Breadwinner Custom URL

Breadwinner was designed to create Sales Orders in NetSuite from any object or objects in Salesforce. This can be accomplished either automatically (through Triggers and Global Classes) or via a guided wizard. The configuration of that guided wizard is detailed below.

Restrictions

Many of our restrictions are restrictions of NetSuite. If you can't create a record in NetSuite manually via copying and pasting, then you probably can't use our Custom URL approach to create that same record.

Additionally, Breadwinner does not include logic that is already native to Salesforce. For instance, this tool cannot merge two Salesforce fields into one NetSuite field. However, you may create a Salesforce Formula that merges those two Salesforce fields, and use this tool to pull that single formula field into Salesforce.

Preparation

Three objects/records are required for this process:

1. A NetSuite object (technically, this is an object that is included in the Breadwinner managed package, though the object will start with "NetSuite…"). This object must have a lookup to the Originating Object.
2. An Originating Object/Record. This is the record that contains the main information you want to go on the Sales Order, such as Date, Customer, etc. (this record may optionally have a child object shown as a single related list, with many records in that related list that become the line items in NetSuite).
3. A NetSuite Customer record. However, most people reference a Salesforce Account instead of the NetSuite Customer, and Breadwinner will prompt the user to either select one of the existing NetSuite Customers associated with that Salesforce Account, or to create a new NetSuite Customer.

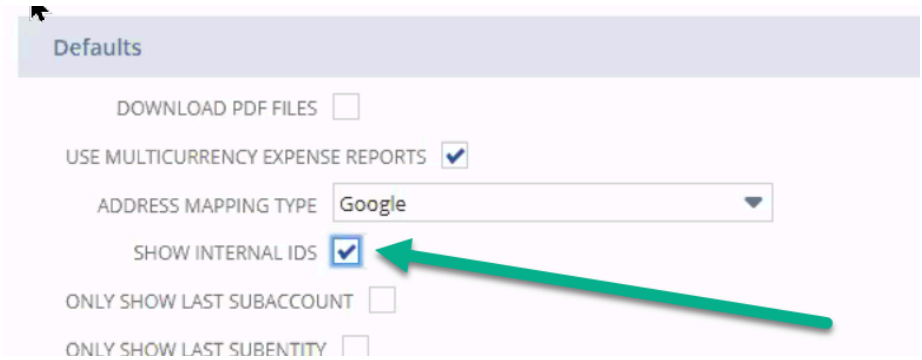
The NetSuite Object (typically either the NetSuite Estimate, Sales Order or Invoice) must have a lookup to the Originating object/record. For instance, if creating Sales Orders from Opportunities, you will need to create a lookup field from the NetSuite Sales Order object in Salesforce to the Opportunity object.

And the Originating object will need either a direct lookup to an Account (either via a lookup or master/detail) or a formula field that references a lookup to an Account. (Or you can create a formula field/lookup pointing directly to a NetSuite Customer, instead of the formula field/lookup pointing to a SF Account).

To start with, you will need to create a lookup field in Salesforce on the “NetSuite Invoice” object or the “Netsuite Sales Order” object, pointing to the Originating object. And, we strongly recommend you add the relevant related list to that object’s page layout.

Exposing the Picklist ID’s

If you wish to create or modify a record with a Picklist in NetSuite, then you will need to pass Breadwinner the ID of the Picklist Option. The visibility of Picklist ID’s might not be enabled in your NetSuite org. To enable visibility of Picklist ID’s (which will only be visible in the Customisation section of NetSuite, and not visible to end users), please go to Home > Set Preferences. On the General Tab, in the Defaults section (usually on the right) you will be able to enable “SHOW INTERNAL IDS”



Conventions in this Document

In the formulas below, the left side of the equation is our field, and the right side of the equation (after the = sign) is for you to populate based on the api names in your Salesforce org. We have also used italics for the section that should be replaced by your values. Do not use the right side values (in italics) in your formulas.

For instance, if you see
&parent_account=*api_field_name__c*

Then your formula would include

&parent_account=

but the *api_field_name__c* would be replaced by the API name of the field in your Salesforce org.

