



ORACLE
NETSUITE

SuiteBuilder (Customization) Guide

2023.1

June 7, 2023



Copyright © 2005, 2023, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

If this document is in public or private pre-General Availability status:

This documentation is in pre-General Availability status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

If this document is in private pre-General Availability status:

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your pre-General Availability trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document may change and 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.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <https://www.oracle.com/corporate/accessibility>.

Access to Oracle Support

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

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

This documentation is in pre-General Availability status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your pre-General Availability trial agreement only. It is not a commitment to deliver any material, code, or functionality, and 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.

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	4
Record Types	7
Transaction Types	11
Segments	11
Advanced Templates	12
Centers	13
Customization Best Practices	15
Optimize Your Custom Fields and Custom Forms	15
Plan Ahead and Make Use of Existing Standard Features	15
Set Up Internal Controls	16
Know When to Use SuiteFlow and SuiteScript	16
Conventions for Naming Custom Objects	17
Changing the ID of a Custom Object	19
Customizing Field Level Help for Standard Fields	20
Custom Fields	23
Custom Fields Videos	24
Field Type Descriptions for Custom Fields	25
Custom Field Types	29
Available Standard Fields and Field Types	30
Transaction Body Fields	31
Transaction Line Fields	33
Transaction Item Options	34
CRM Fields	35
Entity Fields	36
Item Fields	37
Creating a Custom Field	38
Creating Custom Fields by Type	41
Assigning Custom Fields to Specific Record Types	58
Setting Display Options for Custom Fields	60
Setting Validation and Defaulting Properties	64
Setting Sourcing Criteria	68
Sourcing and Filtering Examples	72
Setting Filtering Criteria	74
Dependent Dropdown Lists	76
Restricting Access to Custom Fields	80
Restricting Access to Employee Custom Fields	82
Creating Read-Only Custom Fields	83
Adding Translations for Custom Fields	83
Adding Custom Fields to Transaction Forms	85
Tracking Changes to Custom Fields	85
Inactivating a Custom Field	86
Editing a Custom Field	87
Maintaining Saved Searches that Include Edited Custom Fields	89
Converting the Field Type of a Custom Field	89
Account-Specific Domains in Custom Fields	92
Renaming Custom Fields	92
Mass Updating Custom Fields	93
Advanced Features for Custom Fields	93
Encrypted Custom Field Stored Values	94
Creating Custom Fields with Values Derived from Summary Search Results	95
Dynamic Defaults and Dynamic Hyperlinks	98

Creating Formula Fields	101
Custom Lists	106
Creating a Custom List	106
Adding Translations for Custom Lists	107
Managing Large Custom Lists	109
Custom Forms	111
Form Templates	112
Creating Custom Entry and Transaction Forms	114
Custom Entry Form Properties	115
Custom Transaction Forms Properties	116
Storing Custom Forms with Transactions	118
Configuring Subtabs for Custom Entry and Transaction Forms	120
Moving Fields and Lists Between Subtabs	120
Configuring Field Groups	121
Configuring Fields or Screens	124
Configuring Buttons and Actions	127
Configuring Printing Fields	130
Configuring Sublists	131
Configuring Sublist Fields	132
Configuring QuickViews	132
Associating Custom Code (Client SuiteScript) Files With Custom Forms	135
Defining Preferred Forms	136
Adding Disclaimers to Transaction Form Footers	138
Specifying Check Layout by Subsidiary	138
Customizing Multiple Page Transaction Forms	139
Linking Transaction Forms	140
Transaction Form Printing Preferences	140
Creating Custom Subtabs	142
Creating a Subtab from the Custom Subtabs Page	142
Creating Custom Note Forms	144
Customizing Address Forms	145
Scripting Billing and Shipping Addresses	146
Creating Custom Address Fields	147
Country-Specific Address Forms	147
Custom Sublists	147
Saved Searches for Custom Sublists	148
Applying Custom Sublists to Standard Records	149
Applying Custom Sublists in SDF-Enabled Accounts	152
Applying Custom Sublists to Custom Record Types	152
Custom Child Record Sublist Creation with SuiteScript	154
Customizing a Transaction Sublist	164
Advanced PDF/HTML Templates	167
Enabling the Advanced PDF/HTML Templates Feature	168
Reviewing Available Advanced Templates	169
Advanced PDF/HTML Multi-Currency Statement Templates	170
Setting Custom Forms to Use Advanced Templates	171
Advanced Templates Customization in the Template Editor	173
Viewing an Advanced Template in the Template Editor	174
Template Setup Window	177
Previewing Advanced PDF/HTML Templates	177
Saving an Advanced Template	179
Error Messages in Advanced Templates	180
WYSIWYG Editing in the Template Editor	181
Source Code Editing in the Template Editor	191
Scripting with Advanced Templates	222

Using Custom Data Sources for Advanced Printing	223
Using SuiteScript for Transaction Records	229
Using SuiteScript to Apply Advanced Templates to Non-Transaction Records	231
Using SuiteScript 2.x to Combine Multiple Data Sources in One Advanced Template	232
Changing the Script ID of a Custom Template	234
Advanced Templates for Printing Saved Search Results	235
Advanced Templates Support for Company Printing Preferences	238
FAQs for Advanced Printing	239
Basic Printing Layouts	242
Customizing Transaction Form PDF Layouts	242
Defining Custom Elements	246
Configuring Borders and Placement	247
Formatting Label Text	248
Formatting Data Text	249
Using a Standard #10 Window Envelope With Transactions	249
Transaction Form HTML Layouts	251
Totals Transaction Form Layouts	252
Custom Records	254
Creating Custom Record Types	255
Creating a New Custom Record Type	256
Entering Name and Display Settings	256
Specifying Permission and UI Settings	257
Configuring File and Child Record Settings	259
Defining Search and Edit Settings	260
Adding Fields to Custom Record Types	265
Limiting Search Access to Custom Records	266
Applying Role-Based Restrictions to Custom Records	267
Adding Subtabs to a Custom Record	268
Choosing an Icon for a Custom Record	269
Numbering Custom Record Types	273
Adding Custom Forms for a Record	274
Online Custom Record Forms	274
Setting Up a Permissions List for a Custom Record Type	280
Creating Links to Custom Records	282
Adding Translations for Custom Records	283
Parent-Child Record Relationships	286
Establishing a Parent-Child Relationship	287
Creating a Parent-Child Relationship	288
Types of Parent-Child Relationships	289
Using Child Records	295
Sourcing with Custom Records	296
Available Standard Fields and Field Types for Custom Record Types and Source Lists	298
Updating Custom Record Types	299
Viewing or Editing a Custom Record Type	300
Updating Custom Record Type Print Templates	301
Custom Record Types Associated with a Custom Segment	302
Using Custom Record Entries	303
Searching Custom Record Entries	306
Record Customization (Beta)	307
Navigating Record Customization (Beta)	308
Record Customization (Beta) Page	310
Record Customization Overview Page	311
Record Customization Fields and Sublists Page	312
Record Customization Access Page	316
Custom Transactions	321

Benefits of Custom Transaction Types	321
Custom Transaction Type Naming Enables Better Organization	322
Custom Transaction Types Support Key NetSuite Features	322
Multiple Custom Transaction Styles Supported	324
Sales and Purchase Transaction Types Overview	325
Sales and Purchase Functionality Available in Custom Transactions	326
Sales and Purchase Custom Transaction Types in Integrations	328
Custom Transaction Type Setup	332
Enabling the Custom Transactions Feature	332
Granting a Role Permission to Manage Custom Transaction Types	332
Creating and Editing Custom Transaction Types	333
Custom Transaction Styles Overview	333
Creating a Custom Transaction Type	336
Editing a Custom Transaction Type	339
Locked Custom Transaction Types	341
Custom Transaction Type Classification Fields	342
Custom Fields in Custom Transaction Types	343
Numbering for a Custom Transaction Type	344
Account Field Setup for Custom Transaction Types	353
Statuses for a Custom Transaction Type	355
Creating Links for a Custom Transaction Type	360
Adding Custom Forms for a Custom Transaction Type	362
Permissions for Custom Transaction Instances	364
Adding Translations for a Custom Transaction Type	366
Custom Transaction Type Association with a Custom GL-Lines Plug-in Implementation	366
Custom Transaction Type Association with a SuiteTax Plug-in	367
Deleting Custom Transaction Types	367
Creating Sales and Purchase Custom Transaction Instances	369
Creating a Sales Custom Transaction Instance	369
Creating a Purchase Custom Transaction Instance	374
Printing Custom Transaction Instances	378
Custom Transaction Types in Workflows	379
Custom Segments	384
Benefits of Custom Segments	384
Configure Segment Values to Be Hierarchical	386
Configure a Segment to Default Statically or Dynamically	386
Filter a Segments Values Based on Other Segments	386
Filter a Segment's Values Based on Class, Department, Location, and Subsidiary	387
Configure a Segment to Appear on the GL Impact Page	388
Create Segments as Multi-Select Fields	388
Display Segments Selectively	388
Set Custom Segment Values	389
Custom Segments Overview	389
Enabling the Custom Segments Feature	389
Transaction Types Supported by Custom Segments	389
Settings that Affect Where Custom Segments are Applied	392
Permissions for Managing Custom Segments and Values	393
Granting a Role Permission to Manage Custom Segments	394
Granting a Role Permission to Manage Custom Segment Values	396
Custom Segment Creation	398
Custom Segment Types	398
Creating a Custom Segment	399
Configuring GL Impact for a Custom Segment	403
Filtering for a Custom Segment	404
Applying a Custom Segment to Record Types	410

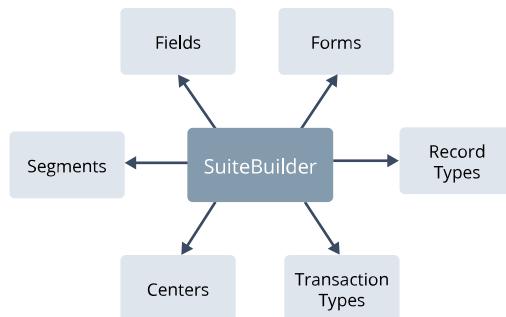
Apply to Kit or Assembly Components Setting for Custom Segments and Transaction Line	
Custom Fields	411
Dynamic Default Value Sourcing for Custom Segments	413
Validation and Static Default Values for Custom Segments	419
User Permissions for a Custom Segment	420
Dependent Segments	423
Setting Display Order of All Custom Segments	423
Editing Custom Segments	424
Required Permissions for Editing Custom Segments	424
Inactivating a Custom Segment	425
Deleting a Custom Segment Definition	426
Custom Segment Values	426
Value Creation for Custom Segments	427
Creating Hierarchies Among a Custom Segment's Values	431
Edit a Custom Segment's Values	432
Changing the Order of a Custom Segment's Values	434
Making a Custom Segment's Values Inactive	435
Deleting a Custom Segment's Values	436
Using the Script ID to Access Custom Segment Body, Line, and Filter By Fields	437
Custom Segments in Record Searches	437
Customizing a Report by Using Custom Segments	438
Using Custom Segments in Workflows	439
Custom Segments as Fields	439
Records that Represent Custom Segment Values	441
SuiteScript and Custom Segments	442
SOAP Web Services and Custom Segments	442
CSV Import and Custom Segments	442
Custom Centers	444
Creating and Editing Custom Centers	445
Creating Center Tabs	446
Creating Center Categories	448
Creating Center Links	450
Assigning a Custom Center to a Custom Role	452
Deploying Upgraded Forms	454
Form Layout Enhancements	454
Field Groups	457
Sublists and Subtabs	458
Custom Form Deployment Process (Summary)	460
Custom Form Deployment Process (In Detail)	461
Deploying Upgraded Custom Forms	463
Previewing Undeployed Custom Forms	463
Editing the Layout of Custom Forms Prior to Deployment	464
Avoid Editing Custom Forms in Tandem	466
Testing Undeployed Custom Forms	466
Deploying Custom Forms	467
Deploying Skipped Custom Forms	467
Understanding Form Deployment Statuses	468
Understanding Form Layout Enhancement Upgrade Logic	469
Upgrade Logic for Custom Forms	469
Upgrade Logic for Subtabs	470
Upgrade Logic for Fields (Diagram)	470
Field Ordering	471

SuiteBuilder (Customization)

- SuiteBuilder Overview
- Customization Best Practices
- Custom Fields
- Custom Forms
- Advanced PDF/HTML Templates
- SuiteBuilder Advanced Templates Reference
- Basic Printing Layouts
- Custom Records
- Record Customization (Beta)
- 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.



SuiteBuilder Users

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.

Component	Description	More Information
Role	Set of permissions that can be assigned to a NetSuite user	Customizing and Creating Roles

Component	Description	More Information
Center	Configuration of NetSuite created for a specific group of roles with similar tasks.	Creating and Editing Custom Centers
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.	Creating Center Tabs

Set Up Data

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

Component	Description	More Information
Record	A single entry of information related to a single business concept. Use custom records to collect information specific to the needs of your business.	Custom Records
Segment	A single entry of information related to a single business concept. Use custom segments to create custom classification fields similar to class, department, and location.	Custom Segments
Transaction	A single entry of information related to a single business concept. Use custom transactions to create transaction types tailored to your business needs.	Custom Transactions
List	List of values that can be selected in a custom field. Use custom lists to set up predefined choices for your employees and customers to select when entering transactions and records.	Custom Lists
Template	A single entry of information related to a single business concept. Use advanced templates to customize printed and emailed forms, records, and saved searches.	Advanced PDF/HTML Templates

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.

Component	Description	More Information
Field	Place on a record or transaction where information is entered.	Custom Fields
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.	Creating Custom Subtabs
Sublist	The results of a saved search displayed on a custom or standard record. Sublists can also be generated through parent-child relationships.	Custom Sublists

Component	Description	More Information
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 perform periodically.	SuiteScript Overview

Customization Objects in SDF

Custom objects can be managed in SuiteCloud Development Framework (SDF). SDF uses XML definitions of custom objects to work with the object attributes, fields, fields structures, field properties, field values, and other details. For more information, see the help topic [SuiteCloud Development Framework XML Reference](#).

If the Multi-Languages feature is enabled in your account, you can define translated labels of customization objects, records, or fields to be used for various language settings. For more information, see the help topics [Translatability Overview](#) and [SuiteApp Translatability Support](#).

When working with custom objects that have translatable fields, you can use reference a translation string from a translation collection. For more information about translatable fields, see the help topics [Translatable Fields on SDF Custom Objects](#) and [Translatable Fields in SuiteCloud Development Framework](#).

For more information about developing custom objects in SDF, see the help topic [SDF Custom Object and File Development in SuiteCloud Projects](#).

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 business-related company data in NetSuite is entered through forms and NetSuite provides several standard forms that you can use. There are forms for every record you can store in your NetSuite account, including transactions, entities (for example, Customers and Vendors), Items, and CRM activities. For a complete list of supported record types, see the help topic [NetSuite Record Types](#).

With customization options, you can customize standard forms to ensure that you are capturing the information that your business needs.

Primary Information

CUSTOM FORM *	Standard Project Form	PROJECT NAME	Project 100	STATUS
PROJECT ID *	9	CUSTOMER	ABC Medical Supplies	LANGUAGE
NAME		Project 100		

Project Overview

CALCULATED WORK 200:00	ACTUAL WORK 16:00	PERCENT WORK COMPLETE 8.0%
---------------------------	----------------------	-------------------------------

Schedule **Resources** **Financial** **P&L** **Relationships** **Communication** **Related Records** **Preferences** **System Information** **Custom**

Project Tasks / Milestones

VIEW: Planning

New Project Task **New Milestone** **View Gantt Chart** **Customize View**

EDIT	ID	MILESTONE	NAME	EXPAND ALL	COLLAPSE ALL	PREDECESSORS	START DATE	END DATE	PLANNED WORK	CALCULATED WORK
Edit	1	No	Task 1				9/11/2013	10/15/2013	200	200

Actions: Save, Cancel, Reset, Search

Forms are composed of the following components:

- 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](#).
- 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. Note that the field area displayed at the top of a form by default is considered to be on the Main subtab. For more information, see the help topic [Creating Custom Subtabs](#)
- Sublists** - You can also add custom sublists to your forms. Sublists can display either saved search results relating to the record on which they are shown, or can display child record instances related to parent record using the relationship field. For example, you can add a custom sublist to track specific milestone details. For more information, see [Custom Sublists](#).

For more information, see the help topic [Custom Forms](#).

Use Case

On the following custom sales order form, you can see that a custom field has been added to show the approval status for the sales order.

Sales Order SORD10003 James Page BILLED

Edit Back | Authorize Return Manage Revenue Recognition | Actions ▾

APPROVAL STATUS
Approved

Primary Information

ORDER # SORD10003	PO #	Summary
CUSTOMER James Page	MEMO	TOTAL (ALT. SALES) 0.00
PROJECT		SUBTOTAL 700.00
DATE 10.3.2003		DISCOUNT ITEM
START DATE		TAX
END DATE		SHIPPING COST
		HANDLING COST
		GIFT CERTIFICATE
		<hr/>
		TOTAL 700.00

Sales Information

SALES REP Jon Baker	SALES EFFECTIVE DATE 10.3.2003	LEAD SOURCE 561 Spring Catalog
OPPORTUNITY	<input type="checkbox"/> EXCLUDE COMMISSIONS	PARTNER

Most forms are divided into subtabs. Subtabs group fields with similar information in one place. For example, the Address subtab is where shipping and billing addresses are entered. You can create subtabs for custom fields.

On the following sales order form, a custom Warranty subtab has been added to track warranty information through the **Warranty Number** and **Warranty Term** custom fields.

Primary Information

ORDER # SORD10003	PO #
CUSTOMER James Page	MEMO
PROJECT	
DATE 10.3.2003	
START DATE	
END DATE	

Sales Information

SALES REP Jon Baker	SALES EFFECTIVE DATE 10.3.2003	LEAD SOURCE 561 Spring Catalog
OPPORTUNITY	<input type="checkbox"/> EXCLUDE COMMISSIONS	PARTNER

Classification

DEPARTMENT Sales	CLASS	LOCATION
DEFERRED REVENUE RECLASSIFICATION ACCOUNT	READY FOR SHIPMENT	LAST MODIFIED DATE 7.10.2014
FOREIGN CURRENCY ADJUSTMENT REVENUE ACCOUNT	SALES REP PHONE	CREATED DATE 10.3.2003
CUSTOMER CATEGORY	POINTS	

Items Shipping Billing Accounting Relationships Sales Team Communication Related Records System Information Address Custom **Warranty EFT**

WARRANTY NUMBER WRNT-EC12409020	WARRANTY TERM 1 year - parts & labor
------------------------------------	---

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 can add a custom sublist to a custom vendor form that shows all of the items that most often purchased from the vendor.

First, you create an item saved search. On the Results subtab of the search, you add the columns you want to show on the sublist. On the Available Filters subtab, you select **Vendor: Name/ID**. The filter is used by the sublist to show only the items purchased from a vendor. For more information about saved search filters, see the help topic [Selecting Available Filters for Saved Searches](#).

Currencies • Pricing Schedules • ACH Transactions • Items • Rates Preferred Items •					
EDIT	NAME	DISPLAY NAME	DESCRIPTION	TYPE	BASE PRICE
Edit	Bio Therapy : Hirudo Medicinalis	Hirudo Medicinalis	Hirudo Medicinalis for Bio Surgery	Inventory Item	140.00
Edit	Bio Therapy : Phaenicia Sericata Larvae	Phaenicia Sericata Larvae	Phaenicia Sericata Larvae for Debridement Therapy	Inventory Item	70.00

You can also create a sublist that displays child record instances using the Parent is Child relationship field. For example, you can create a custom field and assign the parent record and child record so that the parent record pulls in the child record information through the custom field. The following example shows a list of customers associated with case records. In this example, the parent record is Case, the child record is Customer, and the parent subtab where the sublist appears is Related Records.

First, you create an entity custom field called Customer Records. For Type, select List/Record. For List/Record, select Case. Check the Record is Parent box. On the Applies to subtab, check Customer. On the Display subtab, for Parent Subtab, select Related Records.

Now, when you view a Case record, a Customers sublist appears with a Customer field that displays a list of customers associated with the case.

The screenshot shows a NetSuite Case record form. At the top, there are various tabs like Activities, Transactions, Lists, Reports, Analytics, etc. Below the tabs, there are several input fields: SUBJECT (95, AUTO), CONTACT (Bestill, Ella L), COMPANY (Bestill, Ella L), and EMAIL (elbestill@netsuite.com). Under 'Incident Information', there are fields for INCIDENT DATE (10.5.2019), INCIDENT TIME (8:09 am), PRODUCT (dropdown), MODULE (dropdown), ORIGIN (Email), and INBOUND EMAIL ADDRESS (cases.563219.1.5307ae499@cases.netsuite.com). There is also a checkbox for HELP DESK. In the bottom right corner of this section, it says 'CURRENT ASSIGNED TO (2)' and 'INBOUND MEMO (2)'. Below this, there is a 'Communication' tab and a 'Related Records' tab. The 'Related Records' tab is active and shows a list of 'Customers'. The first item in the list is 'Customer Default', with a dropdown menu showing 'CUSTOMER' and a placeholder 'Type then tab'. At the bottom of the form, there are buttons for Save, Cancel, Merge, Enable Spam Lock, and Actions.

Record Types

You can create custom record types to collect and store information that is not included in NetSuite. For example, your company may need to track information about computer and electronic equipment. If

there is no such standard record type in NetSuite, you can create a custom record type called Equipment. Then you can add custom fields to collect and store the equipment information needed. For more information about the platform tools supported by custom record types, see .

The screenshot shows the 'Custom Record Type' setup page for 'Equipment'. It includes fields for Name (Equipment), ID (customrecord26), Owner (Wolfe, J Wolfe, J), and Description. There are sections for Access Type (Use Permission List) and various permissions like Allow UI Access, Allow Mobile Access, etc. A 'Forms' tab is visible at the bottom.

Custom records can be attached to standard records and other custom records making them child record types. 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 record instances and set up forms for various roles. Customizing a form involves determining which fields appear, how these fields are arranged, and which roles use the form.

Equipment Service Example using Custom Record Types

This section provides a high-level overview of the process for creating a custom record type. In this scenario, we'll assume your company needs to track information about computer and electronics equipment.

Complete the following steps:

1. [Creating the Equipment Custom Record Type](#)
2. [Creating the Parent-Child Relationship](#)
3. [Creating a New Service Record Instance](#)
4. [Customizing the Form Used to Enter Custom Records](#)

Creating the Equipment Custom Record Type

Your company needs to track information about computer and electronic equipment. Because there is no such standard record type in NetSuite, you create a custom record type called Equipment. Then, on the

Equipment record, you add custom fields in which you enter serial numbers, location, purchase date, and service and warranty information.

Equipment

Evernon 3000 Laptop 3

Edit | **Back** | Actions ▾

NAME	SERIAL NUMBER
Evernon 3000 Laptop 3	287892898-898982
OWNER	LOCATION
Wolfe, J	
<input type="checkbox"/> INACTIVE	PURCHASE DATE
	8.10.2008
MANUFACTURER	VENDOR
Evernon	Ingram Micro
MODEL	
3000 Laptop	

Notes Files Workflow Warranty Service

WARRANTY NUMBER	WARRANTY VENDOR
39893893	Ingram Micro
WARRANTY TERM	VENDOR PHONE
3 years; parts and labor	650-589-8000

Your custom records can be attached to standard records and other custom records. For example, on the Equipment record type, you want to track details each time you service your equipment, so you create a new custom record called Equipment Service.

Creating the Parent-Child Relationship

To create the parent-child relationship, you add a custom field to the Equipment Service record in which you can select the associated Equipment record. You select a field type of List/Record and select Equipment in the List/Record field. To create the parent-child relationship, check the Record is Parent box.

Equipment Service Field

Save | **Cancel** | **Reset** | **Change ID** | **Apply to Forms** | **Actions ▾**

LABEL *	DESCRIPTION	<input type="checkbox"/> SHOW IN LIST <input type="checkbox"/> GLOBAL SEARCH <input checked="" type="checkbox"/> RECORD IS PARENT <input type="checkbox"/> APPLY ROLE RESTRICTIONS
Equipment		
ID		
custrecord33		
INTERNAL ID	TYPE	
101	List/Record	
OWNER	LIST/RECORD	
Wolfe, J	Equipment	
<input checked="" type="checkbox"/> STORE VALUE <input type="checkbox"/> USE ENCRYPTED FORMAT		

Display **Validation & Defaulting** **Sourcing & Filtering** **Access** **Translation** **History**

INSERT BEFORE	HELP
- Unchanged -	
SUBTAB	
DISPLAY TYPE	PARENT SUBTAB
Normal	Service
<input type="checkbox"/> ALLOW QUICK ADD	

Creating a New Service Record Instance

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

In our example, you click New Equipment Service and enter the service information in the new record. The equipment field fills in automatically. The child service record shows on the parent Equipment record.

The screenshot shows the Equipment record creation screen. At the top, there are buttons for Save, Cancel, Reset, and Actions. Below that, a dropdown for 'CUSTOM FORM' is set to 'Standard Equipment Form'. The main form fields include:

- NAME ***: Evernon 3000 Laptop 3
- OWNER**: Wolfe, J
- INACTIVE**: (unchecked)
- MANUFACTURER**: Evernon
- MODEL**: 3000 Laptop
- SERIAL NUMBER**: 287892898-898982
- LOCATION**: (dropdown menu)
- PURCHASE DATE**: 8.10.2008
- VENDOR**: Ingram Micro

The 'Service' tab is selected at the bottom of the screen. Under the 'Equipment Service' subtab, there is a table listing several service records. One row in the table is highlighted with a red border, corresponding to the 'New Equipment Service' button which is also highlighted with a red box.

VIEW	EQUIPMENT SERVICE	
Default View	Install New Sound Card	Remove
Edit	Routine Maintenance	Remove
Edit	Routine maintenance	Remove
Edit	Routine maintenance	Remove
Edit	Upgrade Memory	Remove

The equipment record in this example is most useful to the IT staff who purchase, maintain, and track the equipment. But there is some information that is generally useful to others in the company.

Customizing the Form Used to Enter Custom Records

When you create a new custom record, a custom form for that custom record is automatically created.

You can customize the forms used to enter custom records in the same way that you customize standard transactions and standard records. You can set up forms for various roles. Customizing a form involves determining which fields show, how these fields are arranged, and which roles use the form.

To customize the forms

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 boxes in the Show column next to the Warranty and Service subtabs. Then click **Save**.
3. In the **Access Type** field, ensure that **Use Permission List** is selected.
4. On the **Permissions** subtab, select the roles that should use the new custom equipment form only. In this example, we add all non-IT roles. 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 relevant to only 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 nonoperational 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.

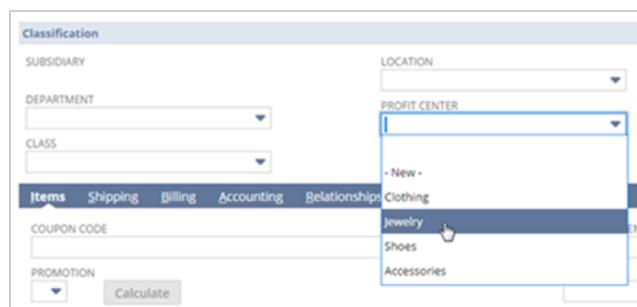
For more information about custom transactions, see [Custom Transactions](#).

For more information about standard transaction types, see the help topic [Transaction Types Included in Monthly Transaction Lines Metric](#).

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 may 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 be customized with a custom template. You can use the Template Editor to customize the existing standard templates.

Custom Transaction Form

Save Cancel Reset | Save & Move Elements

NAME *

TYPE

PRINTING TYPE ADVANCED BASIC

PRINT TEMPLATE

EMAIL TEMPLATE

DISCLAIMER

ADDRESS

LOGO

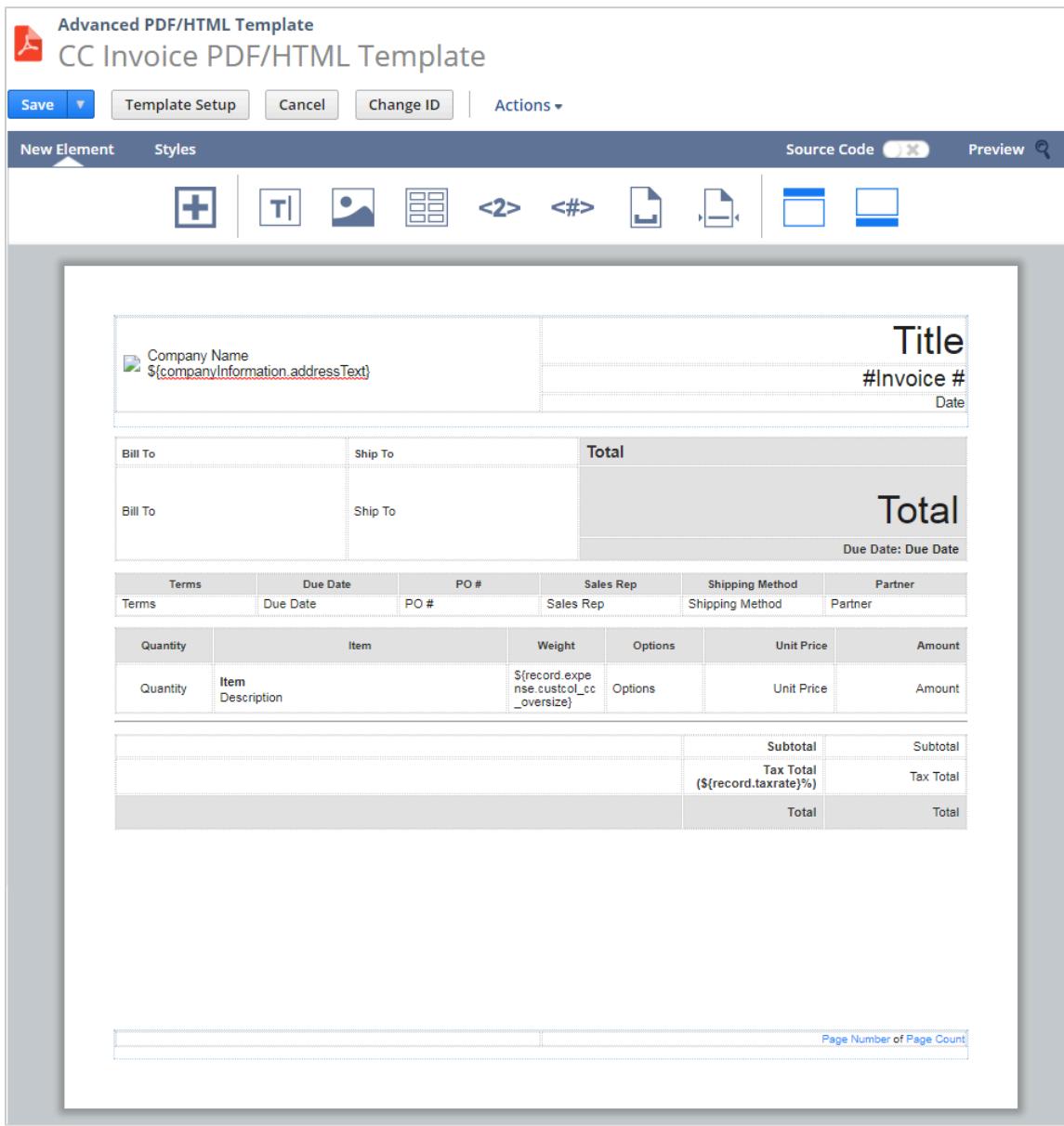
COLUMNS WIDTH 12.75 LAYOUT SPACE 7.5

ALLOW ADD MULTIPLE
 INACTIVE
 FORM IS PREFERRED

Tabs Field Groups Screen Fields Actions Lists Custom Code Roles Linked Forms

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 the New Element toolbar to add HTML-based elements such as fields, images and tables, or printing elements.



For more information about using the template editor, see [WYSIWYG Editing in the Template Editor](#).

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.

The screenshot shows a custom NetSuite center interface titled "Leads". The top navigation bar includes links for Activities, Leads, Opportunities, Customers, Forecast, Reports, Documents, Setup, SuiteSocial, Support, and Knowledge Base. On the right, there are "Personalize" and "Layout" dropdown menus.

The main content area is divided into several sections:

- Leads Links:** Contains a "Find Link..." search bar and links for Leads, Contacts, Lead Source Analysis (with "Detail" and "Gross Lead Source Analysis" options), and a "Quick Search" section.
- Quick Add: New Lead:** A form for adding a new lead with fields for CUSTOMER ID (set to "To Be Generated"), EMAIL, COMPANY NAME, PHONE, and WEB ADDRESS, along with a "Save" button.
- Shortcuts:** A grid of links categorized by role:

Tasks	Enter Sales Order	Customers & Projects
Calendar	Prepare Quote	Leads
Contacts	Enter Cash Sale	Prospects

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 grant custom center access only 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](#).

Customization Best Practices

This section contains best practices for you to consider when building customization. See the following topics in this section:

- [Optimize Your Custom Fields and Custom Forms](#)
- [Plan Ahead and Make Use of Existing Standard Features](#)
- [Set Up Internal Controls](#)
- [Know When to Use SuiteFlow and SuiteScript](#)
- [Conventions for Naming Custom Objects](#)
- [Changing the ID of a Custom Object](#)

Optimize Your Custom Fields and Custom Forms

Remove unnecessary fields when customizing forms. For more information, see [Configuring Fields or Screens](#).

Document and use appropriate naming conventions at all times so that any user can identify the correct object to use and maintain consistency across your organization. IDs of custom objects (except for custom segments) can be changed after they have been created using the **Change ID** button at the top of the page. For more information, see [Conventions for Naming Custom Objects](#) and [Changing the ID of a Custom Object](#).

Check the **Form is Preferred** box when creating a new custom form. Doing this lets you decide which forms your employees use and helps maintain consistency across your company. For more information, see the help topic [Defining Preferred Forms](#).

Choose the correct field type when creating Custom Fields and be aware of the differences between them. For example, a free-form text field type can have up to 300 text characters, whereas a text area field can have up to 4000 characters. For more information about various field types, see the help topic [Field Type Descriptions for Custom Fields](#).

If you create a nonstored custom field that uses formulas or searches, it causes the database to slow down because many logs are created. To avoid affecting performance, check the Store Value box and use a server-side script. For more information, see the help topic [Record.setValue\(options\)](#).

If you work with multiple records and you want to set up ongoing updates, consider using CSV import, mass update, scheduled script or map/reduce script features during off peak hours. For more information, see the help topics [CSV Imports](#), [Mass Update Scripts](#), [SuiteScript 2.x Scheduled Script Type](#) and [SuiteScript 2.x Map/Reduce Script Type](#).

Plan Ahead and Make Use of Existing Standard Features

It's important to understand the scope of a customization before deciding on the appropriate technology to accomplish the business goal. Before building your customization:

- Consider whether the new customization solution is required, nice to have, or can be implemented in a future release.

- Write a roll-out plan and conduct an impact analysis before implementing your solution. Make sure you understand what existing business processes, scripts, workflows and other customizations will be affected.
- Write flexible solutions that consider performance and scalability.
- After your plan is ready, prototype in a sandbox account to avoid potential problems with your production account.
- Think about existing standard features you can use instead of building a customization. For example, you can use Mail Merge to send and attach an email message in a custom record rather than building a new customization. Or you can set the default tax code for line items using existing standard features instead of scripting a solution. To set the default tax code, use the Default Tax Code field in Setup > Accounting > Set Up Taxes. For more information, see the help topics [Working with Mail Merge](#) and [Set Up Taxes](#).

Consider using Suite Script for more complex customizations. For more information, see the help topic [What You Can Do with the SuiteScript API](#).

Set Up Internal Controls

Internal controls help to protect your company from risk and avoid serious errors. NetSuite provides many readily-available controls that do not require customizations. Any customizations made for internal controls are tracked in the System Notes.

Consider the following when setting up your internal controls:

- Ensure logical access and application security. Users should have access only to the information they need to do their jobs.
- Segregate duties and transaction processing.
- Audit permissions regularly for each role.
- Limit the number of users with the Administrator role.

For more information, see the help topics [Internal Controls in NetSuite](#) and [Auditing Data Changes using Searches](#).

Know When to Use SuiteFlow and SuiteScript

SuiteScript and SuiteFlow are powerful tools that have overlapping features and capabilities.

SuiteFlow is designed for automating business processes and does not require any prior programming knowledge. The best practice with SuiteFlow is to know what you want to achieve with your workflow prior to developing it. Modifying it on the go can be difficult if multiple states and actions are involved. For more information, see the help topics [SuiteFlow Overview](#) and [SuiteFlow Best Practices](#).

SuiteScript is a JavaScript-based API used to extend the capabilities of NetSuite. You need to have programming knowledge to use SuiteScript. With SuiteScript you can create a custom user interface, programmatically run saved searches, create custom dashboard portlets, run batch processes, and more. For more information, see the help topics [What You Can Do with the SuiteScript API](#) and [SuiteScript Best Practices](#).

The following actions can be done in both SuiteFlow and SuiteScript:

- Show messages

- Validate fields
- Set up default values
- Change display characteristics
- Set a field as required or optional
- Send email messages
- Create transactional and nontransactional records
- Redirect end user
- Approval routing
- Manipulate sublist

In general, anything that can be done in SuiteFlow can also be done in SuiteScript. When deciding between the two, consider the technical knowledge you have. If your company does not have any JavaScript developers, use SuiteFlow.

SuiteFlow vs. SuiteScript

SuiteFlow helps automate business processes. Consider performing the following actions with SuiteFlow:

- Basic field defaulting. Creating basic field defaulting in SuiteFlow helps users with the Administrator role make edits in the future.
- Complex approval processing
- Approval reminder email messages
- Lead nurturing (or drip marketing)
- Automation based on changes to related records
- Using existing saved searches to process the search data

The following functions are available only in SuiteScript:

- Create custom server requests from the browser
- Redirect user to a Suitelet
- Update related records.
- Copy, print, delete, and attach records
- Create dynamic searches, add filters or results to the search
- Perform web service calls
- Create scripted portlets
- Build new applications

Conventions for Naming Custom Objects

When you create a custom object, such as a custom field, custom record, custom list, and so on, you can specify a unique alphanumeric identifier. You should establish naming conventions for your customizations so that users can identify the correct object to use. Before you enter an ID, consider the purpose of the custom object and the location where it will be used. Create IDs for custom objects that will have meaning to your users. If you leave the ID or Script ID field empty, the system automatically creates an ID when you save the custom object.

If your company name is First Software and your team decides to use the prefix “_fs_” with all custom objects, your list of custom objects would resemble the following:

- custentity_fs_travel_approval_limit
- custitem_fs_gift_wrap_price
- custrecord_fs_system_role



Important: For all custom objects except for custom segments and integration records, you can change an existing ID value using the Change ID button at the top of the page. However, changing the value of an ID that is being used in a script can cause the script to break with no warning messages. You can avoid the need to change IDs in the future by planning your naming conventions before creating the IDs.

When you create an ID for a custom object, the system automatically prepends the file name with a standard string of text. This string varies by custom object type and cannot be changed. As a best practice, you should enter a unique ID name that begins with an underscore. Including the underscore provides a separation between the prepended string of text and your ID.

When planning your naming conventions for custom object IDs, consider including a prefix. For instance, you may want to identify all custom fields by the department to which the custom field applies. We'll assume your company has the following departments: Marketing, Customer Support, Service, and Sales. When creating the unique IDs for each department, you should add prefixes such as **_mktg** for Marketing, **_cs** for Customer Support, **_serv** for Service, and **_sales** for Sales. For example, you want to create a new Keyboard Style item field to be used by the Sales department. The automatic prepended string of text for custom item fields is **custitem**, and you enter an ID of **_sales_keyboard_style**. If we follow the naming conventions mentioned previously, the final ID for the custom field would be **custitem_sales_keyboard_style**.

The following table lists the prepended string used for each custom object type ID or Script ID. It also shows the character limit for the ID or Script ID field.

Custom Object	Prepending String	Character Limit
List	customlist	28
Record Type	customrecord	28
Custom Segment	cseg	15
Transaction Type	<ul style="list-style-type: none"> ■ Purchase: custompurchase ■ Sales: customsale ■ Basic, Journal, and Header Only: customtransaction 	23
Field	<ul style="list-style-type: none"> ■ CRM: custevent ■ Entities: custentity ■ Items: custitem ■ Record: custrecord 	30
Transaction Body Field	custbody	30
Transaction Line Field	custcol	30
Transaction Item Options	custcol	30
Item Number Field	custitemnumber	30
Other Record Fields	custrecord	30

Custom Object	Prepending String	Character Limit
Other Sublist Fields	custrecord	30
Sublist	custsublist	30
Subtab	custtab	40
Forms	<ul style="list-style-type: none"> ■ Entry Forms: custform ■ Transaction Forms: custform ■ Address Forms: custform 	91

Changing the ID of a Custom Object

For all custom objects except for custom segments, if necessary, you can change the ID after the custom object has been created. Use this feature with caution because changing the value of an ID that is being used in a script can cause the script to break with no warning messages.

For information about best practices and naming conventions, see [Conventions for Naming Custom Objects](#).

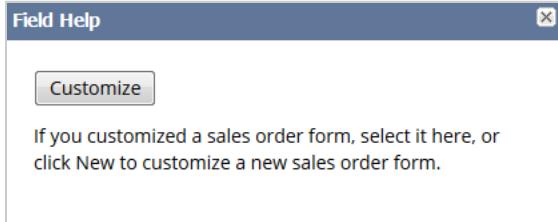
To change the ID or Script ID of a custom object:

1. From the custom object page, click **Change ID**.
2. In the **New ID** field, enter the new ID.
3. Click **Save**.

Customizing Field Level Help for Standard Fields

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.

FORM TYPE	FIELD LABEL	FIELD ID
Employee	Employee ID	entityid

HELP TEXT
Enter the name as it will appear on checks, tax reports and forms.

Translations

LANGUAGE	HELP TEXT
Spanish	Ingrese el nombre tal como aparecerá en los cheques, informes de impuestos y

MAKE AS DEFAULT FOR OTHER FORMS OF THE SAME TYPE

Save **Remove Customization** **Cancel**

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>
-
-
-
- <i>
- <u>
-
-

- <a>

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

- href – Required attribute. Only absolute paths are allowed, that is, URLs starting with http or https. In addition, JavaScript is not supported.
- target – Required 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 popup window.
- title
- download
- hreflang
- rel
- media

For more information about the <a> tag, see the [HTML <a> Tag page](#) on the w3schools website.

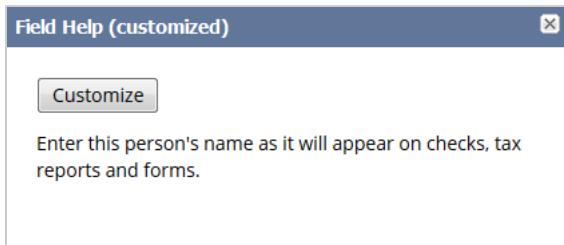
Note: Because HTML tags are supported, use < and > to display the < less than and > greater than symbols in the custom field-level help.

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.

The screenshot shows the 'EMPLOYEEID' field selected in the 'Standard Employee Form' template. The field is highlighted with a red box. To the left of the field name, there is a small blue circular icon with a white letter 'i', which is the standard icon for a tooltip or custom help in SuiteBuilder.

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.

Standard Field Level Help Customization								
	ACTIONS	FORM TYPE	FORM TITLE	FORM NAME	FIELD LABEL	FIELD ID	DEFAULT	HELP TEXT
<input type="checkbox"/>	Edit Remove	Customer/Lead /Prospect	Prospect	Customer_PREFERRED=T_Store=F	Status	entitystatus	<input checked="" type="checkbox"/>	Select the appropriate status of this customer or potential customer. A lead status creates a lead record. A prospect status creates a prospect record.
<input checked="" type="checkbox"/>	Edit Remove	Employee	Employee	Standard Employee Form	Employee ID	entityid	<input checked="" type="checkbox"/>	Enter the name as it will appear on checks, tax reports and forms.
<input type="checkbox"/>	Edit Remove	Vendor	Vendor	Standard Vendor Form	Category	category	<input checked="" type="checkbox"/>	Select the category that applies to this vendor. To add choices to this list, contact admin - do not add your own categories..

Remove Customization of Selected

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. Custom fields can be created for records or for transactions. 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. Custom fields are based on available standard fields and field types. For more information about custom field types, see the following:

- [Field Type Descriptions for Custom Fields](#)
- [Custom Field Types](#)
- [Available Standard Fields and Field Types](#)



Important: SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should access the NetSuite UI only by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic [SuiteScript 2.x Custom Pages](#).

The following table lists required steps you must follow to create a custom field:

Required Steps	Related Help Topics
Set the basic properties for the field	Creating a Custom Field
Assign the field to forms, as needed	Assigning Custom Fields to Specific Record Types
Set the display properties for the field	Setting Display Options for Custom Fields
Set the required validation and defaulting properties and, if needed, create dynamic defaults and hyperlinks	<ul style="list-style-type: none"> ■ Setting Validation and Defaulting Properties ■ Dynamic Defaults and Dynamic Hyperlinks
Set any sourcing criteria needed for the field	Setting Sourcing Criteria
Set any access restrictions to the field based on department, role, or subsidiary	<ul style="list-style-type: none"> ■ Restricting Access to Custom Fields ■ Restricting Access to Employee Custom Fields
Set any filtering criteria needed for the field	Setting Filtering Criteria

The following table lists optional tasks you can perform when creating a custom field:

Optional Tasks	Related Help Topics
Filter choices in a dropdown list field	Dependent Dropdown Lists
Make a field read-only	Creating Read-Only Custom Fields
Add field-level help for a custom field	Adding Field-Level Help for Custom Fields
Translate the label and field help for the field	Adding Translations for Custom Fields
Add a custom transaction field to a custom transaction form	Adding Custom Fields to Transaction Forms
Add a formula to a field	Creating Formula Fields

Optional Tasks	Related Help Topics
Create a custom list	Custom Lists
Convert the field type of a custom field	Converting the Field Type of a Custom Field
Track changes made to a custom field	Tracking Changes to Custom Fields
Inactivate a custom field	Inactivating a Custom Field
Create custom fields with values derived from summary search results	Creating Custom Fields with Values Derived from Summary Search Results
Maintain saved searches that include edited custom fields	Maintaining Saved Searches that Include Edited Custom Fields

For step-by-step instructions on creating and editing custom fields, see the following:

- [Creating a Custom Field](#)
- [Inactivating a Custom Field](#)
- [Editing a Custom Field](#)
- [Advanced Features for Custom 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 deactivate 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. For information about deleting a custom field, see the help topic [Buttons and Menus in NetSuite](#).

Custom fields are supported in SuiteCloud Development Framework (SDF). SDF is a development framework that you can use to create SDF SuiteApps, or to customize NetSuite accounts, using an integrated development environment (IDE) on your local computer. SuiteCloud projects are file-based and use XML definitions of custom NetSuite objects. For more information, see the help topic [SDF Custom Object and File Development in SuiteCloud Projects](#).

For a general overview of custom fields setup options, watch the following video.



To view a list of custom fields training videos, see [Custom Fields Videos](#).

Custom Fields Videos

To view help videos for creating custom fields, see the following:

- [Creating Custom Fields – Overview](#)
- [Creating and Adding a Custom Field to a Record Type](#)
- [Adding a Default Selection to a Custom Field](#)
- [Using Sourcing and the Store Value Feature with a Custom Field](#)
- [Access Control for Custom Field Values](#)
- [Adding a Custom Field that Spans the Top or Bottom of a Page](#)
- [Adding Field-Level Help for Custom Fields](#)

More videos will be added as they become available.

Field Type Descriptions for Custom Fields

When creating a custom field, you select the type of field you want to create depending on the information you want to store in the field.

The following table describes all available field types for custom fields. 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.

Field Type	Description	Example
Check box	Record a true or false answer by placing a check mark in the custom field.	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.
Currency	<p>Enter currency amounts in a custom field (maximum 15 digits and 15 decimal points – 999,999,999,999,999.99999999999999)</p> <p>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.</p>	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.
Date	<p>Enter or select dates.</p> <p>Note that changing Date fields to Date/Time fields will automatically convert Dates in existing records to Date/Times. The time value defaults to midnight of your company's time zone. All existing SuiteScript that referenced the Date field must manually updated.</p>	<p>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 > Company > General Preferences.</p> <div style="border: 1px solid #ccc; padding: 10px; background-color: #f9f9f9;"> i Note: The date and time are based on the user's NetSuite Time Zone preference, set at Home > Set Preferences, not on the browser client time zone. </div>
Date/Time	<p>Combine date and time values in one field.</p> <p>After you enter 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.</p> <p>Also note that time values are stored in NetSuite down to the second.</p> <p>Optionally, you can also add Date/Time custom fields in SuiteScript using the <code>form.addField(options)</code> method. Important: If you use the <code>addField</code> method, for the <code>type</code> parameter, you must specify '<code>datetimetetz</code>'.</p>	<p>For example, you may want a single field to contain date and time timestamp data.</p> <div style="border: 1px solid #ccc; padding: 10px; background-color: #f9f9f9;"> i Note: The date and time are determined by your NetSuite Time Zone preference, set at Home > Set Preferences, not by the browser client time zone. </div>
Decimal Number	Enter a decimal number in a custom field. The largest number you can input is 10,000,000,000,000,000 (maximum 20 digits).	For example, you want to record the distance your customers are from your nearest retail outlet. You create a decimal number custom

Field Type	Description	Example
Number	If you enter a number larger than this, it will be automatically rounded down to the maximum number permitted.	field to store the information on your employee records.
Document	<p>Select a File Cabinet document to preview or download. The field is searchable and can be added to search results.</p> <p>Note: The user of the document custom field must have File Cabinet access to view, select, or upload documents.</p>	For example, you want employees to be able to search customer survey results.
Entity	<p>You can create custom fields of type Entity. To create a custom entity field, set the field type to List/Record, and set the List/Record type to Entity. By doing so the new generic custom entity field will automatically include all records of type contact, customer, employee, group, other name, partner, and vendor.</p> <p>You can then optionally use sourcing and filtering capabilities to limit the choices in the custom field's list to a subset of the supported entities.</p>	<p>For example, on the custom field's definition page, on the Sourcing & Filtering subtab, specify Type in the Filter Using column. In the Value Is column, specify customer, partner, and vendor names.</p> <p>When the custom field of type Entity appears on your record, entities of only type customer, partner, and vendor will appear as available options in the field's list.</p>
Email Address	Enter an email address in a custom field (maximum 254 characters). 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.
Free-Form Text	<p>Enter up to 300 characters of text in a custom field. URLs in custom fields should use your account-specific domain. For more information, see Using Account-Specific Domains.</p>	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.
Help	<p>Place helpful text on record pages where your users enter information (maximum 999 characters). The help that appears is for informational purposes only. It is not stored in your account.</p> <p>URLs in custom fields should use your account-specific domain. For more information, see Using Account-Specific Domains.</p>	<p>When you create a Help custom field, select Help in the Type field and enter your text in the Help field.</p>
Hyperlink	<p>Enter a URL in a custom field that links to another web page (maximum 999 characters, with a hyperlink text limit of 10 characters).</p> <p>Hyperlink fields should begin with http://, https:// or ftp://.</p>	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 URL to open the vendor's site in a new window. You can enter up to 999 characters in the field.



Note: Calculations within free-form text fields display values in scientific notation. To avoid scientific notation, use the following syntax to round digits in free-form text fields: ROUND({cost}/0.65,3).

Field Type	Description	Example
Image	<p>Note: Due to safety concerns, files beginning with file:// or \\ are not currently supported. It is possible to enter a file in this format, but the link will not work.</p> <p>URLs in custom fields should use your account-specific domain. For more information, see Using Account-Specific Domains.</p> <p>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.</p> <p>Image custom fields support the following file formats:</p> <ul style="list-style-type: none"> ▪ .bmp ▪ .gif ▪ .jpg ▪ .jpeg ▪ .pjpg ▪ .pjpeg ▪ .png ▪ .tiff ▪ .tif <p>Note: The TIF/TIFF file format may not be supported by all browsers.</p>	For example, you can attach pictures of your employees to their records.
Inline HTML	<p>Inline HTML can be used on all types of fields. Use HTML, for example, to define a custom field on a record form, or to define a custom field to be included on custom pages generated by Suitelets. For more information about using HTML with Suitelets, see the help topic SuiteScript 2.x Suitelet Script Type.</p> <p>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 fewer.</p> <p>Important: 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 with a new user interface in a future NetSuite release.</p> <p>URLs in custom fields should use your account-specific domain. For more information, see Using Account-Specific Domains.</p>	For example, you could display to customers how many reward points they have earned.

Field Type	Description	Example
Integer Number	Enter integers in a custom field. The maximum number is approximately 18000000000000000000 (20 digits). This maximum can be affected by automatic rounding and limitations of underlying technologies.	For example, you can record a special ID number on your company records.
List/Record	Attach lists and records to custom fields. In the List/Record field, select the list or record type you want to attach. You can also select from a list of public saved searches.	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.
Long Text	<p>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.</p> <p>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.</p>	<p>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.</p>
Multiple Select	<p>Lets you create a field where you can make multiple selections from a list or list of records.</p>	<p>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. Sales representatives use the field to select the various places customers heard about your business.</p>
Password	<p>Lets you create a field that is encrypted in the database (maximum 15 characters). 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 real encrypted value. Ensure that customizations do not improperly use sensitive fields.</p>	<p>For example, you could add a password custom field to a web page.</p>
Percent	<p>Lets you create a field to store a percentage. The information entered in the field can be only an integer from 0 to 100.</p>	<p>For example, you want to track the test results of a training course taken by an employee, and have the result display as a percentage. The percent sign (%) is automatically added to the number.</p>

Field Type	Description	Example
Phone Number	<p>Lets you create a field for a telephone number on records and transactions (maximum 32 characters).</p> <p>If the phone number formatting feature is enabled in your account, phone numbers are localized. The Country field in the subsidiary of the user determines the localization applied.</p>	For example, you can include a contact number on an event record using a CRM field.
Rich Text	<p>Lets you enter and format up to 100,000 characters of text in a custom field.</p> <p>URLs in custom fields should use your account-specific domain. For more information, see Using Account-Specific Domains.</p>	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.
Text Area	<p>Lets you enter up to 4,000 characters of text in a custom field.</p> <p>URLs in custom fields should use your account-specific domain. For more information, see Using Account-Specific Domains.</p>	For example, you want to record special information about how you closed a customer. You could enter it into a text area custom field that appears on customer records.
Time of Day	Lets you enter the time of day in a custom field.	For example, you create a Best Time To Call custom field for your sales representatives to enter the best time of day to call a prospect.

For information about creating a custom field, see [Creating a Custom Field](#).

Using Account-Specific Domains

URLs in custom fields should use your account-specific domain. For more information about account-specific domains in specific field types, see the following:

- Free Form Text

If you include a URL, ensure that it uses your account-specific domain. Data center-specific URLs are automatically modified to an account-specific domain URL, but the 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](#).

- Inline HTML

Use URLs that include your account-specific domain. If the default URL value for this field contains a data center-specific domain, for an active session, it is automatically modified to the account-specific domain. However, the configuration remains unchanged. You should update all links in custom fields to use account-specific domains. For more information, see [Account-Specific Domains in Custom Fields](#).

- Help, Hyperlink, Long Text, Rich Text, and Text Area

Use URLs that include your account-specific domain fields. If you enter a URL that includes a data center-specific domain, it is automatically modified to the account-specific domain. However, the configuration remains unchanged. You should update all links in custom fields to use account-specific domains. For more information, see [Account-Specific Domains in Custom Fields](#).

Custom Field Types

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

Type	Description	Instructions for Creating Field Type
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.	Creating Custom CRM Fields
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.	Creating Custom Entity Fields .
Custom Item Fields	Used to add fields to item records. These records include Accounting and website item records — such as inventory, noninventory, service, other charge, group, kit/package, and assembly/bill of materials item records.	Creating Custom Item Fields
Custom Transaction Body Fields	Used to add fields to the body of transaction records. These records include, for example, purchase, sale, journal entry and expense report records — such as sales orders, invoices, purchase orders, opportunities, Web store transactions, or item receipts.	Creating Custom Transaction Body Fields .
Custom Transaction Line Fields	Used to add fields to the lines 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.	Creating Custom Transaction Line Fields
Custom Transaction Item Fields	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.	Creating Custom Transaction Item Options
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.	Creating Custom Item Number Fields
Other Custom Fields	Used to add fields to custom records not defined by the preceding categories, including campaign events, classes, competitors, departments, and locations.	Creating Other Record Fields
Other Sublist Fields	Used to add fields to the columns of a custom sublist. These fields display in the line-item columns of sublists.	Creating Other Sublist Fields

You can differentiate between custom fields and standard fields on NetSuite pages. To do so, 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 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.

Available Standard Fields and Field Types

Each custom field can be sourced from a standard field based on its field type. The following topics contain tables that list standard fields and the required field type available for custom fields.

- [Transaction Body Fields](#)

- Transaction Line Fields
- Transaction Item Options
- CRM Fields
- Entity Fields
- Item Fields

Transaction Body Fields

You can source from an Entity, a Ship To, or a Sales Rep field.

Source From Entity

The type of entity used to source from depends on the type of transactions you apply your field to. Values in a **purchase** transaction are sourced from **vendor** records. Values in a **sales** transaction are sourced from **customer** records, and values in an **expense report** are sourced from **employee** records.

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

Field	Type	Sales Transactions	Purchase Transactions	Expense Reports
Name	Free-Form Text	×	×	×
Bill To	Text Area	×	×	×
Ship To	Text Area	×	×	—
Phone	Phone Number	×	×	×
Fax	Phone Number	×	×	—
Email	Email Address	×	×	×
City	Free-Form Text	×	×	×
State	Free-Form Text	×	×	×
Zip	Free-Form Text	×	×	×
Country	Free-Form Text	×	×	×
Sales Rep	List/Record	×	—	—
Expected Close Date	Date	×	—	—
Renewal Date	Date	×	—	—
Contact	Free-Form Text	×	×	—
Alt. Contact	Free-Form Text	×	×	—
Alt. Phone	Phone Number	×	×	—
Balance	Currency	×	×	—
Credit Limit	Currency	×	×	—
Account	Free-Form Text	×	×	—

Field	Type	Sales Transactions	Purchase Transactions	Expense Reports
1099 Eligible	Check Box	—	×	—
Tax ID	Free-Form Text	—	×	—
Legal Name	Free-Form Text	—	×	—
Supervisor	List/Record	—	—	×
Soc. Sec. #	Free-Form Text	—	—	×

Source From Ship To

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

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

Field	Type
Name	Free-Form Text
Bill To	Text Area
Ship To	Text Area
Phone	Phone Number
Fax	Phone Number
Email	Email Address
City	Free-Form Text
State	Free-Form Text
Zip	Free-Form Text
Country	Free-Form Text
Sales Rep	List/Record
Expected Close Date	Date
Renewal Date	Date
Contact	Free-Form Text
Alt. Contact	Free-Form Text
Alt. Phone	Phone Number
Balance	Currency
Credit Limit	Currency

Source From Sales Rep

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



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

Field	Type
Name	Free-Form Text
Bill To	Text Area
Phone	Phone Number
Email	Email Address
City	Free-Form Text
Country	Free-Form Text
Supervisor	List/Record
Soc. Sec. #	Free-Form Text

Transaction Line Fields

You can source from an Item or a Customer field.

Source From Item



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

Field	Type
Name	Free-Form Text
Display Name	Free-Form Text
Vendor Name	Free-Form Text
Online Name	Free-Form Text
Available Online	Check Box
Base Price	Currency
Cost	Currency
Preferred Vendor	List/Record
On Hand	Decimal Number

Source From Customer

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



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

Field	Type
Name	Free-Form Text
Bill To	Text Area
Ship To	Text Area
Phone	Phone Number
Fax	Phone Number
Email	Email Address
City	Free-Form Text
State	Free-Form Text
Country	Free-Form Text
Sales Rep	List/Record
Expected Close Date	Date
Renewal Date	Date
Contact	Free-Form Text
Alt. Contact	Free-Form Text
Alt. Phone	Phone Number
Balance	Currency
Credit Limit	Currency
Account	Free-Form Text

Transaction Item Options

You can source from an Item field.

Source From Item



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

Field	Type
Name	Free-Form Text
Display Name	Free-Form Text
Vendor Name	Free-Form Text
Online Name	Free-Form Text
Available Online	Check Box

Field	Type
Base Price	Currency
Cost	Currency
Preferred Vendor	List/Record
On Hand	Decimal Number

CRM Fields

You can source from a Company, Contact, Item, or Assigned field.

Source From Company or Contact

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

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

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

Field	Type
Name	Free-Form Text
Bill To	Text Area
Ship To	Text Area
Phone	Phone Number
Fax	Phone Number
Email	Email Address
City	Free-Form Text
State	Free-Form Text
Country	Free-Form Text
Sales Rep	List/Record
Expected Close Date	Date
Renewal Date	Date
Contact	Free-Form Text
Alt. Contact	Free-Form Text
Alt. Phone	Phone Number
Balance	Currency
Credit Limit	Currency
Account	Free-Form Text

Source From Item

When you select Item in the Source From field, information is sourced from the record selected in the **Item** field of **case** records.



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

Field	Type
Name	Free-Form Text
Display Name	Free-Form Text
Vendor Name	Free-Form Text
Online Name	Free-Form Text
Available Online	Check Box
Base Price	Currency
Cost	Currency
Preferred Vendor	List/Record
On Hand	Decimal Number

Source From Assigned

When you select Assigned in the Source From field, information is sourced from the record selected in the **Assigned To** field of **case** records.



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

Field	Type
Name	Free-Form Text
Bill To	Text Area
Phone	Phone Number
Email	Email Address
City	Free-Form Text
Country	Free-Form Text
Supervisor	List/Record
Soc. Sec. #	Free-Form Text

Entity Fields

You can source from a Sales Rep or Supervisor field.

Source From Sales Rep or Supervisor

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

When you select Supervisor in the Source From field, information is sourced from the records selected in the **Supervisor** field of **employee** records.

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

Field	Type
Name	Free-Form Text
Bill To	Text Area
Phone	Phone Number
Email	Email Address
City	Free-Form Text
State	Free-Form Text
Zip	Free-Form Text
Country	Free-Form Text
Supervisor	List/Record
Soc. Sec. #	Free-Form Text

Item Fields

You can source from a Preferred Vendor field.

Source From Preferred Vendor

When you select Preferred Vendor in the Source From field, information is sourced from the record selected in the **Preferred Vendor** field of **item** records.

Field	Type
Name	Free-Form Text
Bill To	Text Area
Ship To	Text Area
Phone	Phone Number
Fax	Phone Number
Email	Email Address
City	Free-Form Text
State	Free-Form Text
Zip	Free-Form Text

Field	Type
Country	Free-Form Text
Contact	Free-Form Text
Alt. Contact	Free-Form Text
Alt. Phone	Phone Number
Balance	Currency
Credit Limit	Currency
Account	Free-Form Text
1099 Eligible	Check Box
Tax ID	Free-Form Text
Legal Name	Free-Form Text

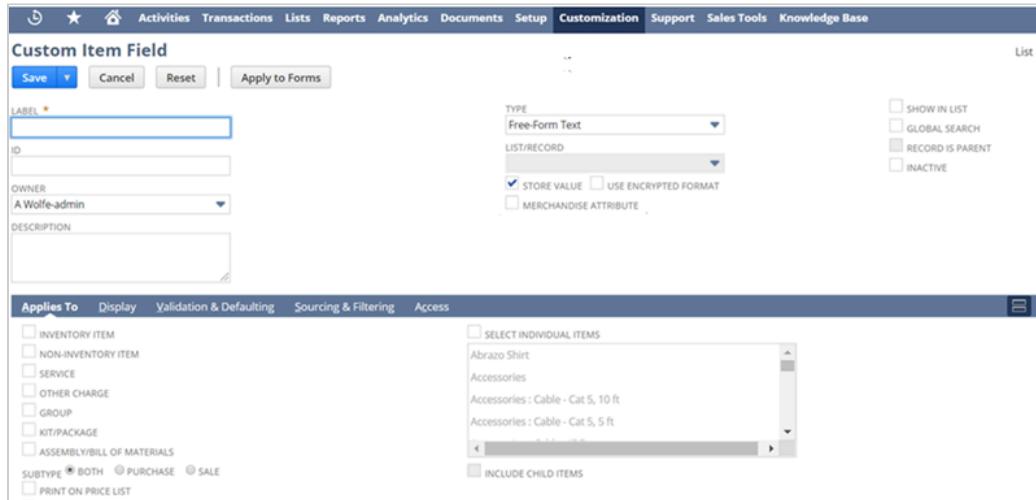
Creating a Custom Field

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

For an overview of the various types of custom fields, see [Custom Field Types](#).

To create a custom field:

1. Go to Customization > Lists, Records, & Fields > [Custom Field] > New, where [Custom Field] is the required field type. The Custom Field page is displayed for the selected type. Options available vary depending on the custom field type selected. The Custom Item 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. In the ID field, enter a unique alphanumeric ID for the field. For information about best practices and naming conventions, see [Conventions for Naming Custom Objects](#). For information about changing an existing ID, see [Changing the ID of a Custom Object](#).
4. Select the owner of the custom field.
Only the owner can modify the record. Your name is selected by default.
5. In the **Description** field, enter a description of the custom field.
6. For custom item fields, to indicate that the field is an option for matrix items, check the **Matrix Items** box. Checking this box automatically sets the Type field to Multiple Select and the Subtab field on the Display subtab to Matrix. Changing any of these settings clears the Matrix Option box. For more information about matrix items, see the help topic [Matrix Items](#).
7. In the **Type** field, select the type of custom field you want to create.

