



DEVELOPING OUTSYSTEMS MOBILE APPS

# Forms



---

# Introduction

Over the course of this set of exercise labs, you will create a web application. The application will focus on creating and managing To Dos. The To Dos will be persisted in a database so they can be accessed from and shared across multiple devices. To Dos will have attributes such as category, priority (low, medium or high), due date and they can be marked as important (starred) by the user.

Users of the To Do application will be able to access all of this information. This back-office application will allow administrators to manage all existing To Dos.

You constantly will be expanding your application, publishing it to the server and testing your application to the server while learning and applying new OutSystems concepts.

At the end of this set of exercise labs, you will have a small, but well-formed web application, spanning multiple screens and concepts that you can easily access from your browser.

In this specific exercise lab, you will:

- Display each Category details
- Allow users to edit and create new Categories

---

# Table of Contents

|   |    |
|---|----|
| Introduction.....                                   | 2  |
| Table of Contents.....                              | 3  |
| Part 1: Retrieve and display category details ..... | 4  |
| Part 2: Add 'Edit Category' feature .....           | 14 |
| Part 3: Add 'New Category' feature .....            | 21 |
| End of Lab .....                                    | 25 |
| List of Figures.....                                | 26 |

## Part 1: Retrieve and display category details

In this part of the exercise, you will create a detail Screen. The detail Screen will display the details of a particular record, in this case a Category.

1. Create the **CategoryDetail** with a Category Identifier parameter and define the Screen **Preparation**.

- a) Switch to the **Interface** tab, by pressing Ctrl+2.
- b) In the elements area, right click the **MainFlow** element and then select 'Add Web Screen'.

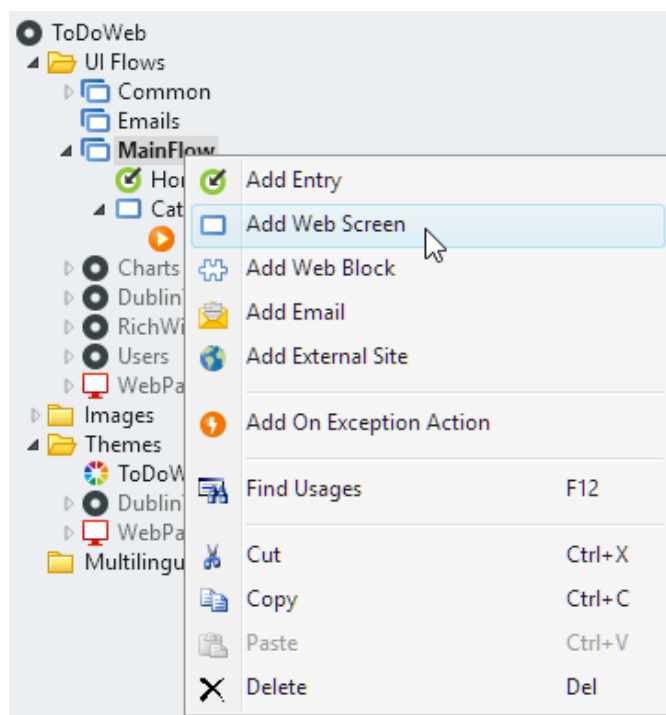


Figure 1. Add Web Screen

- c) Set the name of the new screen to 'CategoryDetail'.
- d) Right click the **CategoryDetail** Screen and select 'Add Input Parameter'.

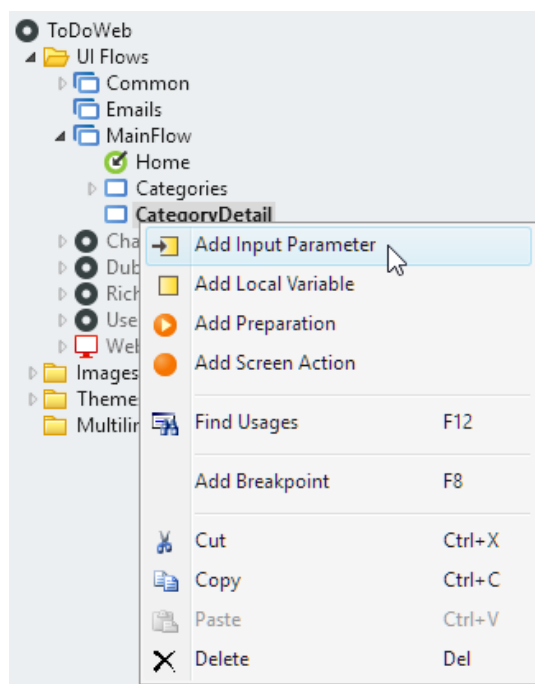


Figure 2. Add an Input Parameter to the CategoryDetail Screen

- e) Enter 'CategoryId' for the name of the Input Parameter, and make sure its **Data Type** is set to **Category Identifier**.

| CategoryId Input Parameter |                       |
|----------------------------|-----------------------|
| Name                       | CategoryId            |
| Description                | ...                   |
| Data Type                  | Category Identifier ▼ |
| Is Mandatory               | Yes ▼                 |
| Default Value              |                       |

Figure 3. CategoryId Input Parameter properties

- f) Right click the **CategoryDetail** Screen and select 'Add Preparation'.
- g) Drag and drop the **CategoryId** Input Parameter onto the **Preparation** flow.

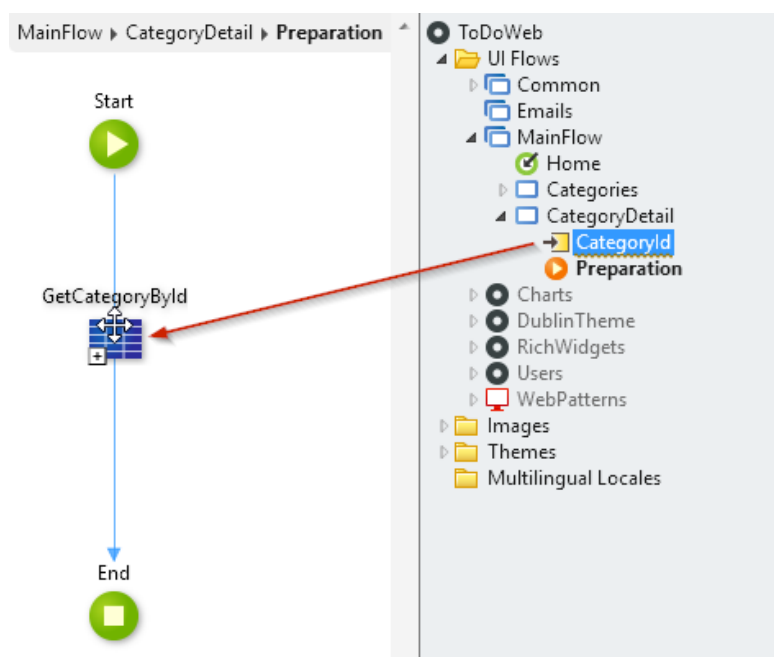


Figure 4. Drag input parameter into Preparation flow

- h) Notice that the created Aggregate has name **GetCategoryById**, and fetches data from the **Category** Entity, filtering it by the **CategoryId** Input Parameter.

| GetCategoryById<br>Aggregate |                 |     |
|------------------------------|-----------------|-----|
| Name                         | GetCategoryById | ... |
| Description                  |                 | ... |
| Timeout in Seconds           |                 | ... |
| Cache in Minutes             |                 | ... |
| Max. Records                 |                 | ▼   |
| Sources                      |                 |     |
| Category                     |                 | ... |
| Filters                      |                 |     |
| Category.Id = CategoryId     |                 | ... |
| Test Values                  |                 |     |
| CategoryId                   |                 | ... |

Figure 5. Aggregate created by dragging CategoryId

## 2. Define the Title of the **CategoryDetail** Screen, to display the Category name.

- Open the **CategoryDetail** Screen.
- Drag and drop an **Expression** to the **Title** placeholder.

