



Jira Service Desk 4.0

Contents

Jira Service Desk Server 4.0 documentation	4
Installing Jira Service Desk	4
Jira applications overview	7
Permissions overview	10
Getting started with Jira Service Desk	16
Getting started for service desk admins	18
Setting up your service desk	20
Creating service desk request types	21
Making queues for your service desk teams	24
Adding service desk agents	27
Customize your service desk channels	29
Bring your service desk to the next level	31
Introduce customers to your service desk	32
Explore a sample project	34
Getting started for service desk agents	35
Administering service desk projects	37
Managing access to your service desk	38
Configuring the customer portal	40
Receiving requests by email	43
Managing the email channel	46
Troubleshooting issues with the email channel	47
Setting up service desk users	52
Managing project role memberships	55
Setting up queues for your team	56
Automating your service desk	57
Managing service desk notifications	61
Setting up request types	66
Troubleshooting issues with request types	70
Setting up SLAs	72
How teams see SLAs	76
Importing SLAs	80
Using JQL queries specific to SLAs	81
Reporting on SLAs	82
Example: creating a basic SLA	84
Example: creating an SLA that doesn't track continuous time	85
Example: creating an SLA with multiple cycles	85
Example: creating an SLA based on due date	86
Setting up approvals	87
Setting up service desk reports	91
Default service desk project configuration	96
Using Jira applications with Hipchat	98
Using Jira applications with Confluence	103
Working on service desk projects	106
Working with issues	107
Attaching files and screenshots to issues	108
Creating issues and sub-tasks	109
Creating issues using the CSV importer	112
Editing and collaborating on issues	117
Linking issues	121
Editing multiple issues at the same time	126
Moving an issue	130
Visual editing	131
Scheduling an issue	132
Logging work on issues	133
Approving a service desk request	137
Customizing the issues in a project	138

Searching for issues	140
Basic searching	143
Quick searching	145
Advanced searching	149
Advanced searching - fields reference	155
Advanced searching - keywords reference	190
Advanced searching - operators reference	192
Advanced searching - functions reference	202
Search syntax for text fields	223
Saving your search as a filter	228
Working with search results	232
Constructing cron expressions for a filter subscription	242
Managing your user profile	244
Allowing OAuth access	247
Requesting apps	247
Using keyboard shortcuts	248
Adding announcements	249
Adding customers	255
Adding request participants	256
Using service desk queues	257
Raising requests on behalf of customers	259
Organizing work with versions	259
Organizing work with components	262
Workflows	263
Using Jira on a mobile device	265
Configuring dashboards	266
Adding and customizing gadgets	268
Gadgets for Jira applications	269
Set up a knowledge base for self-service	291
Knowledge base settings and permissions	294
Write and search for knowledge base articles	297
Using the Help Center	300
Collecting customer satisfaction (CSAT) feedback	302
Jira Service Desk best practices	304
Best practices for designing the customer portal	304
Best practices for IT teams using Jira Service Desk	307
Fulfilling service requests with your IT service desk	309
Managing changes with your IT service desk	313
Managing incidents with your IT service desk	318
Managing problems with your IT service desk	322
Calculating priority automatically	326
Best practices for software teams using Jira Service Desk	328
Collaborate with other Jira teams on Jira Service Desk issues	328
Collect effective bug reports from customers	330
Customize Jira Service Desk's bug report workflow	332
Escalate Jira Service Desk issues to other Jira teams	333
Get set up for customer service	334
Getting help with Jira Service Desk	335

Jira Service Desk Server 4.0 documentation

Put the power of Jira in the hands of your service desk team.

Get started

New to Jira Service Desk? Check out our guides for new administrators and users.

[View guide](#)

What's new

Time to upgrade? Get the lowdown on the latest and greatest in Jira Service Desk 4.0.

[View latest changes](#)

Installing Jira Service Desk

Jira Service Desk is built on the Jira platform and has everything your IT teams need for service request, incident, problem, and change management. It's part of the Jira family of applications (Jira Software, Jira Service Desk, Jira Core) and can be used separately or in any combination, on the same instance.

If you're ready to install or upgrade Jira Service Desk, then this guide has you covered. If you have Jira Service Desk installed and would like to learn how to configure and use it, head to the [Jira Service Desk documentation](#).



1. System requirements

Before you install Jira Service Desk, you'll need to check if it supports your operating system, as well as a few other things.

Read

- [Supported platforms](#)
- [Jira applications installation requirements](#)

If everything looks good, it's time to decide which version and license is right for your organization. If you've already made a decision, jump to [Install Jira Service Desk](#).



2. Versions and licensing

If you plan on managing your own Jira Service Desk instance (not hosted by Atlassian), you have the option of either a Server or Data Center license. Your license determines which features and infrastructure choices are available.

For organizations who need more time to prepare before upgrading to a new version, but still want critical bug fixes, an Enterprise release is a good choice.

Read

- [Jira Service Desk licensing and purchasing](#)
- [Jira Server and Data Center feature comparison](#)
- [Atlassian Enterprise releases](#)

If you're clear on what you're getting and why, it's time to install Jira Service Desk.



3. Install Jira Service Desk

Here we'll talk you through how to install Jira Service Desk to trial and put into production, as well as how to add additional Jira applications to your existing installation.

Install Jira Service Desk to trial

1. Download the installer for your operating system at <https://www.atlassian.com/software/jira/service-desk/download>.
2. Follow the steps at [Evaluation installation](#). Jira Service Desk comes with a handy embedded H2 database that you can use for evaluation purposes.
3. Once installed, read [Getting started for service desk admins](#) to learn what you can do with Jira Service Desk.

If you'd like to move into production, consider starting afresh by following the [Install Jira Service Desk for production](#) guides below.

Install Jira Service Desk for production

1. Set up your database by following the guide that applies to you at [Connecting Jira applications to a database](#).
2. Install with your operating system by following the guide below that applies to you:
 - [Installing Jira applications on Windows](#)
 - [Installing Jira applications on Linux](#)
3. Once installed, follow the tutorial on [Getting started for service desk admins](#) to set up Jira Service Desk for your support teams.

Install additional Atlassian products

If you have Jira Core or Jira Software installed, you can install Jira Service Desk as an additional application, and vice versa.

Head to **Jira Administration**



> **Applications > Versions & licenses** and follow the steps below:

1. Check which version of Jira Service Desk is compatible with your existing installation at [Jira Service Desk version history](#).
2. Ensure you have Jira Administrator Global Permission.
3. Follow the steps at [Installing additional applications and version updates](#).
4. Read the [documentation](#) for the product you've installed, to learn how to get started.

There are over 600 apps available from the [Atlassian Marketplace](#) that will help you supercharge your Jira Service Desk. Read [Managing apps](#) to learn how to set these up.



4. Upgrade to a newer version

There are several ways to upgrade Jira Service Desk. The method you choose will depend on the version of Jira Service Desk you use, and the type of environment you use it in.

Steps to take

1. Decide which version to upgrade to by reading the [release and upgrade notes](#).
2. Determine your upgrade path:

Version	Upgrade path
Earlier than 2.5.x	Upgrade to 3.0, then upgrade to the latest version.
3.0 or later	Upgrade directly to the latest version.

3. Choose your method from the below options and follow the steps on that page:
 - [Using the installer](#) (the easiest way to upgrade your Jira instance)
 - [Upgrading manually](#) (if you're moving to a different operating system or database software)
 - [Upgrading Jira Data Center \(manual\)](#) with downtime
 - [Upgrading Jira Data Center](#) with zero downtime



5. Get the most out of Jira Service Desk

Now you have Jira Service Desk installed, there's plenty of resources to help you set it up, learn the features, and get your dream team using it.

Getting started

- [Getting started for service desk admins](#)
- [Getting started for service desk agents](#)

Top tasks

- [Configuring the Customer Portal](#)
- [Automating your service desk](#)
- [Setting up SLAs](#)

Best practice guides

- [Set up a knowledge base for self-service](#)
- [Best practices for IT teams using Jira Service Desk](#)
- [Best practices for software teams using Jira Service Desk](#)

Jira applications overview

Jira Service Desk licensing overview

The Jira family of applications (Jira Software, Jira Service Desk, Jira Core) are built on the Jira platform and can be used in any combination on the same instance. Depending upon of your setup, users can be licensed to one, all, or any combination of these applications. Read on to understand how Jira Service Desk licensing and roles affect what agents, customers, and other Jira application users can do.

If you're a Jira administrator, check out more information on [Licensing and application access](#).

Application features and project types

Each application delivers a tailored experience for its users, and has an associated project type which in turn offers application specific features. Below is a list of the project types, and their associated application specific features.

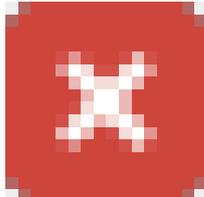
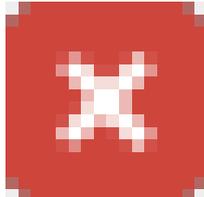
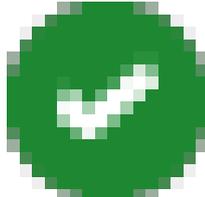
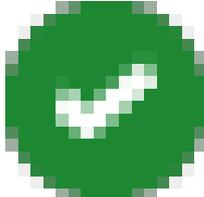
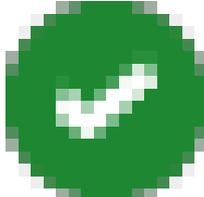
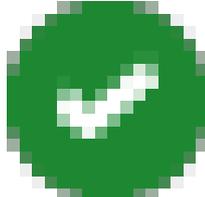
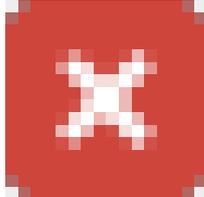
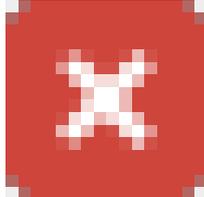
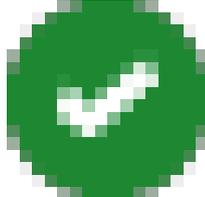
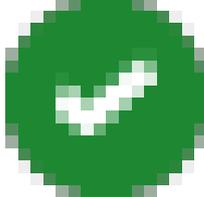
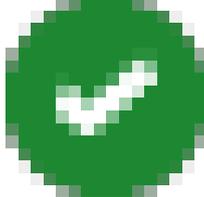
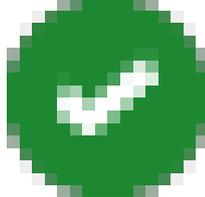
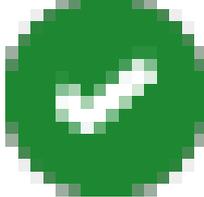
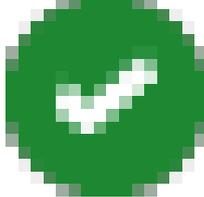
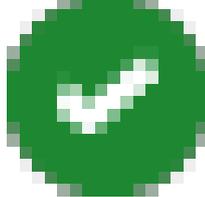
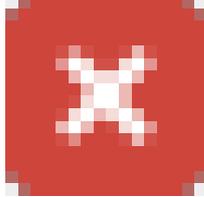
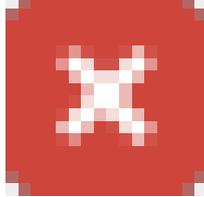
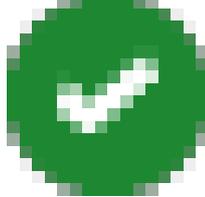
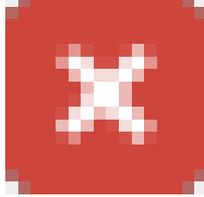
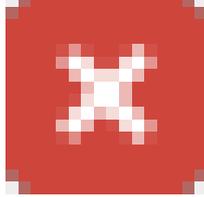
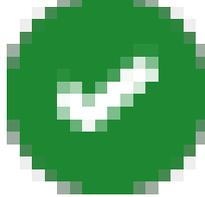
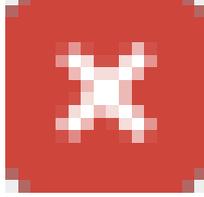
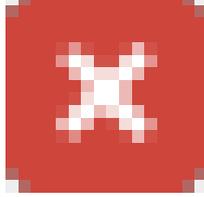
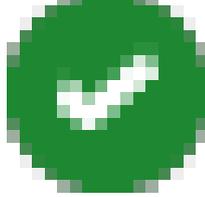
Application	Project type	Application specific feature set
Jira Core	 Business projects	<ul style="list-style-type: none"> Available to all licensed users
Jira Software	 Software projects	<ul style="list-style-type: none"> Integration with development tools Agile boards Release hub for software versions
Jira Service Desk	 Service Desk projects	<ul style="list-style-type: none"> Service Level Agreements (SLAs) A customizable web portal for customers Permission schemes allowing customer access

All users that can log in to a Jira instance will be able to see all the projects in that instance (pending permissions), but they will only be able to see the application-specific features when they have application access. For example, a Software project is able to display information from linked development tools, such as Bitbucket and FishEye, as well as agile boards, but this information is only viewable by a Jira Software user. A Jira Core user would be able to see the Software project, but would not be able to see the Software-specific features, like agile boards or the information from linked development tools. Likewise, a Jira Software user would not be able to see any Jira Service Desk application-specific features on a Service Desk project, only a basic view of the project and its issues.

- Only a Jira administrator can create a project for an installed application. They do not need application access to create the project, but they do need application access if they'd like to view or use the project.
- Anonymous users will have access equivalent to Jira Core users. In other words, they can view issues and work in any type of project, but they won't see application-specific features, e.g. agile boards, which are Jira Software-specific features. To know how to allow anonymous users access to projects, see [Allowing anonymous access to your instance](#).

A list of the applications, their user roles, and their project's application-specific features can be found below:

			Jira Core	Jira Software	Jira Service Desk	
			Jira-Core-user	Jira-Software-user	Jira-ServiceDesk-agent	Cu

Service Desk Projects 	Project level	Create				
		View				
	Issue level	Create				
		View				
		Comment				
		Transition				
	SLA level	Create				
		View				

	Queue level	Create				
		View				
	Jira Service Desk gadgets	View				

Permissions overview

This page describes the different types of permissions and access rights that can be set up in Jira applications.

What are permissions?

Permissions are settings within Jira applications that control what users within those applications can see and do. All Jira applications allow a variety of permissions: from whether users can create new projects to whether a user can see a specific type of comment on an issue. These permissions can differ between applications.

Permissions are different from application access, which is controlled by groups that have **Use** access for an application. For more information about setting application access, see [Managing user access to JIRA applications](#).

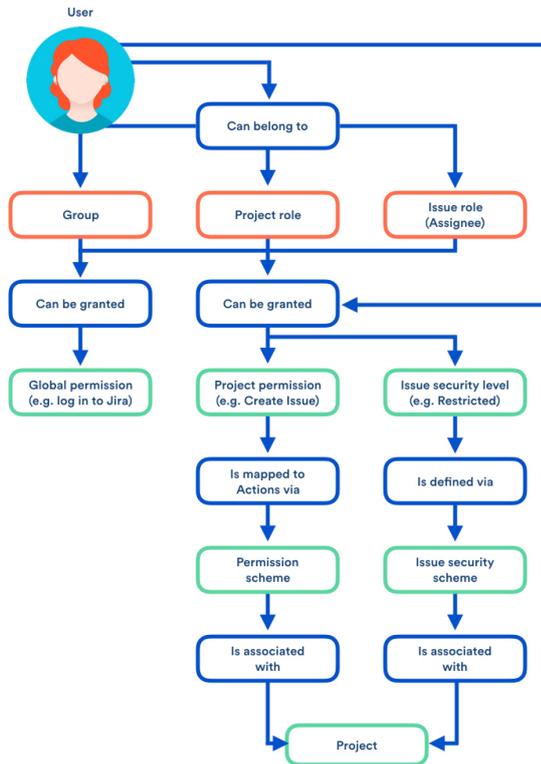
Types of permissions

There are three types of permissions in Jira applications, and they range from the high-level to granular:

- **Global permissions** - These apply to applications as a whole, not individual projects (for example, whether users can see the other users in the application).
- **Project permissions** - Organized into permission schemes, these apply to projects (e.g. who can see the project's issues, create, edit and assign them). While project admins can assign users to a project, they can't customize the permission schemes for a project. There are lots of project-level permissions you can set to control what users can do within a project.
- **Issue security permissions** - Organized into security schemes, these allow the visibility of individual issues to be adjusted (within the bounds of the project's permissions). For example, issue security permissions can let you set up types of issues that can only be seen by project admins or users in specific groups.

How do permissions get assigned?

Permissions can be assigned to groups or to project roles/and or issue roles. This diagram illustrates how permissions are assigned to users:



Who can set permissions?

Permission	Can be set by	For more info, see...
Global permission	A user with the Jira System administrator permission A user in a group with Admin access	Managing global permissions
Project permission	A user with the Jira System administrator permission A user in a group with Admin access	Managing project permissions
Issue security permission	A user with the Jira System administrator permission A user in a group with Admin access A project admin	Configuring issue-level security

Jira Service Desk global and project permissions

Jira Service Desk provides a standard permission scheme (Jira Service Desk Permission scheme for *project*) that automatically gives your service desk users the correct permissions for the project role they are in. For example, adding agents to your service desk will add users to the Service Desk Team role. This role gives them access to Jira Service Desk projects to which they're assigned and also allows them to work on issues.

Global permissions

At installation time, Jira Service Desk creates a global permission named **Jira Service Desk agent access**. If agent based pricing is enabled for the instance, users who require access to agent views or functionality need to have this permission. The number of users who are granted this permission determines how many agent licenses are used on the system.

Project permissions

This table shows the permission configuration for a standard service desk project permission scheme:

Project Permissions	Users / Groups / Project roles	Explanation
Administer Projects	Project Role (Administrators)	Permission to administer a project. This includes the ability to edit project role membership, project components, project versions and certain project details (Project Name, URL, Project Lead, Project Description).
Browse Projects	<ul style="list-style-type: none"> Service Desk Customer - Portal Access Project Role (Service Desk Team) Project Role (Administrators) 	Permission to browse projects, use the Issue Navigator and view individual issues (except issues that have been restricted via issue security). Many other permissions are dependent on this permission , e.g. the 'Work On Issues' permission is only effective for users who also have the 'Browse Projects' permission.
View Development Tools	<ul style="list-style-type: none"> Project Role (Administrators) 	
View (Read-Only) Workflow	<ul style="list-style-type: none"> Project Role (Service Desk Team) Project Role (Administrators) 	Permission to view the project's 'read-only' workflow when viewing an issue. This permission provides the 'View Workflow' link against the Status field of the 'View Issue' page .
Issue Permissions	Users / Groups / Project roles	Explanation
Create Issues	<ul style="list-style-type: none"> Service Desk Customer - Portal Access Project Role (Service Desk Team) Project Role (Administrators) 	Permission to create issues in the project. (Note that the Create Attachments permission is required in order to create attachments.) Includes the ability to create sub-tasks (if sub-tasks are enabled).
Edit Issues	<ul style="list-style-type: none"> Service Desk Customer - Portal Access Project Role (Service Desk Team) Project Role (Administrators) 	Permission to edit issues (excluding the 'Due Date' field — see the Schedule Issues permission). Includes the ability to convert issues to sub-tasks and vice versa (if sub-tasks are enabled). Note that the Delete Issue permission is required in order to delete issues. The Edit Issue permission is usually given to any groups or project roles who have the Create Issue permission (perhaps the only exception to this is if you give everyone the ability to create issues — it may not be appropriate to give everyone the ability to edit too). Note that all edits are recorded in the issue change history for audit purposes.
Transition Issues	<ul style="list-style-type: none"> Service Desk Customer - Portal Access Project Role (Service Desk Team) Project Role (Administrators) 	Permission to transition (change) the status of an issue.

Schedule Issues	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to schedule an issue — that is, to edit the 'Due Date' of an issue. In older versions of Jira this also controlled the permission to view the 'Due Date' of an issue.
Move Issues	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to move issues from one project to another, or from one workflow to another workflow within the same project. Note that a user can only move issues to a project for which they have Create Issue permission.
Assign Issues	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to assign issues to users. Also allows autocompletion of users in the Assign Issue drop-down. (See also Assignable User permission below)
Assignable User	<ul style="list-style-type: none"> • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to be assigned issues. (Note that this does not include the ability to assign issues; see Assign Issue permission).
Resolve Issues	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to resolve and reopen issues. This also includes the ability to set the 'Fix For version' field for issues. Also see the Close Issues permission.
Close Issues	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to close issues. (This permission is useful where, for example, developers resolve issues and testers close them). Also see the Resolve Issues permission.
Modify Reporter	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to modify the 'Reporter' of an issue. This allows a user to create issues 'on behalf of' someone else. This permission should generally only be granted to administrators.

Delete Issues	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to delete issues. Think carefully about which groups or project roles you assign this permission to; usually it will only be given to administrators. Note that deleting an issue will delete all of its comments and attachments, even if the user does not have the Delete Comments or Delete Attachments permissions. However, the Delete Issues permission does not include the ability to delete individual comments or attachments.
Link Issues	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to link issues together. (Only relevant if Issue Linking is enabled).
Set Issue Security	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to set the security level on an issue to control who can access the issue. Only relevant if issue security has been enabled.
Voters & Watchers Permissions	Users / Groups / Project Roles	Explanation
View Voters and Watchers	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to view the voter list and watcher list of an issue. Also, see the Manage Watcher List permission.
Manage Watcher List	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to manage (i.e. view/add/remove users to/from) the watcher list of an issue.
Comments Permissions		Explanation
Add Comments	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to add comments to issues. Note that this does not include the ability to edit or delete comments.

Edit All Comments	<ul style="list-style-type: none"> • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to edit any comments, regardless of who added them.
Edit Own Comments	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to edit comments that were added by the user.
Delete All Comments	<ul style="list-style-type: none"> • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to delete any comments, regardless of who added them.
Delete Own Comments	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to delete comments that were added by the user.
Attachments Permissions	Users / Groups / Project Roles	Explanation
Create Attachments	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to attach files to an issue. (Only relevant if attachments are enabled). Note that this does not include the ability to delete attachments.
Delete All Attachments	<ul style="list-style-type: none"> • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to delete any attachments, regardless of who added them.
Delete Own Attachments	<ul style="list-style-type: none"> • Service Desk Customer - Portal Access • Project Role (Service Desk Team) • Project Role (Administrators) 	Permission to delete attachments that were added by the user.
Time Tracking Permissions	Users / Groups / Project Roles	Explanation

Work On Issues	<ul style="list-style-type: none"> Project Role (Service Desk Team) Project Role (Administrators) 	Permission to log work against an issue, i.e. create a worklog entry. (Only relevant if Time Tracking is enabled).
Edit Own Worklogs	<ul style="list-style-type: none"> Project Role (Service Desk Team) Project Role (Administrators) 	Permission to edit worklog entries that were added by the user. (Only relevant if Time Tracking is enabled). Also, see the Work On Issues permission.
Edit All Worklogs	<ul style="list-style-type: none"> Project Role (Administrators) 	Permission to edit any worklog entries, regardless of who added them. (Only relevant if Time Tracking is enabled). Also, see the Work On Issues permission.
Delete Own Worklogs	<ul style="list-style-type: none"> Project Role (Service Desk Team) Project Role (Administrators) 	Permission to delete worklog entries that were added by the user. (Only relevant if Time Tracking is enabled). Also, see the Work On Issues permission.
Delete All Worklogs	<ul style="list-style-type: none"> Project Role (Administrators) 	Permission to delete any worklog entries, regardless of who added them. (Only relevant if Time Tracking is enabled). Also, see the Work On Issues permission.

Using custom permission schemes

If you are a service desk administrator and you want to customize the standard permission scheme, make sure that the roles have the mandatory permissions. See [Customizing Jira Service Desk permissions](#).

Resolving permission scheme errors

If you encounter any error messages related to your service desk's permission scheme, check out [Resolving Jira Service Desk permission errors](#).

Getting started with Jira Service Desk

Learn about Jira Service Desk's different user types and roles, get a brief introduction to how your customers raise requests, and learn what those requests look like for agents. When you're ready, pick a tutorial and learn how to make your service desk work better for your team and your customers.

User types and roles

Licensed users manage customer requests in your instance of Jira Service Desk. These are most likely your project administrators, service desk team members, employees and even contractors. They track and resolve your customers' requests, transition issues through workflows, and contribute to their team's service level agreements (SLAs).

Unlicensed users raise requests and interact with service desk agents *for free*. These are your customers. They submit requests via your service desk's customer portal or email address, comment on requests, and read knowledge base articles. Customers do not require a license, so you can have an unlimited number of customers who can submit an unlimited number of requests, free of charge.

Most of the information in these documents focuses on two licensed roles: administrators and agents. The administrator sets up and configures service desk projects. The agent works in these projects.

Admins

Project administrators for your service desk can:

- Access all features in Jira Service Desk
- Manage users and roles in service desk projects

- Set up customer portals, request types, queues, reports and SLAs
- Perform all tasks that agents can

Agents

Service desk team members who work on customer requests can:

- View the customer portal, queues, reports and SLA metrics for a project
- Add, edit, and delete customer-facing and internal comments on issues
- Add customers to a project
- Read knowledge base content
- Manage organizations (if allowed at the application level)
- Create knowledge base content (with a Confluence license)

Customers

The people you serve through your service desk can:

- Raise requests via the customer portal or email channel
- Track their requests in the customer portal
- Comment on their requests
- Read knowledge base articles
- Approve other customers' requests
- Share requests with other customers (if allowed by [Customer permissions](#))

All this is free for your customers.

How a service desk works (in 4 simple steps)

Here's how your customers and service desk agents work together to resolve a request:

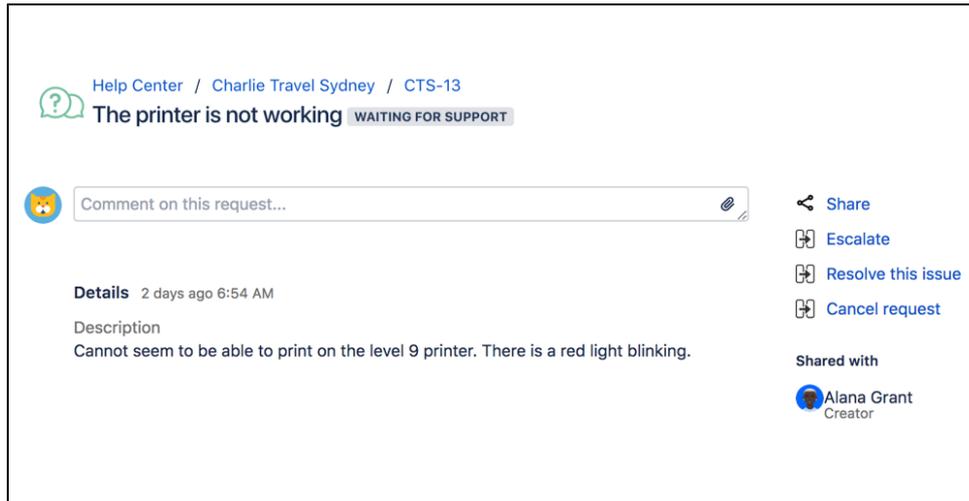
	<p>1. Your customer submits a request to your service agents through a portal or via email.</p>
	<p>2. A service desk agent sees the request in their Jira Service Desk queue and looks into the issue.</p>
	<p>3. Your customer and other participants use the portal or email to discuss the request with your service desk agent, who works in Jira Service Desk.</p>
	<p>4. Your agent completes the request and your customer is satisfied!</p>

"Requests" vs. "issues"

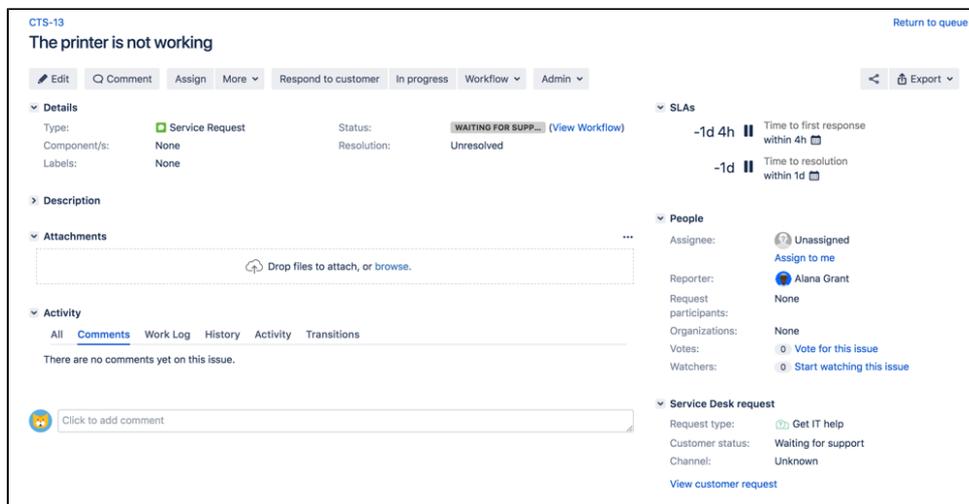
Your customers submit a *request* through the customer portal or by email. These requests become *issues* th

at your agents work on in their agent view.

How customers see their *requests* in the customer portal:



How agents see those corresponding *issues*:



Ready to get started?

Choose your Jira Service Desk tutorial:

[I am an admin](#)

[I am an agent](#)

Getting started for service desk admins

Welcome to Jira Service Desk for admins! In this tutorial, we'll introduce you to your workspace and walk you through the process of setting up a service desk project for your team of agents and a corresponding customer-facing site (which we call the customer portal).

We'll be focusing on basic Jira Service Desk features and tasks to help you get up and running quickly. By the end of this tutorial, you will have:

- Set up 1 service desk project
- Added 3 agents
- Prepared your customer portal to receive customer requests

Let's take a quick look at Jira Service Desk...

Queues



As an admin, you will set up and configure queues for your agents. Your agents will then view and work on

Audience:

- Service desk administrators
- Team managers

Time: 30 minutes

issues from the same tab:

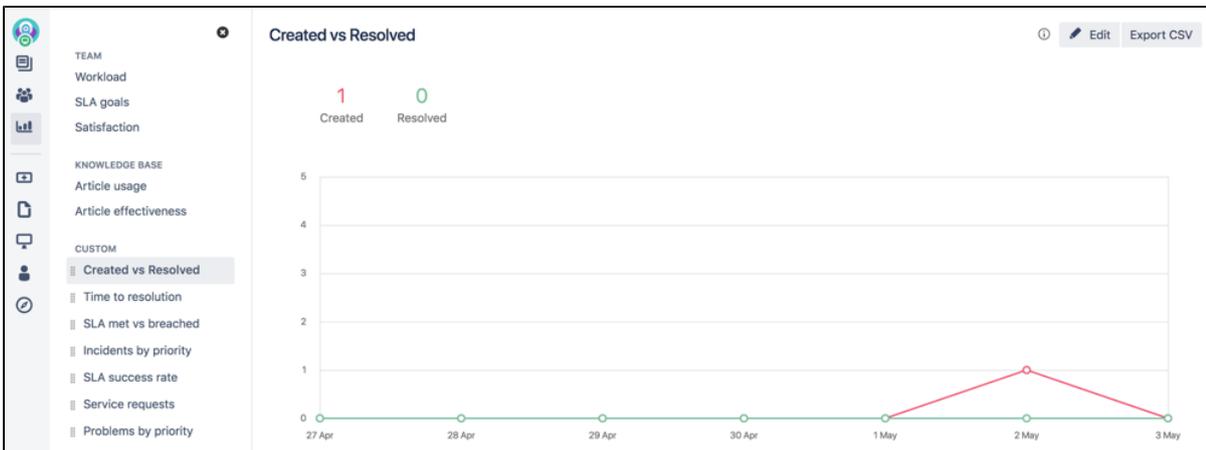
The screenshot shows the 'All open' queue in Jira Service Desk. On the left, there is a sidebar with 'Queues' and 'PROJECT SHORTCUTS'. The main area displays a table of issues:

Time to resolution	Key	Status	Summary	Created	Reporter
-3d	CTF-1	WAITING FOR SUPPO...	What am I looking at?	26/Apr/18	Karen Bywater
1h 19m	CTF-2	WAITING FOR APPRO...	3 day trip to Brisbane	02/May/18	Alana Grant

Reports



Use the Reports tab to view your team's workload. You can also set up custom reports to track your team's progress in more detail:



Project settings



Here, you will set up request types, brand your customer portal, link your service desk to an email account, and manage users:

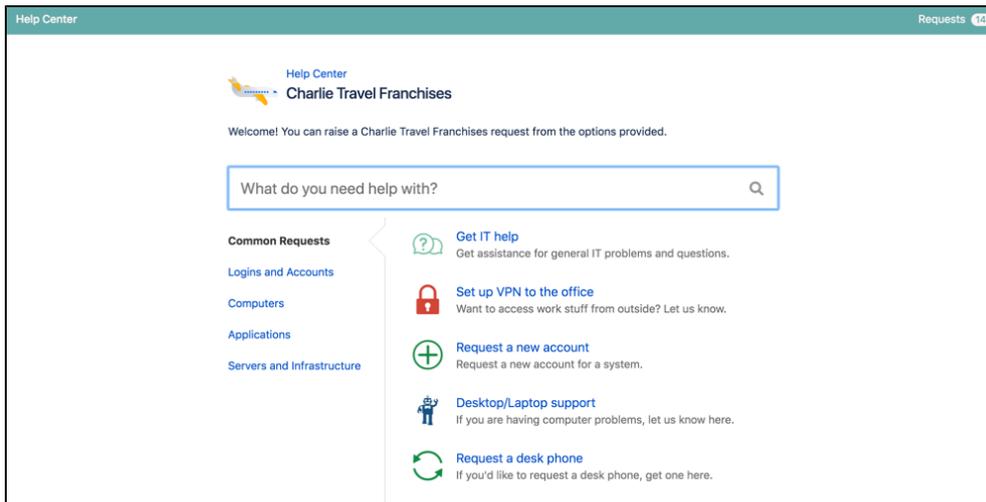
The screenshot shows the 'Project settings' page. The 'Request types' section is active, showing a list of request types under 'Common Requests':

Icon	Request name	Issue type	Description (Optional)
		Service Request	
	Get IT help	Service Request	Get assistance for general IT problems and questions.
	Report a system problem	Incident	Having trouble with a system?
	Set up VPN to the office	Service Request	Want to access work stuff from outside? Let us know.
	Request a new account	Service Request	Request a new account for a system.

Customer portal



This link lets you navigate the customer view of your service desk project:



Now that you're familiar with your service desk workspace, you can set up your own Jira Service Desk site and add your first project.

Let's go!

Setting up your service desk

1. Setting up your service desk
2. Creating service desk request types
3. Making queues for your service desk teams
4. Adding service desk agents
5. Customize your service desk channels
6. Bring your service desk to the next level
7. Introduce customers to your service desk
8. Explore a sample project

You'll need a working Jira Service Desk instance to complete this tutorial, and we have installation instructions for both Windows and Linux operating systems.

- [Windows installation instructions](#)
- [Linux installation instructions](#)

If you have an existing Jira Service Desk site, [skip ahead](#) to create a service desk project. If your administrator has set you up as a project admin for an existing project, jump to [Step 2](#) to create your request types.

Create a project

Jira Service Desk comes with default project templates that you can use to suit your team's needs. There are three templates you can choose from:

- The **Basic Service Desk** template is set up for internal business teams, like HR, finance, or small IT teams. The template comes with just a few recommended request forms and is easy to customize and expand with your needs.
- We recommend the **IT Service Desk** template for IT teams who maintain a more complex infrastructure. The template comes with ITIL-inspired workflows for change, incident, and problem management.
- If you help external customers and want to collect bug reports or take suggestions for new feature, we recommend the **Customer service** desk template. The template comes ready for your customers to request technical or billing help, and report issues.

For now, let's get you set up with a basic service desk project.

1. If you're working with an existing Jira Service Desk instance, select **Projects > Create Project** from the top navigation bar of your instance. If you've just installed and set up Jira Service Desk, you'll have the option to Create a new project by selecting **Create new project**.
2. Select "**Basic Service Desk**" as the project type.
3. Name your project. In this example, we'll use the project name "Charlie Travel Franchises". The project key should be automatically populated, but you can change the key if you'd like. If you see options to link another application, leave these options unchecked.
4. Select **Submit** to create your project.

Nice work! You now have a service desk site with one project. You will now learn to set up request types, which define the requests customers can submit to your team's service desk project.

[Next](#)

Creating service desk request types

1. [Setting up your service desk](#)
2. [Creating service desk request types](#)
3. [Making queues for your service desk teams](#)
4. [Adding service desk agents](#)
5. [Customize your service desk channels](#)
6. [Bring your service desk to the next level](#)

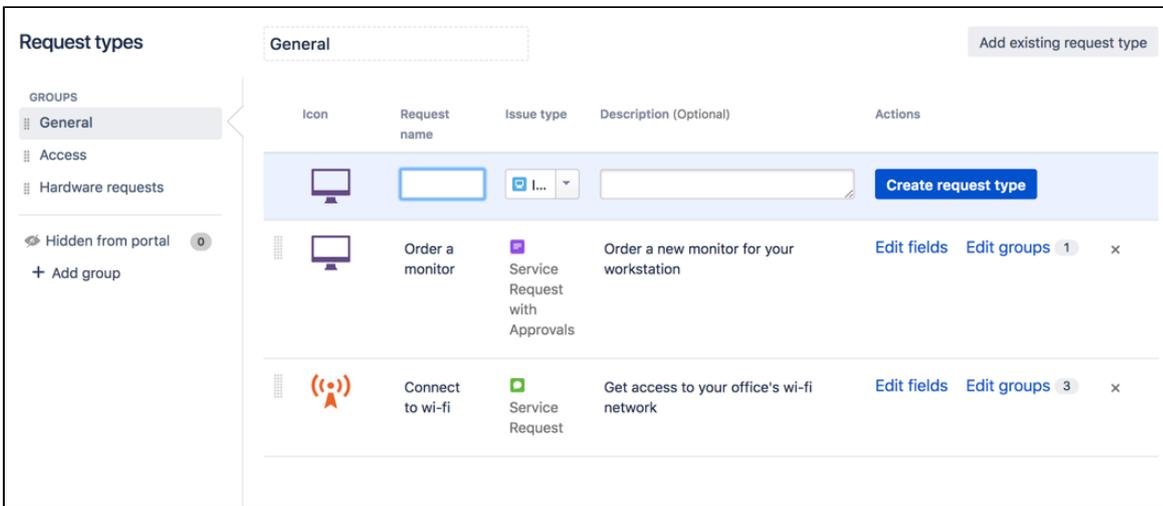
- 7. Introduce customers to your service desk
- 8. Explore a sample project

Request types let you define and organize incoming issues so your service desk team can more efficiently help your customers. If you're moving from an existing help desk application, you can add your existing request categories during this step.

If you're setting up service desk request types for the first time:

- Think about how your customer would write a request, for example 'Order a new monitor' over 'Submit a hardware request'.
- Break things down into smaller chunks, such as 'Get help with printers' or 'Get wi-fi access'.
- Avoid specialist terminology; think 'I need access' more than 'Deploy SSH key'.

By the end of this step, your project's request type page should look something like this:



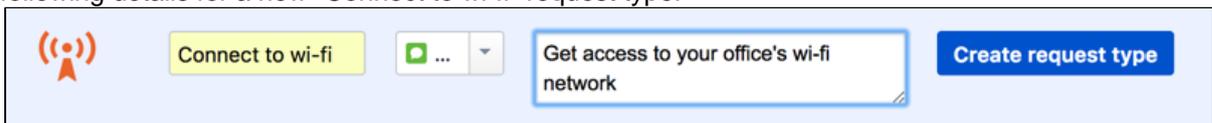
Requests vs. Issues

Remember that customers submit requests to your service desk and your team picks up the corresponding issues to work on.

Create new request types

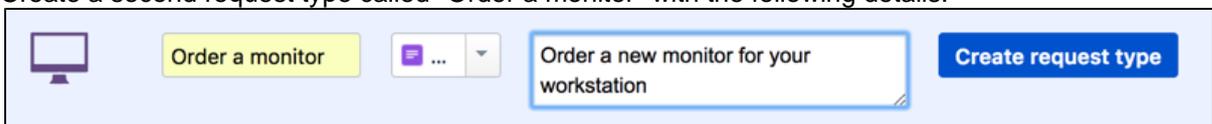
Let's go ahead and add two new request types, so you can familiarize yourself with the request type configuration options.

1. In your new service desk project, select **Project settings > Request types**.
2. In the new request type form at the top of the page, change the request type icon and enter the following details for a new "Connect to wi-fi" request type.



Select **Create request type** when you've finished entering your request type details.

3. Create a second request type called "Order a monitor" with the following details:



Select **Create request type** when finished.

Edit the fields your customers see

Now that you have requests in the customer portal, you can prompt your customers to give you the information you need to help them quickly. These simplified fields help customers understand what information they need to provide when submitting a request.

Let's add some fields to your request types, so you can collect some additional information.

1. For the "Connect to wi-fi" request type, select **Edit fields**.
2. Under **Visible fields**, click the "Summary" display name and rename it to "What do you need?". Add some placeholder text to the **Field help** to gather useful details from your customers. For example:

Display name	Required	Field help (Optional)	Actions
What do you need?	Yes	e.g. Wi-fi access for the Houston office.	Update Cancel

Issue field: Summary

Select **Update** when finished.

3. Select **Add a field** to add the "Priority" field to the request form and select **Apply**.
4. On the **Workflow Statuses** tab, you will see the default Jira workflow status names displayed on the left hand side. You can change how these statuses appear to customers by editing the *Status name to show customer* fields as shown:

Workflow status in JIRA	Status name to show customer
Resolved	Completed
Waiting for customer	Waiting for customer
Waiting for approval	Waiting for approval

Save Discard unsaved changes

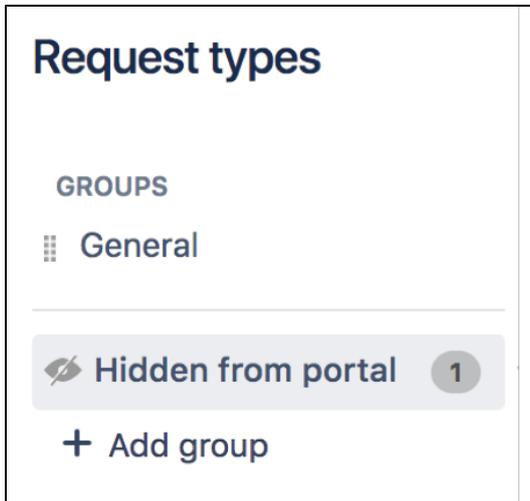
5. Select **View this request form** to see how your changes appear in the customer portal.

Organize your requests with groups

A group is simply a category you can assign to each request type. In the customer portal, your request types are organized vertical tabs based on your groups.

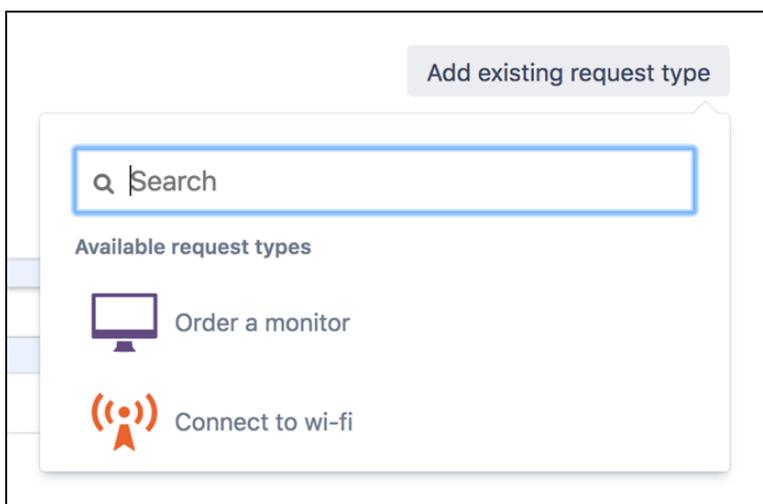
We think groups are helpful if you have *seven or more* request types.

Go to **Project settings > Request types**. You should see your groups in the sidebar:



Let's add a few groups to help your customers find the request they need:

1. To add request groups, select **+ Add group**. Add two groups for your new request types, "Hardware requests" and "Access".
2. When viewing your "Access" request group, select **Add existing request type** and choose your "Connect to wi-fi" request type:



3. Switch to your "Hardware requests" group in the sidebar.
4. Select **Add existing request type** and choose your "Order a monitor" request type.
5. Open the customer portal link from your project sidebar to see your requests organized into groups.
6. To rearrange the order of how your groups appear in the customer portal, go back to your project settings and drag and drop the groups in the request types sidebar.

Create a request from the customer portal

1. Keep the customer portal preview open, so you can create test requests from a customer's perspective.
2. Select the "Connect to wi-fi" request type.
3. Enter "Test wi-fi request" in the open field and select Medium priority.
4. Click **Create** to complete your request and view the open request in the customer portal.
5. Click **Close** to exit the customer view and return to your service desk project.

Excellent work! You now have four request types and a new issue in your project. Next, you will learn how to sort these issues into queues, which will allow you to manage your team's workload.

Next

Making queues for your service desk teams

1. Setting up your service desk
2. Creating service desk request types
3. Making queues for your service desk teams
4. Adding service desk agents
5. Customize your service desk channels
6. Bring your service desk to the next level
7. Introduce customers to your service desk
8. Explore a sample project

Your teams will spend the majority of their time working out of the queues you set up. Agents do not have the permissions to add new queues or configure existing ones; however, Jira Service Desk queues allow you to automatically triage and prioritize issues for them. If you want your team to focus on requests that must be completed by next week, for example, you can set up a queue that only contains requests with a set due date in that week.

Your site comes with preconfigured queues (for example, "Unassigned issues"), but let's go ahead and create three new queues for your team:

1. From your service desk project sidebar, select **Queues** ().
The icon is a square with rounded corners containing a document symbol with a checkmark.
2. Select **New queue** and name your first new queue "Service requests".
3. Define the issues you want to appear in this queue by selecting the following drop-down menus: **Type** (select "Service Request"); **Status** (select "Waiting for support"), and **Resolutions** (select "Unresolved");

New queue

Name

Issues to show

More ▾ Service Request ▾ Waiting for sup... ▾ Unresolved ▾

Columns

More ▾ Key × Summary × Created × Updated × Due Date ×

Create Cancel

4. Select the following column names that will display in this queue from the **More** menu: "Key", "Summary", "Created", "Updated", "Due Date". You can reorder the columns by dragging the name (for example, "Key") across the column field.
5. Select **Create** to add this queue to your team's workspace.
6. Create two new queues with the following two search queries:
"Completed requests" for Service requests that have been successfully resolved.

New queue

Name

Issues to show

More ▾ Service Request ▾ Resolved ▾ Resolution: All ▾

Columns

More ▾ Key × Summary × Created × Updated × Due Date ×

Create Cancel

"Due this week" for requests that must be completed in the next week

New queue

Name

Due this week

Issues to show

More ▾ Type: All ▾ Waiting for sup... ▾ Resolution: All ▾ Label: All ▾

Due Date: All ▾

Now overdue

More than minutes overdue

Due in next weeks or is overdue

Between 11-Jan-2012 and 30-Jan-2012

In range -3w 4d to 3w 4d 1

Update Close

7. Reorder your saved queues by clicking and dragging them to their new location.

You now have three new queues in your project! You will next learn how to add agents to your site so you can get your teams up and running with Jira Service Desk.

Next

Adding service desk agents

1. Setting up your service desk
2. Creating service desk request types
3. Making queues for your service desk teams
4. Adding service desk agents
5. Customize your service desk channels
6. Bring your service desk to the next level
7. Introduce customers to your service desk
8. Explore a

sample
project

There are two default project roles you can assign users to in Jira Service Desk:

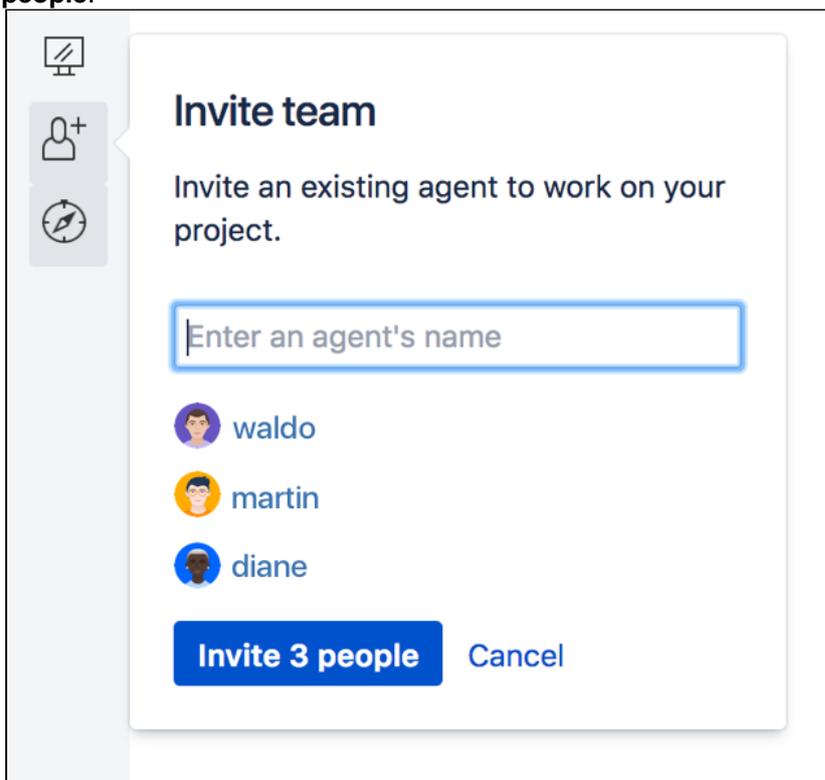
- Service Desk Customers who create requests via email or the customer portal
- Service Desk Team who view and respond to these requests

As the user who created this project, you have been automatically added to the Administrators project role.

Add your agents

Project administrators can add agents with existing user accounts to their project. If you are a project administrator, you will need to contact your site administrator to add user accounts for new agents. Make sure you're signed in as an administrator for this step, you can invite three new agents to your project - **Diane**, **Martin**, and **Waldo**:

1. In your project sidebar, select **Invite team**.
2. Enter the email addresses for your new agents and select **Invite 3 people**.



3. The agents are added to the Service Desk Team role in **Project settings > Users and roles**.

Assign issues to agents

Your agents will generally work out of queues that have issues automatically triaged into them. Let's test out manually assigning issues in case you ever come across a customer request that you want a certain agent or team to handle.

1. From the **Queues** tab, open one of your test requests by clicking the issue summary or issue key.

2. Select **Assign**.
3. Enter Waldo in the **Assignee** field and select **Assign**. When Waldo signs into Jira Service Desk, this issue appears in his queue.
4. Assign another test issue to Diane.

Add your customers

You don't need to add customers to your service desk during this tutorial, but let's check out where you would add them, so you're familiar with the steps:

1. From your project sidebar, select **Customers**.
2. Select **Add customers** in the top right corner and enter one or more email addresses.
3. When you select **Invite**, the customers receive an email invitation with a link to your customer portal, where they can complete the signup process.

Public customer signup

You can have your customers sign up for their own accounts (without an individual email invite) by enabling [public signup](#).

You're almost done! You have now added 3 agents to your service desk project and reviewed the process of assigning issues to these agents. You can now customize your customer portal and share it with the rest of your team.

Next

Customize your service desk channels

1. [Setting up your service desk](#)
2. [Creating service desk request types](#)
3. [Making queues for your service desk teams](#)
4. [Adding service desk agents](#)
5. [Customize your service desk channels](#)
6. [Bring your service desk to the next level](#)
7. [Introduce customers to your service desk](#)
8. [Explore a sample project](#)

Service desk customers can contact your team in two ways. They can log in and create a request via the customer portal or email a request to an email account that you have linked to your service desk project. Let's finish setting up the customer portal and add an email account so your customers can easily contact your team.

Customize the theme and branding of your customer portal

You can rename your customer portal and add a logo so customers can easily associate this service desk with your team and organization when they create requests.