For more information on custom field types, see [Field Type Descriptions for Custom Fields](#).

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



Note: If you later edit a field and change it to a type that is not supported, any data stored for the field is deleted. If you change a field type or a list/record that is used for sourcing or filtering on other custom fields to a field type that is not supported, all sourcing and filtering based on the field is removed.

Type conversion of the field takes a significant amount of time because many records are affected. A timeout can occur. For more information, see [Converting the Field Type of a Custom Field](#).

8. If you selected **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: For List/Record or Multiple Select custom field types, standard security restrictions are not applied to the options in the related List/Record list. It is possible to select a record from the list even if you do not have permission to view the record. However, you cannot edit the record if you do not have edit permission.

9. By default, the **Store Value** option is enabled so that custom field values are stored in your NetSuite account. If you do not want any changes entered stored in the custom field, clear the **Store Value** box. 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. If you clear the **Store Value** box, the field value is accessible in the current workflow instance only.
You cannot clear the **Store Value** box if the value is used for criteria in duplicate detection. If you try to do so, you will receive an error message. For more information about duplicate detection, see the help topic [Setting Up Duplicate Detection](#).
10. Some custom field data types provide a **Use Encrypted Format** option. To encrypt the field values stored in the database, check the box.



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 ensure 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.

11. 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 appears only 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](#)
12. To have the field automatically show in the list of records the field is applied to, check the **Show In List** box. The lists the custom field appears in depends on the field type. For example, entity fields appear in applicable entity lists, transaction body fields appear in transaction lists, CRM fields appear in CRM record lists, and so on.
13. To index this custom field for global search, so that this field's values are searched for matches to global search keywords, check this box. You can index the field for global searches if **Store Value** is checked, and if you have selected 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](#).

14. If you are creating a List/Record custom field, and the record type selected is a parent record, check the **Record is Parent** box.
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](#).
15. If you want to deactivate the field after it is created, check the **Inactive** box. For more information, see [Inactivating a Custom Field](#).
You cannot clear the **Inactive** box if the custom field is used for criteria in duplicate detection. If you try to do so, you will receive an error message. For more information about duplicate detection, see the help topic [Setting Up Duplicate Detection](#).
16. 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 an example of creating and adding a custom field to a record type, watch the following video.

[Creating and Adding a Custom Field to a Record](#)

You can hide a field on a form by setting the field to not display or by hiding the subtab on which the field appears. For more information, see [Setting Display Options for Custom Fields](#) and [Configuring Subtabs for Custom Entry and Transaction Forms](#). Note that if the field is hidden on your preferred form, you cannot make inline edits to the field on saved search results.

You should use your account-specific domains for custom field types.

When you create a custom field, if you attempt to add a URL that includes a data center-specific domain, the URL is automatically modified to your account-specific domain for users in their current session. Your account-specific domain is in the format <https://<accountID>.app.netsuite.com>, where <accountID> represents your account ID. The modification 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](#).

See the following topics.

- [Creating Custom Fields by Type](#)
- [Converting the Field Type of a Custom Field](#)
- [Assigning Custom Fields to Specific Record Types](#)
- [Setting Display Options for Custom Fields](#)
- [Setting Validation and Defaulting Properties](#)
- [Setting Sourcing Criteria](#)
- [Sourcing and Filtering Examples](#)
- [Setting Filtering Criteria](#)
- [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](#)

Creating Custom Fields by Type

You can create the following types of custom fields:

- Custom CRM fields (see [Creating Custom CRM Fields](#)).
- Custom Entity fields (see [Creating Custom Entity Fields](#)).
- Custom Item fields (see [Creating Custom Item Fields](#)).
- Custom Transaction Body fields (see [Creating Custom Transaction Body Fields](#)).
- Custom Transaction Line Fields (see [Creating Custom Transaction Line Fields](#)).

- Custom Item Number fields (see [Creating Custom Item Number Fields](#)).
- Custom Transaction Item options (see [Creating Custom Transaction Item Options](#)).
- Other Custom fields (see [Creating Other Record Fields](#)).
- Other Sublist fields (see [Creating Other Sublist Fields](#)).

Creating 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
- issue records

To create or modify custom CRM fields:

1. Go to Customization > Lists, Records, & Fields > 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. Choose an option:
 - To edit an existing custom CRM field, click the field name in the Description column and then modify the field definition as needed.
 - To add a new custom CRM field, click **New**.
- For more information, see [Creating a Custom Field](#).
4. Complete fields on the Custom CRM Field page as needed, and then click **Save**.



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 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

LABEL * Followup

ID custevent_sdr_phonecall_followup

OWNER Smith, John

DESCRIPTION

TYPE Check Box

LIST/RECORD

Applies To

- TASK
- PROJECT TASK
- MANUFACTURING OPERATION TASK
- PHONE CALL
- EVENT
- CASE
- CAMPAIGN
- PER KEYWORD
- SOLUTION
- ISSUE
- SHOW CHANGES
- AVAILABLE EXTERNALLY

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

Event

Primary Information

CUSTOM FORM * Standard Event Form

LOCATION

TITLE *

STATUS * Confirmed

Date and Time

DATE * 6/26/2015 ALL DAY RESERVE TIME

START TIME * 2:00 pm 2:00 pm

END TIME * 3:00 pm 3:00 pm

FOLLOWUP

Message **Attendees** **Resources** **Availability** **Recurrence** **Related Records** **Communication** **Custom**

Creating 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
- project records
- vendor records
- other name records
- contact records

- partner records
- entity group records
- employee records
- website registration
- generic resource

To create or modify custom entity fields:

1. Go to Customization > Lists, Records, & Fields > 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. Choose an option:
 - To edit an existing custom entity field, click the field name in the Description column and then modify the field definition as needed.
 - To add a new custom entity field, click **New**.
 For more information, see [Creating a Custom Field](#).
4. Complete fields on the Custom Entity Field page as needed, and then click **Save**.

i 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.

The screenshot shows the 'Custom Entity Field' page in NetSuite. At the top, there are buttons for Save, Cancel, Reset, Change ID, and Apply to Forms, along with an Actions dropdown. The main form contains the following fields:

- LABEL ***: Travel Approval Limit
- ID**: custentity_travel_approval_limit
- OWNER**: K Wolfe
- DESCRIPTION**: (empty text area)
- TYPE**: Currency
- LIST/RECORD**: (dropdown menu)
- Access Options** (checkboxes):
 - STORE VALUE (checked)
 - USE ENCRYPTED FORMAT
 - SHOW IN LIST
- Global Search Options** (checkboxes):
 - GLOBAL SEARCH
 - RECORD IS PARENT
 - AVAILABLE TO SUITESIGNON

Below the main form is a tabbed section with tabs: Applies To, Display, Validation & Defaulting, Sourcing & Filtering, Access, Translation, and History. Under the 'Applies To' tab, there are two columns of checkboxes:

CUSTOMER	CONTACT	AVAILABLE EXTERNALLY
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

PROJECT	PARTNER	PRINT ON STATEMENT
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

VENDOR	GENERIC RESOURCE	PRINT ON PRICE LIST
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

EMPLOYEE	WEB SITE	PROJECT TEMPLATE
<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

OTHER NAME	GROUP
<input type="checkbox"/>	<input type="checkbox"/>

At the bottom of the page are Save, Cancel, Reset, Change ID, Apply to Forms, and Actions buttons.

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

Communication	Address	Human Resources	ACH/Direct Deposit	Time Tracking	Payroll	Commission	Related Records	Marketing	Access	System Information
SOCIAL SECURITY	BIRTH DATE									
Job Information										
TYPE	<input type="checkbox"/> SUPPORT REP	<input type="checkbox"/> PROJECT RESOURCE		HIRE DATE 21.10.2014						
EMPLOYEE STATUS			WORK CALENDAR Standard US Employee Work Calendar				LAST REVIEW DATE			
JOB DESCRIPTION			LABOR COST				NEXT REVIEW DATE			
SALES ROLE							TERMINATION DATE			
Expense and Purchasing										
EXPENSE LIMIT		PURCHASE LIMIT		PURCHASE APPROVAL LIMIT						
EXPENSE APPROVER		PURCHASE APPROVER		ACCOUNT						
EXPENSE APPROVAL LIMIT										
TRAVEL APPROVAL LIMIT 1000.00										

Creating 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
- noninventory 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. Choose an option:
 - To edit an existing custom item field, click the field name in the Description column and then modify the field definition as needed.
 - To add a new custom Item field, click **New**.
- For more information, see [Creating a Custom Field](#).
4. Complete fields on the Custom Item Field page as needed, and then click **Save**.



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](#)

[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 could look like the following example.

Creating 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
- transfer orders
- item receipts
- item fulfillments
- fulfillment requests
- store pickup fulfillments
- inventory adjustments
- work orders / assembly builds
- customer payments
- vendor payments
- vendor prepayments
- deposits
- deposit applications
- pay checks
- other transaction types

You can also choose to display custom transaction body fields on the checkout page of your web store transaction. If you do this, the fields are automatically added to the body of sales transaction records.

To create or modify custom transaction body fields:

1. Go to Customization > Lists, Records, & Fields > 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. Choose an option:
 - To edit an existing custom transaction body field, click the field name in the Description column and then modify the field definition as needed.
 - To add a new custom transaction body field, click **New**.
- For more information, see [Creating a Custom Field](#).
4. Complete fields on the Transaction Body Field page as needed, and then click **Save**.



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.

The screenshot shows the 'Transaction Body Field' configuration page. At the top, there are buttons for 'Save' (highlighted in blue), 'Cancel', and 'Apply to Forms'. Below these are fields for 'LABEL *' (set to 'Entered By'), 'ID' (set to 'custbody25'), 'OWNER' (set to 'Jason Toth'), and 'DESCRIPTION' (an empty text area). On the right, there are dropdowns for 'TYPE' (set to 'Free-Form Text') and 'LIST/RECORD' (set to 'LIST'). There are also three checkboxes: 'STORE VALUE' (checked), 'USE ENCRYPTED FORMAT' (unchecked), and 'SHOW IN LIST' (unchecked). To the right of these checkboxes are three checkboxes for 'GLOBAL SEARCH', 'RECORD IS PARENT', and 'INACTIVE'. Below the main configuration area is a tabbed navigation bar with tabs for 'Applies To', 'Display', 'Validation & Defaulting', 'Sourcing & Filtering', 'Access', and 'Translation'. Under the 'Applies To' tab, there are two columns of checkboxes. The left column includes 'PURCHASE' (checked), 'SALE', 'OPPORTUNITY', 'JOURNAL', 'EXPENSE REPORT', 'TRANSFER ORDER', and 'ITEM RECEIPT'. The right column includes 'ITEM FULFILLMENT', 'VIEW FROM ORDER ONLY', 'INVENTORY ADJUSTMENTS', 'ASSEMBLY BUILD', 'CUSTOMER PAYMENT', 'VENDOR PAYMENT', 'DEPOSIT', and 'DEPOSIT APPLICATION'. To the right of the tabs is a 'List' button. At the bottom of the page are 'Save' (highlighted in blue), 'Cancel', and 'Apply to Forms' buttons.

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.

The screenshot shows a 'Purchase Order' form for purchase order 'AMS3339-PO' from 'Acme Medical Supply'. The status is 'FULLY BILLED'. The form includes sections for 'Primary Information', 'Classification', and 'Batch Transaction Information'. In the 'Primary Information' section, there is a checkbox for 'SUPERVISOR APPROVAL' which is checked. In the 'Classification' section, there is a table with columns for 'DEPARTMENT' (Accounting), 'CLASS' (Medical), and 'LOCATION'. In the 'Batch Transaction Information' section, there is a table with columns for 'VIEW' (Default View), 'BATCH TRANSACTION INFORMATION', 'EDIT', 'BATCH', and 'REMOVE'. A note at the bottom says 'No records to show.' The 'Entered By:' field is highlighted with a red box. The bottom of the screen shows the SuiteBuilder logo and the Oracle NetSuite logo.

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.

If the Log System Notes on Update Only preference is not set, and you have the View access level to a custom transaction body field, the default value of your field is displayed in system notes fields when you create a transaction search. For more information, see the help topic [Searching System Notes](#).

Creating 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
- work orders
- paycheck earnings
- paycheck deductions
- paycheck employee taxes
- paycheck company taxes
- paycheck company contributions

To create or modify custom transaction line fields:

1. Go to Customization > Lists, Records, & Fields > Transaction Column 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. Choose an option:
 - To edit an existing custom Transaction Line field, click the field name in the Description column and then modify the field definition as needed.
 - To add a new custom Transaction Line field, click **New**.

For more information, see [Creating a Custom Field](#).

4. Complete fields on the Transaction Line Field page as needed, and then click **Save**.

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 could 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 previously described. 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.

Creating 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 – see the help topic [Lot Numbered Items](#)
- gift certificate items – see the help topic [Gift Certificates](#)



Note: To use custom item number fields, at least one of the features in the preceding list must be enabled.

To create a custom item number field

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. Choose an option:
 - To edit an existing custom item number field, click the field name in the description column and then modify the field definition as needed.
 - To add a new custom item number field, click **New**.
 For more information, see [Creating a Custom Field](#).
3. Complete fields on the Item Number Field page as needed, and then click **Save**.

When you create a lot numbered or serialized inventory item or edit an existing record, you can apply the field to the item if 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 could 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](#).

Creating 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 can record multiple colors of an item as item options.

You can create item options for:

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

i Note: If there are too many characters used in the item option fields on your form, you can encounter performance issues. Try to limit the number of custom item option fields to 50 fields with a maximum total character count of 4000. The total character count includes the field names and their values.

To create or modify custom transaction item option fields:

1. Go to Customization > Lists, Records, & Fields > Transaction Item Option.
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. Choose an option:
 - To edit an existing custom **Transaction Item Option** field, click the field name in the description column and then modify the field definition as needed.
 - To add a new custom **Transaction Item Option** field, click **New**.
 For more information, see [Creating a Custom Field](#).
4. Complete fields on the Item Option page as needed, and then click **Save**.

i 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. However, 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:

1. Go to Customization > Lists, Records, & Fields > Transaction Item Options > New > New.
2. In the **Label** field, enter the name of your option, **Gift Wrapping**.
3. To create a check box field, in the **Type** field, select **Check Box**.
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 select 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 it should appear to customers on your website.

7. 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.

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

Creating 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	Location
Accounting Book	Location Costing Group
Address (See Creating Custom Address Fields)	Manufacturing Cost Template
Allocation Type	Manufacturing Routing
Automatic Location Assignment Configuration	Note
Automatic Location Assignment Rule	Payroll Batch
Bill of Distribution	Payroll Item
Bill of Materials	Planned Order
Bill of Materials Revision	Planned Standard Cost
Billing Rate Card	Planning Item Group
Bin	Planning Rule Group
Campaign Event	Product
Charge	Product Version
Charge Rule	Project Budget
Class	Promotion Code
Company	Region
Competitor	Role
Department	Retirement Plan
Distribution Category	Sales Channel
Distribution Network	Shipping Partner Package
Domain	Shipping Partner Registration
Expense Category	Shipping Partner Shipment
Format Profile	Standard Cost Version
GL Numbering Sequence	Subsidiary
Global Account Mapping	Subsidiary Setting
Global Inventory Relationship	Supply Change Definition
Imported Employee Expense	Supply Change Order
Inbound Shipment	Supply Chain Snapshot
Inventory Cost Template	Supply Chain Snapshot Simulation

Inventory Status	Tax Code
Invoice Group	Tax Type
Item Account Mapping	Timesheet
Item Demand Plan	Vendor-Subsidiary Relationship
Item Location Configuration	Work Breakdown Structure
Item Source	Workflow
Item Supply Plan	Workplace

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. Choose an option:
 - To edit an existing record field, click the field name in the **Description** column and then modify the field definition as needed.
 - To add a new custom record field, click **New**.
 For more information, see [Creating a Custom Field](#).
4. Complete fields on the Other Record Field page as needed, and then click **Save**.

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 could look like the following example.

The screenshot shows the 'Role' setup page in Oracle NetSuite. The 'Edit' button is highlighted. On the right, there's a list of checkboxes for role types: DO NOT RESTRICT EMPLOYEE FIELDS, RESTRICT TIME AND EXPENSES, SALES ROLE (which is checked), SUPPORT ROLE, WEB SERVICES ONLY ROLE, SINGLE SIGN-ON ONLY, PARTNER ROLE, RESTRICT THIS ROLE BY IP ADDRESS (which is checked and highlighted with a red box), SYSTEM ROLE, and INACTIVE.

PERMISSION	LEVEL
Cash Sale	Full
Edit Forecast	Full
Invoice	Full
Opportunity	Full
Estimate	Full
Sales Order	Full

Creating 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.

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. Choose an option:
 - To edit an existing custom sublist field, click the field name in the Description column and then modify the field definition as needed.
 - To add a new custom sublist field, click **New**.
 For more information, see [Creating a Custom Field](#).
4. Complete fields on the Other Sublist Field page as needed, and then click **Save**.

Here is an example of a custom sublist field.

Other Sublist Field

Save ▾ Cancel Reset

SUBLIST LINE TYPE * Bill of Materials Revision Component

LABEL * Update

ID _update

OWNER JL Wolfe

DESCRIPTION

TYPE Free-Form Text

LIST/RECORD

STORE VALUE USE ENCRYPTED FORMAT
 INACTIVE

Display Validation & Defaulting Access Translation

INSERT BEFORE

DISPLAY TYPE Normal

DISPLAY WIDTH

HELP

Save ▾ Cancel Reset

When the other sublist field is included in a form, it could look like the following example.

Bill of Materials Revision

Save Cancel | Customize Form | Actions ▾

NAME * Update 1

MEMO

BILL OF MATERIALS Standard Installation

EFFECTIVE START DATE 11/16/2017

EFFECTIVE END DATE

INACTIVE

DATE CREATED 11/16/2017

Components System Notes

ITEM *	DESCRIPTION	COMPONENT YIELD	BOM QUANTITY	ITEM SOURCE	QUANTITY *	VERSION	UPDATE
:: Ergonomic Keyboard	Ergonomic Keyboard	100.0%	1	Stock	1	6	
		100.0%	1		1		

Add Cancel Insert Remove Move Up Move Down Move To Top Move To Bottom

Save Cancel | Customize Form | Actions ▾

Assigning Custom Fields to Specific Record Types

You can assign custom fields to display on specific record types. When you assign the field 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 you select a record type, the field automatically displays on all forms of that record type, including any custom forms. You can then edit custom forms to not show the new custom field.

Conversely, all custom transaction fields are automatically available in form customization, regardless of what you select on the **Applies To** subtab. For example, if you apply a custom transaction field to sales transaction forms, the field is also available when you customize purchase transaction forms.

For more information about available custom field types, see [Field Type Descriptions for Custom Fields](#).

To apply a custom field to a record type:

1. On the custom field setup page, 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.
- You can add record custom fields to existing and custom subtabs on the entry forms you use to enter records in your NetSuite account.
- You can add transaction custom fields 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, 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](#).
- You can configure the kit or assembly transaction line custom field to copy values from sales order items to fulfillment items. For more information, see [Apply to Kit or Assembly Components Setting for Custom Segments and Transaction Line Custom Fields](#).
- For custom transaction body fields, if the Log System Notes on Update Only preference is not set, and you have the View access level to a field, the default value of your custom transaction body field is displayed in system notes fields when you create a transaction search. For more information, see the help topic [Searching System Notes](#).

You cannot clear the **Applies To** box if the custom field is used for criteria in duplicate detection. If you try to do so, you will receive an error message. For more information about duplicate detection, see the help topic [Setting Up Duplicate Detection](#).

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. On the custom field setup page, click the **Display** subtab.

The fields available on the subtab depend on the field type 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 optionally have the custom field span the column at the top or bottom of a subtab or section. Spanning the column is especially useful to add a text area field at the top or bottom of a subtab or section where you can provide explanatory information. For more information, see [Adding a Field that Spans the Top or Bottom of a Page](#).

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.



Note: To display in saved searches, the field must be set up to display on your preferred form and included on a subtab that displays the form. For more information, see the help topic [Configuring Subtabs for Custom Entry and Transaction Forms](#).

- In the **Display Type** field, 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 only for informational purposes and not have them 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.

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 required 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. The value 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. The information will not be 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. Also, the custom field 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.

In SuiteScript, only user event, scheduled, and Suitelet scripts can set the value of a custom field that has a display type of hidden. For more information about custom coding with SuiteScript, see the help topic [SuiteScript 2.x Script Types](#).

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.

If the Log System Notes on Update Only permission is not set, when you create a new transaction, the default value of a hidden field appears on the System Information subtab on transaction forms where the field is applied. For more information, see the help topic [Setting General Account Preferences](#).



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 about 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.



Important: Field configurations on each custom form interact with the values you set here. Actions such as setting up field groups, adding column breaks, and setting fields to be **Same Row as Previous** on custom forms can cause custom field text to be displayed in an unexpected way based on the custom field width. For details about configuring fields on custom forms, see [Configuring Field Groups](#), [Configuring Fields or Screens](#), and [Associating Related Fields on Custom Forms](#).

6. (Hyperlink fields only) If needed, 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 needed. To enable the setting, **check** the box. To disable the setting, **clear** the box.

- When Apply Formatting 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 Apply Formatting 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.)



Note: Default settings are as follows: disabled for custom fields created prior to 2012.1 upgrade, to preserve pre-upgrade behavior, and enabled for custom fields created after 2012.1 upgrade.

8. In the **Help** field, enter a brief description of the purpose of the field and the information that should be entered into the field.

The help is available when a user clicks the name of the field.

You can enter HTML in the field. Be sure to begin your markup source with <html> and end with </html>.



Note: You should always enter information in the Help field to increase the usability of customized forms or records.

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](#).

Adding Field-Level Help for Custom Fields

Field-level help is a brief description of the purpose of a field and the information that should be entered into a field. To increase the usability of customized forms or records, you should always provide field-level help for custom fields. The help is available when a user clicks the name of the field.

You add custom field-level help to custom fields on the Display subtab of the custom field setup page.

For information about adding field-level help for custom fields, watch the following help video.



[Adding Field-Level Help for Custom Fields](#)

To add custom field-level help:

1. Go to Customization > Lists, Records, & Fields > [Custom Field], where [Custom Field] is the required field type.
2. To create and provide help for a new custom field, click **New**. Otherwise, to add help to an existing custom field, select the field from the list.
3. Click the **Display** subtab.
4. In the **Help** field, enter the field-level help description.
5. Click **Save**.

For more information about the Display settings for custom fields, see [Setting Display Options for Custom Fields](#).

You can define translations for the Field-Level Help of a custom field on the Translation subtab of the custom field page. For more information, see [Adding Translations for Custom Fields](#).

Adding a Field that Spans the Top or Bottom of a Page

When determining where a custom field displays on a form, for some field types you can optionally have the field span the column at the top or bottom of a subtab or section. Spanning the column is especially useful to add a text area field at the top or bottom of a subtab or section where you can provide explanatory information.

On the Insert Before field of the Display subtab, select either **Outside columns at top** or **Outside columns at bottom**. You can optionally enter a display width, display height, or both in the Display Size field to set the initial size of the field. You can later manually change the field's display size on the form, if needed. If you do not specify the initial display size, the size defaults to three columns wide and three lines high.

The field displays on the form above or below the other fields on the subtab or section. You can change the size of the field by dragging the control in the bottom right corner of the field. The column width is determined by the size of the field. If the column spans beyond the page width, a horizontal scroll bar is added to the subtab or section. This scroll bar applies only to the subtab or section where the field appears.

The following screenshot shows the Miscellaneous Instructions field spanned across the page of the Purchasing/Inventory subtab.

For more information about creating a field that spans the top or bottom of a page, watch the following video.



For more information about Display Options, see [Setting Display Options for Custom Fields](#).

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 automatically when a record or transaction is first created.

Watch the following video for an example of setting a default selection to a custom field.



To set validation and defaulting properties:

1. On the custom field setup page, click the **Validation & Defaulting** subtab.

The screenshot shows the 'Validation & Defaulting' tab of the SuiteBuilder interface. At the top, there are tabs for 'Applies To', 'Display', 'Validation & Defaulting' (which is selected), 'Sourcing & Filtering', 'Access', and 'Translation'. Under 'Validation & Defaulting', there are sections for 'MANDATORY' (with checkboxes for 'MANDATORY' and 'CHECK SPELLING'), 'MAXIMUM LENGTH' (with a text input field), 'DEFAULT VALUE' (with a text input field and a 'FORMULA' checkbox), 'SEARCH' (with a dropdown menu), and 'FIELD' (with a dropdown menu).

- Set the validation options. (These options can vary according to the 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 various 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 less than 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.

- 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 **Prevent and Return Error**.

- To permit deletions, null out field values that use the deleted record, and log system notes for deletions, select **Allow and Set Dependent Field Values to Null**.

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 about 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 available only for 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:** You can base values for the custom field on summary search results. Optionally, select a comparison field to join related records in cases where you want to place the custom field on a form for a record type that is different from the summary search record type. For example, you may want to place a custom entity field showing the result of a customer record summary search on an employee custom form. To do so, 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 available only for 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 needed, 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:** To indicate that the check box custom field should display checked by default, check the box. 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 [Creating Custom Transaction Line Fields](#).

- **Default Value:** If needed, 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 Inline HTML field, the default value is applied when you create a new record or when you load an existing record.

When working with Free-form text, Text area, Rich Text or Hyperlink 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 applied only 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 default options specific to the custom field you are creating based on its field type. There are six types of dynamic defaults:

- **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. The preference is 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, 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. This information can help 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, 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, 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, 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 needed.

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 available. 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.

The screenshot shows the 'Custom Entity Field' configuration page. The 'Validation & Defaulting' tab is active. In the 'ALLOW DELETE OF LIST/RECORD VALUES?' dropdown, the 'Prevent and Return Error' option is selected. Other options include 'Prevent and Return Error' and 'Allow and Set Dependent Field Values to Null'. The rest of the page includes fields for Label, Type (Multiple Select), List/Record (Account), and various checkboxes for global search, record is parent, and available to suitesignon.

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 to a null value, 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: custom field type, 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. Therefore, so a setting of **Allow and Set Dependent Field Values to Null** can produce nulled values even for mandatory fields. To avoid nulled values, select 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

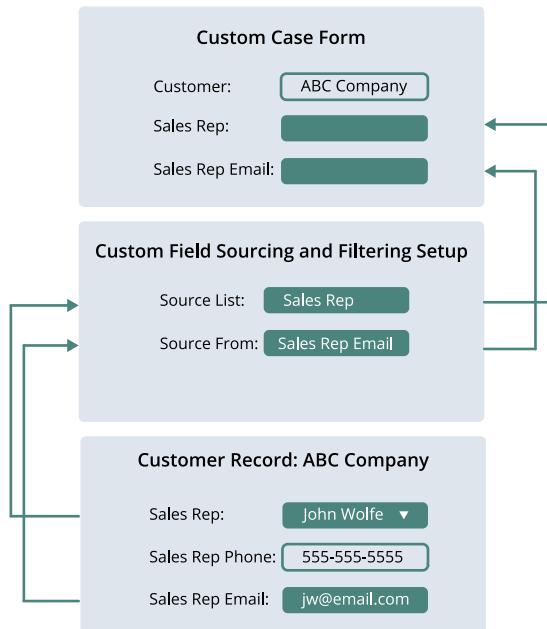
Values in custom fields can be sourced from information from another record in your account. The information populated in the custom field is then dependent on fields associated with a record selected

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.

Sourcing can be done 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 customer record is selected in the Customer field, the sales representative already defined in the selected customer record is sourced to the Sales Rep field on the case form. The Sales Rep Email field defaults to the email address defined for the sourced Sales Rep.

The following diagram shows the custom field sourcing and filtering setup required to populate the Sales Rep and Sales Rep Email fields on a custom case form with data from the customer record:



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

When setting up sourcing, you can store the sourced value in the custom field. When the field value is not stored, the information is not saved in the custom field. By not saving the value in the custom field you can instead view data that is stored elsewhere and populated in the custom field.

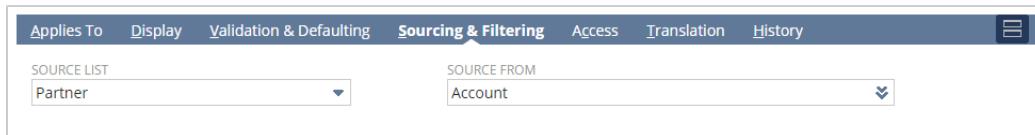
You can store a sourced value so that it will not be overwritten if the source field changes. To use a store value, the custom field must define the Source List and Source From fields on the Sourcing & Filtering subtab. The sourcing automatically fills the custom field with a value when the primary field, defined in the Source List field, is changed. If Store Value is checked, then the value populated in the field is stored on the record. You can optionally change the value on the custom field to have that value stored instead of the populated value. If the value in the source field changes, the stored value will not be overwritten.

For information about custom transaction line field sourcing prior to 2021.2, see [Note about Custom Transaction Line Field Sourcing](#).

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. For an example of a situation where it could make sense to store a value, see [Example of Storing a Value](#).
3. On the **Display** subtab, review the **Display Type** field. If the **Display Type** field is set to **Hidden**, sourcing does not occur. To have sourcing occur, select a **Display Type** other than **Hidden**.
4. Click the **Sourcing & Filtering** subtab.



5. From the **Source List** list, select the field that references the record you want 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 **Vendor** in the **Source List** field instead of **Entity** to obtain the required 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: Information for a Multiple Select field type cannot be sourced.

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.

In the **Source From** list, you can select any fields available on the record you select in the **Source List** field.

In the preceding example, select **Email** from the **Source From** list.

When the record is selected, the field is populated with the value stored in the field selected here. 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.

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

When defining a List/Record Type field, you can choose to populate the custom field with values that meet specific parameters in the sourced list or record. First, select the required item to filter by from the **Source Filter by** list. Then select an item from the **Source List** and, optionally, from the **Source From** list. When you select an element from the source list, it fills 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](#).

Watch the following help video for information about using sourcing and the store value feature with a custom field.

 [Using Sourcing and the Store Value Feature with a Custom Field](#)

Example of Storing a Value

The following example describes a situation where it could make sense to store a value.

John Wolfe is the sales representative for a specific region. He makes some sales on sales order SO-123. John Wolfe's name is automatically sourced to the sales order as the sales representative for the region. You always want John Wolfe to be associated as the sales representative on sales order SO-123. In the future, Mary Brown replaces John Wolfe as the region's sales representative. You don't want Mary's name to replace John's name on sales order SO-123. In this situation, it makes sense to store the sales rep value on the sales order.

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. When you select Customer, the new custom field's values are sourced from the Customer body field on Sales Orders.

Transaction Column Field

Save ▾ Cancel Reset Apply to Forms

LABEL * Customer Info

ID

OWNER Sloan, Cindy

DESCRIPTION

TYPE Free-Form Text

LIST/RECORD

STORE VALUE USE ENCRYPTED FORMAT

Applies To Display Validation & Defaulting Sourcing & Filtering Access Translation

SOURCE LIST Customer

Customer Customer (Line) Customer (Payments and Deposits) Customer Category Dealer Services Decision Maker Deferred Revenue Account for Revenue Reclassification

SOURCE FROM <Type then tab>

Sourcing and Filtering Examples

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

Linking Two Transactions Example

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 to the credit form only). 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, NetSuite populates the new list field with only invoices from that customer.

ORACLE NETSUITE

Activities Transactions Lists Reports Analytics Documents Setup Customization Commerce SuiteApps Support

Transaction Body Field

Save ▾ Cancel Change ID Apply to Forms Actions ▾

LABEL * Related Transaction

ID custbody_we_related_trans

OWNER Jl Wolfe

DESCRIPTION

TYPE List/Record

LIST/RECORD Transaction

STORE VALUE USE ENCRYPTED FORMAT

SHOW IN LIST

GLOBAL SEARCH RECORD IS PARENT INACTIVE

Applies To Display Validation & Defaulting Sourcing & Filtering Access Translation History

SOURCE LIST Entity SOURCE FILTER BY Entity

SOURCE FROM Entity

FILTER USING * IS CHECKED COMPARE TYPE COMPARE VALUE TO VALUE IS IS NOT EMPTY IS EMPTY COMPARE TO FIELD

Add Cancel Insert Remove

Save ▾ Cancel Change ID Apply to Forms Actions ▾

This screenshot shows a custom form in SuiteBuilder. At the top, there's a navigation bar with tabs: Items, Shipping, Billing, Accounting, Relationships, Sales Team, Communication, Address, Custom, E-Document, and Warranty. The 'Custom' tab is selected. Below the navigation bar, there are several input fields and dropdown menus. Some fields have validation errors displayed above them. For example, the 'TEST DELETE FIELD' dropdown has an error message: 'ERROR: Field 'linenumber' Not Found'. The 'Batch Transaction Information' section contains a 'Clear All Lines' button. On the right side, there are sections for Website, Customer Contacts, Transaction Logo, and Related Transaction, each with its own dropdown menu.

Combining Static and Dynamic Filtering Example

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, and select a customer, the Linked Order list is populated with sales orders from that customer.

This screenshot shows the 'Transaction Body Field' configuration screen. At the top, there are buttons for Save, Cancel, Reset, Change ID, and Apply to Forms. Below these are fields for Label (Related Transaction), Description, Type (List/Record), and various checkboxes for visibility and search. The 'Sourcing & Filtering' tab is selected. Under this tab, there are fields for Source List (Entity) and Source Filter By (Entity). A table below shows a static filter rule: 'Type' is checked, 'IS CHECKED' is selected, 'COMPARE TYPE' is 'equal', 'COMPARE VALUE TO' is 'Sales Order', and 'VALUE IS' is 'Sales Order'. There are also buttons for Add, Insert, and Remove.

Filtering Against the Source From Value Example

Suppose you want to have a field on a task record that lists all 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). Because 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 that appears in the Subordinates field on task records filters to cases where the Source Filter By value (Supervisor) equals the Sourced From value (Sales Rep).

Custom CRM Field

LABEL * Subordinates **ID** _subordinates **OWNER** JL Wolfe

TYPE List/Record **LIST/RECORD** Employee **STORE VALUE** USE ENCRYPTED FORMAT SHOW IN LIST

DESCRIPTION

Applies To **Display** **Validation & Defaulting** **Sourcing & Filtering** **Access** **Translation**

SOURCE LIST Opportunity **SOURCE FROM** Sales Rep **SOURCE FILTER BY** Supervisor

FILTER USING *	IS CHECKED	COMPARE TYPE	COMPARE VALUE TO	VALUE IS	IS NOT EMPTY	IS EMPTY	COMPARE TO FIELD
		equal		<Type & tab for single value>			
Add Cancel Insert Remove							

Task

Primary Information

CUSTOM FORM * Standard Task Form **TITLE *** My task today **ASSIGNED TO *** manager, store

PRIORITY * Medium **NOTIFY ASSIGNEE BY EMAIL** **STATUS *** Not Started **PRIVATE TASK**

INSERT BEFORE

Date and Time

START DATE * 16.9.2009 **RESERVE TIME** **REMINDER TYPE** None

DUE DATE * 16.9.2009 **START TIME *** 5:00 pm **END TIME *** 6:00 pm **REMINDER** None

DATE COMPLETED

Message **Related Records** **Availability** **Communication** **Time Tracking** **System Information** **Custom**

NEWCRMFIELD **SUBORDINATES**

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

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 [Dependent Dropdown Lists](#).



Note: Filtering applies only to lists of records. It does **not** apply to custom lists.

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

1. Click the **Sourcing & Filtering** subtab.
2. In the **Filter Using** field, select a field to filter on.

The field you select 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 show only sales representatives, you select the **Sales Rep** field in the Filter Using column. Sales Rep is the name of the field on the employee record being filtered.



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

3. If the field selected 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 **Is Checked** column.

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

For example, select **equal** 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 records only with the state listed as Georgia.

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

7. To include all records with a value entered in your filter field, check the **Is Not Empty** box.

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

8. To include all records with no value entered in your filter field, check the **IsEmpty** box.

9. In the **Compare To Field** column, select 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 can become incompatible. This incompatibility results in unexpected errors when a form containing the field is displayed. If you plan to change a custom field's type, review any existing filters defined for the field, and remove or change these filters to avoid errors. For information about various custom field types, see [Field Type Descriptions for Custom Fields](#).

Dependent Dropdown Lists

You can use a custom field of type List/Record to create dependent dropdown lists. Dependent dropdown lists enable you to filter the choices in a list based on the selections in one or more other fields. For example, you can add two custom transaction line fields to a transaction form and another field that is filtered based on the selections in the other two.

To understand the content in this topic, you should be familiar with creating custom lists, custom record types, and custom transaction types. For more information, see the following help topics:

- [Creating a Custom List](#)
- [Creating Custom Record Types](#)
- [Creating a Custom Transaction Type](#)

To set up field dependencies, you use the Compare To Field column on the Sourcing & Filtering subtab on the custom field page. Use the Compare To Field column to compare values entered in multiple fields on a transaction type or other record type with the values defined on a custom record.

The following screenshot of a Transaction Line Field page for Shirt Style shows the Compare To Field setup.

The screenshot shows the 'Sourcing & Filtering' subtab of a Transaction Line Field page for 'Shirt Style'. The 'SOURCE LIST' dropdown is set to 'Shirt Color'. The 'SOURCE FROM' dropdown is set to 'Shirt Size'. The 'COMPARE TO FIELD' column is highlighted with a red box and contains 'Shirt Color' and 'Shirt Size' under the 'IS CHECKED' and 'COMPARE TYPE' columns respectively. The 'COMPARE VALUE TO' column contains 'equal' for both rows. The 'VALUE IS' column is empty. The 'IS NOT EMPTY' and 'IS EMPTY' columns are also empty. At the bottom, there are buttons for 'Add', 'Cancel', 'Insert', and 'Remove'.

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

For example, you have a Shirt Style transaction line field that filters according to the selections in the Shirt Color and Shirt Size transaction line fields. Shirt Color and Shirt Size are the controlling fields. Shirt Style is the dependent field. You create the Shirt Style field on the custom record type.

For an example that illustrates the use of multiple dependent dropdown lists, see [Creating Dependent Dropdown Lists](#).

Creating Dependent Dropdown Lists

The following procedures provide high-level steps for creating dependent dropdown lists. To understand the process, you should be familiar with creating custom lists, custom record types, and custom transaction types.

For a detailed example, see [Dependent Dropdown Lists Example](#).

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

1. [Creating the Custom Lists and Custom Record Type](#)
2. [Creating Custom Fields and Setting up Filtering](#)

Creating the Custom Lists and Custom Record Type

Before you can create fields that filter based on selections you make in other fields, you first must create a custom list for each controlling field. Then you create the custom record type for the dependent field. You use these custom lists and custom record when you create the fields to include in transaction types or other record types.

To create the custom lists and custom record type:

1. Create a custom list for each controlling field.
2. Create a custom record type whose field provides the options to be available in the dependent field. The custom record type must contain List/Record fields that reference the custom lists you created for the controlling fields in the preceding step.
3. Create a custom record instance for each option you want to be available in the dependent field.

Creating Custom Fields and Setting up Filtering

Next you use the custom lists and custom record created in the preceding procedure to create and add custom fields to the transaction type or other record type. You configure the filtering options on the dependent field. The controlling fields determine the options available in the dependent field.

To create custom fields and set up filtering:

1. Create a custom field of the List/Record type for each controlling field.
2. For the dependent field, create the same type of List/Record custom field that you created for the controlling fields in the preceding step.
In the **List/Record** field, select the custom record type you created in Step 2 of [Creating the Custom Lists and Custom Record Type](#).
3. Click the **Sourcing & Filtering** subtab. Complete the following:
 - a. 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 type that refers to the field you selected in the **Filter Using** column.
 - d. Repeat these steps for each controlling field.
 - e. Click **Save**.

You can create multiple layers of dependencies. In the Shirt Style example, you could have another field that is filtered based on your selection in the Shirt Style field. You must create a custom record type for each additional dependent field in the same way you created the Shirt Style custom record type.

Dependent Dropdown Lists Example

You can create dependent dropdown lists that enable you to filter the choices in a list based on the selections in one or more other fields. The topics in this section describe the steps of creating multiple dependent dropdown lists for a company that sells shirts in various sizes, colors, and styles.

To understand the content in this topic, you should be familiar with creating custom lists, custom record types, and custom transaction types. For more information, see the following help topics:

- [Creating a Custom List](#)
- [Creating Custom Record Types](#)
- [Creating a Custom Transaction Type](#)

A company sells 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.

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

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

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

For the steps to create the dependent dropdown lists for the preceding scenario, see [Steps for Creating Dependent Dropdown Lists](#).

Steps for Creating Dependent Dropdown Lists

This example describes the steps for creating dependent dropdown lists for the shirt style scenario described in [Dependent Dropdown Lists Example](#). For this example, the available shirt style selections depend on what is entered for the shirt size and shirt color.

Complete the following steps:

1. [Creating the Custom Lists and Custom Record Type](#)
2. [Creating Custom Transaction Line Fields and Setting up Filtering](#)

Creating the Custom Lists and Custom Record Type

Before you can create the custom fields, you must first create the custom lists that will be used in those fields. In this example, you need custom lists for Shirt Size and Shirt Color. Then you create the custom record type for Shirt Style. You use these custom lists and the Shirt Style custom record when you create the custom transaction line fields.

[To create the custom lists and custom record type:](#)

1. Create a custom list named **Shirt Color** that includes the colors black and white. Then create a custom list named **Shirt Size** that includes the sizes small and large.
2. Create a custom record type named **Shirt Style** with two fields: **Shirt Size** and **Shirt Color**. For these fields, set the **Type** to **List/Record** and then, in the **List/Record** field, enter the corresponding list from the previous step.

The following screenshot shows the Shirt Style custom record type with the Shirt Color and Shirt Size fields that reference the custom lists:

Fields • Subtabs Sublists Icon • Numbering • Forms • Online Forms Permissions Links Managers Translation • Child Records Parent Records History • System Notes •					
New Field		Move To Top		Move To Bottom	
DESCRIPTION	ID	TYPE	LIST/RECORD	TAB	SHOW IN LIST
Shirt Color	custrecord99	List/Record	Shirt Color		No
Shirt Size	custrecord100	List/Record	Shirt Size		No

3. After you save the Shirt Style custom record type, create a custom record instance for each Shirt Style combination. Name these custom record instances as follows:
- large, black, crew
 - small, black, crew
 - large, white, crew
 - small, white, crew
 - large, white, v-neck
 - small, white, v-neck

The following screenshot shows the shirt style combinations created for the Shirt Style custom record type:

Shirt Style List		List Search Audit Trail	
VIEW	Default	Customize View	New Shirt Style
FILTERS			
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Show Inactives
<input type="button" value="EDIT"/>	<input type="button" value="VIEW"/>	NAME <input type="button" value="▼"/>	QUICK SORT <input type="button" value="▼"/> TOTAL: 6
Edit View		Large, Black, Crew	
Edit View		Large, White, Crew	
Edit View		Large, White, V-neck	
Edit View		Small, Black, V-neck	
Edit View		Small, White, Crew	
Edit View		Small, White, V-neck	

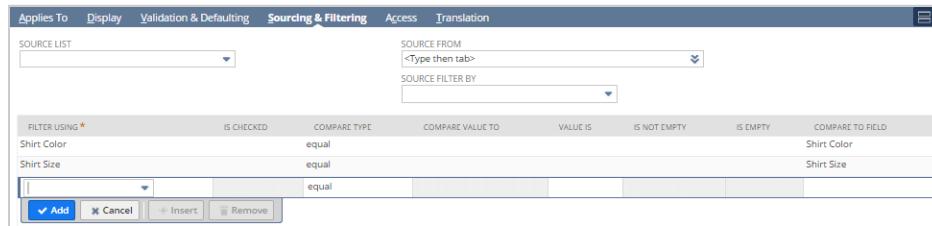
Creating Custom Transaction Line Fields and Setting up Filtering

Next you create the custom fields using the Shirt Color and Shirt Size custom lists and the Shirt Style custom record type. In this example, you need to create three custom transaction line fields: Shirt Color, Shirt Size, and Shirt Style. On the Shirt Style custom field, you configure how the fields are filtered. These custom fields are used on the transaction or record instance.

To create custom transaction line fields and set up filtering:

1. Create a custom transaction line field for each controlling field:
 - **Shirt Color** — Of type List/Record with your shirt color custom list selected in the List/Record field
 - **Shirt Size** — Of type List/Record with your shirt size custom list selected in the List/Record field
2. For the dependent **Shirt Style** field, create the same type of List/Record custom transaction line field that you created for the controlling fields.
In the **List/Record** field, select the **Shirt Style** custom record type.
3. On the **Shirt Style** custom field configuration, click the **Sourcing & Filtering** subtab. Complete the following:
 - a. In the **Filter Using** column, select **Shirt Color**, which represents the field on your custom record that references the **Shirt Color** custom field.

- b. In the **Compare Type** field, select **equal**.
- c. In the **Compare To** field, select **Shirt Color**.
- d. Repeat these steps for the **Shirt Size** field.



- e. Click **Save**.

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



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 the field 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](#).

Watch the following help video for information about access control for custom field values:

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 applicable only to reports and search.)
- **None** – The field cannot be seen, and its contents cannot be changed.

For cases when various 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 remove 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 through 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 in the following steps.
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 in the following steps.
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 for search and reporting.
9. Click **Add**.
10. Repeat these steps for each role, department, or subsidiary.
11. Click **Save**.



Important: The preceding procedure describes how to limit access to a custom field. When available, 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. For more information, 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



Note: SuiteBundler is still supported, but it will not be updated with any new features.

To take advantage of new features for packaging and distributing customizations, you can use the Copy to Account and SuiteCloud Development (SDF) features instead of SuiteBundler.

Copy to Account is an administrator tool that you can use to copy custom objects between your accounts. The tool can copy one custom object at a time, including dependencies and data. For more information, see the help topic [Copy to Account Overview](#).

SuiteCloud Development Framework is a development framework that you can use to create SuiteApps from an integrated development environment (IDE) on your local computer. For more information, see the help topic [SuiteCloud Development Framework Overview](#).

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 because 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 may 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, nonterminated 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. Because any changes you make to the field are not saved, setting the field to read-only prevents you 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 [Custom Field Types](#) to determine the type of field you want to create. You will 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 must 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 enter translations only 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 Field Label

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

Custom Item Field

LABEL *	Export to OpenAir	MATRIX OPTION	<input checked="" type="checkbox"/> SHOW IN LIST
ID	custitemexport_to_openair	TYPE	Check Box
OWNER	Wolfe, J	LIST/RECORD	<input type="checkbox"/> GLOBAL SEARCH <input type="checkbox"/> RECORD IS PARENT
DESCRIPTION	<input checked="" type="checkbox"/> STORE VALUE <input type="checkbox"/> USE ENCRYPTED FORMAT		
<ul style="list-style-type: none"> Applies To Display Validation & Defaulting Sourcing & Filtering Access Translation History 			
LANGUAGE	LABEL	HELP	
Bosnian	<input type="text"/>	<input type="text"/>	
Czech	<input type="text"/>	<input type="text"/>	
English (International)	<input type="text"/>	<input type="text"/>	
Italian	Esporta in OpenAir	<input type="text"/>	

Translating Custom Field Help

You can define translations for the Field Level Help of a custom field on the Translation subtab 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>`.

Custom Item Field

LABEL *	Export to OpenAir	MATRIX OPTION	<input checked="" type="checkbox"/> SHOW IN LIST
ID	custitemexport_to_openair	TYPE	Check Box
OWNER	Wolfe, J	LIST/RECORD	<input type="checkbox"/> GLOBAL SEARCH <input type="checkbox"/> RECORD IS PARENT
DESCRIPTION	<input checked="" type="checkbox"/> STORE VALUE <input type="checkbox"/> USE ENCRYPTED FORMAT		
<ul style="list-style-type: none"> Applies To Display Validation & Defaulting Sourcing & Filtering Access Translation History 			
LANGUAGE	LABEL	HELP	
Bosnian	<input type="text"/>	<input type="text"/>	
Czech	<input type="text"/>	<input type="text"/>	
English (International)	<input type="text"/>	<input type="text"/>	
Italian	Esporta in OpenAir	<input type="text"/> <div style="border: 1px solid black; padding: 5px; width: 150px; height: 100px; margin-top: 5px;"> Fare clic per esportare il record OpenAir. </div>	

Adding Custom Fields to Transaction Forms

After you create a custom transaction field, the field automatically appears on the Custom subtab of standard and custom transaction forms. If you did not specify a subtab for the custom field to appear on, it appears on the Custom subtab. You can then configure if the custom field should or should not be shown on a specific custom transaction form.

To show a custom transaction field on a custom transaction 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. If you did not specify a subtab on the custom field record, the custom field appears on the Custom subtab.
5. If you want the field to show when the transaction is being entered, check the box in the **Show** column next to your custom field. Newly-created custom transaction fields automatically have this box checked.
6. If you use basic printing, click the **Printing Fields** subtab.
 - a. 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**.
 - b. If you want the field to show up when you print or email the form, check the box in the **Print/Email** column next to your custom field.
 - c. Enter a label for each field as you want them to print.
7. Click **Save**.

Your custom transaction field now shows up on your custom transaction form.

To use a specific custom transaction form for your transaction, select the appropriate form in the Custom Form field on the transaction page.

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

The History subtab on a custom field setup page has History and System Notes subtabs. Information on these subtabs provide historical and system notes details about changes made to the custom field.

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

- Date and time the change was made
- User who made the change
- Field label change
- Field type change
- Record type change
- Store value setting change

- Notes related to the change

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

- Date and time the change was made
- User who made the change (Set By)
- How the change was made (Context)
- Type of change
- Field that was changed
- Old field value
- New field value



Note: History and system notes currently are not supported for custom segments.

Inactivating a Custom Field

If required, you can remove a custom field from a specific record type or delete the field completely. Alternatively, you can make a custom field inactive instead of deleting it. Data and associated forms for an inactive custom field are maintained in NetSuite. Maintenance in NetSuite 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.



Warning: 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 deactivate 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.

When a custom field is inactive, the field no longer appears on any forms, is not available in searches, and 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. The **Show Inactives** box is also available for the list of custom fields on each custom record type's Fields subtab. For more information, see the help topic [Adding Fields to Custom Record Types](#). By default, the **Show Inactives** box is cleared, 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** box next to it. You can check the box and submit to make a custom field inactive. You can clear the box and submit 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 verify if custom fields are referenced in SuiteScripts. If you deactivate a custom field, update any SuiteScripts that reference the field, otherwise they will not work.

You cannot make a custom field inactive if other NetSuite records depend on it. For example, you cannot make a custom field 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 preceding examples, there are other dependencies that can prevent you from making a custom field inactive.

When you try to deactivate a custom field that has dependencies, you'll receive an error message. The message contains a link to a Dependent Records page, where you can review details about these dependencies.

Before you try to make custom fields inactive, you can use the Check Inactivate Dependencies option under the Actions menu on each custom field record to check for dependencies. Another option is available to check dependencies that can 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.

You cannot delete or deactivate the custom field if it is used for criteria in duplicate detection. If you try to do so, you will receive an error message. For more information about duplicate detection, see the help topic [Setting Up Duplicate Detection](#).



Note: You cannot make script custom fields 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



Warning: If you edit a custom field ID, the field is removed from any saved searches that it is used in. Before you change a custom field ID, make a list of any saved searches that reference the field, then update the searches with the new ID. For more information, see [Maintaining Saved Searches that Include Edited Custom Fields](#).

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](#)
- [Exposing Internal IDs and Field IDs](#)
- [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)
- custentity (entities)
- custitem (items)
- custcol (transaction column)

- custbody (transaction body)

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
 - Whether the record selected is the parent record. For more information, see the help topic [Parent-Child Record Relationships](#).
 - Whether to make the custom entity field available for SuiteSignOn user identification. For more information, see the help topic [Using Custom Fields as SuiteSignOn User Identification](#).
3. To make other changes to the custom field, select the appropriate subtab:
 - Applies To — Assign the custom field to specific record types.
 - 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.
 - Translation — Translate the label and field-level help for the field.

For more information about the Custom Field page and subtabs, see [Creating a Custom Field](#).
4. Make necessary edits. Then 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](#).

Also, see the following topics:

- [Converting the Field Type of a Custom Field](#)
- [Account-Specific Domains in Custom Fields](#)

Maintaining Saved Searches that Include Edited Custom Fields

If there are saved searches in your account that reference custom fields and you change the custom field ID, the field is removed from the search. To maintain your search results, make a list of any searches that include the custom field before you edit the ID. Then, after you edit the ID, complete the following procedure to update the search.

To update a saved search that includes an edited custom field:

1. Log out of NetSuite and log back in to clear the search cache.
2. Locate and edit the affected searches to add the field with the new ID wherever the field with the old ID was used.
For more information, see the help topic [Editing or Deleting a Saved Search](#).
3. If the field with the old ID was used as part of a custom formula field, you must also update the formula definition to reference the field with the new ID
For more information, see the help topic [Formulas in Search](#).
4. Save the search.
5. Run the search to verify the results.

Converting the Field Type of a Custom Field

NetSuite supports field type conversions for some custom field types. After you have created a custom field, you can edit the type definition to change the Type field to another supported field type. Whenever possible, you should avoid changing a field type. For more information, see [Creating a Custom Field](#).



Note: If you try to convert a field type that is not supported, you will lose data. For a list of supported field types, see [Supported Field Type Conversions](#).



Warning: Custom field filters are type sensitive. If you change the type of a custom field, filters defined for the field can become incompatible. This incompatibility results in unexpected errors when a form containing the field is displayed. If you plan to change a custom field's type, review any existing filters defined for the field and remove or change these filters to avoid errors. For information about field filtering, see the help topic [Setting Filtering Criteria](#).

You cannot change the custom field type if the Type is used for criteria in duplicate detection. For more information about duplicate detection, see the help topic [Setting Up Duplicate Detection](#).

Before you convert the field type of a custom field, confirm that you need to change the field type. You may not need to convert the field type if:

- You use the field only to store data but do not use it anywhere else in NetSuite. In this case, you can have two parallel fields of data, meaning the original field and the new field.
- You use the field only as criteria to search for a specific value. In this case, you have an alternate option to create the new field and add a condition to the field in the Criteria subtab of a search. The results display a row for the original field and a row for the new field.

When large numbers of records are affected, converting a field type can take a significant amount of time to run. During the time that the conversion is running, the system is unusable. Therefore, you should consider running the conversion during off-peak hours. Before running the conversion, you should also verify the number of affected records to decide if you want to continue with the field type conversion. For information, see [Verifying the Amount of Data for Conversion](#).

When large numbers of records are affected, a timeout can occur during the conversion process. If a timeout occurs, all data changes revert to the state they were in prior to the conversion. Field customizations made prior to clicking Save are lost, but the data itself remains in its reverted state. You can try to run the conversion again during off-peak hours.

Alternately, you can convert the field types by performing a mass update or running a mass update script.

For more information, see [Converting Field Types for Custom Fields Using Mass Update \(SuiteAnswers article: 98734\)](#).

See the following Topics:

- [Converting Field Types Through a Bundle or SDF](#)
- [Verifying the Amount of Data for Conversion](#)
- [Supported Field Type Conversions](#)

Converting Field Types Through a Bundle or SDF



Note: SuiteBundler is still supported, but it will not be updated with any new features.

To take advantage of new features for packaging and distributing customizations, you can use the Copy to Account and SuiteCloud Development (SDF) features instead of SuiteBundler.

Copy to Account is an administrator tool that you can use to copy custom objects between your accounts. The tool can copy one custom object at a time, including dependencies and data. For more information, see the help topic [Copy to Account Overview](#).

SuiteCloud Development Framework is a development framework that you can use to create SuiteApps from an integrated development environment (IDE) on your local computer. For more information, see the help topic [SuiteCloud Development Framework Overview](#).

If a field type conversion occurs through a managed bundle or SDF, you cannot control when the field type is changed. For bundles or SDF applications installed manually, you can decide when to update the bundle or SDF applications. For more information, [SuiteBundler Overview](#) and [SuiteCloud Development Framework](#).



Warning: If you change a field type on the target account for custom fields installed by a bundle or SDF, ensure the type conversion is backward compatible. If the type is not backward compatible, you can lose data when the field is updated to the original field type by the bundle or SDF.

Verifying the Amount of Data for Conversion

Large amounts of data can potentially cause a timeout that can result in lost field customizations. Before running a field type conversion, you should verify the number of affected records included in the conversion.

Some ways you can verify the number of affected records include:

- Review field usage. Depending on what the field is used for, you may know that it does or does not affect large numbers of records.
- Create a saved search or workbook of the data with a condition such as the custom field value is not empty, or the custom checkbox field is not checked. The results can help you determine how many records would be affected by a type conversion. If a field is applied to more than one record type, such as Employee and Contact, you must complete two searches or queries and combine the totals.

Supported Field Type Conversions

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.

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 displays 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.

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 displays when you make a change.

The following table lists supported field type conversions.

From Field Type	To Field Type	Notes
<ul style="list-style-type: none"> ■ Decimal Number ■ Percent ■ Currency 	Integer Number	Value rounded to 0 decimal numbers.
<ul style="list-style-type: none"> ■ Decimal Number ■ Percent 	Currency	Value rounded to 2 decimal numbers.
Integer	<ul style="list-style-type: none"> ■ Decimal ■ Currency ■ Percent 	Converted without change.
Currency	<ul style="list-style-type: none"> ■ Decimal ■ Currency ■ Percent 	Converted without change.
Decimal	Percent	Converted without change.
Percent	Decimal	Converted without change.
<ul style="list-style-type: none"> ■ Check Box ■ Currency ■ Date ■ Date/Time ■ Decimal Number ■ Email Address ■ Free-Form Text ■ Help ■ Hyperlink 	<ul style="list-style-type: none"> ■ Free-Form Text ■ Text Area 	<p>Check Box is converted as follows:</p> <ul style="list-style-type: none"> ■ Checked is converted to T ■ Cleared is converted to F

From Field Type	To Field Type	Notes
<ul style="list-style-type: none"> ▪ Inline HTML ▪ Integer Number ▪ Password ▪ Percent ▪ Phone Number ▪ Text Area ▪ Time of Day 		
Select	Multiple Select	List/Record type field remains unchanged. For example, you cannot change from Account to Customer.
<ul style="list-style-type: none"> ▪ Text ▪ Text Area 	<ul style="list-style-type: none"> ▪ Rich Text ▪ Long Text 	Converted without change.
Date	Date/Time	Date/Time is the date and time in the user's current time zone and is converted.

For information about the available field types in NetSuite, see [Field Type Descriptions for Custom Fields](#).

Account-Specific Domains in Custom Fields

You should use URLs that include account-specific domains in all custom fields.

In the past, NetSuite domains contained an identifier of the data center where your account was hosted. Links in custom fields used these data center-specific domains. NetSuite domains no longer include data center identifiers. Instead, the domain identifies your account, not to the data center where your account is hosted. These are called account-specific domains.

Currently, a data center-specific domain in the URL will automatically be modified to an account-specific domain in the NetSuite center. This modification happens for hyperlink, rich text, long text, free-form text, text area, help, and inline HTML custom field types.

The automatic modification from data center-specific URLs to account-specific domain URLs is applied to your active session. Configuration of the URL does not change. Links that use account-specific domains are faster than links modified in the current session. You should update all links in custom fields to use your account-specific domain.

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:

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

The screenshot shows the 'Custom Entity Field' configuration screen. At the top, there are buttons for 'Save', 'Cancel', 'Reset', 'Change ID', and 'Apply to Forms'. The 'Apply to Forms' button is highlighted with a red box. Below these are input fields for 'LABEL *' (containing 'Partner Email') and 'DESCRIPTION'. To the right of the fields are three checkboxes: 'SHOW IN LIST', 'GLOBAL SEARCH', and 'RECORD IS PARENT'. At the bottom right of the screen are navigation icons for 'List', 'Back', and 'Forward'.

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

FORM NAME	FORM SUBTYPE	SHOW	DISPLAY TYPE	FIELD TYPE
				LABEL
Customer Form - Automation	Customer/Lead/Prospect	<input checked="" type="checkbox"/>	Normal	partner name
Custom Customer Form	Customer/Lead/Prospect	<input checked="" type="checkbox"/>	Normal	partner name
Custom Customer Form_2	Customer/Lead/Prospect	<input checked="" type="checkbox"/>	Normal	Partner Email
Custom Pop-up Customer Form	Customer/Lead/Prospect	<input type="checkbox"/>	Normal	Partner Email
Suite Dreams Customer Form	Customer/Lead/Prospect	<input checked="" type="checkbox"/>	Normal	Partner Email
Custom Customer Form 2	Customer/Lead/Prospect	<input checked="" type="checkbox"/>	Normal	Partner Email

If no forms are listed, the Display Type field may be set to Hidden. Hidden fields are not shown on any form.



Note: Custom transaction body and custom transaction line fields are included on all forms, even if they are not displayed. If the Display Type field is set to Hidden, the field is still included on all forms for those two field types. 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.

- 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

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. Use the Advanced PDF/HTML Templates option only 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, the image is shown on the form when you view it. You can interact with the image in two ways:

- When you click the image thumbnail, you are directed to its file in the File Cabinet.
- When you hover over the image, the Open button appears. When you click the Open button, a pop-up window shows a full-size preview of the image.



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 encryption format option is always disabled by default.

You can encrypt the following text types:

- Email Address

- 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 API is not supported for fields with encrypted stored values. Other SuiteScript APIs that rely on search may not be supported.
- Encryption of stored field values increases their size and can 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.

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

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 the values of a custom field.



Note: Note the following

- The rollup functionality is similar to that of custom sublists, except the search results display in a field rather than a sublist.
- To ensure a smoother checkout process for your customers, avoid adding too many values to custom fields.

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

- [Steps for Creating a Summary Search Custom Field](#)
- [Custom Field Types](#)
- [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

You can create a custom field with a value derived from a summary search.

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 more information, 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 more information, 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 place the custom field on a form for a record type that is different from the summary search record type. For example, you may want to place a custom entity field showing the result of a customer record summary search on an employee custom form. To do this, 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, because values for rollup custom fields are not stored.

Custom Fields that Support Summary Search Derived Values

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

- 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
- 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. The value of the field is 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.

The screenshot shows the 'Transaction Body Field' configuration screen. The 'Validation & Defaulting' tab is selected. In the 'SEARCH' dropdown under the 'DEFAULT VALUE' section, the value 'Total Purchase Order Amount' is selected, which is highlighted with a red border. The 'FIELD' dropdown below it is empty. Other tabs visible include 'Applies To', 'Display', 'Sourcing & Filtering', 'Access', and 'Translation'. On the right side of the screen, there are several checkboxes for field properties: 'STORE VALUE', 'USE ENCRYPTED FORMAT', 'SHOW IN LIST', 'GLOBAL SEARCH', and 'RECORD IS PARENT'. The 'OWNER' field is set to 'JL.Wolfe'.

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 can vary for users with various permissions. Summary search results are rolled up for the records to which the current user has access. Because users with various permissions can have access to different sets of records, the calculated value of the field can 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.



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.

See the following topics:

- [NetSuite Tags](#)
- [Dynamic Hyperlinks](#)
- [Setting the Store Value Field](#)
- [Setting the Formula Field](#)
- [Predefined Formula Tags](#)

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](#).

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. These tags work only when the Formula box **is not** checked:

- **{useremail}** - Email of the user currently logged in
- **{now}** - Current date
- **{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 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, in the Default Value field enter the http address as usual followed by **?=** and the required 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
- 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 not be valid.

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. Before checking the Formula box, ensure that you are entering a formula. 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 preceding example, 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.

Predefined Formula Tags

The following tags can be used in formulas:

- Date - Current date and time:
 - `{TODAY}`
 - `{NOW}`
- User ID:
 - `{ME}`
 - `{USER}`
 - `{USER.ID}`
- User role:
 - `{USERROLE}`
 - `{USERROLE.ID}`
- User department:
 - `{USER.DEPARTMENT}`
 - `{USERDEPARTMENT.ID}`
- User location:
 - `{USER.LOCATION}`
 - `{USERLOCATION.ID}`
- User subsidiary:

- {USER.SUBSIDIARY}
- {USERSUBSIDIARY.ID}
- User class:
 - {USER.CLASS}
 - {USERCLASS.ID}

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.

The screenshot shows the 'Validation & Defaulting' subtab of a custom field configuration. At the top, there are tabs for 'Applies To', 'Display', 'Validation & Defaulting' (which is selected), 'Sourcing & Filtering', 'Access', and 'Translation'. Under 'Validation & Defaulting', there are checkboxes for 'MANDATORY' and 'CHECK SPELLING'. Below these are sections for 'MAXIMUM LENGTH' (with a text input field) and 'FIELD' (with a dropdown menu). On the right, there is a 'DEFAULT VALUE' section containing the formula '{amount}-{creditlimit}', a 'SEARCH' section with a dropdown menu, and a 'FORMULA' checkbox which is checked. There is also a small icon with a question mark and a '2' inside a circle.

