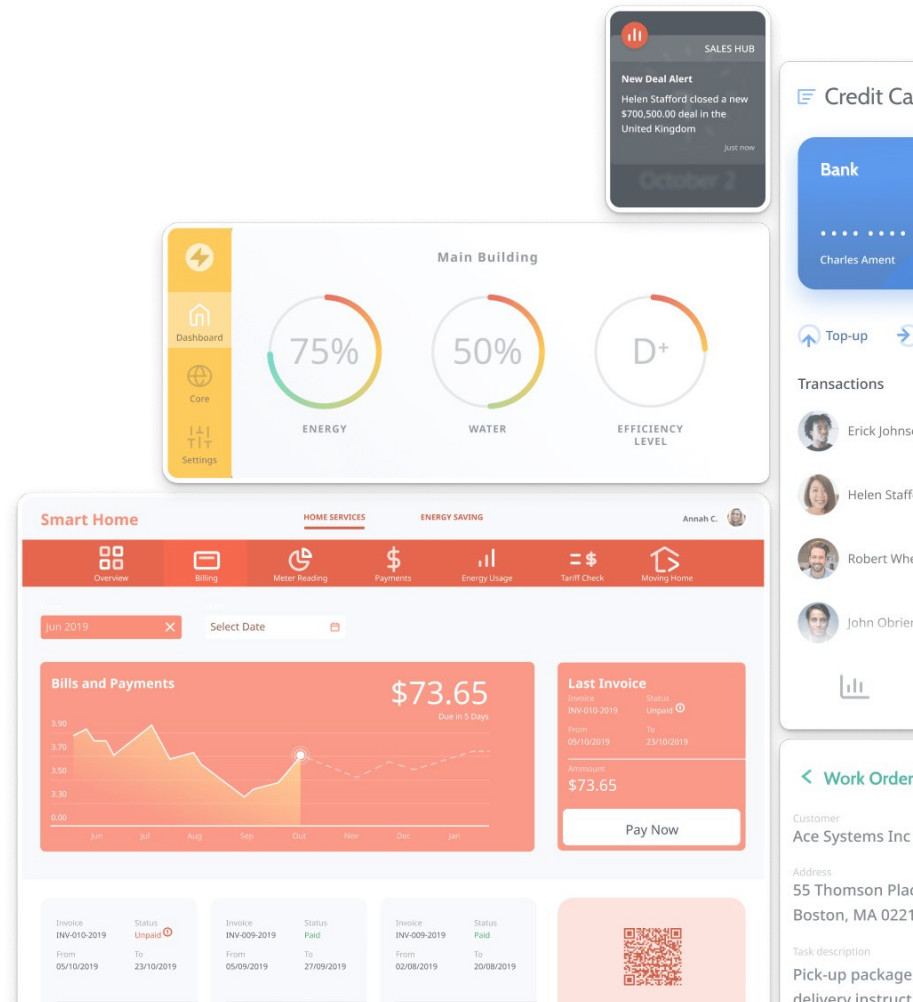


Form Validations

Reactive Developer Boot Camp



What you will learn here

- Built-in Validations
- Custom Validations
- Validation Messages

Validating User Inputs

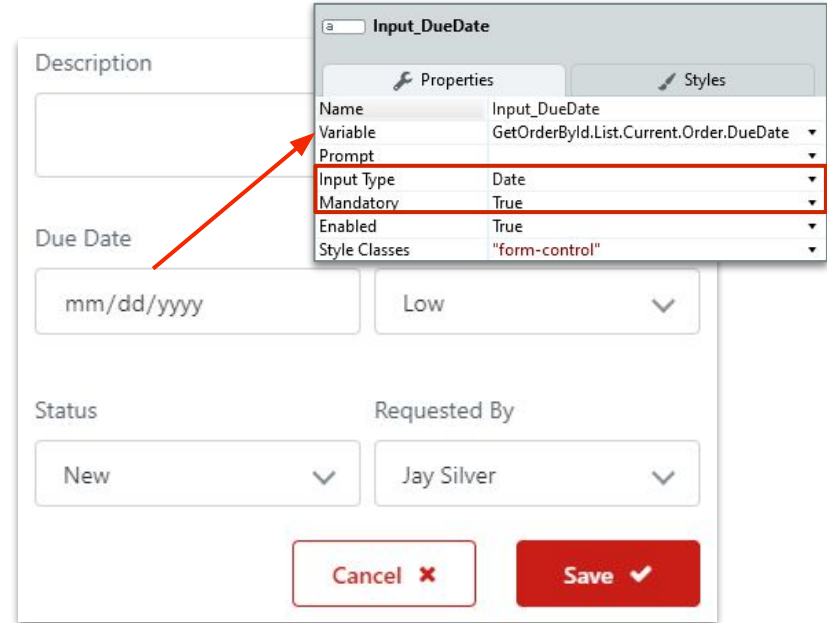
- Applications should limit user mistakes
 - **But they are inevitable**
- Input fields may...
 - Have different data types
 - Depend on other user input values
 - Require business-specific validations
- OutSystems provides mechanisms to help implement input validation