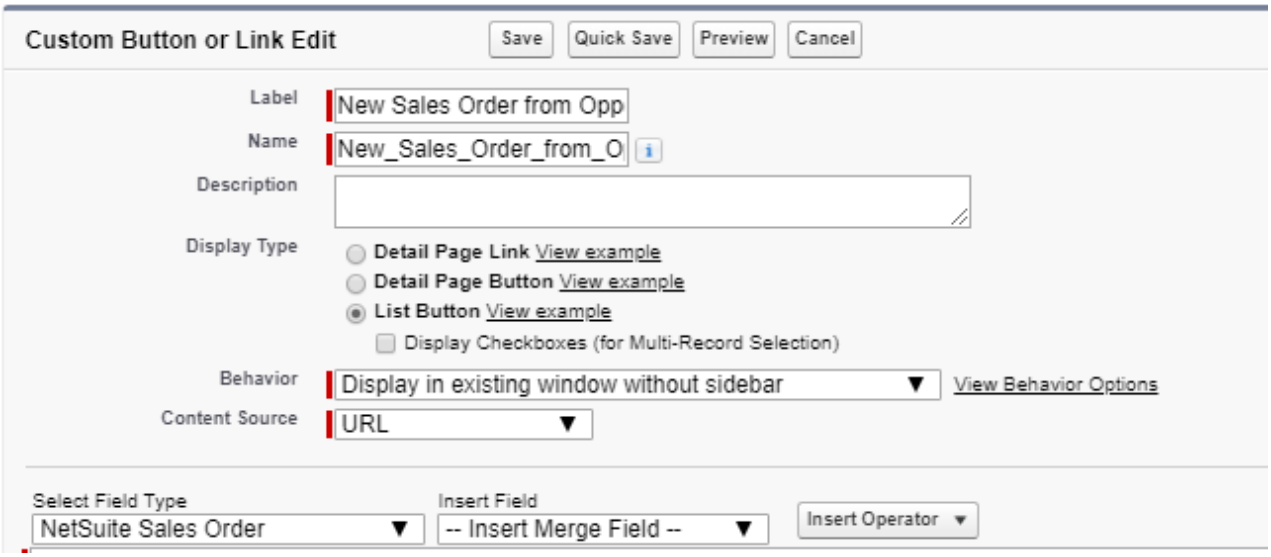
Ease of Use Tip

We have made a Custom URL Wizard to help you create these fields. To view this, simply make a button with the text below, quick-save it, add it to the originating objects Sales Order Related List page layout, and click.

This button should be created on the Sales Order object (as a Button, not an Action or Link). We recommend setting the name of the button to be something like “New NetSuite Sales Order from the [Something Object]”.

Breadwinner for NetSuite
/apex/breadwinner_ns__CreateSalesOrder ?testing=true

Once created, the button should be displayed on the Page Layout of the Object in question (the one you wish to create Sales Orders from), on the Sales Orders related list on that object. (in the page layout for that object, click the wrench on the related list, and then click Buttons to change the buttons on the related list). This is also the time to remove the standard New button by unchecking it, as the standard New button won’t work on Custom Objects.



URL Formula

The Custom URL Wizard will give you the formula you need. However, you may wish to further modify the URL, or just understand it further.

Also, you can create multiple buttons per object, with different formulas to be allowed Sales Orders to be populated in different ways from different fields.

When creating formulas in Salesforce, you can create a formula over many lines, separated by a new line for readability.

Button on Sales Order Related List

You can create a button on the Breadwinner Sales Order object itself, as a List Button (without multi- checkboxes). After creating it, you will need to display it on the related list of the originating object's page layout.

These buttons will be added to the Page Layout of the originating object, by clicking the wrench/spanner on the Sales Orders related list.

Button or Link on Originating Object

You can also create the button, or link, on the originating object. This will be added directly to the page layout of the originating object. However, this can cause slight usability issues, which is why we recommend the button go on the Sales Orders related list.

URL Structure

The URL will always start with a specific line, and contain at minimum two elements, specifying the originating record, and the associated Salesforce Account (or a NetSuite Customer, as noted below)

It is not necessary to add naX.salesforce.com or anything similar; the button can simply start with /apex/….

Breadwinner for NetSuite
/apex/breadwinner_ns__CreateSalesOrder ?originating_record={!*salesforce_record_ID*} &parent_account=*api_field_name__c*

The above presumes you are creating a record under a Salesforce Account (which in turn will look for, or create, a NetSuite Customer under that Salesforce Account). Less common, but equally valid, you might wish to create a NetSuite record under a NetSuite Customer directly. In that case, your URL should look like

Breadwinner for NetSuite
/apex/breadwinner_ns__CreateSalesOrder ?originating_record={!*salesforce_record_ID*}

&netsuite_customer=*api_formula_or_lookup_to_netsuite_customer__c*

However, as this is the less common case, the rest of this document will presume that you are creating a record under a Salesforce Parent Account

Please note that the only field that should contain a Salesforce ID is the originating record field. Additionally, this is the only field that will contain brackets, {}. All other fields will contain the API name of the field, *without* brackets. The API name is easiest copied from the Salesforce Setup page showing all the fields on the originating object.