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 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.
 - In the case of a Field Not Found error, ensure that the appropriate access level is defined for the user or role in the Default **Level for Search/Reporting** list and permissions table. These settings are located on the Access subtab of the custom field record.
- 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, you create a custom entity field to apply to customer records. You can add a formula field that references 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:

Applies To	Display	Validation & Defaulting	Sourcing & Filtering	Access	Translation
<input type="checkbox"/> MANDATORY <input type="checkbox"/> CHECK SPELLING MAXIMUM LENGTH <input type="text"/>		DEFAULT VALUE <input type="text" value="{}partner.email{}"/> <input checked="" type="checkbox"/> FORMULA	SEARCH <input type="text"/>	FIELD <input type="text"/>	

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 following example displays the email address on the record for the partner assigned to each customer.

Customer

3 ABC Co.

Actions: Save Cancel Reset Merge Actions

Primary Information

CUSTOM FORM *	COMPANY NAME *	CATEGORY
Suite Dreams Customer Form	ABC Co.	
CUSTOMER ID *	PARENT COMPANY	DEFAULT ORDER PRIORITY
3		
NAME	STATUS *	COMMENTS
ABC Co.	CUSTOMER-Closed Won	
TYPE	WEB ADDRESS	
<input checked="" type="radio"/> COMPANY		
<input type="radio"/> INDIVIDUAL		

Email | Phone | Address

EMAIL	ALT. PHONE	ADDRESS
smartin@abc.com		
PHONE *	FAX	
212-555-9065		

Classification

SUBSIDIARY	LAST SALES ACTIVITY
Parent Company	
<input type="checkbox"/> EXPORT TO OPENAIR	

Sales **Marketing** **Support** **Financial** **Preferences** **System Info**

TERRITORY	PARTNER EMAIL
	jason@xyzresellers.com



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.

The screenshot shows the 'Custom Entity Field' setup screen. The 'Validation & Defaulting' tab is active. In the 'DEFAULT VALUE' field, the formula '{partner.email}' is entered. The 'FORMULA' checkbox is checked. Other tabs like 'Display', 'Sourcing & Filtering', and 'Access' are also visible.



Note: For more details, refer to [SQL Expressions](#). Also, you can refer to the [Working with the SuiteScript Records Browser](#) for tables of NetSuite field IDs.

For information about field types in formulas, see [Formulas with Various Field Types](#).

Formula Field Example

Suppose, for example, 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. To do so, create a custom field that uses the {linenumber} formula, and apply the field to transaction forms.

To create a custom field that uses the {linenumber} formula:

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. 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. This setup ensures 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.

Formulas with Various Field Types

The following sections describe using various field types in formulas. For information about formula fields, see [Creating Formula Fields](#).

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, can have values for both a memo body field and memo line item fields. Transaction line memo fields are set when transactions are saved. The **memo** field returns both 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 value from the first line item memo field that is not empty.



Note: The **memobody** field is a placeholder that can be used to return the memo body field value, even if it is blank. However, memobody can be used only in a related record, for example, `{salesorder.memobody}`. The **memobody** field cannot be accessed through SuiteScript.

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 of fewer than 1000 values, edit the list as needed. For lists of more than 1000 values, click **Manage Values** on the **Values** subtab 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 Show Inactives 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](#).

For information about running saved searches on custom lists, see [Saved Searches for Custom Sublists](#).

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:

1. Go to Customization > Lists, Records, & Fields > Lists > New. The Custom List page appears.
2. In the **Name** field, enter a name for the list.
3. In the ID field, enter a unique alphanumeric ID for the custom list. For information about best practices and naming conventions, see [Conventions for Naming Custom Objects](#). For information about changing an existing ID, see [Changing the ID of a Custom Object](#).
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 can 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 the **alphabetical order** radio button.
7. If the list is for matrix items, check the **Matrix Option List** box.

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 Option List** box is **not** displayed.

8. In the **Value** field, enter a value for the list.
9. Click **Add**.
10. Add values to the list as needed.
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 should include no more than 1000 values.

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 enter translations only 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 List

Save ▾ Cancel Reset Change ID Actions ▾

NAME * Number

ID customlist40

INTERNAL ID 40

OWNER JL Wolfe

DESCRIPTION

SHOW OPTIONS IN: THE ORDER ENTERED ALPHABETICAL ORDER

MATRIX OPTION LIST
 CONVERT TO CUSTOM RECORD
 INACTIVE

Values	Translation	History	System Notes
LANGUAGE ▲	NAME		
Czech			
German			
Hebrew			
Spanish	Numero		

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:

Custom List

Save ▾ Cancel Reset | Change ID | Actions ▾

NAME *	<input type="text" value="Number"/> SHOW OPTIONS IN: <input checked="" type="radio"/> THE ORDER ENTERED <input type="radio"/> ALPHABETICAL ORDER																																										
ID	<input type="checkbox"/> MATRIX OPTION LIST																																										
INTERNAL ID	<input type="checkbox"/> CONVERT TO CUSTOM RECORD																																										
40	<input type="checkbox"/> INACTIVE																																										
OWNER	<input type="text" value="JL Wolfe"/>																																										
DESCRIPTION	<input type="text"/>																																										
<u>Values</u> • Translation • History • System Notes																																											
<table border="1"> <thead> <tr> <th>VALUE *</th> <th>TRANSLATION</th> <th>ID</th> <th>ABBREVIATION *</th> <th>INACTIVE</th> </tr> </thead> <tbody> <tr> <td>Fifty</td> <td></td> <td>1</td> <td>2</td> <td></td> </tr> <tr> <td>Czech</td> <td></td> <td>own</td> <td></td> <td><input type="button" value="Move To Top"/></td> </tr> <tr> <td>German</td> <td></td> <td></td> <td></td> <td><input type="button" value="Move To Bottom"/></td> </tr> <tr> <td>Hebrew</td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>Spanish</td> <td>Cincuenta</td> <td></td> <td></td> <td></td> </tr> <tr> <td>Done</td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>+ Add Row</td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>				VALUE *	TRANSLATION	ID	ABBREVIATION *	INACTIVE	Fifty		1	2		Czech		own		<input type="button" value="Move To Top"/>	German				<input type="button" value="Move To Bottom"/>	Hebrew					Spanish	Cincuenta				Done					+ Add Row				
VALUE *	TRANSLATION	ID	ABBREVIATION *	INACTIVE																																							
Fifty		1	2																																								
Czech		own		<input type="button" value="Move To Top"/>																																							
German				<input type="button" value="Move To Bottom"/>																																							
Hebrew																																											
Spanish	Cincuenta																																										
Done																																											
+ Add Row																																											



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.

Custom List

Save ▾ Cancel Reset | Change ID | Actions ▾

NAME *	<input type="text" value="Size List"/> SHOW OPTIONS IN: <input checked="" type="radio"/> THE ORDER ENTERED <input type="radio"/> ALPHABETICAL ORDER		
ID	<input type="checkbox"/> MATRIX OPTION LIST		
customlist_listsizes1	<input type="checkbox"/> CONVERT TO CUSTOM RECORD		
OWNER	<input type="checkbox"/> INACTIVE		
Default User			
DESCRIPTION	<input type="text" value="Complete list of size numbers"/>		
<u>Values</u> • Translation • History • System Notes •			
<input type="button" value="Manage Values"/>			

On the Custom List page, the values can be viewed only in alphabetical order. Use the page controls to view, edit, add, or delete any values as required. If your organization permits 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.

Edit View		NAME ▲
Edit View		10007
Edit View		10008
Edit View		10009
Edit View		1001
Edit View		10010
Edit View		10011
Edit View		10012
Edit View		10013
Edit View		10014
Edit View		10015
Edit View		10016
Edit View		10017
Edit View		10018
Edit View		10019
Edit View		1002

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 the following:

- Form preferences are controlled by settings on the custom forms page as well as settings defined for each role.
- The Manage Translations feature enables you to manage your language translations using translation collections. You should edit translations for custom forms using a translation collection because it is more efficient than making the edits on the Translation subtabs in the user interface. For more information, see the help topic [Editing a Translation Collection](#).

Translations used in custom forms are primarily taken from custom field and custom segment definitions. You can override a field label on the custom form definition page, but it is often not the best approach. When you translate a field label on the Translation subtab of a custom field, the translated label overrides the language set up in Home > Set Preferences. Language changes will be visible only to users with the same language preference. For example, if you change the language from English (US) to English (International), only users with the English (International) language preference will see the changes. For more information about translations, see the help topic [Configuring Multiple Languages](#).

- If you use SDF and the Multi-Language feature is enabled, you can enter translations in forms in SDF XML. You should define translations using a translation collection. For more information, see the help topic [Translation Collections as XML Definitions](#).

If you modify a form and enter a translation string directly in the form, the translations are not connected to terms or translations collections and are visible only to users with the same language preference. If you change your language preference, then you must also define the translations for that language.

- 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 access the NetSuite UI only by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic [SuiteScript 2.x Custom Pages](#).

Custom forms are supported in SuiteCloud Development Framework (SDF). SDF is a development framework that you can use to create SDF SuiteApps, or to customize NetSuite accounts, using an integrated development environment (IDE) on your local computer. SuiteCloud projects are file-based and use XML definitions of custom NetSuite objects. For more information, see the help topic [SDF Custom Object and File Development in SuiteCloud Projects](#).

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 required, add custom fields, or apply custom code.

For detailed information about 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 or preferred form for specific user roles. For more information, see [Defining Preferred Forms](#) and [Setting Default Forms for Roles](#).

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 required, add custom fields, or apply custom code.

For detailed information about 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 or preferred form for specific user roles. For more information, see [Defining Preferred Forms](#) and [Setting Default Forms for Roles](#).

You can also link transaction forms together to create transaction workflows. For more information, see [Linking Transaction Forms](#).

Important Note about New Fields and Custom Forms

Be aware that as part of a new release, new fields are sometimes added to NetSuite standard record types. Newly added fields for a record type are automatically added to any forms that you have customized for this record type. These fields are set to show on customized forms. If you do not want new fields to show on any of your customized forms, after your upgrade you need to modify these forms to remove or hide these fields. Keep in mind that newly created fields are typically coupled with forms and can be needed for some forms to function properly. You should test any customized forms where you remove or hide fields, to ensure that the forms still function properly.

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.x 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.

For information about form templates for customizing entry and transaction forms, see [Form Templates](#).

Form Templates

You can customize your transaction and entry forms using basic layouts or advanced PDF/HTML templates. For more information, see the following:

- Advanced PDF/HTML Templates
- Basic Layouts for Transaction Forms

For information about custom entry and custom transaction forms, see [Custom 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 various 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: In a future release, basic layouts will no longer be supported. 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 checking boxes in the Preferred column at Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and clicking Submit, and clicking Submit.

For more details, see [Customizing Transaction Form PDF Layouts](#) and [Transaction Form HTML Layouts](#).

Creating Custom Entry and Transaction Forms

You can create your own custom entry and transaction forms by starting with an existing standard form and customizing it.

To create a custom entry or transaction form:

1. Do one of the following to select the required form to customize.
 - To customize an entry form, go to Customization > Forms > Entry Forms. Click **Customize** or **Edit** next to a form in the Custom Entry Forms list.
 - To customize a transaction form, go to Customization > Forms > Transaction Forms. Click **Customize** or **Edit** next to a form in the Custom Transaction Forms list.
 - If available, in view mode of a custom form, click the **Customize** link in the upper right, and then click **Customize Form**.



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. In the **ID** field, enter a unique alphanumeric ID for the custom form. For information about best practices and naming conventions, see [Conventions for Naming Custom Objects](#). For information about changing an existing ID, see [Changing the ID of a Custom Object](#).



Note: You cannot specify an ID for a transaction form when you create it, but you can change the ID after the form has been created.

4. Set the custom form properties. Options vary depending on the type of form being customized. See the following topics:
 - [Custom Entry Form Properties](#)
 - [Custom Transaction Forms Properties](#)
5. Click **Save**.

For information about how to see what a completed form looks like, see [Viewing a Completed Form](#).

You can use SuiteCloud Development Framework (SDF) to manage custom entry and transaction forms as part of file-based customization projects. For more 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. 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 Copy to Account option in the upper-right corner. For more information about Copy to Account, see the help topic [Copy to Account Overview](#).

For more information about customizing Entry forms, see [Custom Entry Form Properties](#).

For more information about customizing Transaction forms, see [Custom Transaction Forms Properties](#).

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. For more information, see [Configuring Subtabs for Custom Entry and Transaction Forms](#).



Important: As you configure your custom form, consider whether the transactions to use the form require tax data. Only a form that includes required tax-related fields can be used for a transaction with tax consequences. You cannot specify tax field names through form customization. You must go to Setup > Accounting > Set Up Taxes. For more information, 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 can be altered if you later disable this feature. Immediately after you disable Project Management, you should review custom project forms to see if they have been changed, and if necessary, edit them to fit your requirements. For more information, see the help topics [Enabling Project Features](#) and [Using Project Management](#).

Viewing a Completed Form

To see what a completed form looks like, create an entry or transaction that uses the custom form. For example, click the tab where you access transactions, and click the kind of transaction you want to create. From the Custom Entry or Custom Transaction page, in the Custom Form field, select the custom form to use.

To see a list of all transactions that use the custom form, click List in the upper right of the Custom Entry or Custom Transaction page. Then, to view a specific transaction instance, click View beside the transaction you want to view.

For more information, see the help topic [Working with Transactions](#).

Custom Entry Form Properties

Note that the options available on your form vary, depending on the type of entry form being customized.

- **Form is Preferred** – Check to make the form your default form when entering transactions of this type. Only one form can be defined as the preferred form per transaction type. Checking this box

clears any previously defined preferred forms of the same transaction type and replaces it with your new preferred form. For details on how preferred forms are defined, see [Defining Preferred Forms](#).

Note the following about marking an entry or transaction form as 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 nononline order form is marked as preferred for the Customer Center, it is saved as the form for the order. Even if an online form is marked as preferred, it is not saved as the form for an order. Instead, the preferred nononline order form is used.

Employee Center roles have limited access to forms. Only one form is ever made available to this role, and the form set on the Role record takes precedence over the preferred form.

- **Store Form with Record** – Check to store this custom form with each record created using this form. When viewed or edited, any record that was entered with this form is displayed using this form rather than your preferred form.
 - For custom entry forms, the Store Form with Record box is cleared by default.
 - The Store Form with Record box is available only for a subset of transaction forms. For this subset, the box is checked by default. This box is not available for other transaction forms because custom forms are automatically 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 sees the custom form, even if that form is not enabled for their role.

- **Print Template** – This field is 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.
- **Enable Field Editing on Lists** – Check to permit inline editing on this form. Inline editing lets users 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. This option 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. Checking this box clears any previously defined popup forms of the same transaction type and replaces it with your new popup form.



Note: The Use for Pop-ups box is automatically checked for the custom form of a custom record, if there is no previous form already enabled.

- **Popup Only** – When **Use for Pop-ups** is checked, you can 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.

Custom Transaction Forms Properties

Note that the options available on your form vary, depending on the type of transaction form being customized.

- **Form is Preferred** – Check to make the form your default form when entering transactions of this type. Only one form can be defined as the preferred form per transaction type. Checking this box

clears any previously defined preferred forms of the same transaction type and replaces it with your new preferred form. For details on how preferred forms are defined, see [Defining Preferred Forms](#).

Note the following about marking an entry or transaction form as 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 nononline order form is marked as preferred for the Customer Center, it is saved as the form for the order. Even if an online form is marked as preferred, it is not saved as the form for an order. Instead, the preferred nononline order form is used.

Employee Center roles have limited access to forms. Only one form is ever made available to this role, and the form set on the Role record takes precedence over the preferred form.

- **Store Form with Record** – Check to store this custom form with each record created using this form. When viewed or edited, any record that was entered with this form is displayed using this form rather than your preferred form.
 - For custom entry forms, the Store Form with Record box is cleared by default.
 - The Store Form with Record box is available only for a subset of transaction forms. For this subset, the box is checked by default. This box is not available for other transaction forms because custom forms are automatically 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 sees the custom form, even if that form is not enabled for their role.

- **Print Template** – This field is 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.
- **Allow Add Multiples** – Check to permit the **Add Multiple** button to appear on transaction item lists. You should clear the box on any forms that rely on custom code line item validation scripts. The **Add Multiple** button is displayed on Items lists and lets you add multiple items at a time to the item list. However, when items are added with the **Add Multiple** button, any Validate Line custom code events defined for the form are **not** triggered.
- **Printing Type** – This field is available only when the Advanced PDF/HTML Templates feature is enabled. The Advanced option that is selected by default enables your custom form to use an advanced PDF/HTML template. For more details, see [Advanced PDF/HTML Templates](#). Select **Basic** to enable your custom form to use transaction form PDF layouts and HTML layouts.
- **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 use for email attachments when transactions associated with this form are sent by email.
- **Email Message Template** – (Available only for transactions that support custom email templates.) Select which template to use for the message body for this transaction type when NetSuite sends email messages with a PDF attachment. If you select Default Email Template, NetSuite will send transaction email messages using a hard-coded template and body message that you can set up at Setup > Company > Email > Email Preferences (Administrator). For more information, see the help topic [Assigning an Email Template to a Transaction Type](#).
- **PDF Layouts** – Select a PDF layout for your form.
- **HTML Layouts** – Select an HTML layout for your form. Standard and Classic layouts exist for all the standard form types other than shipping labels.
- **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, set the Print Remittance Form with Invoices & Statements feature. For more information, see the help topic [Printing Remittance Forms](#).

- **Disclaimer** – Enter message to appear at the bottom of your form. 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 Width** – This is the total width of all the columns on your form. If the total width of your columns is greater than the available column space, NetSuite adjusts the width proportions to fit on the page.
 - **Layout Space** – This number is the maximum number of inches of printable space permitted 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.

Storing Custom Forms with Transactions

Store Form with Record is an option available for custom entry and some custom transaction forms. The option indicates whether a reference to the form should be stored with each record created by it. When enabled, you see the referenced form, rather than your preferred form, when you view or edit a related record. 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 box is not available to administrators when you are configuring your custom form.

The following table lists custom transaction form types and whether users have the choice of changing the Store Form with Record option.

Custom Transaction Form Type	Can Choose Not to Store Form with Record?
Assembly Build	Yes
Assembly Unbuild	Yes
Bill of Materials	No, form always stored with record
Bill Payment	No, form always stored with record
Cash Refund	No, form always stored with record
Cash Sale	No, form always stored with record
Check	Yes
Credit Card Charge	Yes
Credit Memo	No, form always stored with record
Customer Deposit	No, form always stored with record

Custom Transaction Form Type	Can Choose Not to Store Form with Record?
Customer Refund	Yes
Deposit	Yes
Estimate (Quote)	No, form always stored with record
Expense Report	Yes
Inventory Adjustment	Yes
Inventory Cost Revaluation	Yes
Inventory Worksheet	Yes
Invoice	No, form always stored with record
Item Fulfillment	No, form always stored with record
Item Receipt	No, form always stored with record
Journal Entry	Yes
Opportunity	Yes
Packing Slip	No, form always stored with record
Payment	No, form always stored with record
Picking Ticket	No, form always stored with record
Price List	No, form always stored with record
Purchase Order	No, form always stored with record
Remittance Slip	No, form always stored with record
Return Authorization	No, form always stored with record
Return Form	No, form always stored with record
Sales Order	No, form always stored with record
Shipping Label	No, form always stored with record
Statement	No, form always stored with record
Transfer Inventory	No, form always stored with record
Transfer Order	Yes
Vendor Bill	Yes
Vendor Credit	Yes
Vendor Return Authorization	Yes
Work Order	Yes



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 let users associate vendors with multiple subsidiaries. You can 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.



Note: On saved search results, you can perform inline edits on a field only if the subtab for the field is set to display on your preferred form. Field editing on lists must also be set on your preferred form. For information about setting up field editing on lists for a form, see [Creating Custom Entry and Transaction Forms](#)

If field editing is set up but saved search results will not permit you to edit inline, ensure the subtab for that field is set to display on your preferred form. Alternately, move the field to an alternate subtab that is set to display on the form. For more information, see [Moving Fields and Lists Between Subtabs](#).

2. In the **Label** column, edit the headings for the subtabs as needed.
3. Rearrange the order of subtabs as needed. To rearrange the subtabs, click the required line and then drag it to the preferred position, or click **Move to Top** or **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.



Note: The Move Form Elements page enables you to move elements between subtabs, and a few specific sublists. The available sublists are dependent on the type of form you edit. You can view a full list of available sublists for each transaction type in the SuiteScript Records Browser. For more information, see the help topic [SuiteScript Records Browser](#).

To move fields and lists between subtabs:

1. Click **Save & Move Elements**. Your form is saved, and the Move Form Elements page opens. The **Fields** subtab lists fields that appear on each subtab of the customized form that include fields. Subtabs with lists (the **Contacts** subtab on entry forms or the **Items** subtab on transactions, for example) are shown on the **Lists** subtab of the Move Form Elements page.
2. For each field on each subtab, select the subtab where you want that field to appear. Note that you can select multiple fields and move them to the same subtab by using the same Select and Set Subtab dropdown list. At any time, you can click **Save and Move More** to save your field changes, and reload the page with the fields moved to the specified subtab.
3. Click the **Lists** subtab.
4. Select the subtab where you want each list to appear. It is possible to move a field or list to a subtab that is **not** displayed because it does not contain items of that type. When a field or list is moved to a previously 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**.

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

Use the Field Groups subtab to customize the field groups that appear on your forms. Field groups support body fields only. You cannot create a field group for sublist fields.

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. The order of field groups on the Field Groups subtab determines the order of field groups on your form.

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.

Be aware that adding custom fields to field groups can cause field text to be displayed in a way you may not expect based on the display size attributes set for custom fields. For more information, see [Setting Display Options for Custom Fields](#).

To configure field groups:

1. Click the **Field Groups** subtab.
2. To customize field groups that appear in the main body area of a page, click **Main**. To customize the field groups that appear on other subtabs on the form, click another field group subtab.

LABEL *	SHOW	SHOW FIELD GROUP TITLE	SINGLE COLUMN
Primary Information	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Email Phone Address	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Classification	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

3. In the **Label** column, edit field group headings.
4. In the **Show** column, clear a field group to hide it on the form.
5. In the **Single Column** column, to have the column display vertically rather than horizontally, check a field group. The following screenshot provides an example of the Primary Information field group displayed vertically.

The screenshot shows a SuiteBuilder custom form configuration for a 'Customer' entry. The form is titled '5 Star Day'. It includes the following fields:

- CUSTOM FORM: Custom Customer Form
- CUSTOMER ID: 5 Star Day (with AUTO checkbox checked)
- TYPE: COMPANY (radio button selected)
- COMPANY NAME: 5 Star Day
- PARENT COMPANY: (dropdown menu)
- STATUS: CUSTOMER-Closed Won
- SALES REP: (dropdown menu)
- PARTNER: (dropdown menu)
- WEB ADDRESS: (text input)
- CATEGORY: (dropdown menu)
- DEFAULT ORDER PRIORITY: (dropdown menu)
- COMMENTS: (text area)

The 'Primary Information' field group is highlighted, illustrating the vertical column display when the Single Column option is selected.

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

6. To delete a field group, click the **X** icon. The fields display at the bottom of the page. However, the fields are not assigned to any field group because that field group has been deleted.
7. After you have configured your field groups, you can click the **Fields** subtab (for Entry forms) or the **Screen Fields** subtab (for Transaction forms) to configure the fields that appear in each field group. For more information, see [Configuring Fields or Screens](#).

Field Groups on Custom Forms of Custom Records

You can add field groups to new and existing custom record custom forms.

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 form for the custom record.
- Note:** If you do not yet have a custom form, click **Customize** next to the record type to create a new custom form.
5. Click the **Field Groups** subtab.
 6. Enter 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 **Screen Fields** subtab (for Transaction forms) to configure the fields that appear in each field group. For more information, see [Configuring Fields or Screens](#).

Configuring Fields or Screens

When the following subtabs are available on the current entry or transaction form, you can configure fields for that form:

- **Screen Fields** subtab on transaction forms
- **Fields** subtab on entry forms
- **Sublist Fields** subtab on transaction forms

These fields can also be moved to display on an alternate subtab or in alternate field groups.



Note: Even if you have disabled a subtab as described in [Configuring Subtabs for Custom Entry and Transaction Forms](#), you can still configure fields for that subtab. This is useful if you later decide to include the subtab on your form.

To configure fields in the Screen Fields, Fields, or Sublist Fields subtab:

1. In the **Show** column, check the boxes next to the fields you want to display.



Note: When configuring fields, be aware of the following:

- Hiding fields on a form can hide other related fields. For example, hiding **Credit Card Approved** on the sales order form also hides fields related to **Address Verification System (AVS)**.
- 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 hidden sublist field, the template can fail to print. For more information, see [Using FreeMarker to Work with Hidden Fields Used in Advanced Templates](#).
- If you have the Termination Reason Tracking feature enabled, you cannot save an 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 has both **Quick Add** and the **Enable Field Editing on Lists** checked, then that field can be directly edited, and the record is saved when you click 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](#).

You can add Quick Add portlets to your dashboards. Quick Add portlets permit the quick addition of a selected record type without navigating 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 that are required on your entry form.



Note: 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, 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** — Use the default value set in the field definition.
 - **Checked** — Set the default to checked, possibly overriding the default set in the field definition.
 - **Unchecked** — Set the default to cleared, possibly overriding the default set in the field definition.
6. In the **Label** field, edit the name of any of the fields as needed. 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. For more information, see the help topic [Configuring Multiple Languages](#).
7. In the **Field Groups** list, select the group where you want the field to appear.



Note: After you assign a field to a field group, the field can be moved only within the group. If you want to move the field elsewhere on a form, you must change its field group.

8. In the **Column Break** column, check any field to insert a column break after that field. Be aware that inserting a column break after a field can cause field text to display in a way you may not expect based on the display size attributes set for custom fields. For more information, see [Setting Display Options for Custom Fields](#).
9. To include a blank line before a field, enter the number of blank lines in the **Space Before** column.
10. To associate a field with the field immediately above it, check the **Same Row as Previous** box. An associated field shares the same **Show** or **Hide** setting as the previous field. The field belongs to the same field group as the previous field and is displayed together on the form with the previous field. For more information, see [Associating Related Fields on Custom Forms](#).
11. Rearrange the fields as needed.
 - To move each field to the preferred subtab, click **Move Elements Between Subtabs**. For more information, see [Moving Fields and Lists Between Subtabs](#).
 - Rearrange the order of the fields on each subtab. Select and drag each line item to the preferred position or click **Move to Top** or **Move to Bottom**.



Note: Fields that belong to a field group can be moved only 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 **Check Box Default** column, set the default value for check box fields. If you set the **Check Box Default** field to **Checked** for a transaction form field, the box for that field will automatically be checked. You can clear the box if needed.
 - Click the **Sublist Fields** and arrange the line-item columns for your transaction entry form by moving fields as needed. 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 to use to filter the items that appear in the Items list on this form. For more information, see the help topic [Defining a Saved Search](#).
 14. On the **Total Box** subtab, specify which fields to show in the form totals.
If you use SuiteTax, additional tax fields are available. For more information, see the help topic [Tax Details on Transactions in SuiteTax](#).
 15. Click **Save**.
 16. If required, to create any new custom fields, click **New Field**.



Note: Ensure that you save any changes before creating new fields from within the form. When you click **New Field**, you leave the custom form to go to the custom field page. For more information, see [Custom Fields](#).

17. For transaction forms only, after you have configured the fields or screens, you should configure the printing fields. For more information, see [Configuring Printing Fields](#).



Important: As you configure your custom form, consider whether the transactions to use on the form require tax data. You cannot specify tax field names through form customization. You must go to Setup > Accounting > Set Up Taxes. For more details, 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 **Items** list on transactions. The items in the **Items** dropdown list will meet the criteria of the saved item search. For more information, see the help topic [Defining a Saved Search](#).

To filter the Items dropdown list on transactions:

1. Create a saved items search that filters the items list.
2. Customize the transaction form where you want the filtered items list to appear.
3. On the Custom Transaction Form page, click the **Sublist Fields** subtab.
4. In the **Item Filter** list, select the appropriate 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, 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.

The screenshot shows the 'Custom Entry Form' configuration screen. The 'Fields' subtab is selected. A red box highlights a group of fields: 'Campaign ID', 'Auto Name', 'Title', 'Category', 'Owner', and 'Owner Role'. Each of these highlighted fields has the 'SAME ROW AS PREVIOUS' checkbox checked, indicating they are associated with the previous field in the list.

DESCRIPTION	SHOW	MANDATORY	DISPLAY TYPE	LABEL	FIELD GROUP	COLUMN BREAK	SPACE BEFORE	SAME ROW AS PREVIOUS
Custom Form	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Normal	Custo	Primary Information	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Campaign ID	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Normal	ID	Primary Information	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Auto Name	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Normal	Auto	Primary Information	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Title	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Normal	Title	Primary Information	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Category	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Normal	Categ	Primary Information	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Owner	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Normal	Mana	Primary Information	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Owner Role	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Normal	Mana	Primary Information	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Start Date	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Normal	Start I	Primary Information	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
End Date	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Normal	End D	Primary Information	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Base Cost	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Normal	Base I	Primary Information	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Setting custom fields to be Same Row as Previous on custom forms can cause field text to display in a way you may not expect based on the display size attributes set for custom fields. For more information, see [Setting Display Options for Custom Fields](#).

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 needed.

Configuring Buttons and Actions

On the Actions subtab of custom forms, you can configure standard NetSuite buttons and custom buttons.

You can use the Standard Actions subtab to customize built-in, standard NetSuite buttons. For more information, see [Working with Standard Buttons](#). Use the Custom Actions subtab to create or modify custom buttons you have associated with client SuiteScripts. For more information, see [Working with Custom Buttons](#).

Also note that you can use point-and-click customization to translate the button labels for both standard and custom buttons.



Note: You can also use SuiteScript to hide and rename buttons. For a list of standard buttons that are supported in SuiteScript, see the help topic [Button IDs](#). 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: SuiteScript and point-and-click customization do not support customizing the Save, Edit, Cancel, Back, and Reset buttons.

After you have configured your buttons, you should configure the printing fields. For more information, see [Configuring Printing Fields](#).

See the following topics:

- [Working with Standard Buttons](#)
- [Working with Custom Buttons](#)
- [About Button and Action Layout](#)

Working with Standard Buttons



[Customizing Buttons in Custom Forms](#)

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

To customize standard buttons:

1. On the **Actions** subtab of the Custom Forms page, click the **Standard Actions** subtab.
2. In the **Label** field, enter a UI label for the button.
3. In the **Show** column, to display the button on the form, check the box. Buttons are shown by default.
4. In the **Display As** column, to display the button as an inline button, select **Button**. To have the button appear as an action on the **More Actions** menu or as a menu item in a button group, select **Menu**. For more information, see [About Button and Action Layout](#).
5. Click **Save**.

Working with Custom Buttons

You can add custom buttons to forms to initiate client SuiteScript. For example, you could add Create Invoice button on a customer form that performs a specific function when the button is clicked. The Custom Actions subtab is visible when the Client SuiteScript feature is enabled in your account.



Important: 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 Code** subtab of the form, in the **Script File** field, add the client SuiteScript. For more information, see [Associating Custom Code \(Client SuiteScript\) Files With Custom Forms](#).
The primary object used to encapsulate custom buttons is `serverWidget.Button`. For more information, see the help topic [serverWidget.Button](#).
2. In the **Label** field, enter the UI label for the button. You can enter up to 99 characters.
3. In the **Function** field, enter the name of the function to perform 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, select **Button** to display the button as an inline button. Select **Menu** to have the button to appear as an action in the **More Actions** menu. For more information, see [About Button and Action Layout](#).
5. Click **Save**.

About Button and Action Layout

On the Action subtab of a custom entry form, use the **Display As** list to customize the layout of a button or action.

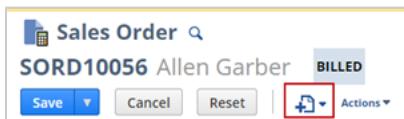
ACTION NAME ▲	LABEL	SHOW	DISPLAY AS
Delete	Delete	<input checked="" type="checkbox"/>	Menu ▾
Make Copy	Make Copy	<input checked="" type="checkbox"/>	Menu ▾
New	New	<input checked="" type="checkbox"/>	Menu ▾
New Event Field	New Event Field	<input checked="" type="checkbox"/>	Button ▾
Print	Print	<input checked="" type="checkbox"/>	Menu ▾
Save & Copy	Save & Copy	<input checked="" type="checkbox"/>	Menu ▾
Save & New	Save & New	<input checked="" type="checkbox"/>	Button ▾
			Menu

On 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.

For certain buttons such as the **Save &** buttons (for example, Save & New and Save & Print), display as **Menu** means that they display as an item in a button group. For these buttons, **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 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 (highlighted in the following image). 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.



Configuring Printing Fields

Configuration of Printing Fields is required for 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 represent the various areas of a printed form.

To configure how each section of your printed form displays:

- For all subtabs on the Printing Fields subtab:

- In the **Print/Email** column, check the boxes next to fields you 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.

- In the **Label** column, edit the labels of the fields as needed.

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.

- For the **Body** and **Columns** subtabs:

- 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.

- Rearrange the fields as needed. Select and drag each line item to the preferred position.

- Click **Save**.

Note: Any configurations made on the Printing Fields subtab are visible only on the printed form.

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 appear 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). For more information, see the help topic [Customizing Tax Fields on Transaction Forms](#).

For fields on the **Header** subtab, some fields represent **values** that are inserted and some represent **labels** of field data that is inserted. For the value 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
- Ship To

After you have configured the printing fields, you should configure sublists. For more information, see [Configuring Sublists](#).

Configuring Sublists

Sublist configuration applies to entry forms only. 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.
2. In the **Never Empty** column, to specify that a sublist entry is required, check the box. If this box is checked, at least one row must be entered for the sublist.

The **Never Empty** box is supported only for sublists that are listed in the SuiteScript Records Browser. The following sublists do not appear on the Sublists subtab because they require at least one row by default.

- Items
- Lines
- Components

3. In the **Label** column, enter headings as needed.
4. Rearrange the sublists as needed.
 - Click **Move Elements Between Subtabs**. On the **Lists** subtab, move each sublist to the preferred subtab. For more information, see [Moving Fields and Lists Between Subtabs](#).
 - To rearrange the sublists on the **Sublists** subtab, select and drag each line item to the preferred position. Or click **Move to Top** or **Move to Bottom**.
5. Click **Save**.

After you have configured the sublists, you should configure sublist fields. For more information, see [Configuring Sublist Fields](#).

Configuring Sublist Fields

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

You can use the **Item Filter** on this subtab to select the saved search to determine which items are shown in the Items list on the transaction form. For more information, see the help topic [Filtering the Items Dropdown List on Transactions](#).

Note: If you use the Item Filter on this page, only saved searches marked as Public appear in the list. For more information about creating a saved search, see the help topic [Defining a Saved Search](#).

To configure sublist fields:

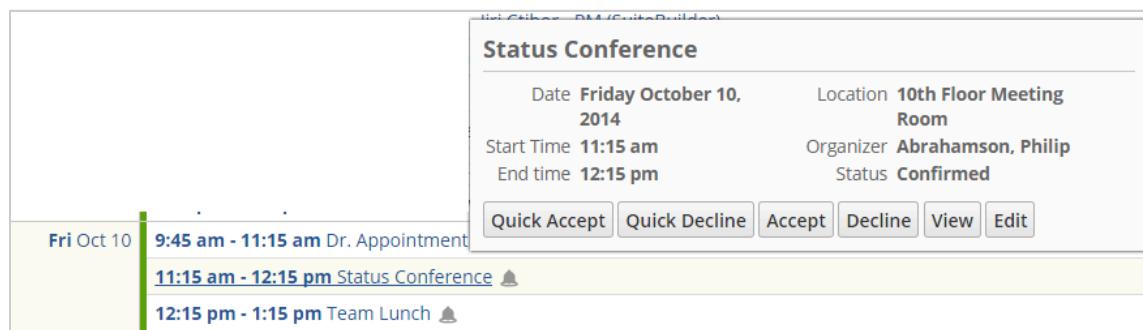
1. In the **Show** column, check the boxes for the sublist fields to display on the form.
2. In the **Label** column enter labels as needed.

Note: 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.

3. Rearrange the fields as needed. To rearrange the fields, select and drag each line item to the preferred position, or click **Move to Top** or **Move to Bottom**.
4. Click **Save**.

Configuring QuickViews

For both entry and transaction forms you use the QuickViews subtab to customize which fields appear in the QuickView for that transaction or record. 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. You can add a maximum of 20 fields to a QuickView.

Note that you can also configure QuickViews on forms associated with custom records. For more information, see [Configuring QuickViews for Custom Records](#) for information.

Configuring QuickViews for Upgraded Forms

QuickView customization work best 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:

1. Open the Custom Form or Custom Transaction form you want to edit. For more information, see the help topic [Creating Custom Entry and Transaction Forms](#).
2. On the Edit Custom Entry [or Transaction] Form page, 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. For more information, see [Deploying Upgraded Forms](#)
3. In the **Field Description** column, add the fields you want to appear in the QuickView for records that use this form. After you select the field, click **Add**. To remove a field, click the X icon on the right side of the field.
4. Click **Save**.

Note 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** subtab, you must save the form customization page before those updates appear on the **QuickView** subtab.
- On the **Fields** subtab, 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** subtab.
- The fields that appear on the QuickView tab match the fields listed in the **Description** column on the **Fields** subtab. However, on the QuickView subtab the custom label for the field displays.
- The list of fields available on the **QuickView** subtab 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. Fields that appear on a record **only** when the record is in Edit mode do not appear in QuickViews. If you add a field to a QuickView, yet the field does not appear when the QuickView displays, you may have added a field that is visible only when the record is in Edit mode.

See also: [Configuring QuickViews for Nonupgraded Forms](#).

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 performing the following steps.

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** subtab.
4. Click **Customize** next to the standard version of the custom record form. Or, 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, click **Add**. To remove a field, click the X icon that appears to the right of the field.
7. Click **Save**.

On the list page for this custom record type, you can hover over a record to see the QuickView.

Configuring QuickViews for Nonupgraded 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. Nonupgraded 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 nonupgraded version of the form show the QuickView customizations of the upgraded form.

Also note the following when attempting to customize QuickViews for nonupgraded forms:

- A nonupgraded 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 nonupgraded 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 nonupgraded and upgraded forms show the NetSuite default QuickView fields for that record type.

See also: [Configuring QuickViews for Upgraded Forms](#).

How to Tell if the Custom Forms in Your Account are Upgraded

Records that use upgraded forms have all fields organized into field groups. For an example of what a field group looks like, see [Configuring Field Groups](#).

If you have noticed any records in your account with a field group layout, then at least 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]. Click **Upgrade Checklist** in the far-right corner. The page shows specifically which forms have been upgraded and that their upgrades have been **Deployed** into NetSuite. Forms that have not been upgraded or deployed have **Skip Upgrade | Deploy Form** links.

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

On 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 (and any built-in NetSuite actions for that form type) are performed.

You can also add a custom button to the form to call the client script. For more information, see [Working with Custom Buttons](#).

To associate client SuiteScript files with a form, click the plus sign to the right of the Script File field, and add a new script file.

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, cannot access the Custom Code tab of custom forms.

Users must have at least the Edit level of the SuiteScript permission to attach a script to a custom form on the Custom Code subtab. 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 subtab but cannot edit it. Users who do not have SuiteScript permission cannot view the Custom Code subtab.

To associate client SuiteScript files with a form:

1. Go to the Custom Code subtab of the form record.
2. In the **Script File** field, select the client SuiteScript file that contains the scripts you want to add to this form. To add a new script file, click the New icon on the right side of the **Script File** field.
3. In one or more of the client event type fields, enter the names of the functions you want to perform. 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.
 - **Page Init Function** — Executing function to be called when this form is first loaded.
 - **Save Record Function** — Executing function to be called when this record is saved.
 - **Validate Field Function** — Executing function to be called when a field on this entry form is changed.
 - **Field Changed Function** — Executing function to be called when a change made to a field is accepted.
 - **Post Sourcing Function** — 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.
 - **Line Init Function** — Function that runs on Line Init events. These events occur when an existing line is selected.
 - **Validate Line Function** — Function that runs on Validate Line events. These events occurs 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.

- **Validate Insert Function** — 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.
 - **Validate Delete Function** — 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.
 - **Recalc Function** — 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.
4. If your client SuiteScript file references any functions in a library file, add the library files in the **Library Script File** list.
 5. 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 you can define form preferences.

- **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 whose role form preferences are **not** defined. On the Roles tab, you can also define the form as preferred for specific roles.
- **On the Custom Forms list page:** On the custom forms list page at Customization > Forms > [Entry Forms or Transaction Forms], you can check the Preferred box for any form you want to set as the default form for all users whose role form preferences are **not** defined.
- **On the Manage Roles page:** 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 select 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.

i Note: In some cases with transactions, the form you choose is selected automatically the next time you create a transaction of that type during your current NetSuite session. For example, if you go to the Invoice Customers page, you'll see the Form list set to the form you used the last time you invoiced customers. When you 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. To define new settings for one of these roles, click **Customize** next to the role, and enter a name for your custom role. Then set the needed custom settings and 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 preferred form for each Customer subtype in the corresponding fields at the top of the Entity subtab — Preferred Lead Form, Preferred Prospect Form and Preferred Subcustomer Form.
 - **CRM** — Set defaults for phone call, campaign, case, event, solution and task entry forms.
 - **Item** — Set defaults for inventory part, noninventory 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 **Enabled** column, for any forms you **do not** want this role to have access to, clear the boxes. This is not available for Customer Center roles.
 - In the **Preferred** column, for any form you want to be set as the preferred form for this role, check the box.
 - If you want this form to be the only form available to this role, check the box in the **Restricted** column. This is 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**.

You can specify the entry or transaction form to use as the preferred form for a role. For more information, see [Defining Preferred Entry and Transaction Forms for Roles](#)

Defining Preferred Entry and Transaction Forms for Roles

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 nononline 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 nononline 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
- Provide space for an approval signature

To add a disclaimer to standard transaction forms:

- Go to Setup > Company > Preferences > Printing & Fax.
- Click the **Printing** subtab.
- 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.
 - Remittance Slip Message** – Remittance slips.
- 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 about customizing transaction forms, read [Custom Forms](#).

Specifying Check Layout by Subsidiary

If your organization uses basic printing, you can configure varying check templates for each subsidiary.

To specify check layout by subsidiary in basic printing:

- Go to Setup > Company > Classifications > Subsidiaries.
- Click **Edit** beside the subsidiary you want to edit.

3. Click the **Preferences** subtab.
4. 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](#).
5. In the **Check Default Chart Type** field, select the template to use when printing checks from this subsidiary.
6. 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.

Adding Page Numbers to your PDF Transaction Forms

You can add page numbers to pages of your PDF transaction forms.

To customize your multiple page transaction forms to have page numbers:

1. Go to Customization > Forms > Transaction Forms.
2. Click **Customize** next to the kind of form you want to change.
3. Click the **Printing Fields** subtab.
4. Click the **Header** subtab.
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. Click **Save**.

Your printed PDF transaction forms now include page numbers for multiple pages.

Adding More Header Information to Multiple Page Transaction Forms

You can add more header information to multiple page transaction forms..

To customize your multiple page transaction forms to have more header information:

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. 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 specify which transaction form is used when you transform 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 transform a transaction you created with a custom transaction form, the custom form set on the Linked Forms subtab is selected by default. In the preceding example, 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 custom entry form or custom transaction form you want to create the linkage from. For more information, see the help topic [Creating Custom Entry and Transaction Forms](#).
2. Click the **Linked Forms** subtab.
A list of the transactions that you can transform to from the type of transaction form you are customizing is shown. For example, if you are customizing an estimate form, you can link forms for cash sales, invoices, and sales orders.
3. In the **Custom Form** column, select the standard or custom form for each transaction.
If you want to use the preferred form for the transaction type, do not specify a linked form for that transaction type.
4. Click **Save**.

When the transaction is transformed, the correct transaction form is used automatically.

 **Note:** If 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 affect 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 affect 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**.

i Note: To use 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 affect printed transactions are on the Transactions subtab of this page.

- Print Using HTML – If you prefer to print transaction forms in HTML rather than the default of PDF, check this box.
- Transaction Email Attachment Form – If you prefer to send email attachments of transactions in PDF rather than HTML, select **PDF**.
- Horizontal Print Offset – To move text in the printed transaction form to the right, enter a positive number in inches. To move text to the left, enter a negative number. This setting applies only to basic layouts.
- Vertical Print Offset – To move text in the printed transaction form lower, enter a positive number in inches. To move text higher, enter a negative number. This setting applies only to basic layouts.

The PDF preferences on the SuiteAnalytics subtab apply only 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 may 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: 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 > New.
2. In the **Script ID** field, enter a unique alphanumeric ID for the custom subtab. For more information about best practices and naming conventions, see [Conventions for Naming Custom Objects](#). For information about changing an existing ID, see [Changing the ID of a Custom Object](#).
3. From the **Type** list, select the type of record for which you want to create the subtab.
4. If applicable, from the **Parent** list, select the parent subtab.
5. In the **Title** field, enter a name for the subtab.
6. Click **Save**.

Now, when you create a custom field, you can select your new subtab in the **Subtab** field. After you assign the custom field to the subtab, the subtab then displays on the record.

You can also create a new subtab by clicking New on the Custom Subtabs list page, or you can create new subtabs from the Custom Subtabs page. For more information, see [Creating a Subtab from the Custom Subtabs Page](#).

Creating a Subtab from the Custom Subtabs Page

You can add new subtabs directly from the Custom Subtabs page, as well as provide translation details for the subtab. This method of creating custom subtabs enables you to quickly create multiple subtabs for a specific record type. You cannot, however, specify your own script ID from this location. NetSuite automatically generates the ID for you. You can later edit this system-generated ID using the Change ID button available on the Custom Subtab page for the subtab. For more information, see [Changing the ID of a Custom Object](#).

To create a custom subtab from the Custom Subtabs page:

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, noninventory, 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. In the **Title** field, enter a name for the sublist.
 4. If required, in the **Translation** field, enter translations for this subtab.
 - 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 be used for website translations only (and are not available at Home > Set Preferences). You should enter translations only 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 required, make the subtab a child of an existing parent subtab. From the **Parent** list, select a parent subtab.
 - The parent list includes 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 a child subtab, you cannot select it as a parent for another custom subtab. In other words, you **cannot** create a child-grandchild relationship.
 6. Click **Add**.
 7. If necessary, 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.
 8. Click **Save**.

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.

Managing Custom Subtabs

You can move an existing field or list from one subtab to another. You can also move a field from a subtab to the main section of a form. For more information, see [Moving Fields and Lists Between Subtabs](#) and [Configuring Fields or Screens](#).

You can rename a custom subtab at any time in one of the following ways:

- From the Custom Subtab list, click **Edit** beside the applicable subtab. In the **Title** field, enter the new subtab name, and click **Save**.
- On the Custom Subtabs page, click the appropriate subtab (Transaction, Entity, Item, CRM). Select the applicable subtab and in the **Title** field, enter the new subtab name. Then click **Save**.

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 a form-level script to custom note forms.

See the following sections:

- [Creating a Custom Note Form](#)
- [Creating a Custom Field for use in Custom Note Forms](#)
- [Using Custom Note Forms](#)

Creating a Custom Note Form

You can create a custom note form that can be used to add notes to a record.

To add a custom note form:

1. Go to Customization > Forms > Entry Forms.
2. Click **Customize** beside the Standard Note Form.
3. On the Custom Entry Form edit page, make changes as needed. For more information, see the help topic [Creating Custom Entry and Transaction Forms](#).
4. Click **Save**.

Creating a Custom Field for use in Custom Note Forms

You can create a custom field that can be displayed on custom note forms. 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:

1. Go to Customization > Lists, Records, & Fields > Other Custom Fields> New.
2. In the **Record Type** field, select **Note**.
3. Complete the remaining fields, as needed. For more information, see the help topic [Creating Other Record Fields](#).
4. Click **Save**.

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 popup windows. 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 popup window. You cannot remove this list from the standard note form if any custom note form exists.

The screenshot shows a 'Note' dialog box. At the top are 'Save' and 'Cancel' buttons. Below them is a 'CUSTOM FORM' dropdown menu with three items: 'Standard Note Form' (selected), 'Custom Note Form', and another 'Standard Note Form'. To the right of the dropdown is a large text area labeled 'MEMO *'. Below the dropdown are four input fields: 'TYPE', 'DIRECTION', 'DATE' (containing '14.10.2014'), and 'TIME' (containing '5:48 pm').

You can set a custom note form to be used by default for the note popup window. 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 permit 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 nonrequired address fields
- Setting address fields to be required
- 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 required, 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 the help topic [Country-Specific Address Forms](#).
6. Click **Save**.

Change to Required Permission for Attaching Scripts to Custom Address Forms

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](#)
- [Using SuiteScript 2.x to Create a New Shipping Address Example](#)
- [Using SuiteScript 2.x to Select an Existing Shipping Address Example](#)
- [Using SuiteScript 2.x to Retrieve a Shipping Address Example](#)
- [Using SuiteScript 2.x to Retrieve one Value from a Shipping Address Example](#)

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 Cusotmization > 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 needed.
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 needed. For details, see the following:
 - [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

This topic has been moved under [Region-Specific SuiteApps](#). For more information, see the help topic [Country-Specific Address Forms](#).

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.

Custom Sublists are supported in SuiteCloud Development Framework (SDF). SDF is a development framework that you can use to create SDF SuiteApps, or to customize NetSuite accounts, using an integrated development environment (IDE) on your local computer. SuiteCloud projects are file-based and use XML definitions of custom NetSuite objects, see the help topic [SDF Custom Object and File Development in SuiteCloud Projects](#).

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 attach a saved search as a sublist only 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 may 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.

FIELD *	SUMMARY TYPE	FUNCTION	FORMULA	WHEN ORDERED BY FIELD
Name	Group			
Email				
Phone				
Office Phone				
Fax				
Primary Contact				
Alt. Email				
sales rep email (Custom)				
Regions (Custom)				
Activity: Date	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.

This example uses Sales Rep as the filter, and assumes that the employee is the sales representative for the records being searched. Filtering by sales representative ensures that sales representatives see

results for their own customers only. If the employee is not the sales representative for the records being searched, the save search will return no results.

The screenshot shows the 'Saved Customer Search' page. At the top, there are buttons for Save, Reset, Cancel, Preview, Pivot Report, Change ID, and Actions. Below these are fields for SEARCH TITLE* (containing 'CC No Recent Activity Customer Search'), APP ID (containing 'customsearch521'), OWNER (containing 'McKinney, Christine'), and PUBLIC (unchecked). On the right, there are checkboxes for 'AVAILABLE AS LIST VIEW', 'AVAILABLE AS DASHBOARD VIEW', 'AVAILABLE AS SUBLIST VIEW', 'AVAILABLE FOR REMINDERS', and 'SHOW IN MENU'. A note below says 'Limit the set of filters available on the form when you reuse this search, or to set filters for the results (such as when used as a list view). Remove all filters to use advanced search.' There are checkboxes for 'MY PREFERRED SEARCH FORM' and 'HIDE FILTER DROPSHDS'. Below this is a 'FILTER' section for 'Sales Rep' with an 'Add' button. At the bottom are Save, Reset, Cancel, Preview, Pivot Report, Change ID, and Actions buttons.

Set up the custom sublist by selecting the saved search and applying it to the required records.

The screenshot shows the 'Custom Sublists' page. At the top, there are buttons for Save and Cancel. Below these are tabs for Transaction, Entity, Item, and CRM. The main area displays a table of available searches, each with a 'LABEL', 'TRANSLATION', 'TAB', and 'FIELD' column. The table includes rows for 'Enhanced Customer/Lead Search Form', 'Item Preferred Vendor - Sublist', 'Custom Transaction Search 3', and 'CC No Recent Activity Customer Search'. Below the table is a 'Move' toolbar with buttons for Add, Cancel, Insert, Remove, Move Up, Move Down, Move To Top, and Move To Bottom. At the bottom are Save and Cancel buttons.

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.

The screenshot shows an employee record with various tabs at the top: Communication, Address, Human Resources, ACH/Direct Deposit, Time Tracking, Payroll, Commission, Related Records, Marketing, Access, System Information, and Custom. The 'Custom' tab is selected. In the 'Custom' tab, there are two sections: 'CUSTENTITY1' and 'CUSTENTITY2'. 'CUSTENTITY1' contains 'MYCOUNTRY USA' and 'WELCOME http://maps.google.com/maps?q=44+Elm+Street%20Athens%20NY'. 'CUSTENTITY2' contains an error message 'ERROR: Invalid Expression' and 'TRAVEL APPROVAL LIMIT (2)'. Below these sections is a 'Travel Request' subtab with tabs for Travel Request, Equipment Service, Travel Request (2), Equipment Service (2), Family Information, AAA, and No Activity. The 'No Activity' tab is selected. Under 'No Activity', there is a table with columns for NAME and MAXIMUM OF DATE. The table lists several entries: '-None-', 'Acme, Inc.', 'Adley Electric Systems', 'Alfonso, Patti', 'Atlanta Braves, Ltd.', 'Balance Test 1', 'Balance Test 1 : Balance Test Sub', 'Brooklyn Dodgers, Inc.', 'Cash, John', 'Contest, Inc.', 'Dallas, Inc.', and 'Dynasty, Ltd.'

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.

The screenshot shows the 'Custom Sublist' configuration dialog. It includes fields for 'SCRIPT ID' (set to '_cc_customer_balance'), 'TAB' (set to 'Accounting'), 'TYPE' (set to 'Entity'), 'SEARCH' (set to 'Customers by Outstanding Balance'), 'LABEL' (set to 'Customers Outstanding Balance'), and 'FIELD' (set to '<Type then tab>'). Under the 'FIELD' dropdown, the 'CUSTOMER' checkbox is checked, while other options like 'JOB', 'VENDOR', 'EMPLOYEE', 'PARTNER', and 'CONTACT' are unchecked.

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, select 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

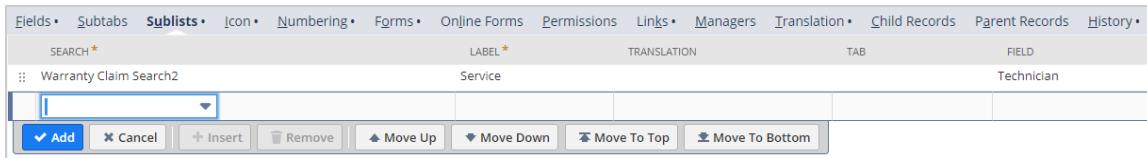
You have a custom record called Warranty Claim, and when a new record is created, you want to show information about previous claims.

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 the warranty claims that were worked by the same technician only.

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

For information about custom child record sublist creation with SuiteScript, see the following topics:

- [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, you should 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 saved to NetSuite.
- when writing server scripts, you must call `Record.commitLine(options)` or `CurrentRecord.commitLine(options)` to save sublist updates. Note that you must do this in addition to calling `Record.save(options)`, which saves 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.

FIXED ASSETS NAME	COST *	USEFUL LIFE IN YEARS *
Fixed asset 1	200.00	10
Fixed asset 2	400.00	2
Fixed asset 3	1000.00	3

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 about creating parent-child relationships between records, see [Parent-Child Record Relationships](#).)

The Fixed Assets fields that appears in the sublist are the required 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.)

The screenshot shows the 'Customer' record for 'Abe Simpson'. The 'Fixed Assets' tab is selected, displaying a sublist of fixed assets. The sublist includes fields for NAME, COST, USEFUL LIFE IN YEARS, and REMOVE. The data in the sublist is:

	NAME	COST	USEFUL LIFE IN YEARS	REMOVE
Edit	2003 Ford Expedition	32,885.00	5	Remove
Edit	Dell Dimension XPS	1,649.00	3	Remove
Edit	Printer	1,000.00	3	Remove

Fields set to **Show in List** on the fixed assets record type appear in the sublist. However, you can update any fixed assets field through 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:

- 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.

The screenshot shows the 'Custom Record Type' page for 'Fixed Assets'. The 'NAME' field is highlighted with a red box. The 'ALLOW CHILD RECORD EDITING' checkbox is also highlighted with a red box. Other visible fields include 'ID', 'INTERNAL ID', 'CUSTOM SEGMENT', 'OWNER', 'DESCRIPTION', and various checkboxes for system settings like 'SHOW LAST MODIFIED', 'ACCESS TYPE', and 'SHOW NOTES'.

DESCRIPTION	ID	TYPE	LIST/RECORD	TAB	SHOW IN LIST
Cost	custrecord1	Currency			No
Salvage Value	custrecord2	Currency			No
Depreciable Cost	custrecord3	Currency			No
Useful Life in Years	custrecord4	Decimal Number			No
Description	custrecord5	Text Area			No

Note: If Allow Child Record Editing is checked, then Allow Delete becomes enabled. You can check the Allow Delete box to enable users to delete records of this type when they appear as child records in a sublist on a parent record.

- 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.
 - Add a custom field to your new custom record type.
 - On the field definition page for the new field, set the field **Type** to **List/Record** (see the following screenshot).
 - Specify the record type that is the parent of your custom record type. In the following screenshot, the customer record type is the parent.
 - 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.

The screenshot shows the 'Fixed Assets Field' configuration in SuiteBuilder. The 'TYPE' dropdown is set to 'List/Record' and 'Customer'. The 'PARENT SUBTAB' dropdown is set to 'Fixed Assets'. A red box highlights the 'RECORD IS PARENT' checkbox in the top right corner.

Your custom child record sublist can appear on a standard or custom subtab of the parent record.

- 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.

- After defining the customer–fixed assets (parent–child) relationship (through 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 screenshot shows the NetSuite Customer Record for Abe Simpson. The main form fields include CUSTOM FORM (Custom Customer Form 4), PARENT COMPANY (<Type then tab>), CATEGORY (From advertisement), and various status and sales rep dropdowns. Below the main form is a section for Email, Phone, and Address. At the bottom, there's a tab bar with Relationships, Communication, Address, Sales, Marketing, Support, Financial, Preferences, System Information, Custom, and Fixed Assets. The Fixed Assets tab is selected. A red box highlights the 'New Fixed Assets' button and the resulting sublist table. The table has columns for EDIT, NAME, COST, USEFUL LIFE IN YEARS, and REMOVE. It lists three items: 2003 Ford Expedition (cost 32,885.00, useful life 5 years), Dell Dimension XPS (cost 1,649.00, useful life 3 years), and Printer (cost 1,000.00, useful life 3 years).

EDIT	NAME	COST	USEFUL LIFE IN YEARS	REMOVE
Edit	2003 Ford Expedition	32,885.00	5	Remove
Edit	Dell Dimension XPS	1,649.00	3	Remove
Edit	Printer	1,000.00	3	Remove

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.

The screenshot shows the NetSuite Customer Record Edit screen for 'Abe Simpson'. The main form includes fields for CUSTOM FORM (Custom Customer Form 4), PARENT COMPANY (dropdown), STATUS (CUSTOMER-Closed Won), SALES REP (Jon Baker), and CATEGORY (From advertisement). Below this is the 'Email | Phone | Address' section with fields for EMAIL (asimpson@netsuite.com), ALT. PHONE (555-1234), and PHONE (504-231-1111). A 'LAST SALES ACTIVITY' section is present. At the bottom, a tab bar is visible with 'Relationships', 'Communication', 'Address', 'Sales', 'Marketing', 'Support', 'Financial', 'Preferences', 'System Information', 'Custom', and 'Fixed Assets' (which is currently selected). Under the 'Fixed Assets' tab, a sublist titled 'Fixed Assets' is displayed with three items: '2003 Ford Expedition' (cost: 32,885.00), 'Dell Dimension XPS' (cost: 1,649.00), and 'Printer' (cost: 1,000.00). Each item has an 'Edit' link and a 'Remove' button.

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\(options\)](#).

```

1 // Add code.
2 ...
3 var sublistFieldValue = objRecord.getSublistValue({
4   sublistId: 'item',
5   fieldId: 'item',
6   line: 3
7 });

```

```

8 | ...
9 | //Add code.

```

Finding the Internal ID of a Custom Child Record Sublist

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. Therefore, the internal ID for the custom child Fixed Assets sublist on customer records is **recmachcustrecord111**.

DESCRIPTION	ID	TYPE	LIST/RECORD	TAB	SHOW IN LIST
Cost	custrecord1	Currency			No
Salvage Value	custrecord2	Currency			No
Depreciable Cost	custrecord3	Currency			No
Useful Life in Years	custrecord4	Decimal Number			No
Description	custrecord5	Text Area			No
New Customer	custrecord111	List/Record	Customer		Yes

Obtaining the Internal ID of the Parent Field

The following is an alternative approach for obtaining the internal ID of the parent field.

To obtain 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).

The screenshot shows the 'Customer' record for 'Abe Simpson'. The 'Fixed Assets' tab is selected. A red box highlights the 'Fixed Assets' sublist, which contains three items:

EDIT	NAME	COST	USEFUL LIFE IN YEARS	REMOVE
Edit	2003 Ford Expedition	32,885.00	5	Remove
Edit	Dell Dimension XPS	1,649.00	3	Remove
Edit	Printer	1,000.00	3	Remove

- 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**.

The screenshot shows the 'Fixed Assets' creation screen. A red box highlights the 'NEW CUSTOMER' dropdown field, which is set to 'Abe Simpson'. A 'Field Help' modal is open, displaying the following information:

This is a custom field of type List/Record
Field ID: custrecord111



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 Customization > Lists, Records, & Fields > Record Types 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)`.

DESCRIPTION	ID	TYPE	LIST/RECORD	TAB	SHOW IN LIST
Cost	custrecord1	Currency			No
Salvage Value	custrecord2	Currency			No
Depreciable Cost	custrecord3	Currency			No
Useful Life In Years	custrecord4	Decimal Number			No
Description	custrecord5	Text Area			No
New Customer	custrecord111	List/Record	Customer		Yes

Example:

```

1 //Get the value of the Cost field on the first line (see the following figure)
2
3 ...
4 var sublistFieldValue = objRecord.getSublistValue({
5   sublistId: 'irecmachcustrecord111',
6   fieldId: 'custrecord1',
7   line: 1
8 });

```

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.

```

1 objRecord.setCurrentSublistValue({
2   sublistId: 'irecmachcustrecord111',
3   fieldId: 'custrecord2',
4 });

```

The screenshot shows a NetSuite customer record for 'Abe Simpson'. The 'Primary Information' section includes fields for CUSTOM FORM (Custom Customer Form 4), PARENT COMPANY (dropdown with placeholder '<Type then tab>'), STATUS (CUSTOMER-Closed Won), and CATEGORY (From advertisement). Below this is an 'Email | Phone | Address' section with fields for EMAIL (asimpson@netsuite.com), ALT. PHONE (555-1234), and FAX. The 'LAST SALES ACTIVITY' section is collapsed. At the bottom, a 'Fixed Assets' tab is selected, showing a sublist titled 'Fixed Assets'. The sublist has columns: EDIT, NAME, COST, USEFUL LIFE IN YEARS, and REMOVE. It lists three items: '2003 Ford Expedition' (cost \$32,885.00, useful life 5 years, remove button), 'Dell Dimension XPS' (cost \$1,649.00, useful life 3 years, remove button), and 'Printer' (cost \$1,000.00, useful life 3 years, remove button).

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 perform validate line functions on custom child record sublists. Validate line functions are performed when a client event occurs prior to a line being added to a sublist.

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.

Sorting by Bill Type Example

You may 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 wanted 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.

Transaction Body Field

		Save	Cancel	Reset	Apply to Forms
LABEL *	<input type="text" value="On Hold"/>				
ID	<input type="text" value="on_hold"/>				
OWNER	<input type="text" value="JL Wolfe"/> ▼				
DESCRIPTION	<input type="text"/>				
Applies To Display Validation & Defaulting Sourcing & Filtering Access Translation					
<input checked="" type="checkbox"/> PURCHASE <input type="checkbox"/> INVENTORY ADJUSTMENTS <input type="checkbox"/> PRINT ON PACKING SLIP <input type="checkbox"/> SALE <input type="checkbox"/> WORK ORDER / ASSEMBLY BUILD <input type="checkbox"/> PRINT ON STATEMENT 					

After bills are entered, go to the Bill Payments page, click Customize View and then select On Hold on the Additional Filters subtab.

Results		Available Filters		Search Title Translation	
Use this tab to specify what filters you would like in the filter region of the search results page.					
<input type="button" value="Remove All"/> <input type="button" value="Add Multiple"/>					
FILTER *		<input type="checkbox"/> SHOW IN FILTER REGION <input type="checkbox"/> SHOW AS MULTI-SELECT		LABEL TRANSLATION	
:: Type <input type="checkbox"/> Yes					
:: Employee <input type="checkbox"/> Yes					
:: UOM Type <input type="checkbox"/> Yes					
:: On Hold (Custom Body) <input type="checkbox"/> Yes					
<input type="button" value="▼ Add"/> <input type="button" value="× Cancel"/> <input type="button" value="+ Insert"/> <input type="button" value="Remove"/> <input type="button" value="▲ Move Up"/> <input type="button" value="▼ Move Down"/> <input type="button" value="▲ Move To Top"/> <input type="button" value="▼ Move To Bottom"/>					

Add the On Hold field to the Results subtab and save.

Results **Available Filters** **Search Title Translation**

Use this tab to indicate columns to be included in the search results as well as sort order.

SORT BY: Date DESCENDING

Remove All Add Multiple

FIELD *	CUSTOM LABEL	CUSTOM LABEL TRANSLATION
Date		
Print		
Type		
On Hold (Custom Body)		
Payroll Batch		

You can now filter results to display only records that are not On Hold.