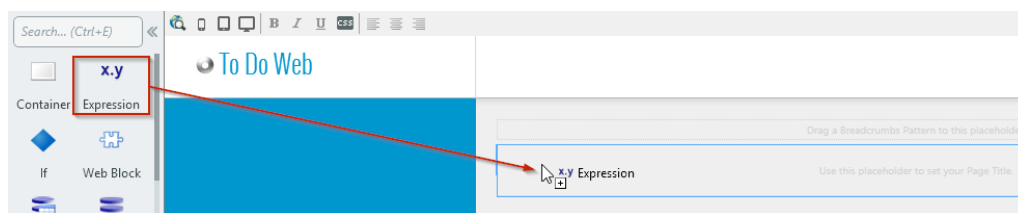


Figure 6. Drag an Expression in the Title placeholder

c) Set its **Value** property to `GetCategoryId.List.Current.Category.Name`

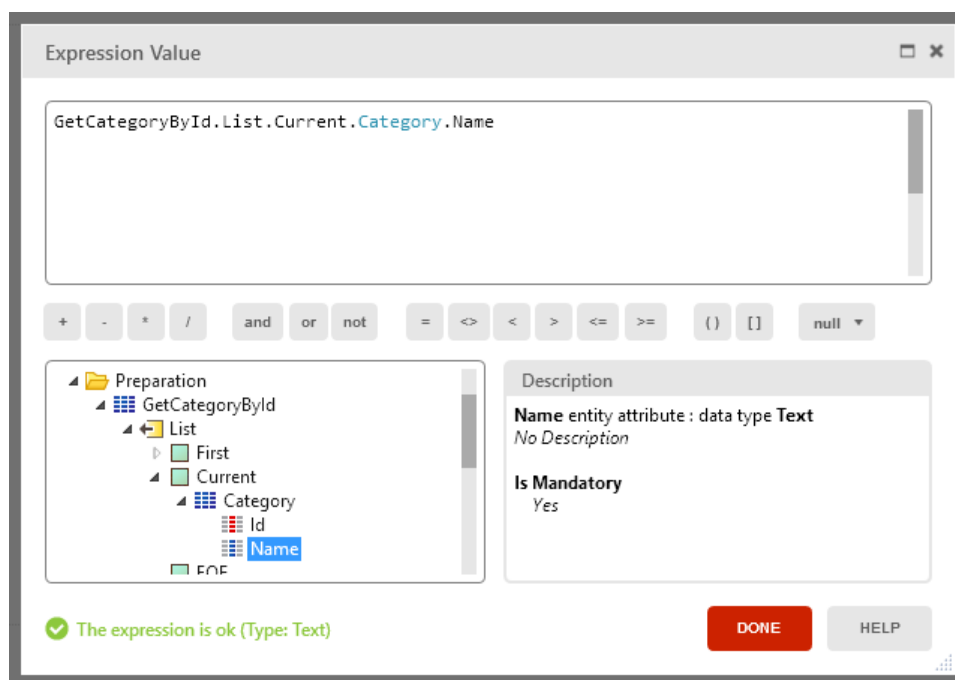


Figure 7. Select the current Category Name

d) Click **Done** to close the Expression Editor.

3. Create a **Form** in the **MainContent** placeholder of the **CategoryDetail**. Retrieve and display the values of the Category attributes.

a) Drag and drop a **Form** Widget to the **MainContent** placeholder.



Figure 8. Add a Form to the Main Content placeholder

- b) Click the **Source Record** drop down and select 'GetCategoryById..Current'

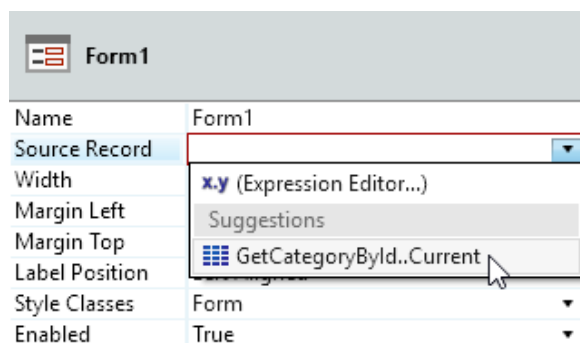


Figure 9. Set Source Record for the Form

- c) Switch to the **Data** tab, then drag and drop the **Category** Entity to **Form1**.

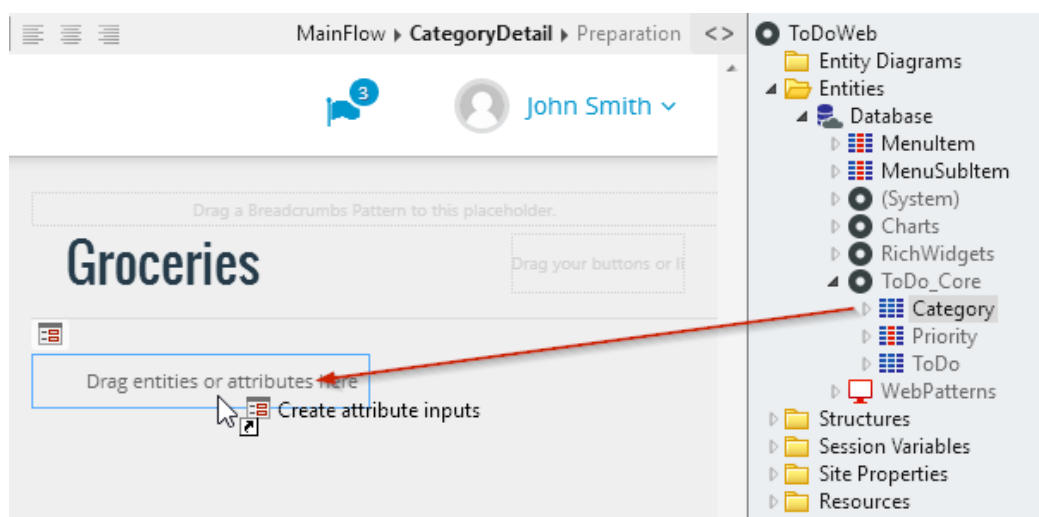


Figure 10. Drag the Category Entity to a Form to create inputs

**NOTE:** Dragging an Entity to a **Form**, will create an Input for each attribute, except the **Id**. It is also possible to drag attributes one by one, to create Inputs individually.

- d) Notice that a **Container** was created. The container contains a **Label** and an **Input** (a text input field), and the form's name has changed from Form1 to **CategoryForm**.