1. In your service desk project, select **Project settings > Portal settings**.
2. Edit your customer portal name and introduction text by typing in the outlined fields. Save any edits by selecting



3. Add a customer portal logo by selecting **Use a custom logo**.
4. Save the following sample image, and then select **Choose logo** to upload it:



5. Select **Save logo**.

Link an email account

In addition to creating requests through the customer portal, customers can create requests and communicate with your team by email. Jira Service Desk Cloud projects come with a default email address, which you can use without having to manage an external email inbox. In this step, you'll link your service desk project to an existing email account used by your team.

1. In your service desk project, select **Project settings > Email requests**.
2. Email requests will be turned off by default, so turn them on now.
3. Select **Add email account** and fill in the requested details. If you use 2-step-verification for Gmail, be sure to generate an [application-specific password](#) when adding your email account details.
4. Once you have linked an email account, look out for the test email that will be sent to your email inbox and the corresponding request that will be created in your service desk project.

Tip:

If you use POP, make sure the email account you choose for this channel has an *empty inbox* so you do not lose any existing emails.

Publicize your service desk

Now that your service desk project is ready to receive requests, you can share the service desk email address (e.g. helpdesk@example.com) and a direct link to the customer portal with your customers.

You can give one or both of the following URLs to your customers.

- The URL to a specific service desk project customer portal. Give this URL to your customers if you've enabled public signup and want them to signup for accounts on their own. The signup link only appears on each individual portal.
- The URL to the global portal where your customers will see all the service desks they have access to. The URL is:
http://<computer_name_or_IP_address>:<HTTP_port_number>/Jira/servicedesk/customer/portals

To publicize your service desk, you can:

- Post a link on your intranet
- Add a hyperlinked button to your web portal
- Email your customers and let them know about the new, easy way to get help!

You've now finished setting up your service desk project! Continue on to learn more advanced tips that will help you better track your team's progress and serve your customers.

Next

Bring your service desk to the next level

1. [Setting up your service desk](#)
2. [Creating service desk request types](#)
3. [Making queues for your service desk teams](#)
4. [Adding service desk agents](#)
5. [Customize your service desk channels](#)
6. [Bring your service desk to the next level](#)
7. [Introduce customers to your service desk](#)
8. [Explore a sample project](#)

Now that you have your basic service desk up and running, you can learn about the following advanced features:

- [Serve your customers and your team better with SLAs](#)
- [Track your team's success with reports](#)
- [Solve requests faster with a knowledge base](#)

Serve your customers and your team better with SLAs

Service-level agreements (SLAs) help you communicate service agreements to your customers and keep track of your team's performance. An SLA consists of a time metric and a corresponding goal or target.

As the administrator, you can configure each SLA metric and goal using the Jira Service Desk SLA designer. SLA information will appear in the internal issue and your agents can also view SLA goals by going to **Report s > Workload** when they log in to your service desk project.

Let's have a quick look at where you can create a new SLA metric:

1. In your service desk project, select **Project settings > SLAs**.
2. Select **Create SLA** to create a new SLA metric for your service desk project.
3. For more information, check out [Setting up service level agreements \(SLAs\)](#).

Track your team's success with reports

Jira Service Desk lets you display selected SLA metrics and goals in interactive reports. Reports can be used

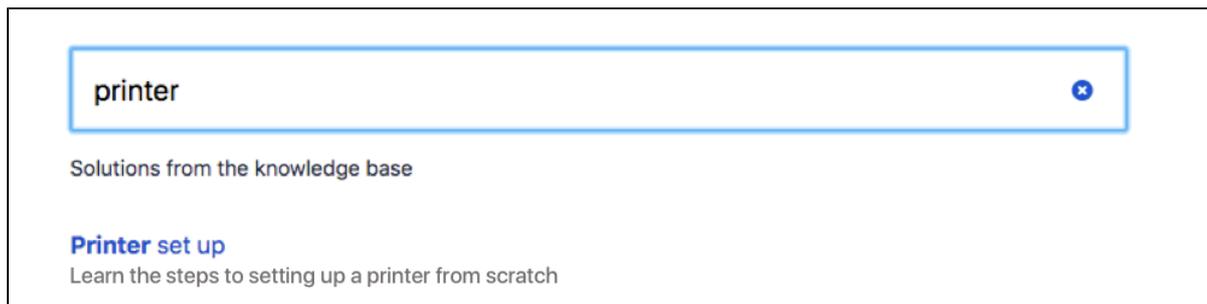
to help you visualize your team's performance so you can identify bottlenecks and optimize your team's workload. Your team of agents can then view the read-only versions of your reports to see how they are tracking towards their goals.

Let's now have a quick look at the **Reports** tab:

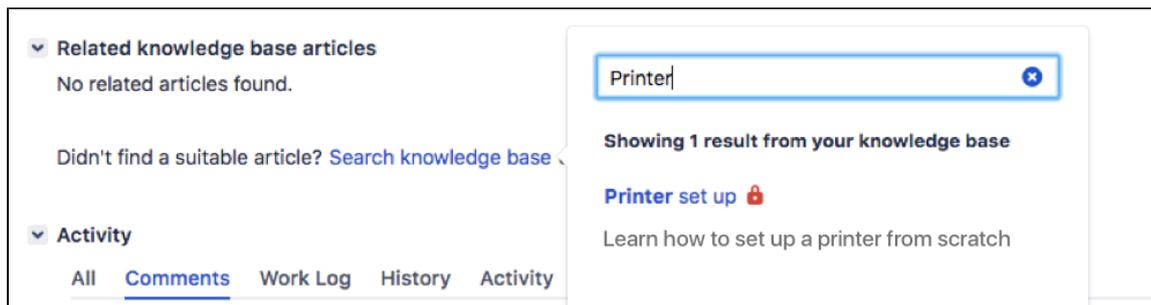
1. From your service desk project sidebar, select **Reports** to view the pre-configured reports in your project.
2. Select **+ New Report** to create a new report, or simply edit one of the pre-configured reports.
3. For more information, check out [Setting up service desk reports](#).

Solve requests faster with a knowledge base

You can deflect common requests and solve issues faster by linking a Confluence knowledge base to your service desk project. When you link a knowledge base to your project, customers can search for solutions in the portal and help center before they raise a request:



A knowledge base helps agents, too. When they work on issues, they can see related knowledge, search for solutions, and create new articles from issues:



Customers can use the articles to self-service problems, and agents can use them to solve requests faster. Everybody wins.

To learn more about linking a knowledge base to your service desk project, see [Set up a knowledge base for self-service](#).

You're almost done! We'll now review the ways customers can contact your team and be informed of updates to their requests.

Next

Introduce customers to your service desk

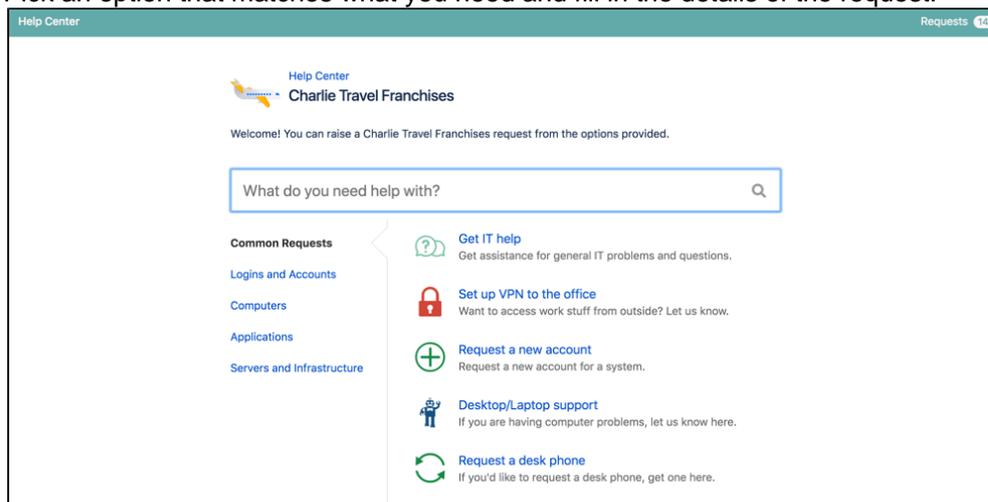
1. [Setting up your service desk](#)
2. [Creating service desk request types](#)
3. [Making queues for your service](#)

- desk teams
- 4. Adding service desk agents
- 5. Customize your service desk channels
- 6. Bring your service desk to the next level
- 7. Introduce customers to your service desk
- 8. Explore a sample project

Now that you have set up your project in a way that serves both your agents and your customers, it's time to show your customers how to start using Jira Service Desk.

Create requests through the customer portal

1. Visit the customer portal.
2. Pick an option that matches what you need and fill in the details of the request:



Create requests by email

Another way of creating requests is by sending emails to a linked service desk. Ask your service team if they are set up to receive email in their service desk project. If they are, simply email them a request directly and keep the conversation going from your inbox.

Create requests in multiple service desks

To send the same request to multiple teams, you have the following options:

- If all of the teams you want to contact have linked their service desk project to an email account, you can easily create the request by sending one email message to all linked service desk email accounts.
- If the teams you want to contact have not all linked their service desk project to an email account, you will need to create the request in each service desk one by one, either through their customer portal or sending emails.

Track and comment on requests

Use the customer portal to see all requests you have created, read comments from agents as they are updated, and check the status of a request. You can add comments and attachments to requests on the customer portal as well.

Another way of tracking requests is through email notifications. You receive email notifications when agents respond to your requests and when the request has been resolved. To add comments to requests, you can simply reply to the email notifications and your reply will be added as a comment to the request.

Congrats! You've completed the Getting started for service desk admins tutorial.

Want to learn more? Check out the home of Jira Service Desk documentation [here](#).

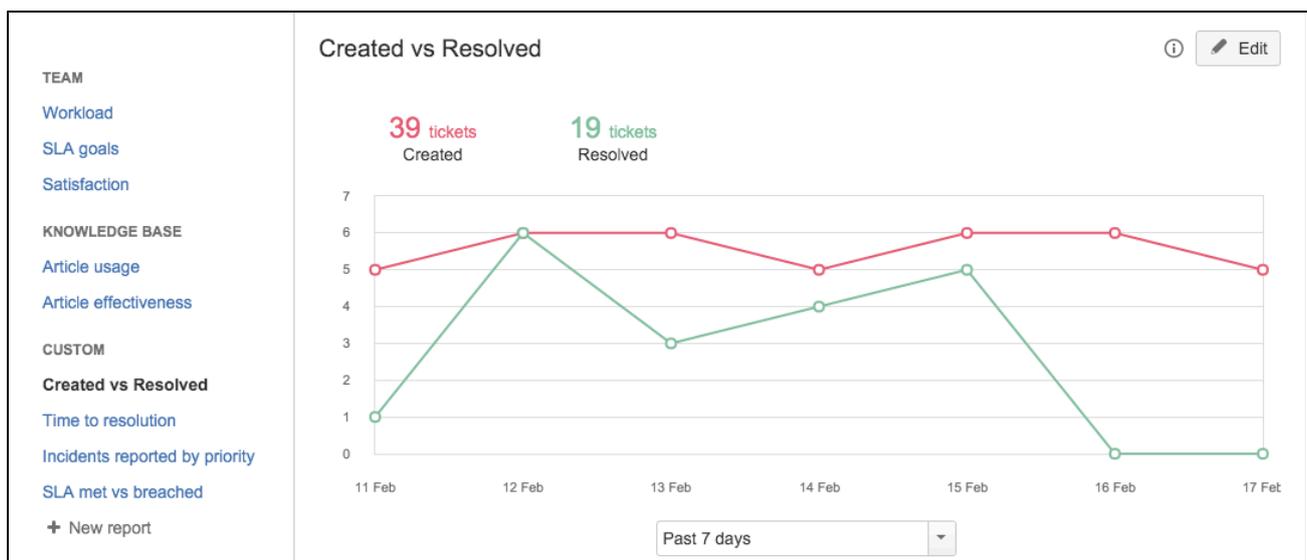
Explore a sample project

Jira Service Desk comes with sample data to help you explore and learn how to use key features.

When you create a sample project, it gets populated with issues that new team members can use to learn about concepts like queues, SLAs, and generate reports like the one below without fear of affecting any real work.

On this page:

- Create a sample project
- Learn about key features

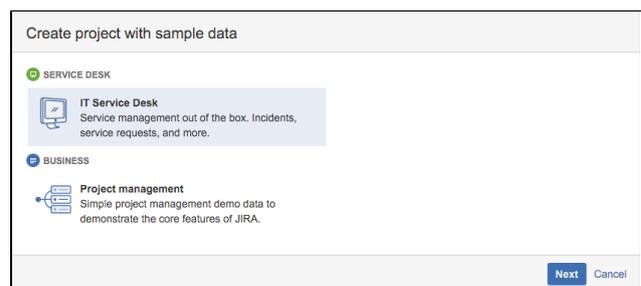


If you are a Jira Service Desk administrator, we suggest you create and explore a sample project to help you and any new team members explore how a service desk project works.

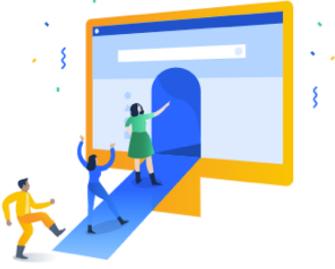
Create a sample project

You need to be a Jira Service Desk administrator.

1. Go to **Project > Create project**.
2. In the Create project screen, click **Create sample data**.
3. In the Create project with sample data screen, select **IT Service Desk** and click **Next**.
4. Enter a name for the sample project.
Tip: If you are creating the project for a specific user, name the project 'Sample - [name user]'. This will make it easier to find and delete later.
5. Click **Submit**.



Sample project access Depending on how user access is set up in your Jira Service Desk version, you may need to [give new users access to the project](#) as well.



Get on board your new service desk

- ✓ Be awesome
- ✓ Learn the key concepts of Jira Service Desk
- ✓ Create a request in the customer portal
- ✓ View issues in queues, where customer requests live
- ✓ See what you can do as a service desk admin

Check out tutorials and best practices in the [documentation](#)

Learn about key features

Here are a list of tasks that we highly recommend you have a go at:

- Explore the customer portal and see what your customers see
- Create a new request and assign it to yourself
- View the queue, edit an issue description, or add a label
- Comment on an issue
- Try out your email channel
- Play with reports

Finished playing with your sample project?
When a sample project has served its purpose, delete it from the project directory. You need to be a Jira Service Desk administrator to do this.

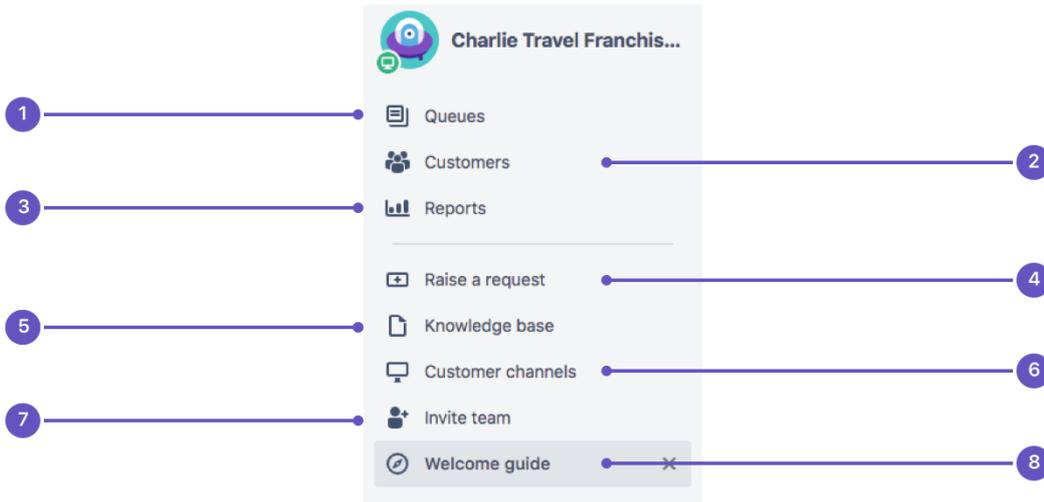
Getting started for service desk agents

On this page, we introduce you to your workspace and walk you through the process of responding to your customers' requests.

On this page
<ul style="list-style-type: none"> • Navigate your workspace • Work on customer issues • View and create knowledge

Navigate your workspace

Use the Jira Service Desk sidebar to navigate your workspace:

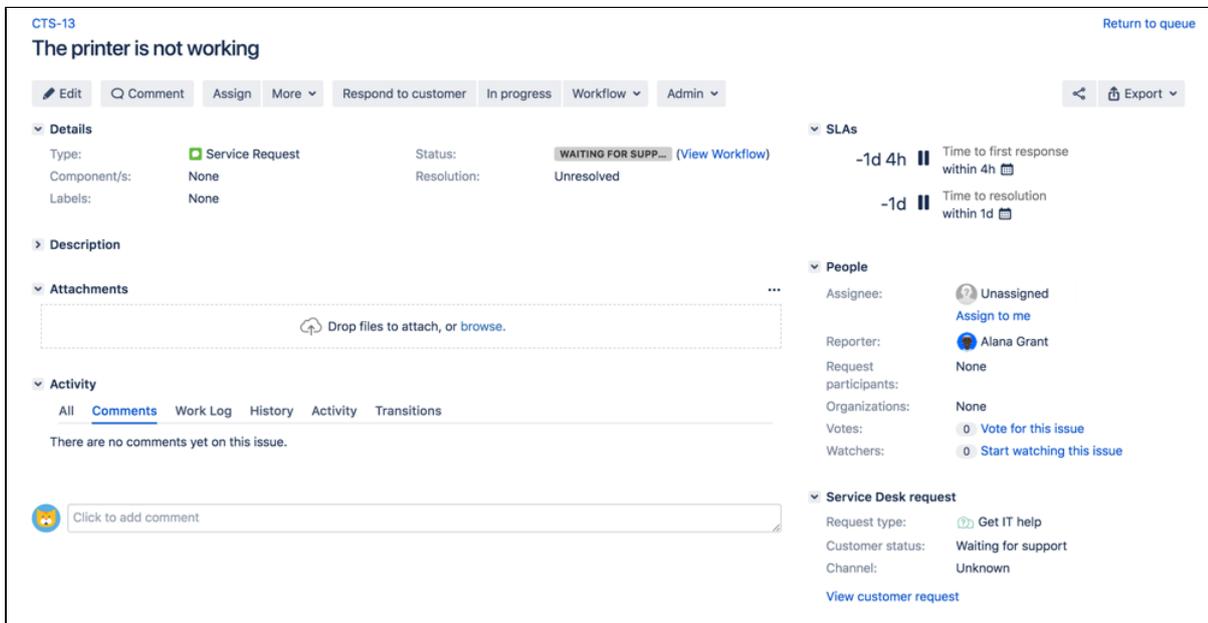


- 1. **Queues:** view issues that customers have submitted to your service desk.
- 2. **Customers:** view your customers and requests, and invite new customers to the service desk.
- 3. **Reports:** view reports about your team's SLA goals, knowledge base and workload.
- 4. **Raise a request:** raise a request on behalf of a customer.
- 5. **Knowledge base:** search your team's knowledge base or create an article.
- 6. **Customer channels:** view your service desk's customer portal and email address.
- 7. **Invite team:** invite an existing agent to work on your project.

Work on customer issues

When customers submit requests to your service desk, the issues are grouped into queues on the **Queues** page.

Click an issue's **Summary** or **Key** to view more information about an issue, or start working on an issue. The issue view looks like this:



From here, you can work with customers and your team to resolve requests. If your service desk is linked to a knowledge base, you can also view, share, and create knowledge from the issue.

Comment on an issue

When you click **Respond to customer**, add an attachment, or comment on an issue, you can share your response with the customer, or comment internally.

- If you share with the customer, the customer is notified and can see the comment or attachment when they view the request in the customer portal.
- If you make an internal comment, the customer isn't notified and can't view the comment or attachment on the customer portal. People who are watching the issue are notified, and your team can see the comment when they view the issue in your service desk.

Collaborate on an issue

You might want to share the issue with other customers who have a similar problem, or with other people on your team who can help you resolve the issue. Here are some ways you can involve other people in the issue:

- Click **Share**

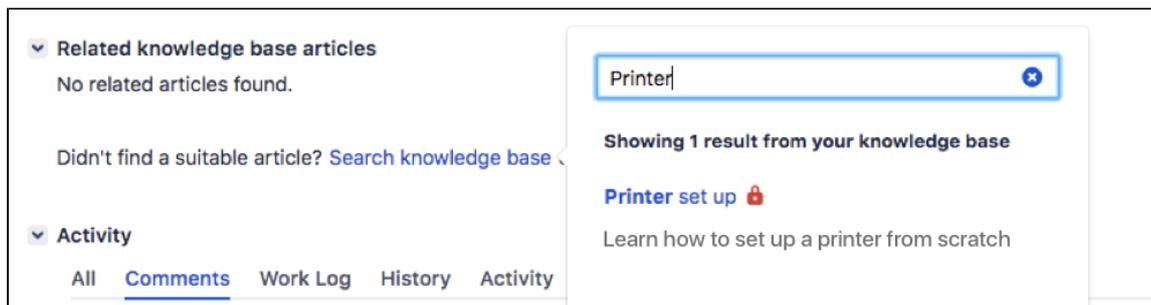


to email a link to the issue to other people on your team.

- Use @ mentions (@username) to mention a team member in a comment.
- Add watchers to involve other users from your Jira site. Someone leaves an internal comment or transitions an issue.
- Add request participants to share the issue with other customers or organizations. Request participants receive the same notifications as the reporter.

View and create knowledge

If your service desk has a linked knowledge base, you can use knowledge articles to solve issues faster:



For example, if you get a lot of requests about printing problems, you can write a step-by-step article that helps customers troubleshoot the problem themselves:

- When customers search for help from the customer portal, they can use articles to solve problems instead of raising requests.
- When you work on an issue, you can share the article with customers so they can try to troubleshoot on their own.

You can also write internal articles and reference them while you work on an issue. Articles that the reporter can't view are marked with a red padlock.

When you create a new article from an issue, you can choose a handy *how-to* or *troubleshooting* template to guide you. The issue summary and description become the article's title and body text.

Nice work! Want to learn more? Proceed to [Working on service desk projects](#) to learn more about what a service desk agent can do.

Administering service desk projects

Welcome to the source of truth for Jira Service Desk administrator knowledge and power.

This section is for Jira Service Desk project administrators. You're in the right place if you're the one who adds agents and sets up your service desk project. If you're new to Jira Service Desk, check out our [getting started guides](#).

Search the topics in Administering service desk projects:

Managing access to your service desk

Go to **Project settings > Customer permissions** to choose who can raise requests in your service desk and who your customers can share requests with.

On this page:

- Choose who can raise requests
- Choose who customers can share requests with
- Which settings are best for my team?

Choose who can raise requests

People need to be customers to raise requests in your service desk. You can let your team control who becomes a customer, or let customers create their own accounts in your service desk.

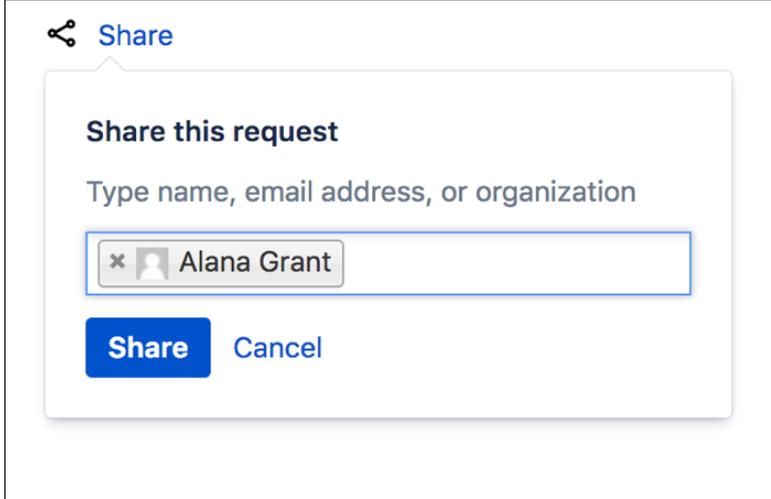
Who can raise requests	Description
Customers who are added to the project	Your team adds customers to the project via the Customers page, or by raising requests on their behalf.
Customers who have an account on this Jira site	People with accounts on your Jira site are automatically added to the Customers list and can raise requests.
Anyone can email the service desk or raise a request in the portal	<ul style="list-style-type: none"> • New customers can create their own accounts in your service desk via the customer portal. • An email request automatically creates an account for the sender. • If you allow customers to share requests, then people they share with also become customers and can raise requests. • A honeypot technique is enabled to help prevent spambots from creating accounts through the customer portal. <p>If this option is disabled, then your Jira administrator has not turned on public signup for service desks on this Jira site. Learn more</p>

Choose who customers can share requests with

You can allow customers to share requests with their organizations, anyone in the service desk, or people who aren't customers yet. The people customers share with become participants in the request. Request participants can comment on and share requests, and receive the same notifications from Jira Service Desk as the reporter. [Learn more about request participants.](#)

The following table describes the ways customers can share requests:

Who customers can share with	Description

<p>Other customers in their organization</p>	<ul style="list-style-type: none"> • Customers can share requests with their organization, or raise a private request. • Customers can search their organization for people to share with: <div data-bbox="491 271 1262 770" style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;">  </div> <ul style="list-style-type: none"> • Customers who aren't in an organization can't share requests.
<p>Any customer, by typing an email address</p>	<p>Other customers in their organization, plus:</p> <ul style="list-style-type: none"> • Customers can share requests with anyone in the service desk, but only if they know their email address. • If anyone can email the service desk or raise a request in the portal, then customers can share requests with people who aren't customers yet. The people they share requests with become customers.
<p>Any customer or organization, by searching in this project</p>	<ul style="list-style-type: none"> • Customers can share their requests with anyone in the project. They can also search the service desk for people to share with. • If anyone can email the service desk or raise a request in the portal, then customers can share the request with people who aren't customers yet. The people they share requests with become customers. <div data-bbox="445 1252 1404 1359" style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p>If your service desk uses user picker custom fields, such as the Approvers field, choose this setting to make sure customers can select users.</p> </div>

Which settings are best for my team?

Not sure how to set up permissions for your team? Here are some suggestions for how to make customer permissions work for you:

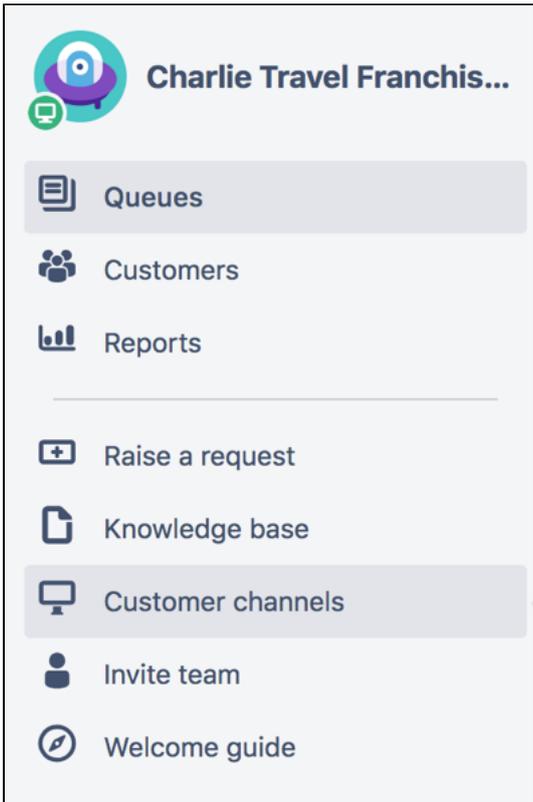
If you're like this	Who can raise requests	Who customers can share with
<p>You have a service desk that handles contractors' leave requests. Only contractors can use the service desk, and you don't want non-contractors don't get confused about where to request leave.</p>	<p>Customers who are added to the project</p>	<p>Other customers in their organization</p>
<p>Your company has an IT service desk, and you want all employees to be able to create their own accounts and email requests.</p>	<p>Customers who have an account on this Jira site</p>	<p>Any customer or organization, by searching in this project</p>

Your team provides software support for individuals. For example, if your company makes a free SAAS application that individuals use to manage finances, you can let your customers email bugs and questions to your service desk email channel.	Anyone can email the service desk or raise a request in the portal	Any customer, by typing an email address
--	--	--

Configuring the customer portal

Your service desk project comes with a customizable customer-facing site called the *customer portal* that customers can use to raise and track requests. If you link your project to a Confluence knowledge base space, customers can also self-service issues by searching for relevant articles.

To access your customer portal, click **Customer channels** in the project sidebar:



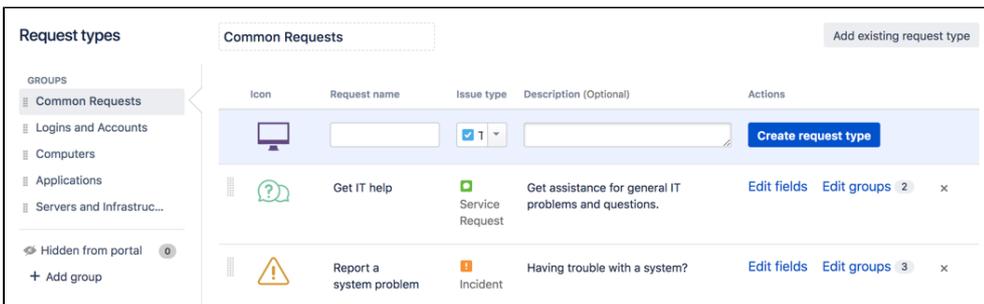
On this page:

- Set up request types
- Brand your portal
- Add transitions
- Manage access to your portal
- View all portals in your Help Center

You need to be an administrator for your project to make the following changes.

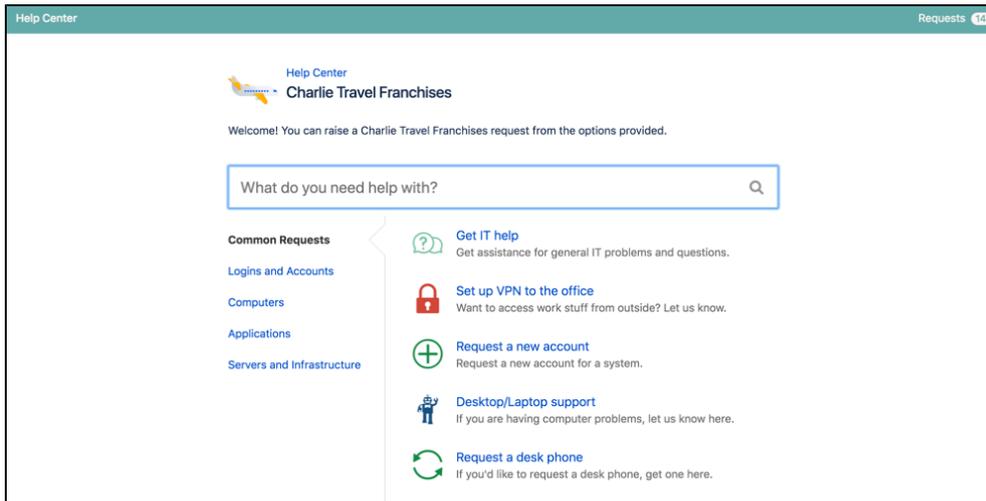
Set up request types

You can customize the types of requests that customers raise from the portal. To create and manage request types, visit **Project settings > Request types**.



Jira Service Desk includes several request types that address common IT help scenarios. The request types are organized into groups to help customers find what they need on the portal.

For example, you can add a "Common Requests" group to help customers address issues like system problems or IT support.



To learn more about customizing request types, check out [Setting up request types](#).

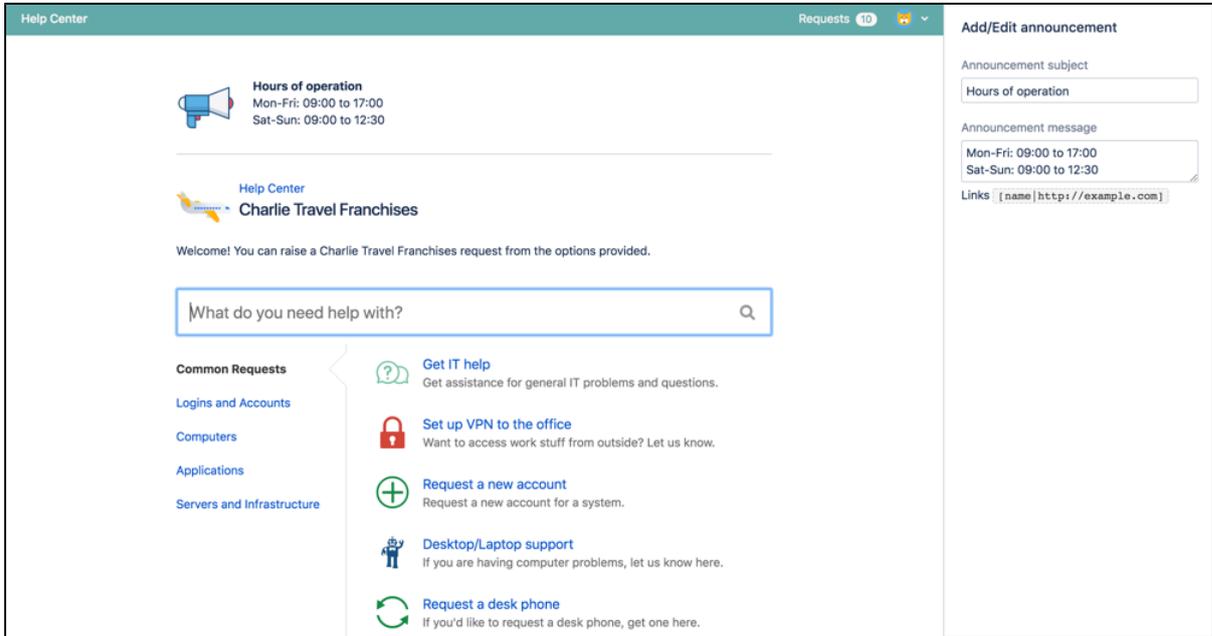
Brand your portal

You can customize your customer portal to reflect your team and company's brand with the following two steps:

1. In **Project settings > Portal settings**, add a portal logo and a short description:

Your logo also appears in notifications sent by your service desk. [Read more about customer notifications](#).

2. Follow the link **View and change** your Customer Portal announcement to add a message unique to your project portal.

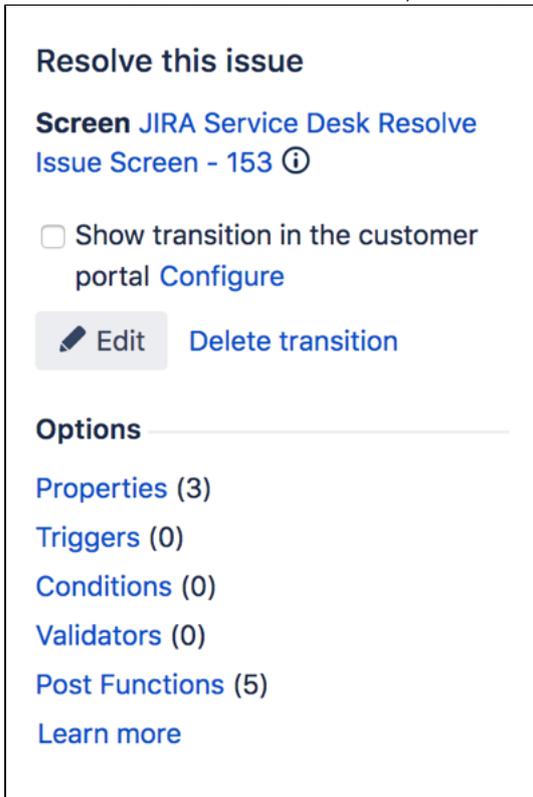


Add transitions

You can show transitions on the customer portal so that customers can transition requests. For example, say an agent shares a knowledge base article with a customer. If the article solves the customer's problem, then the customer can resolve the request.

To add a transition to the portal, you edit an existing transition in the workflow.

1. In your service desk project, click **Project settings > Workflows**.
2. Click  next to the workflow that contains the transition you want to add to the portal.
3. Click **Diagram** to open the diagram view.
4. Click the transition in the workflow, and then select *Show transition on the customer portal*.



Customer transitions behave slightly differently than other workflow transitions:

- Screens don't display on the customer portal. When you add a transition to the portal, you can set a resolution for requests that customers transition.
- When an issue is transitioned from the portal, it bypasses any validators that are defined for the transition.

If it seems like the portal transition isn't working properly, make sure there isn't an automation rule in conflict with the transition.

To learn more about workflows and transitions, see the [advanced workflow](#) configuration page.

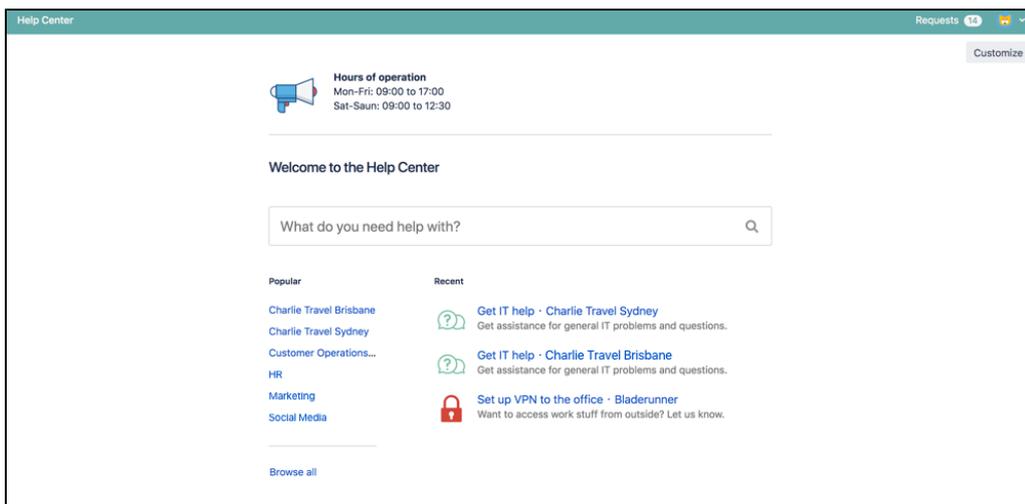
Manage access to your portal

You can allow customers to create their own accounts in the portal, or limit access to customers that your team adds. To learn more about different access options, see [Managing access to your service desk](#).

The customer portal integrates with [Atlassian Crowd](#), Atlassian's single sign-on (SSO) framework. For information about integrating with third-party SSO providers, see [this page](#).

View all portals in your Help Center

The Help Center shows all of the customer portals in your Jira site:



Customers can access the Help Center at the following URL:

```
http://<computer_name_or_IP_address>:<HTTP_port_number>/Jira/servicedesk/customer/portals
```

From the Help Center, customers can raise requests in any of the portals they have access to. They can also view and search their requests in **Requests**.

Receiving requests by email

If your customers prefer to raise requests from the comfort of their email inboxes, you can set up an email address to receive requests in your service desk. Emailed requests are added to your queues, so your team can focus on customers without worrying about missing requests or multiple inboxes.

Here's how it works:

1. A customer emails a request to your service desk email address. The request becomes an issue in your service desk and is added to a queue.
2. An agent comments on the issue.
3. The customer receives an email notification that contains the agent's

- comment.
- The customer replies to the email notification, and the reply displays as a comment on the issue in the service desk.

On this page:

- Before you start
- Add an email account
- Choose a request type
- Verify your linked email account
- Prepare customers for email greatness
- Email channel notes

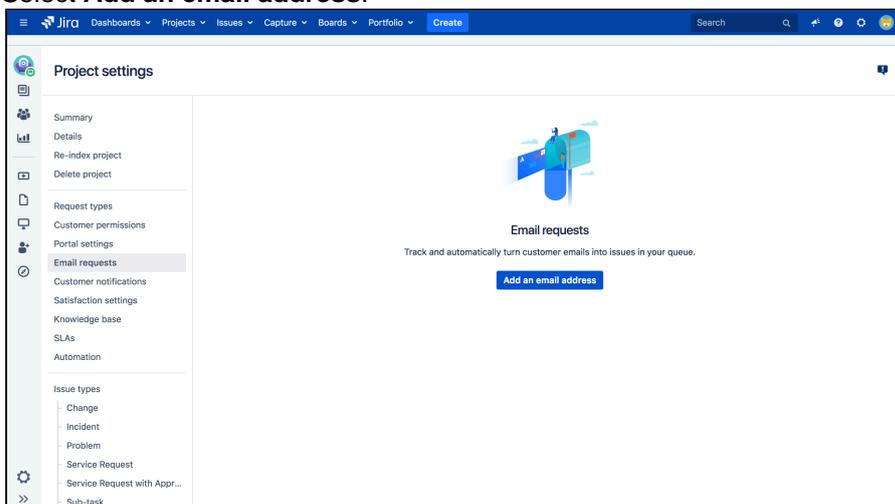
Before you start

- Make sure you have **Jira administrator** and **project administrator** permission.
- Enable [public signup](#), or [add customers](#) to your service desk project to ensure that you receive new requests.
- Set up a suitable [request type](#) with **Summary** and **Description** as required visible fields. Any other fields must be optional.
- Know which emails from your mail client will be [processed](#).

Customer requests and comments are processed differently than [Jira mail](#). Issues created via Jira email handlers don't show up as service desk customer requests. For this reason, we don't recommend using a Jira mail handler for service desk projects.

Add an email account

- Open your service desk project and go to **Project settings > Email requests**.
- Select **Add an email address**.



- Set up your email channel by choosing your email service provider.
- Enter an email address and password, then select **Next**.

If your Gmail or Yahoo! account uses two-step verification, you'll need to set up an [application-specific password](#).

Choose a request type

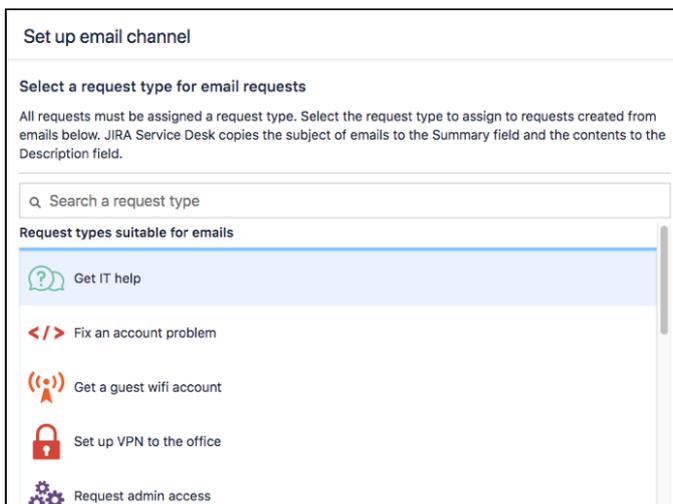
When a customer emails your service desk, a corresponding request is created with the following two fields:

- Summary (from the email subject line)
- Description (from the email content)

To use the email channel, you therefore need to have at least one request type in your project with Summary and Description fields – we call these types of requests "suitable for emails".

Associating email requests with a suitable service desk request type ensures that the emails are successfully filtered into your service desk queues.

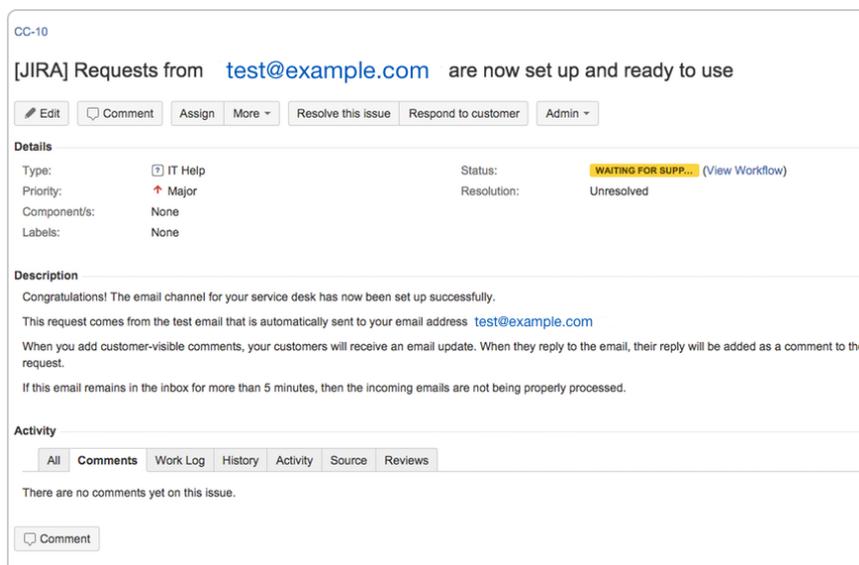
In this example, we have several request types suitable for email:



Select a suitable request type for your email channel and select **Done**.

Verify your linked email account

Once you have chosen a suitable request type, Jira Service Desk sends a test email and creates a corresponding test request in your service desk project. Head on over to the **Queues** tab to find the new request:



New messages sent to your linked email account appear as service desk requests in your project. For more information about which emails are processed by Jira Service Desk, expand the option that applies to you:

Emails using POP

▼ [Jira Service Desk looks for messages in your inbox that have...](#)

1. The "Deleted" flag set to false, and
2. Were received after your email account and service desk project were linked.

To link your email account with a service desk using POP, make sure that your email inbox is empty by moving the existing messages to another folder, archiving them, or deleting them. Starting with an empty inbox ensures that you do not lose emails unintentionally, as POP emails are deleted after they are processed by Jira Service Desk.

Emails using IMAP

▼ [Jira Service Desk looks for messages in your inbox that have...](#)

1. The "Deleted" and "Seen" flags set to false, and
2. Were received after your email account and service desk project were linked.

If you use IMAP, emails are marked as read (not deleted) after they are processed by Jira Service Desk. If you want existing messages to be pulled in by Jira Service Desk, you can move them back to your inbox and mark them as unread after the connection has been established.

Prepare customers for email greatness

Before you share your email address with your customers, you'll want to double check [the customer permissions for your service desk](#):

- If anyone can email your service desk, then you're good to go! People who email your service desk automatically become customers. On the **Email requests** page, you can further control what happens to email addresses that are subsequently added to either the To or CC field of an email associated with a service desk request.
- If customers have to be manually added to your project, then make sure your team [creates new customer accounts](#) for them. If people who aren't customers email your service desk, their requests won't be processed.

On the **Email request** page, you can also choose whether to allow emails from addresses which are currently not registered as customers of your service desk to be added as comments to the associated request. This is useful if a customer forwards an email to an external person regarding a request, and the external person responds with information relating to the request. The information will be recorded as a comment on the request. Note that the service desk will **not** send notifications to the external person regarding the request to preserve privacy.

Email channel notes

- You can only link one email account to your service desk project. If you use more than one email address to interact with your customers, you might be able to set up forwarding rules or aliases to receive requests in the email linked to your service desk project. You will need to configure any forwarding rules or aliases in your email client.
- If you are a Jira administrator, you can refer to [Managing the email channel](#) to learn more about global mail settings.
- If you encountered any issues during the email setup process, check out some common errors and resolutions [here](#).

Managing the email channel

After you [set up your email channel](#), you can change how emails are parsed, if customers are created directly from emails, or view logs and check the email connection for any email channel on the site.

On this page:

- [Manage the email channel for multiple service desk projects](#)
- [Manage global mail settings](#)

Manage the email channel for multiple service desk projects

To manage email channels:

1. Choose  > **Applications**.
2. Under **Jira Service Desk**, select **Email requests**.

Strip quotes

This setting controls whether emailed comments contain the entire email thread, or just the most recent reply.

HTML email parsing

Choose whether HTML emails display in wiki markup or plain text.

Public email comments

This setting controls whether replies are added as comments to existing requests.

Customer account creation

Choose whether new customers are created from the address list of email messages.

View status and logs

Under **Email addresses**, you can view the status of each email channel on your service desk site, view logs, or delete a channel. Note that logging information older than 6 months is deleted daily.

Manage global mail settings

To manage global mail settings:

1. Choose  > **System**.
2. Under **Mail**, select **Global Mail Settings**.

There are two global mail settings – **Email puller** and **Email processor** – that are used by Jira Service Desk only. They don't impact any other email settings you have set up for Jira.

Email puller connects to your mail servers every minute and pulls the email data into the database. Emails with attachments larger than 25MB will not be pulled. Email processor filters the emails (for example, to remove auto-replies and spam) using information stored in the database.

Troubleshooting issues with the email channel

This page contains information about the errors you might run into when setting up the email channel for your service desk. See the [Resolving errors](#) section below for more information about generating an application-specific password (for example, if 2-factor authentication is enabled for your email account), resolving email connection issues, and ensuring that customers who raise requests by email are correctly notified.

Checking the connection

To troubleshoot email channel issues, the first thing to do is to check the connection between Jira Service Desk and your email account. You will see error messages that show you why the email channel does not work for your service desk.

To check the connection:

1. Choose



> **Applications**. Scroll down to the **Jira Service Desk** section and choose **Email requests**.

2. Select **Test**.

Resolving errors

The following table describes the common errors and provides information about how to resolve them when available.

Symptom	Description and resolution
<p>Setting up the email channel</p> <p>Error message:</p> <p>The email address you entered is currently used by another project's email channel. Please choose another email address. Check out our troubleshooting docs for help resolving the issue.</p>	<p>You can only connect one email address to one service desk project in your Jira Service Desk Cloud site or Server instance. If you try to use the same email address to set up an email channel in another service desk project, you'll receive this error message.</p> <p>You can also receive this error message if you are trying to use multiple email aliases that point to the same email account for multiple service desk email channels.</p> <p>To resolve this:</p> <ul style="list-style-type: none"> • Choose another email address for the email channel you're setting up or for the one that already exists.

<p>Setting up a Gmail account</p> <p>Error message:</p> <p>Unfortunately Jira couldn't connect to the mail server. Here is what the mail server said: "[ALERT] Please log in via your web browser: http://support.google.com/mail/accounts/bin/answer.py?answer=78754 (Failure)</p>	<p>If you have two-factor authentication enabled for your Gmail account, you'll likely receive this error when entering your login details. Alternatively, Jira Service Desk checks email accounts every minute, causing Gmail to suspect inappropriate usage of this account and lock it for security reasons.</p> <p>To resolve this:</p> <ul style="list-style-type: none"> • Create an application-specific password for Jira Service Desk in your Gmail account settings. Details can be found here.
<p>Setting up a Yahoo! account</p> <p>Error message:</p> <p>Unfortunately Jira couldn't connect to the mail server. Here is what the mail server said: "[AUTHENTICATIONFAILED] (#MBR1240) Please verify your account by going to https://login.yahoo.com"</p>	<p>If you have two-factor authentication enabled for your Yahoo! account, you'll likely receive this error when entering your login details.</p> <p>To resolve this:</p> <ul style="list-style-type: none"> • Create an application-specific password for Jira Service Desk in your Yahoo! account settings. Details can be found here.
<p>Microsoft Outlook, POP3</p> <p>Error message:</p> <p>Unfortunately Jira Service Desk couldn't connect to the mail server. Here is what the mail server said: "STAT command failed: Exceeded the login limit for a 15 minute period. Reduce the frequency of requests to the POP3 server."</p>	<p>Jira Service Desk checks email accounts every minute. Microsoft Outlook might suspect inappropriate usage of this account and lock it for security reasons.</p> <p>To resolve this:</p> <ul style="list-style-type: none"> • Use IMAP.

Gmail accounts, POP3

Requests are created from archived messages.

When Jira Service Desk checks your email accounts for new messages, it the **inbox folder**.

Gmail uses labels to classify messages into categories and has these folders: **ox**, **Sent Mail** and (or **Trash**). This means that the archived messages are still considered in the inbox folder. With POP3, Jira Service Desk is not able to identify archived messages by label and therefore still brings them in as requests.

To resolve this:

- Use IMAP.

Customers send emails to create requests, but no requests are created and customers do not receive any notifications.

This problem could be due to one or more of the following causes:

- The connection to the email account failed.
- You do not have public signup configured and the customer does not have a user account in the system. Every customer must have an account before they can create requests in a *protected* service desk.
- The default request type for the email channel is unsuitable for the email channel.

▼ [Learn more](#)

A suitable request type for the email channel must be defined in the **Summary** and the **Description** field as visible fields. Any other fields must be optional ones.

To troubleshoot this issue and resolve it:

1. Check the connection as described previously on this page.
2. Check if user accounts exist for your customer. If not, create user accounts for your customers. For instructions, see [Setting up service desk users](#). You can also [configure public signup](#).