Bill Payments List Search Audit Trail

View: Custom Default Edit View New Transaction

FILTERS

On Hold	Status
No	- All -

Advanced PDF/HTML Templates

The Advanced PDF/HTML Templates feature supports an alternative 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.) Administrators and users with the Advanced PDF/HTML Templates permission can customize printed and emailed transactions. 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 on 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 [Advanced Templates Customization 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 also can 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 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 removed in a future release. You are encouraged to use advanced templates and associate them with custom forms. For more information, see [Comparison of Basic Printing and Advanced PDF/HTML Printing](#).

Advanced PDF/HTML Templates are supported in SuiteCloud Development Framework (SDF). SDF is a development framework that you can use to create SDF SuiteApps, or to customize NetSuite accounts, using an integrated development environment (IDE) on your local computer. SuiteCloud projects are file-based and use XML definitions of custom NetSuite objects. For more information, see the help topic [SDF Custom Object and File Development in SuiteCloud Projects](#)

Account-Specific Domains in Advanced Printing Templates

You should use URLs that include account-specific domains in PDF templates. Data center-specific URLs found in printing templates are automatically modified to use the appropriate account-specific domain URL for each PDF printout. However, the configuration remains unchanged. You should update all URLs in stored templates to use account-specific domains. For example, you should change a **system.eu2.netsuite.com** link in your template link to <accountID>.app.netsuite.com, where <accountID> is a variable representing your account ID.

Using Advanced Template Formatting Programmatically



Important: SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should access the NetSuite UI only by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic [SuiteScript 2.x 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 `nlaprintRecord(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](#).

Comparison of Basic Printing and Advanced PDF/HTML Printing

Watch the following video that demonstrates and summarizes the advantages of using advanced PDF/HTML templates instead of basic layouts.



[Comparison of Basic Printing and Advanced Printing](#)

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.

Enable Features

SuiteCloud

[Save](#) [Cancel](#) [Reset](#)

[Company](#) [Accounting](#) [Tax](#) [Transactions](#) [Items & Inventory](#) [Employees](#) [CRM](#) [Analytics](#) [Web Presence](#) [SuiteCloud](#)

VIEW SUITECLOUD [TERMS OF SERVICE](#).

SuiteBuilder

ITEM OPTIONS
ASSIGN CUSTOM TRANSACTION ITEM OPTION FIELDS TO THE LINE ITEMS OF YOUR TRANSACTION RECORDS.

CUSTOM RECORDS
COLLECT INFORMATION SPECIFIC TO YOUR BUSINESS THAT CAN BE INTEGRATED WITH STANDARD NETSUITE RECORDS.

ADVANCED PDF/HTML TEMPLATES
ENABLE POWERFUL, TEMPLATE-BASED RENDERING OF SELECTED TRANSACTIONS.

SuiteScript

CLIENT SUITESCRIPT
USE INDUSTRY-STANDARD JAVASCRIPT TO DO ADVANCED CLIENT-SIDE CUSTOMIZATION OF YOUR FORMS. BY ENABLING THIS FEATURE, YOU AGREE TO [SUITECLOUD TERMS OF SERVICE](#).

You 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](#).

Third-Party Products Used in Advanced Printing

The following third-party products are used in advanced printing:

Product	Version
FreeMarker	2.3.26
Big Faceless Report Generator (BFO)	1.1.70
CKEditor	5

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.

Note: The Advanced PDF/HTML Templates menu option is available only to account administrators and other users who have the Advanced PDF/HTML Templates permission. This is a Setup type permission with only one level: Full. For details about permissions, see the help topic [NetSuite Permissions Overview](#).

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 PDF/HTML Templates				
		Submit	New Template	
EDIT	NAME	SCRIPT ID	TYPE ▲	PREFERRED
Customize	Standard Payment Voucher PDF/HTML Template	STDTMPLPAYMENTVOUCHER	Bill Payment	<input checked="" type="checkbox"/>
Customize	Standard AU Vendor Payment PDF/HTML Template	STDTMPLAUVPENDPYMT	Bill Payment	<input checked="" type="checkbox"/>
Customize	Standard US Vendor Payment PDF/HTML Template	STDTMPLUSVENDPYMT	Bill Payment	<input checked="" type="checkbox"/>
Customize	Standard UK Vendor Payment PDF/HTML Template	STDTMPLUKVENDPYMT	Bill Payment	<input checked="" type="checkbox"/>
Customize	Standard Cash Refund PDF/HTML Template	STDTMPLCASHRFND	Cash Refund	<input checked="" type="checkbox"/>
Customize	Standard Cash Sale PDF/HTML Template	STDTMPLCASHSALE	Cash Sale	<input checked="" type="checkbox"/>
Customize	Standard UK Check PDF/HTML Template	STDTMPLUKCHECK	Check	<input checked="" type="checkbox"/>
Customize	Standard Check PDF/HTML Template	STDTMPLCHECK	Check	<input checked="" type="checkbox"/>
Customize	Standard UK Voucher Check PDF/HTML Template	STDTMPLUKVOUCHERCHECK	Check	<input checked="" type="checkbox"/>
Customize	Standard US Voucher Check PDF/HTML Template	STDTMPLUSVOUCHERCHECK	Check	<input checked="" type="checkbox"/>
Customize	Standard AU Voucher Check PDF/HTML Template	STDTMPLAUVOUCHERCHECK	Check	<input checked="" type="checkbox"/>
Customize	Standard Paycheck PDF/HTML Template	STDTMPLPAYCHECK	Check	<input checked="" type="checkbox"/>
Customize	Standard Credit Memo PDF/HTML Template	STDTMPLCUSTCRED	Credit Memo	<input checked="" type="checkbox"/>
Customize	Standard Customer Deposit PDF/HTML Template	STDTMPLCUSTDEP	Customer Deposit	<input checked="" type="checkbox"/>
Customize	Standard Quote PDF/HTML Template	STDTMPLQUOTE	Estimate	<input checked="" type="checkbox"/>

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 Multi-Currency Statement Templates

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 Invoice 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. Choose 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.

The screenshot shows the 'Custom Transaction Form' configuration page in SuiteBuilder. At the top, there are buttons for Save, Cancel, Reset, and Save & Move Elements. The 'NAME' field is populated with 'Custom Purchase Order'. The 'TYPE' field shows 'Purchase Order'. Under 'PRINTING TYPE', the radio button for 'ADVANCED' is selected. A red box highlights the 'PRINT TEMPLATE' dropdown, which contains 'Custom Purchase PDF/HTML Template'. Below it is another dropdown for 'EMAIL TEMPLATE', showing 'Standard Purchase PDF/HTML Template'. To the right, there are sections for 'DISCLAIMER' and 'ADDRESS'. The 'LOGO' section includes a dropdown menu and settings for 'COLUMNS WIDTH' (12.75) and 'LAYOUT SPACE' (7.5). Checkboxes for 'ALLOW ADD MULTIPLE', 'INACTIVE', and 'FORM IS PREFERRED' are also present. At the bottom, tabs for Tabs, Field Groups, Screen Fields, Actions, Lists, Custom Code, Roles, and Linked Forms are visible.

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.

The screenshot shows the Purchase Order screen with the following details:

- Purchase Order** header with search and navigation buttons.
- AMS7791-PO Acme Medical Supply** transaction number and vendor.
- FULLY BILLED** status indicator.
- CUSTOM FORM ***: A dropdown menu is open, showing "Custom Purchase Order" highlighted with a red border.
- CURRENCY**: US Dollar.
- VENDOR ***: Acme Medical Supply.
- PURCHASE CONTRACT**: A dropdown menu.
- SUPERVISOR APPROVAL**: Checked checkbox.
- DATE ***: 3/29/2003.
- PO #**: AMS7791-PO.
- DEPARTMENT**: Accounting.
- CLASS**: A dropdown menu.
- APPROVAL STATUS**: A placeholder box.
- Summary** section with **TOTAL** 22,497.50.
- Actions** dropdown menu.

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 select **Basic** for **Printing Type**, and choose from the **PDF Layout** and **HTML Layout** dropdown lists. All previously available basic layouts are still available.

When creating a transaction, you 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 Invoice Using Advanced Template

The following invoice uses the Standard Invoice PDF/HTML Template:

 WOLFE ELECTRONICS	Wolfe Electronics 1500 3rd St San Mateo, CA 94403	Invoice #1031 May 1, 2013			
Bill To James A Sample 1234 Any Street Nowhere, ST 94000 USA	Ship To James A Sample 1234 Any Street Nowhere, ST 94000 USA	TOTAL \$5,717.50 <small>Due Date: May 31, 2013</small>			
Terms	Due Date	PO #	Sales Rep	Ship Via	Partner
30 Days	May 31, 2013	21521487	Clark Koozer	FED EX	Rockstar Beverages
Qty	Item		Options	Unit Price	Amount
3	Computer Systems: Desktop: Impressivo 1500 Impressivo 1500 (Superion 1.5 GHz processor 640MB RAM, 40GB Hard Drive) w/17" CRT Monitor, Keyboard, Mouse			\$1,299.00	\$3,897.00
1	Computer Systems: Laptop: Inertio 1600 Inertio 1600 (Moblio 1GHz processor, 512 MB RAM, 40GB HD, DVD/CD-R)			\$1,499.00	\$1,499.00
1	Computer Systems: Laptop: Gratio 1800 Gratio 1800 (Moblio 1.3 GHz processor, 512 MB RAM, 40GB HD, DVD/CD-R)			\$1,699.00	\$1,699.00
				Subtotal	\$5,396.00
				Tax (5.96%)	\$321.50
				Total	\$5,717.50

Advanced Templates Customization in the Template Editor

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](#).

i 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 items to move them 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:

- Viewing an Advanced Template in the Template Editor
- Template Setup Window
- Previewing Advanced PDF/HTML Templates
- Saving an Advanced Template
- Error Messages in Advanced Templates
- WYSIWYG Editing in the Template Editor
- Source Code Editing in the Template Editor

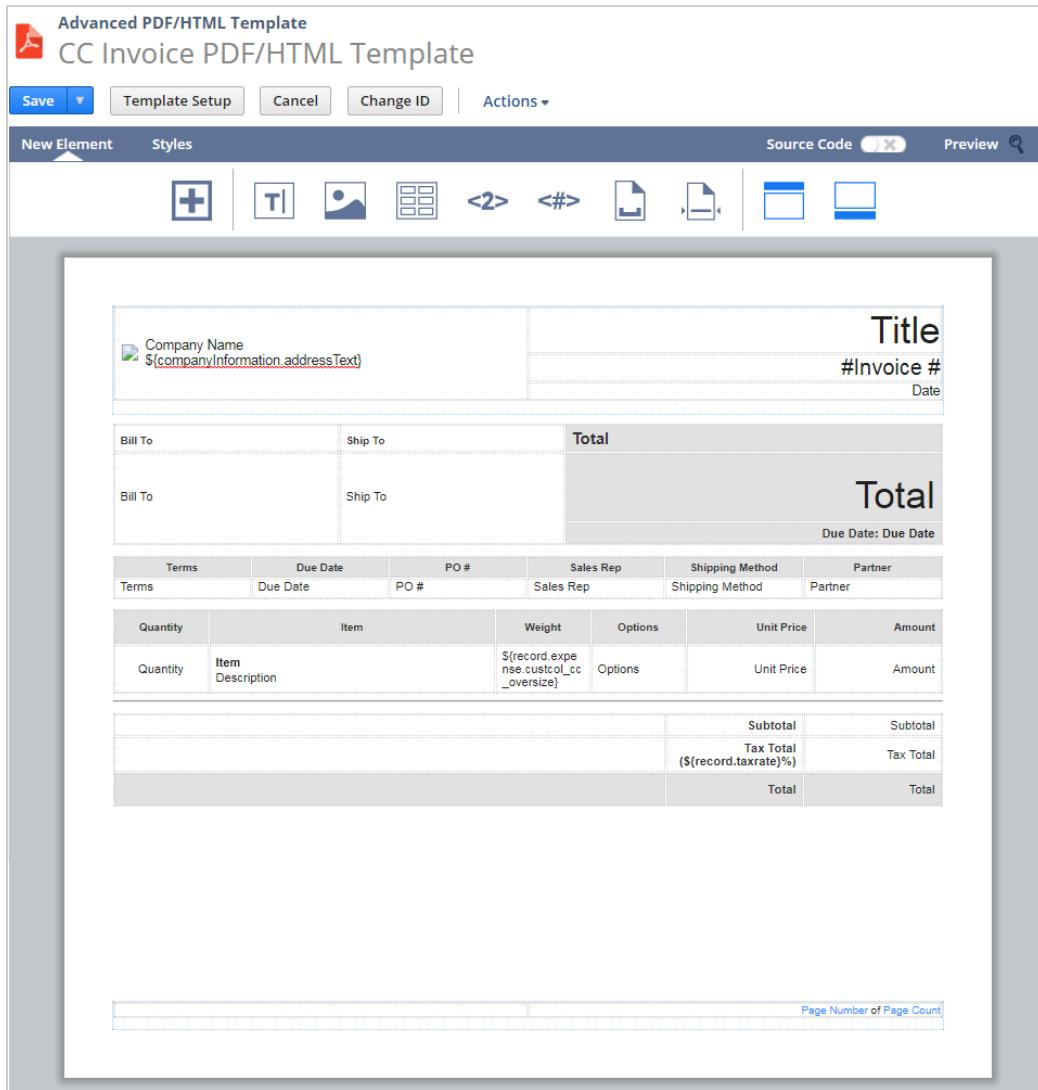


Note: The template editor uses FreeMarker-based syntax. For more information, see the FreeMarker documentation.

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 needed.



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.

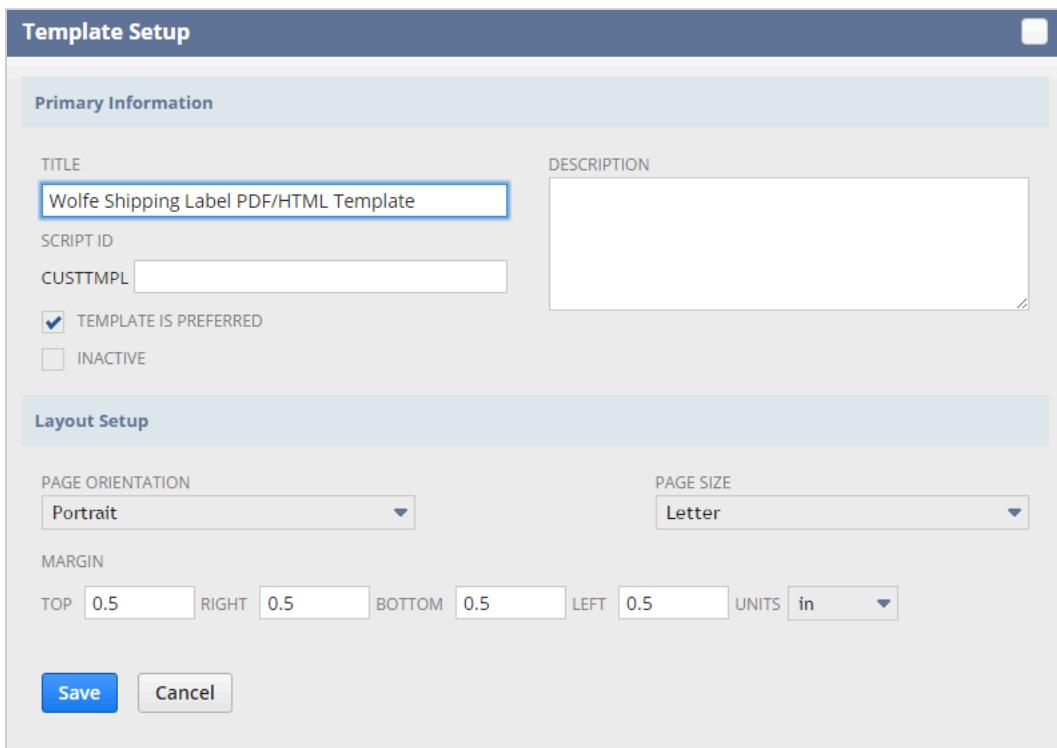
The following table shows the records that are accessed by each template type.

Advanced Template Type	Records Accessed
Bill	Bill
Bill Credit	Bill credit
Bill of Materials	Work order
Bill Payment	Vendor payment
Cash Refund	Cash refund
Cash Sale	Cash sale
Check	Various payment types. For the template preview, the Check record is used. For more information, see Standard Check PDF/HTML Template.
Credit Memo	Credit memo
Custom Transaction	Custom transaction
Customer Deposit	Customer deposit
Expense Report	Expense report
Invoice	Invoice
GL Impact	GL Impact
Item Label	No record is loaded in a way that triggers scripts
Journal	Journal entry
Mailing Label	Address (on all entity records)
Packing Slip	Item fulfillment, and based on the original transaction, one of transfer order or sales order
Payment	Customer payment
Picking Ticket	Sales order, transfer order
Price List	Item search per customer
Purchase Order	Purchase order
Quote	Estimate
Return Authorization	Return authorization
Sales Order	Sales order
Saved Search	Saved search
Shipping Label	No record is loaded in way that triggers scripts
Statement and Multi-Currency Statement	Customer
Work Order	Work order

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:

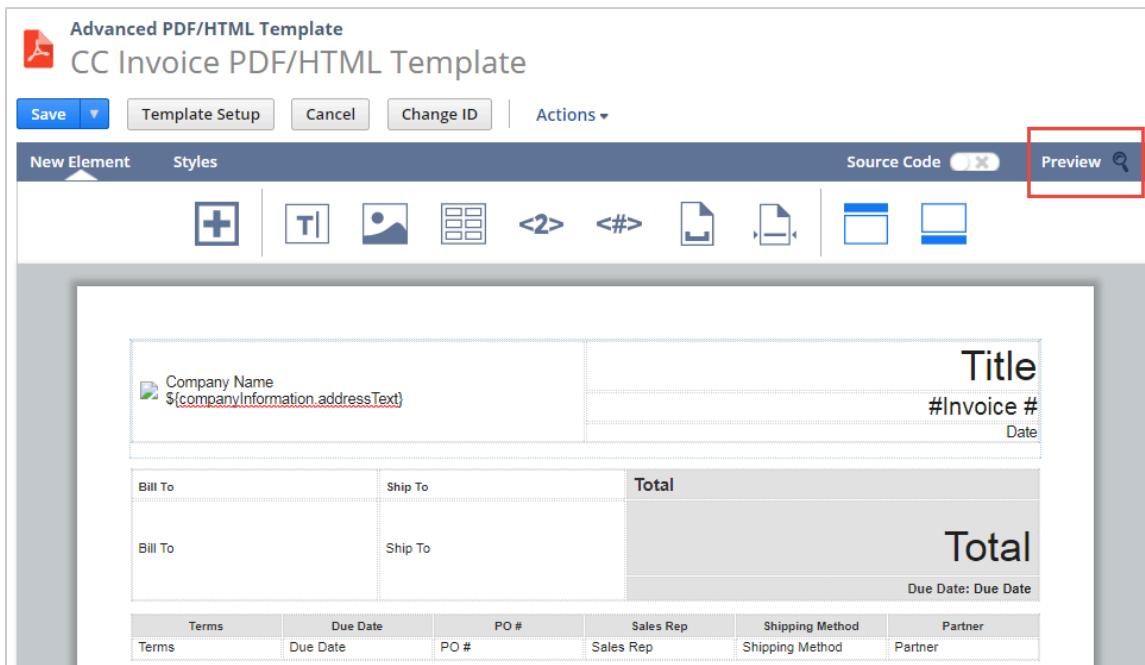
- Basic template properties, including:
 - Title (name)
 - Description
 - Whether the template is preferred for the transaction type
 - Script ID, which is assigned automatically if you leave this field blank
 - Saved Search Template, which generates a template based on an existing saved search (only available for Saved Search templates)
 - Template inactivation, if required
- Template layout settings, including:
 - Page orientation
 - Page size
 - Margins



For information about options for saving templates, see [Saving an Advanced Template](#)

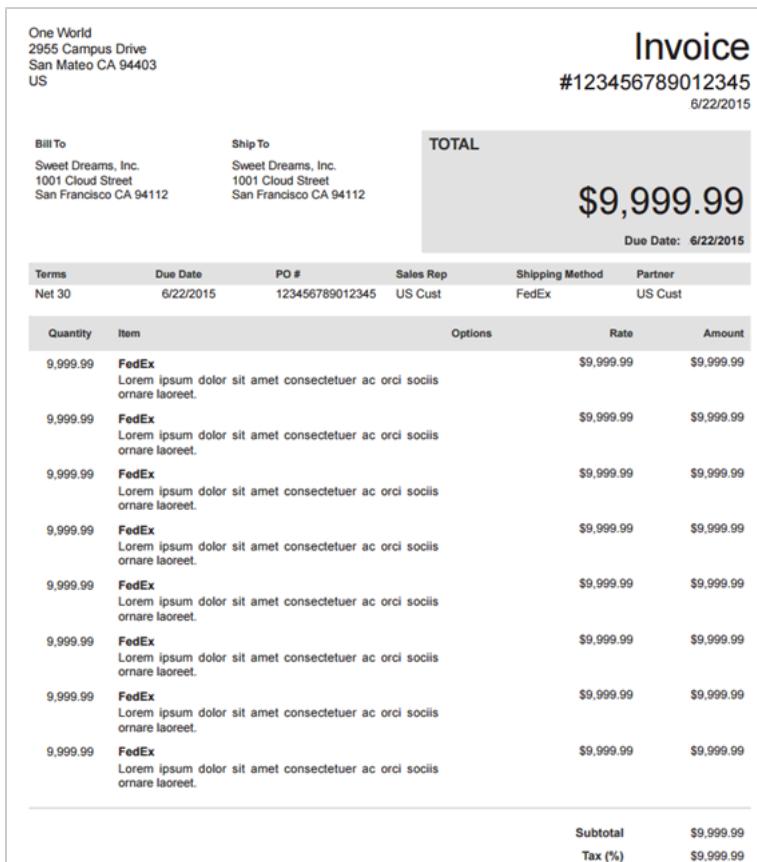
Previewing Advanced PDF/HTML Templates

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

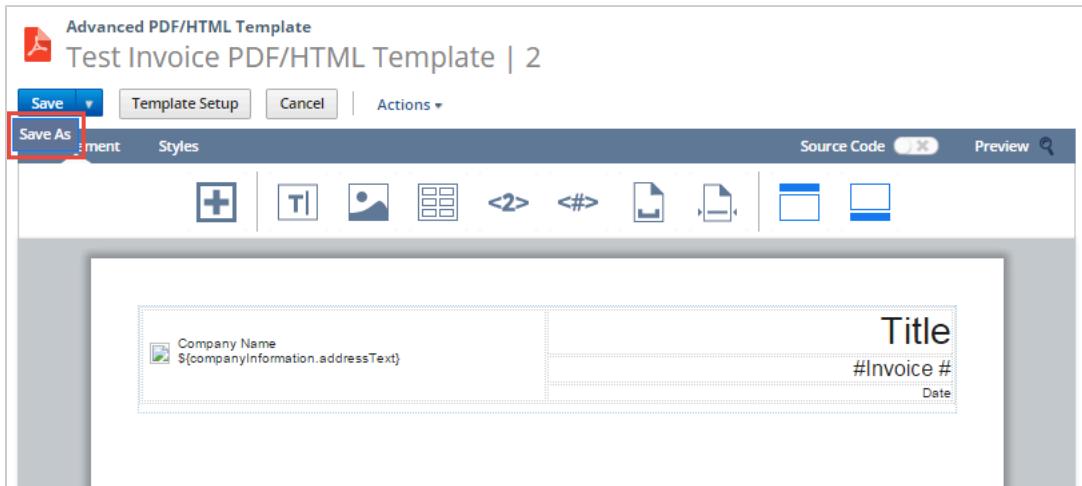
When saving an advanced template, you have the following options:

- **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

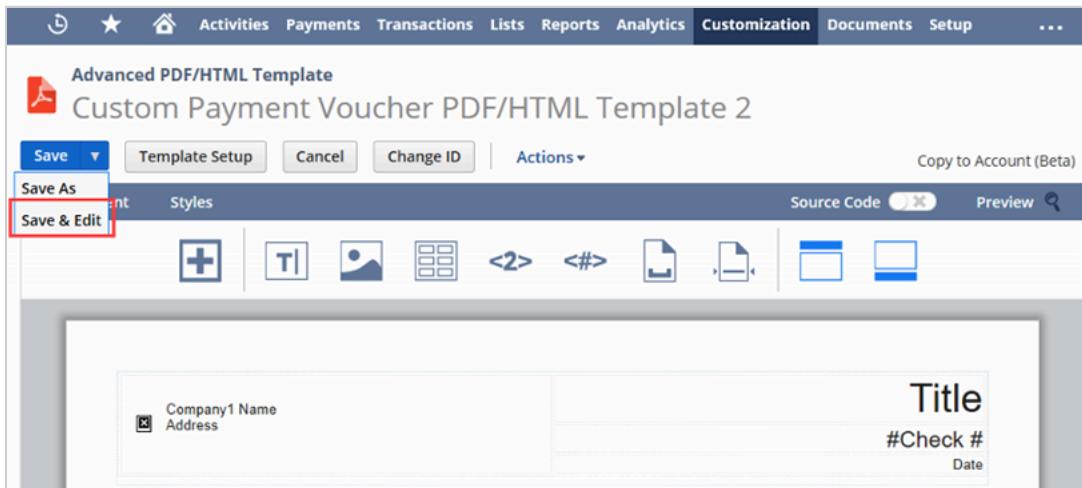
If you want to use an existing customized advanced template as the basis for a new template, you can use the **Save As** button to save a template under another name.

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 [Template Setup Window](#).
3. Click **Save**.
4. In the template editor, click **Save As**. The new template is saved.



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

Error Messages in Advanced Templates

There are two conditions under which error messages can be displayed when working with advanced templates. You may see an error related to the virtual record generated when saving a template. You may also see error messages when you preview, save or print a template.

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 may 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:

```
 ${item.taxcode?substring(4,9)}
```

To ensure that the template saves correctly, add an if statement to verify the length of the string.

```
 1 <#if (item.taxcode?length > 9)>
 2   ${item.taxcode?substring(4,9)}
 3 </#if>
```

Errors when Previewing, Saving or Printing 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.



Note: For PDF creation, there is a size limitation of 100MB for XML file input. If the file size exceeds 100MB, printing results in an error.

WYSIWYG Editing in the Template Editor

Using the Advanced Template Editor

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 window appears 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.

The screenshot shows the 'Advanced PDF/HTML Template' editor interface. At the top, there are buttons for 'Save', 'Template Setup', 'Cancel', 'Change ID', and 'Actions'. Below the toolbar, there are tabs for 'New Element' and 'Styles', along with 'Source Code' and 'Preview' buttons. The main area contains a WYSIWYG editor with a toolbar for text and style formatting. The template layout includes:

- Header:** Company Name \${companyInformation.addressText} and fields for Title, Invoice #, and Date.
- Bill To / Ship To:** Tables for Bill To and Ship To addresses, both labeled 'Total'.
- Payment Terms:** A table with columns for Terms, Due Date, PO #, Sales Rep, Shipping Method, and Partner.
- Item Details:** A table with columns for Quantity, Item Description, Weight, Options, Unit Price, and Amount.
- Tax and Subtotal:** A table with columns for Subtotal, Tax Total, and Total.
- Page Number:** A footer section labeled 'Page Number of Page Count'.

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 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.

Button	Description
	Add a field to the template. For more information, see Adding and Removing Fields in Advanced Templates .
	Add text to the template. For more information, see Adding and Formatting Text in Advanced Templates .
	Add an image to the template. For more information, see Including Images in Advanced Templates .
	Add a table to the template. For more information, see Adding Tables to Advanced Templates .
	Add the page number to the template.
	Add the total number of pages to the template.
	Add a page break to the template. If you want to remove the page breaks, you must switch to Source Code view.
	Add a horizontal line to the template.
	Toggle button to remove or add the template header. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> Note: If you remove the header from the template and then add it again, the header content is erased. </div>
	Toggle button to remove or add the template footer.

Button	Description
	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

Adding Fields and Sublists to Advanced PDF/HTML Templates

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 needed, 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: 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](#).

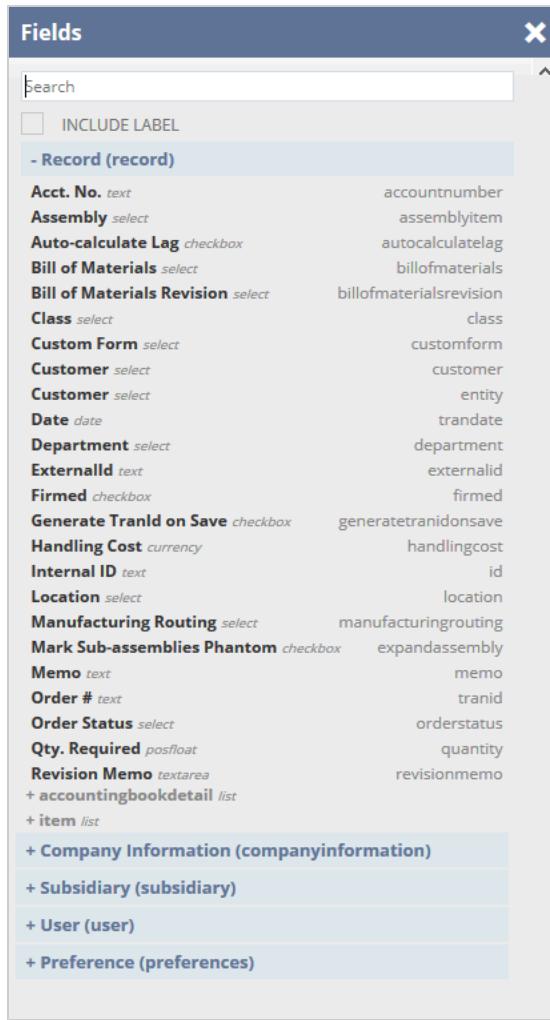
Adding a Field to an Advanced Template

You use the Fields selector to add a field to an advanced template.

To add a field to an advanced template:

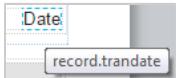
1. Go to Customization > Forms > Advanced PDF/HTML Templates and click **Customize** or **Edit** for the advanced template.
2. Position the cursor in the location of the template editor where you want to add a field.
3. On the New Element toolbar, click the **Fields** button to display a **Fields** selector, 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 the 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 appears:



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](#).

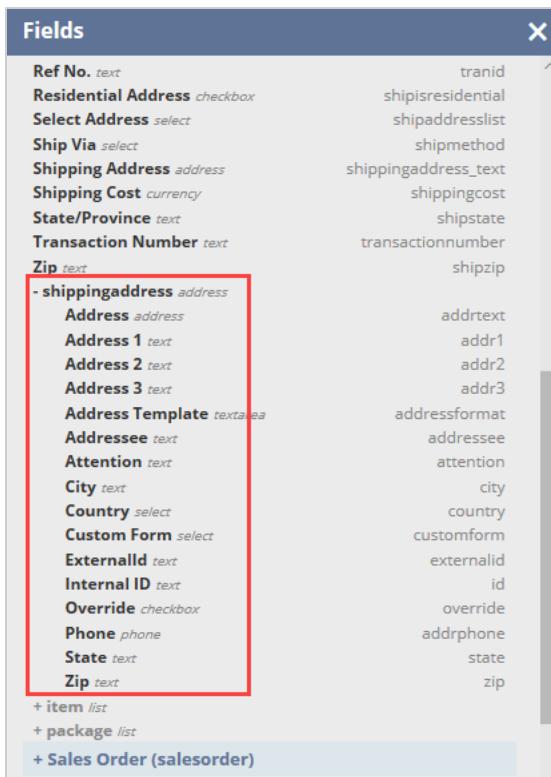
With the appropriate permission, you can include address subrecord fields in advanced templates, as well as reference fields and labels, by using FreeMarker syntax.

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:

```
$(record.currencysymbol)Total
```

Removing a Currency Symbol from an Advanced Template Field

You can use a FreeMarker expression to remove a currency symbol from an advanced template, such as a check.

Use the remove currency format: \${nsformat_number(number|string)}. For more information, see [FreeMarker Formatting Method](#).

Removing a Field from an Advanced Template

There can be some fields on a template that are not required by your organization.

To remove a field from an advanced template, select the field 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.



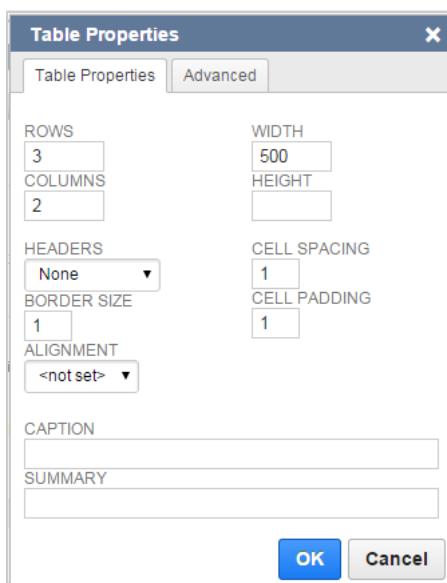
Note: For better performance, you shouldn't use nested tables, that is, tables within tables.

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:



4. Complete the Table Properties window and click **OK**.



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 required 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 element 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:

```

1 <table>
2 <ul>
3 <p>

```

```

4 <pre>
5 <ol>
6 <h1>
7 <h2>
8 <h3>
9 <h4>
10 <blockquote>
```

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. For more information about the File Cabinet, see the help topic [File Cabinet Overview](#).

The New Element toolbar of the template editor includes a button that you can click to insert an image.

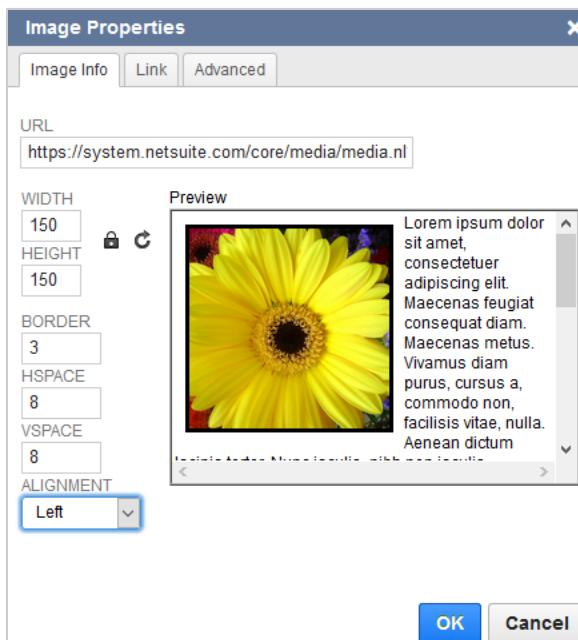
To insert an image on an advanced template:

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.



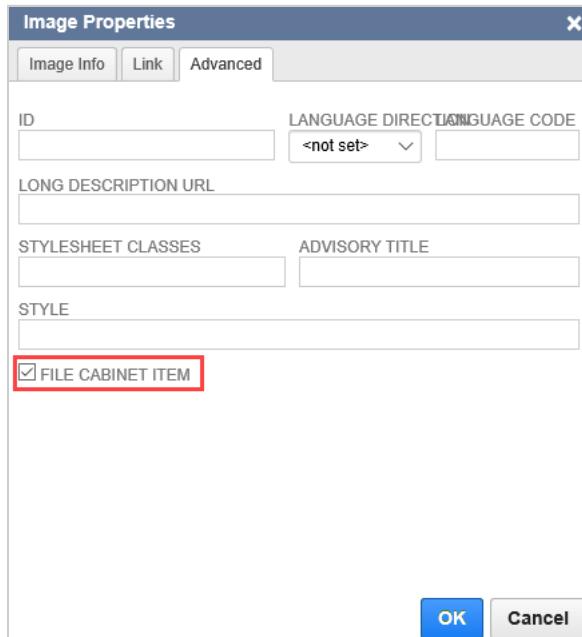
4. In the **URL** field, enter the full path to the image file in the File Cabinet or on the internet. Use **&** for ampersand symbols.

Include fields that dynamically return a URL, for example \${companyInformation.appDomain}. For more information, see [Syntax for Advanced Template Fields](#).



5. Complete the following, as needed:

- a. In the **Width** field, enter the size in pixels to use for the image width.
 - b. In the **Height** field, enter the size in pixels to use for the image height.
 - c. In the **Border** field, enter the thickness in pixels to use for the image border.
 - d. In the **Hspace** field, enter the space in pixels to use for the horizontal space between the image and the surrounding text.
 - e. In the **Vspace** field, enter the space in pixels to use for the vertical space between the image and the surrounding text.
 - f. In the **Alignment** field, select **Left** or **Right** for the image alignment.
6. 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.



7. 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.



Important: To avoid issues when copying and pasting the code, use a plain text editor to check for any hidden characters.

Advanced PDF/HTML Template
CC Invoice PDF/HTML Template

Save ▾ Template Setup Cancel Change ID Actions ▾

Source Code Preview

```

1 <?xml version="1.0"?><!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
2 <pdf>
3   <head>
4     <if .locale == "ru_RU">
5       <link name="verdana" type="font" subtype="opentype" src="${nsfont.verdana}" src-bold="${nsfont.verdana_bold}" bytes="2" />
6     </if>
7     <macrolist>
8       <macro id="nlheader">
9         <table class="header" style="width: 100%; "><tr>
10        <td rowspan="3"><if companyInformation.logoUrl.length != 0> </if> <span class="nameandaddress">${companyInformation.companyName}</span><br /><span class="nameandaddress">${companyInformation.addressText}</span></td>
11        <td align="right"><span class="title">${record@title}</span></td>
12      </tr>
13      <tr>
14        <td align="right"><span class="number">#${record.tranid}</span></td>
15      </tr>
16      <tr>
17        <td align="right">${record.trandate}</td>
18      </tr></table>
19      </macro>
20      <macro id="nlfooter">
21        <table class="footer" style="width: 100%; "><tr>
22          <td><barcode codetype="code128" showtext="true" value="${record.tranid}" /></td>
23          <td align="right"><span>${record@page} of ${totalPages}</span></td>
24        </tr></table>
25      </macro>
26    </macrolist>
27    <style type="text/css">table {
28      <if .locale == "zh_CN">
29        font-family: stsong, sans-serif;
30      <elseif .locale == "zh_TW">
31        font-family: msung, sans-serif;
32      <elseif .locale == "ja_JP">
33        font-family: heiseimin, sans-serif;
34      <elseif .locale == "ko_KR">
35        font-family: hygothic, sans-serif;

```

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
- 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 uppercase directive, for example: {\$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 elements that are closed are subject to HTML formatting, for example, </totalpages> processes as <totalpages> </totalpages>.

Some HTML elements can 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 "
" is displayed.



Important: The following are not supported in advanced templates because of BFO processing.

Do not use the cellspacing attribute. Instead, use the cellmargin 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, <body width="595">.

Source Code Editing to Customize Advanced Templates



Important: NetSuite uses BFO, FreeMarker, and CKEditor. For version details, see [Third-Party Products Used in Advanced Printing](#).

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](#).

An advanced template that is used to print a PDF file is an XML document that uses syntax similar to HTML with FreeMarker and BFO (Big Faceless Organization) report generator elements included. For example:

```

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 </pdf>
```

Line 1 of the template must declare the XML version, and Line 2 must specify the DOCTYPE. Line 3 includes the BFO `<pdf>` wrapping tag, not the `<html>` declaration you would see at the top of an HTML document.

Inside the `<pdf>` tag, 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!". For more information about the Big Faceless Report Generator, see [BFO User Guide](#).

In XML, elements must always be closed. As shown in Line 8, `<pdf>` must always be matched by `</pdf>`, `` by `` and so on. For elements that have no content, like `
`, the closing tag is included in the element: `
`. 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: 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 [Third-Party Products Used in Advanced Printing](#). 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 />` tags 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>` tag. To apply the macro, you reference it in the `<body>` tag 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.

```

1 <?xml version="1.0"?>
2 <!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
3 <pdf>
4   <head>
5     ...
6   <macrolist> <!-- Definition of macros -->
7     <macro id="nlHeader"> <!-- Regular macros -->
8       ... Header Content ...
9     </macro>
10    <macro id="nlFooter">
11      ... Footer Content ...
12    </macro>
13    <macro id="nlWatermark">
14      ... Footer Content ...
15    </macro>
16    <macro id="nlAltHeader"> <!-- Alternative macros -->
```

```

17     ... Header Content ...
18 </macro>
19 <macro id="nlAltFooter">
20     ... Footer Content ...
21 </macro>
22 <macro id="nlAltWatermark">
23     ... Footer Content ...
24 </macro>
25 </macrolist>
26 ...
27 </head>
28 <body header="nlHeader" header-height="2.5in" footer="nlFooter" footer-height="0.5in" background-macro="nlWatermark">
29     ... Body Content each page with regular header, footer and watermark ...
30     <pbr header="nlAltHeader" header-height="2.2in" footer="nlAltFooter" footer-height="0.3in" background-macro="nlAltWater-
31         mark" />
32         ... New page with alternative header, footer and watermark ...
33         ... After that each page with regular header, footer and watermark ...
34     </body>
35 </pdf>

```



Note: If you are having issues with your advanced template, do not contact BFO directly. Always contact NetSuite Customer Support.

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.

```
1 | <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 [Third-Party Products Used in Advanced Printing](#). 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 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} can 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.

```

1 <#list record.item as item>
2   ${item_index} ${item.itemName@label} ${item.itemName} --- ${item.amount}
3 </#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\$

When the sublist ID is the same as the field ID, `@list` is added to the sublist ID. The correct syntax appears in the **Fields** selector automatically. The following example shows the syntax to access a sublist:

```

1 <#list customer.currency@list as customerCurrency>
2   ${customerCurrency.currency} <br/>
3   ${customerCurrency.balance}
4 </#list>
```

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 can 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 can 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 () character is not rendered when using advanced templates. In general, use a line break element (`
`) instead of the character.

If you are using the character to create spacing in HTML table layouts, use the width and height attributes of the `<td>` tag instead to specify how BFO formats the `<table>` tag. For example:

```

1 <table table-layout="fixed" width="200">
2   <tr><td width="60%">Cell 1</td><td width="40%">Cell 2</td></tr>
3   <tr><td>Cell 3</td><td>Cell 4</td></tr>
```

4 | </table>

For more information about tables in BFO, see the BFO TABLE element documentation at <http://bfo.com/products/report/docs/tags/tags/table.html>.

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 similar to HTML. Note that although this syntax can 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 select 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](#)

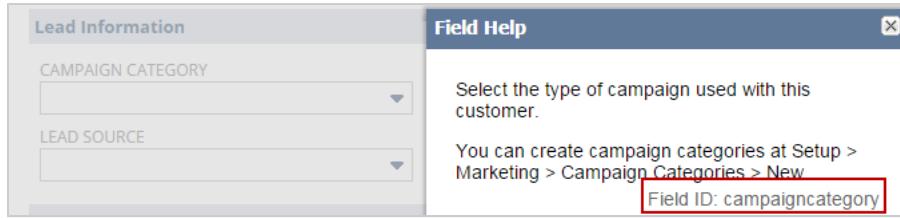
Entering a Field Manually to an Advanced Template

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 window.



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, check 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:



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.

i 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 have the Administrator role or 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 appears 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 element, for example, 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:


```
<a href="http://www.netsuite.com">http://www.netsuite.com</a>
```
 - 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.

```
1 | <#list record.item as item>
```

```

2 |     ${item._index} ${item.itemName@label} ${item.itemName} --- ${item.amount}
3 | </#list>

```

This declaration prints out a block of HTML code to display the whole list.

```

1 | 0 Name: Blue T-Shirt --- 10.00$
2 | 1 Name: Green T-Shirt --- 12.25$
3 | 2 Name: Yellow T-Shirt --- 11.00$

```

To reference one item in the list, you can use \${record.item[index].itemName} to obtain only that line in the sublist. Using the preceding example, \${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}

When the sublist ID is the same as the field ID, @list is added to the sublist ID. The correct syntax appears in the **Fields** selector automatically. The following example shows the syntax to access a sublist:

```

1 | <#list customer.currency@list as customerCurrency>
2 | ${customerCurrency.currency} <br/>
3 | ${customerCurrency.balance}
4 | </#list>

```

You can reference fields on a sublist at the first level only. Referencing sublists at the second level is **not** supported.

For example, only the following is supported.

```

1 | <#list record.item as item>
2 |   ${item.field_first_level}
3 | </#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.addressee}
- \${subsidiary.mainaddress.addr1}
- \${subsidiary.mainaddress.addr2}
- \${subsidiary.mainaddress.city}
- \${subsidiary.mainaddress.dropdownstate}
- \${subsidiary.mainaddress.zip}
- \${subsidiary.mainaddress.country}
- \${subsidiary.mainaddress.addrphone}

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 also can 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.

```

1 | <#list results as salesOrder>
2 |   <p>${salesOrder.customer.phone}</p>
3 | </#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 hardcoded. 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:

Example	Output
\${record.custbody_show_int}	1,234
\${record.custbody_show_int+1}	1235

Example	Output
<code> \${nsformat_number(record.custbody_show_int+1)}</code>	1,235
<code> \${record.custbody_show_date_time}</code>	Dec 18, 2019 5:05:00 AM

The following table shows some examples of the `${nsformat_*()}` formatting method in use:

Method	Example	Output
<code> \${nsformat_date(date string)}</code>	<code> \${nsformat_date("2018-12-04T10:40:00.000Z")}</code>	12/4/2018
<code> \${nsformat_datetime(date string)}</code>	<code> \${nsformat_datetime("2018-12-04T10:40:00.000Z")}</code>	Dec 4, 2018 2:40:00 AM
	<code> \${nsformat_datetime("2018-12-04T10:40:00.000Z")}</code>	Dec 4, 2018 10:40:00 AM
	<code> \${nsformat_datetime(itemDateTime)}</code>	Dec 4, 2018 10:40:00 AM
<code> \${nsformat_time(date string)}</code>	<code> \${nsformat_time("2018-12-04T10:40:00.000Z")}</code>	10:40:00 AM
<code> \${nsformat_currency(number string)}</code> or with optional second string <code> \${nsformat_currency(number string, string)}</code>	<code> \${nsformat_currency(10.20, "EUR")}</code>	€10,20
<code> \${nsformat_rate(boolean string number)}</code> or with optional second string <code> \${nsformat_rate(number string, string)}</code>	<code> \${nsformat_rate(32, "USD")}</code>	\$32.00
<code> \${nsformat_number(number string)}</code> or with optional second locale string <code> \${nsformat_number(number string, string)}</code>	<code> \${nsformat_number("9999999.99")}</code> <code> \${nsformat_number("999999999.99", "cs_CZ")}</code>	999,999,999.99 999 999 999,99
<code> \${nsformat_boolean(boolean string number)}</code>	<code> \${nsformat_boolean("true")}</code>	Yes
<code> \${nsformat_password(string)}</code>	<code> \${nsformat_password("p4sw0rd")}</code>	*****
<code> \${nsformat_percent(number string)}</code>	<code> \${nsformat_percent("100")}</code>	100%
<code> \${nsformat_email(string)}</code>	<code> \${nsformat_email("email@example.com")}</code>	clickable email address
<code> \${nsformat_url(string)}</code>	<code> \${nsformat_url("http://example.com")}</code>	clickable link

Formatters accept numbers in double precision floating point representation.

If the input is an empty string or null value, 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:

```

1 <pdfset>
2 <#list statements as record>
3   ...original template...

```

```

4 | </#list>
5 | </pdfset>
```

Change any record.items references to record.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:

```

1 | <@filecabinet nstype="image" src="https://<accountID>.app.netsuite.com/core/media/media.nl?id=21&c=4130331&h=f
b3b8b4ac4f67b2c369b"/>
```

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 [File Cabinet Overview](#).

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.

```

1 | <?xml version="1.0"?><!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
2 | <pdf>
3 | <head>
4 |   <link name="thai-font" type="font" subtype="opentype" src="https://<accountID>.app.netsuite.com/core/media/media.nl?
id=5967216&#038;c=3809789&#038;h=8609ede0aef4cbfad71d&#038;_xt=.ttf" src-bold="https://<accountID>.app.netsuite.com/core/media/
media.nl?id=5967216&#038;c=3809789&#038;h=8609ede0aef4cbfad71d&#038;_xt=.ttf" bytes="2" />
5 |
6 |   <style type="text/css">
7 |     * {
8 |       font-family: thai-font, sans-serif;
9 |       font-size: 9pt;
10 |      table-layout: fixed;
11 |    }
12 |    th {
13 |      font-weight: bold;
14 |      font-size: 8pt;
15 |      vertical-align: middle;
16 |      padding: 5px 6px 3px;
17 |      background-color: #e3e3e3;
18 |      color: #333333;
19 |    }
20 |    td {
21 |      padding: 4px 6px;
22 |    }
23 |   </style>
24 | </head>
```

```

25 <body>
26   This&#3585;example&#3586;has&#3587;words&#3588;separated&#3589;by&#3590;Thai&#3591;characters .
27 </body>
28 </pdf>

```

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.

Font Name	Language Code	Language
NotoSans_Bold	—	580+ languages supported
NotoSans_BoldItalic	—	For more information, see https://www.google.com/get/noto/
NotoSans_Italic	—	
NotoSans-Regular	—	
NotoSerif_Bold	—	580+ languages supported
NotoSerif_BoldItalic	—	For more information, see https://www.google.com/get/noto/
NotoSerif_Italic	—	
NotoSerif-Regular	—	
NotoSansCJKsc_Bold	zh_CN	Chinese (Simplified)
NotoSansCJKsc-Regular	zh_CN	
NotoSansCJKtc_Bold	zh_TW	Chinese (Traditional)
NotoSansCJKtc-Regular	zh_TW	
NotoSansCJKjp_Bold	ja_JP	Japanese
NotoSansCJKjp-Regular	ja_JP	
NotoSansCJKkr_Bold	ko_KR	Korean
NotoSansCJKkr-Regular	ko_KR	
NotoSansThai_Bold	th_TH	Thai
NotoSansThai-Regular	th_TH	

Font Name	Language Code	Language
NotoSansArabic_Bold	—	Arabic - 60+ languages supported
NotoSansArabic-Regular		https://www.google.com/get/noto/
NotoSansHebrew_Bold	he_IL	Hebrew
NotoSansHebrew-Regular		
NotoSansArmenian_Bold	hy_AM	Armenian
NotoSansArmenian-Regular		
NotoSansBengali_Bold	bn_DB	Bengali
NotoSansBengali-Regular		
NotoSansGujarati_Bold	gu_IN	Gujarati
NotoSansGujarati-Regular		
NotoSansDevanagari_Bold	hi_IN	Hindi
NotoSansDevanagari-Regular		
NotoSansDevanagari_Bold	mr_IN	Marathi
NotoSansDevanagari-Regular		
NotoSansKannada_Bold	kn_IN	Kannada
NotoSansKannada-Regular		
NotoSansTamil_Bold	ta_IN	Tamil
NotoSansTamil-Regular		
NotoSansTelugu_Bold	te_IN	Telugu
NotoSansTelugu-Regular		

For a complete list of supported languages, see the [Google website](#).

Templates that use the Noto fonts have the following code in the <head> section.

```

1  <link name="NotoSans" type="font" subtype="truetype" src="${nsfont.NotoSans_Regular}" src-bold="${nsfont.NotoSans_Bold}" src-
2   italic="${nsfont.NotoSans_Italic}" src-bolditalic="${nsfont.NotoSans_BoldItalic}" bytes="2" />
3   <if .locale == "zh_CN">
4     <link name="NotoSansCJKsc" type="font" subtype="opentype" src="${nsfont.NotoSansCJKsc-Regular}" src-bold="${nsfont.NotoSan-
5      scJKsc_Bold}" bytes="2" />
6   <elseif .locale == "zh_TW">
7     <link name="NotoSansCJKtc" type="font" subtype="opentype" src="${nsfont.NotoSansCJKtc-Regular}" src-bold="${nsfont.NotoSan-
8      cJKtc_Bold}" bytes="2" />
9   <elseif .locale == "ja_JP">
10    <link name="NotoSansCJKjp" type="font" subtype="opentype" src="${nsfont.NotoSansCJKjp-Regular}" src-bold="${nsfont.NotoSan-
11      cJKjp_Bold}" bytes="2" />
12   <elseif .locale == "ko_KR">
13    <link name="NotoSansCJKkr" type="font" subtype="opentype" src="${nsfont.NotoSansCJKkr-Regular}" src-bold="${nsfont.NotoSan-
14      cJKkr_Bold}" bytes="2" />
15   <elseif .locale == "th_TH">
16    <link name="NotoSansThai" type="font" subtype="opentype" src="${nsfont.NotoSansThai-Regular}" src-bold="${nsfont.No-
17      toSansThai_Bold}" bytes="2" />
18 </if>
```

The following styles definitions are also included.

```
1 | <style type="text/css">
```

```

2   * {
3     <if .locale == "zh_CN">
4       font-family: NotoSans, NotoSansCJKsc, sans-serif;
5     <elseif .locale == "zh_TW">
6       font-family: NotoSans, NotoSansCJKtc, sans-serif;
7     <elseif .locale == "ja_JP">
8       font-family: NotoSans, NotoSansCJKjp, sans-serif;
9     <elseif .locale == "ko_KR">
10      font-family: NotoSans, NotoSansCJKkr, sans-serif;
11    <elseif .locale == "th_TH">
12      font-family: NotoSans, NotoSansThai, sans-serif;
13    <else>
14      font-family: NotoSans, sans-serif;
15    </if>
16  }
17 ...
18 </style>
```

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:

```

1 <link name="NotoSansArabic" type="font" subtype="opentype" src="${nsfont.NotoSansArabic_Regular}" src-bold="${nsfont.NotoSansAra
bic_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:

```

1 * { 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.

```

1 td p {
2   align: left;
3 }
```

Note: If you notice missing or corrupted glyphs (usually for ligatures), particularly when using the Arabic locale, add attribute `subset="false"` to the `<link>` tag. 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 can 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:

```

<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:

```
font-family: NotoSans, sans-serif;
```

Change the line to the following:

```
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 if the required fonts are linked.

Steps to achieve such configuration:

1. Load all required fonts by <link> tag.
2. Create CSS selector to pair language and font.
 - Set any CSS properties specific for the particular 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.

```
<?xml version="1.0"?>
<!DOCTYPE pdf SYSTEM "report-1.1.dtd" PUBLIC "-//big.faceless.org//report">
- <pdf>
  - <head>
    <link bytes="2" src-bolditalic="${nsfont.NotoSans_BoldItalic}" src-
          italic="${nsfont.NotoSans_Italic}" src-
          bold="${nsfont.NotoSans_Bold}" src="${nsfont.NotoSans-Regular}">
    <link bytes="2" src-bold="${nsfont.NotoSansCJKkr-Bold}">
      src="${nsfont.NotoSansCJKkr-Regular}" subtype="opentype"
      type="font" name="NotoSansCJKkr"/>
    <link bytes="2" src-bold="${nsfont.NotoSansArabic-Bold}">
      src="${nsfont.NotoSansArabic-Regular}" subtype="opentype"
      type="font" name="NotoSansArabic" subset="false"/>
    <style type="text/css"> * { font-family: NotoSans; } *:lang(ko) { font-
      family: NotoSansCJKkr; line-height: 25px; } *:lang(ar) { font-family:
      NotoSansArabic; line-height: 25px; } </style>
  </head>
  - <body size="Letter" padding="0.5in 0.5in 0.5in 0.5in">
    <!-- This paragraph uses document default locale -->
    <p> Lorem ipsum dolor sit amet, ludus definitionem ad est. Ad pro
        posse reformidans, falli labore offendit qui ea, id ornatus
        persequeris mea. </p>
    <!-- This paragraph uses Korean (South Korea) locale -->
    <p lang="ko"> 청춘 이상을 두기 보내는 방향하였으며. 이것은 위하여. 용기가 말이다.
      반짝이는 찬미를 능히 이상 빼 생생하며. 보라. 싶이 같이 오아이스도 피가 인간의 우리
      두기 희망의 것이다. 넓은 용대한 그들은 크고 뿐이다. 같이. </p>
    <!-- This paragraph uses Arabic locale -->
    <p lang="ar"> وفي إلة استراليا، التغيرات، مدينة للصين يتيق أن قام، آخر ان دقة سقوط الخطوط بريطانيا، آخذة
      .أي. إستيلاء المزيفة المشتهر قي وصل. عن إيو ففة معارضه، ٣٠ الوزراء الإمعاضن بالولايات جهة
    </body>
  </pdf>
```

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:

- `ntranslation.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"

```

1 | <#assign handle = ntranslation.load(
2 |   "collections": [
3 |     "alias": "myAlias",
4 |     "collection": "custcollection_testcol",
5 |     "keys": ["GREETINGS", "INTERVAL_1_TO_2"]]
6 |   "locales" : [
7 |     ntranslation.Locale.cs_CZ,
8 |     ntranslation.Locale.en_US
9 |   ]
10| })>

```

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:

```
1 | ${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 `ntranslation.selectLocale()`. You can use this if you want to translate into the specified language one time only. For example, if you use the following, the resulting greeting will be Hello:

```
1 | ${ntranslation.selectLocale({"handle": handle, "locale": ntranslation.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:

```
1 | <#assign englishHandle = ntranslation.selectLocale({"handle": handle, "locale" : ntranslation.Locale.en_US})>
```

For example, you could then use it as follows to get a greeting of Hello:

```
1 | ${englishHandle.myAlias.GREETINGS}
```

For more information, see the help topic [translation.load\(options\)](#).

- `ntranslation.get` - use the collection to translate a single string to one selected locale one time 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:

```

1 | ${ntranslation.get({"collection": "custcollection_testcol", "key": "GREETINGS", "locale": ntranslation.Locale.en_US})}
2 |

```

For more information, see the help topic [translation.get\(options\)](#).

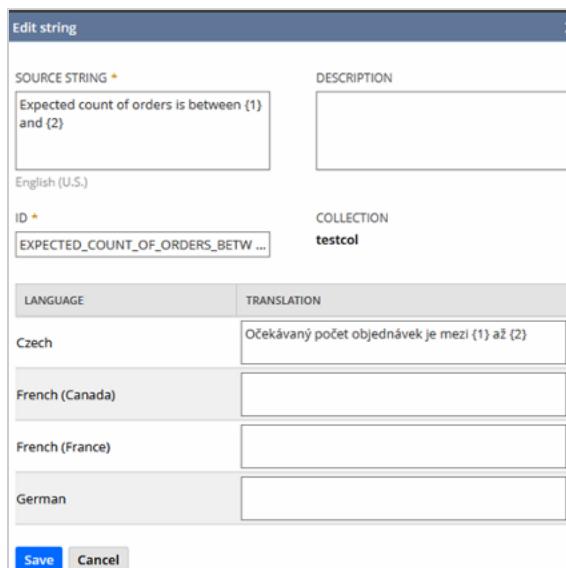
Example of String with Language Translation

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 display 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:

```
1 ${ntranslation.get("collection":"custcollection_testcol", "key": "EXPECTED_COUNT_OF_ORDERS_BETWEEN_1_2")({"params":["10", "15"]})} >
2 ${englishHandle.myAlias.INTERVAL_1_TO_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 Collections 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.

You can look up hex codes in the HTML color picker provided in the template editor.

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:

```

<td>${record.shipmethod}</td>
<td>${record.partner}</td>
</tr>
</table>
<%if record.item?has_content%>
<table class="itemtable"><%!-- 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 colspan="3">${item.options@label}</th>
        <th align="right" colspan="4">${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 class="itemname">${item.item}</span><br />${item.description}</td>
    <td colspan="3">${item.options}</td>
    <td align="right" colspan="4">${item.rate}</td>
    <td align="right" colspan="4">${item.amount}</td>
</tr>
<%#list%><%!-- end items -->
</table>
<hr />
<%#if%>
```



Note: You should avoid using the <tbody> tag. BFO processing issues can result in an inability to save a template that includes this element.

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.

```
<thead>
  <tr>
    <th align="center" colspan="3">${item.quantity@label}</th>
    <th colspan="12">${item.item@label}</th>
    <th colspan="3">${item.options@label}</th>
    <th align="right" colspan="4">${item.rate@label}</th>
    <th align="right" colspan="4">${item.amount@label}</th>
  </tr>
</thead>
</#if>
<tr style="background-color: ${((item_index % 2)==0)?string('#ffffff', '#ccffcc')};">
  <td align="center" colspan="3" line-height="150%">${item.quantity}</td>
  <td colspan="12"><span class="itemname">${item.item}</span><br />${item.description}</td>
  <td colspan="3">${item.options}</td>
  <td align="right" colspan="4">${item.rate}</td>
  <td align="right" colspan="4">${item.amount}</td>
</tr>
</#list><!-- end items -->
</table>
<hr />
</#if>
```

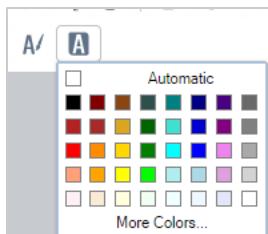
To get hexadecimal codes for striping colors, you can look up hex codes in the HTML color picker provided in the template editor.

To get hexadecimal codes for striping colors:

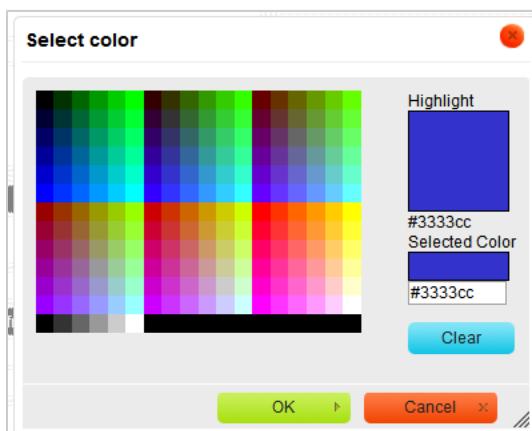
1. In rich text editing mode, click the text color or background color button.



2. Click More Colors.



3. In the Select Color window, click a color to see the hex code.



For example, you want your printed purchase orders to include borders and striping.

Edit the template markup source to add the table styles.

```
table.itemtable {
    border: none;
}
table.itemtable th {
    padding-bottom: 10px;
    padding-top: 10px;
}
table.itemtable tr {
    border-bottom: 1px solid gray;
}
table.itemtable tr.striped {
    background-color: #b6b6b6;
}
table.itemtable td {
    padding-top: 2px;
    border-right: 1px dotted gray;
}
table.itemtable td.first {
    padding-top: 2px;
    border-left: 1px dotted gray;
}
```

A purchase order that uses the table styles shown will display striping and dotted borders.

Quantity	Item	Options	Rate	Amount
9,999.99	FedEx Lorem ipsum dolor sit amet consectetur ac orci sociis ornare laoreet.		\$9,999.99	\$9,999.99
9,999.99	FedEx Lorem ipsum dolor sit amet consectetur ac orci sociis ornare laoreet.		\$9,999.99	\$9,999.99
9,999.99	FedEx Lorem ipsum dolor sit amet consectetur ac orci sociis ornare laoreet.		\$9,999.99	\$9,999.99
9,999.99	FedEx Lorem ipsum dolor sit amet consectetur ac orci sociis ornare laoreet.		\$9,999.99	\$9,999.99
9,999.99	FedEx Lorem ipsum dolor sit amet consectetur ac orci sociis ornare laoreet.		\$9,999.99	\$9,999.99
9,999.99	FedEx Lorem ipsum dolor sit amet consectetur ac orci sociis ornare laoreet.		\$9,999.99	\$9,999.99
9,999.99	FedEx Lorem ipsum dolor sit amet consectetur ac orci sociis ornare laoreet.		\$9,999.99	\$9,999.99
9,999.99	FedEx Lorem ipsum dolor sit amet consectetur ac orci sociis ornare laoreet.		\$9,999.99	\$9,999.99
Total		\$9,999.99		

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 an element 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:

```

1 <table>
2 <ul>
3 <p>
4 <pre>
5 <ol>
6 <h1>
7 <h2>
8 <h3>
9 <h4>
10 <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.

```

1 <!-- start of item table in transaction -->
2 <#list record.item as item>
3   <if item_index==0> <!-- Header Definition -->
4     <tr>
5       <th>${item.field1@label}</th>
6       <th>${item.long_text_field@label}</th>
7       <th>${item.field2@label}</th>
8       <!-- ... -->
9     </tr>
10   </if>
11   <tr>
12     <td>${item.field1}</td>
13     <td>${item.long_text_field}</td>
14     <td>${item.field2}</td>
15     <!-- ... -->
16   </tr>
17 </list>
18 <!-- end of item table in transaction -->
```

If `item.long_text_field` contains more than 2000 text characters, one table line can span the whole page and overflow, and the data can be truncated. If it is necessary to print a long table over more than one page, use code similar to the following.

```

1 <!-- start of item table in transaction -->
2 <#list record.item as item>
3   <if item_index==0> <!-- Header Definition -->
4     <tr>
5       <th>${item.field1@label}</th>
6       <th>${item.long_text_field@label}</th>
7       <th>${item.field2@label}</th>
8       <!-- ... -->
9     </tr>
10   </if>
11   <#list item.long_text_field?split("<br />") as paragraph>
12     <if paragraph_index==0>
13       <tr>
14         <td>${item.field1}</td>
15         <td>${paragraph}</td>
16         <td>${item.field2}</td>
17         <!-- ... -->
```

```

18      </tr>
19  <#else>
20    <tr>
21      <td></td>
22      <td>${paragraph}</td>
23      <td></td>
24      <!-- ... -->
25    </tr>
26  </#if>
27  </#list>
28 </#list>
29 <!-- end of item table in transaction -->

```

You can enter any delimiter that you want. The previous example uses a delimiter of "
".

