



Displaying & selecting multiple records

Pattern #3

When you wish to display multiple records to a user in OutSystems you can use either the Table or the List widgets. When you want to allow the user to input data you will use Inputs, Checkboxes, Dropdowns and other input widgets, typically in a Form. But sometimes you need to allow a user to select one or more records of a list or table.

The first part of this exercise modifies a listing page so the user can select multiple records and perform operations on all of them at once. In the second part, you will store the user selection as a relationship. Finally, in the third part you will need to enhance another listings page by displaying a secondary list in one of the columns of the original listing.

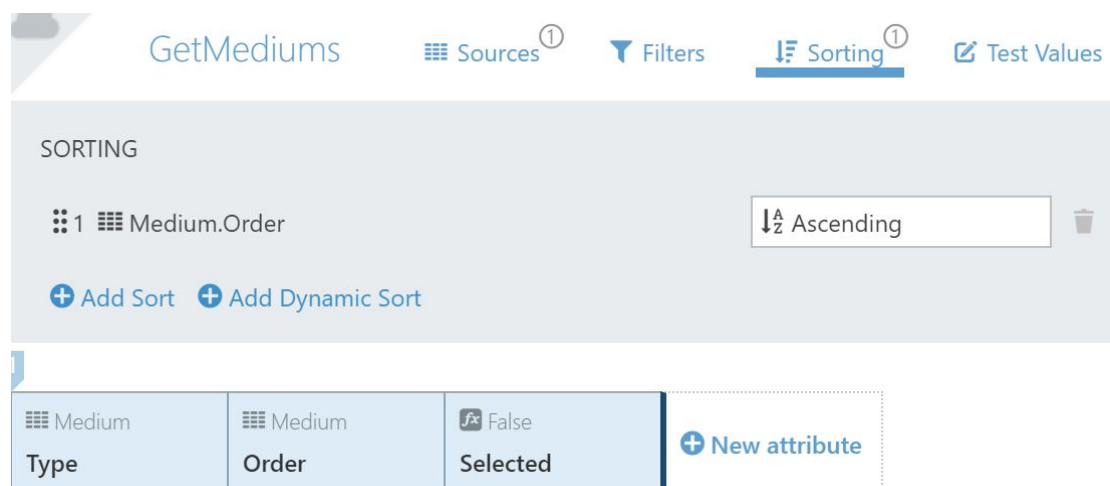
This exercise is based on an existing application that catalogues audio content. You can add new content and choose what is its type. You will extend it with the ability to choose which of the possible distribution types each content it is available on.

Extend the Audio Content Manager application

The Audio Content Manager application is used to catalogue audio content. The user can add new content and choose what is its type. You will improve the management of distribution types and extend it with the ability to choose which of the possible distribution types each content is available on.

Part I - Selecting multiple records in a table

1. Clone the **AudioContentManager** module, rename the clone to **AudioContentManager_<your initials>** and publish it
2. Allow the user to select multiple distribution medium types on the **MediumList** screen and delete them.
 - a. Modify the **GetMediums** Screen Aggregate that is the source of the *Table* being displayed, by adding a new boolean calculated attribute named **Selected**



NOTE: This attribute will be used to indicate if a record was selected or not by the user. That information is only used for interaction on the screen, so the calculated attribute's *Value* doesn't need to rely on any other attribute from the **Medium** entity and can simply be set to **False**.

- b. Add an extra column to the left of the *Table* and place a *Checkbox* on it. The *Checkbox's Variable* should be set to the calculated attribute previously added
- c. Add a link before the *Table* that deletes all the records from the Table that are selected

Medium List

Delete Selected

<input checked="" type="checkbox"/>	Streami
<input checked="" type="checkbox"/>	Streami
<input checked="" type="checkbox"/>	Streami

No items to show...

