Netsuite - Mapping Custom Fields to COs

Last Modified on 05/13/2020 6:22 pm EDT

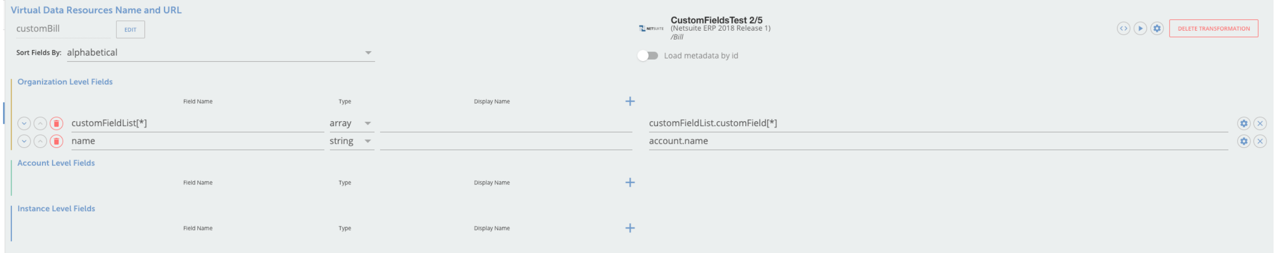
If you are trying to map one of Netsuite's custom fields to a CO, take a moment to review the following approach. In this example, we will use **/bills** as the example.

If you've enabled custom fields on your bills resource, you can see those custom fields through a **GET /bills** request. The custom fields are part of your response, and look like the following:

```
"customFieldList": {
  "customField": [
    {
      "internalId": "100",
      "scriptId": "custbody1",
      "value": "Demo"
    }
  ]
}
```

As you can see from the payload above, custom fields are returned in a customFieldList object that points to a customField array of objects. This format can be a bit unwieldy to work with, so mapping these fields to a CO is viable solution to dealing with the complexities that these custom fields introduce.

First, you're going to want to create a CO that you maps to the **/bills** resource. In this example, we have a customBill CO with only a few fields. You can see the CO below:



Virtual Data Resources Name and URL

customBill

Sort Fields By: alphabetical

CustomFieldsTest 2/5
(Netsuite ERP 2018 Release 1)
/Bill

Load metadata by id

Organization Level Fields

Field Name	Type	Display Name	
customFieldList[*]	array		customFieldList.customField[*]
name	string		account.name

Account Level Fields

Field Name	Type	Display Name

Instance Level Fields

Field Name	Type	Display Name

I created a customFieldList array field in the object definition. This field can be renamed to anything that would be helpful to your app needs. The customFieldList is then mapped to the array inside of the customField object. The results of this mapping are:

```
"customFieldList": [
  {
    "internalId": "100",
    "scriptId": "custbody1",
    "value": "Demo"
  }
]
```

While this example is pretty simplistic, it can help you begin to understand what you would need to start manipulating Netsuite's custom fields using COs.