should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Master Agreement, Oracle License and Services Agreement, Oracle PartnerNetwork Agreement, Oracle distribution agreement, or other license agreement which has been executed by you and Oracle and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Sample Code

Oracle may provide sample code in SuiteAnswers, the Help Center, User Guides, or elsewhere through help links. All such sample code is provided “as is” and “as available”, for use only with an authorized NetSuite Service account, and is made available as a SuiteCloud Technology subject to the SuiteCloud Terms of Service at www.netsuite.com/tos.

Oracle may modify or remove sample code at any time without notice.

No Excessive Use of the Service

As the Service is a multi-tenant service offering on shared databases, Customer may not use the Service in excess of limits or thresholds that Oracle considers commercially reasonable for the Service. If Oracle reasonably concludes that a Customer’s use is excessive and/or will cause immediate or ongoing performance issues for one or more of Oracle’s other customers, Oracle may slow down or throttle Customer’s excess use until such time that Customer’s use stays within reasonable limits. If Customer’s particular usage pattern requires a higher limit or threshold, then the Customer should procure a subscription to the Service that accommodates a higher limit and/or threshold that more effectively aligns with the Customer’s actual usage pattern.

Beta Features

Oracle may make available to Customer certain features that are labeled “beta” that are not yet generally available. To use such features, Customer acknowledges and agrees that such beta features are subject to the terms and conditions accepted by Customer upon activation of the feature, or in the absence of such terms, subject to the limitations for the feature described in the User Guide and as follows: The beta feature is a prototype or beta version only and is not error or bug free and Customer agrees that it will use the beta feature carefully and will not use it in any way which might result in any loss, corruption or unauthorized access of or to its or any third party’s property or information. Customer must promptly report to Oracle any defects, errors or other problems in beta features to support@netsuite.com or other designated contact for the specific beta feature. Oracle cannot guarantee the continued availability of such beta features and may substantially modify or cease providing such beta features without entitling Customer to any refund, credit, or other compensation. Oracle makes no representations or warranties regarding functionality or use of beta features and Oracle shall have no liability for any lost data, incomplete data, re-run time, inaccurate input, work delay, lost profits or adverse effect on the performance of the Service resulting from the use of beta features. Oracle’s standard service levels, warranties and related commitments regarding the Service shall not apply to beta features and they may not be fully supported by Oracle’s customer support. These limitations and exclusions shall apply until the date that Oracle at its sole option makes a beta feature generally available to its customers and partners as part of the Service without a “beta” label.
Send Us Your Feedback

We'd like to hear your feedback on this document. Answering the following questions will help us improve our help content:

- Did you find the information you needed? If not, what was missing?
- Did you find any errors?
- Is the information clear?
- Are the examples correct?
- Do you need more examples?
- What did you like most about this document?

Click here to send us your comments. If possible, please provide a page number or section title to identify the content you're describing.

To report software issues, contact NetSuite Customer Support.
# Table of Contents

**SuiteBuilder (Customization)** ..................................................................................................................... 1  
**SuiteBuilder Overview** ............................................................................................................................... 2  
**Forms** .......................................................................................................................................................... 3  
**Record Types** ............................................................................................................................................. 4  
**Transaction Types** ...................................................................................................................................... 9  
**Segments** .................................................................................................................................................... 10  
**Advanced Templates** .................................................................................................................................. 10  
**Centers** ...................................................................................................................................................... 12  
**Customizing Field Level Help for Standard Fields** .................................................................................. 14  
**Custom Fields** ........................................................................................................................................... 17  
**Table of Custom Field Type Descriptions** ............................................................................................... 17  
**Available Standard Fields and Field Types** ............................................................................................. 24  
**Kinds of Custom Fields** .............................................................................................................................. 31  
**Custom CRM Fields** ................................................................................................................................. 32  
**Custom Entity Fields** ............................................................................................................................... 33  
**Custom Item Fields** .................................................................................................................................. 35  
**Custom Transaction Body Fields** .............................................................................................................. 36  
**Custom Transaction Line Fields** ............................................................................................................. 39  
**Custom Transaction Item Options** ........................................................................................................... 41  
**Custom Item Number Fields** .................................................................................................................... 43  
**Other Record Fields** ................................................................................................................................... 44  
**Other Sublist Fields** ................................................................................................................................... 47  
**Creating a Custom Field** ................................................................................................................................ 48  
**Assigning Custom Fields to Specific Record Types** .................................................................................. 51  
**Setting Display Options for Custom Fields** .............................................................................................. 52  
**Setting Validation and Defaulting Properties** ............................................................................................ 54  
**Setting Sourcing Criteria** ............................................................................................................................ 58  
**Sourcing and Filtering Examples** ................................................................................................................ 61  
**Setting Filtering Criteria** ............................................................................................................................. 64  
**Multiple Dependent Dropdown Lists** ......................................................................................................... 65  
**Restricting Access to Custom Fields** ......................................................................................................... 68  
**Restricting Access to Employee Custom Fields** ...................................................................................... 70  
**Creating Read-Only Custom Fields** .......................................................................................................... 70  
**Adding Translations for Custom Fields** ..................................................................................................... 71  
**Adding Custom Fields to Transaction Forms** ........................................................................................... 72  
**Tracking Changes to Custom Fields** ......................................................................................................... 73  
**Inactivating a Custom Field** ....................................................................................................................... 73  
**Editing a Custom Field** ............................................................................................................................... 74  
**Renaming Custom Fields** ........................................................................................................................... 76  
**Mass Updating Custom Fields** .................................................................................................................. 77  
**Advanced Features for Custom Fields** ...................................................................................................... 77  
**Encrypted Custom Field Stored Values** ..................................................................................................... 77  
**Creating Custom Fields with Values Derived from Summary Search Results** ..................................... 78  
**Dynamic Defaults and Dynamic Hyperlinks** ........................................................................................... 80  
**Creating Formula Fields** ........................................................................................................................... 82  
**Custom Lists** ............................................................................................................................................. 87  
**Creating a Custom List** ............................................................................................................................... 87  
**Adding Translations for Custom Lists** ....................................................................................................... 88  
**Managing Large Custom Lists** .................................................................................................................. 90  
**Custom Forms** .......................................................................................................................................... 92  
**Creating Custom Entry and Transaction Forms** ....................................................................................... 94  
**Storing Custom Forms with Transactions** ............................................................................................... 97  
**Configuring Subtabs for Custom Entry and Transaction Forms** ............................................................... 99
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced PDF/HTML Templates</td>
<td>154</td>
</tr>
<tr>
<td>Scripting with Advanced Templates</td>
<td>204</td>
</tr>
<tr>
<td>WYSIWYG Editing in the Template Editor</td>
<td>163</td>
</tr>
<tr>
<td>Setting Custom Forms to Use Advanced Templates</td>
<td>157</td>
</tr>
<tr>
<td>Reviewing Available Advanced Templates</td>
<td>155</td>
</tr>
<tr>
<td>Custom Child Record Sublist Creation with SuiteScript</td>
<td>140</td>
</tr>
<tr>
<td>Applying Custom Sublists to Standard Records</td>
<td>134</td>
</tr>
<tr>
<td>Applying Custom Sublists in SDF-Enabled Accounts</td>
<td>137</td>
</tr>
<tr>
<td>Applying Custom Sublists to Custom Record Types</td>
<td>138</td>
</tr>
<tr>
<td>Custom Child Record Sublist Creation with SuiteScript</td>
<td>140</td>
</tr>
<tr>
<td>Customizing a Transaction Sublist</td>
<td>151</td>
</tr>
<tr>
<td>Custom Sublists</td>
<td>133</td>
</tr>
<tr>
<td>Saved Searches for Custom Sublists</td>
<td>133</td>
</tr>
<tr>
<td>Configuring Field Groups</td>
<td>100</td>
</tr>
<tr>
<td>Configuring Fields or Screens</td>
<td>102</td>
</tr>
<tr>
<td>Configuring Buttons and Actions</td>
<td>106</td>
</tr>
<tr>
<td>Configuring Printing Fields</td>
<td>108</td>
</tr>
<tr>
<td>Configuring Sublists</td>
<td>110</td>
</tr>
<tr>
<td>Configuring Sublist Fields</td>
<td>110</td>
</tr>
<tr>
<td>Configuring QuickViews</td>
<td>111</td>
</tr>
<tr>
<td>Advanced PDF/HTML Template for Multiple Currencies Customers</td>
<td>157</td>
</tr>
<tr>
<td>Setting Custom Forms to Use Advanced Templates</td>
<td>157</td>
</tr>
<tr>
<td>Customizing Advanced Templates in the Template Editor</td>
<td>159</td>
</tr>
<tr>
<td>Training Videos for Advanced PDF/HTML Templates</td>
<td>163</td>
</tr>
<tr>
<td>Customizing Multiple Page Transaction Forms</td>
<td>118</td>
</tr>
<tr>
<td>Adding Table Breaks to Tables</td>
<td>195</td>
</tr>
<tr>
<td>Adding Striping to Line Items in Advanced Templates</td>
<td>191</td>
</tr>
<tr>
<td>Source Code Editing in the Template Editor</td>
<td>163</td>
</tr>
<tr>
<td>Template Editor Toolbar</td>
<td>165</td>
</tr>
<tr>
<td>Adding and Removing Fields in Advanced Templates</td>
<td>166</td>
</tr>
<tr>
<td>Adding and Formatting Text in Advanced Templates</td>
<td>169</td>
</tr>
<tr>
<td>Including Images in Advanced Templates</td>
<td>172</td>
</tr>
<tr>
<td>Configuring Field Groups</td>
<td>100</td>
</tr>
<tr>
<td>Configuring Field Groups</td>
<td>100</td>
</tr>
<tr>
<td>Configuring Printing Fields</td>
<td>108</td>
</tr>
<tr>
<td>Configuring Sublists</td>
<td>110</td>
</tr>
<tr>
<td>Configuring Sublist Fields</td>
<td>110</td>
</tr>
<tr>
<td>Configuring QuickViews</td>
<td>111</td>
</tr>
<tr>
<td>Advanced PDF/HTML Template</td>
<td>154</td>
</tr>
<tr>
<td>Enabling the Advanced PDF/HTML Templates Feature</td>
<td>155</td>
</tr>
<tr>
<td>Reviewing Available Advanced Templates</td>
<td>155</td>
</tr>
<tr>
<td>Advanced PDF/HTML Template for Multiple Currencies Customers</td>
<td>157</td>
</tr>
<tr>
<td>Setting Custom Forms to Use Advanced Templates</td>
<td>157</td>
</tr>
<tr>
<td>Customizing Advanced Templates in the Template Editor</td>
<td>159</td>
</tr>
<tr>
<td>Training Videos for Advanced PDF/HTML Templates</td>
<td>163</td>
</tr>
<tr>
<td>Linking Transaction Forms</td>
<td>119</td>
</tr>
<tr>
<td>Customizing Multiple Page Transaction Forms</td>
<td>118</td>
</tr>
<tr>
<td>Specifying Check Layout by Subsidiary</td>
<td>118</td>
</tr>
<tr>
<td>Adding Disclaimers to Transaction Form Footers</td>
<td>117</td>
</tr>
<tr>
<td>Configuring QuickViews</td>
<td>111</td>
</tr>
<tr>
<td>Configuring Sublist Fields</td>
<td>110</td>
</tr>
<tr>
<td>Configuring Printing Fields</td>
<td>108</td>
</tr>
<tr>
<td>Configuring Field Groups</td>
<td>100</td>
</tr>
<tr>
<td>Configuring Field Groups</td>
<td>100</td>
</tr>
<tr>
<td>Configuring Printing Fields</td>
<td>108</td>
</tr>
<tr>
<td>Configuring Sublists</td>
<td>110</td>
</tr>
<tr>
<td>Configuring Sublist Fields</td>
<td>110</td>
</tr>
<tr>
<td>Configuring QuickViews</td>
<td>111</td>
</tr>
<tr>
<td>Advanced PDF/HTML Templates</td>
<td>154</td>
</tr>
<tr>
<td>Enabling the Advanced PDF/HTML Templates Feature</td>
<td>155</td>
</tr>
<tr>
<td>Reviewing Available Advanced Templates</td>
<td>155</td>
</tr>
<tr>
<td>Advanced PDF/HTML Template for Multiple Currencies Customers</td>
<td>157</td>
</tr>
<tr>
<td>Setting Custom Forms to Use Advanced Templates</td>
<td>157</td>
</tr>
<tr>
<td>Customizing Advanced Templates in the Template Editor</td>
<td>159</td>
</tr>
<tr>
<td>Training Videos for Advanced PDF/HTML Templates</td>
<td>163</td>
</tr>
<tr>
<td>Linking Transaction Forms</td>
<td>119</td>
</tr>
<tr>
<td>Customizing Multiple Page Transaction Forms</td>
<td>118</td>
</tr>
<tr>
<td>Specifying Check Layout by Subsidiary</td>
<td>118</td>
</tr>
<tr>
<td>Adding Disclaimers to Transaction Form Footers</td>
<td>117</td>
</tr>
<tr>
<td>Configuring QuickViews</td>
<td>111</td>
</tr>
<tr>
<td>Configuring Sublist Fields</td>
<td>110</td>
</tr>
<tr>
<td>Configuring Printing Fields</td>
<td>108</td>
</tr>
<tr>
<td>Configuring Field Groups</td>
<td>100</td>
</tr>
<tr>
<td>Advanced PDF/HTML Template</td>
<td>154</td>
</tr>
<tr>
<td>Enabling the Advanced PDF/HTML Templates Feature</td>
<td>155</td>
</tr>
<tr>
<td>Reviewing Available Advanced Templates</td>
<td>155</td>
</tr>
<tr>
<td>Advanced PDF/HTML Template for Multiple Currencies Customers</td>
<td>157</td>
</tr>
<tr>
<td>Setting Custom Forms to Use Advanced Templates</td>
<td>157</td>
</tr>
<tr>
<td>Customizing Advanced Templates in the Template Editor</td>
<td>159</td>
</tr>
<tr>
<td>Training Videos for Advanced PDF/HTML Templates</td>
<td>163</td>
</tr>
<tr>
<td>Linking Transaction Forms</td>
<td>119</td>
</tr>
<tr>
<td>Customizing Multiple Page Transaction Forms</td>
<td>118</td>
</tr>
<tr>
<td>Specifying Check Layout by Subsidiary</td>
<td>118</td>
</tr>
<tr>
<td>Adding Disclaimers to Transaction Form Footers</td>
<td>117</td>
</tr>
<tr>
<td>Configuring QuickViews</td>
<td>111</td>
</tr>
<tr>
<td>Configuring Sublist Fields</td>
<td>110</td>
</tr>
<tr>
<td>Configuring Printing Fields</td>
<td>108</td>
</tr>
<tr>
<td>Configuring Field Groups</td>
<td>100</td>
</tr>
<tr>
<td>Advanced PDF/HTML Template</td>
<td>154</td>
</tr>
<tr>
<td>Enabling the Advanced PDF/HTML Templates Feature</td>
<td>155</td>
</tr>
<tr>
<td>Reviewing Available Advanced Templates</td>
<td>155</td>
</tr>
<tr>
<td>Advanced PDF/HTML Template for Multiple Currencies Customers</td>
<td>157</td>
</tr>
<tr>
<td>Setting Custom Forms to Use Advanced Templates</td>
<td>157</td>
</tr>
<tr>
<td>Customizing Advanced Templates in the Template Editor</td>
<td>159</td>
</tr>
<tr>
<td>Training Videos for Advanced PDF/HTML Templates</td>
<td>163</td>
</tr>
<tr>
<td>Linking Transaction Forms</td>
<td>119</td>
</tr>
<tr>
<td>Customizing Multiple Page Transaction Forms</td>
<td>118</td>
</tr>
<tr>
<td>Specifying Check Layout by Subsidiary</td>
<td>118</td>
</tr>
<tr>
<td>Adding Disclaimers to Transaction Form Footers</td>
<td>117</td>
</tr>
<tr>
<td>Configuring QuickViews</td>
<td>111</td>
</tr>
<tr>
<td>Configuring Sublist Fields</td>
<td>110</td>
</tr>
<tr>
<td>Configuring Printing Fields</td>
<td>108</td>
</tr>
<tr>
<td>Configuring Field Groups</td>
<td>100</td>
</tr>
</tbody>
</table>
SuiteBuilder (Customization)

- SuiteBuilder Overview
- Custom Fields
- Custom Forms
- Advanced PDF/HTML Templates
- SuiteBuilder Advanced Templates Reference
- Basic Printing Layouts
- Custom Records
- Custom Transactions
- Custom Segments
- Custom Centers
- Deploying Upgraded Forms
SuiteBuilder Overview

With the SuiteBuilder customization tools, you can tailor NetSuite to your individual business needs and processes. SuiteBuilder provides a point-and-click interface for creating fields, forms, record types, transaction types, form layouts, segments, and centers. SuiteBuilder also lets you define how information is accessed and entered by each user of your NetSuite account.

Who Uses SuiteBuilder?

There are three types of SuiteBuilder users:

- **Administrators** - Administrators spend time customizing transaction forms, adding custom record types, and setting up custom centers for roles within the company. Some of the customization work is unique to a specific business. Administrators also assign roles to each NetSuite user. These roles determine which information each user has access to in your NetSuite account.

- **IT Staff** - Issues addressed by the IT department often include requests for changes to the NetSuite account. These issues can range from small tasks such as adding a field to a form, to larger work items like creating custom record types. IT staff members may have access to administrative tools that enable them to manage data in the system, and schedule batch processing jobs.

- **Developers** - For developers of partner solutions and independent software vendors (ISVs), most time with NetSuite is spent coding SuiteScript and SOAP web services. To use these features, developers require a solid understanding of how customization objects interact with their code.

Customizing NetSuite Components

Regardless of your experience with other software applications, using SuiteBuilder can quickly help you set up NetSuite. Use SuiteBuilder to customize the components that control how your users interact with NetSuite.

Access Information

To define how users interact with NetSuite and what data they have access to, you can configure these components.

- **Role** – Set of permissions that can be assigned to a NetSuite user.

- **Center** – Configuration of NetSuite created for a specific group of roles with similar tasks.

- **Center tab** – Section of NetSuite that groups similar links and other information. Standard tabs include Home, Reports, Documents, Activities, and Setup. You can also create custom center tabs.

Set Up Data

To define how to get the most out of your NetSuite implementation, use these components to configure your data.
Collect and Display Data

To collect and display data within NetSuite, you must use forms. A form is a page through which you enter records and transactions. Use these components to build your own forms or customize an existing form.

- **Field** – Place on a record or transaction where information is entered.
- **Subtab** – Section of a record or transaction that groups similar fields.
  
  An example of a standard subtab is the Address subtab where the shipping and billing addresses are entered on transactions and records.
- **Sublist** – The results of a saved search displayed on a custom or standard record. Sublists can also be generated through parent-child relationships.
- **Script** – SuiteScript JavaScript file that runs against a specific form or record type or that creates a custom portlet. Scripts can also be scheduled to execute periodically. For more information about using SuiteScript, see SuiteScript Overview.

Customize Your NetSuite Account

Now that you know what components can be customized with SuiteBuilder, see the following topics for more information:

- Forms
- Record Types
- Transaction Types
- Segments
- Advanced Templates
- Centers

Forms

All information in NetSuite is entered through forms and NetSuite provides a full array of standard forms that you can use. There are forms for every record you can store in your NetSuite account, including transactions, relationships, and CRM activities. For a complete list of supported record types, see the help topic NetSuite Record Types.

With SuiteBuilder, you can customize standard forms to ensure that you are capturing the information that your business needs.
Forms are composed of the following components:

1. **Fields** - Fields are used to display and enter data. If you need information that is not included on a form, you can create your own custom fields and add them to the form. When the form is filled out and submitted, the information in the custom field is stored in the same way as any standard field. For more information, see Custom Fields.

2. **Subtabs** - Most forms are divided into subtabs. Subtabs group fields with similar information in one place. For example, the Schedule subtab contains project tasks and milestones. You can create custom subtabs to organize custom fields on records. For more information, see Creating Custom Subtabs.

3. **Sublists** - You can also add custom sublists to your forms. Sublists display saved search results relating to the record on which they are shown. For example, you might add a custom sublist to track specific milestone details. For more information, see Custom Sublists.

For more information, see Custom Forms.

**Record Types**

You can create custom record types to collect and store information that is not included in NetSuite. For example, your company might need to track information about computer and electronic equipment. If there is no such standard record type in NetSuite, you could create a custom record type called Equipment and add custom fields to collect and store the equipment information needed. For more information, see Custom Forms.
Custom records can be attached to standard records and other custom records. These child record types can be used to track specific information that requires multiple fields on a record. For more information, see Parent-Child Record Relationships.

Similar to standard transactions and records, you can customize the forms used to enter custom records and set up forms for different roles. Customizing a form involves determining which fields appear, how these fields are arranged, and which roles use the form.

For example, your company might need to track information on computer and electronic equipment. Since there is no such standard record type in NetSuite, you could create a custom record type called Equipment and add custom fields in which you enter serial numbers, location, purchase date, and service and warranty information.
Your custom records can be attached to standard records and other custom records. For example, on the preceding equipment record type, you might want to track details each time you service your equipment. You create a new custom record, Equipment Service, used to track these details.

To create the parent-child relationship, you add a field to the Equipment Service record in which you can select the Equipment record it is associated with. The field type is List/Record, and Equipment is chosen in the List/Record field. Checking the Record is Parent box creates the parent-child relationship.

When service is performed, the technician opens the Equipment record, clicks the Service subtab, and clicks a button to create the Equipment Service record.

1. Click the New Equipment Service button to create a new service record.
2. Enter the service information in the new record. The Equipment field is filled in automatically.
3. The child service record shows on the parent Equipment record.
In the same way as with standard transactions and records, you can customize the forms used to enter custom records and even set up forms for different roles. Customizing a form involves determining which fields show, how these fields are arranged, and which roles use the form.

The equipment record in the preceding example is most useful to the IT staff who have to purchase, maintain, and track the equipment, but there is some information that is generally useful to others in the company. A custom form is created for the record.

1. On the Custom Record Type page on the Forms subtab, click Customize next to the standard form.
2. On the Custom Entry Form page, clear the box in the Show column next to the Warranty and Service subtabs.
3. Make sure that Use Permissions List is selected for Access Type, and on the Permissions subtab, restrict your non-IT roles to use only the new custom equipment form.
4. When someone with a non-IT role views the equipment record, the hidden fields and subtabs are not shown.
With the preceding custom record form, employees outside of your IT department can view basic information about the equipment, like manufacturer and model. They cannot see information related to service and warranty that is only relevant to your IT staff.

For more information, see Custom Records and Adding Custom Forms for a Record.

Transaction Types

You can create a custom transaction when you need to capture an event in your general ledger that is specific to your business. With the custom transactions feature, you can create custom transaction types that are configured to reflect your specific needs and business logic. For example, you can create
a custom transaction to record adjustments for non-operational income, such as investment income or currency exchanges.

Custom transactions types use the same features available with standard NetSuite transactions types, such as purchases, payables, sales, billing, customers, and so on. Support for these features makes it possible for each transaction type to have its own unique behavior and processing. For example, you can restrict access to a limited set of users, include custom transactions in bundles, and reference them when you create workflows.

For more information about custom transactions, see Custom Transactions.

Segments

Classifications are used to identify and categorize records in NetSuite and enable you to better organize and manage your data. Using custom segments lets you create custom classifications similar to the class, department, and location default classifications provided by NetSuite. Custom segments can be used as search filters and columns for NetSuite reports. A custom segment can be displayed on the GL Impact page if it is configured to have GL impact.

For example, your company might need to classify a record by its profit center. You could create a custom segment called Profit Center and use it to sort and search for your data.

For more information about custom segments, see Custom Segments.

Advanced Templates

Use the Advanced PDF/HTML Templates feature to customize printed and emailed forms, records, and saved searches. Standard templates are provided for each supported print type but can customized with a custom template. You can use the Template Editor to customize the existing standard templates.
For more information about standard templates, see Advanced PDF/HTML Templates.

You can use the Template Editor to build a custom print template using the WYSIWYG mode, or by editing the source code directly. In WYSIWYG mode, click on the New Element toolbar to add HTML-based elements such as fields, images and tables, or printing elements.
Centers

When employees are assigned roles in NetSuite, they are granted access to the NetSuite pages necessary to complete their work. Each standard role has access to what is called a center. A center is a configuration of NetSuite created for a group of roles with similar tasks. For example, all sales roles – sales representatives, sales managers, and sales administrators – use the Sales Center by default. Although the information available to each role differs, the basic layout of the Sales Center is the same for each of the standard sales roles. Center tabs are the headings at the top of the page.

For more information about using the template editor, see WYSIWYG Editing in the Template Editor.
With SuiteBuilder, you can create custom centers and assign them to custom roles, letting you completely control a role’s experience with NetSuite. You can create custom tabs, add portlets, and add links to the tasks needed by the role. You can only grant custom center access to new custom roles that you create. You cannot give custom center access to standard roles or to customized version of standard roles.

For more information about creating custom centers, see Custom Centers.
Customizing Field Level Help for Standard Fields

View a video about customizing field level help for standard fields.

You can customize the field level help for standard NetSuite fields and enter field level help for other languages, if the Multi-Language feature is enabled. Administrators or users with the full level of the Customize Field Level Help permission can edit or remove the custom help for a field. The view, none, and edit levels of the Customize Field Level Help permission are not used.

To customize the help for a field:

1. View the field level help for the field and then click the Customize button.

2. In the Help Text field, enter the default custom help text to use for the field in the current language.

3. In the Translations section, select a language and enter the help text for the field. Enter custom help for as many languages as required.
   
   If the Multi-Language feature is not enabled, you can customize the help text for English only.

4. Check the Make as Default for Other Forms of the Same Type box to use this custom field level help for any forms derived from this form type.
5. Click Save.

You can enter HTML markup source for the custom field level help. The following HTML tags are supported in upper case or lower case:

- `<p>`
- `<ul>`
- `<ol>`
- `<li>`
- `<i>`
- `<b>`
- `<br>`
- `<a>`

For the `<a>` tag, the following attributes are supported:

- `href` – Mandatory attribute. Only absolute paths are allowed, that is, URLs starting with http or https. In addition, JavaScript is not supported.
- `target` – Mandatory attribute. Only _blank and custom name values are supported. If the target attribute is missing, it is automatically added with the _blank attribute so that clicking a link does not open the link in the field level help dialog.
- `title`
- `download`
- `hreflang`
- `rel`
- `media`

For more information about the `<a>` tag, see the HTML `<a>` Tag page on the w3schools website.

**Note:** Because HTML tags are supported, use `&lt;` and `&gt;` to display the `< less than and > greater than symbols in the custom field level help.

If you want to restore the standard help for a field, open the Customize Standard Field Level Help window and click Remove Customization.

An icon beside the field name indicates that the field has custom help. Only administrators can see this icon.

![Custom Field Level Help](image)

Also, the word (customized) in the Field Help title bar indicates that the field has custom help.
Note: After an administrator or a user with the full level of the Customize Field Level Help permission changes the field level help, users may not see the changes right away. Field level help can be stored in cache for users, and there are multiple caches, so it can take a few hours for all users to see updated field level help entries.

Standard Field Level Help Customization Page

To edit or remove the custom field level help for one or more fields, use the Standard Field Level Help Customization page.

Go to Customization > Lists, Records, & Fields > Custom Help for Standard Fields.

Users with the full level of the Customize Field Level Help permission can edit or remove the custom help for one or more fields. The view, none, and edit levels of the Customize Field Level Help permission are not used.

Filter the help list by form type if required.

To remove the custom help for several fields, check the box in the appropriate rows and click Remove Customization of Selected.
Custom Fields

Custom fields are fields that you can add to your records and transactions to record information specific to your business needs. Record custom fields can be added to existing and custom subtabs on the entry forms you use to enter records in your NetSuite account. Transaction custom fields can be added to the top (body) or the line items (columns) of transactions. For more information, see the following.

- Table of Custom Field Type Descriptions
- Available Standard Fields and Field Types
- Kinds of Custom Fields

**Important:** SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should only access the NetSuite UI by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic SuiteScript 2.0 Custom Pages.

To create a custom field you need to do the following:

- Set the basic properties for the field. For information, see Creating a Custom Field.
- Assign the field to the desired forms. For information, see Assigning Custom Fields to Specific Record Types.
- Set the display properties for the field. For information, see Setting Display Options for Custom Fields.
- Set the desired validation and defaulting properties. If needed, create dynamic defaults and hyperlinks. For information, see Setting Validation and Defaulting Properties and Dynamic Defaults and Dynamic Hyperlinks.
- Set any sourcing criteria for the field. For information, see Setting Sourcing Criteria.
- Set any access restrictions to the field based on department, role, or subsidiary. For information, see Restricting Access to Custom Fields and Restricting Access to Employee Custom Fields.
- Set any filtering criteria for the field. For information, see Setting Filtering Criteria.

You also have several optional setup tasks when creating a custom field:

- Multiple Dependent Dropdown Lists
- Creating Read-Only Custom Fields
- Adding Translations for Custom Fields
- Adding Custom Fields to Transaction Forms
- Creating Formula Fields
- Custom Lists

**Warning:** You should be aware of the consequences of deleting a custom field. Instances of the deleted field will be removed from forms and lists and all associated data will be deleted. Reports and searches containing the deleted field will either have the field removed or may error out, depending on how the field is used. If you inactivate the field, the data is retained in NetSuite. Also note that changing the data type or permissions associated with a custom field can result in errors for reports and searches containing that field.

Table of Custom Field Type Descriptions

When creating a custom field, you select the type of field you want to create, depending on the kind of information you want to store in the field.
The following table outlines all available custom field types. Any character limits mentioned apply to English letters, which require one byte for each letter. If you are using another language, the limit is lower because one character could require up to four bytes. For non-English characters, the limits for the fields vary depending on the characters entered, and the field maximum is lower than the number indicated.

For information on creating a custom field, see Creating a Custom Field.

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check box</td>
<td>Record a true or false answer by placing a check mark in the custom field.</td>
<td>For example, you want to track orders by delivery. You create a check box custom field called Delivery that appears on sales transactions. If orders are for delivery, you check the Delivery box.</td>
</tr>
<tr>
<td>Currency</td>
<td>Enter currency amounts in a custom field. A custom field of type currency behaves differently from standard NetSuite currency fields. That is, if you are not using the Multi-currency feature, the custom field will not be hidden like standard NetSuite currency fields.</td>
<td>For example, you want to track the spending limit of each employee in your company. You create a currency custom field that appears on employee records and enter the spending limit for each employee.</td>
</tr>
<tr>
<td>Date</td>
<td>Enter or choose dates. Note that changing Date fields to Date/Time fields will automatically convert Dates in existing records to Date/Times with the time value defaulting to midnight of your company's time zone. All existing SuiteScript that referenced the Date field will need to be manually updated.</td>
<td>For example, you want to record projected start dates on estimates. You create a date custom field called Projected Start Date. Dates entered must follow the date format selected at Setup &gt; Company &gt; General Preferences. <strong>Note:</strong> The date and time are based on user's NetSuite Time Zone preference, set at Home &gt; Set Preferences, not on the browser client time zone.</td>
</tr>
<tr>
<td>Date/Time</td>
<td>Combine date and time values in one field. After a user enters a date/time value, the data is rendered in the user's preferred date and time format, as well as the user's time zone. Also note that time values are stored in NetSuite down to the second. If you choose, you can also add Date/Time custom fields in SuiteScript using the <code>form.addField(options)</code> method. <strong>Important:</strong> If you use the <code>addField</code> method, for the type parameter, you must specify 'datetimetz'.</td>
<td>For example, you may want a single field to contain date and time timestamp data. <strong>Note:</strong> The date and time are determined by the user's NetSuite Time Zone preference, set at Home &gt; Set Preferences, not by the browser client time zone.</td>
</tr>
<tr>
<td>Decimal Number</td>
<td>Enter a decimal number (maximum 21 digits) in a custom field.</td>
<td>For example, you want to record the distance your customers are from your nearest retail outlet. You create a decimal number custom field to store the information on your employee records.</td>
</tr>
<tr>
<td>Document</td>
<td>Select a File Cabinet document to preview or download. The field is searchable and can be added to search results. <strong>Note:</strong> The user of the document custom field must have File Cabinet access to view, select or upload documents.</td>
<td>For example, you want employees to be able to search customer survey results.</td>
</tr>
<tr>
<td>Entity</td>
<td>You can create custom fields of type Entity. To create a custom entity field, set the field type</td>
<td>For example, on the custom field's definition page, on the Filtering and Sourcing subtab,</td>
</tr>
<tr>
<td>Field Type</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Entity</td>
<td>Specify Type in the Filter Using column. In the Value Is column, specify customer, partner, and vendor. When the custom field of type Entity appears on your record, only entities of type customer, partner, and vendor will appear as available options in the field’s list.</td>
<td></td>
</tr>
<tr>
<td>E-mail address</td>
<td>Enter an email address in a custom field. The email address field creates a link to open your default email client. For example, you create a custom record type for intern records. You can include a field for their email addresses. You can send email to interns by clicking email addresses on their records.</td>
<td></td>
</tr>
<tr>
<td>Free-Form Text</td>
<td>Enter up to 300 characters of text in a custom field. In the Note section, it mentions that if you add a data center-specific URL, the URL automatically redirects to an account-specific domain URL. Configuration remains unchanged. You should update all links in custom fields to use the correct account-specific domain. For more information, see Account-Specific Domains in Custom Fields. For example, you want to offer your customers monogramming on certain items. You can create a Free-Form Text field as an item option to record the monogram they want.</td>
<td></td>
</tr>
<tr>
<td>Help</td>
<td>Place helpful text on record pages where your users enter information. The help that appears is for informational purposes only. It is not stored in your account. In the Note section, it mentions that if you define data-specific URL in help, the URL automatically translates to an account-specific domain URL. Configuration remains unchanged. You should update all links in custom fields to use your account-specific domain. For more information, see Account-Specific Domains in Custom Fields. When you create a Help custom field, choose Help in the Type field and enter your text in the Help field.</td>
<td></td>
</tr>
<tr>
<td>Hyperlink</td>
<td>Enter a URL in a custom field that links to another web page. Hyperlink fields should begin with 'http://', 'https://', or 'ftp://'. For example, you want to link to a vendor’s website. You create a hyperlink custom field called Website that appears on vendor records. First, you enter the company’s URL. When you return to the vendor’s record, you can click the</td>
<td></td>
</tr>
<tr>
<td>Field Type</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>---------</td>
</tr>
<tr>
<td>URL</td>
<td>URL to open the vendor’s site in a new window. You can enter up to 999 characters in the field.</td>
<td></td>
</tr>
<tr>
<td>Note: Files beginning with file:// or \ are permitted, but this format is discouraged. Clicking on these links is not supported due to security concerns.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Note: If you add a data center-specific URL, the URL automatically translates to an account-specific domain URL. Configuration remains unchanged. You should update all links in custom fields to use your account-specific domain. For more information, see Account-Specific Domains in Custom Fields.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Image      | Attach an image to a record. The image is rendered with a maximum width of 250 pixels in the form layout, and the image is resized proportionally. Image custom fields support the following file formats:  
- .bmp  
- .gif  
- .jpg  
- .jpeg  
- .pjpg  
- .pjpeg  
- .png  
- .tiff  
- .tif  
Note: The TIF/TIFF file format may not be supported by all browsers. | For example, you can attach pictures of your employees to their records. |
<p>| Inline HTML| Use HTML to define a custom field to be included on custom pages generated by Suitelets. For more information, see the help topic Suitelets. The field limit of the Inline HTML Field Type is 4000 bytes. In English, 4000 bytes is 4000 characters, but in other languages, the number of characters may be less. | For example, you could display to customers how many reward points they have earned. |</p>
<table>
<thead>
<tr>
<th>Field Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Important:</strong></td>
<td>NetSuite Forms does not support manipulation through the Document Object Model (DOM) and custom scripting in Inline HTML fields. The use of Inline HTML fields on Form pages (except for Forms generated by Suitelets) will be deprecated in a future release.</td>
<td>For example, you can record a special ID number on your company records.</td>
</tr>
<tr>
<td><strong>Note:</strong></td>
<td>If your default value for the field contains data-specific URL, the URL automatically translates to an account-specific domain URL for the active session. Configuration remains unchanged. You should update all links in custom fields to use your account-specific domain. For more information, see Account-Specific Domains in Custom Fields.</td>
<td>For example, you want to track orders by the employee who works on them. You create a custom body field for sales transactions. In the List/Record field, you attach your employee list. On sales transactions, you can then select the appropriate employee for the order.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>Enter integers in a custom field.</td>
<td>For example, you can record a special ID number on your company records.</td>
</tr>
<tr>
<td>List/Record</td>
<td>Attach lists and records to custom fields. In the List/Record field, select the list or kind of record you want to attach. You can also choose from a list of public saved searches. You can also use the list/record option to script a custom printing solution. For more information, see Scripting a Custom Printing Solution.</td>
<td>For example, you want to track orders by the employee who works on them. You create a custom body field for sales transactions. In the List/Record field, you attach your employee list. On sales transactions, you can then select the appropriate employee for the order.</td>
</tr>
<tr>
<td>Long Text</td>
<td>A text area that can hold up to 1,000,000 characters. Use Long Text Custom fields where you need greater than 4,000 characters of text for a text area, and where you do not want the field to contain rich text formatting. When you create a long text field type in the UI, the field character limit is 1,000,000. When you create a long text field type using SuiteScript, the field character limit is 100,000.</td>
<td>For example, use a Long Text custom field in a custom record to hold the text of an item Warranty, or add to an Online Form to hold the text of a End User License Agreement.</td>
</tr>
<tr>
<td>Field Type</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Text Area</td>
<td>Lets you enter up to 4,000 characters of text in a custom field.</td>
<td>For example, you want to record special information about how you closed a customer.</td>
</tr>
<tr>
<td>Rich Text</td>
<td>Lets you enter and format up to 100,000 characters of text in a custom field.</td>
<td>For example, you can include a small biography of your employees with information like marital status, family members and other special information. You can also format the text to create lists, make text bold or italic, and change the font size and color.</td>
</tr>
<tr>
<td>Phone Number</td>
<td>Lets you create a field for a telephone number on records and transactions.</td>
<td>For example, you can include a contact number on an event record using a CRM field.</td>
</tr>
<tr>
<td>Percent</td>
<td>Lets you create a field to store a percentage. The information entered in the field can only be an integer from 0.</td>
<td>100. The percent sign (%) is automatically added to the information.</td>
</tr>
<tr>
<td>Password</td>
<td>Lets you create a field that is encrypted in the database. When you view the record it will always show a fixed number of characters regardless of how long the password is. When validating, you pull the encrypted password value into a hidden field and use custom code to encrypt the value the user typed and compare it with the actual encrypted value.</td>
<td>For example, you could add a password custom field to a web page.</td>
</tr>
<tr>
<td>Multiple Select</td>
<td>Lets you create a field where you can make multiple selections from a list or list of records.</td>
<td>For example, you want to track how customers heard about your business. You create a multiple select custom field that appears on customer records and contains a list of possible sources for how customers heard about your business. Sales representatives use the field to select the different places customers heard about your business.</td>
</tr>
<tr>
<td>Field Type</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>Note: A free form text or text area field can be converted to a long text field. However, after the conversion is complete, the field cannot be converted back.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Note: If you add a data center–specific URL, the URL automatically translates to an account–specific domain URL. Configuration remains unchanged. You should update all links in custom fields to use your account–specific domain. For more information, see Account–Specific Domains in Custom Fields.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table of Custom Field Type Descriptions

<table>
<thead>
<tr>
<th>Field Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>If you add a data center-specific URL, the URL automatically redirects to an account-specific domain URL. Configuration remains unchanged. You should update all links in custom fields to use the correct account-specific domain. For more information, see Account-Specific Domains in Custom Fields.</td>
<td></td>
</tr>
<tr>
<td>Time of Day</td>
<td>Lets you enter the time of day in a custom field.</td>
<td>For example, you create a Best Time To Call custom field, and then your sales representatives could enter the best time of day to call a prospect.</td>
</tr>
</tbody>
</table>

### Scripting a Custom Printing Solution

If you use the List/Record field type, Advanced PDF/HTML Templates is available in the list. The Advanced PDF/HTML Templates option provides users with a list of all available advanced templates. You can incorporate the option in a scripted custom printing solution that lets users select the form to be used for printing.

**Note:** The Advanced PDF/HTML Templates option lists all available templates and is not filtered for users. Only use the Advanced PDF/HTML Templates option with a scripted custom printing solution that specifies what action to perform when the user selects a form from the list.

### Viewing Custom Images and Custom Image Files

After you add a custom image to a form using a custom image field, it is shown on the form when you view it. You can interact with the image in two ways:

1. When you click the image thumbnail, you are directed to its file in the File Cabinet.
2. When you hover over the image, the Open button appears. When you click the Open button, a dialog appears showing a full-size preview of the image.

### Changing Field Types

When you edit a custom field and change the field type, the data that you have previously entered in that field is preserved whenever possible.

The retention of data follows the following rules:

- **Changing to a Different Field Type**
  
  If you change the field type to a fundamentally different field type, existing data you had entered in that field is deleted. A warning message is displayed when you make a change.

  For example, changing a phone number field to a rich text field would preserve your data, whereas changing a rich text field to a percent field would not.

- **Changing the Source List**
If you change the source list for a list/record field or a multiple select field, existing data in that field is not preserved. A warning message is displayed when you make a change.

- **Switching Numeric Field Types**
  If you change a decimal, currency, or percent type field to an integer field, your data is rounded off to the nearest whole number.

  **Warning:** Custom field filters are type sensitive. If you change the type of a custom field, filters defined for the field may become incompatible, resulting in unexpected errors when a form containing the field is displayed. If you plan to change a custom field’s type, you need to review any existing filters defined for the field, and remove or change these filters to avoid errors. For information about field filtering, see Setting Filtering Criteria.

### Available Standard Fields and Field Types

Each kind of custom field has standard fields that can be sourced from. Following are tables that list available standard fields and the required field type for each kind of custom field.

#### Transaction Body Fields

**Source From Entity**

The type of entity used to source from depends on the type of transactions you apply your field to. A **purchase** transaction sources from **vendor** records. A **sales** transaction sources from **customer** records, and an **expense report** sources from **employee** records.

**Note:** In the List/Record field, you must choose the Employee list for sourcing to work properly.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Sales Transactions</th>
<th>Purchase Transactions</th>
<th>Expense Reports</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Bill To</td>
<td>Text Area</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Ship To</td>
<td>Text Area</td>
<td>×</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phone</td>
<td>Phone Number</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Fax</td>
<td>Phone Number</td>
<td>×</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E-mail</td>
<td>E-mail Address</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>City</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>State</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Zip</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Country</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Sales Rep</td>
<td>List/Record</td>
<td>×</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Expected Close Date</td>
<td>Date</td>
<td>×</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Renewal Date</td>
<td>Date</td>
<td>×</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Sales Transactions</td>
<td>Purchase Transactions</td>
<td>Expense Reports</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------</td>
<td>--------------------</td>
<td>-----------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Contact</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Alt. Contact</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Alt. Phone</td>
<td>Phone Number</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Balance</td>
<td>Currency</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Credit Limit</td>
<td>Currency</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Account</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>1099 Eligible</td>
<td>Check Box</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Tax ID</td>
<td>Free-Form Text</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Legal Name</td>
<td>Free-Form Text</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Supervisor</td>
<td>List/Record</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Soc. Sec. #</td>
<td>Free-Form Text</td>
<td></td>
<td>×</td>
<td></td>
</tr>
</tbody>
</table>

**Source From Ship To**

When you choose Ship To in the Source From field, information is sourced from the customer record chosen in the **Ship To** field of purchase transactions.

ℹ️ **Note:** In the List/Record field, you must choose the Employee list for sourcing to work properly.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Bill To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Ship To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Phone</td>
<td>Phone Number</td>
</tr>
<tr>
<td>Fax</td>
<td>Phone Number</td>
</tr>
<tr>
<td>E-mail</td>
<td>E-mail Address</td>
</tr>
<tr>
<td>City</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>State</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Zip</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Country</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Sales Rep</td>
<td>List/Record</td>
</tr>
<tr>
<td>Expected Close Date</td>
<td>Date</td>
</tr>
<tr>
<td>Renewal Date</td>
<td>Date</td>
</tr>
<tr>
<td>Contact</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Alt. Contact</td>
<td>Free-Form Text</td>
</tr>
</tbody>
</table>
Available Standard Fields and Field Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alt. Phone</td>
<td>Phone Number</td>
</tr>
<tr>
<td>Balance</td>
<td>Currency</td>
</tr>
<tr>
<td>Credit Limit</td>
<td>Currency</td>
</tr>
</tbody>
</table>

**Source From Sales Rep**

When you choose Sales Rep in the Source From field, information is sourced from the employee record chosen in the **Sales Rep** field of customer records.

**Note:** In the List/Record field, you must choose the Employee list for sourcing to work properly.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Bill To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Phone</td>
<td>Phone Number</td>
</tr>
<tr>
<td>E-mail</td>
<td>E-mail Address</td>
</tr>
<tr>
<td>City</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Country</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Supervisor</td>
<td>List/Record</td>
</tr>
<tr>
<td>Soc. Sec. #</td>
<td>Free-Form Text</td>
</tr>
</tbody>
</table>

**Transaction Line Fields**

**Source From Item**

**Note:** In the List/Record field, you must choose the Vendor for sourcing to work properly.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Display Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Vendor Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Online Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Available Online</td>
<td>Check Box</td>
</tr>
<tr>
<td>Base Price</td>
<td>Currency</td>
</tr>
<tr>
<td>Cost</td>
<td>Currency</td>
</tr>
<tr>
<td>Preferred Vendor</td>
<td>List/Record</td>
</tr>
</tbody>
</table>
### Available Standard Fields and Field Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>On Hand</td>
<td>Decimal Number</td>
</tr>
</tbody>
</table>

### Source From Customer

When you choose Customer in the Source From field, information is sourced from the customer record chosen in the **Customer:Project** field of **purchase** transactions.

**Note:** In the List/Record field, you must choose the Employee list for sourcing to work properly.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Bill To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Ship To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Phone</td>
<td>Phone Number</td>
</tr>
<tr>
<td>Fax</td>
<td>Phone Number</td>
</tr>
<tr>
<td>E-mail</td>
<td>E-mail Address</td>
</tr>
<tr>
<td>City</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>State</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Country</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Sales Rep</td>
<td>List/Record</td>
</tr>
<tr>
<td>Expected Close Date</td>
<td>Date</td>
</tr>
<tr>
<td>Renewal Date</td>
<td>Date</td>
</tr>
<tr>
<td>Contact</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Alt. Contact</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Alt. Phone</td>
<td>Phone Number</td>
</tr>
<tr>
<td>Balance</td>
<td>Currency</td>
</tr>
<tr>
<td>Credit Limit</td>
<td>Currency</td>
</tr>
<tr>
<td>Account</td>
<td>Free-Form Text</td>
</tr>
</tbody>
</table>

### Transaction Item Options

### Source From Item

**Note:** In the List/Record field, you must choose the Vendor list for sourcing to work properly.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
</tbody>
</table>
### Available Standard Fields and Field Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Vendor Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Online Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Available Online</td>
<td>Check Box</td>
</tr>
<tr>
<td>Base Price</td>
<td>Currency</td>
</tr>
<tr>
<td>Cost</td>
<td>Currency</td>
</tr>
<tr>
<td>Preferred Vendor</td>
<td>List/Record</td>
</tr>
<tr>
<td>On Hand</td>
<td>Decimal Number</td>
</tr>
</tbody>
</table>

### CRM Fields

#### Source From Company or Contact

When you choose Company in the Source From field, information is sourced from the record chosen in the **Company** field of **case** records.

When you choose Contact in the Source From field, information is sourced from the record chosen in the **Contact** field of **case** records.

**Note:** In the List/Record field, you must choose the Employee list for sourcing to work properly.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Bill To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Ship To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Phone</td>
<td>Phone Number</td>
</tr>
<tr>
<td>Fax</td>
<td>Phone Number</td>
</tr>
<tr>
<td>E-mail</td>
<td>E-mail Address</td>
</tr>
<tr>
<td>City</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>State</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Country</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Sales Rep</td>
<td>List/Record</td>
</tr>
<tr>
<td>Expected Close Date</td>
<td>Date</td>
</tr>
<tr>
<td>Renewal Date</td>
<td>Date</td>
</tr>
<tr>
<td>Contact</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Alt. Contact</td>
<td>Free-Form Text</td>
</tr>
</tbody>
</table>
### Source From Item

When you choose Item in the Source From field, information is sourced from the record chosen in the Item field of case records.

**Note:** In the List/Record field, you must choose the Vendor list for sourcing to work properly.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Display Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Vendor Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Online Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Available Online</td>
<td>Check Box</td>
</tr>
<tr>
<td>Base Price</td>
<td>Currency</td>
</tr>
<tr>
<td>Cost</td>
<td>Currency</td>
</tr>
<tr>
<td>Preferred Vendor</td>
<td>List/Record</td>
</tr>
<tr>
<td>On Hand</td>
<td>Decimal Number</td>
</tr>
</tbody>
</table>

### Source From Assigned

When you choose Assigned in the Source From field, information is sourced from the record chosen in the Assigned To field of case records.

**Note:** In the List/Record field, you must choose the Employee list for sourcing to work properly.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Bill To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Phone</td>
<td>Phone Number</td>
</tr>
<tr>
<td>E-mail</td>
<td>E-mail Address</td>
</tr>
<tr>
<td>City</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Country</td>
<td>Free-Form Text</td>
</tr>
</tbody>
</table>
Available Standard Fields and Field Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervisor</td>
<td>List/Record</td>
</tr>
<tr>
<td>Soc. Sec. #</td>
<td>Free-Form Text</td>
</tr>
</tbody>
</table>

Entity Fields

Source From Sales Rep or Supervisor

When you choose Sales Rep in the Source From field, information is sourced from the record chosen in the Sales Rep field of customer records.

When you choose Supervisor in the Source From field, information is sourced from the records chosen in the Supervisor field of employee records.

Note: In the List/Record field, you must choose the Employee list for sourcing to work properly.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Bill To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Phone</td>
<td>Phone Number</td>
</tr>
<tr>
<td>E-mail</td>
<td>E-mail Address</td>
</tr>
<tr>
<td>City</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>State</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Zip</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Country</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Supervisor</td>
<td>List/Record</td>
</tr>
<tr>
<td>Soc. Sec. #</td>
<td>Free-Form Text</td>
</tr>
</tbody>
</table>

Item Fields

Source From Preferred Vendor

When you choose Preferred Vendor in the Source From field, information is sourced from the record chosen in the Preferred Vendor field of item records.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
</tbody>
</table>
Available Standard Fields and Field Types

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bill To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Ship To</td>
<td>Text Area</td>
</tr>
<tr>
<td>Phone</td>
<td>Phone Number</td>
</tr>
<tr>
<td>Fax</td>
<td>Phone Number</td>
</tr>
<tr>
<td>E-mail</td>
<td>E-mail Address</td>
</tr>
<tr>
<td>City</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>State</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Zip</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Country</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Contact</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Alt. Contact</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Alt. Phone</td>
<td>Phone Number</td>
</tr>
<tr>
<td>Balance</td>
<td>Currency</td>
</tr>
<tr>
<td>Credit Limit</td>
<td>Currency</td>
</tr>
<tr>
<td>Account</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>1099 Eligible</td>
<td>Check Box</td>
</tr>
<tr>
<td>Tax ID</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Legal Name</td>
<td>Free-Form Text</td>
</tr>
</tbody>
</table>

Kinds of Custom Fields

Following are the different types of custom record and transaction fields you can create:

- **Custom CRM Fields** – Used to add fields to CRM records. These records include Activity, Marketing and Support records — such as tasks, events, campaigns or cases.
- **Custom Entity Fields** – Used to add fields to entity records. These records include Relationship and Employee records — such as customers, vendors, employees, contacts, partners or groups.
- **Custom Item Fields** – Used to add fields to item records. These records include Accounting and website item records — such as inventory, non-inventory, service, other charge, group, kit/package and assembly/bill of materials item records.
- **Custom Transaction Body Fields** – Used to add fields to the body of transaction records. These records include purchase, sale, journal entry and expense report records — such as sales orders, invoices, purchase orders, opportunities, Web store transactions or item receipts.
- **Custom Transaction Line Fields** – Used to add fields to the columns of transaction records. These fields display in the line-item columns of transaction records and include fields such as expense items, purchase items, sales items, store items or opportunity items.
- **Custom Transaction Item Options** – Used to add fields to the line items of your transaction records such as purchase items, sales items and Web store items. When adding a custom field to the line items of a transaction, you apply the field to the type of line item.
- **Custom Item Number Fields** – Used to add fields to serial and lot numbered inventory records to track information specific to each item or workflow unique to your business. For example, quality control procedures or recall information could be tracked.
- **Other Record Fields** – Used to add fields to custom records not defined by the above categories, including campaign events, classes, competitors, departments, and locations.
- **Other Sublist Fields** — Used to add fields to the columns of a custom sublist. These fields display in the line-item columns of sublists.

**Note:** Currently, only the Bill of Materials Revision sublist is supported.

You can differentiate between custom fields and standard fields on pages in NetSuite if you enable the Show Internal IDs preference at Home > Set Preferences under the Defaults section of the General subtab. With the preference enabled, when you click on a field, the field level help popup shows a field ID in the bottom right corner. If the field you clicked on is a custom field, the ID from the custom field record is shown.

### Custom CRM Fields

Custom CRM fields are fields that you can add to your CRM records to gather information specific to your business needs.

These records include:

- task records
- phone call records
- campaign records
- case records
- solution records
- event records

**To create or modify custom CRM fields:**

2. On the Custom CRM Fields page, each custom field is listed, with columns providing detailed information about the field and which records the field has been applied to.
3. To edit an existing custom CRM field, click the field name in the Description column and then modify the field definition as desired.
4. To add a new custom CRM field, click **New**.

**Note:** Custom CRM fields can be indexed for global search. To include a custom field in global searches, check the Global Search box on its record. You cannot index a custom field for global search if None is selected for any Level for Search/Reporting option on the Access subtab of the custom field record. For more information, see Creating a Custom Field and Including Custom Fields in Global Search.

You can use SuiteCloud Development Framework (SDF) to manage custom CRM fields as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom CRM field.
Kinds of Custom Fields

To another of your accounts. Each custom CRM field page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

Here is an example of a custom CRM field.

![Custom CRM Field Image]

When the custom CRM field is included on a form, it might look like the following example.

![Event CRM Field Image]

Custom Entity Fields

Custom entity fields are fields that you can add to your entity records to gather information specific to your business needs. Entity custom fields can be added to existing and custom subtabs on the entry forms you use to enter entity records in your NetSuite account.

These records include the following relationship and employee records:

- customer records
Kinds of Custom Fields

- project records
- vendor records
- other name records
- contact records
- partner records
- entity group records
- employee records
- website registration

**To create or modify custom entity fields:**

2. On the Custom Entity Fields page, each custom field is listed, with columns providing detailed information about the field and which records the field has been applied to.
3. To edit an existing custom entity field, click the field name in the Description column and then modify the field definition as desired.
4. To add a new custom entity field, click **New**.

**Note:** Custom entity fields can be indexed for global search. To include a custom field in global searches, check the Global Search box on its record. You cannot index a custom field for global search if None is selected for any Level for Search/Reporting option on the Access subtab of the custom field record. For more information, see Creating a Custom Field and Including Custom Fields in Global Search.

You can use SuiteCloud Development Framework (SDF) to manage custom entity fields as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom entity field to another of your accounts. Each custom entity field page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

Here is an example of a custom entity field.

![Custom Entity Field](image)

When the custom Entity field is included on a form, it might look like the following example.
Custom Item Fields

Custom item fields are fields that you can add to your item records to gather information specific to your business needs.

These records include the following Accounting and website item records:

- inventory item
- non-inventory items
- service items
- expense items
- other charges
- item groups
- kit/packages
- assembly/bill of materials

To create or modify custom item fields:

1. Go to Customization > Lists, Records, & Fields > Item Fields.
2. On the Custom Item Fields page, each custom field is listed, with columns providing detailed information about the field and which records the field has been applied to.
3. To edit an existing custom item field, click the field name in the Description column and then modify the field definition as desired.
4. To add a new custom item field, click New.

**Note:** Custom item fields can be indexed for global search. To include a custom field in global searches, check the Global Search box on its record. You cannot index a custom field for global search if None is selected for any Level for Search/Reporting option on the Access subtab of the custom field record. For more information, see Creating a Custom Field and Including Custom Fields in Global Search.

You can use SuiteCloud Development Framework (SDF) to manage custom items fields as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development SuiteBuilder.
Framework Overview. You can use the Copy to Account feature to copy an individual custom item field to another of your accounts. Each custom item field page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

Here is an example of a custom item field.

When the custom item field is included on a form, it might look like the following example.

Custom Transaction Body Fields

Custom transaction body fields are fields that you can add to the body of transaction records to gather information specific to your business needs.
These records include:

- purchase transactions
- sales transactions
- revenue arrangements
- opportunities
- journal entries
- expense reports
- Web store checkout
- item receipts
- item fulfillments
- fulfillment requests
- store pickup fulfillments

To create or modify custom transaction body fields:

2. On the Custom Transaction Body Fields page, each custom field is listed, with columns providing detailed information about the field and which records the field has been applied to.
3. To edit an existing custom transaction body field, click the field name in the Description column and then modify the field definition as desired.
4. To add a new custom transaction body field, click New.

**Note:** Custom transaction body fields can be indexed for global search. To include a custom field in global searches, check the Global Search box on its record. You cannot index a custom field for global search if None is selected for any Level for Search/Reporting option on the Access subtab of the custom field record. For more information, see Creating a Custom Field and Including Custom Fields in Global Search.

You can use SuiteCloud Development Framework (SDF) to manage custom transaction body fields as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom transaction body field to another of your accounts. Each custom transaction body field page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

The following screenshot shows the Custom Transaction Body Field page configured for a custom Entered By: field.
After it has been created, the custom transaction body field can be applied to a transaction form. The following screenshot shows the Entered By: field included on a Purchase Order form.

If the Display Type for the custom transaction body field is set to Hidden, the field is still included on all forms, even if it is not displayed. The system ignores any changes you make to show or hide the field on all forms that apply to the custom transaction body field.
Custom Transaction Line Fields

Custom transaction line fields are fields that you can add to the line items of your transaction records to gather information specific to your business needs.

These records include:

- expense items
- purchase items
- sale items
- store items
- revenue arrangements
- journal entries
- expense reports
- item receipts or fulfillments
- opportunity items
- time cards
- inventory adjustment
- inventory transfer
- fulfillment requests
- store pickup fulfillments

To create or modify custom transaction line fields:

2. On the Custom Transaction Line Fields page, each custom field is listed, with columns providing detailed information about the field and which records the field has been applied to.
3. To edit an existing custom Transaction Line field, click the field name in the Description column and then modify the field definition as desired.
4. To add a new custom Transaction Line field, click New.

For more information, see Creating a Custom Field.

You can use SuiteCloud Development Framework (SDF) to manage custom transaction line fields as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom transaction line field to another of your accounts. Each custom transaction line field page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

Here is an example of a custom transaction line field.
When the custom transaction line field is included in a form, it might look like the following example.

If the Display Type for the custom transaction line field is set to Hidden, the field is still included on all forms, even if it is not displayed. The system ignores any changes you make to show or hide the field on all forms that apply to the custom transaction line field.

**Check Box Type Limitation**

On custom transaction line fields, a limitation exists when all of the following conditions are met:

1. Type = Check Box
2. Store Value = checked
3. Applies To = only Item Fulfillment is checked
4. On the Validation & Defaulting subtab, Default Checked is checked
5. Create a sales order and add at least one item
6. Click Save & Fulfill

On the item fulfillment transaction, the transaction line field is not checked, despite the settings described above. The inconsistency occurs because the custom field information is taken from the sales order, and if the check box is not applied there, NetSuite uses the default system value. As a workaround, you can apply the field to sales orders and hide it on the sales order forms, if required.
Custom Transaction Item Options

Transaction item options are fields that you can add to the line items of your transaction records to gather information specific to your business needs.

Item options can refer to characteristics of an item. For example, you might record different colors of an item as item options.

You can create item options for:

- purchase items
- sales items
- opportunities
- web store items
- kits and assemblies

**Note:** You may encounter performance issues if there are too many item option fields on your form. Try to limit the number of custom item option fields to 50 fields.

**To create or modify custom transaction item option fields:**

2. On the Custom Transaction Item Option Fields page, each custom field is listed, with columns providing detailed information about the field and which records the field has been applied to.
3. To edit an existing custom Transaction Item Option field, click the field name in the description column and then modify the field definition as desired.
4. To add a new custom Transaction Item Option field, click New.
   For more information, see Creating a Custom Field.
5. Complete fields on the Item Option page and click Save.

**Note:** You can define a formula for an item option’s default value (by checking the Formula box on the Setting Validation and Defaulting Properties), but formula references to non-item option fields are not supported. You may be able to save an item option successfully with a reference to a non-item option field, but users will encounter errors when they try to save a line item with the item option selected.

**Offer your customers free promotional gift wrapping**

After you have enabled the Item Options feature, you can offer your customers complementary gift wrapping on select items as a promotion. With item options, create a Gift Wrapping check box on items so customers can indicate if they would like gift wrapping.

**To create an item option for gift wrapping:**

2. In the Label field, enter the name of your option, Gift Wrapping.
3. In the Type field, choose Check Box to create a check box field.
4. On the Applies To subtab, check the Sale and Web Store boxes for the option to appear on sales transactions and in your website.
5. In the Items field, select the individual items you want to offer gift wrapping for.
   You can choose multiple items by holding down the Ctrl key as you select the items with the mouse.
6. On the **Display** subtab, in the **Label for Input** field, enter the name for the option as you want it to appear to customers on your website.

7. When you have finished, click **Save**.

Now, you can offer customers complementary gift wrapping for items. The choice your customers make appears in the **Options** column of each sales transaction.

Here is an example of a custom transaction item option field.

![Item Option](image)

When the custom transaction item option field is included on a form, it might look like the following example.

![Set Options](image)
Custom Item Number Fields

Custom item number fields are fields that you can add to inventory records to track information specific to each item or workflow unique to your business. For example, you can track the status and results of quality control procedures specific to each serialized item, or you can track recall information on lot records.

These records include:

- serial numbered items – see the help topic Serial Numbered Items
- lot numbered items – Lot Numbered Items

**To create or modify custom item number fields:**

1. Go to Customization > Lists, Records, & Fields > Item Number Fields.
2. On the Custom Item Number Field page, each custom field is listed, with columns providing detailed information about the field and which records the field has been applied to.
3. To edit an existing custom item number field, click the field name in the description column and then modify the field definition as desired.
4. To add a new custom item number field, click New.
   
   For more information, see Creating a Custom Field.

When you create a lot numbered or serialized inventory item or edit an existing record, you can apply the field to the item as long as you have not set the field to apply to all items. On the item record, click the Custom subtab. In the Inventory Number Options field, press Ctrl and select all the item number fields you want to apply to the item. Then click Save.

You can use SuiteCloud Development Framework (SDF) to manage custom item number fields as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom item number field to another of your accounts. Each custom item number field page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

Here is an example of a custom item number field.
When the custom item number field is included on a form, it might look like the following example.

The custom item number field is found in the serial number record or lot number record of the inventory detail subrecord. To view the field, enter a new transaction where the custom field is used. Enter an item and then click Inventory Detail. Select and open the details of a serial or lot number. The custom item number field appears on the Serial Number Record page, as shown in the preceding screenshot.

For more information, see the help topic Advanced Bin / Numbered Inventory Management.

Other Record Fields

Other record fields are used for records that do not have custom forms associated with them. You can add other record fields to these record types to gather information specific to your business needs.

Other record fields can be added to the following record types:

- Account
- Accounting Book
- Address (See Creating Custom Address Fields)
- Item Location Configuration
- Item Source
- Item Supply Plan
<table>
<thead>
<tr>
<th>Kinds of Custom Fields</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allocation Type</td>
<td>Location</td>
</tr>
<tr>
<td>Automatic Location Assignment Configuration</td>
<td>Location Costing Group</td>
</tr>
<tr>
<td>Automatic Location Assignment Rule</td>
<td>Manufacturing Cost Template</td>
</tr>
<tr>
<td>Bill of Distribution</td>
<td>Manufacturing Routing</td>
</tr>
<tr>
<td>Bill of Materials</td>
<td>Note</td>
</tr>
<tr>
<td>Bill of Materials Revision</td>
<td>Payroll Batch</td>
</tr>
<tr>
<td>Billing Rate Card</td>
<td>Payroll Item</td>
</tr>
<tr>
<td>Bin</td>
<td>Planned Standard Cost</td>
</tr>
<tr>
<td>Campaign Event</td>
<td>Product</td>
</tr>
<tr>
<td>Charge</td>
<td>Product Version</td>
</tr>
<tr>
<td>Charge Rule</td>
<td>Project Budget</td>
</tr>
<tr>
<td>Class</td>
<td>Promotion Code</td>
</tr>
<tr>
<td>Company</td>
<td>Region</td>
</tr>
<tr>
<td>Competitor</td>
<td>Role</td>
</tr>
<tr>
<td>Department</td>
<td>Shipping Partner Package</td>
</tr>
<tr>
<td>Distribution Category</td>
<td>Shipping Partner Registration</td>
</tr>
<tr>
<td>Distribution Network</td>
<td>Shipping Partner Shipment</td>
</tr>
<tr>
<td>Domain</td>
<td>Standard Cost Version</td>
</tr>
<tr>
<td>Expense Category</td>
<td>Subsidiary</td>
</tr>
<tr>
<td>Format Profile</td>
<td>Subsidiary Setting</td>
</tr>
<tr>
<td>GL Numbering Sequence</td>
<td>Supply Chain Snapshot</td>
</tr>
<tr>
<td>Global Account Mapping</td>
<td>Supply Chain Snapshot Simulation</td>
</tr>
<tr>
<td>Global Inventory Relationship</td>
<td>Tax Code</td>
</tr>
<tr>
<td>Inbound Shipment</td>
<td>Timesheet</td>
</tr>
<tr>
<td>Inventory Cost Template</td>
<td>Vendor-Subsidiary Relationship</td>
</tr>
<tr>
<td>Inventory Status</td>
<td>Work Breakdown Structure</td>
</tr>
<tr>
<td>Item Account Mapping</td>
<td>Workflow</td>
</tr>
<tr>
<td>Item Demand Plan</td>
<td>Workplace</td>
</tr>
</tbody>
</table>

**To create or modify other record fields:**

1. Go to Customization > Lists, Records, & Fields > Other Custom Fields.
2. On the Other Record Fields page, each custom field is listed, with columns providing detailed information about the field and which records the field has been applied to.
3. To edit an existing record field, click the field name in the **Description** column and then modify the field definition as desired.
4. To add a new custom record field, click **New**.
   For more information, see [Creating a Custom Field](#).

You can use SuiteCloud Development Framework (SDF) to manage custom other record fields as part of file-based customization projects. For information about SDF, see the help topic [SuiteCloud Development Framework Overview](#). You can use the Copy to Account feature to copy an individual custom other record field to another of your accounts. Each custom other record field page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic [Copy to Account Overview](#).

Here is an example of an other record field.

When the other record field is included in a form, it might look like the following example.
Other Sublist Fields

Custom sublist fields are fields that you can add to the column of your sublists to gather information specific to your business needs.

⚠️ **Important:** Currently, sublist fields can only be added to the Bill of Materials Revision sublist.

### To create or modify custom sublist fields:

1. Go to Customization > Lists, Records, & Fields > Other Sublist Fields.
2. On the Other Sublist Fields list page, each custom sublist field is listed, with columns providing detailed information about the field and which record type the field applies to.
3. To edit an existing custom sublist field, click the field name in the Description column and then modify the field definition as desired.
4. To add a new custom sublist field, click **New**.
   
   For more information, see Creating a Custom Field.

Here is an example of a custom sublist field.

![Other Sublist Field](image)

When the other sublist field is included in a form, it might look like the following example.
Creating a Custom Field

To record information specific to your business needs, you can create different types of custom record and transaction fields.

See Kinds of Custom Fields for an overview of the different types of custom fields.

To create a custom field:

1. Go to Customization > Lists, Records, & Fields > [Custom Field] > New, where [Custom Field] is the desired field type. The Custom Field page is displayed for the selected type. For example, the Custom Entity Field page looks like the following.

To display all settings for a custom field, click the Expand Tabs icon.
2. In the **Label** field, enter a name or description for the field that is meaningful to your users. You can enter up to 200 characters for the label. However, you should consider how a long label will appear on pages and printed forms.

The label is displayed by the field on the transactions you select. If you change the name of a custom field later, the name is not automatically updated on custom forms that contain the field. For information, see **Renaming Custom Fields**.

3. Enter a unique ID for the field. To identify the location and purpose of the custom field, establish a naming convention such as using an underscore followed by the field name. For example, all custom CRM fields would result in a name custevent_fieldname.

The field can be set to an alphanumeric string up to 30 characters long. NetSuite automatically modifies the ID entered into the field with a prefix that corresponds to the type of field being created as listed below.

- CRM: custevent
- Entities: custentity
- Items: custitem
- Record: custrecord
- Sublist: custrecord
- Transaction Body: custbody
- Transaction Column: custcol

**Note:** Entering your own IDs enables you to maintain them in SuiteScript. Custom field IDs are displayed for each field in the Custom Field lists and on the Record Type page when creating a new record. If you do not enter an ID, NetSuite automatically assigns one when you save the new custom field. After a field ID has been assigned, it can be changed by clicking the Change ID button at the top of the page.

4. Select the owner of the custom field.

Only the owner can modify the record. Your name is selected by default.

5. Enter a description of the field.

6. In the **Type** field, select the kind of custom field you want to create.

   For more information on custom field types, see **Table of Custom Field Type Descriptions**.

   The type of field you select determines the other options you can set on the page.

   **Note:** If you later edit the field and change the type, any data stored for the field is deleted.

   If you change the field type or the list/record for a field that is used for sourcing or filtering on other custom fields, all sourcing and filtering based on the field is removed.

7. If you have chosen **List/Record** or **Multiple Select** in the **Type** field, select the list or record in the **List/Record** field that contains the items for the list field.

   **Important:** The displayed value set for the type of field is not strictly filtered by standard restrictions. Be aware that users may select values from records for which they do not have permission. However, they cannot edit the records represented by these values.

8. By default, the **Store Value** option is enabled so that custom field values are stored in your NetSuite account. Clear the box to indicate that any changes entered not be stored in the custom field. Not storing the value enables you to look at data that is stored elsewhere. If you do not store the value, changes will be discarded, so you may want to make the field read-only.
9. Some custom field data types provide a **Use Encrypted Format** option you can enable to encrypt the field values stored in the database.

**Warning:** After a custom field has been saved, the **Use Encrypted Format** setting cannot be changed. Before you set the option, review **Encrypted Custom Field Stored Values** and be sure that you fully understand the consequences of your selected setting.

When you specify that a field be encrypted, any value in the encrypted field displays as **ENCRYPTED** in all accounts except the account where the value was first created.

10. If you are creating a custom item field, to use this custom item as a merchandise hierarchy attribute, check the **Merchandise Attribute** box. When the box is checked, the custom item field is available to apply to items assigned to a merchandise hierarchy. The Merchandise Attribute box only appears on a custom item field and when the Merchandise Hierarchy feature is enabled.

For more information about merchandise hierarchies, see the help topic **Merchandise Hierarchy**

11. Check the **Show In List** box to have the field automatically show in the list of records the field is applied to.

12. You can index the field for global searches if **Store Value** is checked, and if you have chosen any of the following in the **Type** field: Currency, Decimal Number, Email Address, Free-Form Text, Help, Hyperlink, Inline HTML, Integer Number, Percent, Phone Number, Text Area.

You cannot index a custom field for global search if **None** is selected for any **Level for Search/Reporting** option on the **Access** subtab of the custom field record.

After you check the **Global Search** box for a custom field, each global search compares keywords to that custom field’s values, in addition to comparing with record name and ID field values. For more information, see the help topic **Global Search**.

13. If you are creating a List/Record custom field, check the **Record is Parent** box to indicate that the record type selected is a parent record.

The field is used to create a parent-child relationship between two record types.

For more information about parent-child relationships, see **Parent-Child Record Relationships**.

14. If you later want to inactivate the field, check the **Inactive** box. For more information, see **Inactivating a Custom Field**.

15. After you have created a custom field, you should define which record types the field can be used in. See **Assigning Custom Fields to Specific Record Types**.

**Note:** If you are creating an other record field, you do not assign the field to a specific record type. Continue with **Setting Display Options for Custom Fields**.

For custom field types, if you add a data center–specific URL, the URL automatically changes to an account–specific domain for users in their current session. An account–specific domain uses the format **https://<accountID>.app.netsuite.com**, where `<accountID>` is your account ID. The change happens for the following custom field types:

- hyperlink
- rich text
- long text
- free-form text
- text area
- help
- inline HTML

Configuration remains unchanged. You should update all links in custom fields to use your account–specific domain. For more information, see **Account–Specific Domains in Custom Fields**.
Creating a Custom Field

See the following topics.

- Setting Validation and Defaulting Properties
- Setting Sourcing Criteria
- Sourcing and Filtering Examples
- Setting Filtering Criteria
- Multiple Dependent Dropdown Lists
- Restricting Access to Custom Fields
- Restricting Access to Employee Custom Fields
- Creating Read-Only Custom Fields
- Adding Translations for Custom Fields
- Adding Custom Fields to Transaction Forms
- Tracking Changes to Custom Fields

Assigning Custom Fields to Specific Record Types

Custom fields can be assigned to display on specific record types. When the field is assigned to a record type, it is automatically available as a possible field when creating a custom form for that record type. The field will also be available on all standard forms.

For custom entity fields, you must select the record types where the field is available. If a record type is selected, the field is automatically displayed on all forms of that record type, including any custom forms. Custom forms can then be edited to not show the new custom field.

Conversely, all custom transaction fields are available on all custom forms of all transaction types.

To apply a custom field to a record type:

1. Click the Applies To subtab.
2. Check the boxes to indicate the records you want the field to display on.

Note the following:

- When a new transaction is created from a transaction that has a custom field value, the field value is copied to the new transaction.
- Record custom fields can be added to existing and custom subtabs on the entry forms you use to enter records in your NetSuite account.
- Transaction custom fields can be added to the top (body) or the line items (columns) of transactions. When adding a custom field to the body of a transaction, you apply the field to the type of transaction.
- When adding a custom field to the line items of a transaction, you apply the field to the type of line item.
- To apply a custom item field to a portion of the item, check the Select Individual Items box. Then in the Items field, hold down the Ctrl key and click each item to which the custom field applies.
- To specify that the settings for a custom item field or custom transaction item option field also apply to its child items, check the Include Child Items box. Any child item fields added later will also have the parent settings applied.
- When you have the Advanced Employee Permissions feature enabled and you select Employee from the Applies To subtab, you will notice that an Employee Access subtab appears. From this subtab you can select the custom advanced employee permission you want to associate the custom field with. For more information, see the help topic Advanced Employee Permissions Overview.
3. After you have defined a custom field to display on specific record types, you should define the display properties for that field. See Setting Display Options for Custom Fields.

**Warning:** If you assign a custom field to a record type that is a child of another record, that custom field may not always show on the parent record forms. A child record may not be available on a form for a parent record that was created through transformation from another record type. For example, if you define a custom record as a child record of sales order, the custom child record is not available on forms for sales orders transformed from quotes.

**Setting Display Options for Custom Fields**

For each custom field, you can specify the exact location within the form that the field is to be displayed relative to other fields and subtabs on the page.

**To set the display properties of a custom field:**

1. Click the **Display** subtab.
   The fields available on the subtab depend on the kind of field you are modifying.

2. In the **Insert Before** field, select where to place your new field on records.
   The field lists custom fields of the same type that have already been created.

   **Note:** The Insert Before field affects the placement of fields on standard forms and on the placement of newly created fields. To change the arrangement of fields on a custom form, you must edit the custom form.

   You can also choose to have the custom field span the column at the top or bottom respectively. Spanning the column is especially useful to add a text area field at the top or bottom of a tab or section which provides explanatory information.

   **Note:** Fields can span the width of the page, above or below the other fields of that tab or section but they do not have to span the page. The width is determined by the size of the field.

3. In the **Subtab** field, select the subtab you want the field to display on.
   For example, select **Main** if you want the field to display in the top portion of the record.

   **Important:** If a subtab is not selected, the field is automatically displayed on a custom subtab for the record. Select a subtab that makes sense for the type of information the custom field stores.

4. Select the **Display Type**.
   Display types let you to specify how your custom fields behave in NetSuite. You can use display types to make fields for informational purposes only that are not stored in your account. You can also create fields that are not editable or that have default information or custom code calculations.

   Following are the available display types.

   - **Normal**: A normal field can be edited. You can use the field with custom code calculations, defaulting and sourcing information.
     For example, you create a custom field on employee records for a spending limit. The Spending Limit field has a display type of normal because you want to edit and store the information entered in the field.
     Custom transaction fields are available in the custom form setup for all record types. The custom transaction fields are available in the Screen Fields > Custom subtab.

   - **Display Type**: This is the type of display for the field. It can be **Normal**, **Read Only**, **Hidden**, **Display Only**, **Display and Edit**, **Required**, **Optional**, **Conditional**, **Prevent Update**, **Prevent Create**, **Prevent Delete**, **Prevent Write**, **Prevent Read**, **Prevent Sourcing**, **Prevent Defaulting**, **Prevent Calculations**, and **Prevent Transformations**.

   For example, you create a custom field on employee records for a spending limit. The Spending Limit field has a display type of normal because you want to edit and store the information entered in the field.

   Custom transaction fields are available in the custom form setup for all record types. The custom transaction fields are available in the Screen Fields > Custom subtab.
Creating a Custom Field

If you specify normal display for a transaction field, and then hide it on the custom form, it still appears in the custom form setup.

- **Disabled**: A disabled field cannot be edited. You can use the field with custom code calculations, defaulting and sourcing information only. Any field with a display type of disabled that does not have default, sourced or custom code information will not display on forms.

For example, you create a custom field on expense reports for a spending limit. The Spending Limit field on expense reports is sourced from the Spending Limit custom field on employee records. When an employee enters an expense report, they automatically see their spending limit on the expense report. The field has a display type of disabled because you want the information in the field to be updated but not edited.

Note: You cannot disable a mandatory field unless the field has a default value.

- **Inline Text**: An inline text field is for informational purposes only. The information in the field is the result of custom code calculation, defaulting or sourcing information only. Note that a value for a field with an Inline Text display type will be interpreted as HTML markup source and will be evaluated by the browser when a record containing the field is displayed. Inline text fields of the List/Record or Check Box type are not available for use with custom code. These fields are also not available for transaction line fields.

For example, you create a Tax ID custom field for your purchase orders. When creating the custom field, you can enter your tax ID in the Default Value field. Your Tax ID then appears on purchase orders. The field has a display type of inline text because your tax ID is for informational purposes only, and you do not want the information stored with each transaction.

- **Hidden**: A hidden field cannot be seen on the record or transaction you apply it to. You can perform a search to display the value of the field. The information in the field is the result of custom code calculations and defaulting information. You must use custom code or set a default for the field. Note that you can also define a field as shown in its custom field definition, and then selectively hide it on a form-by-form basis.

If a custom field is marked as hidden, it is not available to add to custom forms, and is not available in the Fields list when customizing the form.

For example, you can use a hidden field to store your support representatives case quota. The support representatives do not see the field but the information can still be searched and reported on. For more information on custom code, read the help topic Working with Online Forms.

Be aware that in SuiteScript, only user event, scheduled, and Suitelet scripts can set the value of a custom field that has a display type of hidden.

Note for custom transaction body and custom transaction line fields: If the Display Type field is set to Hidden, the field is still included on all forms, even if it is not displayed. The system ignores any changes you make to show or hide the field on all forms that apply to that custom transaction body or custom transaction line field.

Warning: Hiding a custom field is a display convenience only and is not field level security. Hidden custom fields are embedded in the page output and can be viewed in the page source.

5. Enter field height and width properties.

The values entered for height and width affect the display size of the field only. The permitted size of the custom field (number of pixels or lines permitted) is controlled by default options. For more information on defaulting options, see Setting Validation and Defaulting Properties.

- **Height**: Enter a height for the custom field in number of lines. The default is three lines.
- **Width**: Enter a width for your custom field in columns.
Creating a Custom Field

6. If desired, in the **Link Text** field, enter text to display instead of a URL for hyperlink fields. Users see the text instead of the URL on records and transactions. Clicking the text links to the URL.

7. (Decimal Number, Integer Number, and Percent fields only) Review the **Apply Formatting** setting and change if desired.
   - When the preference is enabled, the custom field's values use the formatting set in the Number Format and Negative Number Format preferences at Company, Subsidiary, or User level. For details about these preferences, see the help topic Number Formats.
   - When the preference is disabled, the custom field's values are not formatted, meaning they have no commas or decimal points. (The setting is appropriate for Integer Number type fields used as unique identifiers.)

8. In the **Help** field, enter a brief description of the kind of information you want entered in the field. The help is available when the name of the field is clicked.
   You can enter HTML in the field. Be sure to begin your markup source with `<html>` and end with `</html>`.

9. In the **Label for Input** field, enter the label for the item option as you want it to appear to customers on your website.

10. If the custom field is of type list/record or multi-select, and the selection in the List/Record list is hierarchical, a Show Hierarchy box is available.
    - To show the record hierarchy, or the full name, check the Show Hierarchy box. To show the field name without the hierarchy, or the short name, clear the Show Hierarchy box.
    - In the case of accounts, when the Use Account Numbers preference is enabled, custom account fields display the account numbers together with account names.

11. After you have set the display properties, you should define any validation and defaulting properties. See Setting Validation and Defaulting Properties.

---

### Setting Validation and Defaulting Properties

Validation options are constraints that can be placed on your custom fields to help control the information that is entered in the field. Defaults are values you specify for your custom fields that display populate automatically when a record or transaction is first created.

**To set validation and defaulting properties:**

1. Click the **Validation & Defaulting** subtab.
2. Set the validation options. (These options may vary according to the kind of custom field selected and its data type.)

Possible options include the following:

- **Mandatory**: Enable to require information to be entered in the custom fields before a record or transaction can be saved.

  For example, you can create a mandatory contact number field for your sales transactions. When sales representatives enter transactions, a contact number for the customer must be entered before the transaction can be saved.

  **Note:** You can also set fields to mandatory when customizing a form. When using the same custom field on different forms, consider applying required field settings in your custom form rather than on the field itself. Applying the setting at the form level is useful in cases where the field information is required on one form but not on another form.

- **Minimum Value**: Set the minimum number that can be entered in the custom field. A record or transaction cannot be saved with a value below the minimum set here.

  For example, you can create a sales quota field for your employee records. If you set a limit of $500.00 when creating the custom field, any amount less than 500 cannot be entered in the field.

  You can set a minimum value for a percent field that is less than 0. If you do not enter a minimum amount for a percent field, the minimum is 0.

- **Maximum Value**: Set the maximum number that can be entered in the custom field. A record or transaction cannot be saved with a value above the maximum set here.

  For example, you can create a spending limit custom field for your employee records. Set a limit of $700.00 when creating the custom field to ensure that no employee can receive a spending limit of more than $700.00.

  You can set a maximum value for a percent field that exceeds 100. If you do not enter a maximum value for a percent field, the maximum is 100.

- **Maximum Length**: Set the maximum number of characters that can be entered in the custom field. A record or transaction cannot be saved if the information entered in the field exceeds the number of characters permitted.

  For example, you can create a gift message field for your sales transactions to record a special message from your customers. You can limit the number of characters to prevent messages from becoming too long.

- **Allow Delete of List/Record Values**: Indicates how the system handles deletions of records referenced by the custom field. This field is available for custom field definitions that have a type of List/Record or Multiple Select and have the Store Value box checked.

  - Select **Prevent and Return Error** to prevent deletions and return an error message stating that the record could not be deleted due to dependencies and providing a link to dependent records.
Select **Allow and Set Dependent Field Values to Null** to permit deletions, null out field values that use the deleted record, and log system notes for deletions.

For more information, see [Customizing Delete Behavior for Records Referenced by Custom Fields](#).

**Formula**: Enable to validate SQL formula expressions when the field is defined as a formula field. For more information on defining a custom formula field, see [Creating Formula Fields](#).

**Search**: Select a search with summary results to be used to calculate a value for the custom field. Search fields display a rollup value for a selected search results field. The value is dynamically calculated each time a form containing the summary search field is displayed. For example, you can display the total quantity of all line items on a transaction.

The Search option is only available for kinds of custom fields and data types that support summary search derived values. For details, see [Creating Custom Fields with Values Derived from Summary Search Results](#).

**Field**: If you want values for the custom field to be based on summary search results, you can optionally select a comparison field to join related records in cases where you want to put the custom field on a form for a record type that is different from the summary search record type. For example, if you want to put a custom entity field showing the result of a customer record summary search on an employee custom form, you could select an employee record field whose values could be matched to the values for the search’s Available Filter field. Search results field values for all records with matching values for the Compare To field and Available Filter field would be used to calculate the value of the summary search custom field.

The Field option is only available for kinds of custom fields and data types that support summary search derived values. For details, see [Creating Custom Fields with Values Derived from Summary Search Results](#).

3. If desired, enter default parameters for the field.

To set values into an Inline Text or Disabled field, you **must** specify a default value or source the information for the field from another field.

Depending on the field type, various default values can be specified. Possible options include:

- **Default Checked**: Check the box to indicate that the check box custom field should display checked by default. The custom fields can still be cleared on individual transactions and records.

  For example, you can add a Subscribe to Newsletter check box to your customer records. When you set the field to default checked, new customers are automatically subscribed to your newsletter.

  Custom transaction line fields have a limitation that occurs when several conditions are met. For more information, see [Custom Transaction Line Fields](#).

- **Default Value**: If desired, enter a value to display in the field by default. The value can still be changed on individual records and transactions if the field is not locked.