The image shows a web form with the following fields and controls:

- Description:** A single-line text input field.
- Due Date:** A text input field containing the placeholder text "mm/dd/yyyy".
- Priority:** A dropdown menu with "Low" selected and a downward arrow.
- Status:** A dropdown menu with "New" selected and a downward arrow.
- Requested By:** A dropdown menu with "Anthony Howell" selected and a downward arrow.
- Buttons:** Two buttons at the bottom right: a "Cancel" button with a red border and a red 'x' icon, and a "Save" button with a red background and a white checkmark icon.

Built-in Validations

- Built-in validations for Input Widgets are performed automatically
 - **Mandatory fields** must be filled
 - Input values must **comply with the data types** of the Variables bound to the widgets



The screenshot shows a form with several input fields. A red arrow points from the 'Mandatory' property in the 'Input_DueDate' properties window to the 'Due Date' field in the form. The 'Due Date' field has a placeholder 'mm/dd/yyyy' and a 'Low' dropdown menu. The 'Status' field has a 'New' dropdown menu, and the 'Requested By' field has a 'Jay Silver' dropdown menu. At the bottom, there are 'Cancel' and 'Save' buttons.

Input_DueDate	
Properties	Styles
Name	Input_DueDate
Variable	GetOrderById.List.Current.Order.DueDate
Prompt	
Input Type	Date
Mandatory	True
Enabled	True
Style Classes	"form-control"

Built-in Validations

- Built-in validations for Input Widgets are performed automatically
 - **Mandatory fields** must be filled
 - Input values must **comply with the data types** of the Variables bound to the widgets
- Buttons / Links inside a Form have a Built-in Validations Property
 - Set to **Yes** to perform built-in validations

The form contains the following fields:

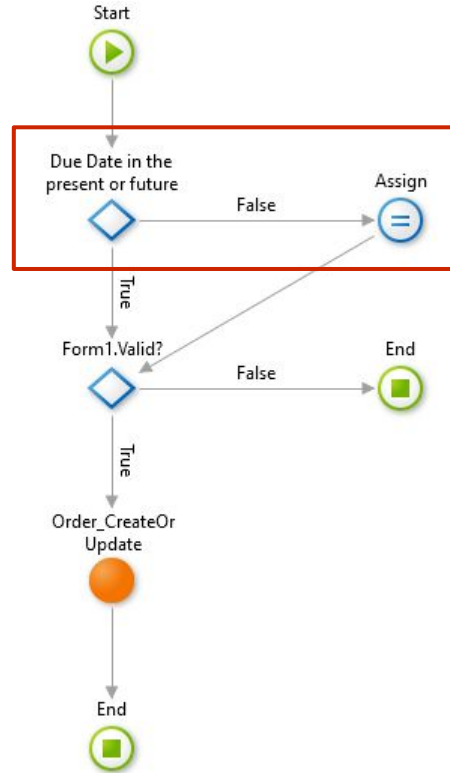
- Description:** A large text input field.
- Due Date:** A date input field with the placeholder text "mm/dd/yyyy".
- Priority:** A dropdown menu with "Low" selected.
- Status:** A dropdown menu with "New" selected.
- Requested By:** A dropdown menu with "Jay Silver" selected.
- Buttons:** A "Cancel" button with a red 'x' icon and a "Save" button with a green checkmark icon.