Warning message:**No suitable request type for the email channel**

You will select the default request type assigned to requests created from the email channel during the setup. However no existing request types are suitable for email requests. A suitable request type must have both the Summary and Description fields as visible fields, and all the other visible fields, if any, must be optional. If you want to enable the email channel, add a new request type that meets the criteria or modify an existing one on the Request types page.

This message appears on the **Er requests** page and prevents you from turning on the email channel.

To resolve this:

1. In your service desk project, select **Project Settings > Request type**
2. Add a new request type (or choose an existing one).
3. Select **Edit fields**
4. Make sure both the Summary and Description fields are added and marked as **Visible = Yes**. You can also add an additional Attachment field with **Required**.
5. Save the request type and head back to **Project settings > Error requests**.

Setting up service desk users

When you set up your service desk project, you add users to the project so that your team can start receiving and resolving requests. Your service desk has the following users:

- **Project administrators** set up the service desk project and users
- **Agents** work on customer requests and add customers to the project.
- **Customers** raise requests in your service desk.
- **Organizations** are groups of customers that are shared across projects.

By default, you need different permissions to manage different types of users in the project:

- **Jira administrators** can manage users and licenses across multiple projects.
- **Project administrators** can add agents from other projects. They can also manage customers and organizations.
- **Agents** can manage customers and organizations.

On this page:

- [Manage agents](#)
- [Manage customers](#)
- [Manage organizations](#)
- [Involve Jira Software or Jira Core users](#)
- [Learn more about managing users](#)

Manage agents**Add agents to a project**

By default, project administrators can add agents from other projects to the project. Jira administrators can add anyone to the project.

To add agents, click **Invite team** in the project sidebar. The agents are emailed a link to the service desk project, and are added to the **Service Desk Team** project role in **Project settings > Users and roles**.

When a Jira administrator adds a new agent to the project, then the agent is also assigned a Jira Service Desk license and added to the **service-desk-users** license group in



> User Management.

If your agents need to collaborate with Jira Core or Jira Software users to resolve an issue, you can grant the Jira users limited access to your service desk project. [Learn more.](#)

Remove agents from a project

1. In your project, go to **Project settings > Users and roles**.
2. Hover over the user or group you'd like to remove from the Service Desk Team project role, and then select



Unlicense agents

Jira administrators can remove an agent's license.

1. Go to  **> User management.**
2. Select the user.
3. In **Application access**, clear the **Jira Service Desk** box.

The agent's license is released, and they are removed from the **Jira-service-desk** user group.

Manage customers

By default, both agents and project administrators can add customers to projects, but only project administrators can remove customers from projects.

Add customers to a project

Add customers to your project via **Customers > Add customers**. Customers are automatically added to the **Customers** list if your service desk is open to users with Jira accounts or allows customers to create their own accounts.

Customers who do not have Jira accounts are added to the **Service Desk Customers** project role in **Project settings > Users and roles**, and are given restricted access to the Customer portal only (not Jira).

For more information about how people become customers, see [Managing access to your service desk](#).

Remove customers from a project

1. Go to **Project settings > Users and roles**.
2. Hover over the user or group you would like to remove, and then select



If the customers have Jira accounts or created their own accounts, then a Jira administrator needs to deactivate them in



> User management.

Manage organizations

Organizations are groups of customers that can be used in multiple projects. When you add an organization to a project, its members can raise requests in the project and share them with the organization. They're also notified about the organization's requests, and can view and search them on the **My Requests** page in the portal. [Learn more about how customers share requests with organizations.](#)

By default, you need the **Service Desk Team** role for a project to manage organizations in it. However, a Jira admin can restrict organization management to Jira admins by turning off the **Organization management** setting in



> Applications > Jira Service Desk Configuration.

Organizations are managed from a project's **Customers** list.

Add organizations

Add organizations to a project via **Customers > Add organizations**. You can add a new organization, or add an existing organization. The organizations that you add display on the **Customers** list. Select the organization to view and manage its members.

Add customers to an organization

You can add customers to an organization from the **Customers** list, or from an organization. Adding customers to an organization that isn't in the project yet adds the organization to the project.

If the customers are new to the Jira site, they are given restricted access to the Customer portal only (not Jira). And because the organization can be used in multiple projects, the customers are not added to the **Service Desk Customers** project role. However, they can still raise requests in all projects that use the organization.

Remove customers from an organization

1. Select an organization in the **Customers** list.
2. Find the customer you want to remove, and then click **X** next to their name.

When you remove customers from an organization, they lose access to projects that use the organization unless they have the **Customers** role for the projects or have access through another organization.

Remove an organization from a project

1. Select an organization in the **Customers** list.
2. Select **Remove from project**.

When you remove an organization from a project, its members lose access to the project unless they have the **Customers** role for the project or have access through another organization.

Delete an organization from a site

You must be a Jira admin to delete an organization from a Jira site.

1. Select an organization in the **Customers** list.
2. Select **Remove from project > Delete organization**.

Deleting an organization does not delete the customers who are in it. The customers still exist on the Jira site, and have access to any projects for which they have the **Customers** role or access via another organization.

Involve Jira Software or Jira Core users

You can give users with Jira Software or Jira Core licenses permission to view and comment on service desk issues without a Jira Service Desk license.

To involve Jira application users, go to **Project settings > Users and roles**, and then add the users to the **Service Desk Team** role.

These users can	These users can't
<ul style="list-style-type: none"> View issues, comments and attachments Add and delete their own attachments and internal comments Watch and vote for issues 	<ul style="list-style-type: none"> Leave comments for customers View queues, the customer list, or reports Transition service desk issues Log work on a service desk issue Be assigned to a service desk issue

For example, Martin, an IT service desk team agent, links an incident ticket in a service desk project to an underlying network problem ticket in a Jira Software project. Andrew, a Jira Software developer on the network operations team, assigns this network issue to himself and starts working on it. After fixing the problem, Andrew opens the linked service desk incident ticket and leaves an internal comment asking Martin to try the network connection again. After receiving the internal comment, Martin verifies the network connection and tells the customer that the problem is resolved.

Learn more about managing users

Check out the following documentation to learn more about managing users and permissions:

Documentation	Details
User management	Add and remove users, manage users with groups, and manage access to Jira applications.
Managing project roles	Add and remove project roles and manage project role membership.
Enabling public signup	Allow customers to create their own accounts by signing up on the customer portal or emailing the email channel
Configuring permissions	Learn how global permissions affect licensing, how project permissions are associated with a project role, and how to customize the default service desk project permission scheme.

Managing project role memberships

You can use project roles to easily associate users and groups with a particular project. For example, you may want to send notifications to a specific set of users associated with your project, and by adding them all to a project role, you can then use that project role to control who receives the notifications.

You can also use project roles to restrict how much access certain users or groups have. Unlike groups, which have the same membership throughout your application, project roles have specific members for each project.

This page contains instructions for managing membership of *existing project roles*. For information on creating and using project roles, see [Managing project roles](#).

Viewing and editing project role members

1. Log in as a project administrator and open your project.
2. Select **Project settings** () > **Users and roles**.

3. You'll see all users and groups associated with each project role.
4. To add users or groups to a project role, select **Add users to a role** in the top right corner. Enter the users or groups and select the project role you wish to add them to.
5. To remove a user or group from a project role, hover over the user or group row, and select



Since group membership can only be edited by users with the **Jira Administrator** global permission, project administrators may therefore prefer to assign users, rather than groups, to their project roles.

Setting up queues for your team

Make sure that your team is working on the right requests at the right time with easily configurable service desk queues. A queue is a filtered set of issues that are displayed to your team. Use your project's default queues or create custom queues to save time triaging requests, and to give your agents more visibility of the number and type of incoming customer requests they need to work on.

You need to be an administrator for your project to set up queues.

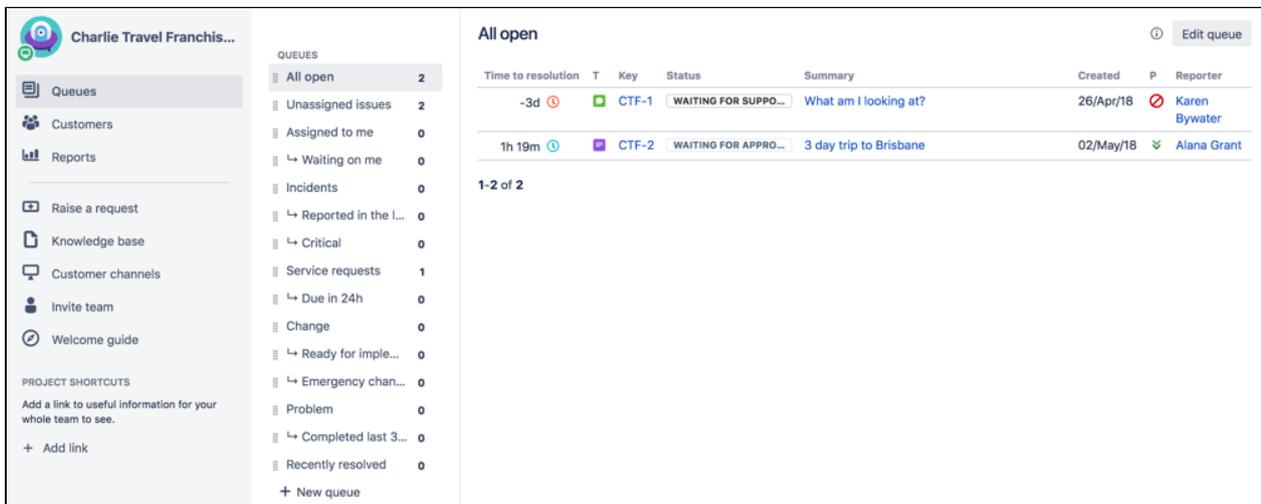
On this page:

- [Creating new queues](#)
- [Managing queues](#)
- [How your team uses queues](#)

You can set up queues on the aptly named **Queues** (



) tab in your service desk project:



Creating new queues

When creating a new queue, you can select the queue name, the issues that will be filtered into this queue, and the columns that appear in the queue to make life easier for your service team. Here's how you create a new queue:

1. From your service desk project sidebar, select **Queues > New queue**.
2. Name your queue using language your team will understand (for example, "Due this week").
3. Select which issues will show up in this queue using the dropdown options in the Basic search view:

You can also select the advanced search view to enter a [Jira Query Language \(JQL\)](#) statement.

4. Add or remove columns to control what issue information, such as the issue key and issue creation date, is displayed in your queue.
5. Select **Create**. If you have existing issues in your project that fit the criteria selected in "Issues to show", these issues will now appear in your new queue.

Managing queues

You can reorder or delete queues at any time by hovering over the Queues sidebar and selecting **Manage**. In the Manage queues dialog that appears, you can see the number of issues in each queue, and drag and drop queues to reorder them.

You can edit existing queues by selecting the queue you wish to configure and selecting **Edit queue** in the top right corner. You can edit the queue name, the issues shown, the columns, and column order. Note that you will see a live preview of the updated issues that appear in this queue as you configure it.

If you're using filters in your queues, make sure these filters include all the priorities defined in the associated priority scheme. See [Associating priorities with projects](#) for more details.

How your team uses queues

Queues give your agents a single view of the work that needs to be done across their team. Agents can view all of the queues in your service desk project; however, they cannot create or edit queues.

Automating your service desk

Create automation rules to perform actions in your service desk based on specific triggers and conditions. For example, you can set an automation rule that alerts an agent when a high-priority issue is created. Or, service desk can reopen an issue if your customer comments on it after its been resolved.

On this page:

- [Set up a preset automation rule](#)
- [Preset automation rules](#)
- [Edit preset automation rules](#)
- [Create a custom automation rule](#)
- [Rule options](#)
- [Disable an automation rule](#)

Set up a preset automation rule

To set up an automation rule:

1. In your service desk project settings, click **Automation** and select **Add rule**.
2. Select a preset rule from the list (see the table below for the available options) and then select **Next**. The rule configuration screen appears.
3. Edit the rule name and description as needed. The rule name appears on the main automation settings page, so changing the name helps you more easily reference what each rule does.
4. Edit and update any fields that appear in red text in the rule's WHEN, IF, and THEN conditions.
5. Select **Save** and you're done.

If your automation rules involve priorities, and you're using filters in your service desk, make sure these filters include all the priorities defined in the associated priority scheme. See [Associating priorities with projects](#) for more details.

Preset automation rules

Your service desk project comes with preset rules that you can use to set up automation. Here are the preset rules that come out-of-the-box to help your team and your customers:

Rule template	What automation does
Transition on comment	Updates the status of an issue after someone comments to: <ul style="list-style-type: none"> • Waiting on Support when a customer comments • Waiting on Customer when your team comments
Reopen on customer comment	Reopens a closed issue when a customer comments
Be aware of urgent issues	Alerts a member of your team via an @mention when a customer submits an urgent request
Keep on top of SLAs	Alerts your team lead via an @mention when a serious issue is about to breach one of your SLAs

Set customer expectations	Lets your customers know when to expect a response from your team based on the priority of their ticket by adding a pre-populated comment to the issue
Update when a linked issue changes	Comment internally on an issue about a change of status for a linked issue
Triage requests sent by email	Update issues received by email with the correct request type, based on keywords present in the request summary or description

Edit preset automation rules

If you want to edit a preset rule, select **Edit** next to the rule in your automation list to see how it's configured.

To edit a rule, select the WHEN, IF, or THEN fields to change your rule's trigger, conditions or resulting actions. Use **Tips for customizing this rule** for suggestions on what to enter in these fields.

If you make changes to the rule, be sure to click **Save** to confirm your edits.

Create a custom automation rule

To create a custom automation rule:

1. In your service desk project settings, click **Automation** and select **Add rule**.
2. Select **Custom rule** from the list and then select **Next**. The rule configuration screen appears.
3. Configure your rule by selecting and defining WHEN, IF, and THEN fields. See the table below for the available options.
4. Select **Save** and you're all set.

Here are the rule sets that are allowed in the automation engine:

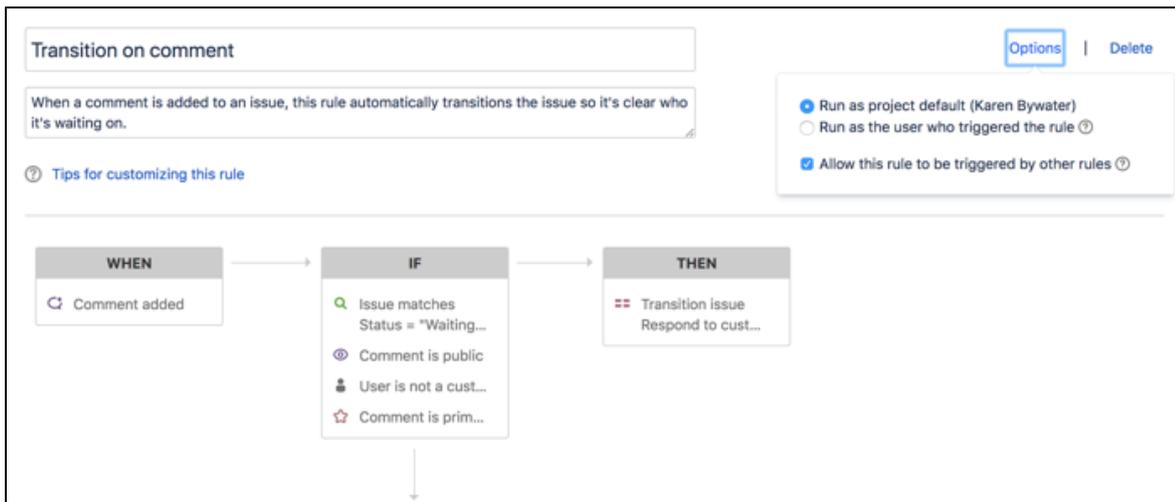
WHEN	IF (optional)	THEN
Comment added	<ul style="list-style-type: none"> • Issue matches a certain filter • Comment visibility is internal or external • User type is a customer or agent • Comment contains a key phrase 	<ul style="list-style-type: none"> • Transition issue to change its position in the workflow • Add comment, either internal or external • Alert user to prompt a specific user or users via an @mention • Edit request type to change the request type (Because request types are mapped to specific issue types, automation isn't able to change issue types. Be sure your request types are the same issue type before applying this rule) • Edit issue to select and change a field in your issue, such as assignee or priority. This affects fields that may not appear in each issue type. • Send email to create an email notification. • Webhook to send a POST request (see our tutorial) • Auto-approve/decline to approve/decline a request based on your IF field
Comment edited	<ul style="list-style-type: none"> • Issue matches a certain filter • Comment visibility is internal or external • User type is a customer or agent • Comment contains a key phrase 	

Issue is created	<ul style="list-style-type: none"> • Issue matches a certain filter • User type is a customer or agent
Issue resolution is changed	<ul style="list-style-type: none"> • Issue matches a certain filter • User type is a customer or agent • Resolution change is either set or cleared
Status changed	<ul style="list-style-type: none"> • Issue matches a certain filter • User type is a customer or agent • Status change visible to customer
A linked issue is transitioned	<ul style="list-style-type: none"> • Link type matches a certain type of link (for example, is related to or blocks) • Issue matches a certain filter • Linked issue matches a certain filter • User type is a customer or agent
Request participant added	<ul style="list-style-type: none"> • Issue matches a certain filter • User type is a customer or agent
Organization added to issue	<ul style="list-style-type: none"> • Issue matches a certain filter • User type is a customer or agent
Approval required	<ul style="list-style-type: none"> • Issue matches a certain filter • User type is a customer or agent

<p>SLA time remaining Select the SLA and goal status that triggers the event</p>	<ul style="list-style-type: none"> • Issue matches a certain filter
---	---

Rule options

Choose Options in the rule's configuration page to change the behavior of how a rule runs, as shown in the image below:



Run rule as: By default, rules run as the person who created the project. Alternatively, you can run rules as the person who triggers the rule. For example, if a rule responds to a customer, then you might want the comment to be from the agent who is working on the request, not from the person who created the project. In this case, you would want to set the **Run rule as** to **User who triggered the rule**. Make sure the person you choose has permission to complete all the actions taken by the rule.

Triggered by other rules: By default, rules can trigger other rules. In some cases, you might need to disable this to prevent two rules from triggering each other infinitely.

To change the default **Run as** user, go to **Automation > Change default event user**. Make sure the person you choose has permission to perform all the actions that you might try to automate in all your rules.

Disable an automation rule

To disable an automation rule:

1. In your service desk project settings, click **Automation**.
2. Select **Edit** next to the rule in your automation list. The rule configuration screen appears.
3. Untick the **Enable rule** checkbox and click **Save**.

Disabled rules appear in your automation list with a **DISABLED** badge.

Managing service desk notifications

When customers submit requests through the customer portal, they receive notifications to keep them informed about the progress of their requests. We call these service desk notifications.

Jira platform notifications are different. When agents and admins work on issues inside of Jira Service Desk, they receive notifications about an issue's activity through the Jira platform. Admins can configure Jira platform notifications through each [project's notification scheme](#).

How different roles receive notifications

A service desk project sends notifications based on two schemes:

1. One for your customer
2. One for your licensed users (most likely your agents and admins)

Customers
When a customer submits a request they receive our default email notifications, as the issue reporter or participant.
For service desks that have incoming email and public sign-up, accounts are actually being set up in the backend. An email is sent to customers asking them to reset their passwords and verify their account details. Service desk admins can disable these account verification emails , if necessary.
Customers in an organization receive notifications when other members of the organization share requests with the organization. They must opt in to notifications to see other activity on the request.
Approvers receive service desk notifications when a request transitions to a stage where their approval is needed. They must opt in to notifications to see other activity on the request.
All other customers receive notifications for all public activities on the requests they're involved in.
Customers can opt in or out of a requests' notifications in the customer portal or email. Reporters who opt out are only notified when the request is resolved.
Agents and admins
When agents and admins work on an issue, they receive email notifications as part of the project's Jira notification scheme .
Agents don't receive notifications of their own changes when they act as a customer on issues with a set request type. Agents acting as a reporter, participant or approver on these issues are always treated as a customer, regardless of the settings in their project's Jira notification scheme. Agents who are part of an organization, will only receive a customer notification when a request is shared with that organization. If the agent is the assignee of the ticket, they won't receive any notifications from the project's Jira notification scheme.

Account verification emails

When customers raise a request by email, the following events take place:

- An account is created for the customer in the backend, so they can access the customer portal
- A notification is sent to the customer, asking them to verify their email address and reset their password, so they can access the customer portal

This is confusing for customers of organizations running email-based service desks; being an email-based service, only the licensed users of the organization (agents and admins) would most likely be using the customer portal.

As Project Administrator, you can disable account verification emails, to avoid confusing your customers.

To disable account verification emails:

You need to be an administrator for your project to make the following changes.

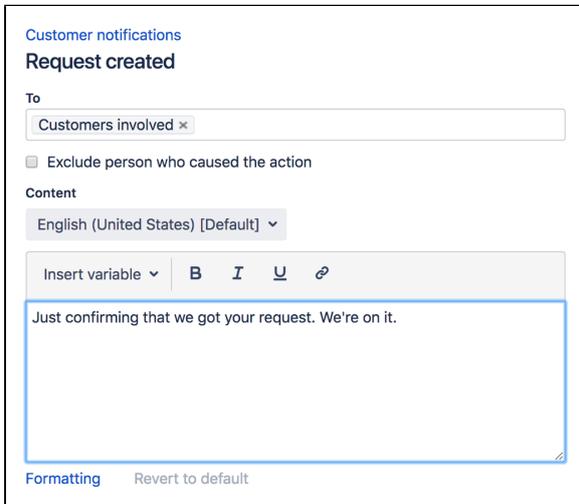
1. From your project's sidebar, select **Project settings > Customer notifications**.

2. Under Account verification email, enable **Do not send account verification emails**.

Edit, disable, and customize notifications

To edit the recipients or message content of your service desk notifications:

1. From your project's sidebar, select **Project settings > Customer notifications**.
2. Select **Edit** next to the notification you wish to alter.



To disable a notification, untick the **Enable** checkbox while editing.

Pro tip: To create custom email triggers, go to **Project setting > Automation** and create a new custom rule. Set up your triggers and use the "send email" THEN action.

Choose recipients

Add at least one of the following to the **To** field:

Recipient options	Details
Reporter (customer)	The reporter of the request. This notification is sent even if the reporter has opted out of receiving notifications on the customer portal.
Customers involved	All customers involved on the request, including the reporter and request participants. This notification is not sent to customers if they have turned off notifications for an individual request in the portal or a request's email thread.
Added participants	The people who have been added as request participants.
Added organizations	The people in an organization that the request has been shared with.
Approvers	The people who need to approve or decline a request.
Exclude person who caused the action	Prevents the person who triggered the rule from receiving a notification. For example, if a customer adds a comment to a request, you can choose to exclude them so they don't receive a notification about their own comment.

Include issue information using message variables

You can use variables to pull blocks of information from issues and insert them into your message. Select the **Insert variable** drop down menu to add valid variables for the notification you're customizing.

Name	Format	Description
Recipient name	\${recipient.name}	The full name of the person receiving the email.
Name of person who caused the action	\${event.user.name}	The full name of the person who triggered the notification, for example by adding a comment.
Issue summary	\${issue.summary}	The summary of the issue, or blank if there's none.
Issue description	\${issue.description}	The description of the issue, or blank if there's none.
Issue key	\${issue.key}	The issue key (for example, IT-123).
Issue reporter	\${issue.reporter.name}	The full name of the user that reported the issue.
Issue resolution	\${issue.resolution}	The resolution of the issue, for example "done".
Request URL	\${request.url}	The URL of the request in the customer portal.
Comment	\${comment}	The comment added to the issue. This is only available using the "Comment added" or "Comment edited" WHEN trigger.
Request status	\${request.status}	The customer-visible status of the request, as shown on the customer portal.
Portal name	\${portal.name}	The portal name, which can be edited on the Portal settings page.
Request details	\${request.details}	The full details of a request. This includes the creation date, request type, summary, and the same fields chosen in the request type settings.
Approval buttons	\${approval.buttons}	The Approve and Decline buttons to action requests from within the email.

Pro tip: Fast-track the time it takes approvers to action pending requests, by inserting **Request details** and **Approval buttons** to your approval notifications template, if they aren't there already.

Support additional languages and translations

Provide translations and regional messages to your service desk's customers. To add or remove a language to your service desk project:

1. Go to **Project settings > Customer notifications**.
2. Under **Language support**, select **Manage languages**.

The languages your service desk supports appear in the table. You can add any languages that your site supports.

We disable new languages by default. When you add a language, your service desk introduces translated notification messages and templates from our default style. Take your time customizing and adjusting these before you decide to enable a language.

If your customer prefers a language that isn't supported in your service desk, they receive notifications in the project's default language.

Translate customer notifications

When creating your message:

1. Select a language from the drop down.
2. Add your translations into the content area, including any relevant variables.

If you leave translations blank, your service desk sends notifications in your instance's default language.

Customize your subject line

Your service desk puts a default subject line on all customer notifications. To change your service desk notifications' subject line:

1. Go to **Project settings > Customer notifications**.
2. Select **Edit templates**.
3. Under **Subject**, edit your subject line.

We require you to keep the issue key someplace in your subject line. This helps thread a request's notifications into a single email conversation.

Style notifications with HTML, CSS, and plain text templates

To change the look and feel of your email notifications:

1. Go to **Project settings > Customer notifications**.
2. Under **Templates**, select **Edit templates**.
3. Adjust the HTML, CSS, and plain text styles as you like.

You can create different styles for different languages using the drop down.

Include more information using template variables

Similar to the variables you can add to individual notification messages, use this set of variables to pull blocks of information and add it to your email template.

Name	Format	Description
Message content	<code>\${message.content}</code>	The content of your notifications. We require this variable. This variable includes batched messages.
Request location	<code>\${request.url}</code>	The URL of the request.
Disable notifications location	<code>\${request.disable.notifications.url}</code>	The URL to turn off notifications for the request. As a best practice, we recommend always including this link.

List of request viewers	<code>\${request.sharedwith}</code>	The participants, approvers and customer organizations who can also view the request.
Help center name	<code>\${helpcenter.name}</code>	The name of your site's help center, an entry point to all customer portals on your site.
Jira Service Desk homepage	<code>\${atlassian.url}</code>	The URL of Jira Service Desk's homepage.

Target additional default CSS classes

You can style more classes than appear in the default template.

Name	Class	Description
Reply marker	<code>.jsd-reply-marker</code>	The style of the reply marker. This series of dashes and hyphens is used to process comments when a customer replies to a notification.
Batched item container	<code>.jsd-activity-item-content</code>	The style of a division used to wrap individual messages added to a batched notification.
Batched message separator	<code>.jsd-activity-item-separator</code>	The style of a separator inserted when batches of messages are sent as a single notification.

Choose HTML or plain text

As a Jira administrator, you can set the default email type for all notifications sent by you site. If the default type is set to HTML, dual-encoded notifications are sent, allowing your customers to then select the HTML or plain text view in their mail client. If your customers use a plain text mail client, you can change your default setting and apply it to new and existing customers.

1. Choose



> **System.**

2. Scroll down to the **User Interface** section and choose **Default User Preferences**.
3. Select **Edit default values**.
4. Change the **Default outgoing email format** to HTML or text and click **Update**.
At this point, the email format you have selected will only be applied to new service desk customers. If you also want to override the email format chosen by existing service desk customers and agents:
5. Under **Operations**, select **Apply**.
6. Select **Update** to finish applying the email preference to all user accounts.

Setting up request types

Jira Service Desk provides a set of default request types that are configured for basic IT help desk scenarios. You can configure the default request types or add new ones to suit the needs of your customers and team. Request types can be organized into groups to help customers find the request they need on the customer portal.

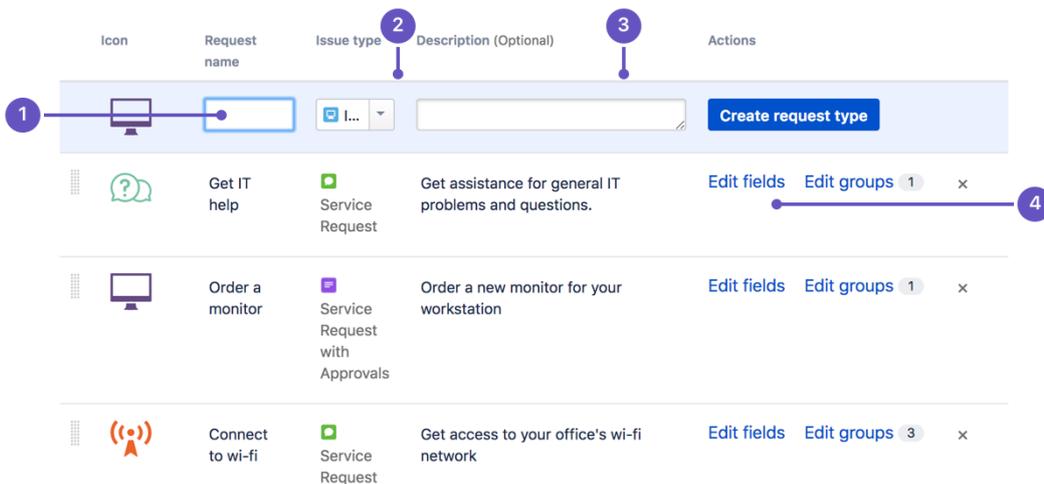
You need to be an administrator to set up request types and workflows in your project.

On this page:

- Set up request types
- Organize request types into groups
- Customize the fields on a request type
- Customize the workflow statuses for a request type
- Hidden fields and unsupported fields

Set up request types

Each request type in a service desk is based on an issue type. Open **Project settings > Request types** to manage your project's request types:

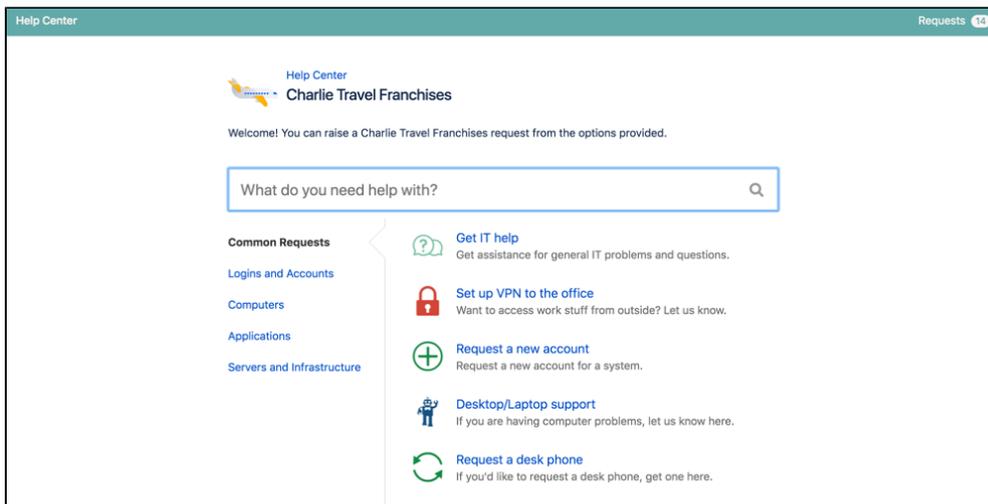


1. **Request name:** give the request an intuitive name by using keywords that your customers look for.
2. **Issue type:** add a new request type by selecting the issue type the request is based on.
3. **Description:** help your customers choose the right request type by providing helpful descriptions.
4. **Edit fields:** customize request fields and workflow statuses, and add request types to groups on your customer portal.

A single issue type can be the basis for many different request types. For example, the *Service Request* issue type serves as the basis for both the "Get IT help" and "Connect to wi-fi" requests.

Organize request types into groups

We recommend using groups if you have seven or more request types, so you can make your request types easier to find on the customer portal. You must have more than one group for the groups to appear in the customer portal. For example: 'Logins and Accounts', 'Applications', and 'Common Requests':

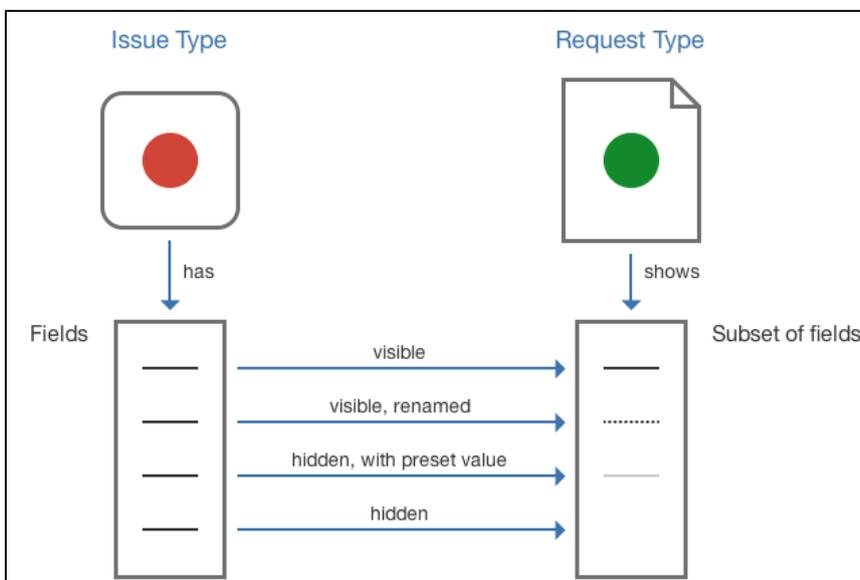


Administrators and project administrators can manage request type groups in **Project settings > Request types**. Click on a group to add new or existing request types to it. You can also create a new group by clicking **+ Add group** and these request type groups are unique to each service desk project.

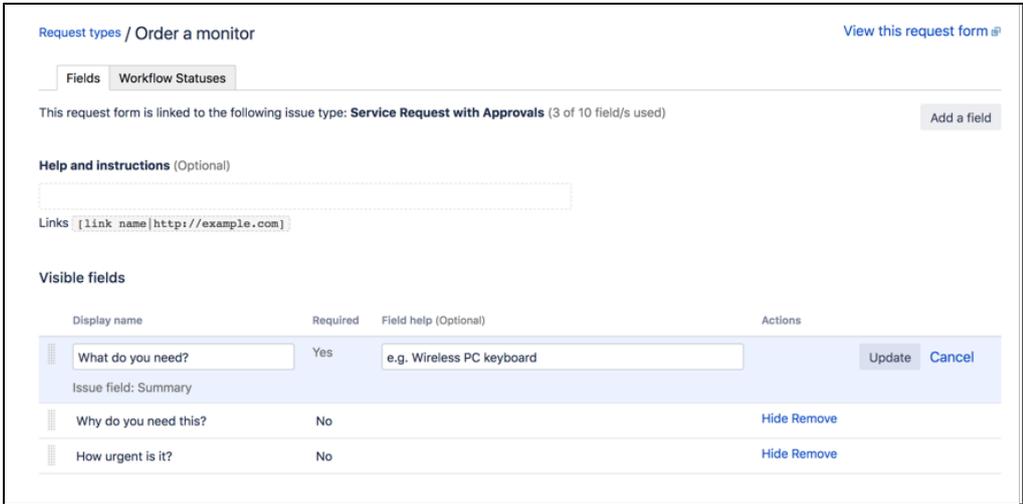
- You can drag and drop request types to re-arrange them in your groups (and, consequently, on your customer portal).
- If you assign multiple groups to a single request type, the request type will appear on multiple tabs.

Customize the fields on a request type

The fields and descriptions that appear in a request type are based on the field configured for the issue type (that is, the issue type the request type is based on).



When editing the request type fields, you can use the **Fields** tab to change the default Jira field names to more customer friendly language. For example, the "Summary" field appears as "What do you need?" for customers.

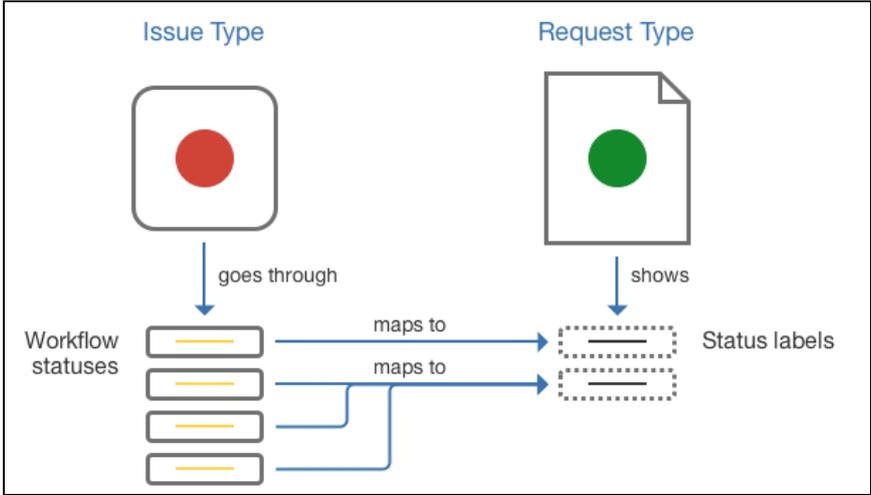


You can also keep fields hidden but available on the request type so that their value can be used for other processes. For more details about how different types of fields work in Jira Service Desk, see [Hidden fields and unsupported fields](#).

If the issue type doesn't have the fields you need, you must [add a field](#) to the Jira issue type that the request type is based on. If the issue type uses multiple screen schemes, the new field must be available in the create screen. See [Associating a screen with an issue operation](#).

Customize the workflow statuses for a request type

Jira Service Desk uses the workflow associated with the request's issue type for the flow of the request.



You can re-map the default workflow statuses to more customer friendly statuses that will appear for customers, and you can also map multiple statuses to a single customer status to simplify the appearance of the workflow. Use the **Workflow Statuses** tab to customize the workflow that customers will see.

Fields

Workflow Statuses

Workflow status in JIRA [view workflow](#)

Resolved

Waiting for customer

Waiting for approval

Status name to show customer

Completed

Waiting for customer

Waiting for approval

Save

Discard unsaved changes

Only changes between these customer-visible 'status names' will be reflected in the Customer Portal and its notifications (e.g. a transition between two workflow statuses can be hidden on the Portal by giving them the same 'status name'). For more information about notifications, see [Managing service desk notifications](#).

If you need to change the workflow of a request, you must edit the workflow associated with the service desk project by going to **Project settings > Workflow**.

Hidden fields and unsupported fields

Each request type in a service desk is based on an issue type. Every issue type has a set of allowed (and possibly required) fields associated with it. As you set up the request type, you can choose to include fields that are hidden on the customer portal but still provide a value to assist with your internal processes and reporting. For example, you might want to set the value of the "Label" field as "hardware" for the "Request new hardware" request type, and set the value as "software" for the "Request new software" request type.

Some fields used by an issue type are not supported for use in the customer portal; if you include these fields on a request type, they will automatically be added to the **Hidden fields with preset values** section and you'll be required to set a value for them.

Other fields aren't supported for use in Jira Service Desk.

These fields can be added to the request type and given a preset value, but you cannot make them visible on the customer portal:

- Assignee
- Linked issues
- Any fields that are defined by other Jira applications
- Group, project, and version picker custom fields

These types of fields can't be added to a request type and won't appear in the in the "Add a field" dialog:

- Issue type
- Log work
- Reporter
- Security level
- Time tracking

Troubleshooting issues with request types

This page contains information about the errors and problems that you might have when setting up request types for your service desk.

Issue	Resolution
-------	------------

<p>Cannot delete the request type because it is the default request type for the email channel.</p> <p>▼ Details...</p> <p>If you see this error when trying to delete a request type, it means that the email channel for your service desk uses this request type as the default one for all the requests coming from emails. When Jira Service Desk pulls emails from the associated email account and creates requests, this request type is assigned to the requests automatically.</p>	<p>Jira administrators can change the default request type for email requests another one by going to Project settings > Email settings in your service more information about the email channel setup, see Receiving requests b</p>
<p>Cannot show a hidden field or make an optional field required because the request type is the default for the email channel.</p> <p>Details...</p> <p>When Jira Service Desk creates new requests from emails sent to the email account associated with your service desk, it copies the email subject to the Summary field and the email content to the Description field. When more fields are required, Jira Service Desk cannot parse emails to fill them in with correct values.</p>	<p>You have the following options:</p> <ul style="list-style-type: none"> • If you want to show a hidden field, make it an optional one. • Ask your Jira administrator to change the default request type for the email channel to use a different request type, and then modify your request type to include more required fields. You can also create a new request type for the email channel if no existing types are suitable. For more information about the email channel setup, see Receiving requests by email.

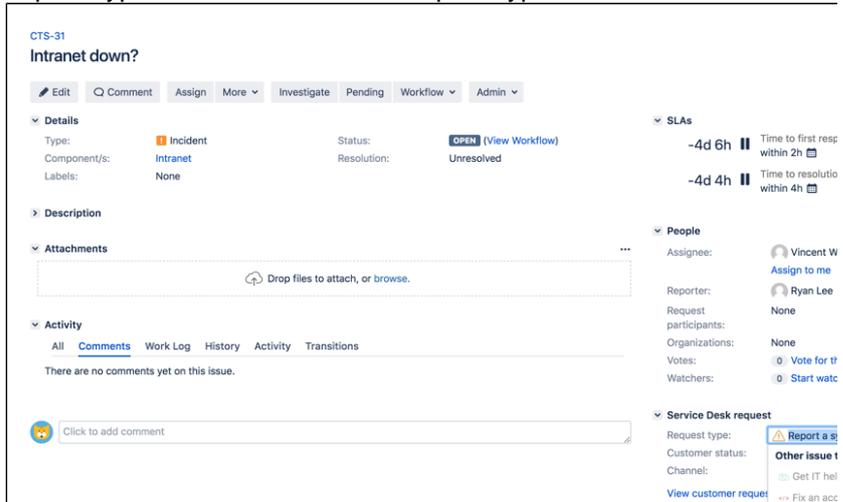
Request type displayed as "No match" in agent view.

Details...

In Jira Service Desk, the service desk request type is stored with the PortalKey/RequestTypeName value. For example, a "New Feature" request created in your "HelpDesk" Customer Portal would have the HelpDesk/NewFeature value. When you move this request to a new project, the HelpDesk/NewFeature value no longer matches the new project's Customer Portal name and request type values.

You have the following options.

- When you move a single issue to a new project, simply edit the service request type field with the correct request type:



- If you need to move a group of issues, you can search for issues with the same issue type in your existing project and then use the Bulk Edit wizard. In step 2, check **Change Customer Request Type** and select the request type that applies to this group of issues.

Setting up SLAs

Good service is what keeps customers coming back, and a key ingredient of good service is responsiveness.

With Jira Service Desk, you can keep your team on track by setting goals for how quickly you manage customer issues. If these goals are set by your customer contracts, you might know them as Service Level Agreements, or SLAs.

SLAs track the progress of things like:

- Respond to all requests within 2 hours.
- Resolve high-priority requests within 24 hours.

See [How teams see SLAs](#) for a detailed look at the Agent's view in different scenarios.

On this page:

- [How to set SLAs](#)
- [How to manage SLA data](#)
- [Best practice SLA usage](#)

How to set SLAs

Your Jira administrators or Project administrators can set SLAs in **Project settings > SLAs**.

When you set an SLA, you choose two things:

- A time metric, which defines how and when time will be measured
- A goal, which defines the target to be met

Set a time metric

The time metric works like a stopwatch, tracking the time between two points in an issue's lifecycle. For example, you might start time when an issue is created, pause time while you wait for the customer to respond, and stop time when the issue is resolved.

Service Desk has pre-configured time metrics to cover the most common IT requirements, but you can

modify these or create your own as needed.

To create a new metric, from your service desk project sidebar, select **Project settings > SLAs > Create SLA** to fill in the following conditions:

Condition	Description
Name	This name (for example <i>Time to resolution</i>) will appear to agents in the SLAs section of issues. Agents should be able to read the name and know what they're being measured on. You can't change the name of an SLA metric after you've saved it.
Start	Time starts being counted against the SLA when any of these occur (for example, <i>Issue Created</i>).
Pause on	Time doesn't get counted against the SLA when at least one of these conditions apply to the issue (for example, <i>Status: Waiting for Customer</i>).
Stop	Time stops being counted against the SLA when any of these occur (for example, <i>Resolution: Set</i>).

You can set multiple conditions for the start, stop, and pause time. Here's an example of the **New SLA** form:

New SLA

Name:

Create Cancel

Time will be measured between the **Start** and **Stop** conditions below.

Start

Begin counting time when

- Clear selected items
- Issue Created
- Resolution: Cleared
- Issue Created
- Assignee: From Unassigned
- Assignee: To Unassigned
- Assignee: Changed
- Entered Status: Awaiting CAB a...
- Entered Status: Defined

→

Pause on

Time is not counted during

- Assignee: Set
- Assignee: Not Set
- Status: Awaiting CAB approval
- Status: Defined
- Status: Resolved
- Status: Waiting for customer
- Status: Peer review / change ...
- Status: Closed
- Status: Under Review

→

Stop

Finish counting time when

- Resolution: Set
- Assignee: From Unassigned
- Assignee: To Unassigned
- Assignee: Changed
- Entered Status: Awaiting CAB ...
- Entered Status: Defined
- Entered Status: Resolved
- Entered Status: Waiting for cu...
- Entered Status: Peer review / a...

Check out [Example: creating an SLA with multiple cycles](#) to see how to create a more complex SLA by starting and stopping the time counter throughout the workflow.

Set a goal

In the **Goals** section of the SLA metric, you choose the type of issue you want to track (**Issues**), how quickly you want to resolve it (**Goal**) and your team's working hours (**Calendar**).

Here are some examples of goals you might set:

Created in 2019 by Atlassian. Licensed under a [Creative Commons Attribution 2.5 Australia License](#).

- Resolve blocker issues within 24 hours.
- Resolve blocker issues created by the Build Engineering team within 12 hours.
- Resolve blocker issues created by the Accounting team within 36 hours.

In the **New SLA** or **Edit SLA** screen, fill out the following fields to define a goal:

Field	Description
Issues	You can enter specific issue criteria using Jira Query Language (JQL) . Base goals on criteria that remain relatively constant throughout an issue's lifecycle (for example, an issue's priority rather than its workflow status).
Goal	This is where you specify the target time for the SLA conditions you previously set. When specifying SLA goals that use a fraction of an hour, write the time as Xh Ym (for example, 3h 30m). You can write SLA goals as hours and minutes, but not days.
Calendar	The calendar allows you to specify working hours when time can be counted against SLAs.

You can drag goals in order of importance. An issue is tracked against the first goal criteria it matches on the list:

Goals

Issues will be checked against this list, top to bottom, and assigned a time target based on the first matching JQL statement.

Issues (JQL)	Goal	Calendar	
<input type="text" value=""/>	<input type="text" value=""/> (e.g. 4h 30m)	Default 24/7 calendar	Add
priority = Highest	36h	Default 24/7 calendar	Delete
priority = Blocker	24h	Default 24/7 calendar	Delete
All remaining issues	No target	Default 24/7 calendar	

Set your working hours

Service Desk lets you create calendars for each goal you assign to an SLA, and that match the working hours of your team. It also lets you exclude lunch breaks, holidays and weekends.

For example, the working hours of your Finance team might be 09:00-17:00 Monday, 08:00-18:00 Tuesday, 10:00-14:00 Wednesday, and 09:00-17:00 Thursday and Friday. Each day would be considered a full working day.

After you have saved your new SLA metric, choose the **Calendars** button to add a calendar with your team's working hours. The **Sample 9-5 Calendar** can be edited to add a 1 hour lunch break that doesn't count toward the SLA. Just delete the 09:00-17:00 line item, and add one for before lunch (08:00-12:00) and after lunch (13:00-17:00). **Save** your calendar and open an SLA metric to associate the calendar with one of your metric goals.

Calendars

Sample 9-5 Calendar

+ Add calendar

Time zone

JIRA default (GMT+00:00) UTC

Work week

Monday 13 : 00 to 17 : 00 Add

Monday	08:00	12:00	Delete
	13:00	17:00	Delete
Tuesday	09:00	17:00	Delete
Wednesday	09:00	17:00	Delete
Thursday	09:00	17:00	Delete
Friday	09:00	17:00	Delete

Save
Cancel

SLA calendars are unique to each service desk project. See [Example: Creating a basic SLA for an example of setting up an SLA that uses a 9-5 working day SLA calendar.](#)

How to manage SLA data

If you create a new name for an SLA, it creates a new custom field that is available to all service desks in your Jira site. As a Jira administrator, you can choose whether users have to be Project administrators or Jira administrators to create new SLA names. You can also view the number of SLA fields being used, and clean up unused fields.

To manage these settings:

1. Choose



> **Applications**. Scroll down to the **Jira Service Desk** section and choose **Configuration**.

2. In the **SLA metric names** section, you can change who can create new SLA metric names. If there are SLA custom fields not in use, click **Clean up** to delete them.

Best practice SLA usage

1. Try to choose an **Assignee** who's not the **Reporter** of an issue. If you assign the same user, your SLA might work incorrectly.
2. If you edit an existing SLA, Service Desk will re-index all the existing issues in the project. All the SLA data for elapsed time will be recalculated to measure against the new metrics. The SLA status is only recalculated for open issues, not for resolved issues.
For example, when the goal for Blocker issues changes from 6 hours to 4 hours, all the closed issues are still considered having met the goal as long as they were resolved in less than 6 hours. This ensures that your reports on closed issues remain accurate for the issues' lifecycle.
3. If issue data changes in such a way that the goals for the issue change (for example, the priority changes from Critical to Blocker), the time against the previous goal will be tracked against the new goal, for open issues only.
For example, if the Support team spent an hour on a Critical issue, when the issue is escalated to Blocker, the hour still counts against the new goal, even if it causes the SLA to be breached.
4. If you're using filters in your service desk to track SLA data, specifically **priorities**, make sure these filters include all the priorities defined in the associated priority scheme. See [Associating priorities with projects](#) for more details.

5. We do not recommend setting up a goal to be dependent on a different SLA.

How teams see SLAs

Support teams rely on SLAs to plan and track their time against issues.

Jira Service Desk shows SLAs in a human readable time, instead of only hours and minutes. That way, Agents don't spend precious time working out what 78:00 means in days.

On this page:
<ul style="list-style-type: none"> • Units of time • How the clock works • Ongoing SLAs • Completed SLAs • Multiple SLA targets • SLA sorting

Your team sees a read-only version of the **SLA** tab, so they know how the SLA is configured. In the issue view, the **SLAs** section lists more information about the SLA that the issue is being measured against:

SLAs

44min Time to first response within 45min

1h 59m Time to resolution within 2h

The times shown by the SLA reflect the calendar settings for that SLA. Hovering over an SLA will give you information about the specific metrics of an SLA.

Units of time

The following units represent time in an SLA:

Seconds	n/a
Minutes	m (min if single unit)
Hours	h
Days	d
Weeks	w
Months	mo
Years	yr

Only 1 or 2 units, at a time, are displayed for simplicity, with a combination of the following:

min	
h	m
d	h
w	d
mo	
yr	

How the clock works

Service Desk calculates minutes, hours, days and weeks by using the working hours set in the associated calendar. It calculates a month and a year by using approximations of 4 weeks and 12 months respectively.

