

CloudCrossing BVBA
Dahlialaan 1
2950 Kapellen
Belgium



FLOW INTRODUCED VARIABLES

Flow Introduced variables

It can be that the input fields of a Screen Flow would be used as variables in the DataSource queries.

This is possible via Flow Introduced variables.

Let's take a flow as an example.

We want to output all opportunities with a close date in a certain period.

Apex Defined Variable Test 2

* Start Date

* End Date

Next

The SOQL we want to run to select the data:

General

Data Source Name: Opportunities for Account

Record Type: SOQL

Description:

Type: List of Objects

SOQL Builder Managed:

Child Data Source Settings

Parent Data Source:

Grouping Field Name:

Parent Query Field Name:

Query Settings

SOQL: SELECT Id, Name FROM Opportunity WHERE CloseDate > :startdate AND CloseDate < :enddate

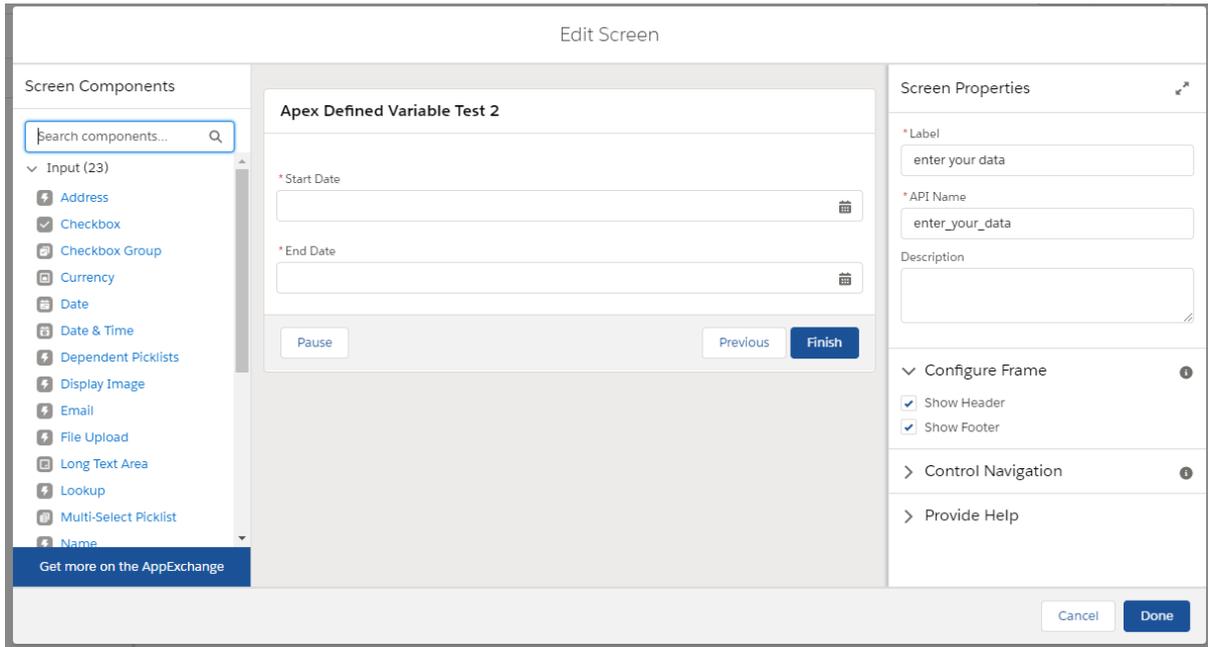
This SOQL takes 2 parameters, the “:startdate” and the “:enddate”. These parameters have to be set by the Flow.

Just as the default parameter “:recordId”, the parameters must be identified by “:”.

The flow:

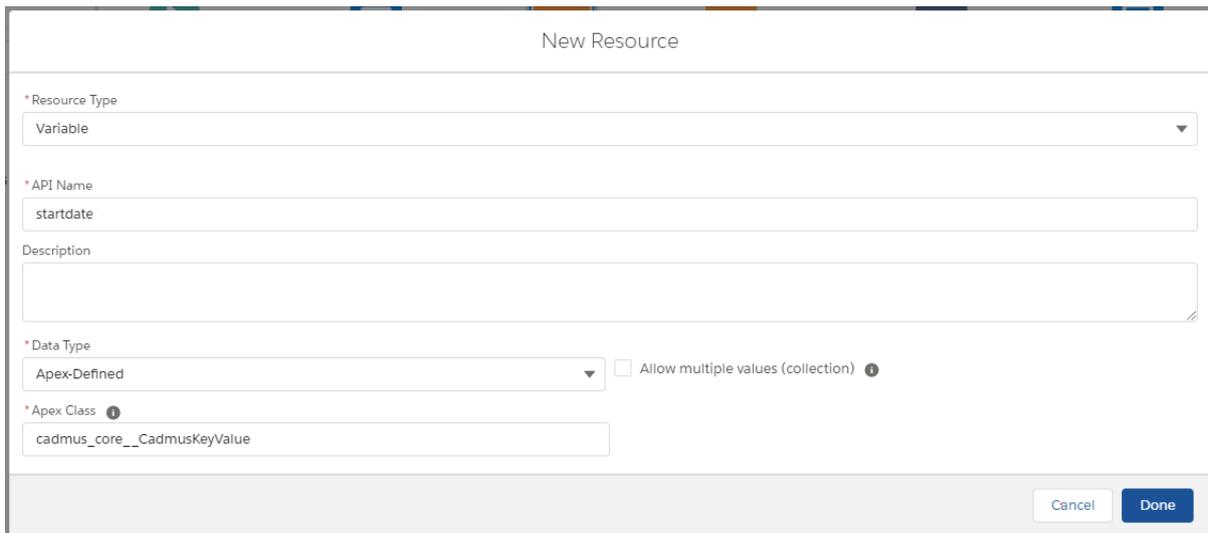


Screen 1 will select the dates:



These are Date fields that allow to select a date, in this case without a time.

Now we want to add these as parameters. So for every parameter, we must create a variable in Flow. The variable must be of type Apex-Defined and set the class `cadmus_core__CadmusKeyValue`.



Do the same for the “enddate” parameter.

These parameters have to wrapped in first a collection and second in a parameter variable to pass on to the “Call PDF Butler” component.

The Collection variable:

See the type, but more important, indicate this is a collection.

Edit Variable

* API Name
values

Description

* Data Type ⓘ
Apex-Defined Allow multiple values (collection) ⓘ

* Apex Class ⓘ
cadmus_core__CadmusKeyValue

Cancel Done

Next, the variable that wraps the collection:

This one has to be of type: "cadmus_core__CadmusParameters"

Edit Variable

* API Name
parameters

Description

* Data Type ⓘ
Apex-Defined Allow multiple values (collection) ⓘ

* Apex Class ⓘ
cadmus_core__CadmusParameters

Cancel Done

Assign the values to the variables:

Important to note, the CadmusKeyValue has to be set:

- 1) key: unique identifier and also maps on the parameter for the SOQL
- 2) value: this has to be specific for the type of the variable
 - a. valueString
 - b. valueDouble
 - c. valueDate
 - d. valueDateTime
 - e. valueBoolean

Edit Assignment

* Label: * API Name:

Description:

Set Variable Values

Each variable is modified by the operator and value combination.

Variable	Operator	Value	
<input type="text" value="{!startdate.key}"/>	<input type="text" value="Equals"/>	<input type="text" value="startdate"/>	<input type="button" value="🗑"/>
<input type="text" value="{!startdate.valueDate}"/>	<input type="text" value="Equals"/>	<input type="text" value="{!Start_Date}"/>	<input type="button" value="🗑"/>
<input type="text" value="{!enddate.key}"/>	<input type="text" value="Equals"/>	<input type="text" value="enddate"/>	<input type="button" value="🗑"/>
<input type="text" value="{!enddate.valueDate}"/>	<input type="text" value="Equals"/>	<input type="text" value="{!End_Date}"/>	<input type="button" value="🗑"/>

Assign even more variables to fill the wrappers:

- add the variables to the collection
- set the collection to the "values" property

Edit Assignment

* Label: * API Name:

Description:

Set Variable Values

Each variable is modified by the operator and value combination.

Variable	Operator	Value	
<input type="text" value="{!values}"/>	<input type="text" value="Add"/>	<input type="text" value="{!startdate}"/>	<input type="button" value="🗑"/>
<input type="text" value="{!values}"/>	<input type="text" value="Add"/>	<input type="text" value="{!enddate}"/>	<input type="button" value="🗑"/>
<input type="text" value="{!parameters.values}"/>	<input type="text" value="Equals"/>	<input type="text" value="{!values}"/>	<input type="button" value="🗑"/>

Call PDF Butler

Add an action of type "Call PDF Butler"

Set at least following input values (these are hard coded for demo purposes).

Most important one is when the {!parameters} are set. This is the wrapper variable we set earlier.

Edit "Call PDF Butler" core action

Use values from earlier in the flow to set the inputs for the "Call PDF Butler" core action. To use its outputs later in the flow, store them in variables.

* Label * API Name

Description

Set Input Values Store Output Values

A _a Alternative Name		Don't Include
A _a Country		Don't Include
A _a Doc Config Id	<input type="text" value="a0J1v00000CysmvEAB"/>	Include
A _a Doc Config Id As String		Don't Include
A _a Language		Don't Include
A _a Pack Id As String		Don't Include
A _a PDF Action Type		Don't Include
A _a Record Id	<input type="text" value="0060Y00000IhdX4QAJ"/>	Include
A _a Record Id As String		Don't Include
∞ Run Async		Don't Include
Σ Set of parameters that can be used in PDF Butler DataSources	<input type="text" value="{!parameters}"/>	Include
A _a Target Type		Don't Include
A _a Timezone		Don't Include