  When working with Free-form text, Text area, Rich Text or Hypertext fields, you can include NetSuite tags in the default definition. These tags are populated with field values when the page is loaded or saved. For more information, see [Dynamic Defaults and Dynamic Hyperlinks](#).

  For example, you can add a spending limit field to your employee records with a default value of $150.00. When you enter new employee records, the field automatically fills with the amount.

  Default values defined here are only applied at creation time for any specified record. After a record has been created, subsequent edits to that record store the previously stored value unless it also has been edited.

- **Dynamic Default**: Dynamic default enables you to select from preset defaulting options specific to the kind of field you are creating. There are six types of dynamic defaults:
Creating a Custom Field

- **Current Date/Time**: For Date fields, your custom field is automatically filled with the current date or time. For example, you can add an information only date field to your sales orders. Setting the field to current date and time lets you track when your sales representatives are most productive.

  **Important**: The date and time for the field are based on the user’s NetSuite Time Zone preference, set at Home > Set Preferences, not on the browser client time zone.

- **Current User**: For Entity List/Record fields, have the name of the entity entering the record or transaction automatically filled in the field. For example, if you have data entry employees as well as sales representatives, you can add a field to your transactions to record who enters the transaction as well as the sales representative who made the sale.

- **Current User’s Supervisor**: For Entity List/Record fields, have the name of the supervisor selected on the entity record automatically filled in the field. For example, you can create a field for your task records that lists the Assignee’s supervisor, helping your management team stay informed of team projects.

- **Current User’s Department**: For Entity List/Record fields, have the department of the entity entering the record or transaction automatically filled in the field. For example, if you have data entry employees in various departments, you can add a field to your transactions to record the department of the employee who enters the transaction.

- **Current User’s Location**: For Entity List/Record fields, have the location of the entity entering the record or transaction automatically filled in the field. For example, if you have data entry employees in various locations, you can add a field to your transactions to record the location of the employee who enters the transaction.

- **Current User’s Subsidiary**: For Entity List/Record fields, have the subsidiary of the entity entering the record or transaction automatically filled in the field. For example, if you have data entry employees in various subsidiaries, you can add a field to your transactions to record the subsidiary of the employee who enters the customer.

  - **Default Selection**: Set a selection list to display in the custom field by default. The choices are limited to the list selected in the List/Record field when creating the custom field. For example, you can create a custom field to record advertising preferences for your customers. You can set a default of Email if you know that the majority of your customers prefer to receive ads by email. When a customer loads the page, Email is displayed in the selection list by default but they can choose Fax or Mail if desired.

4. After you have entered validation and defaulting properties, you should set any sourcing criteria. See Setting Sourcing Criteria.

Customizing Delete Behavior for Records Referenced by Custom Fields

List/record and multiple select type custom fields provide a list for users to select values. The list of values is populated by records of the list/record type set in the custom field definition. Because list/record and multiple select custom fields are dependent on these referenced records, deletion of these records can be problematic. For example, a custom field called Color can be dependent on a custom record type called Color List. If a value is deleted from Color List, for example Purple, any records with a Color custom field value of Purple are impacted.

For custom field definitions that have a type of List/Record or Multiple Select and have the Store Value option enabled, the **Allow Delete of List/Record Values** option is provided. The Allow Delete option
permits overrides of the default system behavior when a delete is attempted of a record referenced by values in the custom field.

Default settings for the option are based on the record type selected in the List/Record field:

- For entity, item, event, and transaction type records: default is **Allow and Set Dependent Field Values to Null**.
  
  When a delete of a referenced record sets a dependent custom field value to null, a system note is logged on the record containing the dependent custom field value. The note specifies the user who deleted the referenced record as the Set by value, and Unset as the Type.

- For other record types (including custom records): default is **Prevent and Return Error**.
  
  When a delete of a referenced record is prevented due to dependent custom field values, the error message includes a link to a page listing the dependent records. The Dependent Records page includes the name of the referenced record at the top of the page. The list on the page includes a line for each dependent custom field, with the following details: kind of custom field, name of custom field, and name of record containing the custom field with a clickable link to the record.

Note the following:

- The **Allow Delete of List/Record Values** setting is not available for Workflow or Workflow Action custom fields.

- The **Allow Delete of List/Record Values** setting for a custom field takes precedence over whether the field is mandatory, so a setting of **Allow and Set Dependent Field Values to Null** can produce nulled values even for mandatory fields. To avoid nulled values, choose a setting of **Prevent and Return Error** for mandatory fields.

- SuiteBundler does not respect the **Allow Delete of List/Record Values** setting. If a referenced record is deleted as part of a bundle operation, such as uninstall, the treatment of dependent custom field values conforms to SuiteBundler rules.

**Setting Sourcing Criteria**

A custom field can source information from another record in your account. The information populated into the custom field is then dependent on fields associated with a record selected on another field within
that form. Sourcing enhances your NetSuite forms by reducing data-entry errors and ensuring that your customers and employees always have the most current information.

You can source from both standard and custom fields.

For example, two custom fields — **Sales Rep** and **Sales Rep Email** — are placed on a custom case form. When a company record is selected in the Company field, the sales representative already defined in the selected company record is sourced to the **Sales Rep** field on the case form and the **Sales Rep Email** field defaults to the email address defined for the sourced Sales Rep.

**Note:** The information is sourced into the custom field only when the record is created or if the specific fields involved are altered when editing the record. In the example above, if you change the sales representative selected on the customer, the sourced field would change to show the email address of the new sales representative.

When setting up sourcing, you have the option to store the value. When the field is not stored, the information is not saved in the custom field. A custom field that does not store the value enables you to look at data that is stored elsewhere when you are setting up searches and reports.

By storing a sourced field, the sourcing will automatically fill the field with a value when the master field is changed. You can then change the value of the custom field. The value is stored independently and has no impact on the source field, so any changes made in the custom field are not updated in the source.

**To set sourcing and filtering criteria:**

1. Edit the custom field you want to add sourcing and filtering criteria to.
2. If you do not want to store the value, clear the **Store Value** box. In most cases you do not want to store the value.
3. On the Display subtab, check the **Display Type** field. If the **Display Type** field is set to **Hidden**, sourcing does not occur.
4. Click the **Sourcing & Filtering** subtab.
5. From the **Source List** list, select the field that references the record you would like to source information from.

For example, you are creating a custom field to appear on the customer record to show the email address of the sales representative assigned to the customer. In the **Source List** field, select **Sales Rep**.

When entering sourcing information for transaction body or transaction line fields, the **Entity** field varies depending on the transaction type. For example, on an expense report, entity means employee, but on a purchase order, the entity field is vendor. For a purchase order, select **Employee** in the Source List field instead of **Entity** to obtain the desired results.

When working with entity fields, you can also define the field to source from a field on the parent record by selecting **Parent** from the **Source List** list.

**Note:** You **cannot** source information for a Multiple Select field type.

6. From the **Source From** list, select the field you want to source from. The source from field is found on the record you selected from the **Source List** list.
Creating a Custom Field

Any fields available on the record you select in the Source List field can be selected from the Source From list.

In the example above, select Email from the Source From list.

The value stored in the field selected here populates the field when the record is selected. The field selected here must be consistent with the type selected for the custom field. For example, if you select E-mail as the field type and then select an address field from the Source From list, you receive an error.

**Note:** If your field is a List/Record field, the field selected for the Source From field must be in the record type selected as the List/Record. Sourcing from body fields to line fields is not supported in SuiteScript 7.

7. If your field is a List/Record field, you can filter the choices that can be selected.

When a List/Record Type field is being defined, you can choose to populate the custom field with values that meet specific parameters in the sourced list or record. First, select the desired item to filter by from the Source Filter by list. Then select an item from the Source List and, optionally from the Source From lists. When you select an element from the source list, it will fill the option-sourced custom field with all elements where the source filter by field matches the source list (or the source from field of the source list).

The record you are sourcing from must be associated with the type of record you want to appear in your custom field.

**Note:** The field selected from the Source Filter by list must be in the record type selected as the List/Record.

For more information, see Sourcing and Filtering Examples.

8. After you have set the sourcing criteria, you should set any filtering criteria. See Setting Filtering Criteria.

**Note:** A custom field with a sourcing relationship is not available for mass updates or inline editing. See the help topic Mass Changes or Updates.

### Note about Custom Transaction Line Field Sourcing

Before Version 2012.2, you could source a custom transaction line field's values from a body field by selecting <Record_Name> (Line) in the Source List list on the Sourcing & Filtering subtab of the Transaction Line Field page. Now, to source a field's values from a body field, you need to select <Record_Name> in the list. If you select <Record_Name> (Line), sourcing is from a field in the sublist and if there is no such field, values for the sourced custom transaction line field are blank.

The following example shows a custom field applied to the Items sublist on Sales Order records. Because the sublist does not include a Customer field, if Customer (Line) is selected, the values for the new custom field will be blank. Selecting Customer sources the new custom field's values from the Customer body field on Sales Orders.
Sourcing and Filtering Examples

Following are some examples of how you can use the Source Filter by field to create dynamic custom fields.

Example 1: Linking Two Transactions

Suppose that you want to link two transactions, such as an invoice and a subsequent credit.

First, add a transaction custom field to customer credits (using a custom form to limit it only to the credit form). The field should be a List/Record Transaction and source Entity in the Source List and Source Filter by fields. When you select a customer on the credit memo, it populates the new list field with only invoices from that customer.
Example 2: Combining Static and Dynamic Filtering

Suppose that you want to create a Linked Order field on a Sales Order that lets you choose another Sales Order from that customer.

First, add a transaction custom field to sales order forms. The field should be a List/Record Transaction and source Entity in the Source List and Source Filter by fields. Add a static Filter to filter the list to only transactions of a particular type (Sales Orders). When you create a new sales order, select a customer, and the Linked Order list populates with sales orders from that customer.
Example 3: Filtering Against the Source From Value

Suppose that you want to have a field on a task record that lists all of the subordinates of a sales representative associated with a selected opportunity.

First, add a custom field to a Task record. The field should be a List/Record Entity (for example Employee). Since you want only the Sales Reps associated with the selected opportunity, define the field Source List as Opportunity. Then filter the Source From by Sales Rep and the Source Filter by as Supervisor. The resultant List filters to cases where the Source Filter By value (Supervisor) equals the Sourced From value (Sales Rep).
Creating a Custom Field

Setting Filtering Criteria

When creating a list/record or multiple select custom field, you can filter the choices available in that field on records and transactions based on selections made in other fields. Filtering enables you to tailor the exact choices offered to users entering records and transactions.

You can filter based on the selections in multiple list fields. For more information, see Multiple Dependent Dropdown Lists.

**To filter a list/record or multiple select custom field:**

1. Click the **Sourcing & Filtering** subtab.
2. In the **Filter Using** field, choose a field to filter on.
   - The field you choose here is a field on the record you selected in the **List/Record** field.
   - The field selected limits the results according to the filter criteria you define. For example, to limit an employee list field to only show sales representatives, you select the **[Is] Sales Rep** field in the Filter Using column, since that is the name of the field on the employee record.

   **Note:** If you specify two or more filters, the custom field uses a popup list, not a dropdown list.

3. If the field chosen in the **Filter Using** column is a check box, check the **Is Checked** box to show only records with that box checked.
In the sales representative example, check the box in the Is Checked column.

4. In the Compare Type column, select how you want the information compared to the criteria you set.

For example, select equals to ensure that the information you set as criteria matches the selections available in the list exactly.

5. In the Compare Value to column, enter the value you want to filter the list by.

For example, if you select State in the Filter Using field, enter GA to filter the list to only records with the state listed as Georgia.

6. If the field chosen in the Filter Using field is a list, choose the value you want to show in the Value Is column.

7. Check the Is Not Empty box to include all records with a value entered in your filter field.

The Is Not Empty option is not available for check box fields.

8. Check the Is Empty box to include all records with no value entered in your filter field.

9. In the Compare To Field column, choose which field on the record selected in the List/Record field you want to compare to the field in the Filter Using column.

10. Click Add/Edit.

11. Add all required filters to the custom field.

   The more filters you add, the fewer choices are offered in the field. Each selection must match each filter to be included.

12. Click Save.

Next, you can define who has access to your custom field. For more information, see Restricting Access to Custom Fields.

**Warning:** Custom field filters are type sensitive. If you change the type of a custom field, filters defined for the field may become incompatible, resulting in unexpected errors when a form containing the field is displayed. If you plan to change a custom field’s type, you need to review any existing filters defined for the field, and remove or change these filters to avoid errors. For information about different custom field types, see Table of Custom Field Type Descriptions.

### Multiple Dependent Dropdown Lists

You can filter the choices in a dropdown list based on the selections in one or more other fields.

See the following example.

- You add two custom transaction line fields to a transaction form and add an additional field that is filtered based on the selections in the other two.

Setting up these field dependencies is accomplished through the use of the Compare To Field column on the Sourcing & Filtering subtab on the custom field page. With the Compare To Field column, you can compare values entered in multiple fields on a transaction or record with the values defined on a custom record.
Creating a Custom Field

The following example illustrates the use of multiple dependent dropdown lists.

A company sells t-shirts of multiple sizes (small and large), colors (black and white), and styles (v-neck and crew). The company sells any combination of size and color, but the selection of styles is limited to a set of color-size combinations. For example, the company does not sell large, black, v-neck shirts, but it does sell large, black, crew neck shirts.

Color and Size are list/record transaction line fields that refer to custom lists. Shirt Style is a custom record type that has the Color and Size fields. There is also a Shirt Style transaction line field that refers to the list of Shirt Style custom records.

To understand how the feature works, it helps to think in terms of controlling versus dependent fields. **Controlling** fields are the fields that are used to determine the selections that are available in **dependent** fields.

The Style transaction line field is filtered according to what is selected in the Color and Size transaction line fields. Color and Size are the **controlling** fields; Style is the **dependent** field.

First create custom lists for color and size.

Next, you create a custom record type with two fields – **Size** and **Color** – that refer to the corresponding custom lists.

Next, you create a custom record for each **Style** combination you sell. These custom records are named:

- large, black, crew
- small, black, v-neck
- large, white, crew
- small, white, crew
- large, white, v-neck
- small, white, v-neck
Finally, you create three custom transaction line fields:

- **Color** – Of type List/Record with your color list selected in the List/Record field
- **Size** – Of type List/Record with your size list selected in the List/Record field
- **Style** – Of type List/Record with your style custom record selected in the List/Record field

For the **Style** field, on the Sourcing & Filtering subtab, in the Filter Using column, select **Color** – the field on your custom record that references the **Color** custom field. Set the Compare Type to **equal**, and select Color in the Compare To Field column. Repeat for the **Size** field.

Now, the Style field is filtered by the selections in the **Size** and **Color** fields.

To create a field that is filtered by multiple other List/Record fields, complete the following steps:

**First, create the custom lists and record type:**

1. Create custom lists for each of the controlling fields.
2. Create a custom record type for the options you want to be available in the dependent field. The custom record must contain List/Record fields that reference the custom lists you created a moment ago for the controlling fields.
3. Create a custom record for each option you want to be available in the dependent field.

**Next, add custom fields to the transaction or record:**

1. Create custom fields for each of the controlling fields.
2. Create the same kind of custom field of type List/Record with the custom record type from step 2 selected in the List/Record field.

On the dependent field record:
   a. On the Sourcing & Filtering subtab, in the Filter Using column, select one of the controlling fields.
   b. In the Compare Type field, select the qualifier you want to use to determine the filtering. For example, if you want the selections in the fields to match, select equal.
   c. In the Compare To column, select the field on your custom record that refers to the field you selected in the Filter Using column.
   d. Repeat for each controlling field.
   e. Click Save.

You can create multiple layers of dependencies. In the previous example, you might have another field that is filtered based on your selection in the Style field. You will need a custom record type for each dependent field as you created for the Style.

Restricting Access to Custom Fields

You can control who can access the information in custom fields, enabling you to maintain the security of your business information. The access you define determines how it can be accessed both on the record as well as through search results and reports.

**Note:** When the Advanced Employee Permissions feature is enabled, use the Employee Access subtab to assign custom fields to custom advanced employee permissions. For more information, see Restricting Access to Employee Custom Fields.

Access to a field can be based on role, department, or subsidiary. The following custom access levels can be assigned to each department and subsidiary.

- **Edit** – The field and its contents can be viewed and changed.
- **View** – The field can be seen, but its contents cannot be changed. (The permission level affects how the form is accessed on records.)
- **Run** – The field can be seen through reports and search results, but its contents cannot be changed. (The permission is only applicable to reports and search.)
- **None** – The field cannot be seen, and its contents cannot be changed.

For cases when different access levels are defined for a user’s role, department, or subsidiary, the highest level of access is granted. For example, an employee is assigned to a department that has Edit access to a custom field, and the employee’s role has been granted View access. The employee has the higher level of access – in the preceding example, Edit access.

In addition to search and reporting, the access level granted to a custom field includes instances where it is referenced by online forms, mail merge operations, and when it is sourced by other custom fields, or referred to by formula fields.

**Note:** If you take away the administrator role’s access to a custom field, the field will not be accessible to scripts that are run by an administrator. To access the field via scripting, you must edit and restore administrator access to the field.

You can set the level of access you want to grant by default to custom fields. The default access level applies to the roles, departments, and subsidiaries, that you do not define on the Role, Department, and Subsidiary subtabs.
To set default access, edit the custom field record, and click the Access subtab. In the Default Access Level, set the level of access you want to give by default. In the Default Level for Search/Reporting field, select the level of access you want to give through search and reporting.

The access you define on the Role, Department, and Subsidiary subtabs overrides the default access levels.

**To set role, department, or subsidiary access restrictions:**

1. Edit the custom field record.
2. Click the Access subtab.
3. In the Default Access Level field, set the access level you want to grant to roles, departments, and subsidiaries that you do not specifically define below.
4. In the Default Level for Search/Reporting field, set the level of access you want to grant through search and reports to roles, departments, and subsidiaries that you do not specifically define below.
5. Click the Role, Department, or Subsidiary subtab.
6. In the first column, select the role, department, or subsidiary you want to define access for.
7. In the Access Level column, select the level of access you want to grant.
8. In the Level for Search/Reporting column, select the level of access you want to give the role, department, or subsidiary via search and reporting.
9. Click Add.
10. Repeat these steps for each role, department or subsidiary.
11. Click Save.

**Important:** The above procedure describes how to limit access to a custom field. You also can check the Apply Role Restrictions box to limit access to an entire custom record, according to a custom field’s values. Applying role restrictions extends access restrictions based on class, department, location, or subsidiary, that are set on role definition pages. For example, if you have set restrictions for roles based on locations, and you want to apply these same restrictions to Subscription custom records, you can check Use Role Restrictions for the Subscription record type’s custom field that stores location values. See Applying Role-Based Restrictions to Custom Records.

**Access Level History**

For auditing purposes, you can view any changes that have been made to custom field access levels.

To view custom field access changes, open the custom field record. Click the Access subtab, and click the History subtab.

You can view the date and time a change was made, the user who made the change, and the changes that were made.

**Bundling Fields With Access Restrictions**

If you include a custom field in a bundle that has access restrictions, custom roles that have access are not automatically included in the bundle. If, however, you include those custom roles in the same bundle, the access restrictions are preserved.

The access level assigned to standard roles is preserved when you bundle a custom field that has access restrictions.
Custom field restrictions based on subsidiary or departments are not carried over into the target account since departments and subsidiaries cannot be included in a bundle.

Restricting Access to Employee Custom Fields

You can control who can access the information in employee custom fields, enabling you to maintain the security of your business information. The access you define determines how it can be accessed both on the record, as well as through search results and reports.

An Employee Access subtab is available on the Custom Entity Field page when the following conditions are met:

- The custom entity field is applied to the employee record, that is, the Employee box on the Applies To subtab is checked.
- The Advanced Employee Permissions feature is enabled.

When the preceding conditions are met, access to the custom field is defined by the configuration on the Employee Access subtab.

When the preceding conditions are not met, access to the field is defined by the configuration on the Access subtab.

To restrict access, edit the custom field record, and click the Employee Access subtab. In the Permissions list, select the appropriate permissions to restrict access to the custom field. Your organization might have custom permissions that you can choose. The standard permissions are:

- Employees and Employee Administration — Uses the restrictions defined on the Role page.
- Employee Confidential — Restricts access to directs reports and below.
- Employee Public — Restricts access to active, non-terminated employees.
- Employee Self — Restricts access to self.

For more information, see the help topic Advanced Employee Permissions.

Creating Read-Only Custom Fields

You can create custom fields that display information but do not store that information with any record or transaction. You can create a read-only field of any type with any display, default, validation or sourcing options.

A read-only field is especially useful when used with information sourcing. Fields that use sourcing show information that is stored elsewhere in NetSuite. That information is retrieved from the source every time the page is loaded. Since any changes users make to the field are not saved, setting the field to read-only prevents user from attempting to edit it.

To create a read-only custom field:

1. Either edit a custom field, or create a new custom field.
   - If you want to create a new field, see Kinds of Custom Fields to determine the type of field you want to create and find step-by-step instructions for each type of custom field.
2. Clear the Store Value box.
3. Click the Display subtab.
4. Set the Display Type field to Disabled.
5. Click **Save**.

For more information on using display types, see **Setting Display Options for Custom Fields**.

### Adding Translations for Custom Fields

If the Multi-Language feature is enabled on your account, you can translate the label and field level help for a custom field so that they match the language of the NetSuite user interface. For details, see the following:

- Translating a Custom Field Label
- Translating Custom Field Help

**Important:** Before you can add these translations, you need to select translation languages at Setup > Company > General Preferences, on the Languages subtab. The Languages subtab lists both system-supported languages that can be used for the NetSuite user interface (and are available at Home > Set Preferences), and additional languages that can used for website translations only (and are not available at Home > Set Preferences). You should only enter translations for system-supported languages, because these are the only languages that can be displayed in the user interface. For details, see **Configuring Multiple Languages**.

### Translating a Custom Field Label

You can define translations for a custom field label on the Translation subtab of the custom field page.

![Custom Field Translation Subtab](image)

### Translating Custom Field Help

You can define translations for the Field Level Help of a custom field on the Translation tab of the custom field page. Field level help is available when the name of the field is clicked. You can enter HTML in the field by beginning the markup with `<html>` and ending it with `</html>`.
Adding Custom Fields to Transaction Forms

After you create a custom transaction field, it automatically appears on standard forms. However, it does not automatically appear on custom forms.

**To add a custom transaction field to a custom form:**

1. Go to Customization > Forms > Transaction Forms.
2. Click **Edit** next to your custom form.
3. On the Custom Transaction Form page, click the **Screen Fields** subtab.
4. Click the subtab you specified on the custom field record and locate your custom field name in the **Description** column.
5. Check the box in the **Show** column next to your custom field if you want the field to show when the transaction is being entered.
6. Click **Printing Fields**.
7. Click the subtab for the area of the printed form your field appears in.
   - For body fields, click **Body**.
   - For column fields and item options, click **Columns**.
8. Check the box in the **Print/Email** column next to your custom field if you want the field to show up when you print or email the form.
9. Enter a label for each field as you want them to print.
10. When you have finished, click **Save**.

Your custom transaction field now shows up on your custom transaction form.
To use a custom transaction form, go to the Transactions page and click the link for the type of transaction you want to enter. When the page appears, select the appropriate form in the Custom Form field.

For more information, see the following topics.

- Adding Subtabs to a Custom Record
- Customizing Transaction Form PDF Layouts
- Transaction Form HTML Layouts

**Tracking Changes to Custom Fields**

In the History subtab for each custom field, information on every saved change is displayed with the following summary information:

- Date/Time
- User
- Label
- Field Type
- Record Type (if Type is List/Record)
- Store Value

The System Notes subtab displays information about any saved setting change for the custom field:

- Date/Time
- User
- Context — how the change was made
- Type of change
- Field changed
- Old value
- New value

**Note:** History and system notes currently are not supported for custom item number fields or custom segments.

**Inactivating a Custom Field**

If required, you can remove a custom field from a specific record type or delete the field completely. Another alternative is to make a custom field inactive instead of deleting it. Data and associated forms for an inactive custom field are maintained in NetSuite, which is useful in cases where you may need to use a custom field again, or you simply want to preserve custom field data in the system.

When a custom field is inactive, it no longer appears on any forms, it is not available in searches, and it is not available to SuiteScript or SuiteAnalytics Connect, the same as a deleted field.

You can make a custom field inactive on the record page for the custom field or in a list of custom fields:

- On a Custom Field record, check the **Inactive** box to make a custom field inactive.
- A **Show Inactives** box is available on all custom field list pages, including the list of custom fields on each custom record type's Fields subtab. By default, the **Show Inactives** box is unchecked, so that inactive custom fields are filtered out of lists. When Show Inactives is checked, the list displays both inactive and active custom fields, and each custom field in the list has an **Inactive** check box next to it. You can check the box to make a custom field inactive, and clear the box to reactivate an inactive custom field.
When a custom field is made inactive, it no longer appears on any forms or reports and it is not returned by global search – it is not available anywhere, the same as a deleted field. However, data and associated forms for an inactive custom field are maintained in NetSuite, whereas a deleted field is completely removed from the system. If an inactive field is later made active again, all of its data is restored, and the field appears on all of the same forms as before it was made inactive.

**Note:** Some settings are not maintained for inactive fields, including display formatting. Also, NetSuite does not check to see if custom fields are referenced in SuiteScripts. If you inactivate a custom field, update any SuiteScripts that reference the field, otherwise they will not work.

A custom field cannot be made inactive if other NetSuite records depend on it. For example, a custom field cannot be made inactive if any of the following conditions apply:

- It is sourced by another custom field.
- It is used in lead conversion mapping.
- It is used in the Parameter, Values section for a workflow action.
- It is used in a workflow definition condition.
- It is used in a filter or a condition for a saved search.

In addition to the above examples, there are other dependencies that may prevent you from making a custom field inactive.

When you try to inactivate a custom field that has dependencies, an error message is returned, with a link to a Dependent Records page, where you can review details about these dependencies.

You can use the Check Inactivate Dependencies option under the Actions menu on each custom field record to check for dependencies before you try to make the custom field inactive. Another option is available to check dependencies that may prevent you from deleting a field. When you do one of these checks, a Dependent Records page appears. If the list has no records, you can make the custom field inactive or delete it without errors.

**Note:** Script custom fields cannot be made inactive.

**Note:** If a bundle contains an inactive custom field and in the target account that field is active, and you update the bundle, the custom field remains active in the target account after the bundle update. If the bundle contains an active custom field and in the target account that field is inactive, the field is changed to be active in the target account. For more information, see the help topic Bundle Update Reference.

## Editing a Custom Field

You can modify custom fields.

If you are viewing a form and want to know if a field is custom, open the field level help and find the field ID.

If the field ID is not visible in the help, you need to show the field IDs in your account.

For more information, see the following topics.

- Getting Field-Level Help
- How do I find a field's internal ID??
  - Showing Record and Field IDs in Your Account

If the field ID begins with one of the following, the field is a custom field.

- custevent (CRM)
To edit a custom field:

1. Go to Customization > Lists, Records, & Fields > [Custom Fields], where [Custom Fields] is the type of custom field you want to modify. The Custom Fields page appears, listing all custom fields configured for that field type.

2. Click the name of the custom field you want to modify. The Custom Field page is displayed for the selected field. You can modify the following:
   - Label
   - Field ID
   - Field Owner
   - Description
   - Type
   - Whether to store values for the field
   - Whether to use an encrypted format
   - Whether to have the field automatically shown in the list of records the field is applied to
   - Whether to index the field for global searches

3. To make other changes to the custom field, select the appropriate subtab:
   - Display — Specify the size of the field and its exact location on the form relative to other fields and subtabs on the page.
   - Validation & Defaulting — Restrict the information that is entered in the field, and specify values that display automatically when a record or transaction is first created.
   - Sourcing & Filtering — Source information for the field from another record in your account, or tailor the choices available when entering records and transactions.
   - Access — Specify who can access the information in the field.
   - Employee Access — If Advanced Employee Permissions is enabled for your organization, specify access restrictions for custom employee fields on the Employee Access subtab.
   - Translation — Translate the label and field level help for the field.

4. When you are finished editing the field, click **Save**.

When making changes to the settings for custom fields, the following options are available:

- Renaming Custom Fields
- Mass Updating Custom Fields

**Note:** To edit a custom field in a custom record type, open the custom record type and select the custom field from the **Fields** subtab. For more information about the **Fields** subtab, see Adding Fields to Custom Record Types.

Account-Specific Domains in Custom Fields

As of 2020.1, NetSuite supports account-specific domain URLs in custom fields. You should use account-specific domains instead of data center-specific URLs.
Previously, NetSuite domains were specific to the data center where your account was hosted. Links in custom fields used these data center–specific domains. NetSuite domains are no longer specific to data centers. Instead, they are account–specific domains, which are domains specific to your account, not to the data center where they are hosted. For more information, see the help topic URLs for Account-Specific Domains.

For users, the account–specific domain feature translates current data center–specific URLs to the appropriate account–specific domain URLs. This translation happens for hyperlink, rich text, long text, free-from text, text area, help, and inline HTML custom field types. For example, a system.eu2.netsuite.com link in your custom field type becomes <accountID>.app.netsuite.com, where <accountID> is your account ID.

The automatic translation from data center–specific URLs to account–specific domain URLs is applied to your active session. Configuration remains unchanged. Links that use account–specific domains are faster than links translated in the current session. You should update all links in custom fields to use your account–specific domain.

For more information, see the help topic How to Transition from Data Center-Specific Domains.

Renaming Custom Fields

When you change the name of a custom field, the name is not automatically updated on custom forms that contain the field.

To apply the new name to custom forms:

1. After making changes to your custom field, click Apply to Forms.

2. On the Apply Custom Field to Forms page, change the field label for each form where you want to reflect the change.

If no forms are listed, you might have the Display Type field set to Hidden, which means that the field will not be shown on any form.
3. Click **Save**.

Mass Updating Custom Fields

To be available for mass update, a custom field must meet the following criteria:

- It must support inline editing. For more information, see the help topic Using Inline Editing.
- It must be displayed on your preferred form for the record type being updated.
- It must be stored.
- It must not have a sourcing relationship.
- It must not be encrypted.
- It must not be an Email custom field.
- If it is an Item custom field, the field script ID must not end with ‘description’.

For more information about performing mass updates, see the help topic Mass Changes or Updates.

Advanced Features for Custom Fields

You can tailor custom fields for your organization by using the advanced features available.

- Encrypted Custom Field Stored Values
- Creating Custom Fields with Values Derived from Summary Search Results
- Dynamic Defaults and Dynamic Hyperlinks
- Creating Formula Fields
- SQL Expressions

Encrypted Custom Field Stored Values

As of 2014.1, a Use Encrypted Format option is available for new custom field definitions with specific text types. When Use Encrypted Format is enabled, field values are encrypted in the database. (Values are still displayed in the user interface.)

**Warning:** After you save a custom field definition, the Use Encrypted Format setting cannot be changed.

When you specify that a field be encrypted, any value in the encrypted field displays as **ENCRYPTED** in all accounts except the account where the value was first created.

The encrypted format option is not supported for Transaction Item Option, Suitelet, Workflow, or Workflow State custom field definitions. For custom field definitions where it is supported, the option is always disabled by default.

You can encrypt the following text types:

- Email Address
### Advanced Features for Custom Fields

- Free-Form Text
- Long Text
- Phone Number
- Rich Text
- Text Area

For all other text types, the Use Encrypted Format box is unavailable.

Be aware of the following precautions for fields with encrypted stored values:

- Fields with encrypted stored values are not available to reporting, SuiteAnalytics Connect, or for sourcing or filtering. Other than the Rich Text and Long Text text types however, they can be returned in the results of searches and saved searches.
- The nlapiLookupField SuiteScript function is not supported for fields with encrypted stored values. Other SuiteScript API functions that rely on search may not be supported.
- Encryption of stored field values increases their size and may have performance implications.
- The data type of a field with encrypted stored values cannot be changed to a type that does not support encrypted stored values.

System notes for fields with encrypted stored values mask old and new field values, displaying asterisks only, for security reasons.

### Creating Custom Fields with Values Derived from Summary Search Results

You can create a custom field that can display values derived from summary saved search results. Each summary search field displays a rollup value for a selected search results field; the value is dynamically calculated each time a form containing the summary search field is displayed. For example, you can use a summary search field to display the total quantity of all line items on a transaction. The summary search field provides an alternative to using SuiteScript to calculate a custom field's values. Note that the rollup functionality is very similar to that of custom sublists, except the search results are displayed in a field rather than a sublist.

For details about custom fields with summary search derived values, see the following:

- Steps for Creating a Summary Search Custom Field
- Kinds of Custom Fields
- Custom Field Data Types that Support Summary Search Derived Values
- Example Summary Search Custom Field
- Current Limitations for Summary Search Custom Fields

### Steps for Creating a Summary Search Custom Field

To create a custom field with a value derived from a summary search:

1. Create or edit a summary saved search that rolls up to the result you want to display in the custom field. (For instructions, see the help topic Defining a Saved Search.)
   - Add search criteria to filter out any records/lines you do not want included in the rollup value. (For instructions, see the help topic Advanced Search Criteria Filters.)
   - Define a search results field for which values will be rolled up, and select a summary type. (Count, Sum, Minimum, Maximum, and Average are supported; Group summary type is not...
supported.) For example, you could set Quantity as the results field and set a summary type of Sum. (For information about summary types, see the help topic Summary Type Descriptions.)

- Define an available filter field. The field is used to filter search results to include only those records with available filter field values that match the available filter field value of the current record. The value displayed for the summary search custom field is a rollup of search results field values for the records that have matching available filter field values. (For more information, see the help topic Selecting Available Filters for Saved Searches.)

2. Create or edit the custom field. (For general instructions, see Creating a Custom Field.)

- Select the search on the custom field's Validation & Defaulting subtab.
- Optionally, you can select a Compare To field. You can use the Compare To field in cases where you want to put the custom field on a form for a record type that is different from the summary search record type. For example, if you want to put a custom entity field showing the result of a customer record summary search on an employee custom form, you could select an employee record field whose values could be matched to the values for the search's Available Filter field. Search results field values for all records with matching values for the Compare To field and Available Filter field would be used to calculate the value of the summary search custom field.
- Do not enable the Store Value option, as values for rollup custom fields are not stored.

Kinds of Custom Fields that Support Summary Search Derived Values

You can select a summary search to provide rollup values for the following kinds of custom fields:

- Entity Fields
- CRM Fields
- Transaction Body Fields
- Other Record Fields
- Custom Record Custom Fields

Custom Field Data Types that Support Summary Search Derived Values

You can select a summary search to provide rollup values for custom fields of the following types:

- Currency
- Date
- Date/Time
- Decimal Number
- Email Address
- Free-Form Text
- Hyperlink
- Integer Number
- Long Text
- Percent
- Phone Number
- Rich Text
Advanced Features for Custom Fields

- Text Area
- Time of Day

Example Summary Search Custom Field

The following example shows the creation of a custom field to be displayed on custom purchase order forms, with the value of the field calculated by a purchase order summary search that sums the values for the purchase order line items’ Amount field. No Compare To field is needed, because the Internal ID field set as the Available Filter for the search can be matched to the Internal ID of the purchase order record displayed on the form.

Current Limitations for Summary Search Custom Fields

- Custom field values from summary search results are never stored. Field values are always calculated dynamically at runtime. A user may be able to edit values, depending on the display options set for the custom field, but the edited values are not stored. Also, because its values are not stored, the field is not available in search results, including lists based on saved searches.
- Calculated values for summary search custom fields may be different for users with different permissions. Summary search results are rolled up for the records to which the current user has access. Because users with different permissions may have access to different sets of records, the calculated value of the field may vary per user.

Dynamic Defaults and Dynamic Hyperlinks

When working with free-form text, text area, rich text or hypertext fields, in the Default Value field on the Validation & Defaulting subtab, you can include NetSuite tags in the default definition. NetSuite tags are populated with field values when the page is loaded or saved.

Dynamic defaults can be used in the hyperlink fields to include information from the record or the current session in the URL for the website.

To include NetSuite tags in the default definition of a field, enclose each tag within curly braces, defining field tags in a dynamic default as `{tag}`, where `tag` is the ID of the field. Each field in NetSuite has a unique ID and therefore a unique tag definition.
**Advanced Features for Custom Fields**

**Note:** Because field IDs are incorporated into tag definitions for fields, when creating custom fields, enter meaningful IDs for each custom field and use consistent naming conventions that meet your business needs. The default NetSuite IDs are meaningless, and in your dynamic defaults it will be difficult to know exactly what the field references.

Dynamic defaults are evaluated and each NetSuite tag is substituted on page load and page save. However, if you check the **Store Value** box, the tag substitution values are saved when the page is created as a true default. The default value is saved and is not dynamically changed when fields on the page change, letting you create a dynamic default that retains its initial value. The field must be edited manually, or updated with custom code to change its initial value.

**Important:** If you need to ensure that NetSuite tags defined in a dynamic default are substituted on each page load and save, clear the **Store Value** box.

### NetSuite Tags

Currently any field on the page that has a custom code ID can be used in a NetSuite tag. You can find the code ID for standard NetSuite fields in the [SuiteScript Records Browser](https://www.netsuite.com/). To determine the code ID of custom fields on your forms, go to the Custom Field list page for the field type, for example, Customization > Lists, Records, & Fields > CRM Fields. The code ID is displayed in the ID column.

There are also some special tags that you can use:

- **{useremail}** — Email of the user currently logged in
- **{today}** — Current date
- **{nlversion}** — The full internal NetSuite release number
- **{nlsessionid}** — The browser’s session ID, which could be used when creating a hyperlink for passing a session to a web service
- **{nluser}** — ID of the user currently logged in
- **{nlrole}** — Role ID of the user currently logged in

### Dynamic Hyperlinks

For hyperlink fields, you can also create a link to a website by defining dynamic defaults. Dynamic defaults are especially useful when the exact URL is unknown until information is collected for the record. You may also want to use information specific to the current logged in session as part of a URL parameter. When creating a dynamic hyperlink, enter the http address as usual followed by `?=` and the desired NetSuite tags embedded in curly braces.

For example, suppose that you want to include an address lookup feature on a customer form. Create a custom Entity field with the following parameters specified:

- Label: Map
- ID: _map
- Type: Hyperlink
- Store Value: Not checked
- Applies To: Customer
- Display / Subtab: Main
- Display / Link Text: Click Here for Google Map
Advanced Features for Custom Fields

- **Validation & Defaulting / Default Value:**
  
  http://maps.google.com/maps?q={(billaddr1)%20{billcity}%20{billstate}%20{billzip})

- **Validation & Defaulting / Formula:** Not checked (after you save the field, return to the custom field configuration page and ensure that Formula is cleared)

The default value includes NetSuite tags that identify the specific address of the current customer. These tags are resolved when the page is loaded so that the URL will direct the user to the customer's address as defined in the current customer record.

**Note:** When creating dynamic hyperlinks, ensure that NetSuite tags embedded in the default value definition represent required fields. If the fields are not required, and the associated form does not include a value for the tag, then the resulting URL will be invalid.

### Setting the Store Value Field

On page load and page save, dynamic defaults are evaluated and each NetSuite tag is substituted. For these substitutions to be made each time, the Store Value box for the custom field must be cleared.

If you check the Store Value box, the tag substitution is performed one time, when the custom field is created. The value is saved and does not change dynamically when the fields on the page are updated.

To ensure that NetSuite tags defined in a dynamic default are substituted every time a page is loaded and saved, clear the Store Value box.

### Setting the Formula Field

When entering a default value for a field, you can enter a formula. When you check the Formula box, the contents of the Default Value field are treated as an SQL expression, executing `SELECT <formula text> FROM dual` in the database to obtain the results. Formulas that contain tags will have the values substituted before the formula is executed.

However, if a value contains tags, it is not necessarily a formula. Ensure that you are entering a formula before checking the Formula box. For example, if you enter Welcome, {firstname} as the default for a text area field, the content is substituted correctly only if the Formula box is cleared.

If you are entering a dynamic default for a hyperlink and you clear the Formula box, the following default value results in a valid URL: http://maps.google.com/maps?q={(billaddr1)%20{billcity}%20{billstate}%20{billzip})

However, if the Formula box is checked, the preceding URL results in an Error: Invalid Expression message. To use a formula for the example above, enter the formula like this: ‘http://maps.google.com/maps?q=’ || {billaddr1} || ‘%20’ || ‘{billcity}’ || ‘%20’ || ‘{billstate}’ || ‘%20’ || ‘{billzip}"

When you are entering information in the Default Value field, the Formula box may be checked automatically. If you see an Error: Invalid Expression error or a default is not appearing as you intended, verify the Formula setting.

### Creating Formula Fields

In addition to defining a custom field that is populated with dynamic data as described in Dynamic Defaults and Dynamic Hyperlinks, you can define fields to be dynamically calculated based on the values returned in the dynamic fields.

To define formula fields, click the Validation & Defaulting subtab of the custom field. Check the Formula box. In the Formula field, use NetSuite Tags to define the dynamically defaulted fields to be used in the calculation and use SQL Expressions to define the formula.
To dynamically recalculate a formula, clear the Store Value box, and if needed, use the NULLIF function in your formula to prevent division by zero.

**Important:** The Formula box must be checked in order for the field to be processed as a formula and, as with any defaulted field, the Store Value box must be cleared to dynamically recalculate the value each time the field is viewed. Also, when a record is loaded, custom formula fields are calculated, but if changes to fields used in the formula definition are made during the time that the record is still loaded, the formula field is **not** recalculated to reflect these changes until the next time the record is loaded.

During validation, the following inline errors can be returned:

- **ERROR: Field Not Found** - returned when either a custom field or search formula is not recognized by the system.
- **ERROR: Invalid Formula** - returned when there is a syntax or data type error in the custom formula field.

When custom formula fields are returned in search results, the displayed value is the result of the dynamically calculated value at the time the search is performed. You can also define search criteria as formula fields without using a custom formula field. For more information, see the help topic Formulas in Search.

**Warning:** If a field on a record is referred to by a formula custom field, you cannot edit the referenced field with inline editing.

**Note:** Knowledge of SQL will help you to fully leverage the flexibility and power of SQL functions to define complex formulas, but the Formula popup window can help you to correctly define formula expressions. The popup includes a Function list that lets you select SQL functions to be included in expressions, and Filter or Field lists that let you to select field names and have their IDs included in expressions. For more details, refer to SQL Expressions. Also, you can refer to the SuiteScript Reference Guide for tables of NetSuite field IDs.

### Referencing Related Records in Formula Fields

When creating a formula field, you can reference data contained in fields on related records.

For example, if you created a custom entity field to apply to customer records, you could add a formula field that referenced a field on the employee record of the sales rep assigned to the customer.

**Note:** When referencing fields on other records, you are restricted to the records with search joins.

The format for formula field references is:
{fieldOnAppliedRecord.fieldOnJoinedRecord}

For example, if you wanted to display the partner email address on customer records, the format for the formula would be:

- **partner** is the field ID for the Partner field on the customer record.
- **email** is the field ID for the email field on the partner record.

The preceding example displays the email address on the record for the partner assigned to each customer.
Note: Knowledge of SQL will help you to fully leverage the flexibility and power of SQL functions to define complex formulas, but you can click Set Formula next to the Formula box to add SQL functions or field IDs to your formula.

Using List/Record Field IDs in Formula Fields

You can reference the ID value for any List/Record type field in a formula field. Use the format `{field_name.ID}`.

Using Transaction Memo Fields in Formulas

Transactions that have line items, such as sales orders, may have values for both a memo body field and memo line item fields. The `memo` field returns the body field and line item field from the transaction, unless the memo body field is blank. If the memo body field is blank, the `memo` field returns the first non-empty line item memo field's value.

Note: The `memobody` field can be used to return the memo body field value, even if it is blank. However, `memobody` can only be used in a related record, for example, `{salesorder.memobody}`.

Formula Field Example

For example, suppose you want to display the remaining credit available to a customer on the customer record. Create a custom entity field of the type Currency called Remaining Credit. Apply the field to the Customer record and set it to display on the Financial subtab. Define the field with the following formula in the Validation & Defaulting subtab:

```
{creditlimit}-nvl({balance},0)
```
(where creditlimit and balance are standard customer fields and the nvl NULL handling function forces the value to be set to the second parameter when the field is NULL)

Make sure that you enable the Formula field and clear the Store Value box to ensure that the value is always dynamically recalculated as a formula.

When a customer record is viewed, the Remaining Credit field returns a calculated value based on the credit limit and customer balance fields.

Creating a Formula Field to Display Transaction Line Numbers

You can display line numbers on the Items subtab of transactions when they are viewed online, and in printed transactions, by creating a custom field that uses the \{linenumber\} formula, and applying the field to transaction forms.

1. To create a custom line number field, go to Customization > Lists, Records, & Fields > Transaction Column Fields > New.
2. Enter a label for the field, select a Type of Integer Number, and clear the Store Value box.
3. On the Applies to subtab, check boxes for the transactions that should display line numbers, and check the Print on Standard Forms box.
4. On the Display subtab, select a Display Type of Disabled.
5. On the Validation & Defaulting subtab, check the Formula box, and enter \{linenumber\} as the Default Value.
6. Save the new field.
7. To enable the line number field to be printed on a custom transaction form, edit the form, and on the Printing Fields, Columns subtab, check the Print/Email box for the new field.

Note the following:

- Line numbers display when a transaction record is in View mode. In Edit mode, line number value is shown as 1.
- Line numbers correspond to the printed or viewed results, meaning they are contiguous even when transaction lines are omitted.
- If you are using a line number formula field for viewed and printed transactions, and you also want to include the line number in search results, set up the search as follows to ensure that search results match viewed and printed transaction items:
  - Set a criteria of Main Line = No (false).
  - Filter out transaction line items related to taxes.
  - Add the Item field as a results field.
  - Add the Amount (Gross) field as a results field; do not use the Amount field.
  - Add a Formula(Numeric) field as a results field, with the following formula expression: RANK() OVER (PARTITION by {internalid} ORDER BY {linesequencenumber}).

Using a Field Formula to Remove Extra Spaces After Date/Time Field Values

You can remove the extra spaces that appear after a Date/Time custom field value by using the \{TO_CHAR(SYSDATE, 'FMMonth DD, YYYY')\} formula.
Custom Lists

A custom list is a list of values that you can use in custom fields on your forms and records. Custom lists enable you to set up predefined choices for your employees and customers to select when entering transactions and records.

**Important:** Custom lists are intended for use with small, fixed, related sets of data. Each custom list should include no more than 1000 values. CSV import is not supported for custom lists with more than 25,000 values.

To see a list page for custom lists, go to Customization > Lists, Records, & Fields > Lists. Choose an option:

- To edit the settings of an existing custom list, click the list name. For lists under 1000 values, edit the list as desired. For lists over 1000 values, click **Manage Values** to edit the list values.
- To edit the list values, click **List**. A list page appears where you can view, edit, add, or delete any values as required.
- To create a new custom list, click **New**
- To show all lists, check the Inactive box

To save time, you should create any custom lists or subtabs that may be needed when implementing the more advanced customizations for your account.

For help on defining a custom list, see **Creating a Custom List**, **Adding Translations for Custom Lists**, and **Managing Large Custom Lists**.

Creating a Custom List

A custom list is a list of values that you can use in custom fields on your forms and records. Custom lists enable you to set up predefined choices for your employees and customers to select when entering transactions and records. You can create an unlimited number of custom lists and an unlimited number of values for each list.

You can use CSV import to import large custom lists. For information, see the help topic **Custom List Import**.

**Important:** Custom lists can be used to set up options for matrix items. For details, see the help topic **Setting Up an Item Matrix**.

To create a custom list:

2. In the **Name** field, enter a name for the list.
3. Enter a unique ID for the custom list. As a best practice, enter a name that begins with an underscore. The text you enter is prepended with the string **customlist**. If you do not enter a value, the system generates one. You use the ID value when scripting to instances of the list.
   
   If you are editing an existing custom list, note that you can change the ID value using the **Change ID** button at the top of the page.

4. Select the owner of the custom list. By default, you are selected as the owner.
Only the owner and users with edit or full permission levels may modify the custom list.

5. Enter a description of the custom list.

6. By default, values are listed in the order in which they are entered. To list values in alphabetical order, click that radio button.

7. Check the **Matrix Options List** box if the list is for matrix items.
   
   If you check the box, an **Abbreviation** column is added to the **Values** list.

   **Note:** The Accounting Matrix Items feature must be enabled to use the matrix lists option. If it is not enabled, the **Matrix Options List** box is not displayed.

8. In the **Value** field, enter a value for the list.

9. Click **Add**.

10. Repeat steps 8 and 9 to add values to the list.

11. Click **Save**.

The custom list can now be used in your custom fields. For details, see **Creating a Custom Field** and **Adding Translations for Custom Lists**.

You can use SuiteCloud Development Framework (SDF) to manage custom lists as part of file-based customization projects. For information about SDF, see the help topic **SuiteCloud Development Framework Overview**. You can use the Copy to Account feature to copy an individual custom list to another of your accounts. Each custom list page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic **Copy to Account Overview**.

**Important:** Custom lists are intended for use with small, fixed, related sets of data. Custom lists that include no more than 1000 values are recommended.

### Adding Translations for Custom Lists

If the Multi-Language feature is enabled in your account, you can translate the name of a custom list, and its available values, so that they match the language of the NetSuite user interface. For details, see the following:

- **Translating a Custom List Name**
- **Translating Custom List Values**

**Important:** Before you can add these translations, you need to select translation languages at Setup > Company > General Preferences, on the Languages subtab. The Languages subtab lists both system-supported languages that can be used for the NetSuite user interface (and are available at Home > Set Preferences), and additional languages that can be used for website translations only (and are not available at Home > Set Preferences). You should only enter translations for system-supported languages, because these are the only languages that can be displayed in the user interface. For details, see the help topic **Configuring Multiple Languages**.

### Translating a Custom List Name

You can define translations for a custom list name on the Translation subtab of the custom list page.
Custom Lists

**Custom List**

**Note:** The maximum length for a custom list name's translation is 30 characters.

**Translating Custom List Values**

You can define translations for a custom list's values, on the Values subtab of the custom list page:
Note: The maximum length for a custom list value's translation is 60 characters.

Managing Large Custom Lists

Custom lists larger than 1000 values are displayed on a list page, enabling you to page through and manage the values of large lists. If you have a large custom list of more than 1000 values that was imported into NetSuite, the Values subtab on the Custom List page contains a Manage Values button. To manage the list values, click Manage Values.
On the Custom List page, the values can only be viewed in alphabetical order. Use the page controls to view, edit, add, or delete any values as required. If your organization allows inline editing, you cannot add or edit the abbreviation or translation from this list page. Edit the individual values to change the value, abbreviation, or translation.
Custom Forms

Forms are the pages used to enter information into the NetSuite database. The standard set of forms provided with your NetSuite account can be customized to better suit your business needs. For example, you may want to reorganize subtabs or rename fields to better match your business workflow and terminology. After you have created a custom form, it can be set as the preferred or default form for a page, or selected as needed from a custom form list.

**Note:** Form preferences are controlled by settings on the custom forms page as well as settings defined for each role.

Any settings defined for a specified role override the preferred form settings on the forms page. For Employee Center roles, only one form is ever made available to this type of role.

**Important:** SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should only access the NetSuite UI by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic SuiteScript 2.0 Custom Pages.

Custom Entry Forms

You use entry forms to enter information into NetSuite. To create your own custom entry forms, start with an existing standard form and customize it. On the custom form, you can rearrange, rename, hide, or disable fields, subtabs, and buttons. You can also make specific fields mandatory, add custom fields, or apply custom code.

For detailed information on how to customize an entry form, see Creating Custom Entry and Transaction Forms and Customizing Address Forms.

After you have created a custom entry form, you can set it as the default form for specific user roles. For more information, see Defining Preferred Forms.

Custom Transaction Forms

You use transaction forms to enter and print transactions in NetSuite. To create your own custom transaction form, start with an existing standard form and customize it. On your custom form, you can rearrange, rename, hide, or disable fields, subtabs, and buttons. You can also make specific fields mandatory, add custom fields, or apply custom code.

For detailed information on how to customize a transaction form, see Creating Custom Entry and Transaction Forms.

After you have created a custom transaction form, you can set it as the default form for specific user roles.

You can also link transaction forms together to create transaction workflows. For more information, see Linking Transaction Forms.

Advanced PDF/HTML Templates

You can use advanced PDF/HTML templates to create custom transaction and entry forms. With custom templates, you can customize the look and feel of the transaction and entry forms that you print or email as part of your business.
You can define different custom transaction layouts for printing forms as PDF and for printing forms as HTML. Using custom advanced templates, you can hide and show fields, move and resize fields and change the font and colors on your forms.

Advanced PDF/HTML templates provide extensive customization capabilities, new feature enhancements, and they support current industry standards for HTML-based editing.

**Important:** To use advanced templates, ensure that the Advanced PDF/HTML Templates feature is enabled. For information, see Enabling the Advanced PDF/HTML Templates Feature.

You can create custom templates for the following types of printed forms:

- Bill of Materials
- Check
- Item Label
- Mailing Label
- Packing Slip
- Payment Receipt
- Payment Voucher
- Picking Ticket
- Price List
- Remittance Slip
- Return Form
- Shipping Label
- Statement
- Transaction

You can set custom forms for supported transaction types to use standard advanced PDF/HTML templates provided by NetSuite. Or, you can create custom templates in a template editor that is available in the NetSuite user interface. This editor supports both rich text editing and HTML markup source editing, and uses industry-standard tools and syntax.

For more details, see Advanced PDF/HTML Templates.

**Basic Layouts for Transaction Forms**

You can create basic layouts for your custom transaction forms. Basic layouts include labels above information and a black background color.

To define whether to print using PDF or HTML, go to Home > Set Preferences > Transactions tab, check or clear the Print Using HTML box, and click Save.

**Important:** Basic layouts will be deprecated in a future release. We encourage you to use Advanced PDF/HTML Templates instead because new features are added exclusively to advanced printing.

To create a basic custom layout, go to Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and click Customize next to a layout. Make your changes and click Save. You can choose default layouts to apply to one or more types of forms by
Optimizing Custom Form Performance

To optimize form performance, use a minimalist approach with your custom forms. Ensure that you:

- Limit the number of fields and sublists.
- Minimize client scripts and workflows on forms, and manage them carefully. For more information, see the help topic SuiteScript 2.0 Client Script Type.
- Implement server callbacks only as required.
- Use the browser built-in code inspection tools to audit the scripts on a form page.

Creating Custom Entry and Transaction Forms

To create a custom entry or transaction form:

1. Do one of the following to select the desired form to customize.
   - Go to the type of form you want to customize, and click Customize.
   - Click Customize or Edit next to the form name at Customization > Forms > Transaction Forms.
   - Click Customize or Edit next to the form name at Customization > Forms > Transaction Forms.

   **Note:** Forms labeled as (External) are used in the Customer Center and My Account section of your website.

2. In the Name field, enter a name for your custom form.
3. Set the custom form properties.
   - Options vary depending on the type of form being customized but may include options described in the following sections.
4. To see what the form looks like, create a transaction that uses the form and then use the list to select the custom form that you created.

You can use SuiteCloud Development Framework (SDF) to manage custom entry and transaction forms as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud.
Development Framework Overview. You can use the Copy to Account feature to copy an individual custom entry or transaction form to another of your accounts. Each custom entry or transaction form page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

For both Entry and Transaction Forms:

- **Form is Preferred** – Check to make this form your preferred form. Only one form can be defined as the preferred form per transaction type. Checking this box clears any previously defined preferred forms. The preferred form is automatically used when entering transactions of this type. For details on how preferred forms are defined, see Defining Preferred Forms.

Note the following about marking an entry or transaction form Preferred for the Customer Center role:

- External forms, meaning forms with names appended with (External), can be marked preferred for Customer Center roles, but not for other roles.
- Forms that are not external cannot be marked preferred for Customer Center roles.
- When a non-online order form is marked preferred for Customer Center, it is saved as the form for the order. However, an online form is not saved as the form for an order, even if it is preferred; instead the preferred non-online order form is used.

Note that Employee Center roles have limited access to forms such that only one form is ever made available to this type of role, and the form set on the Role record takes precedence over the preferred form set here.

- **Store Form with Record** – Check to store a reference to this form with each record created by it. When viewed or edited, any record that was originally created with this form is displayed using this form rather than the user's preferred form.

  - For custom entry forms, the Store Form with Record option is disabled by default.
  - The Store Form with Record option is only available for a subset of transaction forms. For this subset, this option is enabled by default. For other transaction forms, this option is not available and custom forms are always stored with records. For more information, see Storing Custom Forms with Transactions.

**Important:** If a custom form is stored with a record, a user who accesses that record gets that form even if that form is not enabled for their role.

- **Print Template** – (Available only when the Advanced PDF/HTML Templates feature is enabled, and the Printing Type is set to Advanced.) Select a template to be used when transactions associated with this form are printed.

For Entry Forms Only:

- **Enable Field Editing on Lists** – Check to allow inline editing of this form. Inline editing allows users to edit fields on this form from within the record view. When enabled, fields that can be edited from within the record view display the inline editing icon. It also provides a New menu that lists options to create new related records.

- **Use for Pop-ups** – Check to use this form in popup windows when you add a record of this type from another record. This capability is available only for entity forms, item forms, and custom record forms. For each type of form, only one form can be set as the popup form. When set for a new form, the form previously defined as the popup form is unset.
Creating Custom Entry and Transaction Forms

This form is also available on the standard record page in the Custom Form list.

- **Popup Only** – When Use for Pop-ups is checked, check Popup Only to use this form only when adding a record of this type from another record. If the form is a standard popup form, you cannot change the form in the popup window.

  For standard forms, the Custom Form list is available in the popup window, and the options available include this form and the standard forms.

For Transaction Forms Only:

- **Allow Add Multiple** – Clear to hide the Add Multiple button on transaction item lists. You should turn off this option on any forms that rely on custom code line item validation scripts, since the Add Multiple button is displayed on item machines and lets you add multiple items at a time to the item list. However, when items are added in this manner, any Validate Line custom code events defined for the form will not fire.

- **Printing Type** – (Available only when the Advanced PDF/HTML Templates feature is enabled.) The Basic option that is selected by default enables you to set the custom form to use transaction form PDF layouts and HTML layouts. Choose the Advanced option to set the custom form to use an advanced PDF/HTML template instead. For more details, see Advanced PDF/HTML Templates.

  **Note:** Forms that will be used by a Customer Center role must have printing type set to **Basic**.

- **Email Template** – (Available only when the Advanced PDF/HTML Templates feature is enabled, and the Printing Type is set to Advanced.) Select a template to be used when transactions associated with this form are emailed.

- **Transaction Form PDF Layouts** – Select a layout for your form. To customize layouts, click Custom Layouts in the upper-right corner of the page. For more details, see Customizing Transaction Form PDF Layouts.

- **Transaction Form HTML Layouts** – Select a layout for your form. Standard and Classic layouts exist for all the standard form types other than shipping label. These are assumed to be printed using PDF. To customize layouts, click Customize. For more details, see Transaction Form HTML Layouts.

- **Remittance Slip** – Specify which remittance slip is used on invoices, statements, return authorizations and packing slips. To prevent the current transaction from printing with a remittance slip, select None.

  **Note:** To use this feature, the Print Remittance Form with Invoices & Statements preference must be turned on.

- **Disclaimer** – Enter a policy statement or message. You can enter up to 4,000 characters, including spaces, for this message.

- **Address** – Enter an address to be used only on this form. If you do not enter an address, the default address entered at Setup > Company > Company Information is used.

- **Logo** – Select a logo to be used only on this form. You must first upload the image to your File Cabinet at Documents > Files > Images. If you do not select a logo, the default logo selected at Setup > Company > Company Information is used.

  Logos are not displayed on the following transaction forms when basic printing is used: picking ticket, bill of materials, shipping label, opportunity, item fulfillment, item receipt, store pickup fulfillment, and custom transactions.

- **Columns Space** – This number is the maximum number of inches of printable space allowed on your form. The measurement is determined by the Page Width of the layout you choose. You can change
the page width by creating a custom layout. To do this, click **Custom Layouts** in the upper right-hand corner of the page.

- **Columns Width** – This is the total of all the columns on your form. This measurement is determined by the values that you enter on the column tab for your custom form. If your columns’ width totals more than your columns’ space, NetSuite adjusts the widths proportionally to fit on the page.

For more information, see the following topics.

- Moving Fields and Lists Between Subtabs
- Configuring Field Groups
- Configuring Fields or Screens
- Configuring Buttons and Actions
- Configuring Printing Fields
- Configuring Sublists
- Configuring QuickViews
- Associating Custom Code (Client SuiteScript) Files With Custom Forms
- Defining Preferred Forms
- Adding Disclaimers to Transaction Form Footers
- Specifying Check Layout by Subsidiary
- Customizing Multiple Page Transaction Forms
- Linking Transaction Forms

After you have created a custom form, you should configure the subtabs. See **Configuring Subtabs for Custom Entry and Transaction Forms**.

**Important:** As you are configuring your custom form, be sure to consider whether tax data is required for transactions for which you want to use the form. Only a form that includes required tax-related fields can be used for a transaction with tax consequences. Note that you cannot control naming for tax fields through form customization; you must go to Setup > Accounting > Set Up Taxes. See the help topic **Customizing Tax Fields on Transaction Forms**.

**Note:** If you create or edit custom project forms when the Project Management feature is enabled, be aware that these forms may be altered if you later disable this feature. Immediately after you disable Project Management, you need to review custom project forms to see if they have been changed, and if necessary, edit them to fit your requirements. For information about working with this feature, see the help topics **Enabling Project Features** and **Using Project Management**.

### Storing Custom Forms with Transactions

The Store Form with Record option, available for custom entry and some custom transaction forms, indicates whether a reference to the form should be stored with each record created by it. When this reference is stored, whenever a record is viewed or edited, that form is used, rather than the user’s preferred form. This option is enabled by default for custom transaction forms.

For some transaction types, the custom form is always stored with each transaction where it was used. For these transaction types, the Store Form with Record option is not available on custom forms, so users cannot change it.
The following table lists custom transaction form types and whether users have the choice of changing the Store Form with Record option from the default of True.

<table>
<thead>
<tr>
<th>Custom Transaction Form Type</th>
<th>Can Choose Not to Store Form with Record?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembly Build</td>
<td>Yes</td>
</tr>
<tr>
<td>Assembly Unbuild</td>
<td>Yes</td>
</tr>
<tr>
<td>Bill of Materials</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Bill Payment</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Cash Refund</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Cash Sale</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Check</td>
<td>Yes</td>
</tr>
<tr>
<td>Credit Card Charge</td>
<td>Yes</td>
</tr>
<tr>
<td>Credit Memo</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Customer Deposit</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Customer Refund</td>
<td>Yes</td>
</tr>
<tr>
<td>Deposit</td>
<td>Yes</td>
</tr>
<tr>
<td>Estimate (Quote)</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Expense Report</td>
<td>Yes</td>
</tr>
<tr>
<td>Inventory Adjustment</td>
<td>Yes</td>
</tr>
<tr>
<td>Inventory Cost Revaluation</td>
<td>Yes</td>
</tr>
<tr>
<td>Inventory Worksheet</td>
<td>Yes</td>
</tr>
<tr>
<td>Invoice</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Item Fulfillment</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Item Receipt</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Journal Entry</td>
<td>Yes</td>
</tr>
<tr>
<td>Opportunity</td>
<td>Yes</td>
</tr>
<tr>
<td>Packing Slip</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Payment</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Picking Ticket</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Price List</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Purchase Order</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Remittance Slip</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Return Authorization</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Return Form</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Custom Transaction Form Type</td>
<td>Can Choose Not to Store Form with Record?</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>Sales Order</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Shipping Label</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Statement</td>
<td>No, form always stored with record</td>
</tr>
<tr>
<td>Transfer Order</td>
<td>Yes</td>
</tr>
<tr>
<td>Vendor Bill</td>
<td>Yes</td>
</tr>
<tr>
<td>Vendor Credit</td>
<td>Yes</td>
</tr>
<tr>
<td>Vendor Return Authorization</td>
<td>Yes</td>
</tr>
<tr>
<td>Work Order</td>
<td>Yes</td>
</tr>
</tbody>
</table>

⚠️ **Important:** If a custom form is stored with a record, a user who accesses that record gets that form even if that form is not enabled for their role.

### Configuring Subtabs for Custom Entry and Transaction Forms

On the **Subtab** subtab, you can select which subtabs to display on your form and provide a custom heading for each subtab. For example, you have a OneWorld account and you do not want to allow users to associate vendors with multiple subsidiaries. Create a custom vendor entry form, and specify that the Subsidiaries subtab not be shown.

**To modify the available subtabs:**

1. In the **Show** column, check the boxes for the subtabs you want to display on the form. Clear the boxes for the subtabs you do not want to display.
2. In the **Label** column, edit the headings for the subtabs as desired.
3. To rearrange the subtabs, click the desired line to drag and drop it to the desired position or click **Move to Top / Move to Bottom**.
4. After you have configured the subtabs, you should configure the fields or screens. See Configuring Fields or Screens.

To add a subtab that does not yet exist to a form, you need to first create the subtab at Customization > Forms > Subtabs. You can also rename custom subtabs there. See Creating Custom Subtabs.

### Moving Fields and Lists Between Subtabs

You can move fields and lists (the Contacts list on entry forms, for example) between subtabs on entry forms.

After you have made changes on the Custom Form page, click Save & Move Elements to move fields and lists on the form to other subtabs.

**To move fields and lists between subtabs:**

1. Click **Save & Move Elements**.
Creating Custom Entry and Transaction Forms

Your form is saved, and the Move Form Elements page opens.

On the **Fields** subtab are lists of fields that appear on each subtab on the form you are customizing that include fields.

Subtabs with lists (for example, the **Contacts** subtab on entry forms or the **Items** subtab on transactions) are shown on the **Lists** subtab of the Move Form Elements page.

2. For each field on each subtab, select the subtab you want that field to show on. Note that you can also select multiple fields and move them to a subtab to set the dropdown list on those fields. At any time you can click **Save and Move More**, which saves your field changes, and reloads the page with the fields now moved to the specified subtab.

3. Click the **Lists** subtab.

4. Select the subtab you want each list to appear on.

   - It is possible to move a field or list to a subtab that is **not** displayed since it does not contain items of that type (fields or lists respectively). When a field or list is moved to a previously empty, undisplayed subtab, you **cannot** move the field back until the Move Form Items page is saved. If you need to move a field again, you can return to the Move Form Items page and move it after you have saved.

5. Click **Save** to return to the custom form page.

Fields and lists are now shown on the subtabs you selected. If a field or list has been moved to a subtab that is not shown on the form, those items are no longer available on the form.

### Configuring Field Groups

Click the Field Groups subtab to customize the field groups that appear on your forms. Note that the Field Groups subtab appears on custom form pages only **after** you have upgraded and deployed the standard forms in your account.

**Note:** If you are not familiar with the term field groups, see [Field Groups](https://help.oracle.com/sub/1280) in the NetSuite Help Center.

You can use the Field Groups subtab to change the UI label of a field group, the fields within the field group, and the placement of a field group on the page. You can also use the Field Groups subtab to create new field groups and organize all fields into specific field groups.

Be aware that the order of field groups on the Field Groups subtab determines the order of field groups on your form. For example, in the following screenshot, the Primary Information field group appears on top of the Sales Information and Classification field groups.

Any fields that are not assigned to a field group are listed together below all other fields that are assigned to field groups. These unassigned fields are always displayed last and can be reordered among all other fields that are not assigned to a group. If you want to move a field up higher on a subtab, assign it to a field group.

Be aware that adding custom fields to field groups can cause field text to be displayed differently than you might expect based on the display size attributes set for custom fields, as described in [Configuring Field Groups](https://help.oracle.com/sub/1280).

**To work with field groups:**

1. On the Field Groups subtab, click the **Main** subtab to customize field groups that appear in the main body area of a page. Click any of the other field group subtabs to customize the field groups that appear on a form's subtabs.
2. In the **Label** column, edit field group headings.

3. In the **Show** column, clear a field group to hide it on the form.

4. In the **Single Column** column, check a field group if you want it to display vertically rather than horizontally, which is the default display. The following figure provides an example of the Primary Information field group vertically displayed.
Creating Custom Entry and Transaction Forms

Note that if you select **Single Column** for three field groups together, the field groups appear side by side in the UI, as three single (field group) columns make a row.

5. Click the **X** icon in the last column to delete the field group. Clicking the X does not remove the fields from your form. The fields continue to display at the bottom of the page. However, the fields are not be assigned to any field group because that field group no longer exists.

6. After you have configured your field groups, you can click the **Fields** subtab (for Entry forms) or the **Sublist Fields** subtab (for Transaction forms) to configure the fields that appear in each field group. See Configuring Fields or Screens.

**Field Groups on Custom Forms of Custom Records**

You can add field groups to both new and existing custom record custom forms. Note that the following steps are for adding field groups to an existing custom record.

**To add field groups to the custom form of a custom record:**

1. Go to Customization > Lists, Records, & Fields > Record Types.
2. On the Record Types list page, select your record type.
3. Click the **Forms** subtab.
4. On the **Forms** subtab, click **Edit** next to the custom record’s custom form. (Note that if you do not yet have a custom form, click **Customize** next to the record type to create a new custom form.)
5. Next, click the **Field Groups** subtab.
6. Provide a UI label for the first field group, set all other field group attributes, and click **Add** to add the new field group.
7. After adding all field groups, click **Save**.
8. After you have configured your field groups, you can click the **Fields** subtab (for Entry forms) or the **Sublist Fields** subtab (for Transaction forms) to configure the fields that appear in each field group. See Configuring Fields or Screens.

**Configuring Fields or Screens**

In the Fields subtab of entry forms or Sublist Fields subtab of transaction forms, fields available on each subtab for the current form can be configured. These fields can also be moved to display on a different subtab or in different field groups.

**Note:** Even if you have disabled a subtab as described in Configuring Subtabs, you can still configure fields for that subtab. This is useful should you later decide to include the subtab on the form.

**To configure fields for each available tab:**

1. In the **Show** column, check the boxes next to the fields you want to display. Clear the boxes for the fields you do not want to display.

**Warning:** Hiding fields on a form can hide other related fields. For example, hiding the **Credit Card Approved** box on the sales order form also hides fields related to **Address Verification System (AVS)**. Also, if you use multiple locations, you must show the **Location** field on the sales order form you use for your website to calculate shipping correctly.

If an advanced template relies on a sublist hidden field, the template may fail to print. For information about how to avoid this, see Using FreeMarker to Work with Hidden Fields Used in Advanced Templates.
Creating Custom Entry and Transaction Forms

1. If you have the Termination Reason Tracking feature enabled, you will not be able to save the employee record if you hide any of the required termination fields. For more information, see the help topic Termination Reason Tracking.

2. In the Quick Add column (entry forms only), check the boxes next to the fields you want to define as inline editable and available in Quick Add portlets.
   - If a field is checked here, and the Enable Field Editing on Lists box is also checked, then that field can be edited directly and the record is saved when the user clicks off of the field. Inline editable fields are also available when attaching contacts or scheduling activities to records. For more information, see the help topic Using Inline Editing.
   - Quick Add portlets can be made available on users’ dashboards to allow the quick addition of a selected record type without shifting to the record type page. For more information, see the help topic Quick Add Portlet.

3. In the Mandatory column, check the boxes next to the fields you want to require on your entry form.
   - You cannot clear mandatory boxes for fields that NetSuite requires or fields defined as mandatory in the custom field definition.

4. In the Display Type column, select the display type for each field.
   - For more information about display types, see Setting Display Options for Custom Fields.

5. In the Check Box Default column, choose one of the following options for each check box field on the form:
   - Use Field Default, to use the default value set in the field definition.
   - Checked, to set the default to checked, possibly overriding the default set in the field definition.
   - Unchecked, to set the default to checked, possibly overriding the default set in the field definition.

6. In the Label field, edit the name of any of the fields as desired. You can enter up to 200 characters for the label. However, you should consider how the length of a label appears on printed forms.
   - If you change the label of a field on the Sublist Fields subtab, the new label is also applied to the field on the Printing Fields subtab.
   - In accounts where the Multi-Language feature is enabled, the label you edit here applies only to your current language, as set at Home > Set Preferences. To modify the label in other languages, you need to reset your language preference and reedit the label for each one. Note that two different settings are available for languages such as English, French, and Spanish, for example English (U.S.) and English (International).

7. In the Field Groups list, select the field you want the field to appear in.
   - After you assign a field to a field group, the field can only be moved within the group. If you want to move the field elsewhere on a form, change its field group.

8. Check any box in the Column Break column to insert a column break after that field.
   - Be aware that adding a custom field to a field group or inserting a column break after a field can cause field text to be displayed differently than you might expect based on the display size attributes set for custom fields, as described in Configuring Field Groups.

9. If you want to include a blank line before a field, enter the number of blank lines in the Space Before column.

10. If you want to associate a field with the field immediately above it, check the Same Row as Previous box. An associated field shares the same Show/Hide setting as the previous field,
belongs to the same field group as the previous field, and is displayed together on the form with the previous field. See **Associating Related Fields on Custom Forms**.

11. Rearrange the fields as desired.
   - Move each field to the desired subtab. Click **Move Elements Between Subtabs**. For details, refer to **Moving Fields and Lists Between Subtabs**.
   - Rearrange the order of the fields on each subtab. Click the desired line and then drag and drop it to the desired position or click **Move to Top / Move to Bottom**.

   ! **Note:** Fields that belong to a field group can only be moved within the group. If you want to move the field elsewhere on a form, either change the field group for the field or rearrange the field groups.

   Any fields that are not assigned to a field group are listed together below all other fields that are assigned to field groups. These unassigned fields are always displayed last and can be reordered among all other unassigned fields. If you want to move a field up higher on a subtab, assign it to a field group.

12. For transaction forms only, set the following options:
   - In the **Default Checked** column, set the default value for check box fields. This option is only available for toggle fields where the item is either checked or unchecked. If you enable the **Default Checked** box, the field is automatically checked unless the user specifically clears the box.
   - Click the **Sublist Fields** subtab, and arrange the line-item columns for your transaction entry form by dragging and dropping lines as desired. The order of the columns on the screen does not have to match the order of the printed columns of your form.

13. On the **Sublist Fields** subtab in the **Items Filter** field, select a saved search you want to use to filter the items that appear in the Items list on this form.
   
   For information on creating a saved search, see the help topic **Defining a Saved Search**.

14. On the **Total Box** subtab, specify which fields you want to show in the form totals.
   
   If you use SuiteTax, additional tax fields are available on the Total Box subtab. For more information, see the help topic **Tax Details on Transactions in SuiteTax**.

15. Click **Save**.

16. If desired, click **New Field** to create any new custom fields.

   Ensure that you save any changes before creating new fields from within the form. Selecting **New Field** causes you to leave the custom form to go to the custom field page. For detailed information on creating custom fields for your forms, see **Custom Fields**.

17. After you have configured the fields or screens, you should configure the printing fields (for transaction forms only). See **Configuring Printing Fields** for details.

   ! **Important:** As you are configuring your custom form, be sure to consider whether tax data is required for transactions for which you want to use the form. Only a form that includes required tax-related fields can be used for a transaction with tax consequences. Note that you cannot control naming for tax fields through form customization; you must go to Setup > Accounting > Set Up Taxes. See the help topic **Customizing Tax Fields on Transaction Forms**.

---

**Filtering the Items Dropdown List on Transactions**

You can use a saved item search to limit the items that appear in the item list on transactions. For more information, see the help topic **Defining a Saved Search**.
To customize that item list:

1. Create the saved item search to filter the items in the item list. The items in the Items dropdown list will meet the criteria of the saved search.
2. Customize the transaction form on which you want the filtered item list to appear.
3. On the Custom Transaction Form page, click the Sublist Fields subtab.
4. In the Item Filter list, select the saved item search.
5. Click Save.

When this custom transaction form is used, the item field is filtered to display only the items that meet your search criteria. For more information on customizing forms, see Creating Custom Entry and Transaction Forms.

Associating Related Fields on Custom Forms

You can define an association among closely related fields on custom entry and transaction forms. Your definition of one or more fields as associated with a previous field on the form causes the associated fields to:

- Share the same Show/Hide setting as the previous field
- Belong to the same Field Group as the previous field
- Be displayed together on the form with the previous field

You define associated fields on the Fields subtab of a custom form by checking the Same Row as Previous box. You also can remove a field's association with another field by clearing this box.
Be aware that setting custom fields to be Same Row as Previous on custom forms can cause field text to be displayed differently than you might expect based on the display size attributes set for custom fields, as described in Configuring Field Groups.

**Important:** In a previous release a change was made so that all fields previously defined as child fields are defined as associated fields of former parent fields. This change provides greater flexibility because you can remove the association if desired.

### Configuring Buttons and Actions

On the Actions subtab you can configure standard NetSuite buttons and custom buttons. Use the Standard Actions subtab to customize built-in, standard NetSuite buttons. (See Working with Standard Buttons for details.) Use the Custom Actions subtab to create or modify custom buttons you have associated with client SuiteScripts. (See Working with Custom Buttons for details.)

Also note that you can use point-and-click customization to translate the button labels for both standard and custom buttons. For details, see Translating Buttons to Other Languages.

**Note:** You can also use SuiteScript to hide and rename buttons. See the help topic Button IDs for a list of standard buttons that are supported in SuiteScript. Additionally, you can use the Remove Button Action action in SuiteFlow to conditionally hide buttons from specific users or when a record is in a certain state.

**Important:** Customizing the Save, Edit, Cancel, Back, and Reset buttons is not supported in SuiteScript or in point-and-click customization.

After you have configured your buttons, you should configure the printing fields. See Configuring Printing Fields.

### Working with Standard Buttons

Use the following steps to configure standard buttons through point-and-click customization.

**To customize standard buttons:**

1. On the Custom Form page, go to Actions > Standard Actions.
2. In the **Label** field, enter a UI label for the button.
3. In the **Show** column, select whether you want the button to appear or to be hidden on the form. Note that buttons are shown by default. To hide a button, clear the box next to the button.
4. In the **Display As** column, choose **Button** to display the button as an inline button. Choose **Menu** if you want the button to appear as an action in the **More Actions** menu or as a menu item in a button group. See Understanding Button and Action Layout for more details.
5. Click **Save**.

### Working with Custom Buttons

You can add custom buttons to forms to execute client SuiteScript. For example, you could add Create Invoice button on a customer form that would execute a specific function when the button is clicked. Note that for the Custom Actions subtab to appear, you must have the Client SuiteScript feature enabled in your account.

Also note that you must have already added the client SuiteScript file to the Custom Code subtab on the form. See Associating Custom Code (Client SuiteScript) Files With Custom Forms for details.
The primary object used to encapsulate custom buttons is `serverWidget.Button`. For more information, see the help topic `serverWidget.Button`.

⚠️ **Important:** Custom buttons do not appear when a record is in View mode. Custom buttons appear only when a record is in Edit mode. To make a button appear in View mode, use a User Event Script or Workflow.

**To add a custom button to associate with client SuiteScript:**

1. On the Custom Form page, choose Actions > Custom Actions.
2. In the **Label** field, enter the UI label for the button. Although you can enter up to 99 characters, you should keep button names brief to save space.
3. In the **Function** field, enter the name of the function you want executed when the button is clicked. The function can exist in your client SuiteScript file or any library file you have attached to the Custom Code subtab.
4. In the **Display As** column, choose **Button** to display the button as an inline button. Choose **Menu** if you want the button to appear as an action in the More Actions menu. Custom buttons, when changed to **Menu**, are grouped in the More Actions menu.
5. Click **Save**.

**Understanding Button and Action Layout**

On the Action subtab, use the **Display As** list to customize the layout of a button or action. (See the first figure below.)

When you first click the Actions subtab, the display type for each action defaults to either Button or Menu. **Button** means that the action displays as a button that can be clicked on the top and bottom of a form. **Menu** means that the action displays as an action in the More Actions menu, or as a menu item in a button group (See the second screenshot.)
In the case of certain buttons such as the **Save &** buttons (Save & New and Save & Print, and so on), the display type of **Menu** means that they display as an item in a button group. (See the following figure.) **Menu** does not mean the button displays as an item in the More Actions menu.

Actions are grouped in sub-groups represented from left to right. Core actions such as Save appear as buttons and appear in the left-most group. Functional buttons (actions specific to a record) are in a middle group, and More Action actions are in the right-most group.

Also note, if you choose to display a custom button as **Button**, it appears inline in front of the **Create New** button (see figure). If you choose to display both standard and custom buttons as an action in the **Actions** menu, the action appears at the top of the Actions list.

**Translating Buttons to Other Languages**

If you have the Multi-Language feature enabled, you can provide custom translations of standard and custom buttons on your forms. Note, however, you must have the Multi-Language feature enabled in your account to translate your buttons into multiple languages.

**Note:** To enable the Multi-Language feature, go to Setup > Company > Enable Features. On the Company subtab, check the Multi-Language box, and then click Save.

To provide a custom translation for a button, be sure you have logged into NetSuite in the language you want to translate to, navigate to the custom form page, click the Actions subtab, and then perform the translation in place using the Label field.

**Configuring Printing Fields**

(Transaction Forms Only that use basic printing)

On the Printing Fields subtab, you can customize the way your form appears when printed. The Printing Fields subtabs displayed represent the different areas of a printed form.

To configure how each section of your printed form displays:

1. For all subtabs:
Creating Custom Entry and Transaction Forms

1. In the Print/Email column, check the boxes next to fields you want to appear on printed and emailed forms. Clear the boxes for fields you do not want to appear on printed and emailed forms. This column also controls the visibility of a field in the Customer Center. Check the boxes next to fields you want to appear in the Customer Center. Clear the boxes for fields you do not want to appear in the Customer Center.

2. In the Label column, edit the labels of the fields as desired.

   ![Note: If you change the label for a field on the Printing Fields subtab, the label is also automatically applied to the field on the Sublist Fields subtab.]

2. For the Body and Columns subtabs:

   a. In the Width column, enter the width for transaction column fields appearing on your printed and emailed forms.

   ![Note: To change the width of custom body fields, you must make the change on the Custom Form page. The width you set on the custom field does not affect printed transaction body fields.]

3. Rearrange the fields as desired. Drag and drop each line item to the desired position.

3. Click Save.

   ![Important: If the Advanced Taxes feature is enabled in your account, or if you are using NetSuite OneWorld, which requires Advanced Taxes, you cannot directly rename tax fields on a custom transaction form. To change names of tax fields that display on the custom form, you must rename them in the appropriate languages on the Field Naming subtab of the Set Up Taxes page, at Setup > Accounting > Taxes > Set Up Taxes (Administrator). See the help topic Customizing Tax Fields on Transaction Forms.]

Of the fields on the header subtab, some represent actual values that are inserted and some represent labels of field data that is inserted. For the fields that are values, the defaults from the company setup page are used unless overridden on the printing fields subtab.

Following are the available fields listed by type:

**Values:**
- Company Name
- Company Phone
- Company URL
- Form Title
- Page Number

**Labels:**
- Business Number
- Acct. No.
- Date
- Doc. No.
- Bill To
Creating Custom Entry and Transaction Forms

Ship To

After you have configured the fields or screens, you should configure the printing fields. See Configuring Sublists.

Configuring Sublists

**Note:** This task applies to entry forms only.

On the Printing Fields subtab, you can customize the way your form appears when printed. The Printing Fields subtabs displayed represent the different areas of a printed form.

On the Sublists subtab, you can customize which sublists are available on each subtab of the form.

To configure sublists for each available tab:

1. In the **Show** column, check the boxes for the sublists you want to display on the form. Clear the boxes for the lists you do not want to display.
2. In the **Never Empty** column, specify that a sublist entry is mandatory by checking the box. If this box is checked, at least one row must be entered for the sublist. If the user submits a custom form with an empty sublist, a warning message appears that includes the name of the sublist that requires an entry.

   The Never Empty box is not available for all sublists. Only sublists that are listed in the SuiteScript Records Browser are supported. The following sublists do not appear on the Lists subtab because they require at least one row by default:
   - Items
   - Lines
   - Components
3. In the **Label** column, edit the headings for the sublists as desired.
4. Rearrange the sublists as desired.
   - Move each sublist to the desired subtab. Click **Move Elements Between Subtabs > Lists**. For details, refer to Moving Fields and Lists Between Subtabs.
   - Rearrange the order of the sublists on each subtab. Click the desired line and drag and drop it to the desired position or click **Move to Top / Move to Bottom**.
5. Click **Save**.

After you have configured the sublists, you should configure the borders and placement. See Configuring Borders and Placement.

Configuring Sublist Fields

On the Sublist Fields subtab, you can change the label of custom sublist fields and rearrange the columns, if required.

**Note:** If you use the Item Filter on this page, only saved searches marked as Public appear in the list. For more information, see the help topic Defining a Saved Search

To configure sublist fields:

1. In the **Show** column, check the boxes for the sublist fields you want to display on the form. Clear the boxes for the sublist fields you do not want to display.
2. In the Label column, edit the labels for the sublist fields as desired.
3. Rearrange the fields as desired.
4. Click Save.

Configuring QuickViews

For both entry and transaction forms you use the QuickViews tab to customize which fields appear in your QuickViews. The following screenshot shows an example of a QuickView for a Phone Call record.

Body fields and custom fields are supported in QuickViews. However, you cannot add sublist fields. Also, the maximum number of fields you can add to a QuickView is 20.

Note that you can also configure QuickViews on forms associated with custom records. See Configuring QuickViews for Custom Records for information.

Important: Configuring QuickViews on entry and transaction forms is easy if you have deployed any upgraded forms in your account. See Configuring QuickViews for Upgraded Forms for information.

Important: If you have not upgraded and deployed the forms in your account, you can still indirectly affect which fields appear in a QuickView, however, the process is not as straightforward. The benefits of QuickView customization are meant to come with upgraded forms. If you have not upgraded the forms in your account, yet you would like to update your QuickViews, see Configuring QuickViews for Non-Upgraded Forms.

Configuring QuickViews for Upgraded Forms

QuickView customization is easy if you have upgraded the custom forms in your account to include field groups and all other look-and-feel enhancements introduced in NetSuite in a previous release. After a form has been upgraded, a QuickView configuration tab appears on the form customization page. You use this tab to add, remove, and rearrange the fields in a QuickView.