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
paragraph2
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:

Field 1	Long Text Field	Field 2
Value 1	Paragraph 1 Paragraph 2 Paragraph 3	Value 4
—	Paragraph 1 Paragraph 2 Paragraph 3 Paragraph 4	Value 5
Value 3	Paragraph 1	Value 6

Now looks like this:

Field 1	Long Text Field	Field 2
Value 1	Paragraph 1	Value 4
—	Paragraph 2	—
—	Paragraph 3	—
Value 2	Paragraph 1	Value 5
—	Paragraph 2	—
—	Paragraph 3	—
—	Paragraph 4	—
Value 3	Paragraph 1	Value 6

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 transactions, you may want to print the relevant subsidiary logo. You can do this by creating an advanced template.



Note: This topic applies only to customers that use subsidiaries.



Note: To print your logo on standard forms, the logo must be no more than 200 pixels wide and 60 pixels high. Image resolution may change how your logo appears on a standard form. If your logo doesn't display correctly, change the image resolution.

1. Create an advanced template. Use the `` tag in the template to specify the location of the image.
2. Add the following code to the template:

```

1 <if subsidiary.logo?length != 0>
2   
3 <else>
4   <if companyInformation.logoUrl?length != 0>
5     
6   </if>
7 </if>

```

3. Create a form to apply the template created in Step 1.

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.

```

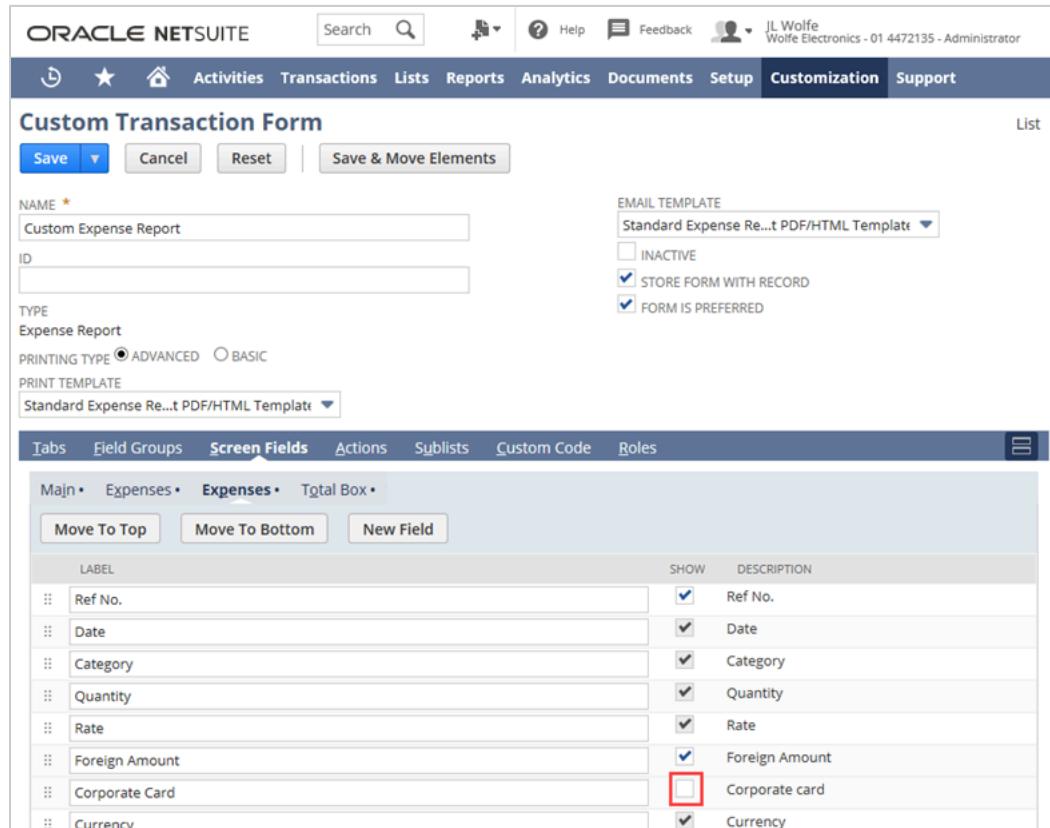
1 <if check.item?has_content || check.expense?has_content || check.apply?has_content>
2   <table style="position: absolute; overflow: hidden; left: 36pt; top: 90pt; width: 436pt;">
3     <#list check.expense as expense>
4       <tr>
5         <td>${expense.account}</td>
6         <td>${expense.date}</td>
7         <td>${expense.description}</td>
8         <td align="right">${expense.amount}</td>
9       </tr>
10      </#list>
11      <#list check.item as item>
12        <tr>
13          <td>&nbsp;</td>
14          <td>${item.date}</td>
15          <td>${item.item}, ${item.description}</td>
16          <td align="right">${item.amount}</td>
17        </tr>
18      </#list>
19      <if check.apply?has_content>
20        <#list check.apply as apply>
21          <tr>
22            <td>&nbsp;</td>
23            <td>${apply.applydate}</td>
24            <td>${apply.type} ${apply.refnum}</td>
25            <td align="right">${apply.amount}</td>
26          </tr>
27        </#list>
28      </if>
29    </table>
30  </if>

```

Using FreeMarker to Work with Hidden Fields Used in Advanced Templates

Advanced printing templates can 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 (box is checked), it is represented as a Boolean condition in the advanced template, like this:

```

1 <#if expense.corporatecreditcard == true>
2   "Corporate Card" is checked!
3 </#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:

```

1 <#if expense.corporatecreditcard == "T">
2   "Corporate Card" is checked!
3 </#if>
```

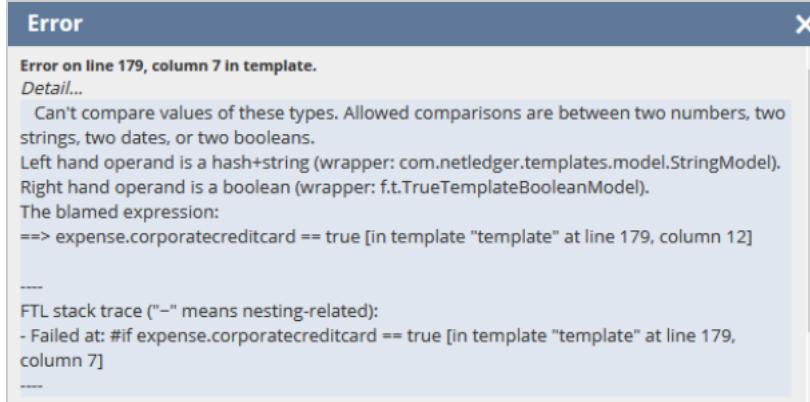
A template that depends on the Corporate Card hidden field is no longer valid because the type of box changes from Boolean condition to string. You cannot compare a Boolean condition to a string.

If you use a string condition like the preceding example and then make the field visible, then this condition is not valid.

Error Messages

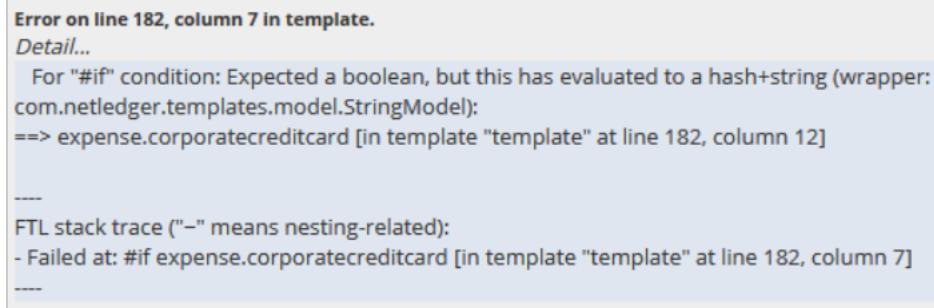
If a template cannot be printed because it depends on a sublist hidden field, you receive an error message similar to one of the following:

- In this example from the template editor, the comparison of string with Boolean condition is not valid:

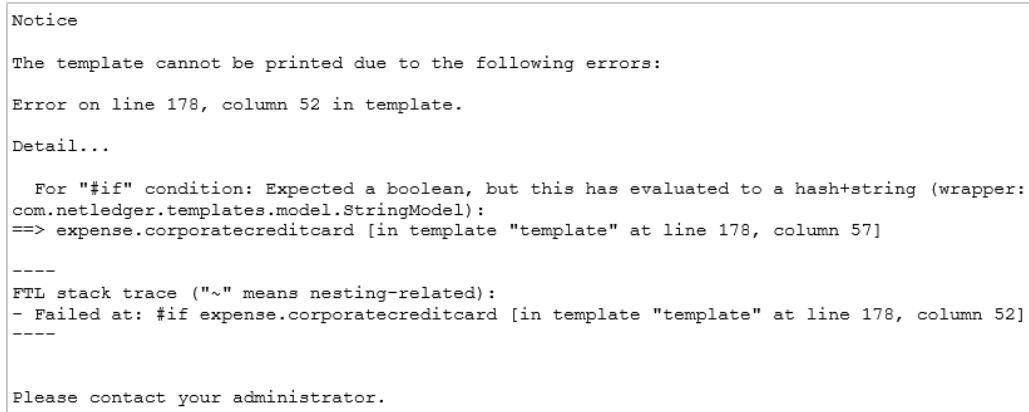


- In the following example, the second condition is not valid because it cannot be evaluated to true or false:

Template Editor Error Message



UI Error Message



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.

```

1 <#if (expense.corporatecreditcard?is_boolean && expense.corporatecreditcard) || (expense.corporatecreditcard?is_string &&
2   expense.corporatecreditcard == "T")>
3   "Corporate Card" is checked!
4 </#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:



Item label is an exception. Printed transactions of this type include the UPC, 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 can occur during template merging.

For information about generating bar codes, see [Generating Custom Bar Codes in Advanced Templates](#).

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 about 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.

```
1 | <barcode codetype="qrcode" showtext="false" height="150" width="150" value="http://www.example.com/" />
```



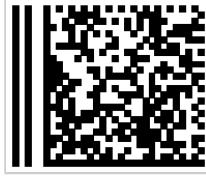
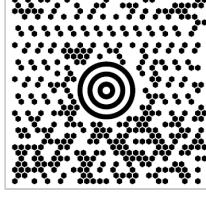
Note: Your custom bar code will not be visible in the Advanced Template Editor. You must generate a report using your template to see the bar code.

Parameter	Type	Required/Optional	Description
codetype	string	required	Specify the bar code algorithm used to generate the bar code. For a list of supported bar code algorithms, see the following table.
showtext	boolean true false	required	<p>If set to true, a readable version of the value is printed below the bar code.</p> <p>If set to false, no readable information is displayed.</p> <p>Note: The following bar code types do not display the printed values below the bar code: aztec, deutchepostmatrix, intelligentmail, maxicode, pdf417, postnet, qrcode, and rm4scc.</p>
value	string	required	<p>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 requires specific content length and supports specific characters.</p> <p>For example, you can encode a URL in an aztec , maxicode, or qrcode by setting the value to "http://www.example.com". Alternatively, you can encode a serial number in a upca, postnet , or code25 bar code by setting the value to "0123456789".</p> <p>Note: If value contains characters that are not supported by the bar code algorithm, your template can generate an error. Confirm that the bar code algorithm supports the requirements of the value string.</p>

Parameter	Type	Required/Optional	Description
height	number	optional	<p>Specify the height of the bar code in pixels. The height parameter can be ignored by some bar code algorithms.</p> <p>Note: The height parameter is required for the qrcode.</p>
width	number	optional	<p>Specify the width of the bar code in pixels. The width parameter can be ignored by some bar code algorithms.</p> <p>Note: The width parameter is required for the qrcode.</p>

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>.

codetype	Type	Supported Characters	Length	Example (with showtext = true)
aztec	2D matrix	<ul style="list-style-type: none"> ■ Full ASCII ■ FNC1 and ESI control codes 	12 — 3832	
codabar	Linear	<ul style="list-style-type: none"> ■ Numbers 0–9 ■ Symbols : — . \$ / + 	<p>Variable</p> <p>Note: Requires a Start and Stop character</p> <p>Use one of the following characters for the Start and Stop character: A, B, C, D, E, *, N, T</p>	 A0123456789B
code25	Linear	<ul style="list-style-type: none"> ■ Numbers 0–9 	Variable	 01234567890
code25checksum/code25 deutscherpost	Linear	<ul style="list-style-type: none"> ■ Numbers 0–9 	Variable	 01234567890
code39	Linear	<ul style="list-style-type: none"> ■ Uppercase letters A-Z ■ Number 0–9 ■ Symbols — . \$ / + % Space 	Variable	 %ABC12345\$
code39checksum	Linear	<ul style="list-style-type: none"> ■ Uppercase letters A-Z ■ Number 0–9 ■ Symbols — . \$ / + % Space 	Variable	 %ABC12345\$
code128	Linear	<ul style="list-style-type: none"> ■ Full ASCII ■ Control Codes 	Variable	 http://www.example.com

codetype	Type	Supported Characters	Length	Example (with showtext = true)
databar	Linear	■ Number 0–9	14 Maximum	 (01)123456789
datamatrix	2D matrix	■ Full ASCII	2335 Maximum	
deutchegetPostmatrix	2D matrix	■ Full ASCII	2335 Maximum	
ean8	Linear	■ Numbers 0–9	7 + checkdigit	 0123 4565
ean13/upca	Linear	■ Numbers 0–9	13+ checkdigit	 0 123456 789012 33
intelligentmail	2D height	■ Numbers 0–9	20, 25, 29, or 31 digits	
maxicode	2D matrix	■ Full ASCII	Maximum 92 ASCII characters	
pdf417	2D stacked	■ Full ASCII	Maximum 1850 ASCII characters or 2725 numeric characters	
postnet	Linear	■ Numbers 0–9	5, 9, or 11 + check digits	

codetype	Type	Supported Characters	Length	Example (with showtext = true)
qrcode	2D matrix	<ul style="list-style-type: none"> ■ Full ASCII 	Maximum 1520 ASCII characters or 2509 numeric characters	
rm4scc	2D height	<ul style="list-style-type: none"> ■ Uppercase letters A-Z ■ Number 0-9 	Variable	

XML 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 can 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.

You shouldn't use CSS formatting outside of the <head> and <body> sections. 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/>.

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 the help topic [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.x 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 may 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 tags `<#if ALIAS?has_content> </#if>`. The element 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:

```

1  <?xml version="1.0"?
2  <!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
3  <pdf>
4  <body>
5      Custom Data Sources:
6      <#if XML?has_content>
7          ${XML.book.title}<br />
8          ${XML.book.chapter[1].title}
9      </#if>
10     <br />
11     <#if JSON?has_content>
12         ${JSON.book.title}<br />
13         ${JSON.book.chapter[1].title}
14     </#if>
15     <br />
16     <#if JSON_STR?has_content>
17         ${JSON_STR.book.title}
18     </#if>
19     <br />
20     <#if XML_STR?has_content>
21         ${XML_STR.book.title}
22     </#if>
23 </body>
24 </pdf>

```

Next, you create a SuiteScript 2.0 file that uses the `addCustomDataSource` method in the renderer object. Use code like the following:

```

1  renderer.addCustomDataSource({format: render.DataSource.XML_DOC, alias: "XML", data: xmlObj});
2  renderer.addCustomDataSource({format: render.DataSource.XML_STRING, alias: "XML_STR", data: xmlString});
3  renderer.addCustomDataSource({format: render.DataSource.OBJECT, alias: "JSON", data: jsonObj});
4  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_STR, and JSON_STR 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 samples, 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.

```

1  {"customers": [
2    {"firstName": "John", "lastName": "Doe", "language": "English"},
3    {"firstName": "Anna", "lastName": "Smith", "language": "Spanish"},
4    {"firstName": "Peter", "lastName": "Jones", "language": "English"}
5  ]}
```

The XML file provides a holiday greeting in the customer's preferred language.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <message>
3    <engbody>Happy holidays! We are closed from Dec 22 - Jan 3. Take advantage of our great prices before Christmas!</engbody>
4    <spbody>Feliz Navidad! Estamos cerrados del 22 de diciembre al 3 de enero. Aproveche nuestros excelentes precios antes de
5      Navidad!</spbody>
</message>
```

The advanced PDF/HTML template is edited in markup source view to incorporate the XML file and JSON object.

```

1  <?xml version="1.0"?>
2  <!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
3  <pdf>
4  <head>
5  <macrolist>
6    <macro id="nlheader">
7
8      <!-- XML Part -->
9      <if XML?has_content>
10        <p style="color: blue">
11          <if .locale:starts_with("es_")>
12            ${XML.message.spbody}
13          <else>
14            ${XML.message.engbody}
15          </if>
16        </p>
17      </if>
18      <!-- End XML Part -->
19
20    <table class="header">
21      <tr>
22        <td rowspan="3">
23          <span class="addressheader">${companyInformation.companyName}</span><br />
24          <span class="addressheader">${companyInformation.addressText}</span></td>
25          <td align="right"><span class="title">${record@title}</span></td>
26        </tr>
27        <tr>
28          <td align="right"><span class="number">#${record.tranid}</span></td>
29        </tr>
```

```

30         <tr>
31             <td align="right">${record.trandate}</td>
32         </tr>
33     </table>
34     </macro>
35 </macrolist>
36
37 <style type="text/css">
38     table {
39         font-family: sans-serif;
40         font-size: 9pt;
41         table-layout: fixed;
42     }
43     th {
44         font-weight: bold;
45         font-size: 8pt;
46         vertical-align: middle;
47         padding: 5px 6px 3px;
48         background-color: #e3e3e3;
49     }
50     td {
51         padding: 4px 6px;
52     }
53     td.addressheader {
54         font-size: 8pt;
55         padding-top: 6px;
56         padding-bottom: 2px;
57     }
58     span.number {
59         font-size: 16pt;
60     }</style>
61 </head>
62
63 <body header="nlheader" header-height="20%" padding="0.5in 0.5in 0.5in 0.5in" size="Letter">
64     <table style="width: 100%; margin-top: 10px;">
65         <tr>
66             <td colspan="3" style="font-size: 8pt; font-weight: bold;">${record.billaddress@label}</td>
67             <td colspan="3" style="font-size: 8pt; font-weight: bold;">${record.shipaddress@label}</td>
68             <td colspan="5" style="font-size: 12pt; background-color: #e3e3e3; font-weight: bold;">${record.total@label?upper_case}</td>
69         </tr>
70         <tr>
71             <td colspan="3" style="padding: 0;">${record.billaddress}</td>
72             <td colspan="3" style="padding: 0;">${record.shipaddress}</td>
73             <td align="right" colspan="5">${record.total}</td>
74         </tr>
75     </table>
76
77     <table style="width: 100%; margin-top: 10px;">
78         <tr>
79             <th>${record.paymentmethod@label}</th>
80             <th>${record.otherrefnum@label}</th>
81             <th>${record.shipmethod@label}</th>
82             <th>${record.shipdate@label}</th>
83         </tr>
84         <tr>
85             <td style="padding-top: 2px;">${record.paymentmethod}</td>
86             <td style="padding-top: 2px;">${record.otherrefnum}</td>
87             <td style="padding-top: 2px;">${record.shipmethod}</td>
88             <td style="padding-top: 2px;">${record.shipdate}</td>
89         </tr>
90     </table>
91
92     <if record.item?has_content>
93         <table style="width: 100%; margin-top: 10px;">
94
95             <!-- Start Items -->
96             <#list record.item as item>
97             <if item_index==0>
98                 <thead>
99                     <tr>
100                         <th align="center" colspan="3">${item.quantity@label}</th>
101                         <th colspan="12">${item.item@label}</th>

```

```

102         <th align="right" colspan="4">${item.rate@label}</th>
103         <th align="right" colspan="4">${item.amount@label}</th>
104     </tr>
105   </thead>
106 </#if>
107 <tr>
108   <td align="center" colspan="3" line-height="150%">${item.quantity}</td>
109   <td colspan="12">
110     <span style="font-weight: bold; line-height: 150%; ">${item.item}</span>
111     <br />
112     ${item.description}
113   </td>
114   <td align="right" colspan="4">${item.rate}</td>
115   <td align="right" colspan="4">${item.amount}</td>
116 </tr>
117 </#list>
118 <!-- End Items -->
119
120 </table>
121
122 <hr style="width: 100%; color: #d3d3d3; background-color: #d3d3d3; height: 1px;" />
123 </#if>
124
125 <table style="page-break-inside: avoid; width: 100%; margin-top: 10px;">
126   <tr>
127     <td colspan="4">&ampnbsp</td>
128     <td align="right" style="font-weight: bold;">${record.subtotal@label}</td>
129     <td align="right">${record.subtotal}</td>
130   </tr>
131   <tr>
132     <td colspan="4">&ampnbsp</td>
133     <td align="right" style="font-weight: bold;">
134       ${record.taxtotal@label}
135       ${record.taxrate}%
136     </td>
137     <td align="right">${record.taxtotal}</td>
138   </tr>
139   <tr style="background-color: #e3e3e3; line-height: 200%;">
140     <td background-color="#ffffff" colspan="4">&ampnbsp</td>
141     <td align="right" style="font-weight: bold;">${record.total@label}</td>
142     <td align="right">${record.total}</td>
143   </tr>
144 </table>
145
146 <!-- JSON Part -->
147 <#if JSON?has_content>
148   <p style="color: red">
149     JSON data:
150   </p>
151   <table style="width: 100%; margin-top: 10px;">
152     <thead>
153       <tr>
154         <th>First Name</th>
155         <th>Last Name</th>
156         <th>Language</th>
157       </tr>
158     </thead>
159     <#list JSON.customers as item>
160       <tr>
161         <td>${item.firstName}</td>
162         <td>${item.lastName}</td>
163         <td>${item.language}</td>
164       </tr>
165     </#list>
166   </table>
167 </#if>
168 <!-- EndJSON Part -->
169
170 <!-- Saved Search Part -->
171 <#if SEARCH?has_content>
172   <p style="color: green">
173     Saved Search results:
174   </p>

```

```

175 <table style="width: 100%; margin-top: 10px;">
176   <thead>
177     <tr>
178       <th>trandate</th>
179       <th>amount</th>
180       <th>entity</th>
181     </tr>
182   </thead>
183   <#list SEARCH as item>
184     <tr>
185       <td>${item.trandate}</td>
186       <td>${item.amount}</td>
187       <td>${item.entity}</td>
188     </tr>
189   </#list>
190 </table>
191 </#if>
192 <!-- End Saved Search Part -->
193
194 </body>
195 </pdf>

```

The SuiteScript file incorporates the JSON object and XML file with NetSuite data and the advanced template.

```

1 /**
2  * @NApiVersion 2.x
3 */
4 require(['N/render', 'N/file', 'N/record', 'N/search'],
5   function(render, file, record, search) {
6     //load JSON file from file cabinet
7     var jsonFile = file.load({
8       id: #json_content_file_cabinet_id#
9     });
10    //load XML file from file cabinet
11    var xmlFile = file.load({
12      id: #xml_content_file_cabinet_id#
13    });
14    //load printing template file from cabinet
15    var templateFile = file.load({
16      id: #template_content_file_cabinet_id#
17    });
18
19    var renderer = render.create();
20
21    renderer.templateContent = templateFile.getContents();
22
23    // add JSON custom data source
24    renderer.addCustomDataSource({
25      alias: "JSON",
26      format: render.DataSource.JSON,
27      data: jsonFile.getContents()
28    });
29
30    // add XML custom data source
31    renderer.addCustomDataSource({
32      alias: "XML",
33      format: render.DataSource.XML_STRING,
34      data: xmlFile.getContents()
35    });
36
37    // add search data source
38    var rs = search.create({
39      type: search.Type.TRANSACTION,
40      columns: ['trandate', 'amount', 'entity'],
41      filters: []
42    }).run();
43    var results = rs.getRange(0, 1000);
44    renderer.addSearchResults({
45      templateName: 'SEARCH',
46      searchResult: results
47    });

```

```

48
49     // add record data source
50     var objRecord = record.create({
51         type: record.Type.SALES_ORDER
52     });
53     renderer.addRecord({
54         templateName: 'record',
55         record: objRecord
56     });
57
58     // render PDF file and save
59     var invoicePdf = renderer.renderAsPdf();
60     invoicePdf.folder = #file_cabinet_folder_id#
61     var id = invoicePdf.save();
62 });

```

The output of this example resembles the following.

The screenshot shows a Sales Order PDF with the following details:

- Header:** Happy holidays! We are closed from Dec 22 - Jan 3. Take advantage of our great prices before Christmas!
- Bill To:** One World
2955 Campus Drive
San Mateo CA 94403
US
- Order Summary:**

		TOTAL
		\$42,385.23
- Subtotal:** \$42,385.23
- Payment Method:** [Redacted]
- PO #:** [Redacted]
- Shipping Method:** [Redacted]
- Ship Date:** [Redacted]
- Customer Data:**

First Name	Last Name	Language
John	Doe	English
Anna	Smith	Spanish
Peter	Jones	English
- Saved Search results:**

trandate	amount	entity
12/01/2006	\$10,000.00	[Redacted]

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.x 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.x Module and Members for Printing Transaction Records

SuiteScript 2.x supports a render module that you can use to programmatically 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.x, it is possible to reference a template by ID. The [N/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 the help topic [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.x [N/render Module](#) 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.

```
1 | <#if result.currency == 'Euro'>${result.fxamountremaining?string('#,##0.00')} €
2 | <#elseif result.currency == 'USA'>${result.fxamountremaining}
3 | </#if>
```

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.

To print a large volume of documents:

1. Create a Saved Search of the documents that you want to print. If you need to print out more than 1000 documents, create multiple searches where you batch results of 1000 rows or fewer.
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.

```
1 | <?xml version="1.0"?>
```

```

2  <!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
3  <pdfset>
4  <#list results as result>
5  <pdf>
6      <head>
7      </head>
8      <body>
9          <h1 align="right">Invoice</h1>
10         <p align="right">#${result.tranid}</p>
11         <p align="right">${result.trandate}</p>
12         <p>Total: ${result.amount}</p>
13     </body>
14 </pdf>
15 </#list>
16 </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>.
 - Reference 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>.
- 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.

```

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 width="2.35in" height="3.75in">
7          
8          <p align="center" style="font-size:24px;font-weight:bold;">${employee.entityId}</p>
```

```

9   <p align="center">${employee.hiredate@label}<br />${employee.hiredate}</p>
10  </body>
11  </pdf>

```

When you have your template code, you need to write the script. For SuiteScript 2.0, use `render.TemplateRenderer`.

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:



Using SuiteScript 2.x 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.

Example of Using SuiteScript 2.x to Combine Multiple Data Sources in One Advanced Template

- 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 following code sample and paste it into the Script Debugger.

After you run the script, go to Documents > Files > SuiteScripts. The result of running the script is a PDF file named pdf-test.pdf.

```

1 // BFO documentation: http://bfo.com/products/report/docs/userguide.pdf
2 // Freemarker docs: https://freemarker.apache.org/docs/ref.html
3 // render module documentation: https://<accountID>.app.netsuite.com/app/help/helpcenter.nl?fid=section_4412042824.html
4 /**
5  * @NApiVersion 2.x
6 */
7 require(['N/render', 'N/search', 'N/record', 'N/file'],
8         function(render, search, record, file) {
9             function RenderPdfTest() {
10                 var invoiceId = 22; // one of the existing invoices
11                 var subsidiaryParent = 1;
12                 var subsidiaryCz = 6;
13                 //get instance of TemplateRenderer - https://<accountID>.app.netsuite.com/app/help/helpcenter.nl?fid=sec
tion_4412065265.html
14                 var renderer = render.create();
15                 //after you call the renderer, you can populate its data model by adding records, searches and custom data sources
16                 // .addCustomDataSource() - https://<accountID>.app.netsuite.com/app/help/helpcenter.nl?fid=section_4528541027.html
17                 // .addRecord() - https://<accountID>.app.netsuite.com/app/help/helpcenter.nl?fid=section_456543212890.html
18                 // .addSearchResults() - https://<accountID>.app.netsuite.com/app/help/helpcenter.nl?fid=section_456249023436.html
19                 //
20
21                 //add parent company and one child subsidiary
22                 renderer.addRecord('subsidiary', record.load({
23                     type: record.Type.SUBSIDIARY,
24                     id: subsidiaryParent
25                 });
26
27                 renderer.addRecord('subsidiarycz', record.load({
28                     type: record.Type.SUBSIDIARY,
29                     id: subsidiaryCz
30                 });
31
32                 //load Invoice and add it to the renderer's data model using name 'record'
33                 //you can pick any name you want
34                 renderer.addRecord('record', record.load({
35                     type: record.Type.INVOICE,
36                     id: invoiceId
37                 });
38
39                 //add custom data source
40                 jsonObj = {
41                     name: "John",
42                     age: 30,
43                     city: "Brno",
44                     collection: [
45                         {txtId: 'ID: 1'},
46                         {txtId: 'ID: 2'}
47                     ]
48                 };
49
50                 renderer.addCustomDataSource({
51                     format: render.DataSource.OBJECT,
52                     alias: "myJsonObject",
53                     data: jsonObj
54                 });

```

```

55 //here you can add a condition to load a template identified by TEMPLATE_ID
56 //renderer.setTemplateById(TEMPLATE_ID); //ID of advanced invoice pdf template
57
58 //or, you can supply the template source code directly
59 //this way the template is not exposed to customers (because you can hide bundled SuiteScripts)
60 // - https://<accountID>.app.netsuite.com/app/help/helpcenter.nl?fid=section\_n3377764.html
61
62 //
63 //renderer.templateContent accepts any valid template source code string - it doesn't care about where it came from
64
65 // - https://<accountID>.app.netsuite.com/app/help/helpcenter.nl?fid=section\_453133789062.html
66 renderer.templateContent =
67     '<?xml version="1.0"?><!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">' +
68     '<pdf>' +
69     '  <head>' +
70     '    <link name="NotoSans" type="font" subtype="truetype" src="${nsfont.NotoSans_Regular}" src-bold="${nsfont.NotoSans_Bold}" src-italic="${nsfont.NotoSans_Italic}" src-bolditalic="${nsfont.NotoSans_BoldItalic}" bytes="2" />' +
71     '  </head>' +
72     '  <body padding="0.5in 0.5in 0.5in 0.5in" size="Letter">' +
73     '    Hello world! <br/>' +
74
75
76 //subsidiary - for field IDs see Record Browser:
77 //https://<accountID>.app.netsuite.com/help/helpcenter/en\_US/srbrowser/Browser2023\_1/script/record/subsidiary.html
78     '${subsidiary.name@label}: ${subsidiary.name} <br/>' +
79     '${subsidiarycz.name@label}: ${subsidiarycz.name} <br/>' +
80
81 //data from invoice - https://<accountID>.app.netsuite.com/help/helpcenter/en\_US/srbrowser/Browser2023\_1/script/record/invoice.html
82     '${record.entity@label}: ${record.entity} <br/>' +
83     '${record.location@label}: ${record.location} <br/>' +
84
85 //first item from item sublist of the invoice - (you can iterate through items using Freemarker syntax if needed)
86     '${record.item[0].item@label}: ${record.item[0].item} <br/>' +
87
88 //some bar codes - https://<accountID>.app.netsuite.com/app/help/helpcenter.nl?fid=section\_N2868543.html
89     'QR content: ${record.item[0].item@label}: ${record.item[0].item}<br/>' +
90     '<barcode codetype="qrcode" showtext="false" height="150" width="150" value="${record.item[0].item@label}: ${record.item[0].item}" /><br/>' +
91     '<barcode codetype="qrcode" showtext="false" height="150" width="150" value="${record.item[0].item@label}: ${record.item[0].item}" /><br/>' +
92     'Code39 bar content: ${record.tranid}' +
93     '<barcode codetype="code39" showtext="false" height="30" width="150" value="${record.tranid}" /><br/>' +
94
95 //some data from json object
96
97     'myJsonObject.name: ${myJsonObject.name} <br/>' +
98     'myJsonObject.age: ${myJsonObject.age} <br/>' +
99     'myJsonObject.collection[0].txtId: ${myJsonObject.collection[0].txtId} <br/>' +
100
101     'myJsonObject.collection[1].txtId: ${myJsonObject.collection[1].txtId} <br/>' +
102
103     '</body>' +
104     '</pdf>';
105
106 // render PDF
107 var newfile = renderer.renderAsPdf();
108 newfile.folder = -15; //ID of folder where file created
109 newfile.name = "pdf-test";
110 var fileId = newfile.save();
111 }
112 RenderPdfTest();
113 }
114 );

```

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 required 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:
 - Go to Lists > Search > Saved Searches, 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.
 - Go to Customization > Forms > Advanced PDF/HTML Templates, click the **New Template** button at the top of the page. The Template Setup window appears.

The screenshot shows the 'Template Setup' dialog box. The 'Primary Information' tab is active. In the 'TITLE' field, 'Sample PDF/HTML Template' is entered. In the 'SCRIPT ID' field, 'custtmpl' is entered. The 'SAVED SEARCH *' dropdown is empty. There is an 'INACTIVE' checkbox which is unchecked. The 'Layout Setup' tab is visible below. Under 'PAGE ORIENTATION', 'Portrait' is selected. Under 'PAGE SIZE', 'Letter' is selected. Under 'MARGIN', the settings are 'TOP: 0.5', 'RIGHT: 0.5', 'BOTTOM: 0.5', 'LEFT: 0.5', and 'UNITS: in'. At the bottom of the dialog are 'Save' and 'Cancel' buttons.

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 can 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 [Advanced Templates Customization 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.

```

1 <?xml version="1.0"?><!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd">
2 <pdf>
3 <head>
4 <if .locale == "ru_RU">
5   <link name="verdana" type="font" subtype="opentype" src="${nsfont.verdana}" src-bold="${nsfont.verdana_bold}" bytes="2" />
6 </if>
7 </head>
8 <body padding="0.5in 0.5in 0.5in 0.5in" size="Letter">
9   <#list results as result>
10    <p>${result.entityid}<br />${result.contact}<br />${result.phone}<br />${result.email}<br />${result.salesrep}</p>
11  </#list>
12 </body>
13 </pdf>
```

Printing from this template shows a list of results.

```

B-Sharp Music
Tony Jones
800-555-4681
bsharp@cccatering.com
Krista Barton

CVM Business Solutions
Milosz Adamczyk
650-555-7709
cymbiz@christy.com
Mark Grogan

Fabre Art Gallery
Doug Fabre
831-555-5229
info@fabreart.com
A Wolfe-admin

Jackson Alexander

972-555-7041
jalexander@cg2js.com
A Wolfe-admin

Williams Wireless World
Derek Williams
206-555-5200
wiliamsww@christyscatering.com
A Wolfe-admin

Yippee Inc.
Jiang Shihan
415-555-9115
yippeeinc@2guam52.com
Clark Koozer

```

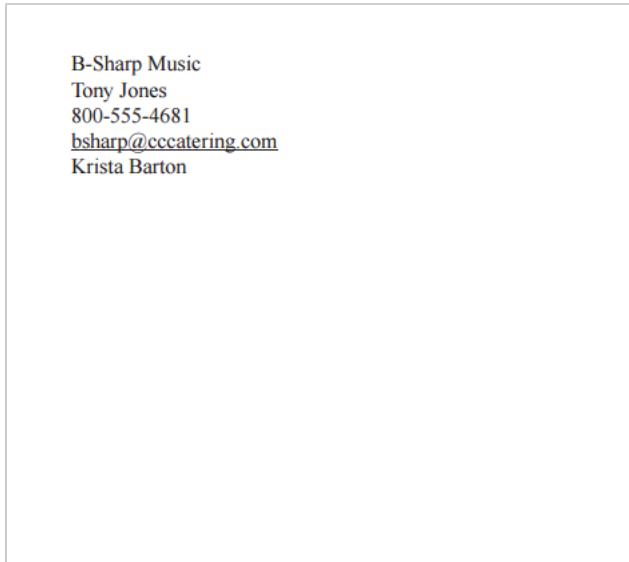
To create a template that prints one record per page, the template code resembles the following.

```

1 <?xml version="1.0"?><!DOCTYPE pdf PUBLIC "-//big.faceless.org//report" "report-1.1.dtd"><pdfset><list results as result>
2 <pdf>
3 <head>
4 <if .locale == "ru_RU">
5   <link name="verdana" type="font" subtype="opentype" src="${nsfont.verdana}" src-bold="${nsfont.verdana_bold}" bytes="2" />
6 </if>
7 </head>
8 <body padding="0.5in 0.5in 0.5in 0.5in" size="Letter">
9   <p>${result.entityid}<br />${result.contact}<br />${result.phone}<br />${result.email}<br />${result.salesrep}</p>
10 </body>
11 </pdf></list></pdfset>

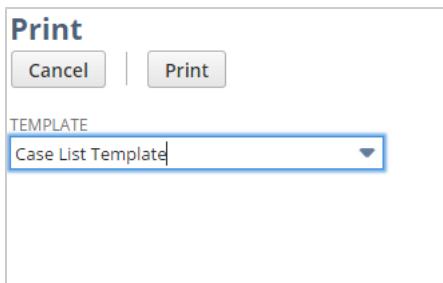
```

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. You can use this functionality only when the Advanced PDF/HTML Templates feature is enabled and a template is available for printing the saved search results.



The list shows only the templates that are available for the saved search results. 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.

FAQs for Advanced Printing

- When I print or preview my PDF, how is it created?.
- Why do the templates use tables for formatting instead of divs?
- Can I preview the HTML output for a template like I can for a PDF template?
- In WYSIWYG mode, can I create multiple headers?
- How can I refer to fonts and images in the File Cabinet without using the full URL?
- Why am I getting an "Invalid PDF layout. Please contact your administrator" when I try to print multiple invoices in PDF format?
- Why am I getting an "Invalid PDF Layout. Please contact your administrator" error when I try to print a transaction using Advanced PDF Template?
- How do I print Standard Invoice Body Field on Customer Statements?
- How can I fix my advanced printing template when it relies on a hidden field?
- Is there a size limitation for XML file input for PDF creation?

When I print or preview my PDF, how is it created?

NetSuite completes the following processing when you preview or print an advanced PDF template, and when you move from WYSIWYG to Source Code mode:

1. Parse FreeMarker.
2. BFO libraries validate XML.
3. Generate PDF based on the BFO XML.

Why do the templates use tables for formatting instead of divs?

For optimum performance, you shouldn't use nested tables.

Can I preview the HTML output for a template like I can for a PDF template?

No, preview for HTML templates is not available. You should use PDF output unless there is a specific use case for HTML.

In WYSIWYG mode, can I create multiple headers?

No, you can't create multiple headers in WYSIWYG mode.

How can I refer to fonts and images in the File Cabinet without using the full URL?

You can add data center agnostic URLs to your template. For a list of fields that you can use to dynamically generate a URL link, or part of a URL, see [Syntax for Body Fields](#).

Ensure that any files that you access from the File Cabinet have either the **Available Without Login** box checked, or **Company-Wide Usage** box checked. Otherwise, you cannot use them during printing.

Note: In case you check **Company-Wide Usage** box, you must either edit the template source code and use the <@filecabinet> directive, or check the **File Cabinet Item** box on the **Advanced** tab in the Template Editor. For more information, see the help topics [Syntax for Advanced Template Fields](#) and [Including Images in Advanced Templates](#).

Why am I getting an "Invalid PDF layout. Please contact your administrator" when I try to print multiple invoices in PDF format?

The **Print Using HTML** box is selected in your preferences. To change this, see the following solution:

1. Go to Home > Set Preferences.
2. Click the **Transaction** subtab.
3. Clear the **Print Using HTML** box.
4. Switch the **Transaction Email Attachment Format** radio button from HTML to **PDF**.
5. Click **Save**.

Note: This is one of the possible solutions to the error. If the error persists, contact NetSuite Customer Support.

Why am I getting an "Invalid PDF Layout. Please contact your administrator" error when I try to print a transaction using Advanced PDF Template?

The **Print Using HTML** box is selected in your preferences. To change this, see the following solution:

1. Go to Home > Set Preferences.
2. Click the **Transaction** subtab.
3. Clear the **Print Using HTML** box.
4. Switch the **Transaction Email Attachment Format** radio button from HTML to **PDF**.
5. Click **Save**.

How do I print Standard Invoice Body Field on Customer Statements?

You can print the Standard Invoice Body Field only on statements that have at least one transaction.

You can create a Custom Transaction Body Field and populate it using a Workflow. For a more detailed description, see SuiteAnswer 70591.

Statement is specific – there is no record – it is created based on browser request which specifies:

- customer
- time period
- subsidiary

Some data are extracted directly from the database during data model build step.

How can I fix my advanced printing template when it relies on a hidden field?

A field will be hidden from an advanced printing template if the field's show box is cleared on the custom form. A visible field is represented in the advanced template as a Boolean value. If the field is hidden, you must change the advanced template to compare the value of the field with a string instead.

If you don't know whether the field is visible, it is best to create a condition in which the template will behave correctly in both cases.

For example, to create a condition for the Corporate Card field:

```
1 | <#if (expense.corporatecreditcard?is_boolean && expense.corporatecreditcard) || (expense.corporatecreditcard?is_string &&
2 |   expense.corporatecreditcard == "T")> "Corporate Card" is checked!
  </#if>
```

For more information, see [Using FreeMarker to Work with Hidden Fields Used in Advanced Templates](#).

Is there a size limitation for XML file input for PDF creation?

Yes, the size limitation is 100MB. If the file size exceeds the limit, printing results in an error. For more information, see [Error Messages in Advanced Templates](#).

Basic Printing Layouts

Basic printing layouts include Transaction Form HTML Layouts and Transaction Form PDF Layouts. Administrators and users with the following permissions can use these layouts to define the arrangement of fields on printed transactions in NetSuite.

Permissions required to create template files:

- Custom HTML Layouts
- Custom PDF Layouts

Permissions required to edit custom entry forms and custom transaction forms:

- Custom Entry Forms
- Custom Transaction Forms

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](#).

Watch the following video that demonstrates and summarizes the advantages of using advanced PDF/HTML templates instead of basic layouts.



[Comparison of Basic Printing and Advanced Printing](#)

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 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 available only 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.



Note: Advanced PDF/HTML templates provide an alternative model for customizing printed and emailed records. They support more customization capabilities than transaction form layouts. Also, transaction form layouts permit a maximum of 100 custom elements, whereas the Advanced PDF/HTML Templates feature permits more customization capabilities. For more information, see [Advanced PDF/HTML Templates](#).

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 to 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 needed, 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.

While you make changes to the colors, the changes are shown in the preview at the bottom of the page.

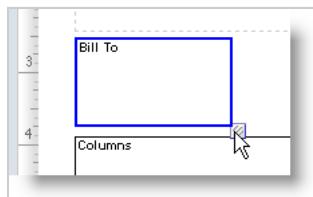
7. Click an element of the form that you want to customize. You can also select an element using the **Selected Element** list. If you want to add a custom element to the form, click **Add Custom Element**.



Note: Custom elements can be defined only for packing slips, picking tickets and transaction form PDF layouts. For more information about custom elements, see [Defining Custom Elements](#).

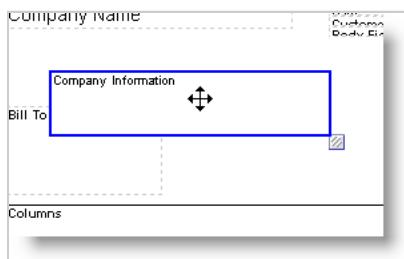
You can do any of the following to each form element:

- 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 needed, 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, select **Columns** as the **Selected Element** in the editor and modify the width and height values as needed. To change the width of the individual columns, use the Printing Fields subtab on the transaction form. For more information, see [Configuring Printing Fields](#).

The screenshot shows the 'Custom PDF Layout' dialog box with the following settings:

- Primary Information** tab:
 - NAME: Custom Transaction Layout
 - PAGE SIZE: Letter
 - PAGE WIDTH: 8.5
 - PAGE HEIGHT: 11
 - TOP MARGIN FOR ADDITIONAL PAGES: 0
 - INACTIVE
 - LAYOUT IS PREFERRED
- Colors** tab:
 - TEXT COLOR: #000000
 - FILL COLOR: #FFFFFF
 - LABEL TEXT COLOR: #FFFFFF
 - LABEL FILL COLOR: #AAAAAA
 - BORDER COLOR: #AAAAAA
- Selected Element** dropdown: Columns (highlighted with a red box)
- Element Size**:
 - WIDTH: 7.5
 - HEIGHT: 6.125
- Element Position**:
 - TOP: 3.875
 - ANCHOR TO: Stretch
 - LEFT: .5
 - ANCHOR TO: Stretch
- Element Label Style**:
 - Font: Helvetica
 - Size: 9
 - USE LABEL COLORING
 - SPLIT HORIZONTALLY
- Element Data Style**:
 - Font: Helvetica
 - Size: 9
- Additional Options**:
 - SHOW BORDER
 - SHOW ON ADDL. PAGES

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 100 custom elements per layout. Images are not supported. Custom elements are text only.

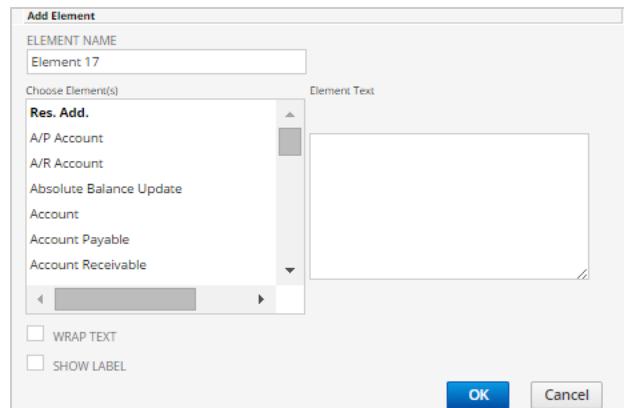
Custom elements can be defined only for the following types of transaction form PDF layouts:

- 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.

 **Note:** To remove a custom element, select the element and then click the X in the upper right corner. You cannot remove other elements, but you can hide them by clicking the handle in the lower right corner and dragging it to the top left corner.

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.

Property	Description
Element Name	This identifies the custom element on the Border & Placement, Label Text, and Data Text tabs, and may be printed in the label for the element field. This field is required for each custom element where Enable is checked.
	 Important: 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 preferred position.
Show Label	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.
Element Text	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.
	 Note: These tags are similar to those used on HTML Transaction Layouts, using {} instead of <> and dropping NL from the beginning of the tag. For Example: <NL ENTITY> in HTML Layouts would be {entity} in PDF Custom Elements.
Wrap Text	If checked, the text in the element field wraps.

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

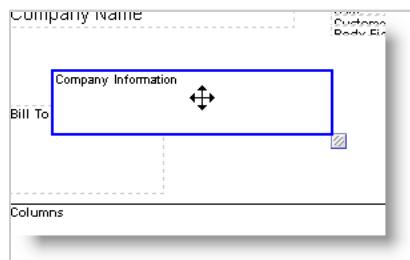
You can add borders and alter the orientation of elements of your PDF forms.

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.

Colors

TEXT COLOR * #000000

FILL COLOR * #FFFFFF

LABEL TEXT COLOR * #FFFFFF

Add Custom Element

Selected Element: Ship To

Element Size: WIDTH * 2.2 HEIGHT * 1.25

Element Position: TOP * 2.4375 ANCHOR TO Page T... Edge LEFT * .5 ANCHOR TO Page L... Edge

Columns

1 2 3 4 5 6 7 8

Company Logo Form Title
Page No.
Page M... Customer
Customer D...

Company Information

Ship To Element 18 {totalamount} Test {actionitem}
{custbody17}

Element Label Style

Font: Helvetica Size: 9 Bold Italic Underline Alignment: Left Center Right

USE LABEL COLORING

SPLIT HORIZONTALLY

Element Data Style

Font: Helvetica Size: 9 Bold Italic Underline Alignment: Left Center Right

Additional Options

SHOW BORDER SHOW ON ADDL. PAGES



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 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**.

To adjust any transaction form layouts that you want to mail in standard #10 window envelopes, repeat the preceding steps. 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.

Fraction	Decimal	Fraction	Decimal
1/2	0.5	1/16	0.0625
1/4	0.25	3/16	0.1875
1/3	0.33	5/16	0.3125
3/4	0.75	7/16	0.4375

Fraction	Decimal	Fraction	Decimal
2/3	0.67	9/16	0.5625
1/8	0.125	11/16	0.6875
3/8	0.375	13/16	0.8125
5/8	0.625	15/16	0.9375
7/8	0.875	—	—

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 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 only on the first page, 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 alternative 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

You can customize 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 needed.

Embedded in the body block are NetSuite tags that correspond to individual content elements. The content elements correspond 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 needed, 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 must begin with a `<tr>`.
- Elements always have data and can have a label.
- The label is represented by the `<nllabel>` tag that must be included in any customization of the label field.
- The data is represented by the `<nldata>` tag that must be included in any customization of the data field.
- There is also a `<nlattributes>` tag that 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 included only 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. Because 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 the help topic [Shortcuts Portlet](#).

For more information, see the following:

- [Creating Custom Record Types](#)
- [Online Custom Record Forms](#)
- [Parent-Child Record Relationships](#)
- [Sourcing with Custom Records](#)
- [Updating Custom Record Types](#)
- [Using Custom Record Entries](#)



Important: SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should access the NetSuite UI only by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic [SuiteScript 2.x Custom Pages](#).

Custom records are supported in SuiteCloud Development Framework (SDF). SDF is a development framework that you can use to create SDF SuiteApps, or to customize NetSuite accounts, using an integrated development environment (IDE) on your local computer. SuiteCloud projects are file-based

and use XML definitions of custom NetSuite objects. For more information, see the help topic [SDF Custom Object and File Development in SuiteCloud Projects](#).

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.
4. Configuring File and Child Record 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 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** – Select 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 select **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 select 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, which enables the employee to modify the custom record type. When defined as a manager, employees are automatically granted custom record view permission. The custom record view permission permits managers 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 about parent and child records, see [Parent-Child Record Relationships](#), [Creating a Parent-Child Relationship](#), 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

You can create new custom record types, as needed.

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 required 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 40.
2. In the **ID** field, enter a unique alphanumeric ID for the custom record. For information about best practices and naming conventions, see [Conventions for Naming Custom Objects](#). For information about changing an existing ID, see [Changing the ID of a Custom Object](#).
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**.



Note: The assignment of internal IDs to new records is sequential. However, the sequence of internal IDs does not indicate when or in which order records were added.

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 permit the record entry owner to be changed, check the **Allow Change** box. If you permit 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 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 Type page, select a permissions model 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 be accessed only 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.

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 Permission 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 Permission 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 any of the following:

- Custom record entries permissions defined on role records
- Permissions defined on the **Permissions** subtab of a custom record type
- No permissions (meaning access to the custom record type is public)

Note: As of 2012.1, the Access Type list replaces the Use Permissions 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 permission to access the custom record type 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 permission to access the custom record type 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:

- You cannot access the custom record type from the NetSuite user interface.
- If you attempt to list, search, view, edit, or create a record of this type in the user interface, the following error message appears: Access to that record type from the user interface is not allowed.
- The following custom record options are locked as disabled: Allow Mobile Access, 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. Later, the child records display 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. 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 permitted.
6. To permit 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 permit 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 available only 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. 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 permit 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 permit 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 available only 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**. 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. Users must drill down from the list to each record entry to edit it.
8. To permit the translated display name of custom record instances to be included in saved searches, check the **Enable Name Translation** box.
9. To permit 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** – Select 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 select **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 select 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, which enables the employee to modify the custom record type. When defined as a manager, employees are automatically granted custom record view permission. The custom record view permission permits managers 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 about parent and child records, see [Parent-Child Record Relationships](#), [Creating a Parent-Child Relationship](#), and [Using Child Records](#).

Defining Hierarchies among Custom Record Values

A custom record type can be set up to permit users to define hierarchies among values for that custom record type. Note that this hierarchy is not for the relationships between various 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.

Custom Record Type

Custom Employee

Actions ▾

NAME *	Custom Employee	SHOW OWNER <input type="checkbox"/> ON RECORD <input type="checkbox"/> ON LIST <input checked="" type="checkbox"/> ALLOW CHANGE	<input type="checkbox"/> ALLOW QUICK SEARCH <input checked="" type="checkbox"/> ALLOW QUICK ADD <input checked="" type="checkbox"/> ENABLE SYSTEM NOTES <input checked="" type="checkbox"/> INCLUDE IN GLOBAL SEARCH <input checked="" type="checkbox"/> INCLUDE IN SEARCH MENU <input checked="" type="checkbox"/> ENABLE OPTIMISTIC LOCKING <input checked="" type="checkbox"/> ENABLE INLINE EDITING <input type="checkbox"/> ENABLE NAME TRANSLATION <input checked="" type="checkbox"/> HIERARCHY <input type="checkbox"/> INACTIVE
PACKAGE			
ID	customrecord3	ACCESS TYPE Require Custom Rec...Entries Permission	
OWNER	Cindy Sloan	<input checked="" type="checkbox"/> ALLOW UI ACCESS <input type="checkbox"/> ALLOW MOBILE ACCESS <input checked="" type="checkbox"/> ALLOW ATTACHMENTS <input checked="" type="checkbox"/> SHOW NOTES <input type="checkbox"/> ENABLE MAIL MERGE <input type="checkbox"/> RECORDS ARE ORDERED <input checked="" type="checkbox"/> SHOW REMOVE LINK <input type="checkbox"/> ALLOW CHILD RECORD EDITING <input type="checkbox"/> ALLOW DELETE <input type="checkbox"/> AVAILABLE OFFLINE	
DESCRIPTION			
<input checked="" type="checkbox"/> INCLUDE NAME FIELD <input type="checkbox"/> SHOW ID SHOW CREATION DATE <input type="checkbox"/> ON RECORD <input type="checkbox"/> ON LIST SHOW LAST MODIFIED <input type="checkbox"/> ON RECORD <input type="checkbox"/> ON LIST			
Fields Subtabs Sublists Icon • Numbering • Forms • Online Forms Permissions Links Managers Translation • Child Records			

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.

Custom Employee

Celia Johnson

Actions ▾

NAME *	Celia Johnson	PARENT
<input type="checkbox"/> INACTIVE		
Child Records Notes Files		
VIEW Default View		
<input style="border: 2px solid red; padding: 2px; margin-right: 10px;" type="button" value="New Custom Employee"/> <input type="button" value="Customize View"/>		
EDIT	NAME ▲	
Edit	Celia Johnson : David Jackson	
Edit	Celia Johnson : Jason Bender	
Edit	Celia Johnson : Sandy Sinclair	

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 screenshot shows the 'Custom Employee' record creation page. At the top, there are buttons for 'Save' (highlighted in blue), 'Cancel', and 'Reset'. Below this, the 'NAME *' field contains 'Sandy Sinclair' and has an 'INACTIVE' checkbox. The 'Notes' tab is selected, showing a list of notes. One note is selected, displaying its details: 'Celia Johnson : David Jackson'. A dropdown menu labeled 'PARENT' is open, showing a hierarchy structure:

- New -
- Celia Johnson
- Celia Johnson : David Jackson** (selected)
- Celia Johnson : Jason Bender
- Jane Doe
- Jane Doe : Bob Jones

Below the notes list, there are fields for 'TITLE' and 'MEMO *'. At the bottom of the notes section, there are buttons for 'Add' (blue), 'Cancel', 'Insert', and 'Remove'.

The Custom Record List page shows the hierarchy for the records in the format <parent record>:<child record>.

The screenshot shows the 'Custom Employee List' page. At the top, there are buttons for 'VIEW Default' (highlighted in blue), 'Customize View', and 'New Custom Employee'. Below this is a toolbar with icons for file operations and a 'FILTERS' button. The main area is a table with columns for 'EDIT | VIEW' and 'NAME'. The data is as follows:

EDIT VIEW	NAME
Edit View	Celia Johnson
Edit View	Celia Johnson : David Jackson
Edit View	Celia Johnson : Jason Bender
Edit View	Celia Johnson : Sandy Sinclair
Edit View	Jane Doe
Edit View	Jane Doe : Bob Jones
Edit View	Jane Doe : John Smith

A yellow box at the bottom left contains an exclamation mark icon and the text: '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.

Custom Record Type

Warranty

NAME *	<input type="text" value="Warranty"/>	ACCESS TYPE	<input checked="" type="checkbox"/> Require Custom Re...tries Permission
ID	customrecord29	SHOW OWNER	<input type="checkbox"/> ON RECORD <input type="checkbox"/> ON LIST <input type="checkbox"/> ALLOW CHANGE
INTERNAL ID	29	ALLOW UI ACCESS	<input checked="" type="checkbox"/>
OWNER	J Wolfe	ALLOW MOBILE ACCESS	<input type="checkbox"/>
DESCRIPTION	<input type="checkbox"/> SHOW NOTES <input type="checkbox"/> ENABLE MAIL MERGE <input type="checkbox"/> RECORDS ARE ORDERED <input type="checkbox"/> ALLOW CHILD RECORD EDITING <input type="checkbox"/> ALLOW DELETE <input type="checkbox"/> AVAILABLE OFFLINE		
<input checked="" type="checkbox"/> INCLUDE NAME FIELD <input type="checkbox"/> SHOW ID		INCLUDE IN SEARCH MENU	<input checked="" type="checkbox"/> ENABLE QUICK SEARCH <input type="checkbox"/> ALLOW QUICK ADD <input checked="" type="checkbox"/> ENABLE SYSTEM NOTES <input checked="" type="checkbox"/> INCLUDE IN GLOBAL SEARCH <input checked="" type="checkbox"/> INCLUDE IN SEARCH MENU <input checked="" type="checkbox"/> ENABLE OPTIMISTIC LOCKING <input checked="" type="checkbox"/> ENABLE INLINE EDITING <input type="checkbox"/> ENABLE NAME TRANSLATION <input type="checkbox"/> INACTIVE

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 backward 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 subtab is not available until the record type has been saved.

To show inactive fields in the list, check Show Inactives.

To add fields to your custom record:

1. To create a new field for this record type, click **New Field** on the **Fields** subtab.
For step-by-step instructions for creating a new field, refer to [Creating a Custom Field](#).
2. Rearrange the fields as needed.
Select a line and drag it to the required 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. Make changes as needed, and 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 can:

- restrict search access to specific custom record type fields on a field-by-field basis
- restrict the audience for custom record type saved searches on a search-by search-basis

Limiting Search and Reporting Access to a Custom Field

You can limit search and reporting access to a custom field.

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. To prevent any searches or reports from returning data for this field, on the **Access** subtab for the field, set the **Default Level for Search/Reporting** option to **None**.

You also can prevent searches or reports run by users with specific roles from returning data for this field. To do so, set 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](#).

Limiting Access for a Custom Record Saved Search

You can limit access for a custom record saved search.

To limit access for a custom record saved search:

1. Go to Lists > Search > Saved Searches.
2. Click **Edit** beside the saved search that you want to restrict.
3. On the search page, click the **Audience** subtab and make changes, as needed, to the users who have access.

For example, you can clear the **Public** box or the **Select All** box for roles. Instead, select only the specific roles that will 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 see only records where they or their subordinates are the sales rep.

For details about setting these restrictions, see the following topics.

- [Setting Employee Restrictions](#)
- [Setting Department, Class, and Location Restrictions](#)
- [Restricting 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. Those users see only custom records where they or their subordinates are the value for the custom field.

When a custom record has a field that permits restrictions, and the Apply Role Restrictions box is checked, empty fields are not included when the restrictions are applied. When role restrictions are enabled for a custom record, there is no way to view or edit a record where the role-related field is empty. For example, if your user role is restricted to Subsidiary A, and a custom record is created that does not specify a subsidiary, your view is restricted to Subsidiary A. You will see only records where Subsidiary A is selected, and you will not see any records with empty subsidiary fields.

Complete one of the following, as needed:

- [Applying Role-Based Access Restrictions to Custom Records if the Class, Department, Location, or Subsidiary Field Does Not Yet Exist.](#)
- [Applying Role-Based Access Restrictions to a Custom Record if the Field Already Exists.](#)

Applying Role-Based Access Restrictions to Custom Records if the Class, Department, Location, or Subsidiary Field Does Not Yet Exist

The following procedure describes how to apply role-based access restrictions to a custom record if the class, department, location, or subsidiary field does not yet exist.

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. In the **Label** field, 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 needed, and click **Save**. For more information, see [Adding Fields to Custom Record Types](#).

Applying Role-Based Access Restrictions to a Custom Record if the Field Already Exists

The following procedure describes how to apply role-based access restrictions to a custom record if the field already exists.

Note: Applying center-based restrictions to custom records is similar to applying role-based restrictions to custom records. For more information about how you can prevent sensitive data from being displayed to unauthorized contacts, see [Limiting Search Access to Custom Records](#).

To apply role-based access restrictions to a custom record 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 required 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.

The screenshot shows the SuiteBuilder interface for defining a custom record type named 'Equipment'. The 'Subtabs' subtab is active. A new subtab titled 'Purchase Details' has been created and is currently selected. The main record details (NAME: Equipment, ID: customrecord26, INTERNAL ID: 26, OWNER: J Wolfe) are shown on the left. On the right, there are several permission checkboxes, some of which are checked (e.g., ALLOW UI ACCESS, SHOW NOTES). The bottom navigation bar includes tabs for Fields, Subtabs, Sublists, Icon, Numbering, Forms, Online Forms, Permissions, Links, Managers, Translation, and Child.

2. If needed, 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: Because the parent field is populated with subtabs created for the current custom record type, you can define the subtab as a child only 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: For the subtabs to be available to place fields on, you must first create the subtabs and save the record. You can place fields on the new subtabs by editing the field record from the **Fields** subtab.

6. Rearrange the fields as needed. If you have not yet created fields for your custom record, you must first define the fields. You can assign a subtab on the Display subtab when you create custom fields. For more information, see [Adding Fields to Custom Record Types](#).
Select a line and drag it to the required position, or use the **Move** buttons.
7. To edit the name of a subtab, click the line for that subtab and change the name in the **Title** column.
8. Click **Save**.

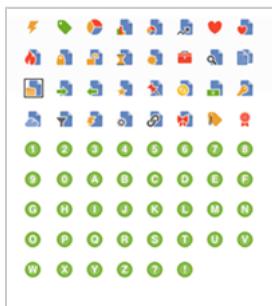
Choosing an Icon for a Custom Record

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:

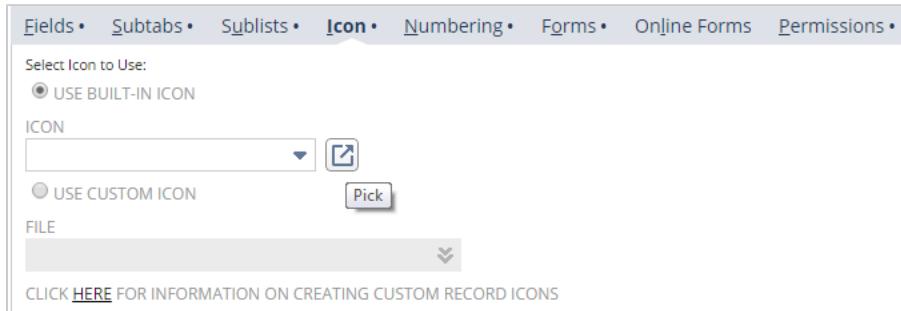


To associate an icon with a custom record:

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.

Alternatively, you can click the picker to view available icons and select the one you want to use.



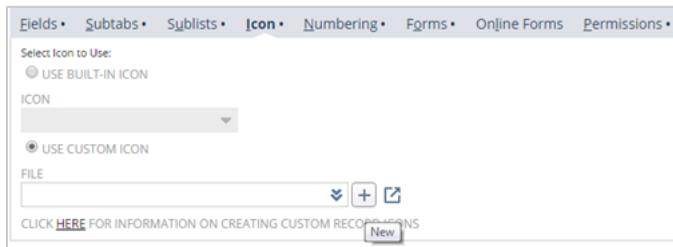
3. If you choose **Use Custom Icon**, select its file. For requirements for custom icons and instructions for creating them, see [Creating Icons for Custom Records](#).
4. Click **Save**.

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 **Icon** subtab, select the **Use Custom Icon** option.
4. Click the **New** button next to the **File** field.



5. In the File popup window, select an icon image.
6. Click **Save**. Then save the custom record.



Important: Custom icons must meet the specific requirements detailed in the following sections. 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 varied 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 backgrounds.



Image 1: Grayscale icon for dark backgrounds. This icon is an outline version of the full-color icon, used on dark color themes. Because the default color theme is dark-colored, this version is used most often.

Image 2: Grayscale icon for light backgrounds. This is another outline version of the icon, used on light color themes. This version is not used as often because few color themes have a light background color.

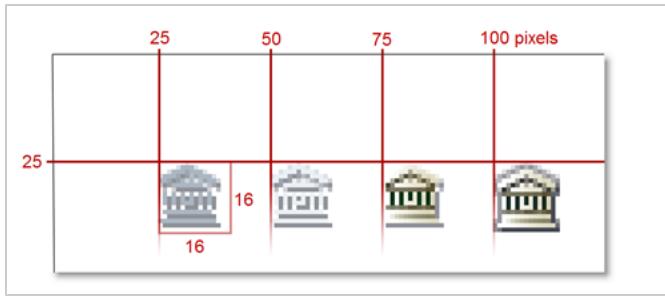
Image 3: Color icon for dark backgrounds. This is a full-color version of the icon, optimized for use on dark backgrounds. This version is used as the color icon that replaces the grayscale version when a cursor is nearby.

Image 4: Color icon for light backgrounds. This is another full-color version of the icon, optimized for use on light backgrounds. This version is used in menus, which always have a light background.

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, 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.



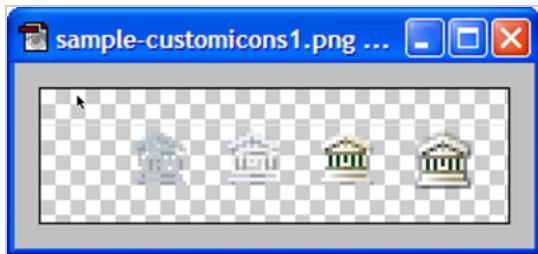
Each icon version must be located at the following locations within the image file, as measured from the upper-left corner of the image:

Icon Version	x Coordinate	y Coordinate
Grayscale icon for dark backgrounds	25	25
Grayscale icon for light backgrounds	50	25
Color icon for dark backgrounds	75	25
Color icon for light backgrounds	100	25

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 various 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**. You should 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.



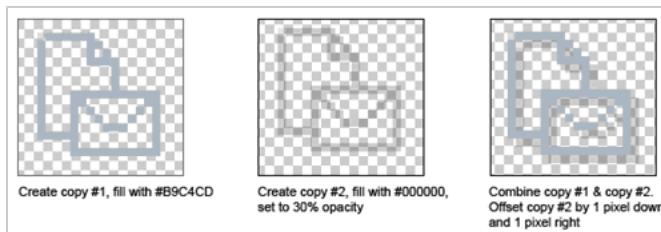
Constructing Your Icon File, Step-by-Step

1. You can create the icon yourself or obtain one from a royalty-free source. When you have the icon, the first step in creating the final icon image file is to create an outline of it. This outline drawing is used to construct the two grayscale icon versions.

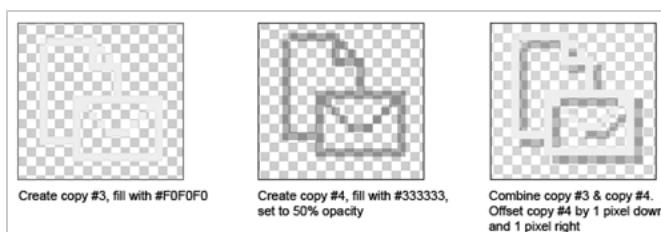


The outline version of your icon should be plain, but it should also have sufficient detail so that it can be recognized when it appears on your dashboard's New Bar. Start by using a 1-pixel pencil tool in your image editing application, and trace an outline of your icon. In addition to the outline, 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 various background colors, you can create the last version of your icon.

If you believe 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 previously described. 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.

 **Note:** Prefixes and suffixes are applied to numbering as configured on the custom record. However, the internal ID assigned to a custom record instance can differ from the autogenerated number or manually-specified ID number.

 **Warning:** After you enable autonumbering, numbers cannot be removed from records. Disabling autonumbering in the future prevents only future records from being numbered.

To assign automatic numbers to a custom record type:

1. Go to Customization > Lists, Records, & Fields > Record Types.
2. In the **Edit** column, click the name of the record type you want to have autonumbered.
3. Click the **Numbering** subtab.
4. To turn on automatic numbering, check the **Enable** box.
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 autogenerated 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. If you want to be able to enter a custom number when you edit the record instance, check the **Allow Override** box.
10. To number record instances that have already been created, check the **Update** box.
If you have already enabled automatic numbering, checking this box does not change any existing numbers, because 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 autonumbered 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 on the Custom Record Type page 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 subtab 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 needed.

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 create a new custom form based on a standard form. Selecting **Edit** enables you customize an existing custom form.
2. Customize the form as needed 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 to 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 may 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, and so on.
- **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

You can create an online custom record form to receive information from customers on your website.

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 are **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 to display 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 **Source**.

 - In the **Detail Message** field on the **Detail Message** subtab, enter a message 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. If you want to link to this form from a website, check the **Enable Online** box.
8. If required, check the **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 needed.
10. Rearrange the order of fields as needed. Click a line and drag it to the required position or click **Move Up**, **Move Down**, **Move to Top** or **Move to Bottom**.
11. In the **Set Up Workflow** subtab:
 - To receive notification when this form is submitted, enter the addresses you want email messages 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.
12. In the **Set Up Appearance** subtab:
With the exception of the Font, the fields described in the following list apply only to default templates.
 - 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.
 - To expand any subtabs on the form, check the **Unlayered Sections** box.
 - 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 Setup > Intranet > Images or Commerce > Site Builder > Content > 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. However, for smaller low-resolution monitors it can be better to construct 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.
13. In the **Custom Code** subtab:
The **Custom Code** subtab is available only if you have a SuiteScript feature enabled. For detailed information about using custom JavaScript files to perform functions, refer to [SuiteScript](#).
 - In the **Script File** field, select the JavaScript file that contains the required 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.
14. Click **Save**.
 15. If needed, preview your new form.
Select the form from the Online Form subtab and then click **Preview**.
 16. 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 about 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. Go to Commerce > Site Builder > Content > Information Items.
9. Click **Edit** next to the information item you want to link to your online custom record form.
10. Select the **Basic** subtab.
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:

```
<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, see the help topic [Using HTML in Description Fields](#).

