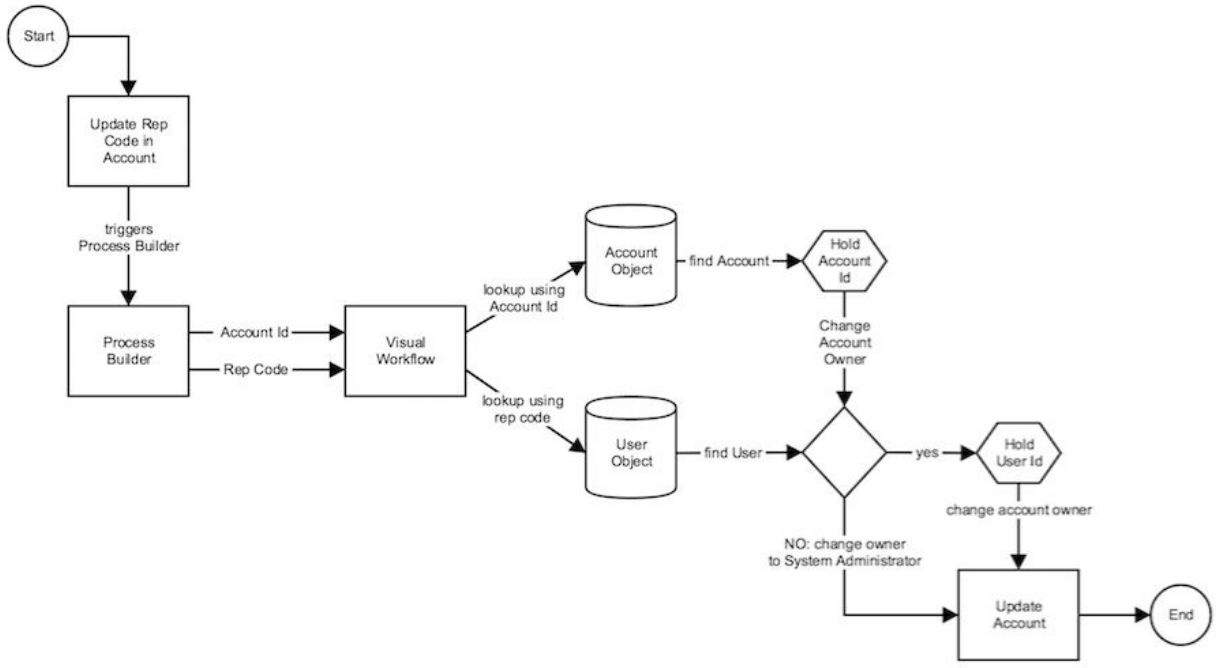
Steps for Building a Process Builder to work with Visual Workflow

Planning:

Before building any further, we drafted a design in order to determine what elements were required for the solution.



The diagram above illustrates how the whole process starts with the update of the Rep Code in the Account record. This invokes the Process Builder to pass the Account Id and Rep Code into the Visual Workflow so it can use these values to find a User with the matching Rep Code. We have one decision point to then decide who to update the Account Ownership to.

Components:

Based on our process diagram we will need:

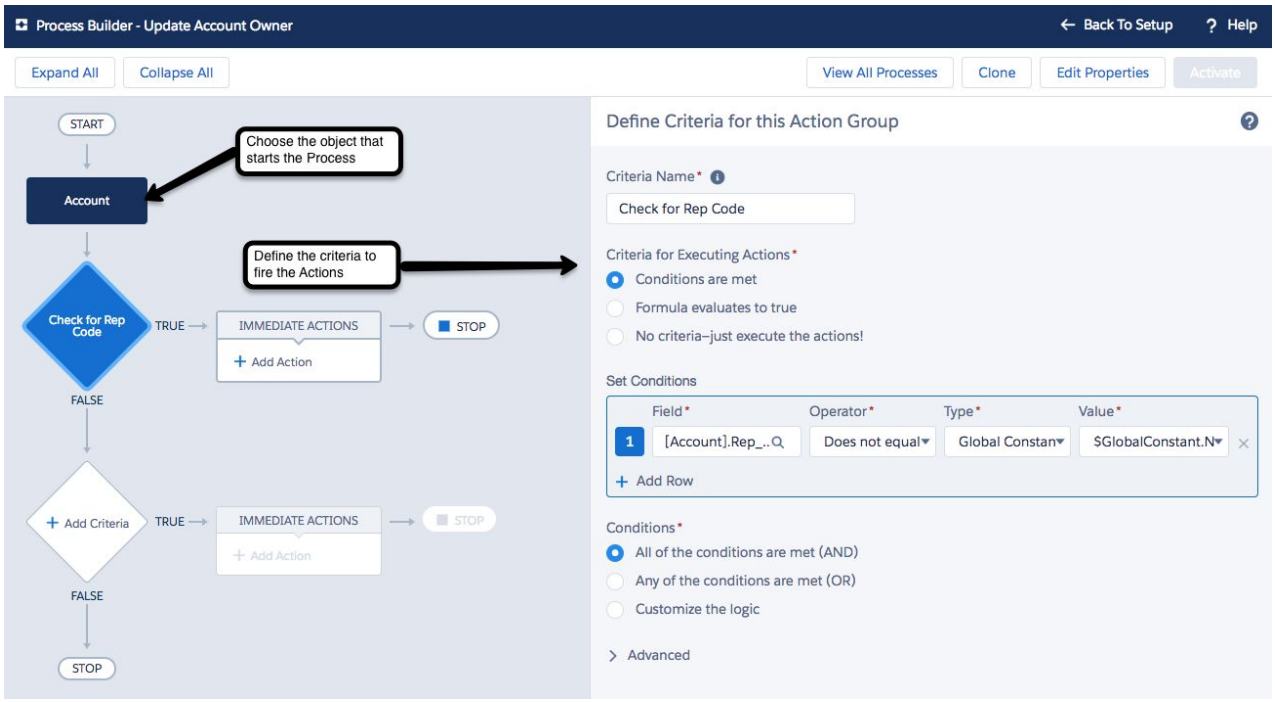
1 x Custom field on the User record to represent the Rep Code
1 x Process Builder to initiate process and pass the Account Id and Rep Code to the Flow
1 x Visual Workflow (aka Flow)
4 x Variables within the Flow

2 x Fast Lookups (for the Account and the User object)
1 x Decision point (to determine how the Record will be updated)
2 x Record Updates (One to update the Account Ownership to the Sys Admin User and one to update the Account Ownership to the User with a matched Rep Code)

Execution:

Step 1: *create the Rep Code* as a custom field on the User record and make it unique. This way there can be no confusion as to who the correct user should be.

Step 2: *create the Process Builder*, but only up to a certain point. As we have yet to build our Visual Workflow, we have no variables to pass our details into. For now, we'll choose the Object that will start the process and define the criteria that will be used to fire the Actions.



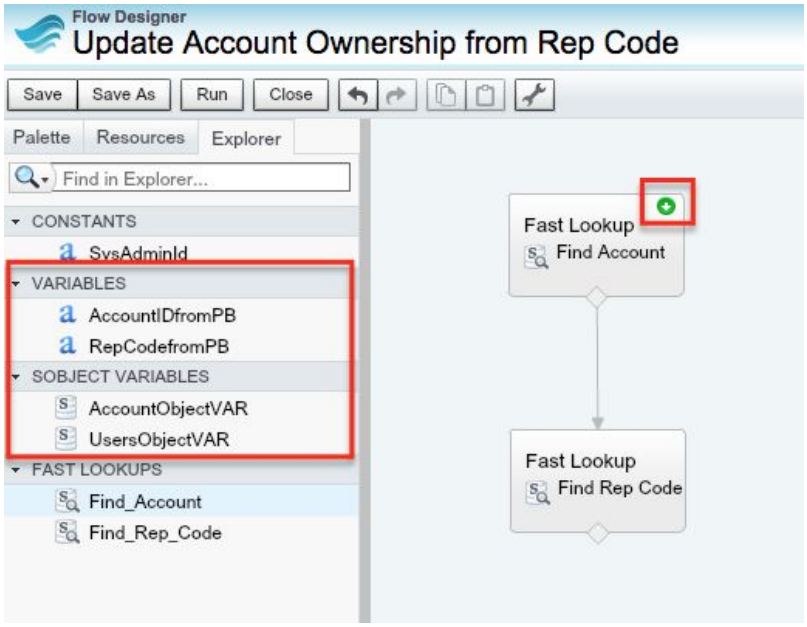
Step 3: *create the Visual Workflow*

1. The first steps to perform with our Flow are to set up the required Variables. As mentioned above we'll need:
    a. 1 Input variable to house the Account Id passed in by the Process Builder (labelled AccountIDfromPB)
    b. 1 input variable to house Rep Code coming from in by the Process Builder (labelled RepCodefromPB)