To configure QuickViews for upgraded entry and transaction forms:

1. Go to Customization > Forms > Entry Form [or Transaction Form].
2. Click Customize next to the Standard version of the form.
   If you already have an existing custom form, click Edit next to this form.

   Important: If you are not sure if the form you want to work with has been upgraded, see How do I tell if the custom forms in my account are upgraded?

3. On the Edit Custom Entry [or Transaction] Form page that appears, click the QuickView tab.
If the page does not have a QuickView tab, this means that the form you want to work with has not yet been upgraded/deployed.

4. In the Field Description column, add the fields you want to appear in the QuickView for records that use this form. After adding each new field, be sure to click the Add button (see figure below). If you want to remove a field, click the X icon that appears to the far right side of the field.

5. Check the Form is Preferred box.

6. Click Save when you are finished configuring your QuickView fields.

Please be aware of these additional guidelines when customizing your QuickViews:

- If you customize a form’s QuickView and select Store Form with Record, these customizations supersede any customization made to a form set to Form is Preferred.
- During form customization, if you make field updates on the Fields tab, you must save the form customization page before those updates appear on the QuickView tab.
- On the Fields tab, if a field is not set to Show, the field does not appear in the QuickView, even if you add the field on the QuickView tab. For example, if the Priority field is not set to Show, then even if Priority is added on the QuickView tab, the Priority field does not appear in the QuickView of a Phone Call record.
- The fields appearing on the QuickView tab match the fields listed in the Description column on the Fields tab. However, in the QuickView it is a field’s custom label that appears.
- Fields that appear on a record only when the record is in Edit mode do not appear in QuickViews. Fields that appear on a record when it is in View mode do appear in QuickViews.

The QuickView list (on the QuickView configuration tab) includes all of the fields that are on a record. The list includes fields that appear on a record when it is in View mode and Edit mode. If you add a field to your customized QuickView that appears only when the record is in Edit mode, the field does not appear in the QuickView when you hover over links to this record. Although there are not many fields in NetSuite that appear on records only when they are in Edit mode, this is one thing to be aware of. If you add a field to a QuickView, yet the field does not appear when the QuickView displays, it may be that you have added a field that is visible only when the record is in Edit mode.

Configuring QuickViews for Non-Upgraded Forms

If you have not upgraded the custom forms in your account to include the look-and-feel changes introduced in a previous version, you cannot directly edit or customize the default QuickView fields assigned by NetSuite. Non-upgraded forms do not include the QuickViews configuration tab on entry and transaction form customization pages.

You can, however, indirectly affect which fields appear in a QuickView by creating an equivalent upgraded form. You can then use the QuickViews tab on the newly upgraded form to make QuickView customizations. You must then set this form to Form is Preferred. By doing so, even the records in your account that use the non-upgraded version of the form show the QuickView customizations of the upgraded form.

Also note the following when attempting to customize QuickViews for non-upgraded forms:

- A non-upgraded stored form shows the customizations of an upgraded preferred form or standard form if there is no preferred form.
- Even if you click Store Form with Record on a non-upgraded form, the QuickViews that appear for records using this form show the QuickView customizations (if any) on the upgraded form.
- If a standard form is marked as preferred, then all Quickviews for both non-upgraded and upgraded forms show the NetSuite default QuickView fields for that record type.
Configuring QuickViews for Custom Records

Unlike built-in standard records, there is no default QuickView for custom records. Because the fields on every custom record are unique, there is no way for the system to set default QuickView fields for all custom records. Therefore, you must create custom record QuickViews yourself by following the steps below.

To configure QuickViews for custom records:

1. Go to Customization > Lists, Records, & Fields > Record Types.
2. Select the custom record type you want to display QuickViews.
3. On the Edit Custom Record Type page, click the Forms tab.
4. Click Customize next to the standard version of the custom record form. If you already have an existing custom form, click Edit next to this form.
5. On the page that appears, click the QuickView tab.
6. In the Field Description column, add the fields you want to appear in the QuickView for this custom record type. After adding each new field, be sure to click the Add button. If you want to remove a field, click the X icon that appears to the far right side of the field.
7. Click Save when you are finished configuring your QuickView fields.

If you go to the list view of this custom record type, you can hover over a record to see the newly configured QuickView. The figure below shows the QuickView for the fixed assets custom record.

How do I tell if the custom forms in my account are upgraded?

Records that use upgraded forms have all fields organized into field groups. (If you are not sure what field groups look like, see Configuring Field Groups.)

If you have noticed any records in your account with a field group layout, then at least one form in your account (the form associated with that record type) has been upgraded.

If you are still not sure if any of the forms in your account have been upgraded, Go to Customization > Forms > Entry Forms [or Transaction Forms]. If any of the forms in your account have been upgraded, you see the Upgrade Status message at the top of your entry or transaction form page. If no forms have been upgraded, the message title says Information and provide details on the form layout enhancements introduced in NetSuite 2010.2.

If forms in your account have been upgraded and you want to see specifically which forms, click Return to Upgrade Checklist.

The page shows specifically which forms have been upgraded and that their upgrades have been Deployed into NetSuite. Forms that have not been upgraded/deployed have Skip Upgrade | Deploy Form links.

To customize a QuickView for any form, the form must be Deployed.

Associating Custom Code (Client SuiteScript) Files With Custom Forms

In the Custom Code subtab, define any existing client SuiteScript files to use with this form. When the script’s executing function is called, the actions defined within the script (as well as any built-in NetSuite
actions for that form type) are performed. For example, if your form calls a Page Init function, the actions defined by that script, as well as any predefined NetSuite actions, are performed prior to the page loading.

⚠️ **Important:** Before associating client SuiteScript files with a form, you must first create the script. For information on creating client SuiteScripts, see the help topic Client Scripts in the NetSuite Help Center.

Note that, if you choose, you can add a custom button to the form to trigger the client script. See Working with Custom Buttons for steps on adding custom buttons to forms.

**Change to Required Permission for Attaching Scripts to Custom Entry and Transaction Forms**

As of 2016.1, users who have the Custom Entry Forms or Custom Transaction Forms permission, but who do not have the SuiteScript permission, can no longer access the Custom Code tab of custom forms. Previously, these users could edit the Custom Code tab of a form record to attach a script to the form.

Now users must have at least the Edit level of the SuiteScript permission to attach a script to a custom form by editing the Custom Code tab of the form record. For users with the Edit or Full level of the SuiteScript permission, the Custom Code tab is displayed and is fully editable. Users with the View or Create level of the SuiteScript permission can see the Custom Code tab, but cannot edit it. For users who do not have SuiteScript permission, the Custom Code tab is not visible.

**To associate client SuiteScript files with a form:**

1. In the **Script File** field, select the client SuiteScript file that contains the desired scripts for this form.
   
   Note that you must upload your file to the File Cabinet before you can select it.

2. In one or more of the client event type fields, enter the names of the functions you want to execute. When entering function names, do not include parentheses or arguments. For example, type `sampleFunction` for a function that appears as `sampleFunction(param1, param2)` in your SuiteScript file.
   
   - In the **Page Init Function** field, enter the name of the executing function to be called when this form is first loaded.
   - In the **Save Record Function** field, enter the name of the executing function to be called when this record is saved.
   - In the **Validate Field Function** field, enter the name of the executing function to be called when a field on this entry form is changed.
   - In the **Field Changed Function** field, enter the name of the executing function to be called when a change made to a field is accepted.
   - In the **Post Sourcing Function** field, enter the name of the function that runs on Post Sourcing events. These events occur following a field change after all the field's child field values are sourced from the server. This enables fieldChange style functionality to occur after all dependent field values have been set.
   - In the **Line Init Function** field, enter the name of the function that runs on Line Init events. These events occur when an existing line is selected.
   - In the **Validate Line Function** field, enter the name of the function that runs on Validate Line events. These events occur prior to a line being added to a sublist (inlineeditor or editor sublists only). It can be thought of as the saveRecord equivalent for sublist line items.
In the **Validate Insert Function** field, enter the name of the function that runs on Validate Insert events. These events occur when you insert a line into an edit sublist. The UI equivalent of this event is when a user selects an existing line in a sublist and then clicks the Insert button. Note that returning false on a validateInsert blocks the insert.

In the **Validate Delete Function** field, enter the name of the function that runs on Validate Delete events. These events occur when you try to remove an existing line from an edit sublist. Returning false blocks the removal.

In the **Recalc Function** field, enter the script name to be called from the attached script file. This script is called when a line item is added. For example, after entering the information you add an item to a transaction.

3. If your client SuiteScript file references any functions in a library file, add the library files in the **Library Script File** list. (You must click the Add button to add your file.)

4. Click **Save**.

### Defining Preferred Forms

Setting preferred forms for your employees lets you control the entry and transaction forms employees use to enter data. This maintains consistency in your company and enables you to capture the information that is most important to your business.

In NetSuite, there are three places where form preferences can be defined.

- **On the Form:** When you create or edit a custom form, you can check the Form is Preferred box to set the current form as the default form for all users where role form preferences are **not** defined. In the Roles tab, you can also define the form as preferred for specific roles.

- **On the Custom Forms page:** On the custom forms page at Customization > Forms > [Forms], you can check the Preferred box for any form you want to set as the default form for all users where role form preferences are **not** defined.

- **On the Manage Roles page:** Administrators can define form preferences for specific roles at Setup > Users/Roles > Manage Roles > Edit [Role]. In addition to setting preferred forms for a role, you can also restrict access to the preferred form.

For custom records, you can choose a preferred form on the Forms subtab or set preferred forms for specific roles on the Permissions subtab. The preferred form you set on the Permissions subtab takes precedence over the preferred form set on the Forms subtab.

For example, you set **Custom Form A** as the preferred form on the Forms subtab. On the Permissions subtab, you set the default form for the Sales Rep role to **Custom Form B**. When a sales rep creates a new record, **Custom Form B** is selected by default.

**Note:** In some cases with transactions, the form that the users choose is selected automatically the next time they create a transaction of that type for their current NetSuite session. For example, users who go to the Invoice Customers page see the Form list set to the form they used the last time they invoiced customers. When users begin a NetSuite session, the preferred form is selected by default on the transaction.

### To set the preferred form for a specific role:

1. Go to Setup > Users/Roles > Manage Roles.
2. Click **Edit** next to the role for which you want to set form preferences.

> **Important:** Standard roles **cannot** be edited. If you want to define new settings for one of these roles, click **Customize** next to the role, enter a name for your custom role and then set the desired custom settings. You must then assign this custom role to the appropriate users.

3. Click the **Forms** subtab.

4. Click the subtab for the section you want to set preferences for.
   - **Transaction**: set defaults for cash refund, cash sale, credit memo, invoice, estimate, opportunity, purchase order, return authorization and sales order transaction forms.
   - **Entity**: set defaults for contact, customer, lead, prospect, employee, project, partner and vendor entry forms. When you define preferences for Entities, you can set a separate preferred form for each Customer subtype in the corresponding fields at the top of the Entity subtab — Lead, Prospect and Subcustomer.
   - **CRM**: set defaults for phone call, campaign, case, event, solution and task entry forms.
   - **Item**: set defaults for inventory part, non-inventory part, service, other charge, description, subtotal, discount, markup, group, kit, assembly and payment item entry forms.

5. Set the form defaults. Note that for some roles, you **cannot** modify all of these settings.
   - In the **Enable** column, clear any boxes for forms you **do not** want this role to have access to (not available for Customer Center roles).
   - Check the box in the **Preferred** column next to any form you want to be set as the preferred form for this role.
   - If you want this form to be the only form available to this role, check the box in the **Restricted** column (not available for Customer Center roles).

   The preferred form settings here override any settings on the Custom Forms page.

   Note the following about marking a transaction or CRM form **Preferred** for the Customer Center role:
   - External forms, meaning forms with names appended with (External), can be marked preferred for the Customer Center roles, but not for other roles.
   - Forms that are not external cannot be marked as preferred for Customer Center roles, so they are not listed on the Forms tab of Customer Center role records.

> **Warning:** If you set preferred forms and do **not** restrict the forms, your employees can still change the form they use when entering transactions or records.

6. Click **Save**.

---

**Defining Preferred Entry and Transaction Forms**

To specify which roles should have this form set as preferred, on the Roles subtab for the form, check the Preferred box for roles that should have this form set as preferred. Preferred forms are selected by default in the Custom Form field when a transaction or record is created.

Note the following about marking an entry or transaction form Preferred for the Customer Center role:

- External forms, meaning forms with names appended with (External), can be marked Preferred for Customer Center roles, but not for other roles. The Customer Center role is available on the Roles subtab only for external forms.
- Forms that are not external cannot be marked Preferred for Customer Center roles.
When a non-online order form is marked Preferred for Customer Center, it is saved as the form for the order. However, an online form is not saved as the form for an order, even if it is preferred. Instead, the preferred non-online order form is used.

**Note:** You can also define preferred forms from the Manage Roles page. **Restrictions** to a form must be defined from the Manage Roles page. For more information, see the help topic **Setting Default Forms for Roles**.

### Adding Disclaimers to Transaction Form Footers

You can add a footer to the bottom of your standard transaction forms. Footers can be used to add text and space for a variety of reasons such as:

- Add a salutation or company slogan
- Familiarize customers with a disclaimer, company policy, or procedure
- Allow space for an approval signature

Footers print at the bottom of your transaction forms.

You can add a disclaimer to your standard forms on the Printing & Fax Preferences page.

With a disclaimer added to the bottom of your forms, you can familiarize customers with your company's policies and procedures.

**To add a disclaimer to standard transaction forms:**

1. Go to Setup > Company > Preferences > Printing & Fax.
2. Click the **Printing** subtab.
3. Enter a disclaimer or other message in one or more of the disclaimer fields. Disclaimers can have up to 4000 characters and appear at the bottom of the form indicated.
   - **Sales Form Disclaimer** – Standard sales forms.
   - **Purchase Form Disclaimer** – Standard purchase forms.
   - **Statement Disclaimer** – Standard statements.
   - **Payment Receipt Disclaimer** – Standard payment receipts.
   - **Packing Slip Message** – Standard packing slips.
   - **Return Policy** – Return forms.
4. When you have finished, click **Save**.

For the disclaimers to appear, you must print your forms using HTML. To set this preference, go to Home > Set Preferences > Transactions tab. Check the Print Using HTML box and click **Save**.

Disclaimers defined here are not used in printed forms based on advanced PDF/HTML templates. To include disclaimers on this kind of printed form, you can add disclaimer text to the template on which the printed form is based.

You can also add disclaimers to specific forms using transaction form customization. For more information on customizing transaction forms, read **Custom Forms**.
Creating Custom Entry and Transaction Forms

Specifying Check Layout by Subsidiary

If your organization uses basic printing, you can configure a different check template for each subsidiary, if required.

To specify check layout by subsidiary in basic printing:

1. Go to.
2. Click Edit beside the subsidiary you want to edit. Then click the Preferences subtab.
3. In the Default Check Type field, to use the standard check type, select Standard. To use the voucher type, select Voucher. For more information, see the help topic Setting Check Printing Preferences.
4. In the Check Default Chart Type field, select the template to use when printing checks from this subsidiary.
5. Click Save.

Customizing Multiple Page Transaction Forms

You can create a custom layout to customize how your multiple-page transaction forms look.

You can add page numbers and header information to subsequent pages of your PDF transaction forms.

To add page numbers to your PDF transaction forms:

1. Go to Customization > Forms > Transaction Forms.
2. Click Customize next to the kind of form you want to change.
3. Click Printing Fields.
4. Click Header.
5. Check the box in the Print/Email column next to Page Number.
   Page number appear beginning on the second page of your transaction.
6. In the Label column you can change the way the numbers appear on your transactions.
   For example, if you wanted the numbers to appear 2/3, you would replace Page {1} of {2} in the label field with (1)/(2). The entries {1} and {2} act as placeholders for your page number formatting.
7. When you have finished, click Save.

Your printed PDF transaction forms now include page numbers for multiple pages.

To add more header information to multiple page transaction forms:

1. Go to Customization > Forms > Transaction Form PDF Layouts (or Transaction Form HTML Layouts).
2. Click Customize next to the kind of layout you want to change.
3. On the Border & Placement subtab, check the boxes in the Show on Addl. Pages column for each field you want to appear on subsequent transaction pages.
4. When you have finished, click Save.

Your printed PDF transaction forms now include additional header information for multiple pages.
Linking Transaction Forms

On the Linked Forms subtab, you can control which transaction form is used when you convert one transaction into another. You can create a chain of transaction forms that mirror your business workflow.

For example, a company has three custom sales order forms that are each used for a certain set of items they sell. When one of these sales orders is used to create a picking ticket, the specific picking ticket form created for each type of sales order is used. The employee creating the picking ticket does not have to search the custom form list to find the proper picking ticket form.

To set up this form workflow, the company administrator edits the custom sales order form, and selects the picking ticket form on the Linked Forms subtab.

When you convert a transaction you created with a custom transaction form, the custom form set on the Linked Forms subtab is selected by default. In the example above, when one of the sales orders is used to print a picking ticket, the custom picking ticket form selected on the sales order form is used by default.

To set up linked forms:

1. Open the list of forms that contains the form you want to create a linkage from.
2. Click Customize or Edit next to the form you want to link from.
3. Enter a name for your custom form.
4. Click the Linked Forms subtab.
   
   A list of the transactions that you can convert to from the type of transaction form you are customizing is shown. For example, if you are customizing an estimate form, you can set linkages for cash sales, invoices, and sales orders.
5. In the Custom Form column, select the standard or custom form for each transaction.
   
   If you want to link to the preferred form, do not select a linked form for that transaction type.
6. Click Save.

When the transaction is converted, the correct transaction form is used automatically.

Note: In the event that the person is assigned a role that is restricted to use only specific transaction forms, the forms set for that role take precedence over the forms you have set on the Linked Forms subtab.

Transaction Form Printing Preferences

NetSuite includes company-level and user-level preferences that impact how transaction forms are printed.

Transaction Forms and Company Printing Preferences

Company printing preferences are defined at Setup > Company > Printing & Fax. The company preferences that impact printed transactions are on the Printing subtab of this page, in the following sections:
Transactions – includes some specialized printing preferences for specific transaction types, as well as general printing preferences for all transaction types.

- For more information about the general printing preference Print Transactions Form Landscape, see Setting Transaction Forms to Print in Landscape.
- For information about which of these preferences are supported for transactions associated with custom forms that use advanced templates, see Advanced Templates Support for Company Printing Preferences.

Advanced PDF/HTML Printing – includes preferences that can be enabled to use advanced templates to format printed Item Detail Statements, Item Labels, and Mailing Labels.

Check Printing – includes preferences for printed checks using basic layouts.

PDFs – includes general preferences that apply to all printed records that use basic layouts except reports.

For more details about the preferences available in these sections, see the help topic Setting Printing and Fax Preferences.

Setting Transaction Forms to Print in Landscape

You can set your transaction forms to print in landscape format on the Printing and Fax Preferences page. When you print in landscape format, the long edge of the page is on top and the short edge is on the side.

By printing in landscape format, you can have wider columns on your transaction forms to include more detailed information for your customers.

**To set transaction forms to print in landscape format:**

1. Go to Setup > Company > Preferences > Printing & Fax.
2. On the Printing subtab, check the **Print Transaction Forms Landscape** box.
3. Click **Save**.

**Note:** To utilize this preference, you must use a standard transaction PDF layout. Use either a standard transaction form with a standard transaction PDF layout or use a custom transaction form associated with a standard PDF layout.

To print transaction forms, go to Transactions > Other > Print Checks and Forms. Alternatively, click **Print** on any transaction page as you save the transaction.

If you print using HTML, you can override the landscape format by changing your printer settings when you print the form.

Transaction Forms and User Printing Preferences

You can set a variety of preferences at Home > Set Preferences to apply to your own instance of NetSuite only. The user preferences that impact printed transactions are on the Transactions subtab of this page.

- Print Using HTML – Check this box if you prefer to print transaction forms in HTML rather than the default of PDF.
- Transaction Email Attachment Form – Choose PDF if you prefer to send email attachments of transactions in PDF rather than HTML.
■ Horizontal Print Offset – Enter a positive number in inches to move text in the printed transaction form to the right. Enter a negative number to move text to the left. This setting applies only to basic layouts.
■ Vertical Print Offset – Enter a positive number in inches to move text in the printed transaction form lower. Enter a negative number to move text higher. This setting applies only to basic layouts.

The PDF preferences on the SuiteAnalytics subtab only apply to printed reports and searches.

Browser Print Settings

When you print a form, a preview page appears to show you what the transaction will look like when it is printed. When you click Print on the preview page, the browser opens a print options page. The print preview looks different in each browser, which is not related to NetSuite. You might need to change the browser print settings or the printer settings so that the printout looks the same as it did in the NetSuite preview.

Creating Custom Subtabs

Custom subtabs are used to organize custom fields on your transaction, entity, CRM, and item records. You should first create custom subtabs and then assign any custom fields to the custom subtabs. For each custom subtab, you can define an existing subtab as the parent subtab. This parent-child definition enables you to include an additional layer of information for your subtab categories.

⚠️ Important: Be aware that for custom subtabs to appear on a record, you must assign a field to the subtab.

To create a custom subtab:

1. Go to Customization > Forms > Subtabs.
2. Click the tab for the type of record for which you want to create a new subtab.
   The following options are available:
   ■ Transaction — subtabs for cash refund, cash sale, credit memo, estimate, invoice, opportunity, purchase order, return authorization, sales order, and other transaction records.
   ■ Entity — subtabs for customer, project, vendor, employee, other name, contact, partner, and group records.
   ■ Item — subtabs for inventory, non-inventory, group, other charge, assembly/bill of materials, kit/package, service item and other item records.
   ■ CRM — subtabs for task, phone call, event, case, campaign, solution, and other CRM records.
3. Enter the name for the subtab in the Title field.
4. If desired, enter translations for this name in the Translation field.
   Before you can add translations, you need to select translation languages at Setup > Company > General Preferences, on the Languages subtab. This subtab lists both system-supported languages that can be used for the NetSuite user interface (and are available at Home > Set Preferences), and additional languages that can used for website translations only (and are not available at Home > Set Preferences). You should only enter translations for system-supported languages, because these are the only languages that can be displayed in the user interface. For details, see the help topic Configuring Multiple Languages.
   The maximum length for a custom subtab's translation is 50 characters.
5. If desired, designate this subtab as a child of an existing subtab in the Parent field, by selecting an existing subtab from the list.

- This list consists of any standard subtabs associated to the selected record type as well as any custom subtabs that you have defined for that record type.
- After you have defined it as a child subtab, a custom subtab is not available to be selected as a parent for another custom subtab. In other words, you cannot create a child-grandchild relationship.

6. Click Add.

7. Repeat these steps for each subtab you want to create.

8. Click Save.

Rearrange your custom subtabs by using the Move Up, Move Down, Move To Top, and Move To Bottom buttons. Insert new subtabs using the Insert button.

Now, when you create a custom field, you can select your new subtab in the Subtab field. After assigning the custom field to the subtab, the subtab then displays on the record.

**Note:** You can rename a custom subtab at any time by going to the appropriate tab on the Custom Subtabs page (Transaction, Entity, Item, CRM), selecting the subtab name and editing it, clicking Done, and saving the Custom Subtabs page.

You can use SuiteCloud Development Framework (SDF) to manage custom subtabs as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom subtab to another of your accounts. Each custom subtab page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

---

### Creating Custom Note Forms

The note form displays as a popup window when a user adds a note to a record. You can vary note form fields, by hiding standard fields, adding custom fields, and changing field order. You can configure note form actions by setting actions to display as buttons or menus, changing labels, hiding standard actions, and adding custom actions. You also can attach form-level script to custom note forms.

To add a custom note form, go to Customization > Forms > Entry Forms, click Customize for the standard note form, and make changes on the Edit Custom Entry Form page.

The note record has a type of Other Record, so its custom fields must be of the type Other Custom Field.

To create a custom field that can be displayed on custom note forms, go to Customization > Lists, Records, & Fields > Other Custom Fields > New. On the New Other Custom Field page, select a Record Type of Note. After you save this type of custom field, the field is displayed by default on all custom note forms.

You can clear the Show box for the custom field on any custom note form where you do not want it displayed.

### Using Custom Note Forms

The standard note form is used by default for all note dialogs. If at least one custom note form has been created in an account, this standard note form includes a Custom Form list where the user can choose to use a custom form for a note dialog. You cannot remove this list from the standard note form as long as any custom note form exists.
Creating Custom Note Forms

You can set a custom note form to be used by default for the note dialog. You can either check the Preferred box for the custom note form in the Entry Forms list, or edit the custom note form and check the Form is Preferred box. If you want to allow users to choose among the standard note form and custom note forms each time they enter a note, you should leave the Custom Form field displayed on custom note forms. To ensure users do not change the note form to anything other than the preferred custom form, on that form you can clear the Show box for the Custom Form field.

Customizing Address Forms

Custom address forms can be used to support address format localization for international customers, and unique business requirements for all customers. A standard address form is provided for all accounts and assigned by default to all countries. Account administrators and other users with the Custom Address Form permission can create as many customized versions of the address form as needed, and assign each custom address form to one or more countries.

When users view or edit addresses on entity, transaction, subsidiary, company information, location, and workplace records, the custom address form assigned to the selected country is the one that displays in the popup address window. Custom address forms inherit the permissions set on the parent record.

Custom address forms support the following capabilities:

- Multiple custom address forms supported per account
- Assignment of custom address forms to countries
- Addition of custom fields to custom address forms (For details, see Creating Custom Address Fields.)
- Rearrangement of standard and custom address fields on forms
- Hiding non-required address fields
- Setting address fields to be mandatory
- Setting address fields display type to be normal, inline text, or disabled
- Customized labels for address fields
- Customized formatting for addresses displayed in UI and printed on shipping labels
- Form-level client scripting

To create a custom address form:

1. Go to Customization > Forms > Address Forms, and click Customize for the standard address form.
2. In the body area of a custom address form record, you can define a name for the form, and modify the template representing how addresses are rendered in the user interface and on printed shipping labels.

   In the Address Template field, address fields are represented by the template field IDs listed on the Fields subtab.

3. On the Fields subtab, you can:
   - Rearrange fields on the form, using the Move to Top and Move to Bottom buttons, and the Column Break, Space Before, and Same Row as Previous fields.
   - Hide fields by clearing the Show box. The Show box cannot be cleared for fields that are required.
   - Make fields required by checking the Mandatory box.

   **Warning:** The Country field is always mandatory, because the value for this field determines the address form to be displayed.

   - Set fields’ display type to inline text or disabled.
   - Modify field labels.
   - Click the New Field button to add a custom field to the address form. For details, see Creating Custom Address Fields.

4. On the Custom Code subtab, you can attach a script file containing functions for any of the supported SuiteScript client events.

   For information about creating scripts to be attached to custom forms, see the help topics Client Scripts and Client Event Types.

5. On the Country subtab, you can assign the custom address form to one or more countries.

   A multi-select list of countries is available when you click the icon to the right of the Country field on this subtab. For more information, see Country-Specific Address Forms.

6. Click Save.

**Change to Required Permission for Attaching Scripts to Custom Address Forms**

As of 2016.1, users who have the Custom Address Form permission, but who do not have the SuiteScript permission, can no longer access the Custom Code tab of custom forms. Previously, these users could edit the Custom Code tab of an address form record to attach a script to the form.

Now users must have at least the Edit level of the SuiteScript permission to attach a script to a custom form by editing the Custom Code tab of the form record. For users with the Edit or Full level of the SuiteScript permission, the Custom Code tab is displayed and is fully editable. Users with the View or Create level of the SuiteScript permission can see the Custom Code tab, but cannot edit it. For users who do not have SuiteScript permission, the Custom Code tab is not visible.

**Scripting Billing and Shipping Addresses**

You can use SuiteScript to create a new address subrecord for an entity and then use that address for custom billing and shipping addresses on transactions.

For more information about using SuiteScript 2.0, see the following.

- Scripting Transaction Shipping and Billing Addresses
Customizing Address Forms

2019.1 Address Scripting Changes to Support Transaction Tax Calculations

Changes to address scripting implemented with the 2019.1 upgrade enhances the accuracy of tax calculations for transactions that have custom addresses.

Some existing scripts may include code to work around previous tax calculation limitations for transactions with custom addresses. These scripts are likely to have unexpected behavior after the 2019.1 upgrade. You may need to update these scripts to ensure that they continue to function correctly after the upgrade.

The 2019.1 release provided the following enhancements to scripts' tax calculations for transactions containing custom addresses.

- **Synchronization of Address Text Values on Subrecord and Parent Record**
- **Elimination of Code Workarounds**
- **Enhanced Override Logic**
- **Enhanced Validation of Country and State Combinations**
- **Avoidance of Unnecessary Form Refreshes**

For examples of how you will need to change existing address script code to adapt to 2019.1 behavior changes, see: Address Scripting Code Snippets.

### Synchronization of Address Text Values on Subrecord and Parent Record

Before 2019.1, in SuiteScript 1.0, the value that scripts returned for addrtext on an address subrecord, and for shipaddress on a parent record, may have been different. After the 2019.1 upgrade, these values are in sync.

**Recommended Action:** Review existing scripts to see if any of them are impacted negatively by the synchronization of subrecord and parent record address text values, and modify scripts as needed to avoid issues.

### Elimination of Code Workarounds

Before 2019.1, scripts that set custom address values on transactions did not automatically result in tax recalculation. This recalculation was required because the tax jurisdiction, or nexus, may change due to address values such as country or state. A nexus change is likely to cause a change in a transaction's tax amount. To avoid this limitation, existing scripts required workaround code to ensure correct tax calculation.

- **SuiteScript 1.0 scripts** that use legacy address fields required code that explicitly set the shipaddress field value, to override the incorrect default tax calculation. They also required code that explicitly set the shipaddresslist field value to -2, to set the correct nexus.
Customizing Address Forms

- SuiteScript 1.0 scripts that use the address subrecord, and SuiteScript 2.0 scripts, required code that explicitly set the shipaddress, shipcountry and shipstate parameters for the transaction, in order to set the correct nexus.

As of 2019.1, any change to custom address fields on transactions automatically recalculates the nexus and recalculates the tax amount according to the updated nexus.

- You no longer have to explicitly set shipaddress or shipaddresslist, or shipaddress, shipcountry, and shipstate, to ensure correct tax calculation. After the 2019.1 upgrade, code that explicitly sets these values will no longer be needed, and this code may cause unwanted overrides of system tax calculation.

- Scripts that set custom address values on transactions are able to automatically trigger nexus changes. Because any nexus change can cause a form to reload without preserving unsaved data, it is possible that some scripts may cause unwanted form reloads and data loss after the 2019.1 upgrade. Whether this issue occurs depends on the point in the transaction creation or update process at which your scripts set custom address values.

**Recommended Actions:** You should review your existing scripts and make the following updates:

- Remove workaround code that is no longer required.
- Verify that your scripts do not interfere with transaction creation in a way that will lead to unwanted form reloads. Also check to ensure that taxes are computed correctly for different nexuses. Update scripts as needed.

**Enhanced Override Logic**

Before 2019.1, when scripts set shipaddress or addrtext field values but did not set shipoverride to T, shipaddress or addrtext was replaced by the default (system-generated) address text value.

As of 2019.1, system logic treats direct manipulation of shipaddress, billaddress, or addrtext values as an intention to override the default address text value. In this case, the system sets the shipoverride field to T, and does not replace values for shipaddress, billaddress, or addrtext with the default address text value.

**Recommended Action:** Review your scripts to determine whether they are dependent on the previous behavior. Update scripts as needed.

**Enhanced Validation of Country and State Combinations**

Before 2019.1, when a script customized a default address on a transaction, the default state was used even when it was not valid for the selected country. This handling could result in invalid country-state combinations.

As of 2019.1, when a script changes the country, and the state is not a valid value for the selected country, the state field value is cleared.

**Recommended Action:** Review your scripts to ensure that they do not depend on the state value always being kept, even when it is invalid. Update scripts as needed.

**Avoidance of Unnecessary Form Refreshes**

Before 2019.1, whenever a script set the country value, the form was refreshed, even if the value remained the same as it was before. This behavior could result in form data being erased.

As of 2019.1, a form refresh does not occur if the country value is set to the same value it was previously.
**Recommended Action:** Review your scripts to ensure that they do not depend on the form being refreshed whenever the country value is set. Update scripts as needed.

**Note:** NetSuite 2019.1 includes another improvement to the handling of address updates involving country values. Before 2019.1 when the country was changed for an address, all other data that was changed at the same time as the country was not saved. So for example, if a script changed both a country and city, only the new country value was saved. As of 2019.1, any other changes that are made to the address are saved along with the new country value.

**Address Scripting Code Snippets**

The following code snippets illustrate the differences between previous code to script addresses and code to be used as of 2019.1.

- **Setting a Custom Address Using Deprecated Legacy Fields in SuiteScript 1.0**
- **Setting a Custom Address Using a Subrecord in SuiteScript 1.0**
- **Setting a Custom Address Using a Subrecord in SuiteScript 2.0**

**Setting a Custom Address Using Deprecated Legacy Fields in SuiteScript 1.0**

**Before 2019.1:**

```javascript
var rec = nlapiCreateRecord('salesorder',{ recordmode: 'dynamic' });
rec.setFieldValue('entity', 2);
rec.setFieldValue('shipaddresslist', '-2'); //CHANGE_INFO: This line is needed to preserve address after form refresh and for correct behavior in UI
rec.setFieldValue('shipcountry', 'GB');
rec.setFieldValue('shipzip', '12345');
rec.setFieldValue('shipaddressee', 'Arrow ECS Australia Pty Ltd');
rec.setFieldValue('shipaddr1', 'Unit 24, Hume Ave');
rec.setFieldValue('shipaddr2', 'Park West Business Park');
rec.setFieldValue('shipcity', 'Dublin');
rec.setFieldValue('shipstate', 'Nairnshire');
rec.setFieldValue('shipaddress', 'Unit 24, Hume Ave
Park West Business Park
Dublin
12345 Nairnshire'); //CHANGE_INFO: This line triggers nexus recalculation
rec.selectNewLineItem('item');
rec.setCurrentLineItemValue('item', 'item', 98);
rec.commitLineItem('item');
var id = nlapiSubmitRecord(rec);
```

**As of 2019.1:**

```javascript
var rec = nlapiCreateRecord('salesorder',{ recordmode: 'dynamic' });
rec.setFieldValue('entity', 2);

//CHANGE_INFO: shipaddresslist is set to -2 automatically right after address components are touched on next lines, so this line is no longer mandatory
rec.setFieldValue('shipcountry', 'GB');
rec.setFieldValue('shipzip', '12345');
rec.setFieldValue('shipaddressee', 'Arrow ECS Australia Pty Ltd');
rec.setFieldValue('shipaddr1', 'Unit 24, Hume Ave');
```
Customizing Address Forms

rec.setFieldValue('shipaddr2', 'Park West Business Park');
rec.setFieldValue('shipcity', 'Dublin');
rec.setFieldValue('shipstate', 'Nairnshire');

//CHANGE_INFO: If you want shipaddress to be generated automatically from address components, you must remove this line. When you set “shipaddress” manually, we'll check "override" field, so you should consider if that’s what you expect to happen. It is no longer needed for nexus to be calculated correctly.
rec.selectNewLineItem('item');
rec.setCurrentLineItemValue('item', 'item', 98);
rec.commitLineItem('item');
var id = nlapiSubmitRecord(rec);

Setting a Custom Address Using a Subrecord in SuiteScript 1.0

Before 2019.1:

var rec = nlapiCreateRecord('salesorder',{ recordmode: 'dynamic' });
rec.setFieldValue('entity', 2);
rec.setFieldValue('shipaddresslist', '-2'); //CHANGE_INFO: This line is needed in order to preserve address after form refresh and for correct behavior in UI
var subrecord = rec.createSubrecord('shippingaddress');
subrecord.setFieldValue('country', 'GB');
subrecord.setFieldValue('zip', '12345');
subrecord.setFieldValue('addressee', 'Arrow ECS Australia Pty Ltd');
subrecord.setFieldValue('addr1', 'Unit 24, Hume Ave');
subrecord.setFieldValue('addr2', 'Park West Business Park');
subrecord.setFieldValue('city', 'Dublin');
subrecord.setFieldValue('state', 'Nairnshire');
subrecord.commit();

rec.setFieldValue('shipaddress', 'Unit 24, Hume Ave
Park West Business Park
Dublin
12345 Nairnshire'); //CHANGE_INFO: This line triggers nexus recalculation
rec.selectNewLineItem('item');
rec.setCurrentLineItemValue('item', 'item', 98);
rec.commitLineItem('item');
var id = nlapiSubmitRecord(rec);

As of 2019.1:

var rec = nlapiCreateRecord('salesorder',{ recordmode: 'dynamic' });
rec.setFieldValue('entity', 2);
//CHANGE_INFO: shipaddresslist is set to -2 automatically right after address components are touched on next lines, so this line is no longer mandatory (but it’s presence does no harm)
var subrecord = rec.createSubrecord('shippingaddress');
subrecord.setFieldValue('country', 'GB');
subrecord.setFieldValue('zip', '12345');
subrecord.setFieldValue('addressee', 'Arrow ECS Australia Pty Ltd');
subrecord.setFieldValue('addr1', 'Unit 24, Hume Ave');
subrecord.setFieldValue('addr2', 'Park West Business Park');
subrecord.setFieldValue('city', 'Dublin');
subrecord.setFieldValue('state', 'Nairnshire');
subrecord.commit();
Customizing Address Forms

//CHANGE_INFO: If you want shipaddress to be generated automatically from address components, you must remove this line. When you set 'shipaddress' manually, we'll check 'override' field, so you should consider if that's what you expect to happen. It is no longer needed for nexus to be calculated correctly.

rec.selectNewLineItem('item');
rec.setCurrentLineItemValue('item', 'item', 98);
rec.commitLineItem('item');
var id = nlapiSubmitRecord(rec);

Setting a Custom Address Using a Subrecord in SuiteScript 2.0

Before 2019.1:

```javascript
/**
 * @NApiVersion 2.x
 * @NScriptType ScheduledScript
 * @NModuleScope SameAccount
 */
require([ 'N/record' ],
function(record) {

    var salesOrderRecord = record.create({
        type : record.Type.SALES_ORDER,
        isDynamic : true,
        defaultValues : {
            entity: 2,
            //CHANGE_INFO: These three parameters had to be added in order to correctly initialize tax nexus on SS 2.0 record
            shipaddresslist: 'cust',
            shipcountry: 'GB',
            shipstate: 'Nairshire'
        }
    });

    var shippingAddressSubrecord = salesOrderRecord.getSubrecord({
        fieldId : 'shippingaddress'
    });

    shippingAddressSubrecord.setValue({
        fieldId : 'country',
        value : 'GB'
    });
    shippingAddressSubrecord.setValue({
        fieldId : 'attention',
        value : 'Arrow ECS Australia Pty Ltd'
    });
    shippingAddressSubrecord.setValue({
        fieldId : 'addr1',
        value : 'nit 24, Hume Ave'
    });
    shippingAddressSubrecord.setValue({
        fieldId : 'city',
        value : 'Dublin'
    });
});
```
shippingAddressSubrecord.setValue({
    fieldId : 'state',
    value : 'Nairshire'
});

shippingAddressSubrecord.setValue({
    fieldId : 'zip',
    value : 2000
});

salesOrderRecord.selectNewLine({
    sublistId : 'item'
});

salesOrderRecord.setCurrentSublistValue({
    sublistId : 'item',
    fieldId : 'item',
    value : 98
});

salesOrderRecord.commitLine({
    sublistId : 'item'
});

var salesOrderInternalId = salesOrderRecord.save();

As of 2019.1:

/**
 * @NApiVersion 2.x
 * @NScriptType ScheduledScript
 * @NModuleScope SameAccount
 */
require([ 'N/record' ],
function(record) {

    var salesOrderRecord = record.create({
        type : record.Type.SALES_ORDER,
        isDynamic : true,
        defaultValues : {
            entity: 2
        }
    });

    var shippingAddressSubrecord = salesOrderRecord.getSubrecord({
        fieldId : 'shippingaddress'
    });

    shippingAddressSubrecord.setValue({
        fieldId : 'country',
        value : 'GB'
    });

    shippingAddressSubrecord.setValue({
        fieldId : 'attention',
        value : 'GB'
    });
Creating Custom Address Fields

If you are an account administrator or have another role with the Custom Fields permission, you can create a custom field to be included in one or more custom address forms.

**To create a custom address field:**

1. Click the **New Field** button on the **Fields** subtab of a custom address form record, or go to Customization > Lists, Records, & Fields > Other Custom Fields > New.
2. In the **Record Type** field, select **Address**.
3. Enter a label for the field and complete other body fields as desired.
4. On the **Applies To** subtab, choose whether to apply the address field to all custom address forms or to selected custom address forms.
   - The default is to apply the field to all custom address forms.
   - If you choose the **Apply to Selected Custom Address Forms** option, select a custom form in the **Address Form** list. Click **Add** to select multiple forms.
5. Complete fields on other subtabs as desired. For details, see the following:
Customizing Address Forms

- Setting Display Options for Custom Fields
- Setting Validation and Defaulting Properties
- Setting Sourcing Criteria
- Setting Filtering Criteria
- Restricting Access to Custom Fields

6. Click Save.

Country-Specific Address Forms

Address forms formatted for certain countries are provided by the Country-Specific Address Forms SuiteApp. When you use a country-specific address form, the system applies the address template to both the user interface and printed forms.

Currently, custom address forms are available for the following countries:
- Australia
- Brazil
- Canada
- Chile
- Colombia
- Mexico
- New Zealand
- Peru
- United Kingdom
- USA

When you view or edit addresses on records, the custom address form assigned to the selected country is displayed in the popup address window. The country-specific address forms apply to the following records:
- Entity (company, subsidiary, contact, vendor, employee, etc.)
- Transaction
- Location
- Workplace

To use country-specific address forms, read the following topic:
- Installing Country-Specific Address Forms

Installing Country-Specific Address Forms

The Country-Specific Address Forms SuiteApp provides custom address forms for Australia, Brazil, Canada, Chile, Colombia, Mexico, New Zealand, Peru, United Kingdom, and USA.

To install the Country-Specific Address Forms SuiteApp:
1. Go to Customization > SuiteBundler > Search & Install Bundles.
2. In the Keywords field, enter Country-Specific Address Forms, or the bundle ID, 64313.
3. Click Search.
4. Click Country-Specific Address Forms in the Name column.
5. On the Bundle Details page, click Install.
Customizing Address Forms

For more information about installing SuiteApps, see the help topic Installing a Bundle.
The Country-Specific Address Forms SuiteApp is a managed bundle. After you install the SuiteApp, your accounts are automatically upgraded when an update to the SuiteApp is available.

Custom Sublists

You can add a custom sublist to any transaction or entry form, including forms for custom record types. Custom sublists use results from a selected saved search of the form record type or a related record type. The custom sublists use the saved search results to present information related to the record you are viewing. The following are examples:

- You could attach a customers sublist on customer records listing other customers with the same sales representative. The sublist is based on a customers search with a filter of Sales Rep.
- You could attach an employees sublist on employee records listing other employees with the same supervisor. The sublist is based on an employees search with a filter of Supervisor.
- You could attach a transactions sublist on customer records listing transactions for that customer, based on a transactions search with a filter of customer name.

The creation of custom sublists is controlled by the Custom Sublists permission. You can assign this permission to a custom role on the Setup subtab of the role record.

You can use SuiteCloud Development Framework (SDF) to manage custom sublists as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom sublist to another of your accounts. Each custom sublist page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

For details about creating saved searches to be used as custom sublists and applying custom sublists to records, see:

- Saved Searches for Custom Sublists
- Applying Custom Sublists to Standard Records
- Applying Custom Sublists in SDF-Enabled Accounts
- Applying Custom Sublists to Custom Record Types
- Custom Child Record Sublist Creation with SuiteScript

Saved Searches for Custom Sublists

To be used for a custom sublist, a saved search must have at least one list/record type field defined as an available filter. Custom sublist results are joined to the form record based on this field’s values. That is, the first item listed on the Available Filters subtab of the saved search must be a list/record type. The custom sublist results that are displayed on each form have the same value for the available filter field as the record currently displayed on the form.

For example, in a saved search for a transactions sublist on a customer record, you could set an available filter of Name (the customer name) so that all transactions for the customer show in the sublist.

For details about available filters, see the help topic Selecting Available Filters for Saved Searches.

Be aware of the following when you choose a field to be the available filter for a custom sublist search:

- If the saved search has more than one available filter, only the first one listed on the Available Filters subtab is used to filter the custom sublist.
- The available filter field must be of the List/Record type, meaning it is displayed in the user interface as a list. Constant value fields and numeric fields do not work.
Note that the searches you can select for custom sublists do not include searches where the type of the first available filter is check box or date, because these types of fields do not work.

- If the sublist represents a different record than the form record, the available filter field should be a logical choice to produce joins with the form record. Many fields that are listed on a search’s Available Filters subtab are not logical choices for this type of join.

For example, in a saved search for a transactions sublist on customer record, it would not work to set an available filter of Type. This choice would most likely result in no joins with the customer record, and a blank sublist would display.

To be a good choice as an available filter, a field should identify a record type.

- The filter field can be the same as the form record. For example, use the Name field for a transactions sublist on a customer record that displays all of that customer’s transactions.

- Or, the filter field can be a field on the form record that is a list of records. For example, use the Location field for a transactions sublist on a customer record that displays the customer’s transactions for a particular location only.

To create a saved search:

1. Go to Reports > Saved Searches > New Saved Search. Click the record type you want to display in the custom sublist.
   The results of this search should include the information you want to show on your sublist.
   For information about creating saved searches, see the help topic Defining a Saved Search.
2. Enter a name for the search.
3. Set criteria for the search.
4. On the Available Filters subtab, add at least one available filter to the search, and ensure that the first filter listed is a List/Record type. Otherwise, the saved search is not available to assign as a sublist.
   You can only attach a saved search as a sublist if the search has at least one available filter. The sublist shown is filtered by the first filter listed on the saved search. Be sure that the search and the filter both apply to the record you are attaching the sublist to.
   For information about setting available filters, see the help topic Selecting Available Filters for Saved Searches.
5. Click Save.

Applying Custom Sublists to Standard Records

You can add a custom sublist to any standard entity or transaction record, including custom transaction and entry forms.

For example, you might want to show a list of customers on the sales order form that are assigned to the sales representative on the order.

**Note:** If SDF is enabled in your account, you use a different page for adding custom sublists. For more information, see Applying Custom Sublists in SDF-Enabled Accounts.

To add a custom sublist to a standard record:

1. First, create a saved search for the information you want to show. The results of this search should include the information you want to show on your sublist. See Saved Searches for Custom Sublists.
2. Go to Customization > Forms > Sublists.
3. Click the subtab that corresponds with the kind of record to which you want to add the sublist.
4. In the **Search** column, select the saved search that returns the results you want to appear on the record.

   If the saved search does not appear in the list, check the saved search settings. The first item listed on the Available Filters subtab must be a List/Record type, otherwise the saved search is not available to assign as a sublist.

5. Enter a label for this sublist.

6. In the **Tab** column, select the subtab where you want the sublist to appear.

7. Check the box for each record where you want this sublist to appear.

   This sublist shows on the standard and custom forms of the types you select.

8. In the **Field** column, select the field by which you are filtering the sublist.

9. Click **Add**.

10. Click **Save**.

Now, these search results show on the records you selected. For an example of applying custom sublists to standard records, see [Example of Adding Custom Sublist to Standard Record](#).

When a transaction is created by transformation from another record type, custom child record sublists are available for editing on the new transaction. Sublists are also available when users select the Make Copy action on an original transaction. The sublists are empty to prevent problems when copying child records, so the user should enter new sublist values.

### Example of Adding Custom Sublist to Standard Record

You want to add a custom sublist to the employee record to display customers who have not had any activity in the last month.

First, create a saved search that shows the information that you want to see in the sublist. On the **Criteria > Summary** subtab, enter the following:

- **Summary Type** — Maximum
- **Field** — Activity : Date
- **Description** — is not after 1 month ago

On the **Results** subtab, set the summary type for Name to Group, and set Activity : Date to Maximum.
On the Available Filters subtab, the first filter listed must be of type List/Record for this saved search to be available as a custom sublist. The ID of a record is used as the parameter for this filter. When a sublist has the Field column defined, the value from the selected field is used instead of the record ID.

Set up the custom sublist by selecting the saved search and applying it to the required records.

In this example, the Field column in the Custom Sublists page was left blank because the filtering done on the Available Filters subtab in the saved search is sufficient.

The sublist appears as a subtab on the employee record.
Applying Custom Sublists in SDF-Enabled Accounts

If SDF is enabled in your account, you use a different page for viewing and adding custom sublists in the UI. For more information about custom sublists in SDF, see the help topic Sublists as XML Definitions.

To view available custom sublists, go to Customization > Forms > Sublists > List.

To add a custom sublist in an SDF-enabled account:

1. Create a saved search for the information you want to show. The results of this search should include the information you want to show on your sublist.
2. Go to Customization > Forms > Sublists > New. The Custom Sublist page appears.
3. In the Script ID field, enter an ID for the custom sublist, starting with an underscore.
4. In the Tab field, select the subtab where you want this sublist to appear.
5. In the Type field, select the sublist type.
6. In the Search field, select the saved search that returns the results you want to appear on the record. If the saved search does not appear in the list, check the saved search settings. The first item listed on the Available Filters subtab must be a List/Record type, otherwise the saved search is not available to assign as a sublist.
7. In the Label field, enter a label for this sublist.
8. In the Field field, select the field by which you are filtering the sublist.
9. Check the box for each record where you want this sublist to appear. This sublist shows on the standard and custom forms of the types you select.
10. Click Save.

Now, these search results appear on the subtab of the records you selected. To view an example of applying a custom sublist to a standard record, see Example of Adding Custom Sublist to Standard Record.
Applying Custom Sublists to Custom Record Types

You can add custom sublists to custom record types.

First, you create a saved search that obtains the results that you want to show on your custom records. See Saved Searches for Custom Sublists. Then you add the search as a custom sublist to your custom record type.

**Note:** If the saved search is not available in the sublist setup, check the saved search settings. The first item listed on the Available Filters subtab must be a List/Record type, otherwise the saved search is not available to assign as a sublist.

To add the search as a custom sublist to a custom record type:

1. Go to Customization > Lists, Records, & Fields > Record Types. Click the name of the record type to which you want to add the sublist.
2. Click the Sublists subtab.
3. In the Search column, select the saved search.
4. In the Label column, enter a label for the subtab where you want the search results to display.
5. (When the Multi-Language feature is enabled) In the Translation column, enter one or more translations of the label. You can enter up to 50 characters in the Translation column.
6. In the Tab column, choose the custom subtab where you want this sublist to display.
7. In the Field column, select the field by which you are filtering this sublist.
8. Click Add.
9. Click Save.

Now, these search results show on the custom record. For an example of applying a custom sublist to a custom record type, see Example of Adding Custom Sublists to Custom Record Types.

**Example of Adding Custom Sublists to Custom Record Types**

You have a custom record called Warranty Claim, and when a new record is created, you want to show information about previous claims.

First you create the saved search. From the custom record list page, click Search beside the record for which you want to create the custom sublist.
You want to filter the search to show only the warranty claims that were worked by the same technician. On the Available Filters subtab, the first filter listed must be of type List/Record for this saved search to be available as a custom sublist.

In the custom record definition, click the Sublists subtab and select the search that you created.

The sublist appears as a subtab on the custom record.
Custom Child Record Sublist Creation with SuiteScript

- Custom Child Record Sublists
- Creating Custom Child Record Sublists
- Custom Child Record Sublist IDs Overview
- Scripting with Custom Child Record Sublists

**Custom Child Record Sublists**

When working with custom child record sublists you can use all the `N/record Module` or `N/currentRecord Module` provided in SuiteScript. If you are not familiar with custom child record sublists, it is recommended that you read these topics in order:

- Custom Child Record Sublists Overview
- Creating Custom Child Record Sublists
- Custom Child Record Sublist IDs Overview
- Scripting with Custom Child Record Sublists

**Custom Child Record Sublists Overview**

Custom child record sublists are Inline Editor Sublists that contain a list of custom records.

You can do the following tasks with Inline Editor Sublists:

- you can add/edit/remove lines dynamically prior to submitting the form
- you can add/edit/remove lines using the UI or SuiteScript
- when writing client scripts, use `Record.commitLine(options)` or `CurrentRecord.commitLine(options)` after each sublist line change. Otherwise your changes will not be committed to NetSuite.
- when writing server scripts, you must call `Record.commitLine(options)` or `CurrentRecord.commitLine(options)` to commit sublist updates. Note that you must do this in addition to calling `Record.save(options)`, which commits the entire record object to the database.

The following screenshot shows a custom child record sublist of fixed assets records. These records appear as line items on a custom Fixed Assets subtab. The parent record that contains the sublist of custom fixed assets records is the customer record.

In the UI, click **New Fixed Assets** to create a new fixed assets child record. The new record is added to the Fixed Assets sublist and is scriptable. In SuiteScript, add a new fixed assets child record by adding a new line to the sublist.
When the **New Fixed Assets** button is clicked, a new fixed assets record opens (see the following screenshot). Note that the fixed assets record contains a **New Customer** field. This field is a List/Record field that references the parent record – in this case, the Abe Simpson customer record.

**Note:** The parent–child relationship between the fixed assets record type and the customer record type was defined on the Custom Record Type definition page for the fixed assets record. (For general information on creating parent–child relationships between records, see [Parent-Child Record Relationships](#).)

The Fixed Assets fields that appears in the sublist are the mandatory fields (those body fields that appear with the asterisk on the fixed assets record) and those fields that have been set to **Show in List** in the Custom Field definition page for the fixed assets record type. You can use **N/record Module** or **N/currentRecord Module** to set or get values for all fixed assets fields in a Fixed Assets sublist.

The New Customer field is a List/Record field that references the parent record of the fixed assets (child) record.

If you know the internal ID of a fixed assets field, you can update the field through sublist scripting. (See [Custom Child Record Sublist IDs Overview](#) to learn how to get field IDs for all fields on a custom child record sublist.)
Fields set to **Show in List** on the fixed assets record type appear in the sublist. However, you can update any fixed assets field via scripting.

When a transaction is created by transformation from another record type, custom child record sublists are available for editing on the new transaction. Sublists are also available when users select the Make Copy action on an original transaction. The sublists are empty to prevent problems when copying child records, so the user should enter new sublist values.

### Creating Custom Child Record Sublists

The following high-level steps are provided for users who are already familiar with NetSuite customization. This procedure provides a general frame of reference for building a custom child record sublist. For more information, see the following.

- Custom Records
- Custom Fields
- Creating Custom Subtabs

**To create a custom child record sublist:**

1. Define a custom record type (such as the fixed assets record mentioned earlier).
Important: On the Custom Record Type page, check the Allow Child Record Editing box. If this preference is not enabled, your records are not scriptable when they appear as sublist line items on the parent record.

2. Establish the parent–child relationship between your new custom record type (fixed assets) and another record type. Parent–child relationships are established through custom fields.
   a. Add a custom field to your new custom record type.
   b. On the field definition page for the new field, set the field Type to List/Record (see the following screenshot).
   c. Specify the record type that is the parent of your custom record type. In the following screenshot, the customer record type is the parent.
   d. Check the Record is Parent box to attach your custom record type (fixed assets) to a parent record type (customer).

In this case, the New Customer field ties the fixed assets record type to the customer record type.
Your custom child record sublist can appear on a standard or custom subtab of the parent record.

3. If you want the custom child record sublist to appear on its own subtab on the parent record, create a subtab with a name that reflects the sublist type. The preceding screenshot shows that a sublist of child fixed assets records appears on a custom Fixed Assets subtab. This subtab appears on all customer (parent) records.

Note: See Adding Subtabs to a Custom Record for steps on creating custom subtabs and adding subtabs to specific record types.

4. After defining the customer–fixed assets (parent–child) relationship (via the New Customer field), go to a customer record in NetSuite and notice the Fixed Assets sublist (see the following screenshot).
**Note:** If you have not specified a parent subtab for this sublist, the custom record child sublist appears on a system-generated subtab called **Custom**.

The preceding screenshot shows the fixed assets sublist. When the **New Fixed Assets** button is clicked, a custom (child) fixed assets record opens. After adding data to the fixed assets record and saving it, the record appears as a sublist line item.

The following screenshot shows that three fixed assets (child) records have been added as sublist line items to the (parent) customer record for Abe Simpson.
Custom Child Record Sublist IDs Overview

Unlike other sublists, there are no standard IDs that can be documented for custom child record sublists. The internal ID for the sublist itself, and all of its associated fields, is unique to each custom child record sublist.

See these topics for guidelines on determining which IDs to reference in N/record Module or N/currentRecord Module:

- Determining the Sublist ID
- Determining Field IDs

Determining the Sublist ID

The internal ID for a custom child record sublist is `recmach + field_id_for_the_parent_field` (for example: `recmachcustrecord111`).

Use `Record.getSublistValue(options)` or `CurrentRecord.getSublistValue(options).`

```javascript
// Add additional code.
...
var sublistFieldValue = objRecord.getSublistValue({
    sublistId: 'item',
    fieldId: 'item',
    line: 3
});
```
The following steps describe where to look in NetSuite to get the internal ID of a custom child record sublist.

**To get the internal ID of a custom child record sublist (the field ID for the parent record):**

1. Go to the record definition for the custom record type (see the following screenshot).
2. On the **Fields** subtab, notice that the **ID** column lists a List/Record field type.

The **New Customer** field (internal ID: **custrecord111**) is the parent field for the fixed assets records that appear as children on customer records. Hence, the internal ID for the custom child Fixed Assets sublist on customer records is **recmachcustrecord111**.

The following is an alternative approach for obtaining the internal ID of the parent field.

1. On the custom child record sublist, click the New child record button (see the following screenshot for a general example).
2. When the new child record opens (see the following screenshot), notice the field that ties the child record to the parent record.

In this case, the **New Customer** field shows that the customer record for Abe Simpson is the parent of the fixed assets record. The field level help popup window for **New Customer** lists the field’s internal ID as custrecord111. Therefore, the internal ID for the Fixed Assets sublist appearing on the (parent) customer record is **recmachcustrecord111**.
**Note:** Internal IDs for custom child record sublists also appear in the SuiteScript Debugger when you load the record that includes the sublist.

### Determining Field IDs

Use these steps to get internal field IDs on a custom child record sublist:

1. Go to the custom record definition page (for example, go to and select your custom record in the **Record Types** list).
2. On the Fields subtab of the Custom Record Type page (see the following screenshot), all field internal IDs appear in the ID column. These are the IDs you reference as the **fieldId** value in `Record.getSublistValue(options)` or `CurrentRecord.getSublistValue(options).

Example:

```javascript
//Get the value of the Cost field on the first line (see the following figure)
...
var sublistFieldValue = objRecord.getSublistValue({
  sublistId: 'irecmachcustrecord111',
  fieldId: 'custrecord1',
  line: 1
});
```

You can get or set values for fields that appear in the Fixed Assets sublist. You can also set or get values that exist on the record, but do not appear in the sublist UI.

For example, the following line sets the value of the **Salvage Value** field in the fixed assets record 3 (see the following screenshot). The internal ID for Salvage Value is custrecord2. See this value on the Custom Record Type definition page for the fixed assets record type.

```javascript
objRecord.setCurrentSublistValue({
  sublistId: 'irecmachcustrecord111',
  fieldId: 'custrecord2',
});
```
Custom Sublists

Scripting with Custom Child Record Sublists

Custom child record sublists are inline editor sublists. Consequently, they support all standard Sublist APIs that run on other inline editor sublists. A custom child record sublist is unique only because it is not identified by a standard sublist internal ID, nor does it contain a standard set of field IDs. Otherwise, like all other inline editor sublists, you can add and remove line items. You can get and set values on existing line items, and the first line number (linenum) for all sublists is 1, not 0.

**Important:** Be aware that you cannot execute validate line functions on custom child record sublists. Validate line functions are executed when a client event occurs prior to a line being added to a sublist.

Adding a record to a custom child record sublist

The following snippet shows how to add a new fixed assets record to the Fixed Assets sublist. This is a server-side script in which the customer record object is loaded into the system, and the Fixed Assets sublist (recmachcustrecord111) is being accessed through methods on the nlobjRecord object.

```javascript
var rec = nlapiLoadRecord('customer', 142);
// Call the nlobjRecord selectNewLineItem method to add a new line.
// Note: Call selectLineItem(...) if the line already exists and you are just updating it.
rec.selectNewLineItem('recmachcustrecord111');
// Set the value for the record name
rec.setCurrentLineItemValue('recmachcustrecord111', 'name', 'Printer');
// Set the value for the Cost field
rec.setCurrentLineItemValue('recmachcustrecord111', 'custrecord1', 1000);
```
//Set the value for the Useful Life in Years field
rec.setCurrentLineItemValue('recmachcustrecord111', 'custrecord4', '3');
//Commit your sublist changes
rec.commitLineItem('recmachcustrecord111');
//Submit the updated Customer record
var id = nlapiSubmitRecord(rec, true);

Customizing a Transaction Sublist

On most pages where you run bulk operations on a list of transactions, you can customize the list displayed. For example, on the Approve Sales Orders page, you can change the default column, filter, and sorting options. If a transaction list is customizable, a Customize button is available in the upper left corner of the sublist. For more information, see the help topic Customizing Sublist Views.

Note: The Customize button is useful for displaying information that is captured by a custom field defined in your account. Custom fields are not included in the transaction sublist by default.

To customize a transaction sublist:

1. On the transaction page, click Customize.
   For example, go to Transactions > Sales > Mark Orders Shipped. Click Customize.
2. On the Additional Columns subtab, check any fields that you want to display as columns in the sublist.
You can add columns to the list, but you cannot remove any of the default columns.

3. On the **Additional Filters** subtab check any fields that you want to add as filters for the sublist.

**Important:** In the **Adjust Inventory Worksheet** and **Print Item Labels** pages, the available column and filter fields are taken from the item record rather than from transactions.

**Example: Sorting by Bill Type**

You might want to delay paying bills until certain criteria are met. To identify these bills on the Bill Payments page, you can add a custom **On Hold** field to the bill record. Add this as filter criteria to the Bill Payments page and then sort by the desired setting to view bills that you must pay versus bills that should wait.

To set up this scenario, create a custom Transaction Body field of the Type Check Box and apply it to purchase records.

After bills are entered, go to the Bill Payments page, click Customize View and then select On Hold on the Additional Filters subtab.

Add the On Hold field to the Results subtab and save.
You can now filter results to display only records that are not On Hold.
Advanced PDF/HTML Templates

The Advanced PDF/HTML Templates feature supports an alternate model for customizing printed and emailed transactions. This model supports more customization capabilities than transaction form layouts, also known as basic layouts. (Basic layouts were previously known as legacy layouts.) To use advanced templates in your account, see Enabling the Advanced PDF/HTML Templates Feature.

When this feature is enabled, you can associate advanced templates with custom transaction forms, so that these templates are used to format printed and email versions of transactions. See Setting Custom Forms to Use Advanced Templates.

You can use advanced PDF/HTML templates to produce either PDF or HTML output, depending upon the settings of your print and email preferences. See Advanced Templates Support for Company Printing Preferences.

Advanced PDF/HTML templates support all transaction and print types supported by basic layouts, including internationalized versions. For a list, see Reviewing Available Advanced Templates.

Standard templates are provided for each supported print type. You can create your own customized templates in a Template Editor that supports current industry standards for HTML-based editing, including rich text editing and HTML markup source editing. You can preview your template as you make changes, and detailed error messages are shown if the template cannot be saved. If required, you can change the script ID of custom templates. See Customizing Advanced Templates in the Template Editor, Source Code Editing to Customize Advanced Templates, Previewing Advanced PDF/HTML Templates, Error Messages in Advanced Templates, and Changing the Script ID of a Custom Template.

You can also use SuiteScript to produce HTML and PDF printed forms that take advantage of advanced template customization capabilities. See Using Advanced Template Formatting Programmatically.

Each printed form that uses an advanced template automatically includes a company logo, based on the image file defined as the Company Logo (Forms) field at Setup > Company > Company Information. For instructions for defining this image file, see the help topic Configuring Company Information.

You can include advanced templates in bundles. See the help topic Objects Available in Customization Bundles.

You can use SuiteCloud Development Framework (SDF) to manage advanced PDF/HTML templates as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual advanced PDF/HTML template to another of your accounts. Each advanced PDF/HTML template page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

⚠️ Important: To use advanced templates, ensure that the Advanced PDF/HTML Templates feature is enabled. Even when this feature is enabled, no changes are made to basic layouts that are currently in use. However, new enhancements are added exclusively to advanced printing, and Transaction Form PDF Layouts and Transaction Form HTML Layouts will be deprecated in a future release. You are encouraged to use advanced templates and associate them with custom forms.

Using Advanced Template Formatting Programmatically

⚠️ Important: SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should only access the NetSuite UI by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic SuiteScript 2.0 Custom Pages.

SuiteScript supports a template engine object and related methods so you can apply advanced template format capabilities programmatically. For SuiteScript 2.0, see the help topic render.TemplateRenderer.
In addition, the SuiteScript function `nlapiPrintRecord(type, id, mode, properties)` supports the use of advanced templates. If you associate an advanced template with the custom form saved for a transaction and use this API to print the transaction, the advanced template is used to format the printed transaction.

You can use SuiteScript to apply advanced templates to printed records that are not transactions. See Using SuiteScript to Apply Advanced Templates to Non-Transaction Records.

### Enabling the Advanced PDF/HTML Templates Feature

The Advanced PDF/HTML Templates feature must be enabled for access to advanced templates and the template editor, and is enabled by default in most customer accounts.

To ensure that the Advanced PDF/HTML Templates feature is enabled, go to Setup > Company > Enable Features. On the SuiteCloud tab, under SuiteBuilder, ensure that the Advanced PDF/HTML Templates box is checked.

Users can enter transaction print settings on the Transactions subtab of the Set Preferences page. For more information, see the help topic Personal Preferences for Transactions.

### Reviewing Available Advanced Templates

When the Advanced PDF/HTML Templates feature is enabled, an Advanced PDF/HTML Templates option is available in the Customization > Forms menu. You can click this option to view a list of the advanced templates in your account.
Advanced templates are available for all printed transactions that support transaction form PDF and HTML layouts, and for expense report transactions. The Advanced PDF/HTML Templates list page includes standard advanced templates and any customized advanced templates that have been created in the account. If a customized template has been installed from a bundle, the list also shows the ID of the bundle.

Advanced templates also support the following additional print types, when their related preferences are enabled:

- If the Print Return Form with Packing Slip preference is enabled, a return form is appended whenever a packing slip is printed. The return form is part of the packing slip advanced template. You can use custom packing slip advanced templates to modify default formatting for printed return forms.

- If the Print Remittance Form with Invoices & Statements preference is enabled, a remittance form is appended whenever an invoice or statement is printed. The remittance form is part of the invoice advanced template and statement advanced template. You can use custom invoice or statement advanced templates to modify default formatting for printed remittance forms.

By default, the standard advanced template is the preferred advanced template for a print type. When you choose the Advanced printing type for a custom form, the preferred advanced template is selected by default as the template for that form. See Setting Custom Forms to Use Advanced Templates.
Advanced PDF/HTML Template for Multiple Currencies Customers

If the Advanced PDF/HTML Templates feature is enabled in your account, a specialized template is available for printing statements for customers that have multiple associated currencies. You can use this Standard Multi-currency Statement Template, or a customized version of it, to ensure that a customer's multiple currencies are accurately represented in printed statements.

To use this type of template to format a printed statement, on the record for the custom form associated with the statement transaction, select Advanced for Printing Type and select the multi-currency template in the Print Template dropdown list. See Setting Custom Forms to Use Advanced Templates.

For customers who began using NetSuite prior to 2014.2, the Standard Multi-currency Statement Template should be used if both the Multi-Currency Customers and Multiple Currencies features are enabled. Otherwise, you should use the Standard Statement PDF/HTML Template, or a customized version of it, for your printed statements.

For customers who began using NetSuite after 2014.2, if the Multiple Currencies feature is enabled, the Standard Multi-currency Statement Template should be used. For multiple currencies to be used in a customer's transactions, these currencies must be associated with the customer on the customer record. For more information, see the help topics Multiple Currencies and Customers and Multiple Currencies.

For customers that do not use multiple currencies, use the Standard Statement PDF/HTML Template, or a customized version of it, for your printed statements. For more information, see the help topic Printing a Statement.

Setting Custom Forms to Use Advanced Templates

When the Advanced PDF/HTML Templates feature is enabled, you can set custom forms for supported transaction types to use advanced templates. When you set a custom form to use an advanced template, that template defines the print and email formatting and contents for transactions that use that custom form.

Note: If your organization uses the multiple currency feature and advanced printing statements, the template needs to allow for multiple currencies by using list record instead of record root. The Standard Multiple Currency Statement provides an example of how to set up the template.

For an example of a printed form that uses an advanced template, see Printed Sales Order Using Advanced Template.

To define an advanced template for a custom form:

1. Go to Customization > Forms > Transaction Forms.
2. On the Custom Transaction Forms page, click Edit for a custom form, or click Customize for a standard form.
3. On the Edit page for the custom form, review the Printing Type options.
   - Basic to associate basic PDF layouts and HTML layouts with the custom form.
   - Advanced to associate advanced templates with the custom form.
4. Select a print template and an email template from the dropdown lists.
- The **Print Template** and **Email Template** dropdown lists contain the standard advanced template and any custom advanced templates for the transaction type.
- You can select one advanced template for printed transactions and a different template for transactions sent by email. These templates are used for both PDF and HTML formatting.
- The preferred advanced template is selected by default in the dropdown lists.

For example, the previous screenshot indicates that any purchase order transaction that has the Custom Purchase Order form selected for Custom Form will use the advanced template **Custom Purchase PDF/HTML Template** for printing and the standard advanced template for sending email messages. Thus, formatting and contents for printed and email versions of the purchase order shown in the following screenshot would be defined by these templates.

**Important:** At any time after you have set a custom form to use an advanced template, you can switch the form back to using basic layouts. Edit the custom form and choose **Basic** for **Printing Type**, and select from the **PDF Layout** and **HTML Layout** dropdown lists. All previously available basic layouts are still available.

When creating a transaction, users can save the transaction and email it as a PDF attachment by clicking **Save & Email**. The PDF attachment uses the customer's preferred language.

**Printed Sales Order Using Advanced Template**

The following sales order uses the Standard Sales Order PDF/HTML Template:
Customizing Advanced Templates in the Template Editor

View a Related Video

To customize your advanced template, you can use the WYSIWYG template editor provided, or you can enter source code markup. For more information about editing, see WYSIWYG Editing in the Template Editor and Source Code Editing in the Template Editor.

**Note:** Some customization options such as adding a bar code or sublist data to your advanced template must be done by editing the source code. For more information, see Source Code Editing to Customize Advanced Templates.

NetSuite provides a WYSIWYG template editor where you can review the formatting and contents of standard advanced templates, and edit them to create custom advanced templates. You drag and drop to move items around on the template.

The template editor supports current industry standards for HTML-based editing, including rich text editing and HTML markup source editing. For more details, see:
Customizing Advanced Templates in the Template Editor

- Viewing an Advanced Template in the Template Editor
- Using the Template Setup Window
- Saving an Advanced Template

**Note:** The template editor uses FreeMarker-based syntax. For more information, see the FreeMarker documentation.

Viewing an Advanced Template in the Template Editor

To see an advanced template in the template editor, go to Customization > Forms > Advanced PDF/HTML Templates and:

- Click **Customize** for a standard advanced template to review it in the template editor and create a customized version of it, or
- Click **Edit** for a custom advanced template to review it in the template editor and make further changes as desired.
**Note:** The default naming convention is **Custom <Print Type> PDF/HTML Template** for the first custom template created for a print type. For each subsequent template of that type, the default is to add a sequence number to the name. For example, the standard advanced template for purchase orders is named Standard Purchase PDF/HTML Template, the first custom advanced template for purchase orders has a default name of Custom Purchase PDF/HTML Template, and the next custom advanced templates for purchase orders have default names of Custom Purchase PDF/HTML Template 2 and Custom Purchase PDF/HTML Template 3.

**Important:** Records are used to populate some values in the template. To open an advanced template for editing, NetSuite runs all before load scripts to get updates on the records and it triggers all Before Record Load Actions in workflows before it loads the template in the editor. If any before load scripts or workflow actions fail, an error message appears and you cannot edit the template until the script or workflow issues are resolved.

The following table shows the records that are accessed by each template type.

<table>
<thead>
<tr>
<th>Advanced Template Type</th>
<th>Records Accessed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bill of Materials</td>
<td>Work order</td>
</tr>
<tr>
<td>Bill Payment</td>
<td>Vendor payment</td>
</tr>
<tr>
<td>Cash Refund</td>
<td>Cash refund</td>
</tr>
<tr>
<td>Cash Sale</td>
<td>Cash sale</td>
</tr>
<tr>
<td>Check</td>
<td>Various payment types. For the template preview, the Check record is used. For more information, see Standard Check PDF/HTML Template.</td>
</tr>
<tr>
<td>Credit Memo</td>
<td>Credit memo</td>
</tr>
<tr>
<td>Custom Transaction</td>
<td>Custom transaction</td>
</tr>
<tr>
<td>Customer Deposit</td>
<td>Customer deposit</td>
</tr>
<tr>
<td>Expense Report</td>
<td>Expense report</td>
</tr>
<tr>
<td>Invoice</td>
<td>Invoice</td>
</tr>
<tr>
<td>GL Impact</td>
<td>GL Impact</td>
</tr>
<tr>
<td>Item Label</td>
<td>No record is loaded in a way that triggers scripts</td>
</tr>
<tr>
<td>Journal</td>
<td>Journal entry</td>
</tr>
<tr>
<td>Mailing Label</td>
<td>Address (on all entity records)</td>
</tr>
<tr>
<td>Packing Slip</td>
<td>Item fulfillment, and based on the original transaction, one of transfer order or sales order</td>
</tr>
<tr>
<td>Payment</td>
<td>Customer payment</td>
</tr>
<tr>
<td>Picking Ticket</td>
<td>Sales order, transfer order</td>
</tr>
<tr>
<td>Price List</td>
<td>Item search per customer</td>
</tr>
<tr>
<td>Purchase Order</td>
<td>Purchase order</td>
</tr>
<tr>
<td>Quote</td>
<td>Estimate</td>
</tr>
<tr>
<td>Return Authorization</td>
<td>Return authorization</td>
</tr>
</tbody>
</table>
Using the Template Setup Window

In the advanced template editor, you can click the Template Setup button to display a Template Setup window where you can modify:

- Basic template properties, including:
  - Title (name)
  - Description
  - Whether the template is preferred for the transaction type
  - Script ID, which is assigned automatically if you leave this field blank
  - Saved Search Template, which generates a template based on an existing saved search (only available for Saved Search templates)
  - Template inactivation, if required

- Template layout settings, including:
  - Page orientation
  - Page size
  - Margins
Customizing Advanced Templates in the Template Editor

For information about options for saving templates, see Saving an Advanced Template

Training Videos for Advanced PDF/HTML Templates

This page includes links to training videos that show how to set up your advanced PDF/HTML templates. Check back to this page because more videos will be coming.

Using the Advanced Template Editor

To customize an advanced PDF/HTML template for your organization, you can use the design view of the Template Editor. View the video.

Adding Fields and Sublists to Advanced PDF/HTML Templates

Advanced PDF/HTML templates use FreeMarker to include NetSuite data, such as fields on transaction records, in the output.

You can include field IDs and sublists from NetSuite transaction records in advanced PDF/HTML templates using FreeMarker expressions. View the video.

WYSIWYG Editing in the Template Editor

View a Related Video

In WYSIWYG mode, you can click a button on the New Element toolbar to add HTML-based elements such as fields, images, and tables, or printing elements such as text, page numbers, page breaks, horizontal lines, headers, and footers. If you are not sure what element a button represents, you can point to the button to display a tooltip. When you click the Image or Table button, a popup displays where you can set properties for the new element.
For printing elements, formatting options are available on the Styles toolbar. You can use these options to control the styling, alignment, and position of elements.

**Warning:** Advanced PDF/HTML templates do not support absolute positioning of elements. If you require exact positioning, you can use Transaction Layouts instead.

Advanced templates use FreeMarker, a Java library used to generate text outputs based on templates and dynamic data. You use FreeMarker interpolations to include NetSuite data in your template. An interpolation is an expression, such as `$(record.entity)` which is replaced in the output with
the actual value of the expression. For complete information about FreeMarker, see the FreeMarker documentation.

For details about customizing templates in the WYSIWYG view:

- Template Editor Toolbar
- Adding and Removing Fields in Advanced Templates
- Adding and Formatting Text in Advanced Templates
- Including Images in Advanced Templates
- Adding Tables to Advanced Templates

Template Editor Toolbar

The buttons on the New Element tab let you add fields, images, and tables, or printing elements such as text, page numbers, page breaks, horizontal lines, headers, and footers. If you are not sure what element a button represents, you can point to the button to display a tooltip. The following table describes each toolbar button in detail.

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a field to the template. For more information, see Adding and Removing Fields in Advanced Templates.</td>
<td></td>
</tr>
<tr>
<td>Add text to the template. For more information, see Adding and Formatting Text in Advanced Templates.</td>
<td></td>
</tr>
<tr>
<td>Add an image to the template. For more information, see Including Images in Advanced Templates.</td>
<td></td>
</tr>
<tr>
<td>Add a table to the template. For more information, see Adding Tables to Advanced Templates.</td>
<td></td>
</tr>
<tr>
<td>Add the page number to the template.</td>
<td></td>
</tr>
<tr>
<td>Add the total number of pages to the template.</td>
<td></td>
</tr>
<tr>
<td>Add a page break to the template. If you want to remove the page breaks, you must switch to Source Code view.</td>
<td></td>
</tr>
<tr>
<td>Add a horizontal line to the template.</td>
<td></td>
</tr>
<tr>
<td>Toggle button to remove or add the template header.</td>
<td></td>
</tr>
</tbody>
</table>
WYSIWYG Editing in the Template Editor

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Note:" /></td>
<td>If you remove the header from the template and then add it again, the header content is erased.</td>
</tr>
</tbody>
</table>

Toggle button to remove or add the template footer.

| Note: | If you remove the footer from the template and then add it again, the footer content is erased. |

---

Adding and Removing Fields in Advanced Templates

View a Related Video

You can see the fields available to include on an advanced template by reviewing the template in the template editor. You can add and remove fields as desired, and also add currency symbols in front of fields, by following the steps in the following procedures:

- Adding a Field to an Advanced Template
- Adding a Currency Symbol to an Advanced Template Field
- Removing a Field from an Advanced Template

![Note:](image) Sublists displayed in the field selector are dimmed and appear for reference only. You cannot add sublists to a template from the field selector. Instead, you must edit the markup source to add a sublist. In addition, you can reference fields on a sublist only at the first level. Referencing sublists at the second level is not supported, with the exception of addresses. For more information about adding sublists to a template, see Source Code Editing in the Template Editor.

---

Adding a Field to an Advanced Template

You use the Fields selector to add a field to an advanced template.

To add a field to an advanced template:

1. Go to Customization > Forms > Advanced PDF/HTML Templates and click Customize or Edit for the advanced template.
2. Position the cursor in the location of the template editor where you want to add a field.
3. On the New Element toolbar, click the **Fields** button to display a **Fields** selector, which lists all fields available to be added to the template.
The **Fields** selector lists all printable fields and sublists that are available to include on a template. The list of available fields is based on the features that are enabled in your account. For a complete list of fields and sublists, see the help topic **SuiteBuilder Advanced Templates Reference**.

- Subrecords are not supported, with the exception of address records.
- Fields and sublists are listed alphabetically and grouped by record name. The data type and script ID are also shown. Each record can be collapsed or expanded to show the fields available. To add a sublist to a template, you edit the markup source to add a sublist. In addition, with the exception of address records, you can reference fields on a sublist only at the first level. Referencing sublists at the second level is not supported. For more information about adding sublists to a template, see **Source Code Editing in the Template Editor**.
- The selector has a search capability so you can quickly find the field you need. Enter part of the field label in the selector’s **Search** field to narrow the list of displayed fields.

4. If you want the field label to be displayed in the template along with the field value, check **Include Label** in the Fields selector.

5. To add multiple fields, use the mouse to separate the fields.

6. When you have finished adding fields, click the **Close** button.
The field are shown on the template. If you point to the label, the FreeMarker identifier for the field displays:

Important: You can enter FreeMarker syntax to add a field or label to an advanced template rather than using the selector, if you prefer. With the exception of address subrecords (as of 2019.2), the selector does not include subrecord fields. You must use FreeMarker syntax to add non-address subrecord fields. For information about this syntax, see Syntax for Advanced Template Fields.

With the appropriate permission, you can include address subrecord fields in advanced templates, as well as reference fields and labels, by using FreeMarker syntax.

Adding Address Subrecord Fields to Advanced Templates

Advanced PDF/HTML Templates directly support the address subrecord. For the address subrecord, you can:

- Use the field selector to directly select subrecord address fields, like City and Street, when you customize advanced PDF printing templates
- Select any custom fields created on the address
- Use text formatting (font, font size, color) to control the layout and appearance of the address in printed content
- Base template logic on address fields (for example, you can use template logic to determine output based on city, state, or zip code)
- Use available address fields to translate addresses, and then print the translated labels of individual address fields

Address fields available in UI:
Adding a Currency Symbol to an Advanced Template Field

You can use a FreeMarker expression to add a currency symbol on an advanced template.

To add a currency symbol to an advanced template:

1. In the template editor, position the cursor in front of the field that represents a currency amount.
2. Enter `${record.currencysymbol}` in front of the field label shown on the template:

Removing a Field from an Advanced Template

There may be some fields on a template that are not required by your organization.

To remove a field from an advanced template:

1. To remove a field, select it and press Delete.

For example, if you do not want sales orders to include the company name and address, you could delete `${companyInformation.companyname}` and `${companyInformation.addressText}`. Note that this example also results in the deletion of the company logo.

Adding and Formatting Text in Advanced Templates

The New Element toolbar of the template editor includes a button that you can click to insert text on advanced PDF/HTML templates. To add text, first place the cursor in the location on the template where the text should be added, and then click the Text button.
If the Styles toolbar does not appear, select the text that you want to format, and then click **Styles**. The Styles toolbar of the template editor includes rich text editing buttons that you can use for the following text formatting functions:

- Formatting Styles
- Paragraph Formats
- Font Name
- Font Size
- Text Color
- Background Color

The editor also includes buttons for font styles and effects. You can point to each button on the Styles toolbar to display a tooltip.

The template editor includes text editing buttons that you can use for the following text alignment functions on advanced templates:

- Decrease Indent
- Increase Indent
- Align Left
- Center
- Align Right
- Justify

**Note:** Custom column fields of type free-form text or text area are always left-aligned, no matter what alignment setting you specify here. To change the alignment of a text or text area column field, add `td p { align: left; }` to the template stylesheet.

### Adding Tables to Advanced Templates

The New Element toolbar of the template editor includes a button that you can click to insert tables on advanced PDF/HTML templates.

**Note:** Nested tables, that is, tables within tables, are not recommended for performance reasons.

To **insert a table on an advanced template:**

1. Open the advanced template in the template editor.
2. Place the cursor in the spot where you want to insert the table.
3. Click the **Table** button:

![Table button](image)

4. Complete the Table Properties dialog and click **OK**.

![Table Properties dialog](image)

The table is added to the template in the template editor.

You can resize the rows, columns, and cells of a table by positioning the cursor over the cell border and dragging it to the desired size.

**Note:** You can also use HTML markup source editing to add a table to an advanced template, but it would be more time-consuming and require knowledge of HTML.

**Important:** If you have text overlapping a footer or missing from a printout, ensure that any long content is enclosed in an HTML tag that will split across pages.

The Report Generator has specific rules for where page breaks can occur. A `<table>` tag nested inside a `<td>` tag is cut off at the bottom if it spreads across multiple pages. Only the following tags split correctly if they are spread across multiple pages:

```html
<table>
  <ul>
    <p>
      <pre>
        <ol>
          <h1>
            <h2>
              <h3>
                <h4>
                  <blockquote>
                  </blockquote>
                </h4>
              </h3>
            </h2>
          </ol>
        </pre>
      </p>
    </ul>
  </table>
```
Including Images in Advanced Templates

Advanced templates support the inclusion of images in printed and emailed forms. The image can be a file that is located on the internet, or it can be an image file that you have uploaded to the File Cabinet. The preferred method of adding an image is to use a File Cabinet ID.

The New Element toolbar of the template editor includes a button that you can click to insert an image.

To insert an image on an advanced template:

1. Open the advanced template in the Template Editor.
2. Place the cursor in the location where you want to insert the image.
3. To insert an image, click the Image button.
4. In the URL field, enter the full path to the image file in the File Cabinet or on the internet. Use & for ampersand symbols.
   Include fields that dynamically return a URL, for example ${companyInformation.appDomain}. For more information, see Syntax for Advanced Template Fields.
5. To make an image a File Cabinet item, click the Advanced tab. Then check the File Cabinet Item box. When this box is checked, the system uses the <@filecabinet> directive to embed the image into the PDF.
6. Enter any other required information in the Image Properties window. Then click **OK**.

The image is added to the template in the template editor.

**Note:** You do not need to add a company logo to a template. Each printed form that uses an advanced template automatically includes a company logo, based on the image file defined as the Company Logo (Forms) field at Setup > Company > Company Information. For instructions for defining this image file, see the help topic Configuring Company Information.

**Warning:** If you reference images online, ensure that the URLs are correct. If any images referenced in a template cannot be found, an error message is displayed and you cannot save the template until the errors are resolved.

### Source Code Editing in the Template Editor

You can use the Source Code toggle to manually edit markup source for a template. You can make template edits directly in this markup source if you have sufficient knowledge of HTML.
**Warning:** Do not modify markup source directly unless you have sufficient CSS and HTML knowledge. NetSuite does not provide support or training in CSS or HTML.

