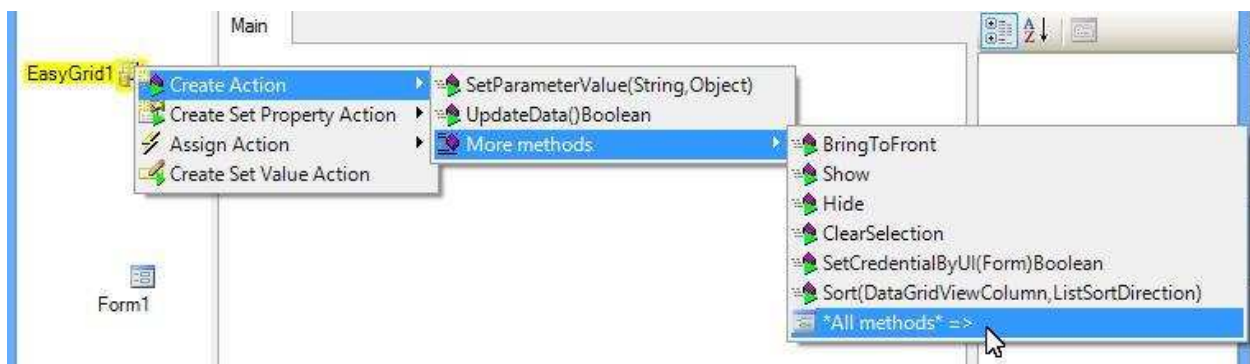
We saw that data can be exported and imported in column arrays. Data can also be exported and imported as row arrays. We'll create another method to demonstrate it.

Array, CSV and Data Binding

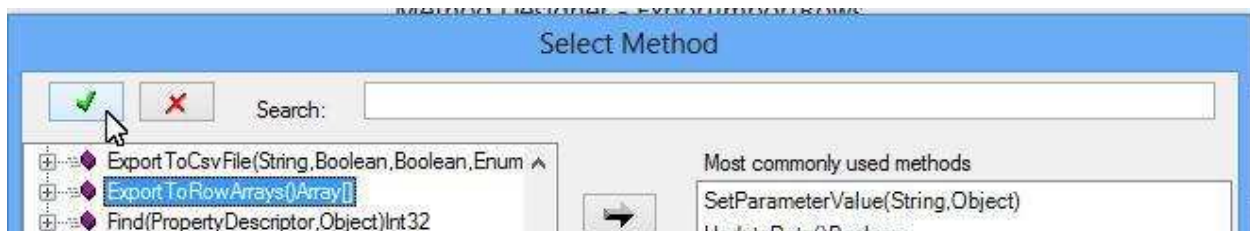


Export row arrays

Create an action to export data to a row array:

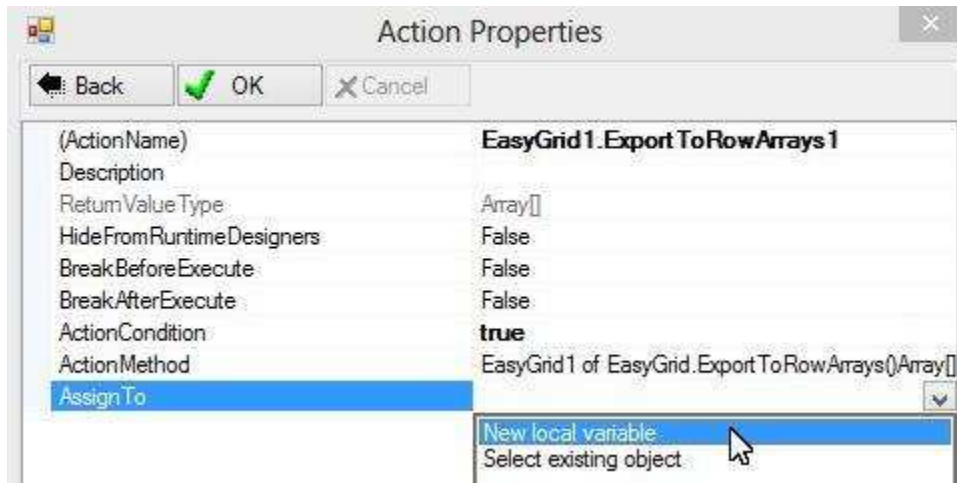


Choose method ExportToRowArrays

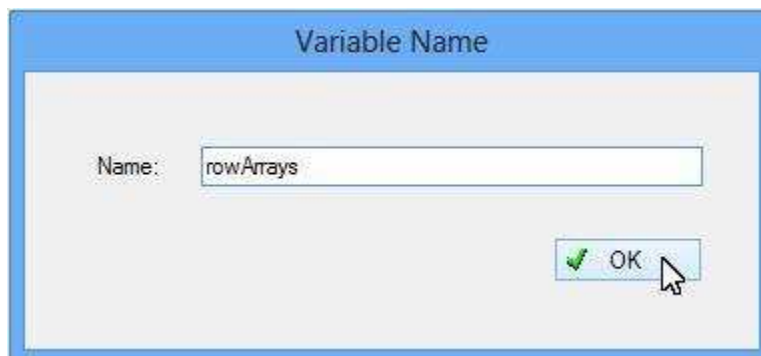


Select "New local variable" to use a new variable for the action result:

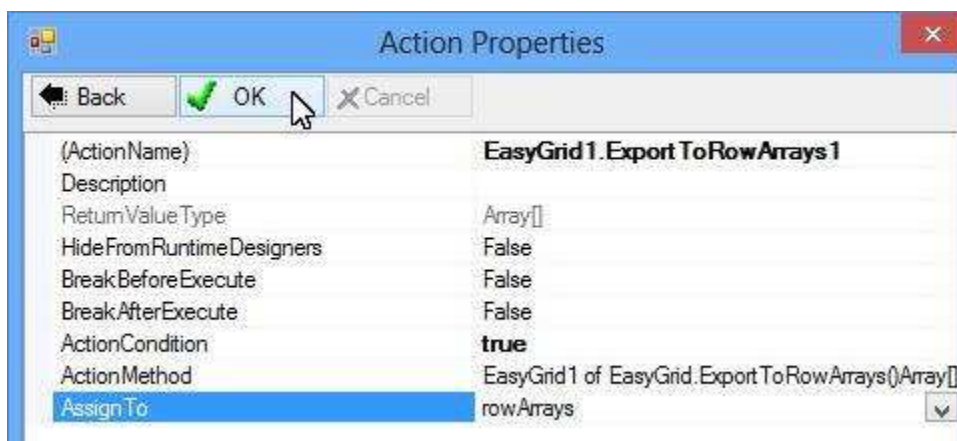
Array, CSV and Data Binding



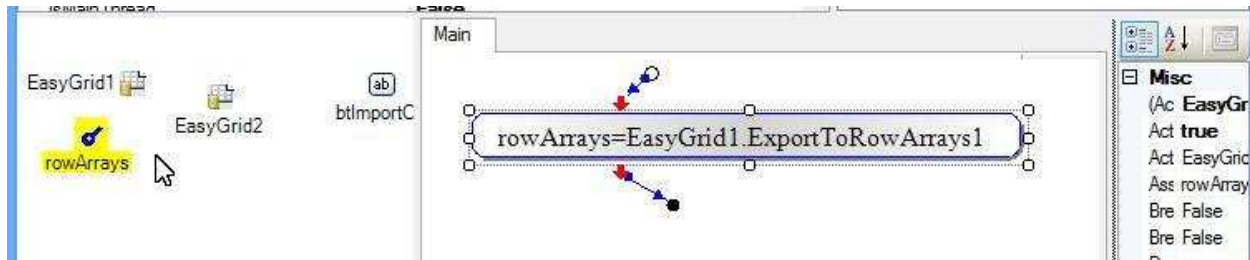
Name the new variable rowArrays:



Click OK. A new action and a new variable are created.



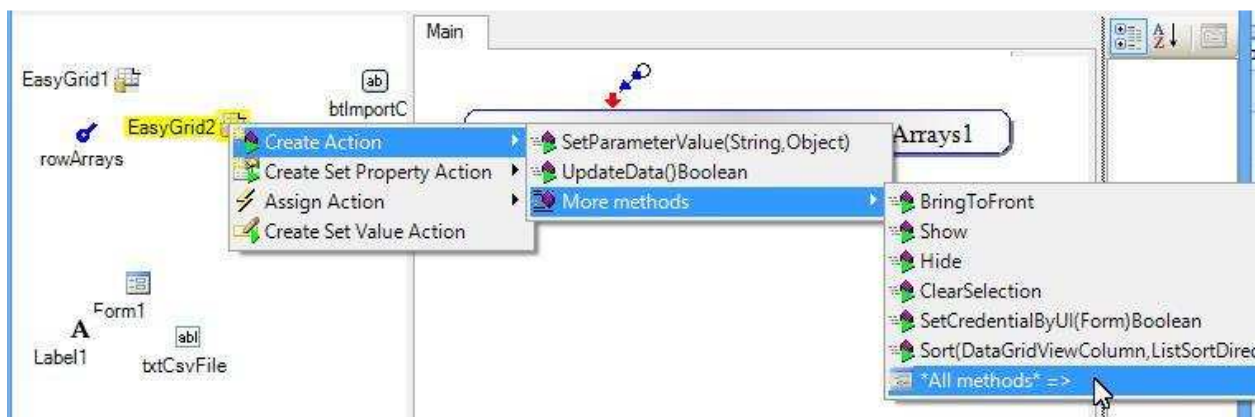
Array, CSV and Data Binding



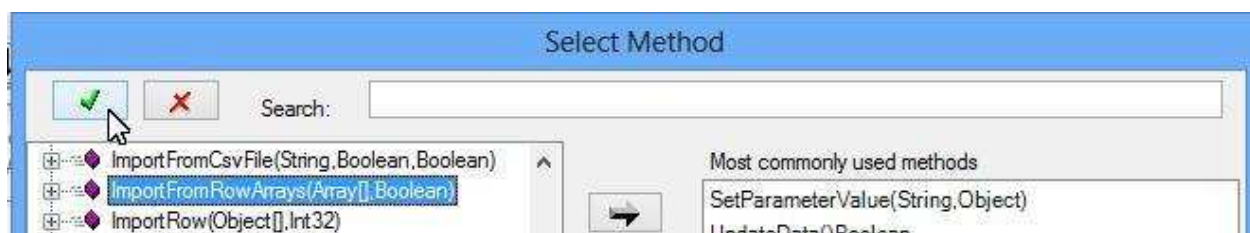
Variable, rowArrays, contains several arrays. Each array is one row of data. If EasyGrid1 loads the sample CSV file, testdata2.txt, then rowArrays contains 3 arrays. The first array is for the first row: [Some text, 123, True]; the second array is for the second row:[Hello World, 100, False], and so on.

Import row arrays

You may use variable rowArrays in your programming. In this sample, we simply import it into the second EasyGrid.

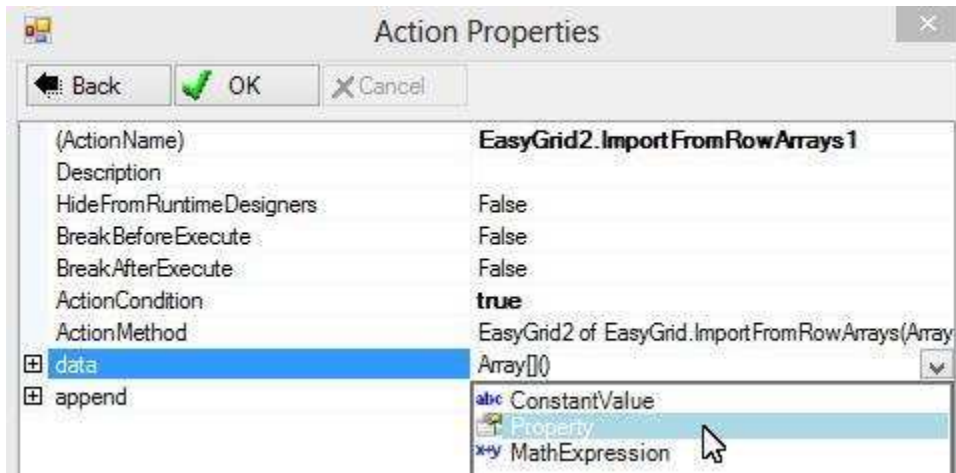


Select ImportFromRowArrays:



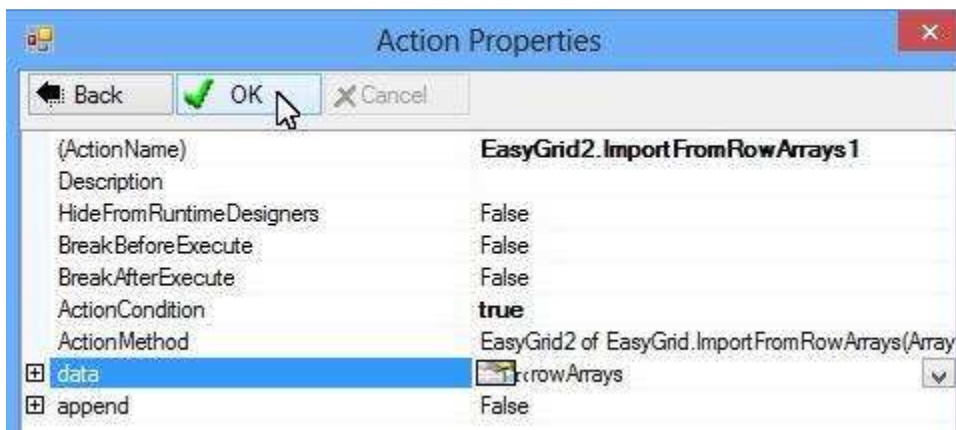
Select the variable, rowArrays, for the “data” of the action:

Array, CSV and Data Binding



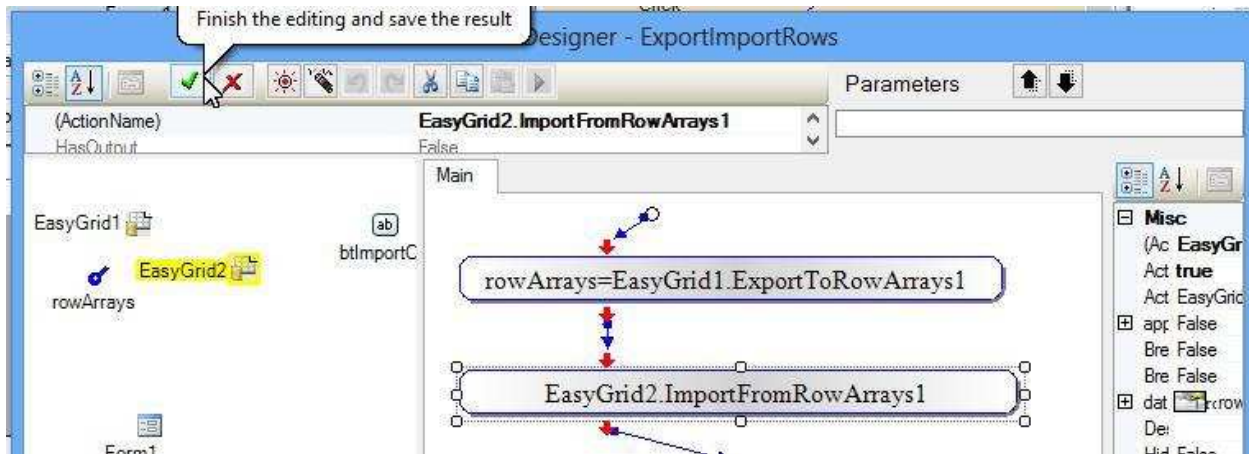
If “append” is True then the new data will be appended at the end of the table. If “append” is False then the existing data will be erased.

Click OK to create the action:



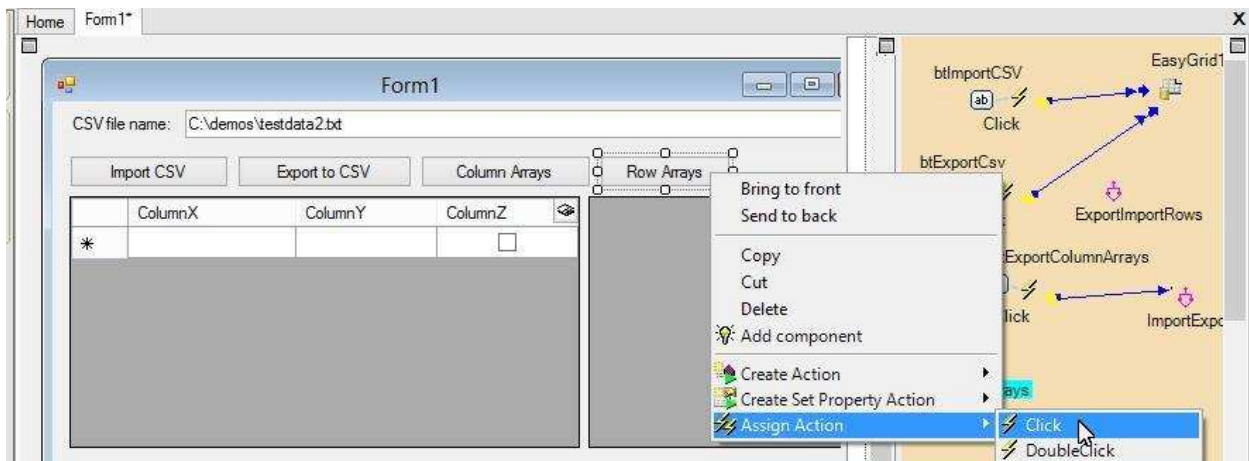
Link the action to the previous action and we are done:

Array, CSV and Data Binding

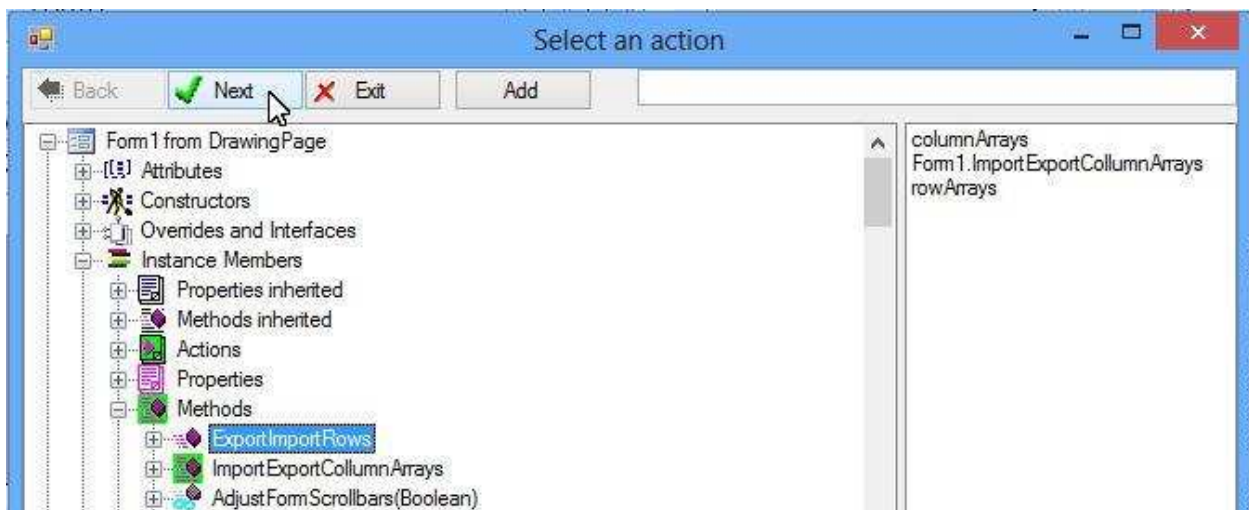


Test

Use a button to execute the new method:

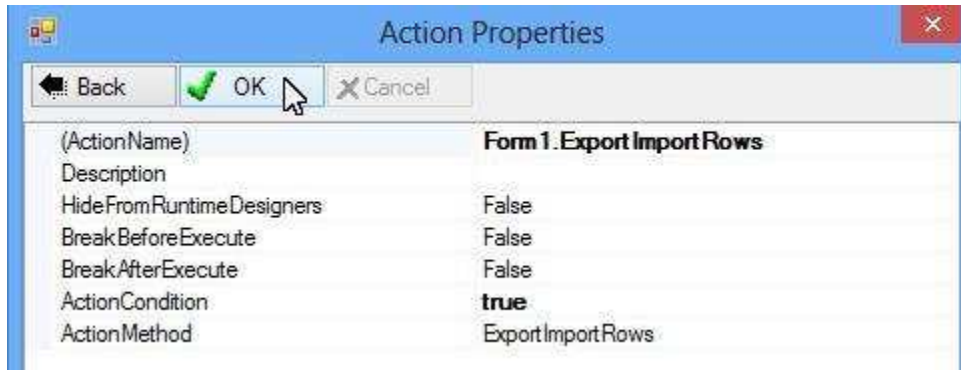


Select the new method:

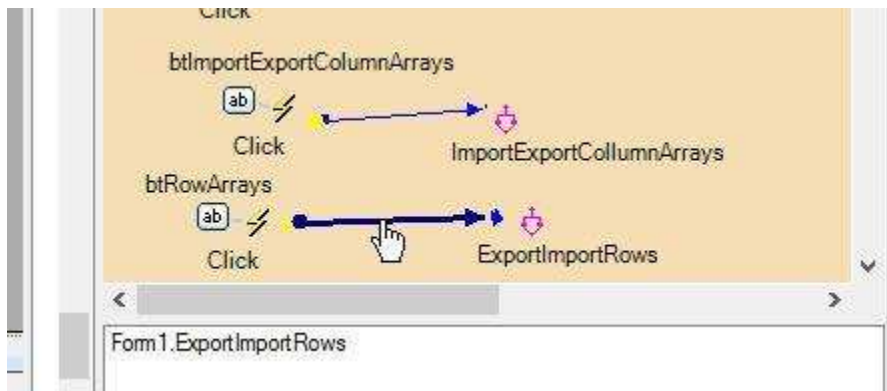


Array, CSV and Data Binding

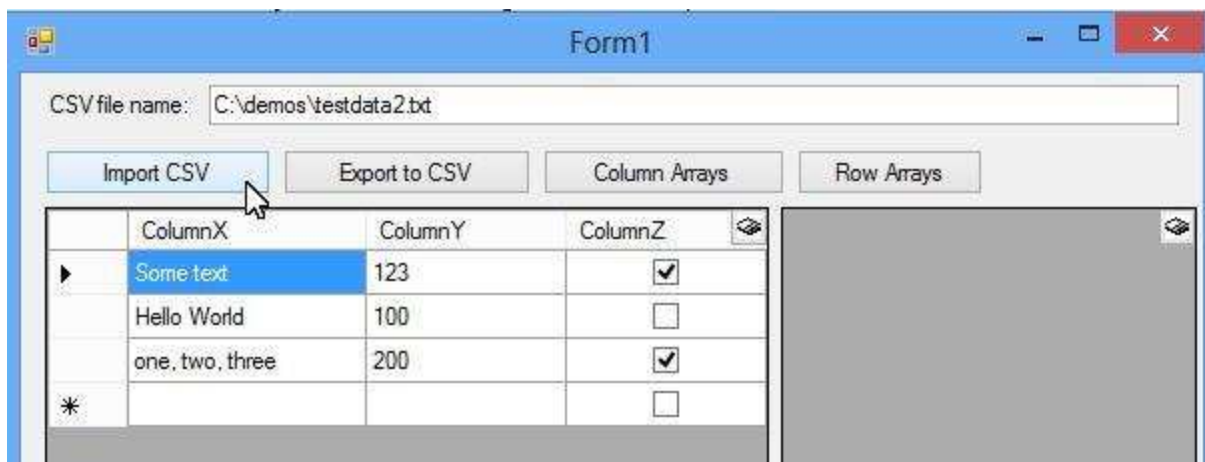
Click OK:



The action is created and assigned to the button:

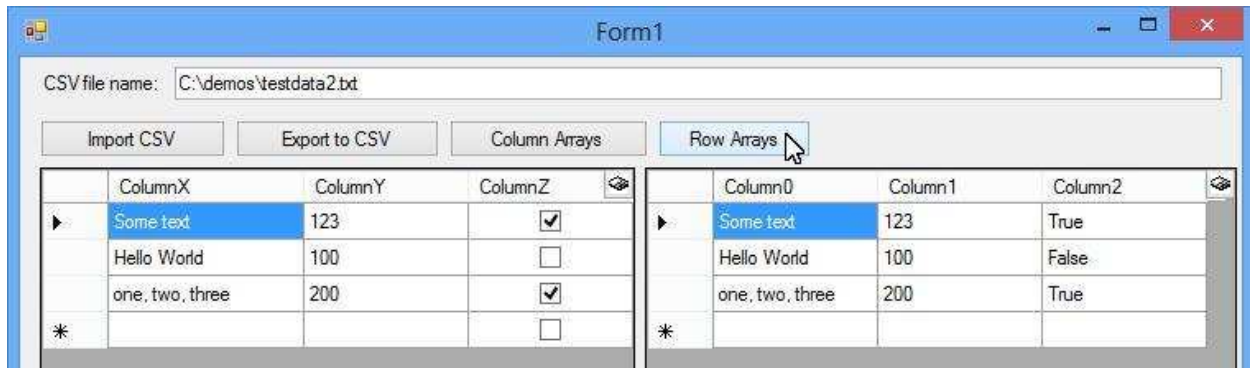


Run the application. Import CSV file, testdata2.txt, into the first EasyGrid:



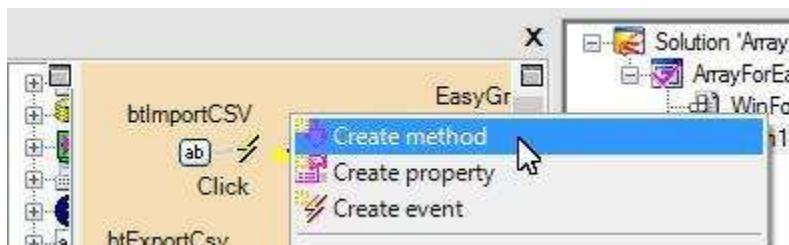
Click button "Row Arrays". Data appear in the second EasyGrid:

Array, CSV and Data Binding



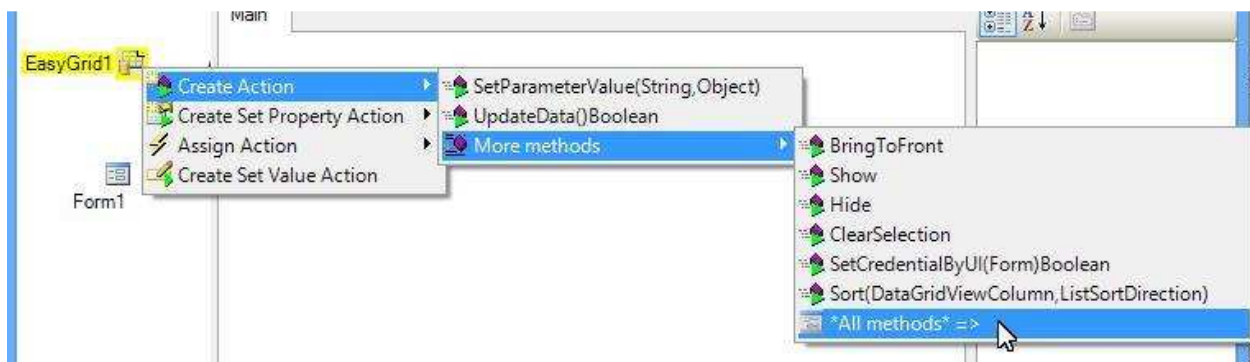
Use single column of data

Column data can be used as a one dimension array. We'll create a new method to demonstrate such usages.



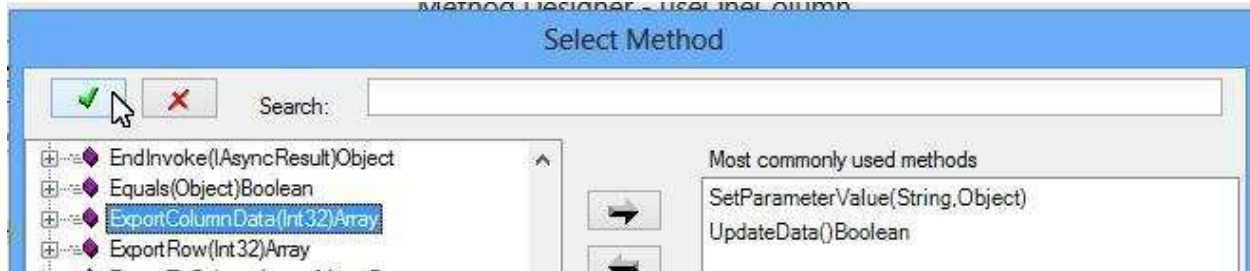
Export one column of data

We create an action to get a column of data to an array.



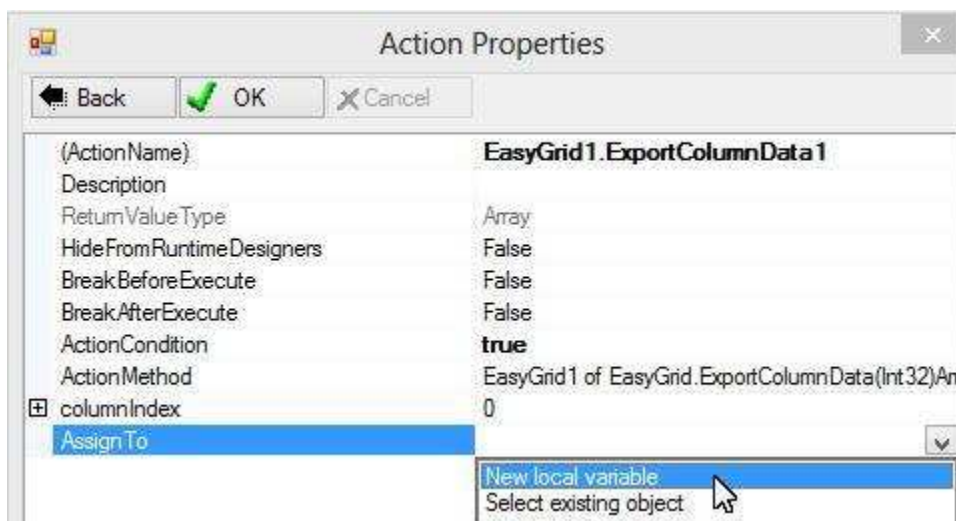
Select ExportColumnData

Array, CSV and Data Binding

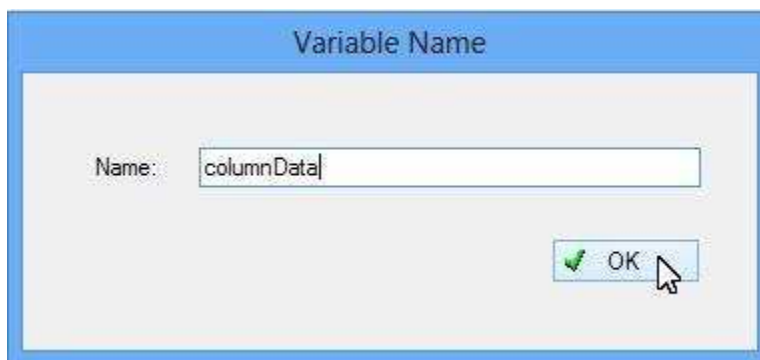


columnIndex indicates which column we want. 0 indicates the first column; 1 indicates the second column; and so on.

Select “New local variable” to use a new variable for the action result:

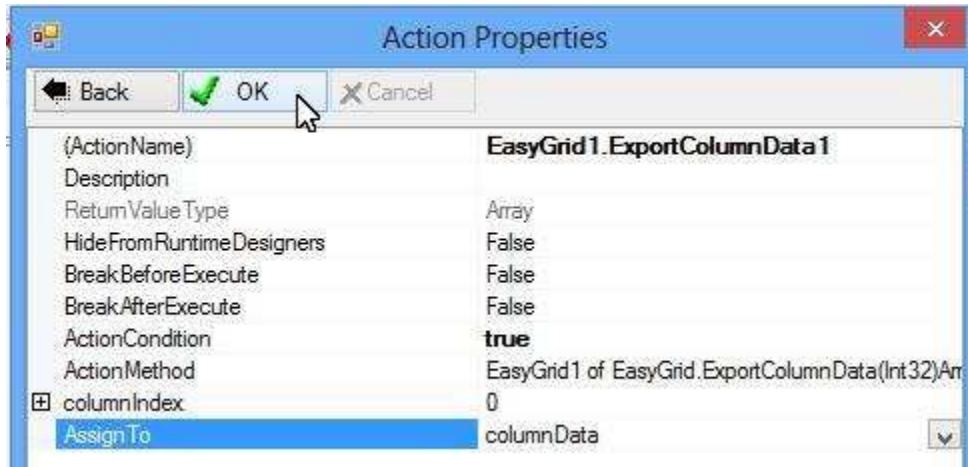


Name the new variable columnData:

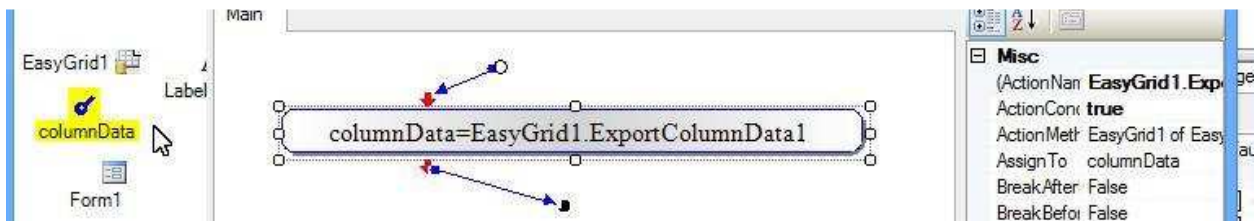


Click OK to create the action:

Array, CSV and Data Binding



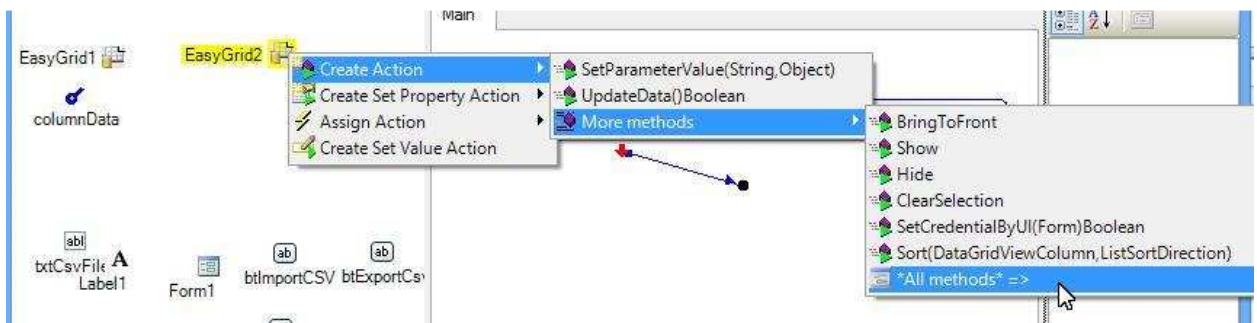
The action is created.



The variable `columnData` is an array containing the first column of data because we used 0 for `columnIndex` of the action.

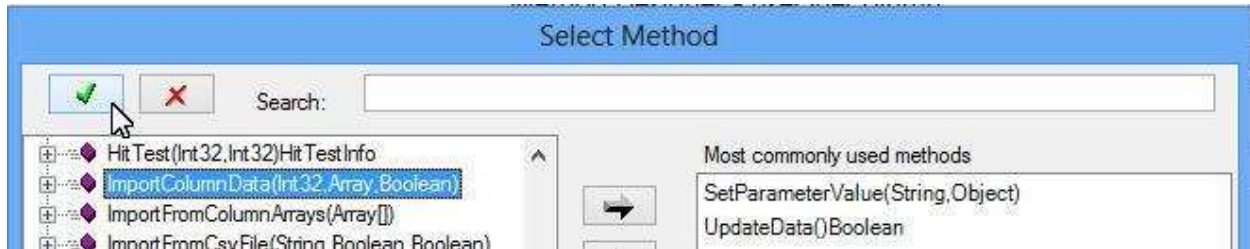
Import one column of data

You may use variable `columnData` in your programming. For this sample, we simply import it to another EasyGrid.

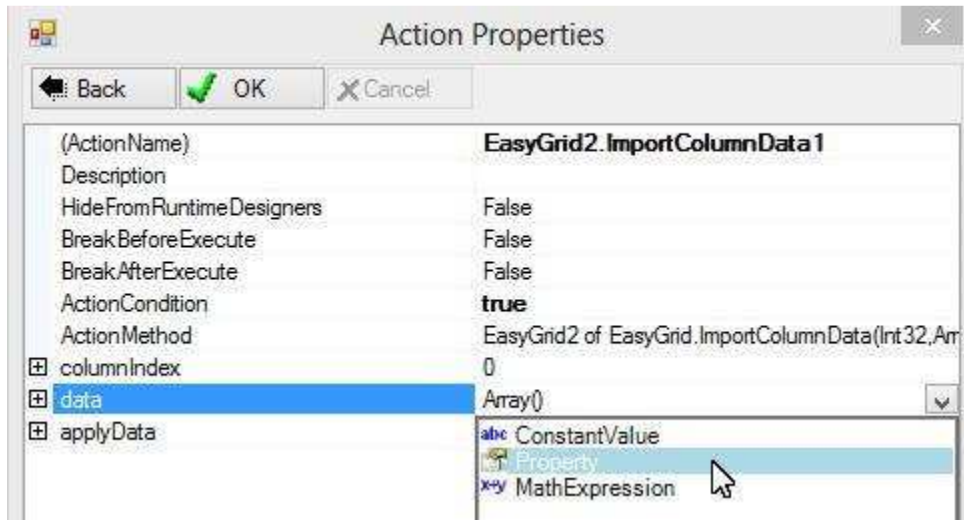


Select `ImportColumnData`

Array, CSV and Data Binding

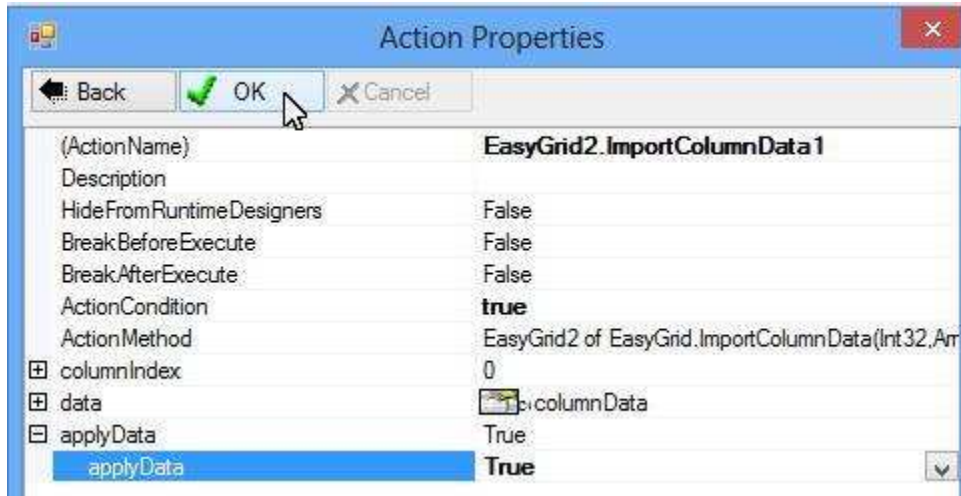


Select variable columnData for “data”:

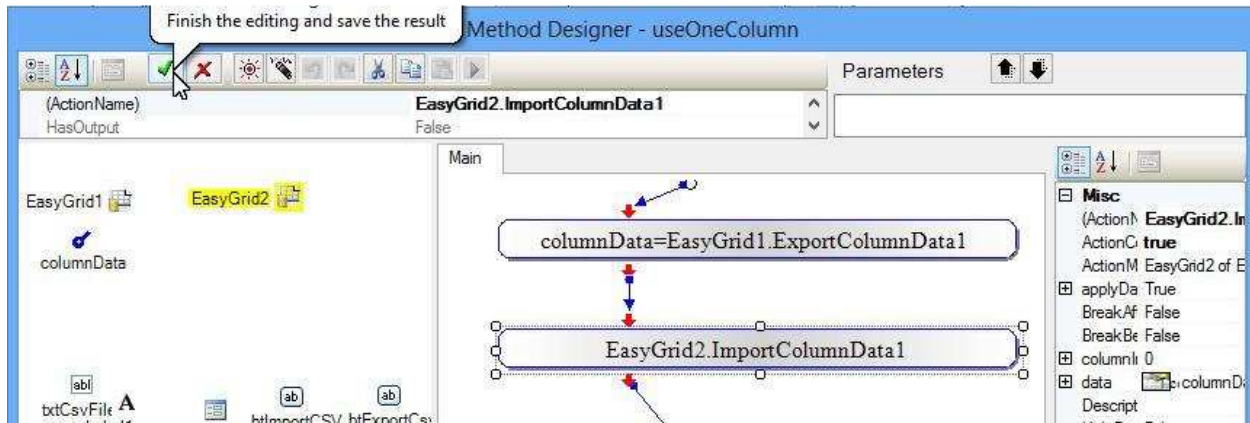


Set applyData to True so that the column of data will appear in the grid. If you want to execute several of such actions then you may set applyData to False except the last action.