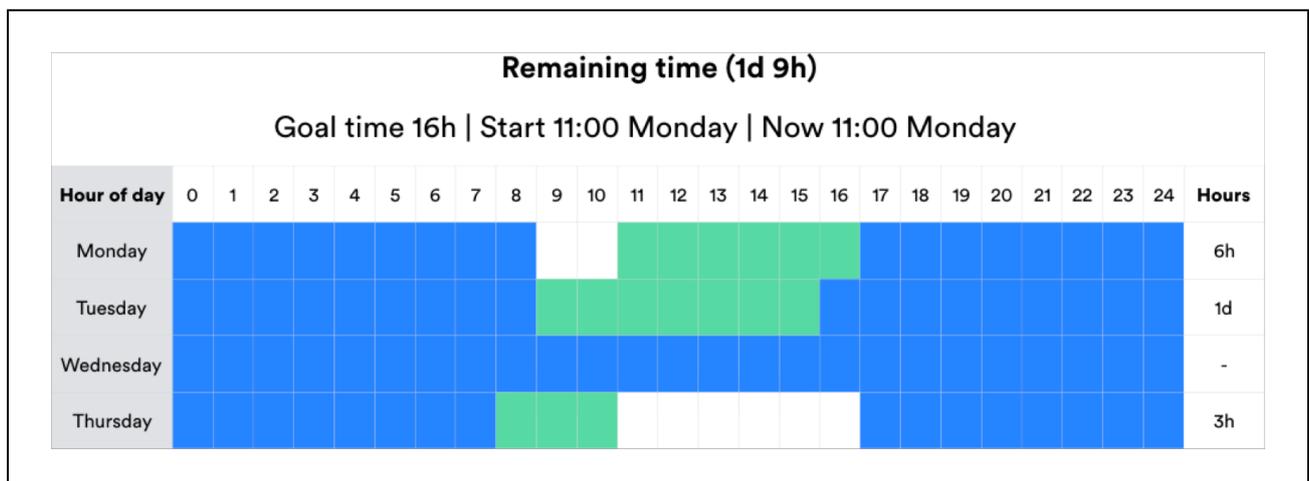
Remaining time

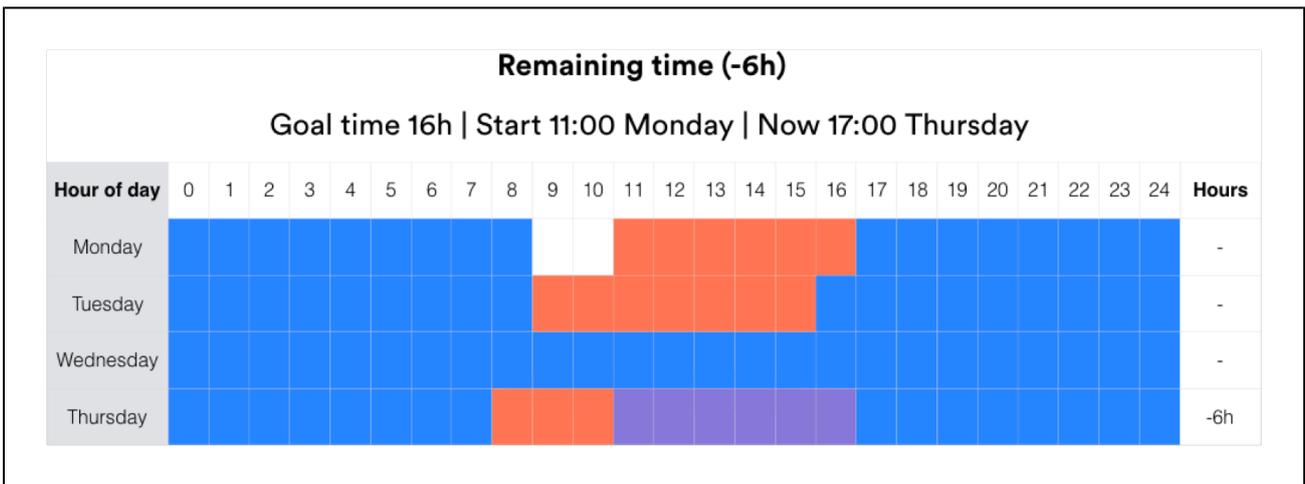
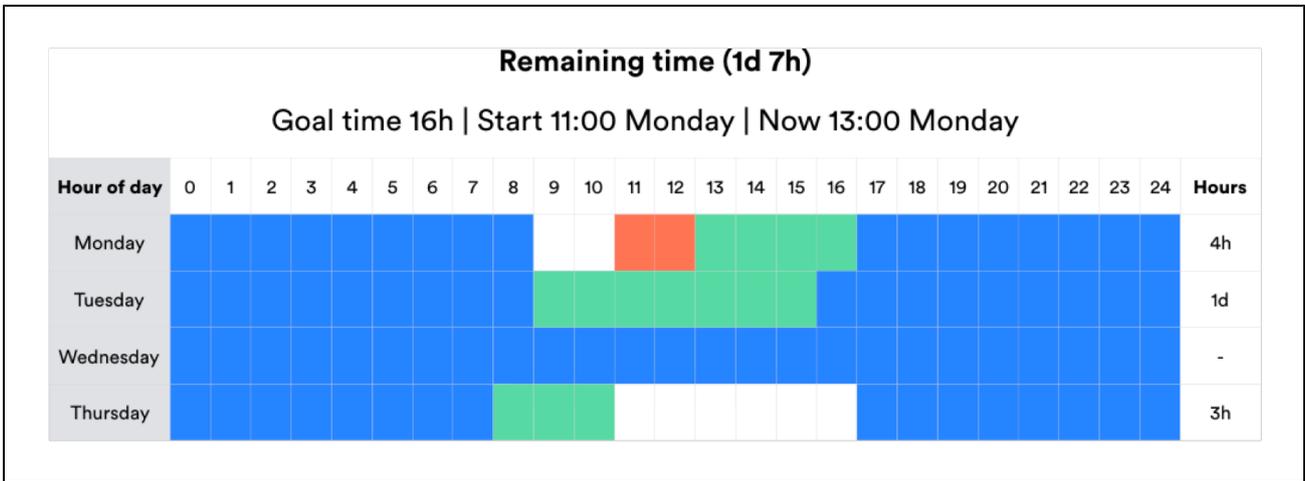
Remaining time is calculated by looking at the time from **now** until **due**, taking into account any pauses.

For example, Request A has an SLA goal time of **16 hours**, with a start time of **11:00** on Monday, and working hours as follows:

- Monday 09:00-17:00
- Tuesday 09:00-16:00
- Wednesday not working
- Thursday 08:00-17:00
- Friday not working
- Saturday not working
- Sunday not working

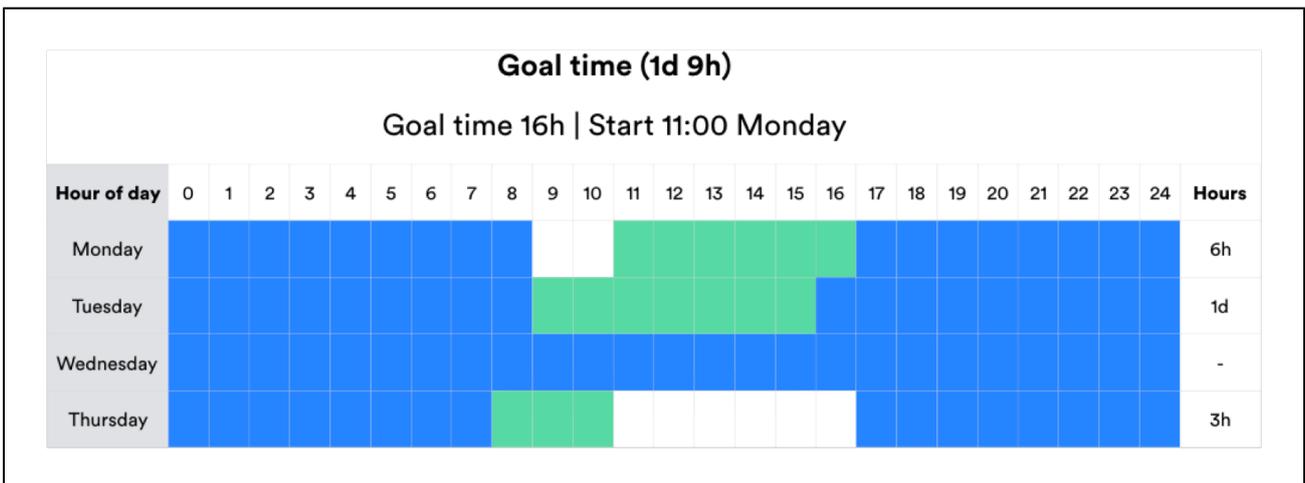
How to read the diagrams:





Goal time

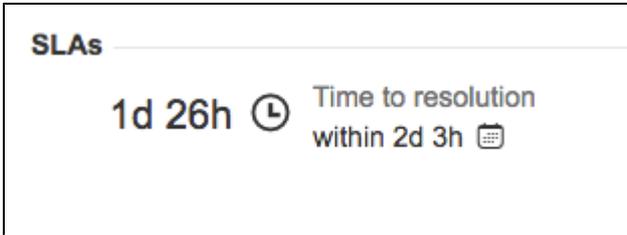
Service Desk calculates goal time similar to remaining time, but **now** is set to **start** of goal. Goal time doesn't take any future pauses into account.



Regardless of the current time, the goal time is always calculated from the start time.

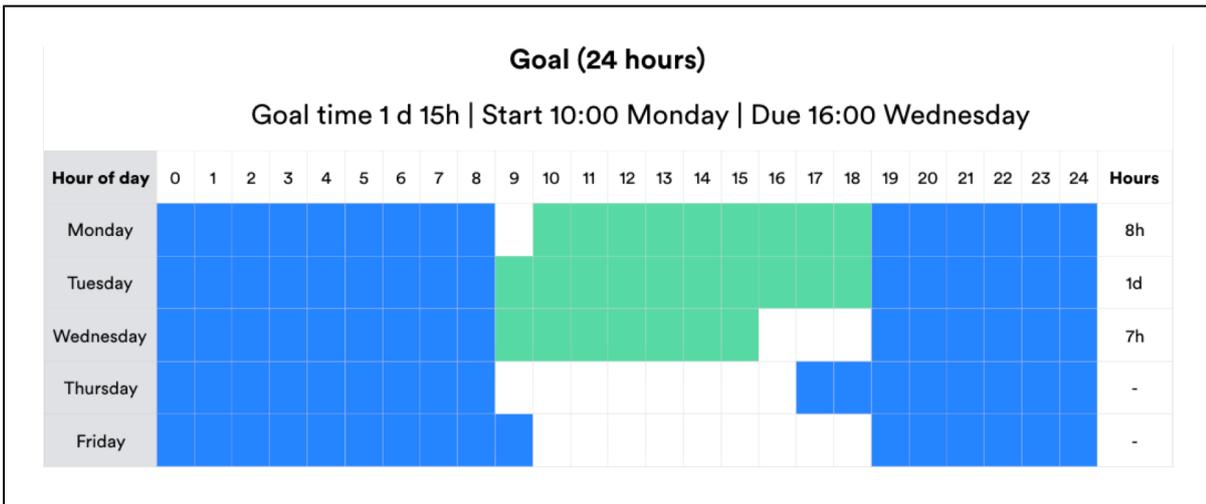
Partial days

In some situations, your Agents might see time displayed as something like 1d 26h. This happens when an SLA starts and stops for a period of less than a day (for example, 7 hours in an 8 hour working day).



All partial days are accumulated as hours and minutes, but not converted to 1 day, because the definition of a day will change depending on the working hours set in the calendar.

For example, Request B has a goal time of 24 hours, with a start time of 10:00 on Monday, and working hours displayed in the diagram below. The SLA goal is **1d 15h** and the SLA is due at 16:00 on Wednesday.



It works like this:

1. From 10:00-18:00 on Monday is not a complete working day (according to the calendar). Total = 8 hours.
 2. From 09:00-18:00 on Tuesday is a complete working day (according to the calendar). Total = 1 day.
 3. From 09:00-16:00 on Wednesday (SLA due time) is not a complete working day (according to the calendar). Total = 7 hours.
- 8 hours + 1 day + 7 hours = **1d 15h**

Ongoing SLAs

The SLA tracker uses colors to indicate the urgency of an SLA for an issue, based on the time remaining.

1h 21m ⌚	The SLA has greater than 1 hour remaining.
36min ⌚	The SLA has less than 1 hour remaining.
19min ⌚	The SLA has less than 30 minutes remaining.
-9min ⌚	The SLA has breached the target. The amount of time past the goal is shown as a negative number.
3min	The SLA has been paused or is currently outside of working hours.

Completed SLAs

A completed SLA displays the time remaining when the SLA was completed (or the amount of time

breached) and an icon to indicate whether the SLA was completed successfully or unsuccessfully.

7min ✓	SLA completed successfully.
-9min ✗	SLA breached its target.

Multiple SLA targets

If the issue meets the criteria for multiple SLAs, trackers for each SLA will appear. In addition, if the SLA has had multiple cycles, you can hover over the symbols for more details on how the SLA was met for that particular cycle. For example, in an SLA that is measured based on when an issue is *Waiting for support*, you can see whether the SLA was met each time the issue started *Waiting for support*.

SLA sorting

When you view a list of issues (in a queue or elsewhere), you can sort them by their SLA remaining times. Ongoing issues are listed first, with the shortest time remaining at the beginning of the list. Completed issues are ranked last, but aren't sorted by the remaining time.

Importing SLAs

Sometimes you may want to reuse SLAs that already exist in another service desk project in your own project. Importing SLAs will import the full configuration of your SLA, including calendars, and allow you to use the same SLAs in your project. These SLAs are **not** linked, so if you make a change to the SLAs in the source project, you'd need to reimport the SLAs to see them in your new project.

Importing an SLA configuration

Before you import your SLAs, we recommend you take a look at how the SLAs are set up in the source project, and make sure you have the required elements in your project. For example, you'll need to make sure you have the same Start, Pause and Stop conditions available, and the same request types available, as these are used to calculate your SLA.

To import an SLA configuration:

1. Navigate to your project and select **Project setting** in the sidebar.
2. Select **SLAs**.
3. Select **+ Import SLA configuration**.
4. Select the project you want to import the SLAs from, and if you want to replace all the existing SLAs, check the relevant box.
Once you've selected the project, if there's any errors or warnings that relate to your import, such as duplicate SLA names, you'll be informed. Some of these you can fix at a later date, or you can choose to stop the import, make the changes, and then reinitiate the import.
5. Select **Import**.

You've just imported the SLAs, and they're now ready to go in your project. Note that these SLAs will **not** affect any closed issues, but if you have open issues their SLAs will be recalculated.

Fixing your SLA import

When you import SLAs, you'll be notified of any issues with your import, and you can fix these after the import. We've pulled together a list of the issues and resolutions to help you get your project up and running as soon as possible.

Issue	Resolution
Goal triggers - the start, pause, and/or stop conditions used by the SLAs you're importing don't exist in the project that you're importing too.	Depending on your project, and what you want to use the SLAs for, you may want to: <ul style="list-style-type: none"> • delete the condition from your imported SLA • create the condition/s that's missing in your imported SLA

Goals - the search filter you're using in the SLAs you're importing is invalid in your instance. Some of the criteria is not available in your instance.	Edit the SLA's search filter and remove the incorrect criteria to fix it.
Calendars - a calendar you're importing has no hours assigned to the work weeks.	Add working hours to the calendar, or you can delete the existing calendar and assign another calendar.
Report - there's an existing report that uses an SLA you're about to delete during your import.	Fix the report/s affected and either: <ul style="list-style-type: none"> • replace the SLA in the affected report • delete the report
Request and issue types - the request and/or issue types used by the SLAs you're importing don't exist in the project you're importing too.	Either delete the references to these in the SLAs, or create the missing request types/issue types and add them to the JQL or condition.

Using JQL queries specific to SLAs

Jira Service Desk includes specific JQL syntax that can help you sort through the details of your requests and issues, and make sure you're hitting your SLA goals. [Read more about JQL syntax.](#)

- If your SLA goals use overlapping JQL filters, your JQL queries may return unexpected results. [Read more about setting up SLA goals.](#)
- If your JQL filters use priorities, make sure these filters include all the priorities defined in the associated priority scheme. See [Associating priorities with projects](#) for more details.

On this page

- Find issues breaching your SLA goals
- Find issues that have paused, completed and are still running an SLA clock
- Find issues based on their SLA clock

Find issues breaching your SLA goals

There are two functions you can use to search for issues that are in a certain state of SLA goal-ness:

- **breached()** filters out issues whose last SLA cycle has failed to meet its target goal
- **everBreached()** filters out issues that have failed to meet their target goal

For example, if you wanted to find all the issues in your project that have successfully completed your first-response goals, use the following query:

```
"Time to first response" != everBreached()
```

Find issues that have paused, completed and are still running an SLA clock

There are three functions you can use to search for issues that are in a certain state of completion:

- **paused()** filters issues whose current SLA cycle is paused. This is determined by your SLA conditions. For example, you may pause an issue's SLA clock when the issue's status is set to "waiting for customer".
- **completed()** filters issues whose SLA cycle is complete, meaning they've reached one of their stop events.
- **running()** filters issues whose current SLA clock is running, meaning they haven't yet reached one of

their stop events.

- **withincalendarhours()** filters issues whose SLA clock is running or not running according to the SLA calendar, *not* conditions.

The paused and running functions do not return issues whose SLA cycles haven't started yet.

For example, if you want to find all the issues that are paused while completing a time to resolution SLA cycle, use the following query:

```
"Time to resolution" = paused()
```

Find issues based on their SLA clock

There are two functions you can use to search for issues that have a certain amount of time on their SLA cycle's clock:

- **elapsed()** filters issues whose SLA cycle's clock meets a specified time condition since the ongoing SLA cycle's start event
- **remaining()** filters issues whose SLA cycle's clock meets a specified time condition before the issue will breach an SLA goal

For example, if you want to find requests that have been waiting for a first response for less than 10 min, use the following query:

```
"Time to first response" < elapsed("10m")
```

Or, if you want to find issues that will breach their resolution target within the next two hours, use this query:

```
"Time to resolution" < remaining ("2h")
```

Reporting on SLAs

The following describes reports we recommend for teams that track SLA goals – which metrics we think are useful and why, and how to create reports.

To view your service desk's reports or create new ones, select **Reports** in your service desk project sidebar. *You must be an administrator to create or edit reports.*

- If your SLA goals use overlapping JQL filters, your custom SLA reports may be inaccurate. [Read more about setting up SLA goals.](#)
- If your JQL filters use priorities, make sure these filters include all the priorities defined in the associated priority scheme. See [Associating priorities with projects](#) for more details.

On this page

- Track success with percent of SLAs met
- Get insight into an agent's performance per SLA

Track success with percent of SLAs met

By default, Jira Service Desk includes an SLA goals report that shows how your team performed against its goals during the past week. This report shows the big picture.

But, if you're interested in trends, want to see a different timeframe or are just more of a visual person, you may find creating a custom report more useful.

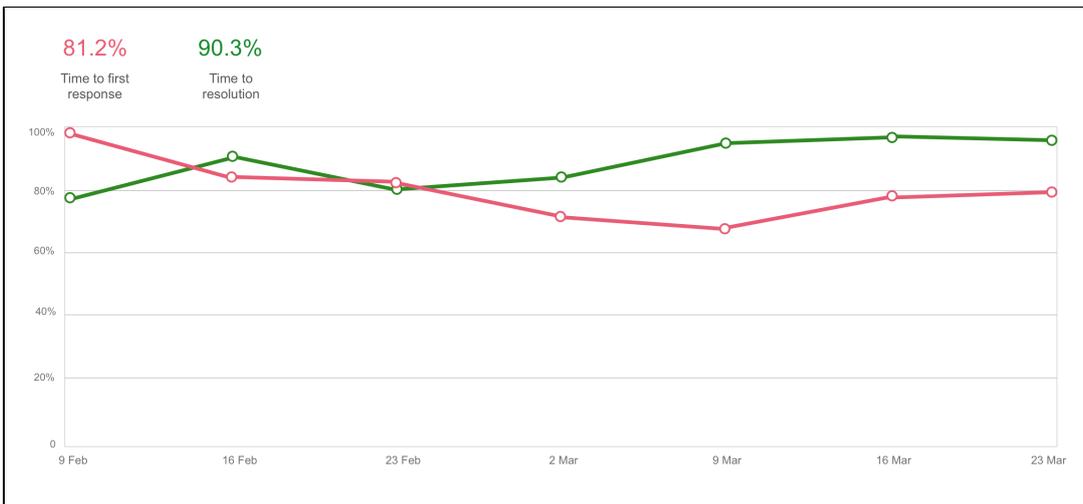
For example, you can create reports to see how well your team performs per request type. Create a report with the following series to drill into how your team is doing on each of your SLA goals for IT help requests:

- **Series** = Time to first response % met
- **Label** = Time to first response
- **Filter by** (advanced) = "Customer Request Type" = "Get IT help"

- **Series** = Time to resolution % met
- **Label** = Time to resolution
- **Filter by** (advanced) = "Customer Request Type" = "Get IT help"

You may find your team is speeding up on responding to customers with these requests but slowing down on resolving these issues. Looking at the details, you might see issues that are breaching your SLA goals feature words like "wi-fi" or "access".

Maybe you need to consider more reliable network hardware or you may need to train your organization about how to properly use the network. You may consider creating a knowledge base article that you can use to quickly answer these questions and increase the percentage of SLA goals met for resolving these requests.



Select a data point on the chart and view all of the issues that were used to calculate the % met metric. We calculate the total from issues that successfully met the goal and those that breached the goal, regardless if they are paused or ongoing.

Get insight into an agent's performance per SLA

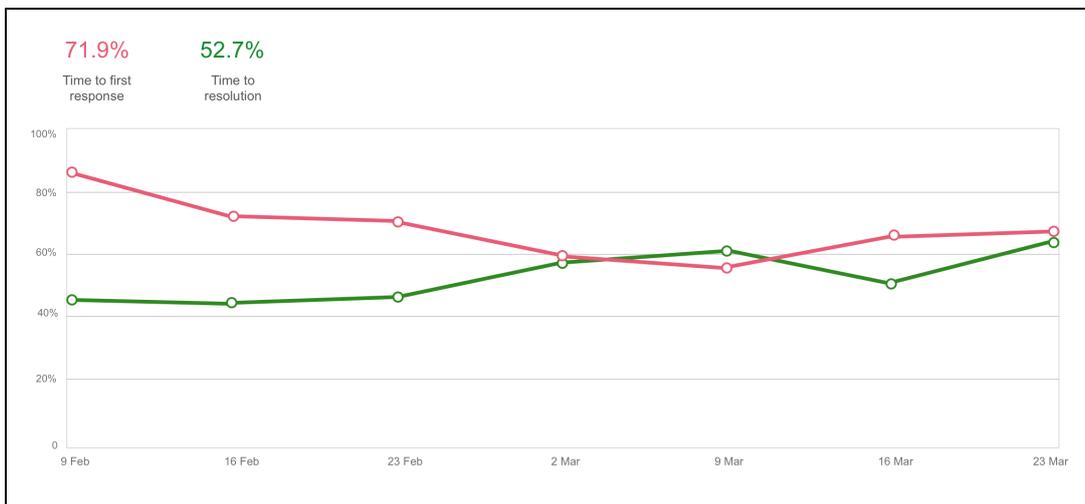
Line managers and other stakeholders may wish to view individual agent performance trends to help determine when agents are being stretched too thin or ensure work is being distributed appropriately.

Use the assignee field and make some reports for your agents to see how they are keeping up with their workload and where their strengths lie.

- **Series** = Time to first response % met
- **Label** = Time to first response
- **Filter by** (advanced) = assignee = agentUserName

- **Series** = Time to resolution % met
- **Label** = Time to resolution
- **Filter by** (advanced) = assignee = agentUserName

You may find that an agent is slow to respond but very quick to resolve requests. Working with the agent, you may be able to strategize how to better notify them about requests when they are raised.



Example: creating a basic SLA

This example looks at how you might create a very basic SLA for a service desk project with a basic workflow:

Basic SLA configuration

All highest and blocker issues must be resolved within 24 hours. You provide 24/7 support for certain customers (these issues are labeled with "24H"). You provide 9-5 support for all other customers, but you don't track SLA metrics for them.

Time to resolution

Time will be measured between the **Start** and **Stop** conditions below.

Start → **Pause on (Optional)** → **Stop**

Begin counting time when: Issue Created

Time is not counted during: [Empty field]

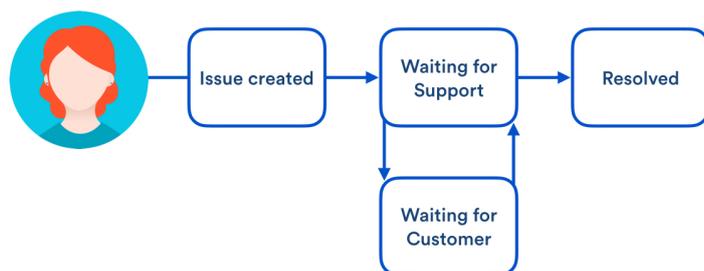
Finish counting time when: Resolution: Set

Goals

Issues will be checked against this list, top to bottom, and assigned a time target based on the first matching JQL statement.

Issues (JQL)	Goal	Calendar
priority = Blocker OR priority = Highest AND labels = 24h	24h	Default 24/7 calendar
priority = Blocker OR priority = Highest	24h	Sample 9-5 Calendar
All remaining issues	No target	Sample 9-5 Calendar

Basic issue workflow



Example: creating an SLA that doesn't track continuous time

This example looks at how you might create a more complex SLA by pausing the time counter during the workflow:

Example SLA configuration

Support wants to complete all issues within 40 hours. Time spent waiting on the customer doesn't count against the 40 hour goal.

Time for support on all issues Edit SLA Delete SLA

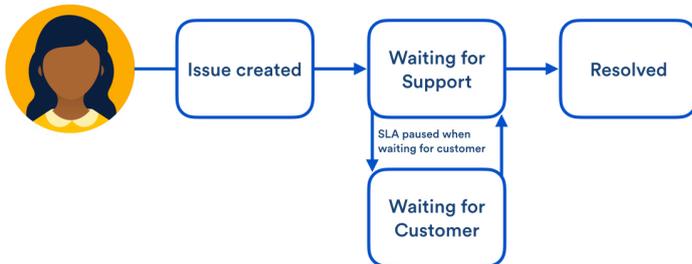
Time will be measured between the **Start** and **Stop** conditions below.

Start Begin counting time when	→ Pause on (Optional) Time is not counted during	→ Stop Finish counting time when
<input type="text" value="Issue Created"/>	<input type="text" value="Status: Waiting for customer"/>	<input type="text" value="Resolution: Set"/>

Goals
Issues will be checked against this list, top to bottom, and assigned a time target based on the first matching JQL statement.

Issues (JQL)	Goal	Calendar
All remaining issues	40h	Default 24/7 calendar

Example issue workflow



Example: creating an SLA with multiple cycles

This example looks at how you might create a more complex SLA by starting and stopping the time counter throughout the workflow. You might set up an SLA like this to track response times (for example, how long it takes your team to respond each time a customer updates an issue with more information). This example also illustrates how goals for different issue criteria can be tracked from a single SLA.

Example SLA configuration

Support wants to respond to **Service requests** within two hours: this includes responding within two hours when the issue is created, as well as each time the issue is updated with more information from the customer.

All other issues have a response time goal of 24 hours.

Time to respond

Time will be measured between the **Start** and **Stop** conditions below.

Start → **Pause on (Optional)** → **Stop**

Begin counting time when → Time is not counted during → Finish counting time when

Entered Status: Waiting for support
Issue Created

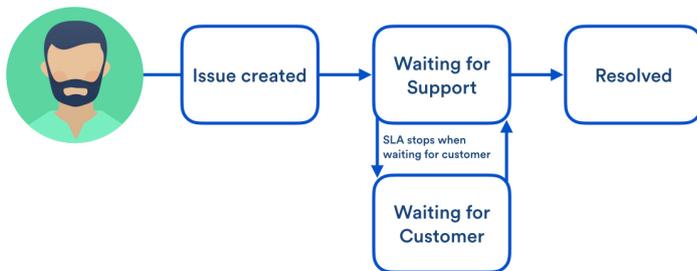
Entered Status: Waiting for customer
Resolution: Set

Goals

Issues will be checked against this list, top to bottom, and assigned a time target based on the first matching JQL statement.

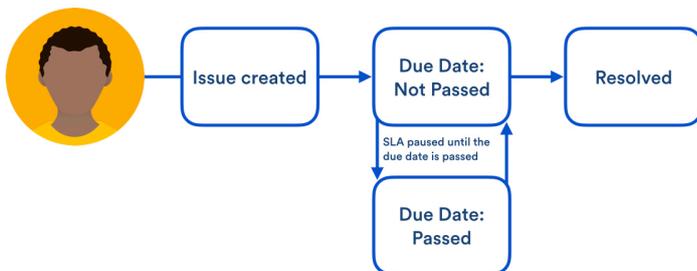
Issues (JQL)	Goal	Calendar
issuetype = "Service Request"	2h	Default 24/7 calendar
All remaining issues	24h	Default 24/7 calendar

Example issue workflow



For further information about how SLAs with multiple start and stop conditions appear in the SLA tracker, see [Setting up SLAs](#).

Example: creating an SLA based on due date



Here's an example of how you might create a more dynamic SLA by pausing the time counter until a specified due date has passed. Set up an SLA like this if your team can't begin their work until a date in the future. For example, setting up a workstation when a new hire starts.

For this SLA to trigger, configure the **Due** field to display on the Issue type screen, and set the **Due** field when the issue gets created. Read [Defining issue type field values](#) to learn how to set this up.

Example SLA configuration

Support want to complete all hardware requests within 24 hours. They want to pause the SLA until they receive the hardware from their supplier (on the expected due date), then unpause the SLA when the due date has passed. All other issues have the same response time goal of 24 hours.

See the image below for how you could set up this SLA in Jira Service Desk:

Issues (JQL)	Goal	Calendar
"Customer Request Type" = "Request new hardware (CTF)"	24h	Default 24/7 calendar
All remaining issues	24h	Default 24/7 calendar

Setting up approvals

Some requests might need approval before your team can work on them. For example, a manager might approve leave requests or an IT manager might approve new system accounts. People don't need a Jira Service Desk license to approve requests; they just have to be customers of the service desk project.

Pro tip: Fast-track the time it takes approvers to action pending requests, by inserting **Request details** and **Approval buttons** to your approval notifications template, if they aren't there already. See [Managing service desk notifications](#).

- On this page:**
- [Setting up an approval step](#)
 - [Best practice customizations](#)
 - [Auto-approve requests](#)
 - [Approval FAQs](#)

Approvers view in the customer portal

How it works:

1. A customer creates a request on the customer portal, and selects the approver required by entering the user name or email address. Alternatively, you can hide the approver field from the customer, and set up a defined list of approvers who are required for that request type.
2. When the request enters the approval status the approver will receive an email.
 - a. If the Request details and Approval buttons variables **have been added** to the approval notifications template, they can view the full details of the request and take action from within the email.
 - b. If the Request details and Approval buttons variables **have not been** added to the approval notifications template, they can view and action the request through the service desk customer portal.
3. The approver can Approve or Decline the request and add an optional comment. The customer receives a standard notification when the request is transitioned out of an approval step, and when the approver leaves a comment.
 - a. If declined, the request moves to the next status in the workflow.
 - b. If approved, the request moves to the next status in the workflow, and an agent is able to work on it.

Setting up an approval step

To set up an approval step on a workflow for your project, you need to have the Jira administrator [global permission](#).

Here are the steps to get approvals working for your project:

- Jira Service Desk creates the **Approvers** custom field automatically. If you want to use another field, make sure you have a user picker custom field available in your Jira instance and on the screens used by your project - this is used on the approval step.
- Add the same user picker custom field to your request type if you want your customers to choose the approver - this will provide the field for your request that allows an approver to be selected.
- Configure the approvals step on the workflow - this allows you to decide if you require one or multiple approvers, and what happens when the approval is declined or approved.

Detailed steps on [setting up an approval](#) are available in the Jira administration documentation.

If you set up your approval step on a status with only two outgoing transitions, they will be used for Approve and Decline. In this case, agents can view requests that require approval, and can modify the approver if required, but they can't change the status until the approver has actioned the request.

If you set up the approval step on a status that has more than two transitions, an agent will be able to transition the request using any of the other transitions that aren't defined as the Approve or Decline transitions. This means the approval step is not enforced.

In certain situations, you may even want to add additional approval steps. For example, if a request needs to be approved by your manager first, and then approved by your finance department. Below are some examples of how you might use approvals:

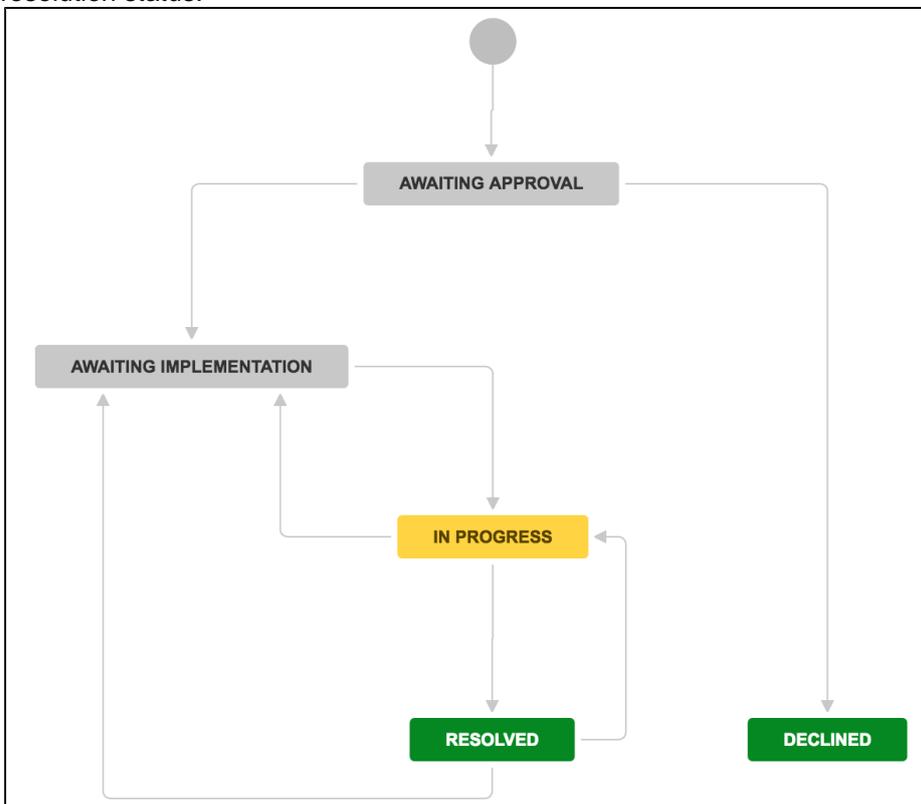
Request type	Approval field	Approver	Notes
Office equipment	Visible on customer portal	Customer selects approver	The customer should select their manager for approval, as each customer could have a different manager.
Computer software	Hidden from customer portal	Set list of approvers	You set the list of approvers, this could be several members of your finance team.

Flights	Two approval steps, one visible, one hidden	First approver is selected by customer. Second approver is a set list	The customer can select the first approver, which could be their manager who approves the business case for their trip. The second approver is from a set list (maybe a finance team) that approves the payment of the flights.
---------	---	---	---

Best practice customizations

You can make the approval process more awesome by making some simple customizations to your service desk project, and that means both you and your customers have a better experience.

1. Make sure you make the name of your multi-user picker custom field, customer friendly on your request. And add a useful help tip, like how many approvers may be required. This will help ensure your customers provide all the correct information first time. Read up on [customizing the fields of your requests](#) for more information on how to do this.
2. When the approval of a request is declined, consider closing the request straight away. This means that if you're setting up a custom approval step, you should ensure that the transition you select for decline leads to a status in the Done category, and the transition has a post function to set the resolution status.



Workflow showing Declined transition leading to Declined status in the Done category

- It's good practice for your approvers to add a comment telling the customer why their request has been declined, and what their next steps should be; for example, the customer should open a new request and provide more information regarding their requirements.

Auto-approve requests

You can use automation rules to auto-approve requests that meet certain conditions. For example, you might auto-approve hardware requests that are under a certain amount.

To set up an auto-approval, create an automation rule with the THEN action auto-approve. For example, any hardware requests created for an item that costs less than \$20 are automatically approved, or any software requests over \$10,000 are automatically declined.

In the example below, we've set up a rule for our customer reimbursement requests, and if the Customer

value is under \$50, the request is automatically approved.



For more information, see [Automating your service desk](#).

Approval FAQs

Why isn't the field I added to my approval step showing on my customer portal?

Make sure you've added the user picker custom field that's used in the approval step on the workflow to your **request type**, and it's visible to customers. If it's still not showing, or it's not listed as a field you can add to your requests, check with your Jira administrator to ensure the field is still available on your project's issue screens.

Declined requests are still showing as open in my service desk project, but they're status shows as Done.

A request will only show as closed when the resolution has been set, so the transition you use to decline the request needs to also set the resolution field. You can achieve this by adding a post function to the transition in your workflow that sets the resolution field of your request. You read up more on post functions on the administration [advanced workflow](#) page.

How do I add, edit or remove approvers on a request?

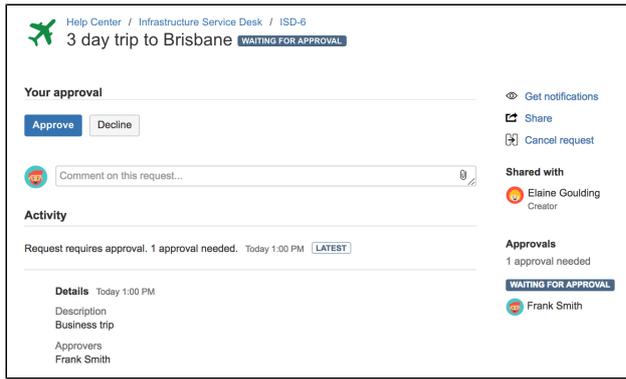
If you need to change the approvers on a request for any reason, you need to edit the user picker custom field that was added in the approval step. View the request in your service desk project, and the editable user picker custom field is displayed in the **People** section of the request. You can make any changes you need to make inline. Note that there is also an **Approvals** section that lists all approvers; however, you can't make any edits here.

If the field isn't showing, you may need to get a Jira administrator to check the field is still available on your project screens.

How can I keep track of the approvals of a request?

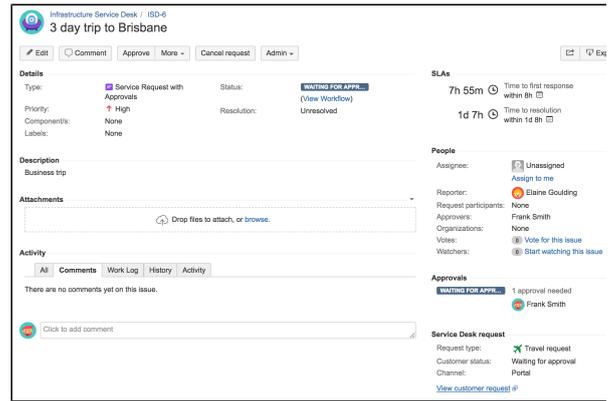
In both your customer portal and your service desk project, you'll see a list of approvers in the **Approvals** section.

Customer portal

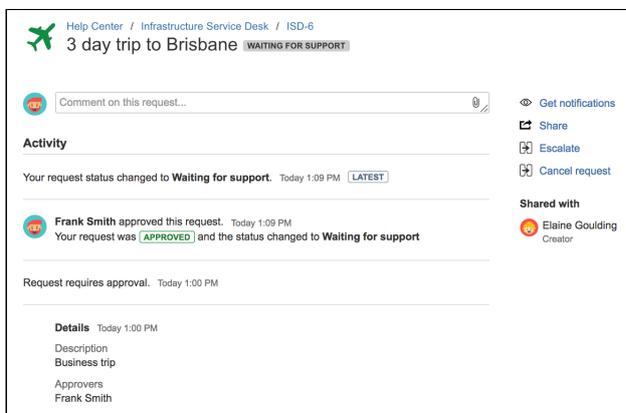


A list of appointed approvers appears in the **Approvals** section, and this section only appears when there are pending approvals for the request.

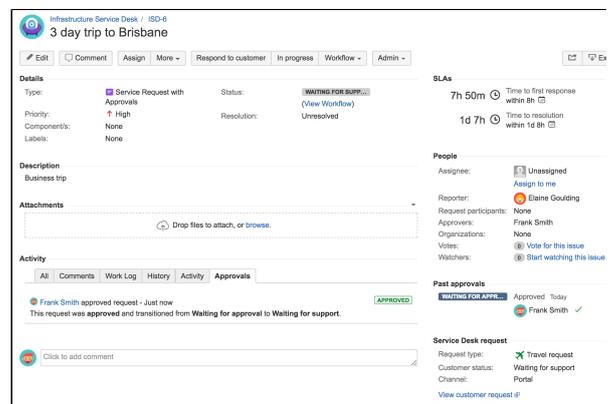
Service desk project



A list of appointed approvers appears in the **Approvals** section, under the **People** section.



When a request is approved, the **Approvals** section disappears, and details about the approval will be added to the Activity section, e.g Your request was **APPROVED** and the status changed to **Waiting for support**.



When a request is approved...

- The approved request displays in the **Past approvals** section, with  beside the approver's name.
- If multiple approvals are required, the request is approved when the minimum number of approvals is reached.
- The **Approvals** tab appears in the **Activity** section, and shows the all the activity involving approvals for the request.

▼ **How do my customers know who to add as an approver?**

Make sure you add **descriptive titles and help text** for the approvals field on your portal. This should include details on who the customer should add (such as their manager, or a member of their IT team), and how many approvers. Depending on your **customer access settings**, your customers may be able to select users or other customers from a drop-down list. Take this into account when adding your title and help text.

Setting up service desk reports

All teams can enjoy a look at their performance and other trends in their service desk. Reports can show the amount and types of requests coming to your team, and how you're resolving them. They can expose areas where you may need more attention or a teammate deserves praise.

We recommend that all teams with a service desk use reports. Besides the following, teams using SLAs can find more value reporting on their SLA goals. [Read more about reporting on SLAs.](#)

To view or create reports, select **Reports** from your service desk project's sidebar. *You must be an administrator to create or edit reports.*

If the JQL filters in your reports use priorities, make sure these filters include all the priorities defined in the associated priority scheme. See [Associating priorities with projects](#) for more details.

On this page:

- [Compare requests created v resolved](#)
 - [View default reports](#)
- [Create custom reports](#)
- [See if your customers are satisfied](#)
- [Track requests created per channel](#)
- [View your average resolution time by issue type](#)
- [Suss out regional trends](#)

Compare requests created v resolved

Knowing how many requests come in and how many your team can resolve helps plan and staff your service desk. This is the most common way to measure any service team's health.

Resolution trends show underlying issues in your organization. These trends help answer questions like:

- Was this week's storm of requests a one-time occurrence or the start of a trend?
- Do we support a service that causes more issues than its worth?
- Is our service desk scaling with our organization or do we need to staff up?

Since this report is the quickest way to check the health of your service desk, we include it by default.

View default reports

Here are the other reports we include by default:

Report name	Details
Workload	Shows the number of requests assigned to your agents
SLA goals	Shows how your team is tracking towards each of the SLA goals you have set
Satisfaction	Shows the average customer satisfaction rating for your team
Article usage	Shows the number of times your customers viewed knowledge base articles
Article effectiveness	Compares knowledge base article viewers with request creators

Created v resolved	Compares the number of requests created and resolved over time
Time to resolution	Compares the length of time taken to resolve requests of type or priority
SLA met v breached	Compares the number of requests that have met or breached an SLA goal
Resolution by component	Compares the resolution times for each component (f or basic service desks only)
Incidents reports by priority	Compares the priority of incidents your customers have reported (f or IT service desks only)

Besides the above, you can dig deeper. *Why* does your performance look like this? Create custom reports for your service desk and explore this question.

Create custom reports

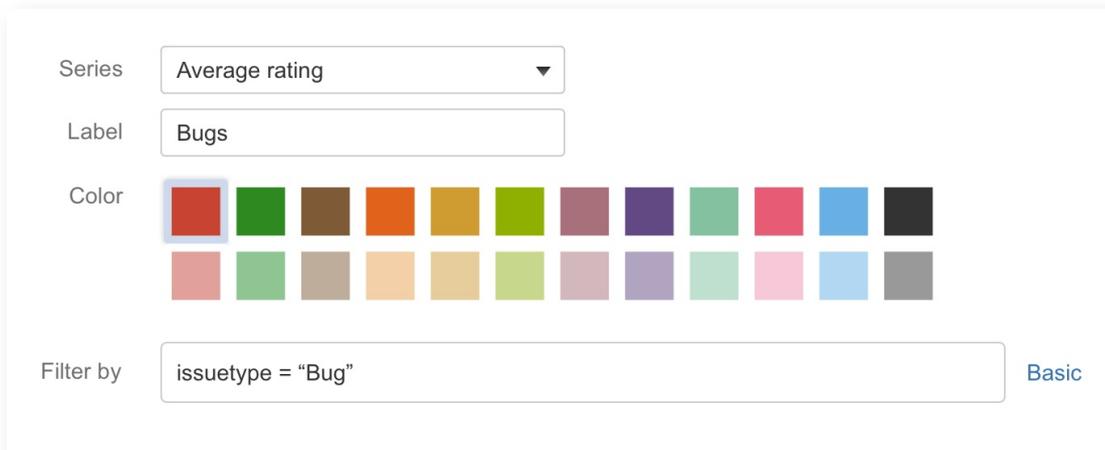
Series make up reports. A series is a set of data points that form a line. For example, the amount of incoming requests on day 1, 2, 3, and so on, for the past week.

Series on their own can point out trends; but, they are more powerful when plotted together. Comparing series can hint at the underlying causes for your service desk's trends.

Create a few custom reports and you'll start to see the value of reporting on your service desk.

To create a custom report:

1. Select **New report**.
2. Choose a report name that you and your team will understand. For example, 'High priority issues'.
3. Select **Add a series**.
4. Fill in the following details: **Series**, **Label**, **Color**, and, optionally a JQL filter.
5. Select **Add** and save your report.
6. Add more series to compare values and create meaning.



Check out the recommended reports below to explore how reports benefit your organization. See examples of what series and their associated JQL filters may be useful to you. [Read more about JQL](#).

See if your customers are satisfied

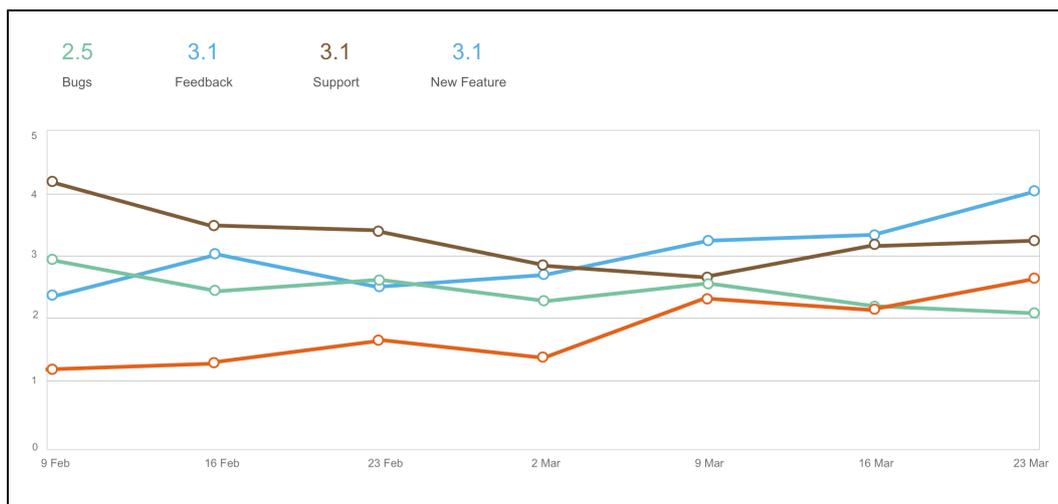
One of the best measure of your service desk's performance is your customers' happiness. Jira Service Desk reports on customer satisfaction straight out of the box. But, you may find more use digging into the details. First, be sure to collect customer satisfaction information on your requests. [Learn how to enable customer satisfaction feedback](#).

Use the average rating customer satisfaction series to see how your team performs. For example, you can use issue types to investigate sections of your organization. To see trends in your customer satisfaction, create a report with the following series:

- **Series** = Average rating
 - **Label** = Bugs
 - **Filter by** (advanced) = issuetype = "Bug"
-
- **Series** = Average rating
 - **Label** = Feedback
 - **Filter by** (advanced) = issuetype = "Feedback"
-
- **Series** = Average rating
 - **Label** = Support
 - **Filter by** (advanced) = issuetype = "Support"
-
- **Series** = Average rating
 - **Label** = New feature
 - **Filter by** (advanced) = issuetype = "New feature"

You might find some interesting results. For example, your response to feature requests may please your customers. But, they aren't happy when they raise requests about billing. Select a data point in the report. You might find these requests feature words like "payment" or "credit card".

Details like these can expose your customers' pain points. Maybe your organization needs an easier form for billing. Maybe you can be clearer about how much your products or services cost or which credit cards you accept.



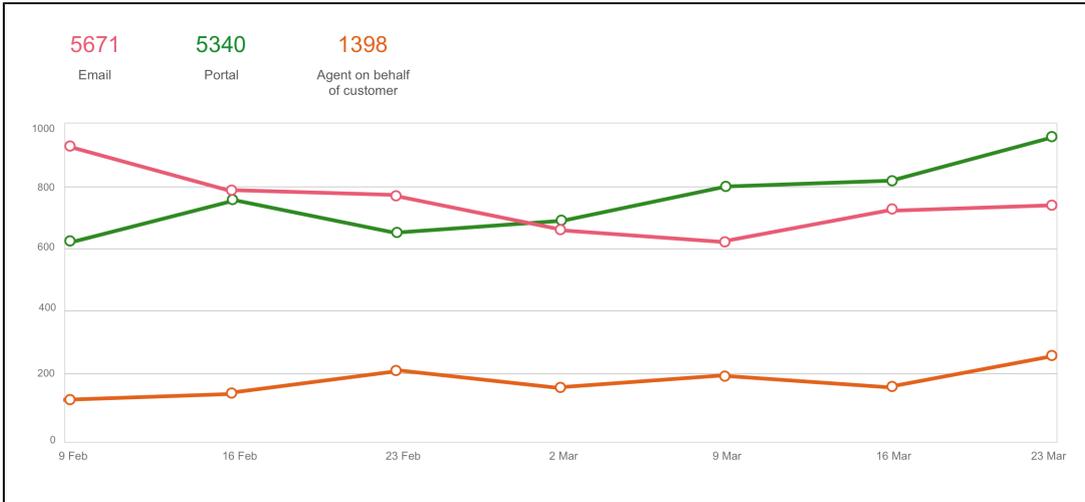
Track requests created per channel

You may consider monitoring *how* your customers submit requests. Are you getting more requests from email than from your customer portal? Create a report with the following series helps answer this question:

- **Series** = Created
 - **Label** = Email
 - **Filter by** (advanced) = request-channel-type = email
-
- **Series** = Created
 - **Label** = Portal
 - **Filter by** (advanced) = request-channel-type = portal
-
- **Series** = Created
 - **Label** = Agent on behalf of customer
 - **Filter by** (advanced) = request-channel-type = Jira

The last series catches issues agents raise outside the portal.

How your customers request help might surprise you. Maybe your agents raise more and more requests on your customers' behalf. If so, you might find ways to direct customers to your portal or email channels. That way your agents have more time to resolve issues, rather than raise them.



View your average resolution time by issue type

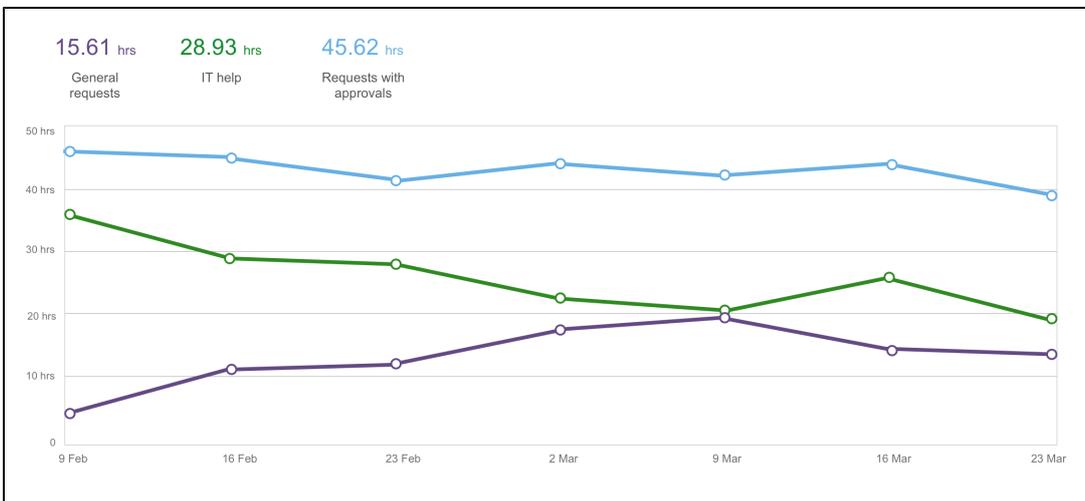
Jira Service Desk tracks requests by time. The time it takes your team to resolve a type of issue can show trends in your teams' efficiency. If you work with a basic service desk, create a report with these series and view the trends that emerge:

- **Series** = Time to resolution (Avg.)
- **Label** = General requests
- **Filter by** (advanced) = "Customer Request Type" = "General requests"

- **Series** = Time to resolution (Avg.)
- **Label** = IT help
- **Filter by** (advanced) = "Customer Request Type" = "IT help"

- **Series** = Time to resolution (Avg.)
- **Label** = Requests with approvals
- **Filter by** (advanced) = "Customer Request Type" = "Request with approval"

You may find that IT help requests take more of your team s' time than general requests. Take into account how many IT help requests come through your service desk. With this info, you can better divide your agents – and their time – to make your customers happier.