Be aware that the template editor may not function properly if you switch back to WYSIWYG mode after you have made edits in markup source mode. Some template content may not be represented correctly, may not be accessible for editing, or may not be displayed at all.

If these issues occur, you can preserve template content by not saving the template in WYSIWYG mode and switching back to markup source mode.

The template editor supports syntax highlighting of markup source, for improved readability.

You can edit the HTML markup source to customize your advanced PDF/HTML template as follows:

- Source Code Editing to Customize Advanced Templates
- Syntax for Advanced Template Fields
- Setting a Template to Use a Font Unavailable in NetSuite
- Languages for Printed Forms that Use Advanced Templates
- Adding Translated Content in Advanced Printouts
- Adding Striping to Line Items in Advanced Templates
- Adding Page Breaks to Tables
- Printing Subsidiary Logo on Advanced Templates
Source Code Editing in the Template Editor

- Adding Apply Sublist to Check Templates
- Adding Bar Codes in Advanced Templates

Use FreeMarker directives to customize date formatting in the template. For example, using a date format containing MONTH does not return the name of the month in uppercase. To display the month in uppercase, use the FreeMarker upper_case directive, for example: \( \text{\$record.duedate?upper_case} \).

In advanced template, some of the markup source syntax relies on BFO (Big Faceless Organization), a set of third party libraries used by NetSuite for generating PDF documents. BFO documentation is available at http://bfo.com/products/report/docs/userguide.pdf

Note: If you are having issues with your advanced template, do not contact BFO directly. Always contact NetSuite Customer Support.

Empty BFO tags that are closed are subject to HTML formatting, for example, \(<\text{totalpages}> \) processes as \(<\text{totalpages}> \) \(<\text{totalpages}>\)\).

Some HTML tags may be displayed as literals in printed text. For example, if a field contains the character for a line break, then the line break literal value of "<br />" is displayed.

Important: The following are not supported in advanced templates because of BFO processing:

- Do not use the \(<\text{tbody}>\) tag, otherwise you may not be able to save the template.
- Do not use the \(<\text{cellspacing}>\) attribute. Instead, use the \(<\text{cellmargin}>\) attribute.
- Do not use the \(<\text{text-align}>\) attribute. Instead, use the \(<\text{align}>\) attribute.
- Do not specify a body width in percentage or you will receive a no size specified error. An absolute value must be set, for example, \(<\text{body width}=595\>\).

Source Code Editing to Customize Advanced Templates

View a Related Video

An advanced template that is used to print a PDF file is an XML document that uses syntax very similar to HTML with FreeMarker and BFO (Big Faceless Organization) report generator elements included. For example:

```xml
1  <?xml version="1.0"?>
2  <!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
3  <pdf>
4    <head>
5      <head>
6      <body>
7        Hello, World!
8      </body>
9    </head>
10  </pdf>
```

Line 1 of the template must declare the XML version, and Line 2 must specify the DOCTYPE. Line 3 includes the BFO \(<\text{pdf}>\) wrapping element, not the \(<\text{html}>\) declaration you would see at the top of an HTML document.
Inside the `<pdf>` element, the head and body elements contain standard HTML, and you can embed CSS2 elements. HTML5 declarations are not allowed. The example displays the text "Hello, World!" The full capabilities of the BFO printing engine are described in the BFO User Guide.

In XML, elements must always be closed. As shown in Line 8, `<pdf>` must always be matched by `</pdf>`, `<b>` by `</b>` and so on. For elements that have no content, like `<br>`, the closing element is included in the element: `<br />`. Any attributes must have quotes around them, for example, `<table width="100%">`.

If a PDF template is used to print in HTML, the `<pdf>` tags are automatically converted to `<html>` when the document is printed.

Important: NetSuite uses BFO, FreeMarker, and CKEditor. For version details, see the help topic Third-Party Notices and Licenses. When using advanced templates, you must follow the syntax and usage guidelines included in the documentation for BFO, FreeMarker, and CKEditor. For more information, see the BFO website, FreeMarker website, and the CKEditor website.

For issues with NetSuite advanced templates, ensure that you contact NetSuite Support.

BFO Elements

BFO (Big Faceless Organization) is a Java application that converts documents written in XML to PDF. BFO is used in NetSuite. For version details, see the help topic Third-Party Notices and Licenses. Some commonly-used BFO elements are described in the following sections.

Page Numbers

The most-commonly used BFO elements in PDF templates are page number and total pages. The `<pagenumber />` and `<totalpages />` elements insert the current page number and total number of pages. Because these values are known at the last part of rendering the page, they cannot be used in FreeMarker declarations as values.

Headers, Footers and Background Macros

You can use BFO functionality to define macros to repeat pieces of HTML code on every page. Each macro is defined in the head part of the template inside the `<macrolist>` element. To apply the macro, you reference it in the `<body>` element definition. It is not possible to create multiple headers for a template. Also, header and footer macros must have a height declared or they will not be applied. See the following example.

```xml
<?xml version="1.0"?>
<!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
<pdf>
  <head>
    ...
    <macrolist> <!-- Definition of macros -->
      <macro id="nlHeader"> <!-- Regular macros -->
        ... Header Content ...
      </macro>
      <macro id="nlFooter">
        ... Footer Content ...
      </macro>
      <macro id="nlWatermark">
        ... Footer Content ...
      </macro>
    </macrolist>
  </head>
</pdf>
```
Bar Codes

When using an advanced printing template, you can add any type of bar code or QR code listed in the Barcodes section of the BFO User Guide. In the template, create a field with a value that can be passed in as the value of that bar code type.

The bar code syntax is shown in the following example.

```html
<barcode codetype="qrcode" showtext="false" height="150" width="150" value="http://www.example.com/" />
```

Using FreeMarker to Include NetSuite Data

FreeMarker is a Java library used to generate text outputs based on templates and dynamic data. FreeMarker is used in NetSuite. For version details, see the help topic Third-Party Notices and Licenses. You use FreeMarker interpolations to include NetSuite data in your template. An interpolation is an expression, such as `${record.entity}` that FreeMarker replaces in the output with the actual value of the expression. A few common uses for FreeMarker declarations are provided in the following sections. For complete information about FreeMarker, see the FreeMarker documentation.

Referencing Fields

FreeMarker is commonly used to reference fields on transaction records. The syntax to reference a field is `${record.fieldId}`. In the output, this interpolation is replaced with a text representation of the field's value.

If you want to print the field's label, the syntax is `${record.fieldId@label}`. For example, `${record.entity@label}: ${record.entity}` might produce something on the Sales Order record that looks like Customer: Fabre Art Gallery.
Sublists and Other Lists

There are some components in NetSuite that can be referenced as a list of objects. The most common example is an item sublist on a transaction, which is represented as a list of lines. You can access values of these lists directly using the index number \( \${record.list[index]} \). For example, adding \( \${record.item[1].itemName} \) to a sales order returns Green T-Shirt.

The more common way of accessing a list is by using the FreeMarker \#list declaration, similar to the following.

```freemarker
<#list record.item as item>
    ${item_index} ${item.itemName@label} ${item.itemName} --- ${item.amount}
</#list>
```

If the item sublist on the sales order had three lines, the output would look like the following.

0 Name: Blue T-Shirt --- 10.00$
1 Name: Green T-Shirt --- 12.25$
2 Name: Yellow T-Shirt --- 11.00$

To view the sublists that are supported for a record, see the SuiteBuilder Advanced Templates Reference.

For more information about using FreeMarker expressions in advanced PDF/HTML templates, see Using FreeMarker to Include NetSuite Data. The training video that is available from the page describes how to include field IDs and sublists from NetSuite transaction records in advanced templates.

Additional Information to Include on Templates

There are additional models that can be accessed on each template. These include `companyInformation` (Company configuration information), `preferences` (user preference settings usually stored as boolean values) and `user` (user information).

Some templates may have additional models attached, enabling you to access additional data. For example, you can access the customer record on each statement, such as the customer email, \( \${customer.email} \).

You can also use \( \${record@title} \) to print out the record title.

Differences from HTML 4 Specification

The implementations of BFO and FreeMarker include some important differences from the HTML 4 specification. These differences might affect the formatting of your advanced templates. For more information, see the Element and Attribute Reference for BFO at http://bfo.com/products/reports/docs/tags/.

Certain characters behave differently when they are used with BFO. For example, the non-breaking space (`&nbsp;`) character is not rendered when using advanced templates. In general, use a line break tag (`<br />`) instead of the `&nbsp;` character.

If you are using the `&nbsp;` character to create spacing in HTML table layouts, use the `width` and `height` attributes of the `<td>` element instead to specify how BFO formats the `<table>` element. For example:

```html
<table table-layout="fixed" width="200">
```
Syntax for Advanced Template Fields

You can include the following field types for a record in an advanced template:

- body fields
- sublist fields
- fields from related records
- record search results fields

The template editor creates an XML document that uses syntax very similar to HTML. Note that although this syntax may appear similar to SuiteScript, the template editor does not support the execution of SuiteScript APIs.

The fastest way to add a field to an advanced template is to choose the field in the template editor’s **Fields** selector. (See *Adding and Removing Fields in Advanced Templates.*) If preferred, you can use FreeMarker syntax to add a field manually instead of using the **Fields** selector.

Subrecords are not supported in advanced templates, except for the list of inventory details values.

See the following:

- Entering a Field Manually to an Advanced Template
- Syntax for Body Fields
- Syntax for Sublist Fields
- Syntax for Address Subrecords
- Syntax for Fields from Joined Records and Searches
- FreeMarker Formatting Method
- Updating a Statement Template to Support Multiple Currencies
- File Cabinet FreeMarker Directive

**Entering a Field Manually to an Advanced Template**

Some fields can be added to advanced templates but are not available in the Fields selector, for example, `$\{record.entity.email\}` refers to the Customer record through the entity field so that all body fields in the customer record can be accessed.

The following procedure describes how to add fields to the template manually.

**To manually add a field to an advanced template:**

1. Obtain the field ID for the field you want to add.
To make field IDs available, go to Home > Set Preferences and ensure that the Show Internal IDs box is checked on the General subtab, Defaults area.

Find the field in the NetSuite user interface and click the field label to display the field level help text.

The field ID is displayed in the popup.

Note: Field ID information is also available in the SuiteBuilder Advanced Templates Reference.

2. Go to Customization > Forms > Advanced PDF/HTML Templates and click Customize or Edit for the advanced template.

3. Place the cursor in the location on the template where you want to add the field.

4. Add a text field, and enter the appropriate syntax for the field that you want to add.

The template editor supports a simplified syntax to get values for the fields to be printed. To see if a field is available in an advanced template, see the help topic SuiteBuilder Advanced Templates Reference.

For information about entering the syntax, read Syntax for Body Fields, Syntax for Sublist Fields, and Syntax for Fields from Joined Records and Searches.

Syntax for Body Fields

For most template body fields, the syntax is ${object.fieldId}, for example, ${record.entity}.

The following objects are currently supported:

- **companyInformation** - company information for the current NetSuite account
  
  For example, ${companyInformation.companyname} is the name associated with the NetSuite account.
  
  Enter ${subsidiary.mainaddress_text} to include the subsidiary address on a template.

- **record** - current transaction record
  
  For example, ${record.trandate} is the date of the transaction.

- **records** - reference multiple data sources, which you typically need to do in label and statements templates, for example, ${statement.amountDue}.

- **customer** - only available on Statement records, this includes customer-related information on the template. For example, you could include ${customer.subsidiary}, ${customer.subsidiary.legalname}, or ${customer.entityId}.

- **salesorder** - available on packing slips

- **preferences** - include settings entered on the Setup > Company > Printing & Fax page.

- **user** - include settings entered on the user's Set Preferences page. These fields are typically used for emails.

You can include fields that dynamically return a URL link to NetSuite pages, or to parts of a URL, such as the account ID:
Source Code Editing in the Template Editor

Note: In the following examples, replace the variable <accountID> with your account ID.

- ${companyInformation.companyId} — Your NetSuite account ID.
- ${companyInformation.appDomain} — Application domain, for example, <accountID>.app.netsuite.com.
- ${companyInformation.customerCenterDomain} — Customer center domain, for example, <accountID>.app.netsuite.com.
- ${companyInformation.customerLoginUrl} — Customer login URL, for example, https://<accountID>.app.netsuite.com/app/login/secure/privatelogin.nl?<accountID>
- ${companyInformation.extformsDomain} — External forms domain, for example, <accountID>.extforms.netsuite.com.

Note: To find your account ID, go to Setup > Company > Company Information. The Account ID field is at the bottom of the right column. To view the correct format for all your account-specific domains, click the Company URLs tab.

To view Account ID and Company URLs, you must be an account administrator or a user with the Setup Company permission.

You can include fields from records related to the current transaction record if they are of type select (not sublists).

- Syntax is ${record.related_record.fieldId}
  For example, to include customer field values from related item records in a sales order template, use the syntax: ${record.entity.email}. You can also refer to custom fields, for example, include a transaction column field on a purchase order template by entering ${record.expense.custcol_cc_oversize}.

The syntax for body fields can include an additional component, such as the following:

- ${object.fieldId@label} - indicates the label of a field should be printed.
  For example, ${record.entity@label}: ${record.entity} indicates the label for the entity should be printed with the entity, that is, Customer: City Art Gallery. The label is the text that displays next to the field on the transaction form.

- ${record@title} - indicates that the name of the transaction record type should be printed, for example, Sales Order for sales order.
  This can be used for record objects only; for other objects it will return a blank string.

Fields that include a link, such as images or URLs, are enclosed in an HTML tag, for example, <img> or <a>. You have two options for link fields:

- Use the field to include the HTML markup source. For example, ${record.hyperlink} returns the following:

- Add @url to the field to return the link without the HTML. For example, ${record.hyperlink@url} returns the following:
  http://www.netsuite.com

Syntax for Sublist Fields

Sublists are NetSuite components that are referenced as lists of objects. The most common example of a sublist is an item sublist on a transaction, which displays a list of lines. Sublists are not available in the Fields selector.
The most common way to access a sublist is by using the FreeMarker `#list` declaration. The following example shows an item sublist on the sales order.

```
<#list record.item as item>
    ${item_index} ${item.itemName@label} ${item.itemName} --- ${item.amount}
</#list>
```

This declaration prints out a block of HTML code to display the whole list.

<table>
<thead>
<tr>
<th>Name</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blue T-Shirt</td>
<td>10.00$</td>
</tr>
<tr>
<td>Green T-Shirt</td>
<td>12.25$</td>
</tr>
<tr>
<td>Yellow T-Shirt</td>
<td>11.00$</td>
</tr>
</tbody>
</table>

To reference one item in the list, you can use `${record.item[index].itemName}` to obtain only that line in the sublist. Using the example above, `${record.item[1].itemName}` returns the line 1 item *Green T-Shirt*.

The syntax for sublist (line item) fields is similar to body field syntax. Sublist fields are denoted with syntax like:

- `${item.item@label}`
- `${item.id}`
- `${item.item}`
- `${item.quantity}`

You can reference fields on a sublist only at the first level. Referencing sublists at the second level is not supported.

For example, only the following is supported.

```
<#list record.item as item>
    ${item.field_first_level}
</#list>
```

If you use `item.inventorydetail`, you will get a listing of all bin/serial numbers.

To sort line items in advanced templates, add the `sort_by()` function to the #list declaration, for example:

```
<#list record.item?sort_by("quantity") as item>
```

**Syntax for Address Subrecords**

For address subrecords, use the following syntax in the source code:

- Record name
- Address subrecord name
- Address field name

For example:

- `${subsidiary.mainaddress.addresee}`
- `${subsidiary.mainaddress.addrl}`
Syntax for Fields from Joined Records and Searches

You can modify advanced templates in the template editor to include fields from records directly joined to the current transaction.

- Syntax is `{record.related_record.fieldId}`
- To include sales description field values from related item records in a sales order template, use the syntax: `{record.item.salesdescription}`

You can also include search results fields from searches joined directly to the current transaction.

- You can use the index number to indicate the search result row to be referenced to obtain a field value. For example, `[0]` would indicate that the field value from the first row returned in search results should be used.
- Syntax is `{results[search_result_line].related_record.fieldId}`.
  - For example, to include the first customer phone number from a sales order search, type: `{results[0].customer.phone}`
- To include a list of search results, use a list directive as shown in the following example.

```html
<#list results as salesOrder>
  <p>${salesOrder.customer.phone}</p>
</#list>
```

FreeMarker Formatting Method

NetSuite formatting is lost when you customize NetSuite standard fields. If you print hard-coded values which came from a custom data source or were calculated from standard fields, the format will not be consistent with the format of standard record fields. When editing the template source code, the `nsformat_*()` formatting method makes it possible to format FreeMarker custom variables in the same way NetSuite formats standard fields. It also decreases the need for hardcoding. The `nsformat_*()` method formats the specific string, number, date, or amount results in the same format used on a standard NetSuite field. With this formatting method, it is possible to apply localization to custom variables, ensuring consistent output.

The following table shows some examples of standard fields in use:

<table>
<thead>
<tr>
<th>Example</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>{record.custbody_show_int}</code></td>
<td>1,234</td>
</tr>
<tr>
<td><code>{record.custbody_show_int+1}</code></td>
<td>1235</td>
</tr>
<tr>
<td><code>{nsformat_number(record.custbody_show_int+1)}</code></td>
<td>1,235</td>
</tr>
</tbody>
</table>
The following table shows some examples of the \$\text{nsformat}()\text{ method in use:}

<table>
<thead>
<tr>
<th>Method</th>
<th>Example</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{nsformat} \text{date}(date</td>
<td>string)</td>
<td>$\text{nsformat} \text{date}(&quot;2018-12-04T10:40:00.000&quot;)</td>
</tr>
<tr>
<td>$\text{nsformat} \text{datetime}(date</td>
<td>string)</td>
<td>$\text{nsformat} \text{datetime}(&quot;2018-12-04T10:40:00.000Z&quot;)</td>
</tr>
<tr>
<td>$\text{nsformat} \text{datetime}(date</td>
<td>string)</td>
<td>$\text{nsformat} \text{datetime}(&quot;2018-12-04T10:40:00.000&quot;)</td>
</tr>
<tr>
<td>$\text{nsformat} \text{datetime}(itemDateTime)</td>
<td>$\text{nsformat} \text{datetime}(itemDateTime)</td>
<td>Dec 4, 2018 10:40:00 AM</td>
</tr>
<tr>
<td>$\text{nsformat} \text{time}(date</td>
<td>string)</td>
<td>$\text{nsformat} \text{time}(&quot;2018-12-04T10:40:00.000&quot;)</td>
</tr>
<tr>
<td>$\text{nsformat} \text{currency}(number</td>
<td>string) or with optional second string $\text{nsformat} \text{currency}(number</td>
<td>string, string)</td>
</tr>
<tr>
<td>$\text{nsformat} \text{boolean}(boolean</td>
<td>string</td>
<td>number) or with optional second string $\text{nsformat} \text{rate}(number</td>
</tr>
<tr>
<td>$\text{nsformat} \text{number}(number</td>
<td>string)</td>
<td>$\text{nsformat} \text{number}(&quot;999999999&quot;)</td>
</tr>
<tr>
<td>$\text{nsformat} \text{boolean}(boolean</td>
<td>string</td>
<td>number)</td>
</tr>
<tr>
<td>$\text{nsformat} \text{password}(string)</td>
<td>$\text{nsformat} \text{password}(&quot;p4sw0rd&quot;)</td>
<td>*****</td>
</tr>
<tr>
<td>$\text{nsformat} \text{percent}(number</td>
<td>string)</td>
<td>$\text{nsformat} \text{percent}(&quot;100&quot;)</td>
</tr>
<tr>
<td>$\text{nsformat} \text{email}(string)</td>
<td>$\text{nsformat} \text{email}(&quot;<a href="mailto:email@example.com">email@example.com</a>&quot;)</td>
<td>clickable email address</td>
</tr>
<tr>
<td>$\text{nsformat} \text{url}(string)</td>
<td>$\text{nsformat} \text{url}(&quot;<a href="http://example.com">http://example.com</a>&quot;)</td>
<td>clickable link</td>
</tr>
</tbody>
</table>

Formatters accept numbers in double precision floating point representation.
If the input is an empty string or null, then the output is also represented as an empty string.

**Updating a Statement Template to Support Multiple Currencies**

If you have an advanced statement template that must be updated to support multiple currencies, edit your template as follows.

Wrap the existing template in the following tags:

```html
<pdfset>
  <#list statements as record>
    ...original template...
  </#list>
</pdfset>
```

Change any \text{record}.\text{items} references to \text{record}.\text{lines}.
File Cabinet FreeMarker Directive

If you edit templates in the source code, you can use the filecabinet FreeMarker directive to reference files directly from the File Cabinet. By using this directive, you can embed text, images, and fonts that are stored in the File Cabinet into the template.

For example:

```freemarker
<@filecabinet nstype="image" src="https://<accountID>.app.netsuite.com/core/media/media.nl?id=21&c=4130331
&h=fb3b8b4ac4f67b2c369b"/>
```

When you add an image to a template and check the File Cabinet Item box, the `<@filecabinet/>` directive is used in the source code.

For more information about the File Cabinet, see the help topic Working with the File Cabinet.

For more information about including images in advanced templates, see Including Images in Advanced Templates.

Setting a Template to Use a Font Unavailable in NetSuite

If you want to print using a font or language that is not available in NetSuite, you can edit the template to do this.

1. Load the .TTF font file into NetSuite.
2. In the template, declare the font as opentype, not truetype.
3. Refer to the font in the File Cabinet.

```xml
<?xml version="1.0"?>
<!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
<pdf>
<head>
  <link name="thai-font" type="font" subtype="opentype" src="https://<accountID>.app.netsuite.com/core/media/media.nl?id=5967216&amp;c=3809789&amp;h=8609ede0ae4cbfad71d&amp;_xt=.ttf" src-bold="https://<accountID>.app.netsuite.com/core/media/media.nl?id=5967216&amp;c=3809789&amp;h=8609ede0ae4cbfad71d&amp;_xt=.ttf"
bytes="2" />
  <style type="text/css">
    * {
      font-family: thai-font, sans-serif;
      font-size: 9pt;
      table-layout: fixed;
    }
    th {
      font-weight: bold;
      font-size: 8pt;
      vertical-align: middle;
      padding: 5px 6px 3px;
      background-color: #e3e3e3;
      color: #333333;
    }
    td {
      padding: 4px 6px;
    }
  </style>
</head>
```
Languages for Printed Forms that Use Advanced Templates

Custom forms that use advanced templates are printed in the locale of the current user, set in the Language field at Home > Set Preferences. Based on the locale, the font family is set.

You also can print these forms in the locale of the customer associated with a transaction, for transaction types that support this option at Transactions > Management > Print Checks and Forms.

ISO standards are followed for the country and culture code for each language (ISO 639 and ISO 3166 respectively).

When printing a template, only one locale can be selected. However, it is possible to combine languages and alphabets within one template if all of the required fonts are linked to the template and the lang attribute is set correctly.

Language Support Available in Advanced Templates

Advanced PDF/HTML Templates use Google Noto fonts. You can create PDF printouts in more than 500 languages, including all languages supported for the NetSuite application. The list of supported languages includes the following.

<table>
<thead>
<tr>
<th>Font Name</th>
<th>Language Code</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>NotoSans_Bold</td>
<td></td>
<td>580+ languages supported</td>
</tr>
<tr>
<td>NotoSans_BoldItalic</td>
<td></td>
<td>For more information, see</td>
</tr>
<tr>
<td>NotoSans_Italic</td>
<td></td>
<td><a href="https://www.google.com/">https://www.google.com/</a></td>
</tr>
<tr>
<td>NotoSans-Regular</td>
<td></td>
<td>get/noto/</td>
</tr>
<tr>
<td>NotoSerif_Bold</td>
<td></td>
<td>580+ languages supported</td>
</tr>
<tr>
<td>NotoSerif_BoldItalic</td>
<td></td>
<td>For more information, see</td>
</tr>
<tr>
<td>NotoSerif_Italic</td>
<td></td>
<td><a href="https://www.google.com/">https://www.google.com/</a></td>
</tr>
<tr>
<td>NotoSerif-Regular</td>
<td></td>
<td>get/noto/</td>
</tr>
<tr>
<td>NotoSansCJKsc_Bold</td>
<td>zh_CN</td>
<td>Chinese (Simplified)</td>
</tr>
<tr>
<td>NotoSansCJKsc-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansCJKtc_Bold</td>
<td>zh_TW</td>
<td>Chinese (Traditional)</td>
</tr>
<tr>
<td>NotoSansCJKtc-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansCJKjp_Bold</td>
<td>ja_JP</td>
<td>Japanese</td>
</tr>
<tr>
<td>NotoSansCJKjp-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansCJKkr_Bold</td>
<td>ko_KR</td>
<td>Korean</td>
</tr>
<tr>
<td>NotoSansCJKkr-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansThai_Bold</td>
<td>th_TH</td>
<td>Thai</td>
</tr>
<tr>
<td>NotoSansThai-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Font Name</td>
<td>Language Code</td>
<td>Language</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>NotoSansThai-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansArabic-Bold</td>
<td></td>
<td>Arabic - 60+ languages supported</td>
</tr>
<tr>
<td>NotoSansArabic-Regular</td>
<td>me_AR</td>
<td>Arabic</td>
</tr>
<tr>
<td>NotoSansHebrew-Bold</td>
<td>he_IL</td>
<td>Hebrew</td>
</tr>
<tr>
<td>NotoSansHebrew-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansArmenian-Bold</td>
<td>hy_AM</td>
<td>Armenian</td>
</tr>
<tr>
<td>NotoSansArmenian-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansBengali-Bold</td>
<td></td>
<td>Bengali</td>
</tr>
<tr>
<td>NotoSansBengali-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansGujarati-Bold</td>
<td>gu_IN</td>
<td>Gujarati</td>
</tr>
<tr>
<td>NotoSansGujarati-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansDevanagari-Bold</td>
<td>hi_IN</td>
<td>Hindi</td>
</tr>
<tr>
<td>NotoSansDevanagari-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansDevanagari-Bold</td>
<td></td>
<td>Marathi</td>
</tr>
<tr>
<td>NotoSansDevanagari-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansKannada-Bold</td>
<td>kn_IN</td>
<td>Kannada</td>
</tr>
<tr>
<td>NotoSansKannada-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansTamil-Bold</td>
<td></td>
<td>Tamil</td>
</tr>
<tr>
<td>NotoSansTamil-Regular</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NotoSansTelugu-Bold</td>
<td></td>
<td>Telugu</td>
</tr>
<tr>
<td>NotoSansTelugu-Regular</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For a complete list of supported languages, see the [Google website](https://www.google.com/get/noto/).

Templates that use the Noto fonts have the following code in the `<head>` section.

```html

#<if .locale == "zh_CN">
  <link name="NotoSansCJKsc" type="font" subtype="opentype" src="${nsfont.NotoSansCJKsc_Regular}" src-bold="${nsfont.NotoSansCJKsc_Bold}" bytes="2" />
</if>

#<if .locale == "zh_TW">
  <link name="NotoSansCJKtc" type="font" subtype="opentype" src="${nsfont.NotoSansCJKtc_Regular}" src-bold="${nsfont.NotoSansCJKtc_Bold}" bytes="2" />
</if>

#<if .locale == "ja_JP">
  <link name="NotoSansCJKjp" type="font" subtype="opentype" src="${nsfont.NotoSansCJKjp_Regular}" src-bold="${nsfont.NotoSansCJKjp_Bold}" bytes="2" />
</if>

#<if .locale == "ko_KR">
  <link name="NotoSansCJKkr" type="font" subtype="opentype" src="${nsfont.NotoSansCJKkr_Regular}" src-bold="${nsfont.NotoSansCJKkr_Bold}" bytes="2" />
</if>

#<if .locale == "th_TH">
  <link name="NotoSansThai" type="font" subtype="opentype" src="${nsfont.NotoSansThai_Regular}" src-bold="${nsfont.NotoSansThai_Bold}" bytes="2" />
</if>
```
The following styles definitions are also included.

```css
* {
    font-family: NotoSans, NotoSansCJKsc, sans-serif;
}
```

To use the other fonts available in the NotoSans font set, use a line similar to the following as the first line in the `<head>` section:

```html
<link name="NotoSansArabic" type="font" subtype="opentype" src="${nsfont.NotoSansArabic-Regular}" src-bold="${nsfont.NotoSansArabic-Bold}" bytes="2" subset="false" />
```

To use this font as the default font for all elements in the XML, edit the `<style>` definition similar to the following:

```css
* { font-family: NotoSansArabic, sans-serif; }
```

**Note:** If you notice issues with font spacing in your custom templates, add the following to the `<style>` definition for the template.

```css
td p {
    align: left;
}
```

**Note:** If you notice missing or corrupted glyphs (usually for ligatures), particularly when using the Arabic locale, add attribute `subset="false"` to the `<link>` element. This embeds the whole font in the PDF file, and results in a slightly larger PDF file size.

### Adding Fonts for Languages that Use Symbols

For languages that use symbols, such as Arabic, values in columns (POS, Quality, and so on) of Advanced PDF/HTML email attachments may appear empty. NetSuite views the symbols as missing fonts, and therefore leaves the fields blank. To have these symbols or fonts appear, users affected by this must customize standard templates based on language or currency.

If a document is printed as an attachment, then it is printed using the customer locale and the number format is determined by currency (for example EGP for Egypt). In the Arabic example, Eastern Arabic
symbols are used for digits. Because the email attachment is printed based on locale, you must add the missing font into your template.

- To add font support to the template, you can link the font by including this line in the `<head>` section:
  
  ```html
  <link name="NotoSansArabic" type="font" subtype="opentype" src="${nsfont.NotoSansArabic_Regular}" src-bold="${nsfont.NotoSansArabic_Bold}" bytes="2" />
  ```

- Use the font as the last option in style.
  
  For example, change this line:

  ```css
  font-family: NotoSans, sans-serif;
  ```
  
  Change the line to the following:

  ```css
  font-family: NotoSans, sans-serif, NotoSansArabic;
  ```

### Combining Multiple Languages and Alphabets in One Document

When printing a template, only one locale is selected, but you can use other locales in the document as long as the required fonts are linked.

Steps to achieve such configuration:

1. Load all required fonts via `<link>` element.
2. Create CSS selector to pair language and font.
   - Set any CSS properties specific for the given language.
   - Typically, line-height will differ for some languages like Arabic
3. Set lang attribute on elements where the language is different from the document language.
   - Nested elements inherit value of the lang attribute.
Adding Translated Content in Advanced Printouts

With the Translation Collection API, you can use FreeMarker code to add translated content in advanced printouts. For example, you could access the Translation Collection to display a localized disclaimer or a greeting. You can use one printing template that pulls the appropriate translation for all languages from the collection to produce translations in multiple languages. Currently, you can do this in source code mode only.

When working with Translation Collections data, you have the following options:

- `nstranslation.load` - loads strings for specific keys of specific collections for specific locales. The locales parameter is optional. If not defined, the current locale from FreeMarker is used.

  In the example that follows, strings with the keys `GREETINGS` and `INTERVAL_1_TO_2` load from the `custcollection_testcol` collection for the `cs_CZ` and `en_US` locales. The translation for `cs_CZ` contains "Ahoj" and translation for `en_US` contains "Hello"

```freemarker
<#assign load = nstranslation.load({
   "collections":{
      "alias": "myAlias",
      "collection": "custcollection_testcol"
   },
   "keys": {"GREETINGS", "INTERVAL_1_TO_2"},
   "locales": [
      nstranslation.Locale.cs_CZ
      nstranslation.Locale.en_US
   ]
})
```

**Use Default Locale**

The first locale specified in `load` is the default. In the previous example, that would be `cs_CZ`. So, if you use the following, the resulting greeting will be Ahoj: `${handle.myAlias.GREETINGS}`

**Get Translation for a Different Locale**

To get translation from `handle` for a different locale than what's in the locale's list, use `nstranslation.selectLocale()`. You might use this if you want to translate into the specified language once only. For example, if you use the following, the resulting greeting will be Hello: `${translation.selectLocale({"handle": handle, "locale": nstranslation.Locale.en_US}).myAlias.GREETINGS}`

**Translate Multiple Strings to Same Language**

When you want to translate multiple strings into the same language, you should store localized `handle` into a variable, like this: `<#assign englishHandle = nstranslation.selectLocale({"handle": handle, "locale": nstranslation.Locale.en_US})>`. For example, you could then use it as follows to get a greeting of Hello: `${englishHandle.myAlias.GREETINGS}`

For more information, see the help topic `translation.load(options)`.

- `nstranslation.get` - use the collection to translate a single string to one selected locale once only. The locale parameter is optional. If not defined, the current locale from FreeMarker is used.

  For example, to return a greeting of Hello, you can access the string in the collection using:

  ```freemarker
  ${nstranslation.get({"collection":"custcollection_testcol", "key": "GREETINGS", "locale": nstranslation.Locale.en_US}}
  ```
Example of String with Language Translation

In the following example, we have a string that will get the expected count of orders within a specified range. From the entry in the Source String field, you can see there are two placeholders ( {1} and {2} ) for the range of numbers to include in the output. The Key is entered in the ID field, and the string translation in the Czech language is shown in the Translation field.

In this example, the default locale is set to output in English (US). When printing the document, you can specify the output be printed in Czech, rather than in the default language.

Source String: Expected count of orders is between {1} and {2}.
Key: EXPECTED_COUNT_OF_ORDERS_BETWEEN_1_2

To use this string in the template source code, enter a line similar to the following:

```javascript
${nstranslation.get({"collection":"custcollection_testcol", "key": "EXPECTED_COUNT_OF_ORDERS_BETWEEN_1_2"})({"params": ["10", "15"]})}
```

The parameters provided for the place holders are 10 and 15.

**Result:**

Expected count of orders is between 10 and 15

For more information, see the help topic Translation Collection Overview.

For more information, see the help topic N/translation Module.

Adding Striping to Line Items in Advanced Templates

You can edit HTML markup source in the template editor to add striping to the line items table in an advanced template.
The syntax for adding striping to the line items tables relies on BFO (Big Faceless Organization), a set of third-party libraries used by NetSuite for generating PDF documents. BFO documentation is available at http://bfo.com/products/report/docs/userguide.pdf.

**Note:** If you are having issues with your advanced template, do not contact BFO directly. Always contact NetSuite Customer Support.

**Warning:** Do not modify markup source directly unless you have sufficient CSS and HTML knowledge. NetSuite does not provide support or training in CSS or HTML.

Be aware that the template editor may not function properly if you switch back to WYSIWYG mode after edits have occurred in markup source mode. Some template content may not be represented correctly, may not be accessible for editing, or may not be displayed at all.

If these issues occur, you can preserve template content by not saving the template in WYSIWYG mode and switching back to markup source mode.

To add striping to a line items table:

1. Open the advanced template in the template editor, and click the **Source Code** toggle.
2. Scroll down to the portion of the HTML markup source relating to the rows in the line items table:

   <table>
   <tr><th align="center" colspan="3">$({item_index % 2==0)?string('#ffffff', '#ccffcc')};</th></tr>
   ...
   </table>

   Note: It is recommended that you avoid using the `<tbody>` tag. BFO processing issues may result in an inability to save a template that includes this tag.

3. Edit the `<tr>` tag relating to rows in the line items table, to specify alternating colors for these rows, like the following example:

   `<tr style="background-color: ${{item_index % 2==0)?string('ffffff', '#ccffcc'));"`>

   In this example, even rows use the color represented in hexadecimal by `ffffff` and odd rows use the color represented by `ccffcc`. 
To get hexadecimal codes for striping colors:

You can look up hex codes in the HTML color picker provided in the template editor:

1. In rich text editing mode, click the text color or background color button.

![Color Picker]

2. Click More Colors.

![More Colors]

3. In the Select Color dialog, click a color to see the hex code.

![Select Color]

For example, you want your printed purchase orders to include borders and striping.

Edit the template markup source to add the table styles.
A purchase order that uses the table styles shown will display striping and dotted borders.

```plaintext
/* Source Code Editing in the Template Editor */

table.itemstable {
  border: none;
}

table.itemstable th {
  padding-bottom: 10px;
  padding-top: 10px;
}

table.itemstable td {
  border-bottom: 1px solid gray;
}

table.itemstable tr,td,th {
  background-color: #F2F2F2;
}

table.itemstable td {
  padding-top: 2px;
  border: 1px dotted gray;
}

table.itemstable td,td,th {
  border-left: 1px dotted gray;
}
```

A purchase order that uses the table styles shown will display striping and dotted borders.
Adding Page Breaks to Tables

If you have text overlapping a footer or missing from a printout, ensure that any long content is enclosed in a tag that will split across pages.

The Report Generator has specific rules for where page breaks can occur. A `<table>` tag nested inside a `<td>` tag is cut off at the bottom if it spreads across multiple pages. Only the following tags split correctly if they are spread across multiple pages:

```
<table>
<ul>
<p>
<ol>
<h1>
<h2>
<h3>
<h4>
<blockquote>
```

For more information about the tags that support table pagination, see Page 17 of the BFO (Big Faceless Organization) Guide.

The following example shows a typical table layout.

```
<!-- start of item table in transaction -->
#list record.item as item
    #if item_index==0
        <!-- Header Definition -->
        <tr>
            <th>${item.field1@label}</th>
            <th>${item.long_text_field@label}</th>
            <th>${item.field2@label}</th>
            <!-- ... -->
        </tr>
    #if

    <tr>
        <td>${item.field1}</td>
        <td>${item.long_text_field}</td>
        <td>${item.field2}</td>
        <!-- ... -->
    </tr>
#list
<!-- end of item table in transaction -->
```

If `item.long_text_field` contains more than 2000 text characters, one table line might span the whole page and overflow, and the data might be truncated. If it is necessary to print a long table over more than one page, use code similar to the following.

```
<!-- start of item table in transaction -->
#list record.item as item
    #if item_index==0
        <!-- Header Definition -->
        <tr>
            <th>${item.field1@label}</th>
            <th>${item.long_text_field@label}</th>
            <th>${item.field2@label}</th>
            <!-- ... -->
        </tr>
    #if
```

SuiteBuilder
You can enter any delimiter that you want. The previous example uses a delimiter of "<br/>".

With the #list statement shown in the example, instead of printing out one table row containing all of the paragraphs, more rows are printed. A split function is applied on long_text_field, so "paragraph1<br />paragraph2<br />paragraph3" becomes a list of values "paragraph1","paragraph2","paragraph3". To print the list of paragraphs, in the first row, all of the fields and the first paragraph of long_text_field are printed. In each subsequent row, one paragraph of long_text_field is printed and all other cells are left blank.

A table that looked like this before:

<table>
<thead>
<tr>
<th>Field 1</th>
<th>Long Text Field</th>
<th>Field 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value 1</td>
<td>Paragraph 1</td>
<td>Value 4</td>
</tr>
<tr>
<td></td>
<td>Paragraph 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Paragraph 3</td>
<td></td>
</tr>
<tr>
<td>Paragraph 1</td>
<td></td>
<td>Value 5</td>
</tr>
<tr>
<td>Paragraph 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paragraph 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Paragraph 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Value 3</td>
<td>Paragraph 1</td>
<td>Value 6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Now looks like this:

<table>
<thead>
<tr>
<th>Field 1</th>
<th>Long Text Field</th>
<th>Field 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value 1</td>
<td>Paragraph 1</td>
<td>Value 4</td>
</tr>
<tr>
<td></td>
<td>Paragraph 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Paragraph 3</td>
<td></td>
</tr>
</tbody>
</table>
With each paragraph in a separate row, as many rows as possible are fit onto the page, and any rows that do not fit are moved to the next page.

Printing Subsidiary Logo on Advanced Templates

For OneWorld organizations, instead of having the main company logo print on invoices, you might want the relevant subsidiary logo to be printed. You can do this by creating one advanced template for each subsidiary.

**Note:** This topic applies only to customers that use subsidiaries.

1. Create one advanced template for each subsidiary. Use the `<img src="" />` tag in the template to specify the location of the image. The URL in the `src` parameter can reference an image in the File Cabinet.
2. Create a form for each subsidiary to apply the template created in Step 1.
3. Add the following code to the template:

```html
<#if subsidiary.logo?length != 0>
  <img src="${subsidiary.logo@url}" style="float: left; margin: 7px"/>
</#if>
<#else>
  <#if companyInformation.logoUrl?length != 0>
    <img src="${companyInformation.logoUrl}" style="float: left; margin: 7px" />
  </#if>
</#if>
```

Adding Apply Sublist to Check Templates

By default, Apply sublist items are not printed on check templates. However, you can customize the check template as follows to include the Apply sublist.

```html
<#if check.item?has_content || check.expense?has_content || check.apply?has_content>
  <table style="position: absolute;overflow: hidden;left: 36pt;top: 90pt;width: 436pt;">
    <#list check.expense as expense>
      <tr>
        <td>${expense.account}</td>
        <td>${expense.date}</td>
        <td>${expense.description}</td>
        <td align="right">${expense.amount}</td>
      </tr>
    </#list>
    <#list check.item as item>
      <tr>
        <td>${item}</td>
      </tr>
    </#list>
  </table>
</#if>
```
Using FreeMarker to Work with Hidden Fields Used in Advanced Templates

Advanced printing templates may fail to print when they depend on a sublist hidden field.

In the following custom transaction form example, the Show box beside the Corporate Card field box is not checked. Therefore, Corporate Card is hidden from templates.
When a field is visible (check box is checked), it is represented as a boolean in the advanced template, like this:

```html
<#if expense.corporatecreditcard == true>
    "Corporate Card" is checked!
</#if>
```

When a field is hidden, you must instead compare the value of the field with a string: "T" (true) or "F" (false), like this:

```html
<#if expense.corporatecreditcard == "T">
    "Corporate Card" is checked!
</#if>
```

A template that depends on the Corporate Card hidden field is no longer valid because the type of check box changes from boolean to string. You cannot compare a boolean to a string.

If you use a string condition like the preceding example and then make the field visible, then this condition is not valid.

**Error Messages**

If a template cannot be printed because it depends on a sublist hidden field, you receive an error message similar to one of the following:

- In this example from the template editor, the comparison of string with boolean is not valid:

```
Error
```

```
Error on line 179, column 7 in template.
Detail...
Can't compare values of these types. Allowed comparisons are between two numbers, two strings, two dates, or two booleans.
Left hand operand is a hash+string (wrapper: com.netledger.templates.model.StringModel).
Right hand operand is a boolean (wrapper: ft PIXI.TrueTemplateBooleanModel).
The blamed expression:
>>> expense.corporatecreditcard == true [in template "template" at line 179, column 12]
```

- In the following example, the second condition is not valid because it cannot be evaluated to true or false:

```
Template Editor Error Message
```

```
Error on line 182, column 7 in template.
Detail...
For "#if" condition: Expected a boolean, but this has evaluated to a hash+string (wrapper: com.netledger.templates.model.StringModel):
>>> expense.corporatecreditcard [in template "template" at line 182, column 12]
```

```
UI Error Message

Notice

The template cannot be printed due to the following errors:

Error on line 178, column 52 in template.

Detail...

For "if" condition: Expected a boolean, but this has evaluated to a hash#string (wrapper: com.ledger.templates.model.StringMode)

== expense.corporatecreditcard [in template "template" at line 178, column 57]

----

PPL stack trace ("~" means nesting-related):
  - Failed at: if expense.corporatecreditcard [in template "template" at line 178, column 52]
  ----

Please contact your administrator.

Solution for Unknown Visibility of the Field

You should avoid a scenario where a template depends on a sublist hidden field.

If you do not know whether the field is visible, you should create a condition in which the template behaves correctly in both cases, like in the following example.

```
<#if (expense.corporatecreditcard?is_boolean && expense.corporatecreditcard) || (expense.corporatecreditcard?is_string && expense.corporatecreditcard == "T")>
  "Corporate Card" is checked!
</#if>
```

Adding Bar Codes in Advanced Templates

Printed transactions based on advanced templates automatically include bar codes, in the same manner as printed transactions based on basic layouts. You do not have to manually edit the HTML markup source of the associated advanced template.

The following types automatically include a bar code in printed transactions, when the Bar Coding and Item Labels feature is enabled:

- Bill of Materials
- Cash Refund
- Cash Sale
- Credit Memo
- Invoice
- Item Label
- Packing Slip
- Picking Ticket
- Purchase Order
- Quote
- Remittance Form
- Return Form
- Return Authorization
Sales Order

Most transaction types print the transaction ID along with the bar code. The following example shows a bar code from a printed sales order:

![Sales Order Bar Code]

Item label is an exception. Printed transactions of this type include the UPC code, serial number of the item with the bar code, or both.

FreeMarker supports 128 ASCII characters for serial number and bar codes. If your serial numbers contain special characters, an error may occur during template merging.

Generating Custom Bar Codes in Advanced Templates

You can edit the HTML markup source of your advanced template to add a custom bar code or QR code for any custom transaction you require. For example, you can create a custom bar code on a warehouse picking ticket. With a bar code scanner, a warehouse employee can retrieve relevant inventory information needed to complete the order.

In the Advanced Template Editor, edit the HTML markup source and use the BFO Tag `<barcode />` to create a custom bar code. For more information on editing the source code, see Source Code Editing in the Template Editor.

**Warning:** Do not modify markup source directly unless you have sufficient CSS and HTML knowledge. NetSuite does not provide support or training in CSS or HTML.

Be aware that the template editor may not function properly if you switch back to WYSIWYG mode after you have made edits in markup source mode. Some template content may not be represented correctly, may not be accessible for editing, or may not be displayed at all.

If these issues occur, you can preserve template content by not saving the template in WYSIWYG mode and switching back to markup source mode.

To add a custom bar code to an advanced template:

1. Open the advanced template in the template editor and click the **Source Code** toggle.
2. Scroll down to the portion of the HTML markup source where you want to create a bar code. Add the following code to the template and specify the `codetype`, `showtext`, and `value` parameters as required.

```html
<barcode codetype="qrcode" showtext="false" height="150" width="150" value="http://www.example.com" />
```

**Note:** Your custom bar code will not be visible in the Advanced Template Editor. You must generate a report using your template to see the bar code.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Required/Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>codetype</td>
<td>string</td>
<td>required</td>
<td>Specify the bar code algorithm used to generate the bar code. For a list of supported bar code algorithms, see the following table.</td>
</tr>
<tr>
<td>showtext</td>
<td>boolean</td>
<td>required</td>
<td>If set to <code>true</code>, a readable version of the value is printed below the bar code. If set to <code>false</code>, no readable information is displayed.</td>
</tr>
</tbody>
</table>

---

*Source: Oracle NetSuite*
### Parameter Types and Descriptions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Required/Optional</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>false</td>
<td>required</td>
<td>Note: The following bar code types do not display the printed values below the bar code: aztec, deutschepostmatrix, intelligentmail, maxicode, pdf417, postnet, qrcode, and rm4scc. Specify the information you want to encode using the bar code algorithm. Use the following table to determine which bar code type meets your requirements. Each bar code type has a specific requirement for content length and supported characters. For example, you can encode a URL in an aztec, maxicode, or qrcode by setting the value to &quot;<a href="http://www.example.com">http://www.example.com</a>&quot;. Alternatively, you can encode a serial number in a upca, postnet, or code25 bar code by setting the value to &quot;0123456789&quot;. Note: If value contains characters that are not supported by the bar code algorithm, your template may generate an error. Confirm that the bar code algorithm supports the requirements of the value string.</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>required</td>
<td>Specify the information you want to encode using the bar code algorithm. Use the following table to determine which bar code type meets your requirements. Each bar code type has a specific requirement for content length and supported characters. For example, you can encode a URL in an aztec, maxicode, or qrcode by setting the value to &quot;<a href="http://www.example.com">http://www.example.com</a>&quot;. Alternatively, you can encode a serial number in a upca, postnet, or code25 bar code by setting the value to &quot;0123456789&quot;. Note: If value contains characters that are not supported by the bar code algorithm, your template may generate an error. Confirm that the bar code algorithm supports the requirements of the value string.</td>
</tr>
<tr>
<td>height</td>
<td>number</td>
<td>optional</td>
<td>Note: The height parameter is required for the qrcode. Specify the height of the bar code in pixels. The height parameter may be ignored by some bar code algorithms.</td>
</tr>
<tr>
<td>width</td>
<td>number</td>
<td>optional</td>
<td>Note: The width parameter is required for the qrcode. Specify the width of the bar code in pixels. The width parameter may be ignored by some bar code algorithms.</td>
</tr>
</tbody>
</table>

The following bar code algorithms are supported by advanced templates and the BFO report generator. For more information about each bar code type, see http://bfo.com/products/report/docs/userguide.pdf.

<table>
<thead>
<tr>
<th>codetype</th>
<th>Type</th>
<th>Supported Characters</th>
<th>Length</th>
<th>Example (with showtext = true)</th>
</tr>
</thead>
<tbody>
<tr>
<td>aztec</td>
<td>2D matrix</td>
<td>Full ASCII</td>
<td>12 — 3832</td>
<td><img src="https://example.com/aztec.png" alt="Aztec Bar Code" /></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FNC1 and ESI control codes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>codabar</td>
<td>Linear</td>
<td>Numbers 0–9</td>
<td>Variable</td>
<td><img src="https://example.com/codabar.png" alt="Codabar Bar Code" /></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Symbols : — . $ / +</td>
<td></td>
<td>Note: Requires a Start and Stop character Use one of the following characters for the Start and Stop character: A, B, C, D, E, *, N, T</td>
</tr>
<tr>
<td>code25</td>
<td>Linear</td>
<td>Numbers 0–9</td>
<td>Variable</td>
<td><img src="https://example.com/code25.png" alt="Code25 Bar Code" /></td>
</tr>
<tr>
<td>deutschepost</td>
<td>Linear</td>
<td>Numbers 0–9</td>
<td>Variable</td>
<td><img src="https://example.com/deutschepost.png" alt="Deutschepost Bar Code" /></td>
</tr>
<tr>
<td>codetype</td>
<td>Type</td>
<td>Supported Characters</td>
<td>Length</td>
<td>Example (with showtext = true)</td>
</tr>
<tr>
<td>---------------</td>
<td>--------</td>
<td>--------------------------------------------------------------------------------------</td>
<td>----------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>code39</td>
<td>Linear</td>
<td>▪ Uppercase letters A-Z&lt;br▪ Number 0-9&lt;br▪ Symbols — . $ / + %&lt;br▪ Space</td>
<td>Variable</td>
<td><img src="image" alt="code39_example" /></td>
</tr>
<tr>
<td>code39checksum</td>
<td>Linear</td>
<td>▪ Uppercase letters A-Z&lt;br▪ Number 0-9&lt;br▪ Symbols — . $ / + %&lt;br▪ Space</td>
<td>Variable</td>
<td><img src="image" alt="code39checksum_example" /></td>
</tr>
<tr>
<td>code128</td>
<td>Linear</td>
<td>▪ Full ASCII&lt;br▪ Control Codes</td>
<td>Variable</td>
<td><img src="image" alt="code128_example" /></td>
</tr>
<tr>
<td>databar</td>
<td>Linear</td>
<td>▪ Number 0-9</td>
<td>14 Maximum</td>
<td><img src="image" alt="databar_example" /></td>
</tr>
<tr>
<td>datamatrix</td>
<td>2D matrix</td>
<td>▪ Full ASCII</td>
<td>2335 Maximum</td>
<td><img src="image" alt="datamatrix_example" /></td>
</tr>
<tr>
<td>deutschpostmatrix</td>
<td>2D matrix</td>
<td>▪ Full ASCII</td>
<td>2335 Maximum</td>
<td><img src="image" alt="deutschpostmatrix_example" /></td>
</tr>
<tr>
<td>ean8</td>
<td>Linear</td>
<td>▪ Numbers 0-9</td>
<td>7 + checkdigit</td>
<td><img src="image" alt="ean8_example" /></td>
</tr>
<tr>
<td>ean13/upca</td>
<td>Linear</td>
<td>▪ Numbers 0-9</td>
<td>13+ checkdigit</td>
<td><img src="image" alt="ean13/upca_example" /></td>
</tr>
<tr>
<td>Intelligentmail</td>
<td>2D height</td>
<td>▪ Numbers 0-9</td>
<td>20, 25, 29, or 31 digits</td>
<td><img src="image" alt="intelligentmail_example" /></td>
</tr>
</tbody>
</table>
### Source Code Editing in the Template Editor

<table>
<thead>
<tr>
<th>codetype</th>
<th>Type</th>
<th>Supported Characters</th>
<th>Length</th>
<th>Example (with showtext = true)</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxicode</td>
<td>2D matrix</td>
<td>■ Full ASCII</td>
<td>Maximum 92 ASCII characters</td>
<td><a href="#">Maxicode Example</a></td>
</tr>
<tr>
<td>pdf417</td>
<td>2D stacked</td>
<td>■ Full ASCII</td>
<td>Maximum 1850 ASCII characters or 2725 numeric characters</td>
<td><a href="#">PDF417 Example</a></td>
</tr>
<tr>
<td>postnet</td>
<td>Linear</td>
<td>■ Numbers 0–9</td>
<td>5, 9, or 11 + check digits</td>
<td><a href="#">POSTNET Example</a></td>
</tr>
<tr>
<td>qrcode</td>
<td>2D matrix</td>
<td>■ Full ASCII</td>
<td>Maximum 1520 ASCII characters or 2509 numeric characters</td>
<td><a href="#">QRCode Example</a></td>
</tr>
<tr>
<td>rm4scc</td>
<td>2D height</td>
<td>■ Uppercase letters A–Z</td>
<td>Variable</td>
<td><a href="#">RM4SCC Example</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Number 0–9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Scripting with Advanced Templates

This section contains SuiteScript solutions that you can use to supplement your SuiteBuilder development.

SuiteScript is a JavaScript-based API that gives developers the ability to extend NetSuite beyond the capabilities provided through SuiteBuilder. With SuiteScript, you can use JavaScript code to perform validations and calculations on forms, create custom forms and portlets, schedule bulk processing of records or customize business process workflows. For more information, see SuiteScript Overview.

- Using Custom Data Sources for Advanced Printing
- Using SuiteScript for Transaction Records
- Using SuiteScript to Apply Advanced Templates to Non-Transaction Records
- Using SuiteScript 2.0 to Combine Multiple Data Sources in One Advanced Template

### Using Custom Data Sources for Advanced Printing

You can combine custom data stored outside of NetSuite with data stored in NetSuite to be displayed together in printed forms for NetSuite transactions. This capability is supported by the SuiteScript 2.0 render module, which includes a method to support XML and JSON data sources in advanced PDF/HTML templates. For more information about the added method, see the help topic N/render Module.
Data from JSON and XML is handled as a string, because there is no information about the data type. You can use JavaScript or FreeMarker functions for formatting, if required. For example, you might want to ensure that dates all use the same format.

To include a custom data source, you customize a standard template and use the source code view to add the new fields. The sections with fields from the custom data source have to be surrounded by the FreeMarker tag `<#if ALIAS?has_content>  </#if>`. The tag is required because custom data sources cannot be displayed in the template editor. If you view a preview of the template, the external data source is not shown.

Because FreeMarker supports only XML, JSON data sources have the same restrictions as XML, for example, a property cannot start with a digit. The JSON data source is converted to XML for printing.

You must know what information is included in the external data source and use syntax like the following:

```xml
<?xml version="1.0"?>
<!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
<pdf>
<body>
  Custom Data Sources:
  <#if XML?has_content>
    ${XML.book.title}<br />
    ${XML.book.chapter[1].title}
  </#if>
  <br />
  <#if JSON?has_content>
    ${JSON.book.title}<br />
    ${JSON.book.chapter[1].title}
  </#if>
  <br />
  <#if JSON_STR?has_content>
    ${JSON_STR.book.title}
  </#if>
  <br />
  <#if XML_STR?has_content>
    ${XML_STR.book.title}
  </#if>
</body>
</pdf>
```

Next, you create a SuiteScript 2.0 file that uses the `addCustomDataSource` method in the renderer object. Use code like the following:

```javascript
renderer.addCustomDataSource({format: render.DataSource.XML_DOC, alias: "XML", data: xmlObj});
renderer.addCustomDataSource({format: render.DataSource.XML_STRING, alias: "XML_STR", data: xmlString});
renderer.addCustomDataSource({format: render.DataSource.OBJECT, alias: "JSON", data: jsonObj});
renderer.addCustomDataSource({format: render.DataSource.JSON, alias: "JSON_STR", data: jsonString});
```

The form is processed as follows:

1. The customized template is loaded into the renderer.
2. Data from the NetSuite record is loaded.
3. Data from the custom data source is added (XML, JSON, XML_STRING, and JSON_STRING in the previous example).
4. The form is printed using the renderer.
For example, you want to print gift certificates for employees that combine name and address information stored in NetSuite with data from an external survey. You customize an advanced template using the source code view of the Template Editor. You also add fields from the XML file that contains exported results from the survey. Then you create a SuiteScript file that uses the `render` module to do the following: load the customized template, add data from the NetSuite record, add custom data from the XML file, and print the gift certificate forms. Data from the two sources are combined to print personalized certificates for employees.

To view an example with code snippets, see Example of Using Custom Data Sources for Advanced Printing.

### Example of Using Custom Data Sources for Advanced Printing

When setting up an advanced template, you can use multiple data sources. This example shows how to use a JSON object and an XML file together with NetSuite data from a saved search to create a packing slip for a customer.

The JSON object provides customer names and language preferences.

```json
{"customers":[
    {"firstName":"John", "lastName":"Doe", "language":"English"},
    {"firstName":"Anna", "lastName":"Smith", "language":"Spanish"},
    {"firstName":"Peter", "lastName":"Jones", "language": "English"}
]}
```

The XML file provides a holiday greeting in the customer's preferred language.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<message>
    <engbody>Happy holidays! We are closed from Dec 22 - Jan 3. Take advantage of our great prices before Christmas!</engbody>
    <spbody>¡Feliz Navidad! Estamos cerrados del 22 de diciembre al 3 de enero. Aproveche nuestros excelentes precios antes de Navidad!</spbody>
</message>
```

The advanced PDF/HTML template is edited in markup source view to incorporate the XML file and JSON object.

```xml
<?xml version="1.0"?><!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd"><pdf><head><macrolist><macro id="nlheader">
<!-- XML Part -->
@if XML?has_content>
    <p style="color: blue">
        @if .locale?starts_with("es")>
            $(XML.message.spbody)
        @else>
            $(XML.message.engbody)
        @/if
    </p>
@endif
<!-- End XML Part -->
</macro></macrolist></head><body> <!-- XML Part -->
@if XML?has_content>
    <p style="color: blue">
        @if .locale?starts_with("es")>
            $(XML.message.spbody)
        @else>
            $(XML.message.engbody)
        @/if
    </p>
@endif
<!-- End XML Part -->
```

SuiteBuilder
```html
<table class="header">
  <tr>
    <td rowspan="3">
      <span class="addressheader">${companyInformation.companyName}</span><br />
      <span class="addressheader">${companyInformation.addressText}</span>
    </td>
    <td align="right">${record@title}</td>
  </tr>
  <tr>
    <td align="right">#${record.tranid}</td>
  </tr>
  <tr>
    <td align="right">${record.trandate}</td>
  </tr>
</table>

</macro>
</macrolist>

<style type="text/css">
  table {
    font-family: sans-serif;
    font-size: 9pt;
    table-layout: fixed;
  }
  th {
    font-weight: bold;
    font-size: 8pt;
    vertical-align: middle;
    padding: 5px 6px 3px;
    background-color: #e3e3e3;
  }
  td {
    padding: 4px 6px;
  }
  td.addressheader {
    font-size: 8pt;
    padding-top: 6px;
    padding-bottom: 2px;
  }
  span.number {
    font-size: 16pt;
  }
</style>

</head>

<body header="nlheader" header-height="20%" padding="0.5in 0.5in 0.5in 0.5in" size="Letter">
<table style="width: 100%; margin-top: 10px;"
  <tr>
    <td colspan="3" style="font-size: 8pt; font-weight: bold;">${record.billaddress@label}</td>
    <td colspan="3" style="font-size: 8pt; font-weight: bold;">${record.shipaddress@label}</td>
    <td colspan="5" style="font-size: 12pt; background-color: #e3e3e3; font-weight: bold;">${record.
      total@label?upper_case}</td>
  </tr>
  <tr>
    <td colspan="3" style="padding: 0;">${record.billaddress}</td>
    <td colspan="3" style="padding: 0;">${record.shipaddress}</td>
</table>
```

SuiteBuilder
<table style="width: 100%; margin-top: 10px;">
<tr>
<th>${record.paymentmethod@label}</th>
<th>${record.otherrefnum@label}</th>
<th>${record.shipmethod@label}</th>
<th>${record.shipdate@label}</th>
</tr>
<tr>
<td style="padding-top: 2px;">${record.paymentmethod}</td>
<td style="padding-top: 2px;">${record.otherrefnum}</td>
<td style="padding-top: 2px;">${record.shipmethod}</td>
<td style="padding-top: 2px;">${record.shipdate}</td>
</tr>
</table>

<#if record.item?has_content>
<table style="width: 100%; margin-top: 10px;">
<!-- Start Items -->
<#list record.item as item>
<#if item_index==0>
<thead>
<tr>
<th align="center" colspan="3">${item.quantity@label}</th>
<th colspan="12">${item.item@label}</th>
<th align="right" colspan="4">${item.rate@label}</th>
<th align="right" colspan="4">${item.amount@label}</th>
</tr>
</thead>
</#if>
<tr>
<td align="center" colspan="3" line-height="150%">${item.quantity}</td>
<td colspan="12">
<span style="font-weight: bold; line-height: 150%;">${item.item}</span>
<br />
${item.description}
</td>
<td align="right" colspan="4">${item.rate}</td>
<td align="right" colspan="4">${item.amount}</td>
</tr>
</#list>
<!-- End Items -->
</table>
</#if>
<table style="page-break-inside: avoid; width: 100%; margin-top: 10px;">
<tr>
<td colspan="4">&nbsp;</td>
</tr>
</table>

<hr style="width: 100%; color: #d3d3d3; background-color: #d3d3d3; height: 1px;"/>
</#if>
<table style="page-break-inside: avoid; width: 100%; margin-top: 10px;">
<tr>
<td colspan="4">&nbsp;</td>
</tr>
</table>
<table>
  <tr><td align="right" style="font-weight: bold;">${record.subtotal@label}</td><td align="right">${record.subtotal}</td></tr>
  <tr><td colspan="4">&nbsp;</td><td align="right" style="font-weight: bold;">${record.taxtotal@label}</td><td align="right">${record.taxrate}%</td></tr>
  <tr style="background-color: #e3e3e3; line-height: 200%;"><td background-color="#ffffff" colspan="4">&nbsp;</td><td align="right" style="font-weight: bold;">${record.total@label}</td><td align="right">${record.total}</td></tr>
</table>

<!-- JSON Part -->
<#if JSON?has_content>
  <p style="color: red">JSON data:</p>
  <table style="width: 100%; margin-top: 10px;">
    <thead>
      <tr>
        <th>First Name</th><th>Last Name</th><th>Language</th>
      </tr>
    </thead>
    <#list JSON.customers as item>
      <tr>
        <td>${item.firstName}</td><td>${item.lastName}</td><td>${item.language}</td>
      </tr>
    </#list>
  </table>
</#if>

<!-- Saved Search Part -->
<#if SEARCH?has_content>
  <p style="color: green">Saved Search results:</p>
  <table style="width: 100%; margin-top: 10px;">
    <thead>
      <tr>
        <th>trandate</th><th>amount</th><th>entity</th>
      </tr>
    </thead>
  </table>
</#if>
The SuiteScript file incorporates the JSON object and XML file with NetSuite data and the advanced template.

```javascript
/** *
@NApiVersion 2.x
*/
require(['N/render', 'N/file', 'N/record', 'N/search'],
function(render, file, record, search) {
    // load JSON file from file cabinet
    var jsonFile = file.load({
        id: '#json_content_file_cabinet_id#
    });
    // load XML file from file cabinet
    var xmlFile = file.load({
        id: '#xml_content_file_cabinet_id#
    });
    // load printing template file from cabinet
    var templateFile = file.load({
        id: '#template_content_file_cabinet_id#
    });

    var renderer = render.create();

    renderer.templateContent = templateFile.getContents();

    // add JSON custom data source
    renderer.addCustomDataSource({
        alias: "JSON",
        format: render.DataSource.JSON,
        data: jsonFile.getContents()
    });

    // add XML custom data source
    renderer.addCustomDataSource({
        alias: "XML",
        format: render.DataSource.XML_STRING,
        data: xmlFile.getContents()
    });

    // add search data source
    var rs = search.create({
```
```javascript
  type: search.Type.TRANSACTION,
  columns: ['trandate', 'amount', 'entity'],
  filters: []
}).run();
var results = rs.getRange(0, 1000);
renderer.addSearchResults(
  templateName: 'SEARCH',
  searchResult: results
});

// add record data source
var objRecord = record.create(
  type: record.Type.SALES_ORDER
});
renderer.addRecord(
  templateName: 'record',
  record: objRecord
});

// render PDF file and save
var invoicePdf = renderer.renderAsPdf();
  invoicePdf.folder = #file_cabinet_folder_id#;
  var id = invoicePdf.save();
});
```

The output of this example resembles the following.
Using SuiteScript for Transaction Records

**How SuiteScript APIs Work**

The SuiteScript API lets you programmatically extend NetSuite beyond the capabilities offered through SuiteBuilder customization. Most SuiteScript APIs pass record, field, sublist, tab, search filter, and search column IDs as arguments.

To determine which script you need, refer to the documentation, where the SuiteScript API is organized by the types of tasks most developers want to perform. To get started with the SuiteScript API for SuiteScript 2.0, see the help topic [SuiteScript 2.0 API Reference](#).

After you have determined which script you need, perform the following steps to get a script to run in NetSuite:

1. Create your script.
2. Create a NetSuite Script record for your script. You will be prompted to load the script file.
3. Create a NetSuite Script Deployment record and specify script runtime options.
For complete details on each step in the process, start with the Running SuiteScript 1.0 in NetSuite Overview topic in the NetSuite Help Center.

**SuiteScript 2.0 Module and Members for Printing Transaction Records**

SuiteScript 2.0 supports a render module that you can use to programatically print, create PDFs, create forms from templates, and create email messages from templates.

The `render.TemplateRenderer` object member provides a template engine object and related methods so you can use advanced PDF/HTML templates to produce HTML and PDF printed forms. For details, see the help topic `render.TemplateRenderer`. If you associate an advanced template with the custom form saved for a transaction and use this API to print the transaction, the advanced template is used to format the printed transaction.

In SuiteScript 2.0, it is possible to reference a template by ID. The SuiteScript 2.0 `render` module includes a method that supports referencing a template by its script ID, `TemplateRenderer.setTemplateByScriptId`. Each template's script ID can be set and reviewed in the Template Setup popup window of the Template Editor.

In addition to the method for referencing templates by script ID, the `TemplateRenderer.setTemplateById` method supports referencing a template by its system-generated internal ID. This method would only be required for specific custom printing solution use cases.

For information about the NetSuite records that support SuiteScript, see SuiteScript Supported Records.

**Advanced Template List for Custom Printing Solutions**

When you create a custom field of type List/Record, Advanced PDF/HTML Templates is available as an option in the List/Records dropdown list on the custom field setup page available at Customization > Lists, Records & Fields > [Custom Field]. This option provides users with a list of all available advanced templates, for use in a case where you create a script that enables users to specify a form to be used for printing, as part of a custom printing solution.

For example, to provide a custom printing solution for your users you could create a purchase order form and add a custom field of type List/Record with Advanced PDF/HTML Templates specified. Then you use the SuiteScript 2.0 `render` module method to refer to each user's selected advanced template by NetSuite internal ID. A Suitelet can then be used to take the user's selection and pass in the internal ID of the template to print the purchase order.

**Notes about Advanced Template List**

- This option should be used only with a scripted custom printing solution.
- The list of available templates is not filtered. All advanced PDF/HTML templates are available, including those that are not for the correct transaction type.
- This field alone does not allow users to specify the template to be used for printing a form. By default, printing preferences for advanced forms are specified by Setting Custom Forms to Use Advanced Templates.

**Printing the Correct Currency Symbol**

If you are using SuiteScript to render saved search results, verify the following to ensure that the correct currency symbol is used in the advanced PDF/HTML template.

1. Add Currency to the saved search columns.
2. Use code something similar to the following in the JavaScript file.

```
<#if result.currency == 'Euro'>${result.fxamountremaining?string('#,##0.00')} €
```

SuiteBuilder
SuiteScript 1.0 Functions and Objects for Transaction Records

SuiteScript 1.0 supports a template engine object and related methods so you can apply advanced template format capabilities programmatically. For details, see the help topics nlapiCreateTemplateRenderer() and nlobjTemplateRenderer. For information about the NetSuite records that support SuiteScript, see the help topic SuiteScript Supported Records.

In addition to this function and object, the SuiteScript function nlapiPrintRecord(type, id, mode, properties) supports the use of advanced templates. If you associate an advanced template with the custom form saved for a transaction and use this API to print the transaction, the advanced template is used to format the printed transaction.

You can also use SuiteScript to apply advanced templates to printed records that are not transactions. For information, see Using SuiteScript to Apply Advanced Templates to Non-Transaction Records.

Printing a Large Volume of Documents

To print hundreds or thousands of documents, you must use the SuiteScript API and follow these steps.

1. Create a Saved Search of the documents that you want to print, or, if you need to print out more than 1000 documents, create multiple searches where you batch results of 1000 rows or less.

2. Create the Template using the <pdfset> tag. The <pdfset> tag specifies that everything between the <pdf> tags will be processed separately, which improves the efficiency of the printing algorithm.

```xml
<?xml version="1.0"?>
<!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
<pdfset>
  <#list results as result>
    <pdf>
      <head>
      </head>
      <body>
        <h1 align="right">Invoice</h1>
        <p align="right">#${result.tranid}</p>
        <p align="right">#${result.trandate}</p>
        <p>Total: ${result.amount}</p>
      </body>
    </pdf>
  </#list>
</pdfset>
```

3. Write the script file, including nlapiSearchRecord to call the saved search and nlapiCreateTemplateRenderer to generate the documents.

4. Create a Suitelet and attach the SuiteScript to it, selecting the appropriate function to execute.

Using SuiteScript to Apply Advanced Templates to Non-Transaction Records

Currently, you must use SuiteScript to apply advanced print templates to non-transaction record types. This topic provides an example of how to use an advanced template with the Employee record type. For general information about SuiteScript 2.0, see the help topic render.TemplateRenderer.
In this example, the goal is to print employee access cards that include name, hire date, and a photo. Before you create a script for this purpose, you need to:

- Use HTML to create a basic print template. This template code should:
  - Conform to FreeMarker syntax. For more information, see the FreeMarker documentation at http://freemarker.apache.org/docs/index.html.
- Understand the data model for the record to determine the fields that will need to be included in your template. You can get the IDs of these fields in the NetSuite UI.
  - To make field IDs available, go to Home > Set Preferences and ensure that the Show Internal IDs option is enabled on the General subtab, Defaults area.
  - Select an Employee record and for each field that you want to include in the template, click the field label to display the field level help text. The field ID is in the bottom right corner.

### Sample Template for Employee Access Card

The following sample template code provides some basic formatting and references 3 field IDs from the employee record: entityId, hiredate, and image.

```xml
<xml version="1.0"/>
<!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
<pdf>
  <head/>
</head>
<body width="2.35in" height="3.75in">
  <img align="center" width="100px" height="125px" src="${employee.image?html}"></img>
  <p align="center" style="font-size:24px;font-weight:bold;">${employee.entityId}</p>
  <p align="center">${employee.hiredate@label}<br />${employee.hiredate}</p>
</body>
</pdf>
```

### Sample SuiteScript for Printing Employee Access Card

When you have your template code, you need to write the actual script. For SuiteScript 1.0, use the `nlapiCreateTemplateRenderer()` function and methods from the `nlobjTemplateRenderer` object. For SuiteScript 2.0, use `render.TemplateRenderer`. For SuiteScript 1.0, the script needs to include the raw string of the template. The following sample SuiteScript 1.0 code provides a model.

```javascript
function generate(request, response) {
  var employee = nlapiLoadRecord('employee', 1074); // Load employee record by ID
  var renderer = nlapiCreateTemplateRenderer();
  var template = '<?xml version="1.0"?><!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">'
      + '<pdf>'
      + '  <head>'
      + '    <body width="2.35in" height="3.75in">
      + '      <img align="center" width="100px" height="125px" src="${employee.image?html}"/>
      + '    </body>'
      + '</head>
      + '</pdf>'
      + '</xml>
  nlapiRenderPDF(renderer, template, ${employee.entityId});
}
```
When you have your script code, you need to save it in a JS file which you load to the NetSuite File Cabinet when you create a script record. Create a script record of the type Suitelet and deploy the script in NetSuite. For instructions for these tasks, see the help topics Suitelets, Steps for Creating a Script Record, and Steps for Defining a Script Deployment. The saved script deployment record provides a URL that you can click to see the PDF after the script is run.

Sample Employee Access Card PDF

The following screenshot shows a printed employee access card based on the sample script code in this example:

Using SuiteScript 2.0 to Combine Multiple Data Sources in One Advanced Template

This example shows how to combine multiple data sources on one advanced printing template. The example includes inline comments describing what the script is doing, and links to the relevant help topics are provided. You can run this example directly in the Script Debugger.

Notes for the example:

- Subsidiaries are included, thus the example will work only in a OneWorld account.
- Specific record numbers are referenced and must exist in the account.
The N/render module example illustrates how to do the following:

- Define and use a template directly from SuiteScript, without exposing the template to end users. You are not limited to the record-specific templates when using SuiteScript. You can enter strings that contain your template source code and provide them directly to the renderer.
- Work with multiple record instances of different record types.
  - Load them.
  - Add them to the template.
  - Use them from the template.
- Add a custom data source to the template, in this case a JSON object with a collection.
- Include static barcodes and generate barcodes based on record values.

To view the example, copy the code snippet below and paste it into the Script Debugger <link>. After you run the script, go to Document > Files > SuiteScripts. The result of running the script is a PDF file named pdf-test.pdf.

```javascript
// Freemarker docs: https://freemarker.apache.org/docs/ref.html
/**
 * @NApiVersion 2.x
 */
require(['N/render', 'N/search', 'N/record', 'N/file'],
function(render, search, record, file) {

  function RenderPdfTest() {
    var invoiceId = 22; // one of the existing invoices
    var subsidiaryParent = 1;
    var subsidiaryCz = 6;
    // get instance of TemplateRenderer
    var renderer = render.create();
    // after you call the renderer, you can populate its data model by adding records, searches and custom data sources

    // add parent company and one child subsidiary
    renderer.addRecord('subsidiary', record.load({
      type: record.Type.SUBSIDIARY,
      id: subsidiaryParent
    }));

    renderer.addRecord('subsidiarycz', record.load({
      type: record.Type.SUBSIDIARY,
      id: subsidiaryCz
    }));

    // load Invoice and add it to the renderer's data model using name 'record'
    SuiteBuilder
  }

  RenderPdfTest();
});
```
// (you can pick any name you want)
renderer.addRecord('record', record.load({
    type: record.Type.INVOICE,
    id: invoiceId
})));

// add custom data source
jsonObj = {
    name: "John",
    age: 30,
    city: "Brno",
    collection: [
        {txtId: 'ID: 1'},
        {txtId: 'ID: 2'}
    ]
};
renderer.addCustomDataSource({
    format: render.DataSource.OBJECT,
    alias: "myJsonObject",
    data: jsonObj
});

// here you can add a condition to load a template identified by TEMPLATE_ID
// renderer.setTemplateById(TEMPLATE_ID); // ID of advanced invoice pdf template

// or, you can supply the template source code directly
// this way the template is not exposed to customers (because you can hide bundled SuiteScripts)

// - renderer.templateContent accepts any valid template source code string - it doesn't care about
// where it came from

renderer.templateContent =
    '<?xml version="1.0"?>
<!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
    <pdf>
        <head>
            <link name="NotoSans" type="font" subtype="truetype" src="${nsfont.NotoSans-Regular}
            <head/>
            <body padding="0.5in 0.5in 0.5in 0.5in" size="Letter">
                Hello world! <br/>
                {subsidiary.name@label}: {subsidiary.name} <br/>
                {subsidiarycz.name@label}: {subsidiarycz.name} <br/>
                {record.entity@label}: {record.entity} <br/>
            </body>
        </head>
    </pdf>
Scripting with Advanced Templates

`' ${record.location@label}: ${record.location} <br/>

// first item from item sublist of the invoice - (you can iterate through items using Freemarker syntax if needed)
' ${record.item[0].item@label}: ${record.item[0].item} <br/>

' QR content: ${record.item[0].item@label}: ${record.item[0].item}<br/>
' <barcode codetype="qrcode" showtext="false" height="150" width="150" value="${record.item[0].item@label}: ${record.item[0].item}" /><br/>
' Code39 bar content: ${record.tranid}
' <barcode codetype="code39" showtext="false" height="30" width="150" value="${record.tranid}" /><br/>

// some data from json object
'
' myJsonObject.name: ${myJsonObject.name} <br/>
' myJsonObject.age: ${myJsonObject.age} <br/>
' myJsonObject.collection[0].txtId: ${myJsonObject.collection[0].txtId} <br/>
'
' myJsonObject.collection[1].txtId: ${myJsonObject.collection[1].txtId} <br/>
'
' </body>' +
' </pdf>;

// render PDF
var newfile = renderer.renderAsPdf();
newfile.folder = -15; // ID of folder where file created
newfile.name = "pdf-test";
var fileId = newfile.save();
}
RenderPdfTest();
};

Previewing Advanced PDF/HTML Templates

The editor for advanced PDF/HTML templates supports previewing of templates. This feature enables you to see how template changes affect PDF output during the time that are editing a template. You can preview the template from both WYSIWYG and markup source editing modes.
Click the **Preview** button to generate and display a PDF based on the current template definition. The PDF that appears uses artificial data that simulates the real values as follows:

- $9,999.99 represents dollar values.
- 6/22/2015 represents a date value in U.S. date format.
- *Lorem ipsum* represents text values.
If a preview of the template cannot be shown, an error message appears.

**Saving an Advanced Template**

When saving an advanced template, you have the following options:

- **Save** – Saves the template using the existing file name
- **Save As** – Creates a copy of the template
- **Save & Edit** – Saves the template and remains in edit mode

**Creating a Copy of an Advanced Template**

If you want to use an existing customized advanced template as the basis for a new template, you can use the **Save As** button to save a template under another name.

1. From the Advanced PDF/HTML Template List page, click **Edit** beside the template that you want to copy.
2. The Template editor appears. Click **Template Setup** and name the new template. For more information, see Using the Template Setup Window.
3. Click **Save**.
4. In the template editor, click **Save As**. The new template is saved.
Saving an Advanced Template

Saving Your Template and Continuing to Make Edits

If you want to save the progress of your template edits but want to continue making further edits, click Save & Edit. If your template is successfully saved, a message appears informing you that the template was saved successfully.

Template Errors When Saving

After clicking Save, Save As, or Save & Edit, if your template contains errors, an error message appears, and you are asked if you want to submit anyway. Choose from the following options:

- To override the error and save the template, click Save. The template is saved. Note that there are valid cases when the template works in production but fails the validation.
- To cancel the save and fix the error, click Cancel. The template is not saved, and you return to edit mode to fix any issues.

Formatting in Advanced PDF/HTML Templates

When working in advanced PDF/HTML templates, such as email, fax, and marketing, it is important that you use valid XML formatting. Invalid formatting can result in broken pages. Code may seem to work
when it is opened in a single page, but the same code can cause broken pages when included in other pages, like Activity View.

Some examples of invalid XML include:

- Cross tags.
- Missing tags or missing end tags, such as `<table>` and `</table>`.

Format tables using CSS. The `<table>` method of formatting tables is obsolete and prone to user error. CSS formatting outside of the `<head>` and `<body>` sections is not recommended. Any content between `</head>` and `<body>` or after `</body>` is suspicious and can be problematic because malicious JavaScripts are often placed there.

To reduce the risk of invalid XML formatting, when in DIV mode, copy the XML code from your editor and paste it in the source code view of the template editor. When you switch the template to WYSIWYG mode or save, NetSuite validates the template code.

For more information about valid formatting, refer to https://www.w3.org/.

**Error Messages in Advanced Templates**

**Virtual Record when Saving Template**

When a template is saved, the printing engine generates a virtual record to make sure that the template can be printed. If your template uses a substring function, a temporary string may be used that does not contain enough characters to let the function to go through correctly. If this occurs you might see an error message similar to the following.

The template cannot be saved due to the following errors: Exception during template merging. java.lang.StringIndexOutOfBoundsException: String index out of range: 9

For example, the following line in a template will work when you are printing the record but not when the template is saved because the temporary string will not be long enough:

```javascript
${item.taxcode?substring(4,9)}
```

To ensure that the template saves correctly, add an if statement to verify the length of the string.

```javascript
<#if (item.taxcode?length > 9)>
  ${item.taxcode?substring(4,9)}
</#if>
```

**Errors when Previewing or Saving Templates**

When you preview or save an advanced template, if the preview cannot be shown, or if the template cannot be saved, an error message appears. The message displays the line and the column where the error occurred.

Also, when a PDF document does not print and fails with an error, the error message includes the template name, line, and column where the error occurred. Click Detail... to see more information about the error encountered. Administrators can use this information to address the issue with the template.

**Changing the Script ID of a Custom Template**

A Change ID button is available on the advanced PDF/HTML templates configuration page for custom templates. To change the script ID of a custom advanced PDF/HTML template, click Change ID.
On the Change ID page, enter the new script ID for the template. Script IDs for custom templates begin with custtmpl.

Click Save.

**Note:** You cannot change the script ID of standard templates.

### Advanced Templates for Printing Saved Search Results

Advanced PDF/HTML Templates support printing saved search results for any record type. You can define multiple print templates for a single saved search using the same Template Editor that you use to edit transaction templates. To create a template for printing saved search results, first you create and save a saved search.

To create a saved search or saved search template, your role must have the permissions and search access for the desired record types. For more information, see the help topic Permissions for Searches.

**To create a template for printing saved search results:**

1. Choose an option:
   - On the Saved Search list page, click Edit on the saved search you want to create a template for. Click the New Template button at the top of the page. A new advanced template appears, and includes all of the results fields from the saved search. To change the template title, click Template Setup to open the Template Setup window.
   - On the Advanced PDF/HTML Templates list page, click the New Template button at the top of the page. The Template Setup window appears.
Change the template title and settings as required. In the **Saved Search** field, select the saved search template and then click **Save**.

The Advanced Template Editor generates a template with the saved search results in a table. All of the fields listed on the Results subtab of the saved search are displayed as table columns. The table columns may become unreadable if the saved search includes more than 10 fields. You can edit the template to remove columns, or edit the saved search to return fewer results.

2. Make changes to the template as required.

   If there is more than one Formula(Currency) field in the saved search, reference the fields by using \$\{result.formulacurrency\} for the first field, \{result.formulacurrency_1\} for the second field, \$\{result.formulacurrency_2\} for the third field, and so on.

   InternalId is not available for any field unless it is specified as a separate output field on the Results subtab of the saved search.

For more information about formatting the template, see Customizing Advanced Templates in the Template Editor.

**Sample Templates**

When printing saved search results, you may want to print a list, or print one record per page.

To print a list of records, modify your template to resemble the following.

```xml
<?xml version="1.0"?>
<!DOCTYPE pdf PUBLIC "-//bs.faceless.org/report/" report-1.1.dtd>

<body padding="0.5in 0.5in 0.5in 0.5in" size="letter">

<list results as result>
  <result entityid=""/>
  <result contact=""/>
  <result phone=""/>
  <result email=""/>
  <result salesrep=""/>
</list>
</body>
```

Printing from this templates shows a list of results.
To create a template that prints one record per page, the template code resembles the following.

The location of the `<#list>` statement results in one page per record.
Printing Saved Search Results

Clicking the Print button on a saved search results page displays a popup window where you can choose from a list of available templates. This functionality is only available when the Advanced PDF/HTML Templates feature is enabled and a template is available for printing the saved search results.

Only templates that are available for the saved search are available. To use browser printing, select Default from the Template dropdown list. Then click Print.

Advanced Templates Support for Company Printing Preferences

Advanced templates support company printing preferences, as defined at Setup > Company > Printing & Faxing Preferences, on the Printing subtab. The following company printing preferences are applied to printed transactions that use advanced templates:

- Customers Default to Print Transactions
- Vendors Default to Print Transactions
- Print Return Form with Packing Slip
- Print Remittance Form with Invoices and Statements
- Print Transaction Forms Landscape
- Print Separate Voucher for Bill Payments
- Print Accounts on Vouchers
- Print Discount and Shipping Lines on Columns
- Use Location Address on Forms
- Check Printing — Default Check Type (Standard or Voucher)

Note that preferences listed in the Messages section of the Printing & Fax Preferences page, such as disclaimers and other message text, are not applied to printed transactions that use advanced templates. You can define specialized message text on any custom advanced template. This alternative enables you to vary message text instead of having one standard message defined on the preferences page.
Basic Printing Layouts

Basic printing layouts include Transaction Form HTML Layouts and Transaction Form PDF Layouts. You can use these layouts to define the arrangement of fields on printed transactions in NetSuite. Basic printing layouts show the labels above the data and use a black background color.

You can customize the layouts for transaction form PDF layouts and transaction form HTML layouts. A custom layout can be used for a custom form, so that any transactions associated with that form use the formatting provided by the custom layout.

**Important:** Basic layouts will be deprecated in a future release. We encourage you to use Advanced PDF/HTML Templates instead because new features are added exclusively to advanced printing. For information, see Advanced PDF/HTML Templates.

For PDF or HTML output, advanced templates provide more customization capabilities than basic layouts, and the built-in template editor can be used in either WYSIWYG or source code mode. The advanced PDF/HTML templates also support current industry standards for HTML-based editing.

Advanced templates support all transaction and print types supported by basic layouts.

In OneWorld accounts, if you print transactions using basic layouts, the logo and address are sourced from the vendor’s primary subsidiary. To use the subsidiary logo and address from the transaction record when printing, use advanced templates. For more information, see Advanced PDF/HTML Templates.