For more information about online custom record forms, see [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 the following topics:

- [Creating an HTML Template Locally](#)
- [Using NetSuite Tags](#)
- [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 Tags](#).
- An input tag that defines the button your customers use to submit the form.

```
1 | <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:

```
1 | <input type="submit" value="Submit Form">
```

For example, the following HTML code is a representation of an acceptable form template:

```
1 | <html>
2 | <head>
3 | </head>
4 | <body>
5 | <NLFORM>
6 | <p>Enter the name of your company: <NLNAME></p>
7 | <p>Enter an email address we can use to contact you: <NLCUSTRECORD1></p>
8 | <p><input type="submit" value="Submit Form"></p>
9 | </form>
10| </body>
11| </html>
```



Important: The resulting file must include all tags, including <html>, <head> and <body> tags, to ensure that it is a valid HTML file recognizable by NetSuite.

Using NetSuite 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: Field IDs are incorporated into tag definitions for fields. When creating custom fields to use in HTML templates, you should 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. The result can make it more difficult to know exactly what the field references.

If you have chosen to include a required 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. The URL for this page is displayed in the Address bar of your browser. The ID for the selected field is at the end of the URL.

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. Click the **Add File** button.
4. Browse your system files to locate and select the HTML template you want to add.
5. Click **Open**. The template is added to the folder.

Creating an HTML Form Template

HTML form templates can be created 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:

1. Go to Lists > Marketing > Marketing Templates > New.
2. On the Select Type page, click **Online Form**.
3. In the **Name** field, enter a name for the template.

4. In the **Description** field, enter any additional information for this template. The information should describe the template — it is not a component of the displayed form. Useful information includes details about where and how this template should be used or descriptions of any unusual fields or field relationships.
 5. Click the **Template** subtab.
 6. If needed, enter a title for the template in the **Title** field. If you do not provide a title, the title is sourced from the **Name** field.
 7. Choose a **Create Template From** option:
 - To base the current template record on an existing record, select **File**. This option enables you to choose a file from the file cabinet or upload a new template file.
 - To create a new template using your own layout, select **Text Editor**. This option enables you to enter text and fields for the new template in the text editor.
 8. If you chose the **File** option, select an existing template to use as the base for the current template. Otherwise, proceed to the next step.
- You can select a template in the following ways:
- Enter the first few letters of the template name in the file selection field and then press the **Tab** key to automatically select a matching template. If multiple templates match the text, a list displays, enabling you to select the required template.
 - To select from a list of templates, click the down arrow in the file selection field to display a list from which you can select the required template.
 - To upload an HTML template from your system, click the **New** icon, and upload the appropriate file. For information about creating your own HTML templates, see [Creating an HTML Template Locally](#).
- After a template is selected, you can view detailed information about the template by clicking the **Open** icon.
9. If you chose the **Text Editor** option, enter text, as needed, in the text editor. To include fields in the template, do the following:
 1. To select the type of record the field appears on, select a field type from the **Field Type** list.
 2. To add a NetSuite field into the HTML template, select a field from the Insert **Field** list.sourcing
- i Note:** If you previously selected the File option instead of the Text Editor option, you do not need to enter template tags because the file selected in the create template from file steps contains the tag information.
10. If required, you can limit access to this template to yourself or to a specific group. To do so.
 1. Click the **Restrict Access** subtab.
 2. To make yourself the only person in your company who can use this template, check the **Private** box
 3. To restrict the template to members of a specific group, in the **Restrict to a Group** field, enter the name of the group.
 11. 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 Permissions subtab lets you restrict access by role to your custom records and the forms used to enter the records. By restricting access to custom records, you can have greater control over who sees the information specific to your business. By restricting access to the entry form, you can control the entry form used by your employees to enter custom records.

When you set permissions for your custom records, you restrict access to the record entries, **not** the record type. To edit, view, or enter a custom record type, you must have a role with permission to access to custom record types.

If you do **not** set permissions here, anyone with access to custom record entries can view, edit, and enter custom records for this type.

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 Permission List 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.
 - **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:
 - To restrict users with this role to viewing or editing the records of this type that only they or their subordinates created, select **Viewing and Editing**.
 - To permit users with this role to view all records of this type but restrict them to editing the records that only they or their subordinates created, select **Editing Only**.
 - To permit users with this role to view and edit all records of this type, leave this column blank.
 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. To make the default form the only entry form available to this role, check the box in the **Restrict Form** column.
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. To make the selected view the only one available to this role, check a box in one of the **Restricted** columns.
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 needed, 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. For more information about creating custom categories, see [Creating Center Categories](#).
4. In the **Links** column, select the type of link you want to add to the menu:
 - To have the custom record list option display in the menu and open to the Custom Record List page, select **Custom Record**, where Custom Record is the name of the custom record. With this option, New and Search are also available on the menu.
 - To have the new custom record option display in the menu and open to the New Custom Record page, select **New Custom Record**, where Custom Record is the name of the custom record.
 - To have the search custom record option display on the menu and open to the Custom Record Search page, select **Search Custom Record**, where Custom Record is the name of the custom record.
5. 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.
6. (Optional) In the **Translation** column, enter any translated names required for the link. Then click **Done**.

7. (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.
8. Click **Add**.
9. Repeat these steps for each link you want to add.
If you want to edit a link, click the line for that link and make changes as needed.
10. 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](#)

i Note: If you use the multi-language feature, you should manage your translations using translation collections maintained in Manage Translations. It is more efficient to use a translation collection for managing translations for multiple languages than it is to define translations on the Translation subtab. For information, see [Multiple Languages Translation Management](#).

Before you can add translations, you must select translation languages in general company preferences on the Languages subtab. For more information, see the help topic [Setting General Account Preferences](#). The language subtab lists:

- System-supported languages that can be used for the NetSuite user interface (available at Home > Set Preferences)
- Additional languages that can be used for website translations only (not available at Home > Set Preferences)

You should enter translations for system-supported languages only, because these are the only languages that can be displayed in the user interface. For custom records, NetSuite uses the user's language preference for translations. For standard records, translations are returned and updated in the company's primary language. For details, see the help topic [Configuring Multiple Languages](#).

Multiple Languages Translation Management

The Manage Translations feature enables you to manage your language translations using translation collections. If you use the multi-language feature, you should edit translations for custom records using a translation collection because it is more efficient than making the edits on the Translation subtab of custom records.

Language customizations made on the Translation subtab of custom records override the language set up in Home > Set Preferences. If you choose to edit translations using the Translation subtab, language changes will be visible only to users with the same language preference.

If you use SDF, you can enter translations in forms in the SDF XML. You should define translations using a translation collection. If you modify a form and enter a translation string directly in the form, the

translations are not connected to terms or translations collections and are visible only to users with the same language preference. If you change your language preference, then you must also define the translations for that language.

For more information, refer to these topics:

- [Configuring Multiple Languages](#)
- [Translation Collections Overview](#)

Translating a Custom Record Name

You can define translations for a custom record type name on the Translation subtab of the custom record page:

The screenshot shows the 'Custom Record Type' page with the 'Translation' subtab selected. The main area contains fields for NAME, ID, OWNER, DESCRIPTION, and various checkboxes for access and search settings. Below this is a table for translating the record name into different languages. The 'NAME' column lists 'Czech', 'German', 'Hebrew', and 'Spanish'. The 'TRANSLATION' column shows the translated names: 'Czech' is empty, 'German' is empty, 'Hebrew' is empty, and 'Spanish' is 'Libro'. A red box highlights the 'Translation' subtab in the navigation bar at the bottom of the table.

LANGUAGE	NAME
Czech	
German	
Hebrew	
Spanish	Libro

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:

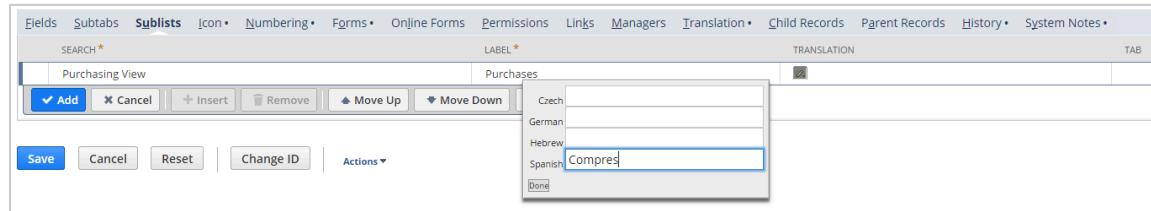
The screenshot shows the 'Subtabs' subtab of the custom record page. It displays a table with a single row for the 'TITLE' 'Chapter'. The 'TRANSLATION' column contains a small icon with a red box around it, indicating a placeholder or a link for translation. The 'PARENT' column is empty.

TITLE*	TRANSLATION	PARENT
Chapter		

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 can provide translations for the individual instances of that record type, such as chair, table, bed.

To permit translation of custom record instance names, check the Enable Name Translation box for the custom record type. This option is disabled by default.

NAME *	<input type="text" value="Furniture"/>	SHOW OWNER <input type="checkbox"/> ON RECORD <input type="checkbox"/> ON LIST <input type="checkbox"/> ALLOW CHANGE	ALLOW QUICK SEARCH <input type="checkbox"/>
ID	customrecord810	ACCESS TYPE Require Custom Re...ries Permission	ALLOW QUICK ADD <input checked="" type="checkbox"/>
INTERNAL ID	810	<input checked="" type="checkbox"/> ALLOW UI ACCESS	ENABLE SYSTEM NOTES <input checked="" type="checkbox"/>
OWNER	Sloan, Cindy	<input checked="" type="checkbox"/> ALLOW MOBILE ACCESS	INCLUDE IN GLOBAL SEARCH <input checked="" type="checkbox"/>
DESCRIPTION		<input checked="" type="checkbox"/> ALLOW ATTACHMENTS	INCLUDE IN SEARCH MENU <input checked="" type="checkbox"/>
		<input checked="" type="checkbox"/> SHOW NOTES	ENABLE OPTIMISTIC LOCKING <input type="checkbox"/>
		<input type="checkbox"/> ENABLE MAIL MERGE	ENABLE INLINE EDITING <input checked="" type="checkbox"/>
		<input type="checkbox"/> RECORDS ARE ORDERED	ENABLE NAME TRANSLATION <input checked="" type="checkbox"/>
		<input type="checkbox"/> ALLOW CHILD RECORD EDITING	INACTIVE <input type="checkbox"/>
		<input type="checkbox"/> ALLOW DELETE	
		<input type="checkbox"/> AVAILABLE OFFLINE	
<input checked="" type="checkbox"/> INCLUDE NAME FIELD <input type="checkbox"/> SHOW ID SHOW CREATION DATE <input type="checkbox"/> ON RECORD <input type="checkbox"/> ON LIST SHOW LAST MODIFIED <input type="checkbox"/> ON RECORD <input type="checkbox"/> ON LIST			

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

LANGUAGE	NAME
Czech	Silla
German	Silla
Hebrew	Silla
Spanish	Silla

Parent-Child Record Relationships

A child record type is a record that is referenced from 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 tables in 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 plural of the name of the parent record type.

Parent Record: Customer

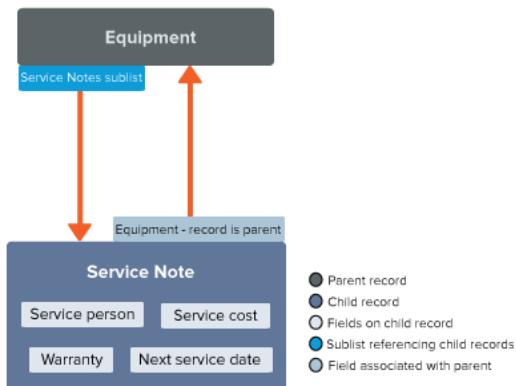
Child Records: Notes, Messages (sublists)

You can also create your own parent-child relationships. For example, you can create a custom record type called Service Note and associate it with an Equipment custom record type. In this example, the Service Note record is associated as the child of the parent Equipment record type. On the Service Note 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, defined as a custom field, 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.



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 reopen the parent record in edit mode, you can then add existing child records or create new ones. 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 subtab, 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 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 the List/Record field, select a record type. Then check the **Record is Parent** box. The record entered in the List/Record field becomes the parent record of the child record. If applicable, you can define multiple custom fields for multiple child records related to the parent record.

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 of the subtab where the child record appears.

The screenshot shows the 'Display' subtab configuration for a child record. The 'PARENT SUBTAB' dropdown is set to 'Normal'. Other visible settings include 'DISPLAY TYPE' (Normal), 'SUBTAB' (dropdown), and 'ALLOW QUICK ADD' (checkbox). The 'Display' tab is currently selected.

Creating a Parent-Child Relationship

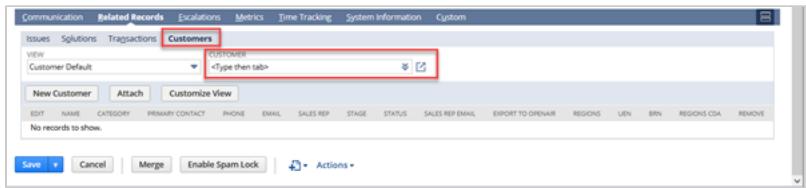
The following process describes an example of creating a parent-child relationship using a custom field. For this example, we are going to create a custom field and assign the parent record and child record. The parent record pulls in the child record information through the custom field. We want to 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 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 required 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, select **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 available only when **Record is Parent** is checked.
9. Click **Save**.

You will now have a parent record of Case with a subtab called Customers that lists all customers affected by this case.



Note: When associating a list with a transaction, you can select only transactions with forms that can be customized. 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.x 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 various 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

The topics in this section refer 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 by workflows or custom scripts only. 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 on the parent record, 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.

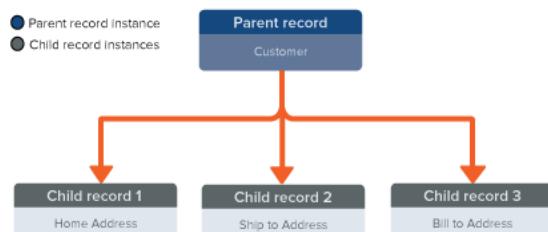
Hierarchy – When checked, users can create parent-child relationships between records of the same custom record type. To do so, create a custom field which will be used to establish the relationship between the custom records, as described in [Hierarchy of Parent-Child Instances of Same Record Type](#).

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 has only one instance. The following example shows a customer record instance with three unique addresses. The three 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 multiple 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

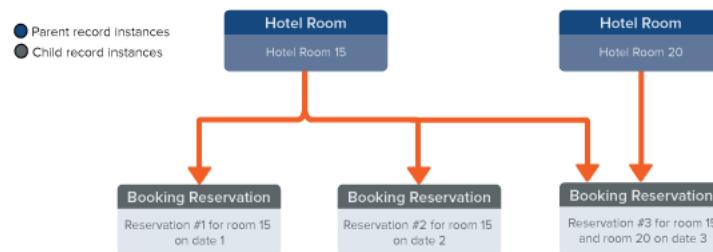
associated child record instance. On the child record instance there is a multi-select field in which many parent records are listed.

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 multiple 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 various child records of various 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 Rate

Hotel rooms have:

- Booking reservations
- Information about seasonal pricing policy

Parking spaces have:

- Booking reservations

Example setup:

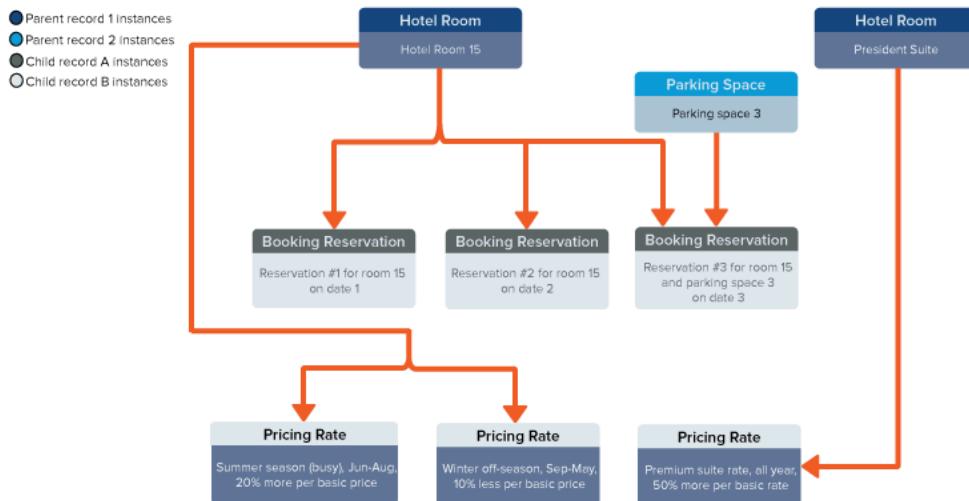
- Two parent record types: 1 - Hotel Room, and 2 - Parking Space
- Two child record types: A - Booking Reservation, and B - Pricing Rate
- Two list/record type custom fields applied to child record type A, where:
 - Record is Parent box is checked for both fields
 - The first field is a list/record type 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 applied 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 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

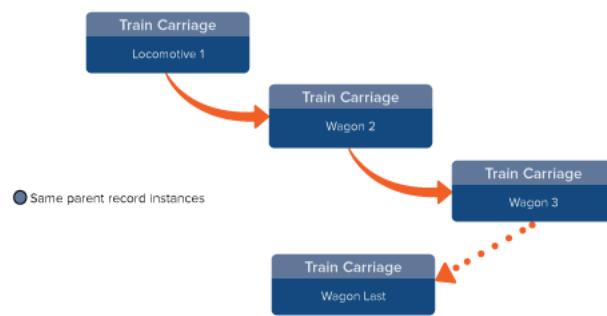
Note: Administrators can alternately model this behavior more easily on custom record types by checking the **Hierarchy** box on the custom record definition. See [Custom Record Type Fields that Affect Parent-Child Relationship on Custom Record](#).

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 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

i 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 exist only 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.x 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.

See the following:

- [Entering Data in Child Records from the Child Record List](#)
- [Entering Data in Child Records from the Parent Record](#)

Entering Data in Child Records from the Child Record List

You can enter data in child records from the child record list.

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, select the record you want this child record to link to.
For example, in the **Equipment** field, you would select 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 **Save** to submit the information and return to the child record list.

- Click **Save & New** to submit the information and add another record.

You can now view this child record from its parent record.

Entering Data in Child Records from the Parent Record

You can enter data in child records from the parent record. To do so, open the list of records where your parent record appears. Choose one of the following options:

- 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 performing the preceding 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. However, for records that are transformed from other records, child record association cannot occur when the records are first created. The association occurs only after the records 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:
 - To submit the information and return to the parent record, click **Save**.
 - To submit the information and add another child record, click **Save & New**.

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 select it in the Source List field.

For example, you have created an Intern custom record type. You want to include information about 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 about sourcing and custom fields, see [Setting Sourcing Criteria](#).

To use sourcing with custom record types:

1. Go to Customization > Lists, Records & Fields > Record Types.
2. In the **Edit** column, click the name of the record type you want to create a sourced field for.
3. Click the **Fields** subtab.
4. Click **New Field**.
5. In the **Label** 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 **Label** 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 select **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, select **Employee** in the **List/Record** field because your sales reps are employees.
14. Click the **Sourcing & Filtering** subtab.
15. 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. To do so, repeat Steps 1-9 for each kind of record you want to be able to source information from.
16. In the **Source From** field, select the field you want to include information from. If you click to view the list, sourcing and filtering options are available.
You must select a **Source List** before you can select a field.
The selected field's type must match the type selected in the **Type** field.
17. When you have finished, click **Save**.
18. 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. For more information about available standard fields and field types, see [Available Standard Fields and Field Types for Custom Record Types and Source Lists](#)

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 in the following table. In addition to standard records, you can use custom fields and record types for sourcing. When using custom fields and record types for sourcing, the field type selected 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 select Check Box in the Type field when creating your CRM field.

Customer, Employee, Partner, and Vendor Fields

Field	Type	Customer	Employee	Partner	Vendor
Name/ID	Free-Form Text	×	×	×	×
Bill To	Text Area	×	×	—	×
Ship To	Text Area	×	—	—	×
Phone	Phone Number	×	×	×	×
Fax	Phone Number	×	—	×	×
E-mail	E-mail Address	×	×	×	×
City	Free-Form Text	×	×	—	×
State	Free-Form Text	×	×	—	×
Zip	Free-Form Text	×	×	—	×
Country	Free-Form Text	×	×	—	×
Sales Rep	List/Record*	×	—	—	—
Expected Close Date	Date	×	—	—	—
Renewal Date	Date	×	—	—	—
Contact	Free-Form Text	×	—	—	×
Alt. Contact	Free-Form Text	×	—	—	×
Alt. Phone	Phone Number	×	—	—	×
Balance	Currency	×	—	—	×
Credit Limit	Currency	×	—	—	×
Account	Free-Form Text	×	—	—	×
1099 Eligible	Check Box	—	—	—	×
Tax ID	Free-Form Text	—	—	—	×
Company Name	Free-Form Text	—	—	×	×
Supervisor	List/Record*	—	×	—	—

Field	Type	Customer	Employee	Partner	Vendor
Soc. Sec. #	Free-Form Text	—	×	—	—

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

Item

Field	Type
Name	Free-Form Text
Display Name	Free-Form Text
Vendor Name	Free-Form Text
Online Name	Free-Form Text
Available Online	Check Box
Base Price	Currency
Cost	Currency
Preferred Vendor	List/Record*
On Hand	Decimal Number

* In the List/Record Field, you must select the Vendor list for sourcing to work properly.

Updating Custom Record Types

If you no longer need a custom record type, you can deactivate or delete the custom record type.

Inactivating a Custom Record Type

If a custom record type is not actively being used, you can deactivate it. The custom record type becomes inactive but is not deleted.

To deactivate 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 deactivate.
4. Click **Submit**.

Deleting a Custom Record Type

If a custom record type is not being used by any custom fields or workflows and is no longer needed, you can delete it.

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 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, custom workflow names, or both 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

The following procedure describes how to view or edit 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 records 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](#).
- **Online Forms** – Create online entry forms to capture information and create new records from outside your NetSuite account. You can customize the appearance of online forms as well as the information you require from anyone who submits these forms.



Note: This subtab is available only after the record type has been saved.

- **Permissions** – select 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 select **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, which enables the employee to modify the custom record type. When defined as a manager, the employee is automatically granted custom record view permission. The custom record view permission permits managers 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**.

For information about editing existing records, see the help topic [Viewing and Editing Records](#).

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 [Advanced Templates Customization 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.

Record Types							
New Type		FILTERS					
<input type="checkbox"/> SHOW INACTIVES		TOTAL: 2					
EDIT ▾	FROM BUNDLE	ID	OWNER	CUSTOM SEGMENT	LIST	NEW RECORD	SEARCH
Business Area		customrecord_cseg_business_area		Business Area	List	New Record	Search
Last Sales Activity Record	53195	customrecord_lsa	Automation User		List	New Record	Search

The custom record has the same name as the custom segment.

Custom Record Type

Business Area

[Save](#) [Cancel](#) [Reset](#) | [Actions ▾](#)

NAME *	<input type="text" value="Business Area"/>	SHOW LAST MODIFIED <input type="checkbox"/> ON RECORD <input type="checkbox"/> ON LIST	SHOW OWNER <input type="checkbox"/> ON RECORD <input type="checkbox"/> ON LIST <input type="checkbox"/> ALLOW CHANGE	ACCESS TYPE Use Permission List	<input checked="" type="checkbox"/> ALLOW UI ACCESS <input type="checkbox"/> ALLOW MOBILE ACCESS <input type="checkbox"/> ALLOW ATTACHMENTS <input type="checkbox"/> SHOW NOTES <input type="checkbox"/> ENABLE MAIL MERGE <input checked="" type="checkbox"/> RECORDS ARE ORDERED <input type="checkbox"/> SHOW REMOVE LINK <input type="checkbox"/> ALLOW CHILD RECORD EDITING <input type="checkbox"/> ALLOW DELETE	<input type="checkbox"/> ALLOW QUICK SEARCH <input type="checkbox"/> ALLOW QUICK ADD <input type="checkbox"/> ENABLE SYSTEM NOTES <input type="checkbox"/> INCLUDE IN GLOBAL SEARCH <input type="checkbox"/> INCLUDE IN SEARCH MENU <input type="checkbox"/> ENABLE OPTIMISTIC LOCKING <input checked="" type="checkbox"/> ENABLE INLINE EDITING <input checked="" type="checkbox"/> ENABLE NAME TRANSLATION <input checked="" type="checkbox"/> HIERARCHY <input type="checkbox"/> INACTIVE
ID	customrecord_cseg_business_area					
INTERNAL ID	329					
ORIGINATING CUSTOM SEGMENT	Business Area					
OWNER	[Owner]					
DESCRIPTION	Edit Description					
<input checked="" type="checkbox"/> INCLUDE NAME FIELD						
<input type="checkbox"/> SHOW ID						
SHOW CREATION DATE <input type="checkbox"/> ON RECORD <input type="checkbox"/> ON LIST						
Fields Subtabs Syblists Icon ▾ Forms ▾ Online Forms Links ▾ Child Records Parent Records History ▾ System Notes						
<input type="checkbox"/> SHOW INACTIVES						
New Field Move To Top Move To Bottom						
DESCRIPTION	ID	TYPE	LIST/RECORD	TAB	SHOW IN LIST	
No records to show.						

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.

Business Area Field
[List](#) [More](#)

[Save](#) ▾
[Cancel](#)
[Reset](#)
[Change ID](#)
[Apply to Forms](#)
[Actions ▾](#)

LABEL *

ID

OWNER

DESCRIPTION

TYPE

LIST/RECORD

STORE VALUE USE ENCRYPTED FORMAT

SHOW IN LIST

GLOBAL SEARCH

RECORD IS PARENT

INACTIVE

APPLY ROLE RESTRICTIONS

[Display](#)
[Validation & Defaulting](#)
[Sourcing & Filtering](#)
[Access](#)
[Translation](#)
[History](#)
[☰](#)

INSERT BEFORE

SUBTAB

DISPLAY TYPE

DISPLAY WIDTH

HELP

ALLOW QUICK ADD

[Custom Segment](#)
[List](#) [Search](#) [More](#)

[Save](#) ▾
[Cancel](#)
[Reset](#)
[Manage Values](#)

Primary Information

LABEL *

ID

RECORD ID

CUSTOM RECORD TYPE

TYPE

INACTIVE

FILTERED BY

Class

Department

Location

Subsidiary

[Accounting](#)

GL IMPACT

[Details](#)

HELP

DESCRIPTION

[Values](#)
[Application & Sourcing](#)
[Validation & Defaulting](#)
[Permissions](#)
[Dependent Segments](#)
[Display Order](#)
[Translation](#)
[☰](#)

DISPLAY ORDER SUBLIST ALPHABETICAL

VALUE *	TRANSLATION	PARENT	INACTIVE	FILTERING
<input type="text" value="Cafe"/>	<input style="outline: 2px solid red;" type="text" value="PASTRIES"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Grocery"/>	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="text" value="Frozen Food"/>	<input type="text"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

[Add](#)
[Cancel](#)
[Insert](#)
[Remove](#)
[Move Up](#)
[Move Down](#)
[Move To Top](#)
[Move To Bottom](#)
[Set Filters](#)

Using Custom Record Entries

Custom record instances or entries are the specific custom records you create. You can create, edit and search your custom record instances from the list of record types. You can also deactivate 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](#)
- [Making a Custom Record's Entries Inactive](#)
- [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

You can view a custom record entries list for a custom record type.

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

You can view or edit a custom record entry for a custom record type.

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 view.

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. For more information, see [Creating Links to Custom Records](#).

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

You can create a new custom record entry for a custom record type.

To create a custom record entry:

1. Go to Customization > Lists, Records & Fields > Record Types, and under the New Record column, click **New Record**.
2. Enter a name for the custom record entry. Then complete fields as needed and include any notes you want to add.
3. Click **Save**.

Copying a Custom Record Entry

You can copy a custom record entry for a record type.

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 view.
2. Click **View** next to the record you want to copy.
3. In the **Actions** menu, select **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 for the name.

Making a Custom Record's Entries Inactive

In some cases, you may need to hide one or more of a custom record's entries. To hide an entry, you make it inactive. Be aware that when you make an entry inactive, it does not appear as a choice in the record's list.

Inactivating a record's entries has the following results:

- The entry no longer appears as a choice in the records list on new forms or on existing forms where it wasn't already selected. However, if the entry is a parent to another entry, it still appears as a parent.
- If an inactive entry was previously selected on an existing form, the entry continues to appear on that form. A search on that entry will also turn up in search results.
- When filtering, if one record is dependent on another record, entries filtered by an inactive entry will not be available. This is true even if the inactive entry on which other entries depend is already selected.

To Make an Entry in a Record's List Inactive:

1. Go to Customization > Lists, Records & Fields > Record Types, and click **List** for the custom record entries list you want to view.
2. Click **List** for the record type
3. Click the **Inactive** box.
4. Click **Save**.

Searching Custom Record Entries

The following procedure describes how to search a list of custom record entries.

To search a list of custom record entries:

1. Go to Customization > Lists, Records & Fields > Record Types.
2. In the Search column, 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:

 - To search for all information of that kind, select **Any**.
 - To enter the information or part of the information you want to find, select **Is, Starts With**, or **Contains**.
 - To enter the information you want to exclude, select **Is Not, Does Not Start With**, or **Does Not Contain**.
 - Select a choice or choices from a list or a list of options.
 - To select from a popup list, click **List** next to an empty list field, or enter the first few letters and press the **Tab** key.
 - If you selected custom in the date field, enter or select dates in the **From** and **To** fields.
 - 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:
 - To run the search and open a NetSuite page with a list of results, click **Submit**.
 - To run the search and save results as a .csv file that you can save to disk or open on your desktop, click **Export**. For more information, see the help topic [Exporting Search Results](#).
 - 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, click **Personalize Search**. See the help topic [Defining a Saved Search as a Preferred Search Form](#).
 - To open a saved search page that includes the filters you defined, click **Create Saved Search**. 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 Criteria**.
 - 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](#).

Record Customization (Beta)



Warning: Record Customization (Beta) is a beta feature. The contents of this feature are preliminary and may be changed or discontinued without prior notice. Any changes may impact the feature's operation with the NetSuite application. NetSuite warranties and product service levels shall not apply to the feature or the impact of the feature on other portions of the NetSuite application. NetSuite may review and monitor the performance and use of this feature. The documentation for this feature is also considered a beta version and is subject to revision. **Please review Beta Software Legal Notices. ORACLE CONFIDENTIAL. For authorized use only. Do not distribute to third parties.** Pre-General Availability Draft: September 25, 2020.



Note: To access this beta feature, you must apply for participation in a beta testing activity by clicking on the associated recruitment form link. On the recruitment form, you will be required to identify an authorized signatory to accept the Oracle Cloud Services Beta Trial License Agreement. After the Agreement is fully executed by your authorized signatory, your company will be provided with the beta feature for testing.

Record Customization (Beta) provides a consolidated view of all business record types in NetSuite. You can view and maintain the record types in your account. You can also create new fields and sublists to include on a record.

A unified location to set up required fields makes it convenient to maintain required settings for multiple fields at one time. When you make a field required, the change is automatically saved to the server.

Record Customization (Beta) settings apply to all other forms. For example, if you make a field required in Record Customization (Beta), the field is marked as required on all forms that include the field.

The access control overview in Record Customization (Beta) provides you with a view of the roles and employees that have access to a specific record or field. You can view a list of all users with access to a record or field as well as the level of access with the applicable role restrictions the users have. From the access control overview, you can also maintain roles and user access.

For more information, see the following:

- [Navigating Record Customization \(Beta\)](#)
- [Record Customization \(Beta\) Page](#)
- [Record Customization Overview Page](#)
- [Record Customization Fields and Sublists Page](#)
- [Record Customization Access Page](#)

Navigating Record Customization (Beta)



Warning: Record Customization (Beta) is a beta feature. The contents of this feature are preliminary and may be changed or discontinued without prior notice. Any changes may impact the feature's operation with the NetSuite application. NetSuite warranties and product service levels shall not apply to the feature or the impact of the feature on other portions of the NetSuite application. NetSuite may review and monitor the performance and use of this feature. The documentation for this feature is also considered a beta version and is subject to revision. [Beta Software Legal Notices](#). **ORACLE CONFIDENTIAL. For authorized use only. Do not distribute to third parties.**



Note: To access this beta feature, you must apply for participation in a beta testing activity by clicking on the associated recruitment form link. On the recruitment form, you will be required to identify an authorized signatory to accept the Oracle Cloud Services Beta Trial License Agreement. After the Agreement is fully executed by your authorized signatory, your company will be provided with the beta feature for testing.

Before you can use Record Customization (Beta), you must enable the feature.

To enable this feature:

1. Go to Setup > Company > Enable Features
2. On the **SuiteCloud** subtab, check the **Record Customization (Beta)** box.

To start Record Customization (Beta), go to Customization > Lists, Records, & Fields > Record Customization (Beta).

The Record Customization (Beta) page groups all available records by record family on the left panel and lists all available records on the right panel. After you select a record, the left panel changes to a menu with tabs and the right panel displays record information. For more information about the menu options, see [Menu](#).

Record Customization (Beta) Information

The following sections describe the recurring elements that may be available on the various Record Customization (Beta) pages.

Record Family

Record family refers to the group to which the record belongs. Record family can be, for example, CRM, Entity, Item, Transaction, or Other.

Record Name

The name of the record whose overview information displays. After you select a record, the record name displays on the upper left of all Record Customization (Beta) pages.

Menu

The following screenshot shows the menu options available after you select a record:



Tab Name	Description
All Record Types	Go to the full list of record types.
Overview	View overview information for the record. When you click a record, the Overview tab is automatically selected.
Fields & Sublists	View fields and sublists information for the record.
Access	View access control information for the record.

Search and Filter Options

Record Customization (Beta) pages have search and filter options, which can include:

- Search boxes for record, field name and id, user, and role.
 - On the Record Customization (Beta) page, search for a record by name. Records listed in the left panel filter as you type.
 - On the Fields and Sublists pages, search for a field or sublist by name or id.
 - On the Record and Fields Access pages, search for a user by name. When searching for a user in Fields access, restrictions may apply by role. If there are one or more roles listed that have restrictions set up for the role, these restrictions can affect the access permitted for a record. When you select a user first, the Search Role filter includes only the roles available to the selected user Restrictions.
 - On the Fields Access page, search for a role by name. When a role is selected first, the Search User filter includes only users with the selected role assigned to them.
- An Origin filter, to filter records, fields, or sublists by origin. For more information, see [Origin](#).

Origin

Origin can appear as a field or a filter. Origin is the source of the record, and can be one of:

- System - records that are a standard part of NetSuite
- Customization - records created by your own in-account or sandbox customization
- SuiteApp - records deployed by a third-party SuiteApp

Or, for custom records that originate from a custom segment, Customization originates from [name of custom segment]. For custom records that originate from a custom segment, there is a link to the custom segment definition page.

Record Customization (Beta) Sections

Refer to the following topics for more information about Record Customization (Beta):

- [Record Customization \(Beta\) Page](#)
- [Record Customization Overview Page](#)
- [Record Customization Fields and Sublists Page](#)
- [Record Customization Access Page](#)

Record Customization (Beta) Page

✖ Warning: Record Customization (Beta) is a beta feature. The contents of this feature are preliminary and may be changed or discontinued without prior notice. Any changes may impact the feature's operation with the NetSuite application. NetSuite warranties and product service levels shall not apply to the feature or the impact of the feature on other portions of the NetSuite application. NetSuite may review and monitor the performance and use of this feature. The documentation for this feature is also considered a beta version and is subject to revision. **Please review Beta Software Legal Notices. ORACLE CONFIDENTIAL. For authorized use only. Do not distribute to third parties. Pre-General Availability Draft: September 25, 2020.**

i Note: To access this beta feature, you must apply for participation in a beta testing activity by clicking on the associated recruitment form link. On the recruitment form, you will be required to identify an authorized signatory to accept the Oracle Cloud Services Beta Trial License Agreement. After the Agreement is fully executed by your authorized signatory, your company will be provided with the beta feature for testing.

The Record Customization (Beta) page groups all available records by record family on the left and lists all available records on the right. From here you can:

- Expand and collapse the record family on the left to view and hide the list of available record types for the record family. Expanding the record families list does not affect the records displayed on the right.
- Click a family of records, for example Entity, on the left to display the list of record types belonging to the family on the right.
- Select a record from the record list on either the left panel or the right panel.

The unique elements of the Record Customization (Beta) page are identified in the following screenshot:

NAME	ID	RECORD FAMILY	ORIGIN
Accounting Book	accountingbook	Other	System
Advanced Intercompany Journal	advintercompanyjournalentry	Transaction	System
Bill	vendorbill	Transaction	System
Bill Credit	vendorcredit	Transaction	System
Campaign	searchcampaign	CRM	System
Case	supportcase	CRM	System
Cash Refund	cashrefund	Transaction	System
Cash Sale	cashsale	Transaction	System
Check	check	Transaction	System
Competitor	competitor	Other	System
Contact	contact	Entity	System
Credit Card Transaction	creditcardcharge	Transaction	System
Credit Memo	creditmemo	Transaction	System

- | | |
|---|--|
| 1 | All – List of all available business record types, displayed on the right. |
| 2 | Record Information – Information about the records, filtered by selected family group, available for the selected record type. Record information includes: <ul style="list-style-type: none"> ■ Name – Record type name, for example Sales Order. |

	<ul style="list-style-type: none"> ■ ID – Record ID. ■ Record Family – Group to which the record belongs. ■ Origin – Source of the record. <p>Sort record details in ascending or descending order by clicking any of the headings.</p>
3	Record families – List of available groups that can be expanded and collapsed to view and hide record types available for the record family. When you click a record family, a list of available records appears on the right.
4	Number of records – Number of records available for the corresponding record family. Expand the record family to view the record list.
5	Record list – List of records available for the record family. When you click a record name, the overview details display on the right and new menu options are available on the left.

For more information about the elements on the Record Customization (Beta) page, see [Record Customization \(Beta\) Information](#)

Record Customization Overview Page

Warning: Record Customization (Beta) is a beta feature. The contents of this feature are preliminary and may be changed or discontinued without prior notice. Any changes may impact the feature's operation with the NetSuite application. NetSuite warranties and product service levels shall not apply to the feature or the impact of the feature on other portions of the NetSuite application. NetSuite may review and monitor the performance and use of this feature. The documentation for this feature is also considered a beta version and is subject to revision. **Please review Beta Software Legal Notices. ORACLE CONFIDENTIAL. For authorized use only. Do not distribute to third parties. Pre-General Availability Draft: September 25, 2020.**

Note: To access this beta feature, you must apply for participation in a beta testing activity by clicking on the associated recruitment form link. On the recruitment form, you will be required to identify an authorized signatory to accept the Oracle Cloud Services Beta Trial License Agreement. After the Agreement is fully executed by your authorized signatory, your company will be provided with the beta feature for testing.

When you select a record, the Overview tab is automatically selected, and overview information for the record appears on the right. The overview includes primary information about the record, and an overview of fields, sublists, and access information. The Customization Overview section includes links to view fields, sublists, and access details, as well as links to create new fields and sublists.

The unique elements of the Overview page are identified in the following screenshot:

Primary Information	Customization Overview
<p>NAME Customer</p> <p>ID customer</p> <p>RECORD FAMILY Entity</p> <p>ORIGIN System</p>	<p>FIELDS 106 standard 5 custom + New Field</p> <p>SUBLISTS 15 standard + New Sublist</p> <p>ACCESS 1 users via global permissions 7 users via 3 roles 32 unused roles</p>

1	Primary Information – Primary information about the record, which includes: <ul style="list-style-type: none"> ■ Name – Record type name. ■ ID – Record ID ■ Description – Record description ■ Record Family – Group to which the record belongs. ■ Origin – Source of the record.
2	Fields – Information about the record fields. Includes: <ul style="list-style-type: none"> ■ Number of standard fields and number of custom fields available for the record. When you click an available link, the Fields & Sublists tab is selected and a list of standard or custom fields available for the record appears. ■ Option to create a new field. For more information, see the help topic Creating a Custom Field.
3	Sublists – Information about the record sublists. Includes: <ul style="list-style-type: none"> ■ Number of standard sublists and number of custom sublists available for the record. When you click an available link, the Fields & Sublists tab is selected and a list of standard or custom sublists available for the record appears. ■ If applicable for the record type, option to create a new sublist. For more information, see Custom Sublists.
4	Access – Number of users by roles and permissions that have access to the record, and number of unused roles that have access to the record. When you click an available link, the Access tab is selected and a list of roles with access to the record appears. The information displayed depends on which link you clicked.
5	Rename – Option to rename the record. This option is not available for all records. For more information, see the help topic Renaming Records and Transactions .

For more information about the elements on the Overview page, see [Record Customization \(Beta\) Information](#)

Record Customization Fields and Sublists Page

 **Warning:** Record Customization (Beta) is a beta feature. The contents of this feature are preliminary and may be changed or discontinued without prior notice. Any changes may impact the feature's operation with the NetSuite application. NetSuite warranties and product service levels shall not apply to the feature or the impact of the feature on other portions of the NetSuite application. NetSuite may review and monitor the performance and use of this feature. The documentation for this feature is also considered a beta version and is subject to revision. Please review [Beta Software Legal Notices](#). **ORACLE CONFIDENTIAL. For authorized use only. Do not distribute to third parties. Pre-General Availability Draft: September 25, 2020.**

 **Note:** To access this beta feature, you must apply for participation in a beta testing activity by clicking on the associated recruitment form link. On the recruitment form, you will be required to identify an authorized signatory to accept the Oracle Cloud Services Beta Trial License Agreement. After the Agreement is fully executed by your authorized signatory, your company will be provided with the beta feature for testing.

The Fields & Sublists page displays information about fields and sublists included on a record. You can access the fields and sublists information in the following ways:

- From the menu, click the **Fields & Sublists** tab.
- From the record overview, under the **Fields** heading, click the standard fields link.
- From the record overview, under the **Sublists** heading, click the standard sublists field.

The Fields & Sublists page has separate and subtabs. Click the subtab for the type of information you want to view. For more information, see [Fields Subtab on Fields and Sublists Page](#) and [Sublists Subtab on Fields and Sublists Page](#).

Fields Subtab on Fields and Sublists Page

When you click the Fields & Sublists tab or click the standard fields link on the record overview, the Fields tab is automatically selected. A list of standard fields available for the record appears with the option to make fields required. For more information, see [Making Fields Mandatory](#).

To view the sublists information, click the Sublists subtab.

The unique elements of the Fields page are identified in the following screenshot:

NAME	ID	TYPE	ORIGIN	MANDATORY
Account	accountnumber	Free-Form Text	System	<input type="checkbox"/>
Alcohol Recipient Type	alcoholrecipienttype		System	<input type="checkbox"/>
Alt. Email	altemail	Email Address	System	<input type="checkbox"/>
Alt. Phone	altphone	Phone Number	System	<input type="checkbox"/>
Annual Revenue (2)	custentity_esc_annual_revenue	Currency	bundle link test	<input type="checkbox"/>
Assigned Web Site	assignedwebsite	Web Site	System	<input type="checkbox"/>
Auto	autoname	Check Box	System	<input type="checkbox"/>
Balance	balance	Currency	System	<input type="checkbox"/>
Billing Rate Card	billingratecard	List/Record (Billing Rate Card)	System	<input type="checkbox"/>
Billing Schedule	billingschedule	Billing Schedule	System	<input type="checkbox"/>
Billing Transaction Form	billingtransactionform	Transaction Form	System	<input type="checkbox"/>
Billing Transaction Type	billingtransactiontype	Transaction Type	System	<input type="checkbox"/>
BRN	custentity_mv_brn	Free-Form Text	s	<input type="checkbox"/>

1	Fields information – Information about fields available for the selected record. Field information includes: <ul style="list-style-type: none"> ■ Name – Field name. ■ ID – Field ID. ■ Type – Type of field. For more information, see Field Types. ■ Origin – Source of the field. ■ Mandatory – Specifies whether the field is required for a specific record type. For more information, see Making Fields Mandatory. Sort fields details in ascending or descending order by clicking any of the headings.
2	New Field – Option to create a new field. For more information, see the help topic Creating a Custom Field .

For more information about the elements on the Fields page, see [Record Customization \(Beta\) Information](#)

Making Fields Mandatory

Customization (Beta) consolidates all fields on a record, enabling you to view and manage required settings all in one location. Record level customization provides a required option for a field. The required setting at the record level takes precedence over the required setting defined on forms, workflows, and user scripts.

When you make a field required, the change is automatically saved to the server. Required field settings specified in Record Customization apply only to the selected record type. If the field is used in another

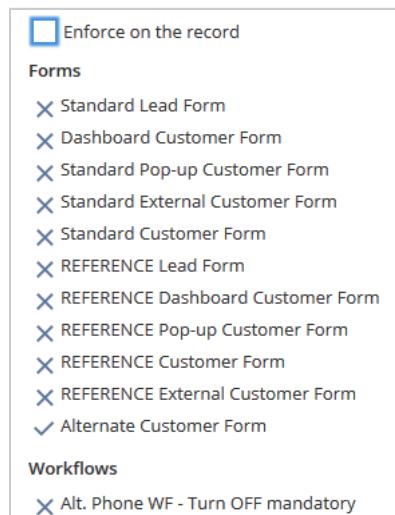
area, on other records for example, the mandatory setting change is not applied there. You can make both standard and custom fields required.

The Fields page shows a Mandatory box for any fields that you can make required. Depending on the required status of the field, this box may appear:

- Checked – always required
- Unchecked – not required
- Indeterminate state – required in some cases

The Mandatory box states are described later in this topic.

For more information about the required details of the field, click the information icon ⓘ. This icon appears when you point to the field name row. The following screenshot shows an example of the required information message:



Forms that use the field appear in a list. If the field has been made mandatory on the form, the form name appears with a check mark. If the field is not mandatory on the form, the form name appears with an X. Use the link to an individual form to change the mandatory status on the form.

To enforce the mandatory setting on all forms at the record level, check the Enforce on the record box.

A record-level mandatory setting takes precedence over the Mandatory box on the Custom Field definition page. The option to check the Mandatory box on the Custom Field page is independent from the record-level setting in Record Customization. The required setting on the Custom Field page does not enforce the record-level required setting.

To reverse the required setting at the record level, clear the Mandatory box or clear the Enforce on the record box.

The following table describes the various Mandatory box states:

Mandatory Box State	Description	Information Message
Mandatory box empty	The field is currently not set to required on any forms or is not defined by any user workflow. The field is not set to required at the record level.	All forms that use the field appear in the list with an X. The Enforce on record box is cleared.
Mandatory box checked	The field is currently set to required on all forms. The required setting is enforced at the record level.	All forms that use the field appear in the list with a check mark. The Enforce on the record box is checked.

Mandatory Box State	Description	Information Message
Mandatory box with indeterminate state	All or some of the individual forms that use this field have the required setting. The required setting is not enforced at the record level.	All or some of the forms that use this field appear with a check mark. The Enforce on the record box is cleared.
No Mandatory box	The field cannot be edited in Record Customization. The required setting is not applicable, for example, because of the type of field, and is therefore not editable.	There is no information message.

You cannot make checkbox, help, or inline HTML fields required, so for these types of fields a Mandatory box is not available.

Field Types

Type of field can be:

- Any of the available customs field types. For more information, see the help topic [Custom Field Types](#).
- Other - field does not belong to any known type of field. For example, Other type can refer to email addresses that are delimited by a semi-colon (;).

For field types of List/Record types, the name of the related record type appears in bold text. Types marked by bold text identify system records that cannot be customized. If the related List/Record type can be opened in Record Customization (Beta), the record type name appears as a link. Click the link to see the overview information for the related record type.

Sublists Subtab on Fields and Sublists Page

When you click the standard sublists link on the record overview, the Sublists tab is automatically selected. You can also access the Sublists tab from the Fields & Sublists tab. A list of all sublists available on the record appears. Expand and collapse the sublist to view and hide a list of fields available on the sublist.

To view the fields information, click the Fields subtab.

The unique elements of the Sublists page are identified in the following screenshot:

NAME	ID	TYPE	ORIGIN
ACT History	recmachcustrecord117	Related Record	Test bundle
Address Book	addressbook	Subrecord	System
Bulk Merge	bulkmerge	Subrecord	System
Campaigns	campaigns	Subrecord	System
Click-Stream hits	clickstreams	Subrecord	System
Contacts	contactroles	Subrecord	System
Access	giveaccess	Check Box	System
Confirm Password	passwordconfirm	Password	System
Contact	contactname	Free-Form Text	System
Contacts	contact	List/Record (Contact)	System
Email	email	Email Address	System
Notify	sendemail	Check Box	System
Password	password	Password	System

<p>1 Sublists information – List of sublists and field for the selected record. Sublist and field information includes:</p> <ul style="list-style-type: none"> ■ Name – Name of sublist (on top level node). ■ ID – script ID of sublist or related field. ■ Type – Type of sublist or type of field. For more information, see Sublist Types. ■ Origin – Source of the field or sublist. <p>Sort sublists details in ascending or descending order by clicking any of the headings.</p>
<p>2 New Sublist – If applicable for the record type, option to create a new sublist. For more information, see Custom Sublists.</p>

For more information about the elements on the Sublists page, see [Record Customization \(Beta\) Information](#)

Sublist Types

Type of sublist can be:

- Subrecord – standard sublists that are an integral part of a specific record.
- Related record – sublists that reference another record through List/Record relation field. Sublist references another record type instance.
- Saved search – sublists that show record instances of saved search results.

When you expand the sublist name, fields available in the sublist display.

Record Customization Access Page

 **Warning:** Record Customization (Beta) is a beta feature. The contents of this feature are preliminary and may be changed or discontinued without prior notice. Any changes may impact the feature's operation with the NetSuite application. NetSuite warranties and product service levels shall not apply to the feature or the impact of the feature on other portions of the NetSuite application. NetSuite may review and monitor the performance and use of this feature. The documentation for this feature is also considered a beta version and is subject to revision. **Please review Beta Software Legal Notices. ORACLE CONFIDENTIAL. For authorized use only. Do not distribute to third parties. Pre-General Availability Draft: September 25, 2020.**

 **Note:** To access this beta feature, you must apply for participation in a beta testing activity by clicking on the associated recruitment form link. On the recruitment form, you will be required to identify an authorized signatory to accept the Oracle Cloud Services Beta Trial License Agreement. After the Agreement is fully executed by your authorized signatory, your company will be provided with the beta feature for testing.

When you click the Access tab, access control details for all roles and employees that have access to a specific record or field appear. The Access page is divided into separate records and Fields subtabs. Click the subtab for the type of information you want to view.

For more information see [Records Subtab on Access Page](#) and [Fields Subtab on Access Page](#).

Records Subtab on Access Page

When you click the Access tab, the Record tab is automatically selected, and access information for the associated record displays. The information includes the:

- Users with global permissions
- Available roles and users who have access to the record
- Level of access a user has to a record
- Restrictions that are applicable for selected role/user and record

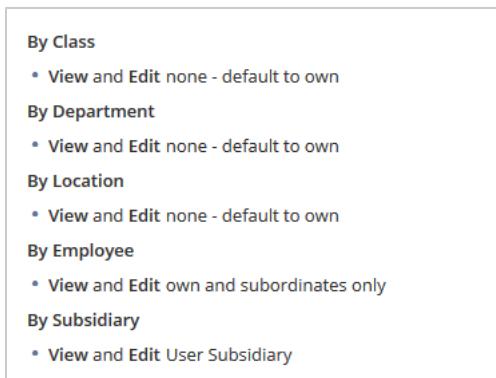
You can expand and collapse roles to view and hide a list of users assigned to the role. You can also view access control details for a role or user by clicking on the role name or user name. For more information, see the help topic [NetSuite Users & Roles](#).

To view fields access information, click the Fields subtab.

The unique elements of the Record page are identified in the following screenshot:

ROLE/USER	LEVEL	RESTRICTIONS
Administrator (1)	Full	
A Wolfe-admin	Full	
Controller (1)	Full	
Customer Center (1)	Edit	
E-Commerce Site Manager (1)	Full	
Employee Center (1)	Edit	By Class, Department ⓘ
Marketing Manager (1)	Full	
A Wolfe-admin	Full	
Receivables Clerk (1)	Full	
A Wolfe-admin	Full	
Sales Manager (1)	Full	

When you click an information icon in the Restrictions column, a message similar to the following appears:



- 1 **Record access control information** – Information about access control for the selected record. Record information includes:
- **Role/User** – All roles with access to the record and employees (expanded under the role) assigned to the role. A number in brackets beside the role name represents how many users are assigned to the role.

	<ul style="list-style-type: none"> ■ Level – Access level each role or user has for the record. If a global permission is assigned to a user, information about the access-level assigned for the user displays here. For more information, see the help topic Using the Global Permissions Feature. <p>Access level information can differ based on the selected role or user. The access level always displays the applicable access level for the selection of role, user, or both.</p> <ul style="list-style-type: none"> ■ Restrictions – Applicable access level restrictions for a role. Restrictions can include, for example, By Subsidiary, By Employee, By Location, By Class, and By Department. For more details about the restrictions, click the information icon.  <p>Depending on the selected user and role combination, restrictions can result in various applicable access levels for the record. For more information, see the help topic Permissions and Restrictions.</p>
2	<p>Hide Empty Roles – Option to filter out roles that can access the record but currently have no users assigned. Click to turn this option on or off.</p>

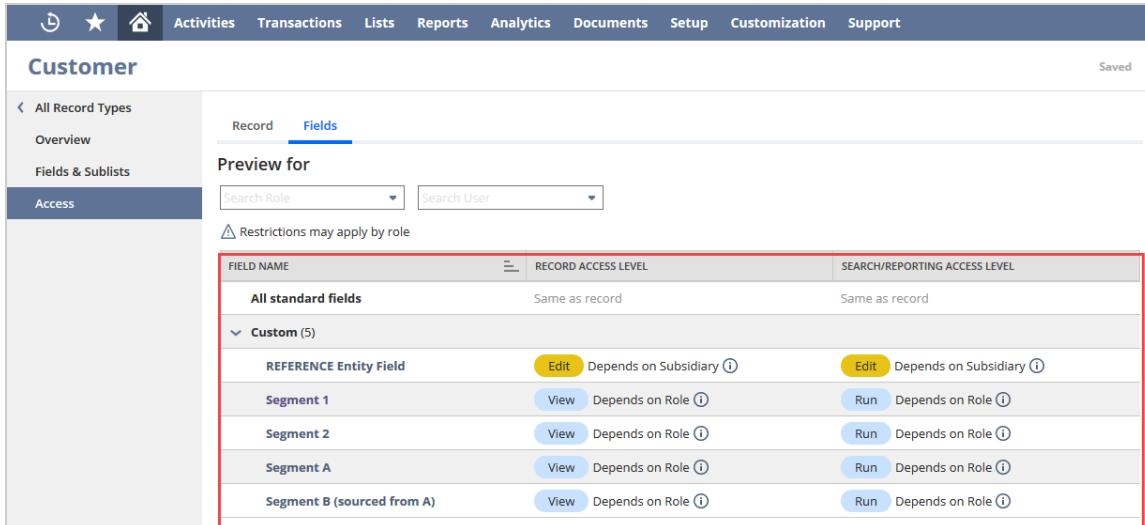
For more information about the elements on the Record page, see [Record Customization \(Beta\) Information](#)

Fields Subtab on Access Page

When you click the Fields subtab, field access information for the record appears. The information includes standard and custom fields and the corresponding access levels for the record and for search and reporting. You can search by role or user to view only the applicable access levels for the fields that a role or user can access.

To view record access information, click the Record subtab.

The unique elements of the Fields page are identified in the following screenshot:



FIELD NAME	RECORD ACCESS LEVEL	SEARCH/REPORTING ACCESS LEVEL
All standard fields	Same as record	Same as record
Custom (5)		
REFERENCE Entity Field	Edit Depends on Subsidiary 	Edit Depends on Subsidiary 
Segment 1	View Depends on Role 	Run Depends on Role 
Segment 2	View Depends on Role 	Run Depends on Role 
Segment A	View Depends on Role 	Run Depends on Role 
Segment B (sourced from A)	View Depends on Role 	Run Depends on Role 

Field access control information – Information about access control for a record field. Field information includes:

- **Field Name** – Name.
- **Record Access Level** – Record access level for the field. For more information, see [Access Levels for Record Fields](#).
- **Search/Reporting Access Level** – Search and reporting access levels for the field. For more information, see [Access Levels for Record Fields](#).

Record access and search/reporting access level restrictions can apply. Restrictions can include, for example, By Subsidiary, By Employee, By Location, By Class, and By Department. For more details about the restrictions, click the information icon. ⓘ

Depending on the selected user and role combination, restrictions can result in various applicable access levels for the record. For more information, see the help topic [Permissions and Restrictions](#).

All standard fields are grouped because their permissions are based on record permissions on the Record tab. Custom fields can have individual access level settings.

Sort fields access details in ascending or descending order by clicking the heading.

For more information about the elements on the (access) Fields page, see [Record Customization \(Beta\) Information](#).

Access Levels for Record Fields

For standard fields, field permissions used on record, and field permissions for search and reporting are based on record type permission available on record access subtab. Role restrictions may be applicable.

Custom fields can have individual access level settings as well as role restrictions. For more information, see the help topic [Restricting Access to Custom Fields](#). If a custom field access level information is dependent on selected role, a message such as **Depends on Role** appears with an information icon ⓘ.

For more details, click the information icon.

The following screenshot shows an example of the By Role information details:



For access levels that have dependencies, when no role is selected, the default access level of the custom field displays. The default access level of a custom field is used for any role that is not set up specifically on the field's access. You can use the Preview for filter to select the role identified in the information details. The field displays both the dimmed default access level and the overriding access level that is effective for the selected role.

The following screenshot shows the default access levels for custom fields on the Customer record:

FIELD NAME	RECORD ACCESS LEVEL	SEARCH/REPORTING ACCESS LEVEL
All standard fields	Same as record	Same as record
Custom (5)		
REFERENCE Entity Field	Edit Depends on Subsidiary ⓘ	Edit Depends on Subsidiary ⓘ
Segment 1	View Depends on Role ⓘ	Run Depends on Role ⓘ
Segment 2	View Depends on Role ⓘ	Run Depends on Role ⓘ
Segment A	View Depends on Role ⓘ	Run Depends on Role ⓘ
Segment B (sourced from A)	View Depends on Role ⓘ	Run Depends on Role ⓘ

The following screenshot shows the access levels for the same fields as in the preceding screenshot when the Administrator role is selected in the Preview for filter. The dimmed View access level is the default access level for the fields and the Edit access level shown in orange is the access level for users with the Administrator role. The applicable access level for the selected role overrides the default access level.

FIELD NAME	RECORD ACCESS LEVEL	SEARCH/REPORTING ACCESS LEVEL
All standard fields	Full	Full
REFERENCE Entity Field	Edit View	Edit Depends on Subsidiary
Segment 1	Run Edit	Run Edit
Segment 2	Run Edit	Run Edit
Segment A	Run Edit	Run Edit
Segment B (sourced from A)	Run Edit	Run Edit

Custom Transactions

The Custom Transactions feature lets Administrators and users with the Custom Transaction Types permission create transaction types tailored to your business needs.

 **Note:** Processed lines for custom transaction types contribute to the Monthly Transaction Lines metric that counts toward maximum limits for your NetSuite service tier. For more information, see the help topic [Transaction Types Included in Monthly Transaction Lines Metric](#).

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 [and](#).

 **Important:** SuiteScript does not support direct access to the NetSuite UI through the Document Object Model (DOM). You should access the NetSuite UI only by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic [SuiteScript 2.x Custom Pages](#).

Custom transactions are supported in SuiteCloud Development Framework (SDF). SDF is a development framework that you can use to create SDF SuiteApps, or to customize NetSuite accounts, using an integrated development environment (IDE) on your local computer. SuiteCloud projects are file-based and use XML definitions of custom NetSuite objects. For more information, see the help topic [SDF Custom Object and File Development in SuiteCloud Projects](#).

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](#)

CSV Import Assistant, SuiteScript, and SOAP and REST 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](#).
- You can interact with custom transaction instances by using REST web services. REST web services support basic, journal, and header only custom transactions. For details, see the help topic [Using CRUD Operations on Custom Records, Custom Lists, and Custom Transactions v1](#).

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 [Multiple Custom Transaction Styles Supported](#).

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 nonoperational 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 is 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.

Financial	Reports	Documents	Setup	Support
Financial Overview				
Banking	▶			
Adjustments	▶	Bad Debt	▶	
Lists	▶	Fixed Asset Depreciation	▶	
Inventory	▶	Non-Operational Income	▶	New
Billing	▶			

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:

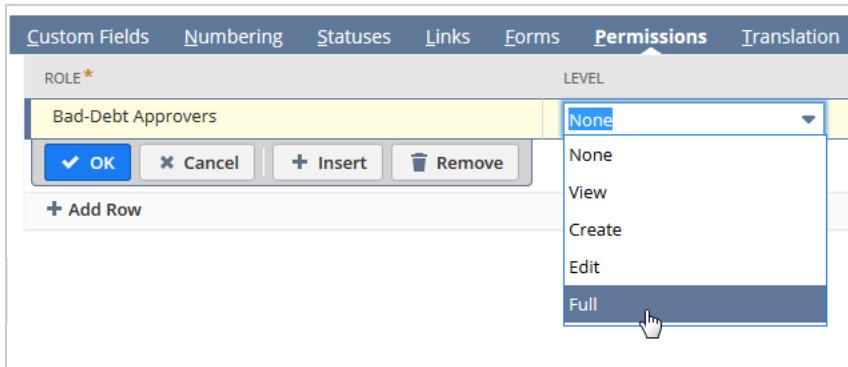
- Each custom transaction type can have its own numbering, permissions, and workflows.
- Custom transaction types can include custom transaction fields that you define.
- You can create multiple custom entry forms for each transaction type.
- You can see what changes have been made to the custom transaction 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 REST web services. REST web services support basic, journal, and header only custom transactions. For details, see the help topic [Using CRUD Operations on Custom Records, Custom Lists, and Custom Transactions v1](#).
- 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 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 included in SuiteApps packaged with SuiteCloud Development Framework (SDF). SDF is a development framework that you can use to create SuiteApps from an integrated development environment (IDE) on your local computer. For more information, see the help topic [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 topics [SuiteCloud Development Framework Overview](#) and [Custom Transaction Record Types as XML Definitions](#).
- 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](#).

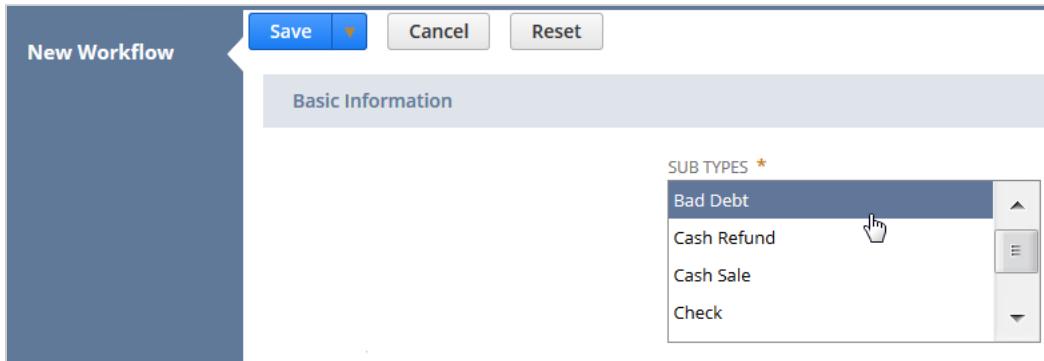
Support for these features makes it possible for each transaction type to have its own unique behavior and processing. These advantages can be critical as you develop a series of discrete transaction types to meet various needs.

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](#).

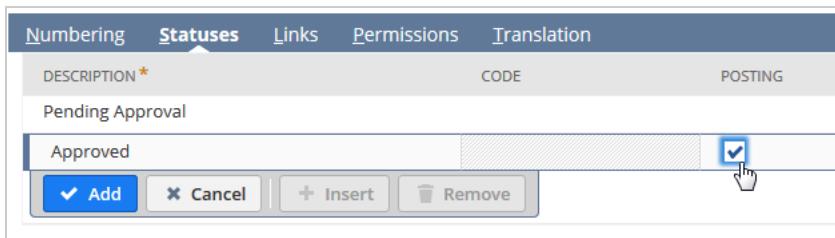
For example, suppose you have a custom transaction type called Bad Debt, which you use to account for debts that are not collectible. For this type, you may want to restrict access to a limited set of users. You can manage this access on the Permissions subtab by using the transaction type's Level sublist.



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.



Multiple Custom Transaction Styles Supported

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, whereas 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 nonprofit 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.

Primary Information		Summary
INVOICE # To Be Generated	END DATE <input type="text"/>	SUBTOTAL 119.70
CUSTOMER/JOB * Green Corp.	POSTING PERIOD * Feb 2019	DISCOUNT ITEM 0.00
JOB <input type="text"/>	DUE DATE <input type="text"/>	TAX
STATUS Pending	PO # <input type="text"/>	SHIPPING COST 11.94
DATE * 2/5/2019	MEMO <input type="text"/>	TOTAL 131.64
START DATE <input type="text"/>		
Sales Information		
SALES REP <input type="text"/>	OPPORTUNITY <input type="text"/>	PARTNER <input type="text"/>
Classification		
DEPARTMENT <input type="text"/>	CLASS <input type="text"/>	
Items Shipping Billing Accounting Relationships Communication		
Account Information ACCOUNT * <input type="text" value="1100 Accounts Receivable"/>		Tax Information <input type="checkbox"/> TAXABLE TAX -Not Taxable- TAX %

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.

For information about available sales and purchase functionality in custom transactions, see [Sales and Purchase Functionality Available in Custom Transactions](#).

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](#).

- Apply Subtab Support - Sales custom credit transactions support the Apply subtab, where you can apply transformed custom transactions to the source transaction. The Apply subtab has the same fields as the Apply subtab for credit memo and displays the same set of transactions. For more information see the help topic [Applying a Customer Credit Memo](#).
- 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 [Setting 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 [Expense Amortization](#).

- Currency Revaluation – You can enter your transaction in various 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 toward 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 Payment Installments](#).
- 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 order to sales custom transaction type
- Sales custom transaction type to customer payment
- Sales custom transaction type to customer refund
- Sales custom transaction to cash sale
- Cash sale to sales custom transaction
- Purchase order to purchase custom transaction
- Return authorization to sales custom transaction
- Vendor bill to purchase custom transaction
- Purchase custom transaction to vendor bill
- Vendor return authorization to purchase custom transaction
- Sales custom transaction to cash refund
- Sales custom transaction to credit memo
- Purchase custom transaction to vendor credit
- Purchase custom transaction to vendor payment

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.

Custom Transaction	Standard Transaction
Debit sales custom transaction	Invoice
Credit sales custom transaction	Credit
Debit purchase custom transaction	Bill
Credit purchase custom transaction	Bill Credit

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 can be used in CSV imports and are available in various areas of NetSuite. To learn more about sales and purchase custom transaction types in integrations, see the following sections.

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 about 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 about 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 | 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 another 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 another 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