Testing

When building the URL, we recommend using *&testing=true* to access our testing page. You can temporarily build this into the URL formula, or append it to the URL after you click on the New Sales Order Button. This testing page will display the API names we have passed and the corresponding values we have retrieved based on the API names. It is only visible to Salesforce Admins, and only visible if testing is set to True.

/apex/breadwinner_ns__CreateSalesOrder
?originating_record={!*salesforce_record_ID*}
&parent_account=*api_field_name__c*
&testing=true

NetSuite Forms

NetSuite uses forms to control the input of a Sales Order, Estimate or Invoice. You can specify a form by using this in the URL

Forms can be referenced either via another field (which contains NetSuite Form Id) on the originating object, or as a text field in the formula. So you may use either approach:

&custom_form=custom_form_api_field__c	&custom_form=”123”

(NetSuite accepts a form ID, rather than a form name)

NetSuite Picklist and Lookup Field IDs

For both Picklist fields and Lookup fields, NetSuite requires the ID Value during record modification or creation.

Please be aware that in NetSuite, a user may be familiar with selecting from a dropdown/picklist menu and looking for a text value. However the data passed to NetSuite by Breadwinner must be a numeric ID. (Please note this is the opposite of Salesforce, which requires the text value when populating a picklist).

Finding the ID of Standard Picklists

The ID's of NetSuite Standard picklists can be found in different locations. Some of the possible places these standard picklist values can be found include:

- Setup > Accounting> Accounting List
- Setup > Sales> CRM List
- Setup > Company> Department
- Setup > Company> Classes

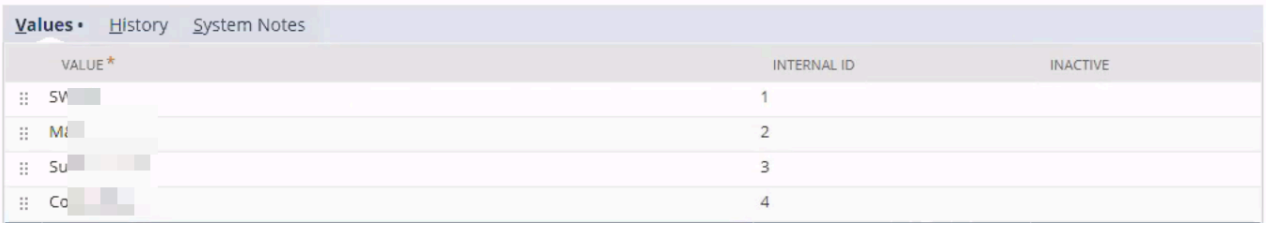
For instance, the Department and Class picklist can be found in their respective locations under the Company Setup. However, other picklists will be in other locations.

Finding the ID of Custom Picklists

The ID's of custom picklists can be found under Customization > Lists, Records & Fields > Lists.

Note: If you can't see the Picklist ID, please go to the section [Exposing Picklist ID's]

Here is an example Picklist. Please note the Internal ID column.



Populating NetSuite Picklist and Lookup Fields

You will need to pass the Internal ID to Breadwinner, rather than the Value. You can pass this as either a field which contains the Internal ID (formula fields work well for this, as they can be based on a Salesforce Picklist) or you can pass this to Breadwinner as raw data within the Custom URL itself without creating any fields (only an appropriate choice if you always use the same value).

Populating Standard Lookup Fields

Once you have the ID of a standard field, you can pass it to Breadwinner either as a field or as a value. As an example, here is a standard picklist, Department.

&department=department_api_field__c	&department=”123”

And here is the Class field being populated.

&class=sales_order_class_api_field__c	&class=”123”

Populating Custom Lookup Fields

NetSuite Custom Picklist and Lookup Fields present an additional layer of complexity. For NetSuite Standard Picklist and Lookup fields, simply passing the ID is enough. However, for custom fields, NetSuite requires that the field type be specified. And, thus, Breadwinner requires the field type to be specified. To specify the field type, we require additional information in the Custom URL.