c. 1 Input/Output SObject variable to hold the Account Record so it can later referenced in the flow (labelled AccountObjectVAR)
d. 1 Input/Output SObject variable to hold the User Record so it can later referenced in the flow (labelled UsersObjectVAR)

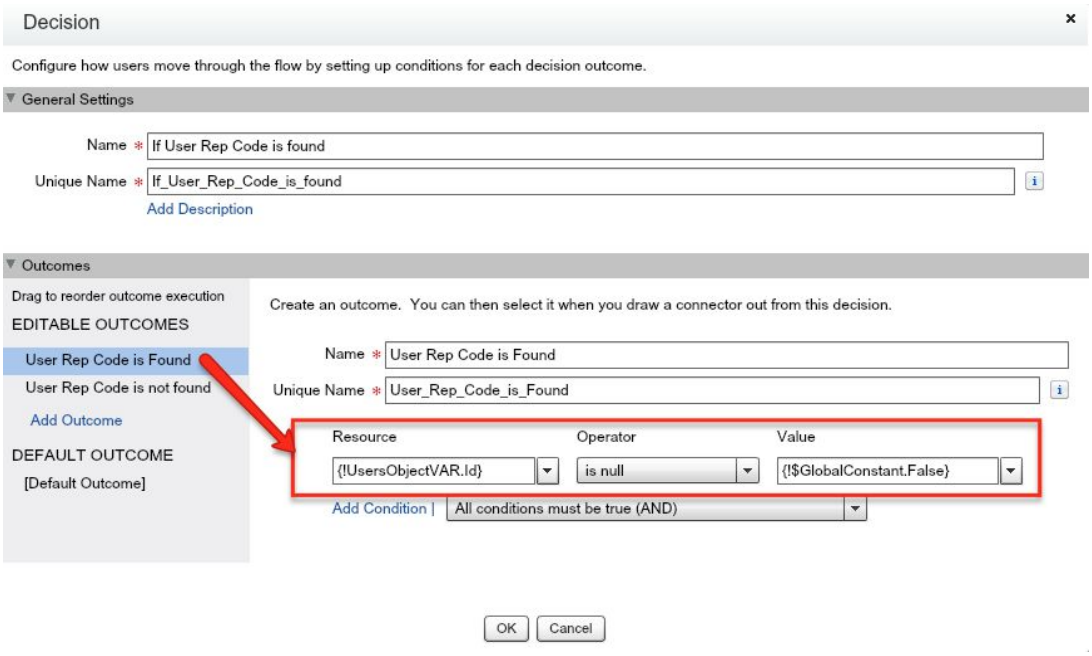
2. We can now create our Fast Lookups. The first Fast Lookup will take the incoming id from the AccountIDfromPB variable, lookup up records in the Account object to find the Account record that matches this Id and then store that record in the SObject variable (AccountObjectVAR)

3. The next step in the Flow will be to find a User record with a Rep Code that matches the Rep Code value passed in from the Process Builder (stored in the RepCodefromPB variable) and store that record in the SObject UsersObjectVAR.