Suss out regional trends

If you service more than one location, you can cut out noise by viewing your regional performance. Start by adding labels to requests, identifying and helping to sort them into regions.

For example, suppose your organization operates in New York and Rio de Janeiro. Your service desk agents add a location label to requests from each region. Create a report to see trends in how many requests come from each location using the following series:

- **Series** = Created
 - **Label** = New York
 - **Filter by** (advanced) = labels = ny
-
- **Series** = Created
 - **Label** = Tokyo
 - **Filter by** (advanced) = labels = rio

If you see an increasing trend in one location or another, you may have to shuffle around some resources. Maybe you've opened a new location without a dedicated service desk team member. The new location finds it difficult to ramp up operations. Perhaps you need to send someone to provide training. Or, maybe there's a language barrier with your knowledge base. You may consider providing support articles in more than one language.

You might see the opposite – a decline in requests coming from one location. Are people going rouge and abandoning your service desk? Do you need to make it clear that the service desk operates for all locations?

Default service desk project configuration

Use this page as a reference for the default configuration of your service desk projects, including custom fields, permissions, and database tables.

- [Custom fields](#)
- [Request types, issue types, and workflows](#)
- [Project permissions](#)
- [Security types](#)
- [Database tables](#)

Custom fields

If required, Jira Service Desk will create the following custom fields:

Custom field	Type	Notes
Viewport Origin	String value, storing the 'Portal' and 'Request Type' if a request was created through the customer portal.	Issues must have this field to be a service desk request.
Time to resolution	An SLA field, stored in JSON format.	This field stores SLA information for time until a request's resolution is set. See Setting up SLAs for more information.
Customer Request Type	String value	Issues must have this field to be a service desk request.

Request types, issue types, and workflows

The default issue types, request types, and workflows are different for each service desk project type. When you create a new service desk project, you can view these defaults by selecting Request types, Issues types, or Workflows from the **Project settings** menu.

Project permissions

At installation time, Jira Service Desk creates a project permission called **Jira Service Desk agent access**. Users who require full access to service desk projects or functionality need to have this permission.

This page shows the permission configuration for a standard service desk project permission scheme.

- To see an overview of how permissions are set up for a service desk, see [Permissions overview](#).
- If you want to customize the permission scheme, see [Customizing Jira Service Desk permissions](#).
- If you run into permission-related problems, see [Resolving Jira Service Desk permission errors](#).

Security types

Jira Service Desk introduces the **Service Desk Customer - Portal Access** security type. A security type is a concept that allows restriction of users to certain permissions, examples of security types include [project roles](#) and groups.

Service Desk Customer - Portal Access is a special security type that only applies to users while they are viewing the customer portal; it was created specifically to allow customers to use the customer portal without giving them access to the internal service desk view and your other Jira applications.

Database tables

When you set up Jira Service Desk, the following tables will be created in your Jira application database.

General Jira Service Desk:

- AO_54307E_AGENTSIGNAUTRES
- AO_54307E_ASYNCUPGRADERECORD
- AO_54307E_CAPABILITY
- AO_54307E_CONFLUENCEKB
- AO_54307E_CONFLUENCEKBBENABLED
- AO_54307E_CONFLUENCEKBLABELS
- AO_54307E_CUSTOMGLOBALTHEME
- AO_54307E_CUSTOMTHEME
- AO_54307E_EMAILCHANNELSETTING
- AO_54307E_EMAILSETTINGS
- AO_54307E_GOAL
- AO_54307E_GROUP
- AO_54307E_GROUPTOREQUESTTYPE
- AO_54307E_IMAGES
- AO_54307E_METRICCONDITION
- AO_54307E_PARTICIPANTSETTINGS
- AO_54307E_QUEUE
- AO_54307E_QUEUECOLUMN
- AO_54307E_REPORT
- AO_54307E_SERIES
- AO_54307E_SERVICEDESK
- AO_54307E_STATUSMAPPING
- AO_54307E_THRESHOLD
- AO_54307E_TIMEMETRIC
- AO_54307E_VIEWPORT
- AO_54307E_VIEWPORTFIELD
- AO_54307E_VIEWPORTFIELDVALUE
- AO_54307E_VIEWPORTFORM

Jira Email Processor Plugin:

- AO_2C4E5C_MAILCHANNEL
- AO_2C4E5C_MAILCONNECTION
- AO_2C4E5C_MAILGLOBALHANDLER
- AO_2C4E5C_MAILHANDLER
- AO_2C4E5C_MAILITEM

- AO_2C4E5C_MAILITEMAUDIT
- AO_2C4E5C_MAILITEMCHUNK
- AO_2C4E5C_MAILRUNAUDIT

Automation:

- AO_9B2E3B_EXEC_RULE_MSG_ITEM
- AO_9B2E3B_IF_CONDITION_CONFIG
- AO_9B2E3B_IF_COND_CONF_DATA
- AO_9B2E3B_IF_COND_EXECUTION
- AO_9B2E3B_IF_EXECUTION
- AO_9B2E3B_IF_THEN
- AO_9B2E3B_IF_THEN_EXECUTION
- AO_9B2E3B_PROJECT_USER_CONTEXT
- AO_9B2E3B_RSETREV_PROJ_CONTEXT
- AO_9B2E3B_RSETREV_USER_CONTEXT
- AO_9B2E3B_RULE
- AO_9B2E3B_RULESET
- AO_9B2E3B_RULESET_REVISION
- AO_9B2E3B_RULE_EXECUTION
- AO_9B2E3B_THEN_ACTION_CONFIG
- AO_9B2E3B_THEN_ACT_CONF_DATA
- AO_9B2E3B_THEN_ACT_EXECUTION
- AO_9B2E3B_THEN_EXECUTION
- AO_9B2E3B_WHEN_HANDLER_CONFIG
- AO_9B2E3B_WHEN_HAND_CONF_DATA

Jira Timed Promises Plugin:

- AO_F1B27B_HISTORY_RECORD
- AO_F1B27B_KEY_COMPONENT
- AO_F1B27B_KEY_COMP_HISTORY
- AO_F1B27B_PROMISE
- AO_F1B27B_PROMISE_HISTORY

Using Jira applications with Hipchat

Integrating Jira applications and [Hipchat](#) gives you and your team the following collaboration power:

- Get notifications in your Hipchat rooms when a customer updates a service desk request, or a developer comments on an issue.
- Create a dedicated Hipchat room from the issue you're working on and want to discuss with your team.
- Preview issues and service desk requests directly in Hipchat when someone on your team mentions them.

On this page:

- [Connecting projects to rooms](#)
- [Invite users](#)
- [Discuss issues in rooms](#)
- [Issue preview](#)
- [Remove OAuth Permissions](#)

Connecting projects to rooms

You can link Jira projects with one or more Hipchat rooms so that when issues are updated or created, messages are sent to the Hipchat rooms that you specify.

1. You must be a logged in as an Administrator or a Project Administrator.
2. Choose



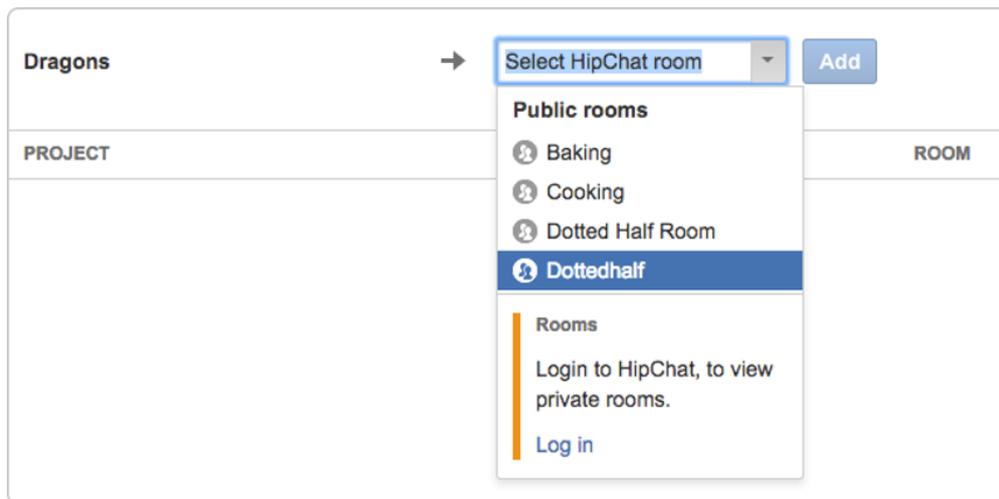
> Projects.

3. Select a project.
4. In the Project settings menu, select **Hipchat Integration**.
5. Choose a Hipchat room and select **Add**.
6. Select the Issue Type, Priority, or select **Advanced** to enter a Jira [JQL Query](#).
7. Select the actions that will send a notification to your room (issue created, assignee changed, new comments, and issue transitions).
8. Select to notify users (using Hipchat notifications) when a message is sent to the room.
9. Changes are saved automatically, continue browsing your project to continue.

Notify Users in This Room uses Hipchat notifications (playing a sound, popups, and bouncing dock icon) to alert users of new messages sent from Jira. This functionality is only available in the web and IOS clients.

Private rooms

Private rooms in Hipchat are by invitation only. In order to in connect Jira to a private room in Hipchat you will need to authorize Hipchat from the **Hipchat Integration** setup screen.



Once you have authorized Jira, all of the private rooms that you are a member of will be displayed in the room selector drop-down menu. When your Jira project and room are integrated, everyone in the private room will be able to see the notifications that are sent to that room.

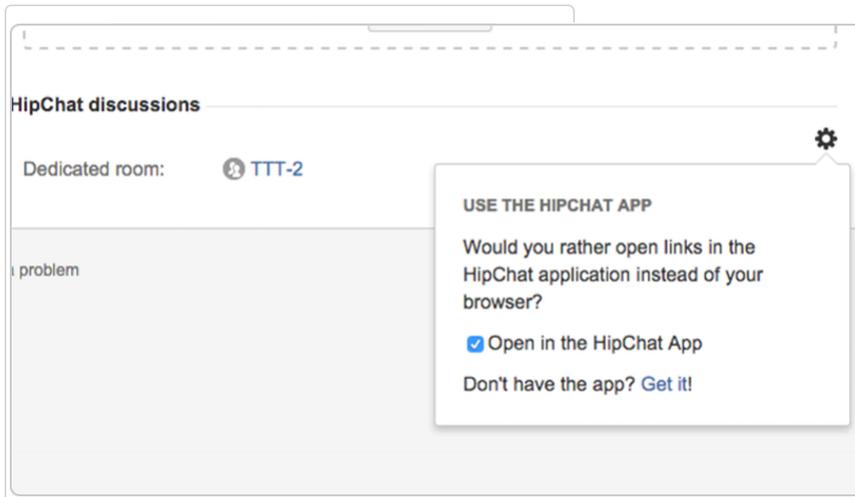
Invite users

If you have administrator permissions, you can invite users to join Hipchat directly from the Integration screen. Follow the instructions in [Linking Jira and your Hipchat site](#) above, to access the integration screen. You must have at least one project integrated with a room to see the invite users link. Select the link to send an email inviting users to Hipchat. To invite users, you will need to confirm access to your Hipchat account to give Jira permission to invite users.

You can remove this access by following the instructions in [Removing OAuth Permissions](#).

Discuss issues in rooms

You can focus your discussion by creating or selecting a Hipchat room to discuss an issue. When Jira is integrated with Hipchat and you are in the issue screen, you can select to "Create a room" or "Choose a room" in the **Hipchat discussions** panel. This will associate the current issue and the room and any changes to the issue will send a notification to that room.

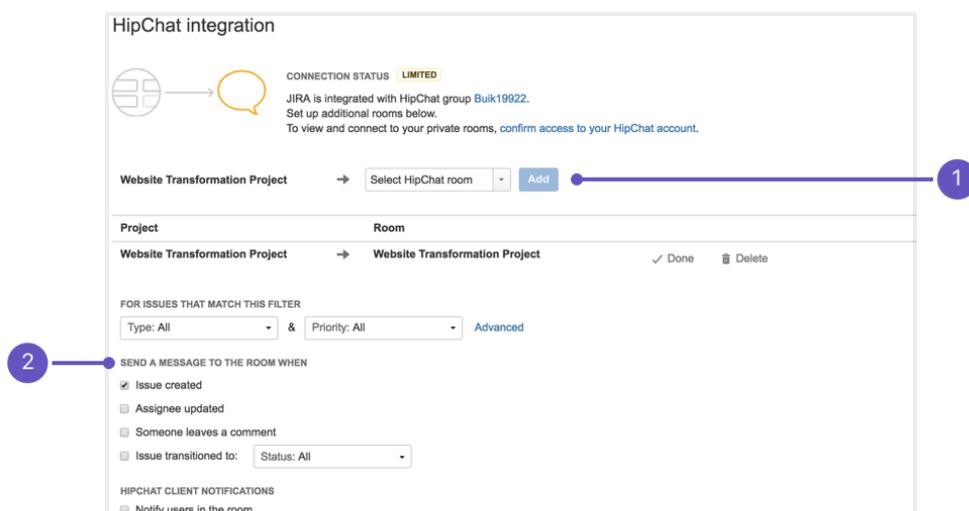


You can also select to have links open in your Hipchat App (OSX only) when you select a link. In the issues screen, select the cog icon in the Hipchat discussion to enable opening links in the application.

Setting up Jira notifications in Hipchat

Project administrators can set up notifications to a Hipchat room whenever work is progressed in their project.

1. Sign in to Jira as a project administrator.
2. Choose  **> Projects**, and select the relevant project.
3. Select **Hipchat integration**.
4. Create a link between your project and a Hipchat room:
 - Select a Hipchat room from the list
 - Click **Add**
5. Alternatively, click **Edit** to change an existing link.
6. Configure the notification settings you'd like to use.

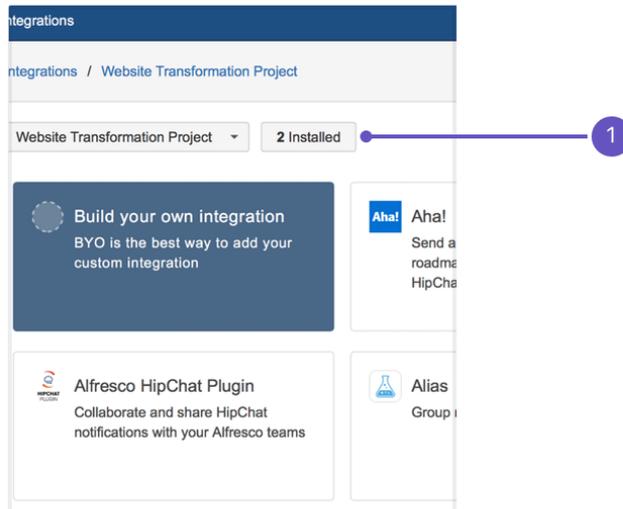


1. **Integration:** Create a link between your project and a Hipchat room.
2. **Messages:** Select messages to send to the room.

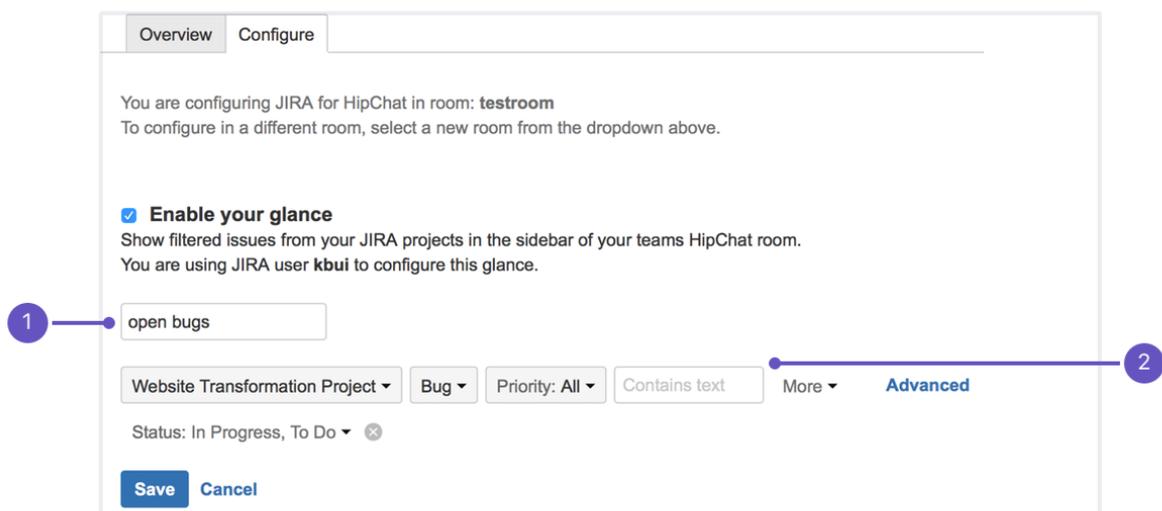
Setting up Jira issues in the Hipchat sidebar

If you have administrator rights in Hipchat, you can also have issues displaying in your Hipchat room's sidebar.

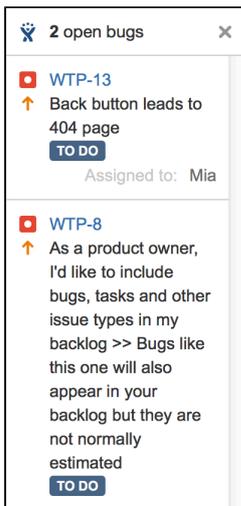
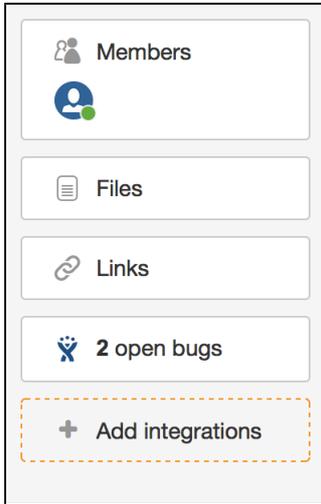
1. Sign in to Hipchat. You'll need to be an administrator of the rooms you want to configure.
2. Select **Integrations** from the top menu.
3. From the drop-down, select the room you'd like to configure.
4. Select **Installed** to see the integrations that have been installed for this room.



1. **Installed:** Click here to see the integrations installed for this room.
5. Select the Jira integration.
6. Select **Enable your glance**. The glance settings will appear.
7. Give your glance a name and set up a basic or JQL filter. The glance name should represent the filter's purpose.

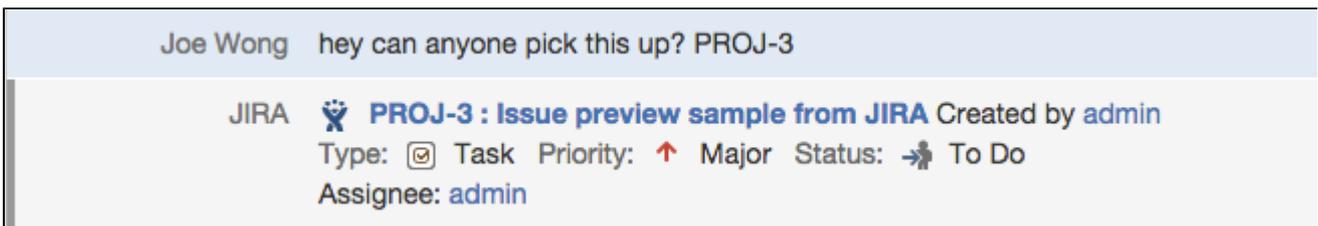


1. **Name:** This will display in the Hipchat room's sidebar above your issues.
2. **Issue filter:** The issues you want to display in the sidebar.
8. Click **Save**. Open your room in Hipchat and click the glance to see these issues in your sidebar.



Issue preview

With issue preview enabled, if you enter an issue key as part of a message, or paste a URL for the issue in any room in Hipchat, you can receive a preview of the issue. This way, the entire room can see and be on the same page when discussing an issue, without ever having to leave the discussion.



If this feature is enabled for a project, a preview will be posted in Hipchat for any issue key/URL for that project. If a project contains sensitive information you don't want shared in Hipchat, make sure to disable this feature for that project.

Connectivity requirements for Jira and/or Hipchat Server customers

For this feature to work, Hipchat needs to be able to talk to Jira, which means that your Jira instance must be addressable and accept inbound connections via HTTPS.

A note on Jira permissions

If this feature is enabled for a project, a preview will be posted in Hipchat for any issue key/URL for that project. If a project contains sensitive information you don't want shared in Hipchat, make sure to disable this

feature for this project.

Configuring issue previews

If you are logged in as a Jira Administrator, you can enable or disable issue preview for all projects. A Project Admin can also override issue preview by individually enabling or disabling this setting for each project.

As a Jira Administrator

1. Log in as a user with the **Jira Administrators** global permission.
2. Choose  **> Applications > Hipchat.**
3. Select **Advanced Settings.**
4. Select the checkbox to enable or disable the Issue Preview globally.
5. Select **Save** to exit.

As a Project Administrator

1. You must be a logged in as a **Project Administrator.**
2. Choose  **> Projects.**
3. Select a project.
4. In the Project settings menu, select **Hipchat Integration.**
5. Select **Advanced Settings.**
6. Select the checkbox to enable or disable Issue Preview for your current room.
7. Select **Save** to exit.

Remove OAuth Permissions

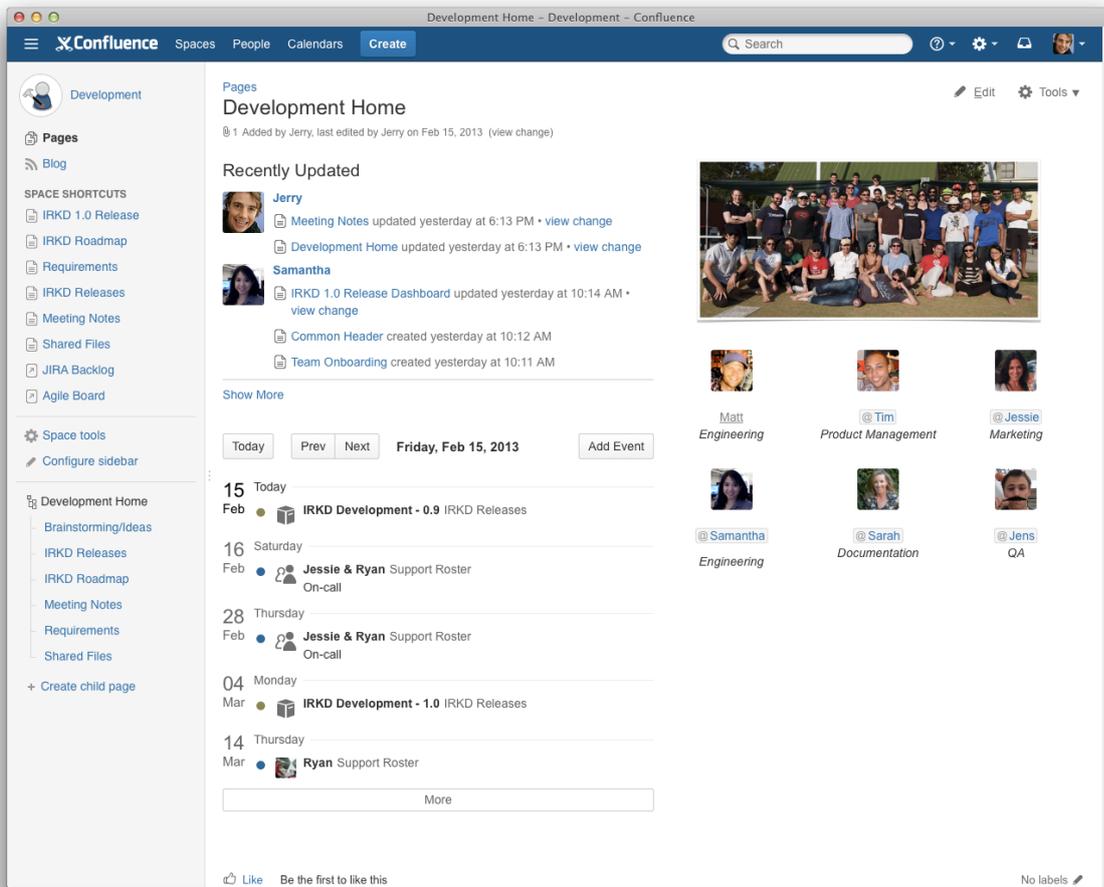
You can remove permissions that you have granted to allow Jira to access Hipchat. For instance, if you have given Jira permission to invite users on Hipchat's behalf.

1. Select your avatar to access your profile.
2. Click **Profile.**
3. Select **Tools.**
4. Click **Hipchat OAuth Sessions.**
5. Select **Remove Access.**

Using Jira applications with Confluence

What is Confluence?

Confluence is a content creation and collaboration platform that connects teams with the content, knowledge, and coworkers they need to get work done, faster. Confluence spaces are great for creating and organizing rich content related to Jira projects using Confluence pages – meeting notes, project plans, requirements documents, release notes, roadmaps, and more.



Why use Confluence with Jira?

Here are some of the reasons we think you might like to add Confluence to your Jira instance:

For a...	You can...
Bug	Create a knowledge base article to document a workaround for a bug.
New Feature	Create a product requirements document for a new feature.
General Jira Use Case	Document and collaborate with your team on an issue in Confluence.

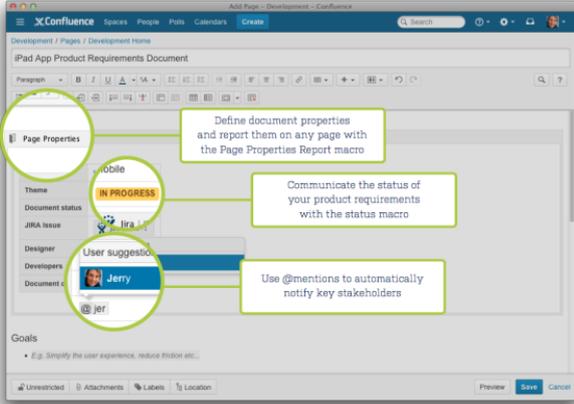
And here are just a few of the things Confluence allows you to do:

- Collaborative commenting, especially through the use of @mentions
- Share pages
- Watch pages
- Form a 'team' network and let them know what you are doing via a status update
- Add images, picture galleries, videos, and more
- Enable various content macros

Confluence features for Jira users

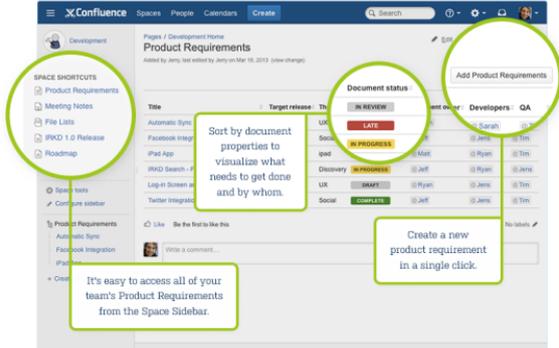
Here are some of the best features of Confluence that Jira users would benefit from.

Define product requirements



Many of our customers write product requirements documents using Confluence to plan new product features. The Product Requirements Blueprint helps development teams collaboratively create, discuss, and organize their product requirements. It's easy to link your product requirements in Confluence to issues in Jira.

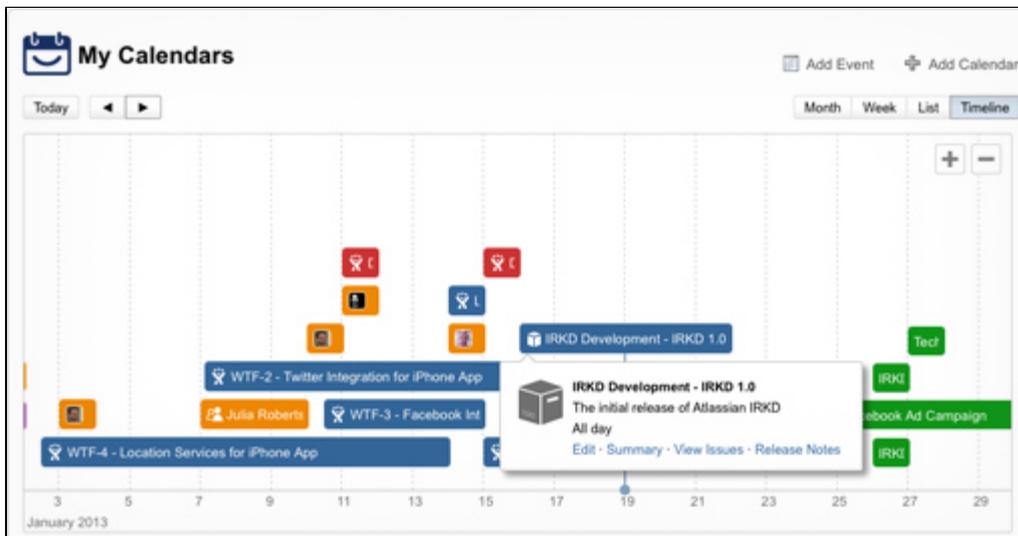
For more information, see [Blueprints](#).



Team Calendars: Your Birds-Eye View of Jira

Surface everything your development team is working on in Jira to the teams that live in Confluence with Team Calendars.

- **Timeline Calendar:** View plans 3 months ahead of time.
- **JQL Support:** Track your versions, issues, and agile sprints.
- **Date Ranges:** Visualize issues over time to understand upcoming workload.



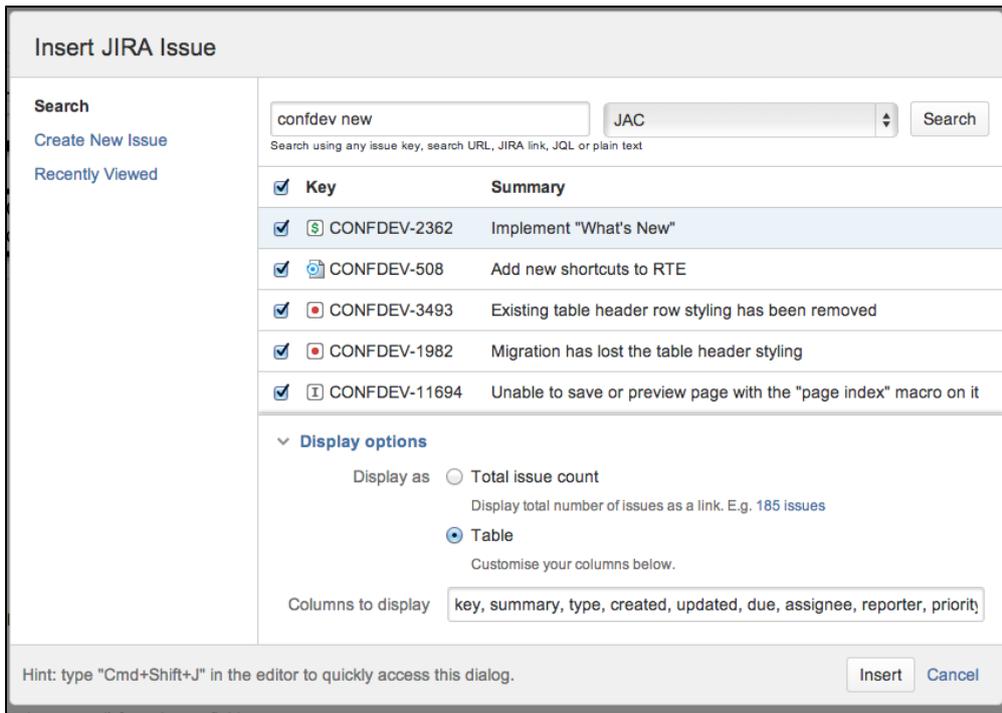
To install this feature, please visit [Atlassian Marketplace](#).

Insert issues on any Confluence page using the Jira Issues macro

Any Jira search result can be embedded in a Confluence page using the [Jira Issues](#) macro with your choice of included fields and field ordering. With the Jira Issues macro, you can:

- Display a table of issues on your page, based on the results of a search using [Jira Query Language \(JQL\) syntax](#) or a Jira URL.
- Display a single issue from the Jira site, or a subset of selected issues from your Jira search results.
- Display a count of issues from the Jira site.

- Create a new issue on the Jira site and display that issue on your page.



Autoconvert pasted issue links

Autoconvert makes producing reports of issues, backlogs, and tasks as easy as copy and paste. With Jira and Confluence connected, you can paste individual issues or Jira query URLs into the editor and watch them immediately transform into the Jira Issues macro.

Automatic links

Whenever an issue is mentioned in a Confluence page using the Jira Issues macro, Jira will create an issue link to that page for you, automatically. Specs to issues, knowledge base articles to support tickets, project outlines to tasks – it all works.

Gadgets

You can embed a Confluence activity stream or a Confluence page in Jira's dashboard. Likewise, Jira gadgets can be rendered on a Confluence page.

Working on service desk projects

If you are an agent working on a Jira Service Desk project, you're in the right place!

If this is the first time you have used Jira Service Desk, check out [Getting started for service desk agents](#) for a brief introduction to your new workspace.

If you're familiar with Jira Service Desk, use the search bar below to find any needed information.

Search the topics in 'Working on service desk projects':

Working on issues

- Work with issues
- Edit and collaborate on issues
- Attach files and

Tracking your work

- Keep on top of SLAs
- Set up dashboards

Serving your customers

- Raise requests on behalf of customers

screenshots to issues

Working with issues

In Jira Service Desk, customer requests are automatically triaged into queues, so you can easily find the issues you need to work on. If you are ready to jump in and learn more about working on and managing customer issues, you're in the right place. This page introduces you to the concept of an issue. You can then learn more about creating, editing, and collaborating issues in the Next steps section.

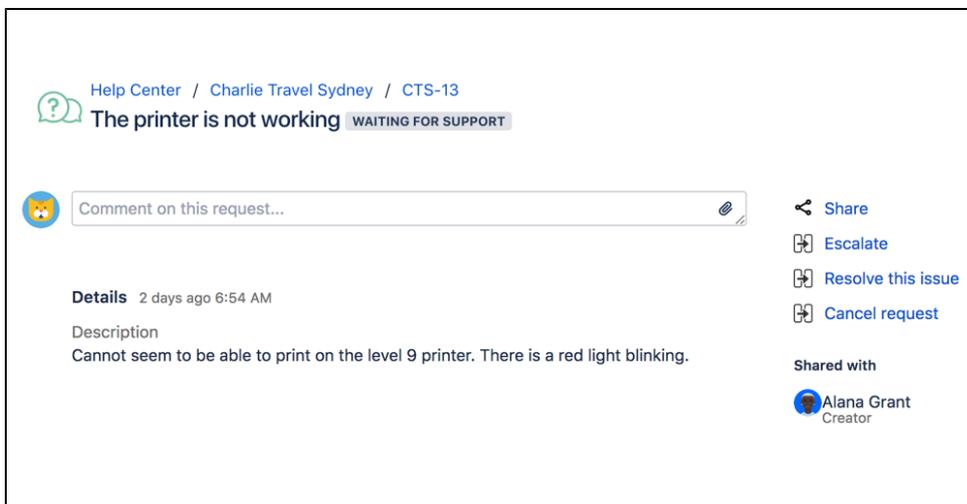
On this page:

- [What is an issue?](#)
- [Next steps](#)

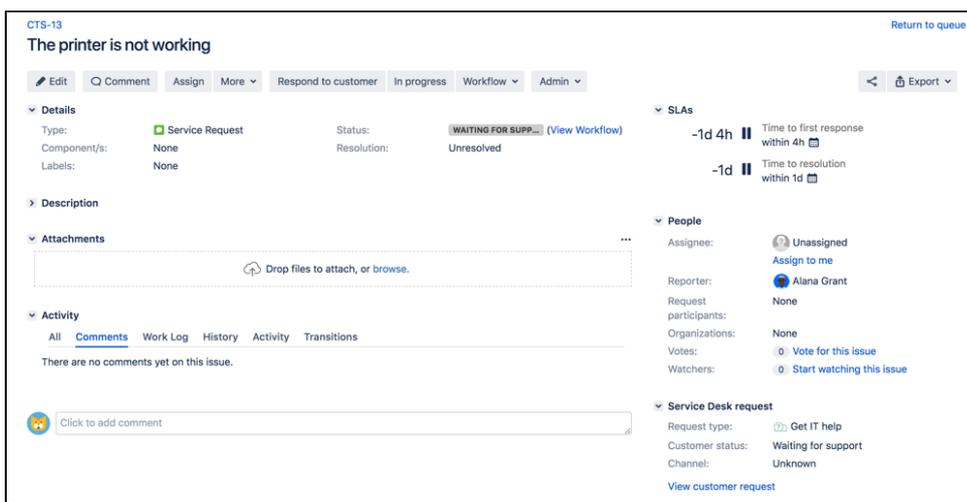
What is an issue?

Different organizations use Jira applications to track different kinds of issues, which can represent anything from a software bug, a project task, to a leave request form.

In Jira Service Desk, an issue is a packet of work that agents work on. In an IT service desk, it represents an incident, a change, and a service request, etc. For example, a customer request of "Our printer is not working" appears as follows in the customer portal:



As an agent, you will pick the issue up internally in the service desk project to work on and it will look like the following:



What activity is shown in the History and Activity tabs?

The **History** tab of an issue records the following information: creator of the issue (this may be the same as the reporter, but can be distinct), changes to an issue field, attachment of a file, deletion of a comment, deletion of a worklog, creation or deletion of an issue link.

The **Activity** tab has the same information, plus additional information, such as comments. However, this may load more slowly, especially if there has been a lot of activity on the issue.

Next steps

Check out the following pages to reach issue ninja status:

- [Creating issues and sub-tasks](#)
- [Attaching files and screenshots to issues](#)
- [Editing and collaborating on issues](#)
- [Logging work on issues](#)

Attaching files and screenshots to issues

To share information with your customers, you can attach documents, images, and screenshots to your Jira Service Desk issues. You can also restrict attachments to be viewed by your internal team only.

On this page:

- [Before you begin](#)
- [Adding attachments](#)
- [Sorting and managing attachments](#)
- [Accessing ZIP file contents](#)
- [Capturing and attaching screenshots](#)

Before you begin

A Jira administrator must enable specific user permissions so that you can add attachments and screenshots into issues. The most common permissions are briefly described below. For more information, your administrator should refer to [Configuring file attachments](#).

▼ [Jira administrator set permissions](#)

- You can attach files and screenshots if your Jira administrator has file attachments enabled.
- You need the **Create Attachments** permission in the appropriate projects.
- The screenshot feature only works with Windows or Mac client. If you use another operating system, you can attach a screenshot using the file attachment feature. For Linux users, please see [our article](#) for enabling this feature.
- If your Jira admin has disabled thumbnails in Jira's attachment settings, the image files will appear as a list.
- If your Jira admin has disabled ZIP support in Jira's attachment settings, the attachments feature will not be available. You must download the zip file to your computer before accessing its individual files.
- To remove attachments from an issue, you need one of the following the project permissions in that issue's project:
 - **Delete Own Attachments** — to delete files that you have added to the issue.
 - **Delete All Attachments** — to delete files that anyone has added to the issue.

▼ [Browser capabilities](#)

- If you're using Google Chrome, Mozilla Firefox, or Internet Explorer 11, attaching screenshots relies on HTML5 compatibility. Safari is not supported.

Adding attachments

You can add files and images to any issue in your service desk project. When working on an issue, simply drag and drop a file onto the issue, or select **browse**. You will then have the option to add a comment with more information about the attachment, and then share the file and comment with your customer or with your

internal team only.

▼ [Acceptable file formats, characters, and sizes](#)

- File formats: GIFs, JPGs, PNGs
- A valid file name cannot contain any of these characters: '\', '/', '\", '%', ':', '\$', '?', '*'.
- By default, the maximum size of any one file is 10MB, although this limit can be customized by your Jira admin.

Sorting and managing attachments

The attachments section of the issue displays a list of options to sort, manage, and download attachments. Select the down-arrow to the right of the attachments section to open the menu. You can reorder the attachments according to a selected criteria. This criteria will be applied to all issues in your project. To remove attachments from the issue, select **Manage Attachments** or hover over the attachment and select



The selected criteria will be lost once you log out.

The screenshot shows the 'Attachments' section of a Jira issue. A file named 'conference_speaker_guide.pdf' is displayed with a thumbnail and metadata: '1 minute ago' and '240 kB'. A dropdown menu is open, showing the following options: 'Sort By Name', 'Sort By Date', 'Ascending', 'Descending', 'Thumbnails', 'List', 'Download All', and 'Manage Attachments'.

Accessing ZIP file contents

You can view the contents of a zip file (including '.zip' or '.jar' file name extensions) in the attachments section. Click the down-arrow and select **List**. In list view, click the arrow icon in front of the zipped file's name to view and download its individual files. If a file is located within a subdirectory of the zipped file, the path to that file is indicated in the content of the zipped file. To download the entire zip file, click **Download Zip**.

Capturing and attaching screenshots

You can capture a screenshot to the system clipboard and paste it directly into an issue.

1. Capture a screenshot using your system keyboard shortcut.
2. Paste the image from your clipboard onto the issue using your system keyboard shortcut or right-click menu. The **Attach screenshot** dialog will display.
3. Enter a filename.
4. Select **Upload**.

Creating issues and sub-tasks

The building blocks of any project are issues. Issues act as the packets of work that travel through their respective workflows within their projects, until

the work is completed. An issue may also have sub-tasks that can be assigned and tracked individually, as well as issue level security to restrict the issue to select members of your team.

On this page, you'll learn more about creating and converting issues and sub-tasks, and setting issue level security. If you are looking to import multiple issues (and sub-tasks) using a CSV file, you can find the import process explained in more detail [here](#).

Before you begin

You need the **Create Issue** project permission for the issue's relevant project.

On this page:

- [Before you begin](#)
- [Creating an issue](#)
- [Cloning an issue](#)
- [Creating a sub-task](#)
- [Converting a sub-task to an issue](#)
- [Converting an issue to a sub-task](#)
- [Restricting access to an issue](#)

Creating an issue

1. Click **Create** at the top of the screen to open the **Create Issue** dialog box.
2. Select the relevant **Project** and **Issue Type** in the **Create Issue** dialog box.
3. Type a **Summary** for the issue and complete any appropriate fields — at least the required ones that are marked by an asterisk.

If you want to access fields that are not shown in this dialog box, or you want to hide existing fields:

- a. Click the **Configure Fields** button at the top right of the screen.
- b. Click **Custom** and select the fields you want to show or hide by selecting or clearing the relevant check boxes respectively, or click **All** to show all fields.

When you next create an issue, these selected fields will be displayed.

4. Optional: To create a series of similar issues – with the same **Project** and **Issue Type** – select the **Create another** checkbox at the bottom of the dialog. Depending on your configuration and the values you may have specified when creating previous issues, some of the fields in the new Create Issue dialog box may be pre-populated. Make sure you check they're all correct before creating the next issue.
5. When you are satisfied with the content of your issue, click the **Create** button.

Cloning an issue

Cloning an issue lets you quickly create a duplicate of an issue within the same project. The cloned issue contains most of the same details stored in the original issue — e.g. Summary, Affects Versions, Components, etc. Other details are not cloned — e.g. Work Log, Comments, Issue history, and Links to Confluence pages. The issue status also returns to the first step of the corresponding workflow, and the resolutions are cleared. The cloned issue can be linked to the original issue, but does not have to be.

1. Open the issue you wish to clone.
2. Select **More > Clone**. The **Clone Issue** screen will appear.
3. You can edit the clone issue's **Summary** if you want.
4. If applicable to the issue you are cloning, you can also select from these options:
 - **Clone sub-tasks** to copy existing sub-tasks
 - **Clone attachments** to add any existing attachments
 - **Clone links** to add any existing linked issues
 - **Clone sprint values** to copy across the issue's current and closed sprint values
5. Click **Create**.

Creating a sub-task

A sub-task can be created for an issue to either split the issue into smaller chunks, or to allow various aspects of an issue to be assigned to different people. If you find a sub-task is holding up the resolution of an

issue, you can convert the sub-task to an issue, to allow it to be worked on independently. If you find an issue is really just a sub-task of a bigger issue, you can also convert an issue to a sub-task.

You can only create sub-tasks if your administrator has enabled sub-tasks, and has added the sub-task issue type to the project's issue type scheme.

1. Navigate to the issue you would like to be the parent issue of the sub-task you are about to create.
2. Select **More > Create Sub-Task**. You will see the **Create sub-task** screen.
3. Fill in the details as needed, and then click **Create** at the bottom of the page.

Note that when you create a sub-task, the following values are inherited from the parent task:

- project
- issue security level
- sprint value, if any (only for Jira Software issues)

Tip: You can customize the **Create sub-task** screen to show fields you use most often. To do this, click **Configure Fields** at the top right corner of the dialog, and use the **All** and **Custom** links to switch between the default screen and your custom settings. Your changes are saved for future use.

Converting a sub-task to an issue

1. Navigate to the sub-task issue you would like convert.
2. Select **More > Convert to Issue**.
3. In the **Step 1. Select Issue Type** screen, select a new issue type (i.e. a standard issue type) and click **Next**.
4. If the sub-task's current status is not an allowed status for the new issue type, the **Step 2. Select New Status** screen is displayed. Select a new status and click **Next**.
5. In the **Step 3. Update Fields** screen, you will be prompted to enter any additional fields if they are required. Otherwise, you will see the message 'All fields will be updated automatically'. Click **Next**.
6. The **Step 4. Confirmation** screen is displayed. If you are satisfied with the new details for the issue, click **Finish**.
7. The issue will be displayed. You will see that it is no longer a sub-task, that is, there is no longer a parent issue number displayed at the top of the screen.

Converting an issue to a sub-task

1. Navigate to the issue you would like to convert.
2. Select **More > Convert to Sub-Task**.
3. In the **Step 1. Select Parent Issue and Sub-Task Type** screen, type or select the appropriate parent issue type and the new issue type (i.e. a sub-task issue type). Click **Next**.
4. If the issue's current status is not an allowed status for the new issue type, the **Step 2. Select New Status** screen is displayed. Select a new status and click **Next**.
5. In the **Step 3. Update Fields** screen, you will be prompted to enter any additional fields if they are required. Otherwise, you will see the message 'All fields will be updated automatically'. Click **Next**.
6. The **Step 4. Confirmation** screen is displayed. If you are satisfied with the new details for the issue, click **Finish**.
7. The issue will be displayed. You will see that it is now a sub-task, that is, its parent's issue number is now displayed at the top of the screen.

Note: You will not be able to convert an issue to a sub-task if the issue has sub-tasks of its own. You first need to convert the issue's sub-tasks to standalone issues; you can then convert them to sub-tasks of another issue if you wish. Sub-tasks cannot be moved directly from one issue to another — you will need to convert them to standard issues, then to sub-tasks of their new parent issue.

Restricting access to an issue

When creating (or editing) an issue, you can restrict access to that issue to members of your team who are part of a chosen security level. To be able to set the security level for an issue, your administrator must add you to the appropriate issue security level, and also grant you the 'Set Issue Security' permission for the appropriate projects.

1. Create/edit the relevant issue.
2. In the **Security Level** drop-down field, select the desired security level for the issue. You will only see the security levels you belong to.
3. Save the issue. It is now only accessible to members of the specified security level. Users who are not members of this security level will not be able to access that issue, or see it in any filters, queries, or statistics.

Creating issues using the CSV importer

If you have the **Create Issue** project permission and the **Bulk Change** global permission for the relevant projects, you can create issues in bulk using a comma-separated value (CSV) file. CSV files are text files that represent tabulated data, and are supported by most systems that handle tabulated data, such as spreadsheets (MS Excel, Numbers) and databases.

The CSV importer allows you to import data from external systems that can export their data in a tabulated format. It also allows you to create your own CSV file to perform bulk issue creation and updates.

Your administrator has access to more import options designed specifically for other systems, such as Github, Fogbugz, and Bugzilla. If you are planning on importing from an external system a large amount of issues, administrators have access to advanced import functionalities by following: [Migrating from other issue trackers](#), including [Importing data from CSV](#).

On this page:

- [Preparing your CSV file](#)
- [Running the CSV file import wizard](#)
- [Tips for importing CSV data into issue fields](#)

There are two steps to using the CSV importer, and an optional third step:

1. Preparing your CSV file
2. Running the CSV import wizard
3. Saving your configuration for future use

Preparing your CSV file

The Jira Importers plugin assumes that your CSV file is based off a default Microsoft Excel-styled CSV file. Fields are separated by commas, and any content that must be treated literally, such as commas and new lines/'carriage returns' themselves are enclosed in quotes.

i For Microsoft Excel and OpenOffice, it is not necessary to quote values in cells as these applications handle this automatically.

CSV file requirements

In addition to being 'well-formed', CSV files have the following requirements:

- Each CSV file must possess a heading row with a Summary column

The CSV file import wizard uses a CSV file's header row to determine how to map data from the CSV file's 2nd row and beyond to fields in your project's issues.

The header row *should avoid containing any punctuation* (apart from the commas separating each column) or the importer may not work correctly.

The header row *must* contain a column for 'Summary' data.

- Commas (as column/field separators) cannot be omitted

For example, this is valid:

```
Summary, Assignee, Reporter, Issue Type, Description, Priority
"Test issue", admin, admin, 1, ,
```

... but this is not valid:

```
Summary, Assignee, Reporter, Issue Type, Description, Priority
"Test issue", admin, admin, 1
```

Encapsulating Jira data structure in your CSV file

Capturing data that spans multiple lines

Use double-quote marks (") in your CSV file to capture data that spans multiple lines. For example, upon import, Jira will treat the following as a valid CSV file with a single record:

```
Summary, Description, Status
"Login fails", "This is on
a new line", Open
```

Treating special characters literally

Use double-quote marks (") around a section of text to treat any special characters in that section literally. Once this data is imported, these special characters will be stored as part of Jira's field data. Examples of special characters include carriage returns/enter characters (as shown in the example above), commas, etc.

To treat a double quote mark literally, you can 'escape' them with another double quote mark character. Hence, the CSV value:

- "Clicking the ""Add"" button results in a page not found error" once imported, will be stored in Jira as:
- Clicking the "Add" button results in a page not found error

Aggregating multiple values into single issue fields

You can import multiple values into an issue field that accepts multiple values (e.g. **Fix (for) Version, Affects Version, Component, Labels**). To do this, your CSV file must specify the same column name for each value you wish to aggregate into the mapped issue field. The number of column names specified must match the maximum number of values to be aggregated into the mapped field. For example:

```
IssueType, Summary, FixVersion, FixVersion, FixVersion, Component,
Component
bug, "First issue", v1, , , Component1,
bug, "Second issue", v2, , , Component1, Component2
bug, "Third issue", v1, v2, v3, Component1,
```

In the above example, the **Component** field of the second issue and the **Fix Version** field of the third issue will generate multiple values in appropriate issue fields upon import.

 Be aware that only a limited number of issue fields support multiple values. The CSV importer will not allow you to import aggregated data into issue fields that only support a single value.

Importing attachments

You can attach files to issues created from your CSV file. To do this, specify the URL of your attachment in an 'Attachments' column within your CSV file.

```
Assignee, Summary, Description, Attachment, Comment
Admin, "Issue demonstrating the CSV attachment import", "Please check
the attached image below.",
"https://Jira-server:8080/secure/attachment/image-name.png",
"01/01/2012 10:10;Admin; This comment works"
Admin, "CSV attachment import with timestamp,author and filename",
"Please check the attached image below.", "01/01/2012
13:10;Admin;image.png;file://image-name.png", "01/01/2012
10:10;Admin; This comment works"
```

i URLs for attachments support the HTTP and HTTPS protocols and can be any location that your Jira instance *must* be able to access.