Picklists have the type *picklist* and we require this to be in the Custom URL.

For Transaction Custom Fields

(for clarity, in NetSuite a ‘Transaction’ is a Sales Order, Estimate, or Invoice)

We designate a custom field on a transaction by starting the line of the Custom URL as

*&cf_*

Which is followed by the type of the custom field (in this case a picklist) followed by the API name of the NetSuite field. Here is the theoretical construction.

&cf_*type*_**netsuite_api_name**= formula_field__c	&cf_*type*_**netsuite_api_name**=”123”

For the sake of readability, we have added italics and bold to different parts of the query string. The italic text refers to the NetSuite field *type* which may be one of: picklist, string, integer, date, and so forth. The bold text refers to the NetSuite API Field name itself. Obviously, you will not need italics or bold in the actual formula when you create it.

An example query string in real-world usage, where a picklist is specified, might be

&cf_*picklist*_**custbody_revenue**= formula_field__c	&cf_*picklist*_**custbody_revenue**= “123”

Where *picklist* refers to the NetSuite Field Type and **custbody_revenue** refers to a custom field on the Sales Order (or any transaction object) called “Revenue”.

For Line Item Custom Fields

NetSuite Transactions (Invoices, Sales Orders, Estimates and Credit Memos) all have line items. While there are several ways to pull data from Salesforce to populate that line item, the usual way to populate that line item in NetSuite is to use a Child Object (i.e. a related list) on the Originating Record in Salesforce.

We designate a custom field which is pulled from a child object and should go onto a line item by starting the line of the Custom URL as

&co_cf

Which is followed by the type of the custom field (in this case a picklist) followed by the API name of the NetSuite field. Here is the theoretical construction.

&co_cf_*type*_**netsuite_api_name**= formula_field__c	&co_cf_*type*_**netsuite_api_name**=”123”

For the sake of readability, we have added italics and bold to different parts of the query string. The italic text refers to the NetSuite field *type* which may be one of: picklist, string, integer, date, and so forth. The bold text refers to the NetSuite API Field name itself. Obviously, you will not need italics or bold in the actual formula when you create it.

An example query string in real-world usage, where a picklist is specified, might be

&co_cf_*picklist*_**custcol_revenue**= formula_field__c	&co_cf_*picklist*_**custcol_revenue**= “123”

Where *picklist* refers to the NetSuite Field Type and **custcol_revenue** refers to a custom field (column) on the Line Item called “Revenue”.

Transaction Fields

These fields modify the “upper half” of the Sales Order, Estimate, or Invoice. By upper half, we mean anything above the line items.

Populating Standard Transaction Fields

Many of these fields are optional to NetSuite, meaning they might not have even been enabled in your NetSuite org. And they are optional in the sense that even if they have been enabled in Salesforce, they might not be required. So you will not necessarily need all of these fields.

Here is an example of populating the PO and Memo field with custom fields from Salesforce.

```
&po=sales_order_po_api_field__c
&memo=sales_order_memo_api_field__c
```

Currency Fields

It is possible to have any combination of currencies, multi-currencies, or single currency, including Salesforce and NetSuite not having the same currency structure. So it is possible to have NetSuite Multi-Currency and Salesforce as Single Currency.

Populating the currency of a Transaction requires being aware of the Currency ID, which can be found in the NetSuite Currency object. You might view a currency as USD, Salesforce might view that currency as USD, and NetSuite will view that currency as the currency with the ID of “1”.

With Breadwinner, you can either pass a three letter text value which we will match to the ISO Code listed in the NetSuite Currency records in Salesforce, or you can directly pass an integer, which Breadwinner will pass to NetSuite directly (that integer will populate whatever currency NetSuite has as associated with that Currency ID).

&currency= sales_order_currency_api_field__c	&currency=”123”

Populating Custom Transaction Fields

(This section does not refer to lookups or picklists which are explained in another section of this document. This section only refers to other data types)

We designate a custom field on a transaction by starting the line of the Custom URL as

*&cf_*

Which is followed by the type of the custom field (in this case a picklist) followed by the API name of the NetSuite field. Here is the theoretical construction.