Array, CSV and Data Binding



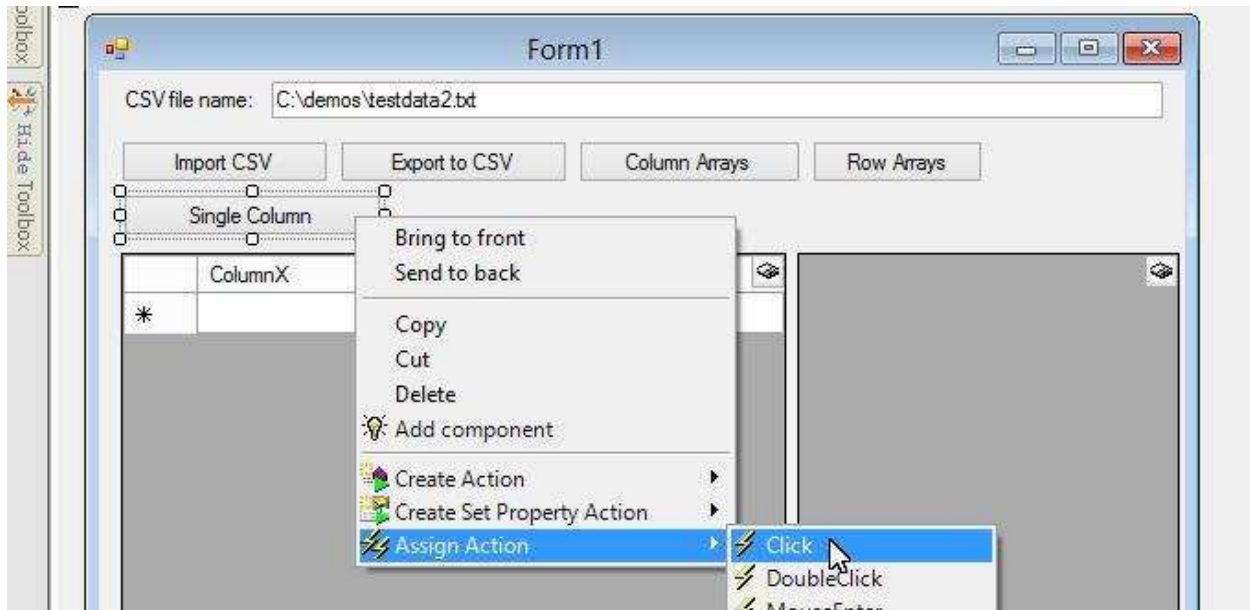
Link the two actions and finish the method:



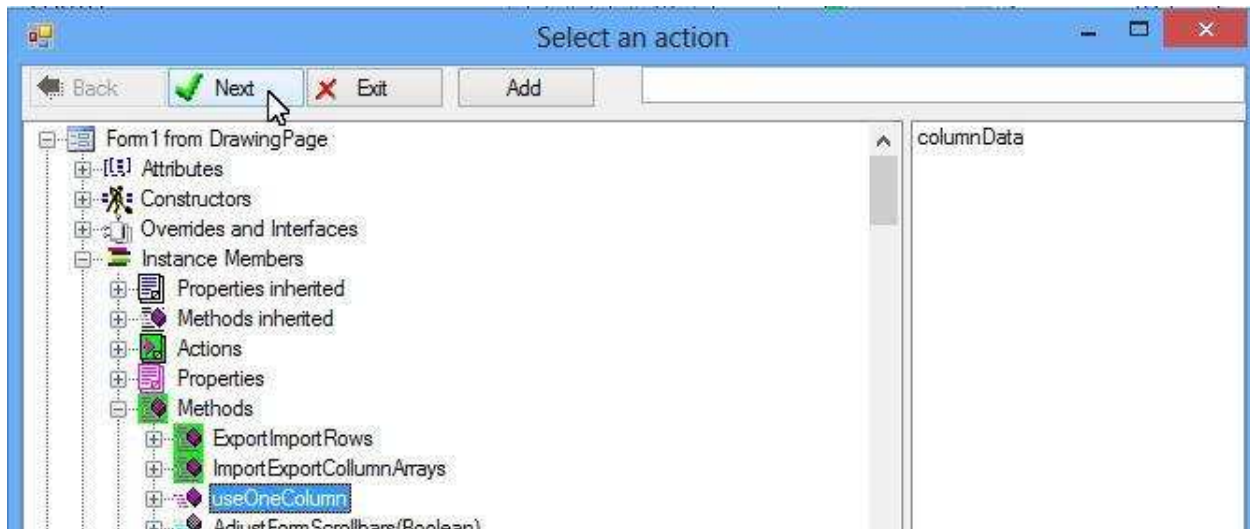
Test

Use a button to execute the new method.

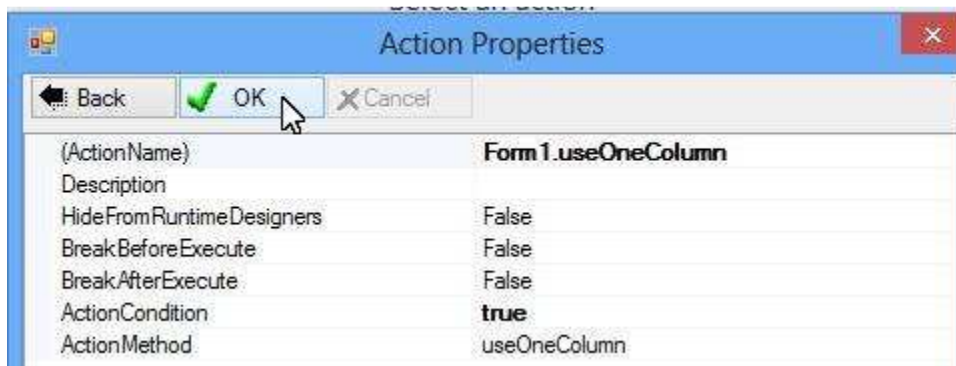
Array, CSV and Data Binding



Select the new method:

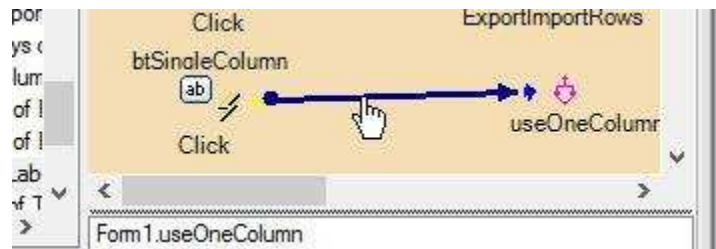


Click OK:



Array, CSV and Data Binding

The action is created assigned to the button:



Run the application. Import CSV file.

The screenshot shows a window titled 'Form1'. At the top, there is a text field for 'CSV file name:' containing 'C:\demos\testdata2.txt'. Below this are four buttons: 'Import CSV', 'Export to CSV', 'Column Arrays', and 'Row Arrays'. The 'Import CSV' button is highlighted with a mouse cursor. Below these buttons is a 'Single Column' button. At the bottom, there is a data grid with four columns: 'ColumnX', 'ColumnY', 'ColumnZ', and an empty column. The data rows are:

| | ColumnX | ColumnY | ColumnZ | |
|---|-----------------|---------|-------------------------------------|--|
| ▶ | Some text | 123 | <input checked="" type="checkbox"/> | |
| | Hello World | 100 | <input type="checkbox"/> | |
| | one, two, three | 200 | <input checked="" type="checkbox"/> | |
| * | | | <input type="checkbox"/> | |

Click "Single Column" button. The first column of the first EasyGrid appears in the second EasyGrid.

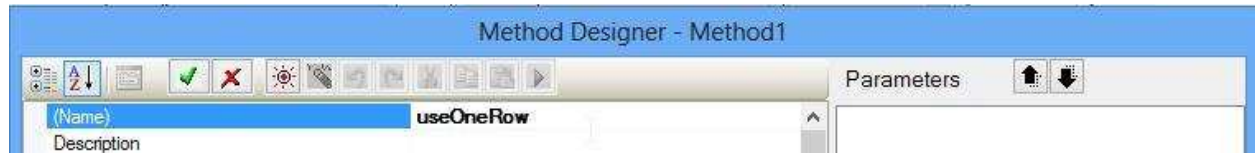
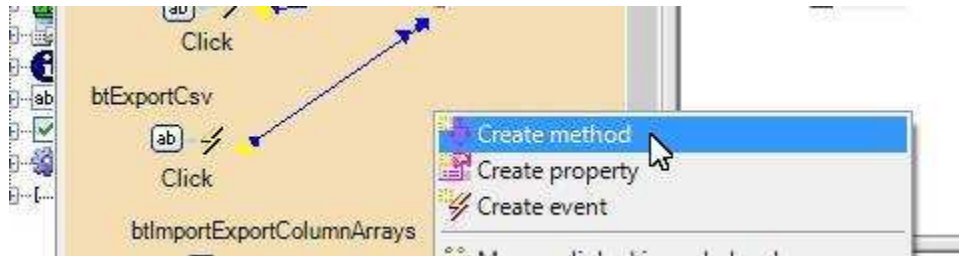
The screenshot shows the same 'Form1' window after clicking the 'Single Column' button. The 'Single Column' button is now highlighted. A second data grid has appeared to the right of the first one. This second grid has a single column labeled 'Column1' and contains the same data as the first grid's 'ColumnX' column:

| | Column1 | |
|---|-----------------|--|
| ▶ | Some text | |
| | Hello World | |
| | one, two, three | |
| * | | |

Use single row of data

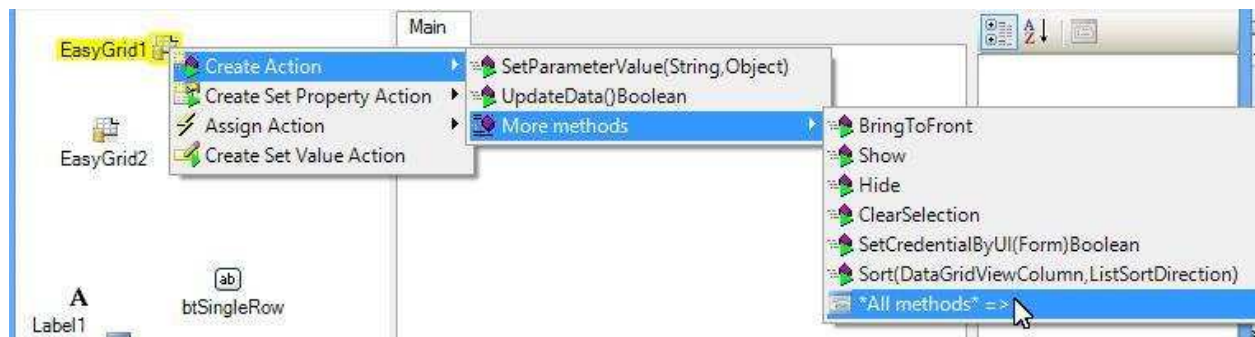
Each row is a one dimensional array. We create a new method to demonstrate the use of single row.

Array, CSV and Data Binding

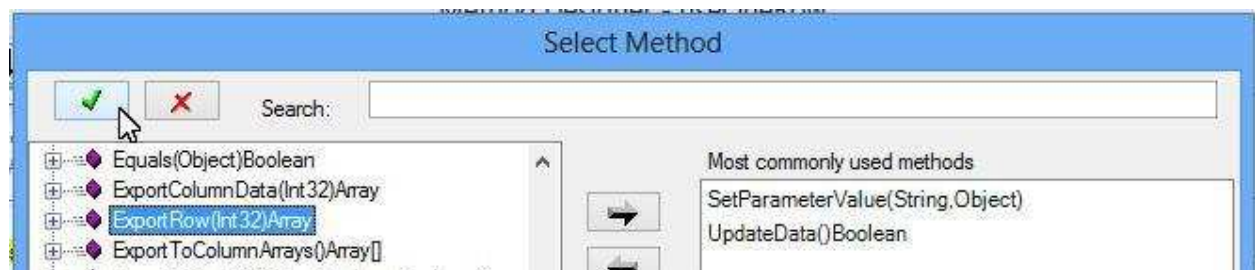


Export one row of data

Create an action to get one row of data:



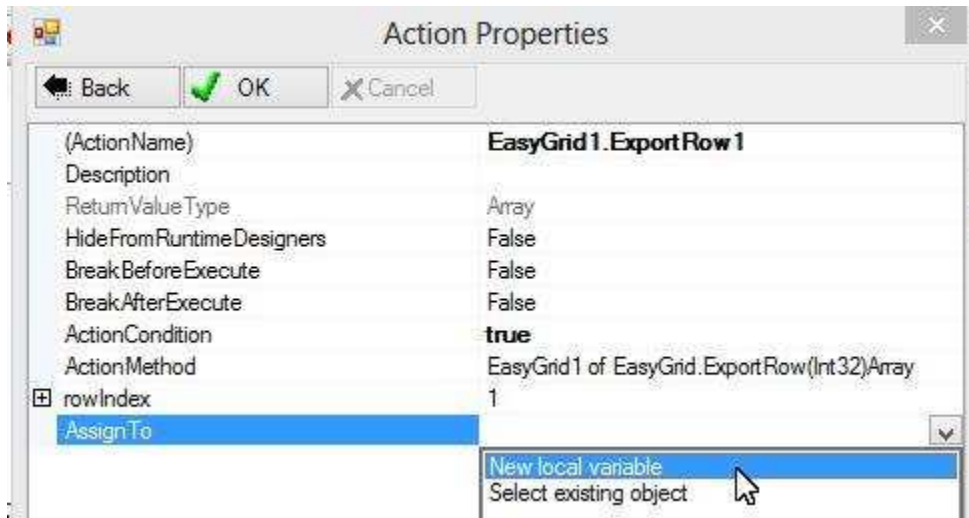
Select ExportRow



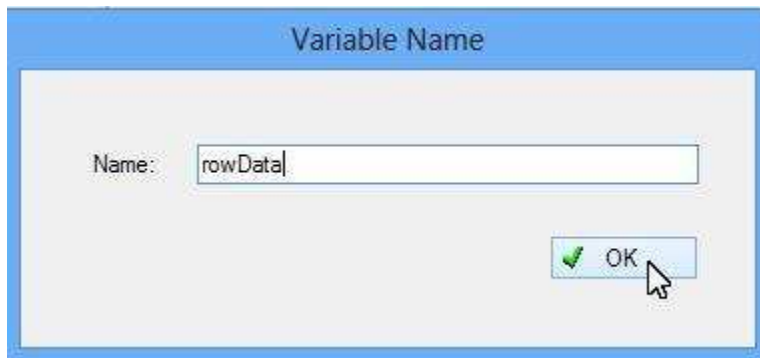
rowIndex indicates which row we want. 0 indicates the first row; 1 indicates the second row; and so on.

Select "New local variable" to use a new variable for action result:

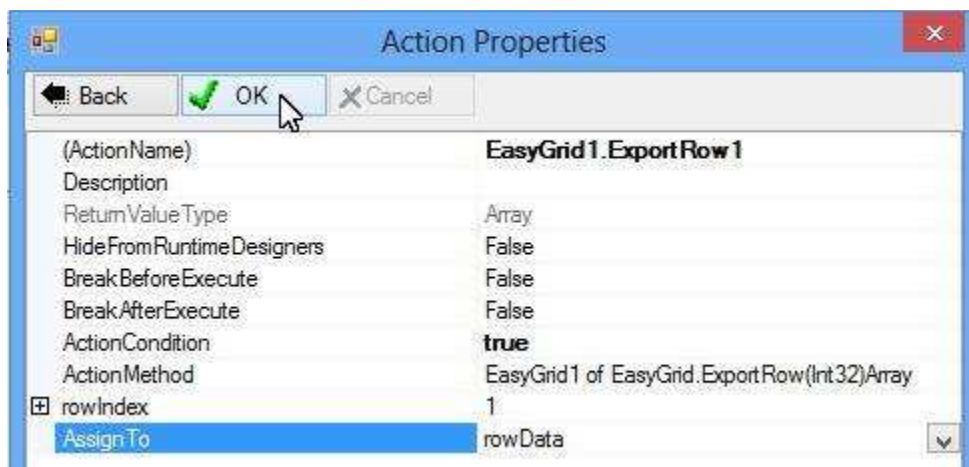
Array, CSV and Data Binding



Name the new variable rowData:

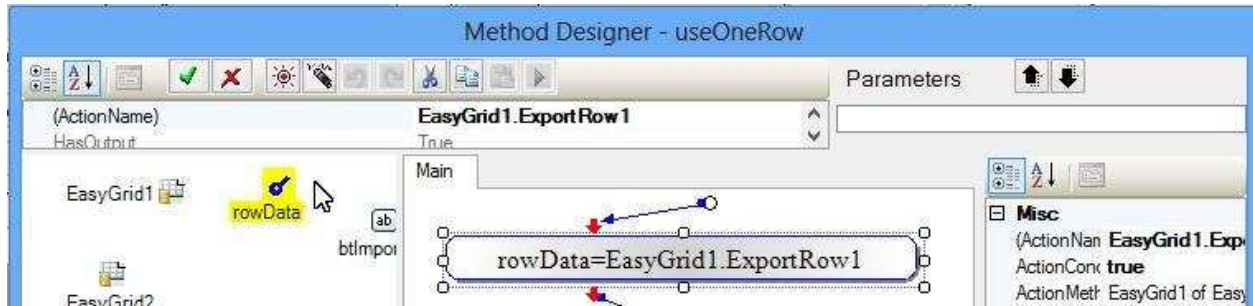


Click OK:



The action and variable are created:

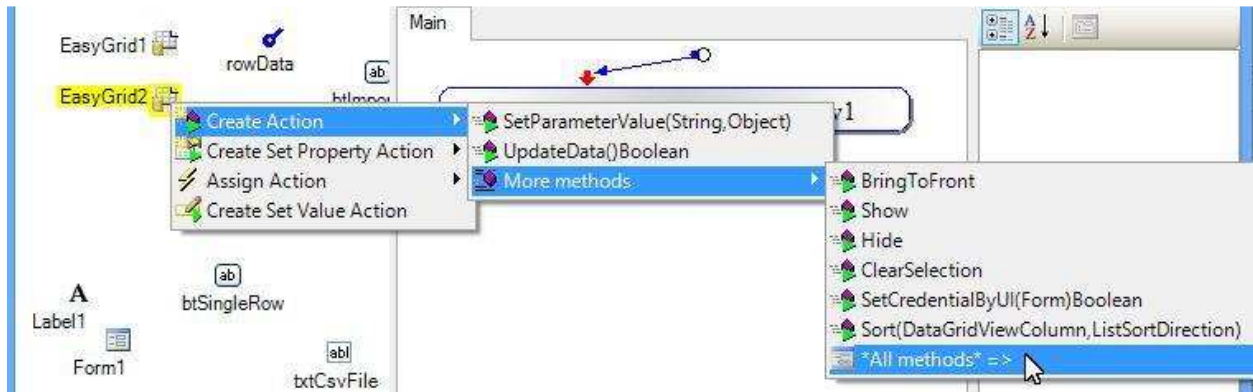
Array, CSV and Data Binding



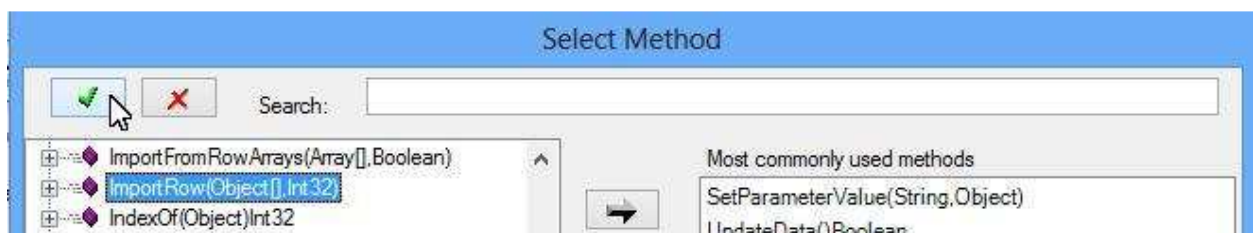
rowData is an array containing the second row because rowIndex is 1 in the action. If the sample CSV file, testdata2.txt, is loaded then rowData should contain [Hello World,100,False]

Import one row of data

You may use rowData according in your programming. In this sample, we simply import it into another EasyGrid:

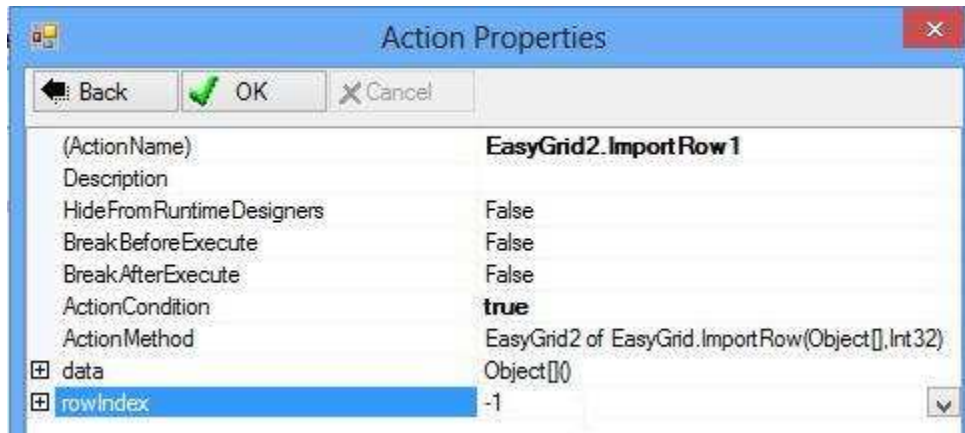


Select ImportRow:

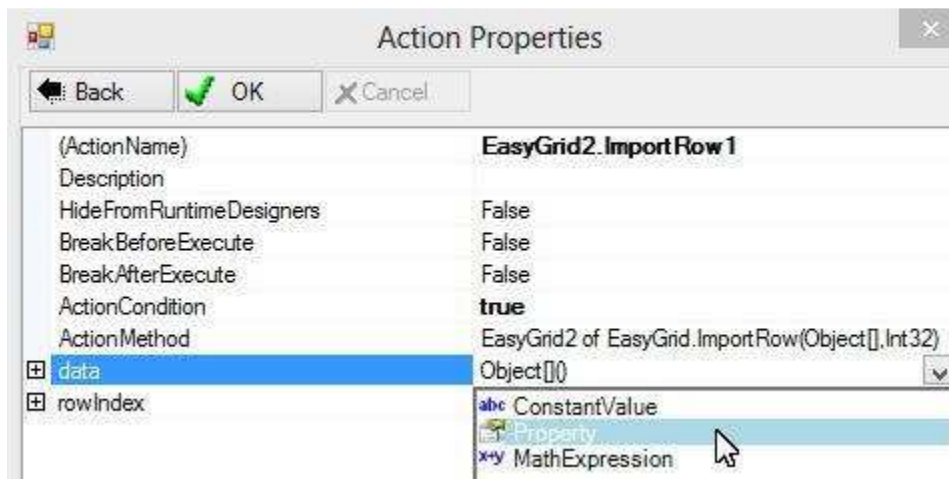


rowIndex indicates which row of data will be replaced by the new data. 0 indicates the first row; 1 indicates the second row; and so on. -1 indicates that we want to append the new data as a new row.