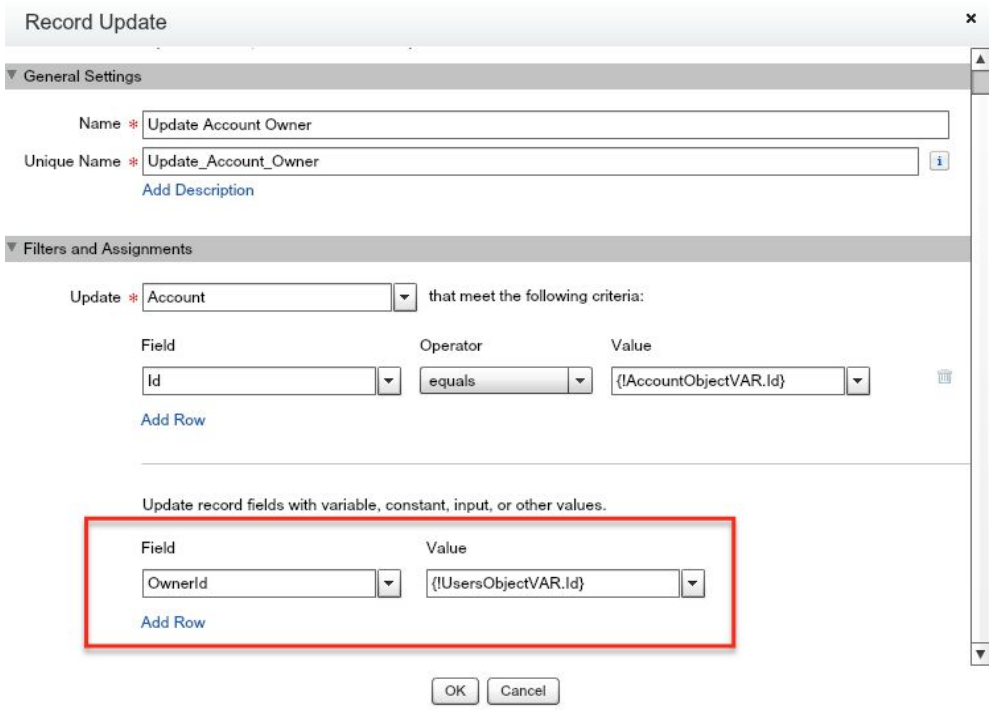
The explorer tab shows the four variables we have created as well the order in which our Fast Lookup will execute. NB. we have established the Find Account Fast Lookup as the First Step in the Flow.

4. The next step of the Flow will be to create the Decision logic. This will determine how the Ownership of the Account record will be updated when a matching User Record is found. Within the condition statement for this outcome we're able to select a specific field value from the User record stored in our SObject Variable - UsersObjectVAR and express the statement as User Id is null = FALSE



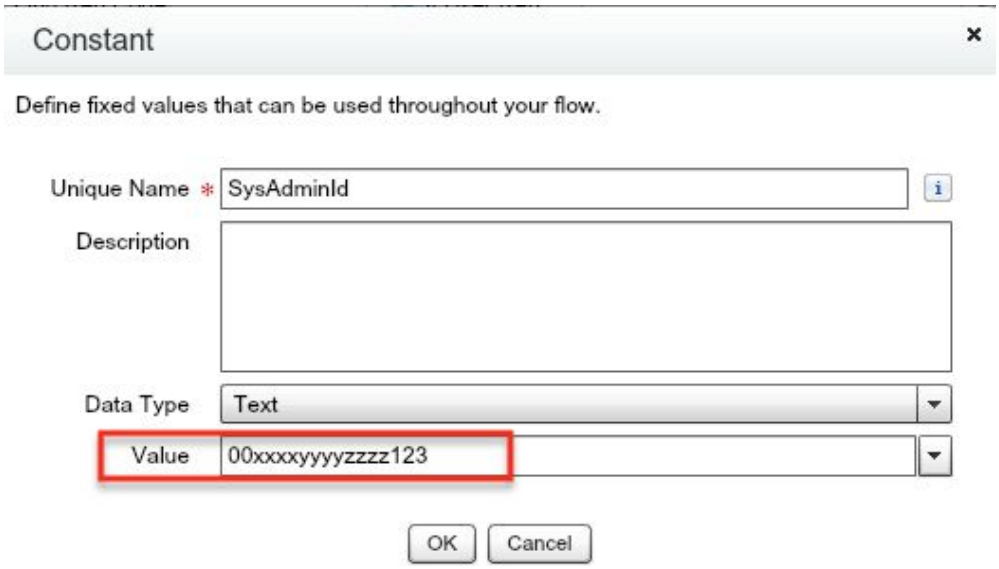
Our alternate Outcome (or the 'Else' statement), details the opposite scenario where the User Id is Null, meaning that the SObject Variable was unable to find a User record with a Rep Code that matches our Rep Code variable (RepCodefromPB).