&cf_*type*_**netsuite_api_name**= formula_field__c	&cf_*type*_**netsuite_api_name**=”123”

NetSuite Field Type	Type Value	Example
Date	date	&cf_***date***_netsuite_api_name=formula_field__c
Datetime	datetime	&cf_***datetime***_netsuite_api_name=formula_field__c
List Record (Lookup)	picklist	&cf_***picklist***_netsuite_api_name=formula_field__c
Integer Number	integer	&cf_**integer_**netsuite_api_name=formula_field__c
Decimal Number	number	&cf_**number_**netsuite_api_name=formula_field__c
Currency	number	&cf_**number_**netsuite_api_name=formula_field__c
Percent	number	&cf_**number_**netsuite_api_name=formula_field__c
Check Box	checkbox	&cf_**checkbox_**netsuite_api_name=formula_field__c
Free Form Text	text	&cf_**text_**netsuite_api_name=formula_field__c
Text Area	textarea	&cf_**textarea_**netsuite_api_name=formula_field__c
Long Text	longtextarea	&cf_**longtextarea_**netsuite_api_name=formula_field__c
Email Address	text	&cf_**text_**netsuite_api_name=formula_field__c
Hyperlink	text	&cf_**text_**netsuite_api_name=formula_field__c

All Fields available for a New Sales Order

The structure of a Custom URL creating a new Sales Order is below, with all possible fields shown. Required fields are in red. All other fields are optional.

/apex/breadwinner_ns__CreateSalesOrder
?originating_record={!*salesforce_record_ID*}
&parent_account=*api_field_name__c*
&custom_form=*sales_order_custom_form_api_field__c*
&currency=*sales_order_currency_api_field__c*
&department=*sales_order_department_api_field__c*
&class=*sales_order_class_api_field__c*
&date=*sales_order_date_api_field__c*
&preceeding_ns_record=*sales_order_created_from_api_field__c*
&order_number=*sales_order_number_api_field__c*
&po=*sales_order_po_api_field__c*
&memo=*sales_order_memo_api_field__c*
&testing=true

Populating Line Items from Child Records

It is possible for Breadwinner to populate all of the Line Items of the Sales Order from custom objects, provided a Custom URL Structure is used.

This requires two things: specifying the child object and specifying the fields on that child object to use.

You may iterate over only one Child Object, i.e. you must pick one and only one Related List to iterate over, not two or more related lists. And, as of now, we do not have the ability to iterate over a subset of those child records; you must iterate over all of them or none of them.

Specifying the Child Object

Please note, *we do not use the API name of the object, we use the Child Relationship Name.* This is because sometimes you will have multiple lookup fields from the child to parent record, and we won't know which one to use. Thus, you must specify the child object by using the Child Relationship Name.

The Child Relationship Name can be tricky to find. In setup, go to the child object, and click on the name of the lookup field (or master/child field) to the parent object. If you have two or more lookup fields to the same parent object, make sure you select the right one. After clicking the name of the lookup field to the parent object, you will be on the setup page of that field.

You will see the Child Relationship Name field. We recommend you append "__r" to that field, as this is best practice, but this is optional because we will append that transparently if it is missing.

Breadwinner for NetSuite
&child_relationship_name=*child_name__r*

Specifying the Child Object Fields

As well as specifying the Child Object, you must specify fields on that Child Object. Most fields are optional, but NetSuite requires an Item in every line item. We have listed the required fields in red.

Also, please note that the field types must match. Item field will need either a direct lookup to NetSuite Item (either via a lookup or master/detail) or a formula field with the NetSuite Item Salesforce Record Id. Quantity and Rate are number fields.

Department

This can be referenced either via another field on the originating object, or as a text field in the formula. So you may use either approaches:

&co_department=dept_api_field__c	&co_department=”123”

Class

This can be referenced either via another field on the originating object, or as a text field in the formula. So you may use either approaches:

&co_class=class_api_field__c	&co_class=”123”

Breadwinner for NetSuite
&co_netsuite_item=*netsuite_item_field__c* **&co_description=*description_field__c*** **&co_quantity=*quantity_field__c*** **&co_rate=*rate_field__c*** **&co_department=*department_field__c*** **&co_class=*class_field__c***

Populating Custom Line Item Fields

(This section does not refer to lookups or picklists which are explained in another section of this document. This section only refers to other data types)

We designate a custom field on a line item by starting the line of the Custom URL as

*&co_cf_*

Which is followed by the type of the custom field (in this case a picklist) followed by the API name of the NetSuite field. Here is the theoretical construction.

