Salesforce CTI Adapter - CTI Flows cheat-sheet

When you build CTI flows, you can use some additional information stored in the internal objects that are provided by the integration layer.

The objects available and the content of the objects depends on the type you handler (event) you are tying your CTI Flow to as well as the source.

Examples:
You can access a result value of an action block by using:
`$.actions.uid-123.results.fieldName`

or you can access the phone number passed in a click to dial command by using:
`$.payload.number`

Here is a description of the objects available for some of the most common source/event combinations

- Voice contact inbound and callback - all events:
  - `contact,`
  - `agent`
- Chat contact
  - for onMessage:
    - `payload,`
      - contactId
      - initialContactId
      - displayName
      - participantRole
      - participantId
      - content
    - `contact,`
    - `agent`
  - for all other events
    - `contact,`
    - `agent`
- Salesforce Agent
  - `payload (details for each event in the openCTI documentation)`
- Salesforce UI ClickToDial:
  - `payload` (details in openCTI documentation),
    - number
    - recordId
    - recordName
    - objectType
    - personAccount
  - `agent,`

- `contact`

The following are high-level name spaces that can be used to retrieve the data from an existing object and use it as a parameter for one of you action blocks.

`$.agent.name`
`$.agent.extension`
`$.agent.dialableCountries`

All the agent properties can be accessed using the dedicated blocks: Get Agent Name, Get Agent Extension, Get Agent Dialable Countries.

All the contact properties available as result fields in Get Contact Properties can also be accessd using the name space model:

`$.contact.contactId (string)`
`$.contact.originalContactId (string)`
`$.contact.type (connect.ContactType)`
`$.contact.status (connect.ContactState)`
`$.contact.statusDuration (number)`
`$.contact.queue (connect.Queue)`
`$.contact.queueTimestamp (number`
`$.contact.attributes (Record<string,value>)`
`$.contact.isSoftphoneCall (boolean)`
`$.contact.isInbound (boolean)`
`$.contact.isConnected (boolean)`
`$.contact.isTransfer (boolean)`
`$.contact.parsedNumber: { country: string; phone: string}`
`$.contact.phone (string)`
`$.contact.country (string)`
`$.contact.channel (string)`
`$.contact.contactMissed (boolean)`
`$.contact.contactStartDate (Date)`
`$.contact.contactStartDateTime (string)`
`$.contact.formattedNumber (string)`
`$.contact.formattedNumberE164 (string)`
`$.contact.rawPhoneNumber (string)`
`$.contact.callType (string)`

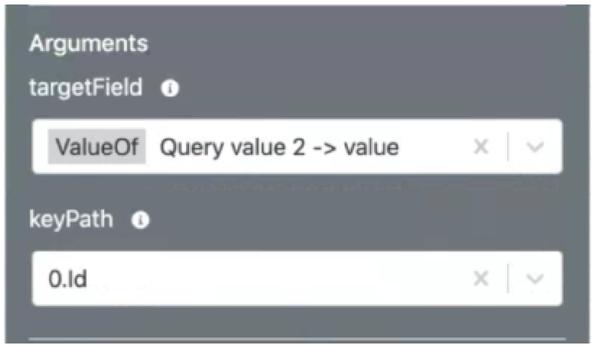
Sample SOQL statements
Looking up contacts by account Id. This will return all the contacts linked to the account
[SELECT Id, Phone FROM Contact WHERE AccountId =: {value_of_acctId)]

Retrieve all the tasks associated with an account and order them from the most recent to last
[SELECT Id, CallDurationInSeconds, WhoId, CallObject, CreatedDate, CallType FROM Task WHERE WhatId =: {accountId} ORDER BY CreatedDate DESC]

Extract Value samples:

1. Accessing array objects

Select the target field as the action bloc that returns or holds an array of elements (example the result for a SELECT SOQL query that returns multiple objects in an Array). If you want to read the property of the first selection refer to the first element of the array by Index 0 and specify the path of the property key as follows