Importing issues into multiple projects

You can import issues from your CSV file into different projects through a CSV file import. To do this:

- Your CSV file requires two additional columns whose headings should be named similarly to **Project Name** and **Project Key**.
 - Ensure that every issue represented in your CSV file contains the appropriate name and key in these columns for the projects to which they will be imported.
- i** The project name and key data is the *minimum project data* required for importing issues from a CSV file into specific projects.

```
IssueType, Summary, Project Name, Project Key
bug, "First issue", Sample, SAMP
bug, "Second issue", Sample, SAMP
task, "Third issue", Example, EXAM
```

In the example above, the first and second issues will be imported into the 'Sample' project (with project key 'SAMP') and the third issue will be imported into the 'Example' project (with project key 'EXAM') , assuming you match the 'Project Name' and 'Project Key' fields in your CSV file to the **Project name** and **Project key** issue fields, respectively during the CSV file import wizard.

Importing work log entries

Your CSV file can contain work log entries. For example:

```
Summary,Worklog
Only time spent (one hour),3600
With a date and an author,2012-02-10 12:30:10;wseliga;120
With an additional comment,Testing took me 3 days;2012-02-10
12:30:10;wseliga;259200
```

To track time spent, you need to use seconds.

Importing to multi select custom fields

Your CSV file can contain multiple entries for the one Multi Select Custom Field. For example:

```
Summary,Multi Select,Multi Select,Multi Select
Sample issue,Value 1,Value 2,Value 3
```

This will populate the Multi Select Custom Field with multiple values.

Importing cascading choice custom fields

You can import values to a cascading choice custom field using the following syntax:

```
Summary, My Cascading Custom Field
Example Summary, Parent Value -> Child Value
```

The '-' separator allows you to import the hierarchy.

NOTE: Currently Jira does not support importing multi-level cascading select fields via CSV (

JRASERVER-34202 - Allow CSV import to support Multi-Level Cascading Select plugin fields
GATHERING INTEREST).

Running the CSV file import wizard

Before you begin: If your Jira installation has existing data, you should [back it up](#).

1. Select **Issues > Import Issues from CSV** to open the **Bulk Create Setup** page. (If you do not have the option **Import issues from CSV**, your Jira Admin must update the Jira Importers plugin to version 6.2.3 or above.)
2. On the **Setup** page, select your **CSV Source File**.
 Leave the **Use an existing configuration file** checkbox cleared if you do not have a configuration file, or if you want to create a new configuration file. Configuration files specify a mapping between column names in your CSV file's header row and fields in your installation.
 - If you select this option, you will be asked to specify an **Existing Configuration File**.
 - If you do not select this option, then at the end of the CSV file import wizard, Jira will ask you if you want create a configuration file that you can use for subsequent CSV imports.
3. Click the **Next** button to proceed to the **Settings** step of the CSV file import wizard. Complete the required fields.
 - If your CSV file uses a different separator character other than a comma, specify that character in the **CSV Delimiter** field. If the separator is a 'Tab', this can be entered using the format **'t'**.
4. Click the **Next** button to proceed to the **Map fields** step of the CSV file import wizard. Here, you can map the column headers of your CSV file to the fields in your selected project. If you want to select specific Jira field values to map specific CSV values to, tick the checkbox for **Map field value**.
i Note: You must map a CSV field to the issue's summary field. This ensures the issues created have a summary.
5. Click the **Next** button to proceed to the **Map values** step of the CSV file import wizard. On this step of the import wizard, you can select which specific CSV field values you want to map to which specific issue field value. For example, your issue types you may have a CSV field value of "Feature Request", which you may want to map to the issue type field value "New Feature".
i Please note:
 - Any fields whose **Map field value** checkboxes were selected in the previous step of the CSV file import wizard will be presented on this page.
 - Leave a field cleared or clear any content within it if you wish to import the value 'as is'.
 - If you are importing a username-based CSV field (e.g. **Reporter** or **Assignee**) and you do not select the **Map field value** checkbox for this field in the previous step of the CSV file import wizard, then the importer will automatically map imported usernames from the CSV file to (lowercase) Jira usernames.
i Regardless of whether or not you select the **Map field value** checkbox, Jira will automatically create usernames based on the data in your CSV file if they have not already been defined in Jira.
6. Click the **Begin Import** button when you are ready to begin importing your CSV data into Jira. The importer will display updates as the import progresses, then a success message when the import is complete.
7. If you're confident your import is correctly set up, click the **Begin Import** button. Your import will begin and once complete you will be informed of any errors. If you'd like to check your import first, click the **Validate** button and Jira will validate your import and inform you of any expected errors or warnings. You can then go back and correct these before running your full import.

i Note:

- If you experience problems with the import (or you are curious), click the **download a detailed**

- **log** link to reveal detailed information about the CSV file import process.
- If you need to import another CSV file with the same (or similar) settings to what you used through this procedure, click the **save the configuration** link to download a CSV configuration file, which you can use at the first step of the CSV file import wizard.

Congratulations, you have successfully imported your CSV data into Jira! If you have any questions or encounter any problems, please contact [Atlassian support](#).

Tips for importing CSV data into issue fields

Below are some helpful tips when importing data from your CSV file into specific issue fields:

Issue Field	Import Notes
Project	CSV data is imported on a per-project basis. You can either specify an existing project(s) as the target, or the importer will automatically create a new project(s) for you at time of import.
Summary	This is the only required field.
Component(s)	You can import issues with multiple components by entering each component in a separate column.
Affects Version(s)	You can import issues with multiple 'Affects Versions' by entering each version in a separate column.
Fix Version(s)	You can import issues with multiple 'Fix Versions' by entering each version in a separate column.
Comment Body	You can import issues with multiple comments by entering each comment in a separate column.
Due Date	Please use the date format specified on the second step of the CSV import wizard.
Issue Type	If not specified in your CSV file, imported issues will be given the default (i.e. first) Issue Type, as specified in your Jira instance. For more information, see Defining issue type field values . You can also create new values on-the-fly during the import process.
Labels	Import issues with multiple labels by: <ul style="list-style-type: none"> • entering each label in a separate column or • putting all labels in one column, delimited by a space
Priority	If not specified in your CSV file, imported issues will be given the default (i.e. first) Priority as specified in your Jira instance. For more information, see Defining priority field values . You can also create new values on-the-fly during the import process.
Original Estimate	The value of this field needs to be specified as number of seconds.
Remaining Estimate	The value of this field needs to be specified as number of seconds.
Time Spent	The value of this field needs to be specified as number of seconds.

Users	<p>You can choose to have the importer automatically create Jira users for any values of the Assignee or Reporter field.</p> <ul style="list-style-type: none"> • Users will be created as active accounts in Jira. Users will need to get their passwords emailed to them the first time they log into Jira. • Users with no real name will get the portion of their email address (login name) before the "@" character as their Full Name in Jira. • If you are using External User Management, the import process will not be able to create users; instead, the importer will give you a list of any new users that need to be created. You will need to create the users in your external user repository before commencing the import. • If you have a user-limited license (e.g. personal license), and the number of required users is larger than the limit, then the import will be stopped. A page will be displayed showing a list of users that can't be created. • If Assignee and Reporter are not mapped, then no usernames are created.
Other fields	<p>If you wish to import any other fields, you can choose to map them to specific Jira custom field(s). If your custom fields don't exist yet in Jira, the importer can automatically create them for you. If your custom field is a date field, please use the date format specified on the second step of the CSV import wizard.</p>

Editing and collaborating on issues

Resolve your customer requests more efficiently with these tips and tricks for editing and collaborating on Jira Service Desk issues.

In addition to learning about the basics of editing and commenting on an issue, you can refer to this page for help with:

- Using the wiki toolbar to make your comments and descriptions pop
- Sharing issues with your team and adding request participants
- Keeping track of issues with labels and issue watchers

On this page:

- [Attaching files and screenshots](#)
- [Collaborating on issues](#)
- [Editing issue details](#)
- [Commenting on issues](#)
- [Canned responses for comments](#)
- [Formatting text with wiki markdown](#)
- [Tracking issues with labels](#)
- [Watching and voting for issues](#)
- [Reordering sub-tasks on an issue](#)

Attaching files and screenshots

If your administrator has enabled file attachments, you and your customers can attach files and screenshots to issues you're working on. See [Attaching files and screenshots to issues](#) for more information.

Collaborating on issues

You can easily keep your team informed by using the



button to share an issue with other Jira users. If your administrator has enabled anonymous access, you can also share issues by entering the email address of a non-Jira user.

If you want to invite members of your team to help you work on an issue, you can mention them by typing @ and their username in the issue description or comment. People already involved in the issue, like the reporter or a commenter, will be listed first in the user list so you can select them faster. Note that the users you mention will be notified once you save the issue description or comment. In Jira Service Desk, your administrator can also enable Request participants, which will appear as another issue field. You can add other agents and customers from your service desk project to help you resolve the original customer's request.

Editing issue details

What permissions do you need?

To edit an issue, you need the **Edit Issue** project permission for the issue's relevant project. If you do not have this permission, please contact your administrator.

To edit an existing issue, select **Edit** to open the Edit Issue dialog box and modify the issue details. If you want to change the fields you need to edit, select **Configure Fields > Custom** and choose the fields you want to show or hide. Select **Update** to save your changes.

Commenting on issues

What permissions do you need?

To add comments to an issue, i.e. to see the **Comment** button, you must have both of the following project permissions for the issue's relevant project:

- **Browse Project** permission to view the issue to be commented on
- **Add Comments** permission to add a comment to the issue.

Note that you automatically become a watcher of the issues that you comment on. You can disable this via the **Preferences > Autowatch** option in your profile.

What	How
Add a comment	Choose Click to add comment and type your comment for the issue.
Delete a comment	On the comment you wish to delete, select the trashcan icon located on the comment. Confirm that you want to remove this comment from the issue by selecting Delete when prompted.
Edit a comment	Select  located on the comment, and edit the text or restrictions (Viewable by...) as needed. When you save your revised comment, you'll see 'edited' displayed to indicate that the comment has been edited. You can hover over 'edited' to see who edited the comment and when.

Link to a comment	<p>Right-click on the Permlink () icon on the comment, then copy the permanent link to the comment. Paste the copied permanent link into your email or chat message.</p> <p>Clicking the permanent link takes you to that particular comment in the Jira issue. If your Jira issue contains an extensive list of comments, the issue page will automatically be scrolled down so that the linked comment is visible.</p>
Restrict a comment	<p>Select Comment internally (for other internal agents and collaborators) or Share with customer (for customers) tab.</p>

Canned responses for comments

Canned responses allow you to create, edit and manage responses that you can then use at any time, directly from the view issue screen. All agents in your project have access to the canned responses saved, so your colleagues can take advantage of your saved responses, and you can take advantage of theirs. Some typical examples of canned responses could be:

- a request for more information from the customer,
- an short message confirming work in ongoing and the next update will be in 24 hours, or
- a lengthy response asking a customer to accept terms and agreements before you continue with the work.

You can access your canned responses when adding a comment by selecting the canned responses icon



. You can add a response directly from the **Popular** or **Frequently** used menu, or you can search for your response by typing the name in the search field, and selecting it. The canned response will be added to the comment field. Feel free to further edit your comment, it won't be added to the issue until you add the comment.

To edit a canned response, select



> **Edit responses**, select the response you'd like to change and click **Edit**.

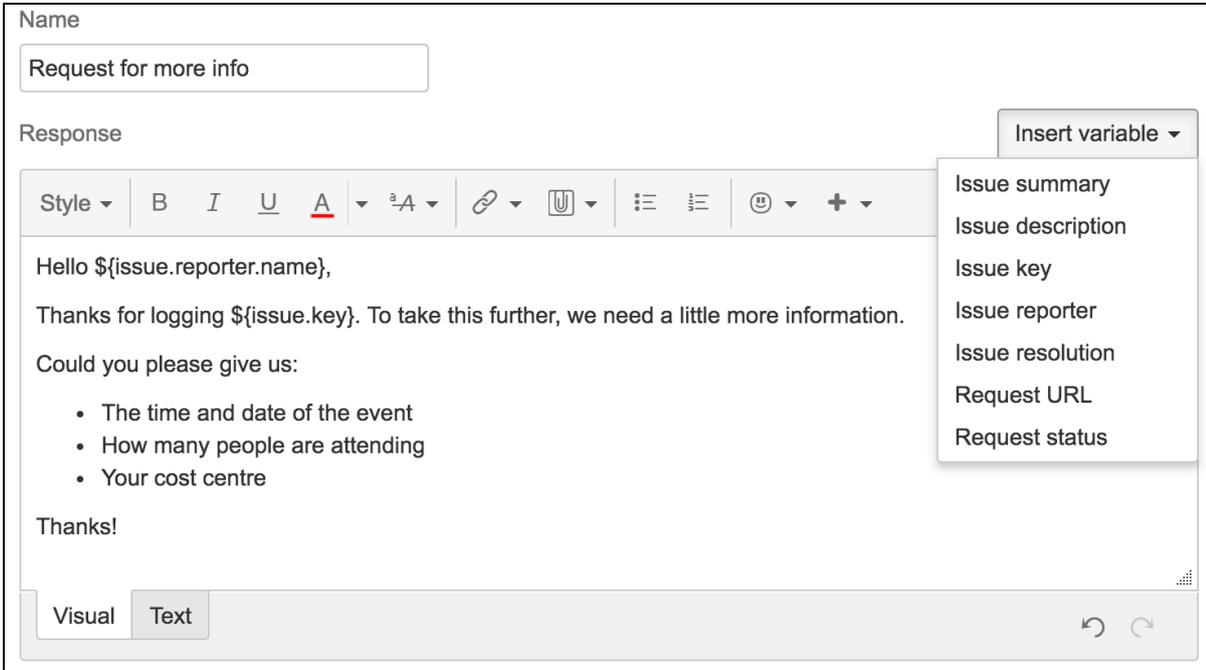
To add a canned response, you can either start with the text you've already added to the comment field, or with a blank comment field. Select



> **Save as a response**, and complete the form with a name for your response, the response, and then select **Save**.

Using variables in canned responses

A variable is a powerful way to customize your response. Using a variable in your response allows the response to automatically contain information related to the issue. For example, you may want to personalize your response by adding the reporter's name, or by adding the issue key. The current list of variables are shown on the image below:



Formatting text with wiki markdown

Jira application [Text Formatting Notation](#) allows you to use rich-text features, such as:

- Italic, bold, underlined text
- Multiple levels of headings
- Bullets, numbered lists, tables, and quotations
- Images
- Macros

When you edit an issue description, comment, or any rich-text field, you can expand the simple wiki editor toolbar to format your text and select **preview** to see how your formatted text will appear. Note that your Jira administrator can enable, disable and configure the renderer which allows you to use wiki markdown, so your options may vary slightly.

HTML macro

When using the HTML macro, which allow you to add HTML code to an issue, you should only use formatting as if you are including something inside the `{{<body>}}` directly. This prevents you from accidentally breaking the page formatting, or overriding Jira's CSS.

Note that if you're administrator has enabled the rich text editor, you'll still be able to format your content using wiki markdown, but if you select the [visual editor](#), you'll see the markdown applied directly.

Tracking issues with labels

Labeling helps you categorize and search for an issue. When viewing an issue, select **More > Labels** to add or remove labels, which will appear in the Details section:



You can click a label (e.g. **doc** in the above screenshot) to jump to the Issue Navigator and see a list of all issues that have this label. You can also add the [Labels Gadget](#) to your dashboard to quickly find issues with labels relevant to you and your team.

Watching and voting for issues

What permissions do you need?

To view other users watching or voting for an issue, you need the **View Voters and Watchers** and **Manage Watcher List** project permissions.

If your administrator has set up the needed notification scheme, you can select **Start watching this issue** to be automatically notified of issue updates. You can also click the number of watchers on the issue to add other Jira users as watchers.

If your administrator has enabled the voting on issues, you can select **Vote for this issue** to encourage the responsible team to resolve or complete the issue.

Reordering sub-tasks on an issue

If you've added sub-tasks to an issue, and need to reorder them, you can drag and drop them on the issue navigator view of the parent issue. If you're using a board in Jira Software, you can also reorder the sub-tasks on the board view. However, these two methods are **independent** of each other. Reordering sub-tasks on the parent issue **will not** reorder tasks on an existing board, and vice-versa.

Linking issues

Issue linking allows you to create an association between two existing issues on either the same or different Jira servers. For example:

- An issue may *relate to* another.
- An issue may *duplicate* another.
- An issue may *block* another.

Issue linking also allows you to:

- Create a new linked issue from an existing issue in a service desk or business project.
- Create an association between an issue and a Confluence page.
- Link an issue to any other web page.

Your Jira administrator can customize the types of links that you can create, see [configuring issue linking](#).

On this page:

- [Creating a link to another issue on the same Jira site](#)
- [Creating a link to an issue on another Jira site](#)
- [Create a new linked issue from an existing issue in a service desk or business project](#)
- [Creating a link to a Confluence page](#)
- [Creating a link to any web page URL](#)
- [Deleting a link](#)
- [Searching for linked issues](#)

Issue links within an issue look like this:

[Screenshot: the 'Issue Links' section within an issue](#)



Note: Resolved issues (i.e. issues with a Resolution set) are displayed in strike-through font, e.g. ~~DEMO-4~~.

To create links on issues, you need to have the Link Issues permission in the project(s) to which the issues belong.

Creating a link to another issue on the same Jira site

1. Open the issue you wish to link to another issue in the same Jira site.
2. Select **More > Link** to display the **Link** dialog box.

3. Ensure that the **Jira Issue** item is selected at the left of the dialog box and then choose the type of link to be created from the **This issue** drop-down list.
 - i** If your Jira system administrator has configured *fully reciprocal application links* between your Jira site and another one, a **Server** drop-down list may appear above the **This issue** list. If this is the case, ensure your Jira site appears or has been selected from the **Server** list.
4. In the **Issues** field, specify the issue(s) to be linked to your currently viewed/selected issue. There are two ways to do this:
 - Type the full issue key (e.g. **ABC-123**) — or to link to multiple issues, press the 'Enter' key between each typed issue key.
 - i** If you have previously browsed an issue, you can quickly find the issue by typing the first few letters of the issue key (or part of the Summary), which will appear in an 'autocomplete' drop-down list for selection:
 - OR:**
 - Click the **search for an issue** link to use the **Find Jira issues** popup, which allows you to perform either a simple **text search** or an **advanced search** for issues.
5. Optional: Add a **Comment** to describe why you are linking these issues.
6. Click the **Link** button at the bottom of the dialog.

Creating a link to an issue on another Jira site

 To create this type of link, your Jira system administrator should have configured *fully reciprocal application links* between your Jira site and the other Jira site containing the issue(s) you want to link to.

1. Open the issue you wish to link to another issue.
2. Select **More > Link** to display the **Link** dialog box.
3. Ensure that the **Jira Issue** item is selected at the left of the dialog box.

 **Note:**

- This option will not be available if your Jira system administrator has not configured an application link between your Jira site and the remote Jira site.
- If, after selecting this option, you are prompted for authorization, you may be required to log in to the remote Jira site, which will allow your Jira site to access the remote Jira site *on behalf of your account on the remote Jira site*.

 This behavior means the application links configured between your Jira site and the remote Jira site use OAuth authentication.

4. If your Jira site is connected to multiple remote Jira sites, choose the relevant Jira site from the **Server** drop-down list.
5. Choose the type of link to be created from the **This issue** drop-down list.
6. Type the **Issue** key of the issue on the remote Jira site that you want to link to. Alternatively, you can search for issues on the remote Jira site by clicking the **search for an issue** link, which opens the **Find Jira issues** popup.

 You can link to any issue on the remote Jira site to which you have access on that site.
7. Select the **Create reciprocal link** checkbox to create the complementary link on the remote issue you are linking to, back to your issue. For example, if you create a **blocks** link type to a remote issue, the reciprocal link generated on the remote issue will be a **is blocked by** link type back to your local issue.
8. Optional: Add a **Comment** to describe why you are linking these issues.
9. Click the **Link** button at the bottom of the dialog.

Troubleshooting

 **Problem:** If you selected the **Create reciprocal link** checkbox, but after clicking the **Link** button, you discover that a reciprocal link from the remote issue back to your issue has not been created, then your Jira system administrator has most likely created only a one-way link from your Jira site to the remote Jira site.

 **Solution:** Ask your Jira system administrator to configure *fully reciprocal application links* between your Jira site and the remote Jira site.

 **Problem:** If you attempted to create a reciprocal link but received the following message:

'A reciprocal link from issue 'XYZ-123' back to this issue was not created as the remote Jira server returned the following error: No Link Issue Permission for issue 'XYZ-123'.' (where 'XYZ-123' is the issue key on the remote Jira site),

then a reciprocal link on the remote Jira site will not have been created, because the user account through which you authenticated on the remote Jira site (at step 3 above) does not have the Link Issues project permission.

 **Solution:**

- Ask the Jira project administrator(s) on the remote Jira site to grant your user account the Link Issues project permission for the relevant project(s) to which you need to create issue links.
- Alternatively, if the application link between your Jira site and the remote Jira site use OAuth authentication and you suspect you may have authenticated on the remote site with another user account that does not have the Link Issues project permission, repeat the procedure above but during the authorization step (at step 3), authenticate on the remote site with a user account which has this permission.

 If you are not prompted for authentication during authorization, try clearing your browser's cookies first and repeat the procedure again.

Create a new linked issue from an existing issue in a service desk or business project

To create a linked issue, you need to have Create issue and Linked Issues permissions in the

destination project(s).

To create a linked issue:

1. Open the issue from which you wish to create the linked Jira issue.
2. In the Issue screen, select **More > Create linked issue** to display the **Create Linked Issue** dialog box

The newly created linked issue contains the same Project, Issue Type, and Summary information stored in the original issue. It is also linked to the service desk issue, in this case CTF-2.

The screenshot shows the 'Create linked issue' dialog box. It has the following fields and options:

- Project:** Charlie Travel Sydney (CTS)
- Issue Type:** Problem
- Created issue:** causes
- Linked issues:** CTF-2 (with a search bar below it: 'Search for issues to link to from the one you're creating.')
- Summary:** Fix software bug in app XYZ
- Description:** The print dialog is missing the orientation features. Unable to print to landscape.

At the bottom right, there are 'Create' and 'Cancel' buttons.

3. Select the destination **Project** in which the new linked issue is to be created.
4. Select the correct Issue Type for the new linked issue.
5. In the **Linked issues** field, specify issue(s) to be linked to your new linked issue.
6. Edit the linked issue **Summary**.
7. Edit the **Description** and describe why you are linking these issues.
8. Select the **Copy attachments** checkbox to include any attachments from the original issue.
9. Select the **Copy links** checkbox to include any URLs from the original issue.
10. Click the **Create** button at the bottom of the dialog.

Your linked issue has now been created.

Creating a link to a Confluence page

This feature is only supported in Confluence versions 4.0 or later.

! To create this type of link, your Jira system administrator needs to have configured an *application link* between your Jira site and the Confluence site containing the pages you want to link to.

1. Open the issue you wish to link to another issue.
2. Select **More > Link** to display the **Link** dialog box.
3. Click the **Confluence Page** option at the left of the dialog box.
 - i** This option is not available if your Jira system administrator has not configured an application link between your Jira site and Confluence site.
4. If more than one application link has been configured between your Jira site and other Confluence sites, then choose the appropriate Confluence site from the **Server** drop-down list.
5. Specify the Confluence page to be linked to your currently viewed issue. There are two ways to do this:

- In the **Page URL** field, enter the URL of a page on the Confluence site you want to link to. For example:

```
http://<confluence-server>/display/ds/Welcome+to+the+Confluence+Demonstration+Space
```

- Click the **search for a page** link. The **Link** dialog box is replaced by the **Find a Confluence page** dialog box.
 - If you are prompted for authorization, you may be required to log in to the Confluence site, which will allow your Jira site to access the Confluence site *on behalf of your account on the Confluence site*. This behavior means the application links configured between your Jira site and the remote Confluence site use OAuth authentication.
 - a. In the first **Search** field, specify one or more search terms that appear in the page you want to link to. This field is mandatory.
 - b. Optional: In the second **Search** field, select the Confluence space to further narrow down the search.
 - c. Click the **Search** button and then the title of the page you want to link to.
- 6. Optional: Add a **Comment** to describe why you are linking these issues.
- 7. Click the **Link** button at the bottom of the dialog.

Troubleshooting

✘ Problem: If Confluence page links you create show **Failed to load** on the issue or if you attempted to search for a Confluence page but received the following message:

'Content on the Confluence site could not be accessed because the Confluence server's 'Remote API' feature is disabled. The Confluence system administrator must enable this 'Remote API' feature for Jira to successfully access this content.'

then Jira was unable to communicate with the Confluence server to either:

- retrieve information about the link or
- conduct a Confluence page search in the **Find a Confluence page** dialog box.

✔ Solution:

Ask the Confluence system administrator to enable the **Remote API (XML-RPC & SOAP)** feature, since this Confluence feature is disabled by default. See [Enabling the Remote API](#) in the Confluence documentation for details.

Creating a link to any web page URL

1. Open the issue you wish to link to another issue.
2. Select **More > Link** to display the **Link** dialog box.
3. Click the **Web Link** option at the left of the dialog box.
4. Specify the **URL** of the web page you want to link to.
5. Specify the **Link Text** that will appear in the **Issue Links** section of the 'view issue' page and will be hyperlinked to your URL.
6. Optional: Add a **Comment** to describe why you are linking these issues.
7. Click the **Link** button at the bottom of the dialog.

Deleting a link

1. Go to an issue that contains links, and locate the **Issue Links** section.
2. Hover your mouse over the link you wish to delete, and click the **Delete** (trashcan) icon that appears.

Searching for linked issues

You can search for issues that are linked to a particular issue. See [Advanced searching](#) for more information.

• Be aware that this functionality does not extend to issues on a remote Jira server.

Editing multiple issues at the same time

At some point, you may need to change multiple issues at the same time. You can do this by performing a bulk operation.

There are restrictions placed on some of the bulk operations. For example, if you select multiple issues with different workflows, you can only transition them in groups with the same workflow, and one group at a time. The restrictions are explained further in the relevant sections.

On this page:

- [Before you begin](#)
- [Transition multiple issues](#)
- [Delete multiple issues](#)
- [Move multiple issues](#)
- [Edit multiple issues](#)
- [Watch / stop watching multiple issues](#)

Before you begin

Required permissions - To perform a bulk operation, you'll need the appropriate project-specific permission and the global Bulk Change permission. For example, you would need to have both the **Move Issue** and **Bulk Change** permissions to perform the **Bulk Move** operation.

Disable notifications for bulk operations - You can disable mail notifications for a particular bulk operation by deselecting the **Send Notification** checkbox in the bulk operation wizard. For this option to be available, you must be an administrator or project administrator of all the projects associated with your selected issues. Deselecting **Send Notification** only disables Jira notifications. It doesn't affect notifications that are sent to your service desk customers.

Disable customer notifications - Some bulk operations, such as **Change comment**, might trigger email notifications to your customers. To prevent a flurry of emails, a Jira admin can temporarily disable outgoing mail in



> **System** > **Mail** > **Outgoing mail**. This setting controls both Jira and customer notifications, so remember to turn it back on when you're done with your bulk edit.

Using the bulk change wizard - The bulk change wizard will progress you through your bulk change. To step back at any step of the operation, select the relevant step in the menu on the left-hand side. Selecting Cancel will cancel the entire process.

Transition multiple issues

This bulk operation allows you to transition multiple issues through a workflow at the same time. You can only perform one transition bulk operation at a time. You will also need to provide any values required to complete the transition. For example, to close multiple issues, you will need to provide a value for the Resolution field, such as Done, Fixed, or Won't Fix.

▼ [How to transition multiple issues](#)

1. Perform a search with the required filters to produce a list of issues.
2. Select **Tools** > **Bulk Change**.
3. Select the issues you'd like to perform the bulk operation on, and select **Next**.
4. Select **Transition Issues**, and select **Next**.

5. Select the available workflow action. The actions available are dependent on the issues (and their associated workflows) that you have selected. Select **Next**.
6. Select a value for any required fields for this transition, and if available, decide whether you'd like to send email notifications. Select **Next**.
7. Review your bulk operation, and select **Confirm** when you are happy with the operation.

Delete multiple issues

This bulk operation allows you to delete multiple issues at the same time.

▼ How to delete multiple issues

1. Perform a search with the required filters to produce a list of issues.
2. Select **Tools > Bulk Change**.
3. Select the issues you'd like to perform the bulk operation on, and select **Next**.
4. Select **Delete Issues**, and select **Next**.
5. If available, decide whether you'd like to send email notifications. Select **Next**.
6. Review your bulk operation, and select **Confirm** when you are happy with the operation.

Move multiple issues

This bulk operation allows you to move multiple issues at the same time. The issues you're moving need to be mapped to both a project and an issue type, and in doing this, you may need to also map the status and fields of the issues. Subtasks need to be mapped, too.

▼ How to move multiple issues

1. Perform a search with the required filters to produce a list of issues.
2. Select **Tools > Bulk Change**.
3. Select the issues you'd like to perform the bulk operation on, and select **Next**.

▼ More information...

The bulk move operation can be performed on both standard issues and sub-task issues. Standard issues can be moved to another project and issue type, whereas a sub-task can only have its issue type changed. (Note that it is possible to convert a sub-task to an issue, and vice versa.)

It is **not** possible to select *both* a sub-task and its parent to bulk move. This is so as to adhere to the parent/sub-task relationship (i.e. the sub-task is always located in the same project as the parent issue). Any sub-tasks of selected parent issues that were also selected will be automatically discarded from the move.

For example, you have issue B being a sub-task of issue A and you try to bulk move both A and B simultaneously. You will see a warning message (see below) and will be prompted to select a target project and issue type for issue A. If you select a new project for A, you will be prompted to move the sub-task to a new issue type based on issue A's new project. If you *don't* change the project for issue A, the sub-task will not be required to be moved.

4. Select **Move Issues**, and select **Next**.

The bulk move operation may require additional information dependent on which issues you have selected to move. This information is requested as follows:

- a. Select Projects and/or Issue Types

▼ More information...

The first step of the Bulk Move wizard is to choose which projects and issue types you will move your issues to. The target project and issue type will determine whether extra steps will be required to migrate statuses and fields.

Selected issues are grouped by their current project and issue type. You can either select a new project and issue type for each one or choose to move all standard issues to a single project and issue type.

i Note: This *does not apply to sub-tasks* since they cannot be moved to a standard issue type.

- b. Select Projects and/or Issue Types for Sub-Tasks

▼ More information...

If you are moving issues with sub-tasks to another project, you will also need to move the sub-tasks to the new project. You can also elect to change the issue types of the

sub-tasks being moved if you need to.

c. Select status migration mappings for invalid statuses

▼ [More information...](#)

As multiple workflows can be active simultaneously, some statuses associated with the collection of selected issues may not be valid in the target workflow. In this case, you should map invalid statuses to valid statuses in your new workflow.

d. Select values for required fields and fields with invalid values

▼ [More information...](#)

In order to adhere to the field configuration scheme associated with the target project and issue type, it may be necessary to update/populate required fields (e.g. fields that are required in the target project, but may not have been in the original project).

For each field that needs to be populated, you will be prompted to supply a value. This value will be applied to all issues that are being *Bulk Moved* together.

For the following fields, you can select from a list of possible values provided for you:

- Component
- Affects Version
- Fix Version
- Custom fields of type 'Version-Picker'

Note that versions which have been archived in the target project cannot be selected as the target when performing a bulk move. If you need to move issues into an archived version, you will need to first unarchive the version in the target project.

It is possible to retain original field values that are valid in the target destination by checking the **Retain** checkbox associated with the field. For example, some issues may already include a valid custom field value — these values can be retained, while issues that require an update will adopt the value specified on the **Field Update** screen.

- **Checked:** the original value is retained where possible¹. The field will not be updated with the specified new value.
- **Unchecked:** all fields will be updated with the specified new value.

Note that the '**Retain**' checkbox is not available for the following fields, since an explicit mapping is required:

- Component
- Affects Version
- Fix Version
- Custom fields of type 'Version-Picker'

2. Confirm changes to be made and complete the operation

▼ [More information...](#)

When all move parameters — e.g. target project, status mappings and field updates — have been specified for all issues, you will be presented with a confirmation screen displaying all changes that will be made to the issues being moved. The following details are displayed as applicable:

- **Issue Targets:** the target project and issue type
- **Workflow:** the target workflow and invalid status mappings
- **Updated Fields:** new values for fields that require updating
- **Removed Fields:** values to be removed in fields that are not valid in the target

The issues will only be moved once the **Confirm** button is clicked from the confirmation page. If the operation is exited anytime before this step, no changes will be made to the issues.

Note that steps C and D above will occur once for each different target project and issue type combination.

Edit multiple issues

This bulk operation allows you to edit multiple issues at the same time. The bulk edit operations available

depend on the issues selected and the nature of the field/s you want to change.

▼ [How to edit multiple issues](#)

1. Perform a search with the required filters to produce a list of issues.
2. Select **Tools > Bulk Change**.
3. Select the issues you'd like to perform the bulk operation on, and select **Next**.
4. Select **Edit Issues**, and select **Next**.
5. Select the bulk edit operation from the list of available operations (expand more information for a full list of available and unavailable operations, and their conditions).

▼ [More information...](#)

Available Operations	Conditions
Change Affects Version/s	<ul style="list-style-type: none"> • Selected issues belong to one project, and that project has version/s • This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues
Change Assign To	<ul style="list-style-type: none"> • This field is not hidden in any field configurations the selected issues belong to • Current user has 'assign issue' permission for all the selected issues
Change Comment	<ul style="list-style-type: none"> • This field is not hidden in any field configurations the selected issues belong to • Current user has 'comment issue' permission for all the selected issues
Change Component/s	<ul style="list-style-type: none"> • Selected issues belong to one project, and that project has component/s • This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues
Change Due Date	<ul style="list-style-type: none"> • This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues • Current user has 'schedule issue' permission for all the selected issues
Change Fix For Version/s	<ul style="list-style-type: none"> • Selected issues belong to one project, and that project has version/s • This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues
Change Issue Type	<ul style="list-style-type: none"> • Current user has 'edit issue' permission for all the selected issues
Change Priority	<ul style="list-style-type: none"> • This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues
Change Reporter	<ul style="list-style-type: none"> • This field is not hidden in any field configurations the selected issues belong to • Current user has 'edit issue' permission for all the selected issues • Current user has 'modify reporter' permission for all the selected issues
Change Security Level	<ul style="list-style-type: none"> • This field is not hidden in any field configurations the selected issues belong to • All the selected projects are assigned the same issue level security scheme • Current user has 'edit issue' permission for all the selected issues • Current user has 'set issue security' permission for all the selected issues

Change Custom Fields	<p>The 'Change Custom Fields' operation is available only if:</p> <ul style="list-style-type: none"> • a global custom field exists OR • an issue type custom field exists and the issues are all of this specific issue type OR • a project custom field exists and the issues are all of the same project
Edit a Closed Issue	<ul style="list-style-type: none"> • Your workflow must allow editing of closed issues
Change Sprint	<p>You need to specify the sprint ID.</p> <ul style="list-style-type: none"> • This operation only affects active and future sprints, i.e. closed/completed sprints are not included when bulk editing the Sprint field.

Unavailable Operations

The fields listed in this section have no operations for bulk editing. This is because there is an alternative method or it is not logical to perform bulk edit on them.

The following system fields are unavailable for bulk editing:

- Attachments
- Summary
- Description
- Environment
- Project — Please use 'Bulk Move' to move issues between projects
- Resolution — Please use 'Bulk Workflow Transitions' to modify the resolution of issues
- Time Tracking fields — Original Estimate, Remaining Estimate, Time Spent

The following custom field types are unavailable for bulk editing:

- Import Id
- Read Only Text

6. Select a value for any required fields for this operation, and if available, decide whether you'd like to send email notifications. Select **Next**.
7. Review your bulk operation, and select **Confirm** when you are happy with the operation.

Watch / stop watching multiple issues

These bulk operations allows you to start watching or stop watching multiple issues at the same time.

▼ [How to watch multiple issues](#)

1. Perform a search with the required filters to produce a list of issues.
2. Select **Tools > Bulk Change**.
3. Select the issues you'd like to perform the bulk operation on, and select **Next**.
4. Select **Watch Issues**, and select **Next**.
5. Review your bulk operation, and select **Confirm** when you are happy with the operation.

▼ [How to stop watching multiple issues](#)

1. Perform a search with the required filters to produce a list of issues.
2. Select **Tools > Bulk Change**.
3. Select the issues you'd like to perform the bulk operation on, and select **Next**.
4. Select **Stop Watching Issues**, and select **Next**.
5. Review your bulk operation, and select **Confirm** when you are happy with the operation.

Moving an issue

Sometimes, an issue may belong to a different project, and you may want to move this issue to another project. You can easily do this by using the **Move Issue** wizard.

Before you begin:

- You must have the Move Issues permission for the project that has the issue that you want to move.

- You must have the Create Issues permission for the project that you wish to move your issue to.

If you do not have either of these permissions, please contact your Jira administrator to have these added to your user profile.

If you wish to move multiple issues between projects at the same time, please refer to the documentation on [bulk moving issues](#).

Moving an issue

The **Move Issue** wizard allows you to specify another project in your Jira instance to move your selected issue to. As there may be significant differences in the configuration of your original project and target project, the **Move Issue** wizard allows you to change certain attributes of the issue. These include:

- **Issue Type** — If your issue is a custom issue type that does not exist in your target project, you must select a new issue type. You can also choose to arbitrarily change the issue type.
- **Issue Status** — You may have set up custom issue statuses as part of a workflow. If you have assigned a custom status to your issue, and it does not exist in your target project, you must select a new issue status for your issue. You cannot arbitrarily change the issue status, i.e. the option to change the issue status will only appear if you are required to change it.
- **Custom Fields** — If you have defined **required** custom fields for your issue that do not exist in your target project, you must set values for them. You will only be prompted to enter the values for **required custom fields** in the target project that are missing values. If the custom fields of your original project also exist in your target project, and these custom fields are not required in the target project, you may need to set values for them, to move the issue successfully. If you wish to change the existing values for other fields on your issue, you can do this after the move is complete.

To move an issue:

1. View the issue that you wish to move.
2. Select **More > Move**.
3. The first page of the **Move Issue** wizard is displayed. Complete the steps required.
4. The confirmation page will display with all of your changes. If you wish to revise any of your changes, you can click the appropriate step in the left-hand menu to return to that page of the wizard. Once you are happy with your changes, click **Move** to move the issue to the target project.
5. Your issue will be moved to the target project and displayed on screen. You can now edit the issue to make further changes, if you wish.

Moving related issues

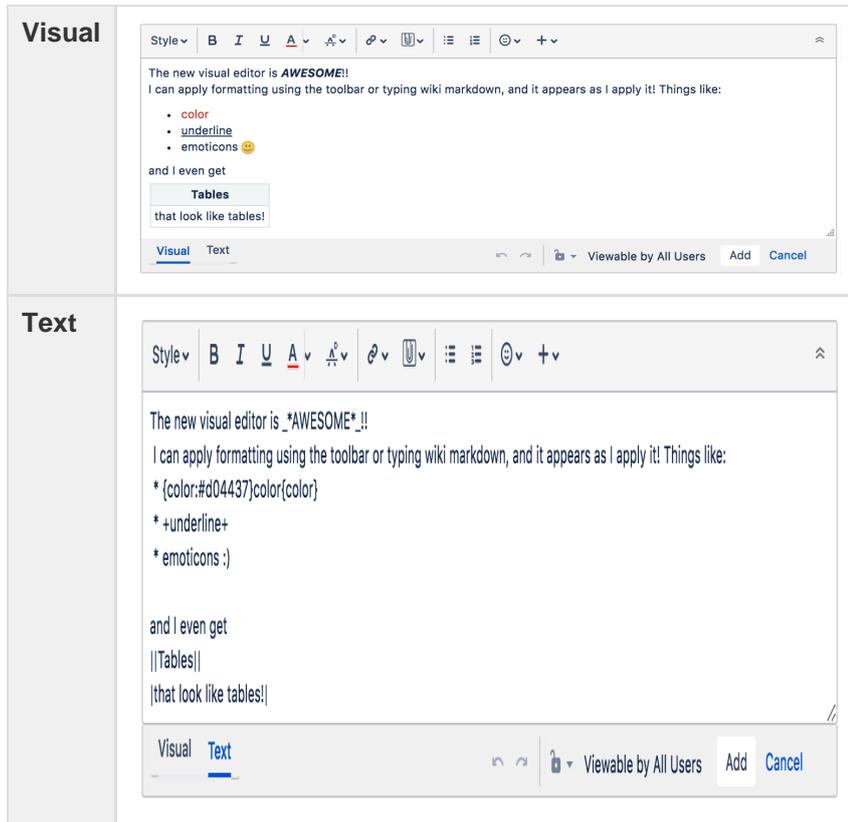
- If your issue has sub-tasks, the 'Move Issue' wizard will also move the sub-tasks to the target project.
- If you are moving an epic, the 'Move Issue' wizard will not move the issues in the epic. The epic and the issues in the epic will still be linked to each other, but the issues in the epic will remain in the original project. You will need to move them separately.

Troubleshooting

- Restricted comments appear to be removed after moving the issue. See this article: [Restricted comments disappear after moving an issue to a new project](#).

Visual editing

Formatting content in Visual mode gives you a What You See Is What You Get (WYSIWYG) experience. Formatting appears as you apply it, and you no longer have to flip to a Preview to see what your content will look like when saved. You still have the option to view the wiki markup by selecting the Text tab. You'll know you have access to the visual editor because you'll see the Visual and Text tabs.



In Visual mode, you can still enter wiki markup syntax as you add your content, and it'll be rendered exactly as it'll display when you save. You can even flip between modes to view the formatted content, and the wiki markup syntax. You can also use the toolbar to format and style your content.

As Visual editing is really a preview of what we're working on, there's a few things that may not work quite as you'd expect them:

- Formatting content in a complex way can affect it's ability to be rendered, things like tables in the cells of other tables, and adding images to table cells won't work.
- Pasting content may not work as expected, as the source content may really be formatted using a method we don't support. So pasting tables may work, and it may not, depending on the source. Pasting plain text is absolutely fine.
- If you have macros provided by 3rd party apps, and they're incompatible with Visual mode, you won't be able to edit the macro header, and it's content will be rendered as text (wiki markup).

Scheduling an issue

You can schedule issue due dates in Jira Service Desk to help your agents prioritize incoming customer requests and find overdue issues that need urgent attention. The powerful scheduling feature allows you to perform fixed and relative date searches based on specific due dates as well as arbitrary search periods. You can also perform advanced searches using Jira Query Language.

Scheduling an issue

To schedule an issue, populate its **Due** date field. This can be done either when creating an issue, or at a later stage by editing the issue.

To enable Issue Scheduling, at least one group or project role must be given the Schedule Issues permission by your Jira administrator. Only users with the Schedule Issues permission can populate the **Due** date field.

Searching by due date

You can use either [basic search](#) or [advanced search](#) to search for issues by their Due Date.

Using simple search

You can search for issues using the search form in Issue Navigator (see [Searching for Issues](#)). There are two ways to search for issues based on the **Due** date field. The first way is using fixed date values, the second is using periods that are relative to the current date.

Fixed date searches

There are two text fields in the search form that allow searching based on the **Due** date field.

- To search for all issues that are due after a certain date, enter the date in the Due After text field. For example, to find all issues that are due after 1st June 2010, enter 1-6-2010 in the Due After field. You can also use the Calendar popup to select a date by clicking the calendar icon to the right of the field.
- To search for issues that are due before a certain date, enter the date in the Due Before text field. For example, to find all issues that are due before 1st July 2010, enter 1-7-2010 in the Due Before field.

To search for issues that are due between two dates, populate both the Due After and the Due Before fields.

Relative period search

It is possible to perform a search that is relative to the time when it is run. For example, it is possible to do a search for issues that are due seven days from now. To do this, enter 7d in the Due Date To text field of the Issue Navigator. If the search is saved and run the next day, the issues that are due in seven days from the time that the search is run will be retrieved. Thus, this search will find all issues that are due within a week every time it is run.

The values that are entered in the Due Date From and Due Date To fields have to conform to a special syntax (described below). However, it is also possible to use the Due Date popup by clicking the icon to the right of the Due Date To text field to specify the search period.

Due Date Popup

Use the Due Date popup to do the following:

- To search for issues that are overdue at the time of the search, select the first radio button, and click **OK**.
- To search for issues that are overdue by more than a certain number of days, populate the text field in the second row, and click **OK**.
- To search for issues that are due in the next certain amount of days, and are not overdue at the time of the search, populate the text field in the third row with the number of days, and choose **and not** from the select box in the third row. Select the third radio button, and click **OK**.
- To search for issues that are due in the next certain amount of days, and are overdue at the time of the search, populate the text field in the third row with the number of days, and choose **and** from the select box in the third row. Select the third radio button, and click **OK**.
- The fourth row of the popup is used for arbitrary period searches. Use the **to** text field to specify the upper bound of the search, and the **from** text field to specify the lower bound of the search. A blank text field means no bound. Populating the text fields in the fourth row actually has the same effect as populating the Due Date From and Due Date To text boxes. The syntax is described below.

Relative Period Search Syntax

The Due Date From and Due Date To fields use a special syntax to denote time period bounds. The syntax uses numbers and abbreviations that follow the numbers to represent what the numbers actually mean. The abbreviations are "w" for weeks, "d" for days, "h" for hours, and "m" for minutes. For example, to specify 10 days in the future, use "10d" or "1w and 3d". To specify a period bound in the past, prefix the value with the "-" sign. For example, to specify 2 days, 4 hours, and 3 minutes ago, use "-2d 4h 3m".

Using advanced search

You can also use Jira Query Language (JQL) to search for issues by due date — see [Advanced Searching](#), and particularly the documentation on the Due field.

Logging work on issues

In Jira Service Desk, you use Service Level Agreements (SLAs) configured by your administrator to help you track how well you're meeting customer expectations (e.g. responding to a request within 4 hours). You can use the Time Tracking feature in addition to SLAs to generate a workload report

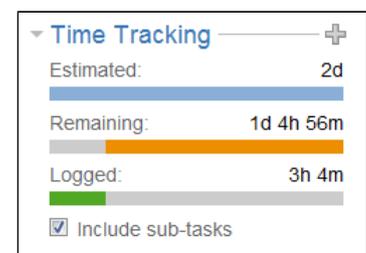
when you're working on a customer request with other agents, or when you need to track time spent fixing a problem that affects multiple customer requests.

On this page:

- Before you begin
- Setting a time estimate for an issue
- Logging work on an issue
- Editing a work log entry
- Deleting a work log entry
- Customized Jira installations

Here's how time tracking appears on an issue:

- The Estimated field displays the amount of time originally anticipated to resolve the issue
- The Remaining field displays the amount of time currently anticipated to resolve the issue
- The Logged field displays the amount of time logged working on the issue so far
- Choosing to include sub-tasks displays the aggregated time of an issue and all its sub-tasks



When you log time for the first time, the time spent is subtracted from the original estimate, and the resulting value is automatically presented in the remaining estimate. When subsequent work is logged, any time spent is subtracted from the remaining estimate.

Before you begin

- Make sure your Jira administrator has enabled the [Time Tracking](#) feature.
- Make sure you have the Work on Issues, Delete Work Logs, and Edit Work Logs project permissions.

Note that anyone with the Browse Project permission can view time tracking information on an issue.

Setting a time estimate for an issue

Teams can set a time estimate for an issue in order to calculate how long it will take to solve the issue.

1. Open the issue and select **Edit**.
2. Scroll down the Edit issue window to fill in the following time tracking fields:

Field	Description
Original Estimate	Amount of time you believe is required to solve the issue. If you want to change original estimate values once they have logged work time, ask your Jira administrator to disable legacy mode on time tracking.

Remaining Estimate	Amount of time you believe is required to solve the issue in its current state.
--------------------	---

If the Jira time tracking feature is in legacy mode, you will only see the original estimate field if work has not been logged. Once work time has been logged, you will only see the remaining estimate field.

Tips:

- You can specify additional time units after a time value 'X', such as Xw, Xd, Xh, or Xm, to represent weeks (w), days (d), hours (h), and minutes (m), respectively. If you type a number without specifying a time unit (e.g. if you type '2' instead of '2h'), the default time unit that your Jira administrator specified will apply.
- Default conversion rates are 1w = 5d and 1d = 8h.

3. Select **Update**.

When work is first logged against the issue, the **Time Spent** is subtracted from the **Original Estimate**, and the resulting value is automatically presented in the **Remaining Estimate**. When subsequent work is logged, any **Time Spent** is subtracted from the **Remaining Estimate**.

Additionally, once work has been logged on an issue, various reports based on the time tracking information become available.

Logging work on an issue

Once you have started to work on a specific issue, you can log your work by following these steps:

1. Select the issue you want to log time on.
2. Go to **More > Log Work**.
3. Fill in the following **Log Work** fields, and select **Log**:

Log Work field	Description
Time spent	The amount of time spent on the issue. This is the aggregate amount of time that has been logged against this issue.
Date started	Date and time when you started this unit of work.
Remaining estimated	Amount of time anticipated to resolve the issue after completing this unit of work. You can adjust this value using the following options: <ul style="list-style-type: none"> • Adjust Automatically - Adjust the remaining estimate value by subtracting the amount of work logged in the Time Spent field from the remaining estimate current value. • Leave Estimate unset - This option is displayed only if no time estimate has been specified on the issue. You can use this option when you want to keep track of work, but you don't necessarily have a time estimate for an issue. • Use Existing Estimate of - Select this option if you do not want to change the issue remaining estimate value. • Set to - You can adjust the remaining estimate value to the amount of time you specify in this field. • Reduce by - Select this option to manually adjust the remaining estimate value by subtracting the amount of time you specify in this field.

Work description	<p>Type a description related to the achieved work.</p> <p>Comments are copied to the Workflow Description by default, but your Jira administrator can change this option in the 'Copy Comment to Workflow Descriptions' settings. If this setting is disabled:</p> <ul style="list-style-type: none"> • The work log entry may be visible to anyone. If this is a concern, you need to edit this work log entry after creating it to modify its visibility. • You have to manually copy comments to a workflow description once you have logged work.
------------------	--

You can also log work while resolving or closing an issue by closing it and editing the log work fields. Select the padlock icon to set the work logged to be viewable only by members of a particular project role or group.

Editing a work log entry

You can edit your own work log entries if you have been granted the Edit Own Work Logs permission. You can also edit other people's work log entries if you have been granted the Edit All Work Logs permission.

Deleting a work log entry

You can delete your own work log entries if you have been granted the Delete Own Work Logs permission. You can also delete other people's work log entries if you have been granted the Delete All Work Logs permission.

1. Go to the desired issue, and open the **Work Log** tab.
2. Hover over the work log entry to display the actions for the entry on the right side.
3. Select the entry you want to delete, and click the trash can icon. You will be prompted to choose how the Remaining Estimate is affected by deleting the work log:

Option field	Description
Auto adjust	Choose this option to automatically add the time spent value to the current remaining estimate value.
Leave existing estimate	Select this option if you do not want to change the issue remaining estimate value.
Set estimated time remaining	Choose this option to manually set the issue's remaining estimate value to the specified amount.
Increase estimated time remaining	Select this option to increase the estimated remaining.

4. Click **Delete**.

Customized Jira installations

Jira applications can be customized by your Jira administrator by adding the Log Work and Time Tracking fields to the customized screens. This way, you can log work and specify time estimates on the same Jira screen when performing any Jira operation, such as editing, creating an issue, or transitioning an issue to another status.

If you want to work *and/or* specify time estimates on the same Jira screen:

1. Navigate to the issue and view its details.
2. Perform the customized Jira operation that allows you to log work *and* specify time estimates on the same Jira screen. For example, assuming that your Jira administrator has added the **Time Tracking** fields to the **Resolve Issue Screen**, and assuming this screen also retains the default **Log Work** fields, select **Workflow > Resolve Issue** at the top of the issue.

- If your Jira administrator has configured the Log Work fields as optional, then you can choose whether or not to log work by checking the Log Work checkbox.
- If your Jira administrator has made logging work mandatory, you will not see the Log Work checkbox, and will instead need to log work when transitioning an issue.

Approving a service desk request

Jira Service Desk projects have an option to include an approval step and assign approvers to their issues. You may be asked to approve a service desk request if you've been assigned the role of "approver". When this happens, you'll receive an email notifying you that your approval is required. If the **Request Details** and **Approval buttons** variables have been added to the project's approval notifications template, you can action the request from within the email. If not, there will be a link to the customer portal where you can view and action the request.