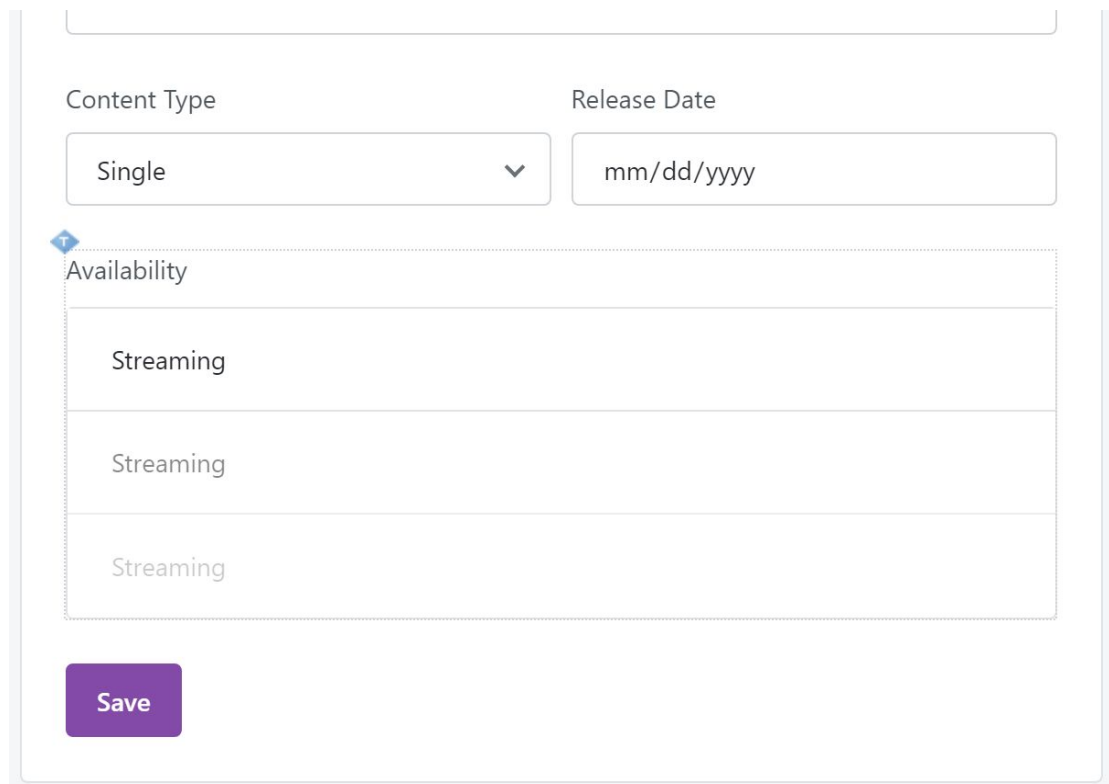
SUGGESTION: You can cycle through the aggregate's List and delete only the records that have been **Selected**

END OF PART I

Part II - Use Checkboxes to establish relationships

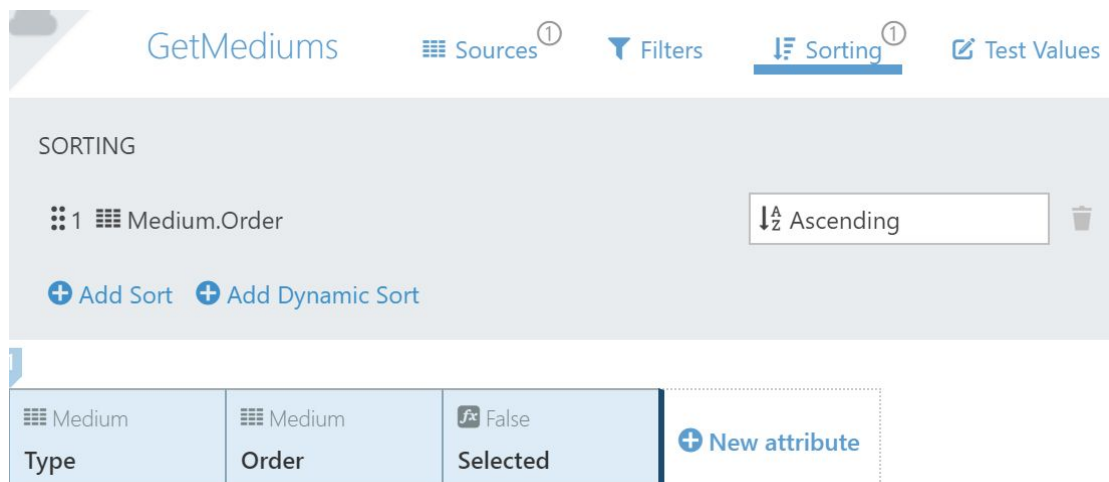
The second part of this exercise we will allow the user to choose which distribution mediums each content is available on. To that extent we will update the content details to show the list of all available mediums and which ones are currently selected for the particular record.