&co_cf_*type*_**netsuite_api_name**= formula_field__c	&co_cf_*type*_**netsuite_api_name**=”123”

NetSuite Field Type	Type Value	Example
Date	date	&co_cf_***date***_netsuite_api_name=formula_field__c
Datetime	datetime	&co_cf_***datetime***_netsuite_api_name=formula_field__c
List Record (Lookup)	picklist	&co_cf_***picklist***_netsuite_api_name=formula_field__c
Integer Number	integer	&co_cf_**integer_**netsuite_api_name=formula_field__c
Decimal Number	number	&co_cf_**number_**netsuite_api_name=formula_field__c
Currency	number	&co_cf_**number_**netsuite_api_name=formula_field__c
Percent	number	&co_cf_**number_**netsuite_api_name=formula_field__c
Check Box	checkbox	&co_cf_**checkbox_**netsuite_api_name=formula_field__c
Free Form Text	text	&co_cf_**text_**netsuite_api_name=formula_field__c
Text Area	textarea	&co_cf_**textarea_**netsuite_api_name=formula_field__c
Long Text	longtextarea	&co_cf_**longtextarea_**netsuite_api_name=formula_field__c
Email Address	text	&co_cf_**text_**netsuite_api_name=formula_field__c
Hyperlink	text	&co_cf_**text_**netsuite_api_name=formula_field__c

All Fields available for a New Sales Order with Child Records

The list and structure of a Custom URL with all possible fields, including those for Line Items that are drawn from Child Records, utilized is below. Required fields are in red. All other fields are optional.

Breadwinner for NetSuite
/apex/breadwinner_ns__CreateSalesOrder **?originating_record={!*salesforce_record_ID*}** **&parent_account=*api_field_name__c*** **&custom_form=*sales_order_custom_form_api_field__c*** **&currency=*sales_order_currency_api_field__c*** **&department=*sales_order_department_api_field__c*** **&class=*sales_order_class_api_field__c*** **&location_id=*sales_order_location_api_field__c*** **&date=*sales_order_date_api_field__c*** **&preceeding_ns_record=*sales_order_created_from_api_field__c*** **&order_number=*sales_order_number_api_field__c*** **&po=*sales_order_po_api_field__c*** **&memo=*sales_order_memo_api_field__c*** **&sales_rep=*sales_order_sales_rep_api_field__c*** ***&bill_address_list=sales_order_bill_address_list__c*** ***&ship_address_list=sales_order_ship_address_list__c*** **&*billing*_addressee=*sales_order_billing*_addressee__c** ***&billing_attention=sales_order_billing_attention__c*** **&*billing*_addressee=*sales_order_billing*_addressee__c** **&*billing*_addr1=*sales_order_billing*_addr1__c** **&*billing*_addr2=*sales_order_billing*_addr2__c** **&*billing*_city=*sales_order_billing*_city__c** **&*billing*_state=*sales_order_billing*_state__c** **&*billing*_country=*sales_order_billing*_country__c** **&*billing*_zip=*sales_order_billing*_zip__c** ***&shipping_attention=sales_order_shipping_attention__c*** **&*shipping*_addressee=*sales_order_shipping*_addressee__c** **&*shipping*_addr1=*sales_order_shipping*_addr1__c** **&*shipping*_addr2=*sales_order_shipping*_addr2__c** **&*shipping*_city=*sales_order_shipping*_city__c** **&*shipping*_state=*sales_order_shipping*_state__c** **&*shipping*_country=*sales_order_shipping*_country__c**

&*shipping*_zip=*sales_order_shipping*_zip__c
&child_relationship_name=*child_name__r*
&co_netsuite_item=*netsuite_item_field__c*
&co_description=*description_field__c*
&co_quantity=*quantity_field__c*
&co_rate=*rate_field__c*
&co_department=*department_field__c*
&co_class=*class_field__c*
&co_location_id=*location_field__c*
&co_rev_rec_schedule=*rev_rec_schedule_field__c*
&co_rev_rec_start_date=*rev_rec_start_date_field__c*
&co_rev_rec_end_date=*rev_rec_end_date_field__c*
&testing=true

All Fields available for a New Invoice with Child Records

The list and structure of a Custom URL with all possible fields, including those for Line Items that are drawn from Child Records, utilized is below. Required fields are in red. All other fields are optional.

