Using the Multi-select Text Edit Field Setup for Creatio

From Customer FX Corporation – customerfx.com

To use the Multi-select text edit field setup for Creatio, some development skills are required. However, the steps are fairly simple. This control module will convert a regular text field into a multi-select field where the user can select items from a lookup to populate the text control. The selected items will be added to the control as a comma-delimited list and stored as text.

Step 1

The first step is to use the page wizard to add a text field as you would normally. Drag a new text field onto the page (or select an existing one).

Step 2

Click the Source Code tab for the page. In the very top line, you will need to add "CfxMultiselectEdit" to the modules list in the [] at the top of the page code, with quotes. For example, if you're adding this to the Contact page, the top line will look like this:

```
define("ContactPageV2", [], function() {
```

You will change that to look like this:

```
define("ContactPageV2", ["CfxMultiselectEdit"], function() {
```

Step 3

You will need to locate the item you added to the page in step 1. To do this, simply click into the code part of the window and CTRL+F to search for the column name for the text control. Once located, you will need to add the following lines into the “values” section, in this example, I will be using the ContactType lookup for the control. **Change ContactType to the object you want for the lookup**. The first two lines will stay as-is, without changes. You'll modify the 3rd and 4th line as needed.

```
"itemType": Terrasoft.ViewItemType.COMPONENT,
"className": "Terrasoft.CfxMultiselectEdit",
"lookupEntity": "ContactType",
```

Note, if you'd like to override the caption, you can use a string, as shown above or use a localizable string as follows (if no override, the column's title will be shown):

```
"labelConfig": {
    "caption": { "bindTo": "Resources.Strings.MyCaption" }
}
```

Or

```
"labelConfig": {
    "caption": "My caption"
}
```

Step 4

The last change to make is to the bindTo line of the text field. Change that from this:

```
"bindTo": "UsrMyProperty",
```

To the following:

```
"value": { "bindTo": "UsrMyProperty" },
```

That is all. The control should now be working. You can see a complete video tutorial of these steps here: https://www.youtube.com/watch?v=agoghd5CnVo

The complete change to the text field on the page would change from this:

```
{
      "operation": "insert",
      "name": "STRING26a06da4-f972-4f2e-98bf-47e0b37b5cc2",
      "values": {
            "layout": {
                  "colSpan": 12,
                  "rowSpan": 1,
                  "column": 0,
                  "row": 4,
                  "layoutName": "ContactGeneralInfoBlock"
            },
            "bindTo": "UsrMultiContactType",
            "enabled": true
      },
      "parentName": "ContactGeneralInfoBlock",
      "propertyName": "items",
      "index": 7
}
```

To this:

```
{
      "operation": "insert",
      "name": "STRING26a06da4-f972-4f2e-98bf-47e0b37b5cc2",
      "values": {
            "itemType": Terrasoft.ViewItemType.COMPONENT,
            "className": "Terrasoft.CfxMultiselectEdit",
            "lookupEntity": "ContactType",
            "layout": {
                  "colSpan": 12,
                  "rowSpan": 1,
                  "column": 0,
                  "row": 4,
                  "layoutName": "ContactGeneralInfoBlock"
            },
            "value": { "bindTo": "UsrMultiContactType"},
            "enabled": true
      },
      "parentName": "ContactGeneralInfoBlock",
      "propertyName": "items",
      "index": 7
}
```

Tip: It’s also a good idea to give the “name” a better, more descriptive value

Lookup Filters (optional)

Optionally, you can also add a filter for the lookup by adding a “filters” property to the control in the diff with a function that returns a valid EntitySchemaQuery filter or filter group. To add filters, add a function that returns an EntitySchemaQuery filter in the methods. Then, bind the function to the filters property.

For example, first bind a function to the filters property:

```
      {
            "operation": "insert",
            "name": "STRING26a06da4-f972-4f2e-98bf-47e0b37b5cc2",
            "values": {
                  "itemType": Terrasoft.ViewItemType.COMPONENT,
                  "className": "Terrasoft.CfxMultiselectEdit",
                  "lookupEntity": "ContactType"
                  "filters": { "bindTo": "getMultiLookupFilters" },
                  "layout": {
                        "colSpan": 12,
                        "rowSpan": 1,
                        "column": 0,
                        "row": 4,
                        "layoutName": "ContactGeneralInfoBlock"
                  },
                  "value": { "bindTo": "UsrMultiContactType"},
                  "enabled": true
            },
            "parentName": "ContactGeneralInfoBlock",
            "propertyName": "items",
            "index": 7
      }
```

Then, in the methods, add the function to return the ESQ filters:

```
getMultiLookupFilters: function() {
      return Terrasoft.createColumnFilterWithParameter(
            Terrasoft.ComparisonType.EQUAL, "Name", "Customer");
}
```

Allowing Text Edit (optional)

Another option allows for whether the user can edit the text in the control without using the lookup. By default, the control will not allow the user to type into the text field and only populate it by opening the lookup and selecting items. You can change this behavior to allow the user to use either the lookup, or to type directly into the text field by adding the following property:

```
      "allowTextEdit": true
```