NetSuite Workflows Guide

States, actions, transitions, triggers and conditions, plus how to build, read and troubleshoot SuiteFlow workflows.

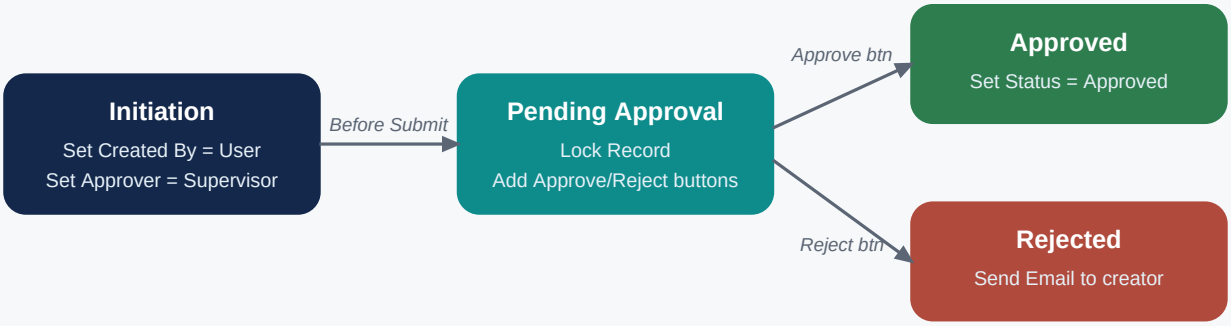
1. What Is a NetSuite Workflow?

A workflow is a set of automated rules that runs against a NetSuite record, such as a purchase order, journal entry, vendor bill or customer. NetSuite's workflow tool is called **SuiteFlow**. It lets administrators automate business processes (approvals, field defaults, alerts, record locking, email notifications) through a point-and-click editor, with no code required.

Every workflow is built from the same five building blocks. Learn these and you can read, build or troubleshoot almost any workflow in your account.

Building block	In plain terms	Example
State	A stage the record sits in. The boxes on the diagram.	Initiation, Pending Approval, Approved
Action	Something the workflow does while in a state.	Set a field, send an email, lock the record
Transition	A rule that moves the record from one state to another. The arrows.	Move to Approved when the Approve button is clicked
Trigger	The moment in the record's life when an action or transition fires.	On Entry, Before Record Submit, After Record Submit
Condition	A filter that decides whether something should happen at all.	Only when Amount > 10,000 or Subsidiary = UK

STATES (boxes) hold ACTIONS. TRANSITIONS (arrows) move the record between states.



Example: a simple journal / purchase approval flow. Each arrow fires on a trigger and can carry a condition.

2. Setup Before You Start

Before you can build workflows, make sure the account and your role are ready:

- **Enable the feature:** go to *Setup* > *Company* > *Enable Features* > *SuiteCloud* and tick **SuiteFlow** (Workflow). Once enabled, records gain workflow subtabs under System Information.
- **Check permissions:** your role needs the **Workflow** permission (Setup type). Administrators have it by default.

- **Create a workflow:** go to *Customization* > *Workflow* > *Workflows* > *New*. Existing workflows are listed on the same page, ideally each with a clear description.
- **Use a sandbox if you have one.** Build and test there before deploying to production.

The workflow definition

When you open a workflow, the **workflow workspace** (the diagram) is on the left and the **workflow definition** is on the right. The definition holds the high-level settings that control where and when the workflow runs:

Setting	What it controls
Record Type / Sub Types	The record the workflow runs on, e.g. Transaction > Purchase Order, Journal, or Customer.
Description	Plain-English summary of the purpose. Always fill this in; it is the first thing others will read.
Owner	The responsible user. In Testing mode, the workflow only runs for the owner.
Release Status	**Not Initiating** (off), **Testing** (owner only), **Released** (live for all users), **Suspended** (paused).
Execute as Admin	Runs with administrator privileges instead of the current user's. Powerful, so use deliberately.
Enable Logging	Records detailed step-by-step logs of what the workflow tried to do. Essential for troubleshooting.
Initiation	**Event Based** (fires when a user or script touches the record) or **Scheduled** (runs on a timetable against a saved search).
Event Definition	Which events start it: **On Create**, **On View or Update**, or both, plus trigger type and event types (Create, Edit, Approve, etc.).
Condition	Restricts which records start the workflow. No condition means every matching record qualifies.