Array, CSV and Data Binding

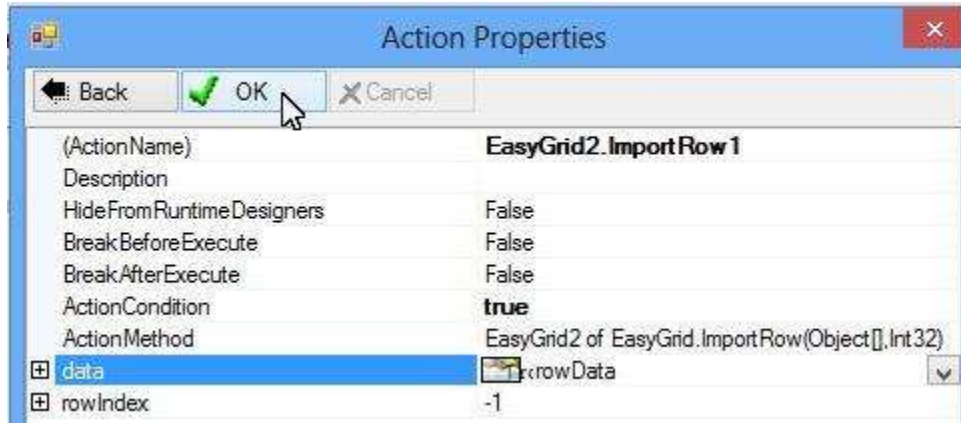


Select variable rowData as the new data to be imported:

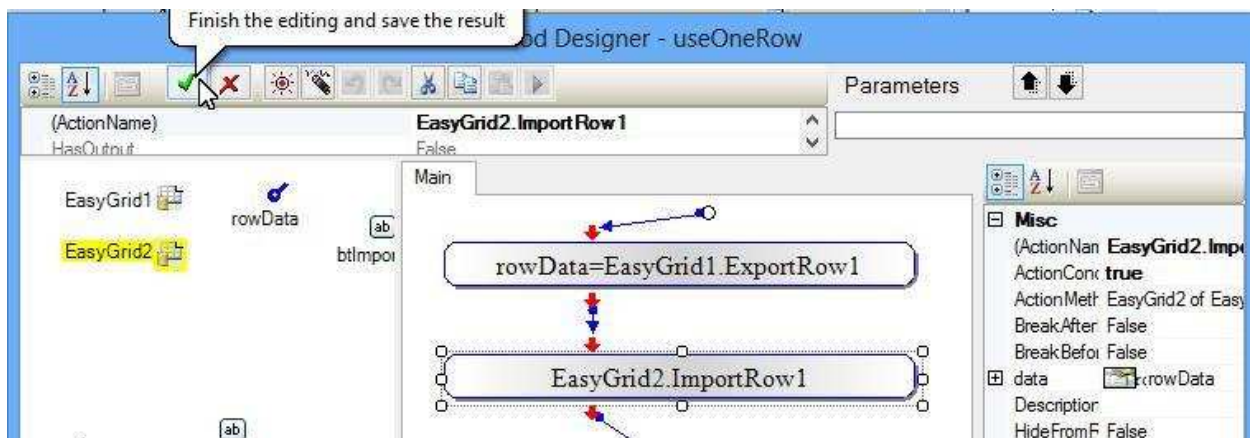


Click OK:

Array, CSV and Data Binding

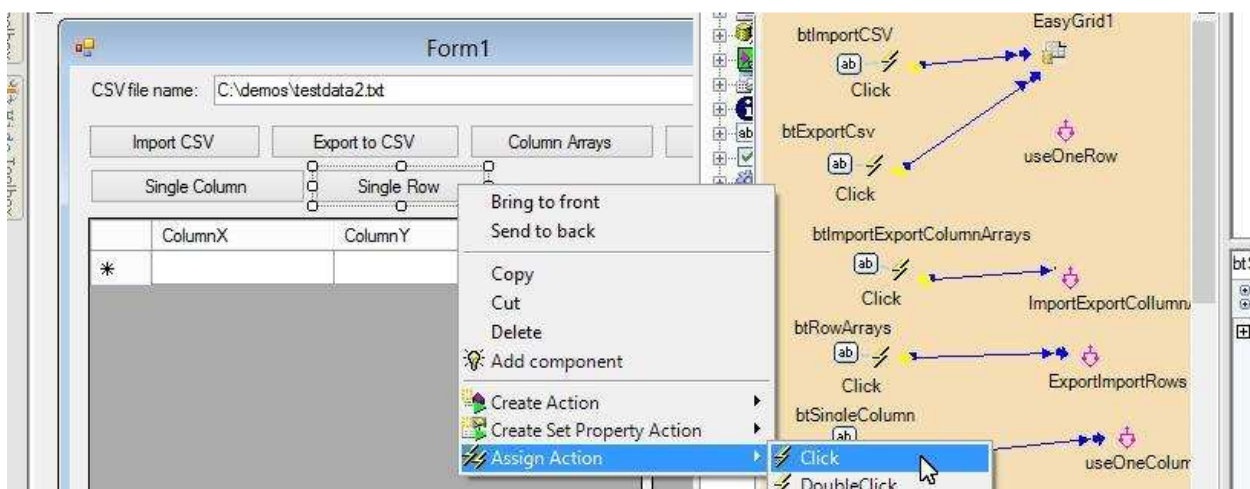


Link the actions and we are done:



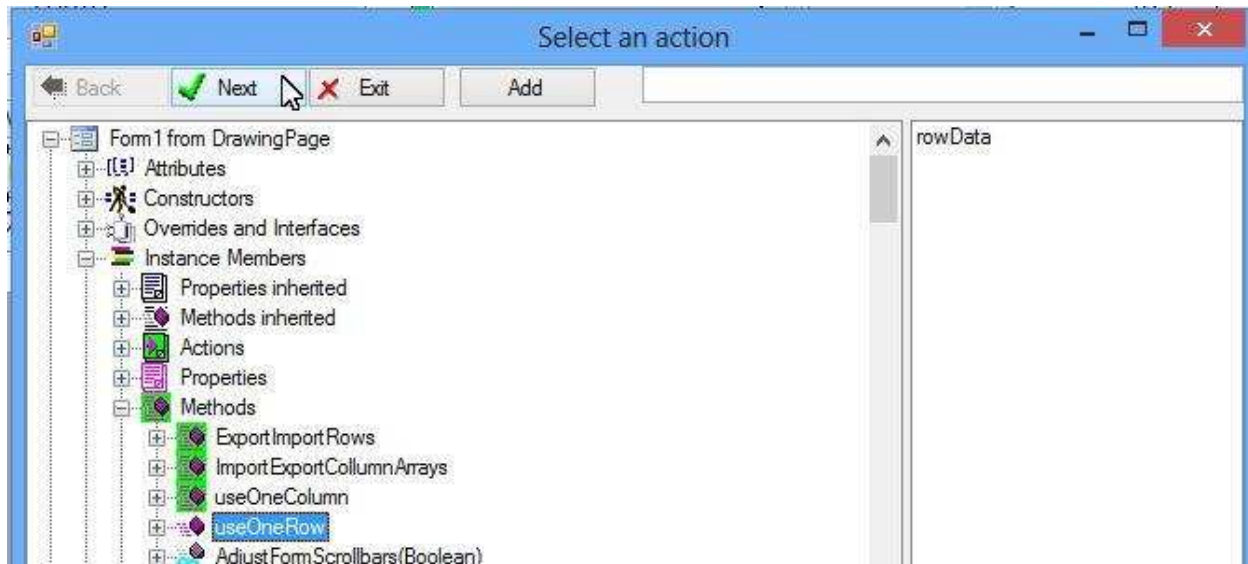
Test

We use a button to execute the new method:

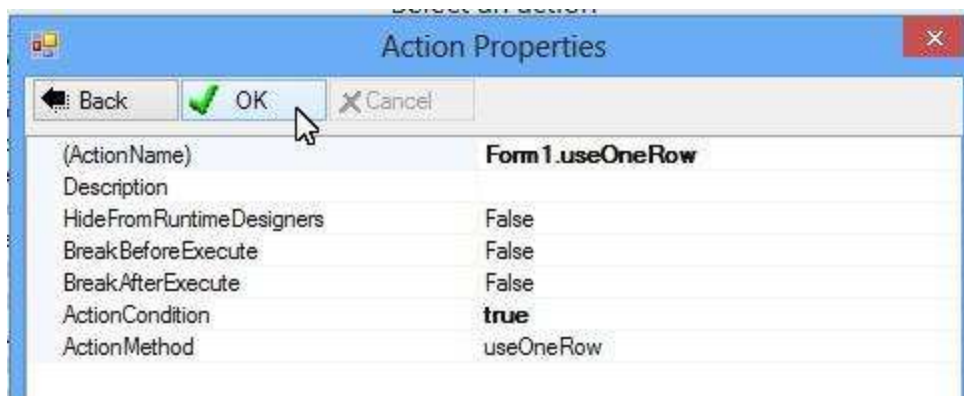


Select the new method:

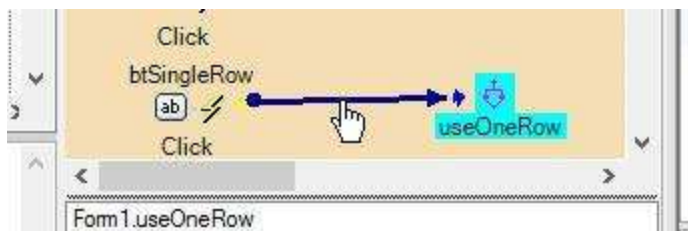
Array, CSV and Data Binding



Click OK:

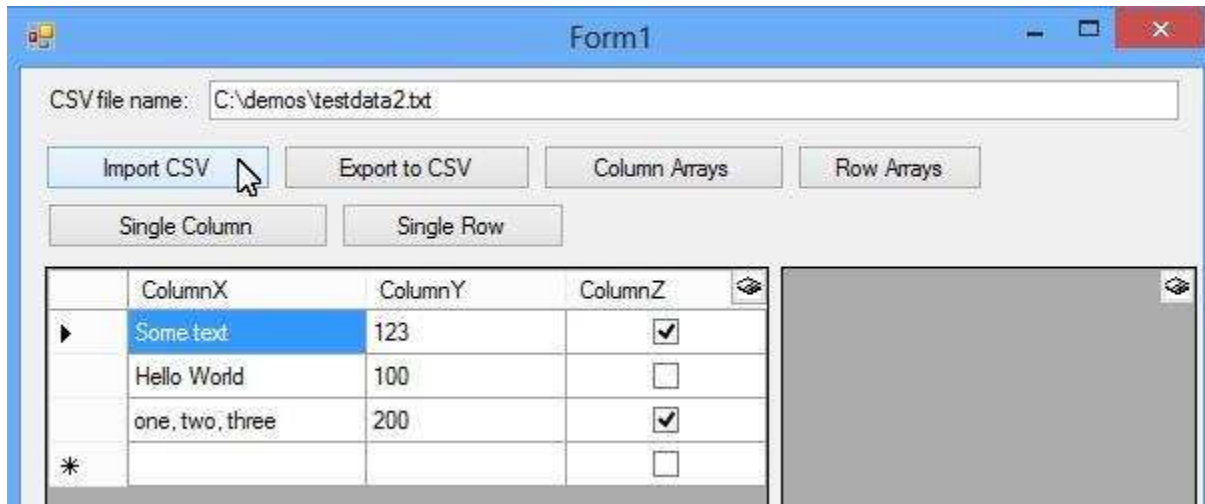


The action is created and assigned to the button:

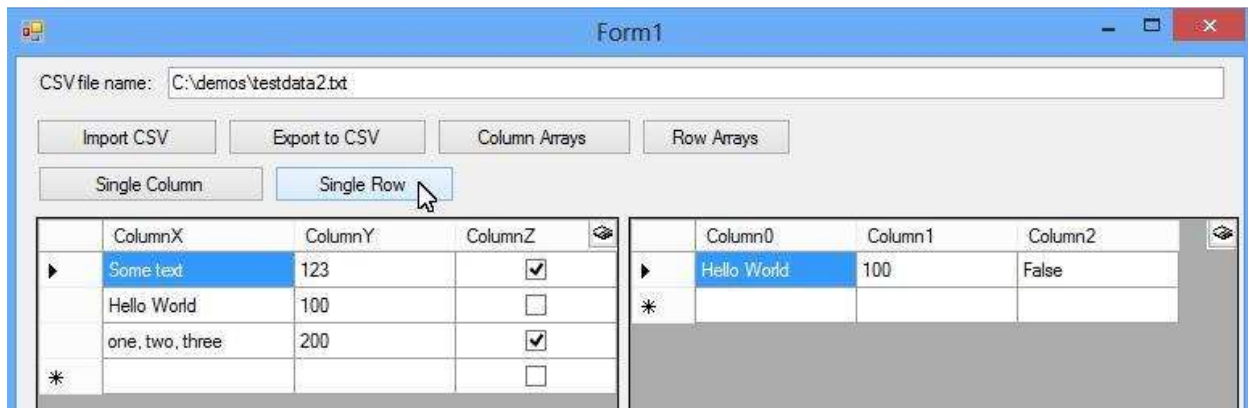


Run the application and import the CSV file:

Array, CSV and Data Binding



Click button "Single Row". The second row appears in the second EasyGrid:

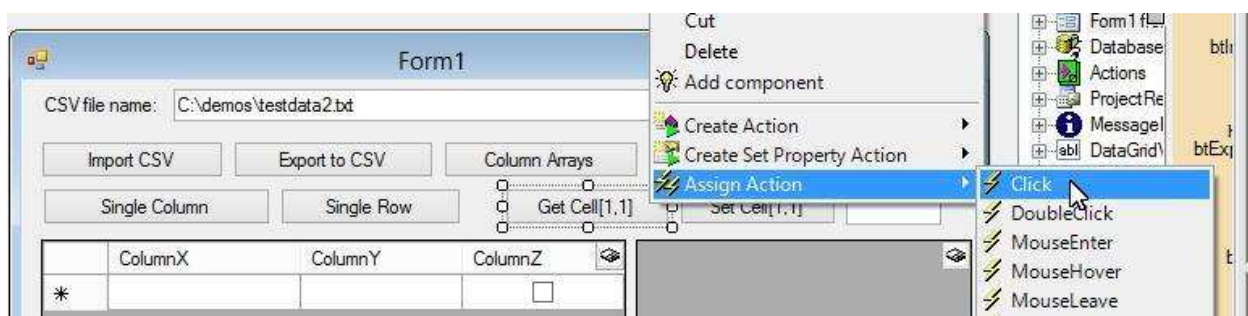


Use single elements

We may get and set value for each cell.

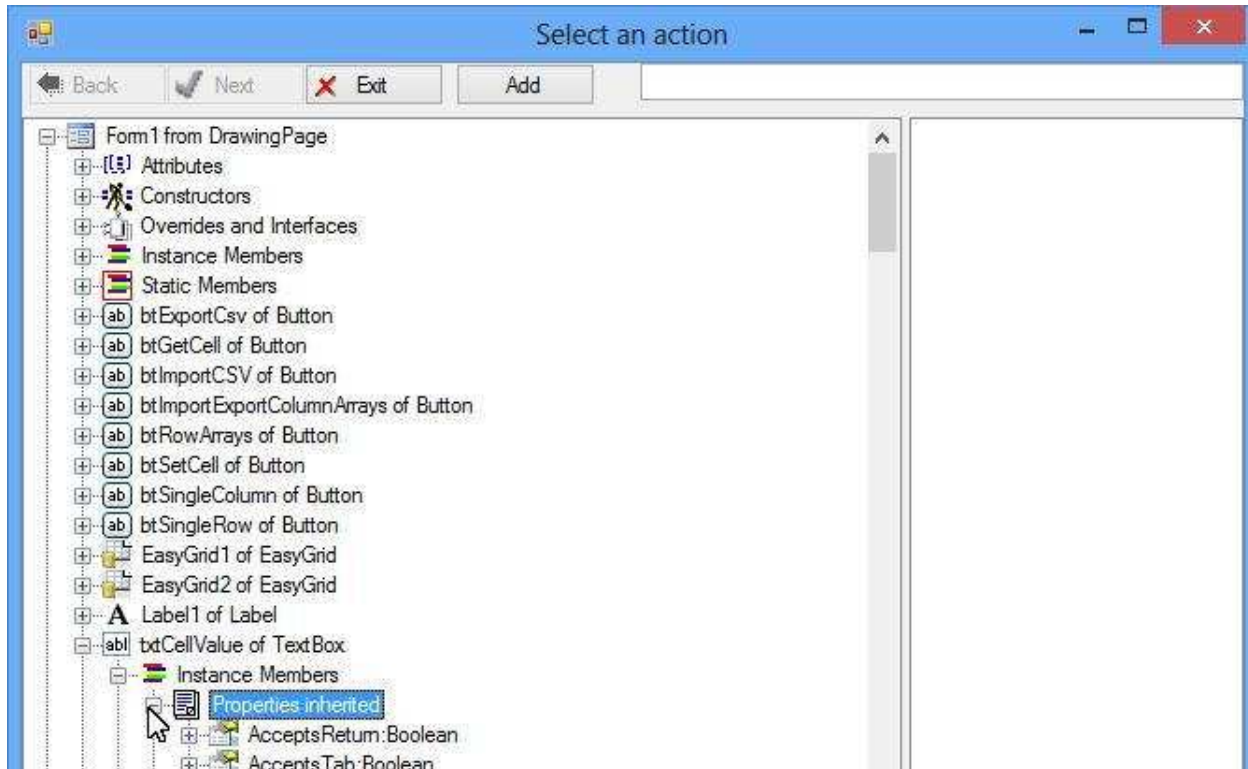
Get Element

Let's create an action to copy the element on the second row and second column to a text box; execute the action when clicking a button.

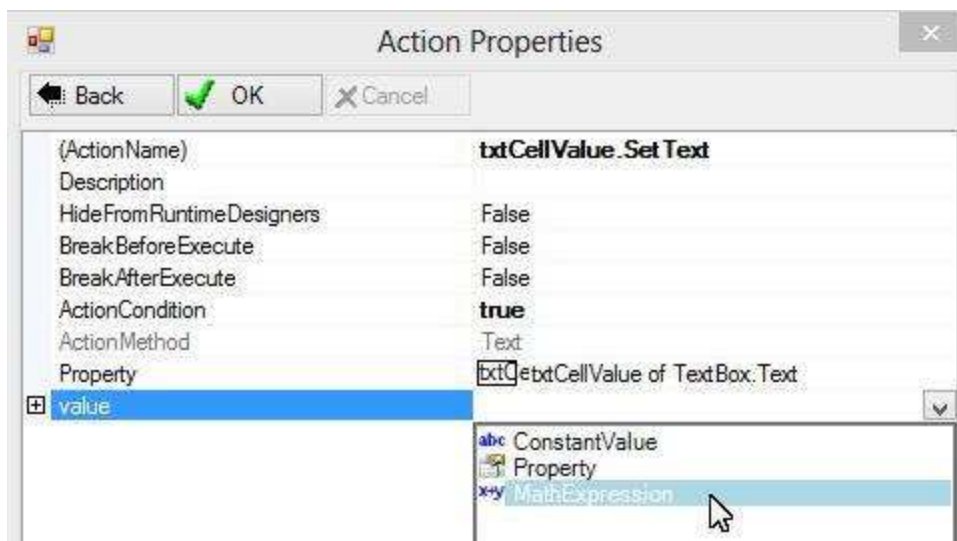


Array, CSV and Data Binding

Select the Text property of the text box:

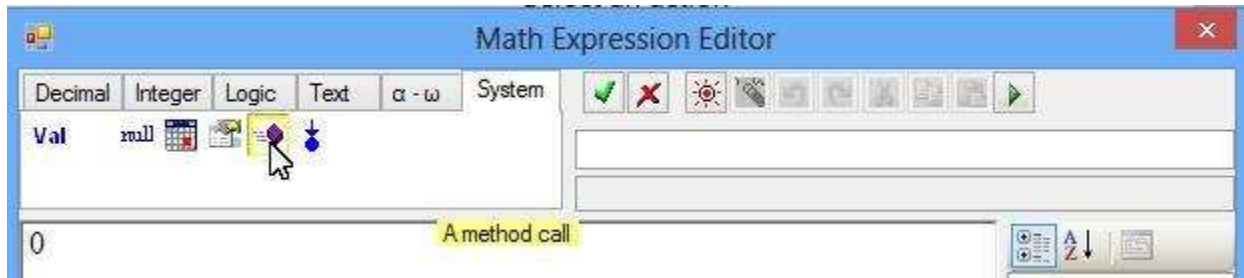


Select Math Expression:

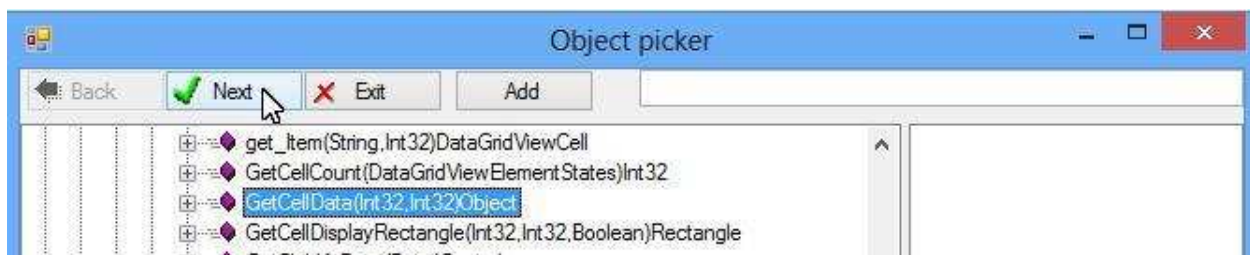
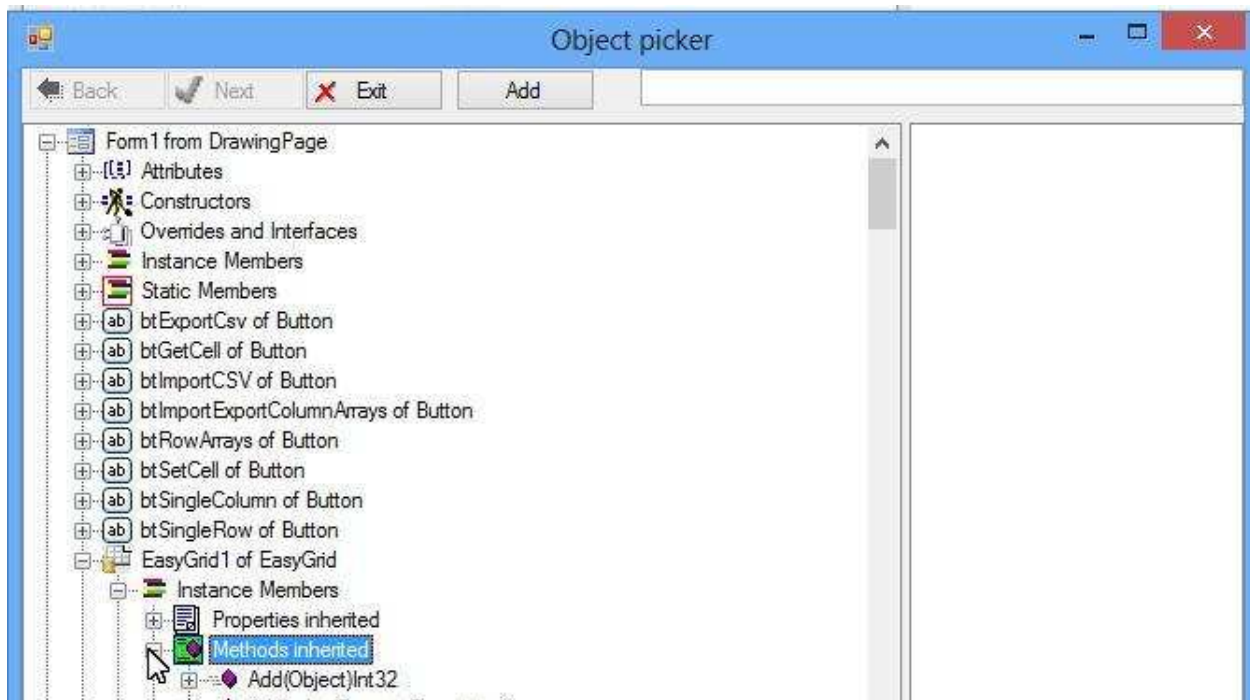