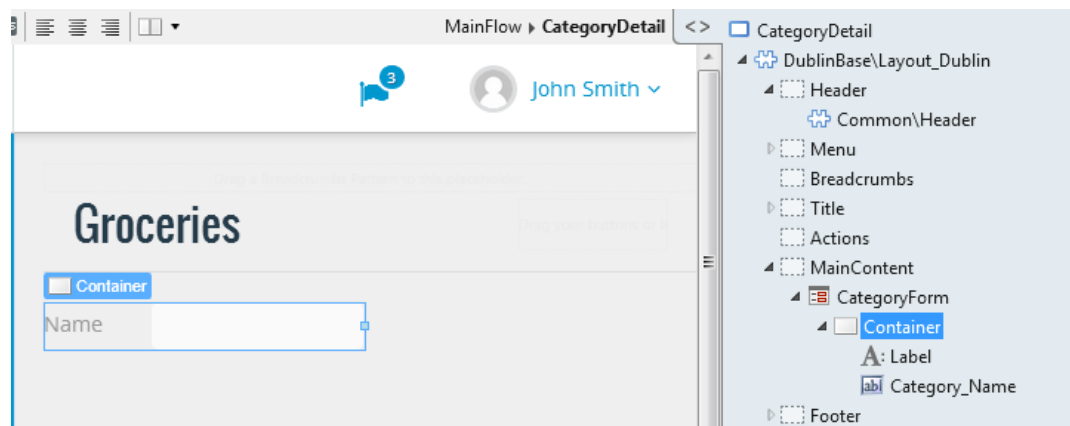


Figure 11. Category\_Name Input field

**NOTE:** There is a warning in **TrueChange** stating that the inputs on the Screen are not being submitted to the server and will be ignored. For now, this Screen will be used only for displaying, not modifying, category data so this warning can be safely ignored.

#### 4. Create a 'Back' button that returns the user to the **Categories** Screen.

- a) Drag and drop a Button below the input container, but inside the **CategoryForm**.

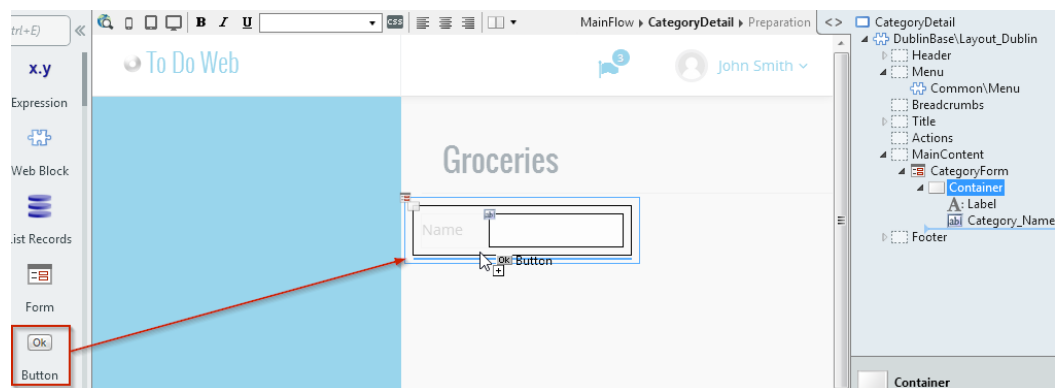


Figure 12. Drag and drop a Button on the bottom of the Form

- b) Right click the Button and select **Enclose in Container**, to create a Container around the Button.

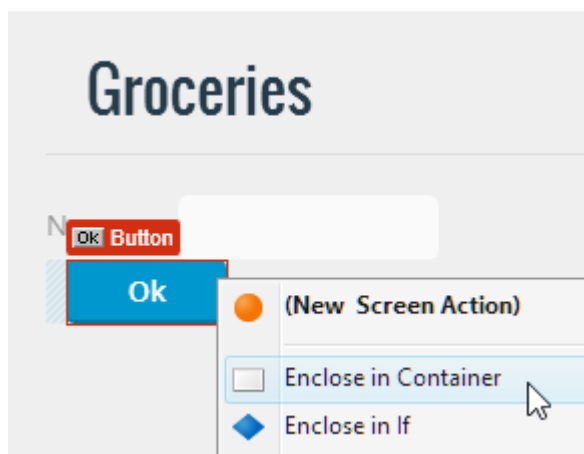


Figure 13. Enclose in Container

- c) Select the Button and set its **Label** to “Back” and **Method** to ‘Navigate’.

---

**NOTE:** The Method Navigate indicates that when clicking on the Button, the user will navigate to a different page, similar to a HTTP GET request. No data is submitted to the server, only a new request for a different page is made.

---

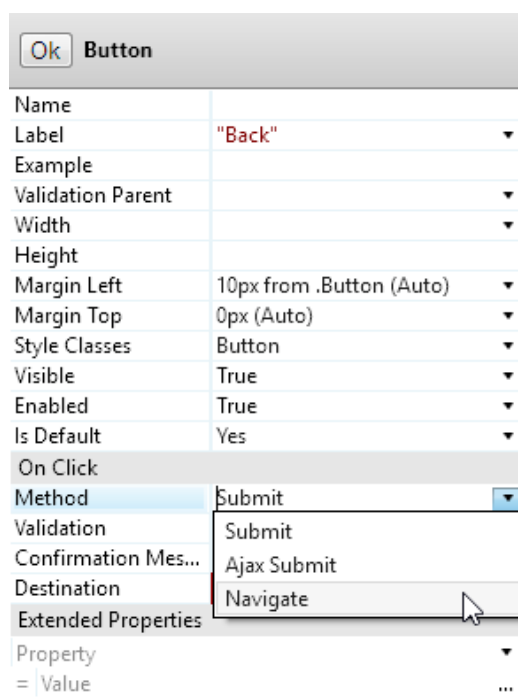


Figure 14. Set Label and Method for Back Button

- d) Double click the **Destination** property to open the **Select On Click Destination** dialog. Select the **Categories** Screen and click **Ok**.

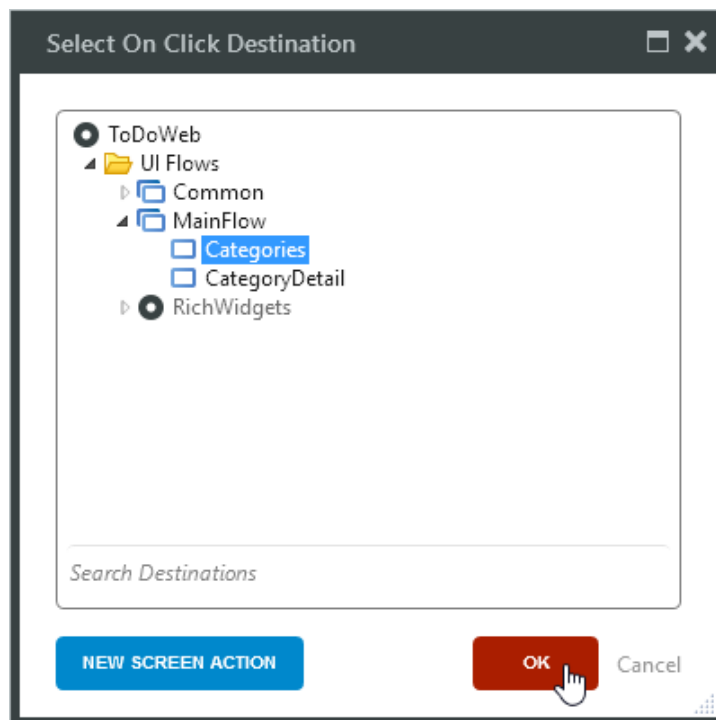


Figure 15. Set On Click Destination for Back Button

5. Update the **Categories** Screen to be able to link to the details of the Category. Create a Link from the Category name to the **CategoryDetail** Screen.

a) Open the **Categories** Screen.

b) In the **CategoryTable**, right click the **Expression** in the Name column and select 'Link to MainFlow\CategoryDetail Web Screen'.