The 'Events' panel shows the following configuration:

Events	
On Click	SaveOnClick
(New Argument)	
Built-in Validations	Yes

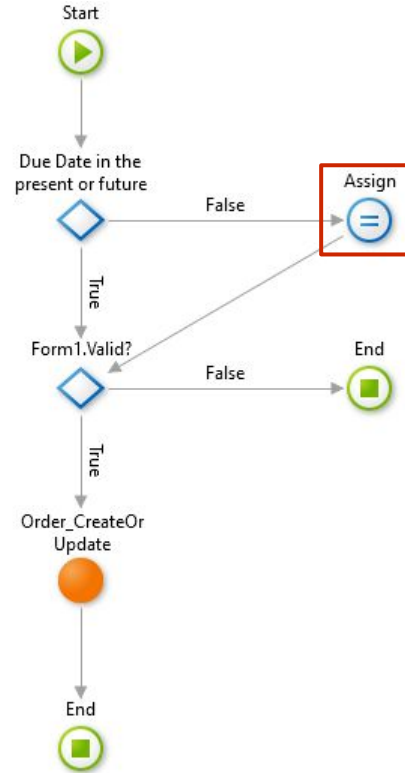
Custom Validations

- Inputs can have custom validations
 - Must be performed in the Action flow
 - For invalid Form Input fields
 - Set the **Valid** property to False
 - Set the **ValidationMessage**



Custom Validations

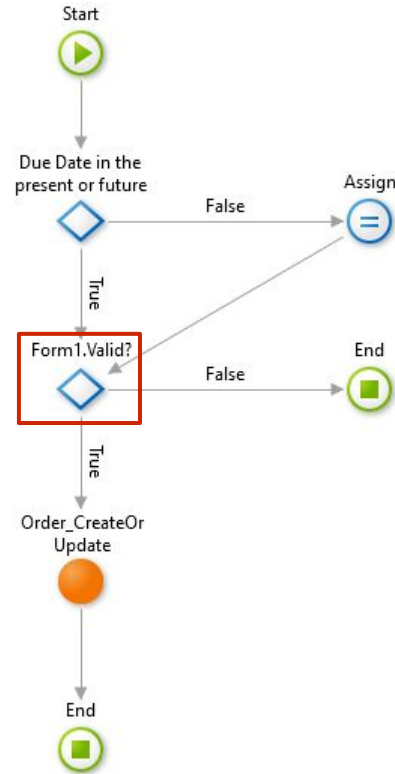
- Inputs can have custom validations
 - Must be performed in the Action flow
 - For invalid Form Input fields
 - Set the **Valid** property to False
 - Set the **ValidationMessage**



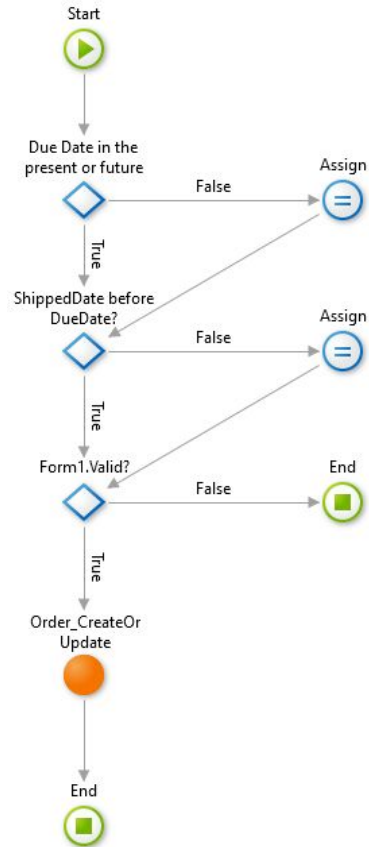
Assign	
Label	
Assignments	
Input_DueDate.Valid	▼
= False	▼
Input_DueDate.ValidationMessage	▼
= "Due date cannot be in the past"	▼