See the following topics:

- Customizing Transaction Form PDF Layouts
- Transaction Form HTML Layouts
- Totals Transaction Form Layouts

Customizing Transaction Form PDF Layouts

A transaction form PDF layout defines the arrangement of fields on printed standard and classic (or legacy) PDF transaction documents in NetSuite. You can customize transaction form PDF layouts by customizing borders and content position, repositioning and resizing fields, or by changing the fonts and colors used.

**Important:** Basic layouts will be deprecated in a future release. We encourage you to use Advanced PDF/HTML Templates instead because new features are added exclusively to advanced printing. For information, see Advanced PDF/HTML Templates.

Custom transaction form PDF layouts are only available when you print your forms in PDF. If you print using HTML, a custom layout does not affect the look of your form. To print using PDF, go to Home > Set Preferences > Transactions tab. Clear the Print Using HTML box, and click Save.

To customize a PDF layout, you first set up the color of text and backgrounds on the form and set the page height and width of printed forms. Then you use the Form Editor to change the size and orientation of elements of the form.

You can choose to enter your measurements in inches or in centimeters by setting the Form Size Measurements preference at Setup > Company > Preferences > Printing & Fax Preferences on the Printing subtab.
To customize a transaction form PDF layout:

1. Go to Customization > Forms > Transaction Form PDF Layouts.
2. Click Customize next to the layout you want to customize.

   **Note:** A transaction form PDF layout cannot be customized for check vouchers.

To create a new custom layout based on an existing custom layout, click Edit next the layout. Then enter a name for the new layout, and click Save As.

3. On the Custom PDF Layout page, enter a name for your custom layout.
4. If desired, edit the width and height properties for your page.
5. Check the Layout is Preferred box to make this layout your preferred layout for this type of transaction.
6. Set the following colors for your form layout:
   - **Text Color** – Controls the color of the text in the form.
   - **Fill Color** – Sets the background color in the sections of the form.
   - **Label Text Color** – Controls the color of the text at the top of the main section of the form.
   - **Label Fill Color** – Determines the background color of the label for the main section of the form.
   - **Border Color** – Sets the border color for each section of the form.

   You can click the color palette to choose a color or enter the hexadecimal code for the color you want to set.

   As you make changes to the colors, the changes are shown in the preview at the bottom of the page.

7. Click an element of the form that you want to customize. You can also select an element using the Selected Element list. If you want to add a custom element to the form, click Add Custom Element.

   **Note:** Custom elements can only be defined for packing slips, picking tickets and transaction form PDF layouts. For more information on custom elements, see Defining Custom Elements.

You can do any of the following to each form element:

- Resize a section by clicking and dragging the handle in the lower right corner. Note that the cursor does not change to indicate that you can resize the item.
You cannot remove an element, but you can hide it by clicking the handle in the lower right corner and dragging it to the top left corner.

- Click and drag an element to change the orientation of the PDF layout.

- Under the **Selected Element** heading, select the element, and if desired, determine the element’s width and height.

- Under the **Element Position** heading, determine the element’s position on the form. You can anchor/align the element to the top or bottom edge of the page with a fixed size (the size you
specify under the **Selected Element** heading), or align the element to the specified top position and stretch it to fit the height of the page (within the margins).

**Note:** You can place one element on top of another, but those elements will be stacked when the form is printed and results may not appear as expected.

- Under the **Element Label Style** heading, determine the font and style of the element’s label.  
  Check the **Use Label Coloring** box if you want the color set in the **Label Text Color** field to be used for the label of a form section.
  Check the **Split Horizontally** box if you want the text beside the label. Keep this box clear if you want the text beneath the label.
- Under the **Element Data Style** heading, choose the font and style of the text in the form section.
- Check the **Show Border** box if you want the section to have a border.

**Note:** To change the size of the area where the columns appear, choose **Columns** as the **Selected Element** in the editor and modify the width and height values as desired. To change the width of the individual columns, use the Printing Fields subtab on the transaction form. For more information, see **Configuring Printing Fields**.
8. Click **Preview** to see how this PDF layout will look when printed.

For any element that continues on an additional page, the bottom of the element must be high enough on the page to allow line items to print below it. For example, a logo element on a PDF layout cannot be set to be the whole page.

9. Click **Save**.

**Note:** When the PDF opens and you click the **Print** button, the Print window opens. In the **Page Handling** section of the Print window, verify that **Page Scaling** is set to **None**, and the boxes beneath it are not checked. Automatic scaling can cause difficulties with printing checks.

**Note:** If you have text overlapping a footer or missing from a printout, ensure that any long content is enclosed in an element that will split across pages.

See the following topics.

- Formatting Label Text
- Formatting Data Text
- Defining Custom Elements

### Defining Custom Elements

Custom elements are blocks of static or dynamic text that you can add to and position on PDF transaction layouts. You can define up to ten custom elements per layout. Images are not supported. Custom elements are text only.

Custom elements can only be defined for the following types of transaction form PDF layouts:

- packing slip
- picking ticket
- transaction

For example, you can follow these steps to add the Bill To element to a packing slip:

1. Go to Customization > Forms > Transaction Form PDF Layouts.
2. Click **Customize** next to your picking ticket form.
3. Click **Add Custom Element**.

The Add Element window opens.
4. In the **Element Name** field, enter a name for the element, such as **Bill To**.

5. In the **Choose Element(s)** list on the left side of the window, scroll down and click **Bill To**.

6. Click **OK**.

7. The **Bill To** box appears in the top left corner of the layout.

8. Click the **Bill To** element to drag it to the appropriate place on the form.

9. Complete additional changes to the element as necessary.

10. Click **Save**.

Now when you use this custom form, the form shows the Bill To address.

For each custom element, define the properties as described in the following table. After an element is added, click **Save & Edit**. The custom element is then added to the other tabs and can be modified like any other element on the form.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Element Name</strong></td>
<td>This identifies the custom element on the Border &amp; Placement, Label Text, and Data Text tabs, and may be printed in the label for the element field. This field is mandatory for each custom element where Enable is checked.</td>
</tr>
<tr>
<td><strong>Important:</strong></td>
<td>When a Label is created and the Custom Element is saved, the element is automatically available in the Borders and Placement tab with the coordinates defaulted to 0,0,0,0. To view the Label, you must edit these coordinates to display the label in the desired position.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Show Label</strong></td>
<td>If checked, the Label for this custom element is printed on transaction forms using this layout, using the same formatting as other element labels as defined on the Label Text tab (see Formatting Label Text). The element does not appear on the Label Text tab if Show Label is not checked.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Element Text</strong></td>
<td>Text entered here is displayed in the element field when printed. The text is formatted the same as other element text as defined on the Data Text tab (see Formatting Data Text). You can use NetSuite tags to dynamically retrieve and display text associated with a specific instance of a form field. To construct a tag enter the field name and enclose it in braces — {entity}. The fields available are the same body fields as those available in custom code. Refer to Custom Code Names for a complete list of available fields.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Note:</strong></td>
<td>These tags are similar to those used on HTML Transaction Layouts, using {} instead of &lt;&gt; and dropping NL from the beginning of the tag. For Example: &lt;NLENTITY&gt; in HTML Layouts would be {entity} in PDF Custom Elements.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Wrap Text</strong></td>
<td>If checked, the text in the element field wraps.</td>
</tr>
</tbody>
</table>

If you want to add personalized text, click **Add Custom Element** and instead of selecting an element, enter your text in the **Element Text** field and click **OK**.

**Configuring Borders and Placement**

You can add borders and alter the orientation of elements of your PDF forms.

**To add a border to a form:**

1. Go to Customization > Forms > Transaction Form PDF Layouts.

2. Click **Customize** or **Edit** next to the form you want to change.
3. In the Form Editor click the element you want to add a border to.
4. Under the **Additional Options** heading, check the **Show Border** box.
5. Click **Preview** to see how this PDF layout will look when printed.
6. Click **Save**.

To change an element's orientation in the PDF layout, click and drag it to where you want it on the form or use the fields under the **Element Position** heading. For more information, see Customizing Transaction Form PDF Layouts.

### Formatting Label Text

You can change the style and alignment of labels for PDF form elements.

If the **Element Label Style** or **Element Data Style** fields are not available, they are not applicable to the field selected.
**Customizing Transaction Form PDF Layouts**

**To format element labels:**

1. Go to Customization > Forms > Transaction Form PDF Layouts.
2. Click **Customize** or **Edit** next to the form you want to change.
3. In the Form Editor, click the element with the label you want to change.
4. Under the **Element Label Style** heading to the right of the preview, set the font style and alignment options for this label.
5. Check the **Use Label Coloring** box if you want the color set in the **Label Text Color** field to be used for the label of a form section.
6. Check the **Split Horizontally** box if you want the text beside the label. Clear the box if you want the text beneath the label.
7. To see how this PDF layout will look when printed, click **Preview**.
8. Click **Save**.

**Formatting Data Text**

You can change the style and alignment of the text in the sections of PDF forms.

**Note:** In transaction form PDF layouts, you cannot change the line spacing, nor can you change the space between label text and data text. For information about using advanced templates to change the spacing between label text and data text, see **Advanced PDF/HTML Templates**.

**To format element text:**

1. Go to Customization > Forms > Transaction Form PDF Layouts.
2. Click **Customize** or **Edit** next to the form you want to change.
3. In the Form Editor, click the element with the label you want to change.
4. Under the **Element Data Style** heading to the right of the preview, set the font style and alignment options for this label.
5. Click **Preview** to see how this PDF layout will look when printed.
6. Click **Save**.

**Using a Standard #10 Window Envelope With Transactions**

If you use window envelopes, you may need to adjust the printing position of your PDF transaction forms to show the correct address through the window.

If your account is set up to use metric measurement, convert the imperial values provided in the following procedure.

**To adjust your standard layouts to fit in a standard #10 window envelope:**

1. Go to Customization > Forms > Transaction Form PDF Layouts (or Transaction Form HTML Layouts).
2. Click **Customize** next to the kind of layout you want to change.
3. In the **Name** field, enter a name for your custom layout.
4. Click the **Elements** subtab.
5. Click **Border & Placement**.
6. In the **Border & Placement** section, locate the **Bill To** row.
7. In the **Left** column, enter 1.
   This moves the left edge of the field 1 inch from the left margin of the page.
8. In the **Top** column, enter 2.25.
   This moves the top edge of the field 2.25 inches from the top margin of the page.
9. In the **Right** column, enter 4.2.
   This moves the right edge of the field 4.2 inches from the left margin.
10. In the **Bottom** column, enter 3.5.
    This moves the bottom edge of the field 3.5 inches from the top of the page.
11. Locate the **Ship To** row.
12. In the **Left** field, enter 4.55.
    This moves the left edge of the field 4.55 inches from the left margin of the page.
13. In the **Top** column, enter 2.25.
    This moves the top edge of the field 2.25 inches from the top margin of the page.
14. In the **Right** column, enter 8.
    This moves the right edge of the field 8 inches from the left margin.
15. In the **Bottom** column, enter 3.5.
    This moves the bottom edge of the field 3.5 inches from the top of the page.
16. When you have finished, click **Submit**.
    You are returned to the Custom Layouts list.
17. Check the box in the **Preferred** column next to your custom layout.
    This ensures that your layout is applied to all forms of that type.
18. Click **Save**.

Repeat the steps above to adjust any transaction form layouts that you want to mail in standard #10 window envelopes. You can adjust any of your transaction form layouts for any size window envelope. Measure your envelope and the placement of the window to determine where to place the address.

When you have the measurements for your envelope and determined where to place the address field, you need to convert the fraction measurements into decimal measurements to adjust the layout. The following table is a quick reference guide to converting fraction measurements into decimal measurements.

<table>
<thead>
<tr>
<th>Fraction</th>
<th>Decimal</th>
<th>Fraction</th>
<th>Decimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/2</td>
<td>0.5</td>
<td>1/16</td>
<td>0.0625</td>
</tr>
<tr>
<td>1/4</td>
<td>0.25</td>
<td>3/16</td>
<td>0.1875</td>
</tr>
<tr>
<td>1/3</td>
<td>0.33</td>
<td>5/16</td>
<td>0.3125</td>
</tr>
<tr>
<td>3/4</td>
<td>0.75</td>
<td>7/16</td>
<td>0.4375</td>
</tr>
<tr>
<td>2/3</td>
<td>0.67</td>
<td>9/16</td>
<td>0.5625</td>
</tr>
</tbody>
</table>
Transaction Form HTML Layouts

Transaction form HTML layouts define the arrangement of fields on HTML transaction documents in NetSuite. Standard and Classic HTML layouts exist for all the standard form types other than shipping label and item label - these are assumed to be printed using PDF (see Customizing Transaction Form PDF Layouts). HTML layouts could be used to generate customized email versions of transactions forms.

Important: Basic layouts will be deprecated in a future release. We encourage you to use Advanced PDF/HTML Templates instead because new features are added exclusively to advanced printing. For information, see Advanced PDF/HTML Templates.

You can customize transaction form HTML layouts by editing the templates on which they are based. The templates for each layout consist of two blocks of HTML:

- **Style Block**: this block begins with `<STYLE>` and ends with `</STYLE>`. The style block defines the styles used by the layout.
- **Body Block**: this block begins with `<TABLE>` and ends with `</TABLE>`. The body block defines the actual content of the HTML page.

When you print using the Classic HTML Transaction Layout template, the header information, including the logo, is printed on every page of the HTML printout. To print the logo and other header information on the first page only, use the Standard HTML Transaction Layout template. You can also customize the classic template and remove the `<thead class="headrepeat">` and `</thead>` tags.

Totalling is handled in the same way as for Transaction Form PDF Layouts (see Totals Transaction Form Layouts).

Note: Advanced PDF/HTML templates provide an alternate model for customizing printed and emailed records. They support more customization capabilities than transaction form layouts. For details, see Advanced PDF/HTML Templates.

Creating Custom Transaction Form HTML Layouts

To customize a transaction form HTML layout:

1. Go to Customization > Forms > Transaction Form HTML Layouts.
2. Click Customize next to the layout you want to customize.
3. In the Label field, enter the name of your customized layout.
4. Check the Layout is Inactive box to remove this layout as an option in dropdown lists. You can always clear this box later and the layout is available again.
5. Check the Layout is Preferred box to make this layout your preferred layout for this type of transaction.
6. On the **Templates** subtab, edit the **Style** and **Body** blocks as desired.
   Embedded in the body block are NetSuite tags that correspond to individual content elements. The content elements correspond very closely to the layout elements in the existing PDF layouts. Basic customization consists of editing the body or style templates but leaving all of the NetSuite tags in the body block.
   For example, to create a collection layout you could add a block of text at the top of the body layout.

7. If desired, click **Elements** and customize the HTML that corresponds to each element.
   On the **Elements** subtab, each element is listed with the corresponding element ID, the HTML code that will be used to display the label that corresponds to the element, and the HTML code that will be used to display the data that corresponds to the element.
   When editing the HTML for elements, follow these guidelines:
   ■ Ensure that the HTML used correlates to any formats as defined in the body block. For example, if the element is included in a `<TABLE>` tag in the body block, the element itself should begin with a `<TR>`.
   ■ Elements always have data and may or may not have a label.
   ■ The actual label is represented by the tag `<NLLABEL>` tag which should be included in any customization of the label field.
   ■ The actual data is represented by the `<NLDATA>` tag which should be included in any customization of the data field.
   ■ There is also a `<NLATTRIBUTES>` tag which is used in the COLUMNS element to indicate where text formatting elements (such as alignment) are inserted.
   ■ The BODY, COLUMNS, and AGING elements are repeated to produce the HTML of the form.

**Totals Transaction Form Layouts**

Totaling fields on transaction forms display as follows:
■ Discount, shipping, and tax totals are displayed on separate lines of the total.
■ Each field is only included if it has a value.
■ The discount, shipping item, and tax names are used.
■ There are multiple tax lines in CA and VAT-enabled accounts.
■ The total area is automatically expanded to fit the necessary information.

**Amount Paid and Amount Remaining on Invoices**

Amount paid and amount remaining are optionally included on transaction forms. They display as follows:
■ The Amount Remaining field is a footer field instead of a body field for purposes of printing.
■ There is an additional Amount Paid footer field. This field is omitted if none of the invoice has been paid.
■ If these fields are shown in a customized form, they are displayed in the total section underneath the invoice total.
Custom Records

In NetSuite, you track all the information in your account using records. Administrators and users with the Custom Record Entries permission can create custom records that collect information specific to the needs of your business. You can attach information from custom records to entities, items, or transactions using custom fields.

For example, you may want to keep track of training courses your employees have taken. Since a record type specific for this purpose does not exist in NetSuite, create a custom record type titled Employee Courses. In the custom record type, create custom fields to store information that you want to be included on the record. You could have fields for the course names, class start date, class end date, course level, certificate achieved, certification expiry date, and so on.

You can also set up the custom record so that it appears as a subtab on another record. So, you could have an Employee Courses subtab on the employee record.

To enable the Custom Records feature, go to Setup > Company > Enable Features > SuiteCloud. Then, on the SuiteCloud subtab, check the Custom Records box.

To modify Custom Records, go to Customization > Lists, Records, & Fields > Record Types. You can do the following:

- Create a new custom record type
- Edit an existing custom record type
- View a list of record instances that have been created using each custom record type
- Create a new record instance for a selected custom record type, by clicking New Record
- Create a new search record for a selected custom record type

For information about adding a new record instance using the Lists menu, see the help topic NetSuite Record Types.

To make adding new records quicker, add shortcuts from the new instance page using the shortcuts menu (the star icon on the navigation menu). Or, add a shortcut to your shortcut portal. For more information, see Shortcuts Portal.

For more information, see the following:

- Creating Custom Record Types
- Online Custom Record Forms
- Parent-Child Record Relationships
- Sourcing with Custom Records
- Using Custom Record Entries

**Important:** SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should only access the NetSuite UI by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic SuiteScript 2.0 Custom Pages.

Creating Custom Record Types

To create a custom record type, perform the following steps:
1. Creating a New Custom Record Type.
2. Entering Name and Display Settings.
3. Specifying Permission and UI Settings.
5. Defining Search and Edit Settings.
6. To create a custom record type, click **Save**.

**Note:** To enable QuickViews for your custom record, use form customization for the custom record. For details, see Configuring QuickViews for Custom Records.

Before you save the custom record type, the following subtabs are displayed for you to further define your custom records:

- **Subtabs** – Create and arrange subtabs on your custom record type.
  
  For more information, see Adding Subtabs to a Custom Record.

  **Note:** To save time, create and arrange subtabs for your custom records before defining your custom fields.

- **Sublists** – Add search results as sublists on your custom record type.
  
  For more information, see Applying Custom Sublists to Custom Record Types.

- **Icon** – Select the PNG sprite you want to use to represent this record type in the New Bar, Create New menu, Recent Records menu, Recent Records portlet, and QuickViews. You can choose from built-in icons or create your own custom icon.
  
  For more information, see Choosing an Icon for a Custom Record.

- **Numbering** – Specify the numbering format for the custom record types.
  
  For more information, see Numbering Custom Record Types.

- **Permissions** – Choose the roles you want to access custom record entry forms, choose a default form, and restrict the forms available here.
  
  For more information, see Setting Permissions for a Custom Record Type and Applying Role-Based Restrictions to Custom Records.

  **Important:** For these permissions to apply, you must choose **Use Permission List** from the **Access Type** list.

- **Links** – Create links that take you to the list of record entries for this custom record type and choose where to place the links.
  
  For more information, see Creating Links to Custom Records.

- **Managers** – Define specific employees as managers of the current record type. This allows them to modify the custom record type. When defined as a manager, employees are automatically granted custom record view permission. This allows them to see the list of custom record types but not drill down on them.

  **Note:** If an employee has a role that includes the Custom Record Type permission, they have edit access to all custom record types. The Managers subtab enables you to grant permission for an employee to the current record type only.

- **Translation** – (when Multi-Language feature is enabled) Define translations for the custom record type name to be used when users change the language preference.
For more information, see Adding Translations for Custom Records.

After you save a custom record type, the following subtabs are added:

- **Fields** – Create and arrange the fields for your custom record type.
  For more information, see Adding Fields to Custom Record Types.
- **Forms** – Customize and select a preferred entry form for your custom record type.
  For information, see Adding Custom Forms for a Record.
- **Online Forms** – Create and manage online forms for your custom record types.
  For more information, see Adding Custom Online Forms for a Record.
- **Child Records** – If this record type is a parent record, its child records are listed here.
- **Parent Records** – If this record type is a child record, its parent records are listed here.
  For more information on parent and child records, see Creating Built-in Child Records and Using Child Records.

You can use SuiteCloud Development Framework (SDF) to manage custom record types as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom record type to another of your accounts. Each custom record type page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

Creating a New Custom Record Type

**To create a new custom record type**

1. Go to Customization > Lists, Records, & Fields > Record Types > New.

   **Note:** The custom record type is created after you enter information in all mandatory fields, and then click Save.

2. See Entering Name and Display Settings.

You can use SuiteCloud Development Framework (SDF) to manage custom record types as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom record type to another of your accounts. Each custom record type page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

Entering Name and Display Settings

After you create a new custom record type, you enter a name and description for the custom record type as well as display settings for custom record entries.

**To enter a name and display settings for the custom record type:**

1. In the **Name** field, enter a name for the record type.
   The maximum number of characters you can enter in the **Name** field is 300.

2. If you use custom code, enter a unique ID for this custom record in the **ID** field.
3. In the **Owner** field, select the owner of this custom record type.
   Only the owner can modify this record type.

4. In the **Description** field, enter a description of this record type.

5. To include a required **Name** field on each record entry, check the **Include Name Field** box.
   If you check this box, the **Name** field appears as the first field on the record and in the record list. If you do not check this box, your records are automatically assigned a number based on the order in which they are entered.

6. To display the record entry ID that is automatically assigned by NetSuite to each new record entry, check **Show ID**.

7. In the **Show Creation Date** field, to display the creation date and time on each record entry, check the **On Record** box. To display the creation date and time for each record entry in your list for this record type, check the **On List** box.

8. In the **Show Last Modified** field, to display the last modified date and time on each record entry, check the **On Record** box. To display the last modified date and time for each record entry in your list for this record type, check the **On List** box.

9. In the **Show Owner** field, choose an option:
   - To display the record owner on each record entry, check the **On Record** box. The record owner is the person who creates the record entry.
   - To display the record owner for each record entry in your list for this record type, check the **On List** box.
   - To allow the record entry owner to be changed, check the **Allow Change** box. If you allow the owner to be changed, you must also show the owner on the record entry. An Owner field is displayed on your record entries as a dropdown list of people with login access to your NetSuite account. The Owner field on each record entry defaults to the current person entering the record.

10. Enter permission and UI settings. See **Specifying Permission and UI Settings**.

**Specifying Permission and UI Settings**

After you have entered name and display settings for the custom record type, you enter permission and UI settings.

**To enter permission and UI settings:**

1. On the Custom Record Types page, choose a permissions mode from the **Access Type** list:
   - **Require Custom Record Entries Permission** indicates that only users logging in with a role with permission granted to the custom record type can access it.
   - **Use Permission List** grants access to the custom record type according to the permissions set up on the Permissions subtab of this page.
   - **No Permission Required** makes the custom record type publicly available.

   For more information, see **Setting Permissions for a Custom Record Type**.

2. To indicate that this record can only be accessed through SuiteScript, clear the **Allow UI Access** box. By default, this box is checked.

   When this box is not checked, users cannot access the custom record type from the NetSuite user interface. Also, the following custom record options are disabled: Allow Mobile Access, Allow Quick Search, Allow Quick Add, and Include in Search Menu.
Creating Custom Record Types

3. To indicate that this record should be accessible on mobile devices through the NetSuite iPhone application, check the **Allow Mobile Access** box.
   This box, which is disabled by default, is not available if the **Allow UI Access** box is cleared.

4. Enter file and child record settings. See Configuring File and Child Record Settings.

Setting Permissions for a Custom Record Type

To manage access to custom record type data, you can:

- **Define the Permissions Model** to use permissions on role records, use permissions defined on the custom record itself, or provide public access to the custom record type.
- **Prevent Access through the User Interface** so that users cannot access custom record type data through the NetSuite user interface.

Define the Permissions Model

You can use the **Access Type** list on a custom record type page to define a permissions model for a custom record type. This model can be based on: custom record entries permissions defined on role records, permissions defined on the Permissions subtab of a custom record type, or no permissions (meaning access to the custom record type is public).

**Note:** As of 2012.1, the Access Type list replaces the Use Permissions check box that was available in prior releases.

The **Access Type** list includes the following options:

- **Require Custom Record Entries Permission**
  □ This option is the default.
  □ Custom record types created prior to 2012.1 that did not have the Use Permissions box checked have this option set.
  □ This option indicates that only users logging in with a role with permission granted to the custom record type can access it. This permission can be set on the Lists subtab of the Permissions subtab on each Role page. See the help topic Customizing or Creating NetSuite Roles.
  Note that this limitation does not apply to the owner of the custom record type. The owner always has full access in any role.

- **Use Permission List**
  □ Custom record types created prior to 2012.1 that had the Use Permissions box checked have this option set.
  □ This option indicates the users logging in with a role with permissions defined on the Permissions subtab of the custom record type can access it. This permission can also be set on the Custom Records subtab of the Permissions subtab on each Role page.
  Note that this limitation does not apply to the owner of the custom record type. The owner always has full access in any role.
  □ For details about creating a permission list, see Setting Up a Permissions List for a Custom Record Type.

- **No Permission Required**
  □ This option indicates that it is considered public and all users can access this custom record type.
With this option, any user can modify the record if they get access to its entry form, which they could do through a URL, even if they do not have a link to the form.

You can use this option for records that must be accessible to scripts, but that you do not want users to access. After testing, clear the Allow UI Access box for the record. With this combination of settings, there are no restrictions on programmatic access to the record type, but no access through the user interface. See Prevent Access through the User Interface.

Prior to 2012.1, this option was not available for custom record types.

Prevent Access through the User Interface

You can clear the Allow UI Access box for a custom record type, to indicate that it can only be accessed programmatically. For example, this could be done through SuiteScript or SOAP web services. By default, the Allow UI Access box is checked.

When this box is cleared:

- Users cannot access the custom record type from the NetSuite user interface.
- If a user attempts to list, search, view, edit, or create a record of this type in the user interface, the following error message appears:

  ![Error Message](image)

  The following custom record options are locked as disabled: Allow Quick Search, Allow Quick Add, and Include in Search Menu.

Important: You need to take additional steps to control access to custom record data through searches. See Limiting Search Access to Custom Records.

Configuring File and Child Record Settings

After you have specified permission and UI settings, you enter file and child record settings.

To enter file and child record settings:

1. On the Custom Record Type page, to use a File Cabinet subtab to attach documents and images to your record entries, check the Allow Attachments box.
2. To add a Notes subtab to your child record entries, check the Show Notes box.
   
   Notes are added to this subtab automatically when any change is made to individual records.
3. To enable mail merge capabilities for records of this type, check the Enable Mail Merge box.
   
   For information about using mail merge, see the help topic Working with Mail Merge.
4. To be able to edit the order in which your child records appear on each parent record, check the Records are Ordered box.
   
   If you do not check this box, child records display in alphabetical order in both View and Edit modes.
If you check this box, in View mode, child records continue to display in alphabetical order. In Edit mode, child records initially display in the order in which they were entered and later, in the order in which you have set them through editing.

5. To hide child record sublist Remove links and prevent users from separating child records from the parent record, clear the Show Remove Link box.

By default, each child record in a sublist on a parent custom record includes a Remove link for users with Edit permission. Clicking this link results in the removal of the child record from the sublist but not from the system. This removal separates the child record from the parent record, in effect deleting the relationship between records, but this removal does not delete the child record from the system.

If Remove links are available, users can separate child records even if child record editing is not allowed.

6. To allow records of this type to be edited directly when they display as child records in a sublist on a parent record, check the Allow Child Record Editing box. Note the following:
   - When the Show Remove Link option is disabled, the Allow Child Record Editing box is not available.
   - Checking the box does not supersede users’ role-based permissions. Only users who have permission to edit the record type can edit child records when this option is enabled.
   - The box was formerly labeled Allow Inline Editing. Its label was changed because it is not related to the Inline Editing feature or to the Inline Editing toggle that displays on list pages when that feature is enabled. The name change also was intended to avoid confusion with the Enable Inline Editing option.
   - **Warning:** Child records are never editable in parent records that have more than 10,000 child record lines, even when the Allow Child Record Editing box is checked for the record type.

7. To allow users to delete records of this type when they are child records in a sublist on a parent record, check the Allow Delete box. The box is only available if you have checked the Allow Child Record Editing box. When the box is checked, the following occurs:
   - When the parent record is in Edit mode and users click the Remove button on the child record in the sublist, the entire child record type is deleted from NetSuite.
   - When the parent record is in View mode and users click the Remove button on the child record in the sublist, the child record is removed from the sublist and the child record type remains in the system.
   - **Warning:** Child records cannot be deleted in parent records that have more than 10,000 child record lines, even when the Allow Delete box is checked for the record type.

8. Enter search and edit settings for the custom record type. See Defining Search and Edit Settings.

Defining Search and Edit Settings

After you have specified child and record settings, you enter search and edit settings.

**To enter search and edit settings:**

1. On the Custom Record Type page, if you want to allow this record type to be searched using the Quick Search portlet on dashboards, check the Allow Quick Search box.
   - This option is not available if the Allow UI Access box is cleared.
2. If you want to allow this record type to be added using the Quick Add portlet on dashboards, check the **Allow Quick Add** box. For more information, see the help topic [Quick Add Portlet](#). This option is not available if the **Allow UI Access** box is cleared.

3. If you do not want system notes to be created for changes to this record type, clear the **Enable System Notes** box. By default, system notes are enabled.

4. If you do not want keywords entered in the global search box in the upper right corner of the page to apply to this record type, clear the **Include in Global Search** box.

5. If you do not want this record type to be available for searches in the UI, clear the **Include in Search Menu** box.

   If the **Allow UI Access** box is cleared, this option is disabled and cannot be changed.

6. Review the setting for the **Enable Optimistic Locking** option.

   Enabling this option causes the system to check for conflicting updates whenever a user or script attempts to save updates to a custom record entry. If another user or script has saved updates to the same custom record entry during the time that the first user or script was entering updates, an error appears. For more information, see [Enabling Optimistic Locking for Custom Records](#).

   **Note:** By default, this option is enabled for custom record types created in 2012.2 and later, and disabled for custom record types created prior to that release. You should enable this option.

7. If available, review the setting for the **Enable Inline Editing** option.

   - This option is only available if the Inline Editing feature has been enabled at Setup > Company > Setup Tasks > Enable Features, on the Company subtab, Data Management area.
   - This option is enabled by default.
   - When this option is enabled, an Inline Editing switch on list pages for this custom record type is set to **on**, and users can update record instances quickly by changing data directly in each record row. For more information, see the help topic [Using Inline Editing](#).
   - When this option is disabled, the Inline Editing switch is not available on list pages for this custom record type, and users must drill down from the list to each record entry to edit it.

8. To allow the translated display name of custom record instances to be included in saved searches, check the **Enable Name Translation** box.

9. To allow the definition of hierarchical relationships between records of this type, check **Hierarchy**. This hierarchy can be defined either on the parent record entry or on the child record entry. For more information, see [Defining Hierarchies among Custom Record Values](#).

10. Note that if you check the **Inactive** box, this record type no longer appears on the Record Types list unless you check the **Show Inactives** box at the top of the page. Also, you can no longer select this kind of record from any lists on entities, items, or transactions.

Before you save the custom record type, the following subtabs display for you to further define your custom records:

- **Subtabs** – Create and arrange subtabs on your custom record type. For more information, see [Adding Subtabs to a Custom Record](#).

  **Note:** To save time, create and arrange subtabs for your custom records before defining your custom fields.

- **Sublists** – Add search results as sublists on your custom record type.

  For more information, see [Applying Custom Sublists to Custom Record Types](#).
■ **Icon** – Select the PNG sprite you want to use to represent this record type in the New Bar, Create New menu, Recent Records menu, Recent Records portlet, and QuickViews. You can choose from built-in icons or create your own custom icon.

For more information, see Choosing an Icon for a Custom Record.

■ **Numbering** – Specify the numbering format for the custom record types. For more information, see Numbering Custom Record Types.

■ **Permissions** – Choose the roles you want to access custom record entry forms, choose a default form and restrict the forms available here. For information, see Setting Permissions for a Custom Record Type and Applying Role-Based Restrictions to Custom Records.

**Important:** For these permissions to apply, you must choose Use Permission List from the Access Type list.

■ **Links** – Create links that take you to the list of record entries for this custom record type and choose where to place the links. For more information, see Creating Links to Custom Records.

■ **Managers** – Define specific employees as managers of the current record type. This allows them to modify the custom record type. When defined as a manager, employees are automatically granted custom record view permission. This allows them to see the list of custom record types but not drill down on them.

**Note:** If an employee has a role that includes the Custom Record Type permission, they have edit access to all custom record types. The Managers subtab enables you to grant permission for an employee to the current record type only.

■ **Translation** – (when Multi-Language feature is enabled) Define translations for the custom record type name to be used when users change the language preference. For more information, see Adding Translations for Custom Records.

After you save a custom record type, the following subtabs are added:

■ **Fields** – Create and arrange the fields for your custom record type. For more information, see Adding Fields to Custom Record Types.

■ **Forms** – Customize and select a preferred entry form for your custom record type. For more information, see Adding Custom Forms for a Record.

■ **Online Forms** – Create and manage online forms for your custom record types. For more information, see Adding Custom Online Forms for a Record.

■ **Child Records** – If this record type is a parent record, its child records are listed here.

■ **Parent Records** – If this record type is a child record, its parent records are listed here.

For information on parent and child records, see Parent-Child Record Relationships, Creating Built-in Child Records, and Using Child Records.

### Defining Hierarchies among Custom Record Values

A custom record type can be set up to allow users to define hierarchies among values for that custom record type. Note that this hierarchy is not for the relationships between different custom record types, but rather for the relationships between instances of the same custom record type.

For example, a custom record type has been created to store employee information. You can set up hierarchies between instances of this record type to indicate supervisory relationships.
First the custom record type must be set up with the Hierarchy option enabled.

When the hierarchy functionality is enabled on the custom record type definition page, you can define a parent-child hierarchy between two record instances of the same type. This hierarchy can be defined either on the parent record or on the child records.

On a parent record instance, you can click **New Custom Employee** on the Child Records subtab to create a child record.

On a child record instance, you can select the parent record from the list, or create a new record instance to be the parent.
The Custom Record List page shows the hierarchy for the records in the format <parent record>::<child record>.

**Important:** Although this example illustrates only two levels of hierarchy, this feature supports multiple levels of hierarchy among custom record instances.

**Enabling Optimistic Locking for Custom Records**

Each custom record type has a Enable Optimistic Locking option that can be enabled to protect custom record data integrity.
Enabling this option causes the system to check for conflicting updates whenever a user or script attempts to save updates to an instance of this custom record type. If another user or script has saved updates to the same custom record instance during the time that the first user or script was entering updates, the following message is returned:

"Unable to save record. Record was changed by a different user. Please reload and try again."

The Enable Optimistic Locking option is enabled by default for all custom record types created as of 2012.2 and later. For backwards compatibility, this option is disabled by default for custom record types created prior to 2012.2. You should enable this option, but first review any scripts that may be affected by this change and edit them as needed.

This support for optimistic locking makes custom records' concurrency control consistent with the optimistic locking used generally for NetSuite standard records. Optimistic locking assumes that multiple concurrent transactions can usually complete without affecting each other, so data resources do not have to be locked during the time that transactions are in process. Instead, a check for conflicts occurs before each transaction is committed. For more information about optimistic locking, you can review a related article at http://en.wikipedia.org/wiki/Optimistic_concurrency_control.

Adding Fields to Custom Record Types

The Fields subtab lets you add fields to your record. You can add as many fields as necessary to your custom record. If any custom segments have been applied to this custom record type, they are listed on the Fields subtab. Custom segments have an ID that begins with cseg.

Important: This tab is not available until the record type has been saved.

To add fields to your custom record:

1. Click New Field to create a new field for this record type.
   For step-by-step instructions for creating a new field, refer to Creating a Custom Field.
2. Rearrange the fields as desired.
   Select a line and drag and drop it to the desired position, or click Move To Top or Move To Bottom. If you have placed the fields on subtabs, moving a field here changes its position in relation only to the other fields on the same subtab.
3. To edit a field, click the name of the field in the **Description** column.

4. Click **Save**.

   Before saving, NetSuite validates any parent-child combinations to ensure that the values are unique. If the same parent-child combination already exists, an error message appears showing the existing entry, and you cannot save the duplicate.

**Note:** If you want this field to be available for data entry in the Quick Add portlet, check the **Allow Quick Add** box. By default, this option is not enabled. See the help topic *Quick Add Portlet*.

---

### Limiting Search Access to Custom Records

The restrictions to custom record type access set up on the Permissions subtab do not apply to custom record data access by searches. If you want to prevent searches from returning custom record type data, you have the option of restricting search access to specific custom record type fields on a field by field basis, and restricting the audience for custom record type saved searches on a search by search basis.

**To limit search and reporting access to a custom field:**

1. Go to Customization > Lists, Records, & Fields > Record Types.
2. Click a custom record type.
3. On the **Fields** subtab for the custom record, click a field.
4. On the **Access** subtab for the field, set the **Default Level for Search/Reporting** option to **None** to prevent any searches or reports from returning data for this field.

   You also can prevent searches or reports run by users with specific roles from returning data for this field by setting the **Default Level for Search/Reporting** option to **None** for specific roles on the **Role** subtab.

   For more information, see *Restricting Access to Custom Fields*.

**To limit access for a custom record saved search:**

1. Go to Lists > Search > Saved Searches.
2. Click the saved search that you want to restrict.
3. On the search page, click the **Audience** subtab and make changes to the users who have access as desired.

   For example, you can clear the **Public** box or the **Select All** box for roles, and instead select only the specific roles that you want to have access to the custom record.

   For more information, see the help topic *Defining Audiences for Saved Searches*.

---

### Applying Role-Based Restrictions to Custom Records

On a role record, you can restrict the access of users with that role to standard records, based on these records’ values for department, class, location, employee, and subsidiary (OneWorld) fields. For example, you could set an employee-based restriction for the Sales Manager role so that those users only see records where they or their subordinates are the sales rep.

For details about setting these restrictions, see the following topics.
Creating Custom Record Types

- Set Employee Restrictions
- Set Department, Class, and Location Restrictions
- Restrict Role Access to Subsidiaries (OneWorld Only)
- Customizing or Creating NetSuite Roles

You can apply the restrictions set on role records for a particular category (department, class, location, employee, or subsidiary) to a custom record, by checking the Apply Role Restrictions box for a list/record custom field that stores values in one of these categories. For example, if you check this box for an Employee list/record custom field, the employee-based restriction set on the Sales Manager role record is applied to this custom record, so that those users only see custom records where they or their subordinates are the value for the custom field.

To apply role-based access restrictions to a custom record:

If the class, department, location, or subsidiary field does not yet exist:

1. On the Fields subtab of a custom record definition page, click New Field.
2. On the new field definition page:
   a. Enter a name for the field.
   b. From the Type list, select List/Record.
   c. From the List/Record list, select Class, Department, Location, Employee, or Subsidiary.
   d. Check the Apply Role Restrictions box.
   e. Complete other settings for the custom field as desired, and click Save. For more information, see Adding Fields to Custom Record Types.

If the field already exists:

1. On the Fields subtab of a custom record definition page, click the field name.
2. On the field definition page, check the Apply Role Restrictions box and click Save.

Adding Subtabs to a Custom Record

The Subtabs subtab of a custom record form lets you add custom subtabs to your record to better organize fields. To save time when creating custom fields, create any desired custom subtabs first.

To add subtabs:

1. In the Title column of the Subtabs subtab, enter the name of your new subtab for this record type.
2. If desired, designate this subtab as a child of an existing subtab.
   In the Parent column, select an existing subtab from the list. This list consists of any custom
   subtabs already saved for the current custom record type as well as any custom subtabs that you
   have defined for that record type.

   **Note:** Since the parent field is populated with subtabs created for the current custom
   record type, you can only define the subtab as a child if the custom record has already been
   saved with predefined subtabs.

3. Click Add.
4. Repeat these steps for each subtab you want to add.
5. Click Save.

   **Important:** You must save the record after you have created subtabs in order for the
   subtabs to be available to place fields on. You can place fields on your new subtabs by
   editing the field record from the Fields subtab.

6. Rearrange the fields as desired (if you have not yet created fields for your custom record, go on to
   defining fields. You can assign a subtab when you create custom fields.
   Select a line and drag and drop it to the desired position, or use the Move buttons.
7. To edit the name of a subtab, click the line for that subtab.
8. Click Save.

**Choosing an Icon for a Custom Record**

You can choose an icon to represent a custom record in the NetSuite user interface. Users can quickly
identify the record type through this visual cue. These icons are used in the following places:
■ Create New menu on records and New column on list pages and list portlets
■ Recent Records menu
■ Recent Records portlet
■ QuickViews

You can choose from 70 prebuilt icons or create your own custom icon.

The following image shows the prebuilt icons from which you can choose:

To associate an icon with a custom record:

1. On the Icon subtab of the custom record, choose the Use Built-In Icon or the Use Custom Icon option.
2. If you choose Use Built-In Icon, you can select an icon from the list.
   You can click the picker to view available icons and click the one you want to use.
3. If you choose Use Custom Icon, select its file. For requirements for custom icons and instructions for creating them, see Creating Icons for Custom Records.
4. Click Save.

Creating Icons for Custom Records

You can create custom icons to represent custom records in the NetSuite user interface.

To use a custom record icon:

1. Create an image file for your icon.
2. Open the custom record for which you want to add the icon.
3. On the Icons subtab, choose the Use Custom Icon option.
4. Click New next to the File field.
5. In the File dialog, choose your icon image.
6. Click **Save** in the File dialog, then save the custom record.

**Important:** Custom icons must meet the specific requirements detailed below. You should read these requirements thoroughly and give yourself enough time to test your icon. You will need at least a basic understanding of an image-editing application like Adobe Photoshop or another application that enables you to edit artwork and save the icons as transparent .png files.

### The Four Icon Versions

A custom record icon is made up of a set of four slightly different versions of your icon, ranging from grayscale to full-color. These four versions are required so that the icon can be displayed with maximum clarity and contrast on a variety of different backgrounds.

**Image 1:** Grayscale icon for dark backgrounds. This icon is an “outline” version of the full-color icon. It is used in the dashboard’s New Bar when a dark color theme is used. Since the default color theme is dark-colored, this is the version that is most often seen.

**Image 2:** Grayscale icon for light backgrounds. This is another outline version of the icon, used in the dashboard’s New Bar when a light color theme is used. This version is not used as often since few color themes have a light background color for the New Bar.

**Image 3:** Color icon for dark backgrounds. This is a full-color version of the icon, optimized for use on dark backgrounds. It is used in the New Bar as the color icon that replaces the grayscale version when a cursor is nearby.

**Image 4:** Color icon for light backgrounds. This is another full-color version of the icon, optimized for use on light backgrounds. It is used in menus, which always have a light background — such as the Create New menu and the Recent Records menu.
Creating Custom Record Types

The Icon Sprite Image File

The four icon versions reside, side-by-side, in a single image file known as a sprite. Using coordinates and other data stored in CSS, NetSuite displays the proper icon version needed and crops out and hides the rest of the sprite image. Therefore, in order for your icons to display properly, they must be an exact size and at an exact location within the image file.

Each icon must be no larger than 16 pixels by 16 pixels. Any artwork that goes beyond the 16x16 boundary will not be displayed.

![Icon Sprite Diagram]

Each icon version must be located at the following locations within the image file, as measured from the upper-left corner of the image:

<table>
<thead>
<tr>
<th>Icon Version</th>
<th>x Coordinate</th>
<th>y Coordinate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grayscale icon for dark backgrounds</td>
<td>25</td>
<td>25</td>
</tr>
<tr>
<td>Grayscale icon for light backgrounds</td>
<td>50</td>
<td>25</td>
</tr>
<tr>
<td>Color icon for dark backgrounds</td>
<td>75</td>
<td>25</td>
</tr>
<tr>
<td>Color icon for light backgrounds</td>
<td>100</td>
<td>25</td>
</tr>
</tbody>
</table>

Most image-editing applications have guides that you can set up to help you keep track of these spacing requirements. You can also download the sample icon file pack at the bottom of this topic, which contains different template files that will help keep your icons in order.

File Format

Save the icon image file as a 24-bit .png file with an 8-bit alpha channel for the transparent background. Your image editing application may refer to this file format as PNG-24. NetSuite strongly recommends that you always use the PNG-24 format when saving your icon image file. Using .gif or .jpeg formats is discouraged due to their limited (or lack of) transparent backgrounds. Other image file formats are not accepted.
Constructing Your Icon File, Step-by-Step

1. When you have your icon, whether you created it yourself or obtained it from a royalty-free source, the first step in creating your final icon image file is to create an outline of it. This outline drawing will be used to construct the two grayscale icon versions.

The outline version of your icon should be plain, but it should also have sufficient detail so that it can be recognized when it appears on your dashboard's New Bar. Start by using a 1-pixel pencil tool in your image editing application, and trace an outline of your icon. In addition to the outline, trace some of your icon's internal components so that it will be easier to recognize.

2. Take the outline tracing of your icon and follow these directions to create your grayscale icon for dark backgrounds:

3. Take the outline tracing of your icon and follow these directions to create your grayscale icon for light backgrounds:
4. After the two grayscale versions are complete, create the color versions. This involves placing the color icon that you already have on a variety of background colors to check its appearance. Try the following colors:
   - #5A759C - default color for the New Bar
   - #F1F4F9 - default color for the Recent Records menu
   - #EBECEF - default color for the Create New menu

   Try other colors, such as light and dark reds and then test the colors of your NetSuite color theme. After you see how your color icon looks on different background colors, you can create the last version of your icon.

   If you feel that your color icon looks fine regardless of background color, make a copy of it to be placed in the fourth slot of your .png image.

5. Place the four versions of your icon in a single .png image, using the proper spacing described above. Save your icon image file as a PNG-24 file, and upload it to your NetSuite account.

Sample Icons

Click here to download our sample set of custom icons (.zip archive, 14 KB), designed to help you get started.

Numbering Custom Record Types

You can have numbers automatically assigned to your custom record instances for easier tracking and designation.

**Warning:** After you enable auto-numbering, numbers cannot be removed from records. Disabling auto-numbering in the future only prevents future records from being numbered.

**To assign automatic numbers to a custom record type:**

1. Go to Customization > Lists, Records, & Fields > Record Types.
2. Click **Edit** next to the record type you want to have auto-numbered.
3. Click the **Numbering** subtab.
4. Check the **Enable** box to turn on automatic numbering.
5. In the **Prefix** field, enter any numbers or letters you want added before each automatically-generated number.
6. In the **Suffix** field, enter any numbers or letters you want added after each automatically-generated number.
7. In the **Minimum Digits** field, enter the number or total digits you want as the minimum for auto-generated numbers.
   
   This adds placeholder zeros to numbers that do not have the number of minimum digits you set.
   
   For example, enter 4 to have the first number added as 0001. Valid values for this field range from 0–20.
8. In the **Initial Number** field, enter the number you want to use to start automatic numbering.
   
   For example, if you enter 20, the first record instance created is numbered 20, and no record instances are numbered below 20. The next record instance numbered would be 21.
9. Check the **Allow Override** box if you want to be able to enter a custom number when you edit the record instance.
10. Check the **Update** box to number record instances that have already been created.
   If you have already enabled automatic numbering, checking this box does not change any existing numbers, as those numbers have already been referenced in the system.

11. Click **Save**.

**Important:** Be aware that, as of 2010.2, handling of the Name criteria for custom records searches has been modified. These changes particularly affect auto-numbered custom record types. Previously, the value specified for the Name criteria was compared to the Name field values of custom records, to IDs containing prefixes, and to IDs with prefixes. Now, the Name criteria value is compared only to Name field values, providing a more clearly delineated set of results. To search for ID values, users can use the ID criteria.

### Adding Custom Forms for a Record

The **Forms** subtab lets you create custom entry forms for your records. You can create an unlimited number of entry forms for your record type. After you have created custom entry forms, you can select which form is the preferred entry form.

**Important:** This tab is not available until the record type has been saved.

By default, at least one form is automatically assigned to the custom record type. This custom record form can be customized as desired.

**To create custom forms for the record:**

1. In the **Forms** subtab, click **Customize** or **Edit** next to the entry form you want to customize.
   Selecting **Customize** enables you to create a new custom form based on the selected one. Click **Edit** to customize the selected form.
2. Customize the form as desired and then save it.
   For step-by-step instructions on how to customize a form, see *Creating Custom Entry and Transaction Forms*.
3. In the **Preferred** column, check the entry form that you want to be set as the preferred form for this record type.
   **Note:** If you have also set preferred forms on the **Permissions** subtab, the preferred form set on the **Permissions** subtab takes precedence over the role set on the **Forms** subtab.

For example, you set Custom Form A as the preferred form on the **Forms** subtab. On the **Permissions** subtab, you set the default form for the Sales Rep role to Custom Form B. When a sales rep creates a new record, Custom Form B is selected by default.

4. Click **Save**.

### Online Custom Record Forms

An online custom record form is used to receive information from customers on your website. Information received from online forms automatically creates or updates records in your NetSuite account.

For example, you might link to a warranty custom record form from your website. You require your customers to include their name, the end date of the warranty, the item purchased, serial number, the start date and the type of warranty.
You can customize the appearance of online forms as well as the information you require from anyone who submits these forms. When creating an Online Form, start with either a default template or your own HTML template and then modify as needed for the current form.

- **Default NetSuite Template**: provides the ability to customize page messages, field labels and properties, etc.
- **HTML Template**: provides the ability to customize HTML templates that have already been created.

For more information, see the following:

- Adding Custom Online Forms for a Record
- Creating Online Custom Record Forms
- Linking Online Custom Record Forms to My Website
- Creating HTML Templates for Online Custom Record Forms

### Adding Custom Online Forms for a Record

The Online Forms subtab lets you create online entry forms to capture information and create new records from outside of your NetSuite account.

**Important:** This subtab is not available until the record type has been saved.

To add custom online forms:

On the **Online Forms** subtab, do one of the following:

- Click **New Online Form** to create an online form for this record type.
- Click **New Online HTML Form** to create an online form based on an HTML template.
- To edit an existing online form, click the name of the form.

### Creating Online Custom Record Forms

#### To create an online custom record form:

1. Go to Customization > Lists, Records, & Fields > Record Types.
2. In the **Edit** column, click the name of the record you want to create an online form for.
3. In the **Online Forms** subtab, click **New Online Form** or **New Online HTML Form**.
   - Selecting **New Online Form** enables you to create a form based on the NetSuite default template.
   - Selecting **New Online HTML Form** lets you create a form based on a custom HTML template that you have created. You must first create the custom HTML template before you can use it for an online form.
   - The steps involved for creating an Online Form versus an Online HTML Form are the same with a couple of exceptions as noted in the following procedure.

4. Enter a title for this form.
   - This title is displayed at the top of the form.

5. If creating an HTML template (these options not available for Default templates):
   - In the **Template** field, select the HTML template you want to use for this form.
   - You can create online HTML form templates at Lists > Marketing > Marketing Templates.
To have NetSuite insert labels for your form fields, check the Include Field Labels box. If you leave this box unchecked, you must include field labels in your HTML template file.

6. If creating a Default template (these options not available for HTML templates):
   - In the Message field, enter a message that your customers will see at the top of the form. This message can include up to 500 HTML characters.

   **Note:** The message can be formatted using built-in rich text formatting tools. If you prefer to edit the message directly with HTML tags, click HTML Source Code.

   - In the Detail Message field, enter a message you would like to display at the bottom of the form. This message can include up to 4000 HTML characters. Again, you can use the built-in rich text formatting tools or view as HTML only.

7. Check the Enable Online box if you want to link to this form from a website.

8. If desired, check the Form is Inactive box. You can always reactivate it at a later date.

9. In the Select Fields subtab, edit any existing fields or add new fields as desired. Rearrange the order of fields as desired.

Click on a line and drag and drop it to the desired position or click Move Up, Move Down, Move to Top or Move to Bottom.

1. In the Set Up Workflow subtab:
   - To receive notification when this form is submitted, enter the addresses you want email to be sent to in the Notify by EMail field.
   - To specify a page for customers to be sent to after they submit the form, enter the URL for that page in the Redirect to URL field. By default, the user who submits the form is redirected the home page of your primary website.
   - In the Handle Duplicate Records field, select how you want NetSuite to handle records that are duplicates of existing records.

2. Click the Set Up Appearance subtab.
   - In the Number of Columns Shown field, select the number of columns for the form.
   - In the Color Theme field, select a color theme for the form.
   - In the Font field, select a font for the form.
   - Check the Default Browser Style box to make the form's appearance match the customer's browser.
   - Check the Unlayered Sections box to expand any subtabs on the form.
   - In the Button Alignment field, select where to place the buttons on the form.
   - In the Form Logo field, select a logo to place at the top of the form.
     You can upload new logos at Lists > Web Site > Images.

As you define the appearance options for your form, keep in mind that your end users use various sized browsers and types. For example, if you are designing on a high-resolution large monitor, several columns may display fine, but for smaller low-resolution monitors it may be better to design your layout with fewer columns. Also, if customers may need to print the form, ensure that the colors you select will properly display for both color and black and white printers.

3. In the Custom Code subtab:
The Custom Code tab is only available if you have a SuiteScript feature enabled. For detailed information on using custom JavaScript files to perform functions, refer to SuiteScript 1.0.

- In the Script File field, select the JavaScript file that contains the desired scripts for this form.

  **Important:** You must upload your file to the File Cabinet before you can select it.

- In the Page Init Function field, enter the script name to be called from your script file when this entry form is first loaded.
- In the Save Record Function field, enter the script name to be called from your script file when this record is saved.
- In the Validate Field Function field, enter the script name to be called from your script file when a field on this entry form is changed.
- In the Field Changed Function field, enter the script name to be called from your script file when a change made to a field is accepted.

4. Click Save.
5. If desired, preview your new form.
   Select the form from the Online Form subtab and then click Preview Form.
6. Add links to your online form to your website.
   For details on how link to your online form see Creating Links to Custom Records.

Now, customers can enter information on your website, and records are automatically created or updated in NetSuite.

### Linking Online Custom Record Forms to My Website

You can link to an online custom record form from your website.

If you have a NetSuite website, you can create a link to an online form in one of your information items, category descriptions and from any other HTML description field.

**To link to an online custom form from your NetSuite website:**

1. Click the Setup tab.
2. On the Setup page, under the Customization heading, click Record Types.
   The Custom Record Types list opens.
3. In the Edit column, click the name of the record type you want to edit.
4. Click the Online Forms subtab.
5. Click the name of the form you want to link to.
6. On the Online Custom Record Form page, click the External subtab.
7. Copy the URL from the Publishable Form URL field.
   You can highlight the URL with your mouse, right-click, and then click Copy.
8. Click the Lists tab.
9. On the Lists page under the Web Store heading, click the kind of item you want to paste the link into.
10. On the list, click Edit next to the information item or category you want to link to your online custom record form.
11. Enter or paste the link in the description field you want the link to appear in.
For example, the account administrator of Wolfe Electronics wants to include a line in a Detailed Description field that says, "Click here to register for your warranty." The word "here" links to the custom record form.

The HTML markup source entered would be:

```html
<p>Click <a href='the Online Custom Record Form's URL'>here</a> to register for your warranty.</p>
```

12. Click Save.

Now your customers can follow the link to your online custom record form on your website. After this form is submitted, a record is created with the customer's information.

You can also link to your online custom record form from an external website or from an email message. To do this, copy and paste the form's URL into a link in your HTML document.

For more information about entering HTML in your website, read the help topic Using HTML in Description Fields.

For more information about online custom record forms, read Creating Online Custom Record Forms.

### Creating HTML Templates for Online Custom Record Forms

To use an HTML template when creating your online forms, you must first create the template and store it within NetSuite. You can create your HTML templates locally on your own machine and then upload them to the NetSuite File Cabinet or you can use the built-in NetSuite template creator. See:

- Creating an HTML Template Locally
- Uploading an HTML Template
- Creating an HTML Form Template

**Note:** Online HTML form templates are especially useful if you do not use a NetSuite website, but you use NetSuite online forms.

### Creating an HTML Template Locally

When creating an online form template on your local machine, you can define how the fields are arranged, which fields to include on the form and the style of the page. Use standard HTML code to create the template as you would for any other HTML form and include the following elements:

- `<NLFORM>` and `</form>` tags to define the beginning and end of the form.
- Tags for each NetSuite field included on the form. For details, see Using NetSuite Field Tags.
- An input tag that defines the button your customers use to submit the form.

```html
<input type="submit" value="Button Text">
```

You can substitute text in the button by changing the value in the code. For example, if you want the text in the button to read Submit Form, the code would be:

```html
<input type="submit" value="Submit Form">
```

For example, the following HTML code is a representation of an acceptable form template:

```html
<html>
```
Using NetSuite Field Tags

Field tags in an HTML form are defined as <NLTAG>, where TAG is the ID of the field. Each field in NetSuite has a unique ID and therefore a unique tag definition.

Note: Because field IDs are incorporated into tag definitions for fields, when creating custom fields that will be used in your HTML templates, it is useful if you specify IDs for each field and use consistent naming conventions that make sense in your business environment. If the default NetSuite IDs are accepted when creating your custom fields, the tags may not make sense in your HTML code making it more difficult to know exactly what the field references.

If you have chosen to include a mandatory Name field on your custom record, you must include the tag, <NLNAME>, for that field in your template.

To determine the tags to use for each field on your custom record:

1. Click Customization > List, Records, & Fields > Record Types.
2. In the Edit column, click the name of the record type you want to create a template for.
3. On the Fields subtab, click the name of the field you want to place in your template.
4. In the Address bar of your browser, the URL for this page is displayed. At the end of the URL is the ID for the selected field.

Uploading an HTML Template

After you create your own HTML template, you must upload the template to NetSuite to make it available when creating online HTML forms.

Note: HTML templates are saved by default in the Marketing Templates folder in your File Cabinet. To change the folder you use to store your templates, an account administrator or sales administrator can go to Setup > Sales > Preferences > Sales Preferences.

To upload an HTML template:

1. Click Documents > Files > File Cabinet.
2. In the File Cabinet, browse to Templates > Marketing Templates.
3. In the File field, click Browse.
4. In the Choose File window locate your HTML template on your hard drive.
5. Select the file.
6. Click Open.
7. On the Folder Contents page, click Add This File.

Creating an HTML Form Template

HTML form templates can be creating by selecting an existing template and setting additional properties for it or by designing a new HTML template in NetSuite.

To create an HTML form template record:
2. On the Select Type page, click Online Form.
3. In the Name field, enter a name for the template.
4. If desired, enter a title for the template.
   If no title is provided, this field sources the name field.
5. In the Description field, enter any additional information for this template.
   This is information that describes the template — it is not a component of the displayed form. It may be useful to describe where and how this template should be used or describe any unusual fields or field relationships.
6. To base the current template record on an existing template, in the Template File field, select the template.
   You can select a template in a number of ways:
   - Enter the first few letters of the template name in the Template File field and then press the Tab key to automatically select a matching template — if multiple templates match the text, a list is displayed in a pop-up where you can select the desired template.
   - Click the List icon to display a pop-up where you can select the desired template.
   - Click the New icon to upload an HTML template from your hard drive. See Creating an HTML Template Locally for information on creating your own HTML templates.
   After a template is selected, you can view detailed information about the template by clicking the Open icon.
7. To design a new template, enter your text and template tags in the Template field.
   - To add NetSuite fields into your HTML template, select the desired field from the Insert Field list.
   - To view the HTML code, click HTML Source Code.
     If you have selected a template in step 6, skip this step. You do not need to enter template tags since the file selected in step 6 contains all of this information.
8. If desired, limit access to this template by selecting a group from the Restrict to Group field.
9. Click Save Now. When you create an HTML online form, you can select this template.

Setting Up a Permissions List for a Custom Record Type

You can manage access to a custom record type's data by setting up a permissions list on the Permissions subtab of the record type page.
The role-based restrictions you set on this subtab are also available on the record for each role. Changes made on role records related to this custom record’s permissions are reflected here.

**Important:** For the permission settings in the Permissions subtab to take affect, the Use Permissions option must be selected for Access Type. Be aware that these permission settings are not used to restrict search access to custom record data. You can limit searches’ access to custom record data on a per-field or per-search basis. See Limiting Search Access to Custom Records.

To set up a permissions list for a record type:

1. On a custom record type page, click the **Permissions** subtab.

   **Note:** If the custom record type is associated with a custom segment, the Permissions subtab is not available. The permissions must be set on the custom segment configuration page.

2. In the **Role** column, select the role you want to have access to custom record entries of this type.

3. In the **Level** column, select a level of access for this role. Available options include:
   - **None**: People with this role cannot use custom records of this type at all.
   - **View**: People with this role can view custom records of this type.
   - **Create**: People with this role can view and create custom records of this type.
   - **Edit**: People with this role can view, create and edit custom records of this type.
   - **Full**: People with this role can view, create, edit and delete records of this type.

4. Select a value in the **Restrict** column to limit the access of users with the selected role to custom records of this type:
   - Select **Viewing and Editing** to restrict users with this role to viewing or editing only the records of this type that they or their subordinates created.
   - Select **Editing Only** to restrict users with this role to editing only the records of this type that they or their subordinates created. They can view all records of this type.
   - Leave this column blank to allow users with this role to view and edit all records of this type.

5. In the **Default Form** column, select a default entry form for this role to use when entering records of this type.

   **Note:** The default form you set here for a role takes precedence over the preferred form setting on the **Forms** subtab.

   For example, you set Custom Record Form A as the preferred form on the **Forms** subtab. On the Permissions subtab, you set the default form for the Sales Rep role to Custom Record Form B. When a sales rep creates a new record, Custom Form Record B is selected by default.

6. Check the box in the **Restrict Form** column to make the default form the only entry form available to this role.

7. From the **Search Form** list, select a custom search form to be used for searches of this record type, if one is available.

8. From the **Search Results** list, select a custom search to be used to limit results for searches of this record type, if one is available.

9. From the **List View**, **Dashboard View**, and **Sublist View** lists, select a custom search to be used to limit displayed results for these lists, if one is available. Check a box in one of the **Restricted** columns to make the selected view the only one available to this role.
10. Click **Add**.
11. Repeat these steps for each role you want to give access to.
12. Click **Save**.

Creating Links to Custom Records

The Links subtab lets you create links throughout your account to access your custom records. When determining where to place links for the records, it is important to consider all roles that will be using the record type and how they will be using it to determine the most logical place for the links.

**Important:** Even after you create a link to a custom record type, this link does not display for users that do not have permission to access that custom record type.

To add a link to a custom record:

1. In the **Center** column of the **Links** subtab, select the center you want the link to be visible in.
   If desired, you can add links to this custom record in each center.
2. In the **Section** column, select the tab you want the link to display on.
   The tabs available vary depending on the selected center. If a center has **not** been selected, no tabs are listed.
3. In the **Category** column, select a standard, built-in NetSuite category, or select a custom category you have already created. (See **Creating Center Categories** for information on creating custom categories.)
4. In the **Label** column, enter a name for this link.
   If you do not enter a label, the label name defaults to the name of the custom record type.
5. (Optional) In the **Translation** column, type the translated name for the link.
6. (Optional) In the **Insert Before** column, choose where you want to place the link. If you do not provide a value in the **Insert Before** column, the link will appear below the last link in your chosen category.
7. Click **Add**.
8. Repeat these steps for each link you want to add.
   If you want to edit a link, click the line for that link.
9. Click **Save**.

Adding Translations for Custom Records

If the Multi-Language feature is enabled in your account, you can translate the name of a custom record, its custom subtab titles, and its custom sublist labels, so that they match the language of the NetSuite user interface. You can also translate the names of instances of a custom record. For details, see the following:

- Translating a Custom Record Name
- Translating Custom Record Subtab Titles
- Translating Custom Record Sublist Labels
- Translating Custom Record Instance Names
Important: Before you can add these translations, you need to select translation languages at Setup > Company > General Preferences, on the Languages subtab. This subtab lists both system-supported languages that can be used for the NetSuite user interface (and are available at Home > Set Preferences), and additional languages that can be used for website translations only (and are not available at Home > Set Preferences). You should only enter translations for system-supported languages, because these are the only languages that can be displayed in the user interface. For details, see the help topic Configuring Multiple Languages.

Translating a Custom Record Name

You can define translations for a custom record type name on the Translation subtab of the custom record page:

![Custom Record Type](image)

The maximum number of characters you can enter in the Name field is 300.

Note: Translated names that are available on a menu are truncated to 128 characters.

Translating Custom Record Subtab Titles

You can define translations for a custom record type's subtab titles, on the Subtabs subtab of the custom record page:

![Subtabs](image)
For details about custom record subtabs, see Adding Subtabs to a Custom Record.

**Translating Custom Record Sublist Labels**

You can define translations for custom record type sublist labels on the Sublists subtab of the custom record page:

Custom sublists present information related to the record you are viewing, based on results from a selected saved search of the record type or a related record type. You can apply a custom sublist to a custom record, so it displays on forms for that record. For more details, see Applying Custom Sublists to Custom Record Types.

**Translating Custom Record Instance Names**

In addition to providing translations for the name of the custom record type itself, you can provide translations for individual instances of that record type. For example, in addition to providing the Spanish translation “Muebles” for a custom record type named Furniture, you also can provide translations for the individual instances of that record type, such as chair, table, bed.

To allow translation of custom record instance names, check the Enable Name Translation box for the custom record type. This option is disabled by default.

You can enable the Enable Name Translation option for a custom record type if all of the following are true:

- The Multi-Language feature is enabled for the account.
The Include Name Field option is enabled for the custom record type.

Numbering is not enabled for the custom record type. (The Enable box on the Numbering subtab is not checked.)

When the Enable Name Translation box is checked for a custom record type, each custom record instance has a Translation subtab.

In addition, you can use the translated display name in saved searches to help users find custom record instances in their language preference. When setting up a saved search, add Display Name (Translated) and Language to the Results subtab, and add Language as a filter to the Criteria subtab. With these settings, the saved search results are filtered by the user's language preference.

Parent-Child Record Relationships

A child record type is a record that is referenced in another record in NetSuite. The information in the child record is associated with another record at a higher level, which is the parent record. You can use a child record to track multiple fields of specific information that are related to the parent record. Child records are always of the List/Record type. For more information, see Using Child Records.

For example, a typical customer record includes sublists, represented as subtabs on a form, for note and message record instances. In this case the customer record is a parent of the note and message records, and therefore the notes and messages are child records of the customer record. The name of a child record sublist on the parent record is the same as the name of the child record type.

**Parent Record:** Customer

**Child Records:** Notes, Messages (sublists)

You can also create your own parent-child relationships. For example, you can create a custom record type called Service Notes and associate it with an Equipment custom record type. In this example, the Service Notes record is associated as the child of the parent Equipment record type. On the Service Notes record type definition, you can create custom fields for service person, price of service, whether the service was covered under warranty, and the equipment's next date of service. With this parent-child relationship established, employees can then view these service notes when they are working on the parent Equipment records.

Parent-child relationships can exist between:

- Two standard records
- Two custom records
One standard record and one custom record

The following diagram demonstrates the parent-child record relationship. In this example both the parent and child records are custom records.

![Diagram of parent-child record relationship]

To add a child record to a parent record, the parent record must already exist, meaning that previously it must have been created and saved. When you open the parent record again in edit mode, you can then add the child records. You cannot add child records at the time when you are creating a new parent record.

**Note:** A child record may not be available on a form for a parent record that was created through transformation from another record type. For example, if you define a custom record as a child record of sales order, this custom child record is not available on forms for sales orders transformed from quotes.

For more information, see the following topics:

- Establishing a Parent-Child Relationship
- Creating a Parent-Child Relationship
- Types of Parent-Child Relationships
- Using Child Records

**Establishing a Parent-Child Relationship**

To establish a parent-child relationship between records, you must create a custom field and define both the parent and the child. This applies to both custom and standard records.

You can define the parent and child on the:

- Custom Field configuration page, when creating a new custom field
- New Field configuration page from a Custom Record Type page

**Custom Field Configuration for New Custom Field**

You can define parent and child records on the Custom Field configuration page.

- To define the parent, select a field type of List/Record or Multiple Select. Then in the List/Record field, select a record type. Check the Record is Parent box. The record entered in the List/Record field is the parent record.
To define the child, on the Applies To tab, check the box of the child record. If applicable, you can select multiple records as child records.

New Field Configuration for Custom Child Record

You can define parent and child records using the New Field configuration page from a custom record type page. To do so:

To define parent child records:

1. Create a custom child record as described in Creating Custom Record Types.
2. On the Fields subtab, to define the parent record, click New Field. Then select a field type of List/Record or Multiple Select.
3. In Fields subtab, to define the parent record, click New Field. Then select a field type of List/Record or Multiple Select.

Define a Subtab on a Parent Record

You can also define the subtab on the parent record where the child record appears. On the child field definition page, click the Display subtab. Then on the Parent Subtab field, enter the subtab where the child record appears.

Creating a Parent-Child Relationship

The following process describes an example of creating a parent-child relationship using a custom field. For this example, we are going to create a custom field and assign the parent record and child record. The parent record pulls in the child record information through the custom field. We want to associate a list of customers to our case records, so we will create an entity custom field called Customer Records. The parent record in this example is Case, and the child record is Customer. This Customer Records field displays a list of customers associated with the case.

To create the parent-child relationship:

1. Go to Customization > Lists, Records, & Fields [Custom Field] > New, where [Custom Field] is the desired field type.
   The type of field you select depends on the type of record you want to associate.
2. In the Label field, enter a name for your custom field. For this example, enter Customer Records.
3. In the Type field, select List/Record or Multiple Select.
To associate multiple records to another record, choose **Multiple Select**. In our example, Multiple Select lets you associate multiple customer records to a case.

4. In the **List/Record** field, select the parent record you want to associate this list (of customer records, in this case) with. For this example, select **Case**. Note that this field is available only if you selected a type of List/Record or Multiple Select.

5. To make the Case record a parent of the Customer child record, check the **Record is Parent** box. With this box checked, a child sublist appears on the parent record. The Record is Parent box is available after you select a valid record type in the List/Record field.

6. On the **Applies To** subtab, you specify the child record. Check the box beside the type of child record you want to associate with the parent record previously selected in the List/Record field. If applicable, you can check more than one record type. For this example, check **Customer**.

7. Click the **Display** subtab.

8. In the **Parent Subtab** field, select the subtab where you want this list to display on the parent record. Note that this field is only available when **Record is Parent** is checked.

9. Click **Save**.

You will now have a parent record of Case with a subtab called Customer Records that lists all customers affected by this case.

**Note:** When associating a list with a transaction, you can only choose transactions that have customizable forms. For example, you can associate a list of cases with a cash sale but not to a bill.

The following transaction types are available as parents:

- Cash Refund
- Cash Sale
- Credit Memo
- Estimate
- Invoice
- Opportunity
- Purchase Order
- Return Authorization
- Sales Order

**Triggering User Events**

User events are triggered when a custom child record is associated with or separated from its parent record, unless the custom field referencing the parent record is of type Multiple Select. A beforeLoad, beforeSubmit, or afterSubmit event occurs if a record is associated or separated.

The user events support scripting and workflows based on association and separation of custom child records. The event is audited on the custom child record as an edit execution event type and is supported in SuiteScript 2.0. For more information about user events, see the help topic *SuiteScript 2.0 User Event Script Type*.

**Types of Parent-Child Relationships**

Parent-child relationships can exist between two standard records, two custom records, or a combination of standard and custom records. For more information, see *Parent-Child Record Relationships*. 
There are different ways in which you can create a parent and child record relationship, including:

- One parent record type to one child record type
- Many parent record types to many child record types
- Hierarchy of parent-child instances of the same record type

This topic refers to both record types and record instances. Parent-Child relationships are established between the record types during record type configuration. Record types can have multiple instances. For example, a child record type of Address can have multiple instances such as home address, mailing address, and shipping address.

To learn more about parent-child record relationships, see the following sections:

- Types of Relationships Between Parent-Child Record Instances
- Custom Record Type Fields that Affect Parent-Child Relationship on Custom Record
- One Parent Record Type to One Child Record Type
- Many Parent Record Types to Many Child Record Types
- Hierarchy of Parent-Child Instances of Same Record Type
- Parent-Child Relationship Limitations

Types of Relationships Between Parent-Child Record Instances

There are limits to how many child record instances can be associated with or separated from a parent record instance. As well, there are limits to the number of parent record instances to which a child record instance can associate. These quantities are determined by the relationships between the parent and child record types.

The most common association between record instances permits a one-to-many relationship, where you can have either of the following:

- One parent record instance associated with many child record instances
- One child record instance associated with many parent record instances

In these cases, a single child record type is associated with a single parent record type. However, there can be multiple instances of either the child record or the parent record. See One Parent Record Type to One Child Record Type.

You also have an association between instances that permits a many-to-many relationship. In this case, you have many parent record types and many child record types. Each parent record instance can have many associated child record instances. At the same time, each child record instance can have many associated parent record instances. See Many Parent Record Types to Many Child Record Types.

Currently, it is possible to define a one-to-one relationship only by workflows or custom scripts. For more information, see One-to-One Parent-Child Relationship Between Instances of Different Record Types.

Custom Record Type Fields that Affect Parent-Child Relationship on Custom Record

The following fields on the custom record definition page affect the functionality of the parent-child relationship on instances of custom records. For more information about these fields, see:

- Configuring File and Child Record Settings
Defining Search and Edit Settings

Show Remove Link – When cleared, child record sublist Remove links are not visible, which prevents users from separating child records from the parent record.

Allow Child Record Editing – When checked, users can directly edit record instances of this custom record type when they display as children in a sublist on a parent record.

Allow Delete – When checked, users can delete record instances of this type when they appear as child records in a sublist on a parent record.

Hierarchy – When checked, users can create parent-child relationships between records of the same custom record type. To do so, create a custom field which will be used to establish the relationship between the custom records, as described in Hierarchy of Parent-Child Instances of Same Record Type.

One Parent Record Type to One Child Record Type

In a one record type to another record type parent-child association, you have one field on a child record with Record is Parent checked. When the Record is Parent box is checked on the child record, a child record sublist is automatically added to the parent record. Therefore, the parent record has one sublist to which it can associate with one or more child record instances. These record instances all are of the same record type. The records can be custom or standard.

Example of One Parent Record Type to One Child Record Type

An example of a one-to-one record type relationship is a customer parent record type with an address child record type. Typically, in a one record type to another record type scenario, the child record has multiple instances, but the parent record only has one instance. The following example shows a customer record instance with three addresses. The three different addresses are the child record instances.

One-to-One Parent-Child Relationship Between Instances of Different Record Types

A parent-child relationship where one parent instance can associate with only one child instance is currently possible only by creating a custom script or workflow. By default, users can associate one or more instances.

One Child Instance Associated with Many Parent Instances

The one child instance associated with many parent instances scenario has the same relationship as the one parent instance associated with many child instances scenario. There is one parent record type and one child record type. It is, however, special in terms of its relationship between instances. Usually, one parent instance associates with one or more child instances. The parent record has a sublist where each row in the list shows a specific associated child record instance.
On the child record instance, there is a link to the parent record instance. An example, illustrated by the Equipment field with Record is Parent box checked, is shown in the Parent-Child Record Relationships topic diagram.

In the less common one-to-many scenario, you associate the same child record instance with different parent record instances of the same record type. In other words, many parent record instances associate with the same child record instance. The individual parent record instances have sublists showing the associated child record instance. On the child record instance there is a multi-select field in which many parent records are listed.

You create this type of relationship by checking Record is Parent on the custom field and selecting a field type of Multiple Select instead of List/Record.

**Example of Many Parent Instances to One Child Instance**

The following example shows a relationship of one child record type to one parent record type. In this case, there are multiple parent record instances for one child record instance. The specific hotel room, the parent record, has multiple booking reservations, the child records, for that room.

In this example, Child record 3 (Reservation#3) associates with two parent records.

Hotel Rooms 15 and 20 are both instances of the Hotel Room custom record type.

Child record instances are of the custom record type Booking Reservation, where:

- The first Booking Reservation instance is booked for date 1 and is a child record of Hotel Room 15. On the parent record instance for Hotel Room 15, the booking reservation instance is shown as the first row in a Booking Reservation sublist. For example, Booking Reservation - name of child record type.

  **Note:** The child record sublist on the parent record, in this case, Booking Reservation, has the same name as the child record type.

- The second Booking Reservation for Hotel Room 15 is similar but reserved for date 2. On the parent record instance for Hotel Room 15, the booking reservation instance is shown as the second row in the Booking Reservation sublist.

- The third Booking Reservation for date 3 has two parent record instances. On the parent record instance for Hotel Room 15, the booking reservation instance is shown as the third row in the Booking Reservation sublist. On the parent record instance for Hotel Room 20, the booking reservation instance is shown as the first row in the Booking Reservation sublist.

**Many Parent Record Types to Many Child Record Types**

In a many-to-many record type relationship, more than one field is defined on different child record types with Record is Parent checked. Each of these fields can refer to one or more parent record types as well. If many parent to many child associations are created, many sublists show on the parent record type instances. Each parent record can associate with different child records of different types.
**Example of Many Parent Record Types to Many Child Record Types**

An example of a many parent record types to many child record types relationship is one where there are:

- Two parent record types: Hotel Room and Parking Space
- Two child record types: Booking Reservation and Pricing

Hotel rooms have:

- Booking reservations
- Information about seasonal pricing policy

Parking spaces have:

- Booking reservations

Example setup:

- Two parent record types: 1 - Hotel Room, and 2 - Parking Space
- Two child record types: A - Booking Reservation, and B - Pricing Rate
- Two list/record type custom fields applied to child record type A, where:
  - Record is Parent box is check for both fields
  - The first field is a list/record type that refers to parent record type 1
  - The second field is a list/record type that refers to parent record type 2, creating a many-to-many relationship

When the field definitions are saved, the Hotel Room parent record type shows sublists for Booking Reservation and Pricing Rate.

- One list/record type custom field applies to child record type B, has the Record is Parent box checked, and the List/Record selection refers to parent record type 1

On parent record instances of type 1, there are two sublists. The first can associate instances of child record type A, and the second can associate instances of child record type B. On parent record instances of type 2, there is only one sublist which can associate instances of child record type A.

The following diagram shows the relationships between the parent and child instances.
Hotel Room 15 and the President Suite are instances of parent record type Hotel Room. These instances have two sublists:

- The first sublist refers to Booking Reservations
- The second sublist refers to Pricing Rates of hotel rooms

Hotel Room 15 has three associated reservations, shown as rows in a sublist on the parent record instance. These reservations are numbered 1, 2, and 3. Reservation #3 is also associated as a child of Parking Space 3, which is of record type Parking Space. This means that the customer reserved the hotel room together with the parking space.

Hotel Room instances have associated Pricing Rate instances that determine how hotel room rates are calculated. In this example, Hotel Room 15, which is a standard room, has two different seasonal rates. Based on the reservation date, the room could have rates associated in a:

- Summer Season Rate (Jun-Aug, 20% more) sublist
- Winter Off-season Rate (Sep-May, 10% less) sublist

The President Suite hotel room has only one year-round Premium Rate associated with it (50% more). In this example, the room has no associated reservation.

**Hierarchy of Parent-Child Instances of Same Record Type**

You can create a hierarchy of records where parent and child records are of the same type. Each instance can be a parent of another instance of the same type. A parent record can be a parent of another instance of the same record type. Such behavior can be configured on the custom field definition of the child record, where:

- The Record is Parent box is checked
- List/Record selection is the same record type as the record type where the relation field is applied

**Note:** Administrators can alternately model this behavior more easily on custom record types by checking the Hierarchy box on the custom record definition. See Custom Record Type Fields that Affect Parent-Child Relationship on Custom Record.

**Example of Hierarchy of Parent Instances of the Same Record Type**

An example of a hierarchy of the same record type relationship is a locomotive with several wagons. The record type is Train Carriage. Parent record instance 1 is Locomotive. Parent record instance 2 is Wagon 2 and is a child of parent record 1. Parent record 3 is Wagon 3 and is a child of parent record 2. This pattern continues until you reach the last wagon. The last wagon is the terminal child record. This terminal child record has an empty sublist with no other child records and is a child of the preceding parent record wagon.

The following example has four instances of the same the record type, Train Carriage.

- Parent record instance Locomotive 1 has no associated parent but has associated instance (Wagon 2) in the child sublist
- Parent record Wagon 2 has an association with its parent instance (Locomotive 1) and has an associated child record Wagon 3 in the child sublist
- Parent record Wagon 3 has an association with its parent instance (Wagon 2) and has an associated child record Wagon Last in the child sublist
- Terminal Wagon Last has no instances associated in its child sublist but holds an association with its parent instance (Wagon 3)
Parent-Child Relationship Limitations

**Note:** This section describes some limitations to the types of parent-child relationships you can create in the current record customization UI. However, through scripting, you can enforce the type of relationship described here.

An administrator cannot enforce an exclusive parent-child relationship, where:

- One child record belongs exclusively to one parent record, and
- The child record cannot exist without the parent record

This means that the child record is indivisible from the parent record and can only exist if the parent exists. On its own, the child record has no meaning.

For example, you cannot have a serial number child record belong exclusively to an item number parent record. The serial number is considered an essential part of the item number and has no meaning without the item number. You cannot load the serial number record without the context of the item.

The following behaviors must be possible:

- You can modify the parent field on the child record
- The child record exists only with its parent
- The life cycle of the child record is the same as the life cycle of the parent record.
- You can access the child record only from the parent record (the child record does not have its own entry form)
- The child record is deleted when the parent record is deleted

You must use scripting to enforce the behaviors required for exclusive parent-child relationships. For more information, see the help topic [SuiteScript 2.0 Scripting Records and Subrecords](#).

Using Child Records

After you have created your child record types, you can enter data into records from the child record list or parent records.

**To enter data in child records from the child record list:**

1. Go to Customization > Lists, Records & Fields > Record Types.
2. Click **New Record** next to the name of the child record type you want to enter records for.
3. In the field that links to the parent record, choose the record you want this child record to link to.
For example, in the Equipment field, you would choose the machine you are entering service notes for.

4. Enter data in the remaining fields on the record.

5. When you are finished, choose one of two options to submit the information to NetSuite:
   - Click Submit to submit the information and return to the child record list.
   - Click Submit & New to submit the information and add another record.

You can now view this child record from its parent record.

**To enter data in child records from the parent record:**

1. Open the list of records where your parent record appears.
   - If your parent record is a custom record:
     1. Go to Customization > Lists, Records & Fields > Record Types.
     2. Click the name of the custom record list you want to open.
     If you have a link in your center directly to your parent record list, you can click that link instead of the above steps.
   - If your parent record is a standard NetSuite record, on the Lists page, click the name of the list you want.
   You can associate child records with any of the standard NetSuite records, but for records that are transformed from other records, child record association cannot occur when the records are first created, only after they have been saved.
     1. Click View next to the name of the record you would like to enter data for.
     2. Click the subtab where the child record list appears.
     3. Click New above the list to enter a new child record.
     4. Enter data in the fields on the record.
     5. When you are finished, choose one of two options to submit the information to NetSuite:
       - Click Submit to submit the information and return to the parent record.
       - Click Submit & New to submit the information and add another child record.

To view a child record from the parent record, click the link for the child record in the list of child records. To edit a child record from the parent record, click Edit.

By default, inactive child records are not shown on the parent record. If you customize the sublist and the saved search filters by the Inactive box, inactive records can be shown in the sublist.

**Warning:** A child record may not be available on a form for a parent record that was created through transformation from another record type. For example, if you define a custom record as a child record of sales order, this custom child record is not available on forms for sales orders transformed from quotes.

**Sourcing with Custom Records**

You can use sourcing with custom record types for both standard and custom records. You must first create a List/Record field on your custom record type for the kind of record you want to source information from. After you have created that field, you can then choose it in the Source List field.
For example, you have created an Intern custom record type. You want to include information on your intern records about the employees supervising each intern. You can do this by first creating an Employee field with a list of your employees. Then you can source information from employee records to other fields on your intern records.

For information on sourcing and custom fields, see Sourcing with Custom Records.

To use sourcing with custom record types:

1. Go to Customization > Lists, Records & Fields > Record Types.
2. Click Edit next to the record type you want to create a sourced field for.
3. Click the Fields subtab.
4. Click New Field.
5. In the Description field, enter the name of your new field. For example, if you are creating a field to list your employees you would enter Employee.
6. In the Type field, select List/Record.
7. In the List/Record field, select the kind of record you want to include information from in other fields on your record type. For example, if you want to include information about your employees you would select Employee.
8. Click Save.
   Now, you can create your sourced fields.
9. Click the Fields subtab.
10. Click New Field.
11. In the Description field, enter a description for your field. For example, if you wanted to include an employee’s email address you would enter Employee Email.
12. In the Type field, select a type of custom field. The type you select must match the type of field you want to source from. For example, if you wanted to include an email address you would choose Email Address in the Type field.

For a list of standard fields and their types, see Available Standard Fields and Field Types for Custom Record Types and Source Lists.
13. If you are creating a List/Record field, select the appropriate list or record in the List/Record field. For example, if you are sourcing from the Sales Rep field on Customer records, choose Employee in the List/Record field because your sales reps are employees.
14. In the Source List field, select the record type you want to source information from. This field lists the field you created in steps 1-9. You can create multiple source lists for your custom records by repeating steps 1-9 for each kind of record you want to be able to source information from.
15. In the Source From field, select the field you want to include information from. You must choose a Source List before you can choose a field. The selected field’s type must match the type selected in the Type field.
16. When you have finished, click Save.
17. To add Source List fields, repeat steps 1-9. To add sourced fields, repeat steps 10-17.
You can now include information from standard and custom fields and records on your custom record types.

**Available Standard Fields and Field Types for Custom Record Types and Source Lists**

Custom record types and source lists can use sourcing with the standard fields and records listed below. In addition to standard records, you can also use custom fields and record types for sourcing. When using custom fields and record types for sourcing, the field type chosen depends on the field type of the custom field being sourced from.