Breadwinner for NetSuite
/apex/breadwinner_ns__CreateInvoice **?originating_record={!*salesforce_record_ID*}** **&parent_account=*api_field_name__c*** **&custom_form=*invoice_custom_form_api_field__c*** **&currency=*invoice_currency_api_field__c*** **&department=*invoice_department_api_field__c*** **&class=*invoice_class_api_field__c*** **&location_id=*invoice_location_api_field__c*** **&date=*invoice_date_api_field__c*** ***&due_date=invoice_due_date_api_field__c*** **&po=*invoice_po_api_field__c*** **&memo=*invoice_memo_api_field__c*** **&sales_rep=*invoice_sales_rep_api_field__c*** **&child_relationship_name=*child_name__r*** **&co_netsuite_item=*netsuite_item_field__c*** **&co_description=*description_field__c*** **&co_quantity=*quantity_field__c*** **&co_rate=*rate_field__c*** **&co_department=*department_field__c*** **&co_class=*class_field__c*** **&co_location_id=*location_field__c*** **&co_rev_rec_schedule=*rev_rec_schedule_field__c*** **&co_rev_rec_start_date=*rev_rec_start_date_field__c*** **&co_rev_rec_end_date=*rev_rec_end_date_field__c*** **&testing=true**

All Fields available for a New Credit Memo with Child Records

The list and structure of a Custom URL with all possible fields, including those for Line Items that are drawn from Child Records, utilized is below. Required fields are in red. All other fields are optional.

Breadwinner for NetSuite
/apex/breadwinner_ns__CreateCreditMemo **?originating_record={!*salesforce_record_ID*}** **&parent_account=*api_field_name__c*** **&custom_form=*credit_custom_form_api_field__c*** **&currency=*credit_currency_api_field__c*** **&department=*credit_department_api_field__c*** **&class=*credit_class_api_field__c*** **&location_id=*invoice_location_api_field__c*** **&date=*credit_date_api_field__c*** **&po=*credit_po_api_field__c*** **&memo=*credit_memo_api_field__c*** **&sales_rep=*credit_sales_rep_api_field__c*** **&child_relationship_name=*child_name__r*** **&co_netsuite_item=*netsuite_item_field__c*** **&co_description=*description_field__c*** **&co_quantity=*quantity_field__c*** **&co_rate=*rate_field__c*** **&co_department=*department_field__c*** **&co_class=*class_field__c*** **&co_location_id=*location_field__c*** **&co_rev_rec_schedule=*rev_rec_schedule_field__c*** **&co_rev_rec_start_date=*rev_rec_start_date_field__c*** **&co_rev_rec_end_date=*rev_rec_end_date_field__c*** **&testing=true**

All Fields available for a New Return Authorization (RMA) with Child Records

The list and structure of a Custom URL with all possible fields, including those for Line Items that are drawn from Child Records, utilized is below. Required fields are in red. All other fields are optional.

Breadwinner for NetSuite
/apex/breadwinner_ns__CreateReturnAuthorization **?originating_record={!*salesforce_record_ID*}** **&parent_account=*api_field_name__c*** **&custom_form=*rma_custom_form_api_field__c*** **&currency=*rma_currency_api_field__c*** **&department=*rma_department_api_field__c*** **&class=*rma_class_api_field__c*** **&location_id=*rma_location_api_field__c*** **&date=*rma_date_api_field__c*** **&po=*rma_po_api_field__c*** **&memo=*rma_memo_api_field__c*** **&sales_rep=*rma_sales_rep_api_field__c*** **&child_relationship_name=*child_name__r*** **&co_netsuite_item=*netsuite_item_field__c*** **&co_description=*description_field__c*** **&co_quantity=*quantity_field__c*** **&co_rate=*rate_field__c*** **&co_department=*department_field__c*** **&co_class=*class_field__c*** **&co_location_id=*location_field__c*** **&co_rev_rec_schedule=*rev_rec_schedule_field__c*** **&co_rev_rec_start_date=*rev_rec_start_date_field__c*** **&co_rev_rec_end_date=*rev_rec_end_date_field__c*** **&testing=true**

All Fields available for a New Purchase Order with Child Records

The list and structure of a Custom URL with all possible fields, including those for Line Items that are drawn from Child Records, utilized is below. Required fields are in red. All other fields are optional.