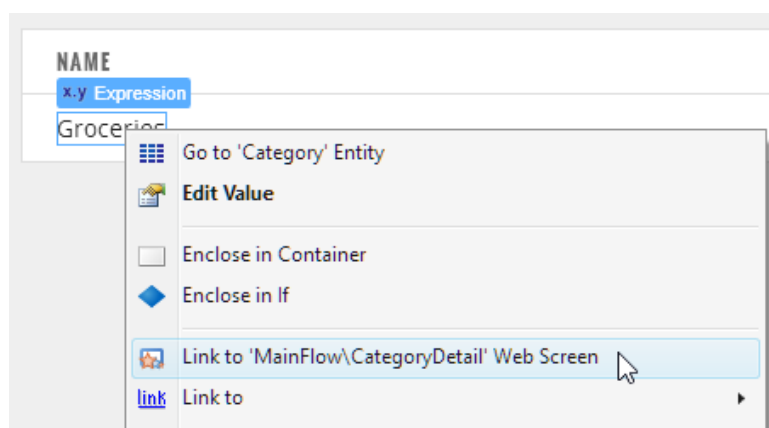


Figure 16. Link to CategoryDetail

c) Notice that the create link option had a star in its icon. This indicates that the selection is an accelerator. The Link was created, given a **Title**, the **Method** set to 'Navigate', and the **Destination** set to the **CategoryDetail** Screen, with the respective **Category Identifier** Input.

| link Link            |  |
|----------------------|--|
| Name                 |  |
| Title                | "Edit '" + CategoryTable.List.Current.Categori   |
| Validation Parent    |  |
| Style Classes        |  |
| Visible              | CategoryTable.List.Current.Category.Id <         |
| Enabled              | True   |
| Is Default           | No   |
| On Click             |  |
| Method               | Navigate   |
| Validation           | (none)   |
| Confirmation Message |  |
| Destination          | <input type="checkbox"/> MainFlow\CategoryDetail |
| CategoryId           | CategoryTable.List.Current.Category.Id           |
| (New Argument)       |  |

Figure 17. The new Link's automatically populated properties

6. Publish the **Categories** and **CategoryDetail** Screens changes to the server.

- a) Click the **1-Click Publish** to publish the module.
- b) Click the **Open in Browser** button. The **Categories** Screen should be displayed, since it is the **Home** Entry Point (which is the default entry Screen for the **ToDoWeb** module).

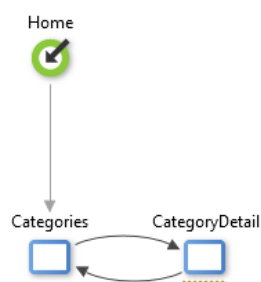


Figure 18. Home Entry Point, in the MainFlow

- c) Select a **Name** in the Categories Table.

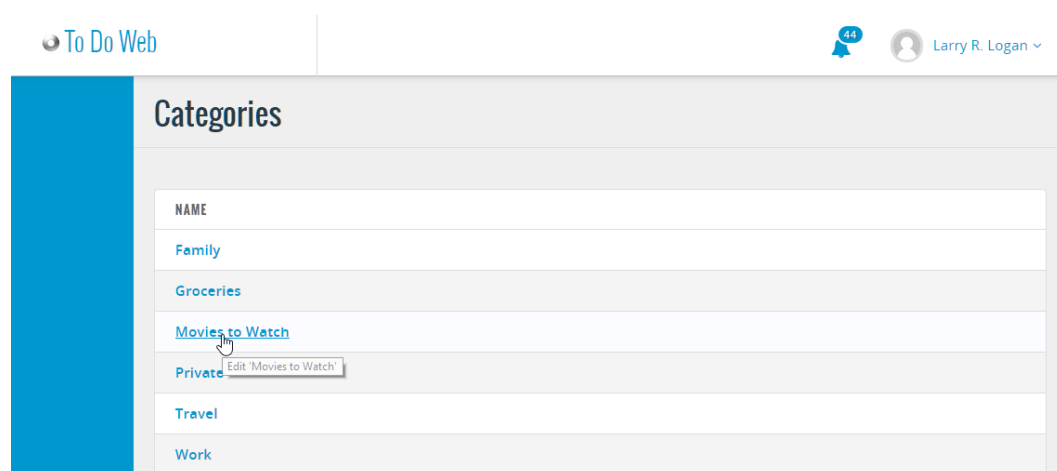


Figure 19. Categories Screen in the browser

d) Click the **Back** button to return to the **Categories** Screen.

Figure 20. CategoryDetail Screen in the browser

## Part 2: Add 'Edit Category' feature

In this part of the exercise, you will expand the **CategoryDetail** Screen so that any changes made to its **Form**, can be submitted back to the server. Submission to the server will execute a Screen Action, that will save the entered information to the database.

1. Add a 'Save' Button to **CategoryDetail**. Configure it to execute a Screen Action on clicking, to save the **Form** data to the database.

a) Open the **CategoryDetail** Screen.

b) Drag and drop a new **Button** to the left of the existing **Back** button.

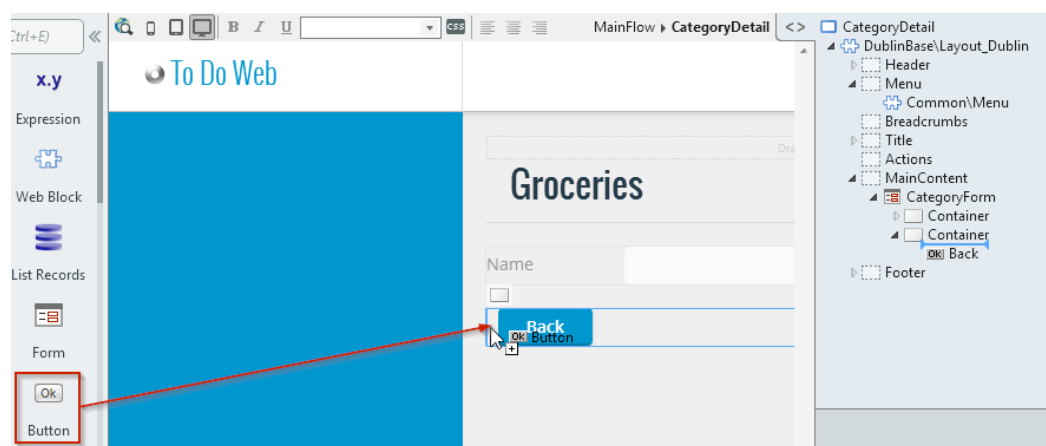


Figure 21. Add a new Button to the CategoryDetail Screen

c) In the properties editor, set the button's **Label** to "Save".

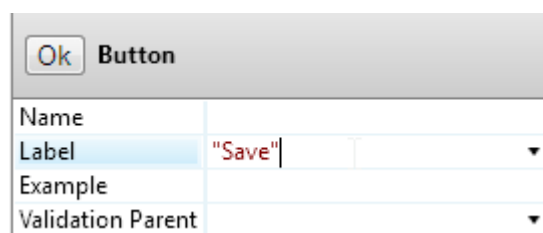


Figure 22. Set Save Button Label

d) Notice that both Buttons do not have the same color. Check their **Is Default** property.

| Ok Button         |                          | Ok Button         |              |
|-------------------|--------------------------|-------------------|--------------|
| Name              |                          | Name              |              |
| Label             | "Save"                   | Label             | "Back"       |
| Example           |                          | Example           |              |
| Validation Parent |                          | Validation Parent |              |
| Width             |                          | Width             |              |
| Height            |                          | Height            |              |
| Margin Left       | 10px from .Button (Auto) | Margin Left       | 1.96% (Auto) |
| Margin Top        | 0px (Auto)               | Margin Top        | 0px (Auto)   |
| Style Classes     | Button                   | Style Classes     | Button       |
| Visible           | True                     | Visible           | True         |
| Enabled           | True                     | Enabled           | True         |
| Is Default        | No                       | Is Default        | Yes          |

Figure 23. Save and Back button properties

- e) Set **Save Button Is Default** property to **Yes**.
- f) Notice that the **Is Default** property of the **Back Button** is now set to **No**.