Array, CSV and Data Binding

Select Method icon:

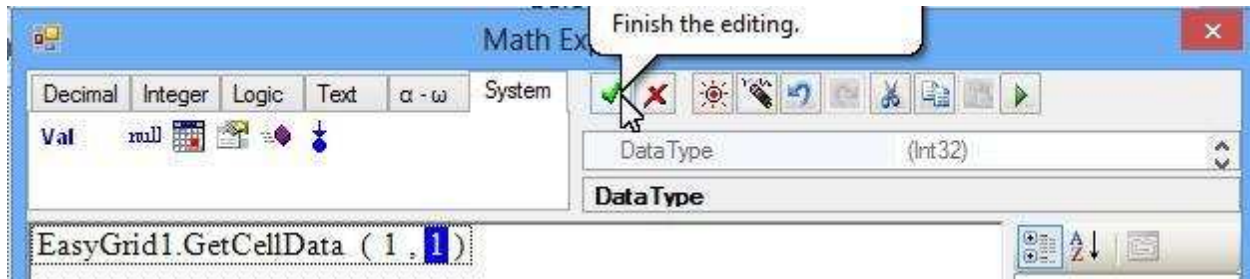


Select GetCellData of the EasyGrid:

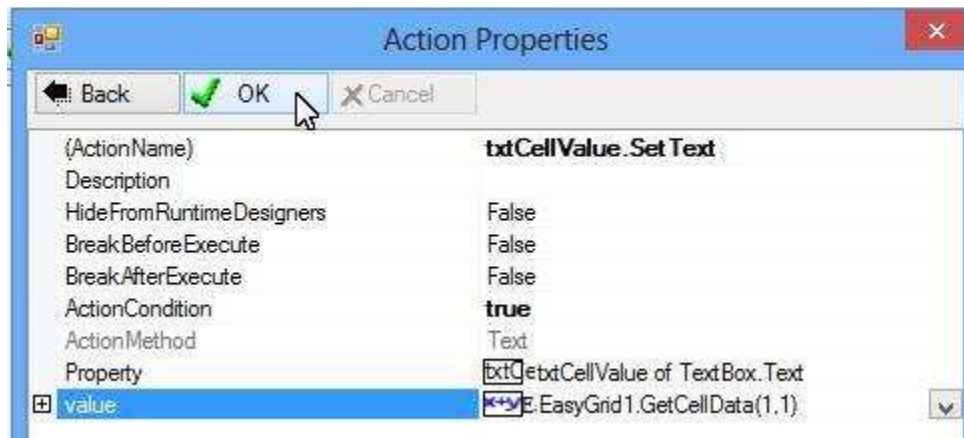


Use 1 and 1 to indicate the second row and the second column:

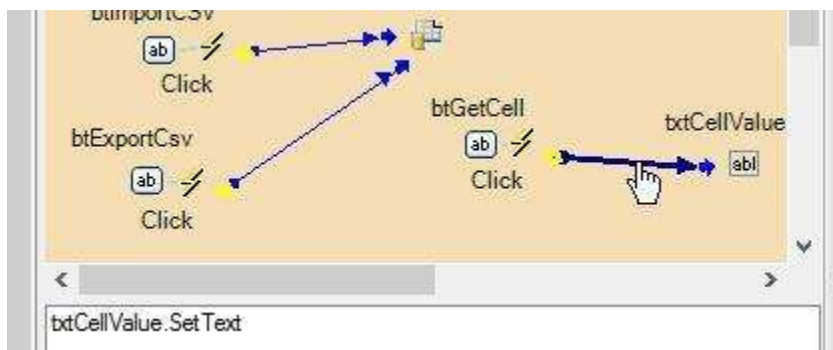
Array, CSV and Data Binding



Click OK:

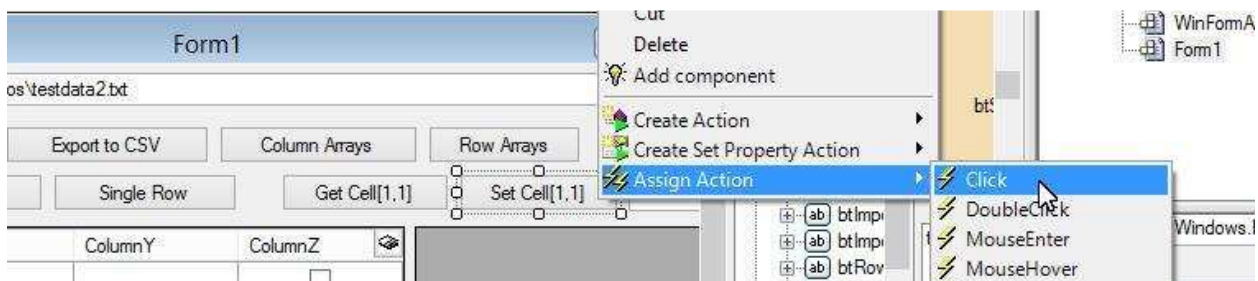


The action is created and assigned to the button:



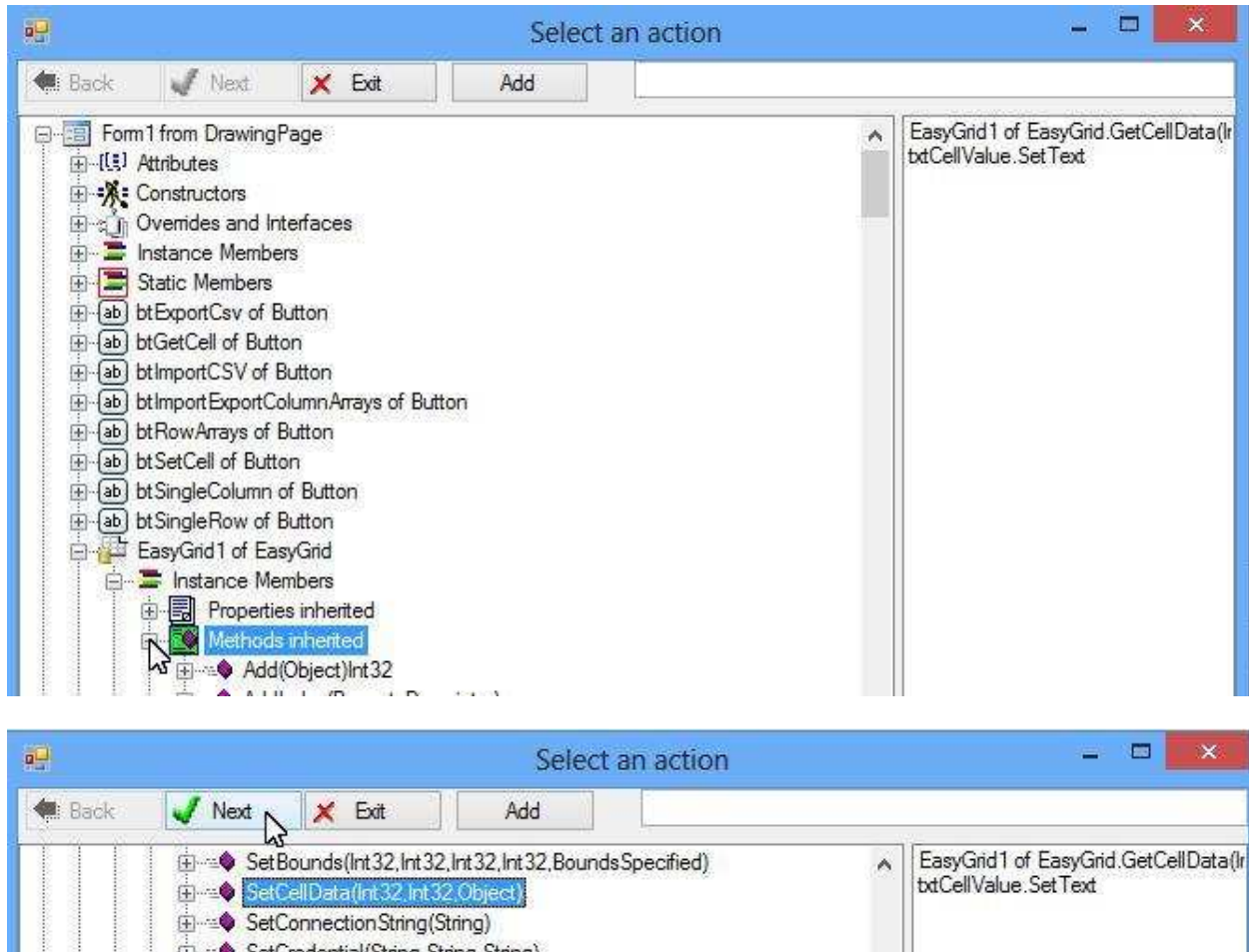
Set single element

Let's create an action to set the cell on the second row and second column to the content of a text box.



Array, CSV and Data Binding

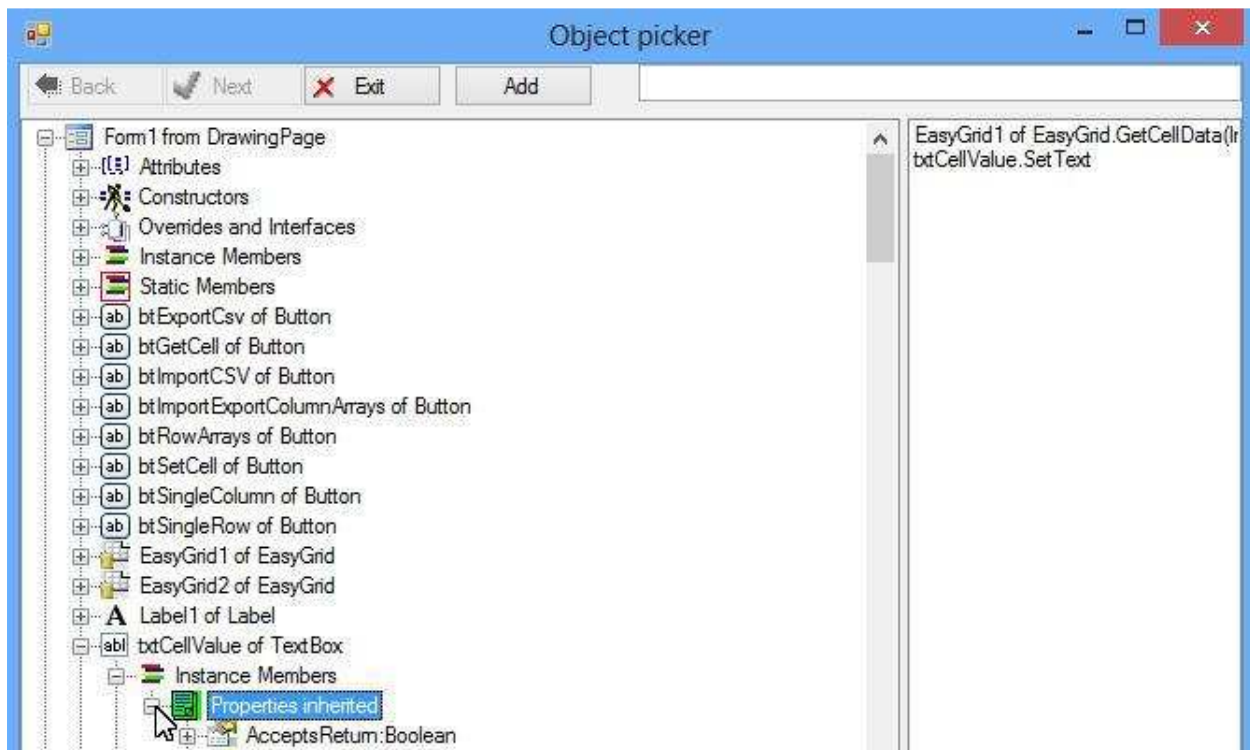
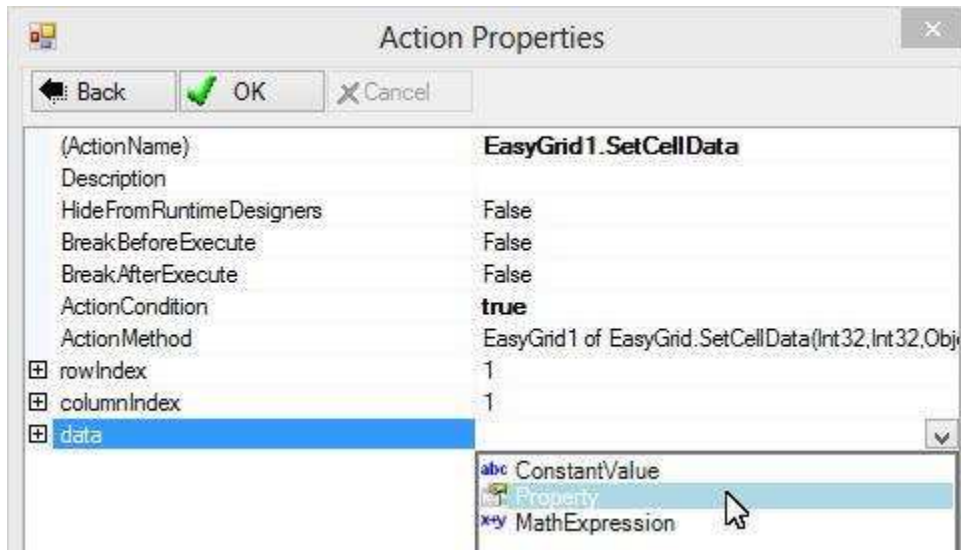
Select SetCellData of the EasyGrid:



Set rowIndex to 1 to indicate the second row; set columnIndex to 1 to indicate the second column.

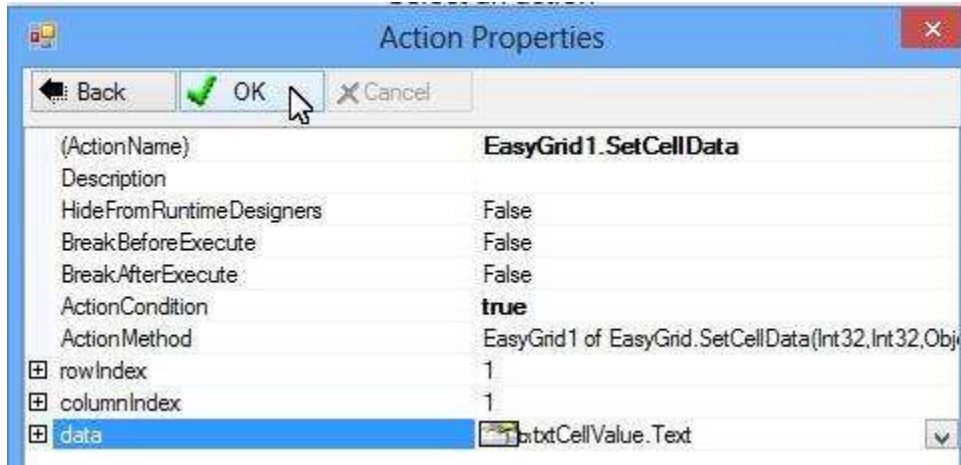
Select the Text property of the text box for the data:

Array, CSV and Data Binding

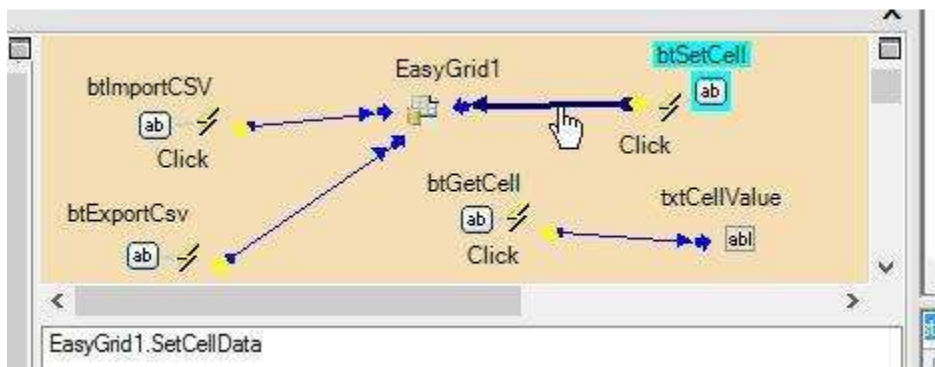


Click OK:

Array, CSV and Data Binding

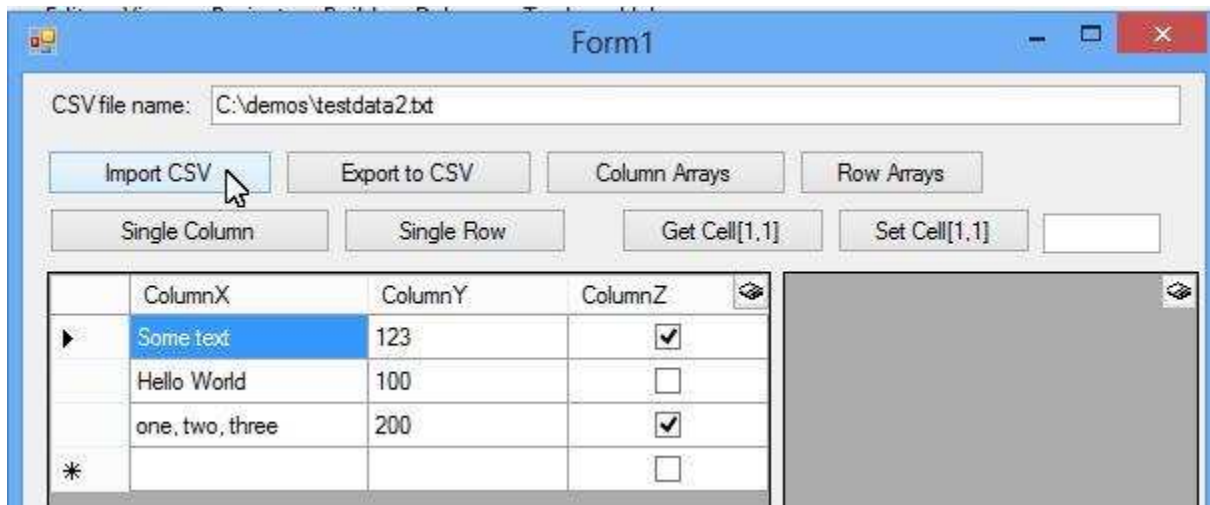


The action is created and assigned to the button:



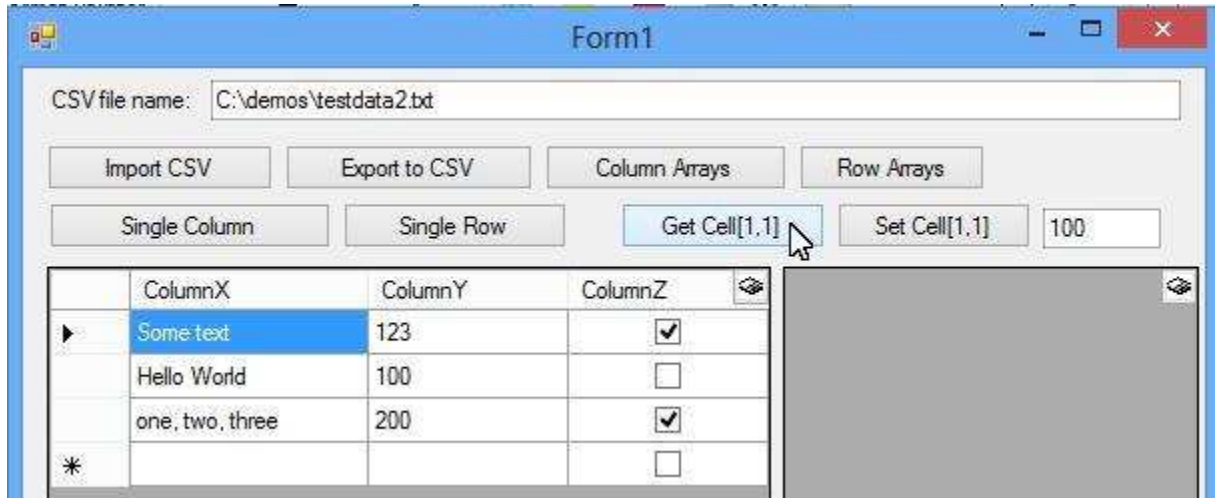
Test

Run the application. Load data from the CSV file:

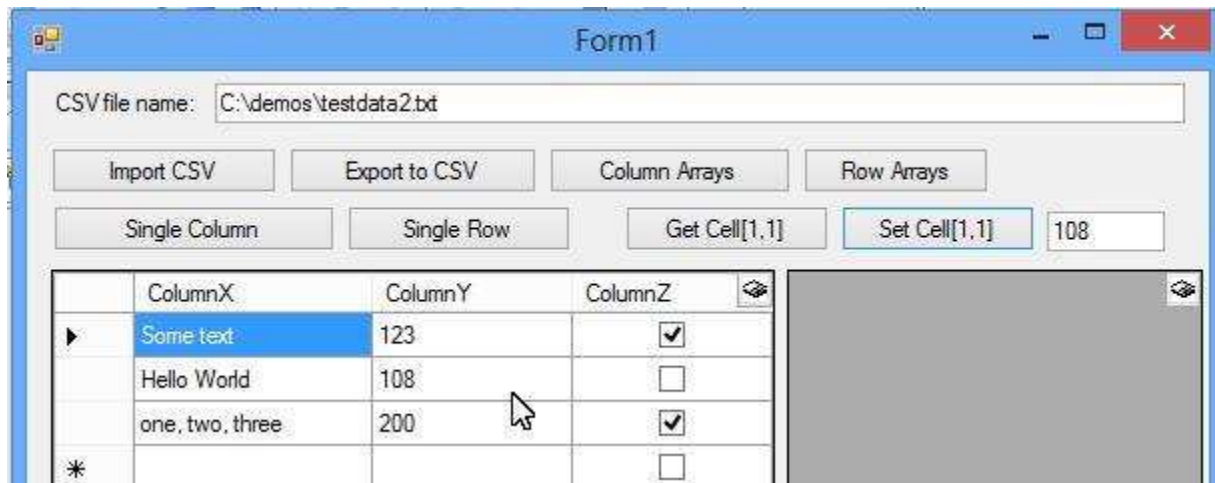


Click button "Get Cell [1,1]". The value on the second row and second column appears on the text box:

Array, CSV and Data Binding

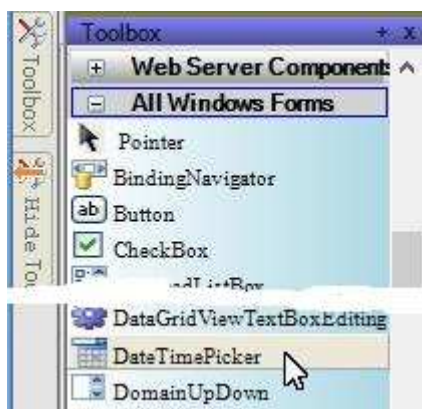


Modify value on the text box; click button “Set Cell [1,1]”. The modified value appears in the EasyGrid:

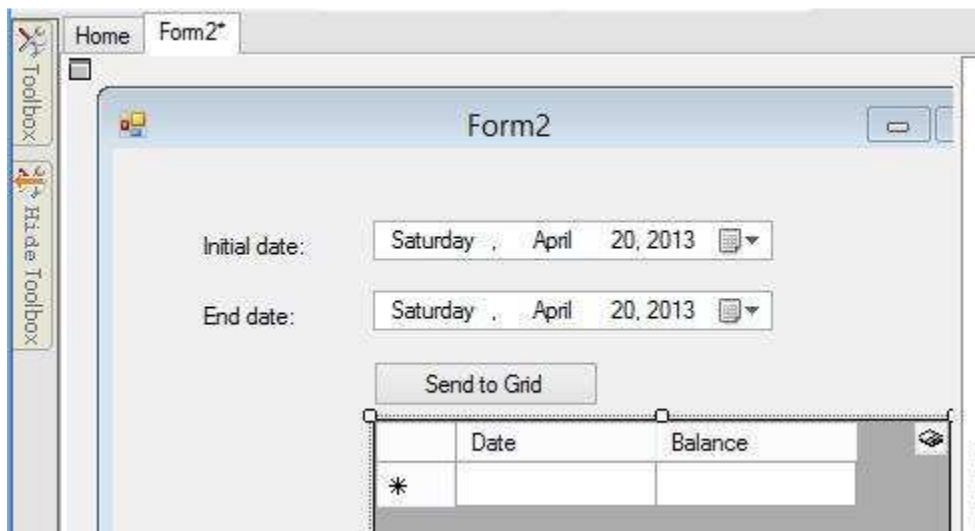
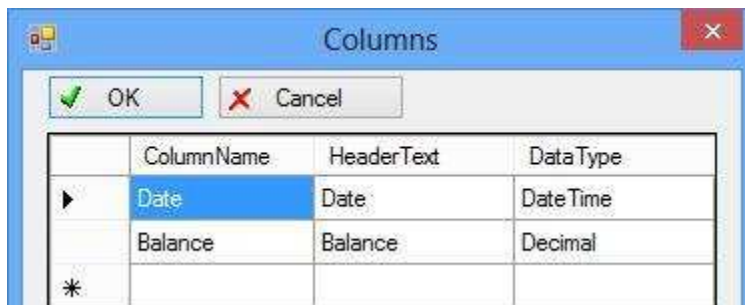
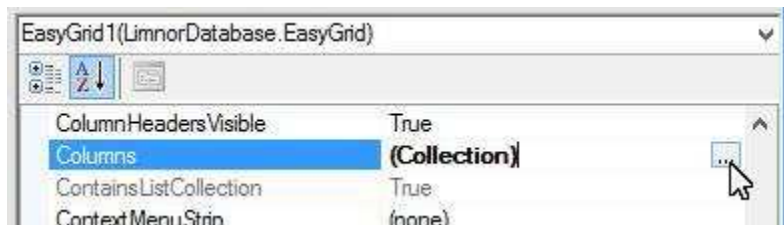
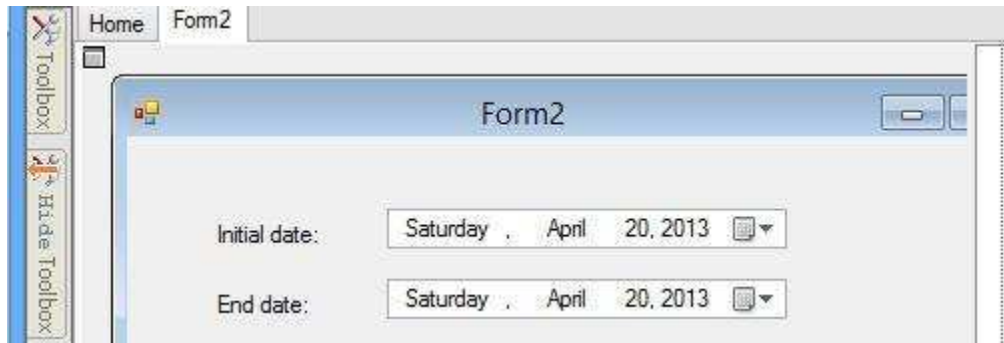


Sample: List all days between two dates

Suppose we use two DateTimePickers to specify two dates. Click a button; all days between the two dates are listed in an EasyGrid’s first column.



Array, CSV and Data Binding



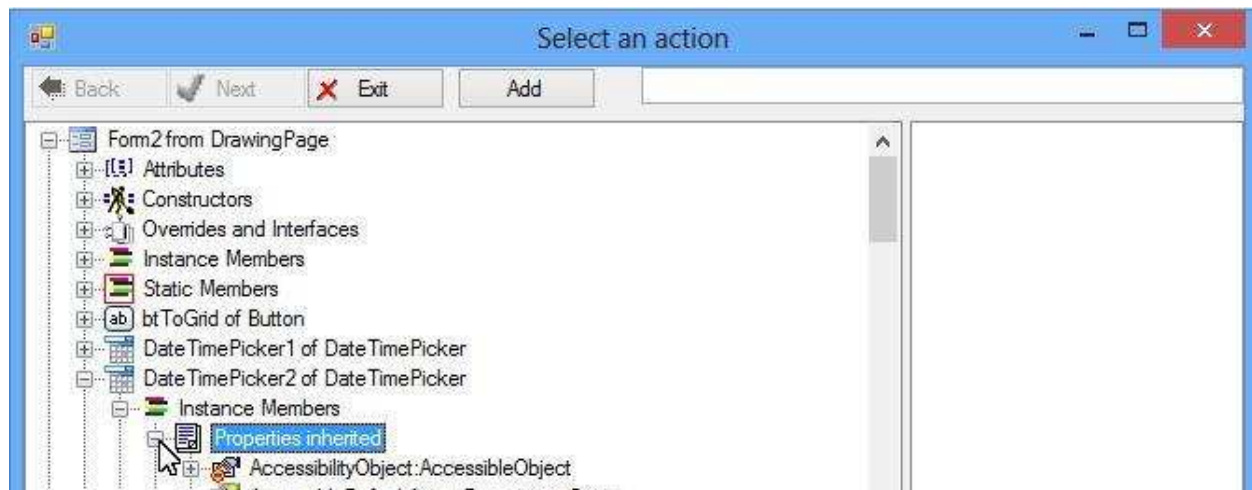
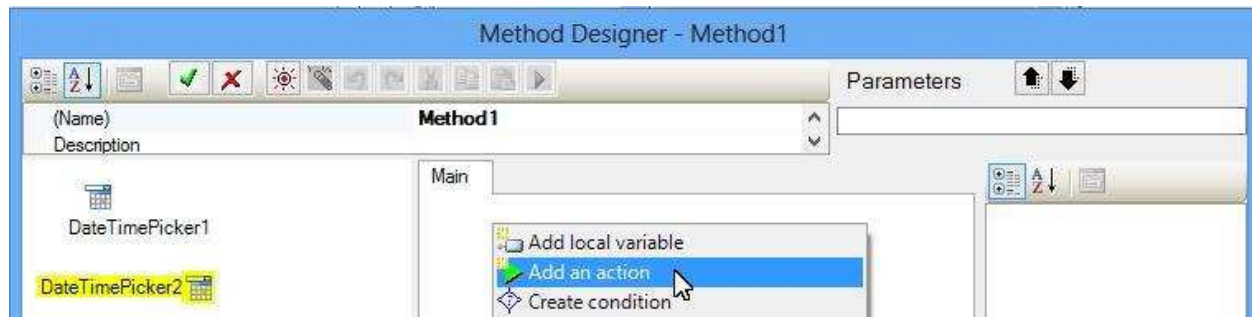
We know that we can use `ImportColumnData` action to do it if we can get an array of days. The question is how to get an array of days between two dates. There are many ways to do it. One example is to create a `TimeSpan` object between the two dates. The `TimeSpan` object has a property `Days` which tells us how many days there are. That will be our array size. Then, starting from the first date, we can use `AddDays` action to fill the array.

Array, CSV and Data Binding

We create a method to do it.



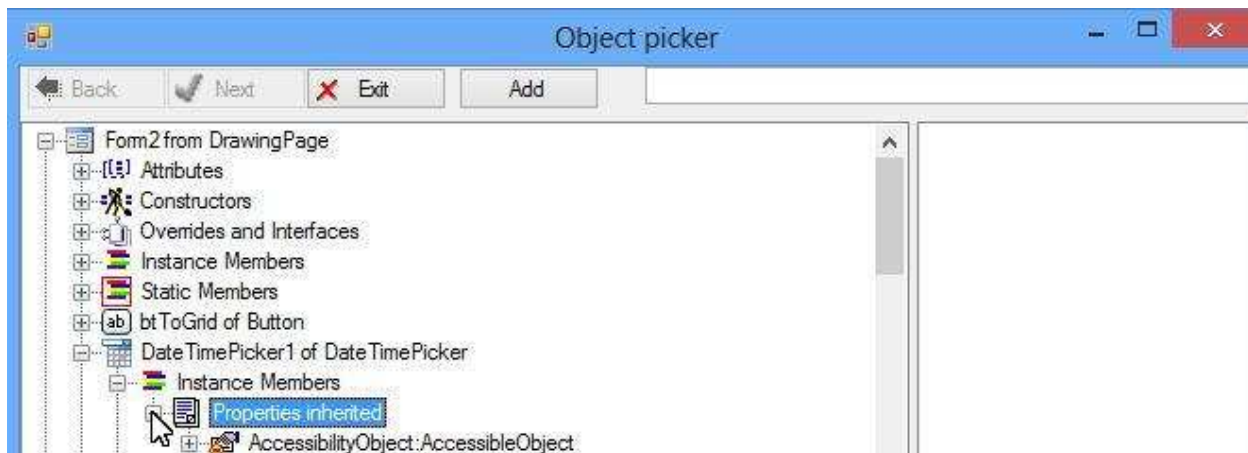
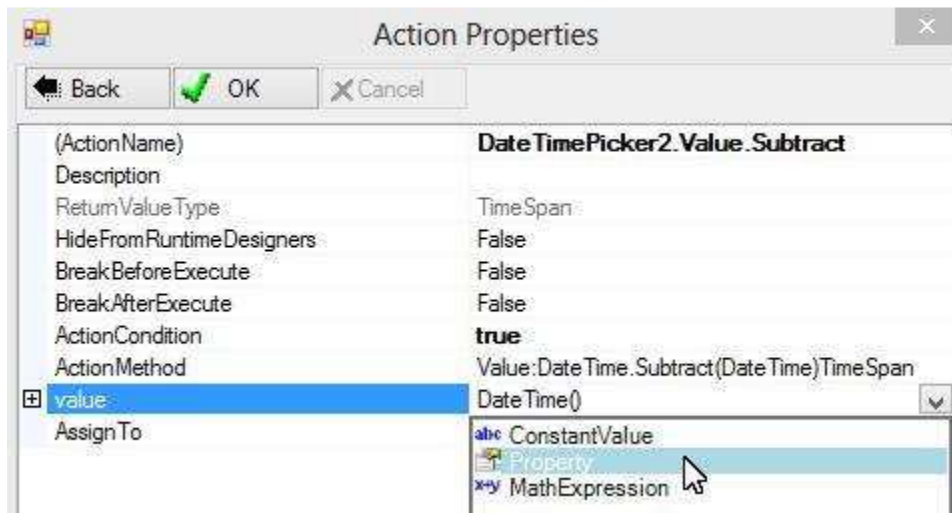
Create a Subtract action:



Array, CSV and Data Binding

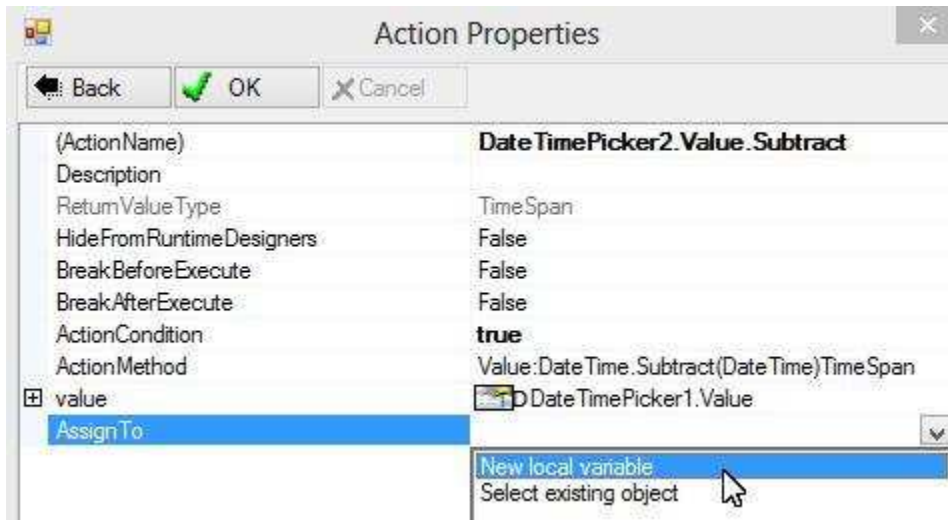


Select the Value of the first DateTimePicker:

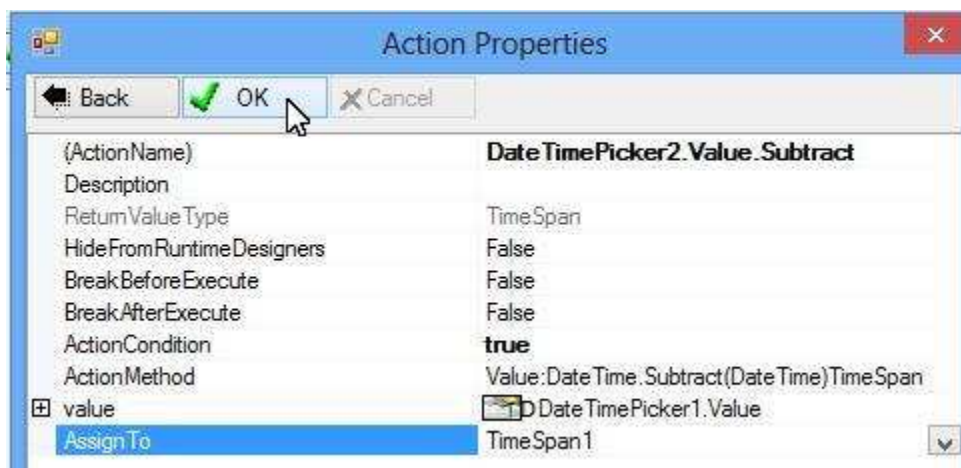


Select "New local variable" for "AssignTo":

Array, CSV and Data Binding

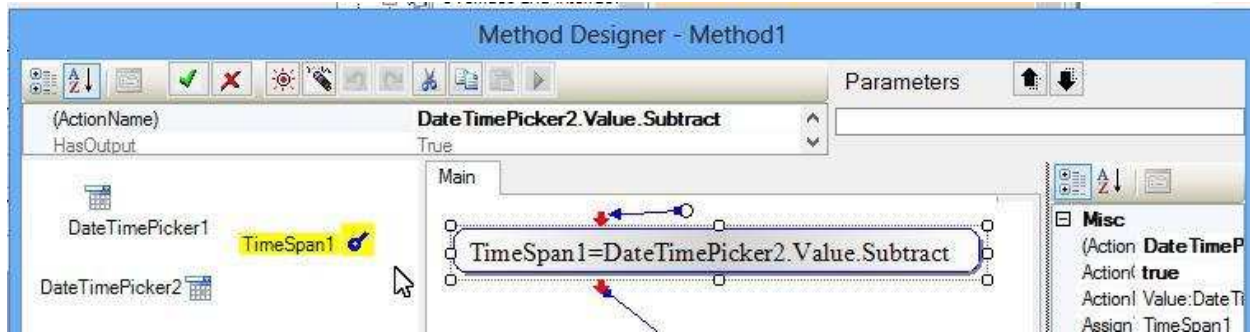


Click OK:

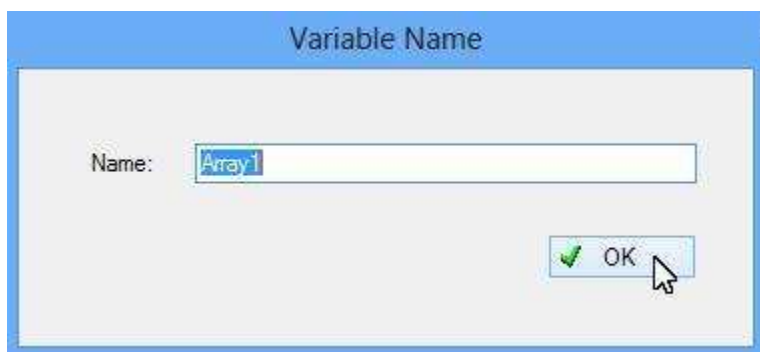
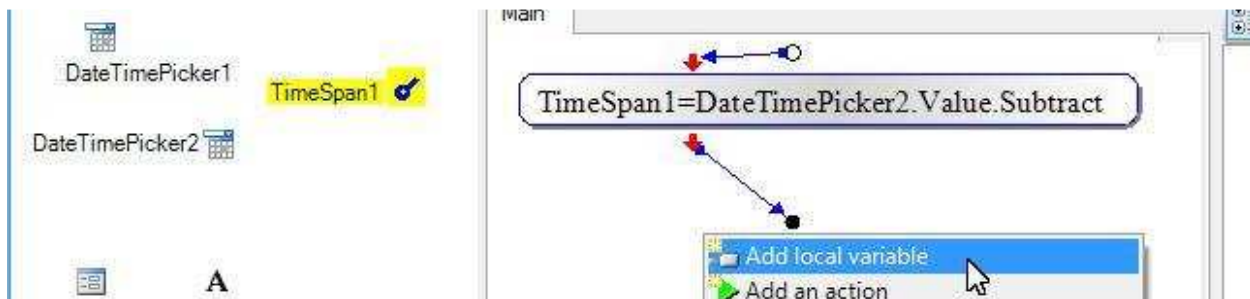


The Action and the variable are created:

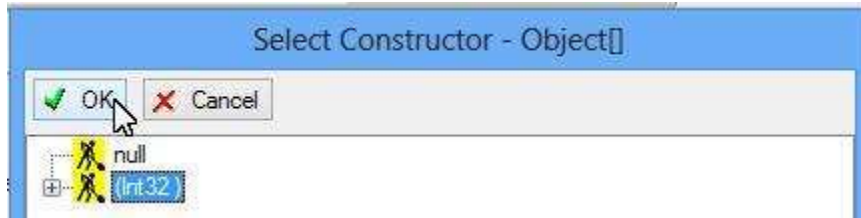
Array, CSV and Data Binding



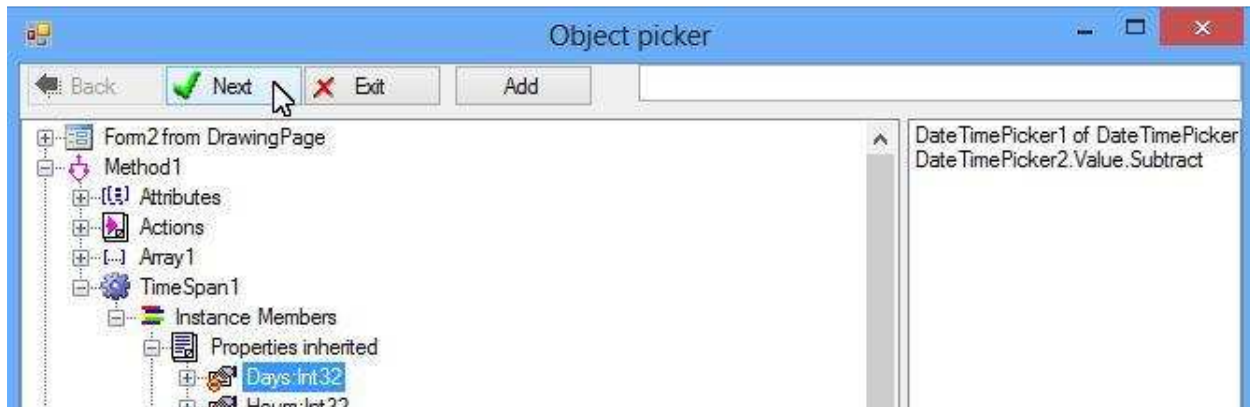
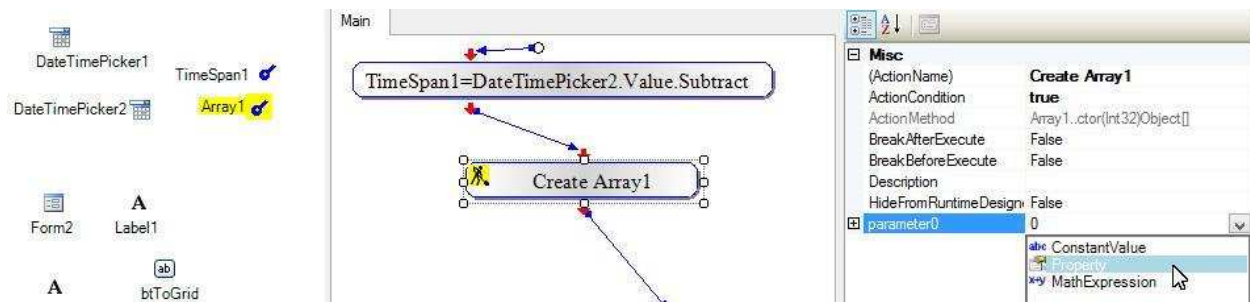
Create an array variable:



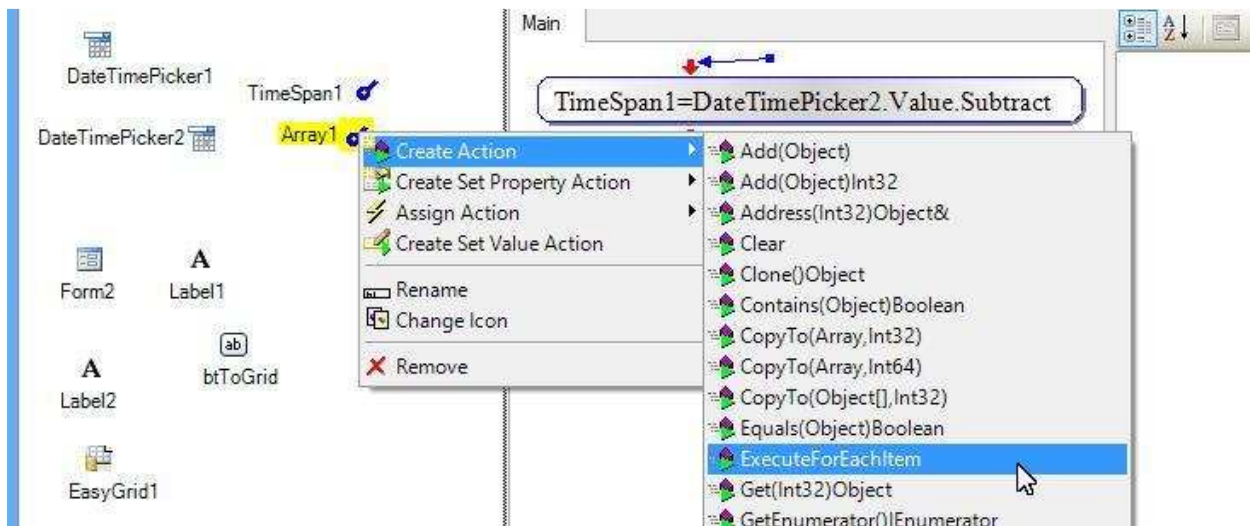
Array, CSV and Data Binding



An action is created for creating the array variable. Specify its size to the Days property of the TimeSpan:

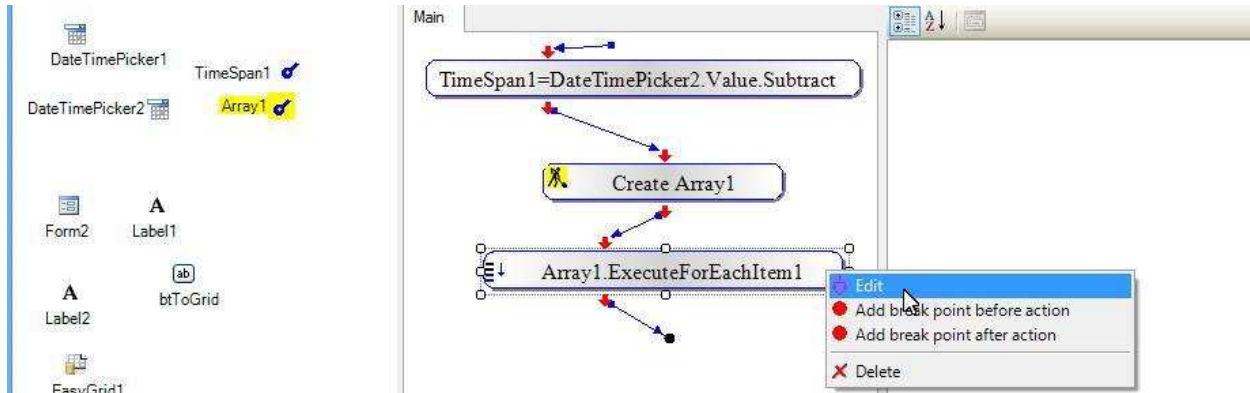


Create an action to go through the array and fill it with dates by adding days to the starting date:

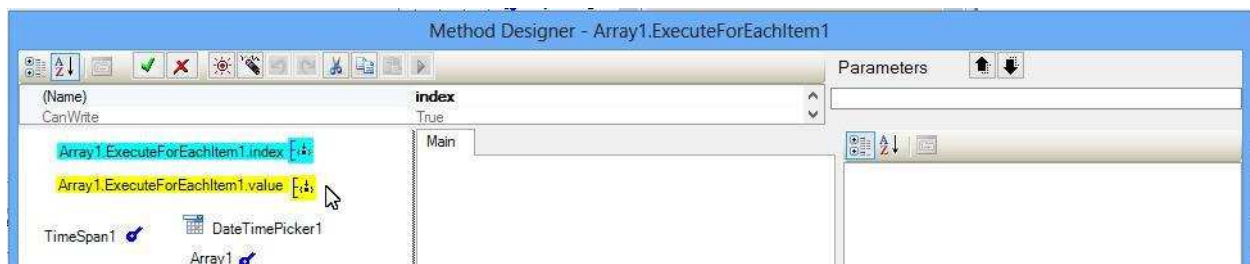


Array, CSV and Data Binding

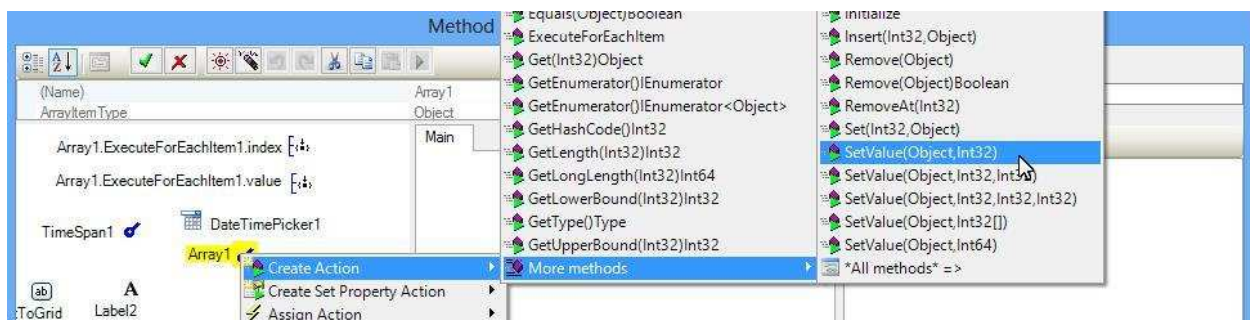
Edit the “for-each” action to set array item:



A new Method Editor appears:

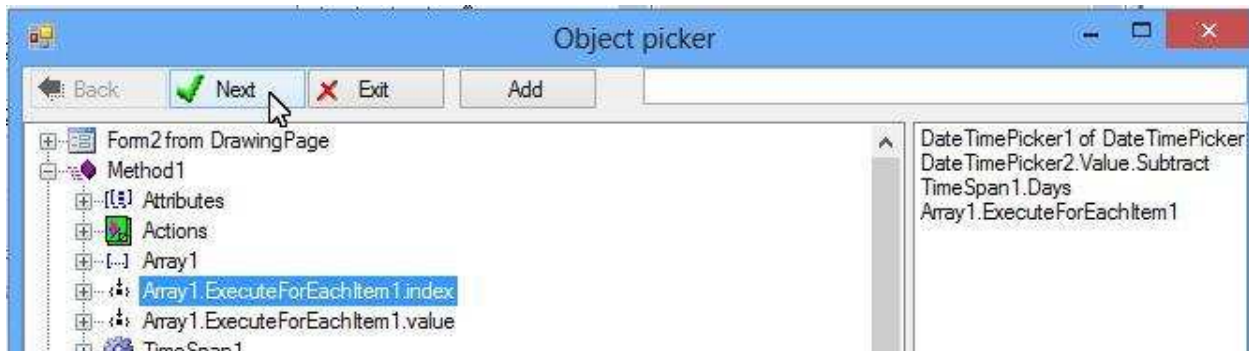
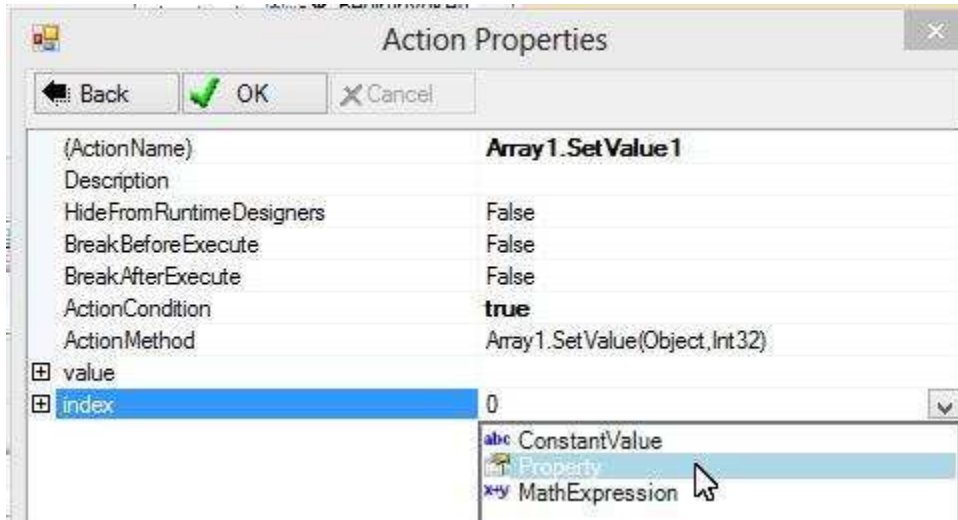


Note that “index” is the array index, which goes from 0, 1, 2, ..., to the array size minus 1. “value” is the array item. We may set array item identified by “index” to the first date adding “index” days:

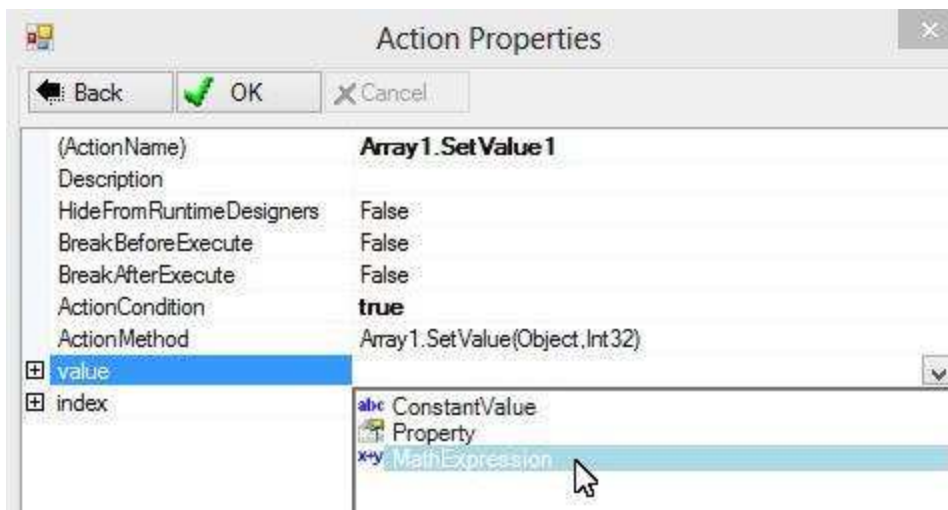


Set index to “index”:

Array, CSV and Data Binding

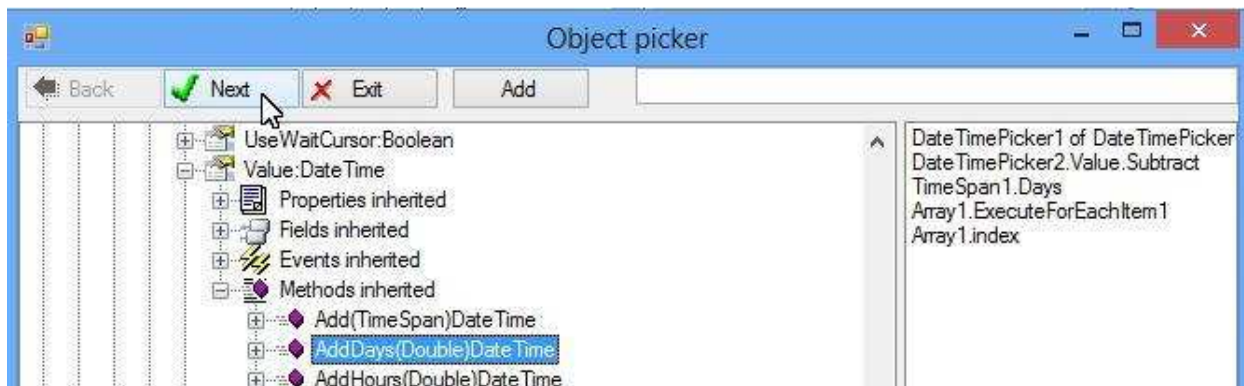
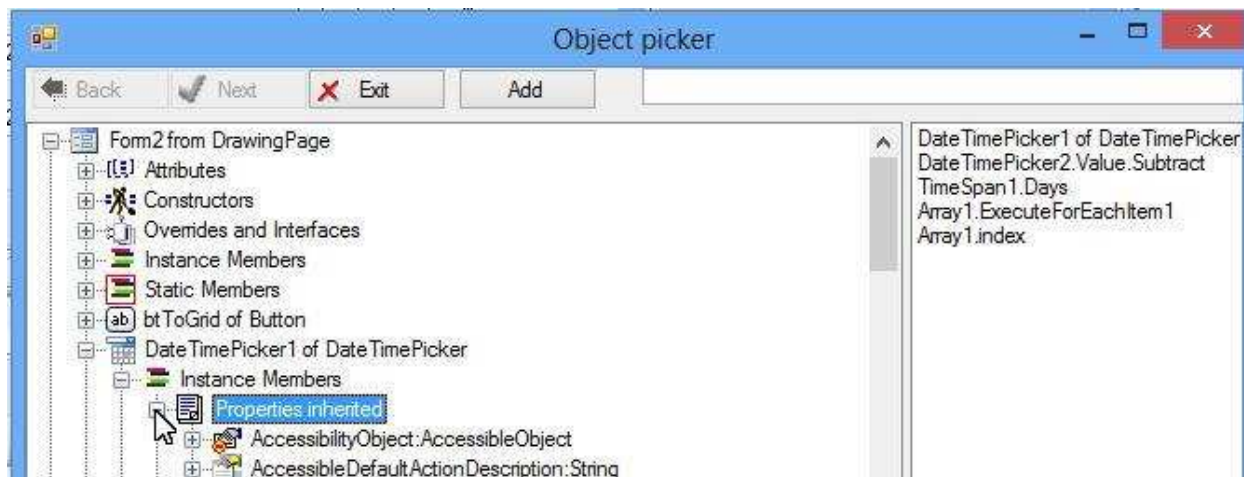
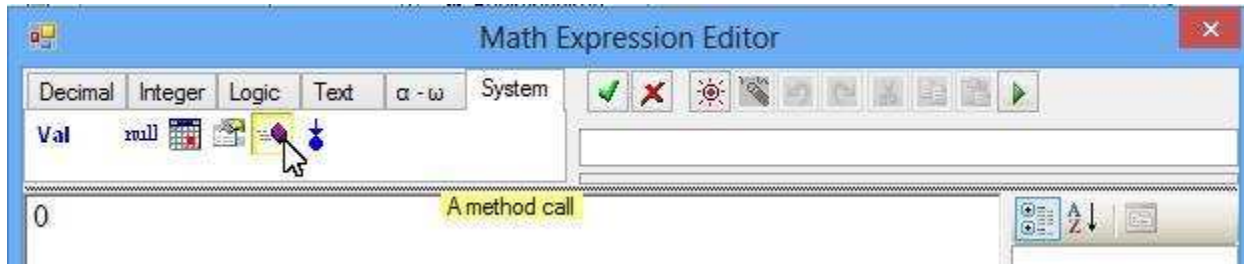


Set value to the first date adding "index" days:

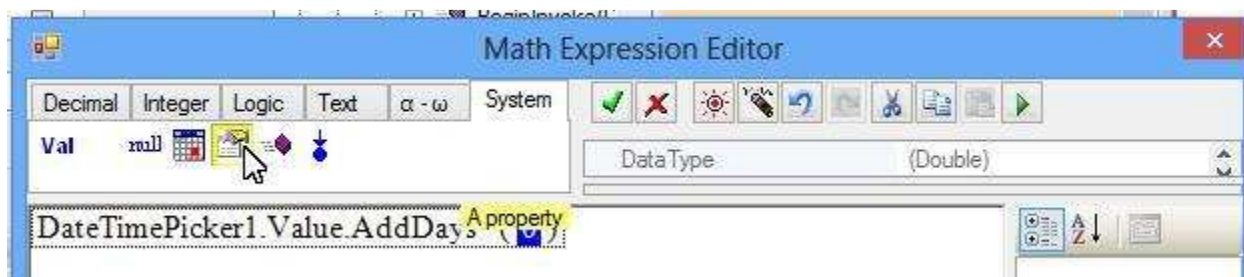


Choose AddDays method of the first DateTimePicker's Value property:

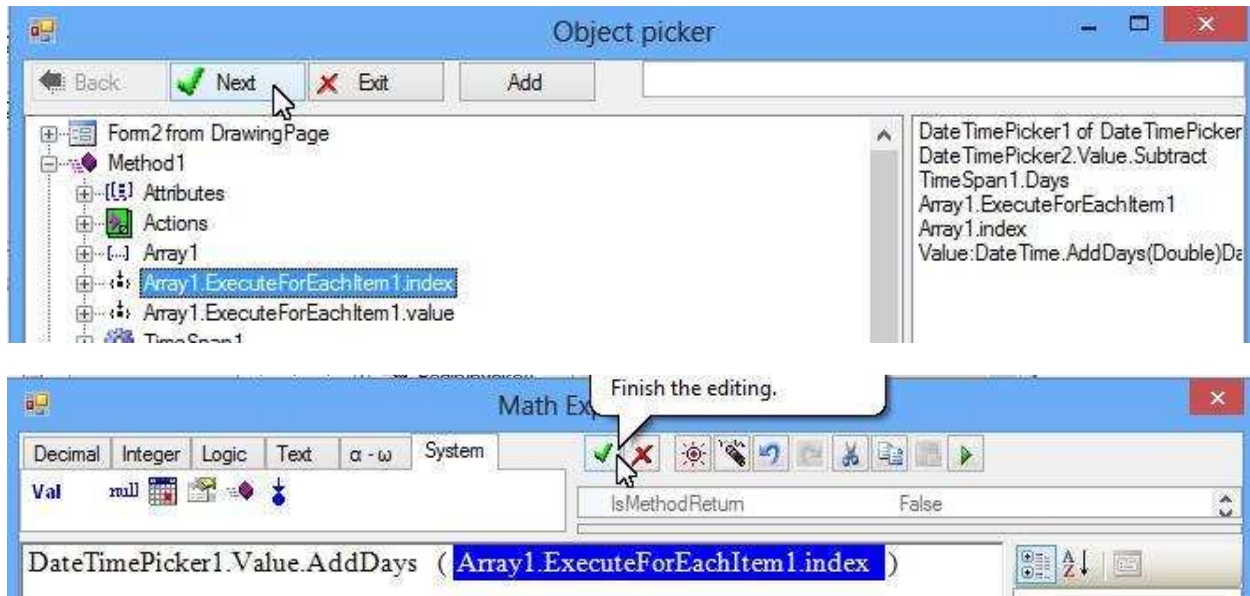
Array, CSV and Data Binding



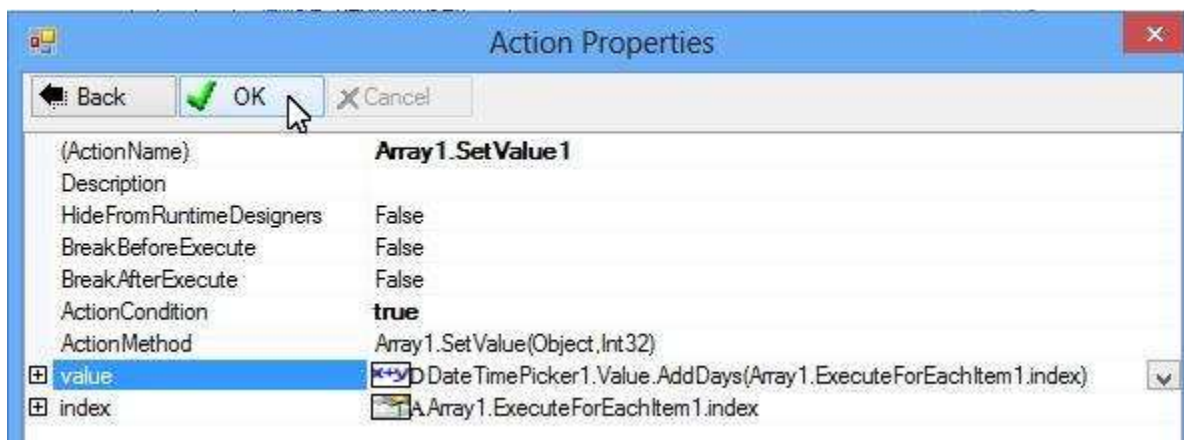
Select "index" for the days:



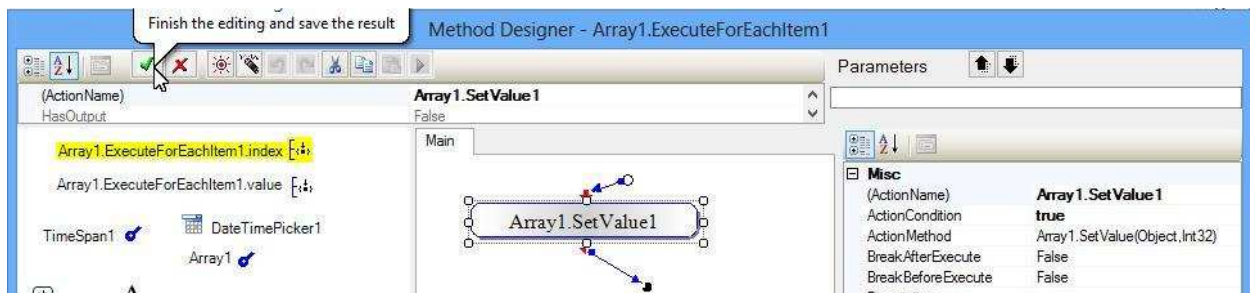
Array, CSV and Data Binding



Click OK to finish creating the action:

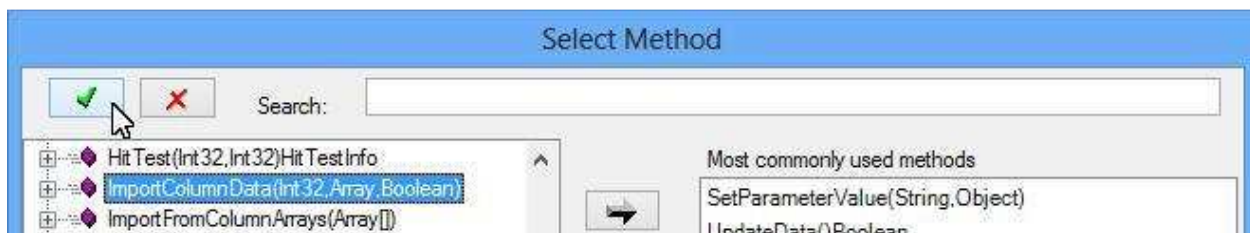
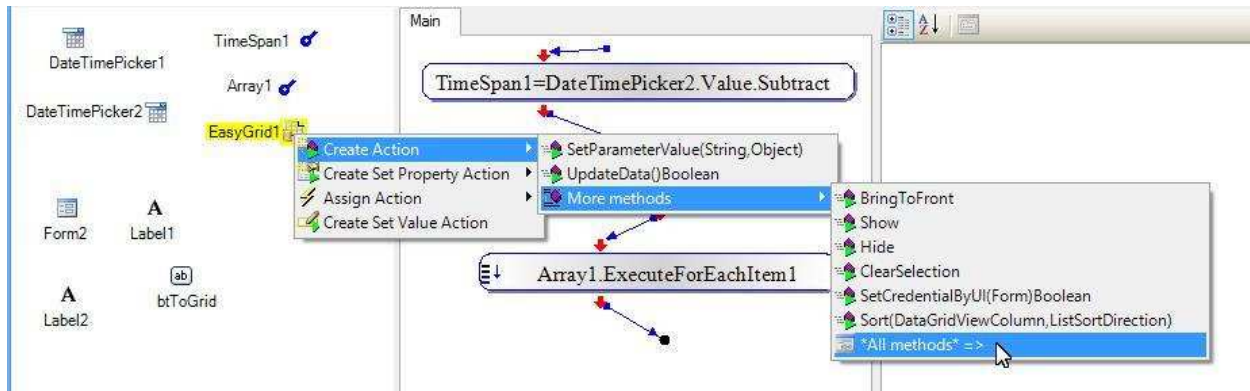


We just need this one action for processing the array item:

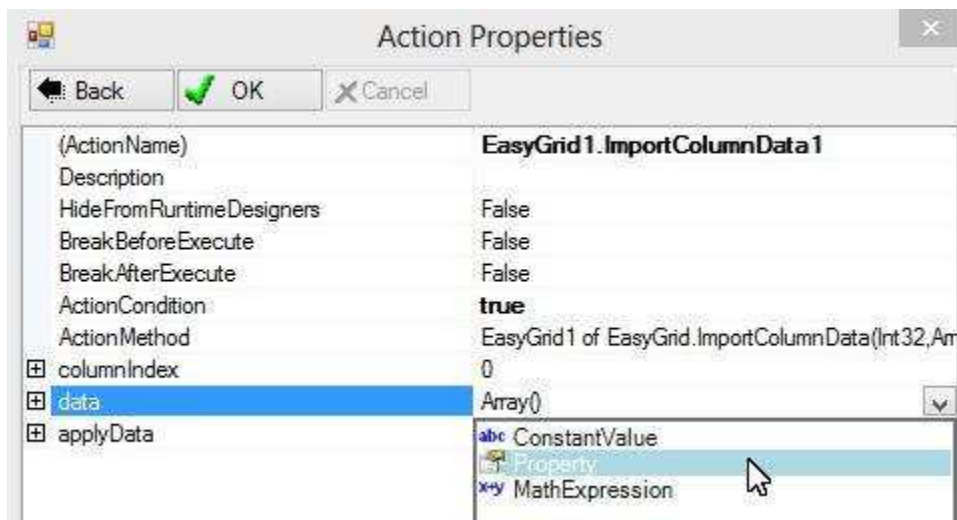


Now we have an array of dates. We may import it to the EasyGrid:

Array, CSV and Data Binding

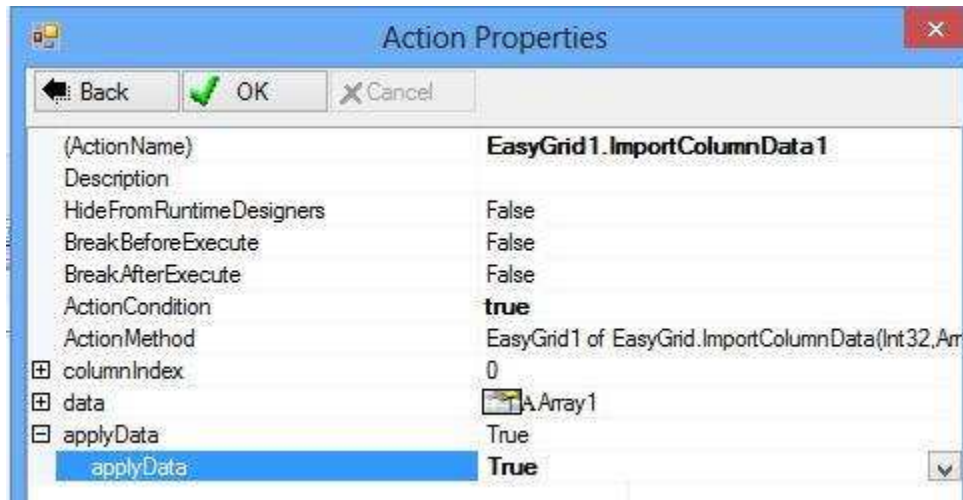


Select the array variable as the data:

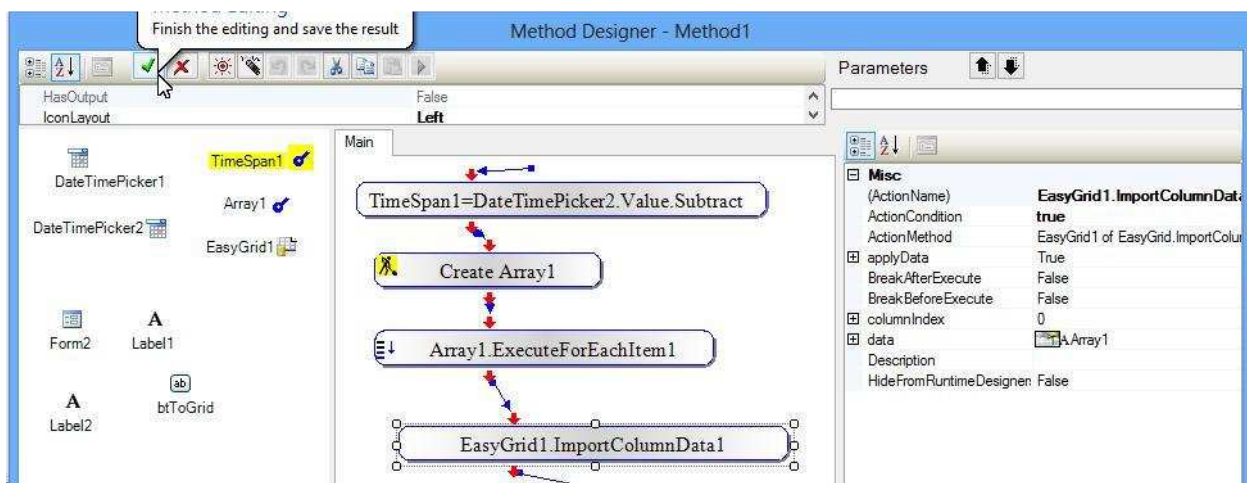
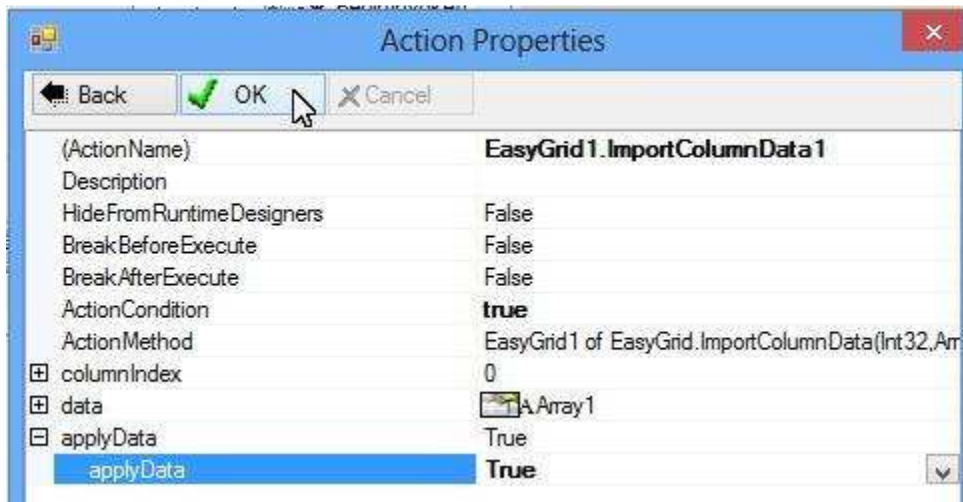


Set applyData to true:

Array, CSV and Data Binding

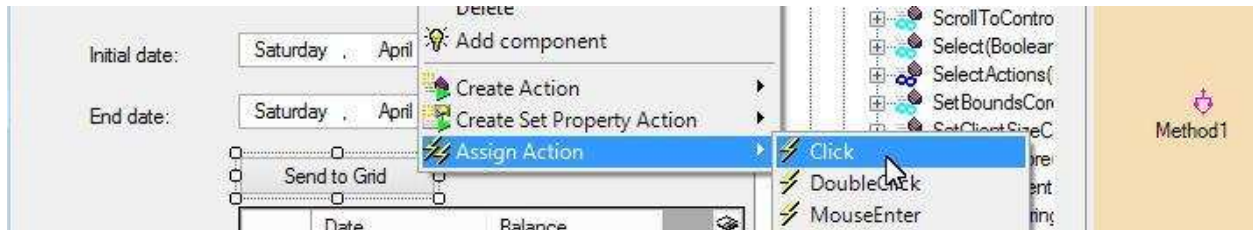


columnIndex is 0 to indicate the first column. Click OK to finish creating the action:

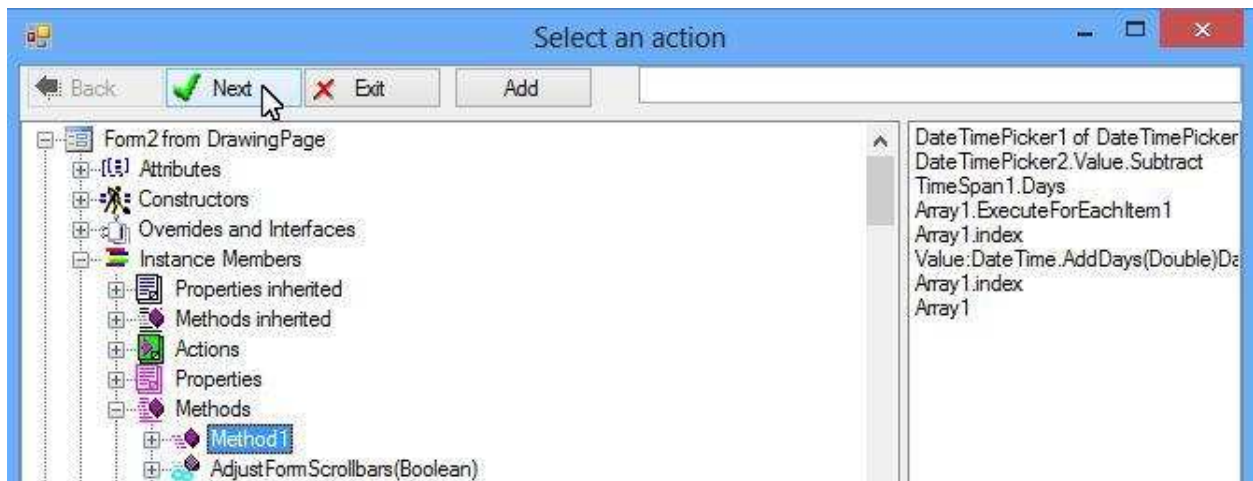


Let's execute the method at Click event of the button:

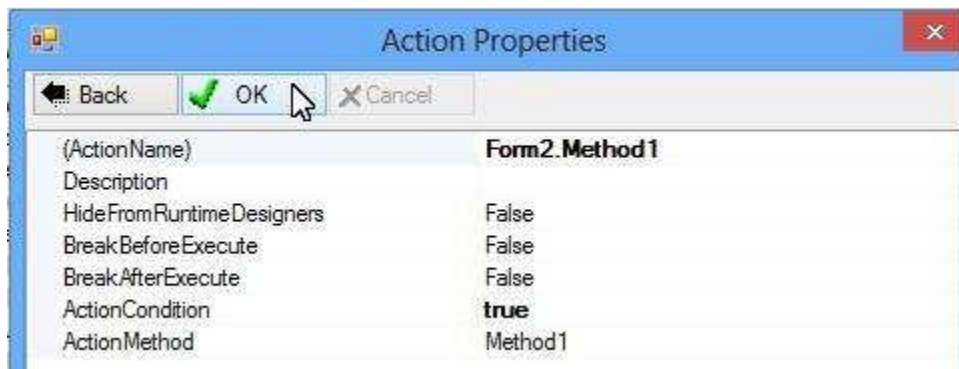
Array, CSV and Data Binding



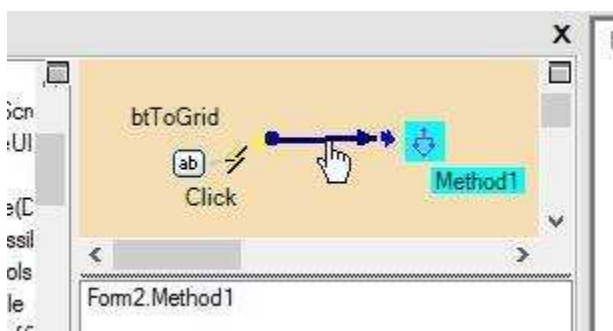
Select the method:



Click OK:

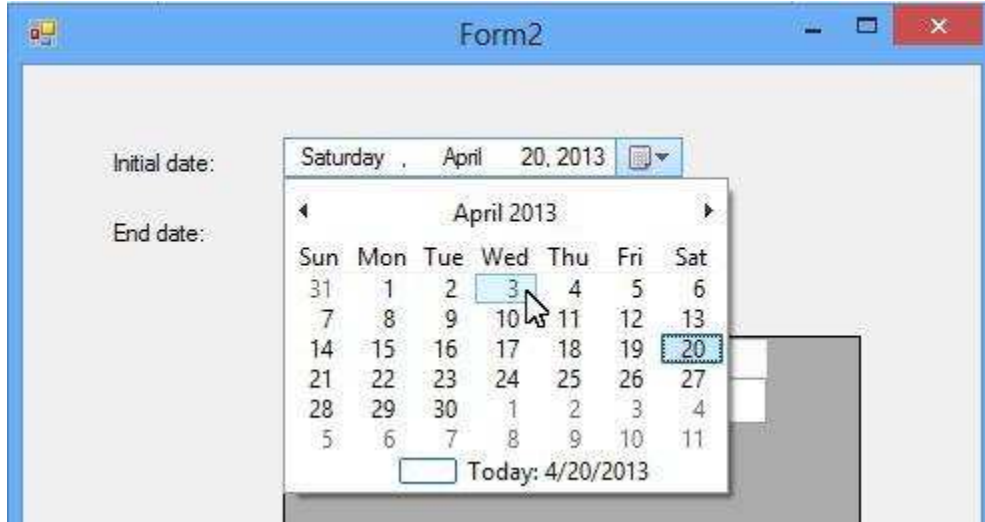


The action is created and assigned to the button:

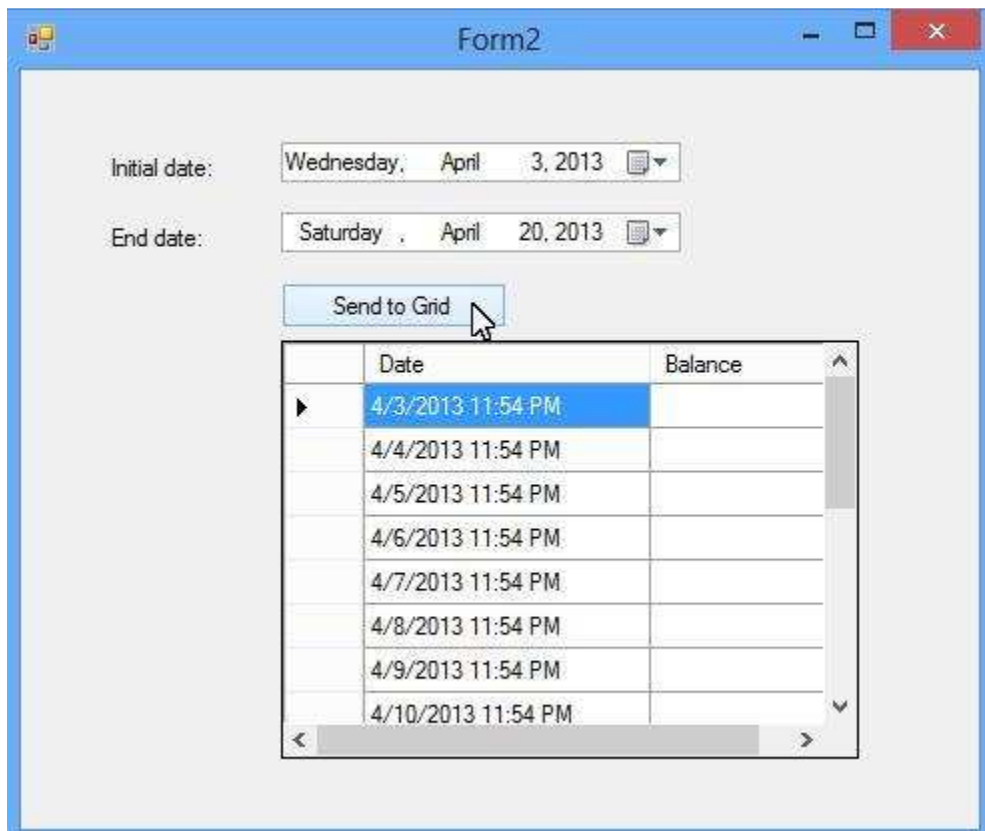


Array, CSV and Data Binding

Run the application. The form appears. Choose dates:



Click the button. The dates appear in the first column:



| | Date | Balance |
|---|--------------------|---------|
| ▶ | 4/3/2013 11:54 PM | |
| | 4/4/2013 11:54 PM | |
| | 4/5/2013 11:54 PM | |
| | 4/6/2013 11:54 PM | |
| | 4/7/2013 11:54 PM | |
| | 4/8/2013 11:54 PM | |
| | 4/9/2013 11:54 PM | |
| | 4/10/2013 11:54 PM | |

Feedback

Please send your suggestions and questions to support@limnor.com