1. Show the list of available mediums on the content details and allow the user to Select them
 - a. On the **ContentDetail** screen, display a table with all the Mediums available, sorted by their Order attribute



The screenshot shows a form titled "ContentDetail". It has two input fields at the top: "Content Type" with a dropdown menu showing "Single" and a "Release Date" field with a placeholder "mm/dd/yyyy". Below these is a table titled "Availability" with three rows, each containing the text "Streaming". At the bottom left of the form is a purple "Save" button.

- b. Add an extra boolean calculated attribute to the source *Aggregate* of the available mediums, named **Selected** and with a *Value* of **False**



The screenshot shows the "GetMediums" screen. At the top, there are tabs: "Sources", "Filters", "Sorting", and "Test Values". The "Sorting" tab is selected. Below the tabs, there is a "SORTING" section with a list of sorts. The first sort is "1 Medium.Order" with a dropdown menu set to "Ascending". Below the sorting section are two buttons: "Add Sort" and "Add Dynamic Sort". At the bottom, there is a table with three columns: "Medium Type", "Medium Order", and "Medium Selected". The "Medium Selected" column has a calculated attribute "fx False". To the right of the table is a button "New attribute".

Medium Type	Medium Order	Medium Selected
		fx False

- c. Add an extra column to the left of the *Table*, and place a *Checkbox* bound to the calculated attribute

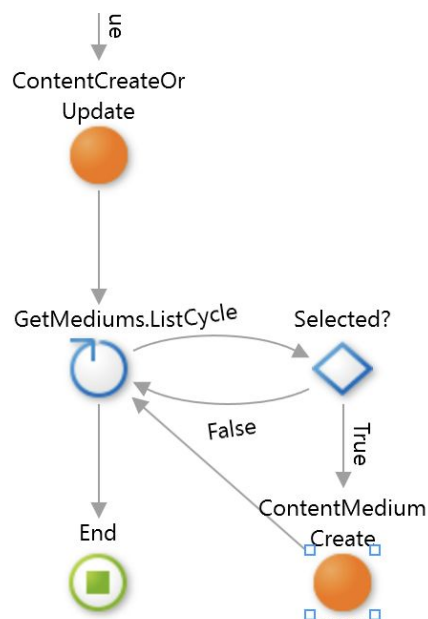
The screenshot shows a table with three rows, each labeled 'Streaming'. A checkbox is added to the left of the table. The properties panel for the checkbox is open, showing the following settings:

Checkbox1	
Name	Checkbox1
Variable	Items.List.Current.Selected
Enabled	True
Style Classes	"checkbox"
Attributes	
Property	= Value
Events	
On Change	
Event	
Handler	

A 'Save' button is visible below the table.

2. Change the **SaveOnClick** screen action to update the selection
 - a. For each Medium selected, create a new **ContentMedium** record with the corresponding content and medium details

SUGGESTION: You can cycle through the **GetMediums** aggregate's List and only establish the relationship for the records have been **Selected**

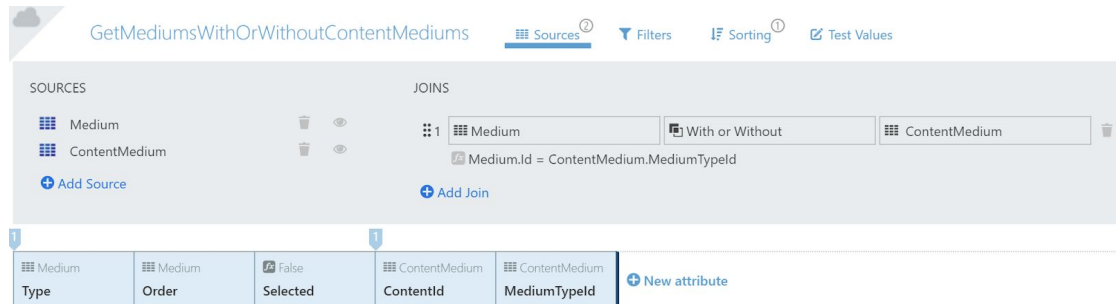


The screenshot shows the configuration for the **ContentMediumCreate** screen action. The action is a 'Run Server Action'.