3. The Core Concepts, Briefly

States

A **state** is a stage in the process. A record is always in exactly one state of a given workflow at a time. Every workflow has a **starting state** (the first one the record enters) and usually one or more **end states**, where the workflow finishes because there is nowhere further to go. States can also hold **state fields**, and the workflow as a whole can hold **workflow fields**: custom values the workflow can store and reuse (for example, the number of times a bill has been rejected).

Actions within a state

Actions are the work a state performs. A state can hold many actions, and they run in the order listed for each trigger. Common action types include:

Action	What it does
Set Field Value	Populates or changes a field, e.g. set Created By to the current user.
Set Field Display Type	Makes a field Hidden, Disabled, Inline or Normal on the form.
Set Field Mandatory	Forces the user to fill a field before saving.
Send Email	Notifies a user, approver or customer.
Lock Record	Prevents edits while the record is in this state (useful during approval).
Add Button	Adds a custom button (e.g. Approve / Reject) that can drive a transition.
Create Record	Creates a related record, such as a follow-up task.

Action	What it does
Show Message / Return User Error	Displays a message, or blocks the save with an error.
Initiate Workflow	Starts another workflow, useful for breaking big processes into smaller ones.

Transitions between states

A **transition** moves the record from its current state to another. Each transition can have a trigger, a condition, a button that fires it, or a scheduled delay (e.g. move to Escalated after 3 days). A single state can have several outgoing transitions, which is how branching works. NetSuite evaluates them in order, and the first one whose trigger and condition are met wins, so the order of your transitions matters.

Triggers

A **trigger** defines *when* an action or transition fires. Server triggers run on NetSuite's servers as the record loads or saves; client triggers run in the user's browser while they edit the form.

Trigger	Fires when...
Entry / Exit	The record enters or leaves the state.
Before Record Load	The record is being loaded, before the user sees it. Good for hiding fields or adding buttons.
Before Record Submit	The user clicks Save, before the data is written. Good for setting or validating values.
After Record Submit	After the record has been saved. Good for emails and creating related records.
Before User Edit / Before & After Field Edit / Before User Submit	Client triggers that react to the user's activity on the form in real time.
Scheduled	On a timed basis, for scheduled workflows or delayed transitions.

Conditions

A **condition** decides *whether* something happens. Conditions can sit on the workflow's initiation, on individual actions and on transitions. You can build them three ways:

- **Visual builder:** pick a field, a compare type and a value (e.g. Subsidiary = UK).
- **Formula:** write a SQL-style or field-based formula for more complex logic.
- **Saved search:** point to a saved search; the condition passes when the record appears in its results (e.g. bill amount is greater than the PO amount).

Conditions can also compare the **old record** (before saving) with the **new record** (after saving). For example: run only when a customer's status changed *from* Closed Won *to* Lost.

4. Creating a Workflow: The Basic Steps

Step	What to do
1. Plan it on paper	Sketch the states, the actions in each, and what moves the record between them. Note the record type and the exceptions.
2. Create the definition	Customization > Workflow > Workflows > New. Set name, record type, sub type, a clear description, and Release Status = **Testing**.
3. Set initiation	Choose Event Based or Scheduled, pick On Create / On View or Update, and add any initiation condition.
4. Build the states	Add states in the workspace and rename them meaningfully (Pending Approval, not State 2).

Step	What to do
5. Add actions	Inside each state, add actions, set their triggers and conditions, and order them.
6. Add transitions	Drag from one state to the next, or use New Transition. Set the trigger, condition or button.
7. Enable logging and test	Tick Enable Logging, then test as the owner with real-world scenarios, including the edge cases.
8. Release	Switch Release Status to **Released**. Consider turning logging off once stable.