```

1  CustomSaleItem item = new CustomSaleItem();
2  item.setItem(createRecordRef("39"));
3
4  CustomSaleItemList itemList = new CustomSaleItemList();
5  itemList.setItem(new CustomSaleItem[] {item});
6

```

```

7 | CustomSale record = new CustomSale();
8 | record.setTranType(createRecordRef("105"));
9 | record.setEntity(createRecordRef("84"));
10 | record.setLocation(createRecordRef("2"));
11 | record.setItemList(itemList);
12 |
13 | c.addRecord(record);

```

SOAP Request

```

1 <add xmlns="urn:messages_2019_2.platform.webservices.netsuite.com">
2   <record xsi:type="ns7:CustomSale" xmlns:ns7="urn:customization_2019_2.setup.webservices.netsuite.com">
3     <ns7:tranType internalId="105" xsi:type="ns8:RecordRef" xmlns:ns8="urn:core_2019_2.platform.webservices.netsuite.com"/>
4     <ns7:entity internalId="84" xsi:type="ns9:RecordRef" xmlns:ns9="urn:core_2019_2.platform.webservices.netsuite.com"/>
5     <ns7:location internalId="2" xsi:type="ns10:RecordRef" xmlns:ns10="urn:core_2019_2.platform.webservices.netsuite.com"/>
6     <ns7:itemList replaceAll="false" xsi:type="ns7:CustomSaleItemList">
7       <ns7:item xsi:type="ns7:CustomSaleItem">
8         <ns7:item internalId="39" xsi:type="ns11:RecordRef" xmlns:ns11="urn:core_2019_2.platform.webservices.netsuite.com"/>
9       </ns7:item>
10      </ns7:itemList>
11    </record>
12  </add>

```

SOAP Response

```

1 <addResponse xmlns="urn:messages_2019_2.platform.webservices.netsuite.com">
2   <writeResponse>
3     <platformCore:status isSuccess="true" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
4       <platformCore:statusDetail>
5         <platformCore:afterSubmitFailed>false</platformCore:afterSubmitFailed>
6       </platformCore:statusDetail>
7     </platformCore:status>
8     <baseRef internalId="176" type="customSale" xsi:type="platformCore:RecordRef" xmlns:platformCore="urn:core_2019_2.platform.web
9       </writeResponse>
10    </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

```

1 InitializeRef reference = new InitializeRef();
2   reference.setInternalId("76");
3   reference.setType(InitializeRefType.customPurchase);
4
5   InitializeAuxRef auxReference = new InitializeAuxRef();
6   auxReference.setType(InitializeAuxRefType.tranType);
7   auxReference.setscriptId("custompurchase105"); // or auxReference.setInternalId("105");
8
9   InitializeRecord initializeRecord = new InitializeRecord();
10  initializeRecord.setType(InitializeType.customPurchase);
11  initializeRecord.setReference(reference);
12  initializeRecord.setAuxReference(auxReference);
13
14  CustomPurchase recordToAdd = (CustomPurchase) c.initialize(initializeRecord);

```

SOAP Request

```

1 <initialize xmlns="urn:messages_2019_2.platform.webservices.netsuite.com">
2   <initializeRecord>
3     <ns7:type xmlns:ns7="urn:core_2019_2.platform.webservices.netsuite.com">customPurchase</ns7:type>
4     <ns8:reference internalId="76" type="customPurchase" xmlns:ns8="urn:core_2019_2.platform.webservices.netsuite.com"/>
5     <ns9:auxReference scriptId="custompurchase105" type="tranType" xmlns:ns9="urn:core_2019_2.platform.webservices.netsuite.com"/>
6   </initializeRecord>

```

7 | </initialize>

SOAP Response

```

1 <readResponse>
2   <platformCore:status isSuccess="true" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com"/>
3   <record xsi:type="setupCustom:CustomPurchase" xmlns:setupCustom="urn:customization_2019_2.setup.webservices.netsuite.com">
4     <setupCustom:createdDate>2019-09-10T07:08:00.000-07:00</setupCustom:createdDate>
5     <setupCustom:lastModifiedDate>2019-09-10T07:08:00.000-07:00</setupCustom:lastModifiedDate>
6     <setupCustom:tranType internalId="105" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
7       <platformCore:name>my Purchase B</platformCore:name>
8     </setupCustom:tranType>
9     <setupCustom:billAddressList internalId="38" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
10       <platformCore:name>Default Billing</platformCore:name>
11     </setupCustom:billAddressList>
12     <setupCustom:account internalId="6" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
13       <platformCore:name>Fees Receivable</platformCore:name>
14     </setupCustom:account>
15     <setupCustom:entity internalId="105" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
16       <platformCore:name>Acme Medical Supply</platformCore:name>
17     </setupCustom:entity>
18     <setupCustom:subsidiary internalId="1" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
19       <platformCore:name>Parent Company</platformCore:name>
20     </setupCustom:subsidiary>
21     <setupCustom:postingPeriod internalId="353" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
22       <platformCore:name>Sep 2019</platformCore:name>
23     </setupCustom:postingPeriod>
24     <setupCustom:tranDate>2019-09-10T00:00:00.000-07:00</setupCustom:tranDate>
25     <setupCustom:currencyName>US USD</setupCustom:currencyName>
26     <setupCustom:billingAddress xmlns:platformCommon="urn:common_2019_2.platform.webservices.netsuite.com">
27       <platformCommon:country>_unitedStates</platformCommon:country>
28       <platformCommon:addressee>Acme Medical Supply</platformCommon:addressee>
29       <platformCommon:addr1>1234 Sepulveda Blvd</platformCommon:addr1>
30       <platformCommon:city>Los Angeles</platformCommon:city>
31       <platformCommon:state>CA</platformCommon:state>
32       <platformCommon:zip>94321</platformCommon:zip>
33       <platformCommon:addrText>Acme Medical Supply1234 Sepulveda BlvdLos Angeles CA 94321</platformCommon:addrText>
34       <platformCommon:override>false</platformCommon:override>
35     </setupCustom:billingAddress>
36     <setupCustom:exchangeRate>1.0</setupCustom:exchangeRate>
37     <setupCustom:dueDate>2019-09-10T00:00:00.000-07:00</setupCustom:dueDate>
38     <setupCustom:paymentHold>false</setupCustom:paymentHold>
39     <setupCustom:memo>tralala memo</setupCustom:memo>
40     <setupCustom:currency internalId="1" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
41       <platformCore:name>US USD</platformCore:name>
42     </setupCustom:currency>
43     <setupCustom:transactionNumber>To Be Generated</setupCustom:transactionNumber>
44     <setupCustom:expenseList>
45       <setupCustom:expense>
46         <setupCustom:line>1</setupCustom:line>
47         <setupCustom:category internalId="2" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
48           <platformCore:name>Entertainment</platformCore:name>
49         </setupCustom:category>
50         <setupCustom:account internalId="59" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
51           <platformCore:name>Advertising</platformCore:name>
52         </setupCustom:account>
53         <setupCustom:amount>123.0</setupCustom:amount>
54         <setupCustom:isBillable>false</setupCustom:isBillable>
55       </setupCustom:expense>
56     </setupCustom:expenseList>
57     <setupCustom:accountingBookDetailList xmlns:platformCommon="urn:common_2019_2.platform.webservices.netsuite.com">
58       <platformCommon:accountingBookDetail>
59         <platformCommon:accountingBook internalId="2" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
60           <platformCore:name>Secondary accounting book</platformCore:name>
61         </platformCommon:accountingBook>
62         <platformCommon:currency internalId="1" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
63           <platformCore:name>US USD</platformCore:name>
64         </platformCommon:currency>
65         <platformCommon:exchangeRate>1.0</platformCommon:exchangeRate>
66       </platformCommon:accountingBookDetail>
67       <platformCommon:accountingBookDetail>
68         <platformCommon:accountingBook internalId="3" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">

```

```

69 |     <platformCore:name>Third accounting book</platformCore:name>
70 |     </platformCommon:accountingBook>
71 |     <platformCommon:currency internalId="1" xmlns:platformCore="urn:core_2019_2.platform.webservices.netsuite.com">
72 |         <platformCore:name>US USD</platformCore:name>
73 |     </platformCommon:currency>
74 |     <platformCommon:exchangeRate>1.0</platformCommon:exchangeRate>
75 |     </platformCommon:accountingBookDetail>
76 |     </setupCustom:accountingBookDetailList>
77 | </record>
78 | </readResponse>

```

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 required 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.

Level	Users can:	Users cannot:
View	View transaction type definitions.	Edit, create, or delete transaction types.
Create	View and create transaction types.	Edit or delete transaction types.
Edit	View, create, and edit transaction types.	Delete transaction types.
Full	View, create, edit, and delete transaction types.	—

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, select 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 about 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
- Journal
- Header Only
- Sales
- Purchase

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.

Non-Operational Income Entry

NOIE1002

Primary Information

ENTRY NO.	DATE
NOIE1002	2/6/2015
CURRENCY	POSTING PERIOD
USA	Feb 2015
EXCHANGE RATE	
1.00	

Classification

SUBSIDIARY
Parent Company

Lines

ACCOUNT	DEBIT	CREDIT	MEMO
694 Rental Income		2,675.00	
6261 CDPS	2,675.00		

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 permit postings to elimination subsidiaries. Basic and Header Only styles do not permit 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 originate from standard or custom fields on the transaction instance entry form. It can also originate 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.

Bad Debt Accrual

Primary Information

CUSTOM FORM *	EXCHANGE RATE *
Custom Bad Debt Accrual Form	1.00

ENTRY NO.	DATE *
To Be Generated	1/20/2015

CURRENCY *	CLASS
USA	

Classification

SUBSIDIARY *	CLASS
Parent Company	

DEPARTMENT

Transactions

USE	NAME	NUMBER	DAYS OVERDUE	DATE	DUE DATE
<input checked="" type="checkbox"/>	Invoice #3	3	71		11/10/2014
<input type="checkbox"/>	Invoice #1	1	66		11/15/2014
<input type="checkbox"/>	Invoice #2	2	56		11/25/2014

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 Plug-in Lines feature. Similarly, users can enter Header Only transaction instances before the corresponding transaction type is linked to a plug-in implementation, if the type does not have a posting status.

For more information, see the following topics.

- [Custom GL Lines Plug-in Overview](#)
- [Custom Transaction Type Association with a Custom GL-Lines Plug-in Implementation](#)

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 nonprofit 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. In the **ID** field, enter a unique alphanumeric ID for the transaction type. For information about best practices and naming conventions, see [Conventions for Naming Custom Objects](#). For information about changing an existing ID, see [Changing the ID of a Custom Object](#).
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 permits 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 void only transactions that do not use inventory items.
6. If the transaction should close the sales order, check the **Ability to Close Sales Order** box. The sales custom transaction can close a sales order only if the transaction is of debit type and if it is posting. For more information, see [Ability to Close Sales Order](#).
7. 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.

8. Specify any custom segments as appropriate. For more information, see [Custom Segments](#).
9. 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](#).

10. By default, a new custom transaction type has a nonposting status. If you want instances of the transaction to post, go 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 nonposting. For help creating statuses, see [Statuses for a Custom Transaction Type](#).
11. (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 autogenerated 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** – Select 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](#).

12. 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](#).

Ability to Close Sales Order

Sales custom transactions can close the sales order. When configuring a sales custom transaction, choose one of the following options:

Check the Ability to Close Sales Order Box

Uses the custom transaction in the sales workflow instead of the invoice. Sales custom transactions can close the sales order only if the transaction is of debit type and if it is posting. To specify that the Sales custom transaction is of debit type and is posting:

- On the Accounting subtab, clear the Credit box.
- On the Statuses subtab, check the Posting box, or add a status and check the related Posting box.

When you transform the standard sales order transaction to a custom transaction, the results are as follows:

- The sales order transactions are marked as billed, which effectively closes the sales order.
- A standard link of type Order Bill/Invoice is created between the two transactions. The relationship between the order bill and invoice transactions is, therefore, the same as the relationship between a standard sales order and standard invoice.
- A standard item fulfillment transaction is created as part of this flow to cover the Cost of Goods Sold (COGS) impact for the inventory items sold. The custom transaction does not handle the COGS impact.
- For revenue recognition on the sales order transaction, the custom transaction is considered as a billing event for recognizing the revenue.

Clear the Ability to Close Sales Order Box

The custom transaction should be a stand-alone transaction or an additional transaction to be used within the existing sales workflow. The results are as follows:

- A generic **Transformation** link type is created between sales orders and custom transactions. This link recognizes only the **created from** relationship. In this case, the custom transaction behaves as a stand-alone transaction.
- If you sell an inventory item, COGS is covered on the custom transaction because there is no link to the item fulfillment.

System Notes for Custom Transaction Type Configuration



Note: This topic applies to System Notes v2 only. For information about System Notes, see the help topic [System Notes Overview](#).

NetSuite logs system notes for changes to custom transaction type configuration. Use the System Notes v2 Workbook to view changes to the configuration of custom transaction types. For more information, see the help topic [System Notes v2 Workbook](#).

You can also access system notes for custom transaction types using the System Notes link on the Custom Transaction Types page.

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 can 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 void only 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**.

For information about best practices and naming conventions, see [Conventions for Naming Custom Objects](#). For information about changing an existing ID, see [Changing the ID of a Custom Object](#).

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 autonumbering for the type. For more information, see [Numbering for a Custom Transaction Type](#).

- **Transaction Numbers** – Configure autogenerated 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 nonposting 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** – Select 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 upper 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

i 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 can 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](#)
- [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 autonumbering scheme for instances of this transaction type. Similarly, if you have a OneWorld account, you can refer to subsidiary when configuring your autonumbering 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. 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](#).

i 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 [Creating Custom Transaction Body Fields](#) and [Creating 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 required 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 autonumbered. Autonumbering 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 individual transactions are drawing money from the same account, then their document numbers are part of a single sequence even if they are not the same types.

Document numbers can also be required because of government numbering requirements. In those cases, NetSuite customers may 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

Autogenerated 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 autogenerated numbering is not set up in your account, transaction numbers begin at 1. For more information about the records and transactions available for autonumbering, see the help topic [Records and Transactions Available for Auto-Numbering](#).



Important: Historical transaction records can 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 autogenerated 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.

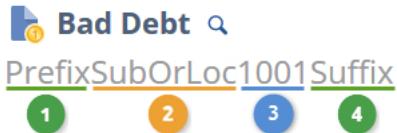
Components of a Custom Transaction Type's Entry Number

At a minimum, the entry number for a transaction instance includes an autogenerated 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, NetSuite 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.



Each component, along with information about configuring it, is described in the following table.

Callout	Description
1	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 .
2	This value corresponds with the Transaction Number 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: <ul style="list-style-type: none"> 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. The transaction instance includes a value in the relevant field (Subsidiary or Location). The relevant subsidiary or location record includes a value in its Transaction Number Prefix field. For more details, see Defining Numbering Preferences for Subsidiaries and Locations.
3	This value is the autogenerated number. In some cases, all instances of a transaction type are numbered as part of a single sequence. In other cases, there can be multiple sequences. The system uses the following rules: <ul style="list-style-type: none"> If the entry number does not include a Transaction Number Prefix value derived from a subsidiary or location record, the system uses a generic numbering sequence not tied to any classification. The initial number of this sequence is the value saved in the transaction type's Initial Number field. For help setting this field, see Defining Numbering for a Custom Transaction Type. In some cases, the entry number of the transaction instance does include a Transaction Number Prefix value derived from a subsidiary or location record. In these situations, the system uses numbering specific to that combination of subsidiary and transaction type (or the combination of location and transaction type). The initial number for each sequence is taken from the location or subsidiary record. For more details, see Defining Numbering Preferences for Subsidiaries and Locations.
4	This value corresponds with the transaction type's Suffix 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 .

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. Go 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. Then click the **Transaction Numbers** subtab.
 - Go to Setup > Company > Auto-Generated Numbers. Then click the **Transaction Numbers** subtab and locate the 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 autogenerated number. It also occurs before the subsidiary or location prefix, which you may have enabled in the previous step.
4. (Optional) Enter numbers or letters in the **Suffix** field. The suffix is displayed after each autogenerated number.
5. If appropriate, enter a value from 1 to 20 in the **Minimum Digits** field. This minimum applies only to the autogenerated 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 autogenerated number of 1 becomes 0001.
6. If appropriate, check **Update Initial Number** and then 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, NetSuite (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 to let users override an autogenerated 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 autogenerated numbers begin at the next number. Each number can be manually set to a lower value, but autogenerated 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**.

For examples of how to set up transactions to use transaction type and location prefixes, see [Autogenerated Transaction Number with Transaction Type and Location Prefixes Examples](#).

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 autogenerated number 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. Go 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 Number Prefix** field includes the prefix for this record. Make any changes as appropriate.

 **Important:** The value in the Transaction Number 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 **Transaction Numbers** subtab. If the Transaction Numbers subtab is not displayed, you have not yet created a transaction type that has the Use Subsidiary or Use Location option enabled. For more information, see [Creating a Custom Transaction Type](#).
 - b. In the **Type** column, locate the name of your transaction type.
 - c. If you want to update the initial number used for transaction numbering, check **Update Initial Number**.
 - d. 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. For example, if the Initial Number field is set to 100 and the value in the current Number column is 105, then the numbering begins at 106.

 **Important:** If the Transaction Number Prefix field (in the location or subsidiary record) 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. Click **Save**.

For examples of how to set up transactions to use transaction type and location prefixes, see [Autogenerated Transaction Number with Transaction Type and Location Prefixes Examples](#).

Autogenerated Transaction Number with Transaction Type and Location Prefixes Examples

A custom transaction instance can have a predefined transaction type prefix and a location or subsidiary prefix. The following example demonstrates how to set up a custom purchase type transaction to include both a transaction type prefix and a location prefix. It also describes how to set the initial number NetSuite uses for the autogenerated number.



Note: Setting up transaction numbers that use a subsidiary (available in OneWorld accounts) prefix is similar to setting up transaction numbers that use a location prefix.

See the following sections:

- [Transaction Type Prefix](#)
- [Location Prefix](#)
- [Transaction Instance](#)
- [Transaction Number Examples](#)

Transaction Type Prefix

You set up the transaction type prefix in the custom transaction type record. In the following example, on the Transaction Number sublist:

- The Use Location box is checked. With this box checked, the resulting transaction number includes the location prefix if one is defined on the location record.
- The Prefix field has a transaction type prefix of PURCH. This prefix is added to the transaction numbers of all instances of this transaction type. In the generated transaction number, this prefix precedes the location (or subsidiary) prefix if one is used.
- The Update Initial Number box is checked, and an initial number is defined in the Initial Number field. This setting is optional and defines 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. This initial number is used for transactions that do not use a location (or subsidiary) prefix. If a location (or subsidiary) prefix is used, this number is ignored and the initial number setting on the location (or subsidiary) record is used instead.

The screenshot shows the 'Custom Transaction Type' setup screen in NetSuite. At the top, there are navigation links: Activities, Transactions, Lists, Reports, Analytics, Documents, Setup, Customization, and more. Below the title 'Custom Transaction Type', there are buttons for Save, Cancel, Change ID, and Actions. The 'NAME *' field contains 'ABC Cust Purchase Type Trans'. The 'ID' field is 'custompurchase_jlz_cust_purch_trans' and the 'INTERNAL ID' is '111'. The 'TRANSACTION STYLE' is set to 'Purchase'. On the right, there are checkboxes for 'ALLOW VOID TRANSACTIONS USING REVERSING JOURNALS' and 'UPDATE INITIAL NUMBER'. Under 'CLASS', 'DEPARTMENT', and 'LOCATION', dropdown menus show 'None' with 'MANDATORY' checked. The 'Transaction Numbers' subtab is active, showing 'USE LOCATION' checked, 'PREFIX' set to 'PURCH', and 'MINIMUM DIGITS' empty. The 'INITIAL NUMBER' is set to '35' with 'CURRENT NUMBER' at '31'. At the bottom, there are Save, Cancel, Change ID, and Actions buttons.

For more information, see [Creating a Custom Transaction Type](#).

Location Prefix

You set up the location prefix on the location record.

Note: If you are using a custom entry form, ensure that the custom form makes the Location field visible. For help with custom forms, see [Adding Custom Forms for a Custom Transaction Type](#).

On the location record in the following example:

- The Transaction Number Prefix has a location prefix of KWLOC, prefix is added to the transaction numbers of all instances that are associated with this location. In the generated transaction number, this prefix follows the location (or subsidiary) prefix if one is used.
- The Transaction Numbers subtab, the Update Initial Number box is checked for the transaction type, and an initial number is defined in the Initial Number field. This setting is optional and defines the initial number that should be used for the first transaction instance that uses the location. If you do not enter a value, the system (in most cases) starts the sequence at 1. If a value is entered here, it overrides the initial number on the custom transaction type record, if one is defined.

The screenshot shows the 'Location' setup screen in SuiteBuilder. At the top, there are buttons for Save, Cancel, and Search, along with an Actions dropdown. Below these are various configuration fields: 'LOCATION IS INACTIVE' (checkbox), 'NAME *' (text input 'Warehouse 1'), 'PARENT LOCATION' (dropdown), 'LOCATION TYPE' (dropdown 'Warehouse'), 'TIME ZONE' (dropdown '(GMT-05:00) Eastern Time (US & Canada)'), 'LATITUDE' (text input), 'LONGITUDE' (text input), 'DOCUMENT NUMBER PREFIX' (text input), 'TRANSACTION NUMBER PREFIX' (text input 'KWLOC'), 'LOGO' (text input), 'LOCATION COSTING GROUP' (dropdown), and 'DEFAULT ALLOCATION PRIORITY' (checkboxes for 'MAKE INVENTORY AVAILABLE' and 'INCLUDE IN SUPPLY PLANNING'). There are also 'INCLUDE IN CONTROL TOWER' and 'BUSINESS AREA' dropdowns with '+' icons. A 'USE BINS' checkbox is checked. Below this is a table titled 'Transaction Numbers' with columns: TYPE, UPDATE INITIAL NUMBER, INITIAL NUMBER, and CURRENT NUMBER. It lists three rows: 'ABC Cust Purchase Type Trans' (checked), 'Meal' (unchecked), and 'Rewards Credit' (unchecked). The 'INITIAL NUMBER' column contains '110' for the first row, and the 'CURRENT NUMBER' column contains '102'. At the bottom are Save, Cancel, Search, and Actions buttons.

For more information, see [Defining Numbering Preferences for Subsidiaries and Locations](#)

Transaction Instance

The following screenshot of a custom transaction instance shows the results of the first custom purchase transaction instance that uses prefixes and autogenerated transaction numbering, based on the preceding setup examples.

The transaction number generated is PURCHKWLOC110:

- **PURCH** is the transaction type prefix defined on the custom transaction type record.
- **KWLOC** is the location prefix defined on the location record.
- **110** is the initial number defined on the location record.

ABC Cust Purchase Type Trans

Acme Medical Supply

Primary Information

TRANSACTION NUMBER PURCHKWLOC110	CURRENCY USA	DATE 15.12.2021
REFERENCE NO.	EXCHANGE RATE 1.00	POSTING PERIOD Dec 2021
VENDOR Acme Medical Supply	<input type="checkbox"/> PAYMENT HOLD	MEMO
ACCOUNT Petty Cash	DUE DATE 13.2.2022	

Classification

LOCATION Warehouse 1

EXCLUDE FROM GL AUDIT NUMBERING

Expenses and Items **Billing** **Landed Cost** **Relationships** **Communication** **Related Records** **System Information** **Custom** **GL Impact** **E-Document**

LANDED COST PER LINE

Expenses 175.00 • Items 0.00

CATEGORY	ACCOUNT	AMOUNT	MEMO	DEPARTMENT	CLASS	LOCATION	CUSTOMER	PROJECT	BILLABLE	AMORT.	SCHEDULE
Lodging	63500 Travel & Ent	175.00		Marketing		Warehouse 1					

Transaction Number Examples

The following examples show the results of autogenerated transaction numbers based on prefixes used.

No Prefixes Used

If no transaction or location (or subsidiary) prefix is used, the transaction number is based on the initial number in the transaction type record. No prefixes are included

The following screenshot shows a transaction number that does not use prefixes.

ABC Cust Purchase Type Trans

TR Fictitious Company 1

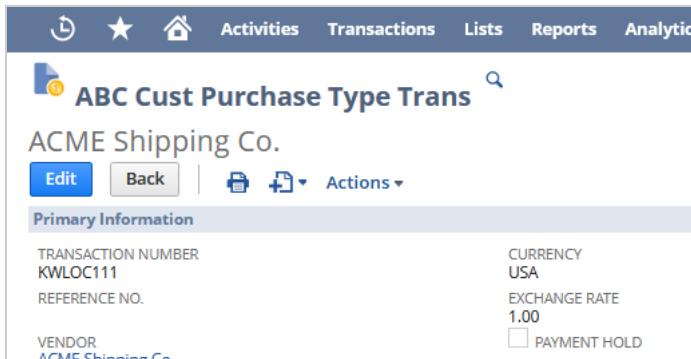
Primary Information

TRANSACTION NUMBER 30	CURRENCY USA
REFERENCE NO.	EXCHANGE RATE 1.00
VENDOR TR Fictitious Company 1	<input type="checkbox"/> PAYMENT HOLD
ACCOUNT	DUE DATE 12.2.2021

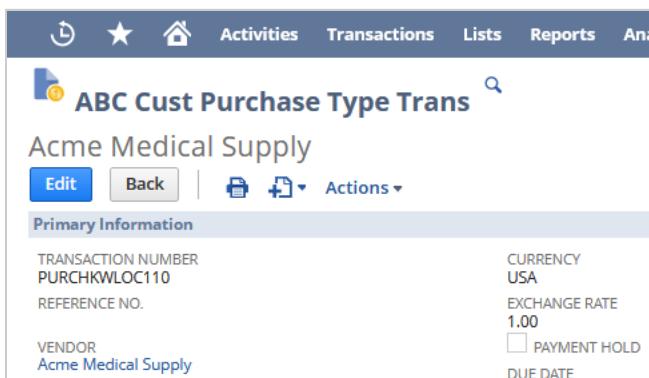
Location Prefix Used

If a location prefix is used (either alone or with a transaction type prefix) the transaction number is based on the initial number on the location record.

The following screenshot shows a transaction number that uses a location prefix only.

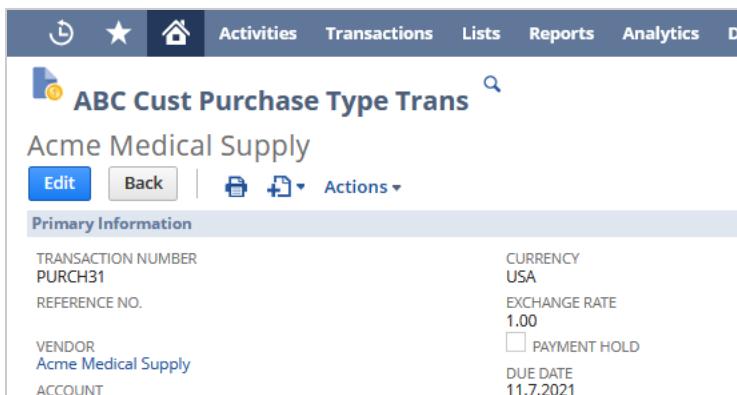


The following screenshot shows a transaction number that both a transaction prefix and a location prefix.



Transaction Type Prefix Used

If a transaction type prefix is used without a location (or subsidiary) prefix, the transaction number is based on the transaction number on the transaction type record.



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. Go 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 transaction 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:

- 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.

- Choose an option:
 - To let 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.

- 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**.
 - 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.
- 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 nonposting.
- 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 instances of the transaction post – but only if no statuses exist for the transaction type.

DESCRIPTION *	ID	POSTING
Awaiting Approval		Yes
Approved		Yes
Rejected		Yes

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.

DESCRIPTION *	ID	POSTING
Awaiting Approval		Yes
Approved		Yes
Rejected		No



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**.



Note: The **ID** field is populated with an A-Z value that is automatically generated when the transaction type is saved. You cannot enter a value in this column.

- 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 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 may have a situation where other users may need to create and edit transaction instances, but these users should not be permitted 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 may 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, select **Inline Text**.



Note: For more details about working with custom forms, see [Adding Custom Forms for a Custom Transaction Type](#).

- c. Click **Save** to create the form.
2. Configure the role to permit access to the custom form that you 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**.

Transaction • Other Record • Custom Record • CRM • Inventory Detail • Time • Item • Entity •							
ENABLED	RESTRICTED	TYPE ▾	FORM NAME	PREFERRED			
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Bad Debt	Custom Bad Debt Form	<input type="checkbox"/>			
<input type="checkbox"/>	<input type="checkbox"/>	Bad Debt	Standard Bad Debt Form	<input type="checkbox"/>			



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:

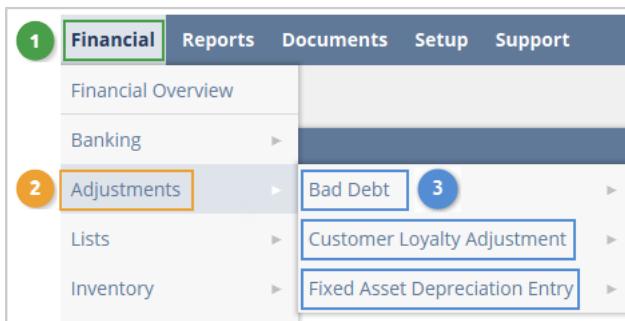
- [Planning for Custom Transaction Type Links](#)
- [Creating Links by Editing the Transaction Type](#)

- 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 about 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:

- Go to Customization > Lists, Records, & Fields > Transaction Types, and click the name of the appropriate transaction type.
- 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 select the appropriate label.
 - g. 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 process 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.

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 needed. 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.

Level	Users can:	Users cannot:
View	View instances of the transaction type.	Edit, create, or delete transaction instances.
Create	View and create instances of the transaction type.	Edit or delete transaction instances.
Edit	View, create, and edit instances of the transaction type.	Delete transaction instances.
Full	View, create, edit, and delete instances of the transaction type.	—

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.
- Note:** The Employee Center role is not available for transaction types that use a Sales or Purchase transaction style.
- 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 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 upper 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 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 upper 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.



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 needed.
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](#).

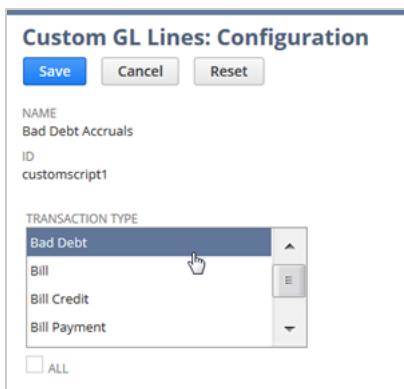
Custom Transaction Type Association with a Custom GL-Lines Plug-in Implementation

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 originate 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.



For more information, see the help topic [Configuring 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 various 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 originate from values entered in a custom UI created by using SuiteScript objects. For more details, see the SuiteTax documentation available to SuiteTax 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 (which uses the SuiteScript [N/record Module](#)):

```

1 | customTransactionTypeID = customTransaction.getValue('customtype')
2 | customTransactionType = record.load({
3 |   type: 'customtransactiontype',
4 |   id: customTransactionTypeID
5 | })
6 | customTransactionType.getValue('scriptid')
```

Deleting Custom Transaction Types

In some cases, it can 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. To display a popup menu, click **Actions**. Then click **Delete**.
The system displays a popup message 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:
 - [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](#).

To learn more about the available features, see the following subtab topics:

- [Items](#)
- [Promotions](#)
- [Shipping](#)
- [Billing](#)
- [SuiteTax](#)

Items

On the Items subtab of a new sales custom transaction instance, 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](#).

For more information about creating a new sales custom transaction instance, see [Creating Sales and Purchase Custom Transaction Instances](#).

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

i 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 and fields may 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.

i 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 \(Essentials\)](#).

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 noninventory 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 (Essentials)

Advanced Revenue Management (Essentials) automates revenue forecasting, recognition, reclassification, deferral, and auditing through a rule-based event handling framework. If Advanced Revenue Management (Essentials) 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 [Setup for Advanced Revenue Management \(Essentials\)](#).

Advanced Revenue Management (Revenue Allocation)

The Advanced Revenue Management (Revenue Allocation) is an add-on feature to Advanced Revenue Management (Essentials). This feature supports the use of fair value pricing, range checking, and fair value formulas to allocate revenue across several performance obligations.

Advanced Revenue Management (Revenue Allocation) supports fair values based on the method of your choice. You can use standalone selling price (SSP), vendor-specific objective evidence (VSOE), best

estimate of selling price (ESP), third party evidence (TPE), or some other method. These fair values are used to determine the revenue allocation ratios for multi-element transactions. For more information, see the help topic [Setup for Advanced Revenue Management \(Revenue Allocation\)](#).

Apply



Note: The Apply subtab is available only for sales credit transactions.

On a sales custom credit transaction, you can apply a transformed custom transaction to the source transaction. To apply a credit transaction to the source transaction, check the **Apply** box beside the debit transaction you want to apply the credit to.

The Apply subtab has the same fields as the Apply subtab for credit memo and displays the same set of transactions. For more information see the help topic [Applying a Customer Credit Memo](#).

Promotions

On the Promotions subtab of a new sales custom transaction instance, you can apply [SuitePromotions](#), shipping discounts, and more.

For more information about creating a new sales custom transaction instance, see [Creating Sales and Purchase Custom Transaction Instances](#).

SuitePromotions



Note: The Promotions subtab appears only if you have SuitePromotions enabled. For information about using SuitePromotions, see the help topic [Migrating to SuitePromotions](#).

On the SuitePromotions subtab, you can calculate apply 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 can 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 of a new sales custom transaction instance, you can calculate shipping costs.

For more information about creating a new sales custom transaction instance, see [Creating Sales and Purchase Custom Transaction Instances](#).

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.



Note: You must have View permissions for both Item Fulfillments and Sales Orders to retrieve shipping costs using the calculate the shipping costs calculator.

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 of a new sales custom transaction instance, you can apply [Gift Certificates](#) and [Installments](#), as well as perform other billing-related tasks like setting terms, adding billing addresses, and so on.

For more information about creating a new sales custom transaction instance, see [Creating Sales and Purchase Custom Transaction Instances](#).

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 on the Tax Details subtab of a new sales custom transaction instance.

For more information about creating a new sales custom transaction instance, see [Creating Sales and Purchase Custom Transaction Instances](#).

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 subtab topics:

- [Expenses and Items](#)
- [Landed Costs](#)
- [Billing](#)

- SuiteTax

Expenses and Items

On the Expenses and Items subtab of a new purchase custom transaction instance, 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](#).

For more information about creating a new purchase custom transaction instance, see [Creating Sales and Purchase Custom Transaction Instances](#).

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.

i Note: You can have multiple categories for custom cost categories. The following procedures use example categories of Shipping and Insurance. These categories can 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 [Expense Amortization](#).

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 noninventory 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 of a new purchase custom transaction instance, 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.

For more information about creating a new purchase custom transaction instance, see [Creating Sales and Purchase Custom Transaction Instances](#).



Note: You can have multiple categories for custom cost categories. The following procedures use example categories of Shipping and Insurance. These categories can differ from the custom categories your company uses. For more information, see the help topic [Creating Cost Categories](#).

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 the help topic [Landed Cost Overview](#).

Billing

On the Billing subtab of a new purchase custom transaction instance, you can create [Installments](#), as well as perform other billing-related tasks like setting terms and International Commercial Terms (incoterms), and so on.

For more information about creating a new purchase custom transaction instance, see [Creating Sales and Purchase Custom Transaction Instances](#).

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 on the Tax Details subtab of a new purchase custom transaction instance.

For more information about creating a new purchase custom transaction instance, see [Creating Sales and Purchase Custom Transaction Instances](#).

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 required.

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.



Note: Basic printing is not supported on custom transactions. For information about advanced printing templates, see [Advanced PDF/HTML Templates](#).

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 Customization > Forms > Advanced PDF/HTML Templates. The Record Type column shows the custom transaction type ID.

Advanced PDF/HTML Templates					
	NAME	SCRIPT ID	TYPE	RECORD TYPE ▾	PREFERRED
EDIT	Standard Cash Sale PDF/HTML Template	STD TMP LCAHSALE	Cash Sale		<input checked="" type="checkbox"/>
Customize	Standard Paycheck PDF/HTML Template	STD TMP LPAYCHECK	Check		<input checked="" type="checkbox"/>
Customize	Standard UK Voucher Check PDF/HTML Template	STD TMP LPU VOUCHERCHECK	Check		<input checked="" type="checkbox"/>
Customize	Standard US Voucher Check PDF/HTML Template	STD TMP LPU VOUCHERCHECK	Check		<input checked="" type="checkbox"/>
Customize	Standard UK Check PDF/HTML Template	STD TMP LPU CHECK	Check		<input checked="" type="checkbox"/>
Customize	Standard Check PDF/HTML Template	STD TMP LCHECK	Check		<input checked="" type="checkbox"/>
Customize	Standard AU Voucher Check PDF/HTML Template	STD TMP LAU VOUCHERCHECK	Check		<input checked="" type="checkbox"/>
Customize	Standard Credit Memo PDF/HTML Template	STD TMP LCUST CRED	Credit Memo		<input checked="" type="checkbox"/>
Customize	Standard Non-Typical Expenses PDF/HTML Template	STD TMP LCT NT EXP	Custom Transaction	CUSTOMTRANSACTION_NT_EXP	<input checked="" type="checkbox"/>
Customize	Standard Customer Deposit PDF/HTML Template	STD TMP LCUST DEP	Customer Deposit		<input checked="" type="checkbox"/>
Customize	Standard Quote PDF/HTML Template	STD TMP LQUOTE	Estimate		<input checked="" type="checkbox"/>
Customize	Standard Expense Report PDF/HTML Template	STD TMP LPLEXPREP	Expense Report		<input checked="" type="checkbox"/>
Customize	Standard Invoice PDF/HTML Template	STD TMP LCUST INV	Invoice		<input checked="" type="checkbox"/>
Customize	Standard Item Label PDF/HTML Template	STD TMP LITEM LABEL	Item Label		<input checked="" type="checkbox"/>

2. Click **Customize** beside the template that you want to customize.
3. Make the required changes to the template. For information, see [Advanced Templates Customization 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](#).

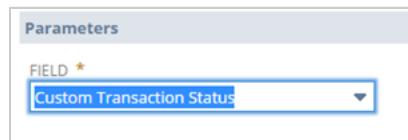
Custom Transaction Types in Workflows

If appropriate, you can reference your custom transaction types when you create workflows. For details, see the following sections:

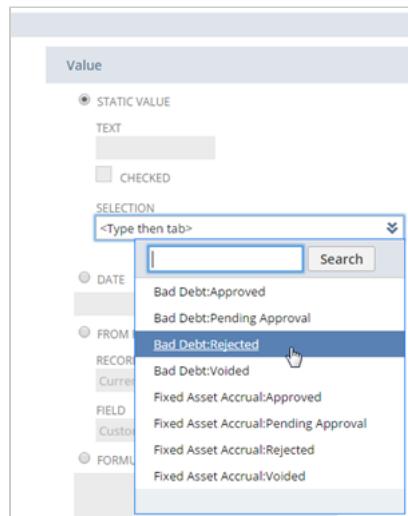
- [Referencing Status in a Workflow Action](#)
- [Using Status as a Workflow Condition](#)
- [Sales Custom Transaction Transform Action Workflow Example](#)
- [Using Workflows with SuiteScript for Custom Transaction Transformations](#)

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.

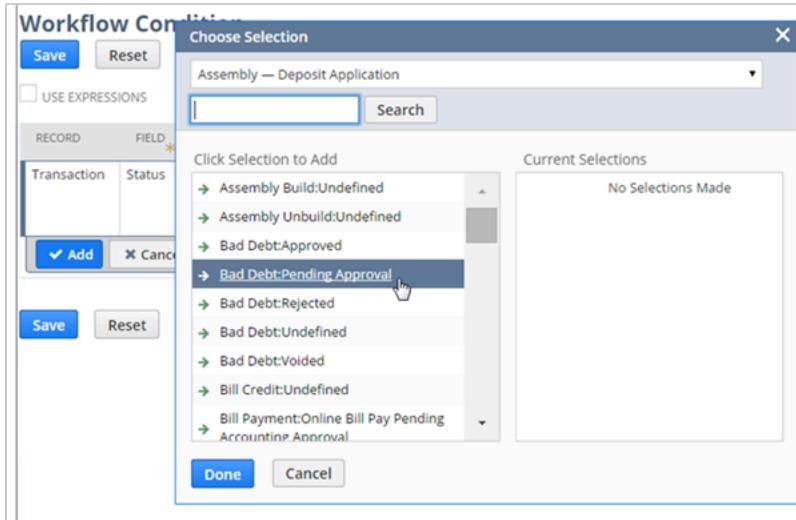


If you are using the Set Field Value action, you can choose the Static Value radio button and select the status from the Selection list. The list shows all of the available custom transaction statuses, listed according to custom transaction type. Select the appropriate combination of transaction type 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, so you can add custom buttons to custom transaction pages.

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 new instance of the custom transaction opens, with the Customer, Date, Period, Subsidiary, Class, Department, Location, and Items fields prefilled. The user verifies the information and clicks Save.

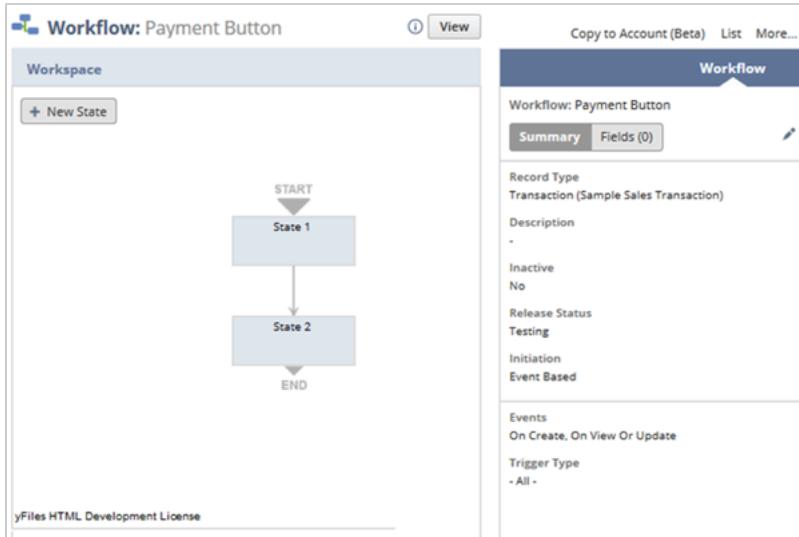
You can also add a custom button to the custom transaction page, for example, a Pro-forma Invoice. When the user clicks the button, a new instance of the standard transaction opens, with the Customer, Date, Period, Subsidiary, Class, Department, Location, and Items fields prefilled. The user verifies the information and clicks Save.

Payment links and generic transformation links are used to transform custom transactions. A Transformation type link is created only if a Payment type link does not already exist. For example, transforming a Custom Sales Transaction to a standard Customer Payment record creates a payment link, but not a transformation link.

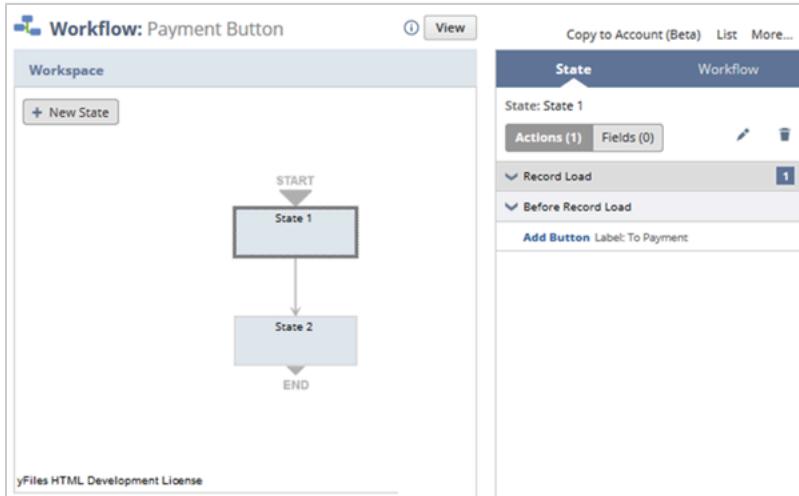
The Transformation type link creates a link between transformed transactions in custom transactions when you save the custom transaction. A transformation link is created whenever an instance of a custom transaction is the source or destination of a transformation. The link appears on target transaction in the Created From field and at the source transaction in Related Records.

Example:

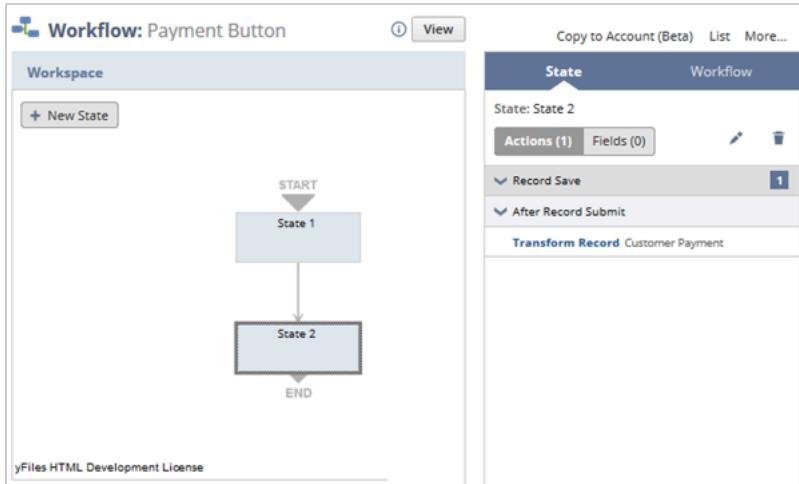
We want to add a button to a sales custom transaction type for payment. Our example triggers the workflow on create, view, or update of a Sample Sales Transaction record.



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. To permit the user to edit the transaction before saving, check the Redirect Without Saving box,



The following screenshot shows the button on the Sample Sales Transaction record.

Using Workflows with SuiteScript for Custom Transaction Transformations

To use SuiteScript 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.
- 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 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 include 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 access the NetSuite UI only by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic [SuiteScript 2.x Custom Pages](#).

Custom segments are supported in SuiteCloud Development Framework (SDF). SDF is a development framework that you can use to create SDF SuiteApps, or to customize NetSuite accounts, using an integrated development environment (IDE) on your local computer. SuiteCloud projects are file-based and use XML definitions of custom NetSuite objects. For more information, see the help topic [SDF Custom Object and File Development in SuiteCloud Projects](#).

For more details about custom segments, see the following topics:

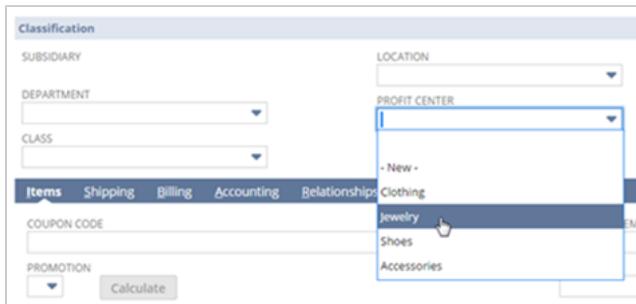
- [Benefits of Custom Segments](#)
- [Custom Segments Overview](#)
- [Permissions for Managing Custom Segments and Values](#)
- [Custom Segment Creation](#)
- [Filtering for a Custom Segment](#)
- [Custom Segment Values](#)
- [Using the Script ID to Access Custom Segment Body, Line, and Filter By Fields](#)
- [Custom Segments in Record Searches](#)
- [Customizing a Report by Using Custom Segments](#)
- [Using Custom Segments in Workflows](#)
- [Deleting a Custom Segment Definition](#)

For more information about working with classifications, departments and locations, and using the custom GL lines plug-in, see the following.

- [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.



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.

Sales by Audience and Profit Center View Detail				
PROFIT CENTER	QTY. SOLD	ITEM NAME	TOTAL REVENUE	SALES CHANNEL
Women's				
Shoes				
1 Ballet Flat		\$65.00	Web Store	Madrid
2 Jeweled Sandal		\$90.00	Brick-and-Mortar Stores	New York
1 Patent Heel		\$452.00	Partner Outlet	Paris
1 Slingback Sandal		\$115.00	Web Store	Sao Paulo
Total - Shoes	5	\$722.00		
Accessories	16	\$2,575.00		
Clothing	141	\$12,156.00		
Outerwear	46	\$5,519.00		
Jewelry	16	\$11,048.00		
Total - Women's	224	\$32,020.00		
Men's				
Outerwear	28	\$6,154.00		
Clothing	36	\$13,852.61		
Total - Men's	64	\$20,006.61		
Unisex				
Accessories	20	\$1,270.00		
Total - Unisex	20	\$1,270.00		
Total	308	\$53,296.61		

In these ways and more, custom segments enhance your ability to categorize data and meet your organization's unique reporting and analysis needs.

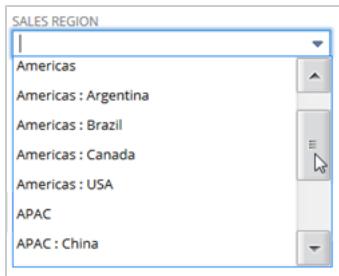
Before you can create custom segments, you must enable the feature, as described in [Enabling the Custom Segments Feature](#). After you have done so, you can create custom segments at Customization > Lists, Records, & Fields > Custom Segments > New. Almost all segment-configuration options can be managed from this page, including the following:

- Configure Segment Values to Be Hierarchical
- Configure a Segment to Default Static or Dynamically
- Filter a Segment's Values Based on Other 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
- Set Custom Segment Values
- 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 hierarchical relationships. For example, a segment called Sales Region can 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 select.



For more details, see [Creating Hierarchies Among a Custom Segment's Values](#).

Also, see [Benefits of Custom Segments](#).

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](#).

Also, see [Benefits of 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.

The screenshot shows the SiteBuilder Classification page. In the top right corner, under 'SALES REGION', the value 'EMEA: Denmark' is selected. Below it, under 'SALES CHANNEL', a dropdown menu is open, showing two options: 'Web Store' and 'Catalog'. The 'Web Store' option is highlighted with a blue border and has a cursor icon pointing at it. Other fields visible include 'SUBSIDIARY', 'LOCATION', 'DEPARTMENT', 'CLASS', 'ITEMS' (selected), 'COUPON CODE', 'PROMOTION', 'DISCOUNT ITEM', 'RATE', and a 'Calculate' button.

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](#).

Also, see [Benefits of Custom Segments](#).

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 may 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.

The screenshot shows the SiteBuilder Classification page. Under 'SUBSIDIARY', 'Parent Company' is listed. Under 'DEPARTMENT', 'Men's' is selected. Below it, under 'ITEMS' (selected), a dropdown menu for 'PROFIT CENTER' is open, showing three options: 'Clothing', 'Shoes', and 'Jewelry'. The 'Clothing' option is highlighted with a blue border and has a cursor icon pointing at it. Other fields visible include 'CLASS', 'COUPON CODE', 'PROMOTION', 'DISCOUNT ITEM', 'RATE', and a 'Calculate' button.

For details on filtering, see [Filtering for a Custom Segment](#).

Also, see [Benefits of Custom Segments](#).

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												Return to Invoice #3
ACCOUNT	AMOUNT (DEBIT)	AMOUNT (CREDIT)	POSTING	MEMO	NAME	SUBSIDIARY	DEPARTMENT	CLASS	PRODUCT LINE	PROFIT CENTER	SALES CHANNEL	SALES REGION
1100 Accounts Receivable	165.00		Yes		Anonymous Customer	Parent Company	Women's	Full-Priced	Winter 2015-16	Clothing	Web Store	Denmark
4000 Sales		165.00	Yes		Anonymous Customer	Parent Company	Women's	Full-Priced	Winter 2015-16	Clothing	Web Store	Denmark

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](#).

Also, see [Benefits of Custom Segments](#).

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.



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](#).

Also, see [Benefits of Custom Segments](#).

Display Segments Selectively

The Custom Segments feature includes a variety of permissions options. One option is that you can choose to make a segment visible to users only in certain roles. For example, you can create a segment called Rating that is applied to the Employee record type. This segment may 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](#).

For details on managing permissions for segments, see [Permissions for Managing Custom Segments and Values](#).

Also, see [Benefits of Custom Segments](#).

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.x 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](#).

Also, see [Benefits of Custom Segments](#).

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. Go 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)
- Customer payment
- 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 fulfillment
- Item fulfillment (transaction lines)
- Item receipt (expense transaction columns)
- Item receipt (transaction lines)
- Journal entry (transaction lines)
- Opportunity (transaction line)
- Other charge item (transaction line)
- Partner commission
- Payable Tegata
- Paycheck (derives custom segments from time entries)
- Paycheck earning
- Paycheck earning (transaction lines, derived from time entries)

- Paycheck journal
- Payment end journal
- Payroll adjustment
- Payroll liability check
- Period end journal
- Period end journal (transaction line)
- 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 payment
- 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
- Store with item groups

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 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 may configure a segment so that it is filtered by another segment. For example, you can 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. Therefore, if you edit a segment to create a filtering relationship between it and a second segment, you may receive a message regarding record application. This message may state that the system will apply the second segment, the one named in the Filtered by list, to more record types.

Additionally, you may 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.

Level	Users can:	Users cannot:
View	<ul style="list-style-type: none"> ■ View custom segment definitions. ■ View the values defined for existing custom segments. ■ Use the Custom Segment search type. 	<ul style="list-style-type: none"> ■ Create, edit, or delete custom segments. ■ Create, edit, or delete values on a custom segment (unless the role has the appropriate Value Management Access level for that segment).
Create	<ul style="list-style-type: none"> ■ View custom segment definitions. ■ View the values defined for existing custom segments. ■ Use the Custom Segment search type. ■ Create custom segments. ■ Create values for new segments that the user is creating. 	<ul style="list-style-type: none"> ■ Edit or delete custom segments. ■ Create, edit, or delete values on an existing custom segment (unless the role has the appropriate Value Management Access level for that segment).
Edit	<ul style="list-style-type: none"> ■ View custom segment definitions. ■ View the values defined for existing custom segments. ■ Use the Custom Segment search type. 	<ul style="list-style-type: none"> ■ Delete custom segments. ■ 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.

Level	Users can:	Users cannot:
Read	<ul style="list-style-type: none"> ▪ Create custom segments. ▪ Create values for new segments that the user is creating. ▪ Edit body fields and sublist fields on an existing segment. 	
Full	<ul style="list-style-type: none"> ▪ View custom segment definitions. ▪ View the values defined for existing custom segments. ▪ Use the Custom Segment search type. ▪ Create custom segments. ▪ Create values for new segments that the user is creating. ▪ Edit body fields and sublist fields on an existing segment. ▪ Delete custom segments. 	<ul style="list-style-type: none"> ▪ 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.

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 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 can 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 topics:

- [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.

Description	When Can the User Add Values?	Permitted Methods for Creating Values
Users assigned to the Administrator role	At any time	Creating Values Within the Segment Definition Creating Values by Clicking Manage Values Creating Values Using the New Button
Users with the Create level of the Custom Segments permission	Only during the time that the user is creating the segment (unless the user has additional privileges).	Creating Values Within the Segment Definition
Users with the following: <ul style="list-style-type: none"> ▪ The Create, Edit, or Full level of the Value Management Access permission 	At any time	Creating Values Within the Segment Definition Creating Values by Clicking Manage Values
Users with both of the following: <ul style="list-style-type: none"> ▪ 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.

Value Management Access Level	Users can:	Users cannot:
View	<ul style="list-style-type: none"> ■ 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.) 	<ul style="list-style-type: none"> ■ Create, edit, or delete the segment's values.
Create	<ul style="list-style-type: none"> ■ Create values for the segment for which they have the Value Management Access permission. 	<ul style="list-style-type: none"> ■ Edit or delete the segment's values.
Edit	<ul style="list-style-type: none"> ■ Create and edit the values for the segment. 	<ul style="list-style-type: none"> ■ Delete the segment's values.
Full	<ul style="list-style-type: none"> ■ Create, edit, and delete the values for the segment. 	—

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, select the required 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](#).
 - 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](#)

Custom segment types can be included in SuiteApps packaged with SuiteCloud Development Framework (SDF). SDF is a development framework that you can use to create SuiteApps from an integrated development environment (IDE) on your local computer. For more information, see the help topics [SuiteCloud Development Framework Overview](#) and [Custom Segments as XML Definitions](#).

Limitations of Multiple Select

If you select 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 can 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 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 [Custom Field Types](#).

To create a custom segment:

1. Go to Customization > Lists, Records, & Fields > Custom Segments > New.

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.

- The label cannot be a duplicate of an existing segment label.
 - The label cannot be a duplicate of an existing custom record 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. In the **ID** field, enter a unique alphanumeric ID for the transaction type. For information about best practices and naming conventions, see [Conventions for Naming Custom Objects](#). 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 can stop working, or may 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.

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 deactivate it by checking the **Inactive** box. For details, see [Inactivating a Custom Segment](#).
8. Select the **Display Type**. Display types let you specify how your custom segments 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 custom segments that are not editable or that have default information or custom code calculations.
 - **Normal**: A normal segment can be edited and can be used in defaulting and sourcing information.
 - **Disabled**: A disabled segment cannot be edited. You can use the custom segment with defaulting and sourcing information only. Any segment with a display type of disabled that does not have default or sourced information does not appear on forms.



Note: You cannot disable a required custom segment unless the segment has a default value.

- **Hidden:** A hidden segment cannot be seen on the record or transaction you apply it to. You can perform a search to display the value of the segment. The information in the segment is the result of defaulting information. You must set a default for the field.

If a custom segment is marked as hidden, it is not available to add to custom forms.

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



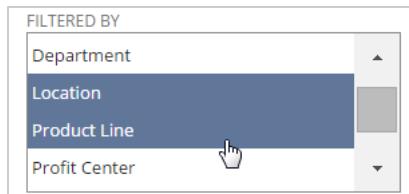
Warning: Hiding a custom segment is a display convenience only and is **not** field level security. Hidden custom segments are embedded in the page output and can be viewed in the page source.