5. We're now ready to create our Record Update actions. The first Record Update will branch from the Decision Outcome of when a matching User is found. The update will firstly reference the Id from the record stored in our SObject variable (AccountObjectVAR) to find the correct record to update. It will then update the Account Owner field with the Id of the User record stored in the Users SObject variable (UsersObjectVAR).

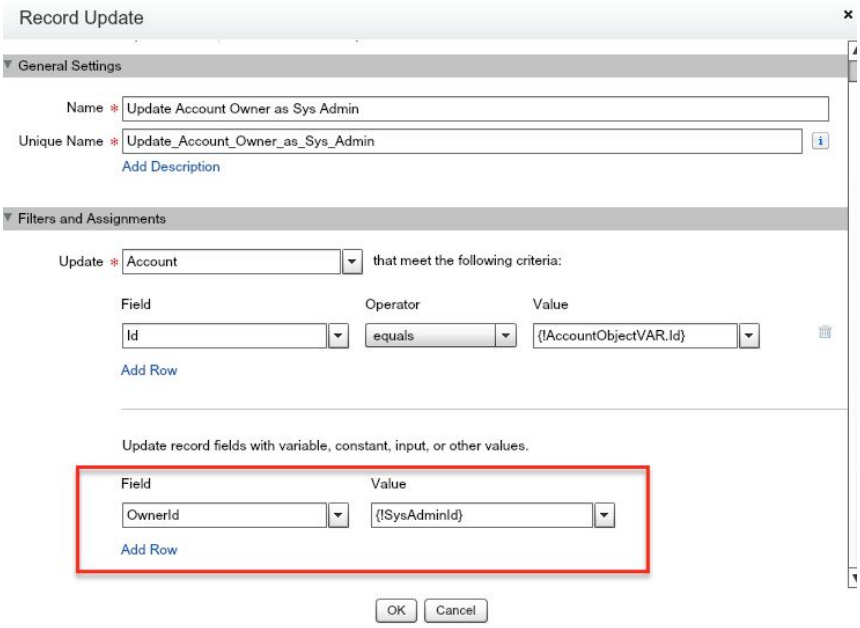


NB. the id field has been appended the end of the Value fields in the Record Update screen.

6. The second Record Update is for the User Rep Code not found Outcome and will update the Ownership field on the Account record with Id of the Sys Admin user. As we're updating the Account Record with a static value (ie the Id of the Sys Admin User record), we'll create a Constant to store this id.