5. How to Read an Existing Workflow

Inheriting a workflow someone else built can be daunting. The list description gives you an idea of its purpose, but not the precise actions or how they connect. Open it (in view or edit mode) and work through it in the same order every time:

- **Read the definition first.** What record and sub type does it run on? Is it executing as admin (high privileges)? Is it in Testing or Released?
- **Check the event definition and conditions.** On Create only, On View or Update, or both? Which trigger type? No conditions means it fires for every matching record.
- **Find the start and the end.** Locate the starting state and any end states before following the arrows.
- **Open each state** and drill into its actions and their parameters, not just the summary in the workspace.
- **Follow the transitions in sequence.** Opening a state shows all its outgoing transitions in one list, which is the easiest way to read branches (if this, go there; if that, go elsewhere).
- **Open any saved searches** used as conditions, in **edit** mode, to see the criteria rather than the results.
- **Write it down.** For large workflows, noting each step on paper or in a notepad makes the logic far easier to follow.

> **Read every action as a sentence**
>
> **On [trigger], [action] [parameter] when [condition].**
>
> *Example:* "On Entry, Set Field Value of Employee to the current User when Subsidiary is UK." Apply the same pattern to transitions: "Before Submit, move from Initiation to Pending Approval when..." The logic holds no matter how complex the workflow gets.

6. Logs and Troubleshooting

Turning on logging

Tick **Enable Logging** in the workflow definition. NetSuite then records every step it tried to take and the logic it applied at each one.

Where to find the history

Open the record the workflow runs on (e.g. the customer or the purchase order) and go to the **System Information** subtab. With SuiteFlow enabled, it has three sections:

Section	What it shows
System Notes	The standard audit trail: every change to fields on the record.
Active Workflows	Which workflows are running on this record right now, and which state each is in.
Workflow History	When the record entered and exited each state, in each workflow, newest first, with a **Log** link when logging is enabled.

Workflow History is invaluable when several workflows fire on the same record in different sequences. Click **Log** to open the detailed log (in a pop-up or new tab). It is grouped by trigger (Entry, Before Load, Before Submit, After

Submit, etc.). Expand each one to see the exact actions attempted and any failures. Ticking **Show Rejected Actions** and **Show IDs** adds detail, including actions skipped because their condition was not met.

> **Real example from the log**
> A workflow tried to create a Task record, set the due date and message, and failed. The log showed why: the required **Title** field was never supplied by the workflow. Without the log, the action simply would not have appeared to work.

Stopping a stuck workflow

Sometimes an instance must be stopped mid-flow, for example an approval whose approver has left the company, or a test instance that locked a record. Under **Active Workflows** on the record, click **Cancel** on that instance and confirm. Any lock it applied is released and normal buttons return.

> **Admin tip: control who can cancel**
> Users who can see the Active Workflows list could cancel an approval and bypass your process. Customize the form to hide the workflow sublists from standard users. Controlling your forms matters as much as controlling your workflows.

Quick troubleshooting checklist

- Is the Release Status **Released**? In Testing, it only runs for the owner.
- Does the record match the record type, sub type, event type and initiation condition?
- Is the context right (UI, CSV import, web services, scripts)? Workflows can be limited by context.
- Are the transitions in the right order? The first one that matches wins.
- Did an action's condition quietly fail? Check rejected actions in the log.
- Is another workflow or a script changing the same fields? Check Workflow History and System Notes.
- Does the role running it have permission? If not, consider whether Execute as Admin is appropriate.

7. Best Practices

- Give every workflow a clear name and a full description of what it does and why.
- Name states and actions descriptively so the diagram reads like a process map.
- Prefer several small, focused workflows over one giant one, but document how they interact.
- Always test in Testing mode (and a sandbox) with real scenarios before releasing.
- Use logging while building and troubleshooting; turn it off when stable to keep history lean.
- Use Execute as Admin only when the process genuinely needs higher privileges.
- Review workflows when processes, people or approval hierarchies change.

Want to go further?

This guide is a starting point. For full, step-by-step video training on NetSuite workflows, including building approval processes, advanced conditions, scheduled workflows and real-world troubleshooting, **sign up for the full courses on the SuperTraining website.**