---

**NOTE:** Only one Button can be the default. By setting the **Save** Button as default, the **Back Button Is Default** property automatically changed to **No**.

---

- g) Your Screen should look like this

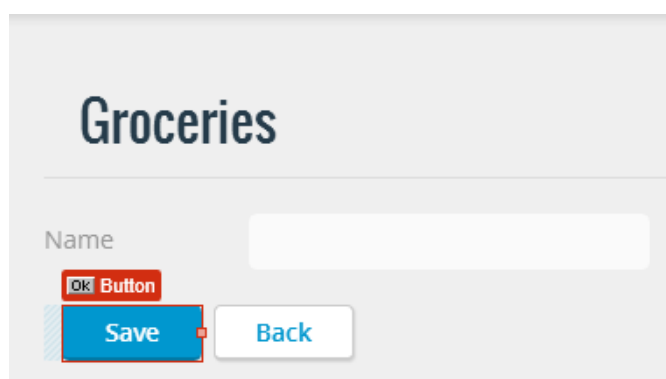


Figure 24. CategoryDetail Screen, with Save as default button

- h) Set the **Destination** of the **Save** Button to '(New Screen Action)'.

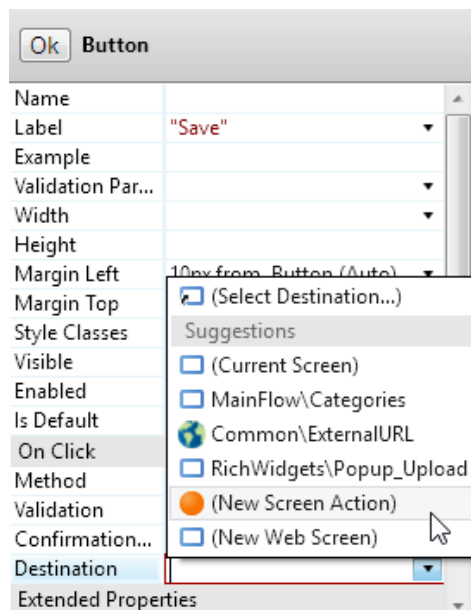


Figure 25. Set Save button destination

- i) Inside the created **Save** Screen Action flow, drag a **Run Server Action** and drop it in connector between Start and End.

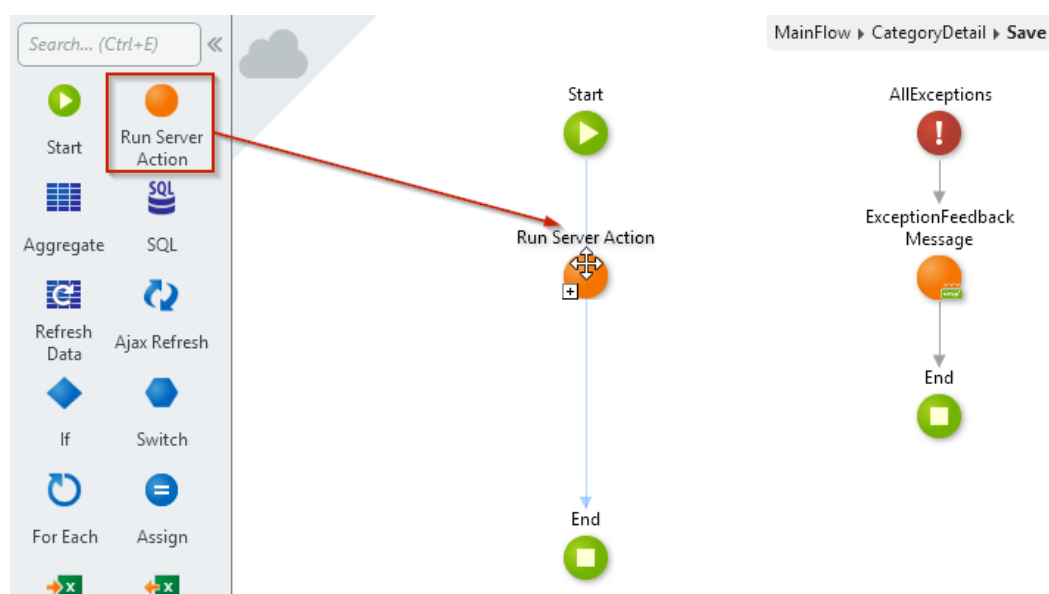


Figure 26. Insert Execute Server Action between Start and End

- j) In the **Select Action** dialog, select the **CreateOrUpdateCategory** Entity Action.



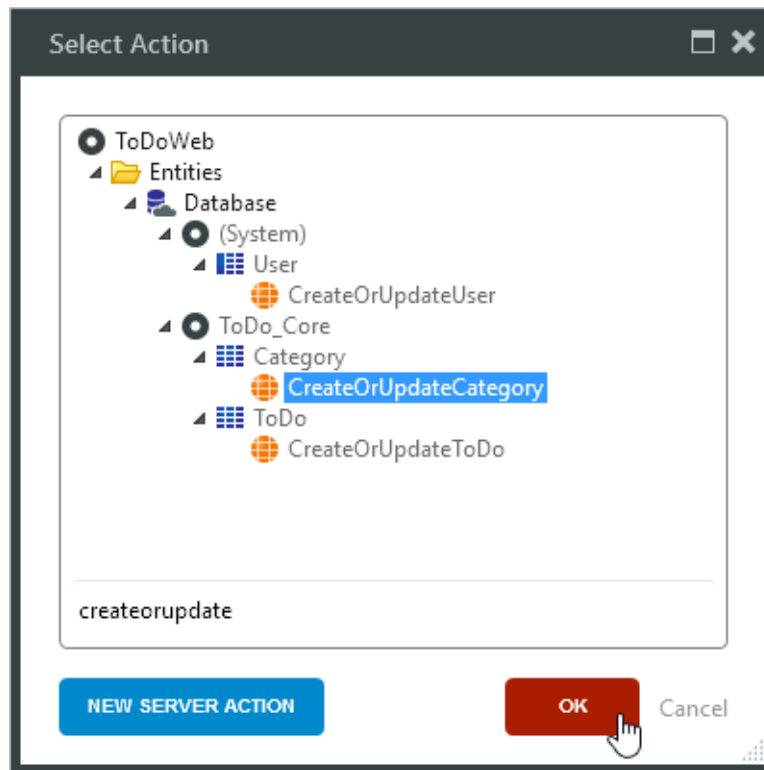


Figure 27. Select Action dialog

- k) Set the **Source** argument for the **CreateOrUpdateCategory** Action to 'CategoryForm.Record'.

---

**NOTE:** To recap the flow of data in this Screen:

- 1) In the **Preparation**, the category to be displayed is fetched from the database using an Aggregate (and the Screen parameter **CategoryId**);
  - 2) In the Screen, the **Form** Widget sources its value from the output of the Aggregate and presents it to the user;
  - 3) The user changes the values in the **Form** inputs and presses the **Save** Button, submitting changes to the server;
  - 4) The **Save** Screen Action handles the submit and updates (or creates) the record in the database. It uses the value from the Form, which contains the changed category with the user inputs.
- 

- l) Drag and drop a **Destination** over the existing **End** statement in the main editor.