Custom Validations

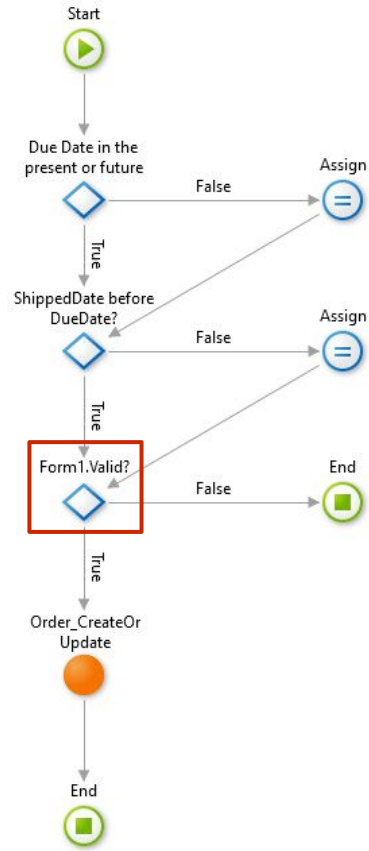
- Inputs can have custom validations
 - Must be performed in the Action flow
 - For invalid Form Input fields
 - Set the **Valid** property to False
 - Set the **ValidationMessage**
- The **Form.Valid** property should be checked after the last custom validation
 - If one Input is not Valid, the Form is automatically not Valid
 - The *Form.Valid* cannot be explicitly changed