9. Optionally, check the **Show In List** box to display the custom segment column on a custom record list or sublist.

CURRENCY	AMOUNT (FOREIGN CURRENCY)	AMOUNT	UNIT SIZE
USA	\$0.00	0.00	Medium
Canadian Dollar	\$0.00	0.00	Large
British pound	£0.00	0.00	Large

10. 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.



Later, you must manage the exact filtering configuration for each value by using the **Set Filters** button, as described later in this procedure.

11. 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](#).

12. 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.
13. 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.

14. 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 required 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 required). 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 Source List field on 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.

15. 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 wanted, 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 autogenerated COGS and revenue lines get the segment value from the item line or transaction body, where applicable, even if they belong to alternate 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 (Essentials) feature.



Important: The GL impact setting can be specified only 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 autogenerated 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:

1. Go to Customization > Lists, Records, & Fields > Custom Segments > New .
2. Check or clear the **GL Impact** box as appropriate.
3. Click **Save**.

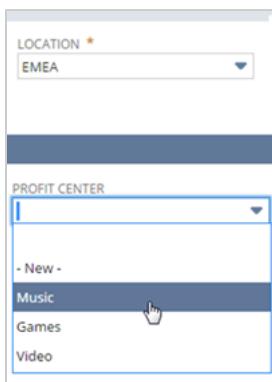
Filtering for a Custom Segment

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 can 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 may want to select another custom segment in the Filtered by list, but that segment is not listed. In these cases, the problem may be that the other segment already has a filtering relationship with the segment you are editing. To find out, go 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](#)



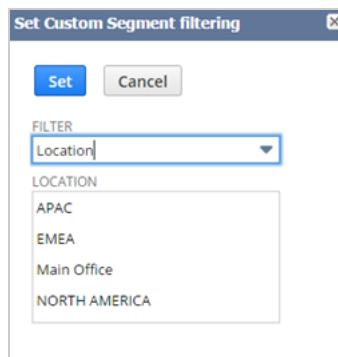
Important: Make sure you edit every value, if you want every value to be available under some circumstance. If you do not set the filtering conditions for a value, that value is never displayed in the segment's list.

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 preceding editing steps 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, child segments of only that parent are available for selection on transaction lines.

Note: You can have issues with filtering if you have the following scenario:

- A parent field exists on the expense line.
- You hide the parent field on the expense line using form customization.

Custom segment filtering on an expense line uses the parent field to filter. When you hide the parent field, the field still exists as a hidden field. Filtering from a parent body field will not work because the custom segment on the line continues to filter by the hidden parent line field. The custom segment does not filter by the parent body field.

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 various areas of the business, such as grocery, bakery, and cafe.

Value	Parent	Inactive	Filtering
Grocery	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Cafe	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Bakery	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

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.

The screenshot shows the 'Custom Segment' creation screen. In the 'Primary Information' section, the 'LABEL' is set to 'Product Line' and the 'TYPE' is 'List/Record'. A 'FILTERED BY' dropdown menu is open, showing 'Business Unit' selected, along with other options like 'Class', 'Department', and 'Location'. Below this, there are sections for 'Accounting' (GL IMPACT) and 'Details' (HELP and DESCRIPTION fields). At the bottom, there's a navigation bar with tabs for 'Values', 'Application & Sourcing', 'Validation & Defaulting', 'Permissions', 'Dependent Segments', and 'Display Order'. The 'Values' tab is active, displaying a list of items under 'Food': 'Hot Beverages', 'Cold Beverages', and 'Desserts'. To the right of the list, there are columns for 'PARENT', 'INACTIVE', and 'FILTERING', with specific values listed for each item.

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.

The screenshot shows a purchase order line item. The 'SUBSIDIARY' and 'CLASS' fields are empty. The 'DEPARTMENT' field has a dropdown arrow. The 'LOCATION' field is highlighted in blue and contains a dropdown menu with options: '- New -', 'Hot Beverages', 'Cold Beverages', and 'Desserts'. Below the location field, there are tabs for 'Items', 'Shipping', 'Billing', 'Accounting', 'Relationships', and 'Sales Team'. The 'BUSINESS UNIT' field has a dropdown arrow and contains the value 'Cafe'. At the bottom, there are buttons for 'Save', 'Cancel', 'Auto Fill', 'Reset', and 'Actions'.

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 edit 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
 - To apply the custom segment to Collect Tegata and Pay Tegata transactions, check the Tegata box
 - To apply the custom segment to single vendor payments and bill payments to multiple vendors, check the Vendor Payment box.
- Transaction columns, and custom transaction sublists

You can configure the kit or assembly custom segment column field to copy values from sales order items to fulfillment items. For more information, see [Apply to Kit or Assembly Components Setting for Custom Segments and Transaction Line Custom Fields](#).

- Entities
- 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 the help topic [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**.

Apply to Kit or Assembly Components Setting for Custom Segments and Transaction Line Custom Fields

You can configure kit or assembly custom segments or transaction line custom fields to copy values from sales order items to fulfillment items.

Custom Segments

You can configure the kit or assembly custom segment column field to copy values from sales order items to fulfillment items.

To configure the custom segment column field on items with kit or assembly components:

1. Go to Customization > Lists, Records, & Fields > Custom Segments.
2. Click the **Application and Sourcing** subtab. Then click the **Transaction Columns** subtab.
3. To have kit or assembly segments copy values from sales orders to related item fulfillments, check the **Kit/Assembly Components** box.

Transaction Line Custom Field

You can configure the kit or assembly transaction line custom field to copy values from sales order items to fulfillment items.

To configure the transaction line custom field on items with kit or assembly components:

1. Go to Customization > Lists, Records, & Fields > Transaction Column Fields.
2. Click the **Applies To** subtab.
3. To have kit or assembly transaction line fields copy values from sales orders to related item fulfillments, check the **Apply to Kit/Assembly Components** box.

Kit and Assembly Components Behavior

The following table describes the behavior of the Kit/Assembly Components box for custom segment columns and the Apply to Kit/Assembly Components box for transaction line custom fields.

Behavior	Kit/Assembly Component box or Apply to Kit/Assembly Components box Checked	Kit/Assembly Component box or Apply to Kit/Assembly Components box Cleared
Field is visible on item sublist	Yes*	Yes*
Field value is saved on item sublist	Yes	No
Field value is carried over from transformed record (sales order to item fulfillment)	Yes	No
Can override segment of line field when kit component items are visible	Yes (can override)	No (cannot override)

*Even if the Apply to Kit/Assembly Component box is used to hide the field, the kit or assembly field is available on the record through SuiteScript

*You can configure whether the field or segment is visible on the item fulfillment lines of the form. To make the field or segment visible, check the Item Fulfillment box on the Transaction Columns subtab of the Application & Sourcing subtab. You can override the Item Fulfillment setting of the custom field or custom segment on the Custom Form definition page where the fields and segments are listed.

Example of Overriding the Value of a Segment of Field on a Line with an Assembly Component

In the following example, for the Inventory and Child terms, you can override the original values in the Kitlinefield and the Kitsegment columns.

The screenshot shows the 'Item Fulfillment' screen in Oracle NetSuite. At the top, there's 'Primary Information' with fields like 'REF. NO.', 'CUSTOMER', and 'ITEM'. Below that is the 'Classification' section with dropdowns for 'SEGMENT 1' (set to '1'), 'SEGMENT A' (empty), and 'SEGMENT B (SOURCED FROM A)' (empty). Underneath is a navigation bar with tabs: 'Items', 'Shipping', 'Packages', 'Relationships', 'Communication', 'System Information', and 'Custom'. The main area is titled 'SELECT ITEM' and contains a grid of items. The first item listed is 'Kit Item' with a checked checkbox. To its right, under 'EXCLUDE ITEM FROM RATE REQUEST', the value 'One' is highlighted with a red box. In the 'ITEM' column of the grid, the values 'Inventory Item', 'Child', and 'Inventory Item' are listed. To the right of these, under 'KITLINEFIELD', the values '1', '3', and '2' are listed, also highlighted with red boxes. The grid has columns for 'FULFILL', 'ITEM', 'DESCRIPTION', 'LOCATION', 'ON HAND', 'REMAINING', 'QUANTITY', 'INVENTORY DETAIL', 'DROP SHIP PO', 'OPTIONS', 'REQUEST', 'KITLINEFIELD', and 'KITSEGMENT'.

Example of Kit and Assembly Values Being Copied to the Item Fulfillment

The following example shows kit and assembly custom segment columns or transaction line custom field values (S01 and S02) that have been copied from the sales order to the item fulfillment. The values are copied based on the configuration of the Kit/Assembly Components box on the custom segment definition page or the Apply to Kit/Assembly Components box on the transaction line custom field definition page.

This screenshot illustrates the relationship between the 'Item Fulfillment' screen and the 'Custom Segment' definition screen. On the left, the 'Item Fulfillment' screen shows a list of items: 'KitPackageU0' and 'KitPackageU0'. The 'KitPackageU0' row has a 'SO1' value in the 'SERIAL.NUM' column. On the right, the 'Custom Segment' definition screen shows a 'Transaction Line Field' configuration with 'APPLIES TO KIT/ASSEMBLY COMPONENTS' checked. It also shows a 'Custom Segment' record with 'NAME' set to 'SO1' and 'SO2'. Red arrows point from the 'SO1' value in the 'Item Fulfillment' grid to the 'NAME' field in the 'Custom Segment' definition screen, indicating the flow of values.

Dynamic Default Value Sourcing for Custom Segments

In some cases, you can 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 Default Values in Custom Segments Example](#)

Dynamic Defaulting for Custom Segments

In some cases, you may want to apply a custom segment to two record types that have a relationship. 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.



Call out	Description
1	This term represents the name of a field on the target record types. The selection that you make 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.
2	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.

Choose a Target

The Application & Sourcing subtab includes several child subtabs. Each of these subtabs represents a 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 go to certain other subtabs, you can select a alternate 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. To avoid this situation, make the segment required, 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, not all record types have all of the fields listed. If you select an option 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 an option 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 selected **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 is populated 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. Go 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.

APPLIED	RECORD TYPE
<input type="checkbox"/>	Work Order / Assembly Build
<input type="checkbox"/>	Customer Payment
<input type="checkbox"/>	Deposit
<input type="checkbox"/>	Expense Report
<input type="checkbox"/>	Inventory Adjustment
<input type="checkbox"/>	Item Fulfillment
<input type="checkbox"/>	Item Receipt
<input type="checkbox"/>	Journal Entry
<input type="checkbox"/>	Meal
<input type="checkbox"/>	Opportunity
<input type="checkbox"/>	Paycheck Journal
<input type="checkbox"/>	Transfer Order

NetSuite automatically checks the Customer box on the Entities subtab to make the custom segment available on the customer record.

The screenshot shows the SuiteBuilder interface for creating a custom segment. The top navigation bar includes 'Values', 'Application & Sourcing', 'Validation & Defaulting', 'Permissions', 'Dependent Segments', and 'Display Order'. The 'Application & Sourcing' tab is active. In the 'Sourcing' section, there is a dropdown menu labeled 'SOURCE LIST'. The 'Application' section contains a checkbox for 'WEB SITE REGISTRATION FORM'. Below this, there are two buttons: 'Mark All' and 'Unmark All'. A list of record types is shown, with 'Customer' checked. Other record types listed include Contact, Employee, Entity Group, Generic Resource, Other Name, Partner, Project, Project Template, and Vendor. At the bottom are 'Save', 'Cancel', and 'Reset' buttons.

If the preferred contact method is specified on the customer record, it defaults in automatically when you select the customer on a sales order.

The screenshot shows a sales order form with the 'Custom' tab selected. On the right side, there is a section for 'PREFERRED CONTACT METHOD' which has a dropdown menu set to 'Email'. This field is highlighted with a red box. Other fields in the same row include 'WEBSITE' and 'CUSTOMER CONTACTS'. The form also includes sections for 'NOTES FROM OPPORTUNITY', 'DOES THIS FIELD DISPLAY ON CHECKOUT?', 'SHIPINFO', 'REQUESTED SHIP DATE', and a 'Batch Transaction Information' section with a 'Clear All Lines' button and a 'Batch' table with various actions like 'Add', 'Cancel', 'Insert', etc. At the bottom are 'Save', 'Cancel', 'Auto Fill', 'Reset', and 'Actions' buttons.

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.

Dynamic Default Values in Custom Segments Example

You can use sourcing to create a dynamic default on a filtered segment. The steps in this example outline how you can set up a state to filter by city on a sales order. Then when you enter the city name on a sales order, the state fills in automatically.

To complete the steps in this example, you'll need to create custom segments. For more information, see [Creating a Custom Segment](#).

Complete the following steps

1. Creating a Custom Segment Type that Determines the Dynamic Default Value.
2. Creating a Custom Segment Type that Displays the Dynamic Default Value.
3. Setting Dynamic Default Values.
4. Using the Segments on a Transaction Form.

Creating a Custom Segment Type that Determines the Dynamic Default Value

Create the custom segment whose value is used to determine the dynamic default value of another field. In this example, the custom segment is for the name of the city and is used to determine the default state name.

To create the segment:

1. Go to Customization > Lists, Records, & Fields > Custom Segments > New.
2. In the **Label** field, enter the name of the segment, for example, **City**.
3. Complete the fields, as required.
4. Select the **Application & Sourcing** subtab. Then select the **Transactions** subtab.
5. Under the **Application** subheading, check the box beside the transactions you want to apply this segment to. For this example, check the **Sales Transactions** box.
6. Click **Save**.

Creating a Custom Segment Type that Displays the Dynamic Default Value

Create the custom segment whose value dynamically displays in the field based on the value entered in the custom segment field created in the preceding procedure. In this example, the custom segment is for the name of the state, and it will automatically appear based on the city name entered.

To create the segment:

1. Go to Customization > Lists, Records, & Fields > Custom Segments > New.
2. In the **Label** field, enter the name of the segment, for example, **State**.
3. Complete the fields, as required.
4. Select the **Application & Sourcing** subtab. Then select the **Transactions** subtab.
5. In the Application field group, check the box beside the transactions you want to apply this segment to. For this example, check the **Sales Transactions** box.
6. In the **Source List** field, select the label of the first custom segment created. For this example, select **City**.
7. A message appears informing you that setting the source list (custom segment) requires the system to apply the segment to the record type. To proceed, click **OK**.
8. Click **Save**.

Setting Dynamic Default Values

After the segments are created, you must set the default values for custom segment values. For example, if you want a value for state to dynamically display when a specific city is entered in the City field, you set this up on the City entry form.

To set the dynamic default values for a segment:

1. Go to Customization > Lists, Records, & Fields > Record Types.
2. Locate the custom record that is associated with the City segment that you created and click **List**.
3. To create a new record, click **New**. To edit an existing record, click **Edit**.
4. If this is a new record, in the **Name** field, enter the name of the city.
5. In the **State** field, enter the state. When the city name specified is entered on a sales order, the state name set up here appears automatically in the **State** field.
6. Click **Save**.

Using the Segments on a Transaction Form

You can now use the segments you created to fill in values dynamically on sales forms. In this example, when you create a new sales order, the newly-created segments appear on the Custom subtab of the sales order. When you enter a city name, the state automatically fills in with the state that you specified in the dynamic defaulting setup in the preceding procedure.

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 an alternate default. For details, see [Configuring Static Defaults for Custom Segments](#).
- You can configure logic that dynamically generates default values. You can specify varying logic for various 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 required. 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 required 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 may have permission to create and edit a specific record type, but they may 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 select 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. Therefore, 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 permit or deny 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 permit or deny a role from being able to use the segment as a search field or column. You can also permit or deny 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.

Level	Users can:	Users cannot:
None	—	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.
View	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.	Set values for the segment.
Edit	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).	—

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, select the required 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 Permissoin](#)

Scope of the Search/Reporting Access Permission

The following table describes the access associated with the various levels of the Search/Reporting Access permission.

Level	Users can:	Users cannot:
None	—	<ul style="list-style-type: none"> ■ Use the segment as a search filter. ■ Use the segment as a column in search results. ■ 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. ■ View reports that use the segment as a filter or include it as a column. ■ Customize reports to use custom segments as filters or columns.
View	<ul style="list-style-type: none"> ■ View saved searches that include custom segments in their results. ■ View reports that use the segment as a filter or include it as a column. 	<ul style="list-style-type: none"> ■ Use the segment as a search filter ■ Use the segment as a column in search results. ■ Customize reports to use custom segments as filters or columns.
Edit	<ul style="list-style-type: none"> ■ View saved searches that include custom segments in their results. ■ View reports that use the segment as a filter or include it as a column. ■ Use the segment as a search filter. ■ Use the segment as a column in search results. ■ Customize reports to use custom segments as filters or columns. 	—

Assigning the Search/Reporting Access Permissoin

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, select the required 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 may 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.

Display Order	
Move To Top	Move To Bottom
LABEL	
:: Sales Region	
:: Geographic Region	

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.

Custom Segments			
VIEW		Custom Segment	New Custom Segment
FILTERS		TOTAL: 2	
<input type="button" value="Edit View"/>	<input type="button" value="Edit View"/>	<input type="button" value="DISPLAY ORDER ▲"/>	LABEL ID GL IMPACT
1	Sales Region cseg_sales_region Yes	2	Geographic Region cseg_geo_region Yes

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 may 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 dimmed, click **Manage Values** to open a separate 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.

Group	Users can:	Users cannot:
Users assigned to the Administrator role	<p>Change any value on the segment definition, except for ID, which can never be changed.</p> <p>Users in this group are the only ones who can change the following value:</p> <ul style="list-style-type: none"> ■ 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.) 	Change the ID field.
Users with both of the following: <ul style="list-style-type: none"> ■ The Create, Edit, or Full level of Value Management Access permission ■ The Edit or Full level of the Custom Segments permission 	<ul style="list-style-type: none"> ■ Change most values on the segment definition. ■ Work with values by directly editing the Values sublist. Users can also click the Manage Values button to complete these tasks on another page, if they prefer. (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.) 	Change any of the following fields: <ul style="list-style-type: none"> ■ ID. ■ Type, if the segment's GL Impact box has been checked.
Users with either the Edit or Full level of the Custom Segments permission	Change most values on the segment definition.	<ul style="list-style-type: none"> ■ Change any of the following fields: <ul style="list-style-type: none"> □ ID. □ Type, if the segment's GL Impact box has been checked. ■ Add, create, or delete values. ((However, these users can give themselves that permission by making changes on the Permissions subtab of the segment definition.)
Users with both of the following: <ul style="list-style-type: none"> ■ The Create, Edit, or Full level of Value Management Access permission ■ The View level of the Custom Segments permission 	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 .)	Edit the segment definition.

Inactivating a Custom Segment

You can deactivate a custom segment by checking the **Inactive** box on the custom segment definition page. When you make a custom segment inactive, the segment no longer appears on any forms, reports, searches, or GL Impact pages, and is not available for SuiteBuilder, 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.

When you deactivate a parent custom segment but leave its child segment active, the child segment is no longer filtered by the parent.

You cannot deactivate a custom segment that is used as a search filter or in a workflow. If you try to do so, you will receive an error message indicating why the custom segment cannot be deactivated.

You cannot deactivate the custom field if it is used for criteria in duplicate detection. If you try to do so, you will receive an error message. For more information about duplicate detection, see the help topic [Setting Up Duplicate Detection](#).

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.

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 deactivate 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.

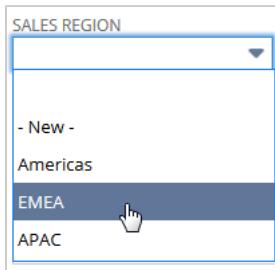
You cannot delete the custom segment if it is used for criteria in duplicate detection. If you try to do so, you will receive an error message. For more information about duplicate detection, see the help topic [Setting Up Duplicate Detection](#).

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. In the **Action** menu, click **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**.



4. 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.
5. 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 stand-alone 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.
6. 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. Select the appropriate values, and then click **Set**. Click **OK**. For more information about filtering, see [Filtering for a Custom Segment](#).
7. If you have more values to add, repeat Step 3.
8. Choose **Sublist** or **Alphabetical** to specify the display order of the values.
9. Click **Save** on the segment definition.

Creating Values by Clicking Manage Values

In some cases, you may 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 get 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 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 stand-alone 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.

- 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 individual 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 notice 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:

- 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.



- Click **New**.
- The system displays a popup form that lets you create a new value.
- 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.



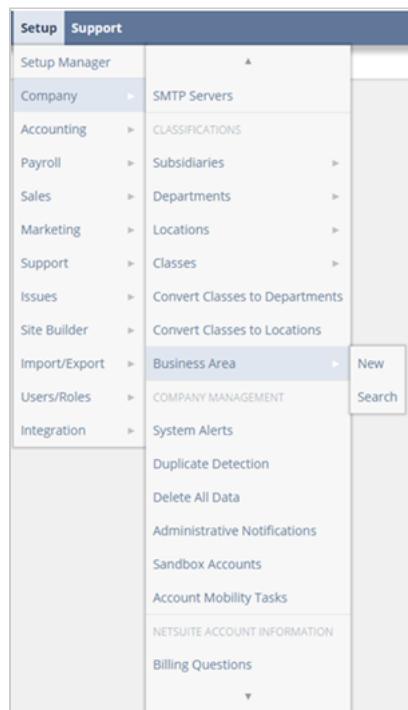
For more information about filtering, see [Filtering for a Custom Segment](#).

- 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 stand-alone 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**.

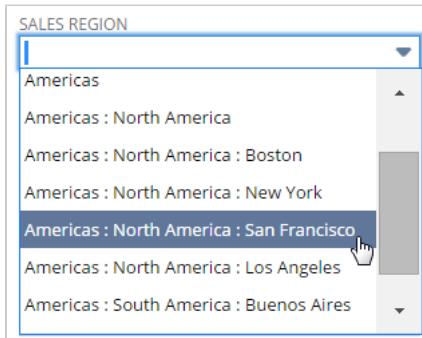
If the display order for the custom segment values is set to **Sublist**, the new value appears at the end of the list of values. If the display order for the custom segment values is set to **Alphabetical**, the new value is incorporated into the alphabetical list of values.

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 may want to make the value inactive. Making a value inactive prevents the value from appearing as a stand-alone 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.

ID	VALUE*	PARENT	INACTIVE
1	Americas		Yes
2	North America	Americas	<input checked="" type="checkbox"/>

Buttons at the bottom: OK, Cancel, Insert, Remove, Move Up, Move Down, Move To Top, Move To Bottom, Set Filters.

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 required 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.

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.

Description	Permitted Methods for Editing Values
Users assigned to the Administrator role	Editing a Custom Segment's Values by Updating the Segment Definition Editing a Custom Segment's Values by Clicking Manage Values
Users with both of the following: <ul style="list-style-type: none"> ■ The Edit or Full level of the Value Management Access permission ■ The Edit or Full level of the Custom Segments permission 	Editing a Custom Segment's Values by Updating the Segment Definition Editing a Custom Segment's Values by Clicking Manage Values
Users with both of the following: <ul style="list-style-type: none"> ■ The Edit or Full level of the Value Management Access permission ■ The View level of the Custom Segments permission 	Editing a Custom Segment's Values by Clicking Manage Values



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. On the segment definition, click **Save**.

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 Ctrl key to select more than one value or to clear a value. For more information about filtering, see [Filtering for a Custom Segment](#).

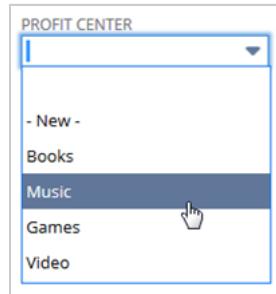


- If you want to change whether this value is visible to users, check or clear the **Inactive** box. Note that making the value inactive prevents the value from appearing as a stand-alone choice. However, if the value is a parent to another value, the value still appears as a parent.

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 in alphabetical order, 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, select **Sublist**.
 - To make the values appear in alphabetical order, select **Alphabetical**.

ID	VALUE*
1	Books
2	Music
3	Games
4	Video

4. Click **Save**.

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. You can always reactivate it, if needed.

Inactivating a segment's values has the following results:

- The value no longer appears as a choice in the segments list on new forms or on existing forms where it wasn't already selected. However, if the value is a parent to another value, it still appears as a parent.
- If an inactive value was previously selected on an existing form, the value continues to appear on that form. A search on that value will also turn up in search results.
- When filtering, if one segment is dependent on another segment, values filtered by an inactive value will not be available. This is true even if the inactive value on which other values depend is already selected.

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 may 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 get 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 may 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 get 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.

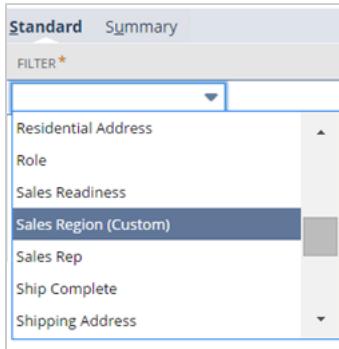
Custom segment IDs in searches and filters show the field type so that you can distinguish among body, line and filter fields.

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:

```
1 | CONCAT({cseg_section},{cseg_section.cseg_section_filterby_subsidy})
```

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 in alphabetical order with other available fields.

The screenshot shows the Report Builder interface with the following components:

- SEARCH FIELDS:** A search bar with a magnifying glass icon.
- ADD FIELDS:** A list of fields categorized by record type:
 - Sales:**
 - Account Based Number
 - Amount (Discount)
 - Amount (Foreign Currency)
 - Amount (Gross)
 - Amount (Net)
 - Audience** (highlighted in blue)
- Report Preview:** A table showing data for the 'Item' record type. The table has three columns: 'Item', 'Audience', and 'Sales Channel'. The 'Audience' column contains five rows labeled 'Item Type 1' through 'Item 5', each corresponding to a row in the 'Audience' list above.

Item	Audience	Sales Channel
Item Type 1	Audience 0	Sales Channel 0
Item 2	Audience 1	Sales Channel 1
Item 3	Audience 2	Sales Channel 2
Item 4	Audience 3	Sales Channel 3
Item 5		

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.

 **Note:** Custom segments are not included as dimensions in the Budget and Financial fields of the Financial Report Builder for budget-related reports. The budget-related reports are Budget Income Statement, Budget Income Statement Detail, and Budget vs. Actual.

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](#).

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](#)

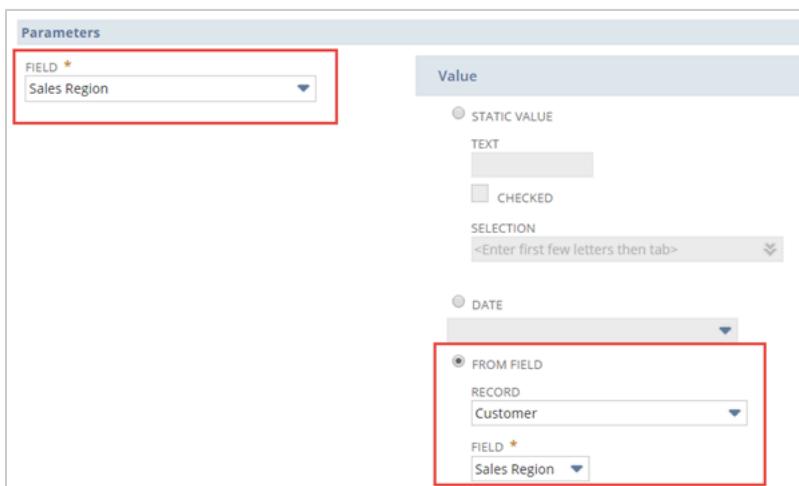
Setting Field Values

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.

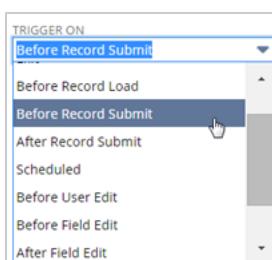
For example, with a workflow, you can set a distinct sourcing method for each specific record type. By contrast, in some cases, the custom segment's Source List field lets you choose only one defaulting method, and that method is shared by multiple record types. For example, you can select only one Source List value for all transaction types. However, with a workflow, you could use the Set Field Value action to create a unique defaulting method for each specific transaction type.

For example, suppose you were editing a workflow that affects the Cash Sale transaction type. For 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 may 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.

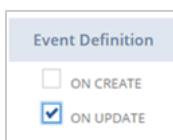
To configure this behavior, use the fields under the Parameters heading in the Workflow Action window. You identify the segment for which you want to create the default by using the Field list on the left side of the page, then use the fields under the Value heading to configure the sourcing behavior. 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 NetSuite populates the source field automatically, or when the user is editing the record. With a workflow, you could configure the value to be populated at various 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.



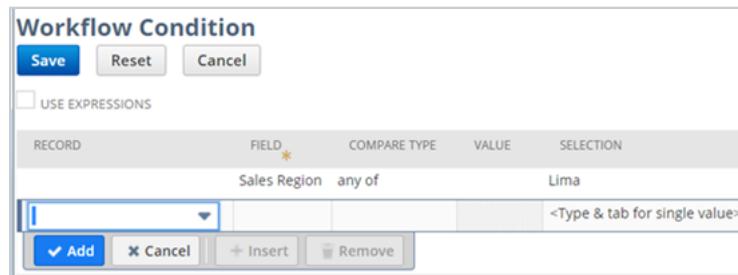
Similarly, with a workflow, you could choose to have the segment value be set only when the record is updated, rather than when it is created. Conversely, you could choose to have the segment value be set only when the record is created, rather than when it is updated. You do this by editing the workflow 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 if the segment is set to a certain value only. 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.



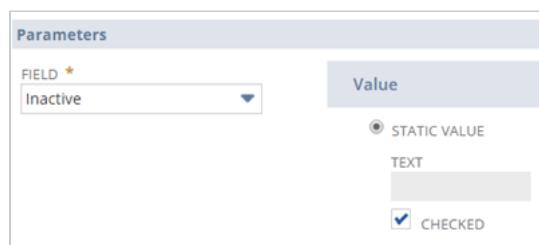
For more information on workflow conditions, see the help topic [Workflow Conditions](#).

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 can 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 may 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, select 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 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:

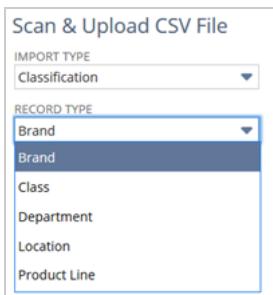
- Enable the Custom Segments feature at Setup > Company > Enable Features, on the SuiteCloud subtab.
- Create at least one custom segment in your account. You can create custom segments at Customization > Lists, Records, & Fields > Custom Segments > New.

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 can 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

Administrators and users with the Custom Centers permissions can create custom centers and apply the custom centers 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 not see the same links on the Forecast tab as 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 where you want the tab to appear. 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 customize centers and tabs only 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 access the NetSuite UI only by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic [SuiteScript 2.x Custom Pages](#).

Custom Centers are supported in SuiteCloud Development Framework (SDF). SDF is a development framework that you can use to create SDF SuiteApps, or to customize NetSuite accounts, using an integrated development environment (IDE) on your local computer. SuiteCloud projects are file-based and use XML definitions of custom NetSuite objects. For more information, see the help topic [SDF Custom Object and File Development in SuiteCloud Projects](#)

Account-Specific Domains in Custom Center Links

You should use URLs that include account-specific domains in all custom center links.

In the past, NetSuite domains contained an identifier of the data center where your account was hosted. Links in custom centers used these data center-specific domains. NetSuite domains no longer include data center identifiers. Instead, the domain identifies your account, not to the data center where your account is hosted. These are called account-specific domains.

When you create custom center links, if you attempt to add a URL that includes a data center-specific domain, a warning message appears. The message suggests an account-specific domain URL to use instead. You should make the suggested change. If you do not make the change, the URL as you entered it is saved.

Currently, a data center-specific domain in the URL will automatically be modified to an account-specific domain in the NetSuite center. The automatic modification from a data center-specific domain to your

account-specific domain is applied to your active session. The configuration of the URL does not change. Links that use account-specific domains are faster than links modified in the current session. You should update all links in custom centers to use your account-specific domain.

See the following topics.

- [Creating and Editing Custom Centers](#)
- [Creating Center Tabs](#)
- [Creating Center Categories](#)
- [Creating Center Links](#)
- [Assigning a Custom Center to a Custom Role](#)

Creating and Editing Custom Centers

You can create custom centers that you apply to custom roles. You can then edit the center, for example, to create tabs and links for the custom center, define the order of tabs, and translate labels.

You can customize centers only if the Custom Records feature is enabled.

See the following sections:

- [Creating a Custom Center](#)
- [Entering Translation Labels for a Custom Center](#)
- [Changing the Order of Tabs in a Custom Center](#)
- [Managing Centers with SuiteCloud Development Framework \(SDF\)](#)

Creating a Custom Center

To create a custom center:

1. Go to Customization > Centers and Tabs > Centers > **New**.
2. In the **Label** field, enter a name for the center.
3. In the **ID** field, enter an internal ID for the center, if required.
4. Click **Save**.

Entering Translation Labels for a Custom Center

After you create and save a custom center, you can edit the center to define translated labels.

To translate labels in a custom center:

1. On the **Centers** list page, click **Edit** next to the center for which you want to define translations.
2. On the **Translation** subtab, enter the translated labels.
3. Click **Save**.

For more information about translating languages, see the help topic [Configuring Multiple Languages](#).

After you create a custom center, you can create the tabs and links to appear in the center. You can create a custom tab from the following locations:

- [Creating Center Tabs](#)
- [Creating Center Categories](#)

Changing the Order of Tabs in a Custom Center

From the Centers page, you can change the order of the tabs in a custom center. The definition page for the center lists the tabs in the order in which they are displayed. To change the order, select a tab listing and drag it to another place in the order. Or select a tab listing and click **Move to Top** or **Move to Bottom**.

Managing Centers with SuiteCloud Development Framework (SDF)

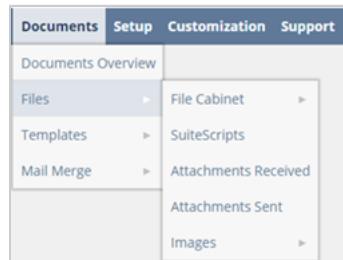
You can use SuiteCloud Development Framework (SDF) to manage custom centers as part of file-based customization projects. For information about SDF, see the help topic [SuiteCloud Development Framework Overview](#). You can use the Copy to Account feature to copy an individual custom center to another 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](#).

Creating Center Tabs

Administrators and users with the Custom Center Tabs permission can create 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 customize center tabs only if the Custom Records feature is enabled.

In the following screenshot, the tabs are Documents, Setup, Customization, and Support. Off the Documents tab, the categories are Files, Templates, and Mail Merge. Off the Files category, the links to the File Cabinet are the SuiteScripts, Attachments Received, and so on.



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.

The screenshot shows the 'Custom Center Tab' creation dialog. At the top, there are 'Save' and 'Cancel' buttons. Below them are fields for 'LABEL' and 'CENTER'. A tabs bar at the bottom includes 'Content', 'Audience', and 'Translation'. The 'Content' tab is active, showing a 'Categories' section with a 'NAME' field and a 'NAME*' label. Below this is a toolbar with buttons for 'Add', 'Cancel', 'Insert', 'Remove', 'Move Up', 'Move Down', 'Move To Top', and 'Move To Bottom'. The 'Categories' section is currently empty.

2. In the **Label** field, enter a name for the tab.
Users click the name to view the contents of the tab.
3. From the **Center** list, select the center for which you want to add the tab.
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. Click the **Content** subtab. Then, on the **Categories** subtab, in the **Name** column, enter a name for a category of links. After creating the categories, you will later follow the steps in [Creating Center Links](#) to assign specific links to each category.



Warning: This step 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 the category links that should appear on the tab.
7. Click the **Portlets** subtab.
8. In the **Type** column, add **Links**.
9. In the **Column** column, enter left, right, or center to define the location where the portlet appears on the tab.
10. To have the link show, check the **Show** box.
11. Click **Add**.
12. Add all the required portlets to the tab.
13. Click the **Audience** subtab.

The screenshot shows the 'Custom Center Tab' configuration interface. The 'Audience' subtab is active. It contains several dropdown menus and checkboxes for selecting users and groups. The 'ROLES' section has a checked 'Select All' checkbox and lists 'AVP Clerk', 'AV/Clerk', 'Accountant', 'Accountant (Reviewer)', and 'Administrator'. The 'EMPLOYEES' section also has a checked 'Select All' checkbox and lists '5. Sabina test', 'Allan Stours ABC Medical Supplies : A', 'Adams, William', and 'Adeline Tenney'. The 'GROUPS' section lists '2005_springsale_emailgroup', 'All Employees', 'balance test group', and 'healthcenter group'. The 'CUSTOMERS' section is empty. The 'VENDORS' section lists '1099 test', 'Acme Medical Supply', 'ACME Shipping Co.', and 'Adjustment Partner'. The 'PARTNERS' section lists 'Abe Simpson', 'Abe Simpson', 'Adjustment Partner', 'Commission Partner', and 'Business Test'. There are also sections for 'DEPARTMENTS' and 'Accounting'.

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
15. On the **Translation** subtab, enter translated labels for the custom center tab. The Translation subtab is available only if the Multi-Languages feature is enabled in your account. For more information, see the help topic [Configuring Multiple Languages](#).
16. Click **Save**.
17. Next, add links to the categories. See [Creating Center Links](#) for details.

For information about changing the order of tabs, see [Changing the Order of Tabs in a Custom Center](#).



Note: If you return to the tab and do not see the changes you have made, clear your browser cache.

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 of your accounts. Each center tab 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](#).

You can also use the Dashboard Tiles SuiteApp to create a dashboard with tiles that display your business critical KPIs in a bold visual layout with images and blinking alerts. For more information about the Dashboard Tiles SuiteApp, see the help topic [Dashboard Tiles](#).

Creating Center Categories

Administrators and users with the Custom Center Categories permission can create 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 customize center categories only 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 the 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**.

LABEL *
Contact Details

ID
_contact_detail

CENTER TYPE *
CC Sales Representative Center

CENTER TAB *
New Leads

INSERT BEFORE

Values **Translation**

LINK *	LABEL	TRANSLATION	SHORT LIST
:: New Corporate Leads	New Corporate Lead		
:: Customer Profile	Customer Info		
:: Referrer	Referral		

<Type then tab>

Add Cancel Insert Remove Move Up Move Down Move To Top Move To Bottom

2. On the Center Category page, complete the following steps:
 - a. In the **Label** field, enter a name for the category.
 - b. In the **ID** field, enter an ID for the category, if required.
 - c. From the **Center Type** list, select an existing center.
 - d. From the **Center Tab** list, select one of the standard, built-in NetSuite tabs (also referred to as sections).
 - e. From the **Insert Before** list, select where you want to insert the custom category.

3. On the **Values** subtab, in the **Link** list, select 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 appear by default.

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, the link appears in the **Link** list.

4. In the **Label** field, enter a UI label for the link.
5. The short list setting is applicable to some center types. 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.
6. Click **Add**.
7. Click **Save**.
8. On the **Translation** subtab, enter translated labels to use for the category. The Translation subtab is available only if the Multi-Languages feature is enabled in your account. For more information, see the help topic [Configuring Multiple Languages](#).



Note: If you return to the tab and do not see the changes you have made, clear your browser cache.

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

Administrators and users with the Creating Center Links permission can create 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 customize center links only if the Custom Records feature is enabled.

You can create links to NetSuite pages and web pages. For more information, see [Creating Links to NetSuite Pages](#) and [Creating Links to Web Pages](#).

Creating Links to NetSuite Pages

You can create center links to NetSuite pages.

To create links to NetSuite pages, custom records, and Suitelets:

1. Go to Customization > Centers and Tabs > Center Categories.
2. On the **Center Categories** list page, click **Edit** next to the category for which you want to add links.

LINK *	LABEL	TRANSLATION	SHORT LIST
::: New Corporate Leads	New Corporate Lead		
::: Customer Profile	Customer Info		
::: Referrer	Referral		

3. On the **Values** subtab, from the **Link** list, select the NetSuite page, custom record, or Suitelet to link to. If you are adding a link to a NetSuite page, make sure you 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. To have a link appear on a standard category, you must set the link on the Custom Record Type definition page. Also, 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. Note that a link to a custom record type does not display for users who do not have permission to access that custom record type.

4. In the **Label** field, enter a UI label for the link. The label appears off to the side of the category in the UI.
5. The short list setting is applicable to some center types. In the **Short List** column, you can indicate that the link should appear in a portlet when the link is in a narrow column of the dashboard. To do so, check the box.

Note: If the link is on a custom tab, the short list setting does not control whether the link is included on the links portlet. For custom tabs the link is always included.

- Click **Add**.

Note: The link will open in the existing window. You cannot set up a link to open in a new window.

- On the **Translation** subtab, enter translated labels for the custom link. The Translation subtab is available only if the Multi-Languages feature is enabled in your account. For more information, see the help topic [Configuring Multiple Languages](#).
- For centers that support it, you can indicate that the link should appear in a portlet when the link is in a narrow column of the dashboard. To do so, check the **Short List** column box. Note that for custom tabs, the link always displays on the links portlet regardless of the Short List setting.
- Add other values required. To change the order, select an item in the list and drag it to another place in the order. Or select a item and click **Move Up**, **Move Down**, **Move to Top**, or **Move to Bottom**.
- Click **Save**.

Creating Links to Web Pages

You can create center links to web pages.

To create links to web pages:

- Go to the website or NetSuite page for which you want to create a link. Right-click the address bar and select **Copy**.
- Go to Customization > Centers and Tabs > Center Links > New.

Label *	URL *	Script ID
New User Interface	https://system.netsuite.com/help/helpcenter/en_US/Output/Help/SetupConfiguration/UserExperience/EUpdates_Version_2010_Release_2_User_Interface_Overview.html?NS_VER=2010.2.0	
View Transaction Saved Searches	https://system.netsuite.com/app/common/search/savedsearches.nl?sortcol=title&sortdir=ASC&csv=HTML&OfficeXML=F&pdf=F&use=GENERAL&type=Transaction&accesslevel=ALL&scheduled=ALL	
View Saved Reports	https://system.netsuite.com/core/reporting/savedreports.nl?whence=	
Mass Email Contacts	https://system.netsuite.com/app/crm/common/merge/mailmerge.nl?templatetype=EMAIL	
View Contact Saved Searches	https://system.netsuite.com/app/common/search/savedsearches.nl?sortcol=title&sortdir=ASC&csv=HTML&OfficeXML=F&pdf=F&use=GENERAL&type=Contact&accesslevel=ALL&scheduled=ALL	
Create Target Group for Campaign	https://system.netsuite.com/app/crm/crmgroup.nl	
Form Layout Enhancements	https://system.netsuite.com/help/helpcenter/en_US/Output/Help/SetupConfiguration/UserExperience/EUpdates_Version_2010_Release_2_User_Interface_Overview.html?NS_VER=2010.2.0#1111021	
Form Deployment	https://system.netsuite.com/help/helpcenter/en_US/Output/Help/SuiteFlex/Customization/Forms_CustomForms_2.html?NS_VER=2010.2.0	
Go to the Web	www.google.com	
View Case Saved Searches	https://system.netsuite.com/app/common/search/savedsearches.nl?sortcol=title&sortdir=ASC&csv=HTML&OfficeXML=F&pdf=F&use=GENERAL&type=Case&accesslevel=ALL&scheduled=ALL	
Mass Email Groups	https://system.netsuite.com/app/crm/common/merge/mailmerge.nl?templatetype=EMAIL	
View Item Saved Searches	https://system.netsuite.com/app/common/search/savedsearches.nl?sortcol=title&sortdir=ASC&csv=HTML&OfficeXML=F&pdf=F&use=GENERAL&type=Item&accesslevel=ALL&scheduled=ALL	
Manage Customer Duplicates	https://system.netsuite.com/app/common/entity/manageDuplicates.nl?id=&type=&sortcol=+DupMatchString&sortdir=ASC&csv=HTML&OfficeXML=F&pdf=F&name=basic&type=custJob&mopt=and	
View Customer Saved Searches	https://system.netsuite.com/app/common/search/savedsearches.nl?sortcol=title&sortdir=ASC&csv=HTML&OfficeXML=F&pdf=F&use=GENERAL&type=Customer&accesslevel=ALL&scheduled=ALL	
Opportunities	https://system.netsuite.com/app/common/search/searchresults.nl?searchid=157	

- In the **Label** column, enter a name for the custom link.
- In the **URL** column, enter the URL for the link.
- In the **Script ID** field, enter a unique alphanumeric ID for the custom center link.
- Click **Add**.

Note: The link will open in the existing window. You cannot set up a link to open in a new window.

- Click **Save**.

8. To add the link to a custom category, go to Customization > Centers and Tabs > Center Categories.
9. Click **Edit** next to the category for which you want to add a link.

LINK*	LABEL	TRANSLATION	SHORT LIST
⋮ New Corporate Leads	New Corporate Lead		
⋮ Customer Profile	Customer Info		
⋮ Referrer	Referral		
⋮ Mass Email Contacts	Send Email Blast		

10. From the **Link** list, select the label of the custom link.
11. Click **Add**.
12. Click **Save**.

Note: On the **Translation** subtab, enter translated labels for the custom center link. The Translation subtab is available only if the Multi-Languages feature is enabled in your account. For more information, see the help topic [Configuring Multiple Languages](#).

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, NetSuite uses the standard center.

To assign a custom center to a custom role:

1. Go to Setup > User/Roles > Manage Roles > New > New.
2. In the **Name** field, enter a name for the new role.
3. In the **ID** field, enter an ID for the new role, if required.
4. From the **Center Type** list, select the custom center that you created.
5. Enter all other required settings for the new role. For information, see the help topic [Customizing or Creating NetSuite Roles](#).
6. 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, 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. From the **Role** list, 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 [Roles and Accounts](#). 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 [Sublists and Subtabs](#). 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] [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 access the NetSuite UI only by using SuiteScript APIs. For information about using SuiteScript APIs to customize the UI, see the help topic [SuiteScript 2.x 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 [Sublists and Subtabs](#).

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 screenshots show the difference between a form that has been upgraded to include the [Form Layout Enhancements](#) and one that has not.

The first screenshot is a view of a sales order with a custom sales order form applied. In this screenshot, the [Field Groups](#) and the [Sublists and Subtabs](#) changes associated with the Form Layout Enhancements have not yet been applied to the custom form used for this record.

Customer		CLASS	Summary																																																																																								
2 Acme Inc.		New Business	JOB		LOCATION	TOTAL (ALT.SALES)	0.00	DATE		East Coast	8/19/2014		TERMS	SUBTOTAL	660.00	ORDER #		SALES REP	SORD100384		SALES REP PHONE	DISCOUNT		OPPORTUNITY			SUBSIDIARY		START DATE	TAX TOTAL	6.05	Parent Company		END DATE	SHIPPING COST	7.00	PAIRED INTERCOMPANY TRANSACTION			HANDLING COST		INTERCOMPANY STATUS			GIFT CERTIFICATE		DEFERRED REVENUE RECLASSIFICATION ACCOUNT			TOTAL	673.05	29001 Deferred Revenue Sys					FOREIGN CURRENCY ADJUSTMENT REVENUE ACCOUNT					CURRENCY		EST. EXTENDED COST			US Dollar		58.48			DEPARTMENT		EST. GROSS PROFIT			Accounting		601.52					EST. GROSS PROFIT PERCENT					91.1394%		
JOB		LOCATION	TOTAL (ALT.SALES)	0.00																																																																																							
DATE		East Coast	8/19/2014		TERMS	SUBTOTAL	660.00	ORDER #		SALES REP	SORD100384		SALES REP PHONE	DISCOUNT		OPPORTUNITY			SUBSIDIARY		START DATE	TAX TOTAL	6.05	Parent Company		END DATE	SHIPPING COST	7.00	PAIRED INTERCOMPANY TRANSACTION			HANDLING COST		INTERCOMPANY STATUS			GIFT CERTIFICATE		DEFERRED REVENUE RECLASSIFICATION ACCOUNT			TOTAL	673.05	29001 Deferred Revenue Sys					FOREIGN CURRENCY ADJUSTMENT REVENUE ACCOUNT					CURRENCY		EST. EXTENDED COST			US Dollar		58.48			DEPARTMENT		EST. GROSS PROFIT			Accounting		601.52					EST. GROSS PROFIT PERCENT					91.1394%										
8/19/2014		TERMS	SUBTOTAL	660.00																																																																																							
ORDER #		SALES REP	SORD100384		SALES REP PHONE	DISCOUNT		OPPORTUNITY			SUBSIDIARY		START DATE	TAX TOTAL	6.05	Parent Company		END DATE	SHIPPING COST	7.00	PAIRED INTERCOMPANY TRANSACTION			HANDLING COST		INTERCOMPANY STATUS			GIFT CERTIFICATE		DEFERRED REVENUE RECLASSIFICATION ACCOUNT			TOTAL	673.05	29001 Deferred Revenue Sys					FOREIGN CURRENCY ADJUSTMENT REVENUE ACCOUNT					CURRENCY		EST. EXTENDED COST			US Dollar		58.48			DEPARTMENT		EST. GROSS PROFIT			Accounting		601.52					EST. GROSS PROFIT PERCENT					91.1394%																		
SORD100384		SALES REP PHONE	DISCOUNT																																																																																								
OPPORTUNITY			SUBSIDIARY		START DATE	TAX TOTAL	6.05	Parent Company		END DATE	SHIPPING COST	7.00	PAIRED INTERCOMPANY TRANSACTION			HANDLING COST		INTERCOMPANY STATUS			GIFT CERTIFICATE		DEFERRED REVENUE RECLASSIFICATION ACCOUNT			TOTAL	673.05	29001 Deferred Revenue Sys					FOREIGN CURRENCY ADJUSTMENT REVENUE ACCOUNT					CURRENCY		EST. EXTENDED COST			US Dollar		58.48			DEPARTMENT		EST. GROSS PROFIT			Accounting		601.52					EST. GROSS PROFIT PERCENT					91.1394%																										
SUBSIDIARY		START DATE	TAX TOTAL	6.05																																																																																							
Parent Company		END DATE	SHIPPING COST	7.00																																																																																							
PAIRED INTERCOMPANY TRANSACTION			HANDLING COST																																																																																								
INTERCOMPANY STATUS			GIFT CERTIFICATE																																																																																								
DEFERRED REVENUE RECLASSIFICATION ACCOUNT			TOTAL	673.05																																																																																							
29001 Deferred Revenue Sys																																																																																											
FOREIGN CURRENCY ADJUSTMENT REVENUE ACCOUNT																																																																																											
CURRENCY		EST. EXTENDED COST																																																																																									
US Dollar		58.48																																																																																									
DEPARTMENT		EST. GROSS PROFIT																																																																																									
Accounting		601.52																																																																																									
		EST. GROSS PROFIT PERCENT																																																																																									
		91.1394%																																																																																									

The second screenshot 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.

The screenshot shows the Sales Order form for record SORD100384. At the top, there's a header bar with buttons for Edit, Back, Approve, Cancel Order, Create Deposit, Manage Revenue Recognition, and Actions. A 'PENDING APPROVAL' status indicator is also present. Below the header, the form is divided into several sections:

- Primary Information:** Contains fields for ORDER # (SORD100384), PO #, CUSTOMER (2 Acme Inc.), MEMO, PROJECT, DATE (8/19/2014), START DATE, and END DATE.
- Summary:** A summary table showing financial details:

TOTAL (ALT. SALES)	0.00
SUBTOTAL	660.00
DISCOUNT ITEM	
TAX TOTAL	6.05
SHIPPING COST	7.00
HANDLING COST	
GIFT CERTIFICATE	
TOTAL	673.05
- Sales Information:** Contains fields for SALES REP, SALES EFFECTIVE DATE (8/19/2014), LEAD SOURCE (555 lead source test), OPPORTUNITY, EXCLUDE COMMISSIONS, and PARTNER.
- Classification:** Contains fields for SUBSIDIARY (Parent Company), CLASS (New Business), and LOCATION (East Coast). It also includes DEPARTMENT (Accounting).
- Intercompany Management:** Contains fields for PAIRED INTERCOMPANY TRANSACTION and INTERCOMPANY STATUS.
- Deferred Revenue Reclassification Account:** Contains fields for DEFERRED REVENUE RECLASSIFICATION ACCOUNT, FOREIGN CURRENCY ADJUSTMENT REVENUE ACCOUNT, and DOCUMENT DATE.

- Updated **Sublists and Subtabs** – 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.

The screenshot shows the Items tab of a form. The top navigation bar includes tabs for Items, Shipping, Billing, Accounting, Relationships, Sales Team, Communication, Related Records, System Information, Custom, and Accounting Books. Under the Items tab, there are several input fields and a table:

ITEM	COMMITTED	PICKED	PACKED	FULFILLED	INVOICED	REV. COMMITTED	BACK ORDERED	QUANTITY	UNITS	INVENTORY DETAIL	DESCRIPTION	PRICE LEVEL	RATE
\$100 shake	0	0	0	0	0	0		1				Base Price	100.00
Assorted Bandages, Sterile, Large, Green	0	0	0	0	0	0		1				Custom	
Avantyx bottles	0	0	0	0	0	0		1				Online Price	555.00

See the following topics:

- [Field Groups](#)
- [Sublists and Subtabs](#)

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 screenshot shows fields organized into six 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 can 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 in the following screenshot), fields have been automatically organized into the **Primary Information**, **Email | Phone | Address**, and **Classification** field groups.

The screenshot displays the 'Vendor' record page for 'Acme Medical Supply'. The page is organized into three main sections: **Primary Information**, **Email | Phone | Address**, and **Classification**.

- Primary Information:** Contains fields for CUSTOM FORM (Standard Vendor Form), VENDOR ID (Acme Medical Supply), NAME, TYPE (selected: COMPANY), COMPANY NAME (Acme Medical Supply), and WEB ADDRESS.
- Email | Phone | Address:** Contains fields for EMAIL, PHONE, ALT. PHONE, FAX, and ADDRESS (Acme Medical Supply, 1234 Sepulveda Blvd., Los Angeles, CA 94321). A 'Map' link is also present.
- Classification:** Contains SUBSIDIARY (Parent Company) and an EXPORT TO OPNAIR checkbox.

At the bottom, a navigation bar includes tabs for Relationships, Communication, Address, Marketing, **Financial**, Preferences, System Information, Custom, and a menu icon.

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.

The screenshot shows the 'Financial' tab selected on the top navigation bar of the NetSuite Account form. The form is organized into several sections:

- Account Information:** Contains fields for Legal Name, Primary Currency (set to US Dollar), Account, Default Expense Account, Default Payables Account, Terms (set to Net 60), Credit Limit, and Incoterm.
- Tax Information:** Contains fields for TAX ID and a checkbox for 1099 ELIGIBLE.
- Balance Information:** Shows Balance (37,473.64) and Unbilled Orders (9,797.53). It also shows Balance (Base) (37,473.64) and Unbilled Orders (Base) (9,797.53).
- Project Information:** Contains fields for Project Resource (checkbox) and Work Calendar (set to Default Work Calendar).

Administrators can use SuiteBuilder to customize field group titles, order, and layout. For information about customizing field groups, see [Configuring Field Groups](#) in the NetSuite Help Center.

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 about 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.

Sublists and Subtabs

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. For more information, see [Added Subtabs and Sublists](#). They will also notice that certain subtabs and sublists have been removed. For more information, 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)

- **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 (appears only 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 unsupported 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] [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.



Note: As of 2012.2, the upgrade process has been simplified to remove the Skip Upgrade option.

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.

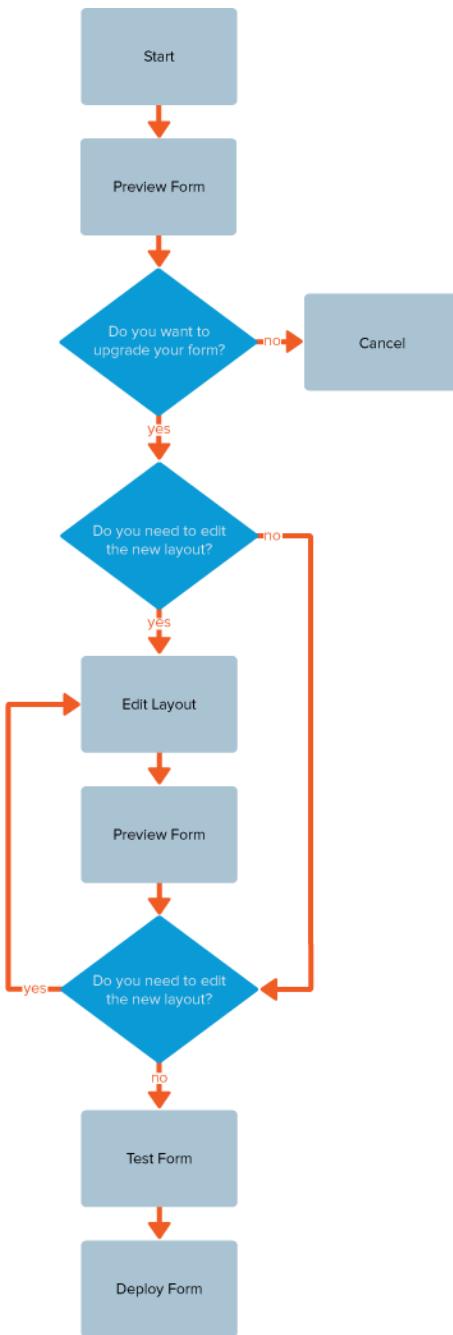
Custom Transaction Forms						
<input type="button" value="Submit"/> Item Receipt — Work Order Close ▾						
						Total: 102
EDIT	INTERNAL ID	NAME	TYPE A	SCRIPT	LIBRARY SCRIPT	PREFERRED
Customize	39	Standard Item Receipt	Item Receipt		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Customize	30	Standard Journal Entry	Journal		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Edit	138	Custom Journal Entry	Journal		<input type="checkbox"/>	<input type="checkbox"/>
Customize	77	Standard Opportunity	Opportunity		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Edit	116	Custom Opportunity	Opportunity		<input type="checkbox"/>	<input type="checkbox"/>



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.

Note: To enlarge the diagram, press Ctrl+Plus Sign. To return your screen to its usual size, press Ctrl+Minus Sign.



Note: As of 2012.2, the upgrade process has been simplified to remove the Skip Upgrade option.

The following steps outline the process for deploying upgraded forms.

1. Open the Upgrade Checklist. Go to Customization > Forms > Entry Forms [or Transaction Forms] [or Transaction 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 [Sublists and Subtabs](#).

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. For more information, (see [Deploying Skipped Custom Forms](#)). 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 autogenerated field groups (which you can later customize).

Previewing Undeployed Custom Forms

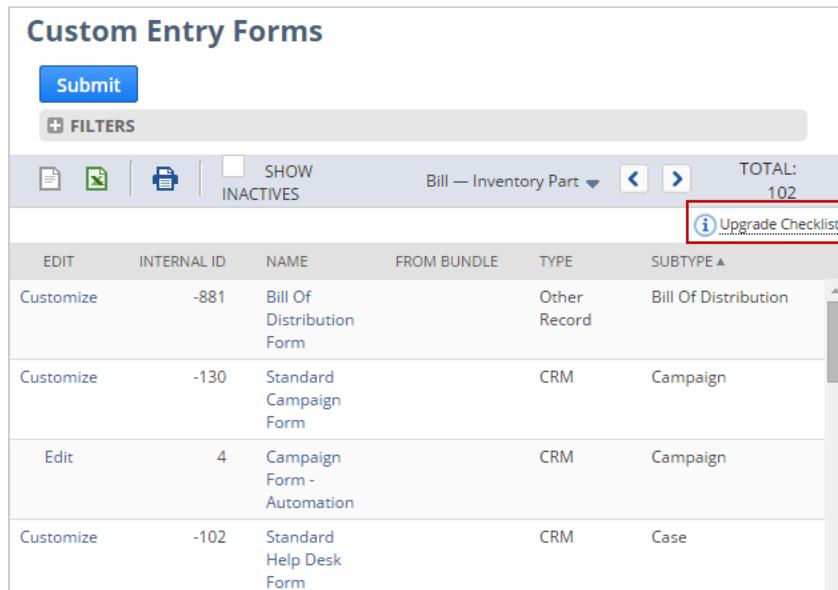
Before deploying upgraded custom forms to end users, NetSuite lets you preview what the forms will look like after you deploy them. The preview process for custom forms is **extremely important**, as it lets you consider the following:

- 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. Go to the **Upgrade Checklist**:
 - a. Go to Customization > Forms > Transaction Forms [or Entry Forms] [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**.



Custom Entry Forms					
<input type="button" value="Submit"/> <input type="button" value="FILTERS"/>					
EDIT	INTERNAL ID	NAME	FROM BUNDLE	TYPE	SUBTYPE ▲
Customize	-881	Bill Of Distribution Form		Other Record	Bill Of Distribution
Customize	-130	Standard Campaign Form		CRM	Campaign
Edit	4	Campaign Form - Automation		CRM	Campaign
Customize	-102	Standard Help Desk Form		CRM	Case

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. Go to the Upgrade Checklist:

- a. Go to Customization > Forms > Transaction Forms [or Entry Forms] [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 **Edit Layout** column, click **Edit** next to the custom form you have previewed and now want to modify.

i 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 end 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

		Edit Layout	Preview Form	Status	Deploy Options
<input type="checkbox"/> Show Inactives					
Custom Service Invoice	Edit	Progress Invoice - Automation		Deploy Form	
Custom Product Invoice	Edit	Service Invoice - Automation		Deploy Form	
Custom Journal Entry	Edit	Professional Invoice - Automation		Deploy Form	
Standard Opportunity	Edit	Custom Opportunity		Deploy Form	
Standard Drop Ship Purchase Order	Edit	Custom Opportunity 2		Deploy Form	
Standard Purchase Order					

3. Use the page that appears to make your additional layout changes.



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. Return to the Upgrade Checklist:
 - a. Go to Customization > Forms > Transaction Forms [or Entry Forms] [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 the Steps 2 to 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.

Avoid Editing Custom Forms in Tandem

You should not simultaneously edit the forms that are currently in use (in your production account) and their corresponding forms accessed through the Upgrade Checklist. For example, you should not 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 sync with the version in production.

Instead, you should select 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. Go to the Upgrade Checklist:
 - a. Go to Customization > Forms > Transaction Forms [or Entry Forms] [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.



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.

6. If you have any client or user event scripts associated with the form or the record type your are testing, verify that the scripts process as expected.
7. After testing your custom form, return 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. Go to the Upgrade Checklist:
 - a. Go to Customization > Forms > Transaction Forms [or Entry Forms] [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. Go to the Upgrade Checklist:
 - a. Go to Customization > Forms > Transaction Forms [or Entry Forms] [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 [Sublists and Subtabs](#) 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 may have placed on a custom field group will automatically be placed 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:

Status	Description
< No status at all >	The account administrator has not performed any action on the form. The form has not been deployed (by clicking the Deploy Form link).
Deployed	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.
Skipped Upgrade	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.

Status	Description
	Important: 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.

Understanding Form Layout Enhancement Upgrade Logic

When you preview custom forms, NetSuite performs 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 where ever 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 about 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 be moved only 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.

Existing Form	Upgraded Form
Custom Fields	
Custom fields that are on a valid standard subtab	Will be kept on the same subtab when the form is upgraded.

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 autogenerated 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, under the autogenerated field groups.

After previewing upgraded **custom** forms, if you are not satisfied with the autogenerated layout, you can reassign fields to alternate groups, relabel or remove field groups, and create your own custom fields groups. You can also move fields between the new subtabs that were autogenerated. 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 boxes 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 [Sublists and Subtabs](#) 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 boxes 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**.

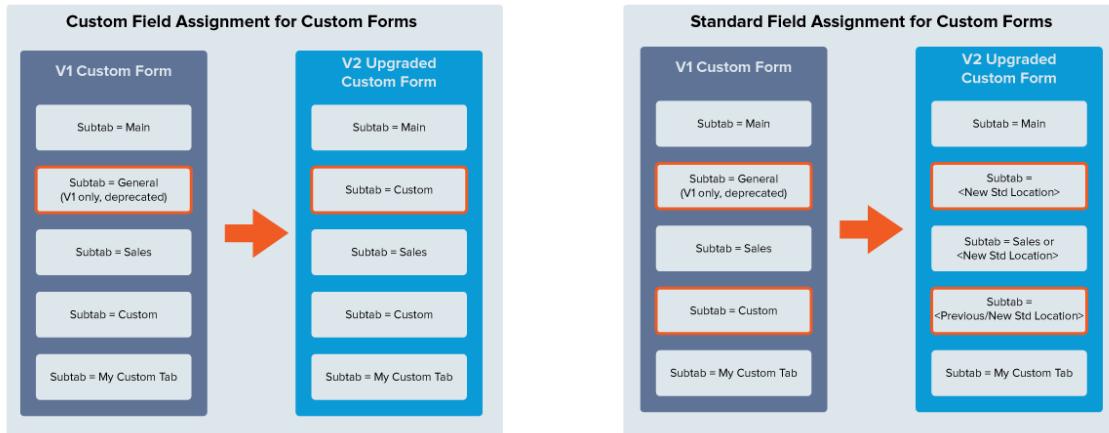
The following diagram shows the subtab assignment for fields in a "V1" (nonupgraded) environment and a "V2" (upgraded) environment.

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 removed 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 usual 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 boxes 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 boxes in the Preferred column at Customization > Forms > Transaction Form PDF Layouts or Customization > Forms > Transaction Form HTML Layouts, and clicking Submit.