Arguments

targetField: ValueOf Query value 2 -> value

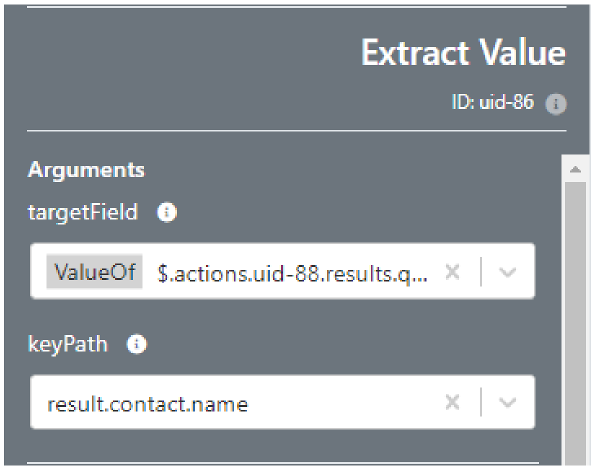
keyPath: 0.Id

2. Nested objects

Select the object you want to extract the value from in the targetField, same as in the sample above
Specify the key path by indicating the structure of the object you are trying to access as follows:
If the object that represents the value of the target object has this structure

```
{
 result:
        {
                contact:
                {
                        name: “Jane Doe”
                }
        }
}
```

your keyPath is result.contact.name



Extract Value

ID: uid-86

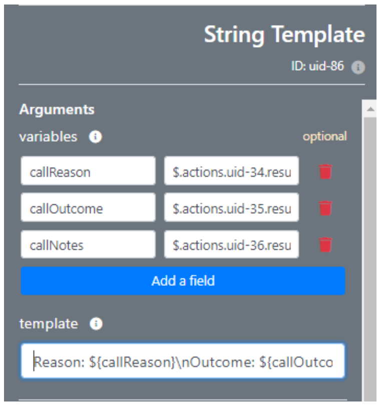
Arguments

targetField: ValueOf $.actions.uid-88.results.q...

keyPath: result.contact.name

String Template example

If your string needs to be formatted by using multiple constant and dynamic values like in this example (building the comments line for an activity logged at the end of a call) subject = `Reason: ${callReason}\nOutcome: ${callOutcome}\nNotes:${callNotes}`



String Template

ID: uid-86

Arguments

variables — optional

callReason	$.actions.uid-34.resu
callOutcome	$.actions.uid-35.resu
callNotes	$.actions.uid-36.resu

Add a field

template

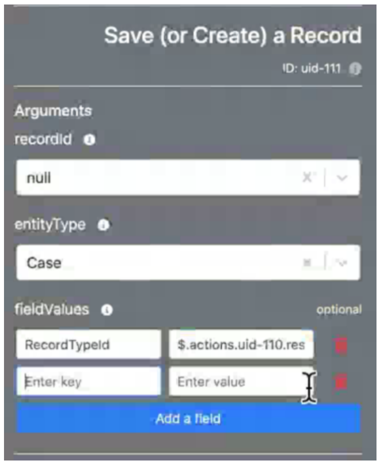
Reason: ${callReason}\nOutcome: ${callOutco

Save (or Create) Record

This bloc allows you to create or update entities such as Activities, Tasks, Cases. It is mostly used at the end of a call to either create a call log or to create a case.

1. If you specify a recordId, the action block will update the field Values you are configuring as parameters and the entityType is not used
2. If you set the recordId value to null, the action block creates a new entity of the specified type (Case, Contact, Account, etc)

Note: use Create Task if you want to create a task, the Save/Create saveLog based action should be used for all other objects.



Save (or Create) a Record

ID: uid-111

Arguments

recordId: null

entityType: Case

fieldValues — optional

RecordTypeId	$.actions.uid-110.res
Enter key	Enter value

Add a field