How to approve and decline from the customer portal

Pro tip: Ask your Project Administrator to add **Request details** and **Approval buttons** to your approval notifications template, so you can quickly action a request from within your email. See [Managing service desk notifications](#).

It's likely that you've upgraded from a version of Jira Service Desk prior to 3.12, and the approval notifications template hasn't been updated.

1. Navigate to the service desk customer portal by either selecting the link in your email, or entering the URL.
2. View the approval request and review the supporting information.
3. Leave a comment by selecting **Add** below the comment field. You don't have to leave a comment, but if you're declining a request it helps to let the customer know why you declined it.
4. Select **Approve** or **Decline**.
5. The customer will receive a notification of the selected action you have taken.

The screenshot displays the Jira Service Desk customer portal interface for an approval request. At the top, the breadcrumb navigation shows 'Help Center / Infrastructure Service Desk / ISD-6'. The request title is '3 day trip to Brisbane' with a 'WAITING FOR APPROVAL' badge. Under 'Your approval', there are 'Approve' and 'Decline' buttons, and a comment field with a placeholder 'Comment on this request...'. The 'Activity' section shows a status update: 'Request requires approval. 1 approval needed. Today 1:00 PM LATEST'. The 'Approvals' section lists 'Elaine Goulding' as the creator and 'Frank Smith' as the approver, with a 'WAITING FOR APPROVAL' badge and a '1' indicator. A blue circle with the number '2' points to the 'Activity' section, and a blue circle with the number '1' points to the 'Approvals' section.

1. **Approvals section**, which lists the appointed approvers for the request. This section only appears when there are pending approvals for the request.
2. When a request is approved, the Approvals section disappears, and details about the approval will be added to the **Activity** section.
For example, Your request was **APPROVED** and the status changed to **Waiting for support**.

How to approve and decline from email

Good news! You're using the new default approval notifications template.

1. View the approval request and decide what action you'd like to take.
 2. Select either the **Approve** or **Decline** button within the email. If you aren't logged into the customer portal, you will be prompted to do so. You'll then see:
 - a. If you hit **Approve**, a confirmation screen.
 - b. If you hit **Decline**, a screen prompting you to leave a comment and confirm your decision. You don't have to leave a comment, but if you're declining a request it helps to let the customer know why you declined it.
 3. The customer will receive a notification of the selected action you have taken.
-
1. **Approval buttons**, that let the approver action requests from within their email.
 2. **Request details**, that show the full details of a request (incl request type, summary, creation date, and the same fields chosen in the request type settings).

If you're the only approver required on the request, and you approve it, the request will be moved to the status defined in the workflow for the approve transition. If there is more than one approval required, the status will remain the same until all approvers have responded, and your approval will be noted on the request.

If you decline a request (or any of the approvers decline it), it's automatically moved to the status as defined in the workflow for the decline transition, and your response is noted on the request.

Customizing the issues in a project

Issues are the packets of work that need to be completed in a project. These issues are made up of issue fields, and the issue fields contain data about the issue. This data is important, as it helps define the issue, and can contain important information about the issue, such as a summary, a description, due dates, and when and where the work is required. This information is presented on a screen. A screen groups all available fields (or a subset of all available fields) defined in Jira applications, and organizes them for presentation to a user. Jira Service Desk allows you to customize the configuration and behavior of issues to better suit the needs of your customers and agents. You may choose to:

- Change a field's behavior (such as change a field's description, make a field hidden or visible, or make a field required or optional)
- Add your own values for fields that have default values assigned (e.g. Resolution and Status)
- Create new 'custom' fields
- Configure different renderers for (some) fields
- Change the position fields on a screen
- Choose which screen should be displayed for each issue operation (e.g. 'Create Issue', 'Edit Issue') or workflow transition (e.g. Resolve Issue, Close Issue)

A simple example of how customizing an issue could benefit your team could be marking fields as 'Required' when an issue is created. This would ensure you always capture the required information you need to get the work done to resolve the issue. If you couple this with positioning the required fields at the top of the screen, and even hiding fields you know the issue creator won't use, you'll make sure your users can see and complete the required fields as quickly as possible.

You can make this...

Project **Development Strategy (DSE)**

Issue Type **Task**

Field Tab **Group**

Summary

Account **Please select**
Tempo Account custom field

Due Date

Priority **Medium**

Environment

For example operating system, software platform and/or hardware specifications (include as appropriate for the issue).

Description

Create another **Create** Cancel

into this...

Project **Development Strategy (DSE)**

Issue Type **Task**

Summary

Environment

For example operating system, software platform and/or hardware specifications (include as appropriate for the issue).

Description

Create another **Create** Cancel

Jira administrators and project administrators have different permissions when it comes to customizing issues.

Project administrators

As a project administrator, you're able to customize some aspects of the issues in your project if:

- you have the *Extended project administration* permission, which is enabled by default (you can check that in **Project settings > Permissions**),
- the screens you make the changes on must not be used by other projects, or used as a transition screen in a workflow.
- the screen isn't the default Jira screen (no-one can edit these screens).

If a screen is shared by another project, you'll see this information when you view the screen. If your screen suits the criteria above, you can:

- add and remove tabs that will appear on a screen, and edit the name of the tab,
- add, remove and rearrange system fields, and
- add, remove and rearrange existing custom fields, but you can't create custom fields.

Editing a screen

1. Select **Projects** and choose the project whose screen/s you want to view.
2. Select **Project settings** in the sidebar.
3. Select **Screens** in the Project settings sidebar.

You'll see the screen scheme/s used by your project, and the issue types that use that scheme. Expand the screen scheme to see the screens associated with that scheme and issue types. Select the screen you wish to edit. You can perform the following actions:

Action	Instructions
Add a tab	Click Add Tab  . Enter the name of the new tab in the dialog that appears and click Add .
Move a tab	Hover over the dotted part of the tab (next to the tab name) and drag the tab to the desired position.
Rename a tab	1. Hover over the tab name and click the pencil icon . 2. Enter the new name and click OK .
Delete a tab	Hover over the tab name and click the X .
Add a field	1. Click the tab that you want to add the field to. 2. Type the name of the field in the drop-down displayed at the bottom of the current fields. Field suggestions will appear as you type. 3. Click Add Field to add it to the current tab.
Move a field	Hover over the dotted part of the field (next to the field name) and drag the field to the desired position. Move a field to a different tab by dragging it to the name of the tab and dropping it.
Remove a field	Hover over the field and click the X that appears.

Jira administrators

As a Jira administrator, you can view more conceptual information on [customizing issues](#) in the Jira administrator's documentation.

Searching for issues

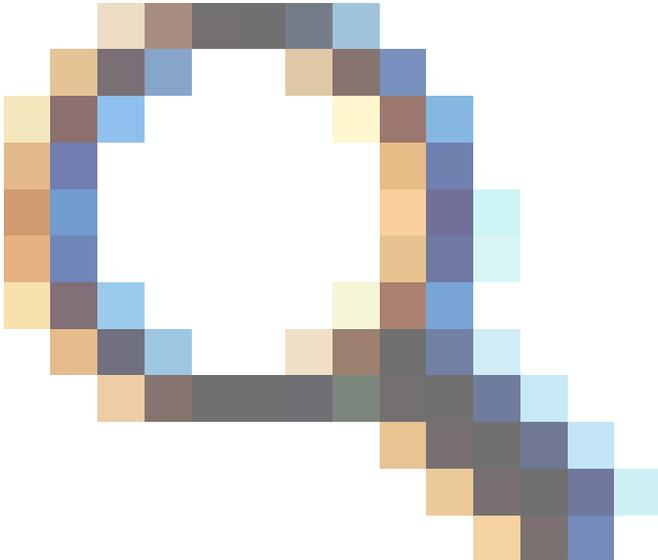
Can't find the customer issue you've been working on? This page will show you how to search for issues in Jira Service Desk. Any agent can search for issues, although they will only see results from projects they have access to. You'll find a step-by-step guide below that will show you how to run a search and use the search results. If you want more details on anything described on this page, see the related topics at the bottom of the page.

On this page:

- [1. Define your search criteria](#)
- [2. Change your view of the search results](#)
- [3. Working with the search results](#)
- [4. Save your search](#)
- [Next steps](#)

1. Define your search criteria

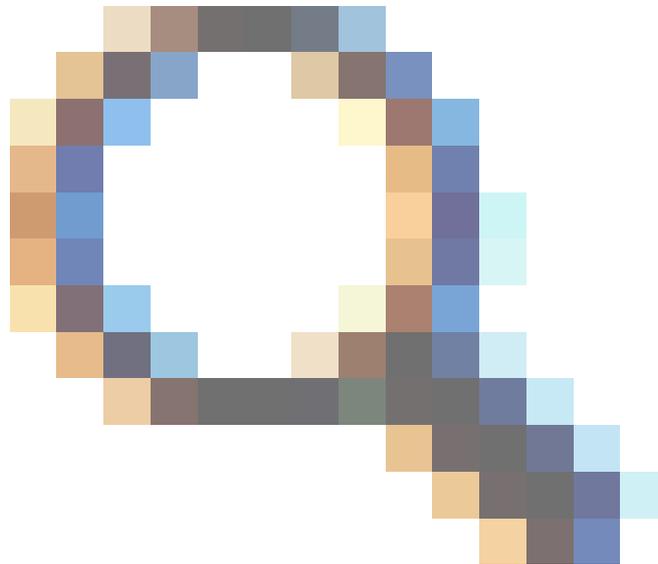
The first step in searching for issues is to define the criteria for your new search. You can define your search criteria in three different ways: using the quick search, using the basic search, or using the advanced search.

Quick search	<p>The quick search is the fastest way to define search criteria. However, it is less precise than other complex queries (e.g. <code>project = Jira AND status = Open AND priority = High</code>). If your search criteria is not complex, for example, you know the project key and some key words for</p> <p>To use the quick search: Enter your search criteria in the search box in the header bar of Jira a</p> <p><i>Tip: If you know the issue key or project key, enter it before other search terms, e.g. "JIRA help lir</i></p>
Basic search	<p>The basic search is more precise than the quick search, but easier to use than the advanced search. It has a user-friendly interface that lets you define complex queries, without needing to know how to use advanced searching).</p> <p>To use the basic search: Navigate to Issues (in header) > Search for issues, then enter your search criteria</p> <p><i>Tip: If the advanced search is shown instead of the basic search, click Basic next to the</i></p>  <p><i>icon.</i></p> 

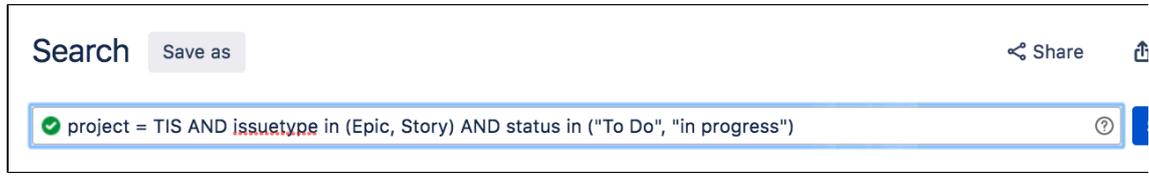
Advanced search

The advanced search is the most powerful of the three search methods. You can specify criteria in the other searches (e.g. ORDER BY clause). However, you need to know how to construct the Jira Query Language (JQL) to use this feature.

To use the advanced search: Navigate to **Issues** (in header) > **Search for issues**, then enter y
*Tip: If the basic search is shown instead of the advanced search, click **Advanced** next to the*



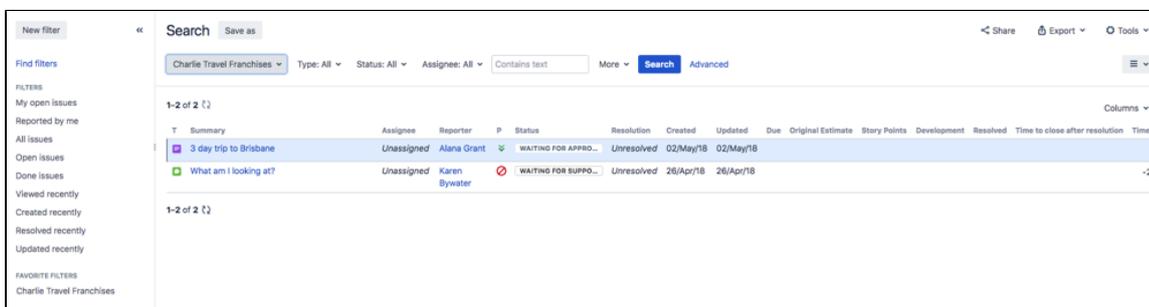
icon.



2. Change your view of the search results

You have crafted the perfect search criteria and run the search. Your search results will be displayed in the issue navigator. The issue navigator allows you to change how the search results are displayed. For example, you may want to bring high priority issues to the top or hide certain fields.

- **Change the sort order:** Click the column name.
- **Show/hide columns:** Click **Columns** and choose the desired columns.



3. Working with the search results

You've got the search results displaying the way that you want. Now you can work with the actual issues in the search results. The issue navigator lets you action individual issues, as well as the entire set of issues returned by your search.

Individual issues:

- **View the issue:** Click the issue key or name.

- **Action individual issues:** Click the cog icon next to the issue row and select an option.

All issues in the search results:

- **Export the search results to different formats, like Excel and XML:** Click **Export** and select the desired format.
- **Share the search results:** Click **Share**, then enter the recipient's details.
- **Create an RSS feed:** Click **Export > RSS (Issues)** or **RSS (Comments)**.
- **Bulk modify issues in search results:** Click **Tools** and select **all <n> issue(s)** under **Bulk Change**.

4. Save your search

If you frequently run the same search, you can save the search criteria as a filter. This saves you from having to manually redefine the search criteria every time. Jira applications also include a number of predefined system filters for common queries, such as 'My Open Issues', 'Reported by Me', 'Recently Viewed', and 'All Issues'.

To save your search as a filter: On the search results page, click **Save as** and enter a name for the filter. Your new filter will be shown in the left panel with your other favorite filters, filters shared with you, and the system filters. To run a filter, just click it.

Next steps

Read the following related topics:

- [Quick searching](#)
- [Basic searching](#)
- [Advanced searching](#)
- [Saving your search as a filter](#)
- [Working with search results](#)

Basic searching

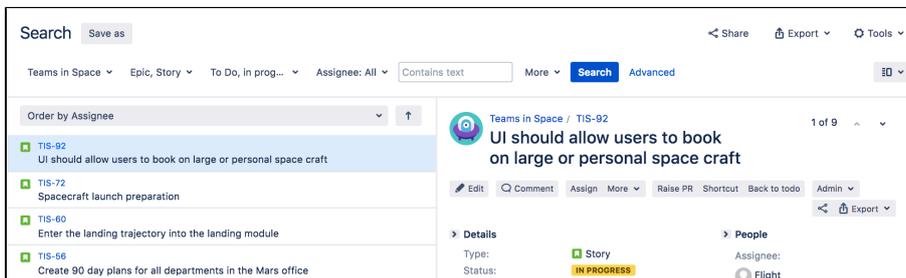
The basic search provides a user-friendly interface that lets you define complex queries, without needing to know how to use JQL (advanced searching).

- If you don't have complex search criteria, you may want to use [quick search](#) instead.
- If you are comfortable with the Jira Query Language (JQL), you may want to use [advanced search](#) instead. This search is more powerful than than the basic search.

On this page:

- [Basic searching](#)
- [Running a saved search](#)
- [Troubleshooting](#)
- [Next steps](#)

Screenshot: *Basic search*



Basic searching

1. Choose **Issues > Search for issues**.

- If there are existing search criteria, click the **New filter** button to reset the search criteria.
- If the advanced search is shown instead of the basic search, click **Basic** (next to the **Search** button).
- [Why can't I switch between basic and advanced search?](#)
In general, a query created using basic search will be able to be translated to advanced search, and back again. However, a query created using advanced search may not be able

to be translated to basic search, particularly if:

- the query contains an OR operator (note you can have an IN operator and it will be translated, e.g. `project in (A, B)`)
 - i.e. even though this query: `(project = JRA OR project = CONF)` is equivalent to this query: `(project in (JRA, CONF))`, only the second query will be translated.
 - the query contains a NOT operator
 - the query contains an EMPTY operator
 - the query contains any of the comparison operators: `!=`, `IS`, `IS NOT`, `>`, `>=`, `<`, `<=`
 - the query specifies a field and value that is related to a project (e.g. version, component, custom fields) and the project is not explicitly included in the query (e.g. `fixVersion = "4.0"`, without the `AND project=JRA`). This is especially tricky with custom fields since they can be configured on a Project/Issue Type basis. The general rule of thumb is that if the query cannot be created in the basic search form, then it will not be able to be translated from advanced search to basic search.
2. Enter the criteria for the search. You can search against specific fields and/or search for specific text.
 - If you are searching against a field and can't find the field you want, or the field is displaying greyed out text, see the [Troubleshooting section](#) below.
 - If you are searching for text, you can use special characters and modifiers in your search text, such as wildcards and logical operators. See [Search syntax for text fields](#).
 3. The search results will automatically update in the issue navigator, unless your administrator has disabled automatic updates of search results. If so, you will need to click the **Update** button on the field drop-down after every change.

Running a saved search

Saved searches (also known as [filters](#)) are shown in the left panel, when using basic search. If the left panel is not showing, hover your mouse over the left side of the screen to display it.

To run a filter, e.g. **My Open Issues**, simply click it. The search criteria for the basic search will be set, and the search results will be displayed.

Note, clicking the **Recently Viewed** filter will switch you to the advanced search, as the basic search cannot represent the `ORDER BY` clause in this filter.



Troubleshooting

Why can't I find the field I want to choose?

Some fields are only valid for a particular *project/issue type context*. For these fields, you must select the applicable project/issue type. Otherwise, the field is not available for selection.

Why are the field criteria displaying in grey text?

Some fields are only valid for a particular *project/issue type context*. If you choose a field in your search, then remove all projects/issue types that reference the field, then the field is invalid. The invalid field does not apply to your search and displays in grey text.

Why is there a red exclamation mark in my field?

Some field values are only valid for a particular *project/issue type context*. For example, you may have configured a project to use a status *In QA Review* in its workflow. If you select this project and status in your search, then change the search to filter for a project that doesn't use *In QA Review*, the status will be invalid and ignored in the search.

Why don't my search results automatically update?

Your search results will always update automatically whenever any fields are changed, provided that your administrator has not disabled automatic updates of search results. Ask your administrator whether they

have disabled automatic updates of search results.

Next steps

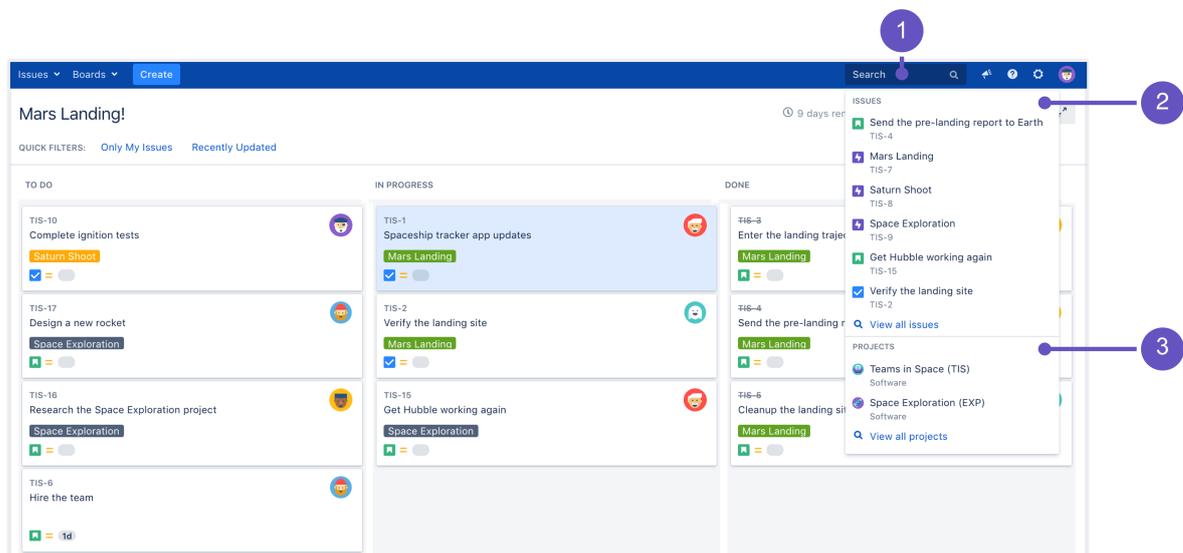
Read the following related topics:

- [Searching for issues](#)
- [Advanced searching](#)
- [Saving your search as a filter](#)
- [Working with search results](#) — find out how to use the issue navigator, export your search results, bulk modify issues, and share your search results.

Quick searching

Sometimes, you just want to be able to get to the particular issue that you're interested in. Other times, you can't remember what the issue was, but you remember that it was an open issue, assigned to you, or you have its name on the tip of your tongue. Quick search can help you in these scenarios.

The **Search** box is located at the top right of your screen, in the Jira header bar. To use quick search, just start typing what you're looking for.



1. **Search:** Click anywhere in the box to display your recent work, or start typing to search through all your issues and projects.
2. **Issues:** Recent issues (before searching), or issues that match your search.
3. **Projects:** Recent projects (before searching), or projects that match your search.

Using quick search by many users at once can affect performance. You can limit the number of concurrent searches, or monitor how your users are searching in real-time. [Learn more](#)

Understanding quick searching

Read the following topics to learn how to get the most out of quick searching:

[Jumping to an issue](#) | [Searching as you type](#) | [Free-text searching](#) | [Smart querying](#)

[Jumping to an issue](#)

If you type in the **key** of an issue, you will jump straight to that issue. For example, if you type in 'ABC-107' (or 'abc-107'), and press the **Enter** button, you will be redirected to the issue 'ABC-107'.

In many cases, you do not even need to type in the full key, but just the numerical part. If you are currently working on the 'ABC' project, and you type in '123', you will be redirected to 'ABC-123'.

Searching as you type

When you start typing the word you're looking for, the quick search will react instantly by showing and refreshing the list of most relevant results. To display these results, your search term is matched against the following fields:

- Summary (projects and issues)
- Description (issues)

Free-text searching

You can additionally search through comments or use extra operators for fuzzy or wildcard search. These results won't be displayed as 'instant results', but you can view them after pressing **Enter** in the search box.

You can combine free-text and keywords together, e.g. "my closed test tasks". You can also use wildcards, e.g. "win*8".

For more information on free-text searching, see [Search syntax for text fields](#).

Smart querying

Quick search also enables you to perform 'smart' searches with minimal typing. For example, to find all the open bugs in the 'TEST' project, you could simply type 'test open bugs' and quick search would locate them all for you.

Your search results will be displayed in the Issue Navigator, where you can view them in a variety of useful formats (Excel, XML, etc).

The search terms that quick search recognizes are:

Search Term	Description	Examples
my	Find issues assigned to me.	my open bugs
r:	Find issues reported by you, another user or with no reporter, using the prefix <i>r:</i> followed by a specific reporter term, such as <i>me</i> , a username or <i>none</i> . <i>Note that there can be no spaces between "r:" and the specific reporter term.</i>	r:me — finds issues reported by you. r:samuel — finds issues reported by the user whose username is "samuel". r:none — finds issues with no reporter.
<project name> or <project key>	Find issues in a particular project.	test project TST tst
overdue	Find issues that were due before today.	overdue

created: updated: due:	Find issues with a particular Created, Updated, or Due Date using the prefixes <i>created:</i> , <i>updated:</i> , or <i>due:</i> , respectively. For the date range, you can use <i>today</i> , <i>tomorrow</i> , <i>yesterday</i> , a single date range (e.g. '-1w'), or two date ranges (e.g. '-1w,1w'). Note that date ranges cannot have spaces in them. Valid date/time abbreviations are: 'w' (week), 'd' (day), 'h' (hour), 'm' (minute).	created:today created:yesterday updated:-1w — finds issues updated in the last week. due:1w — finds issues due in the next week. due:-1d,1w — finds issues due from yesterday to next week. created:-1w,-30m — finds issues created from one week ago, to 30 minutes ago. created:-1d updated:-4h — finds issues created in the last day, updated in the last 4 hours.
<priority>	Find issues with a particular Priority.	blocker major trivial
<issue type>	Find issues with a particular Issue Type. Note that you can also use plurals.	bug task bugs tasks
<resolution>	Find issues with a particular Resolution.	fixed duplicate cannot reproduce
c:	Find issues with a particular Component(s). You can search across multiple components. <i>Note that there can be no spaces between "c:" and the component name.</i>	c:security — finds issues with a component whose name contains the word "security".

v:	<p>Find issues with a particular Affects Version(s). To find all issues belonging to a 'major' version, use the wildcard symbol ' * '.</p> <p><i>Note that there can be no spaces between "v:" and the version name.</i></p>	<p>v: 3.0 — finds issues that match the following versions (for example):</p> <ul style="list-style-type: none"> • 3.0 • 3.0 eap • 3.0 beta <p>...but will not match against the following versions (for example):</p> <ul style="list-style-type: none"> • 3.0.1 • 3.0.0.4 <p>That is, it will match against any version that contains the string you specify followed immediately by a space, but not against versions that do not contain a space immediately after the string you specify.</p>
ff:	<p>Find issues with a particular Fix For Version(s). Same usage as v: (above).</p>	
*	<p>Wildcard symbol ' * '. Can be used with v: and ff:.</p>	<p>v: 3.2* — finds any issue whose version number is (for example):</p> <ul style="list-style-type: none"> • 3.2 • 3.2-beta • 3.2.1 • 3.2.x

In Mozilla-based browsers, try creating a bookmark with URL `http://<your-Jira-site>/secure/QuickSearch.jspa?searchString=%s` (substituting `<your-Jira-site>` with your Jira instance's URL) and keyword (such as 'j'). Now, typing 'j my open bugs' in the browser URL bar will search your Jira instance for your open bugs. Or simply type your search term in the Quick Search box, then right-click on the Quick Search box (with your search term shown) and select "Add a Keyword for this search...".

Searching issues from your browser's search box

If you are using Firefox or Internet Explorer 8 (or later), you can add your Jira instance as a search engine/provider via the drop-down menu next to the browser's search box. Once you add your Jira instance as a search engine/provider in your browser, you can use it at any time to conduct a Quick Search for issues in that Jira instance.

OpenSearch

Jira supports this browser search feature as part of the autodiscovery part of the [OpenSearch](#) standard, by supplying an [OpenSearch description document](#). This is an XML file that describes the web interface provided by Jira's search function. Any [client applications](#) that support OpenSearch will be able to add Jira to their list of search engines.

Next steps

Read the following related topics:

- [Searching for issues](#)

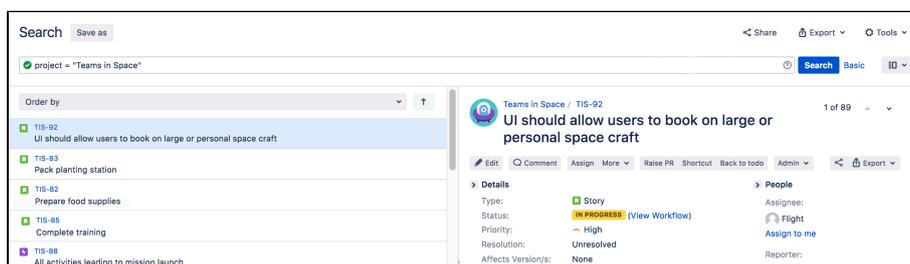
Advanced searching

The advanced search allows you to build structured queries using the Jira Query Language (JQL) to search for issues. You can specify criteria that cannot be defined in the quick or basic searches (e.g. `ORDER BY` clause).

- If you don't have complex search criteria, you may want to use [quick search](#) instead.
- If you are not comfortable with the Jira Query Language (JQL), you may want to use [basic search](#) instead.

Note, JQL is not a database query language, even though it uses SQL-like syntax.

Screenshot: Advanced search



On this page:

- [Advanced searching](#)
- [Understanding advanced searching](#)
- [Reference](#)
- [Running a saved search](#)
- [Next steps](#)

Advanced searching

1. Navigate to **Issues** (in header) > **Search for issues**.

- If there are existing search criteria, click the **New filter** button to reset the search criteria.
- If the basic search is shown instead of the advanced search, click **Advanced** (next to the **Search** button).

Why can't I switch between basic and advanced search?

In general, a query created using basic search will be able to be translated to advanced search, and back again. However, a query created using advanced search may not be able to be translated to basic search, particularly if:

- the query contains an OR operator (note you can have an IN operator and it will be translated, e.g. `project in (A, B)`)
 - i.e. even though this query: `(project = JRA OR project = CONF)` is equivalent to this query: `(project in (JRA, CONF))`, only the second query will be translated.
- the query contains a NOT operator
- the query contains an EMPTY operator
- the query contains any of the comparison operators: `!=`, `IS`, `IS NOT`, `>`, `>=`, `<`, `<=`
- the query specifies a field and value that is related to a project (e.g. version, component, custom fields) and the project is not explicitly included in the query (e.g. `fixVersion = "4.0"`, without the `AND project=JRA`). This is especially tricky with custom fields since they can be configured on a Project/Issue Type basis. The general rule of thumb is that if the query cannot be created in the basic search form, then it will not be able to be translated from advanced search to basic search.

2. Enter your JQL query. As you type, Jira will offer a list of "auto-complete" suggestions based on the context of your query. Note, auto-complete suggestions only include the first 15 matches, displayed alphabetically, so you may need to enter more text if you can't find a match.

Why aren't the auto-complete suggestions being shown?

- Your administrator may have disabled the "JQL Auto-complete" feature for your Jira instance.
- Auto-complete suggestions are not offered for function parameters.
- Auto-complete suggestions are not offered for all fields. Check the [fields](#) reference to see which fields support auto-complete.

3. Press Enter or click **Search** to run your query. Your search results will display in the issue navigator.

Understanding advanced searching

Read the following topics to learn how to get the most out of advanced searching:

[Constructing JQL queries](#) | [Setting the precedence of operators](#) | [Restricted words and characters](#) | [Performing text searches](#)

Constructing JQL queries

A simple query in JQL (also known as a 'clause') consists of a *field*, followed by an *operator*, followed by one or more *values* or *functions*. For example:

```
project = "TEST"
```

This query will find all issues in the "TEST" project. It uses the "project" *field*, the EQUALS *operator*, and the *value* "TEST".

A more complex query might look like this:

```
project = "TEST" AND assignee = currentuser()
```

This query will find all issues in the "TEST" project where the assignee is the currently logged in user. It uses the "project" *field*, the EQUALS *operator*, the *value* "TEST", the "AND" keyword and the "currentuser()" function.

For more information on fields, operators, keywords and functions, see the [Reference section](#) below.

Setting the precedence of operators

You can use parentheses in complex JQL statements to enforce the precedence of operators.

For example, if you want to find all resolved issues in the 'SysAdmin' project, as well as all issues (any status, any project) currently assigned to the system administrator (bobsmith), you can use parentheses to enforce the precedence of the boolean operators in your query, i.e.

```
(status=resolved AND project=SysAdmin) OR assignee=bobsmith
```

Note that if you do not use parentheses, the statement will be evaluated left-to-right.

You can also use parentheses to group clauses, so that you can apply the NOT operator to the group.

Restricted words and characters

Reserved characters

JQL has a list of reserved characters:

space	"	"	+	.	,	;	?		*	/	%	^	\$	#	@	[]
-------	---	---	---	---	---	---	---	--	---	---	---	---	----	---	---	---	---

If you wish to use these characters in queries, you need to:

- surround them with quote-marks (you can use either single quote-marks (') or double quote-marks ("));
- **and,**
- if you are searching a text field and the character is on the list of [reserved characters for text searches](#), precede them with two backslashes.

For example:

- `version = "[example]"`

- `summary ~ "\\[example\\]"`

Reserved words

JQL also has a list of reserved words. These words need to be surrounded by quote-marks (single or double) if you wish to use them in queries.

▼ Show me...

"abort", "access", "add", "after", "alias", "all", "alter", "and", "any", "as", "asc", "audit", "avg", "before", "begin", "between", "boolean", "break", "by", "byte", "catch", "cf", "char", "character", "check", "checkpoint", "collate", "collation", "column", "commit", "connect", "continue", "count", "create", "current", "date", "decimal", "declare", "decrement", "default", "defaults", "define", "delete", "delimiter", "desc", "difference", "distinct", "divide", "do", "double", "drop", "else", "empty", "encoding", "end", "equals", "escape", "exclusive", "exec", "execute", "exists", "explain", "false", "fetch", "file", "field", "first", "float", "for", "from", "function", "go", "goto", "grant", "greater", "group", "having", "identified", "if", "immediate", "in", "increment", "index", "initial", "inner", "inout", "input", "insert", "int", "integer", "intersect", "intersection", "into", "is", "isempty", "isnull", "join", "last", "left", "less", "like", "limit", "lock", "long", "max", "min", "minus", "mode", "modify", "modulo", "more", "multiply", "next", "noaudit", "not", "notin", "nowait", "null", "number", "object", "of", "on", "option", "or", "order", "outer", "output", "power", "previous", "prior", "privileges", "public", "raise", "raw", "remainder", "rename", "resource", "return", "returns", "revoke", "right", "row", "rowid", "rownum", "rows", "select", "session", "set", "share", "size", "sqrt", "start", "strict", "string", "subtract", "sum", "synonym", "table", "then", "to", "trans", "transaction", "trigger", "true", "uid", "union", "unique", "update", "user", "validate", "values", "view", "when", "whenever", "where", "while", "with"

Note for Jira administrators: this list is hard coded in the `JqlStringSupportImpl.java` file.

Performing text searches

You can use Lucene's text-searching features when performing searches on the following fields, using the CONTAINS operator:

Summary, Description, Environment, Comments, custom fields that use the "Free Text Searcher" (i.e. custom fields of the following built-in custom field types: Free Text Field, Text Field, Read-only Text Field).

For more information, see [Search syntax for text fields](#).

Reference

	Description	Reference
--	-------------	-----------

<p>Fields</p>	<p>A field in JQL is a word that represents a Jira field (or a custom field that has already been defined in Jira).</p>	<p>Fields reference page</p> <ul style="list-style-type: none"> ▼ Show list of fields <ul style="list-style-type: none"> • Affected version • Approvals • Assignee • Attachments • Category • Comment • Component • Created • Creator • Custom field • Customer Request Type • Description • development[builds].failing • development[commits].all • development[pullrequests].all • development[pullrequests].open • development[reviews].all • development[reviews].open • Due • Environment • Epic link • Filter • Fix version • Issue key • Issue link type • Labels • Last viewed • Level • Original estimate • Parent • Priority • Project • Remaining estimate • Reporter • Request channel type • Request last activity time • Resolution • Resolved • SLA • Sprint • Status • Summary • Text • Time spent • Type • Updated • Voter • Votes • Watcher • Watchers • Work log author • Work log comment • Work log date • Work ratio
----------------------	---	--

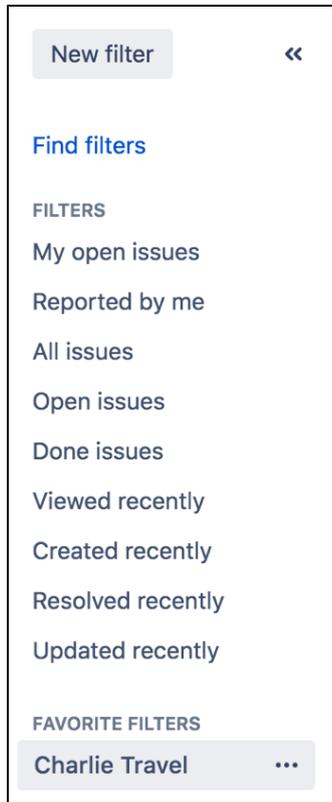
Operators	<p>An operator in JQL is one or more symbols or words that compare the value of a field on its left with one or more values (or functions) on its right, such that only true results are retrieved by the clause. Some operators may use the NOT keyword.</p>	<p>Operators reference page Show list of operators</p> <ul style="list-style-type: none"> • EQUALS: = • NOT EQUALS: != • GREATER THAN: > • GREATER THAN EQUALS: >= • LESS THAN: < • LESS THAN EQUALS: <= • IN • NOT IN • CONTAINS: ~ • DOES NOT CONTAIN: !~ • IS • IS NOT • WAS • WAS IN • WAS NOT IN • WAS NOT • CHANGED
Keywords	<p>A keyword in JQL is a word or phrase that does (or is) any of the following:</p> <ul style="list-style-type: none"> • joins two or more clauses together to form a complex JQL query • alters the logic of one or more clauses • alters the logic of operators • has an explicit definition in a JQL query • performs a specific function that alters the results of a JQL query. 	<p>Keywords reference page Show list of keywords</p> <ul style="list-style-type: none"> • AND • OR • NOT • EMPTY • NULL • ORDER BY

<p>Functions</p>	<p>A function in JQL appears as a word followed by parentheses, which may contain one or more explicit values or Jira fields.</p> <p>A function performs a calculation on either specific Jira data or the function's content in parentheses, such that only true results are retrieved by the function, and then again by the clause in which the function is used.</p>	<p>Functions reference page</p> <ul style="list-style-type: none"> <ul style="list-style-type: none"> <ul style="list-style-type: none"> • approved() • approver() • breached() • cascadeOption() • closedSprints() • completed() • componentsLeadByUser() • currentLogin() • currentUser() • earliestUnreleasedVersion() • elapsed() • endOfDay() • endOfMonth() • endOfWeek() • endOfYear() • everbreached() • issueHistory() • issuesWithRemoteLinksByGlobalId() • lastLogin() • latestReleasedVersion() • linkedIssues() • membersOf() • myApproval() • myPending() • now() • openSprints() • paused() • pending() • pendingBy() • projectsLeadByUser() • projectsWhereUserHasPermission() • projectsWhereUserHasRole() • releasedVersions() • remaining() • running() • standardIssueTypes() • startOfDay() • startOfMonth() • startOfWeek() • startOfYear() • subtaskIssueTypes() • unreleasedVersions() • updatedBy() • votedIssues() • watchedIssues() • withinCalendarHours()
-------------------------	--	--

Running a saved search

Saved searches (also known as [Saving your search as a filter](#)) are shown in the left panel, when using advanced search. If the left panel is not showing, hover your mouse over the left side of the screen to display it.

To run a filter, e.g. **My Open Issues**, simply click it. The JQL for the advanced search will be set, and the search results will be displayed.



Next steps

Read the following related topics:

- [Searching for issues](#)
- [Basic searching](#)
- [Search syntax for text fields](#)
- [JQL: The most flexible way to search Jira \(on the Atlassian blog\)](#)
- [Saving your search as a filter](#)
- [Working with search results](#)—find out how to use the issue navigator, export your search results, bulk modify issues, and share your search results.

Advanced searching - fields reference

This page describes information about fields that are used for advanced searching. A field in JQL is a word that represents a Jira field (or a custom field that has already been defined in your Jira applications). In a clause, a field is followed by an [operator](#), which in turn is followed by one or more values (or [functions](#)). The operator compares the value of the field with one or more values or functions on the right, such that only true results are retrieved by the clause. Note: it is not possible to compare two fields in JQL.

Affected version

Search for issues that are assigned to a particular affects version(s). You can search by version name or version ID (i.e. the number that Jira automatically allocates to a version). Note, it is better to search by version ID than by version name. Different projects may have versions with the same name. It is also possible for your Jira administrator to change the name of a version, which could break any saved filters that rely on that name. Version IDs, however, are unique and cannot be changed.

Syntax	<code>affectedVersion</code>
Field Type	VERSION

On this page:

- [Affected version](#)
- [Approvals](#)
- [Assignee](#)
- [Attachments](#)
- [Category](#)
- [Comment](#)
- [Component](#)
- [Created](#)
- [Creator](#)
- [Custom field](#)
- [Customer Request Type](#)
- [Description](#)
- [development\[builds\].failing](#)

Auto-complete	Yes
Supported operators	<p>= , != , > , >= , < , <= , ~ , !~</p> <p>IS , IS NOT , IN , NOT IN</p> <ul style="list-style-type: none"> • The comparison operators (e.g. ">") use the version order that has been set up by your project administrator, not a numeric or alphabetic order. • For this field, the contain operators (~ and !~) find exact matches, and can be used to search through versions with a wildcard.
Unsupported operators	<p>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</p>
Supported functions	<p>When used with the IN and NOT IN operators, this field supports:</p> <ul style="list-style-type: none"> • releasedVersions() • latestReleasedVersion() • unreleasedVersions() • earliestUnreleasedVersion()
Examples	<ul style="list-style-type: none"> • Find issues with an AffectedVersion of 3.14: affectedVersion = "3.14" <p><i>Note that full-stops are reserved characters and need to be surrounded by quote-marks.</i></p> <ul style="list-style-type: none"> • Find issues with an AffectedVersion of "Big Ted": affectedVersion = "Big Ted" • Find issues with an AffectedVersion ID of 10350: affectedVersion = 10350

[^ top of page](#)

- development[commits].all
- development[pullrequests].all
- development[pullrequests].open
- development[reviews].all
- development[reviews].open
- Due
- Environment
- Epic link
- Filter
- Fix version
- Issue key
- Issue link type
- Labels
- Last viewed
- Level
- Original estimate
- Parent
- Priority
- Project
- Remaining estimate
- Reporter
- Request channel type
- Request last activity time
- Resolution
- Resolved
- SLA
- Sprint
- Status
- Summary
- Text
- Time spent
- Type
- Updated
- Voter
- Votes
- Watcher
- Watchers
- Work log author
- Work log comment
- Work log date

Approvals

Only applicable if Jira Service Desk is installed and licensed, and you're using the Approvals functionality.

Search for issues that have been approved or require approval. This can be further refined by user.

Syntax	approvals
Field Type	USER
Auto-complete	No
Supported operators	=
Unsupported operators	~ , != , !~ , > , >= , < , <= IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Supported functions	<ul style="list-style-type: none"> • approved() • approver() • myApproval() • myPending() • pending() • pendingBy()
Examples	<ul style="list-style-type: none"> • Find issues that require or required approval by John Smith: approval = approver(jsmith) • Find issues that require approval by John Smith: approval = pendingBy(jsmith) • Find issues that require approval by the current user: approval = myPending() • Find all issues that require approval: approval = pending()

[^ top of page](#)

Assignee

Search for issues that are assigned to a particular user. You can search by the user's full name, ID, or email address.

Syntax	assignee
Alias	cf[CustomFieldID]
Field Type	USER
Auto-complete	Yes
Supported operators	= , != IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED <i>Note that the comparison operators (e.g. ">") use the version order that has been set up by your project administrator, not a numeric or alphabetic order.</i>

Unsupported operators	<code>~ , !~ , > , >= , < , <=</code>
Supported functions	<p>When used with the IN and NOT IN operators, this field supports:</p> <ul style="list-style-type: none"> membersOf() <p>When used with the EQUALS and NOT EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> currentUser()
Examples	<ul style="list-style-type: none"> Find issues that are assigned to John Smith: <code>assignee = "John Smith"</code> or <code>assignee = jsmith</code> Find issues that are currently assigned, or were previously assigned, to John Smith: <code>assignee WAS "John Smith"</code> or <code>assignee WAS jsmith</code> Find issues that are assigned by the user with email address "bob@mycompany.com": <code>assignee = "bob@mycompany.com"</code> <p><i>Note that full-stops and "@" symbols are reserved characters and need to be surrounded by quote-marks.</i></p>

[^ top of page](#)

Attachments

Search for issues that have or do not have attachments.

Syntax	<code>attachments</code>
Field Type	ATTACHMENT
Auto-complete	Yes
Supported operators	IS, IS NOT
Unsupported operators	=, != , ~ , !~ , > , >= , < , <= IN, NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Search for issues that have attachments: <code>attachments IS NOT EMPTY</code> Search for issues that do not have attachments: <code>attachments IS EMPTY</code>

[^ top of page](#)

Category

Search for issues that belong to projects in a particular category.

Syntax	<code>category</code>
---------------	-----------------------

Field Type	CATEGORY
Auto-complete	Yes
Supported operators	=, != IS, IS NOT, IN, NOT IN
Unsupported operators	~ , !~ , > , >= , < , <= WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues that belong to projects in the "Alphabet Projects" Category: category = "Alphabet Projects"

[^ top of page](#)

Comment

Search for issues that have a comment that contains particular text. [Jira text-search syntax](#) can be used.

Syntax	comment
Field Type	TEXT
Auto-complete	No
Supported operators	~ , !~
Unsupported operators	= , != , > , >= , < , <= IS, IS NOT, IN, NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues where a comment contains text that matches "My PC is quite old" (i.e. a "fuzzy" match): comment ~ "My PC is quite old" Find issues where a comment contains the exact phrase "My PC is quite old": comment ~ "\"My PC is quite old\""

[^ top of page](#)

Component

Search for issues that belong to a particular component(s) of a project. You can search by component name or component ID (i.e. the number that Jira automatically allocates to a component).

Note, it is safer to *search by component ID than by component name*. Different projects may have components with the same name, so searching by component name may return issues from multiple projects. It is also possible for your Jira administrator to change the name of a component, which could break any saved filters that rely on that name. Component IDs, however, are unique and cannot be changed.

Syntax	component
Field Type	COMPONENT
Auto-complete	Yes

Supported operators	<code>= , !=</code> <code>IS , IS NOT , IN , NOT IN</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <=</code> <code>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Supported functions	When used with the IN and NOT IN operators, component supports: <ul style="list-style-type: none"> • <code>componentsLeadByUser()</code>
Examples	<ul style="list-style-type: none"> • Find issues in the "Comp1" or "Comp2" component: <code>component in (Comp1, Comp2)</code> • Find issues in the "Comp1" and "Comp2" components: <code>component in (Comp1) and component in (Comp2)</code> or <code>component = Comp1 and component = Comp2</code> • Find issues in the component with ID 20500: <code>component = 20500</code>

[^ top of page](#)

Created

Search for issues that were created on, before, or after a particular date (or date range). Note that if a time-component is not specified, midnight will be assumed. Please note that the search results will be relative to your configured time zone (which is by default the Jira server's time zone).

Use one of the following formats:

"YYYY/MM/dd HH:mm"

"YYYY-MM-dd HH:mm"

"YYYY/MM/dd"

"YYYY-MM-dd"

Or use "w" (weeks), "d" (days), "h" (hours) or "m" (minutes) to specify a date relative to the current time. The default is "m" (minutes). Be sure to use quote-marks (""); if you omit the quote-marks, the number you supply will be interpreted as milliseconds after epoch (1970-1-1).

Syntax	<code>created</code>
Alias	<code>createdDate</code>
Field Type	DATE
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code> <code>IS , IS NOT , IN , NOT IN</code>
Unsupported operators	<code>~ , !~</code> <code>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>

Supported functions	<p>When used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> • <code>currentLogin()</code> • <code>lastLogin()</code> • <code>now()</code> • <code>startOfDay()</code> • <code>startOfWeek()</code> • <code>startOfMonth()</code> • <code>startOfYear()</code> • <code>endOfDay()</code> • <code>endOfWeek()</code> • <code>endOfMonth()</code> • <code>endOfYear()</code>
Examples	<ul style="list-style-type: none"> • Find all issues created before 12th December 2010: <code>created < "2010/12/12"</code> • Find all issues created on or before 12th December 2010: <code>created <= "2010/12/13"</code> • Find all issues created on 12th December 2010 before 2:00pm: <code>created > "2010/12/12" and created < "2010/12/12 14:00"</code> • Find issues created less than one day ago: <code>created > "-1d"</code> • Find issues created in January 2011: <code>created > "2011/01/01" and created < "2011/02/01"</code> • Find issues created on 15 January 2011: <code>created > "2011/01/15" and created < "2011/01/16"</code>

[^ top of page](#)

Creator

Search for issues that were created by a particular user. You can search by the user's full name, ID, or email address.

Syntax	<code>creator</code>
Field Type	USER
Auto-complete	Yes
Supported operators	<code>=</code> , <code>!=</code> <code>IS</code> , <code>IS NOT</code> , <code>IN</code> , <code>NOT IN</code> , <code>WAS</code> , <code>WAS IN</code> , <code>WAS NOT</code> , <code>WAS NOT IN</code>
Unsupported operators	<code>~</code> , <code>!~</code> , <code>></code> , <code>>=</code> , <code><</code> , <code><=</code> <code>CHANGED</code>
Supported functions	<p>When used with the IN and NOT IN operators, this field supports:</p> <ul style="list-style-type: none"> • <code>membersOf()</code> <p>When used with the EQUALS and NOT EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> • <code>currentUser()</code>

Examples	<ul style="list-style-type: none"> Search for issues that were created by Jill Jones: <pre>creator = "Jill Jones"</pre> or <pre>creator = "jjones"</pre> Search for issues that were created by the user with email address "bob@mycompany.com": <pre>creator = "bob@mycompany.com"</pre> <p><i>(Note that full-stops and "@" symbols are reserved characters, so the email address needs to be surrounded by quote-marks.)</i></p>
-----------------	--

[^ top of page](#)

Custom field

Only applicable if your Jira administrator has created one or more custom fields.

Search for issues where a particular custom field has a particular value. You can search by custom field name or custom field ID (i.e. the number that Jira automatically allocates to an custom field).

Note, it is safer to search by custom field ID than by custom field name. It is possible for a custom field to have the same name as a built-in Jira system field; in which case, Jira will search for the system field (not your custom field). It is also possible for your Jira administrator to change the name of a custom field, which could break any saved filters that rely on that name. Custom field IDs, however, are unique and cannot be changed.

Syntax	CustomFieldName
Alias	cf[CustomFieldID]
Field Type	<p><i>Depends on the custom field's configuration</i></p> <p><i>Note, Jira text-search syntax can be used with custom fields of type 'Text'.</i></p>
Auto-complete	Yes, for custom fields of type picker, group picker, select, checkbox and radio button fields
Supported operators	<i>Different types of custom field support different operators.</i>
Supported operators: number and date fields	<pre>= , != , > , >= , < , <= IS , IS NOT , IN , NOT IN</pre>
Unsupported operators: number and date fields	<pre>~ , !~ WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</pre>
Supported operators: picker, select, checkbox and radio button fields	<pre>= , != IS , IS NOT , IN , NOT IN</pre>
Unsupported operators: picker, select, checkbox and radio button fields	<pre>~ , !~ , > , >= , < , <= WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</pre>

Supported operators: text fields	~ , !~ IS , IS NOT
Unsupported operators: text fields	= , != , > , >= , < , <= IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Unsupported operators	~ , !~ , > , >= , < , <= WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Supported functions	<i>Different types of custom fields support different functions.</i>
Supported functions: date/time fields	When used with the EQUALS , NOT EQUALS , GREATER THAN , GREATER THAN EQUALS , LESS THAN or LESS THAN EQUALS operators, this field supports: <ul style="list-style-type: none"> • currentLogin() • lastLogin() • now() • startOfDay() • startOfWeek() • startOfMonth() • startOfYear() • endOfDay() • endOfWeek() • endOfMonth() • endOfYear()
Supported functions: version picker fields	Version picker fields: When used with the IN and NOT IN operators, this field supports: <ul style="list-style-type: none"> • releasedVersions() • latestReleasedVersion() • unreleasedVersions() • earliestUnreleasedVersion()
Examples	<ul style="list-style-type: none"> • Find issues where the value of the "Location" custom field is "New York": location = "New York" • Find issues where the value of the custom field with ID 10003 is "New York": cf[10003] = "New York" • Find issues where the value of the "Location" custom field is "London" or "Milan" or "Paris": cf[10003] in ("London", "Milan", "Paris") • Find issues where the "Location" custom field has no value: location != empty

[^ top of page](#)

Customer Request Type

Only applicable if Jira Service Desk is installed and licensed.

Search for Issues matching a specific Customer Request Type in a service desk project. You can search for a Customer Request Type either by name or description as configured in the Request Type configuration screen.