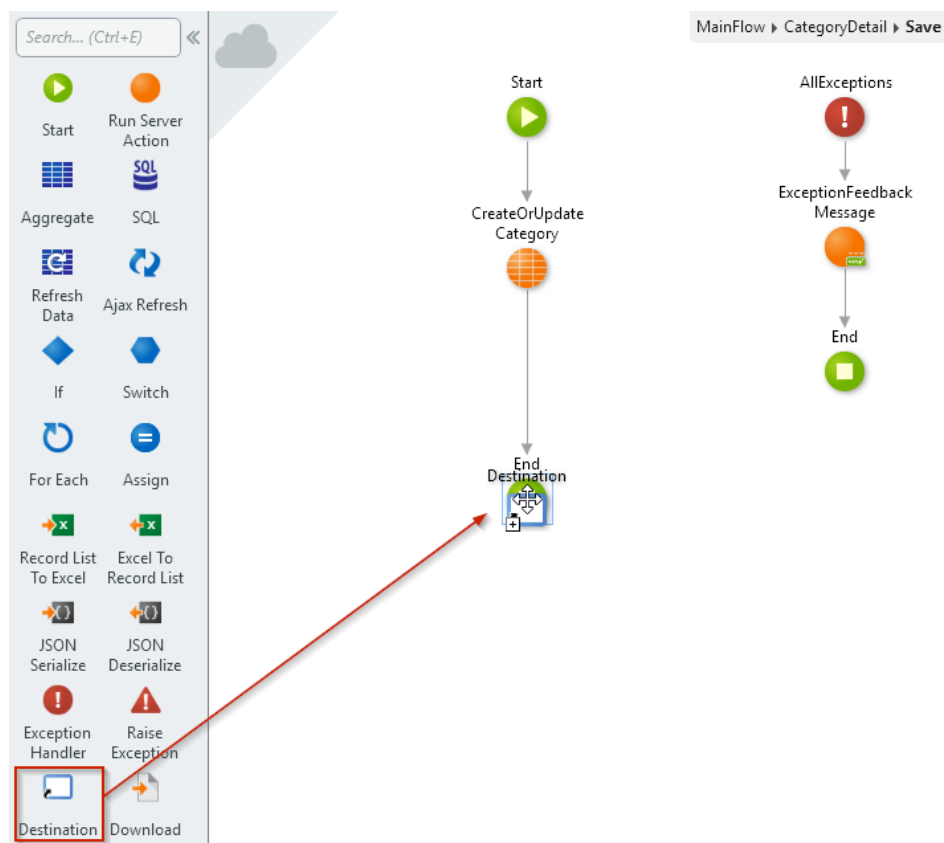


Figure 28. Drag a Destination to replace End

- m) In the **Select Destination** dialog, type 'categories' then select the **Categories** screen.

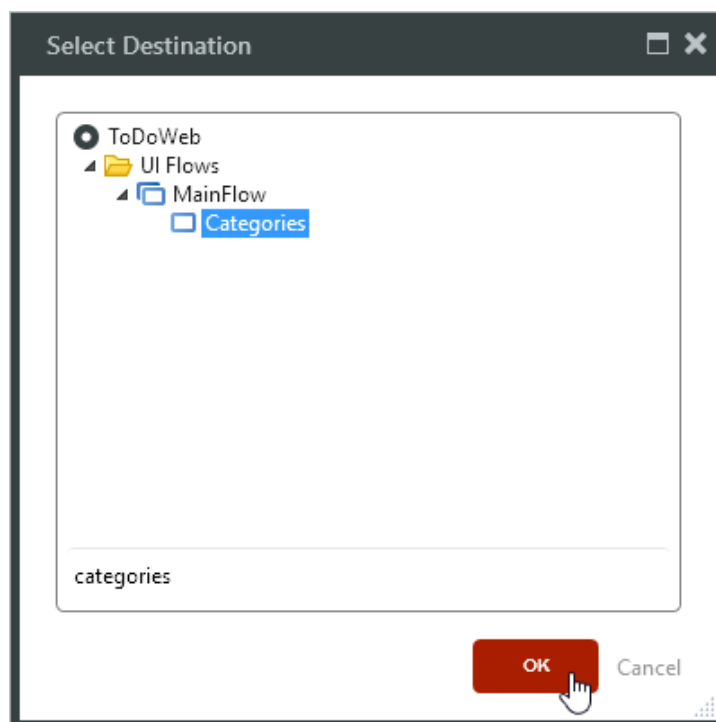


Figure 29. Select Destination dialog

**NOTE:** Screen Actions by default terminate in an **End** statement. This causes execution to cycle back to the current Screen's **Preparation** and the Screen is re-rendered automatically.

By replacing the **End** statement with a **Destination**, execution can be redirected to a different Screen, after the Screen Action completes.

n) Your screen action should look like this

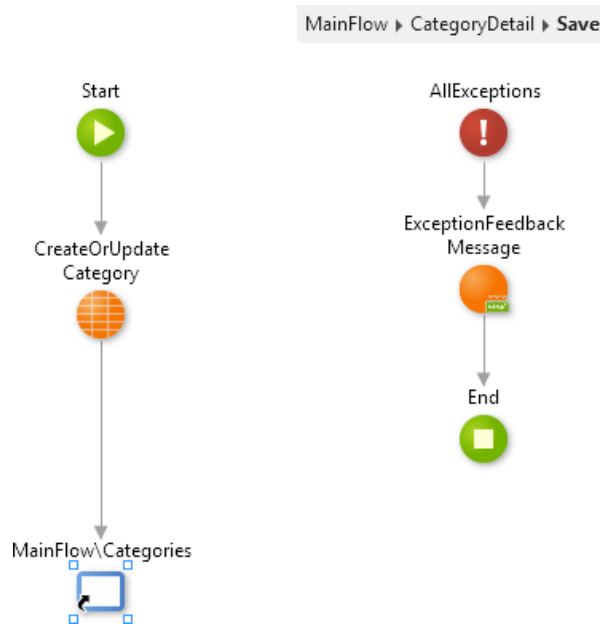
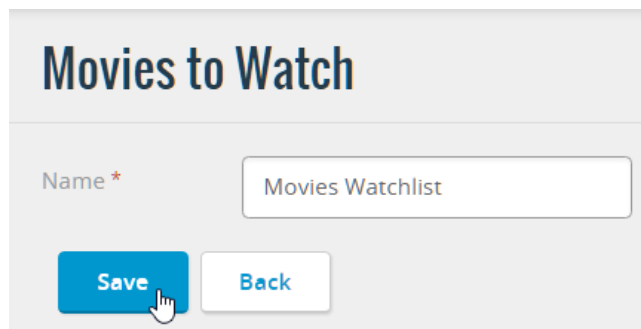


Figure 30. The 'Save' Screen Action

2. Publish the module and test the edit Category feature.

- a) Click the **1-Click Publish** button to publish the module.
- b) Click the **Open in Browser** button to view the **Categories** Screen. If required, login again with the same credentials as before.
- c) Click the 'Movies to Watch' Category to navigate to the respective **CategoryDetail** Screen.
- d) Change the **Name** to 'Movies Watchlist', then click **Save**.



The screenshot shows a form titled "Movies to Watch". It has a label "Name \*" and a text input field containing "Movies Watchlist". Below the input field are two buttons: a blue "Save" button and a white "Back" button. A mouse cursor is clicking on the "Save" button.

Figure 31. Category name update

- e) You should now see the **Categories** list screen, and the updated category name.



The screenshot shows a list screen titled "Categories". It contains a table with the following categories:

| NAME             |
|------------------|
| Family           |
| Groceries        |
| Movies Watchlist |
| Private          |
| Travel           |
| Work             |

A mouse cursor is hovering over the "Movies Watchlist" category.