Breadwinner for NetSuite
/apex/breadwinner_ns__CreatePurchaseOrder **?originating_record={!*salesforce_record_ID*}** **&parent_account=*api_field_name__c*** **&custom_form=*purchase_order_custom_form_api_field__c*** **&currency=*purchase_order_currency_api_field__c*** **&department=*purchase_order_department_api_field__c*** **&class=*purchase_order_class_api_field__c*** **&location_id=*purchase_order_location_api_field__c*** **&date=*purchase_order_date_api_field__c*** **&due_date=*purchase_order_due_date_api_field__c*** **&other_ref_num=*purchase_order__other_ref_api_field__c*** **&memo=*purchase_order__memo_api_field__c*** **&child_relationship_name=*child_name__r*** **&co_netsuite_item=*netsuite_item_field__c*** **&co_description=*description_field__c*** **&co_quantity=*quantity_field__c*** **&co_rate=*rate_field__c*** **&co_department=*department_field__c*** **&co_class=*class_field__c*** **&co_location_id=*location_field__c*** **&testing=true**

All Fields available for a New Bill with Child Records

The list and structure of a Custom URL with all possible fields, including those for Line Items that are drawn from Child Records, utilized is below. Required fields are in red. All other fields are optional.

Breadwinner for NetSuite
/apex/breadwinner_ns__CreateBill **?originating_record={!*salesforce_record_ID*}** **&parent_account=*api_field_name__c*** **&custom_form=*bill_custom_form_api_field__c*** **&currency=*bill_currency_api_field__c*** **&department=*bill_department_api_field__c*** **&class=*bill_class_api_field__c*** **&location_id=*bill_location_api_field__c*** **&date=*bill_date_api_field__c*** **&due_date=*bill_due_date_api_field__c*** **&memo=*sales_order_memo_api_field__c*** **&child_relationship_name=*child_name__r*** **&co_netsuite_item=*netsuite_item_field__c*** **&co_description=*description_field__c*** **&co_quantity=*quantity_field__c*** **&co_rate=*rate_field__c*** **&co_department=*department_field__c*** **&co_class=*class_field__c*** **&co_location_id=*location_field__c*** **&testing=true**

Populating Line Items from Originating Record Fields

You may have a number of fields on the Originating Record that you wish to populate various line items on the Sales Order/Purchase Order/Bill. This is possible, as long as you specify each and every individual line. This is different from populating child records, where we iterate over each child record. Here, you specify the number of lines in each URL.

The Sales Order/Purchase Order/Bill URL will start the same as always, with the structure taken from the basic Custom URL structure for Sales Order/Purchase Order/Bill fields, where you can specify dates, etc.

However, the URL will then specify line items, using the structure &line#_field=*your_field__c*

Breadwinner for NetSuite
&line1_netsuite_item=*netsuite_item_field__c* **&line1_description=*description_field__c*** **&line1_quantity=*quantity_field__c*** **&line1_rate=*rate_field__c*** **&line1_department=*department_field__c*** **&line1_class=*class_field__c*** **&line1_location_id=*location_field__c*** **&testing=true**

Internet Explorer, depending on the version, may have a limit of 2000 characters in the URL. Firefox and Chrome have potentially much longer URL limits, of 30,000 characters and up. Salesforce may have it's own limits. We cannot be responsible for either your browser's URL character limitation or Salesforce's URL character limitation. As such, we recommend this approach for just a few lines, however it may work for dozens of lines or more.

Specifying more than one line

Should you want more than one line, you simply need to start with &line2… and so forth.

Breadwinner for NetSuite
&line1_netsuite_item=*netsuite_item_field__c*
&line1_quantity=*quantity_field__c*
&line1_rate=*rate_field__c*
&line2_netsuite_item=*netsuite_item_field__c*
&line2_quantity=*quantity_field__c*
&line2_rate=*rate_field__c*
&line3_netsuite_item=*netsuite_item_field__c*
&line3_quantity=*quantity_field__c*
&line3_rate=*rate_field__c*

NetSuite Company Creation

The Custom URL requires a Salesforce Account to be associated with the creation process. Breadwinner will iterate over all the NetSuite Companies associated with that Salesforce Account.

If there is no associated NetSuite Company, you will be required to create a new NetSuite Company. And, even if a NetSuite Company is associated, you will be given the option to create another NetSuite Company.

We will prepopulate fields onto the NetSuite Company creation page, though you will be able to edit it before creation.

Specific Fields

If you have enabled Multi-Currency in both Salesforce and in NetSuite, we will prepopulate the Currency field on the NetSuite Company.

You may also have NetSuite Customer fields such as the Territory field. You can map custom fields in Salesforce to NetSuite by configuring this in the Breadwinner tab, under the Configuration > Company.

Provisional Features

These features are available in our Release Candidate package and can be tested and configured in your Salesforce Sandbox. Talk to your Breadwinner AE for access to this testing / release candidate environment.