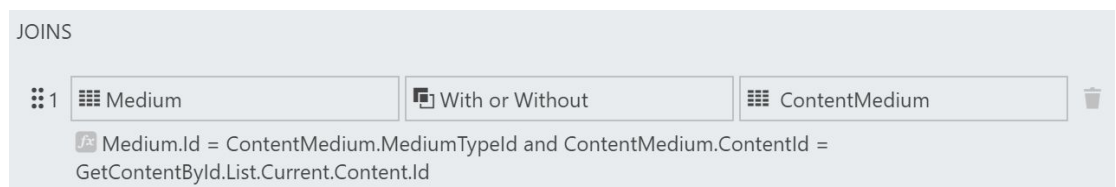
ContentMediumCreate	
Name	ContentMediumCreate
Server Requ...	(Module Default Timeout)
Action	ContentMedium\ContentMedi
Source	
Id	NullIdentifier()
ContentId	ContentCreateOrUpdate.Id
Medium...	GetMediums.List.Current.Medi
(New Argu...	

3. Display existing Medium selection when showing the details of a Content and update the SaveOnClick to correctly handle pre-existing selections

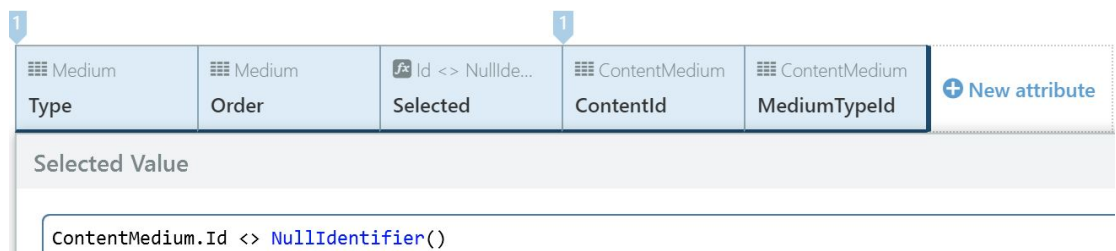
- a. Add the **ContentMedium** entity to the **GetMediums** aggregate



- b. Change the *Join* condition to restrict the join to records relative to the Content record being displayed



- c. Change the Selected expression to reflect the pre-existing relationship



- d. Update the **SaveOnClick** screen action to only create records for newly selected distribution mediums, and to delete records for previously selected distribution mediums that have been deselected



END OF PART II

Part III - Displaying multiple records per Table row

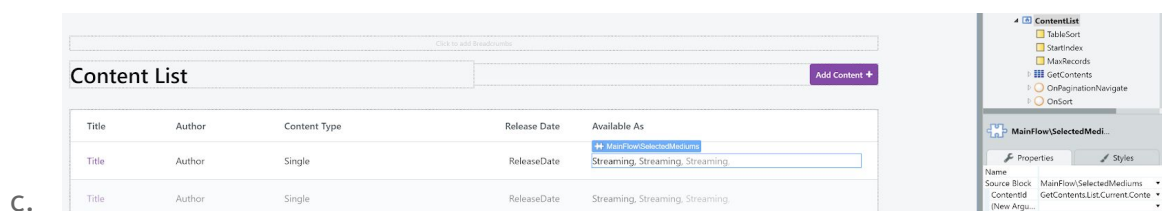
In this third, and final, part of the exercise we will change the Content listing page to show which distribution mediums each record is available on. In order to do this, for each content record displayed, we will need to obtain the list of mediums that have been configured for it. This list is different for each record, and in order to implement this solution we will need to use a Block to obtain the medium list and display it, based on the specific content record we pass as input.

1. Create a new *block* that receives a *Content Identifier* and displays the distribution mediums currently assigned to that content
 - a. In the **MainFlow UI Flow** create the **SelectedMediums** block, add an input parameter of type *Content Identifier*
 - b. Add an *Aggregate* to the *block* to obtain all **ContentMedium** (and associated Medium) records for the specific content
 - c. Add a *List* to the block, bound to the *Aggregate*'s output
 - d. Add a container inside the List, and an Expression inside the container, to display the Medium name for that record

SUGGESTION: Place an *If* next to the *Expression* to add a coma and a space next to the medium name if there will be more elements displayed after



2. Add the distribution medium list to the Content listing screen
 - a. On the **ContentList** screen, add a column to the right of the *Table* with a meaningful header
 - b. Drag an instance of the **SelectedMediums** block to the the newly created column, and configure it to use the Content Id of the current record being displayed



END OF PART III