Figure 32. Updated Category name

## Part 3: Add 'New Category' feature

In this part of the exercise, you will add a new Link in the Categories screen that will allow the creation of brand new category using the **CategoryDetail** Screen.

1. Add a 'Create a new Category' link to **Categories**. Configure it to navigate to **CategoryDetail** Screen in "create new" mode.

a) Open the **Categories** Screen.

b) Type 'Create a new Category' inside the **Actions** placeholder of the Screen.

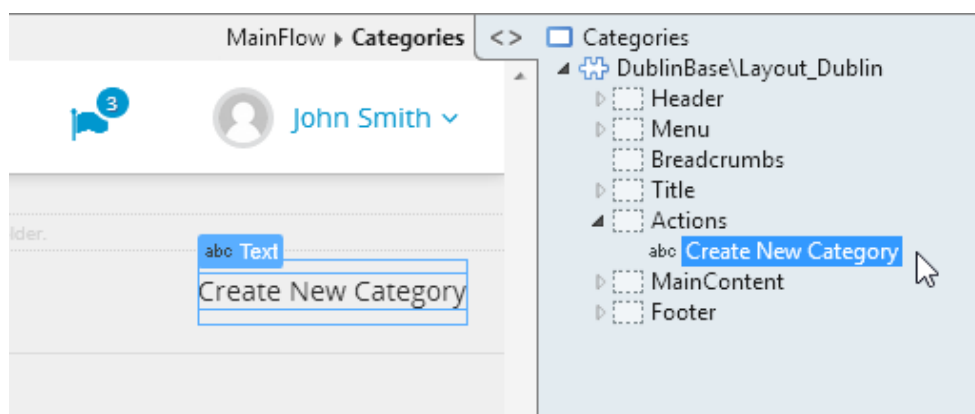


Figure 33. Categories Screen Actions placeholder

c) Right click the text element, and Link it to 'MainFlow\CategoryDetail'.

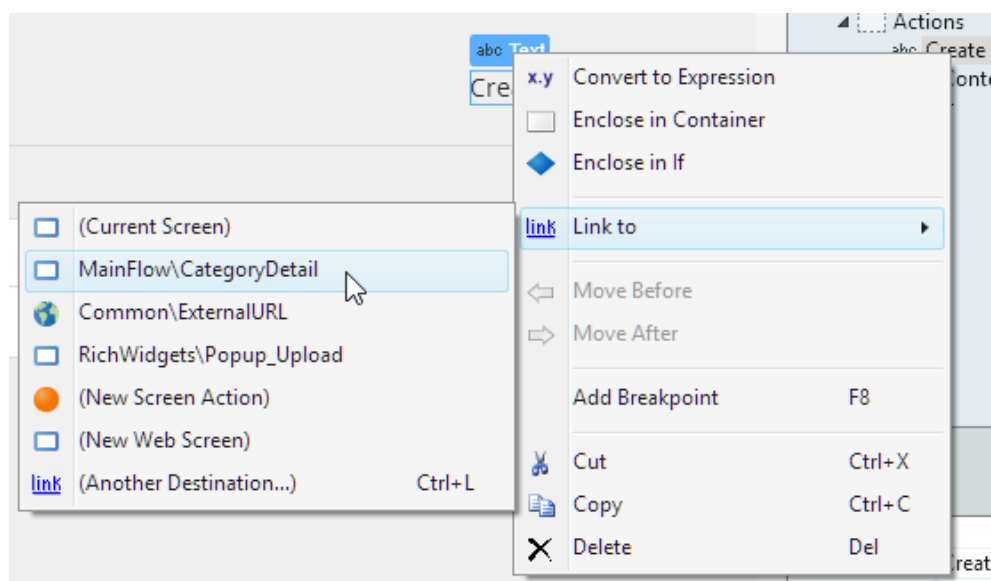


Figure 34. Link action text to CategoryDetail Screen

- d) In the properties editor of the recently created Link, set the **CategoryId** Parameter to `NullIdentifier()`

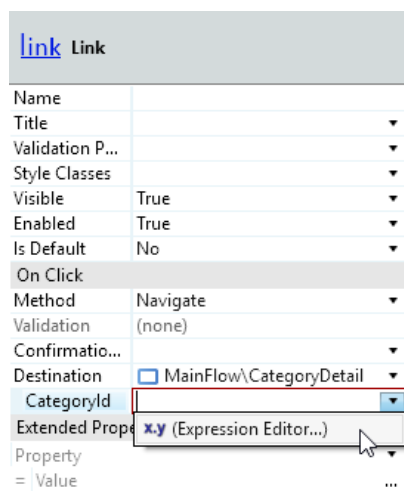


Figure 35. Link CategoryId Parameter

- e) Click **Done** to close the editor.

**NOTE:** Recall that the **CategoryId** Input Parameter that is passed into the **CategoryDetail** Screen, is used in that Screen's **Preparation** to retrieve the category to be displayed.

When the application navigates to **CategoryDetail** from the category list, it passes into that Parameter the Identifier of the line selected. This causes an existing record to be retrieved and displayed (for showing and editing).

When it navigates from the 'Create a new Category' link, by passing `NullIdentifier()` into that Parameter, no record will be retrieved (because identifier in the database are never null). This causes a blank record to be displayed (for creation).

- f) Set the link **Style Classes** property to 'ActionAdd'.

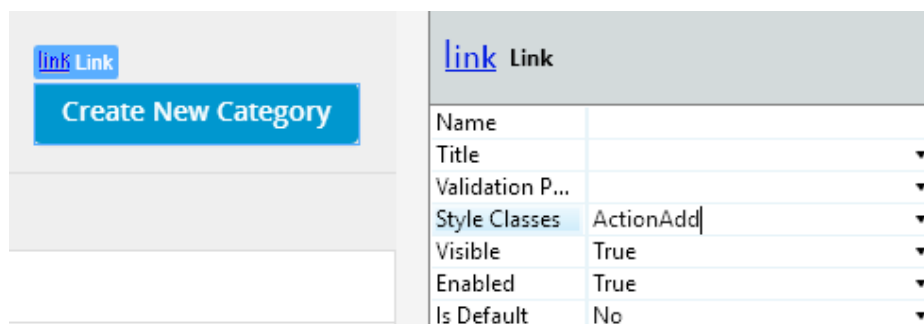


Figure 36. Link Style Classes property

2. Make the **CategoryDetail** Screen title display 'New Category', if adding a new category.

a) Open the **CategoryDetail** screen.

b) Right click the **Expression** in the Title placeholder of the Screen and select 'Enclose in If'.

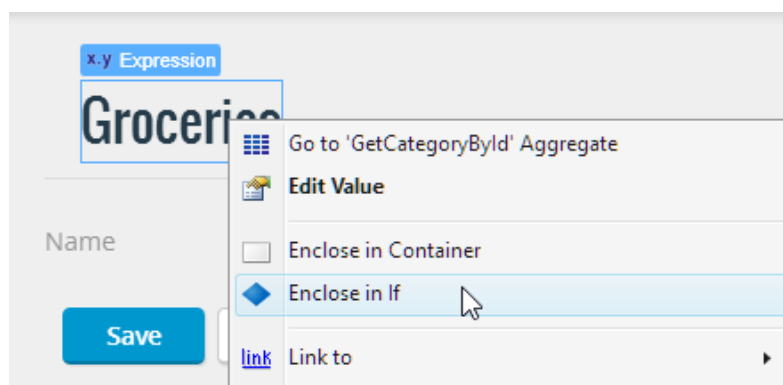


Figure 37. Enclose Expression in If

c) Set the **Condition** property for this new If to

```
CategoryId <> NullIdentifier()
```

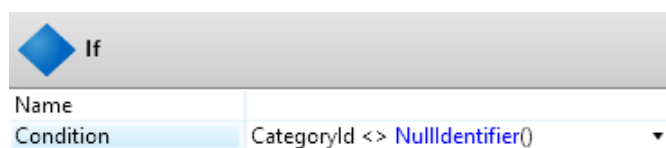


Figure 38. If Widget Condition property

d) In the false branch of the **If**, type in 'New Category'.

e) The contents of the **Title** placeholder should look like this in the **Widget Tree**.