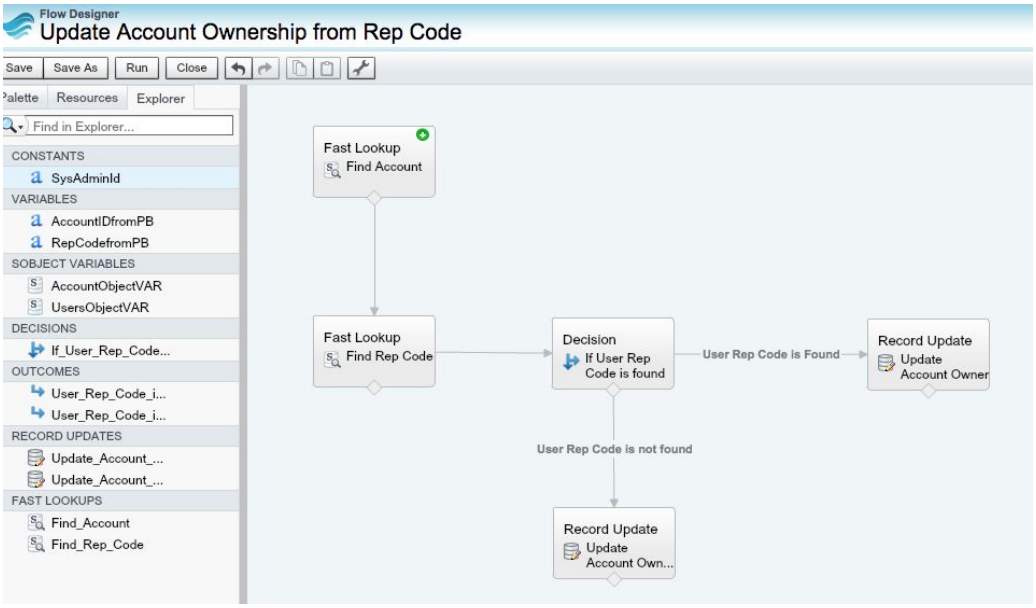


We can either create the Constant before creating the Record Update action or we have the option to create from within the Record Update action. Either way it needs to store the Id of the Sys Admin User.



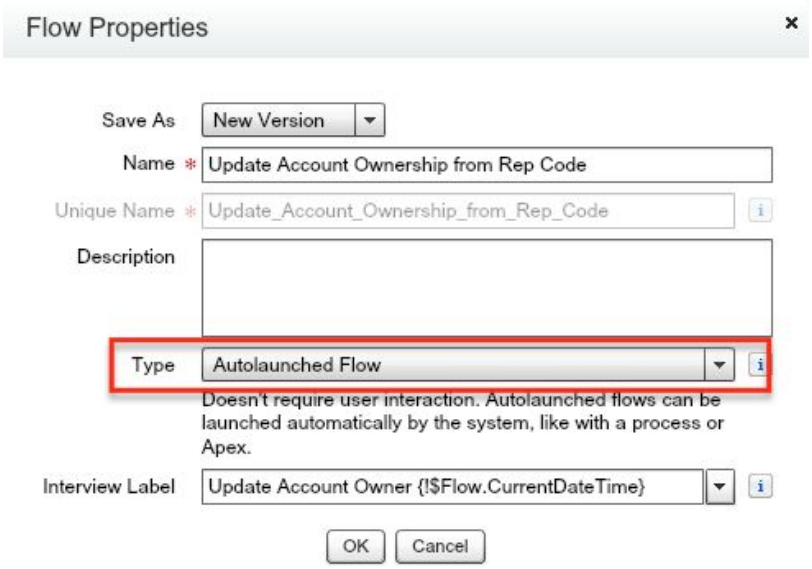
Similar to first Record Update we created, we need to tell it which record to update and which field will be updated and by what. Instead of referencing a variable to update the Account Owner, we're using the Constant (SysAdminId).

Tying it all Together: Now that we have created all of our Flow elements, they can be linked together in a logical sequence.