For example, you have created a custom entity check box field for Quarterly Mailing. If you want to source information from the Quarterly Mailing field to a custom CRM field for cases, you must choose Check Box in the Type field when creating your CRM field.

### Customer, Employee, Partner and Vendor Fields

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Customer</th>
<th>Employee</th>
<th>Partner</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name/ID</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Bill To</td>
<td>Text Area</td>
<td>×</td>
<td>×</td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>Ship To</td>
<td>Text Area</td>
<td>×</td>
<td></td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Phone</td>
<td>Phone Number</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>Fax</td>
<td>Phone Number</td>
<td>×</td>
<td></td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>E-mail</td>
<td>E-mail Address</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>City</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>State</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>Zip</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>Country</td>
<td>Free-Form Text</td>
<td>×</td>
<td>×</td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>Sales Rep</td>
<td>List/Record*</td>
<td>×</td>
<td></td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>Expected Close Date</td>
<td>Date</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Renewal Date</td>
<td>Date</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contact</td>
<td>Free-Form Text</td>
<td>×</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Alt. Contact</td>
<td>Free-Form Text</td>
<td>×</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Alt. Phone</td>
<td>Phone Number</td>
<td>×</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Balance</td>
<td>Currency</td>
<td>×</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Credit Limit</td>
<td>Currency</td>
<td>×</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Account</td>
<td>Free-Form Text</td>
<td>×</td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>1099 Eligible</td>
<td>Check Box</td>
<td></td>
<td></td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Tax ID</td>
<td>Free-Form Text</td>
<td></td>
<td></td>
<td></td>
<td>×</td>
</tr>
</tbody>
</table>
### Sourcing with Custom Records

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Customer</th>
<th>Employee</th>
<th>Partner</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Company Name</td>
<td>Free-Form Text</td>
<td></td>
<td>×</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>Supervisor</td>
<td>List/Record*</td>
<td></td>
<td></td>
<td></td>
<td>×</td>
</tr>
<tr>
<td>Soc. Sec. #</td>
<td>Free-Form Text</td>
<td></td>
<td></td>
<td>×</td>
<td></td>
</tr>
</tbody>
</table>

* In the List/Record field, you must choose the Employee list for sourcing to work properly.

### Item

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Display Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Vendor Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Online Name</td>
<td>Free-Form Text</td>
</tr>
<tr>
<td>Available Online</td>
<td>Check Box</td>
</tr>
<tr>
<td>Base Price</td>
<td>Currency</td>
</tr>
<tr>
<td>Cost</td>
<td>Currency</td>
</tr>
<tr>
<td>Preferred Vendor</td>
<td>List/Record*</td>
</tr>
<tr>
<td>On Hand</td>
<td>Decimal Number</td>
</tr>
</tbody>
</table>

* In the List/Record Field, you must choose the Vendor list for sourcing to work properly.

### Updating Custom Record Types

#### Inactivating a Custom Record Type

**To inactivate a custom record type:**

1. Go to Customization > Lists, Records & Fields > Record Types.
2. At the top of the page, check the **Show Inactives** box.
3. Check the box in the **Inactive** column next to the record type you want to inactivate.
4. Click **Submit**.

#### Deleting a Custom Record Type

**To delete a custom record type:**

1. Go to Customization > Lists, Records & Fields > Record Types.
2. In the Edit column, click the name of the custom record type you want to delete.
3. In the More Actions menu, click **Delete**.

After the deletion, the list of custom records no longer includes the deleted record type.

**Important:** You cannot delete a custom record type that is used by any custom fields or workflows. When you attempt to delete a custom record type that has this kind of dependency, you receive error messages with lists of custom field names and/or custom workflow names that depend on the custom record. You must delete any dependent custom fields and workflows before you can delete the record type.

Viewing or Editing a Custom Record Type

**To view or edit a custom record type:**

1. Go to Customization > Lists, Records & Fields > Record Types.
2. In the Edit column, click the name of the custom record type you want to view or edit.
3. Record types have the following subtabs for you to further define them:
   - **Fields** – Create and rearrange the fields for your custom record type. For information, see [Adding Fields to Custom Record Types](#).
   - **Subtabs** – Create and arrange subtabs for your custom record type. For information, see [Adding Subtabs to a Custom Record](#).

   **Note:** To save time, create and arrange subtabs for your custom records before defining your custom fields.

   - **Sublists** – Add search results as sublists on your custom record type. For more information, see [Applying Custom Sublists to Custom Record Types](#).
   - **Icons** – Select the .png sprite you want to use to represent this record type in the New Bar, Create New menu, Recent Records menu, Recent Records portlet, and QuickViews. You can choose from built-in icons or create your own custom icon. For more information, see [Choosing an Icon for a Custom Record](#).
   - **Numbering** – Specify the numbering format for custom record types. For information, see [Numbering Custom Record Types](#).
   - **Forms** – Customize and select a preferred entry form for your custom record type. For more information, see [Adding Custom Forms for a Record](#). To print your custom record, you can choose the standard print template or a custom print template when editing the preferred entry form for your custom record type. For more information, see [Updating Custom Record Type Print Templates](#).
   - **Permissions** – Choose the roles you want to access custom record entry forms, choose a default form and restrict the forms available here. For information, see [Setting Up a Permissions List for a Custom Record Type](#).

   **Important:** You must choose **Use Permission List** from the Access Type list for these permissions to take effect. For custom record types that are associated with a custom segment, you cannot edit the permissions of the custom record. The custom record permissions are set on the custom segment definition page.

   - **Links** – Create links that take you to the list of record entries for this type and choose where to place the links. See [Creating Links to Custom Records](#).
■ **Managers** – Define specific employees as managers of the current record type. This allows them to modify the custom record type. When defined as a manager, the employee is automatically granted custom record view permission. This allows them to see the list of custom record types but **not** drill down on them.

**Note:** If an employee has a role that includes the Custom Record Type permission, they have edit access to **all** custom records types. The **Managers** subtab enables you to grant permission for an employee to the current record type only.

■ **Translation** – (when Multi-Language feature is enabled) Define translations for the custom record type name to be used when users change the language preference. See Adding Translations for Custom Records.

■ **Child Records** – If this record type is a parent record, the child records are listed here. See Using Child Records.

■ **Parent Records** – If this record type is a child record, the parent records are listed here. See Parent-Child Record Relationships.

4. When you have finished making changes, click **Save**.

### Updating Custom Record Type Print Templates

Each custom record type has a standard print template that is dynamically generated when you create a custom record type. If you edit the custom record type, the standard template is automatically updated to reflect the changes made in the custom record type.

You can use the advanced template editor to create a custom print template for the custom record.

**Note:** Customized record type print templates are not updated when the custom record type is changed.

For more information about using the Template Editor, see Customizing Advanced Templates in the Template Editor.

After you have created a custom print template, you update the custom record type configuration to use the custom template.

**To update the custom record type print template:**

1. Go to Customization > Lists, Records, & Fields > Record Types (Administrator).
2. In the **Edit** column, click the name of the custom record type you want to view or edit.
3. On the **Forms** subtab, click **Customize** on the form you want to edit.
4. Select the custom print template from the **Print Template** list. The standard print template is listed as **Default** on the Print Template list on the Custom Entry Form configuration page.
5. Click **Save**.

### Custom Record Types Associated with a Custom Segment

On the Record Types list page, the custom segment name in the Segment column means that the custom record type is associated with a custom segment.
The custom record has the same name as the custom segment.

If you edit a custom record that is associated with a custom segment, note the following:

- Some of the fields on the Custom Record Types page are read-only to avoid conflict with the custom segment settings.
- Some of the subtabs on the Custom Record Types page are not available because the record is associated with the custom segment.
- You cannot edit the permissions of the custom record. The custom record permissions are set on the custom segment definition page.
- The custom record type cannot be inactivated because it is associated with a custom segment.

When you add a new field to the record, it appears as a column on the Values subtab of the custom segment definition page.
Using Custom Record Entries

Custom record instances or entries are the actual custom records you create. You can create, edit and search your custom record instances from the list of record types. You can also inactivate or delete a
record type that is no longer being used. To provide users with access to all custom records, user roles must have the Custom Record Entries permission enabled. For more information about role permissions, see the help topic Setting Permissions for Custom Records.

For details, see the following:

- Inactivating a Custom Record Type
- Deleting a Custom Record Type
- Viewing a Custom Record Entries List
- Viewing or Editing a Custom Record Entry
- Creating a Custom Record Entry
- Copying a Custom Record Entry
- Viewing or Editing a Custom Record Type
- Custom Record Types Associated with a Custom Segment
- Searching Custom Record Entries

Viewing a Custom Record Entries List

To view a custom record entries list:

1. Go to Customization > Lists, Records & Fields > Record Types.
   You can also reach the custom record entries list by placing a link directly to the list in your center. See Creating Links to Custom Records.

2. Click List for the custom record entries list you want to see.
   Your list of custom records appears.
   You can also reach the custom record entries list by placing a link directly to the list in your center. See Creating Links to Custom Records.

Viewing or Editing a Custom Record Entry

To view or edit a custom record entry:

1. Go to Customization > Lists, Records & Fields > Record Types, and click List for the custom record entries list you want to see.
   You can also reach the custom record entries list by placing a link directly to the list in your center. If you want a link to appear on a standard category, you must set the link on the Custom Record Type definition page. Also, if you want to copy the custom record and center link to another account, you must set the link on the Custom Record Type definition page. See Creating Links to Custom Records for details.

2. In the records list, click View or Edit for the record you want.

3. If you are editing, click Save when you are finished.

Creating a Custom Record Entry

To create a custom record entry:

1. Go to Customization > Lists, Records & Fields > Record Types, and click New Record.
2. Enter a name and any notes you want to add, and click **Save**.

### Copying a Custom Record Entry

**To copy a custom record entry:**

1. Go to Customization > Lists, Records & Fields > Record Types, and click **List** for the custom record entries list you want to see.
2. Click **View** next to the record you want to copy.
3. In the **Actions** menu, choose **Make Copy**.
4. Enter a name for your new record, and click **Save**.
   
   You cannot create another record with the same name. The information from the original record is copied except the name.

### Searching Custom Record Entries

**To search a list of custom record entries:**

1. Go to Customization > Lists, Records & Fields > Record Types.
2. Click **Search** for the record type you want to search.
3. On the Search page, enter or select criteria to filter search results:
   
   To enter specific criteria:
   - Select **Any** to search for all information of that kind.
   - Select **Is, Starts With, or Contains** and enter the information or part of the information you want to find.
   - Select **Is Not, Does Not Start With, or Does Not Contain** and enter the information you want to exclude.
   - Select a choice or choices from a list or a list of options.
   - Click **List** next to an empty list field to select from a popup, or enter the first few letters and press the **Tab** key.
   - Enter or pick dates in **From** and **To** fields if you selected custom in the date field.
   - Select **Either, Yes, or No** for the option you want.

   Using a combination of these fields is the best way to find what you are looking for.
4. After entering search criteria, choose one of the following options:
   - Click the **Submit** button to run the search and open a NetSuite page with a list of results.
   - Click the **Export** button to run the search and save results as a .csv file that you can save to disk or open on your desktop. For more information, see the help topic [Exporting Search Results](#).
   - Click the **Reset** button to clear the filters you defined.
   - Click the **Customize** button to open a saved search page with no filters defined, where you can define a custom search form to be your default search form for the record type. See the help topic [Defining a Saved Search as a Preferred Search Form](#).
   - Click the **Create Saved Search** button to open a saved search page that includes the filters you defined. See the help topic [Saved Searches](#). In a saved search, you can include the translated display name of custom record instances in saved searches. See the help topics [Saved Searches](#) and [Adding Translations for Custom Records](#).
5. After you have submitted a search and a search results page has opened, you can do the following:
   ■ Export search results as a .csv or .xls file. See the help topic Exporting Search Results.
   ■ Email search results to one or more recipients. See the help topic Emailing Search Results.
   ■ Create a saved search with the same definitions as the search. See the help topic Saved Searches.

6. If the list of results is too large or too small, you can add, remove or expand criteria. To return to
   the search criteria page, click Return To Previous Search at the bottom of the Search Results page.
   ■ To set more complex search criteria and define search results display options, enable the Use
     Advanced Search box on the Search page. For information about using advanced search
     functionality, see the help topic Defining an Advanced Search.
   ■ Saved searches can be run on demand, include all advanced search functionality and provide
     additional capabilities, including scheduling, email alerts, audience definition, and highlighting
     of results. See the help topic Saved Searches.
Custom Transactions

The Custom Transactions feature lets you create transaction types tailored to your business needs.

This feature, along with Custom GL Lines and Custom Segments, is part of the SuiteGL feature set. For more about the SuiteGL feature set, see the help topics SuiteGL Features Overview, Custom Segments, and Custom GL Lines Plug-in.

![Important: SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should only access the NetSuite UI by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic SuiteScript 2.0 Custom Pages.]

For information about custom transactions, see the following topics:

- Benefits of Custom Transaction Types
- Sales and Purchase Transaction Types Overview
- Custom Transaction Type Setup
- Creating and Editing Custom Transaction Types
- Custom Transaction Type Association with a Custom GL-Lines Plug-in Implementation
- Deleting Custom Transaction Types
- Creating Sales and Purchase Custom Transaction Instances
- Printing Custom Transaction Instances
- Custom Transaction Types in Workflows
- Custom Transaction Types in Bundles

CSV Import Assistant, SuiteScript, and SOAP Web Services are supported by custom transaction instances.

- You can interact with custom transaction instances by using the CSV Import Assistant. For details, see the help topic Custom Transactions Import.
- You can interact with custom transaction instances using SuiteScript. For details, see the help topic Custom Transaction.
- You can interact with custom transaction instances by using SOAP Web Services. For details, see the help topic Custom Transaction.

Benefits of Custom Transaction Types

The Custom Transactions feature has the following benefits:

- You can name your custom transaction types in a way that reflects your business logic. For details, see Custom Transaction Type Naming Enables Better Organization.
- Like standard transactions, each custom transaction type can have its own numbering scheme, permissions, and workflow logic. For details, see Custom Transaction Types Support Key NetSuite Features.
- You can create custom transaction types in multiple styles. For example, the transaction type can resemble a journal entry, or it can behave more like an expense report. For details, see Custom Transaction Styles.
Custom Transaction Type Naming Enables Better Organization

In your business, there may be a wide variety of events that can require an adjustment to your general ledger. For example, you may need to record adjustments for non-operational income, such as interest income that your company receives through investments. Conversely, you may need to record debits for rewards you give customers through customer loyalty programs. Without the Custom Transactions feature, your options for recording these various adjustments may be limited. One strategy might be to record all adjustments as journal entries. However, when you rely solely on the journal entry record, all of these varying transactions are grouped together in a single list view. Moreover, when employees enter journal entries, they have limited choices for distinguishing one type of journal entry from another.

By contrast, with the Custom Transactions feature, you can create custom transaction types that are clearly labeled for specific purposes. With this approach, each custom transaction type has its own list view and its own menu path, which you can customize. In the following example, three custom transaction types have been added to the Accounting Center’s Financial tab.

These enhancements make it possible for employees to automatically classify each transaction when they create it. And if they want to view a list of all transactions of a particular type, they can do so using that type’s list view.

Custom Transaction Types Support Key NetSuite Features

When you use custom transaction types, you can leverage many of the same features that are available with standard NetSuite transaction types. For example:

- Each custom transaction type can have its own numbering, permissions, and workflows.
- Custom transaction types can include custom transaction fields that you define.
- You can create multiple custom entry forms for each transaction type.
- You can see what changes have been made to the custom transaction type configuration. For more information, see System Notes v2 for Custom Transaction Types.
- You can interact with custom transaction instances using SuiteScript. For details, see the help topic Custom Transaction.
- You can interact with custom transaction instances by using SOAP web services. For details, see the help topic Custom Transaction.
- You can interact with custom transaction instances by using the CSV Import Assistant. For details, see the help topic Custom Transactions Import.
- You can reference your custom transaction types when creating saved searches. To find custom transaction types, use the Transaction search type. In the Types list, your custom transaction types are listed with standard transaction types.
You can use global search or SuiteAnalytics Workbook to find a custom transaction.

You can review system notes for custom transaction type configuration changes. Use the System Notes v2 (Beta) workbook to view changes to the configuration of custom transaction types. For more information, see the help topic System Notes v2 Workbook.

You can interact with custom transaction instances by using SuiteAnalytics Connect. To reference your custom transaction types, use the Transaction table. Use the transaction_type column to identify your custom transaction type. For more details, see the help topic Connect Schema.

Custom transaction types can be bundled, as described in Custom Transaction Types in Bundles.

Deploy custom transaction types to other accounts using SuiteCloud Development Framework, with some limitations. For more information, see the help topic SuiteCloud Development Framework.

Custom transaction types can be used in conjunction with the Custom GL Lines plug-in. With the Custom GL Lines plug-in, you can create logic that automatically creates a GL impact. For more information, see Custom Transaction Type Association with a Custom GL-Lines Plug-in Implementation.

Support for these features makes it possible for each transaction type to have its own unique behavior and processing. These advantages may be critical as you develop a series of discrete transaction types to meet different needs.

**Note:** Custom transactions appear on the Deposits and Credits subtab of the Reconcile Bank Statement if the Show All Transaction Types in Reconciliation box is checked on the Accounting Preferences page. For more information, see the help topic General Accounting Preferences.

For example, suppose you have a custom transaction type called Bad Debt, which you use to account for debts that are not collectible. For this type, you may want to restrict access to a limited set of users. You can manage this access on the Permissions subtab by using the transaction type’s Level sublist.

You may also want to create a custom workflow for this type. On the New Workflow page, custom transaction types are listed with standard transaction types.
If appropriate, your workflow can reference any custom statuses that you have defined for your transaction type. Custom statuses are a unique feature of custom transaction types and are not available with standard transaction types. You create statuses by using the transaction type’s Statuses subtab.

### Custom Transaction Styles

The Custom Transactions feature supports multiple transaction formats. That is, you can create transaction types in any of the following styles:

- **Basic** – Lets users record credit or debit lines to specified accounts. The corresponding account to be adjusted for balancing purposes is defined on the transaction type record. This approach is similar to the expense report transaction, which always debits the same predefined account.

- **Journal** – Lets users record sets of debits and credits to accounts that a user manually specifies when entering the transaction. As with a standard journal entry record, the total value of credits must equal the total balance of debits.

- **Header only** – Relies on a GL plug-in implementation to calculate the GL impact. That is, the transaction does not include a Lines sublist for users to manually enter debits and credits to specific accounts. Rather, the plug-in implementation calculates the impact based on other data. This data can consist of values that users enter on the transaction header or of values they enter on a custom form created by using SuiteScript objects.

- **Sales** – Provides functionality similar to a sales transaction, including the Item sublist, taxes (SuiteTax only), and inventory impact. Sales custom transactions behave similar to invoices, cash sales, or credit memos.

- **Purchase** – Provides functionality similar to a purchase transaction, including the Item and Expenses sublists, inventory impact, taxes (SuiteTax only), and amortization. Purchase custom transactions behave similar to vendor bills.

For full details on the various transaction styles, see [Custom Transaction Styles Overview](#).

### Sales and Purchase Transaction Types Overview

Sales and purchase custom transaction types provide behavior similar to sales and purchase transactions, enabling you to create custom transaction types that use the Item sublist and define the GL impact of transactions. The behavior of sales custom transactions is similar to invoices, while purchase custom transactions are similar to vendor bills.

**Note:** Sales and purchase transaction types support SuiteTax only, not legacy tax.

A sales custom transaction provides sales transaction functionality that you can incorporate into your sales workflow. For example, Large and Associates is a non-profit company that relies on donations. The company needs the ability to enter a cash donation in NetSuite with payment options including check or credit card. This transaction should not be recorded in accounts receivable. When creating the transaction, users need to choose items on the Items subtab, and choose the cash, check, or credit card account to post the transaction. A sales custom transaction meets these requirements.
A purchase custom transaction provides purchase transaction functionality that you can incorporate into your purchase workflow. For example, when Seven Company in Brazil imports goods, the company is responsible for issuing a Nota Fiscal to be able to carry the goods from customers at the port of arrival, to its own premises. A purchase custom transaction can meet these requirements.

Sales and Purchase Functionality Available in Custom Transactions

Generally, sales and purchase custom transactions function the same as standard transactions, for example, invoice or vendor bill. Currently, the following features are supported by sales and purchase custom transactions:

- Items Subtab Support - Sales and purchase custom transactions support the Items subtab, where you can include details of items for sale and for purchase. For more information, see the help topic Adding Items on a Sales Transaction.

- Expenses Subtab Support - Purchase custom transactions support the Expenses subtab, where you can include expense details related to purchases. For more information, see the help topic Vendor Bills.
- **Tax** – Sales and purchase custom transactions support SuiteTax only, not legacy tax. For more information, see the help topic SuiteTax.
  The Standard Tax Report includes sales and purchase custom transaction details. For more information, see the help topic Generating a Standard Tax Report in SuiteTax.

- **Inventory impact and costing** – Sales and purchase transactions can increase or decrease your stock level. If you purchase an inventory item using a posting credit custom purchase transaction, your stock level increases. Your last purchase price and average inventory price are updated. If you sell an inventory item using a posting debit custom sales transaction, your stock level decreases. Cost of Goods Sold accounting lines are automatically posted to the transaction, depending on your inventory valuation method. For more information, see the help topic Inventory Costing Preferences.

- **Revenue Recognition** – You can use both revenue recognition and advanced revenue management features with sales custom transactions. Revenue recognition and advanced revenue management work the same with sales custom transactions as they do when created from an invoice or credit memo. For more information, see the help topic Revenue and Expense Recognition Overview.

- **Amortization** – You can attach amortization schedules to your purchase custom transactions in the same way as you do it for vendor bills. For more information, see the help topic Amortization Feature Overview.

- **Currency Revaluation** – You can enter your transaction in different currencies. They get revalued at the end of the month or at the time of their payment. For more information, see the help topic Currency Revaluation Transactions.

- **Multi–Subsidiary Customer and Vendor** – You can use multi–subsidiary customers and vendors on sales and purchase custom transactions. For more information, see the help topic Transactions Available for Multi-Subsidiary Vendors.

- **Elimination** – Intercompany transactions are automatically eliminated at the end of the month if you have the Automated Intercompany Management feature enabled. For more information, see the help topic Automated Intercompany Management Overview.

- **Promotions, SuitePromotions, and Discounts** – You can apply promotions, SuitePromotions, and discounts to sales custom transactions. For more information, see the help topic Promotions Overview.

- **Shipping** – You can use the shipping feature with sales custom transactions to manage your company's shipping needs. For more information, see the help topic Shipping. When an eligible SuitePromotion is added, the shipping discount is applied after you calculate the shipping costs. For more information, see the help topic Shipping Promotions and Multiple Shipping Routes.

- **Gift Certificates** – You can sell gift certificates on sales custom transactions. Then you can apply those gift certificates towards payments of future transactions. For more information, see the help topic Gift Certificates.

- **Installments** – You can use the installment feature with sales custom transactions (debit type) and purchase custom transactions (credit type). Installments on sales custom transactions work the same as they do with creating the installment from an invoice. Installments on purchase custom transactions work the same as they do with vendor bills. For more information, see the help topics Creating Installments and Vendor Installment Payments.

- **Landed Costs** – You can calculate landed costs for purchase custom transactions. Landed costs work the same for purchase custom transactions as they do for vendor bills. For more information, see the help topic Landed Cost Overview.

Additionally, you can create transformation workflows to support your business transaction flows. The following transaction transformations are supported by sales and purchase custom transactions:

- Sales custom transaction type to sales custom transaction type
- Purchase custom transaction type to purchase custom transaction type
- Invoice to sales custom transaction type
- Sales custom transaction type to invoice
Sales and Purchase Transaction Types Overview

- Sales order to sales custom transaction type
- Sales custom transaction type to customer payment
- Sales custom transaction type to customer refund

For more information, see the help topic Supported Transformation Types.

To view a sample of a sales and purchase transaction workflow, see Sales Custom Transaction Transform Action Workflow Example. For more information about workflows, see the help topic SuiteFlow Overview.

Taxation for sales and purchase custom transactions is calculated the same way as it is for standard transactions. The following table shows the relationship between sales and purchasing custom transactions and the standard transactions that calculate tax the same way.

<table>
<thead>
<tr>
<th>Custom Transaction</th>
<th>Standard Transaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debit sales custom transaction</td>
<td>Invoice</td>
</tr>
<tr>
<td>Credit sales custom transaction</td>
<td>Credit</td>
</tr>
<tr>
<td>Debit purchase custom transaction</td>
<td>Bill</td>
</tr>
<tr>
<td>Credit purchase custom transaction</td>
<td>Bill Credit</td>
</tr>
</tbody>
</table>

For more information, see Custom Transaction Type Association with a SuiteTax Plug-in.

Sales and Purchase Custom Transaction Types in Integrations

Sales and Purchase Custom Transaction Types CSV Import

The CSV Import Assistant can be used to import sales and purchase custom transactions. For more information on how to import custom transactions, see the help topic Custom Transactions Import.

Sales and Purchase Custom Transaction Types in SuiteScript

Sales and purchase custom transactions are also available in SuiteScript. For more information on custom transactions through SuiteScript, see the help topic Custom Transaction.

Sales and Purchase Custom Transaction Types in SOAP Web Services

Sales and purchase custom transaction types are also available in SOAP web services. As sales and purchase custom transactions in SOAP web services differ in a few things from other custom transaction types, it is important to read through the whole topic.

You can also find more information in Sales and Purchase Transaction Types Overview.

You can find more information about Custom Transactions in SOAP web services in Custom Transaction.

Supported Operations

The following operations can be used with the sales and purchase custom transaction types:

add | addList | attach / detach | delete | deleteList | get | getCustomizationId | getDeleted | getList | getSavedSearch | getSelectValue | initialize / initializeList | search | searchMore | searchNext | update | updateList | upsert | upsertList |
Note: You can also use the asynchronous equivalents of SOAP web services list operations. For information about asynchronous operations, see the help topic SOAP Web Services Asynchronous Operations. For more information about request processing, see the help topic Synchronous Versus Asynchronous Request Processing.

Field Definitions

The SuiteTalk Schema Browser includes definitions for all body fields, sublist fields, search filters, and search joins available to this record. For details, see the Schema Browser's CustomSale and CustomPurchase reference pages.

Note: For information on using the SuiteTalk Schema Browser, see the help topic SOAP Schema Browser.

Usage Notes

Consider the following information when you work with sales and purchase custom transactions:

- When working with the add, update and upsert operations you must specify a custom transaction type in the tranType field. You can use either the scriptId or the internalId of the specific custom transaction type. Sales and purchase custom transaction types have dedicated classes customSale and customPurchase respectively.
- For the get, getList, delete, deleteList and attach / detach operations, you must use CustomTransactionRef.
- When using any search operation, you must search as for any other custom transaction and then set the criteria to sale or purchase.
- With the initialize operation you can do the following transformations:
  - CustomSale transaction type to a different customSale transaction type.
  - CustomSale transaction type to customerPayment.
  - CustomSale transaction type to customerRefund.
  - Invoice to customSale transaction type.
  - SalesOrder to customSale transaction type.
  - CustomPurchase transaction type to a different customPurchase transaction type.
- When using the get operation for a record created through the initialize operation, the returned record does not contain the createdFrom field.

Note: The createdFrom field is usually returned with a record created by the initialize operation and contains the internal id of the source record specified by the element reference.

Code Sample

In the following example the add operation is used for a sales custom transaction.

Java

```java
CustomSaleItem item = new CustomSaleItem();
item.setItem(createRecordRef("39"));

CustomSaleItemList itemList = new CustomSaleItemList();
```
itemList.setItem(new CustomSaleItem[] {item});

CustomSale record = new CustomSale();
record.setTranType(createRecordRef("105"));
record.setEntity(createRecordRef("84"));
record.setLocation(createRecordRef("2"));
record.setItemList(itemList);
c.addRecord(record);

SOAP Request

```xml
<add xmlns="urn:messages_2019_2.platform.webservices.netsuite.com">
  <record xsi:type="ns7:CustomSale" xmlns:ns7="urn:customization_2019_2_setup.webservices.netsuite.com">
    <ns7:tranType internalId="105" xsi:type="ns8:RecordRef" xmlns:ns8="urn:core_2019_2.platform.webservices.netsuite.com"/>
    <ns7:entity internalId="84" xsi:type="ns9:RecordRef" xmlns:ns9="urn:core_2019_2.platform.webservices.netsuite.com"/>
    <ns7:location internalId="2" xsi:type="ns10:RecordRef" xmlns:ns10="urn:core_2019_2.platform.webservices.netsuite.com"/>
    <ns7:itemList replaceAll="false" xsi:type="ns7:CustomSaleItemList">
      <ns7:item xsi:type="ns7:CustomSaleItem">
        <ns7:item internalId="39" xsi:type="ns11:RecordRef" xmlns:ns11="urn:core_2019_2.platform.webservices.netsuite.com"/>
      </ns7:item>
    </ns7:itemList>
  </record>
</add>
```

SOAP Response

```xml
<addResponse xmlns="urn:messages_2019_2.platform.webservices.netsuite.com">
  <writeResponse>
    <platformCore:status isSuccess="true" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:statusDetail/>
      <platformCore:afterSubmitFailed>false</platformCore:afterSubmitFailed>
    </platformCore:status>
    <baseRef internalId="176" type="customSale" xsi:type="platformCore:RecordRef" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com"/>
  </writeResponse>
</addResponse>
```

In the following example the initialize operation is used to transform a purchase custom transaction type 76 to a purchase custom transaction type 105.

Java

```java
InitializeRef reference = new InitializeRef();
reference.setInternalId("76");
reference.setType(InitializeRefType.customPurchase);
InitializeAuxRef auxReference = new InitializeAuxRef();
```
auxReference.setType(InitializeAuxRefType.tranType);
auxReference.setScriptId("custompurchase105"); // or auxReference.setInternalId("105");

InitializeRecord initializeRecord = new InitializeRecord();
initializeRecord.setType(InitializeType.customPurchase);
initializeRecord.setReference(reference);
initializeRecord.setAuxReference(auxReference);

CustomPurchase recordToAdd = (CustomPurchase) c.initialize(initializeRecord);

SOAP Request

```xml
<initialize xmlns="urn:messages_2019_2.platform.webservices.netsuite.com">
  <initializeRecord>
    <ns7:type xmlns:ns7="urn:core_2019_2.platform.webservices.netsuite.com">customPurchase</ns7:type>
    <ns8:reference internalId="76" type="customPurchase" xmlns:ns8="urn:core_2019_2.platform.webservices.netsuite.com"/>
    <ns9:auxReference scriptId="custompurchase105" type="tranType" xmlns:ns9="urn:core_2019_2.platform.webservices.netsuite.com"/>
  </initializeRecord>
</initialize>
```

SOAP Response

```xml
<readResponse>
  <platformCore:status isSuccess="true" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com"/>
  <record xsi:type="setupCustom:CustomPurchase" xmlns:setupCustom="urn:customization_2019_2.setup.webservices.netsuite.com">
    <setupCustom:createdDate>2019-09-10T07:08:00.000-07:00</setupCustom:createdDate>
    <setupCustom:lastModifiedDate>2019-09-10T07:08:00.000-07:00</setupCustom:lastModifiedDate>
    <setupCustom:tranType internalId="105" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>my Purchase B</platformCore:name>
    </setupCustom:tranType>
    <setupCustom:billAddressList internalId="38" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>Default Billing</platformCore:name>
    </setupCustom:billAddressList>
    <setupCustom:account internalId="6" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>Fees Receivable</platformCore:name>
    </setupCustom:account>
    <setupCustom:entity internalId="105" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>Acme Medical Supply</platformCore:name>
    </setupCustom:entity>
    <setupCustom:subsidiary internalId="1" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>Parent Company</platformCore:name>
    </setupCustom:subsidiary>
    <setupCustom:postingPeriod internalId="353" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>Sep 2019</platformCore:name>
    </setupCustom:postingPeriod>
    <setupCustom:tranDate>2019-09-10T00:00:00.000-07:00</setupCustom:tranDate>
  </record>
</readResponse>

Sales and Purchase Transaction Types Overview

```xml
<setupCustom:billingAddress xmlns:platformCommon="urn:common_2019_2.platform.webservices.netsuite.com">
  <platformCommon:country>unitedStates</platformCommon:country>
  <platformCommon:addresssee>Acme Medical Supply</platformCommon:addresssee>
  <platformCommon:addr1>1234 Sepulveda Blvd</platformCommon:addr1>
  <platformCommon:city>Los Angeles</platformCommon:city>
  <platformCommon:state>CA</platformCommon:state>
  <platformCommon:zip>94321</platformCommon:zip>
  <platformCommon:addrText>Acme Medical Supply1234 Sepulveda BlvdLos Angeles CA 94321</platformCommon:addrText>
</setupCustom:billingAddress>
<setupCustom:exchangeRate>1.0</setupCustom:exchangeRate>
<setupCustom:dueDate>2019-09-10T00:00:00.000-07:00</setupCustom:dueDate>
<setupCustom:paymentHold>false</setupCustom:paymentHold>
<setupCustom:memo>tralala memo</setupCustom:memo>
<setupCustom:currency internalId="1" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
  <platformCore:name>US USD</platformCore:name>
</setupCustom:currency>
<setupCustom:transactionNumber>To Be Generated</setupCustom:transactionNumber>
<setupCustom:expenseList>
  <setupCustom:expense>
    <setupCustom:line>1</setupCustom:line>
    <setupCustom:category internalId="2" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>Entertainment</platformCore:name>
    </setupCustom:category>
    <setupCustom:account internalId="59" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>Advertising</platformCore:name>
    </setupCustom:account>
    <setupCustom:amount>123.0</setupCustom:amount>
    <setupCustom:isBillable>false</setupCustom:isBillable>
  </setupCustom:expense>
</setupCustom:expenseList>
<setupCustom:accountingBookDetailList xmlns:platformCommon="urn:common_2019_2.platform.webservices.netsuite.com">
  <platformCommon:accountingBookDetail>
    <platformCommon:accountingBook internalId="2" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>Secondary accounting book</platformCore:name>
    </platformCommon:accountingBook>
    <platformCommon:currency internalId="1" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>US USD</platformCore:name>
    </platformCommon:currency>
  </platformCommon:accountingBookDetail>
  <platformCommon:accountingBookDetail>
    <platformCommon:accountingBook internalId="3" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>Third accounting book</platformCore:name>
    </platformCommon:accountingBook>
    <platformCommon:currency internalId="1" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
      <platformCore:name>US USD</platformCore:name>
    </platformCommon:currency>
  </platformCommon:accountingBookDetail>
</setupCustom:accountingBookDetailList>
```

SuiteBuilder

Oracle NetSuite
Custom Transaction Type Setup

For help getting started with the Custom Transactions feature, see the following topics:

- Enabling the Custom Transactions Feature (required)
- Granting a Role Permission to Manage Custom Transaction Types (optional)

After you have enabled the feature, authorized users can create custom transaction types, as described in Creating and Editing Custom Transaction Types. After custom transaction types have been created, authorized users can enter transaction instances by going to Customization > Lists, Records, & Fields > Transaction Types, and clicking the New Transaction link for the desired transaction type.

If appropriate, you can also configure other menu paths for entering custom transactions. This process is described in Creating Links for a Custom Transaction Type.

Enabling the Custom Transactions Feature

Before you can create custom transaction types, you must enable the feature. To enable the feature, go to Setup > Company > Enable Features. On the SuiteCloud subtab, check the Custom Transactions box, and click Save.

After you enable the feature, you can begin creating custom transaction types, as described in Creating a Custom Transaction Type.

Granting a Role Permission to Manage Custom Transaction Types

By default, only account administrators have permission to create custom transaction types. However, account administrators can grant access to other roles. The available access levels are described in the following table.

<table>
<thead>
<tr>
<th>Level</th>
<th>Users can:</th>
<th>Users cannot:</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>View transaction type definitions.</td>
<td>Edit, create, or delete transaction types.</td>
</tr>
<tr>
<td>Create</td>
<td>View and create transaction types.</td>
<td>Edit or delete transaction types.</td>
</tr>
<tr>
<td>Edit</td>
<td>View, create, and edit transaction types.</td>
<td>Delete transaction types.</td>
</tr>
<tr>
<td>Full</td>
<td>View, create, edit, and delete transaction types.</td>
<td></td>
</tr>
</tbody>
</table>

To grant a role permission to manage custom transaction types:

1. Go to Setup > Users/Roles > Manage Roles.
2. Locate the role you want to modify, and click Edit or Customize.
3. On the Permissions subtab, click the Setup subtab.
4. Do one of the following:
   - To grant the role access, add a line to the sublist. In the Permission column, set the list to Custom Transaction Types. In the Level column, choose the appropriate access level, and then click Add.
   - To make changes to the role's existing access, locate the Custom Transaction Types permission and edit the corresponding value in the Level column. Then click OK.
   - To remove a role's access, locate the Custom Transaction Types permission and click it to enable a series of buttons. Then click Remove.

5. Click Save.

**Note:** Giving a role permission to manage custom transaction types does not give the role permission to enter transaction instances. For information on permitting a role to work with transaction type instances, see Permissions for Custom Transaction Instances.

Creating and Editing Custom Transaction Types

After you have enabled the Custom Transactions feature, the next step is to create custom transaction types.

For help creating and updating custom transaction types, see the following topics.

- Custom Transaction Styles Overview
- Creating a Custom Transaction Type
- Editing a Custom Transaction Type
- Locked Custom Transaction Types
- Custom Transaction Type Classification Fields
- Account Field Setup for Custom Transaction Types
- Custom Fields in Custom Transaction Types
- Numbering for a Custom Transaction Type
- Account Field Setup for Custom Transaction Types
- Statuses for a Custom Transaction Type
- Creating Links for a Custom Transaction Type
- Adding Custom Forms for a Custom Transaction Type
- Permissions for Custom Transaction Instances
- Adding Translations for a Custom Transaction Type

**Note:** For help enabling the Custom Transactions feature, see Enabling the Custom Transactions Feature.

Custom Transaction Styles Overview

For each custom transaction type you create, you must choose a transaction style. The style determines how instances of the transaction type will appear and behave, how users will interact with transaction instances, and how they will affect your general ledger. After a transaction type has been saved, its style cannot be changed. For this reason, you may find it useful to review this section prior to creating your custom transaction types.

The available transaction styles include the following:
Basic

Users entering instances of a Basic style transaction type can enter either a series of credits or a series of debits. Users specify the accounts to be credited (or debited) in each transaction’s Lines sublist. The corresponding account to be adjusted for balancing purposes is defined on the transaction type’s definition. This account is sometimes called the offset account.

With this approach, the custom transaction is similar to the expense report transaction, which always debits the same predefined account. However, one unique feature of the Basic style is that you can choose whether the offset account named on the transaction type is to be credited or debited. You make this choice using the transaction type’s Credit box. When this box is checked, the offset account is credited, and the accounts that the user identifies in the Lines sublist are debited. When the box is cleared, the reverse behavior is used.

After a transaction type has been created, the value of the Credit check box cannot be changed.

Journal

The Journal transaction style lets users record sets of debits and credits to accounts they specify when entering the transaction. As with a standard journal entry record, the total value of credits must equal the total value of debits.

Line item entries on journal style custom transactions are treated as main line entries. This behavior is consistent with the behavior of standard journal entry transaction types.

Journal style custom transactions allow for postings to elimination subsidiaries. Basic and Header only styles do not allow for postings to elimination subsidiaries.
Header Only

With the Header Only transaction style, the system uses a Custom GL Lines plug-in implementation to determine the GL impact of transactions that users enter. That is, the transaction instances do not include a Lines sublist for users to manually enter debits or credits to specific accounts. Rather, the plug-in implementation calculates the impact based on other data that users enter. This data can come from standard or custom fields on the transaction instance entry form. It can also come from values entered in a custom UI created by using SuiteScript objects.

For example, you may want to create a Header Only transaction that uses SuiteScript objects to create a sublist based on a saved search. With this approach, you could configure the transaction type to display a list of transactions on which the system can take some action, depending on choices the user makes when entering each transaction instance. The plug-in implementation could then create GL lines based on those choices.

The following screenshot illustrates this approach. In this example, a Bad Debt Accrual transaction instance shows a list of overdue invoices. The user can select those invoices which should be designated as bad debt.

Note: When using custom transactions with the Custom GL Lines Plug-in, a posting must have at least one transaction line.

A Header Only transaction instance has no GL impact until it is associated with a plug-in implementation. However, you can create Header Only transaction types prior to enabling the Custom GL Lines feature. Similarly, users can enter Header Only transaction instances before the corresponding transaction type is linked to a plug-in implementation, as long as the type does not have a posting status.

For more information, see the following topics.

- Custom GL Lines Plug-in Overview
Sales

A sales custom transaction provides sales transaction functionality that you can incorporate into your sales workflow. With sales custom transactions, you can use the Item sublist and define the GL impact of the transaction. Inventory impact, taxes (SuiteTax only), and revenue recognition are also supported. The behavior of sales custom transactions is similar to invoices, cash sales, or credit memo transactions.

The following transaction transformations are supported:

- Sales transaction type to sales transaction type
- Sales order to sales transaction type
- Invoice to sales transaction type
- Sales transaction type to payment
- Sales transaction type to refund

For example, Large and Associates is a non-profit company that relies on donations. The company needs the ability to enter a cash donation in NetSuite with payment options including check or credit card. This transaction should not be recorded in accounts receivable. Users should be able to choose items on the Items subtab and choose a revenue account to post the transaction. A sales custom transaction meets these requirements.

Purchase

A purchase custom transaction provides purchase transaction functionality that you can incorporate into your purchase workflow. You can use purchase custom transactions to use the Item and Expenses sublists and define the GL impact of the transaction. Amortization, taxes (SuiteTax only), and inventory impact are also supported. The behavior of purchase custom transactions is similar to vendor bills.

For example, when Seven Company in Brazil imports goods, the company is responsible for issuing a Nota Fiscal to be able to carry the goods from customers at the port of arrival, to its own premises. A purchase custom transaction can meet these requirements.

Creating a Custom Transaction Type

When you create a custom transaction type, you must choose a name and a transaction style for the type. If appropriate, you can also define other qualities, such as the numbering style the type will use, the menu paths that will provide access to the transaction form, and more. After you save your new type, you can also add custom fields and create custom entry forms.

![Note: Custom transactions appear on the Deposits and Credits subtab of the Reconcile Bank Statement if the Show All Transaction Types in Reconciliation box is checked on the Accounting Preferences page. For more information, see the help topic General Accounting Preferences.]

To create a custom transaction type:

1. Go to Customization > Lists, Records & Fields > Transaction Types > New.
2. In the Name field, enter a name for the type. This value must be unique. As a best practice, enter a singular noun, because for certain locales the system uses the plural form of the name on the transaction type's List view.
3. Enter a value in the ID field. You use this value when scripting to instances of the transaction type. As a best practice, enter a name that begins with an underscore. If you do not enter a value, the system generates one.

4. In the Transaction Style field, specify a value for transaction style. This choice determines how users will interact with instances of this type. Your choices are:
   - **Basic** – Users entering transaction instances can use a Lines sublist to enter either a series of credits or a series of debits to accounts that they specify. With this style, you must also identify the corresponding account to be adjusted for balancing purposes in the transaction type's Account field. For more information, see Account Field Setup for Custom Transaction Types.
   - **Journal** – Users record sets of credits and debits to accounts that they manually specify when entering the transaction. This is the only style that allows for postings to elimination subsidiaries.
   - **Header only** – With this style, users do not manually identify accounts to credit or debit. Instead, this style relies on a Custom GL Lines plug-in implementation to calculate the GL impact.
   - **Sales** – Provides functionality similar to a sales transaction, including the Item sublist, inventory impact, taxes (SuiteTax only), and revenue recognition. Sales custom transactions provide functionality similar to invoice, cash sale or credit memo transactions.
   - **Purchase** – Provides functionality similar to a purchase transaction, including the Item and Expenses sublists, amortization, taxes (SuiteTax only), and inventory impact. Purchase custom transactions provide functionality similar to vendor bills.

For more details on the available transaction styles, see Custom Transaction Styles Overview.

**Important:** After you save your custom transaction type, the Transaction Style cannot be changed.

5. If appropriate, check the Allow Void Transactions Using Reversing Journals box. This option enables users to create reverse journal entries for posting transactions. This box is active only when the global Void Transactions Using Reversing Journals preference is enabled. You can view and set the global preference at Setup > Accounting > Accounting Preferences, on the General subtab.

**Note:** For sales and purchase transaction types, users can only void transactions that do not use inventory items.

6. If appropriate, use the Class, Department, and Location lists to specify that these fields appear on instances of this transaction type. For each field, you can specify that it is displayed either on the transaction header or as a column in the Lines sublist. For each classification, you can also check the corresponding Mandatory box to make the field required. For more information about these fields, see Custom Transaction Type Classification Fields.

**Note:** The Class, Department, and Location fields are not available for sales and purchase transaction types because they are available under the same conditions as standard sales and purchase transactions. You can specify class, department, and location by customizing the form for the sales or purchase custom transaction.

7. Specify any custom segments as appropriate. For more information, see Custom Segments.

8. If you chose a Transaction Style of Basic, Sales, or Purchase, click the Accounting subtab and configure the account details for the type. For more information, see Account Field Setup for Custom Transaction Types.

9. By default, a new custom transaction type has a non-posting status. If you want instances of the transaction to post, navigate to the Statuses subtab and do one of the following:
■ Check the **Posting** box. With this choice, every instance of the transaction posts when it is saved.

■ Create statuses for the transaction type. Each status can be configured to be either posting or non-posting. For help creating statuses, see **Statuses for a Custom Transaction Type**.

10. (Optional) Use any of the following subtabs to further define your custom transaction type:

   ■ **Document Numbers** – Configure external numbers for transaction types. For more information, see **Numbering for a Custom Transaction Type** and **Defining Numbering for a Custom Transaction Type**.

   ■ **Transaction Numbers** – Configure auto-generated transaction numbering for the type. For more information, see **Numbering for a Custom Transaction Type** and **Records and Transactions Available for Auto-Numbering**.

   ■ **Statuses** – Create statuses for the type. For more information, see **Statuses for a Custom Transaction Type**.

   ■ **Links** – Create menu paths for the transaction type. For more information, see **Creating Links for a Custom Transaction Type**.

   ■ **Permissions** – Choose the roles that should be permitted to work with instances of this custom transaction type. For more information, see **Permissions for Custom Transaction Instances**.

   ■ **Translation** – Define translations for the custom transaction type's name. This subtab appears only if the Multi-Language option is enabled at Setup > Company > Enable Features, on the Company subtab. For more information, see **Adding Translations for a Custom Transaction Type**.

11. Click **Save**.

   The system saves your new custom transaction type and assigns full-level permission to the role of the user who created the custom transaction type. The permission lets users search for custom transaction instances in the global search.

   NetSuite also creates an advanced template to print custom transaction instances.

   After you save, two additional subtabs are available to further define the custom transaction type, as follows:

   ■ **Custom Fields** – Create custom fields and automatically add them to the transaction type. Be aware that any custom field you create is available to be added to any standard or custom transaction type that exists in your system. For details, see **Custom Fields in Custom Transaction Types**.

   ■ **Forms** – Create custom forms for entering instances of the transaction type. For details, see **Adding Custom Forms for a Custom Transaction Type**.

You can use SuiteCloud Development Framework (SDF) to manage custom transaction types as part of file-based customization projects. For information about SDF, see the help topic **SuiteCloud Development Framework Overview**. You can use the Copy to Account feature to copy an individual custom transaction type to another of your accounts. Each custom transaction type page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic **Copy to Account Overview**.

### System Notes for Custom Transaction Type Configuration

NetSuite logs system notes for changes to custom transaction type configuration. Use the System Notes v2 (Beta) Workbook to view changes to the configuration of custom transaction types. For more information, see the help topic **System Notes v2 Workbook**.

---

**SuiteBuilder**
You can also access system notes for custom transactions types using the System Notes link on Custom Transaction Types page. For more information, see System Notes v2 for Custom Transaction Types.

Editing a Custom Transaction Type

Use the following procedure to make changes to an existing custom transaction type. Some fields, such as the Transaction Style field, cannot be changed after you have saved the type. Additionally, after transaction instances have been created, the Credit box cannot be changed.

You can change key fields such as Account, Filter Account Type, and Posting. When you edit those fields, the changed settings are not applied automatically to historical transactions. However, if you edit a historical transaction, the changes are applied and the GL impact of the transaction may change.

**Note:** In some cases, a transaction type that was installed from a bundle may be locked to editing. In these cases, you cannot directly edit the transaction type as described in this topic. However, you can still make changes to certain aspects of the transaction type's behavior. For details, see Locked Custom Transaction Types.

**To edit a custom transaction type:**

1. Go to Customization > Lists, Records & Fields > Transaction Types.
2. In the **Edit** column, click the name of the transaction type you want to edit.
3. Change any of the following values, as appropriate:
   - **Name** – This value must be unique. As a best practice, enter a singular noun. (For certain locales, the system uses the plural form of the name in the transaction type's List view.) When you make a change to the type's name, the old name is no longer used, even for existing transaction instances.
   - **Account** – The system displays the Account field only if the transaction type has a Transaction Style of Basic, Sales, or Purchase. In these cases, you can use the list to modify the offset account, which is the account to be debited or credited each time a user enters an instance of the transaction type. Any change you make to the Account field affects only those transaction instances created or edited after you save your change.
     
     The way this account is used varies depending on whether the Credit box is checked. For more information about the Account field and Credit box, see Account Field Setup for Custom Transaction Types.
   - **Allow Void Transactions Using Reversing Journals** – This option enables users to create reverse journal entries for posting transactions. To use this option, you must have enabled the global Void Transactions Using Reversing Journals preference, which is available at Setup > Accounting > Accounting Preferences.
     
     **Note:** For sales and purchase transaction types, users can only void transactions that do not use inventory items.
   - **Class, Department, and Location** – Use these fields to make changes to whether class, department, and location can be set on instances of this transaction type. Any changes you make affect existing transaction instances as well as new ones. For more details on these fields, see Custom Transaction Type Classification Fields.
4. The ID field is used to reference this transaction type during scripting. All IDs have a prefix that cannot be changed:
   - Basic, Journal, and Header Only styles have a prefix of customtransaction.
   - Purchase styles have a prefix of custompurchase.
   - Sales styles have a prefix of customsale.
To change the latter part of the ID, complete the following steps:

a. Click the **Change ID** button to display the **Change Script ID** popup
b. In the **New ID** field, enter the desired new ID.
c. Click **Save**.

5. (Optional) To further define your custom transaction type, make changes on any of the following subtabs:

- **Custom Fields** – Create custom fields and automatically add them to the transaction type (and to any transaction type). Be aware that any custom field you create is available to be added to any standard or custom transaction type that exists in your system. For more information, see [Custom Fields in Custom Transaction Types](#).
- **Document Numbers** – Configure auto-numbering for the type. For more information, see [Numbering for a Custom Transaction Type](#).
- **Transaction Numbers** – Configure auto-generated transaction numbering for the type. For more information, see [Numbering for a Custom Transaction Type](#) and [Records and Transactions Available for Auto-Numbering](#).
- **Accounting** – Configure account details for the type. The fields on the Accounting subtab apply to Basic, Sales, and Purchase Transaction Styles. In these cases, you can use the list to modify the offset account, which is the account to be debited or credited each time a user enters an instance of the transaction type. Any change you make to the Account field affects only those transaction instances created or edited after you save your change. The way this account is used varies depending on whether the Credit box is checked. For more information about the Account field and Credit box, see [Account Field Setup for Custom Transaction Types](#).
- **Statuses** – Create posting and non-posting for statuses the type. When editing existing statuses, some limits exist. For more information, see [Statuses for a Custom Transaction Type](#).
- **Links** – Create menu paths for the transaction type. For more information, see [Creating Links for a Custom Transaction Type](#).
- **Forms** – Create custom entry forms for the transaction type. For more information, see [Adding Custom Forms for a Custom Transaction Type](#).
- **Permissions** – Choose the roles that should be permitted to work with instances of this custom transaction type. For more information, see [Permissions for Custom Transaction Instances](#).
- **Translation** – Define translations for the custom transaction type’s name. This subtab is displayed only if the Multi-Language option is selected at Setup > Company > Enable Features, on the Company subtab. For more information, see [Adding Translations for a Custom Transaction Type](#).

6. Click **Save**.

### System Notes v2 for Custom Transaction Types

To access System Notes from the Custom Transaction types page, click System Notes located in the top right of the page.

Custom Transaction configuration changes are logged using System Notes v2. For more information, see the help topic [Viewing System Notes v2](#).

### Locked Custom Transaction Types

In some cases, you may be working with a custom transaction type that has been locked to editing. For example, transaction types that were imported by a bundle are sometimes locked. In these cases, you
cannot directly modify the transaction type definition. However, you can use other options in the NetSuite UI to refine aspects of how the transaction type functions in your account. For more details, see the following:

- Account Field Setup for Custom Transaction Types
- Permissions for Custom Transaction Instances
- Numbering for a Custom Transaction Type
- Adding an Existing Custom Field to a Custom Transaction Type
- Adding Custom Forms for a Custom Transaction Type
- Creating Links for a Custom Transaction Type

Custom Transaction Type Classification Fields

**Note:** Classification fields, including the Mandatory boxes, are not available for the sales and purchase custom transaction types. For sales and purchase transaction types, class, department, and location are determined automatically based on the features and preferences enabled in your account. You can use form customization to configure whether classification fields are available for sales and purchase custom transactions.

For each transaction type, you can decide whether instances of the type include fields for class, department, and location.

Your choices for each of these fields are as follows:

- **None** – (Default choice) The field is not used.
- **Header** – The field is displayed on the transaction's body.
- **Lines** – The field is a column in the transaction's Lines sublist.

For each field, you can also check a Mandatory box. If you check this box, users entering transaction instances are required to enter a value for the field.

Your choices about these fields are all reflected on the standard form for the transaction type. Your choices are also reflected in the default configuration of any custom forms you create, although you can make changes to the custom forms. For example, on a custom form, you can hide any of the classification fields, even if the transaction type was configured to make the field visible and mandatory. In this manner, you can bypass the transaction type's settings.

**Important:** Your custom transaction type may be associated with a plug-in implementation. If it is, consider the logic of the plug-in implementation before setting these fields. Specifically, the plug-in implementation may set line values for class, department, or location. If it does, you must set the Class, Department, or Location list to **Lines**. If you do not, the values provided by the plug-in are not used.

You can set values for the classification fields both when you create a custom transaction type and when you edit one. For details, see Creating a Custom Transaction Type and Editing a Custom Transaction Type.

For more information about the classification fields, see the following topics:

- Impact of Global Preferences
Creating and Editing Custom Transaction Types

- Classification Fields in Numbering
- Behavior of Classification Fields Following Edits

Impact of Global Preferences

The choices that you make on the custom transaction type for class, department, and location override your global accounting preferences. These preferences are set at Setup > Accounting > Accounting Preferences. However, if your transaction type's Mandatory setting is less restrictive than your global accounting preferences, the system displays a warning. In this case, you can dismiss the warning and save the less-restrictive setting.

Classification Fields in Numbering

If you configure your transaction type to include the Location field on the transaction header, you may also want to refer to location in the auto-numbering scheme for instances of this transaction type. Similarly, if you have a OneWorld account, you can refer to subsidiary when configuring your auto-numbering sequence. For details, see Numbering for a Custom Transaction Type.

Behavior of Classification Fields Following Edits

When you edit an existing custom transaction type, you can change the settings related to class, department, and location. Any changes you make affect existing instances of the transaction type, as well as new ones. If you make changes, note the following:

- If you move a field from the Lines sublist to the header, the new header field is populated only if all of the fields on the transaction instance's Lines sublist shared the same value. Otherwise, the field's value is not defined. If you change the location back to Lines, the former values are restored.

- If you choose to make a field mandatory that was not previously required, existing transaction instances may not have the required value. However, a user editing one of these instances is required to enter a value before being permitted to save changes.

Custom Fields in Custom Transaction Types

If appropriate, you can add custom fields to your custom transaction type. For more details, see the following sections:

- Creating a Custom Field and Adding it to a Custom Transaction Type
- Adding an Existing Custom Field to a Custom Transaction Type

Creating a Custom Field and Adding it to a Custom Transaction Type

If appropriate, you can create new custom fields and automatically add them to your custom transaction type. However, be aware of the following:

- When you create a custom field, it becomes available for all transaction types, including all standard and custom types. It is not exclusive to the custom transaction type that you are working with.
Creating and Editing Custom Transaction Types

However, custom fields created for custom transactions are hidden on standard transaction forms even if they are available for scripts and searches.

- When you add a custom field to your transaction type, you add it to all forms that exist for the type, including the standard form and any custom forms that exist. However, if appropriate, you can customize the field on your custom forms. For example, you can change the field's label, make it read-only, or hide it. For help working with custom forms, see Configuring Custom Forms for a Custom Transaction Type.

If your transaction type has been locked to editing and you want to add custom fields to it, see Adding an Existing Custom Field to a Custom Transaction Type.

**Note:** You create and maintain custom fields by using the transaction type's Custom Fields subtab. This subtab appears only when you are editing an existing transaction type. The subtab is not available when you are creating a new transaction type.

For more information, see Custom Transaction Body Fields and Custom Transaction Line Fields.

**To create a custom field and add it to your custom transaction type:**

1. Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type. The **Custom Fields** subtab should be displayed by default.
2. Do one of the following:
   - To add to the body of transaction records, click **New Body Field**, or
   - To add to the line items of transaction records, click **New Column Field**.

   The system displays a page that lets you create the new field. On the **Applies To** subtab, the custom transaction type is already selected. If you want to apply your new field to additional transaction types, you can select the types on this subtab.
3. Configure the field as appropriate. For more details on configuring a custom field, see **Creating a Custom Field**.
4. Click **Save**.

   The system saves your custom field and adds it to your transaction type.

**Adding an Existing Custom Field to a Custom Transaction Type**

Every custom transaction field you create is available to be added to all transaction types, both custom and standard types. This procedure explains how to add one of these fields to your custom transaction type.

Be aware that when you add a custom field to your transaction type, you add it to all forms that exist for the type, including the standard form and any custom forms you have created. However, if appropriate, you can customize the field on your custom forms. For example, you can change the field's label, make it read-only, or hide it. For help working with custom forms, see Configuring Custom Forms for a Custom Transaction Type.

For help creating a custom field, see **Creating a Custom Field**.

**To add an existing custom field to a custom transaction type:**

1. Do one of the following:
To add an existing body field, go to Customization > Lists, Records, & Fields > Transaction Body Fields.

To add an existing column field, go to Customization > Lists, Records, & Fields > Transaction Column Fields.

2. Click the name of the desired field.

3. On the Applies to subtab, locate the Custom Transactions box, which lists all of the existing custom transaction types. Select the name of the appropriate type.

4. Click Save.

Numbering for a Custom Transaction Type

All NetSuite transactions are auto-numbered. Auto-numbering helps you determine the sequence in which transactions were entered. Numbering also helps you find transaction instances.

All transaction types, including custom transaction types, use document numbers and transaction numbers.

Document Numbers

Numbers for document types are external numbers, such as the number that an external vendor uses on a bill, or account-based numbers such as a check number. Document numbers can also store a number for type-based transactions. Type-based numbers are available as custom columns in searches and lists.

Note: The Document Numbers subtab is not relevant for Purchase custom transaction types. Purchase custom transactions use external numbering, similar to that used for vendor bills, where users specify the number themselves. If an automated numbering sequence is needed for a purchase custom transaction type, use Transaction Numbers.

For external numbering sequences such as vendor bill, the document number is unique for each vendor, which permits the same document number to be assigned to more than one vendor. NetSuite prevents a vendor from having a duplicated document number on the same transaction type.

Document numbers are useful in transactions where the account field has special significance. These types of transactions must have numbering that is unique to the account named on the transaction. For example, if five different transactions are drawing money from the same account, then their document numbers are part of a single sequence even if they are of different types.

Document numbers may also be required because of government numbering requirements. In those cases, NetSuite customers might want to use the document numbering system to meet the government requirements and transaction numbering to satisfy their own internal needs and requirements.

Transaction Numbers

Auto-generated numbers are internal, gapless numbers that cannot be overwritten and are generated when a transaction record is saved. These internal numbers are generated for each transaction type.

If auto-generated numbering is not set up in your account, transaction numbers begin at 1. For more information about the records and transactions available for auto-numbering, see the help topic Records and Transactions Available for Auto-Numbering.
Creating and Editing Custom Transaction Types

**Important:** Historical transaction records may have duplicate internal numbering. NetSuite prohibits overwriting transaction type numbering sequences but permits the overwriting of document numbers. NetSuite does not renumber historical internal transaction numbers.

For more information about setting up auto-generated document numbers and transaction numbers, see the help topic Set Auto-Generated Numbers.

For each of your custom transaction types, you can customize the way these entry numbers are created. These options are described in Components of a Custom Transaction Type's Entry Number. You may want to review these options before configuring numbering.

The process of configuring numbering includes the following tasks:

- **Defining Numbering for a Custom Transaction Type** – Set the main values that define your numbering scheme.
- **Defining Numbering Preferences for Subsidiaries and Locations** – (Optional) Tie some aspect of numbering to subsidiary or location.

**Components of a Custom Transaction Type's Entry Number**

At a minimum, the entry number for a transaction instance includes an automatically generated number. However, you can customize the way the numbers are generated. For example, for each transaction type, you can do any of the following:

- Define a static prefix, to be used for all instances of this type.
- Add a prefix derived from the subsidiary or the location record with which the transaction instance is associated. When you choose this configuration, the system ties the numbering sequence to the transaction instance's subsidiary or location value. In this way, you can have multiple numbering sequences for each transaction type: one sequence for each subsidiary (or one sequence for each location).
- Define a static suffix, to be used for all instances of this type.

The following illustration shows a sample entry number that uses all of these options.

![Bad Debt Prefix Sub Or Loc 1001 Suffix](image)

Each component, along with information about configuring it, is described in the following table.

<table>
<thead>
<tr>
<th>Callout</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>This value corresponds with the transaction type's Prefix field. This static value is used for all instances of the transaction type. For help setting this value, see Defining Numbering for a Custom Transaction Type.</td>
</tr>
<tr>
<td>2</td>
<td>This value corresponds with the Transaction Prefix field of either the subsidiary or location record with which the transaction instance is associated. This value appears only if all of the following occur:</td>
</tr>
<tr>
<td></td>
<td>- The transaction type's Use Subsidiary or Use Location option is enabled. For help setting this field, see Defining Numbering for a Custom Transaction Type.</td>
</tr>
<tr>
<td></td>
<td>- The transaction instance includes a value in the relevant field (Subsidiary or Location).</td>
</tr>
<tr>
<td></td>
<td>- The relevant subsidiary or location record includes a value in its Transaction Prefix field. For more details, see Defining Numbering Preferences for Subsidiaries and Locations.</td>
</tr>
</tbody>
</table>
Creating and Editing Custom Transaction Types

Defining Numbering for a Custom Transaction Type

You can set most of the fields relevant to custom transaction numbering by using the following procedure.

If you decide to enable the Use Subsidiary or Use Location option as described in the second step of this procedure, afterward you may want to complete the steps described in Defining Numbering Preferences for Subsidiaries and Locations.

For information about the components of a custom transaction type's entry number, see Components of a Custom Transaction Type's Entry Number.

To define numbering for a custom transaction type:

1. Navigate to one of the pages that lets you configure numbering, as follows:
   - Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type. Click the Document Numbers subtab.
   - Go to Setup > Company > Auto-Generated Numbers. Click the Transactions subtab, and locate your transaction type in the list.

   Both of these pages offer a view into the same set of fields, so a change made on one page is visible on the other.

2. If appropriate, add a prefix tied to the transaction instance's subsidiary or location field:
   - **Use Subsidiary** – This option is available only in OneWorld accounts.
   - **Use Location** – If you choose this option, you may want to make sure that the transaction type definition is configured to include the Location field on the transaction header (not the Lines sublist). For help, see Custom Transaction Type Classification Fields. If you are using a custom entry form, you may also want to make sure that the custom form makes the Location field visible. For help with custom forms, see Adding Custom Forms for a Custom Transaction Type.

   When a subsidiary or location prefix is used, the system also uses a unique numbering sequence for transaction instances that have that prefix.

3. (Optional) Enter numbers or letters in the Prefix field. This prefix is displayed before each automatically-generated number. It also occurs before the subsidiary or location prefix, which you might have enabled in the previous step.

4. (Optional) Enter numbers or letters in the Suffix field. The suffix is displayed after each automatically-generated number.
5. If appropriate, enter a value from 1 to 20 in the Minimum Digits field. This minimum applies only to the automatically-generated number, not to the characters in the prefix and suffix. To meet the minimum for generated numbers that are too short, the system prepends zeroes to the number. For example, if Minimum Digits is set to 4, an automatically generated number of 1 becomes 0001.

6. If appropriate, enter a value in the Initial Number field. This number does not apply to sequences associated with subsidiary or location. Additionally, this number is used only if it is higher than the value in the Current Number field.

   If you do not enter a value in the Initial Number field, the system in most cases starts the sequence at 1. (There is one exception: if the Current Number field is displayed and has a value, the numbering begins with the number that follows the value in that field.)

   **Note:** The Current Number field is not displayed if you are in the process of creating a new transaction type. If the field appears, it has a value only if an instance of the transaction type exists. This read-only field reflects the value of whichever existing transaction instance has the highest entry number. In most cases, this field shows the number of the most recently created transaction instance.

7. If you want users to be able to override an automatically generated number, check the Allow Override box.

   **Warning:** Overriding transaction numbers permits users to create duplicate numbers and gaps in the sequence. Additionally, when a transaction number is overridden with a higher value, future auto-generated numbers begin at the next number. Each number can be manually set to a lower value, but auto-generated numbering always follows the highest number that has been used so far. Use this option with caution. Ensure that you understand the risks associated with overriding transaction numbers.

8. Click Save.

**Defining Numbering Preferences for Subsidiaries and Locations**

As described in Components of a Custom Transaction Type's Entry Number, you can configure transaction instance entry numbers to include a prefix defined on the subsidiary or location record.

When the entry number includes this type of prefix, the auto-numbering sequence used is specific to the unique combination of subsidiary and transaction type (or the unique combination of location and transaction type). In these cases, the initial number is taken from the subsidiary or location record. The subsidiary or location records in your account include fields where you can set initial numbers for all custom transaction types that have the Use Subsidiary or Use Location option enabled.

This procedure describes how to create both of the values specific to this configuration: the transaction prefix and the initial number.

You may have to complete these steps multiple times, depending on how many subsidiaries and locations exist.

**To define numbering preferences for a subsidiary or location:**

1. Navigate to one of the following pages, as appropriate:
   - Setup > Company > Classifications > Subsidiaries
   - Setup > Company > Classifications > Locations

   Click **Edit** for the appropriate subsidiary or location.
2. The Transaction Prefix field includes the prefix for this record. Make any changes as appropriate.

![Important:](image)
The value in the Transaction Prefix field is used for all transaction types configured to include a subsidiary or location prefix, including standard transaction types, so proceed with caution when making changes.

3. Configure the initial numbering sequence for your transaction type, for this subsidiary or location:
   a. Click the Numbering subtab. If the Numbering subtab is not displayed, you have not yet created a transaction type that has the Use Subsidiary or Use Location option enabled.
   b. In the Type column, locate the name of your transaction type.
   c. In the Initial Number field, enter the number that should be used for the first transaction instance. If you do not enter a value, the system in most cases starts the sequence at 1. There is one exception: if there is a value in the Current Number column, the numbering begins with the number that follows the value in that field.

![Important:](image)
If the Transaction Prefix field does not include a value, the system ignores the value in the Initial Number field. In these cases, the system numbers the instance as part of a sequence not specific to subsidiary or location. The initial number for this sequence is defined in the transaction type's Initial Number field. For help setting that value, see Defining Numbering for a Custom Transaction Type. Additionally, consider adding a value to the custom transaction type's Prefix field, to distinguish instances of this type from those of other transaction types.

4. If appropriate, enter a value in the Initial Number field. This number does not apply to sequences associated with subsidiary or location. Additionally, this number is used only if it is higher than the value in the Current Number field.

   If you do not enter a value in the Initial Number field, the system in most cases starts the sequence at 1.

5. Click Save.

Account Field Setup for Custom Transaction Types

The behavior of the Account field varies depending on the value of the transaction type's Credit box. If the box is checked, the offset account is credited when an instance of the transaction type posts. The accounts that the user identifies on the transaction instance's Lines sublist are debited. If the box is cleared, the reverse behavior occurs.

After a transaction instance is created, the value of the Credit box cannot be changed. However, you can change the offset account. If you do make a change to a transaction type's Account field, existing instances are not affected. If you edit existing instances, the new settings are applied and the account is changed for that specific instance.

To set up the account field for a custom transaction type, select one of the following:

- Entering the Account for a Basic Transaction Type
- Entering the Account for Sales and Purchase Transaction Types

**Entering the Account for a Basic Transaction Type**

When a transaction type has a style of basic, you must specify an offset account. The offset account shows the account to be debited (or credited) each time an instance of the transaction type is saved. You define this value by using the Account field for the transaction type.
To enter the account for a basic transaction type:

1. Navigate to one of the pages that lets you configure the Account field, by doing one of the following:
   - Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type. Then, click the Accounting subtab.
   - Go to Setup > Accounting > Accounting Preferences, and click the Items/Transaction subtab. On this page, under Accounts, the system lists several transaction types. This list includes all custom transaction types that use the Basic style.

Both of these pages provide a view of the same set of fields, so a change made on one page is visible on the other.

2. Use the Account list to select an account. This account is debited or credited each time users enter instances of this transactions type.

3. If you want this account to be credited when users enter instances of this transaction type, check the Credit box. In this case, the account named in the Account field is credited each time an instance of this transaction type is saved. The accounts that the user identifies in each transaction instance's Lines sublist are debited. If you want the reverse behavior, leave the box cleared.

   **Important:** After you create an instance of the custom transaction, the Credit box cannot be changed.

4. Click Save.

Entering the Account for Sales and Purchase Transaction Types

When the transaction type list style is sales or purchase, account information is required when you configure the custom transaction type. You can specify account settings that apply to all custom transaction instances, or you can give users the ability to enter account information when they create a transaction of that type.

To enter the account for sales and purchase transaction types:

1. If you want this account to be credited when users enter instances of this transaction type, check the Credit box. In this case, the account specified is credited each time an instance of this transaction type is saved. The accounts that the user identifies in each transaction instance's Lines sublist are debited. If you want the reverse behavior, leave the box cleared.

   **Important:** After you create an instance of the custom transaction, the Credit box cannot be changed.

2. Choose an option:
   - To allow users to specify the account when creating a transaction, check the Specify Account on Transaction box.

     The Filter Account Type, All, and Default Account fields are available only when the Specify Account on Transaction box is checked.

     1. To restrict the accounts available to the user when creating a transaction, select the account types in the Filter Account Type field. Or, to make all accounts available, check All.

     2. To specify a default account to be used for transactions of this type, select the account from the Default Account list.
To specify one account to be used for all transactions of this type, select an account from the Account list. This account is debited or credited each time users enter instances of this transaction type.

3. Click Save.

Statuses for a Custom Transaction Type

If appropriate, you can create statuses for each transaction type. You use statuses to represent the various stages of business processing required for instances of your transaction type. Advantages of this approach include the following:

- By using statuses, you can keep track of each instance's progress in the context of your required business processing.
- You can designate each status as either posting or non-posting.
- You can choose to make the Status field visible and available to users, or you can hide it.
- You can create workflows and scripting solutions that use and set the Status field.

For more information about statuses, see the following topics:

- Custom Transaction Type Statuses Overview
- Creating Statuses for Custom Transaction Types
- Modifying or Deleting a Custom Transaction Type Status
- Displaying or Hiding the Status Field for a Custom Transaction Type
- Referencing Custom Transaction Type Status in a Workflow

Custom Transaction Type Statuses Overview

As described in Statuses for a Custom Transaction Type, there are many advantages to creating statuses. However, before you create statuses, you should review the following:

- When Statuses Exist, All Instances Must Have a Status
- When Statuses Exist, the Posting Body Field is Ignored
- Statuses Are User-Facing, Even When the Status Field is Hidden

When Statuses Exist, All Instances Must Have a Status

When you create statuses for a transaction type, you also create a requirement that every instance of the transaction type have a status.

Because every transaction instance must have a status, you may want to consider how statuses should be assigned to transaction instances, both at the time they are created and at other key points. You have the following choices:

- You can expose the Status field on the transaction instance form. With this approach, a user entering or editing the transaction in the UI can manually assign a status to the instance. For more details on this choice, see Displaying or Hiding the Status Field for a Custom Transaction Type.
- You can create a workflow or SuiteScript that sets the status. For details, see Referencing Custom Transaction Type Status in a Workflow.
You can also use a combination of these approaches. However, be aware that it is preferable to have some method of assigning statuses, particularly to new transaction instances. If you do not actively assign a status to a new transaction instance, the system automatically assigns a status. Specifically, the system assigns the first status that was created.

**Note:** There is only one exception to the rule that every transaction must have a status. If instances of the transaction were entered before the statuses were created, these instances have an undefined status. They retain their undefined status unless they are opened for editing and a user saves changes. At this point, a status is assigned.

**When Statuses Exist, the Posting Body Field is Ignored**

The Statuses subtab includes a Posting box, situated above the Statuses sublist. This box indicates whether or not instances of the transaction post – but only if no statuses exist for the transaction type.

If statuses are defined for the transaction type, the system ignores the Posting box. In this case, for each transaction, the system uses the Posting value associated with the appropriate status. These values are shown in the Posting column.

**Note:** There is only one exception to this rule. If a transaction was entered before the statuses were created, in general, the posting status depends on the value of the body field. However, if the transaction is opened or edited following the creation of the statuses and then saved, a status is assigned to the transaction. After that, the system uses the Posting value associated with the status.

**Statuses Are User-Facing, Even When the Status Field is Hidden**

Users can see the status of a transaction instance by referring to a label displayed in the upper left corner of the page. This label is displayed both when the transaction instance is in edit mode and in view mode.
Although you can hide the Status field, you cannot hide the status label. For this reason, the status names you choose should be appropriate for viewing by anyone with permission to view the transaction instances.

Creating Statuses for Custom Transaction Types

Two statuses, Open and Fully Applied, are available automatically if the following conditions are met:

- The header line posts to an Accounts Receivable or Accounts Payable account
- The transaction is a posting sales or purchase transaction.

The Open and Fully Applied statuses are not searchable.

Use the following procedure to create statuses for your custom transaction types. Note that you can create no more than 24 statuses for any type.

**Important:** After an instance of a transaction type has been created, your options for changing any of the type's statuses are limited to changing its name and the translations of its name. So proceed with caution when creating statuses.

**To create statuses for a custom transaction type:**

1. Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type.
2. Click the **Statuses** subtab.
3. To create a status, add a line to the sublist:
   a. In the **Description** field, enter a name for the status.
   b. (Optional) Enter values in any of the following columns:
      - **Posting** – If you want the transaction instance to post to the GL when this status is reached, designate the status as posting. Click in the Posting column, and then check the box.
      - **Translation** – If your site uses the Multi-Language feature, enter translations for the name of the status. Click in the Translation column to display a popup window. In this window, you can enter a translation for each language your account uses. (The available languages are set at Setup > Company > Preferences > General Preferences, on the Languages subtab.) When you are finished entering translations, click **Done**.
   c. After you have entered values for all needed fields, click **OK** to add the status to the list.
4. If you want the Status field to be visible and available on transaction instances in the UI, check the Show Status Field box. For more information about this field, see Displaying or Hiding the Status Field for a Custom Transaction Type.

5. Click Save.

Modifying or Deleting a Custom Transaction Type Status

After you create statuses for your custom transaction type, you may want to make changes. Your ability to make changes varies, as follows:

- Before users have created transaction instances, you can change any field's value except ID. You can also delete statuses. Note that when you delete a status, its ID value is never reused.
- After an instance of a transaction type has been created, your options are limited to changing the name of the status and changing the translations of the name. You cannot make other changes, and you cannot delete statuses.

**Note:** You can add new statuses at any time, as described in Creating Statuses for Custom Transaction Types.

To modify or delete a status:

1. Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type.
2. Click the Statuses subtab.
3. If you want to edit an existing status, do any of the following:
   - To change the status name, edit the value in the Description column.
   - To change whether or not the status is posting, click in the Posting column and check or clear the box. If you cannot display the box, then an instance of the transaction type already exists, and this field cannot be changed.
   - To change or add translations of the status name, click in the Translation column. The system displays a popup window that lists all the existing translations. Make changes as needed. When you are finished, click Done.
4. If you want to delete a status, complete the following steps:
   a. Click in the row that represents the status you want to delete. The system displays several buttons.
   b. Click Remove. If you do not see a Remove button, then an instance of the transaction type already exists, and the status cannot be deleted.
5. Click Save.

Displaying or Hiding the Status Field for a Custom Transaction Type

If you have created statuses, you can also choose to display the Status field on transaction instances. When the Status field is displayed, it appears on the standard entry form as a list that the user can set.

Be aware that when you display the Status field, you make it available to anyone who has permission to edit instances of the transaction type using the standard entry form. However, you may want the field...
to be available only to certain people. You might have a situation where other users may need to create and edit transaction instances, but these users should not be allowed to set the Status field. In these situations, you should complete two tasks:

- Make the field available for the transaction type, as described in Configuring the Transaction Type to Include the Status Field in the UI.
- Use custom forms and permissions to refine how and for which roles the field is available, as described in Refining the Availability of the Status Field.

Configuring the Transaction Type to Include the Status Field in the UI

Before the Status field can be available to anyone in the UI, you have to configure it to be available to the transaction type. When you choose this configuration, you make the field viewable and available on the standard entry form for the type.

To configure the transaction type to include the Status field in the UI:

1. Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type.
2. Click the Statuses subtab.
3. If you want the Status field to be displayed on the transaction, check the Show Status Field box. If you want to hide the Status field, clear the box.
4. Click Save.

Refining the Availability of the Status Field

If you have configured a transaction type so that the Status field is visible, you may want to restrict access to the field for certain roles. For example, you might prefer that, for some users, the field is either hidden or read-only. To create these restrictions, use the following procedure.

To refine the availability of the status field:

1. Create a custom form, and modify the settings for the Status field:
   a. Create the form by opening the transaction type and clicking the Forms subtab. Locate the standard entry form for the type and click Customize. In response, the system displays details about a new custom entry form, which you can configure.
   b. Click the Screen Fields subtab. Locate the row that represents the Status field, and make one of the following changes:
      - To hide the field from view, clear the box in the Show column.
      - To make the field read-only, change the Display Type field by doing one of the following: To gray out and disable the field, select Disabled. To make the field's current setting display as on-screen text, choose Inline Text.
   c. Click Save to create the form.
2. Configure the role to allow access to the custom form that you just created:
   a. Open the role for editing by going to Setup > Users/Roles > Manage Roles. Identify the role you want to modify and click Edit or Customize.
b. Click the Forms subtab. The system displays a list of transactions that the role has permission to work with. This list includes one row for each of the transaction's available entry forms.

c. To remove the role's ability to use the standard entry form, locate the row that represents the form. Clear the box in the Enabled column.

d. To verify that the role has permission to use the new custom form, locate the row that represents the custom form. Make sure that the box in the Enabled column is checked.

e. Click Save.

Note: Before the role can work with any entry form for the transaction type, the role must also have permission to view or use the transaction type. For details, see Permissions for Custom Transaction Instances.

Referencing Custom Transaction Type Status in a Workflow

After you create statuses, a common next step is to create a workflow. As described in Custom Transaction Type Statuses Overview, a workflow offers a useful way to set statuses for transactions instances.

For more details on working with workflows, see Custom Transaction Types in Workflows.

You can also manipulate statuses using SuiteScript. For details on scripting with custom transaction types, see the help topic Custom Transaction in the SuiteScript Reference section.

Creating Links for a Custom Transaction Type

For each of your custom transaction types, you can create links, or menu paths, in the NetSuite UI. People in your organization can use these menu paths to access the transaction type's list view and entry form. For more details, see the following:

- Planning for Custom Transaction Type Links
- Creating Links by Editing the Transaction Type
- Creating Links by Using the Customization Menu

Planning for Custom Transaction Type Links

When you create a link to a custom transaction type, first you should consider the center that will host the link. A center is a view of NetSuite. Each role is linked to only one center. The center determines which tabs and menus display when a user logs in to NetSuite with the associated role. Before you begin creating links, you should identify which roles need to access the transaction type, and the centers that are associated with these roles. For more information on centers, see the help topic Centers Overview.

For each link you create, you specify a section, category, and label. The following illustration shows how each of these elements appears in the UI.
In the preceding illustration, the highlighted areas are defined as follows:

- 1 – A section. Sections are sometimes called tabs.
- 2 – A category
- 3 – Labels

When creating links, you can use standard NetSuite centers, sections, and categories, or you can create your own.

**Note:** Before members of a role can use links to a transaction type, the role must also have permission to view or use the transaction type. For details on granting these permissions, see Permissions for Custom Transaction Instances.

### Creating Links by Editing the Transaction Type

If appropriate, you can create links by directly editing the custom transaction type.

**Note:** In some cases, a transaction type that was installed from a bundle may be locked to editing. In these cases, you cannot directly edit the transaction type as described in this topic. However, you can still add links for the type. For details, see Creating Links by Using the Customization Menu.

**To create links for a custom transaction type:**

1. Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type.
2. Click the **Links** subtab.
3. Create a link by adding a line to the sublist, as follows:
   a. Use the **Center** list to select a center.
   b. Use the **Section** list to select the appropriate section, or tab. The available tabs vary depending on the center you selected. If you did not select a center, no tabs are listed.
   c. Use the **Category** list to select the appropriate menu category.
   d. (Optional) In the **Label** column, enter the label text you want to use. If you do not enter label text, the system uses the name of the transaction type as the label.
   e. (Optional) Enter translations for the link. Click in the **Translation** column to display a popup window. In this window, you can enter a translation for each language your account uses. (The available languages are set at Setup > Company > Preferences > General Preferences, on the Languages subtab.) When you are finished entering translations, click **Done**.
   f. (Optional) Specify where the label will appear in relation to other labels in the category. Click in the **Insert Before** column to display a list, and choose the appropriate label.
Click Save.

4. Create additional links as required.

5. Click Save.

For full details on creating and editing custom transaction types, see Creating a Custom Transaction Type and Editing a Custom Transaction Type.

Creating Links by Using the Customization Menu

If you installed your custom transaction type from a bundle and the type has been locked to editing, you cannot update the transaction type to include new links. Instead, you can use the Customization menu. With this method, you can update a custom category to include links to your custom transaction type.

For details on creating custom categories, see Creating Center Categories. For details on updating your custom category to include a link to your custom transaction type, see Creating Center Links.

Adding Custom Forms for a Custom Transaction Type

When you create a custom transaction type, the system automatically creates a standard form for the type. People in your organization can use this form to enter instances of the transaction.

The standard form can be modified in some ways. For example, you can add custom fields to the standard form. However, you may prefer to make further changes, such as defining some fields as read-only, or changing the placement of fields. If you want to make these kinds of changes, you should create a custom form for the transaction type. For details, see the following:

- Adding Custom Forms for a Custom Transaction Type
- Designating a Custom Form as Preferred

**Note:** One of the ways you can access custom forms is by using the transaction type's Forms subtab. This subtab appears only when you are editing an existing transaction type. The subtab is not available when you are creating a new transaction type.

Configuring Custom Forms for a Custom Transaction Type

When you create a custom form, you create an alternative to the standard form. With your custom form, you can do any of the following:

- Change the names, visibility, and placement of subtabs, field groups, fields, sublists, and actions.
- Change whether a field is read-only or editable.
- Add custom actions that execute client SuiteScripts.
- Associate custom code (existing client SuiteScripts) with the form.
- Set the form as preferred for one or more roles.
- Choose whether to store the form with the record.
- Include a Memo field, which allows users to enter a header level description of their entries that is propagated to the GL impact. Journal entries do not have a memo header, so adding a memo header field to custom transaction types enables users to provide meaningful descriptions of accounting entries. Information entered in the Memo field for a custom transaction is put in the Memo (Main) field on transaction lines.
To add or modify a form for a custom transaction type:

1. Display a list of the existing forms by doing one of the following:
   - Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type. Then click the Forms subtab. In response, the system displays a list of the existing forms for this transaction type. The list includes the standard form for the transaction type and any custom forms that have been created.
   - Go to Customization > Forms > Transaction Forms. A list of all existing transaction forms appears.

2. Do one of the following:
   - To create a form based on the standard form for the type, click Customize beside the standard form.
   - To edit an existing custom form, click Edit.

3. Customize the form as desired. For more details on working with custom forms, see Creating Custom Entry and Transaction Forms.

4. Click Save.

Designating a Custom Form as Preferred

Every custom transaction type has a default preferred form, which is displayed for roles that do not have a preferred form set.

To set a default form for roles that do not have their own preference, use the following procedure. For help designating a form as preferred for a particular role, see Defining Preferred Forms.

To designate a custom form as preferred:

1. Display a list of the existing forms by doing one of the following:
   - Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type. Then click the Forms subtab. The system displays a list of the existing forms for this transaction type. The list includes the standard form for the transaction type and any custom forms that have been created.
   - Go to Customization > Forms > Transaction Forms. The system displays a list of all existing transaction forms.

2. Locate the form that you want to designate as preferred. In the Preferred column, check the box.

3. Click Save.

For full details on editing and creating custom transaction types, see Creating a Custom Transaction Type and Editing a Custom Transaction Type.

Permissions for Custom Transaction Instances

For each custom transaction type, you can specify which roles can work with instances of the transaction type. When users create a new custom transaction type, NetSuite automatically adds a full-level permission for the role of the current user. The permission is added so that users can use the global search to find and work with the new custom transaction type.

When you grant a role access to transaction type instances, you can choose from several access levels. These levels are described in the following table.
Creating and Editing Custom Transaction Types

<table>
<thead>
<tr>
<th>Level</th>
<th>Users can:</th>
<th>Users cannot:</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>View instances of the transaction type.</td>
<td>Edit, create, or delete transaction instances.</td>
</tr>
<tr>
<td>Create</td>
<td>View and create instances of the transaction type.</td>
<td>Edit or delete transaction instances.</td>
</tr>
<tr>
<td>Edit</td>
<td>View, create, and edit instances of the transaction type.</td>
<td>Delete transaction instances.</td>
</tr>
<tr>
<td>Full</td>
<td>View, create, edit, and delete instances of the transaction type.</td>
<td></td>
</tr>
</tbody>
</table>

There are two ways to configure permissions:

- Configuring Permissions by Editing the Custom Transaction Type
- Configuring Permissions by Editing the Role

The ability of a role to interact with custom transaction instances can be further refined by restricting the role's access to particular forms for the transaction type. For details on creating forms, see Adding Custom Forms for a Custom Transaction Type. For details on disabling a role's access to a particular form, see the help topic Setting Default Forms for Roles.

**Note:** For help giving a role permission to create and edit custom transaction types, see Granting a Role Permission to Manage Custom Transaction Types.

Configuring Permissions by Editing the Custom Transaction Type

Use this procedure to grant a role permission to work with instances of a custom transaction type. You can also use these steps to modify or remove the role's access.

**Note:** In some cases, a transaction type that was installed from a bundle may be locked to editing. In these cases, you cannot directly edit the transaction type as described in this topic. However, you can still configure permissions for the type. For details, see Configuring Permissions by Editing the Role.

To grant a role access by editing the custom transaction type:

1. Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type.
2. Click the Permissions subtab. If any roles have previously been granted access to this transaction type, they are listed on this subtab.
   As of 2019.2, the role of the user who created the custom transaction type is automatically assigned full-level permission. The permission level lets users search for custom transaction types in the global search.
3. To grant a role access, add a line to the sublist, as follows:
   a. Use the Role list to select the appropriate role.
   b. By default, the Level is set to View. If you want to change this level, click in the column to display a list. Use the list to select the appropriate level.
   c. Click Add.
4. Add additional lines to the sublist to grant access to additional roles.
5. If you want to make changes to any existing role's access, edit the appropriate Level value, then click OK.
6. If you want to remove a role's access, click the name of the role to enable a series of buttons, then click **Remove**.

7. Click **Save**.

For full details on editing and creating custom transaction types, see Creating a Custom Transaction Type and Editing a Custom Transaction Type.

**System Notes v2 for Custom Transaction Types**

To access system notes from the Custom Transaction Type page, click System Notes located in the top right of the page.

Custom Transaction configuration changes are logged using System Notes v2. For more information, see the help topic Viewing System Notes v2.

**Configuring Permissions by Editing the Role**

Use this procedure to grant a role permission to work with instances of a custom transaction type. You can also use these steps to modify the role's access or remove the role's access.

If you installed your custom transaction type from a bundle and the type has been locked to editing, this method is the only one you can use to configure permissions.

**To grant a role access by editing the role:**

1. Go to Setup > Users/Roles > Manage Roles. Locate the role that needs access to the transaction type, and click the corresponding **Edit** or **Customize** link. The system displays a page that lets you edit the role.

2. Locate the Transactions subtab on the Permissions subtab, which should be displayed by default. Do one of the following:
   - If you want to grant the role access to the transaction type, add a line to the sublist:
     1. In the **Permission** column, use the list to select the appropriate transaction type.
     2. In the **Level** column, select the appropriate access level.
     3. Click **Add**.
   - If the role already has access, and you want to make a change, modify the value in the **Level** column as appropriate. Then click **OK**.
   - If the role already has access, and you want to remove the role's access, click on the name of the transaction type to enable a series of buttons. Click **Remove**.

3. Click **Save**.

**System Notes v2 for Roles and Permissions**

To access System Notes from the Role page, click System Notes located in the top right of the page.

For more information, see the help topic Viewing System Notes v2.

**Adding Translations for a Custom Transaction Type**

If your account uses the Multi-Language feature, you can add translations for your transaction type's name.
Creating and Editing Custom Transaction Types

Note: For help entering translations for link labels, see Creating Links for a Custom Transaction Type.

To add translations for a custom transaction type:

1. Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type.
2. Click the Translations subtab.
   The system displays a list of the languages configured for your account. These languages are configured at Setup > Company > Preferences > General Preferences, on the Languages subtab.
3. In the Name column, enter the appropriate translation for each language, as desired.
4. Click Save.

For full details on editing and creating custom transaction types, see Creating a Custom Transaction Type and Editing a Custom Transaction Type.

Custom Transaction Type Association with a Custom GL-Lines Plug-in Implementation

With the Custom GL Lines plug-in, you can create logic that automatically creates a GL impact. A plug-in implementation can calculate GL impact based on data that users enter in the standard or custom fields on the transaction instance entry form. GL impact can also come from values entered in a custom UI created by using SuiteScript objects. For more information, see the help topic Custom GL Lines Plug-in Overview.

To use a plug-in implementation, you can configure the implementation to apply to one or more transaction types.

You can associate any custom transaction type with a Custom GL Lines plug-in implementation. If a custom transaction type has a list style of Header Only, it must be associated with a plug-in implementation before it will have any accounting impact.

After you complete the setup, configure the plug-in the same way you would for a standard transaction type. On the configuration page for the plug-in implementation, custom transaction types are listed with standard types.
Custom Transaction Type Association with a Custom GL-Lines Plug-in Implementation

For more information, see the help topic Configure the Custom GL Lines Plug-in Implementation.

**Custom Transaction Type Association with a SuiteTax Plug-in**

With the SuiteTax plug-in, you can create logic that automatically calculates different tax amounts on individual sales or purchase custom transaction types. A plug-in implementation can calculate tax based on data that users enter in the standard or custom fields on the transaction instance entry form. Calculated tax can also come from values entered in a custom UI created by using SuiteScript objects. For more details, see the SuiteTax documentation available to SuiteTax engine providers.

To use a plug-in implementation, you can configure the implementation to apply to one or more sales or purchase transaction types. After you complete the setup, configure the plug-in the same way you would for a standard transaction type. On the configuration page for the plug-in implementation, custom transaction types are listed with standard types.

In the SuiteTax plug-in, use the hidden customtype field to obtain information about a specific sales or purchase custom transaction type. For example, you can use the customtype field to get the custom transaction script ID with the following code:

```javascript
customTransactionTypeID = customTransaction.nlapiGetFieldValue('customtype');
customTransactionType = nlapiLoadRecord('customtransactiontype', customTransactionTypeId);
customTransactionScriptId = customTransactionType.getFieldValue('scriptid');
```

**Deleting Custom Transaction Types**

In some cases, it may be necessary to delete a custom transaction type. You can delete individual custom transaction types, or you can delete custom transaction types that were installed as part of a bundle.

**Deleting Individual Custom Transaction Types**

If required, you can delete custom transaction types. However, note that you cannot delete a custom transaction type in any of the following situations:

- If instances of the type exist.
- If other dependent records exist (for example, a custom workflow that references the transaction type).
- If the transaction type was created by a bundle and was locked to editing.

For situations where deletion is not prohibited, you can use the following steps to remove the type.

**To delete a custom transaction type:**

1. Go to Customization > Lists, Records, & Fields > Transaction Types.
2. In the **Edit** column, click the name of the transaction type you want to delete.
3. Click **Actions** to display a pop-up menu, then click **Delete**.
   - The system displays a pop-up asking if you are sure you want to delete the transaction type.
4. Click **OK**.
The custom transaction type and any printing templates associated with the transaction type are deleted from your account.

Deleting Custom Transaction Types with a Bundle

Custom transaction types can be removed as part of a bundle removal or a bundle update.

Note: Deleting custom transaction types applies only to bundle operations. If a custom transaction type was created manually in the UI and a transaction instance exists, you cannot delete the transaction type.

When a bundle operation is used to delete a transaction type, the system takes the following actions with any existing instances of that transaction type:

- The transactions are migrated to a new transaction type called Deprecated Custom Transaction.
- The transactions are locked to editing, although they can still be voided.
- If the transactions are in open periods, they can be deleted.
- Body fields are added to the transaction showing:
  - The name of the custom transaction type that was deleted.
  - The status of the transaction at the time it was deleted.
- Associated printing templates are deleted.
- Custom fields are discarded unless they are related to a custom segment that has GL impact.

To see a list of transactions that were instances of deleted custom transaction types, use the Transaction search type. Set the Type field to Deprecated Transaction Type.

Creating Sales and Purchase Custom Transaction Instances

After a sales or purchase transaction type has been created, authorized users can create transaction instances.

To create a custom sales or purchase transaction instance:

1. From the menu or global search, locate and open the required sales or purchase custom transaction. For more information, contact your administrator.
2. Click New Transaction.
3. Complete the transaction instance by referring to the following sections, as required.

For more information, see the following:

- Creating a Sales Custom Transaction Instance
- Creating a Purchase Custom Transaction Instance

Creating a Sales Custom Transaction Instance

After custom sales transaction types have been created, authorized users can enter transaction instances. For more information, see the help topic Creating an Invoice.
Creating Sales and Purchase Custom Transaction Instances

To learn more about the available features, see the following subtabs:

- Items
- Promotions
- Shipping
- Billing
- SuiteTax

**Items**

On the Items subtab, you can sell items, decrease stock levels for inventory, and calculate COGS. You can apply Promotions, Discounts, and Revenue Recognition, as well as sell Gift Certificates, and more. For more information, see Sales and Purchase Functionality Available in Custom Transactions.

**Complete the following steps as needed:**

1. Click the **Items** subtab.
2. Add an item to the sales transaction.

See the following sections:

- Promotions
- Discounts
- Gift Certificates
- Revenue Recognition

**Promotions**

**Note:** The Promotion and Coupon fields appear on the Items subtab only if SuitePromotions is not enabled. For information about adding promotions using SuitePromotions, see SuitePromotions.

To apply a promotion, in the **Coupon Code** field or the **Promotion** field, enter a valid code. The remaining **Coupon Code** or **Promotion** field fills in automatically. If applicable, the **Discount Item** and **Rate** fields fill in automatically, because they are set up during promotion creation as a flat rate or a percentage. For more information, see Discounts.

Depending on the promotion setup, one of the following occurs:

- The promotion discount is calculated automatically. The Discount Item line in the Summary on the upper right corner of the page shows the promotion discount amount.
- The promotion is not calculated automatically. Click **Calculate**. The Discount Item line in the Summary on the upper right corner of the page shows the promotion discount amount.
- The promotion is added as a separate line item. The amount is deducted directly from the Subtotal line in the Summary on the upper right corner of the page.

For more information, see the help topic Review or Create a Discount Item.

For more information, see the help topic Promotions.

**Discounts**

You can apply discounts in the following ways:
Automatically – Discount items and rates are set up during the promotion creation as a flat rate or a percentage. If they apply to the sales transaction, they are automatically added to the Discount Item and Rate fields. You can optionally change these entries. The Discount Item line in the Summary on the upper right corner of the page shows the promotion discount amount.

On the Item line – After adding an item line, create a discount item code or select an existing code. This discount is applied to the item. The Rate and Amount fields might fill in automatically. You can change these entries.

When you apply a line item discount, the discount appears as a separate line item. The amount is deducted directly from the Subtotal in the Summary on the upper right corner of the page instead of appearing as a discount on the Discount Item line.

On the Items subtab, when SuitePromotions is enabled – In the Discount Item field, select a discount item. The Rate field is filled in automatically. The Discount Item line in the Summary on the upper right corner of the page shows the promotion discount amount.

Note: It is possible to apply discounts using both the line item discount and the overall order discount on the same sales transaction. If you apply discounts on line items and on the overall order, two discounts are applied. The discount on the individual item is applied first and then the overall Discount Item field discount is applied to the remaining balance owing.

For more information, see the help topic Review or Create a Discount Item.

**Gift Certificates**

Gift certificate items enable customers to purchase store credit that they can send to someone as a gift. To sell a gift certificate, you add the gift certificate as an item on a sales transaction.

**To sell a gift certificate:**

1. Enter a gift certificate item or select one from the list.
2. In the Amount field, enter the gift certificate amount.
3. Select the Gift Certificate field. In the popup window, enter details in the From, Recipient Name, Recipient Email, and Gift Message fields.
4. Click Save.

After you sell a gift certificate, you can apply it to a sales transaction. For more information about applying a gift certificate, see Gift Certificates on the Billing subtab.

For more information, see the help topic Gift Certificates.

**Revenue Recognition**

Revenue recognition features enable you to defer revenue for recognition in multiple future time periods. You can add a revenue recognition schedule on the item line, as described in the following procedure. For more information, see the help topics Revenue Recognition and Editing a Revenue Recognition Schedule.

If enabled, the system uses Advanced Revenue Management.

**To add a revenue recognition schedule:**

1. On the item line, in the Rev. Rec. Schedule field, select an existing schedule to use. If you want to create a new schedule, click New. For more information, see the help topic Defining a Revenue Recognition Template.
2. If promoted, enter the start and end dates in the Rev. Rec. Start Date and Rev. Rec. End Date fields.
3. Save the transaction.
4. Under Rev. Rec. Schedule, a schedule link replaces the template name. To open the Revenue Recognition Schedule page to see how the amounts will be split, click the link.

When you select an item and click the open icon, a page opens. The title of the page is based on the item or non-inventory item you selected. On the Revenue Recognition/Amortization subtab, configure advanced revenue recognition. For more information, see the help topic Revenue and Expense Recognition Overview.

Advanced Revenue Management

Advanced Revenue Management (ARM) automates revenue forecasting, allocation, recognition, reclassification, and auditing through a rule-based event handling framework. If ARM is enabled, you don't enter anything at the transaction. Revenue elements and arrangements are automatically created for the custom transaction when you next run Update Revenue Arrangements and Revenue Recognition Plans. For more information, see the help topic Advanced Revenue Management.

Promotions

On the Promotions subtab, you can apply SuitePromotions, shipping discounts, and more.

SuitePromotions

Note: The Promotions subtab appears only if you have SuitePromotions enabled. For information about using SuitePromotions, see the help topic Migrating to SuitePromotions.

To apply a promotion:

1. Click the Promotions subtab.
2. To automatically apply promotions, check the Automatically Apply Promotions box. All applicable promotions are applied. To clear promotions that have been automatically applied, clear the Automatically Apply Promotions box and click Clear All Lines.
3. To manually add promotions, in the Promotion field, enter a SuiteItem promotion and click Add. Repeat for any other applicable promotions.
4. To view the applied promotions on the line item list, click the Items subtab. The Discount Item line in the Summary on the upper right corner of the page shows the promotion discount amount.

Note: Depending on your discount accounting settings, the discount might be deducted directly from the item total. In this case, the promotion discount does not display as a separate item on the Discount Item line in the Summary. For more information, see the help topic Review or Create a Discount Item.

Shipping

On the Shipping subtab, you can calculate shipping costs.

To calculate shipping costs:

1. Click the Shipping subtab.
2. In the **Shipping Carrier** field, select a shipping carrier, such as FedEx or UPS.
3. In the **Shipping Method** field, select a shipping method, such as FedEx Ground UPS.
4. In the **Shipping Cost** field, to have the amount calculated automatically, click the calculator icon. Alternately, to manually set the shipping cost, enter a dollar value.

For more information, see the help topic **Shipping**.

When an eligible SuitePromotion is added, the shipping discount is applied after you calculate the shipping costs. For more information, see the help topic **Shipping Promotions and Multiple Shipping Routes**.

**Billing**

On the Billing subtab, you can apply **Gift Certificates** and **Installments**, as well as perform other billing-related tasks like setting terms, adding billing addresses, and so on.

**Gift Certificates**

You can apply a gift certificate to a custom sales transaction instance. Before you can apply a gift certificate, you must sell it. For information about selling a gift certificate, see **Gift Certificates** on the Items subtab.

**To apply a gift certificate:**

1. Click the **Billing** subtab.
2. In the Payment section **Gift Certificate** field, enter the gift certificate number or select the appropriate gift certificate number from the list.
3. The value in the **Amount Applied** field defaults to one of the following:
   - The amount of the purchased item, if the item is less than the gift certificate value.
   - The full value of the gift certificate, if the item cost is more than the gift certificate value.
4. The **Available Credit** field shows any remaining balance on the gift certificate.
5. The Gift Certificate line in the Summary on the upper right corner of the page shows the gift certificate amount applied. You can apply additional gift certificates to the same sales transaction.

For more information about gift certificates, see the help topic **Gift Certificates**.

**Installments**

You can specify installments for your customers who pay for items over time using installment-based payments.

**To set up sales custom transaction installment-based payments:**

1. Click the **Billing** subtab.
2. In the **Terms** field, select an installment-based term. An Installments subtab is added beside the Payment subtab.
3. Click the **Installments** subtab. Installment due dates and amounts are listed based on the installment terms.
4. To override the automatically generated installment amounts, check the **Override** box. Then select a date and manually enter the installment payment amount.
Note: The remaining amounts do not automatically adjust to cover any difference created by the manual adjustment. You must manually ensure the totals entered equal the amount owed. You cannot save the transaction until the installment amounts equal the total.

For more information about creating an installment term for a sales transaction, see the help topic Creating Installments.

SuiteTax

When the SuiteTax feature is enabled, you can apply taxes to sales custom transaction types.

To use SuiteTax to apply taxes to a transaction:

1. Click the Tax Details subtab.
2. Modify the values in the Tax Type, Tax Code, Tax Basis, Tax Rate, and Tax Amount columns for existing tax lines. All tax information fields are mandatory.

For more information, see the help topic Tax Details on Transactions in SuiteTax.

Creating a Purchase Custom Transaction Instance

After custom purchasing transaction types have been created, authorized users can enter transaction instances. For more information, see the help topic Vendor Bills.

To learn more about available features, see information about the following subtabs:

- Expenses and Items
- Landed Costs
- Billing
- SuiteTax

Expenses and Items

On the Expenses and Items subtab, you can purchase items, add expenses, apply Amortization and Landed Costs, and more. For more information, see Sales and Purchase Functionality Available in Custom Transactions.

Complete the following steps as needed:

1. Click the Expenses and Items subtab.
2. Click the Items subtab.
3. Add an item to the purchase transaction.

See the following Sections:

- Landed Costs
- Amortization
Landed Costs

Landed cost is the total amount paid for a product including shipping, and other additional costs.

**Note:** You can have multiple categories for custom cost categories. The following procedures use example categories of Shipping and Insurance. These categories might differ from the custom categories your company uses. For more information, see the help topic Creating Cost Categories.

**To calculate landed costs per line item on the Expenses and Items subtab:**

1. Click the Expenses and Items subtab. Then in the Items sublist, enter item details.
2. Check the Landed Cost Per Line box. Note that when this box is checked, the fields on the Landed Costs subtab are disabled.
3. Click the Landed Cost field. In the popup window, complete the following:
   a. In the Cost Category field, enter a custom cost category such as Shipping or Insurance.
   b. In the Amount field, enter a dollar amount for the cost.
   c. Click Add.
   d. Create additional cost categories as required.
   e. To finish, click OK.
4. Click Save.

You can also calculate landed costs on the Landed Costs subtab.

For more information about landed costs, see the help topic Landed Cost Overview.

Amortization

The amortization feature enables you to record expenses independently from receiving bills and making payments, so you can defer expenses and spread their impact across multiple future time periods. For more information, see the help topic Amortization Feature Overview.

**To create an amortization schedule:**

1. In the Amort. Schedule field, select a template such as Amortization Template, or select New and enter details. For more information, see the help topic Creating Amortization Templates.
2. In the Amort. Start and Amort. End fields, enter start and end dates for the amortization period.
3. Click Save.
4. A link appears in the Amortization Schedule Field. To open the Amortization Schedule page to see how the amounts will be amortized, click the link.

When you select an item and click the open icon, a page opens. The title of the page is based on the item or non-inventory item you selected. On the page, there is a Revenue Recognition/Amortization subtab. Use the options on the Revenue Recognition/Amortization subtab to configure amortization. For more information, see the help topic Revenue and Expense Recognition Overview.

Landed Costs

On the Landed Costs subtab, you can calculate landed cost by weight, quantity, or value. You can enter the cost amount manually or use the amount from the current transaction or another transaction.
To calculate Landed Costs on the Landed Costs subtab:

1. Click the Expenses and Items subtab and clear the Landed Cost Per Line box. If this box is checked, the fields on the Landed Costs subtab are unavailable.
2. On the Items subtab, enter item details.
3. In the Landed Cost Category field, enter a category for the landed cost. This category is required if the landed cost source will be anything other than Manual.
4. Click the Landed Costs subtab.
5. In the Cost Allocation Method field, select Weight, Quantity, or Value.
6. For each applicable cost field (for example, Shipping and Insurance) and source combination, choose an option:
   - In the Source field, enter Manual. Then enter a weight, quantity, or dollar value based on the cost allocation method selected.
   - In the Source field, enter This Transaction. The cost field fills in automatically with the amount calculated on the line items.
   - In the Source field, enter Other Transaction or Other Transaction (exclude tax). Then in the related Transaction field, select a transaction. The cost field fills in automatically with the amount from the selected transaction.
7. Click Save.

You can also calculate landed costs by line item on the Landed Costs subtab.

For more information about landed costs, see Landed Costs Overview.

**Billing**

On the Billing subtab, you can create Installments, as well as perform other billing-related tasks like setting terms and International Commercial Terms (incoterms), and so on.

**Installments**

You can create installments for paying bills using installment-based payments. Each installment is treated as a separate bill and has its own reference number.

**To set up purchase custom transactions for installment-based payments:**

1. Click the Billing subtab.
2. In the Terms field, select an installment-based term. An Installments subtab is added. On the Installments subtab, installment due dates and amounts are listed based on the installment terms.
3. To override the automatically generated installment amounts, check the Override box. Then select a date and manually enter the installment payment amount.

**Note:** The remaining amounts do not automatically adjust to cover any difference created by the manual adjustment. You must manually ensure the totals entered equal the amount owed. You cannot save the transaction until the installment amounts equal the total.

For information about creating an installment term for a purchase transaction, see the help topic Creating Installments.
**SuiteTax**

When the SuiteTax feature is enabled, you can apply taxes to Purchase custom transaction types.

**To use SuiteTax to apply taxes to the transaction:**

1. Click the Tax Details subtab.
2. Modify the values in the Tax Type, Tax Code, Tax Basis, Tax Rate, and Tax Amount columns for existing tax lines. All tax information fields are mandatory.

For more information, see the help topic Tax Details on Transactions in SuiteTax.

**Printing Custom Transaction Instances**

When you print custom transaction instances, advanced PDF/HTML templates are used to format the output. You can customize an advanced template for the transaction type to ensure that the information you want is included in the printout.

**Customizing the Template for Printing Custom Transactions**

After you create a custom transaction type, a standard advanced PDF/HTML template for that transaction type is created automatically. You can customize the template as required.

**To customize the advanced template:**

1. Go to . The Record Type column shows the custom transaction type ID.
2. Click Customize beside the template that you want to customize.
3. Make the required changes to the template. For information, see Customizing Advanced Templates in the Template Editor.

**Associating the Template with the Transaction Type**

After you customize an advanced template, set the custom forms for the custom transaction type to use the template. For information, see Setting Custom Forms to Use Advanced Templates.
Custom Transaction Types in Workflows

If appropriate, you can reference your custom transaction types when you create workflows. For details, see the following sections:

- Creating a Workflow for a Custom Transaction Type
- Referencing Status in a Workflow Action
- Using Status as a Workflow Condition
- Sales Custom Transaction Transform Action Workflow Example
- Using Workflows for Custom Transaction Transformations

Referencing Status in a Workflow Action

For certain types of workflow actions, you may want to reference the status of a custom transaction type. In these cases, you can identify the status by using the Parameters section of the Workflow Action page. To reference the status field, set the Field value to Custom Transaction Status.

If you are using the Set Field Value action, you can select the Static Value radio button and select the status from the Selection list. The list shows all of the available custom transaction statuses, listed according to custom transaction type. Select the appropriate combination of transaction type and status.
Using Status as a Workflow Condition

If appropriate, you can use a custom transaction type’s status as a workflow condition. On the Workflow Condition page, set Record Type to Transaction and Field to Status. (Note that this approach is different from the way you reference status in a workflow action. With that form, you set Status to Custom Transaction Status.) In the Selection field, choose the appropriate combination of transaction type and status.

Sales Custom Transaction Transform Action Workflow Example

Sales and purchase custom transaction types are available in the Transform Action in workflows. For example, we want to add a button to a sales custom transaction type for payment. Our example triggers the workflow on create, view, or update of a Sample Sales Transaction record.
The first state occurs before the record loads to add a To Payment button.
The second state specifies that when the button is clicked, the transaction is transformed to a customer payment. You could transform the transaction to a customer payment, customer refund, or a sample sales transaction.

The following screenshot shows the button on the Sample Sales Transaction record.
Using Workflows for Custom Transaction Transformations

You may want to create a custom transaction from an existing transaction type instance, or you may want to create a transaction instance from a custom transaction. You can use workflows to specify transformations with custom transactions. Use standard workflow functionality to specify when the button appears.

For example, on your Invoice page, you want to display a custom button that is available in View mode. When the user clicks the button, a defined custom transaction is created, with the Customer, Date, Period, Subsidiary, Class, Department, Location, and Items fields already filled in. The user verifies the information and clicks Save.

You can also add a custom button to the custom transaction page, for example, a Pro-forma Invoice. When the user clicks the button, a defined standard transaction is created, with the Customer, Date, Period, Subsidiary, Class, Department, Location, and Items fields already filled in. The user verifies the information and clicks Save.

**Note:** Only the Payment link is supported to transform custom transactions.

To create a custom action to perform a transformation, use the `redirect.toRecordTransform(options)` method. The method takes the source transaction, opens the form of the destination transaction and fills data in the form based on the source transaction. If a field is available in both the source and destination transactions, it is transferred. However, the following fields are not automatically filled in:

- **Date** – Today's date is displayed by default.
- **Due Date** – Calculated from the Terms using today's date.
- **Accounts** – If the main line account from the source transaction is not available in the target transaction, the default transaction account is used.
  
  You may receive a warning message that you must select a subsidiary that can use the account set for this custom transaction. This message appears if the account on the custom transaction is restricted to a certain subsidiary that is different from the subsidiary in the target transaction. If this message appears, the transaction cannot be saved.

- **Status** – The value of this field is not transferred.

For more information about the `redirect.toRecordTransform(options)` method, see the help topic `N/redirect Module`.

For more information about supported transformation types, see the help topic Supported Transformation Types.

Custom Transaction Types in Bundles

If appropriate, you can create bundles that include custom transaction types. For details, see the following topics:

- Pre-Bundling Checklist
- Including a Custom Transaction Type in a Bundle
- Other Recommended Objects
- Post-Installation Steps
- Removing Custom Transaction Types
Pre-Bundling Checklist

Before bundling your custom transaction type, you may want to review the type. In particular, check the type's settings for each of the following:

- Transaction Style
- Statuses
- Class, Department, and Location
- Custom Fields

Transaction Style

Make sure that you have chosen the most appropriate list style for your custom transaction type. For example, you might have created a transaction type for which the entire GL impact should be determined by a Custom GL Lines plug-in implementation. For these types, the list style should be Header Only. For more details on list styles, see Custom Transaction Styles Overview.

Statuses

If your transaction type includes statuses, check that the statuses are set up correctly. For example, you may want to make sure that the posting flag has been set as desired for each status. After the bundle is installed, the Statuses sublist cannot be modified in the target account. This limitation exists even if you do not select the Lock On Install option for the transaction type. You can always make changes to the type's statuses in the source account and update the bundle. However, users in the target account are never able to change existing statuses or add new statuses. For more information about statuses, see Statuses for a Custom Transaction Type.

Class, Department, and Location

Check that your custom transaction type has the desired settings for the class, department, and location fields. For help with these fields, see Custom Transaction Type Classification Fields.

Custom Fields

Check the Custom Fields subtab of the custom transaction type. This subtab lists all custom body fields and column fields that are linked to the transaction type. If any of these fields should be omitted from the bundled version of the type, remove them before creating the bundle. To remove a field, click the name of the field to open it for editing. On the Applies To subtab, cancel the selection of the custom transaction type. For more details about custom fields, see Custom Fields in Custom Transaction Types.

Including a Custom Transaction Type in a Bundle

You can add any custom transaction type to a bundle. In the Bundle Builder’s Object Types list, choose Custom Lists/Records > Transactions. In response, the system updates the Choose Objects list to include all of your custom transaction types. You can then select the appropriate types to include.
Other Recommended Objects

In some cases, you may want to consider including the following objects with your custom transaction type:

- **A Custom GL Lines plug-in implementation** – If your bundle will include a Header Only style custom transaction type, consider adding a Custom GL Lines plug-in implementation to the bundle. An instance of a Header Only transaction type has no GL impact unless the type is associated with a Custom GL Lines plug-in implementation. For more information, see the help topic Custom GL Lines Plug-in Overview.

- **A workflow** – If your transaction type includes statuses, consider adding a workflow to your bundle. You can use a workflow to set the status flag and take actions based on status values. For details on why a workflow is recommended, see Custom Transaction Type Statuses Overview. For information on creating a workflow, see the help topic SuiteFlow Overview.

Post-Installation Steps

Certain tasks cannot be managed by the bundle. These tasks must be completed by an administrator after the bundle has been installed. They include the following:

- If your bundle includes a Basic, Sales, or Purchase style transaction type, an administrator in the target account must define the offset account for the type. For more information, see Account Field Setup for Custom Transaction Types.

- If the bundle includes a Custom GL Lines plug-in implementation, an administrator in the target account must configure the plug-in implementation. For details on this process, see the help topic Configure the Custom GL Lines Plug-in Implementation.

- By default, only account administrators have permission to work with instances of any custom transaction type. If other roles require permission, an administrator must configure it. For details, see Permissions for Custom Transaction Instances.

Additionally, an administrator may want to review and customize the transaction type's numbering logic. For details, see Numbering for a Custom Transaction Type.
Removing Custom Transaction Types

When a custom transaction type is installed as part of a bundle, it can also be removed, even if instances of that transaction type exist. You can remove the type by uninstalling the bundle. Additionally, a bundle update can remove the type.

This behavior differs from how the system treats transaction types that were created in the UI. Transaction types created in the UI cannot be deleted if an instance of the type exists.

When a bundle operation is used to delete a transaction type, the system takes the following actions with any existing instances of that transaction type:

- The transactions are migrated to a new transaction type called Deprecated Custom Transaction.
- The transactions are locked to editing, although they can still be voided.
- If the transactions are in open periods, they can be deleted.
- Body fields are added to the transaction showing:
  - The name of the custom transaction type that was deleted.
  - The status of the transaction at the time the transaction type was deleted.
- Associated printing templates are deleted.
- Custom fields are discarded unless they are related to a custom segment that has GL impact.

To see a list of transactions that were instances of deleted custom transaction types, use the Transaction search type. Set the Type field to Deprecated Transaction Type.

Accounting Preferences for Custom Transactions Applied to a Bundle

Partners can create bundles for customers that include custom transactions. Setup accounts defined in custom transactions cannot be transferred, except for Default Accounts Receivable and Default Accounts Payable accounts. Customers are responsible for specifying the accounts to use after the bundle installation.

If the bundle is not locked, you can specify an account or a default account to use in the custom transaction type account settings. For more information, see Entering the Account for Sales and Purchase Transaction Types.

If the bundle is locked, you select the account or default account on the Accounting Preferences page. If the Default Account is Default Accounts Receivable or Default Accounts Payable, then neither the Account nor Default Account fields display.

To select the account:

1. Go to Setup > Accounting > Account Preferences and click the Items/Transactions sublist.
2. Under the Accounts heading, locate the Account or Default Account field for the custom transaction.
   For a Basic custom transaction type, an Account field is displayed. For Sales and Purchase custom transaction types, the field can be either Account or Default Account, depending on what is required for the custom transaction.
3. Select the required account or default account.
4. Click Save.
Custom Segments

The Custom Segments feature lets you create custom classification fields similar to class, department, and location.

This feature, also known as SuiteSegments, is part of the SuiteGL feature set. Other SuiteGL features includes Custom Transactions and Custom GL Lines. For more about SuiteGL, see the help topic SuiteGL Features Overview.

For information about the Balancing Segments feature, which depends on custom segments, see the help topic Balancing Segments and Journals.

**Important:** SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should only access the NetSuite UI by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic SuiteScript 2.0 Custom Pages.

For more details about custom segments, see the following topics:

- Benefits of Custom Segments
- Custom Segments Overview
- Permissions for Managing Custom Segments and Values
- Custom Segment Creation
- Filtering for a Custom Segment
- Custom Segment Values
- Using the Script ID to Access Custom Segment Body, Line, and Filter By Fields
- Custom Segments in Record Searches
- Customizing a Report by Using Custom Segments
- Using Custom Segments in Workflows
- Deleting a Custom Segment Definition
- Adding a Custom Segment to a Bundle

For more information about working with classifications, departments and locations, and using the custom GL lines plug-in, see the following.

- Custom GL Lines Plug-in
- Classifications Overview
- Departments and Classes Overview
- Locations Overview

**Benefits of Custom Segments**

The Custom Segments feature lets you create custom classification fields similar to class, department, and location. You can create an unlimited number of custom segments, define possible values for each segment, and add the segments to specific record types. People working in NetSuite can then use the segments to classify records appropriately.
Benefits of Custom Segments

You can configure segments to display on the GL Impact page. Additionally, standard NetSuite reports can be customized to use custom segments as filters and columns. When a segment is used as a column, it can also be grouped with other columns. For example, the following screenshot shows a customized version of the Sales by Item report. This report groups two segments in the column at the left.

In these ways and more, custom segments enhance your ability to categorize data and meet your organization's unique reporting and analysis needs.

Before you can create custom segments, you must enable the feature, as described in Enabling the Custom Segments Feature. After you have done so, you can create custom segments at Customization > Lists, Records, & Fields > Custom Segments > New. Almost all segment-configuration options can be managed from this page, including the following:

- Configure Segment Values to Be Hierarchical
- Configure a Segment to Default Statically or Dynamically
- Filter a Segment's Values Based on Other Segments
Benefits of Custom Segments

- Filter a Segment's Values Based on Class, Department, Location, and Subsidiary
- Configure a Segment to Appear on the GL Impact Page
- Create Segments as Multi-Select Fields
- Display Segments Selectively
- Custom Segments

Custom segments are part of the SuiteGL feature set. For more on SuiteGL, see the help topic SuiteGL Features Overview.

Configure Segment Values to Be Hierarchical

In the list of available segment values, you can create hierarchal relationships. For example, a segment called Sales Region might have values such as Americas, APAC, and EMEA. However, you may also want to add options that are more specific. When you create the new options, you can specify in the segment definition that they are children of the broader geographic categories. These relationships are shown in the list of values from which the user can choose.

For more details, see Creating Hierarchies Among a Custom Segment's Values.

Configure a Segment to Default Statically or Dynamically

If appropriate, you can configure a segment to default to a specific value. You have the following options:

- You can configure a segment to default to a static choice. This default is used on any record type where the segment appears, unless overridden. For example, suppose you have a segment called Sales Region, which includes the values Americas, EMEA, and APAC. If the majority of your records require a value of Americas, you could specify that Americas is the default.

- You can create dynamic defaulting for specific categories of record types, such as transactions and entities. For example, you may have a segment called Sales Region that you have applied both to the Customer record type and to all sales transactions. You can configure the segment so that, when it occurs on a sales transaction, it defaults to the value selected on the corresponding customer record. This type of defaulting overrides static defaulting.

For details on defaults, see Validation and Static Default Values for Custom Segments.

Filter a Segment's Values Based on Other Segments

In some cases, you may want to limit a segment's available values based on choices the user made for other segments on the same record.

For example, suppose your organization sells merchandise worldwide. You may have multiple sales channels, such as brick-and-mortar stores, a web store, and partner outlets. However, although you can
permit customers everywhere to use your web store, other channels may be available only in certain geographic regions.

In this case, you could create two segments, Sales Region and Sales Channel. You could configure a record's Sales Channel values to be filtered based on the selection the user makes for Sales Region. For example, you could configure a Sales Region called Denmark to permit the selection of only two Sales Channel values.

You can also set up more complex filtering. You can configure a segment's available values to be dependent on multiple segments and on other classification fields, as described in the next section.

For details on filtering, see Filtering for a Custom Segment.

Filter a Segment's Values Based on Class, Department, Location, and Subsidiary

You can set up a segment so that its values are filtered based on choices the user made in the Class, Department, Location, and Subsidiary fields. This capability is similar to the behavior described in Filter a Segment's Values Based on Other Segments.

For example, consider a company that sells clothing and accessories. This company may have a segment called Profit Center, with values such as Clothing, Shoes, and Jewelry. If the company markets products to both men and women, it may also have departments called Men's and Women's.

Both the Men's and Women's departments may sell clothing and shoes, but jewelry might be available only for women. In this case, you could configure the Profit Center segment so that its values are filtered based on the value of the Department field. A record with a Department value of Men's would show only two of the three Profit Center values.
Benefits of Custom Segments

For details on filtering, see Filtering for a Custom Segment.

Configure a Segment to Appear on the GL Impact Page

In some cases, you may want to apply segments to transactions. You can apply segments to the body of a transaction or make them columns in a transaction sublist. In both cases, you can configure these segments so that their values appear on the GL Impact page for the transactions where they are used.

![GL Impact Page](image)

For custom segments that have GL impact, the Custom GL Lines Plug-in can set and read custom segment values from custom and standard lines and from transaction records.

For more details, see Configuring GL Impact for a Custom Segment.

Create Segments as Multi-Select Fields

If appropriate, you can set up a segment so that the user can save multiple selections. This option differs from the Class, Department, and Location fields, where users can select only one option.

![Multi-Select Fields](image)

Some limitations exist for this capability. For example, if a segment is configured to display on the GL Impact page, it cannot be a multi-select field. For more details, see Custom Segment Types.

Display Segments Selectively

The Custom Segments feature includes an array of permissions options. One option is that you can choose to make a segment visible only to users in certain roles. For example, you might choose to create a segment called Rating that is applied to the Employee record type. This segment might contain sensitive information that should be available only to supervisors. In this case, you can configure the segment to be available to users in a Supervisor role but hidden for all other users. Additionally, you can choose to make the segment read-only for certain roles.

You also have many options for deciding how users can manage custom segments. For example, your account will probably have many roles that do not have permission to create or edit segments. However, if appropriate, you can give these roles permission to create and edit values for one particular segment. Note also that values can be created and updated by using the CSV Import Assistant. For details, see the help topic Custom Segment Value Import.
Benefits of Custom Segments

For details on managing permissions for segments, see Permissions for Managing Custom Segments and Values.

Set Custom Segment Values

Custom segment values can be set or created using Custom GL Lines Plug-in, SuiteScript, SOAP web services, or CSV import.

The Custom GL Lines Plug-in can set and read custom segment values from custom and standard lines and from transaction records, if the custom segment has GL impact. The default values for column segments are sourced from the body. The Custom GL Lines Plug-in can change this value, even for segments that are applied only to the body. For details, see the help topic Custom GL Lines Plug-In Interface Definition.

On record types that are exposed to SuiteScript, you can use SuiteScript to set values for custom segments that exist as fields. You can also use SuiteScript to create values for existing custom segments. For details see the help topic SuiteScript 2.0 API Reference.

In many cases, you can use SOAP web services to set values for custom segments that exist as fields. You can set values on instances of record types that are exposed to SOAP web services and have a CustomFieldList defined as part of the exposure. You can also use SOAP web services to create values for existing custom segments. For details, see the help topic CustomFieldLists for Setting Custom Segment Values.

Custom segment values can be created and updated by using the CSV Import Assistant. For details, see the help topic Custom Segment Value Import.

Custom Segments Overview

For help getting started with the Custom Segments feature, see the following topics:

- Enabling the Custom Segments Feature
- Transaction Types Supported by Custom Segments
- Permissions for Managing Custom Segments and Values

Enabling the Custom Segments Feature

Before you can create custom segments, you must enable the feature. Navigate to Setup > Company > Enable Features. On the SuiteCloud subtab, check the Custom Segments box and click Save.

After you enable the feature, you can begin creating custom segments, as described in Creating a Custom Segment.

Transaction Types Supported by Custom Segments

Custom segments are supported for the following transaction types. Support is for defining a custom segment in the body of the transaction, unless otherwise indicated.

- Assembly build
- Bad debt (transaction lines)
- Bill
- Bill (expense transaction columns)
- Bill credit
- Bill credit (expense transaction columns)
- Bin putaway worksheet
- Bin transfer
- Blanket purchase order
- Blanket purchase order (expense transaction columns)
- Budget
- Cash refund
- Cash sale
- Check
- Check (expense transaction columns)
- Commission
- Credit card
- Credit card (expense transaction columns)
- Credit card refund
- Credit memo
- Currency revaluation (Fx)
- Custom (expense transaction columns)
- Deposit application
- Deprecated custom transactions
- Estimate
- Expense report (transaction lines)
- Expense transaction columns
- Finance charge
- Inventory
- Inventory adjustment (transaction lines)
- Inventory count
- Inventory distribution
- Inventory part (transaction lines)
- Inventory transfer (transaction lines)
- Invoice
- Item receipt (expense transaction columns)
- Item receipt (transaction lines)
- Journal entry (transaction lines)
- Opportunity (transaction line)
- Other charge item (transaction line)
- Partner commission
Custom Segments Overview

- Payable Tegata
- Paycheck (derives custom segments from time entries)
- Paycheck earning (transaction lines, derived from time entries)
- Payroll adjustment
- Payroll liability check
- Period end journal
- Purchase contract
- Purchase order
- Purchase order (expense transaction columns)
- Receivable Tegata
- Request for quote
- Requisition
- Requisition (expense transaction columns)
- Return authorization
- Revenue arrangement
- Revenue contract
- Revenue commitment
- Revenue commitment reversal
- Sales tax payment
- Statement charge
- Store pickup fulfillment
- Tax liability check
- Time tracking (time transaction columns)
- Transfer order
- Vendor request for quote
- Vendor return authorization
- Vendor return authorization (expense transaction columns)
- Transfer order (transaction lines)
- Work order
- Work order (transaction lines)

Custom segments are also supported on transaction lines for the following item types.

- Apply to Kit
- Assembly/bill of materials
- Item fulfillment
- Item group
- Kit item
- Non-inventory part
- Nonmodifiable type
- Service
Settings that Affect Where Custom Segments are Applied

Be aware that certain configurations of a custom segment can affect where that segment is applied. These configurations include the following:

- Dynamic Defaults
- Filtering by Another Custom Segment
- Filtering by Class, Department, Location, or Subsidiary
- GL Impact
- Type

Dynamic Defaults

When you use dynamic defaults, you essentially create a relationship between multiple record types that use a segment. If you try to create this relationship before you have applied the segment to all of the applicable record types, the system displays a message regarding record application. This message states that the system will apply the segment to the record type referenced in the Source List choice that you have made. You can cancel out of the change, or you can let the system apply the segment to the additional record types.

For more information about sourcing, see Dynamic Default Value Sourcing for Custom Segments.

Filtering by Another Custom Segment

In some cases, you might configure a segment so that it is filtered by another segment. For example, you might have two segments, Sales Region and Sales Channel. You could configure the Sales Channel segment's values to be filtered based on the selection the user makes for Sales Region.

To achieve this type of configuration, edit the Sales Channel segment's Filtered by field. In this field, select Sales Region.

When you use this type of configuration, the segment in the Filtered by field must be applied to all of the same records as the segment that you are editing. For that reason, if you edit a segment to create a filtering relationship between it and a second segment, you might receive a message regarding record application. This message might state that the system will apply the second segment, the one named in the Filtered by list, to more record types.

Additionally, you might receive this message if you are editing a segment that is already filtered by another segment. If you apply the segment to additional record types, this change can affect the segment named in the Filtered by list. If that segment is not already applied to the new types you have selected, the system displays a message stating that it will be applied to them.
In both of these cases, you can cancel the change, or you can let the system apply the segment to the additional record types.

For more details about filtering, see Filtering for a Custom Segment.

Filtering by Class, Department, Location, or Subsidiary

In some cases, you may want to create a filtering relationship between a custom segment and the Class, Department, Location, or Subsidiary field. In this case, you edit the custom segment and select one of these classifications in the Filtered by field.

As described in Filtering by Another Custom Segment, the classifications selected in the Filtered by field must be applied to all of the same record types as the segment that you are editing. However, in the standard NetSuite configuration, Class, Department, Location, and Subsidiary do not exist on certain record types. By using custom fields, you can customize the entry forms for these record types and make these classifications available. However, custom fields cannot be used for filtering custom segments, so these custom fields are not listed in the Filtered by field. Even if you name the fields Class, Department, Location, and Subsidiary, they are not represented by the options listed in the custom segment's Filtered by field.

For this reason, in some cases when you try to filter a segment by Class, Department, Location, or Subsidiary, the system displays a warning. You are prompted to let the system change either the Filtered by field or the record application of the segment you are editing. The following limitations exist:

- Class, Department, and Location cannot be used for filtering on the record types listed on the following subtabs: CRM, Other Record Types, Custom Record Types, and Custom Segments.
- Subsidiary cannot be used for filtering on the record types listed on the Custom Record Types or Custom Segments subtabs.

GL Impact

The purpose of the GL Impact option is to add the segment to the GL Impact page for transactions where it is used. For this reason, you may receive a warning if you check the GL Impact box without applying the segment to a transaction. The GL Impact option has no function if the segment is not applied to a transaction type. However, you are not required to apply these segments to transaction types.

You set the GL Impact when you create a custom segment. After that, the option cannot be changed.

For more information about GL Impact, see Configure a Segment to Appear on the GL Impact Page.

Type

If a custom segment's Type field is set to Multiple Select, that segment cannot be applied to a transaction sublist.

For more information about custom segment types, see Custom Segment Types.

Permissions for Managing Custom Segments and Values

By default, only administrators can create and configure segments and segment values. However, you can permit other roles to complete these tasks. At a high level, you can permit a role to do the following:
- **Create and configure custom segments and their values** – To grant a role permission to view, edit, and delete custom segment definitions, you use the global Custom Segments permission. For details, see *Granting a Role Permission to Manage Custom Segments*.

- **Create and edit values for a particular custom segment** – For each segment, you can grant a role permission to create, edit, and delete the segment’s values. You grant this access by using the segment’s Value Management Access permission. For details, see *Granting a Role Permission to Manage Custom Segment Values*.

- **Use segments after they have been created, configured, and applied to records types** – To give a role this type of access, you use two permissions, each of which are segment-specific: Record Access and Search/Reporting Access. For more information, see *User Permissions for a Custom Segment*.

Note that the first two types of permissions have interdependencies. For example, if a role has some level of the Custom Segments permission, the role also has permission to view segment values, even if the role has not specifically been granted Value Management Access for any segment.

### Granting a Role Permission to Manage Custom Segments

By default, only administrators have permission to view custom segment definitions, create segments, edit segments, and delete them. However, you can assign this permission to other roles. You grant this access by using a permission called Custom Segments, which is available on the Permissions > Setup subtab of the role record.

For details, see the following sections:

- **Scope of the Custom Segments Permission**
- **Assigning the Custom Segments Permission**

### Scope of the Custom Segments Permission

The following table describes the access associated with the various levels of the Custom Segments permission. Except where noted, this table assumes that the user has no other permissions.

<table>
<thead>
<tr>
<th>Level</th>
<th>Users can:</th>
<th>Users cannot:</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>- View custom segment definitions.</td>
<td>- Create, edit, or delete custom segments.</td>
</tr>
<tr>
<td></td>
<td>- View the values defined for existing custom segments.</td>
<td>- Create, edit, or delete values on a custom segment (unless the role has the appropriate Value Management Access level for that segment).</td>
</tr>
<tr>
<td></td>
<td>- Use the Custom Segment search type.</td>
<td></td>
</tr>
<tr>
<td>Create</td>
<td>- View custom segment definitions.</td>
<td>- Edit or delete custom segments.</td>
</tr>
<tr>
<td></td>
<td>- View the values defined for existing custom segments.</td>
<td>- Create, edit, or delete values on an existing custom segment (unless the role has the appropriate Value Management Access level for that segment).</td>
</tr>
<tr>
<td></td>
<td>- Use the Custom Segment search type.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Create custom segments.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Create values for new segments that the user is creating.</td>
<td></td>
</tr>
</tbody>
</table>
Permissions for Managing Custom Segments and Values

<table>
<thead>
<tr>
<th>Level</th>
<th>Users can:</th>
<th>Users cannot:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>View custom segment definitions.</td>
<td>Delete custom segments.</td>
</tr>
<tr>
<td></td>
<td>View the values defined for existing custom segments.</td>
<td>Edit, create, or delete values on an existing custom segment (unless the role has the appropriate Value Management Access level for that segment). Note that these users have the ability to give themselves that permission. They can change the Value Management Access level on the Permissions subtab of the Custom Segments definition.</td>
</tr>
<tr>
<td></td>
<td>Use the Custom Segment search type.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Create custom segments.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Create values for new segments that the user is creating.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Edit body fields and sublist fields on an existing segment.</td>
<td></td>
</tr>
<tr>
<td>Edit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Full</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

As shown in the preceding table, even if a role does not have the Value Management Access permission for any segment, the Custom Segments permission always gives the user permission to view segment values. Similarly, the Create, Edit, and Full levels of the Custom Segments permission give the role some ability to create values, even if they do not have the Value Management Access permission.

However, if you want any of these users to be able to edit values on an existing custom segment, you may want to actively assign them the Value Management Access permission for the appropriate segments. Be aware that they can also grant themselves this permission on the Permissions subtab of the custom segment.

Assigning the Custom Segments Permission

Use the following procedure to assign the Custom Segments permission to a role.

To assign the Custom Segments permission:

1. Go to Setup > Users/Roles > Manage Roles.
2. Locate the role you want to modify, and click Edit or Customize.
3. On the Permissions subtab, click Setup.
4. Do one of the following:
   - To grant the role access, add a line to the sublist: In the **Permission** column, set the list to **Custom Segments**. In the **Level** column, select the appropriate access level. Then click **Add**.
   - To remove a role’s existing access, locate the Custom Segments permission in the sublist. Click on it to enable a series of buttons. Then click **Remove**.
   - To modify the role’s existing access, locate the Custom Segments permission in the sublist. Edit the corresponding value in the **Level** column. Then click **OK**.

5. Click **Save**.

Granting a Role Permission to Manage Custom Segment Values

In some cases, you may want to give a role permission to add values for a particular segment. This capability may be useful if you have certain users who are responsible for one segment but should not have the ability to change the values of another segment.

A user must have the Value Management Access permission to be able to edit or delete the values of an existing segment.

You grant users the ability to manage values for a segment by using the Value Management Access permission. You can configure this permission by making changes on the Permissions subtab of the custom segment definition.

For details, see the following sections:

- Required Permissions for Creating Custom Segment Values
- Scope of the Value Management Access Permission
- Assigning the Value Management Access Permission

### Required Permissions for Creating Custom Segment Values

To be able to create values, a user must have the appropriate privileges. There are several permission configurations that can result in users having this access.

Examples of users with various permission configurations are described in the following table, along with the procedures that each group can use to create values.

<table>
<thead>
<tr>
<th>Description</th>
<th>When Can the User Add Values?</th>
<th>Permitted Methods for Creating Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users assigned to the Administrator role</td>
<td>At any time</td>
<td>Creating Values Within the Segment Definition</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Creating Values by Clicking Manage Values</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Creating Values Using the New Button</td>
</tr>
<tr>
<td>Users with the Create level of the Custom Segments permission</td>
<td>Only during the time that the user is creating the segment (unless the user has additional privileges).</td>
<td>Creating Values Within the Segment Definition</td>
</tr>
<tr>
<td>Users with the following:</td>
<td>At any time</td>
<td>Creating Values Within the Segment Definition</td>
</tr>
</tbody>
</table>
Permissions for Managing Custom Segments and Values

### Description

**When Can the User Add Values?**

- The Create, Edit, or Full level of the Value Management Access permission

**Permitted Methods for Creating Values**

- Creating Values by Clicking Manage Values

**Users with both of the following:**

- The Create, Edit, or Full level of the Value Management Access permission
- Permission to work with the segment when it is displayed on another record

**At any time**

- Creating Values by Clicking Manage Values
- Creating Values Using the New Button

**Note:** Be aware that even if users have only the Edit or Full value of the Custom Segments permission, they can assign themselves the Value Management Access permission for any custom segment.

### Scope of the Value Management Access Permission

Refer to the following table for details on the privileges associated with the Value Management Access permission.

<table>
<thead>
<tr>
<th>Value Management Access Level</th>
<th>Users can:</th>
<th>Users cannot:</th>
</tr>
</thead>
<tbody>
<tr>
<td>View</td>
<td>View values for the segment. (In addition, by having the View level of the Custom Segments permission, the user can view values for all segments.)</td>
<td>Create, edit, or delete the segment's values.</td>
</tr>
<tr>
<td>Create</td>
<td>Create values for the segment for which they have the Value Management Access permission.</td>
<td>Edit or delete the segment's values.</td>
</tr>
<tr>
<td>Edit</td>
<td>Create and edit the values for the segment.</td>
<td>Delete the segment's values.</td>
</tr>
<tr>
<td>Full</td>
<td>Create, edit, and delete the values for the segment.</td>
<td></td>
</tr>
</tbody>
</table>

### Assigning the Value Management Access Permission

Use this procedure to grant a role permission to work with a custom segment's values.

**To assign the Value Management Access permission:**

1. Go to Customization > Lists, Records, & Fields > Custom Segments.
2. Locate the segment for which you want to grant access, and click **Edit**.
3. Click the **Permissions** subtab.
4. For a role to have access, the role must be referenced in the Permissions sublist. Review the sublist to see if the role is already listed, then do one of the following:
   - If the role is not listed, add a line to the sublist: In the **Role** column, select the appropriate role.
   - In the **Value Management Access Level** column, choose the desired access level. Review the values for this role in the columns labeled **Record Access Level** and **Search/Reporting Access Level**. Make any changes as needed. Then click **Add**. For details on the Record Access and Search/Reporting Access permissions, see User Permissions for a Custom Segment.

SuiteBuilder
Permissions for Managing Custom Segments and Values

- If the role is already listed but does not have the ability to manage values, then edit the role's access. Locate the role in the sublist. Edit the corresponding value in the Value Management Access Level column. Then click OK.

**Note:** If the Role has not been granted the Custom Segments permission, None is the only option available on the Value Management Access Level list. You must grant the role the Custom Segments permission before you can specify the value management access level. Go to Setup > Users/Roles > User Management > Manage Roles. On the Permissions > Setup subtab, add the Custom Segments permission to the role.

5. Click Save.

**Note:** You can also set the Value Management Access Level by editing the role. On the role record, go to the Permissions > Custom Record subtab. The names of all custom segments are displayed in the Record column.

Custom Segment Creation

After the Custom Segments feature has been enabled, authorized users can create and configure segments. For more details, see the following topics:

- Custom Segment Types
- Creating a Custom Segment
- Configuring GL Impact for a Custom Segment
- Applying a Custom Segment to Record Types
- Validation and Static Default Values for Custom Segments
- Making a Custom Segment Mandatory
- User Permissions for a Custom Segment
- Editing Custom Segments

Custom Segment Types

Every custom segment has a type. The type determines how the segment behaves when a user encounters a segment on a record and sets a value for it.

A segment’s type can be either of the following:

- **List/Record** (default) – The user can save only one selection.
- **Multiple Select** – The user can save multiple selections.

For more information about types, see the following:

- Limitations of Multiple Select
- Ramifications of Changing a Segment’s Type Field
- Changes that Can Indirectly Modify a Segment’s Type Field

Limitations of Multiple Select

If you choose a type of Multiple Select for a segment, be aware of the following limitations:
- The segment cannot have GL Impact.
- The segment cannot be applied to transaction columns.
- You cannot configure dynamic defaulting for the segment. (Dynamic defaulting is configured by using the Source List field on the Application & Sourcing subtab of the segment definition, as described in Creating a Custom Segment.)
- Multiple selects on multiple custom segments are not supported.

**Ramifications of Changing a Segment’s Type Field**

It is possible to change a segment’s type during editing of the segment. However, exercise caution when changing a segment’s type. This type of change can have the following effects:

- Changing the type from Multiple Select to List/Record can cause data loss. For example, suppose you create a segment of type Multiple Select and apply it to a record type. In this case, users may create records that store multiple values for that segment. If the segment’s type is later changed to List/Record, the saved values are discarded.

- Changing the type from List/Record to Multiple Select can affect where the segment has been applied. For example, suppose you have a segment of type List/Record that is applied to one or more transaction sublists. If you change the type to Multiple Select, the system generates a warning explaining that the segment will be removed from these transaction sublists. You have the ability to cancel out of this change, or you can proceed with changing the type and application of the segment.

**Changes that Can Indirectly Modify a Segment’s Type Field**

Be aware that in some cases, you can indirectly change a segment’s type. For example, this type of change can occur when you have a filtering relationship between two segments, one of type List/Record and one of type Multiple Select.

For example, suppose that Segment A is of type List/Record. You may decide to filter Segment A’s values by the choice a user made in another segment, Segment B, on the same record. Suppose that Segment B is of type Multiple Select.

With filtering, both segments must be applied to the same record types and transaction columns. So in this example, you are choosing to filter Segment A’s values according to Segment B. In this case, if Segment B is not already applied to the same records as Segment A, the system makes changes. It automatically updates Segment B to apply it to all of the records where Segment A has been applied.

Now, suppose that Segment A has been applied to a transaction sublist, such as Sales Item or Expense. In this scenario, when you create the filtering relationship between Segment A and Segment B, the system must apply Segment B to the transaction sublist that Segment A is associated with. However, because a segment of type Multiple Select cannot be applied to transaction sublists, the system changes Segment B’s type to List/Record. In this case, you are not warned that the type of Segment B will be changed.

For more details about filtering, see Filtering for a Custom Segment.

**Creating a Custom Segment**

When you create a custom segment, you can choose a label for the segment, define its values, and configure other settings. Of the available fields, only Label is required for the segment to be saved.

To create a segment, you must have the appropriate permissions. Authorized users include the following:
- **Administrative users** – Users who belong to the Administrator role.
- **Other users** – Users who belong to a role that has the Create, Edit, or Full level of the Custom Segments permission.

A completely defined custom segment must contain one custom record component, and one each of the following five custom field component types: Body, Column, Entity, Event, and Item. For more information, see Custom Records and Kinds of Custom Fields.

**To create a custom segment:**

2. In the **Label** field, enter a label for the segment. This text will be used as the segment’s label when it appears on records or as a transaction column.
   The name of the custom segment cannot be the same as any existing custom field name.

   ![Important: This value must be unique across all classifications in your system. Consider the following restrictions when entering a label for your custom segment.](image)

   - The label cannot be a duplicate of an existing segment label.
   - The label cannot be a duplicate of an existing custom field label.
   - If you have classifications called Class, Department, Location, and Subsidiary, you cannot use any of those words as labels.
   - You cannot name a segment Account.

3. Optionally, enter a value in the **ID** field. You can enter up to 15 characters in this field. As a best practice, enter a name that begins with an underscore. If you do not enter a value, the system generates one. Note that this value cannot be changed after the new segment is saved.

4. As of 2019.1, any new custom segments that you create automatically use the unified ID, and the **Use as Field ID** box is not visible.
   If you are editing a custom segment definition that was created before 2019.1, the **Use as Field ID** box is available.
   To use a unified ID for the entire custom segment definition, check the **Use as Field ID** box. When the box is checked, no field ID fields or columns are shown on the Application & Sourcing subtabs because one ID is used for all fields.

   ![Important: If you change the Use as Field ID setting on existing custom segments, your scripting solutions might stop working, or might not work as expected. Verify SuiteScripts, CSV imports, SOAP web services, workflows, formula fields, bundles, SDF, searches, printing templates, and any other customizations that include custom segments. SuiteAnalytics Connect does not use the unified ID.](image)

5. After the new custom segment is saved, the **Custom Record Type** field displays the custom record type associated with the custom segment. You can click the custom record type name to open the configuration page. For information see Custom Record Types Associated with a Custom Segment.

6. Optionally, change the **Type** list from its default of List/Record to Multiple Select. This choice determines whether a user setting a value for the segment can save multiple selections, as follows:
   - **List/Record** – The user can save only one selection.
   - **Multiple Select** – The user can save multiple selections.
   Some limitations exist with Multiple Select. For details, see Custom Segment Types.

7. If a custom segment is no longer needed, you can inactivate it by checking the **Inactive** box. For details, see Inactivating a Custom Segment.
8. Optionally, check the **Show In List** box to display the custom segment column on a custom record list or sublist.

![Custom Segment Creation](image)

9. In some cases, you may want to set up filtering for a segment's values. With this option, you can specify that, for any of the segment's values, the value's availability is conditional based on selections the user made in other classification fields on the same record. For full details about filtering, see [Filtering for a Custom Segment](#).

If you want to use this capability, in the **Filtered by** list, select the classifications you will use to filter this segment's available values. Hold down Ctrl to select more than one option.

![Filtered by](image)

Later, you must manage the exact filtering configuration for each value by using the **Set Filters** button, as described later in this procedure.

10. If appropriate, check the **GL Impact** box. Checking this box means that, when the segment is used on a transaction, the segment's value is displayed on the GL Impact page.

**Important**: After the segment is created, the value of the GL Impact option cannot be changed. For more details about GL impact, see [Configure a Segment to Appear on the GL Impact Page](#).

11. Enter field-level help for the segment in the **Help** field. When you enter help information, a user working with the segment can click the segment's label to display a popup window containing your help text.

12. If required, enter notes about the segment in the **Description** field. This text is visible only on this page, for people who have permission to view or edit the segment.

13. Optionally, enter additional configuration settings on the segment's subtabs, as follows:

   - **Values** – Create values for the custom segment by adding lines to the Values sublist. Note that if you made a selection in the **Filtered by** box, you must populate the **Filtering** column for each value that you want to be available to users. For full details on configuring filtering for a value, see [Setting Filters for Each of the Segment's Values](#). For more details about each column in the Values sublist, see [Creating Values Within the Segment Definition](#).

   - **Application & Sourcing** – On this subtab, you can do both of the following:
     
     - Apply the segment to one or more record types by checking the record types. You can also make the segment available on another custom segment or as a column in a transaction sublist. For more details, see [Applying a Custom Segment to Record Types](#).
     
     - If the segment applies to more than one record type, you can set up dynamic defaulting using the **Source List** field. With this approach, you can make the segment value on one record default to the segment value saved on another record. For full details on configuring dynamic defaulting, see [Dynamic Default Value Sourcing for Custom Segments](#).
- **Validation & Defaulting** – On this subtab, you can do both of the following:
  - Make the segment mandatory when it appears on a record type by checking the **Mandatory** box. Be aware, however, that a custom form for the record type can be designed to prevent the field from being visible (and therefore prevent it from being mandatory). For more details, see [Making a Custom Segment Mandatory](#).
  - Choose a static default value for the segment by using the **Default Selection** list. If you have configured dynamic defaulting using the **Application & Sourcing** subtab, the dynamic defaulting overrides the default selection value specified. For more details, see [Configuring Static Defaults for Custom Segments](#).

- **Permissions** – Grant roles permission to work with this segment. You can configure the following:
  - **Value Management Access Level** – Specify which roles can create and edit values for the segment. For details, see [Granting a Role Permission to Manage Custom Segment Values](#).
  - **Record Access Level** – Specify which roles can view and set values for a segment when it appears on a record. For details, see [Granting Roles Permission to Set Segment Values on Records](#).
  - **Search/Reporting Access Level** – Specify which roles can do the following: search based on segment values, customize reports to include segment values as columns and filters, and view segment values on custom reports. For details, see [Granting Roles Permission to Use Segments in Searches and Reports](#).

- **Dependent Segments** — When editing an existing segment, use this subtab to see a list of the segments that use the current segment for filtering their values.

  **Note:** The **Dependent Segments** subtab lists custom segments that are configured to have their values filtered by the current segment. When you create a new segment, this subtab is empty.

- **Display Order** – Specify the order in which custom segments appear in the body and lines of transactions, on other records, and on the GL Impact page, if applicable. You can set the display order to reflect the priority and dependencies of custom segments. For details see [Setting Display Order of All Custom Segments](#).

- **Translation** – Define translations for the segment's label and help text. Be aware that the Translation subtab is displayed only if the Multi-Language option is enabled, at Setup > Company > Enable Features, on the Company subtab.

14. **Click Save.**

   The custom segment and an associated custom record type are created. The custom record type has the same name as the custom segment and is available on the Custom Record Types list page. You can edit the custom record type directly to add values to the custom segment.

   You can use SuiteCloud Development Framework (SDF) to manage custom segments as part of file-based customization projects. For information about SDF, see the help topic [SuiteCloud Development Framework Overview](#). You can use the Copy to Account feature to copy an individual custom segments to another of your accounts. Each custom segment page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic [Copy to Account Overview](#).

### Configuring GL Impact for a Custom Segment

If you have applied a custom segment to transaction types or columns, the segment can have GL impact. When you enable GL Impact, segment values that are saved on transaction instances are displayed on the GL Impact page for those transactions.
If desired, you can select the GL Impact option even when the segment is not applied to transaction types or columns. However, until you apply the segment to a transaction type or column, this option has no effect.

Any user who has permission to create a custom segment can enable GL impact for that segment. However, after the segment has been created, this option cannot be modified.

For custom segment values, if a period is closed, you cannot change custom segment values that impact GL on any transactions in the period.

**GL Impact Hidden Lines**

Custom segments on GL Impact hidden lines are supported to ensure appropriate financial reporting. The hidden lines get the custom segment values from the item line or transaction body, where applicable. Hidden line support ensures that GL Impact is correctly reflected for landed cost, tax lines, and currency revaluations.

**Segment Values Available for Generated COGS and Revenue Lines**

When a sale order or sale order item custom segment has GL impact, the appropriate GL segment value is assigned to the generated COGS and revenue GL lines to ensure appropriate financial reporting. The auto-generated COGS and revenue lines get the segment value from the item line or transaction body, where applicable, even if they belong to different business transactions.

**Automatically-Generated Journals**

Custom segments are supported on automatically-generated journals that include classification, department, and location. The following automatically-generated journals include custom segments:

- Advanced Revenue Recognition Journal
- Amortization Journal
- Bill Variances Journal (SCM)
- Collect Tegata and Pay Tegata
- Entity Open Balance Journal Entry
- Intercompany Elimination Journal Entry
- Offset Journal Entry (Absolute Balance Update)
- Recognize Gift Certificate (SCIS)
- Revaluation Journal (FX Revaluation)
- Revenue Reclassification Journal
- Revenue Recognition Journal Entry
- Time Posting to Journal Entry (PSA)
- Transactions created during historical transaction processing (HTP)

You can apply custom segments to revenue arrangement and revenue element records, which are part of the Advanced Revenue Management feature.

**Important:** The GL impact setting can only be specified when you create a new custom segment. After the custom segment is saved, the GL Impact option cannot be changed.

**Examples**

You enter a standard invoice or cash sale and set or source a GL segment value on the body, line, or both. The auto-generated COGS and revenue lines on the transaction have the same segment value.
You enter a sales order with the GL segment value on the item, and then you fulfill the item. The COGS line on the GL impact page gets the segment value from the item line. When you invoice the sales order later, the revenue line on the GL Impact gets the same segment value. Both COGS and revenue lines source the GL segment value from the sales order item line unless the user manually overwrites the values.

For more information, see the help topics GL Impact Page and Cost of Goods Sold (COGS) GL Impact.

**To configure GL impact for a custom segment:**

2. Check or clear the **GL Impact** box as appropriate.
3. Click **Save**.

### Filtering for a Custom Segment

If appropriate, you can set up a segment so that, when it appears on a record, its values are filtered. Values can be filtered based on choices the user made in other classification fields on the same record.

For example, consider a company that markets entertainment media. This company has a segment called Profit Center, with values such as Books, Games, Music, and Video. The company operates in APAC, EMEA, and North America, and has Location values that represent each of those geographic areas.

Suppose that both Profit Center and Location are available on sales transactions, but for some locations, only certain Profit Centers are available. For example, all profit centers may be available in North America. However, in the other three locations, only games, music, and video are available.

In this case, you might configure the Profit Center segment so that its values are filtered based on the value selected for Location. With this configuration, you can specify that, if the selected Location is EMEA, the Books value is hidden from the list of Profit Center values.

In addition to Location, you can filter based on the values selected in the Class and Department fields, if those features are enabled. Additionally, you can filter based on choices the user made in another segment. In OneWorld accounts, you can also filter by the Subsidiary field.

Note also that you can filter by one field or by multiple fields.

For more details about filtering, see the following topics:

- Setting Up Filtering for a Custom Segment
- Removing Filtering from a Custom Segment
- Viewing a Custom Segment's Filtering Relationships
Setting Up Filtering for a Custom Segment

As described in Filtering for a Custom Segment, you can configure each segment so that its available values are determined by selections made in another classification field.

To configure filtering, at a high level, you complete two steps:

- Setting the Segment's Filtered by Field
- Setting Filters for Each of the Segment's Values

**Note:** When you are configuring filtering, be aware that filtering choices can be overridden by a static default. That is, if you configure a static default for a segment, that value can always be saved on a new record, even if your filtering configuration would otherwise make the value impermissible. If you want to avoid this behavior, avoid choosing a static default that is not permitted by your filtering configuration. For details on static defaults, see Configuring Static Defaults for Custom Segments.

Setting the Segment's Filtered by Field

To enable filtering, you edit the Filtered by field of the segment whose values are being filtered. For example, if you had a Profit Center segment whose available values depend on the selection made in the Location field, you edit the Profit Center segment. No changes are required to the other classification field (in this case, the Location field).

**To configure the segment's Filtered by field:**

1. Edit the custom segment for which you want to add filtering.
2. In the Filtered by field, select all of the classifications to use for filtering. (In the example from the beginning of this topic, you select Location.) To select more than one value, hold down the Ctrl key.

   **Note:** In some cases, you might want to select another custom segment in the Filtered by list, but that segment is not listed. In these cases, the problem might be that the other segment already has a filtering relationship with the segment you are editing. To find out, navigate to the other segment definition, and view the Dependent Segments subtab. This subtab lists all of the segments that currently filter by the segment you are viewing.

3. Click Save.

Setting Filters for Each of the Segment's Values

After the segment's Filtered by field has been configured, you must edit each of the segment's values. (In the example from the beginning of this topic, you edit the Profit Center segment's values.) If you fail to edit any value, that value will never be available for users to select. You can use either of the following approaches to edit the values:

- Editing Values Within the Segment Definition
- Editing Each Value Directly

**Important:** Make sure you edit every value, if you want every value to be available under some circumstance. If you do not set the filtering conditions for a value, that value is never displayed in the segment's list.

Editing Values Within the Segment Definition

If you have permission to edit the segment definition, you can set filters for each value by editing the Values sublist.
To edit values within the segment definition:

1. Edit the custom segment for which you want to add filtering.
2. In the Values sublist, edit each value as follows:
   a. For the value that you want to configure, click anywhere in that row to enable a series of buttons.
   b. Click Set Filters to display a popup window. The popup window includes a Filter list. Its value is the field selected in the segment's Filtered by field. Directly below the list is a list of that field's values.
   c. In the popup window, select the values of the Filter field that permit this value to be displayed as an option. For example, suppose you are editing the Book value of the Profit Center segment. If you want this value to be available only when the North America location is selected, select North America. To select more than one value, hold down the Ctrl key.
   d. If the segment's Filtered by field includes more than one selection, use the Filter list to select another classification and set its values.
   e. When you have made selections for all of the fields listed in the Filter list, click Set.
   
   ! Important: You must set values for all of the fields available in the Filter list, or the value is never available to users.
   f. Click OK.
3. If there are other values that should be available, and do not have filters set, repeat the editing steps above for that value.
4. Click Save on the segment definition.

Editing Each Value Directly

If you do not have permission to edit the segment definition, but you do have permission to edit its values, use the following procedure.

To edit each value directly:

1. If it is not already open, open the value for editing. For more information, see Editing a Custom Segment's Values by Clicking Manage Values.
2. Locate the Filter by [Classification Field] field, where [Classification Field] is the field selected in the segment's Filtered by field. This list includes all possible values of the field by which you are filtering.
3. Select the values of the classification field that should permit the user to select the value that you are editing. For example, suppose you are editing the Book value of the Profit Center segment. If you want this value to be available only when the North America location is selected, select North America. To select more than one value, hold down the Ctrl key.

4. If the page includes additional fields labeled Filtered by [Classification Field], make selections in each of these fields.

Important: You must set values for all of the Filtered by fields, or the value you are editing is never available to users.

5. Click Save.

Filtering Across Body and Line Segments

Segment values on transaction lines can be filtered by the segment value set in the body of a transaction. When you select a value for a parent segment, only child segments of that parent are available for selection on transaction lines.

For example, you have a parent custom segment called Business Unit that is available on purchases at the body level and includes a list of different areas of the business, such as grocery, bakery, and cafe.
The child custom segment, Product Line, is filtered by Business Unit and available on purchase transaction lines. Items include food, cold beverages, hot beverages, and frozen desserts. For each list value, you set filtering to specify for which business unit the item will be available.

On a purchase order, selecting a business unit at the body level filters the items available in the Product Line list at the line level, making order entry less cumbersome.

Removing Filtering from a Custom Segment

When you remove filtering from a segment, all of its values become available for selection. Use the following procedure to remove filtering from a segment.

To remove filtering from a segment:

1. Edit the custom segment that no longer requires filtering.
2. To remove filtering, clear the selections in the Filtered by field. To clear a selection, hold down the Ctrl key and click a selected field. Note that you may have to use the scroll bars to see all classifications that have been selected.
3. Click Save.
Viewing a Custom Segment’s Filtering Relationships

When working with a custom segment, you may want to know what filtering relationships it has with other segments. There are two areas on the segment definition that show this information:

- The Filtered by body field – Shows the segments that the current custom segment uses to filter its values.
- The Dependent Segment subtab – Shows the segments that use the current segment for filtering their values.

To navigate to a custom segment definition, go to Customization > Lists, Records, & Fields > Custom Segments. Locate the appropriate segment, and click Edit or View.

To view a custom segment definition, you need at least the View level of the Custom Segments permission. For details about this permission, see Granting a Role Permission to Manage Custom Segments.

Applying a Custom Segment to Record Types

For a custom segment to be available on records, you must apply the segment to one or more record types or transaction sublists. After you do, the segment is available as a field on instances of that records. Depending on where you apply the segment, it is available either as a body field or as a column in a transaction sublist. Authorized users can use the segment to classify the record or transaction line. On standard forms, the segment is displayed on the Custom subtab of the record instance.

You can also apply segments to groups of record types, in some cases, or make them columns in transaction sublists, or groups of transaction sublists. For example, you can apply a segment to all sales transactions or to all sales item sublists. You can also apply a segment to another segment.

You apply segments to record types when creating or editing a segment by using the Application & Sourcing subtab.

The options available in the Application field group of the Transactions and Transaction Columns subtabs enable you to specify the transactions and transaction columns where the custom segment is available.

For more information about the transactions where custom segments can be used, see Transaction Types Supported by Custom Segments.

**Note:** Custom segments cannot be applied to custom lists.

To apply a segment to a record type, transaction sublist, or group:

1. Edit the custom segment.
2. Click the Application & Sourcing subtab.
   
   The subtabs available from the Application & Sourcing subtab include the following:
   
   - Transactions, including the following items:
     
     - Custom transaction types
     
     - Check the Tegata box to apply the custom segment to Collect Tegata and Pay Tegata transactions
     
     - Check the Vendor Payment box to apply the custom segment to single vendor payments and bill payments to multiple vendors
   
   - Transaction columns, and custom transaction sublists
   
   - Entities
Custom Segment Creation

- CRM
- Items
- Other Record Types, including the following items:
  - Advanced revenue recognition records
  - Allocation Schedule destination line – Applies the custom segment to the destination lines on an allocation schedule
  - Allocation Schedule source line – Applies the custom segment to the source lines on an allocation schedule

For more information about allocation schedules, see Expense Allocation Overview.

- Budget Import — If a custom segment is applied to a budget record, and budget transactions exist, the custom segment cannot be inactivated, deleted, or removed from the budget record.

**Important:** To avoid data corruption, the Custom Segments feature cannot be disabled if a custom segment is applied to a budget record.

- Multi-book accounting records

**Note:** Some records are not available unless the feature is enabled. For example, the accounting book record is not available if the Multi-Book Accounting feature is not enabled.

- Custom Record Types, which includes any custom record types available in the account
- Custom Segments, which includes all custom segments available in the account

If the Use as Field ID box is checked for the custom segment, no field ID fields or columns are shown on the Application & Sourcing subtabs because one ID is used for all fields.

3. Specify where the custom segment is available by checking the appropriate boxes on each of the subtabs.
   - Some of the subtabs have an Application field group, where you specify the transactions where this custom segment is available.
   - All of the subtabs include a list of record types, where you specify the record types where the custom segment is available.

4. Click Save.

**Dynamic Default Value Sourcing for Custom Segments**

In some cases, you might apply a custom segment to two record types that have a relationship with each other. In these situations, you may want the segment value on one record to populate with the value selected on the other.

For example, in the standard NetSuite configuration, each sales transaction is associated with a particular customer record. You may want the segment value on a new sales transaction to populate with the segment value saved on the corresponding customer record. You can configure this behavior by specifying dynamic default logic.

For more information about dynamic defaults, see the following topics:

- Dynamic Defaulting for Custom Segments
- Creating Dynamic Default Logic for Custom Segments
Dynamic Defaulting for Custom Segments

In some cases, you may want to apply a custom segment to two record types that have a relationship with each other. In these cases, you may want the segment on one record to populate automatically with the value of the segment on the other. To configure this behavior, you must specify dynamic default logic for the segment.

With this approach, at a high level, you create a relationship between the following:

- One or more **target record types** – The group of record types, or single record type, whose instances receive the dynamically loaded default value.
- A field on the target record type – This field is used to identify the particular record instance that provides the default value. This record is the **source record instance**.

For more details, see the following sections:

- Source List Choices
- Choose a Target
- Prerequisites for Creation of a Dynamic Default

### Source List Choices

When you configure dynamic defaulting, you use the Source List field.

Each choice in the Source List shows two terms. The first term represents the label of a field. The second term represents a record type.

<table>
<thead>
<tr>
<th>Call out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>This term represents the name of a field on the target record types. The selection that a user makes in this field always represents another record. For example, some transaction types have a field called Sales Rep. When a value for that field is saved on a transaction, that value identifies a specific employee record.</td>
</tr>
<tr>
<td>2</td>
<td>This term represents the record type of the source record instance. To follow the example from the preceding row, a field labeled Sales Rep identifies a record of type Employee.</td>
</tr>
</tbody>
</table>

### Choose a Target

The Application & Sourcing subtab includes several child subtabs. Each of these subtabs represents a different category of record type.

When you display some of these subtabs, you will see only one Source List field. The choice you make in this field affects all of the record types selected on that subtab. When you navigate to certain other subtabs, you can select a different Source List option for those record types.

On two of the subtabs, you can specify a Source List choice for each record type listed. This option is possible for the types listed on two subtabs: Other Record Types and Custom Record Types.

### Prerequisites for Creation of a Dynamic Default

Sometimes you may save a Source List choice, but a dynamic default is not always generated for the applicable record types. For a default to be generated, the following statements must be true:
The source record instance must have a value saved in the segment field – If the segment field on the source record instance has not been populated with a value, no dynamic default is created. For example, suppose you have configured the segment on sales transactions to populate with the segment value saved on the corresponding customer record. If the corresponding customer record does not have a value saved in the segment field, no dynamic default is created. If you want to avoid this situation, you can make the segment mandatory, as described in Making a Custom Segment Mandatory.

The Source List field must exist on the entry form for the target record type – On the subtabs where one Source List is used for many record types, note that not all record types have all of the fields listed. If you select a choice in the Source List field, and one of the record types does not have that field, no default is ever created for instances of that record. For example, on the Transactions subtab, the Source List includes a choice labeled Sales Rep (Employee). If you select that option, then dynamic defaults are created for transactions that have a Sales Rep field. For example, instances of the Cash Sale record type could receive the dynamic default value. However, for transactions such as Journal Entry, which does not have a Sales Rep field, no dynamic default can be created.

A value for the Source List field must be selected on the instance of the target record type – Sometimes not every field on a record has a value saved. If an instance of the target record type is saved with no value selected for the field identified by the Source List, no default is created. For example, suppose you have configured account records to populate with a default value. In the Source List for the Account record type, you may have chosen Restrict to Location (Location). If an instance of the account record does not have a value selected in the Restrict to Location field, no dynamic default can be created for that record.

Creating Dynamic Default Logic for Custom Segments

If appropriate, you can create dynamic defaulting for a custom segment. This capability can be useful when a segment is applied to multiple record types. For example, suppose you have a segment that has been applied to sales transactions and customer records. You can configure the segment so that, when it occurs on a sales transaction, the segment populates with the segment value selected on the corresponding customer record. This type of defaulting overrides static defaulting.

You can configure dynamic defaults at the time you are creating the segment. You can also edit the segment later to add this configuration.

To create dynamic default logic for a custom segment:

1. Edit the custom segment.
2. Click the Application & Sourcing subtab.
3. Navigate to the subtab that represents the group or record type for which you want to configure dynamic defaulting. For example, to configure a default for transaction types, click the Transactions subtab. If you want to configure a default for one of the record types listed on the Other Record Types or Custom Record Types subtab, click one of those subtabs.
4. Set the Source List field to the appropriate value. The choices in this list each represent a field on the target record type, which is used to identify another record. For help understanding these options, see Source List Choices.
5. Click Save.

Example

In the following example, a Preferred Contact Method custom segment is set up to appear on sales orders, and be sourced from the customer record.
NetSuite automatically checks the Customer box on the Entities subtab to make the custom segment available on the customer record.
If the preferred contact method is specified on the customer record, it defaults in automatically when you select the customer on a sales order.

When the customer is selected on the sales order, the Preferred Contact Method field is filled in automatically with the value from the customer record.

Validation and Static Default Values for Custom Segments

If appropriate, you can create default values for custom segments. For any segment, you can use one or both of the following options:

- You can choose any of the segment's available values and make it the static default. That value is automatically selected on all new records that use the segment, unless other logic creates a different default. For details, see Configuring Static Defaults for Custom Segments.
- You can configure logic that dynamically generates default values. You can specify different logic for different groups of record types. With this approach, the default for a record is derived from the segment value that was saved on a related record. If a dynamic default value is loaded on a record, it overrides a static default. For details, see Dynamic Default Value Sourcing for Custom Segments.

Making a Custom Segment Mandatory

If you want a segment to be a required field on records where it appears, you can configure it to be mandatory. You implement this configuration by checking the Mandatory box on the segment definition. This option affects all record types to which the segment has been applied.

However, be aware that even with this configuration, the segment is not necessarily mandatory in all situations. For example:

- A custom form for the record type can be designed to prevent the custom segment from being visible. When a user creates or updates a record using this type of form, the segment is not mandatory.
- Some users might have permission to create and edit a specific record type, but they might not have permission to set a value for the segment. When these users work with the record, the segment is not mandatory.

You can configure a segment to be mandatory when you are creating the segment. You can also edit the segment later to change this configuration.

To make a segment mandatory:

1. Edit the custom segment.
2. Click the Validation & Defaulting subtab.
3. Check the **Mandatory** box.
4. Click Save.

**Configuring Static Defaults for Custom Segments**

When you create a static default, you choose a single value from the segment’s list of available values. When the segment appears on a new record as a field, it is populated with that value, unless other configuration overrides it.

Be aware of the following:

- If you have configured dynamic defaulting for a segment, any dynamic default that is generated overrides the static default. For details about dynamic defaults, see Dynamic Default Value Sourcing for Custom Segments.
- If the segment uses filtering, note that the value you choose as the static default can always be saved on a new record, even if your filtering configuration would otherwise make that value impermissible. If you want to avoid this behavior, choose a static default that is always permitted by your filtering configuration. For more details, see Filtering for a Custom Segment.

You can configure static defaults when you are creating a segment. You can also edit a segment later to add this configuration.

**To configure a static default value for a custom segment:**

1. Edit the custom segment.
2. Click the **Validation & Defaulting** subtab.
3. Set the **Default Selection** list to the appropriate value.
4. Click Save.

**User Permissions for a Custom Segment**

Every segment that has been applied to a record becomes a field on that record. For that reason, you can use the Permissions subtab to manage permissions for each segment in its capacity as a field on another record. At a high level, you can permit a role to do any of the following:

- **View and use a segment when it appears on a record** – For each segment, you can explicitly allow or disallow a role from being able to set a value for the segment. You can also make the segment hidden or read-only for a particular role. You manage these types of privileges by using the segment’s Record Access permission. For details, see Granting Roles Permission to Set Segment Values on Records.
- **View and refer to the segment when using searches and reports** – For each segment, you can explicitly allow or disallow a role from being able to use the segment as a search field or column. You can also allow or disallow a role from being able to use segments when customizing reports. You manage these types of privileges by using the segment’s Search/Reporting Access permission. For details, see Granting Roles Permission to Use Segments in Searches and Reports.

**Granting Roles Permission to Set Segment Values on Records**

For every segment, you can control which roles can view and interact with the segment as a field. You control this access by defining the Record Access level for each role. You can also set a default access level, which applies to all roles that do not have an access level explicitly defined.

For details, see the following sections:

- **Scope of the Record Access Permission**
Assigning the Record Access Permission

Scope of the Record Access Permission

The following table describes the access associated with the various levels of the Record Access permission.

<table>
<thead>
<tr>
<th>Level</th>
<th>Users can:</th>
<th>Users cannot:</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>View or set values for the segment on the records where the segment has been applied. For these users, the segment — both its label and value — are hidden.</td>
<td>Set values for the segment.</td>
</tr>
<tr>
<td>View</td>
<td>View segments on the records where the segment has been applied (if the user has permission to view or edit the record type). Users can view both the segment label and the selected value, if a selection has been saved. For users with this access level, the segment field is read-only, even if the user has permission to edit the record.</td>
<td></td>
</tr>
<tr>
<td>Edit</td>
<td>View and set values for the segment on records where the segment has been applied (if the user has permission to edit the record type).</td>
<td></td>
</tr>
</tbody>
</table>

Assigning the Record Access Permission

Use the following steps to assign the Record Access permission to a role.

To assign the Record Access permission to a role:

1. Edit the custom segment.
2. Click the Permissions subtab.
3. For a role to have access, the role must be referenced in the Permissions sublist. Review the sublist to see if the role is already listed, then do one of the following:
   - If the role is not listed, add a line to the sublist: In the Role column, select the appropriate role. In the Record Access Level column, choose the desired access level. Review the values for this role in the columns labeled Value Management Access Level and Search/Reporting Access Level. Make any changes as needed. Be aware that the Value Management Access permission gives the user permission to create values, so review this column with care. For details, see Granting a Role Permission to Manage Custom Segment Values. Then click Add.
   - If the role is already listed but does not have the ability to manage values, then edit the role’s access. Locate the role in the sublist. Edit the corresponding value in the Record Access Level column. Then click OK.
4. From the Default Record Access Level list, set the default record access level for a segment. This access level applies to any role that is not listed in the Permissions sublist with a specific access level.
5. Click Save.

Granting Roles Permission to Use Segments in Searches and Reports

For every segment, you can control which roles can view and interact with the segment when working with searches and reports. You control this access by defining the Search/Reporting Access level for each
role. You can also set a default access level, which applies to all roles that do not have an access level explicitly defined.

For details, see the following sections:

- Scope of the Search/Reporting Access Permission
- Assigning the Search/Reporting Access Permission

**Scope of the Search/Reporting Access Permission**

The following table describes the access associated with the various levels of the Search/Reporting Access permission.

<table>
<thead>
<tr>
<th>Level</th>
<th>Users can:</th>
<th>Users cannot:</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>■ Use the segment as a search filter.</td>
<td>■ Use the segment as a column in search results.</td>
</tr>
<tr>
<td></td>
<td>■ Use the segment as a column in search results.</td>
<td>■ View columns that reference custom segments in saved searches. The user can view the saved search, but columns referencing custom segments are omitted from the view.</td>
</tr>
<tr>
<td></td>
<td>■ View reports that use the segment as a filter or include it as a column.</td>
<td>■ View reports that use the segment as a filter or include it as a column.</td>
</tr>
<tr>
<td></td>
<td>■ Customize reports to use custom segments as filters or columns.</td>
<td>■ Customize reports to use custom segments as filters or columns.</td>
</tr>
<tr>
<td>View</td>
<td>■ View saved searches that include custom segments in their results.</td>
<td>■ Use the segment as a search filter</td>
</tr>
<tr>
<td></td>
<td>■ View reports that use the segment as a filter or include it as a column.</td>
<td>■ Use the segment as a column in search results.</td>
</tr>
<tr>
<td></td>
<td>■ Customize reports to use custom segments as filters or columns.</td>
<td>■ Customize reports to use custom segments as filters or columns.</td>
</tr>
<tr>
<td>Edit</td>
<td>■ View saved searches that include custom segments in their results.</td>
<td>■ Use the segment as a search filter</td>
</tr>
<tr>
<td></td>
<td>■ View reports that use the segment as a filter or include it as a column.</td>
<td>■ Use the segment as a column in search results.</td>
</tr>
<tr>
<td></td>
<td>■ Use the segment as a search filter.</td>
<td>■ Use the segment as a column in search results.</td>
</tr>
<tr>
<td></td>
<td>■ Use the segment as a column in search results.</td>
<td>■ Customize reports to use custom segments as filters or columns.</td>
</tr>
<tr>
<td></td>
<td>■ Customize reports to use custom segments as filters or columns.</td>
<td>■ Customize reports to use custom segments as filters or columns.</td>
</tr>
</tbody>
</table>

**Assigning the Search/Reporting Access Permission**

Use the following steps to assign the Search/Reporting Access permission to a role.

To assign the Search/Reporting Access permission to a role:

1. Edit the custom segment.
2. Click the Permissions subtab.
3. For a role to have access, the role must be referenced in the Permissions sublist. Review the sublist to see if the role is already listed, then do one of the following:
   - If the role is not listed, add a line to the sublist: In the Role column, select the appropriate role. In the Search/Reporting Access Level column, choose the desired access level. Review the values for this role in the columns labeled Value Management Access Level and Record Access Level. Make any changes as needed. Be aware that the Value Management Access...
permission gives the user permission to create values, so review this column with care. For details, see Granting a Role Permission to Manage Custom Segment Values. Then click Add.

- If the role is already listed but does not have the ability to manage values, then edit the role’s access. Locate the role in the sublist. Edit the corresponding value in the Search/Reporting Access Level column. Then click OK.

4. From the Default Search/Reporting Access Level list, set the default search/reporting access level. This access level applies to any role that is not listed in the Permissions sublist with a specific access level.

5. Click Save.

Dependent Segments

When editing an existing segment, you might want to know what filtering relationships it has with other segments. From the custom segment definition, use the Dependent Segments subtab to see a list of the segments that use the current segment for filtering their values.

Setting Display Order of All Custom Segments

You can specify the display order of custom segments in the body and lines on transactions, on other records, and on the GL Impact page, if the custom segment has GL impact. This display order reflects the priority and dependencies of custom segments.

The Display Order subtab lists all active custom segments in display order.

By default, the current segment is selected in the list. When you create a new custom segment, the name is listed as - Segment currently being created - until you enter a name in the Label field. To change the display order of custom segments, drag and drop the list items or click the Move To Top and Move To Bottom buttons.

On the Custom Segments list page, numbers in the Display Order column show the segment display order.

The default display order is based on the order in which segments were created. As new segments are created, they are added to the end of the display order unless you change it. If you check Show Inactives
on the list page, inactive custom segments appear at the bottom of the list because they have no display order.

**Editing Custom Segments**

In some cases, you may want to make changes to an existing custom segment. For example, you might want to change the segment’s label, the record types where the segment is applied, or the segment’s default value. Most fields on a custom segment can be changed after the segment is created, but note the following exceptions:

- The **ID** field cannot be changed.
- The **GL Impact** option cannot be changed.
- Your ability to change the **Type** field varies depending on your permissions. For details, see [Required Permissions for Editing Custom Segments](#).

For more information about editing custom segments, see [Inactivating a Custom Segment](#) and [Deleting a Custom Segment Definition](#).

When a custom segment is created, an associated custom record type is also created. The custom record type has the same name as the custom segment, and you can edit the custom record at any time to add custom segment values. For more information about creating custom record types, see [Creating Custom Record Types](#).

Custom segment definitions can be updated at any time, even if the segment is used in a transaction that is in a closed period.

Note also that changing the **Type** field from Multiple Select to List/Record can cause data loss. For details, see [Custom Segment Types](#).

To edit a custom segment:

1. Go to Customization > Lists, Records, & Fields > Custom Segments.
2. In the list, locate the appropriate segment and click **Edit**.
3. Change any fields as appropriate. If you need to add or update values, and the Values sublist is grayed out, click **Manage Values** to open a different page for working with values. For details, see [Creating Values by Clicking Manage Values](#).
   
   If you need help understanding other fields on the segment definition, see the following topics.
   - [Filtering for a Custom Segment](#)
   - [Applying a Custom Segment to Record Types](#)
   - [Dynamic Default Value Sourcing for Custom Segments](#)
   - [Validation and Static Default Values for Custom Segments](#)
   - [User Permissions for a Custom Segment](#)
   - [Dependent Segments](#)
   - [Setting Display Order of All Custom Segments](#)
4. After making all needed changes on the segment definition, click **Save**.

**Required Permissions for Editing Custom Segments**

To be able to open a custom segment for editing and save changes, you must have the appropriate privileges. Further, not all user groups can make changes. The following table gives examples of groups with permission configurations that let them edit segments.
### Note: After a custom segment is created, the GL Impact option cannot be changed.

<table>
<thead>
<tr>
<th>Group</th>
<th>Users can:</th>
<th>Users cannot:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users assigned to the Administrator role</td>
<td>These users can change any value on the segment definition, except for ID, which can never be changed.</td>
<td>Change the ID field.</td>
</tr>
<tr>
<td></td>
<td>Users in this group are the only ones who can change the following value:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The Type of an existing segment, if the segment's GL Impact box has been checked. (If the GL Impact box has not been checked, other users can also change the type.)</td>
<td></td>
</tr>
<tr>
<td>Users with both of the following:</td>
<td>Change most values on the segment definition.</td>
<td>Change any of the following fields:</td>
</tr>
<tr>
<td></td>
<td>- The Create, Edit, or Full level of Value Management Access permission</td>
<td>□ ID.</td>
</tr>
<tr>
<td></td>
<td>- The Edit or Full level of the Custom Segments permission</td>
<td>□ Type, if the segment's GL Impact box has been checked.</td>
</tr>
<tr>
<td>Users with either the Edit or Full level of the Custom Segments permission</td>
<td>Change most values on the segment definition.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>These users cannot directly modify anything on the segment definition, but they can work with the segment's values. They can open the segment definition in View mode and click Manage Values to display a page that lets them work with values. (Whether these users can add, edit, or delete values depends on the exact value of the user's Value Management Access permission. For details, see Required Permissions for Creating Custom Segment Values.)</td>
<td>Edit the segment definition.</td>
</tr>
<tr>
<td>Users with both of the following:</td>
<td>Edit the segment definition.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The Create, Edit, or Full level of Value Management Access permission</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- The View level of the Custom Segments permission</td>
<td></td>
</tr>
</tbody>
</table>

### Inactivating a Custom Segment

You can inactivate a custom segment by checking the Inactive box on the custom segment definition page. When a custom segment is made inactive, it no longer appears on any forms, reports, searches, or GL Impact pages, and is not available for SuiteBundler, SuiteFlow, SOAP web services, SuiteAnalytics Connect, Mass Update, or the Custom GL Lines Plug-in. However, an inactive segment is not permanently removed, and can be used again when needed.
Editing Custom Segments

When you inactivate a parent custom segment but leave its child segment active, the child segment is no longer filtered by the parent.

If you attempt to inactivate a custom segment that is used as a search filter or in a workflow, the inactivation is not allowed and a message appears indicating why the custom segment cannot be inactivated.

Inactive segments and their values are not visible on historical transactions, nor are they available when entering new transactions.

To view inactive segments, check Show Inactives on the Custom Segments list page. Inactive segments appear at the bottom of the list because they have no display order specified.

Note: Inactive custom segments cannot be included in bundles. For more information see Adding a Custom Segment to a Bundle.

Deleting a Custom Segment Definition

Depending on the Allow GL Custom Segment Deletion setting in General Accounting Preferences, you may be able to delete custom segments. If custom segments cannot be deleted, you can inactivate them.

Warning: If you delete a custom segment definition, both the custom segment definition and all custom segment values and instances on records are removed anywhere they are used. Consequently, custom segment values are removed from transactions, even in closed periods. You cannot reverse the deletion of a custom segment.

To delete a custom segment, you must have the appropriate permissions. Authorized users include the following:

- Administrative users – Users assigned to the Administrator role.
- Other users – Users assigned to a role that has the Full level of the Custom Segments permission.

You cannot delete a custom segment if it has dependent segments. For example, if another segment references this segment in the Filtered by field, the first segment cannot be deleted.

To delete a custom segment definition:

1. Go to Customization > Lists, Records, & Fields > Custom Segments.
2. Locate the name of the segment you want to delete, and click Edit.
3. Click Actions to display a popup menu, then Delete.
   The system displays a warning asking if you are sure you want to delete the custom segment.
4. Click OK.

Custom Segment Values

When a custom segment is displayed on a record, the segment's values are displayed as choices in a list. Depending on the configuration of the segment's Type field, the user can save one or more selections.
For details on designing and working with a segment's list of values, see the following:

- Value Creation for Custom Segments
- Creating Hierarchies Among a Custom Segment’s Values
- Edit a Custom Segment’s Values
- Changing the Order of a Custom Segment’s Values
- Making a Custom Segment’s Values Inactive
- Deleting a Custom Segment’s Values

Value Creation for Custom Segments

Values can be created during the time that a custom segment is being created. Additionally, authorized users can create values for existing segments.

For more details, see the following topics:

- Required Permissions for Creating Custom Segment Values
- Creating Values Within the Segment Definition
- Creating Values by Clicking Manage Values
- Creating Values Using the New Button
- Creating Values from the Setup Menu

Creating Values Within the Segment Definition

In some cases, you may have permission to edit custom segments. In these cases, you can create values by editing the segment's Values sublist. One advantage of this approach is that you can view all of the values in the list as you add new values.

To create segment values by editing the segment definition:

1. Edit the custom segment. Go to Customization > Lists, Records, & Fields > Custom Segments, and click Edit.
2. On the Values subtab, add a line to the sublist. In the Value column, enter a name. This text is displayed in the segment's list, when the segment appears on a record.
3. If your account uses the Multi-Language feature and you want to create translations for the value's name, click in the Translation column. The system displays a popup window. Enter the appropriate translations in the fields provided. When you are finished, click Done.
If another value in this sublist should be the parent of the value that you are adding, click in the **Parent** column. The system displays a list. Select the name of the parent. For more information about parent-child relationships, see Creating Hierarchies Among a Custom Segment's Values. NetSuite validates parent-child combinations to ensure that the values are unique. If the same parent-child combination already exists, an error message appears.

If you want to hide the new value from users, make the value inactive. Note that making the value inactive prevents the value from appearing as a standalone choice. However, if the value is a parent to another value, the value still appears as a parent. To make a value inactive, click in the **Inactive** column to display a box, and check the box.

If this segment uses filtering, you enter criteria for determining when this value is available. Click **Set Filters**. A popup window appears, showing the classification fields you can use to set conditions for this value's availability. Choose the appropriate values, and then click **Set**. Click **OK**. For more information about filtering, see Filtering for a Custom Segment.

If you have more values to add, repeat step 3.

Choose **Sublist** or **Alphabetical** to specify the display order of the values.

Click **Save** on the segment definition.

**Creating Values by Clicking Manage Values**

In some cases, you might not have permission to edit segments, but you do have permission to add values for a segment. In these cases, to add a value, you must open a page designed specifically for creating values. One way to navigate to this page is by viewing the segment definition and clicking a button labeled Manage Values.

**To create segment values by clicking Manage Values:**

1. Go to Customization > Lists, Records, & Fields > Custom Segments.
2. In the list, select the name of the segment for which you want to create values. The system opens the segment definition in view mode.
3. Click **Manage Values**. A list of the existing segment values appears.
4. Locate the button that lets you create a new value. The button's name incorporates the segment label. For example, if the segment's label is Profit Center, the button is labeled New Profit Center. Click the button to open a form for creating a new value.
5. Fill out the form as follows:
   - In the **Name** field, enter a name. This text is displayed in the segment's list, when the segment appears on a record.
   - If another value should be the parent of the value that you are adding, select that value in the **Filter** list. NetSuite validates parent-child combinations to ensure that the values are unique. If the same parent-child combination already exists, an error message appears. For more information about parent-child relationships, see Creating Hierarchies Among a Custom Segment's Values.
   - If this segment uses filtering, the page displays one or more lists with headings that begin with the words **Filter by**. In each box, select the appropriate value. To select more than one value, use the control key. For more information about filtering, see Filtering for a Custom Segment.
If you want to hide the new value from users, make the value inactive. Note that making the value inactive prevents the value from appearing as a standalone choice. However, if the value is a parent to another value, the value still appears as a parent. To make a value inactive, check the Inactive box.

If you want to hide the new value from users, check the Inactive box.

If your account uses the Multi-Language feature and you want to create translations for the value's name, enter the translations in the fields provided.

6. Click Save.

**Creating Values Using the New Button**

In some cases, you may have permission to create values for a segment, and you may also have permission to work with the segment in another context. For example:

- You may have permission to select a value for a segment when it appears on a record that you have permission to edit.
- You may have permission to create values for two different segments that have some relationship to each other. For example, one segment may be used to filter the other segment's values.

In these cases, when you come across the segment on another record, you can use the New button to create values. This approach lets you create a new segment value without navigating away from the record you are working on. The new value is immediately available for selection in the record.

**To create values using the New button:**

1. During the time that you are working on the record that shows the segment, move your cursor to the right of the segment label.
   
   The system displays a New button.

2. Click New.
   
   The system displays a popup form that lets you create a new value.

3. Fill out the form as follows:
   - In the Name field, enter a name. This text is displayed in the segment's list, when the segment appears on a record.
   - If another value should be the parent of the value that you are adding, select that value in the Filter list. NetSuite validates parent-child combinations to ensure that the values are unique. If the same parent-child combination already exists, an error message appears. For more information about parent-child relationships, see Creating Hierarchies Among a Custom Segment's Values.
If this segment uses filtering, the page displays one or more lists with headings that begin with the words Filter by. In each box, select the appropriate value. To select more than one value, use the Ctrl key.

For more information about filtering, see Filtering for a Custom Segment.

If you want to hide the new value from users, make the value inactive. Note that making the value inactive prevents the value from appearing as a standalone choice. However, if the value is a parent to another value, the value still appears as a parent. To make a value inactive, check the Inactive box.

If your account uses the Multi-Language feature and you want to create translations for the value's name, enter the translations in the fields provided.

4. Click Save.

Creating Values from the Setup Menu

Custom segments appear on the Classification menu in Setup > Company > Classifications at the bottom of the classification group. In the following example, the user created a Business Area custom segment.
To add a value for the custom segment:

1. Go to Setup > Company > Classifications > Custom Segment Name > New.
2. In the **Name** field, enter the value.
3. In the **Parent** list, select the parent for this value, if required.
4. Click **Save**.

If the display order for the custom segment values is set to **Sublist**, the new value appears at the end of the list of values. If the display order for the custom segment values is set to **Alphabetical**, the new value is incorporated into the alphabetical list of values.

Creating Hierarchies Among a Custom Segment’s Values

Within a custom segment’s list of values, you can create a hierarchy, which makes a value a parent to one or more other values. When the segment is available on a record, the parent-child relationships are denoted with a colon.
A segment value can also have a grandparent and more senior levels of hierarchy.

If you have a value that should exist only to be the parent of another value, you might want to make the value inactive. Making a value inactive prevents the value from appearing as a standalone choice in the segment's list but permits the value to remain visible as a parent. To make a value inactive, check the Inactive box associated with that value.

To create a parent-child relationship, you edit the value that should be the child, as described in the following procedure.

**To make a value the child of another value:**

1. Open a page where you can edit the value. Depending on your privileges, you may be able to make your changes by editing the custom segment's Values sublist. Otherwise, you can view the segment definition, click the Manage Values button, and open the appropriate value for editing. For details, see Edit a Custom Segment's Values.
2. Do one of the following:
   - If you are editing the Values sublist, locate the value in the sublist. Click in the Parent column to enable a list. Set the list to the desired parent value. Click Save.
   - If you have opened a single value for editing, the Parent list should be visible on the page. Set the list appropriately. Click Save.

NetSuite validates parent-child combinations to ensure that the values are unique. If you have entered a duplicate of an existing parent-child combination, an error message appears.
Custom Segment Values

Edit a Custom Segment’s Values

For details on making changes to a custom segment’s values, see the following topics:

- Required Permissions for Editing Custom Segment Values
- Editing a Custom Segment’s Values by Updating the Segment Definition
- Editing a Custom Segment’s Values by Clicking Manage Values

Required Permissions for Editing Custom Segment Values

To be able to edit a custom segment’s values, a user must have the appropriate privileges. There are several permission configurations that can result in users having this access.

Examples of users with various permission configurations are described in the following table, along with the procedures that each group can use to modify values.

<table>
<thead>
<tr>
<th>Description</th>
<th>Permitted Methods for Editing Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users assigned to the Administrator role</td>
<td>Editing a Custom Segment’s Values by Updating the Segment Definition</td>
</tr>
<tr>
<td></td>
<td>Editing a Custom Segment’s Values by Clicking Manage Values</td>
</tr>
<tr>
<td>Users with both of the following:</td>
<td>Editing a Custom Segment’s Values by Updating the Segment Definition</td>
</tr>
<tr>
<td></td>
<td>Editing a Custom Segment’s Values by Clicking Manage Values</td>
</tr>
<tr>
<td>The Edit or Full level of the Value Management Access permission</td>
<td>Editing a Custom Segment’s Values by Updating the Segment Definition</td>
</tr>
<tr>
<td>The Edit or Full level of the Custom Segments permission</td>
<td>Editing a Custom Segment’s Values by Clicking Manage Values</td>
</tr>
<tr>
<td>Users with both of the following:</td>
<td>Editing a Custom Segment’s Values by Clicking Manage Values</td>
</tr>
<tr>
<td>The Edit or Full level of the Value Management Access permission</td>
<td>Editing a Custom Segment’s Values by Clicking Manage Values</td>
</tr>
<tr>
<td>The View level of the Custom Segments permission</td>
<td>Editing a Custom Segment’s Values by Clicking Manage Values</td>
</tr>
</tbody>
</table>

**Note:** Be aware that even if users have only the Edit or Full value of the Custom Segments permission, they can assign themselves the Value Management Access permission for any custom segment.

Editing a Custom Segment’s Values by Updating the Segment Definition

If you have the appropriate privileges, you can modify a custom segment’s values by editing the segment’s Values sublist. One advantage of this approach is that you can view all of the values in the list as you make changes.

For details on the permissions required to complete this procedure, see Required Permissions for Editing Custom Segment Values. For an alternate method of editing values, see Editing a Custom Segment’s Values by Clicking Manage Values.

To edit a custom segment’s values by updating the segment definition:

1. Edit the custom segment.
2. Locate the value you want to change in the Values subtab. Make any of the following changes, as appropriate:

- In the **Value** column, you can change the value's name. The value's name is displayed in the segment's list when the segment appears on a record.

- If your account uses the Multi-Language feature, and you want to change translations for the value's name, click in the **Translation** column. The system displays a popup window. Modify the translations as needed. When you are finished, click **Done**.

- If you want to create a parent or change the value's parent, click in the **Parent** column. The system displays a list. Adjust the list as appropriate. For details about parent-child relationships, see Creating Hierarchies Among a Custom Segment's Values.

- If you want to change whether this value is visible to users, click in the **Inactive** column to display a check box. Check or clear the box as appropriate. Note that making the value inactive prevents the value from appearing as a stand-alone choice. However, if the value is a parent to another value, the value still appears as a parent.

- If this segment uses filtering, you may want to create, change, or clear criteria for determining when this value is available. In this case, click the name of the value to display a series of buttons. Click **Set Filters** to display a popup window. The popup window shows the other classification fields you can use to set conditions for this value's availability. Make the appropriate changes. Then click **Set** to close the window. Click **OK** to close the series of buttons and save the filtering choices. For more information about filtering, see Filtering for a Custom Segment.

3. If you have more values to modify, repeat step 2.

4. Click **Save** on the segment definition.

### Editing a Custom Segment's Values by Clicking Manage Values

One way to edit a custom segment's values is to view the segment definition and click the button labeled Manage Values. You should use this method if you do not have permission to edit the segment definition.

**Note:** Custom segment values on transactions that impact GL cannot be edited in closed periods.

**To edit a custom segment’s values by clicking Manage Values:**

1. Go to Customization > Lists, Records, & Fields > Custom Segments.
2. In the list, select the name of the segment for which you want to edit values.
   
   The system opens the segment definition in view mode.
3. Click **Manage Values**. A list of the existing segment values appears.
4. Locate the value you want to modify and click the corresponding **Edit** button.
5. Make any of the following changes, as appropriate:
In the **Name** column, you can change the value's name. The value's name is displayed in the segment's list when the segment appears on a record.

If you want to create a parent or change the value's parent, adjust the **Parent** list. For more information about parent-child relationships, see Creating Hierarchies Among a Custom Segment's Values.

If this segment uses filtering, the page displays one or more list boxes. These lists have headings that begin with the words **Filter by**. In each list, change the selections as appropriate. You can use the control key to select more than one value or to clear a value. For more information about filtering, see Filtering for a Custom Segment.

If you want to change whether this value is visible to users, check or clear the **Inactive** box. Note that making the value inactive prevents the value from appearing as a stand-alone choice. However, if the value is a parent to another value, the value still appears as a parent.

6. Click **Save**.

Changing the Order of a Custom Segment's Values

When you add values to a custom segment, those values are displayed in a list on a record.

There are two ways to configure the sequence of values. You can sort them alphabetically, or you can have them display in the sequence in which they are listed on the segment definition. The latter option is the default.

To change the way values are ordered, you must have permission to edit the segment definition.

**To reorder segment values:**

1. Edit the custom segment.
2. On the Values subtab, locate the **Display Order** label.
3. Do one of the following:
   - To make the values appear in the order shown on the segment's Values sublist, choose **Sublist**.
   - To make the values appear in alphabetical order, choose **Alphabetical**.
Making a Custom Segment's Values Inactive

In some cases, you may need to hide one or more of a custom segment's values. To hide a value, you make it inactive. Be aware that when you make a value inactive, it does not appear as a choice in the segment's list. However, if the value is a parent to another value, it still appears as a parent.

There are two ways to make a segment's value inactive:

- Making a Value Inactive by Editing the Segment Definition
- Making a Value Inactive by Clicking Manage Values

If appropriate, you can also delete a segment's values. For details, see Deleting a Custom Segment's Values.

Making a Value Inactive by Editing the Segment Definition

In some cases, you may have a set of permissions that includes the ability to edit the segment definition. In these cases, you can make a value inactive by directly editing the segment's Values sublist. One advantage of this approach is that you can view all values as you modify the list.

To make a value inactive by editing the segment definition:

1. Edit the custom segment.
2. Locate the value you want to hide. Click in the Inactive column to display a check box.
3. Check the Inactive box.
4. Click Save.

Making a Value Inactive by Clicking Manage Values

In some cases, you might not have permission to edit segments, but you do have permission to edit values for a segment. In these cases, to make a value inactive, you must open a page designed specifically for managing the value. You navigate to this page by viewing the segment definition and clicking the button labeled Manage Values.

To make a value inactive by clicking Manage Values:

1. Go to Customization > Lists, Records, & Fields > Custom Segments.
2. In the list, select the name of the segment for which you want to edit values.
The system opens the segment definition in view mode.

3. Click **Manage Values**. A list of the existing segment values appears.
4. Locate the value you want to modify and click the corresponding **Edit** button.
5. Check the **Inactive** box.
6. Click **Save**.

Deleting a Custom Segment's Values

In some cases, you may need to delete one or more of a custom segment's values. There are two ways to delete a segment's values:

- **Deleting Values Within the Custom Segment Definition**
- **Deleting Values by Clicking Manage Values**

You cannot delete a value if that value has been saved on an instance of a record. If you try to delete this type of value, the system displays the following error: *The segment value cannot be deleted because a child record exists.*

As an alternative to deletion, consider making the value inactive.

If a period is closed, you cannot change custom segment values that impact GL on any transactions in the period.

**Deleting Values Within the Custom Segment Definition**

In some cases, you may have a set of permissions that includes the ability to edit the segment definition. In these cases, you can delete values by editing the segment's Values sublist. One advantage of this approach is that you can view all values as you modify the list.

**To delete a custom's segment value from within the segment definition:**

1. Edit the custom segment.
2. Locate the value you want to delete. Click anywhere in that row to enable a series of buttons.
3. Click **Remove**.
4. Click **Save**.

**Deleting Values by Clicking Manage Values**

In some cases, you might not have permission to edit segments, but you do have permission to delete values for a segment. In these cases, to delete a value, you must open a page designed specifically for managing the value. You navigate to this page by viewing the segment definition and clicking the button labeled Manage Values.

**To delete values by clicking Manage Values:**

1. Go to Customization > Lists, Records, & Fields > Custom Segments.
2. In the list, select the name of the segment for which you want to edit values. The system opens the segment definition in view mode.
3. Click **Manage Values**. A list of the existing segment values appears.
4. Locate the value you want to modify and click the corresponding **Edit** button.
5. Point to the **Actions** label to display a popup menu.
6. Click **Delete**.
   The system displays a message asking if you are sure you want to delete the record.
7. Click **OK**.

**Using the Script ID to Access Custom Segment Body, Line, and Filter By Fields**

The field ID of custom segments is prefixed with `cseg`. When creating your custom segments, establish a clear naming convention for your custom segment definition IDs. Meaningful script names are helpful when you are creating solutions for scripts, bundles, formulas, workflows, searches, and more.

Custom segment definitions use a unified script ID, which lets you use one ID in your scripts to refer to all of the record types in a custom segment. With the unified ID, you are not required to know the applied record type when referring to a custom segment field on a specific record.

Custom segment IDs in searches and filters show the field type so that you can distinguish among body, line and filter fields.

To filter custom segments, use the pattern `_filterby_` to include custom segment fields in the filtering criteria. For example, to filter `cseg_section` by subsidiary:

```
CONCAT({cseg_section},{cseg_section.cseg_section_filterby_subsidiary})
```

The following example uses SuiteScript 1.0 to find contacts.

```javascript
var filters = new Array();
filters[0] = new nlobjSearchFilter( 'entityid', null, 'doesnotcontain', ['q'] ); // filter only contacts that do not have "q" in the name
var columns = new Array();
columns[0] = new nlobjSearchColumn( 'entityid' );
columns[1] = new nlobjSearchColumn( 'email' );
columns[2] = new nlobjSearchColumn( 'cseg3', 'transaction' ); // in search result, show the value of custom segment that is on transaction body for contact transactions
columns[3] = new nlobjSearchColumn( 'line.cseg3', 'transaction' ); // in search result, also show the value of custom segment on transaction line
columns[4] = new nlobjSearchColumn( 'cseg2_filterby_location', 'cseg2' ); // I want to know the value of segment filter field that is joined via custom segment2 on the contact record
var search1 = nlapiCreateSearch( 'contact', filters, columns );
var result = search1.runSearch();
result.getResults(0,5); // show first five result lines
```

**Custom Segments in Record Searches**

After you have applied a custom segment to a record type, authorized users can reference that segment when searching for records. When you use advanced search, you can use custom segments as search filters and search columns. On both the Criteria and Results subtab, each segment appears among the other available fields from the record. Each segment's label is followed by the word Custom in parentheses.
Note that if you are searching a transaction, the name of the custom segment appears twice in the list of fields. One instance represents the custom segment as a transaction column. The other represents the segment as a body field. The segment appears twice even if you did not apply the segment to both a transaction sublist and the transaction body.

To use a custom segment as a search filter or column, you must have the Edit level of the Search/Reporting Access permission for that segment.

To view a saved search that includes the segment in its results, you must have at least the View level of the Search/Reporting Access permission. Otherwise, the column referencing segment data is hidden.

For more details on the Search/Reporting Access permission, see Granting Roles Permission to Use Segments in Searches and Reports.

For full details on running an advanced search, see the help topic Defining an Advanced Search.

**Note:** If you want to search for a custom segment definition, you can use the Custom Segment search type. Go to Reports > New Search and click Custom Segment.

### Customizing a Report by Using Custom Segments

After you have applied a custom segment to a record type, authorized users can reference that segment when customizing reports. You can use custom segments as columns or filters.

When you click **Edit Columns** or **Filters** in the Report Builder, you can find segments in the Add Fields list. Each segment appears under the label representing the record type where it was applied. Segments are listed alphabetically with other available fields.
For transactions, the name of the custom segment appears twice in the list of fields. One instance represents the custom segment as a transaction column. The other represents the segment as a body field. The segment appears twice even if you did not apply the segment to both a transaction sublist and the transaction body.

For reports other than financial statements, you can add a custom segment as a report dimension to the current report. Select the custom segment in the Column list in the report footer. The selection applies only to the current report, so you need to repeat the selection each time you run the report.

In the Financial Report Builder, you can add a custom segment as a dimension for a financial statement. Select the custom segment in the View Columns By list on the Edit Columns page. This selection persists whenever the custom financial statement is run.

To customize a report to include a custom segment, you must have the appropriate Search/Reporting Access permission level for that segment. Similarly, to view a report that filters by or references custom segments, you must have the appropriate level. For more details on this permission, see Granting Roles Permission to Use Segments in Searches and Reports.

For full details on customizing reports, see the help topic Report Customization.

Adding a Custom Segment to a Bundle

You can create bundles that include custom segments. When creating the bundle, custom segments are listed under Custom Lists/Records on the Select Objects page of the Bundle Builder. For more information about SuiteBundler, see the help topics SuiteBundler Overview and Creating a Bundle with the Bundle Builder.

For the limitations of custom segments in bundles, see the following:

- Creating and Installing Bundles
- Updating Bundles
- Uninstalling Bundles
- Correcting a Damaged Custom Segment
- Updates Required to Existing Objects for Custom Segment Unified IDs

Creating and Installing Bundles

When a bundle is installed, all selected objects are copied as well as their settings, such as GL impact and where the segment will be used. Whenever a custom segment that is part of a parent/child hierarchy is included in a bundle, all segments in the hierarchy are included in the bundle automatically.

Locked Custom Segments in Bundles

If a custom segment is locked in the Bundler, you cannot change it from the user interface in the target account.

If permissions are assigned for a locked custom segment and a standard role, those permission levels are included with the bundle. For locked custom segments, no one can edit the custom segment definition and only an administrator and the roles specified can manage custom segment values.

To add or change the roles that are allowed to manage custom segment values, you can assign the value management permission to a custom role for specific custom segments.
To assign the value management permission to a custom role:

1. Go to Setup > Users/Roles > Manage Roles.
2. Click Edit next to the custom role that you have created.

   **Note:** You cannot assign access to locked custom segments for built-in NetSuite roles.

3. On the Permissions > Setup subtab, select Custom Segments from the Permission list.
4. In the Level field, select the level of access that this role requires for custom segments.
5. On the Permissions > Custom Record subtab, select the custom segment from the Record list.
6. In the Level field, select the level of access that this role requires for custom segment values.
7. Click Save.

On the Permissions subtab of the Custom Segment configuration, the added custom role and value management access level are shown. No record access or search/reporting access levels are set for the custom role, so the default settings are applied.

---

## Updating Bundles

When an installed bundle is updated in a target account, the custom segment definitions in the source bundle overwrite the settings in the account, with the following exceptions:

- Segment use settings are merged
- Display order is not changed
- GL Impact flags are not changed
- Segment values are preserved or merged.

For any custom segment that includes data, a preference can be set that is applied during bundle updates. This preference indicates whether: the target account custom segment data should be preserved, or data from the bundled custom segment and target account custom segment should be merged, during the update. The default is to preserve data.

A list is available for custom segments on the Set Preferences page of the Bundle Builder. The list includes Preserve Data and Merge Data options. When the Preserve Data option is selected, the bundle update does not make any changes to target account custom segment values. When the Merge Data option is selected, the bundle update merges custom segment values, without deleting any values from the target account custom segment. The avoidance of target account custom segment value deletion is due to the possibility that a custom segment may have GL impact. In addition, no option is provided to replace data for a custom segment, also to avoid unintentional changes to GL impact.

For more information about setting bundle preferences, see the help topics Step 4 Set Preferences and Bundle Object Preferences.
If the custom segment type in the source account is changed, the type is updated on the custom segment in the target account. Be aware of the effect the change will have:

- If the custom segment type is changed from list/record to multi-select, the segment is removed from any transaction sublists. Transaction column applications are not valid for multi-select lists.
- If the custom segment type is changed from multi-select to list/record, data may be lost.

If the Allow GL Custom Segment Deletion preference is enabled, an update to a bundle can delete a custom segment. If the Allow GL Custom Segment Deletion preference is not enabled, an error appears when you attempt to update a bundle containing a deleted custom segment.

You should avoid deleting custom segments using the bundle update. The best practice is to inactivate any custom segments that are no longer needed.

### Related Topics
- Adding a Custom Segment to a Bundle
- Creating and Installing Bundles
- Uninstalling Bundles
- Correcting a Damaged Custom Segment

### Uninstalling Bundles

If the Allow GL Custom Segment Deletion setting is enabled, bundles containing custom segments can be uninstalled. If the Allow GL Custom Segment Deletion setting is not enabled, an error appears when you attempt to uninstall a bundle containing a custom segment.

### Related Topics
- Adding a Custom Segment to a Bundle
- Creating and Installing Bundles
- Updating Bundles
- Correcting a Damaged Custom Segment

### Correcting a Damaged Custom Segment

A custom segment can sometimes become corrupted if any of its required components are missing. A completely defined custom segment must contain one custom record component, and one each of the following five custom field component types: Body, Column, Entity, Event, and Item. For information about the components of a custom segment, see Creating a Custom Segment

If any components are missing from the custom segment, you receive a segment corrupted error on the Custom Segment configuration page. However, you can continue to use the custom segment configuration page with limited operations available. As of 2020.1 if you installed a custom segment from a bundle and it is damaged, you can reinstall the bundle to correct the error. If you created a custom segment manually and it is damaged, delete the custom segment definition and recreate it.
Adding a Custom Segment to a Bundle

When you edit the definition of a corrupted custom segment, only the Cancel button is available. Depending on your permissions, the Delete button might also be available.

If a form or applied record accesses a corrupted custom segment, you can save or view the record without the custom segment value.

Related Topics
- Adding a Custom Segment to a Bundle
- Creating and Installing Bundles
- Updating Bundles
- Uninstalling Bundles

Using Custom Segments in Workflows

If appropriate, you can reference custom segments when working with workflows. You can do either of the following:

- Refer to custom segments when they appear as fields on other record types. For example, you can configure a segment field to default to a particular value. For details, see Custom Segments as Fields.
- Use workflows to manage the records that represent custom segment values. For details, see Records that Represent Custom Segment Values.

Custom Segments as Fields

When a custom segment has been applied to a record type, your workflow can reference the segment as it would any of that record type's fields. For examples, see the following sections:

- Setting Field Values
- Setting Workflow Conditions

Setting Field Values

If appropriate, you can use the Set Field Value action to populate a custom segment field with a value. This approach can be an alternative to using the custom segment's Source List field (which is described in Dynamic Default Value Sourcing for Custom Segments). Compared with the Source List field, the Set Field Value action allows for greater granularity when configuring the behavior of defaults.

For example, with a workflow, you can set a different sourcing method for each specific record type. By contrast, in some cases, the custom segment's Source List field lets you choose only one defaulting method, and that method is shared by multiple record types. For example, you can select only one Source List value for all transaction types. However, with a workflow, you could use the Set Field Value action to create a different defaulting method for each specific transaction type.

For example, suppose you were editing a workflow that affects the Cash Sale transaction type. For the purpose of this example, assume also that you have a custom segment called Sales Region that has been applied to both the Cash Sale transaction type and the Customer record type. You might want the Sales Region value used on each cash sale to match the value of the segment as it appears on the corresponding customer record.
To configure this behavior, use the fields under the Parameters heading in the Workflow Action window. You identify the segment for which you want to create the default by using the Field list on the left side of the page, then use the fields under the Value heading to configure the sourcing behavior. Specifically, you select the From Field radio button. Then you use the Record list to identify the record that will provide the default value. Use the Field list to identify the source field.

The Set Field Value action also provides multiple options for determining when the segment field will be populated with a default value. By contrast, when you use the custom segment's Source List field, the segment is populated with the default value only when the value of the source field is set or changes. Typically, this event takes place when the user opens a new record, if the source field populates automatically, or when the user is editing the record. With a workflow, you could configure the value to be populated at different times, including after the user clicks Save and is no longer looking at the record. You configure this behavior in the Workflow Action window. Specifically, you use the Trigger On list.

Similarly, with a workflow, you could choose to have the segment value be set only when the record is updated, rather than when it is created (or vice versa). You do this by editing the workflow and making choices under the Event Definition heading.

For full details on the Set Field Value action, see the help topic Set Field Value Action.
Setting Workflow Conditions

Another time you may want to refer to a segment as a field is when setting workflow conditions. For example, you could create a condition that a workflow runs only if the segment is set to a certain value. In this case, open the Workflow Condition window as you would when setting any condition. In the Field list, select the name of the segment. In the Value column, add the specific text that you want to reference. For example, in the following screenshot, a condition dictates that a workflow will run only when the Sales Region segment is set to the value Lima.

Records that Represent Custom Segment Values

When you create a value for a custom segment, NetSuite creates a record to represent that value. For example, consider a custom segment called Sales Region that has values such as Americas and EMEA. In this case, the values Americas and EMEA are records of type Sales Region. If appropriate, you could create a workflow that handles these records.

This type of workflow is used only if you open a form to create or work with the segment value. The workflow is not used when you add or update values by modifying the Values sublist of the custom segment definition.

You might use a workflow that manipulates segment values in the following type of situation: Suppose users need to create new values for the Sales Region custom segment. You might prefer that these values be inactive by default. In this case, you could create a workflow that automatically checks the Inactive box when the user opens a form for creating a new value. To configure this behavior, create a Set Field Value action. In the Workflow Action window, choose Inactive in the Field list. Under Value, select a static value of checked.

You could also take other actions. For example, you could configure a workflow to automatically create a new value for an existing custom segment. To configure this behavior, you use the Create Record action. In the Workflow Action window, you make changes under the Parameters heading. Specifically, in the
Using Custom Segments in Workflows

Record Type list, select the custom segment for which you want to create a value. Minimally, you also have to add logic for populating the Name field of the new value.

For more details on the Create Record action, see the help topic Create Record Action.

**Important:** Any workflow you create to manage a custom segment's values affects the behavior of values only in certain cases. The workflow is used only if you open a value in a separate form, as described in Creating Values by Clicking Manage Values, Editing a Custom Segment's Values by Clicking Manage Values, and Creating Values Using the New Button. This type of workflow does not affect changes that you make when editing the Values sublist of the custom segment definition.

### SuiteScript and Custom Segments

Scripting is supported for custom segment values and for custom segments as fields. The following are supported:

- On record types that are exposed to SuiteScript, you can use SuiteScript to set values for custom segments that exist as fields.
- You can use SuiteScript to create values for existing custom segments.

Scripting is not supported for custom segment definitions, and the segment definition type is not shown in the Applies To list on script deployments. However, each custom segment you create is associated with a custom record type that has the same name. These segment-managed record types are similar to other custom record types and are fully scriptable. The name of a custom segment in the scripting area refers to the custom record type or one of the custom segment fields. It does not refer to the custom segment definition itself.

### SOAP Web Services and Custom Segments

For detailed information about working with custom segments in SOAP web services, see the help topic Custom Segment in the SOAP Web Services Records Guide.

### CSV Import and Custom Segments

You can import values for an existing custom segment using the CSV Import. For example, with the segment called Sales Channel, you could use the Import Assistant to import new values such as Catalog and Partner Outlet.

For more information about this import, see the following:

- Prerequisites for Importing Values
- Segments Appear Under Classification Import Type
Prerequisites for Importing Values

Before you can import values for custom segments, you must do both of the following:

- Enable the Custom Segments feature at Setup > Company > Enable Features, on the SuiteCloud subtab.
- Create at least one custom segment in your account. You can create custom segments at Customization > Lists, Records, & Fields > Custom Segments > New.

By default, only administrators have permission to import custom segment values. However, you can grant permission to other roles. The following permissions are required:

- **The global Custom Segments permission** – You can add this permission on the Permissions > Setup subtab of the role record. At least the View level is required.
- **The Value Access Level permission for the custom segment** – You can add this permission on the Custom Record subtab of the role record. To be able to add values, the role needs at least the Create level. To be able to update values, or to be able to set the parent for a value, the user needs at least the Edit level.

Segments Appear Under Classification Import Type

To import values, set the Import Type to Classification. In the Record Type list, the system displays both the names of your custom segments and any other classifications enabled in your account. For example, custom segments named Brand and Product Line would be listed with Class, Department, and Location, if those features were enabled.

For each import, you can define values for the following fields:

- The name of the value.
- A parent for the value. For example, if your segment has a value called Retail Store, that value might be the parent of values such as Express Kiosk and Superstore.
- Inactive, which determines whether the segment value is available to be selected.
- Translations of the value's name, if your account uses the Multi-Language feature.

For more information, see the help topic Custom Segment Value Import.
Custom Centers

You can create custom centers that you apply to custom roles. To use the Custom Centers feature, you must enable the Custom Records feature, at Setup > Company > Setup Tasks > Enable Features > SuiteCloud.

NetSuite Centers determine which tabs and links are available for groups of similar user roles. For example, the Sales Center is shared by the Sales Rep, Sales Manager, and Sales Administrator roles and includes tabs such as Leads, Opportunities and Forecast. (For a visual representation of a center, see Centers in the SuiteBuilder Overview section.)

Each tab contains links to transactions, lists and setup pages. The links that appear are based on the user’s role and the permissions the role is granted. For example, users assigned to the Sales Rep role would see different links on the Forecast tab than users with the Sales Administrator role because of permissions granted to each role. However, both roles share the Sales Center.

To create a custom center, you create a center record and then create its custom tabs. When you create custom tabs, you choose the center you want the tab to appear in. The center can be either an existing or custom center. You also choose the links and portlets that will appear in the center. You can only customize centers and tabs if the Custom Records feature is enabled.

Note: Users can only use links and information that their roles have access to. To customize roles, go to Setup > Users/Roles > Manage Roles. You must create a new role to apply a custom center to it. To create a new role, click New on the Manage Roles page, select your custom center in the Center Type field and customize your new role.

Be aware that newly enabled feature menu items are not automatically added to custom centers. You must manually add menu items to custom centers after a feature is enabled.

Important: SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should only access the NetSuite UI by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic SuiteScript 2.0 Custom Pages.

Account–Specific Domains in Custom Center Links

As of 2020.1, NetSuite supports account–specific domains in custom center links.

Previously, NetSuite domains were specific to the data center where your account was hosted. Links in custom centers used these data center–specific domains. NetSuite domains are no longer specific to data centers. Instead, they are account–specific domains, which are domains specific to your account, not to the data center where they are hosted. For more information, see the help topic URLs for Account-Specific Domains.

For users, the account-specific domains feature translates current data center–specific URLs to the appropriate account-specific domain URLs. For example, a system.eu2netsuite.com link in your custom center becomes <accountID>.app.netsuite.com, where <accountID> is your account ID.

The automatic translation from data center–specific URLs to account-specific domain URLs is applied to your active session. Configuration remains unchanged. Links that use account-specific domains are faster than links translated in the current session. You should update all links in custom centers to use the appropriate account-specific domain.
Creating and Editing Custom Centers

When you create custom center links, if you attempt to add a data center–specific URL, a warning message appears suggesting an account–specific domain URL to use instead. Consider making the suggested change. If you choose not to make changes, the URL as you entered it is saved. In the NetSuite center, the link translates to an account–specific domain URL.

For more information, see the help topic How to Transition from Data Center-Specific Domains.

See the following topics.

- Creating and Editing Custom Centers
- Creating Center Tabs
- Creating Center Categories
- Creating Center Links
- Translating Custom Centers, Tabs, Categories, and Links
- Assigning a Custom Center to a Custom Role

Creating and Editing Custom Centers

To create a custom center, go to Customization > Centers and Tabs > Centers > New. Enter a name for your center and click Save.

Next, you create the tabs and links you want to appear in your center. To add tabs to a center, go to Customization > Centers and Tabs > Centers, click Edit next to the center, and click the New Tab button. Or, you can go to Customization > Centers and Tabs > Center Tabs > New.

To change the order of the tabs in a custom center, go to Customization > Centers and Tabs > Centers and click Edit next to the center. The definition page for the center lists the tabs in the order in which they are displayed. To change the order, you can select a tab listing and drag it to a different place in the order. You can also select a tab listing and click the Move to Top or Move to Bottom buttons.

You can only customize centers if the Custom Records feature is enabled.

You can use SuiteCloud Development Framework (SDF) to manage custom centers as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom center to another of your accounts. Each custom center page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

Note: If the Multi-Languages feature is enabled in your account, you can define translated center names to be used for different language settings. See Translating Custom Centers, Tabs, Categories, and Links.

Creating Center Tabs

You use center tabs to add custom tabs to roles. Center tabs can include categories of links to NetSuite pages, custom records, Suitelets, or external web pages. You can also designate which portlets you want to appear on the dashboard when users click the tab. You can only customize center tabs if the Custom Records feature is enabled.
In the following figure, Documents, Setup, Customization, and Support are the tabs. Off the Documents tab, Files, Templates, and Mail Merge are the categories. Off of the Files category are the links to the File Cabinet, SuiteScripts, Attachments Received, and so on.

Note that if you want to create a tab that displays internal information in your account, you should create your own custom intranet tab. For more information, see the help topic Publishing Information to an Internal Site.

To create a center tab:

1. Go to Customization > Centers and Tabs > Center Tabs > New. A Custom Center Tab page appears.

2. In the Label field, enter a name for the tab. Users click the name to view the tab’s contents.

3. In the Center field, choose the center where you want the tab added. If you want to show the tab in all centers, select -All-.

   **Note:** If you create a custom center tab and specify that it be shown in all centers, it will not appear in a dashboard that is already published.

4. On the Categories subtab, in the Name column, enter a name for a category of links. After creating your categories, you will later follow the steps in Creating Center Links to assign specific links to each category.

   **Warning:** Step 4 is for adding custom categories to custom tabs. If you want to add custom categories to standard, built-in NetSuite tabs, see Creating Center Categories.

5. Click Add.

6. Add all of the category links that you want to appear on the tab.

7. Click the Portlets subtab.

8. In the Type column, choose Links.
9. In the **Column** section, choose to place the portlet on the left, right or in the middle of your tab.

10. If you want the link to show, check the **Show** box.

11. Click **Add**.

12. Add all the required portlets to the tab.

13. Click the **Audience** subtab.

14. Specify who can access the custom center. For each area, you can make the custom center available to all by checking the **Select All** box. Indicate the areas that have access. You can specify:
   - roles
   - employees — giving permission to specific employees could create extra maintenance requirements when employees change roles or leave the company. The best practice is to assign a custom center to a role instead of directly to an employee.
   - departments
   - groups
   - customers
   - vendors
   - partners

**Note:** If the Multi-Languages feature is enabled in your account, you can define translated tab names to be used for different language settings. See **Translating Custom Centers, Tabs, Categories, and Links**.

15. Click **Save**.

16. Next, add links to your categories. See **Creating Center Links** for details.

**Note:** If you return to the tab and do not see the changes you have made, clear your browser cache. To do so in Internet Explorer, click **Tools**, and select **Internet Options**. In the **Temporary Internet Files** section, click **Delete Files**, and then click **OK**.

You can use SuiteCloud Development Framework (SDF) to manage center tabs as part of file-based customization projects. For information about SDF, see the help topic **SuiteCloud Development Framework Overview**. You can use the Copy to Account feature to copy an individual center tab to another
Creating Center Categories

Use the following steps to add custom categories to standard, built-in NetSuite tabs. After creating custom categories for standard tabs, you can then add links to each category. The links can go to NetSuite pages, custom records, Suitelets, or external websites. You can only customize center categories if the Custom Records feature is enabled.

**Note:** To add custom categories to custom tabs, see Creating Center Tabs. On the Categories subtab, in the Label column, enter a name for a category of links. After creating your categories, you will later follow the steps in Creating Center Links to assign specific links to each category.

**To add custom categories to standard tabs:**

1. Go to Customization > Center and Tabs > Center Categories > New. A Center Category page appears.

2. On the Center Category page:
   a. In the **Label** field, provide a UI label for the category.
   b. In the **ID** field, provide an ID for the category if desired.
   c. From the **Center Type** list, choose an existing center.
   d. From the **Center Tab** list, choose one of the standard, built-in NetSuite tabs (also referred to as sections).
   e. From the **Insert Before** list, specify where you want to insert your custom category.
   f. From the **Link** list, choose the appropriate link.

   The links that appear in the lists are links to other NetSuite pages, custom records, Suitelets, and external web pages. Links to NetSuite pages, custom records, and Suitelets will appear by default.

   If you want to select a link to an external website, you must have already created that link by going to Customization > Centers and Tabs > Center Link. (See Creating Center Links for details.) After the link is created, it appears in the **Link** list.
Creating Center Categories

g. In the **Label** field, provide a UI label for the link.

h. In the **Short List** column, to indicate that the link should appear in a portlet when the link is in a narrow column of the dashboard, check the box.

**Note:** If the Multi-Languages feature is enabled in your account, you can define translated category labels to be used for different language settings. See Translating Custom Centers, Tabs, Categories, and Links.

3. Click **Add**.
4. Click **Save**.

**Note:** If you return to the tab and do not see the changes you have made, clear your browser cache. To do so in Internet Explorer, click **Tools**, and select **Internet Options**. In the **Temporary Internet Files** section, click **Delete Files**, and then click **OK**.

You can use SuiteCloud Development Framework (SDF) to manage custom center categories as part of file-based customization projects. For information about SDF, see the help topic SuiteCloud Development Framework Overview. You can use the Copy to Account feature to copy an individual custom center category to another of your accounts. Each custom center category page has a clickable Copy to Account option in the upper right corner. For information about Copy to Account, see the help topic Copy to Account Overview.

Creating Center Links

You can create links that appear on your own custom categories as well as on standard, built-in NetSuite categories. Center links can take users to other pages within NetSuite, custom records, Suitelets, or external websites. You can only customize center links if the Custom Records feature is enabled.

To create links to NetSuite pages, custom records, and Suitelets:

1. Go to Customization > Centers and Tabs > Center Categories.
2. On the Custom Center Categories list page, click **Edit** next to the category you want to add links to.
3. On the Custom Center Category page, in the **Link** field, select the NetSuite page, custom record, or Suitelet you want to link to. If you are adding a link to a NetSuite page, make sure to use an
account-specific domain. For more information, see Account-Specific Domains in Custom Center Links

**Note:** For custom records, you can also create links using the Links subtab on the Custom Record Type definition page. If you want a link to appear on a standard category, you must set the link on the Custom Record Type definition page. Also, if you want to copy the center link and custom record to another account, you must set the link on the Custom Record Type definition page. See Creating Links to Custom Records for details. Also note that a link to a custom record type does not display for users that do not have permission to access that custom record type.

4. Provide a UI label for the link. The label appears off to the side of the category in the UI.

**Note:** If the Multi-Languages feature is enabled in your account, you can define translated link labels to be used for different language settings. See Translating Custom Centers, Tabs, Categories, and Links.

5. Click Add.

6. Add all of the center links required.

7. Click Save.

**To create links to web pages:**

1. Go to the website or NetSuite page for which you want to create a link. Right-click the address bar and select Copy.

2. Go to Customization > Centers and Tabs > Center Links.

3. In the **Label** column, enter a name for your custom link.

4. In the **URL** column, enter the URL for the link.

5. Click Add.

**Note:** The link will open in the existing window. You cannot set up a link to open in a new window.

6. Add all of the web links required.

7. Click Save.

8. Add the link to a custom category by going to Customization > Centers and Tabs > Center Categories.

9. Click Edit next to the category.
10. From the Link list, select the label of your custom link.
11. Click Add.
12. Click Save.

Translating Custom Centers, Tabs, Categories, and Links

If the Multi-Language feature is enabled in your account, you can translate labels for custom centers, tabs, categories, and links, so that they match the language of the NetSuite user interface, set by each user at Home > Set Preferences. For details, see the following:

- Translating Custom Centers
- Translating Custom Center Tabs
- Translating Custom Center Categories
- Translating Custom Center Links
- Displaying Translated Centers

**Important:** Before you can translate these labels, you need to select translation languages at Setup > Company > Preferences > General Preferences, on the Languages subtab. The Languages subtab lists both system-supported languages that can be used for the NetSuite user interface (and are available at Home > Set Preferences), and additional languages that can used for website translations only (and are not available at Home > Set Preferences). You should only enter translations for system-supported languages, because these are the only languages that can be displayed in the user interface. For details, see the help topic Configuring Multiple Languages.

Translating Custom Centers

You can define translated labels for a custom center when you edit the center. (You must first create and save it.) After the center has been created, go to Customization > Centers and Tabs > Centers, click Edit, and enter translated labels on the Translation subtab.
Translating Custom Center Tabs

You can define translated labels for a custom center tab when you first create it or when you edit it later. Go to Customization > Centers and Tabs > Center Tabs > New and enter translated labels on the Translation subtab. These labels translate the Name field.

Translating Custom Center Categories

You can define translated labels for a custom center category when you first create it or when you edit it later. Go to Customization > Centers and Tabs > Center Categories > New and enter translated labels on the Translation subtab. These labels translate the Category field.
Translating Custom Center Links

You can define translated labels for custom center links on the Custom Center Category page that lists the links. Go to Customization > Centers and Tabs > Center Categories, select a center category, and enter link translations on the Values subtab.

Displaying Translated Centers

The following screenshot shows a custom center, tab, category, and link in U.S. English:
Assigning a Custom Center to a Custom Role

You can assign the custom center to a custom role.

**Note:** You can assign a custom center only to a new custom role. If you customize an existing NetSuite role, the standard center is used.

To assign a custom center to a custom role:

1. Go to Setup > User/Roles > Manage Roles > New. A Role window appears.
2. Enter a name and ID for the new role.
3. In the **Center Type** field, select the custom center that you created.
4. Enter all of the other required settings for the new role. For information, see the help topic [Customizing or Creating NetSuite Roles](#).
5. Click **Save**.

**Important:** When you create a custom role and click **Save**, you cannot change the center assigned to that role.

Testing a new Custom Center

To see the new custom center, you assign the new role to yourself and switch to the new role.

To test a new custom center:

1. Go to List > Employees > Employees.
2. Click **Edit** next to your name.
3. Click the **Access** subtab.
4. In the **Role** field, select the new role that you created.
5. Click **Add**.
6. Click **Save**.
7. Switch to the new role. For information, see the help topic **Switching Between Roles**. The dashboard for the role appears with your custom center.
8. Test the links in the new custom center.
Deploying Upgraded Forms

NetSuite administrators can use the Upgrade Checklist to preview custom forms with 2010.2 Form Layout Enhancements applied. These enhancements include Field Groups and the standardization of form Subtabs and Sublists. After previewing the new layout, administrators can then deploy the upgraded custom forms to end users.

To access the Upgrade Checklist, go to Customization > Forms > Entry Forms [or Transaction Forms], and click the link in the message area at the top. If you have already upgraded forms in your account, the link in the message area is called Return to Upgrade Checklist.

After you deploy an upgraded form, you cannot “roll back” the deployment. When you deploy an upgraded form, you are replacing the existing form.

**Important:** As of 2012.2, all standard forms have been automatically upgraded to use form layout enhancements, so there is no need for administrators to deploy upgraded standard forms. Also, if your NetSuite account was established in 2010.2 or later, form layout enhancements are automatically applied to custom forms as well. The custom form deployment process described here is applicable only to NetSuite accounts established prior to 2010.2.

**Important:** SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should only access the NetSuite UI by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic SuiteScript 2.0 Custom Pages.

Review the following to get an understanding of form layout enhancements and the process for previewing, upgrading, and deploying custom forms to apply these enhancements:

- **Form Layout Enhancements** – Summarizes the changes that are made when you deploy upgraded forms.
- **Custom Form Deployment Process (Summary)** – Provides a high-level overview of the deployment process.
- **Custom Form Deployment Process (In Detail)** – Provides a workflow diagram that depicts each step in this process.

See the following for detailed steps:

- Deploying Upgraded Custom Forms
- Understanding Form Deployment Statuses
- Understanding Form Layout Enhancement Upgrade Logic

**Form Layout Enhancements**

Version 2010.2 introduced a new user interface (UI), which included a set of changes categorized as **Form Layout Enhancements**. These form layout enhancements were not automatically applied to custom forms that were created prior to 2010.2. NetSuite administrators have the option of applying form layout enhancements to each custom form, as described in Deploying Upgraded Custom Forms.

Form layout enhancements include the following:
The addition of Field Groups to organize all fields on a record into logical groups.

The consistent naming and placement of Subtabs and Sublists.

NetSuite administrators can upgrade the transaction and entry forms in their account to include these enhancements. Administrators must then deploy the upgraded forms to their NetSuite users before the users can begin working with the enhanced forms.

Administrators should use the Upgrade Checklist to manage the form upgrade and deployment processes. If you are an account administrator, see Deploying Upgraded Forms to learn more.

Important: As of 2012.2, form layout enhancements have been applied to all standard forms.

Form Layout Enhancements (The Big Picture)

The following figures show the difference between a form that has been upgraded to include the Form Layout Enhancements and one that has not.

The first figure is a view of a sales order with a custom sales order form applied. In this figure, the Field Groups and the Subtabs and Sublists changes associated with the Form Layout Enhancements have not yet been applied to the custom form used for this record.

The second figure shows the same record, but with an “upgraded” standard sales order form. The upgraded form now includes:

- **Field Groups** – All fields in the main header area are organized into the Primary Information, Sales Information, and Classification field groups. Based on the data in the record, NetSuite automatically created these field groups when the account administrator upgraded the form.
Updated **Subtabs and Sublists** – Subtab and sublist data have been reorganized into more meaningful categories. In this example, the Address, Payment, Messages, and History tabs have been removed. Content that was previously on these subtabs has been moved to the main header area or other subtabs/sublists. The Billing, Accounting, Relationships, Communication, Related Records, and System Information tabs have been added.
Field Groups

When your account administrator upgrades your existing forms to include the Form Layout Enhancements, the fields on your record pages are reorganized into field groups. Field groups organize related data into logical groups.

The following figure shows fields organized into six different field groups. By default, the Primary Information field group will appear on all record pages when the Form Layout Enhancements are deployed by your account administrator. Some of the other field groups on this page may also appear, depending on the form type, the data in the form, and the features enabled in your account.

In the main body area of the standard vendor form (shown below), fields have been automatically organized into the Primary Information, Email | Phone | Address, and Classification field groups.

Notice that field groups can also appear on subtabs. The Financial subtab shows the Account Information, Balance Information, Tax Information, and Project Information field groups.
Administrators can use SuiteBuilder to customize field group titles, order, and layout. See Configuring Field Groups in the NetSuite Help Center for information on customizing field groups.

When administrators upgrade the standard and custom transaction and entry forms in your account, NetSuite automatically groups all standard, built-in fields. Custom fields remain on the subtabs they were previously assigned to, unless they were assigned to subtabs that have been removed. In this case, after the upgrade occurs, custom fields will automatically be placed on a subtab called Custom. Account administrators can then reassign these fields to other subtabs if they choose.

For detailed information on how standard and custom fields are reorganized into field groups, NetSuite administrators should see Understanding Form Layout Enhancement Upgrade Logic in the NetSuite Help Center.

Subtabs and Sublists

After administrators deploy a form with the Form Layout Enhancements applied, the subtabs and sublists that appear on the form will change. End users will notice that new subtabs and sublists have been added (see Added Subtabs and Sublists). They will also notice that certain subtabs and sublists have been removed (see Removed Subtabs and Sublists).

The subtab and sublist changes associated with the Form Layout Enhancements were implemented by NetSuite to create more consistency across the application. Across all records, subtabs and sublists now have a more consistent pattern of naming and placement.

Added Subtabs and Sublists

If you are viewing forms with the Form Layout Enhancements applied, you will notice that the following subtabs have been added:

- **Access** (appears on all entity records when the record is in View or Edit mode)
Form Layout Enhancements

- **Accounting** (appears on certain entity, item, and transaction forms)
- **Billing** (appears on certain transaction forms)
- **Builds** (appears on certain transaction forms)
- **Communication** (appears on certain entity, item, and transaction forms and will contain these subtabs in a consistent order)
  - Messages (appears when record is in View or Edit mode)
  - Activities (appears when record is in View or Edit mode)
  - Events (appears when record is in New)
  - Tasks (appears when record is in New)
  - Calls (appears when record is in New)
  - Files
  - User Notes
  - Bulk Merge (often hidden by default)
- **Fulfillments and Credits** (appears on certain transaction forms)
- **Fulfillments and Receipts** (appears on certain transaction forms)
- **Journal** (appears on certain transaction forms)
- **Locations** (appears on certain item forms)
- **Message** (appears on certain CRM forms)
- **Preferences** (appears on certain entity and item forms)
- **Purchasing / Inventory** (appears on certain item forms)
- **Receipts and Refunds** (appears on certain transaction forms)
- **Relationships** (appears on certain entity and transaction forms and will contain all entities associated with the primary record. The entities appearing on this subtab will vary depending on the primary record you are on.)
- **Related Records** (appears on certain entity, item, CRM, and transaction forms, and will contain transactions and other records associated with the primary record. The records appearing on this subtab will vary depending on the primary record you are on.)
- **Sales** (appears on certain transaction forms)
- **Subscriptions** (appears on certain entity forms)
- **System Information** (appears on certain entity, item, CRM, and transaction forms, and will contain these subtabs in a consistent order)
  - System Notes (appears when record is in View or Edit mode)
  - Workflow
  - Translation (only appears on certain items)
- **Time Tracking** (appears on some entity records)
- **Vendors** (appears on some item records)

**Important:** Be aware that the Date Created field and the Inactive field both appear on the System Information subtab after the Form Layout Enhancements have been deployed to your account.

**Removed Subtabs and Sublists**

If your account administrator has not yet deployed Form Layout Enhancements to your account, you will notice that there are no changes to the organization of the fields or subtabs on a page.
However, if you are viewing forms with the Form Layout Enhancements applied (meaning that your account administrator has deployed the upgraded forms to your account), you will notice the following subtabs have been removed:

**Entity Forms:**
- General
- Info

**Item Forms:**
- Basic
- History
- Rev Rec / Amort (note: subtab fields have been moved to the Revenue Recognition / Amortization subtab)
- Specials (the fields on this subtab are moved to the Web Store subtab when you deploy the upgraded form)

**CRM Forms:**
- General
- History

**Transaction Forms:**
- Carrier
- General
- History
- International
- Packages
- Revenue (the fields on this subtab are moved to the new Accounting subtab when you deploy the upgraded form)

**Other subtabs not removed but relabeled:**
- Related Info (CRM forms) relabeled to Related Records

Fields that used to reside on these subtabs have been moved to new locations on the form or to another subtab.

**Important:** If you are a SuiteScript developer and you have referenced any of these deprecated tabs in your scripts, you will need to modify your scripts to reference existing subtabs. Note that the IDs for all new subtabs are provided in the SuiteScript Records Browser.

---

**Custom Form Deployment Process (Summary)**

The custom form deployment process involves previewing each custom form, further modifying the layout of each form (if necessary), testing each form in the context of a record, and then deploying the form to end users. Custom forms must be deployed individually.

The following diagram summarizes the form deployment process. Use the NetSuite Upgrade Checklist to manage this process. To access the Upgrade Checklist, go to Customization > Forms > Entry Forms [or Transaction Forms], and click the link in the message area at the top. If you have already upgraded forms in your account, the link in the message area is called Return to Upgrade Checklist.
Custom Form Deployment Process (In Detail)

You begin the custom form deployment process by going to the Upgrade Checklist.

To access the Upgrade Checklist for custom transaction forms, go to Customization > Forms > Transaction Forms, and click the Upgrade Checklist link. To access the Upgrade Checklist for custom entry forms, go to Customization > Forms > Entry Forms, and click the Upgrade Checklist link. If you have already upgraded forms in your account, the link is called Return to Upgrade Checklist.

Important: If you do not see this link in the upper right of the custom transaction forms or custom entry forms list, it means your account does not include any custom forms that require upgrading.

The following diagram shows the entire workflow for deploying upgraded forms.
The following steps outline the process for deploying upgraded forms.

1. Open the Upgrade Checklist. Go to Setup > Customization > Transaction or Entry Forms.
2. In the Preview Forms column, click the link for the custom form.
3. To edit the new layout of the form, click **Edit Layout**, make changes, and click **Save**. Preview the form by clicking the link for the form in the Preview Name column.

4. When you have finished editing the form layout, click **Enable Test Mode**. Test the form to ensure that you get the expected output.

5. When testing is complete, click **Disable Test Mode**.

6. To deploy the form, click the **Deploy Form** link next to the form.

After reviewing the overall form deployment workflow, go to **Deploying Upgraded Custom Forms** to begin the form upgrade and deployment processes.

### Deploying Upgraded Custom Forms

Use the following steps to deploy custom forms that include **Form Layout Enhancements**. When you deploy upgraded custom forms to end users, NetSuite **automatically** applies the layout enhancements to the form. These enhancements include **Field Groups** and the standardization of form **Subtabs and Sublists**.

**Note:** As of 2012.2, all standard forms have been automatically deployed to use form layout enhancements, so there is no need to deploy standard forms. Also, if your NetSuite account was established in 2010.2 or later, form layout enhancements are automatically applied to custom forms as well. The custom form deployment process described here is applicable only to NetSuite accounts established prior to 2010.2.

To deploy upgraded custom forms, see these sections in the following order:

1. Previewing Undeployed Custom Forms
2. Editing the Layout of Custom Forms Prior to Deployment
3. Avoid Editing Custom Forms in Tandem
4. Testing Undeployed Custom Forms
5. Deploying Custom Forms

Be aware of the following:

- You are NOT required to upgrade any of your custom forms if you prefer that they keep their existing layout. If you choose not to upgrade a custom form, you can take no action at all.
- As of 2012.2, the form upgrade process has been simplified to remove the Skip Upgrade option. If prior to that release you used this option, and later decide you want Form Layout Enhancements applied to the form, you can perform an Undo Skip operation (see **Deploying Skipped Custom Forms** for details.) However, if you have added custom field groups to a “skipped” form, you will lose all field group formatting when you deploy the upgraded custom form. The fields in the field groups will remain, however they will be reorganized into auto-generated field groups (which you can later customize).

### Previewing Undeployed Custom Forms

Before deploying upgraded custom forms to end users, NetSuite lets you preview what the forms will look like after you deploy them. The preview process for custom forms is **extremely important**, as it lets you consider the following:

- Do you like how NetSuite has applied the **Form Layout Enhancements** to your custom forms?
- Do you see additional layout modifications you need to make to the form so that it suits your business needs?
- Do you want users to start working with the upgraded custom form? (If so, you will deploy the form.)
Would you prefer to keep the custom form in its existing layout?

To preview undeployed custom forms:

1. Navigate to the Upgrade Checklist:
   a. Go to Customization > Forms > Transaction (or Entry) Forms.
   b. On the forms list page, in the right column near the top, click the Upgrade Checklist link.
      If you have already upgraded forms in your account, the link in the message area is called Return to Upgrade Checklist.

2. In the Preview Form column, click the custom form you want to preview.
   The form that appears will have the form layout enhancements applied according to the logic described in Understanding Form Layout Enhancement Upgrade Logic.

3. If you generally approve of the upgraded layout, but see a few areas you need to modify, see Editing the Layout of Custom Forms Prior to Deployment.
   If you decide you want to retain the existing layout that your end users are currently working with, click Cancel and take no further action.

Editing the Layout of Custom Forms Prior to Deployment

After previewing a custom form, you will probably notice parts of the layout you want to modify. You will go through the Upgrade Checklist to make these modifications. If you do not see this link in the upper right of the custom transaction forms or custom entry forms list, it means your account does not include any custom forms that require upgrading.

Important: As of 2012.2, all standard forms have been automatically upgraded to use form layout enhancements, so there is no need for administrators to deploy upgraded standard forms. Also, if your NetSuite account was established in 2010.2 or later, form layout enhancements are automatically applied to custom forms as well. The custom form deployment process described here is applicable only to NetSuite accounts established prior to 2010.2.

To edit the layout of undeployed custom forms:

1. Navigate to the Upgrade Checklist:
Deploying Upgraded Custom Forms

1. Go to Customization > Forms > Transaction (or Entry) Forms.
   a. On the forms list page, in the message area at the top, click the Upgrade Checklist link. If you have already upgraded forms in your account, the link in the message area is called Return to Upgrade Checklist.

2. In the Edit Layout column, click Edit next to the custom form you have previewed and now want to modify.

   ![Information](image)

   **Information**
   Use the Upgrade Checklist to apply Form Layout Enhancements to the forms in your account. Through the Upgrade Checklist, you can preview and further modify upgraded custom forms. Then, if you choose, you can replace the existing custom form in your and users’ accounts by deploying the upgraded custom forms. To guide you through this process, see Deploying Upgraded Forms in the NetSuite Help Center.

   ![Upgrade Checklist: Review and Deploy Transaction Forms](image)

   **Upgrade Checklist: Review and Deploy Transaction Forms**

   ![Table](image)

   **Table:**
<table>
<thead>
<tr>
<th>Edit Layout</th>
<th>Custom Form Name</th>
<th>Status</th>
<th>Deploy Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit</td>
<td>Progress Invoice</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
<tr>
<td>Edit</td>
<td>Service Invoice</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
<tr>
<td>Edit</td>
<td>Professional Invoice</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
<tr>
<td>Edit</td>
<td>Custom Service Invoice</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
<tr>
<td></td>
<td>Custom Product Invoice</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
<tr>
<td></td>
<td>Custom Journal Entry</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
<tr>
<td></td>
<td>Standard Opportunity</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
<tr>
<td></td>
<td>Custom Opportunity</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
<tr>
<td></td>
<td>Custom Opportunity 2</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
<tr>
<td></td>
<td>Standard Drop Ship Purchase Order</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
<tr>
<td></td>
<td>Standard Purchase Order</td>
<td>Deployed</td>
<td>Deploy Form</td>
</tr>
</tbody>
</table>

3. Use the page that appears to make your additional layout changes (see figure below).

   ![Important](image)

   **Important:** The custom form edit page you access through the Upgrade Checklist should be used to modify ONLY the layout aspects of the form. Use this page to modify tabs, field groups, screen fields, and lists. Do not use this page to modify any other properties of the form! All other form properties related to roles, form default settings and permissions are inherited from the custom form currently in use. If you want to change any of these settings, you must do so by accessing the form through the custom forms list page (Customization > Forms > Transaction (or Entry) Forms).

4. After making your layout changes, click Save.

5. Navigate back to the Upgrade Checklist:
   a. Go to Customization > Forms > Transaction (or Entry) Forms.
   b. On the forms list page, in the message area at the top, click Upgrade Checklist. If you have already upgraded forms in your account, the link in the message area is called Return to Upgrade Checklist.

6. Preview the form. The layout changes you made in Step 3 will reflect in the preview.

7. If you want to further modify the layout, repeat steps Step 2–Step 6.

8. When you are satisfied with the layout, you can test the form to verify it works as expected. See Testing Undeployed Custom Forms for details.

SuiteBuilder
Avoid Editing Custom Forms in Tandem

NetSuite does not recommend that you simultaneously edit the forms that are currently in use (in your production account) and their corresponding forms accessed through the Upgrade Checklist. For example, it is not recommended that you edit the layout of Custom Form A that is currently used in production, during the time that you also edit the “preview version” of Custom Form A accessible through the Upgrade Checklist. Editing both versions will cause the undeployed preview version to get out of synch with the version in production.

Instead, you should pick one version of the form to modify: either the version currently in production, or the (upgraded) undeployed preview version accessed through the Upgrade Checklist.

If you choose to edit the production version, then when you preview the form through the Upgrade Checklist, the preview will show the Form Layout Enhancements – as they apply to ALL the modifications you have made to the form. Note that if you continue to modify the production version after you have previewed it, the two versions of the form will be out of sync.

If you choose to make all form layout edits to the upgraded/undeployed “preview” version, you will know that when you are ready to deploy the form, none of the modifications you have made will be lost. This cannot be said of the modifications made to the “production” version currently in use.

Testing Undeployed Custom Forms

When you test an undeployed custom form, you are viewing the form in the context of a record page. When in test mode, you have the opportunity of viewing the form as it appears in New mode and Edit mode.

When it comes to testing SuiteScripts on custom forms, you will be testing any client or user event scripts associated with the record type or the form.

To test undeployed custom forms:

1. Navigate to the Upgrade Checklist:
   a. Go to Customization > Forms > Transaction (or Entry) Forms.
   b. On the forms list page, in the message area at the top, click Upgrade Checklist. If you have already upgraded forms in your account, the link in the message area is called Return to Upgrade Checklist.

2. Click the Enable Test Mode button on the bottom of the Upgrade Checklist.

3. Use the navigation menus to go to the form you want to test.
   For example, if you want to test a custom sales order form, do one or all of the following:
   - Go to Transactions > Sales > Enter Sales Orders > New (to test a form in New mode)
   - Go to Transactions > Sales > Enter Sales Orders > List > and click Edit next to an existing record (to test a form in Edit mode)

4. In the page that appears, from the Custom Form list, select the custom form you want to test (see figure).

   **Note:** When in Test Mode, all custom forms that appear in the Custom Form list will appear with the upgraded layout. This includes both deployed and undeployed custom forms.

5. Notice that the page layout changes to include field group and subtab Form Layout Enhancements. Verify that the upgraded layout will suit your business needs.
Deploying Upgraded Custom Forms

6. If you have any client or user event scripts associated with the form or the record type you are testing, verify that the scripts execute as expected.

7. After testing your custom form, navigate back to the Upgrade Checklist by going to Customization > Forms > Transaction (or Entry) Forms, and clicking Upgrade Checklist at the top of the page. If you have already upgraded forms in your account, the link in the message area is called Return to Upgrade Checklist.

8. On the Upgrade Checklist, click the Disable Test Mode button to take the form out of test mode.

9. After testing a custom form, you can now deploy the upgraded form to end users. See Deploying Custom Forms for details.

Deploying Custom Forms

After you have previewed a custom form, edited its layout (if necessary), and tested it thoroughly, you can deploy the upgraded form into the accounts of all users.

After you deploy a custom form, you will notice that its status on the Upgrade Checklist changes to Deployed. You will also notice that all links to the form are removed from the Upgrade Checklist. After a custom form has been deployed, if you want to access the form to make further changes, you will access the form through the custom forms list page (Customization > Forms > Transactions Forms (or Entry Forms). The Upgrade Checklist is used to facilitate the form deployment process only. It is not meant to be used to access forms that have been deployed.

To deploy custom forms:

1. Navigate to the Upgrade Checklist:
   a. Go to Customization > Forms > Transaction (or Entry) Forms.
   b. On the forms list page, in the message area at the top, click the Upgrade Checklist link. If you have already upgraded forms in your account, the link in the message area is called Return to Upgrade Checklist.

2. In the Upgrade Checklist, in the Deploy Options column, click the Deploy Form link for the custom form you want to deploy.
   After clicking the link, notice that the status for the form changes to Deployed.

After you have deployed a custom form, the form will be available in the accounts of all of your end users. If the custom form you deploy is currently set as the “Preferred Form” for all users or for users with specific roles, your users will begin using this form immediately after it is deployed.

Deploying Skipped Custom Forms

Use the following steps to deploy a custom form that you had previously “skipped.”

Note that, as of 2012.2, the form upgrade process has been simplified to remove the Skip Upgrade option. The following procedure applies to any custom forms for which you skipped upgrade prior to that release.

**Important:** Field groups created on existing custom forms that were skipped will not be carried over in the new upgraded form. You will notice this when you preview the upgraded custom form.

To deploy skipped custom forms:

1. Navigate to the Upgrade Checklist:
a. Go to Customization > Forms > Transaction (or Entry) Forms.

b. On the forms list page, in the message area at the top, click Upgrade Checklist. If you have already upgraded forms in your account, the link in the message area is called Return to Upgrade Checklist.

2. Next to the custom form you want to deploy, in the Deploy Options column, click Undo Skip and Upgrade Form. After clicking this link, the Edit link reappears next to the form, and the form itself becomes a link.

3. In the Preview Name column, click the custom form to preview.

The form that appears will have the layout enhancements applied according to the NetSuite logic for applying these enhancements. See Understanding Form Layout Enhancement Upgrade Logic for details.

Notice the following:
- The new Field Groups that automatically render when you preview the form.
- The Subtabs and Sublists that have been added or removed.
- If you had previously added custom field groups to these forms, when you upgrade the layout and deploy the form, all standard fields you might have placed on a custom field group will automatically be put into the fields groups NetSuite considers appropriate. All custom fields you had placed in a custom field group will appear at the bottom of the main area of the form and will not have a field group title. You will have to manually edit those fields and place them on new field groups.

4. If you decide you approve of most of the changes to your form, but you need to make additional layout modifications before deploying the form to users, see Editing the Layout of Custom Forms Prior to Deployment.

5. Next, test the upgraded form. See Testing Undeployed Custom Forms.

6. After thoroughly previewing and testing the custom form, click the Deploy Form link.

Understanding Form Deployment Statuses

Form deployment statuses appear in the Upgrade Checklist and on the transaction and entry custom form list pages.

Upgrade Checklist Deployment Statuses

In the Status column on the Upgrade Checklist you will see these statuses for custom forms:

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; No status at all &gt;</td>
<td>The account administrator has not performed any action on the form. The form has not been deployed (by clicking the Deploy Form link).</td>
</tr>
<tr>
<td>Deployed</td>
<td>The account administrator has clicked the Deploy Form link and deployed the form. The deployed form includes the 2010.2 Form Layout Enhancements and is now in the accounts of all end users.</td>
</tr>
<tr>
<td>Skipped Upgrade</td>
<td>The account administrator has clicked the Skip Upgrade link and has explicitly chosen to skip the deployment of the form. The form will retain its existing layout.</td>
</tr>
</tbody>
</table>
Understanding Form Deployment Statuses

**Status**

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Important:</strong></td>
<td>As of 2012.2, the Skip Upgrade option is no longer available, but any form that was skipped prior to that release continues to display a status of Skipped Upgrade until the account administrator deploys the form.</td>
</tr>
</tbody>
</table>

Understanding Form Layout Enhancement

**Upgrade Logic**

When you preview custom forms, NetSuite executes a backend process that automatically applies form layout enhancements to the existing versions of the forms. These enhancements include the arrangement of fields into logical Field Groups and the standardization of Subtabs and Sublists.

The goal of the backend upgrade process is to create a form layout that is better organized and more consistent. The other goal is to upgrade the layout and maintain form customizations wherever possible.

See the following sections to learn about the NetSuite upgrade logic as it applies to custom forms, standard and custom fields, subtabs, and custom field record pages used to add or edit custom fields. These topics do not need to be read in order.

- Upgrade Logic for Custom Forms
- Upgrade Logic for Subtabs
- Upgrade Logic for Fields (Diagram)

For information on field ordering as it applies to form upgrades, see Field Ordering.

**Upgrade Logic for Custom Forms**

When the backend upgrade process runs against a custom form, the standard fields on the form are rearranged to match the new field location of the standard fields on the upgraded standard form. For example, when a custom Customer form is upgraded, the standard fields on the custom Customer form are rearranged to match the layout of the standard fields on the upgraded standard Customer form.

**Important:** The exception to this is standard fields placed on a custom subtab. When a custom form is upgraded, standard fields that have been placed on a custom subtab will remain on the custom subtab.

The location of custom fields on custom forms will also remain unchanged during the upgrade process. Custom fields will only be moved if they are on a subtab that is removed during the upgrade process.

See the following table for details. Also see the Upgrade Logic for Fields (Diagram) for a visual description of these field upgrade changes.

<table>
<thead>
<tr>
<th>Existing Form</th>
<th>Upgraded Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Custom Fields</td>
<td></td>
</tr>
<tr>
<td>Custom fields that are on a valid standard subtab.</td>
<td>Will be kept on the same subtab when the form is upgraded.</td>
</tr>
</tbody>
</table>
Existing Form | Upgraded Form
--- | ---
Custom fields on a subtab that is removed when the form is upgraded. | Will be moved to a subtab called **Custom**.
After previewing the new auto-generated layout, you can manually edit the layout and move your custom fields off of the Custom subtab.
See [Editing the Layout of Custom Forms Prior to Deployment](#).

Custom fields that are on a custom subtab. | Will be kept on the same custom subtab.

Custom fields that are in the main area of the form. | Will be placed at the bottom of the main area, below the auto-generated field groups.

After previewing upgraded **custom** forms, if you are not satisfied with the auto-generated layout, you can reassign fields to different groups, relabel or remove field groups, and create your own custom fields. You can also move fields between the new subtabs that were auto-generated. For more information, see [Previewing Undeployed Custom Forms](#) and [Editing the Layout of Custom Forms Prior to Deployment](#).

To create custom layouts, go to Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and click the Customize link next to a layout. Make your changes and click Save. You can choose default layouts to apply to one or more types of forms by checking box(es) in the Preferred column at Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and clicking Submit.

### Upgrade Logic for Subtabs

In addition to the creation of field groups on upgraded custom forms, the backend upgrade process also reorganizes and relabels certain subtabs and sublists. When forms are upgraded, certain subtabs are added (depending on the form type), and others are removed. Although certain subtabs are removed during the upgrade process, the data on these subtabs are moved to another (more logical) area on the form. This new location can be another subtab or it can be under a field group. (See [Subtabs and Sublists](#) for more information about the subtabs that are added or removed during the form upgrade process.)

To create custom layouts, go to Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and click the Customize link next to a layout. Make your changes and click Save. You can choose default layouts to apply to one or more types of forms by checking box(es) in the Preferred column at Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and clicking Submit.

### Upgrade Logic for Fields (Diagram)

The following diagram visually represents the field arrangement concepts discussed in Upgrade Logic for Custom Forms. Technically, all NetSuite fields appear on subtabs, even fields that appear in the main body of the form are (technically speaking) located on a subtab called **Main**.

The following diagram shows the subtab assignment for fields in a "V1" (non-upgraded) environment and a "V2" (upgraded) environment.

Note the following in the diagram:

- V1 indicates the existing form, and V2 indicates the upgraded form
- Boxes outlined in red highlight the scenarios where a field's location (subtab) changes when a form is upgraded.
- New Std Location means that NetSuite automatically moves the field to a new subtab, existing subtab, or to the Main subtab based on the field's data.
- Main is one example of a subtab that is valid for both V1 and V2 forms.
- General is one example of a subtab that is deprecated when forms are upgraded.
- Some standard fields that are common across forms have been moved to a new location for better logical grouping. The move is consistent across forms.

**Note:** To enlarge the diagram, press Ctrl+Plus Sign. To return your screen to its normal size, press Ctrl+Minus Sign.

To create custom layouts, go to Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and click the Customize link next to a layout. Make your changes and click Save. You can choose default layouts to apply to one or more types of forms by checking box(es) in the Preferred column at Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and clicking Submit.

### Field Ordering

Customized ordering of fields will not be respected when a form is upgraded. The exception is the order of fields on custom tabs. Their ordering will not change when Form Layout Enhancements are applied.

To create custom layouts, go to Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and click the Customize link next to a layout. Make your changes and click Save. You can choose default layouts to apply to one or more types of forms by checking box(es) in the Preferred column at Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and clicking Submit.