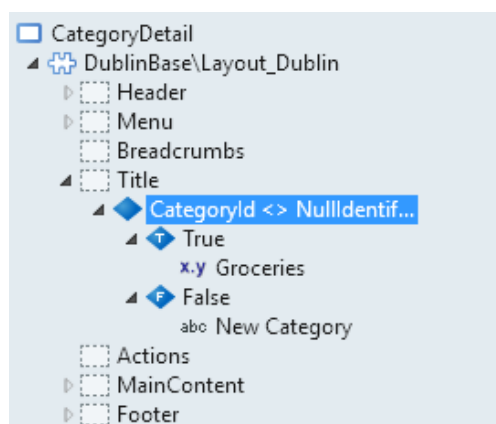


Figure 39. Title placeholder contents in CategoryDetail Screen

### 3. Publish the module and test the create new Category feature.

- a) Click the **1-Click Publish** button to publish the module.
- b) Click the **Open in Browser** button to view the **Categories** Screen. If required, login again with the same credentials as before.
- c) Click the 'Create a new Category' link to navigate to the **CategoryDetail** Screen.
- d) Enter 'To Do Back Office App' in the **Name** input, then click **Save**.

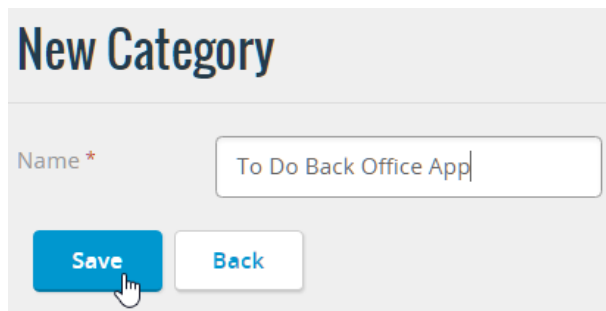
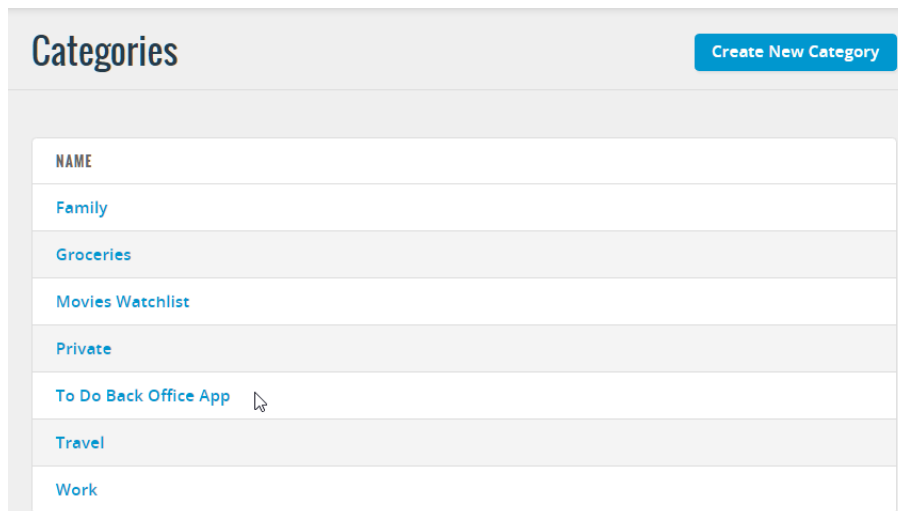


Figure 40. Create a New Category

- e) You should now see the new Category in the **Categories** Screen.



| NAME                  |
|-----------------------|
| Family                |
| Groceries             |
| Movies Watchlist      |
| Private               |
| To Do Back Office App |
| Travel                |
| Work                  |

Figure 41. New Category in Categories Screen



---

## End of Lab

In this exercise, you created a new Screen to display the details of a specific Category. Then implemented the logic to allow users to modify categories.

Finally, you have extended the **CategoryDetail** Screen to also allow to create new Categories.

## List of Figures

Here is the list of screenshots and pictures used in this exercise.

|  |    |
|--|----|
| Figure 1. Add Web Screen.....  | 4  |
| Figure 2. Add an Input Parameter to the CategoryDetail Screen.....   | 5  |
| Figure 3. CategoryId Input Parameter properties.....                 | 5  |
| Figure 4. Drag input parameter into Preparation flow .....           | 6  |
| Figure 5. Aggregate created by dragging CategoryId.....              | 6  |
| Figure 6. Drag an Expression in the Title placeholder.....           | 7  |
| Figure 7. Select the current Category Name .....                     | 7  |
| Figure 8. Add a Form to the Main Content placeholder .....           | 7  |
| Figure 9. Set Source Record for the Form .....                       | 8  |
| Figure 10. Drag the Category Entity to a Form to create inputs ..... | 8  |
| Figure 11. Category_Name Input field .....                           | 9  |
| Figure 12. Drag and drop a Button on the bottom of the Form.....     | 9  |
| Figure 13. Enclose in Container.....                                 | 10 |
| Figure 14. Set Label and Method for Back Button.....                 | 10 |
| Figure 15. Set On Click Destination for Back Button.....             | 11 |
| Figure 16. Link to CategoryDetail .....                              | 11 |
| Figure 17. The new Link's automatically populated properties .....   | 12 |
| Figure 18. Home Entry Point, in the MainFlow.....                    | 12 |
| Figure 19. Categories Screen in the browser.....                     | 12 |
| Figure 20. CategoryDetail Screen in the browser .....                | 13 |
| Figure 21. Add a new Button to the CategoryDetail Screen .....       | 14 |
| Figure 22. Set Save Button Label.....                                | 14 |
| Figure 23. Save and Back button properties.....                      | 15 |
| Figure 24. CategoryDetail Screen, with Save as default button .....  | 15 |
| Figure 25. Set Save button destination .....                         | 16 |
| Figure 26. Insert Execute Server Action between Start and End .....  | 16 |
| Figure 27. Select Action dialog.....                                 | 17 |
| Figure 28. Drag a Destination to replace End .....                   | 18 |
| Figure 29. Select Destination dialog.....                            | 18 |
| Figure 30. The 'Save' Screen Action.....                             | 19 |
| Figure 31. Category name update .....                                | 20 |
| Figure 32. Updated Category name .....                               | 20 |
| Figure 33. Categories Screen Actions placeholder .....               | 21 |
| Figure 34. Link action text to CategoryDetail Screen.....            | 21 |
| Figure 35. Link CategoryId Parameter .....                           | 22 |
| Figure 36. Link Style Classes property .....                         | 22 |
| Figure 37. Enclose Expression in If .....                            | 23 |
| Figure 38. If Widget Condition property .....                        | 23 |

|  |    |
|--|----|
| Figure 39. Title placeholder contents in CategoryDetail Screen ..... | 23 |
| Figure 40. Create a New Category.....                                | 24 |
| Figure 41. New Category in Categories Screen.....                    | 24 |