Multiple Validations

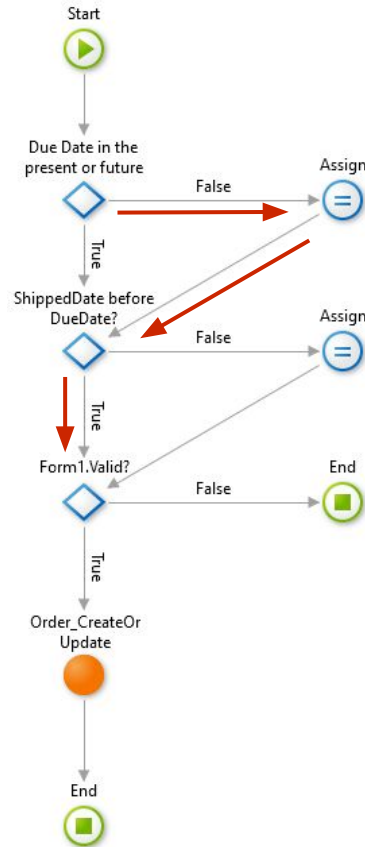


Multiple Validations



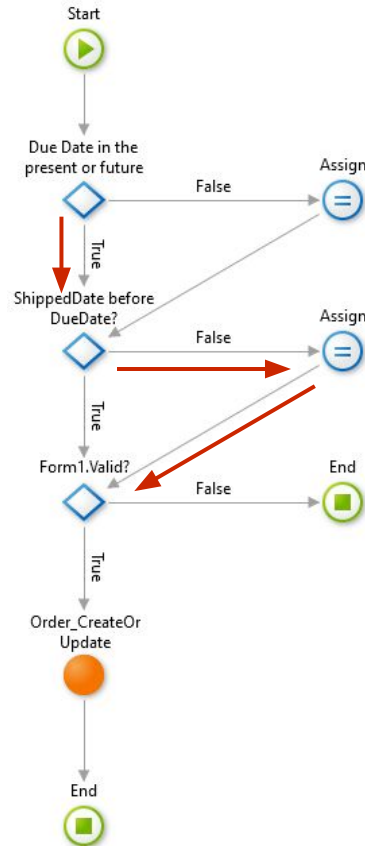
Multiple Validations

- First validation fails
- Second one passes



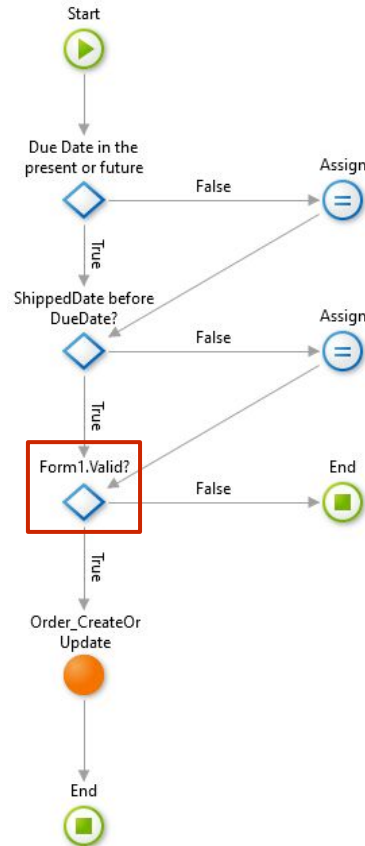
Multiple Validations

- First validation passes
- Second one fails



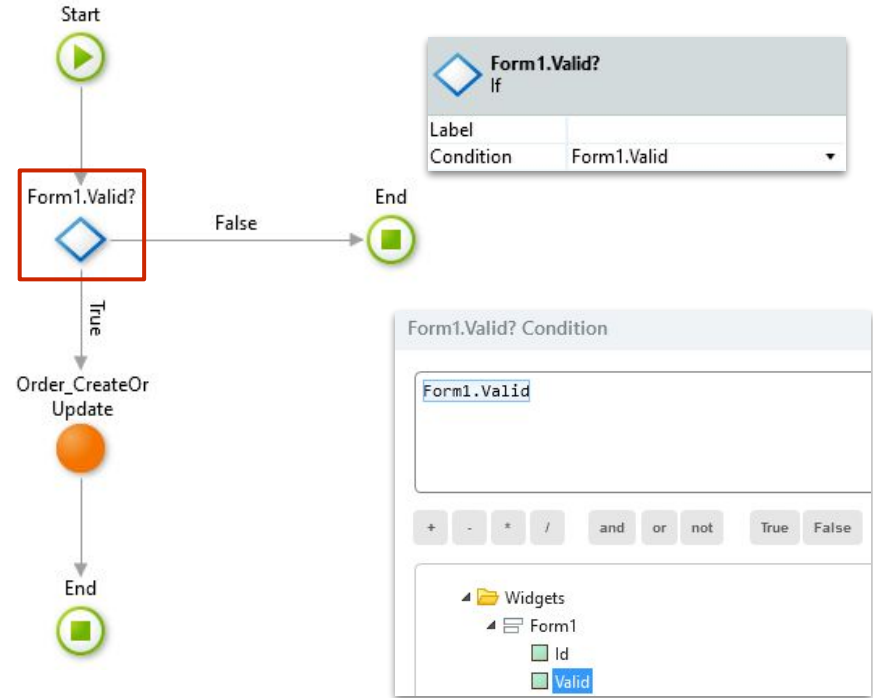
Multiple Validations

- If the Form is not valid the flow ends
- If it is valid, data can be safely used in logic



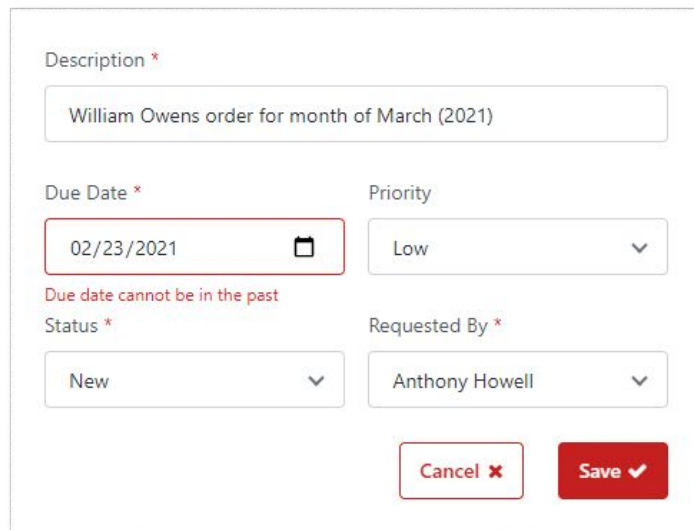
Checking Built-in Validations

- The *Form.Valid* property should be checked even if there are no custom validations
 - To avoid storing / manipulating invalid data
- OutSystems automatically performs the validations
 - But it is up to the developer to leverage that in the logic




Validation Messages


- The Valid property of the Inputs are checked when rendering the Screen
- If Valid property is set to True:
 - Displays the regular widget
- If Valid property is set to False:
 - Displays the regular widget
 - Applies the specific styling
 - e.g. a red border




Description *


William Owens order for month of March (2021)

Due Date * 02/23/2021 

Priority Low 

Due date cannot be in the past

Status * New 

Requested By * Anthony Howell 

Cancel ✕ Save ✓

Questions?



Thank You!