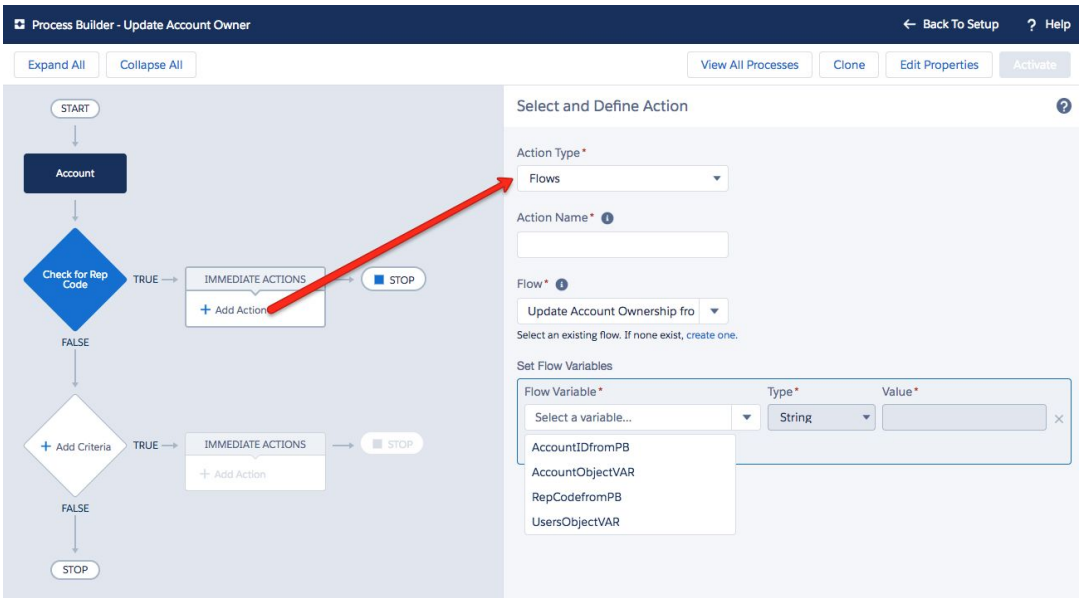
Step 4: Finalising the Process Builder

In order for the Flow to referenced by the Process Builder, it needs to be an Autolaunched Flow. By default, Visual Workflow will save the Flow as Type ‘Autolaunched Flow’ if no Input screens have been added.



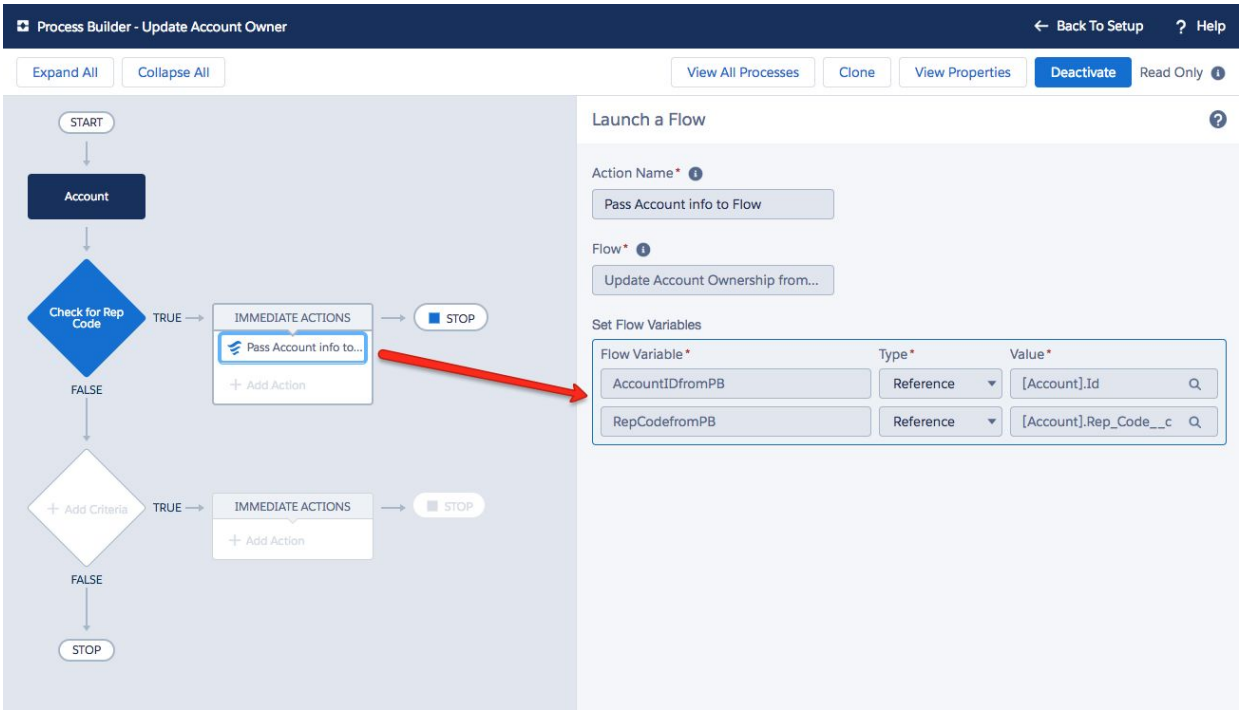
An Autolauched Flow is a type of Flow where no user interaction is required.

Returning to our Process Builder, we can now add an Actions to pass values into the Flow. By selecting Flow as the Action Type, choose the Flow we have just finished and then select the Variables created within the Flow.



As per our process diagram earlier, we can set the Flow variables to pass data into so that;

1. The Account Id will pass into the AccountIDfromPB variable
2. The Rep Code on the Account will pass into the RepCodefromPB variable



Now we're ready to Activate the Process Builder and test it out!