Syntax	"Customer Request Type"
Field Type	Custom field

Auto-complete	Yes
Supported operators	= , != IN , NOT IN
Unsupported operators	~ , !~ , > , >= , < , <= IS , IS NOT, WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED
	<div style="border: 1px solid black; padding: 5px;"> <p>Note that the Lucene value for Customer Request Type, is <code>portal-key/request-type-key</code>. While the portal key cannot be changed after a service desk portal is created, the project key can be changed. The Request Type key cannot be changed once the Request Type is created.</p> </div>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues where Customer Request Type is Request a new account in projects that the user has access to: "Customer Request Type" = "Request a new account" Find issues where the Customer Request Type is Request a new account in SimpleDesk project, where the right operand is a selected Lucene value from the auto-complete suggestion list. "Customer Request Type" = "sd/system-access" Find issues where Customer Request Type is either Request a new account or Get IT Help. "Customer Request Type" IN ("Request a new account", "Get IT Help")

[^ top of page](#)

Description

Search for issues where the description contains particular text. [Jira text-search syntax](#) can be used.

Syntax	description
Field Type	TEXT
Auto-complete	No
Supported operators	~ , !~ IS , IS NOT
Unsupported operators	= , != , > , >= , < , <= IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues where the description contains text that matches "Please see screenshot" (i.e. a "fuzzy" match): description ~ "Please see screenshot" Find issues where the description contains the exact phrase "Please see screenshot": description ~ "\"Please see screenshot\""

[^ top of page](#)

development[builds].failing

Search for issues that have failed builds on a linked Bamboo instance or instances.

Syntax	<code>development[builds].failing</code>
Field Type	TEXT
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code>
Unsupported operators	<code>~ , !~, IS , IS NOT , IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find all issues with more than 1 failing build: <code>development[builds].failing > 1</code>

[^ top of page](#)

development[commits].all

Search for issues that have code commits on a linked Bitbucket instance or instances.

Syntax	<code>development[commits].all</code>
Field Type	TEXT
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code>
Unsupported operators	<code>~ , !~, IS , IS NOT , IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find all issues with more than 15 commits: <code>development[commits].all > 15</code>

[^ top of page](#)

development[pullrequests].all

Search for issues that have open or merged pull requests on a linked Bitbucket instance or instances.

Syntax	<code>development[pullrequests].all</code>
Field Type	TEXT
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code>
Unsupported operators	<code>~ , !~, IS , IS NOT , IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED</code>
Supported functions	None

Examples	<ul style="list-style-type: none"> Find all issues with more than 5 open or merged pull requests: <code>development[pullrequests].all > 5</code>
-----------------	--

[^ top of page](#)

development[pullrequests].open

Search for issues that have open pull requests on a linked Bitbucket instance or instances.

Syntax	<code>development[pullrequests].open</code>
Field Type	TEXT
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code>
Unsupported operators	<code>~ , !~, IS , IS NOT , IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find all issues with more than 2 open pull requests: <code>development[pullrequests].open > 2</code>

[^ top of page](#)

development[reviews].all

Search for issues that have open or closed reviews on a linked Crucible instance or instances.

Syntax	<code>development[reviews].all</code>
Field Type	TEXT
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code>
Unsupported operators	<code>~ , !~, IS , IS NOT , IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find all issues with more than 2 open or closed reviews: <code>development[review].all > 2</code>

[^ top of page](#)

development[reviews].open

Search for issues that have open reviews on a linked Crucible instance or instances.

Syntax	<code>development[reviews].open</code>
Field Type	TEXT
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code>

Unsupported operators	<code>~ , !~, IS , IS NOT , IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find all issues with open reviews: <code>development[review].open > 0</code>

[^ top of page](#)

Due

Search for issues that were due on, before, or after a particular date (or date range). Note that the due date relates to the *date* only (not to the time).

Use one of the following formats:

`"YYYY/MM/dd"`

`"YYYY-MM-dd"`

Or use "w" (weeks) or "d" (days) to specify a date relative to the current date. Be sure to use quote-marks (").

Syntax	due
Alias	dueDate
Field Type	DATE
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code> <code>IS , IS NOT , IN , NOT IN</code>
Unsupported operators	<code>~ , !~</code> <code>WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED</code>
Supported functions	<p>When used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> currentLogin() lastLogin() now() startOfDay() startOfWeek() startOfMonth() startOfYear() endOfDay() endOfWeek() endOfMonth() endOfYear()

Examples	<ul style="list-style-type: none"> Find all issues due before 31st December 2010: <code>due < "2010/12/31"</code> Find all issues due on or before 31st December 2010: <code>due <= "2011/01/01"</code> Find all issues due tomorrow: <code>due = "1d"</code> Find all issues due in January 2011: <code>due >= "2011/01/01" and due <= "2011/01/31"</code> Find all issues due on 15 January 2011: <code>due = "2011/01/15"</code>
-----------------	--

[^ top of page](#)

Environment

Search for issues where the environment contains particular text. [Jira text-search syntax](#) can be used.

Syntax	<code>environment</code>
Field Type	TEXT
Auto-complete	No
Supported operators	<code>~ , !~</code> <code>IS , IS NOT</code>
Unsupported operators	<code>= , != , > , >= , < , <=</code> <code>IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues where the environment contains text that matches "Third floor" (i.e. a "fuzzy" match): <code>environment ~ "Third floor"</code> Find issues where the environment contains the exact phrase "Third floor": <code>environment ~ "\"Third floor\""</code>

[^ top of page](#)

Epic link

Search for issues that belong to a particular epic. The search is based on either the epic's name, issue key, or issue ID (i.e. the number that Jira automatically allocates to an issue).

Syntax	<code>"epic link"</code>
Field Type	Epic Link Relationship
Auto-complete	No
Supported operators	<code>= , !=</code> <code>IS , IS NOT, IN , NOT IN</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <=</code> <code>WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED</code>

Supported functions	When used with the IN or NOT IN operators, epic link supports: <ul style="list-style-type: none"> • <code>issueHistory()</code> • <code>linkedIssues()</code> • <code>votedIssues()</code> • <code>watchedIssues()</code>
Examples	<ul style="list-style-type: none"> • Find issues that belong to epic "Jupiter", where "Jupiter" has the issue key ANERDS-31: <pre>"epic link" = ANERDS-31</pre> or <pre>"epic link" = Jupiter</pre>

[^ top of page](#)

Filter

You can use a saved filter to narrow your search. You can search by filter name or filter ID (i.e. the number that Jira automatically allocates to a saved filter).

Note:

- It is safer to search by filter ID than by filter name. It is possible for a filter name to be changed, which could break a saved filter that invokes another filter by name. Filter IDs, however, are unique and cannot be changed.
- An unnamed link statement in your typed query will override an ORDER BY statement in the saved filter.
- You cannot run or save a filter that would cause an infinite loop (i.e. you cannot reference a saved filter if it eventually references your current filter).

Syntax	<code>filter</code>
Aliases	<code>request</code> , <code>savedFilter</code> , <code>searchRequest</code>
Field Type	Filter
Auto-complete	Yes
Supported operators	<code>=</code> , <code>!=</code> <code>IN</code> , <code>NOT IN</code>
Unsupported operators	<code>~</code> , <code>!~</code> , <code>></code> , <code>>=</code> , <code><</code> , <code><=</code> <code>IS</code> , <code>IS NOT</code> , <code>WAS</code> , <code>WAS IN</code> , <code>WAS NOT</code> , <code>WAS NOT IN</code> , <code>CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> • Search the results of the filter "My Saved Filter" (which has an ID of 12000) for issues assigned to the user jsmith: <pre>filter = "My Saved Filter" and assignee = jsmith</pre> or <pre>filter = 12000 and assignee = jsmith</pre>

[^ top of page](#)

Fix version

Search for issues that are assigned to a particular fix version. You can search by version name or version ID (i.e. the number that Jira automatically allocates to a version).

Note, it is safer to search by version ID than by version name. Different projects may have versions with the same name, so searching by version name may return issues from multiple projects. It is also possible for

your Jira administrator to change the name of a version, which could break any saved filters that rely on that name. Version IDs, however, are unique and cannot be changed.

Syntax	<code>fixVersion</code>
Field Type	VERSION
Auto-complete	Yes
Supported operators	<p><code>= , != , > , >= , < , <= , ~ , !~</code> <code>IS , IS NOT, IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN ,</code> <code>CHANGED</code></p> <ul style="list-style-type: none"> • The comparison operators (e.g. ">") use the version order that has been set up by your project administrator, not a numeric or alphabetic order. • For this field, the contain operators (~ and !~) find exact matches, and can be used to search through versions with a wildcard.
Unsupported operators	
Supported functions	<p>When used with the IN and NOT IN operators, this field supports:</p> <ul style="list-style-type: none"> • <code>releasedVersions()</code> • <code>latestReleasedVersion()</code> • <code>unreleasedVersions()</code> • <code>earliestUnreleasedVersion()</code>
Examples	<ul style="list-style-type: none"> • Find issues with a Fix Version of 3.14 or 4.2: <code>fixVersion in ("3.14", "4.2")</code> <i>(Note that full-stops are reserved characters, so they need to be surrounded by quote-marks.)</i> • Find issues with a Fix Version of "Little Ted": <code>fixVersion = "Little Ted"</code> • Find issues with a Fix Version ID of 10001: <code>fixVersion = 10001</code>

[^ top of page](#)

Issue key

Search for issues with a particular issue key or issue ID (i.e. the number that Jira automatically allocates to an issue).

Syntax	<code>issueKey</code>
Aliases	<code>id , issue , key</code>
Field Type	ISSUE
Auto-complete	No
Supported operators	<p><code>= , != , > , >= , < , <=</code> <code>IS , IS NOT, IN , NOT IN</code></p>
Unsupported operators	<p><code>~ , !~</code> <code>WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED</code></p>

Supported functions	When used with the IN or NOT IN operators, <code>issueKey</code> supports: <ul style="list-style-type: none"> • <code>issueHistory()</code> • <code>linkedIssues()</code> • <code>updatedBy()</code> • <code>votedIssues()</code> • <code>watchedIssues()</code>
Examples	<ul style="list-style-type: none"> • Find the issue with key "ABC-123": <code>issueKey = ABC-123</code>

[^ top of page](#)

Issue link type

Search for issues that have a particular link type, like *blocks* or *is duplicated by*.

Syntax	<code>issueLinkType</code>
Alias	<code>issueType ???</code>
Field Type	ISSUE_TYPE ???
Auto-complete	Yes
Supported operators	<code>= , !=</code> <code>IN , NOT IN</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <=</code> <code>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED , IS , IS NOT</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> • Find issues with a link type of "blocks": <code>issueLinkType = blocks</code> • Find issues with an issue type of "duplicates" or "is duplicated by": <code>issueLinkType in (duplicates,"is duplicated by")</code>

[^ top of page](#)

Labels

Search for issues tagged with a label or list of labels. You can also search for issues without any labels to easily identify which issues need to be tagged so they show up in the relevant sprints, queues or reports.

Syntax	<code>labels</code>
Field Type	LABEL
Auto-complete	Yes
Supported operators	<code>= , != , IS , IS NOT , IN , NOT IN</code> <i>We recommend using IS or IS NOT to search for a single label, and IN or NOT IN to search for a list of labels.</i>
Unsupported operators	<code>~ , !~ , , > , >= , < , <=</code> <code>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>

Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues with an existing label: labels = "x" Find issues without a specified label, including issues without a label: labels not in ("x") or labels is EMPTY

Last viewed

Search for issues that were last viewed on, before, or after a particular date (or date range). Note that if a time-component is not specified, midnight will be assumed. Please note that the search results will be relative to your configured time zone (which is by default the Jira server's time zone).

Use one of the following formats:

"YYYY/MM/dd HH:mm"

"YYYY-MM-dd HH:mm"

"YYYY/MM/dd"

"YYYY-MM-dd"

Or use "w" (weeks), "d" (days), "h" (hours) or "m" (minutes) to specify a date relative to the current time. The default is "m" (minutes). Be sure to use quote-marks (""); if you omit the quote-marks, the number you supply will be interpreted as milliseconds after epoch (1970-1-1).

Syntax	lastViewed
Field Type	DATE
Auto-complete	No
Supported operators	= , != , > , >= , < , <= IS , IS NOT, IN , NOT IN
Unsupported operators	~ , !~ WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED
Supported functions	<p>When used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> currentLogin() lastLogin() now() startOfDay() startOfWeek() startOfMonth() startOfYear() endOfDay() endOfWeek() endOfMonth() endOfYear()

Examples	<ul style="list-style-type: none"> Find all issues last viewed before 12th December 2010: <code>lastViewed < "2010/12/12"</code> Find all issues last viewed on or before 12th December 2010: <code>lastViewed <= "2010/12/13"</code> Find all issues last viewed on 12th December 2010 before 2:00pm: <code>lastViewed > "2010/12/12" and created < "2010/12/12 14:00"</code> Find issues last viewed less than one day ago: <code>lastViewed > "-1d"</code> Find issues last viewed in January 2011: <code>lastViewed > "2011/01/01" and created < "2011/02/01"</code> Find issues last viewed on 15 January 2011: <code>lastViewed > "2011/01/15" and created < "2011/01/16"</code>
-----------------	---

[^ top of page](#)

Level

Only available if issue level security has been enabled by your Jira administrator.

Search for issues with a particular security level. You can search by issue level security name or issue level security ID (i.e. the number that Jira automatically allocates to an issue level security).

Note, it is safer to search by security level ID than by security level name. It is possible for your Jira administrator to change the name of a security level, which could break any saved filter that rely on that name. Security level IDs, however, are unique and cannot be changed.

Syntax	<code>level</code>
Field Type	SECURITY LEVEL
Auto-complete	Yes
Supported operators	<code>= , !=</code> <code>IS , IS NOT, IN , NOT IN</code>
Unsupported operators	<code>> , >= , < , <= , ~ , !~</code> <code>WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Search for issues with a security level of "Really High" or "level1": <code>level in ("Really High", level1)</code> Search for issues with a security level ID of 123: <code>level = 123</code>

[^ top of page](#)

Original estimate

Only available if time-tracking has been enabled by your Jira administrator.

Search for issues where the original estimate is set to a particular value (i.e. a number, not a date or date range). Use "w", "d", "h" and "m" to specify weeks, days, hours, or minutes.

Syntax	<code>originalEstimate</code>
Alias	<code>timeOriginalEstimate</code>
Field Type	DURATION

Auto-complete	No
Supported operators	= , != , > , >= , < , <= IS , IS NOT, IN , NOT IN
Unsupported operators	~ , !~ WAS , WAS IN, WAS NOT, WAS NOT IN , CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues with an original estimate of 1 hour: <code>originalEstimate = 1h</code> Find issues with an original estimate of more than 2 days: <code>originalEstimate > 2d</code>

[^ top of page](#)

Parent

Only available if sub-tasks have been enabled by your Jira administrator.

Search for all sub-tasks of a particular issue. You can search by issue key or by issue ID (i.e. the number that Jira automatically allocates to an Issue).

Syntax	<code>parent</code>
Field Type	ISSUE
Auto-complete	No
Supported operators	= , != IN , NOT IN
Unsupported operators	> , >= , < , <= , ~ , !~ IS , IS NOT, WAS , WAS IN, WAS NOT, WAS NOT IN , CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues that are sub-tasks of issue TEST-1234: <code>parent = TEST-1234</code>

[^ top of page](#)

Priority

Search for issues with a particular priority. You can search by priority name or priority ID (i.e. the number that Jira automatically allocates to a priority).

Note, it is safer to search by priority ID than by priority name. It is possible for your Jira administrator to change the name of a priority, which could break any saved filter that rely on that name. Priority IDs, however, are unique and cannot be changed.

Syntax	<code>priority</code>
Field Type	PRIORITY
Auto-complete	Yes
Supported operators	= , != , > , >= , < , <= IS , IS NOT, IN , NOT IN , WAS , WAS IN, WAS NOT, WAS NOT IN , CHANGED

Unsupported operators	~ , !~
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues with a priority of "High": <code>priority = High</code> Find issues with a priority ID of 10000: <code>priority = 10000</code>

[^ top of page](#)

Project

Search for issues that belong to a particular project. You can search by project name, by project key or by project ID (i.e. the number that Jira automatically allocates to a project). In the rare case where there is a project whose project key is the same as another project's name, then the project key takes preference and hides results from the second project.

Syntax	<code>project</code>
Field Type	PROJECT
Auto-complete	Yes
Supported operators	= , != IS , IS NOT, IN , NOT IN
Unsupported operators	> , >= , < , <= , ~ , !~ WAS , WAS IN, WAS NOT, WAS NOT IN , CHANGED
Supported functions	When used with the IN and NOT IN operators, <code>project</code> supports: <ul style="list-style-type: none"> <code>projectsLeadByUser()</code> <code>projectsWhereUserHasPermission()</code> <code>projectsWhereUserHasRole()</code>
Examples	<ul style="list-style-type: none"> Find issues that belong to the Project that has the name "ABC Project": <code>project = "ABC Project"</code> Find issues that belong to the project that has the key "ABC": <code>project = "ABC"</code> Find issues that belong to the project that has the ID "1234": <code>project = 1234</code>

[^ top of page](#)

Remaining estimate

Only available if time-tracking has been enabled by your Jira administrator.

Search for issues where the remaining estimate is set to a particular value (i.e. a number, not a date or date range). Use "w", "d", "h" and "m" to specify weeks, days, hours, or minutes.

Syntax	<code>remainingEstimate</code>
Alias	<code>timeEstimate</code>
Field Type	DURATION

Auto-complete	No
Supported operators	= , != , > , >= , < , <= IS , IS NOT, IN , NOT IN
Unsupported operators	~ , !~ WAS , WAS IN, WAS NOT, WAS NOT IN , CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues with a remaining estimate of more than 4 hours: remainingEstimate > 4h

[^ top of page](#)

Reporter

Search for issues that were reported by a particular user. This may be the same as the creator, but can be distinct. You can search by the user's full name, ID, or email address.

Syntax	reporter
Field Type	USER
Auto-complete	Yes
Supported operators	= , != IS , IS NOT, IN , NOT IN , WAS , WAS IN, WAS NOT, WAS NOT IN , CHANGED
Unsupported operators	~ , !~ , > , >= , < , <=
Supported functions	<p>When used with the IN and NOT IN operators, this field supports:</p> <ul style="list-style-type: none"> membersOf() <p>When used with the EQUALS and NOT EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> currentUser()
Examples	<ul style="list-style-type: none"> Search for issues that were reported by Jill Jones: reporter = "Jill Jones" or reporter = jjones Search for issues that were reported by the user with email address "bob@mycompany.com": reporter = "bob@mycompany.com" <p><i>(Note that full-stops and "@" symbols are reserved characters, so the email address needs to be surrounded by quote-marks.)</i></p>

[^ top of page](#)

Request channel type

Only applicable if Jira Service Desk is installed and licensed.

Search for issues that were requested through a specific channel (e.g. issues submitted via email or through a Service Desk portal).

Syntax	request-channel-type
---------------	----------------------

Field Type	TEXT
Auto-complete	Yes
Supported operators	= , != IS, IS NOT, IN, NOT IN
Unsupported operators	~ , !~ , > , >= , < , <= WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED
Supported functions	When used with the IN and NOT IN operators, this field supports: <ul style="list-style-type: none"> • email: requests submitted via email • Jira: requests created using Jira • portal: requests created using a Service Desk portal • api: requests created using a REST API
Examples	<ul style="list-style-type: none"> • Find issues where the request channel was email: request-channel-type = email • Find issues where the request channel was something other than a service desk portal: request-channel-type != portal

[^ top of page](#)

Request last activity time

Only applicable if Jira Service Desk is installed and licensed.

Search for issues that were created on, before, or after a particular date (or date range). Note that if a time-component is not specified, midnight will be assumed. Please note that the search results will be relative to your configured time zone (which is by default the Jira server's time zone).

Use one of the following formats:

"YYYY/MM/dd HH:mm"

"YYYY-MM-dd HH:mm"

"YYYY/MM/dd"

"YYYY-MM-dd"

Or use "w" (weeks), "d" (days), "h" (hours) or "m" (minutes) to specify a date relative to the current time. The default is "m" (minutes). Be sure to use quote-marks (""); if you omit the quote-marks, the number you supply will be interpreted as milliseconds after epoch (1970-1-1).

Syntax	request-last-activity-time
Field Type	DATE
Auto-complete	Yes
Supported operators	= , != , > , >= , < , <= IS, IS NOT, IN, NOT IN
Unsupported operators	~ , !~ WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED

Supported functions	<p>When used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> • <code>currentLogin()</code> • <code>lastLogin()</code> • <code>now()</code> • <code>startOfDay()</code> • <code>startOfWeek()</code> • <code>startOfMonth()</code> • <code>startOfYear()</code> • <code>endOfDay()</code> • <code>endOfWeek()</code> • <code>endOfMonth()</code> • <code>endOfYear()</code>
Examples	<ul style="list-style-type: none"> • Find all issues last acted on before 23rd May 2016: <code>request-last-activity-time < "2016/05/23"</code> • Find all issues last acted on or before 23rd May 2016: <code>request-last-activity-time <= "2016/05/23"</code> • Find all issues created on 23rd May 2016 and last acted on before 2:00pm that day: <code>created > "2016/05/23" AND request-last-activity-time < "2016/05/23 14:00"</code> • Find issues last acted on less than one day ago: <code>request-last-activity-time > "-1d"</code> • Find issues last acted on in January 2016: <code>request-last-activity-time > "2016/01/01" and request-last-activity-time < "2016/02/01"</code>

[^ top of page](#)

Resolution

Search for issues that have a particular resolution. You can search by resolution name or resolution ID (i.e. the number that Jira automatically allocates to a resolution).

Note, it is safer to search by resolution ID than by resolution name. It is possible for your Jira administrator to change the name of a resolution, which could break any saved filter that rely on that name. Resolution IDs, however, are unique and cannot be changed.

Syntax	<code>resolution</code>
Field Type	RESOLUTION
Auto-complete	Yes
Supported operators	<code>= , != , > , >= , < , <=</code> <code>IS , IS NOT, IN , NOT IN , WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED</code>
Unsupported operators	<code>~ , !~</code>
Supported functions	None

Examples	<ul style="list-style-type: none"> Find issues with a resolution of "Cannot Reproduce" or "Won't Fix": <code>resolution in ("Cannot Reproduce", "Won't Fix")</code> Find issues with a resolution ID of 5: <code>resolution = 5</code> Find issues that do not have a resolution: <code>resolution = unresolved</code>
-----------------	---

[^ top of page](#)

Resolved

Search for issues that were resolved on, before, or after a particular date (or date range). Note that if a time-component is not specified, midnight will be assumed. Please note that the search results will be relative to your configured time zone (which is by default the Jira server's time zone).

Use one of the following formats:

"YYYY/MM/dd HH:mm"

"YYYY-MM-dd HH:mm"

"YYYY/MM/dd"

"YYYY-MM-dd"

Or use "w" (weeks), "d" (days), "h" (hours) or "m" (minutes) to specify a date relative to the current time. The default is "m" (minutes). Be sure to use quote-marks (""); if you omit the quote-marks, the number you supply will be interpreted as milliseconds after epoch (1970-1-1).

Syntax	<code>resolved</code>
Alias	<code>resolutionDate</code>
Field Type	DATE
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code> <code>IS , IS NOT , IN , NOT IN</code>
Unsupported operators	<code>~ , !~</code> <code>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Supported functions	When used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports: <ul style="list-style-type: none"> <code>currentLogin()</code> <code>lastLogin()</code> <code>now()</code> <code>startOfDay()</code> <code>startOfWeek()</code> <code>startOfMonth()</code> <code>startOfYear()</code> <code>endOfDay()</code> <code>endOfWeek()</code> <code>endOfMonth()</code> <code>endOfYear()</code>

Examples	<ul style="list-style-type: none"> • Find all issues that were resolved before 31st December 2010: <code>resolved <= "2010/12/31"</code> • Find all issues that were resolved before 2.00pm on 31st December 2010: <code>resolved < "2010/12/31 14:00"</code> • Find all issues that were resolved on or before 31st December 2010: <code>resolved <= "2011/01/01"</code> • Find issues that were resolved in January 2011: <code>resolved > "2011/01/01" and resolved < "2011/02/01"</code> • Find issues that were resolved on 15 January 2011: <code>resolved > "2011/01/15" and resolved < "2011/01/16"</code> • Find issues that were resolved in the last hour: <code>resolved > -1h</code>
-----------------	---

[^ top of page](#)

SLA

Used in Jira Service Desk only

Search for requests whose SLAs are in a certain

Syntax	<p>Time to resolution</p> <p>Time to first response</p> <p><your custom SLA name></p>
Field Type	SLA
Auto-complete	No
Supported operators	<code>= , !=, > , >= , < , <=</code>
Unsupported operators	<p><code>~ , !~</code></p> <p><code>IS , IS NOT , IN , NOT IN , WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED</code></p>
Supported functions	<ul style="list-style-type: none"> • <code>breached()</code> • <code>completed()</code> • <code>elapsed()</code> • <code>everBreached()</code> • <code>paused()</code> • <code>remaining()</code> • <code>running()</code> • <code>withinCalendarHours()</code>
Examples	<ul style="list-style-type: none"> • Find issues where Time to First Response was breached: <code>"Time to First Response" = everBreached()</code> • Find issues where the SLA for Time to Resolution is paused due to a condition: <code>"Time to Resolution" = paused()</code> • Find issues where the SLA for Time to Resolution is paused due to the SLA calendar: <code>"Time to Resolution" = withinCalendarHours()</code> • Find issues that have been waiting for a response for more than 1 hour: <code>"Time to First Response" > elapsed("1h")</code> • Find issues that that will breach Time to First Response in the next two hours: <code>"Time to First Response" < remaining("2h")</code>

[^ top of page](#)

Sprint

Search for issues that are assigned to a particular sprint. This works for active sprints and future sprints. The search is based on either the sprint name or the sprint ID (i.e. the number that Jira automatically allocates to a sprint).

If you have multiple sprints with similar (or identical) names, you can simply search by using the sprint name — or even just part of it. The possible matches will be shown in the autocomplete drop-down, with the sprint dates shown to help you distinguish between them. (The sprint ID will also be shown, in brackets).

Syntax	sprint
Field Type	NUMBER
Auto-complete	Yes
Supported operators	= , != IS , IS NOT, IN , NOT IN
Unsupported operators	~ , !~ , > , >= , < , <= WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED
Supported functions	<ul style="list-style-type: none"> • openSprints() • closedSprints()
Examples	<ul style="list-style-type: none"> • Find issues that belong to sprint 999: sprint = 999 • Find issues that belong to sprint "February 1": sprint = "February 1" • Find issues that belong to either "February 1", "February 2" or "February 3": sprint in ("February 1", "February 2", "February 3") • Find issues that are assigned to a sprint: sprint is not empty

[^ top of page](#)

Status

Search for issues that have a particular status. You can search by status name or status ID (i.e. the number that Jira automatically allocates to a status).

Note:

- It is safer to search by status ID than status name. It is possible for your Jira administrator to change the name of a status, which could break any saved filter that rely on that name. Status IDs, however, are unique and cannot be changed.
- The WAS, WAS NOT, WAS IN and WAS NOT IN operators can only be used with the name, not the ID.

Syntax	status
Field Type	STATUS
Auto-complete	Yes
Supported operators	= , != IS , IS NOT, IN , NOT IN , WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED
Unsupported operators	~ , !~ , > , >= , < , <=

Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues with a status of "Open": status = Open Find issues with a status ID of 1: status = 1 Find issues that currently have, or previously had, a status of "Open": status WAS Open

[^ top of page](#)

Summary

Search for issues where the summary contains particular text. [Jira text-search syntax](#) can be used.

Syntax	summary
Field Type	TEXT
Auto-complete	No
Supported operators	~ , !~ IS , IS NOT
Unsupported operators	= , != , > , >= , < , <= IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues where the summary contains text that matches "Error saving file" (i.e. a "fuzzy" match): summary ~ "Error saving file" Find issues where the summary contains the exact phrase "Error saving file": summary ~ "\"Error saving file\""

[^ top of page](#)

Text

This is a "master-field" that allows you to search all text fields, i.e.:

- Summary
- Description
- Environment
- Comments
- custom fields that use the "free text searcher"; this includes custom fields of the following built-in custom field types:
 - Free text field (unlimited text)
 - Text field (< 255 characters)
 - Read-only text field

Notes:

- The `text` master-field can only be used with the CONTAINS operator ("~" and "!~").
- [Jira text-search syntax](#) can be used with these fields.

Syntax	text
Field Type	TEXT

Auto-complete	No
Supported operators	~
Unsupported operators	= , != , !~ , > , >= , < , <= IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues where a text field matches the word "Fred": text ~ "Fred" or text ~ Fred Find all issues where a text field contains the exact phrase "full screen": text ~ "\"full screen\""

[^ top of page](#)

Time spent

Only available if time-tracking has been enabled by your Jira administrator.

Search for issues where the time spent is set to a particular value (i.e. a number, not a date or date range). Use "w", "d", "h" and "m" to specify weeks, days, hours, or minutes.

Syntax	timeSpent
Field Type	DURATION
Auto-complete	No
Supported operators	= , != , > , >= , < , <= IS , IS NOT , IN , NOT IN
Unsupported operators	~ , !~ WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues where the time spent is more than 5 days: timeSpent > 5d

[^ top of page](#)

Type

Search for issues that have a particular issue type. You can search by issue type name or issue type ID (i.e. the number that Jira automatically allocates to an issue type).

Note, it is safer to search by type ID than type name. It is possible for your Jira administrator to change the name of a type, which could break any saved filter that rely on that name. Type IDs, however, are unique and cannot be changed.

Syntax	type
Alias	issueType
Field Type	ISSUE_TYPE

Auto-complete	Yes
Supported operators	= , != IS , IS NOT , IN , NOT IN
Unsupported operators	~ , !~ , > , >= , < , <= WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues with an issue type of "Bug": type = Bug Find issues with an issue type of "Bug" or "Improvement": issueType in (Bug,Improvement) Find issues with an issue type ID of 2: issueType = 2

[^ top of page](#)

Updated

Search for issues that were last updated on, before, or after a particular date (or date range). Note that if a time-component is not specified, midnight will be assumed. Please note that the search results will be relative to your configured time zone (which is by default the Jira server's time zone).

Use one of the following formats:

"YYYY/MM/dd HH:mm"

"YYYY-MM-dd HH:mm"

"YYYY/MM/dd"

"YYYY-MM-dd"

Or use "w" (weeks), "d" (days), "h" (hours) or "m" (minutes) to specify a date relative to the current time. The default is "m" (minutes). Be sure to use quote-marks (""); if you omit the quote-marks, the number you supply will be interpreted as milliseconds after epoch (1970-1-1).

Syntax	updated
Alias	updatedAt
Field Type	DATE
Auto-complete	No
Supported operators	= , != , > , >= , < , <= IS , IS NOT , IN , NOT IN
Unsupported operators	~ , !~ WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED

Supported functions	<p>When used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> • <code>currentLogin()</code> • <code>lastLogin()</code> • <code>now()</code> • <code>startOfDay()</code> • <code>startOfWeek()</code> • <code>startOfMonth()</code> • <code>startOfYear()</code> • <code>endOfDay()</code> • <code>endOfWeek()</code> • <code>endOfMonth()</code> • <code>endOfYear()</code>
Examples	<ul style="list-style-type: none"> • Find issues that were last updated before 12th December 2010: <code>updated < "2010/12/12"</code> • Find issues that were last updated on or before 12th December 2010: <code>updated < "2010/12/13"</code> • Find all issues that were last updated before 2.00pm on 31st December 2010: <code>updated < "2010/12/31 14:00"</code> • Find issues that were last updated more than two weeks ago: <code>updated < "-2w"</code> • Find issues that were last updated on 15 January 2011: <code>updated > "2011/01/15" and updated < "2011/01/16"</code> • Find issues that were last updated in January 2011: <code>updated > "2011/01/01" and updated < "2011/02/01"</code>

[^ top of page](#)

Voter

Search for issues for which a particular user has voted. You can search by the user's full name, ID, or email address. Note that you can only find issues for which you have the "View Voters and Watchers" permission, unless you are searching for your own votes. See also [votedIssues](#).

Syntax	<code>voter</code>
Field Type	USER
Auto-complete	Yes
Supported operators	<code>= , !=</code> <code>IS , IS NOT , IN , NOT IN</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <=</code> <code>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Supported functions	<p>When used with the IN and NOT IN operators, this field supports:</p> <ul style="list-style-type: none"> • <code>membersOf()</code> <p>When used with the EQUALS and NOT EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> • <code>currentUser()</code>

Examples	<ul style="list-style-type: none"> Search for issues that you have voted for: <code>voter = currentUser()</code> Search for issues that the user "jsmith" has voted for: <code>voter = "jsmith"</code> Search for issues for which a member of the group "Jira-administrators" has voted: <code>voter in membersOf("Jira-administrators")</code>
-----------------	---

[^ top of page](#)

Votes

Search for issues with a specified number of votes.

Syntax	<code>votes</code>
Field Type	NUMBER
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code> <code>IN , NOT IN</code>
Unsupported operators	<code>~ , !~</code> <code>IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find all issues that have 12 or more votes: <code>votes >= 12</code>

[^ top of page](#)

Watcher

Search for issues that a particular user is watching. You can search by the user's full name, ID, or email address. Note that you can only find issues for which you have the "View Voters and Watchers" permission, unless you are searching for issues where you are the watcher. See also [watchedIssues](#).

Syntax	<code>watcher</code>
Field Type	USER
Auto-complete	Yes
Supported operators	<code>= , !=</code> <code>IS , IS NOT , IN , NOT IN</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <=</code> <code>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Supported functions	<p>When used with the IN and NOT IN operators, this field supports:</p> <ul style="list-style-type: none"> <code>membersOf()</code> <p>When used with the EQUALS and NOT EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> <code>currentUser()</code>

Examples	<ul style="list-style-type: none"> • Search for issues that you are watching: <code>watcher = currentUser()</code> • Search for issues that the user "jsmith" is watching: <code>watcher = "jsmith"</code> • Search for issues that are being watched by a member of the group "Jira-administrators": <code>watcher in membersOf("Jira-administrators")</code>
-----------------	---

[^ top of page](#)

Watchers

Search for issues with a specified number of watchers.

Syntax	<code>watchers</code>
Field Type	NUMBER
Auto-complete	No
Supported operators	<code>= , != , > , >= , < , <=</code> <code>IN , NOT IN</code>
Unsupported operators	<code>~ , !~</code> <code>IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Supported functions	<p>When used with the IN and NOT IN operators, this field supports:</p> <ul style="list-style-type: none"> • <code>membersOf()</code> <p>When used with the EQUALS and NOT EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> • <code>currentUser()</code>
Examples	<ul style="list-style-type: none"> • Find all issues that are being watched by more than 3 people: <code>watchers > 3</code>

[^ top of page](#)

Work log author

Only available if time-tracking has been enabled by your Jira administrator.

Search for issues a particular user has logged work against. You can search by the user's full name, ID, or email address. Note that you can only find issues for which you have "Time Tracking" permissions, unless you are searching for issues that you've logged work against.

Syntax	<code>worklogAuthor</code>
Field Type	USER
Auto-complete	Yes
Supported operators	<code>= , !=</code> <code>IS , IS NOT , IN , NOT IN</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <=</code> <code>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>

Supported functions	<p>When used with the IN and NOT IN operators, this field supports:</p> <ul style="list-style-type: none"> membersOf() <p>When used with the EQUALS and NOT EQUALS operators, this field supports:</p> <ul style="list-style-type: none"> currentUser()
Examples	<ul style="list-style-type: none"> Search for issues that you've logged work against: <code>worklogAuthor = currentUser()</code> Search for issues that the user "jsmith" has logged work against: <code>worklogAuthor = "jsmith"</code> Search for issues that a member of the group "Jira-software-users": <code>worklogAuthor in membersOf("Jira-software-users")</code>

[^ top of page](#)

Work log comment

Only available if time-tracking has been enabled by your Jira administrator.

Search for issues that have a comment in a work log entry which contains particular text. [Jira text-search syntax](#) can be used.

Syntax	<code>worklogComment</code>
Field Type	TEXT
Auto-complete	No
Supported operators	<code>~ , !~</code>
Unsupported operators	<code>= , != , > , >= , < , <=</code> <code>IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues where a comment in a work log entry contains text that matches "test sessions" (i.e. a "fuzzy" match): <code>comment ~ "test sessions"</code> Find issues where a comment contains the exact phrase "test sessions": <code>summary ~ "\"test sessions\""</code>

[^ top of page](#)

Work log date

Only available if time-tracking has been enabled by your Jira administrator.

Search for issues that have comments in work log entries that were created on, before, or after a particular date (or date range). Note that if a time-component is not specified, midnight 00:00 will be assumed. Please note that the search results will be relative to your configured time zone (which is by default the Jira server's time zone).

Use one of the following formats:

`"YYYY/MM/dd HH:mm"`

`"YYYY-MM-dd HH:mm"`

`"YYYY/MM/dd"`

`"YYYY-MM-dd"`

Or use "w" (weeks), "d" (days), "h" (hours) or "m" (minutes) to specify a date relative to the current time. The default is "m" (minutes). Be sure to use quote-marks (""); if you omit the quote-marks, the number you supply will be interpreted as milliseconds after epoch (1970-1-1).

Syntax	worklogDate
Field Type	DATE
Auto-complete	No
Supported operators	= , != , > , >= , < , <= IS , IS NOT , IN , NOT IN
Unsupported operators	~ , !~ WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Supported functions	When used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports: <ul style="list-style-type: none"> • currentLogin() • lastLogin() • now() • startOfDay() • startOfWeek() • startOfMonth() • startOfYear() • endOfDay() • endOfWeek() • endOfMonth() • endOfYear()
Examples	<ul style="list-style-type: none"> • Find issues that have comments in work log entries created before midnight 00:00 12th December 2010: worklogDate < "2010/12/12" • Find issues that have comments in work log entries created on or before 12th December 2010 (but not 13th December 2010): worklogDate <= "2010/12/13" • Find issues that have comments in work log entries created on 12th December 2010 before 2:00pm: worklogDate > "2010/12/12" and worklogDate < "2010/12/12 14:00" • Find issues that have comments in work log entries created less than one day ago: worklogDate > "-1d" • Find issues that have comments in work log entries created in January 2011: worklogDate > "2011/01/01" and worklogDate < "2011/02/01" • Find issues that have comments in work log entries created on 15 January 2011: worklogDate > "2011/01/15" and worklogDate < "2011/01/16"

[^ top of page](#)

Work ratio

Only available if time-tracking has been enabled by your Jira administrator.

Search for issues where the work ratio has a particular value. Work ratio is calculated as follows: **workRatio = timeSpent / originalEstimate) x 100**

Syntax	workRatio
---------------	-----------

Field Type	NUMBER
Auto-complete	No
Supported operators	= , != , > , >= , < , <= IS , IS NOT , IN , NOT IN
Unsupported operators	~ , !~ WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues on which more than 75% of the original estimate has been spent: workRatio > 75

[^ top of page](#)

Advanced searching - keywords reference

This page describes information about keywords that are used for advanced searching. See [Advanced searching](#).

A keyword in JQL is a word or phrase that does (or is) any of the following:

- joins two or more clauses together to form a complex JQL query
- alters the logic of one or more clauses
- alters the logic of [operators](#)
- has an explicit definition in a JQL query
- performs a specific function that alters the results of a JQL query

On this page:

- [AND](#)
- [OR](#)
- [NOT](#)
- [EMPTY](#)
- [NULL](#)
- [ORDER BY](#)

AND

Used to combine multiple clauses, allowing you to refine your search.

Note that you can use parentheses to control the order in which clauses are executed.

Examples

- Find all open issues in the "New office" project:

```
project = "New office" and status = "open"
```

- Find all open, urgent issues that are assigned to jsmith:

```
status = open and priority = urgent and assignee = jsmith
```

- Find all issues in a particular project that are not assigned to jsmith:

```
project = JRA and assignee != jsmith
```

- Find all issues for a specific release which consists of different version numbers across several projects:

```
project in (JRA,CONF) and fixVersion = "3.14"
```

- Find all issues where neither the Reporter nor the Assignee is Jack, Jill or John:

```
reporter not in (Jack,Jill,John) and assignee not in
(Jack,Jill,John)
```

[^ top of page](#)

OR

Used to combine multiple clauses, allowing you to expand your search.

Note that you can use parentheses to control the order in which clauses are executed.

(Note: also see [IN](#), which can be a more convenient way to search for multiple values of a field.)

Examples

- Find all issues that were created by either jsmith or jbrown:

```
reporter = jsmith or reporter = jbrown
```

- Find all issues that are overdue or where no due date is set:

```
duedate < now() or duedate is empty
```

[^ top of page](#)

NOT

Used to negate individual clauses or a complex JQL query (a query made up of more than one clause) using parentheses, allowing you to refine your search.

(Note: also see [NOT EQUALS](#) ("!="), [DOES NOT CONTAIN](#) ("!~"), [NOT IN](#) and [IS NOT](#).)

Examples

- Find all issues that are assigned to any user except jsmith:

```
not assignee = jsmith
```

- Find all issues that were not created by either jsmith or jbrown:

```
not (reporter = jsmith or reporter = jbrown)
```

[^ top of page](#)

EMPTY

Used to search for issues where a given field does not have a value. See also [NULL](#).

Note that **EMPTY** can only be used with fields that support the **IS** and **IS NOT** operators. To see a field's supported operators, check the individual [field](#) reference.

Examples

- Find all issues without a DueDate:

```
duedate = empty
```

or

```
duedate is empty
```

[^ top of page](#)

NULL

Used to search for issues where a given field does not have a value. See also [EMPTY](#).

Note that NULL can only be used with fields that support the **IS** and **IS NOT** operators. To see a field's supported operators, check the individual [field](#) reference.

Examples

- Find all issues without a DueDate:

```
duedate = null
```

or

```
duedate is null
```

[^ top of page](#)

ORDER BY

Used to specify the fields by whose values the search results will be sorted.

By default, the field's own sorting order will be used. You can override this by specifying ascending order ("asc") or descending order ("desc").

Examples

- Find all issues without a DueDate, sorted by CreationDate:

```
duedate = empty order by created
```

- Find all issues without a DueDate, sorted by CreationDate, then by Priority (highest to lowest):

```
duedate = empty order by created, priority desc
```

- Find all issues without a DueDate, sorted by CreationDate, then by Priority (lowest to highest):

```
duedate = empty order by created, priority asc
```

Ordering by **Components** or **Versions** will list the returned issues first by **Project**, and only then by the field's natural order (see [JRA-31113](#)).

[^ top of page](#)

Advanced searching - operators reference

This page describes information about operators that are used for advanced searching.

An operator in JQL is one or more symbols or words, which compares the value of a [field](#) on its left with one or more values (or [functions](#)) on its right, such that only true results are retrieved by the clause. Some operators may use the [NOT](#) keyword.

EQUALS: =

The "=" operator is used to search for issues where the value of the specified field exactly matches the specified value. (Note: cannot be used with text fields; see the [CONTAINS](#) operator instead.)

To find issues where the value of a specified field exactly matches *multiple* values, use multiple "=" statements with the [AND](#) operator.

Examples

- Find all issues that were created by jsmith:

```
reporter = jsmith
```

- Find all issues that were created by John Smith:

```
reporter = "John Smith"
```

[^top of page](#)

On this page:

- [EQUALS: =](#)
- [NOT EQUALS: !=](#)
- [GREATER THAN: >](#)
- [GREATER THAN EQUALS: >=](#)
- [LESS THAN: <](#)
- [LESS THAN EQUALS: <=](#)
- [IN](#)
- [NOT IN](#)
- [CONTAIN S: ~](#)
- [DOES NOT CONTAIN: !~](#)
- [IS](#)
- [IS NOT](#)
- [WAS](#)
- [WAS IN](#)
- [WAS NOT IN](#)
- [WAS NOT](#)
- [CHANGED](#)

NOT EQUALS: !=

The "!=" operator is used to search for issues where the value of the specified field does not match the specified value. (Note: cannot be used with text fields; see the [DOES NOT MATCH](#) ("!~") operator instead.)

Note that typing `field != value` is the same as typing `NOT field = value`, and that `field != EMPTY` is the same as `field IS_NOT EMPTY`.

The "!=" operator will not match a field that has no value (i.e. a field that is empty). For example, `component != fred` will only match issues that have a component **and** the component is not "fred". To find issues that have a component other than "fred" **or have no component**, you would need to type: `component != fred` or `component is empty`.

Examples

- Find all issues that are assigned to any user except jsmith:

```
not assignee = jsmith
```

or:

```
assignee != jsmith
```

- Find all issues that are not assigned to jsmith:

```
assignee != jsmith or assignee is empty
```

- Find all issues that were reported by me but are not assigned to me:

```
reporter = currentUser() and assignee != currentUser()
```

- Find all issues where the Reporter or Assignee is anyone except John Smith:

```
assignee != "John Smith" or reporter != "John Smith"
```

- Find all issues that are not unassigned:

```
assignee is not empty
```

or

```
assignee != null
```

[^top of page](#)

GREATER THAN: >

The ">" operator is used to search for issues where the value of the specified field is greater than the specified value.

Note that the ">" operator can only be used with fields that support ordering (e.g. date fields and version fields), and cannot be used with text fields. To see a field's supported operators, check the individual field reference.

Examples

- Find all issues with more than 4 votes:

```
votes > 4
```

- Find all overdue issues:

```
duedate < now() and resolution is empty
```

- Find all issues where priority is higher than "Normal":

```
priority > normal
```

[^top of page](#)

GREATER THAN EQUALS: >=

The ">=" operator is used to search for issues where the value of the specified field is greater than or equal to the specified value.

Note that the ">=" operator can only be used with fields that support ordering (e.g. date fields and version fields), and cannot be used with text fields. To see a field's supported operators, check the individual field reference.

Examples

- Find all issues with 4 or more votes:

```
votes >= 4
```

- Find all issues due on or after 31/12/2008:

```
duedate >= "2008/12/31"
```

- Find all issues created in the last five days:

```
created >= "-5d"
```

[^top of page](#)

LESS THAN: <

The "<" operator is used to search for issues where the value of the specified field is less than the specified value.

Note that the "<" operator can only be used with fields which support ordering (e.g. date fields and version fields), and cannot be used with text fields. To see a field's supported operators, check the individual field reference.

Examples

- Find all issues with less than 4 votes:

```
votes < 4
```

[^top of page](#)

LESS THAN EQUALS: <=

The "<=" operator is used to search for issues where the value of the specified field is less than or equal to than the specified value.

Note that the "<=" operator can only be used with fields which support ordering (e.g. date fields and version fields), and cannot be used with text fields. To see a field's supported operators, check the individual field reference.

Examples

- Find all issues with 4 or fewer votes:

```
votes <= 4
```

- Find all issues that have not been updated in the past month (30 days):

```
updated <= "-4w 2d"
```

[^top of page](#)

IN

The "IN" operator is used to search for issues where the value of the specified field is one of multiple specified values. The values are specified as a comma-delimited list, surrounded by parentheses.

Using "IN" is equivalent to using multiple `EQUALS (=)` statements, but is shorter and more convenient. That is, typing `reporter IN (tom, jane, harry)` is the same as typing `reporter = "tom" OR reporter = "jane" OR reporter = "harry"`.

Examples

- Find all issues that were created by either jsmith or jbrown or jjones:

```
reporter in (jsmith,jbrown,jjones)
```

- Find all issues where the Reporter or Assignee is either Jack or Jill:

```
reporter in (Jack,Jill) or assignee in (Jack,Jill)
```

- Find all issues in version 3.14 or version 4.2:

```
affectedVersion in ("3.14", "4.2")
```

[^top of page](#)

NOT IN

The "NOT IN" operator is used to search for issues where the value of the specified field is not one of multiple specified values.

Using "NOT IN" is equivalent to using multiple `NOT_EQUALS (!=)` statements, but is shorter and more convenient. That is, typing `reporter NOT IN (tom, jane, harry)` is the same as typing `reporter != "tom" AND reporter != "jane" AND reporter != "harry"`.

The "NOT IN" operator will not match a field that has no value (i.e. a field that is empty). For example, `assignee not in (jack,jill)` will only match issues that have an assignee **and** the assignee is not "jack" or "jill". To find issues that are assigned to someone other than "jack" or "jill" **or are unassigned**, you would need to type: `assignee not in (jack,jill) or assignee is empty`.

Examples

- Find all issues where the Assignee is someone other than Jack, Jill, or John:

```
assignee not in (Jack,Jill,John)
```

- Find all issues where the Assignee is not Jack, Jill, or John:

```
assignee not in (Jack,Jill,John) or assignee is empty
```

- Find all issues where the FixVersion is not 'A', 'B', 'C', or 'D':

```
FixVersion not in (A, B, C, D)
```

- Find all issues where the FixVersion is not 'A', 'B', 'C', or 'D', or has not been specified:

```
FixVersion not in (A, B, C, D) or FixVersion is empty
```

[^top of page](#)

CONTAINS: ~

The "~" operator is used to search for issues where the value of the specified field matches the specified value (either an exact match or a "fuzzy" match — see examples below). For use with version and text fields only, i.e:

Text fields:

- Summary
- Description
- Environment
- Comments
- Custom fields that use the "Free Text Searcher"; this includes custom fields of the following built-in Custom Field Types
 - Free Text Field (unlimited text)
 - Text Field (< 255 characters)
 - Read-only Text Field

Version fields:

- Affected version
- Fix version
- Custom fields that use the "Version Picker".

The JQL field "text" as in `text ~ "some words"` searches an issue's Summary, Description, Environment, Comments. It also searches all text custom fields. If you have many text custom fields you can improve performance of your queries by searching on specific fields, e.g.

```
Summary ~ "some words" OR Description ~ "some words"
```

Note: when using the "~" operator, the value on the right-hand side of the operator can be specified using [Jira a text-search syntax](#).

Examples

- Find all issues where the Summary contains the word "win" (or simple derivatives of that word, such as "wins"):

```
summary ~ win
```

Note that for version fields, the ~ operator returns an exact match. For example, to find the version "9.0", you would use the following query:

```
fixVersion ~ "9.0"
```

- Find all issues where the Summary contains a wild-card match for the word "win":

```
summary ~ "win*"
```

- Find all issues where the Summary contains the word "issue" and the word "collector":

```
summary ~ "issue collector"
```

- Find all issues where the Summary contains the exact phrase "full screen" (see [Search syntax for text fields](#) for details on how to escape quote-marks and other special characters):

```
summary ~ "\"full screen\""
```

- Find all issues where the Fix Version field contains a wild-card match for version "9", e.g. 9.1 or 9.0.1:

```
fixVersion ~ "9*"
```

- Find all issues where the Fix Version field contains "9", e.g. 0.9.1 or 9.1:

```
fixVersion ~ "*9*"
```

[^top of page](#)

DOES NOT CONTAIN: !~

The "!~" operator is used to search for issues where the value of the specified field matches the specified value. For use with version and text fields only, i.e.:

Text fields:

- Summary
- Description
- Environment
- Comments
- Custom fields that use the "Free Text Searcher"; this includes custom fields of the following built-in Custom Field Types
 - Free Text Field (unlimited text)
 - Text Field (< 255 characters)
 - Read-only Text Field

Version fields:

- Affected version
- Fix version
- Custom fields that use the "Version Picker".

The JQL field "text" as in `text ~ "some words"` searches an issue's Summary, Description, Environment, Comments. It also searches all text custom fields. If you have many text custom fields you can improve performance of your queries by searching on specific fields, e.g.

```
Summary ~ "some words" OR Description ~ "some words"
```

Note: when using the "!~" operator, the value on the right-hand side of the operator can be specified using [Jira text-search syntax](#).

Examples

- Find all issues where the Summary does not contain the word "run" (or derivatives of that word, such as "running" or "ran"):

```
summary !~ run
```

Note that for version fields, the ~ operator returns an exact match. For example, to find issues where

Fix Version is not "9.0", you would use the following query:

```
fixVersion !~ "9.0"
```

- Find all issues where the Fix Version field does not contain any version from the 9.x line:

```
fixVersion !~ "9.*"
```

[^top of page](#)

IS

The "IS" operator can only be used with **EMPTY** or **NULL**. That is, it is used to search for issues where the specified field has no value.

Note that not all fields are compatible with this operator; see the individual field reference for details.

Examples

- Find all issues that have no Fix Version:

```
fixVersion is empty
```

or

```
fixVersion is null
```

[^top of page](#)

IS NOT

The "IS NOT" operator can only be used with **EMPTY** or **NULL**. That is, it is used to search for issues where the specified field has a value.

Note that not all fields are compatible with this operator; see the individual field reference for details.

Examples

- Find all issues that have one or more votes:

```
votes is not empty
```

or

```
votes is not null
```

[^top of page](#)

WAS

The "WAS" operator is used to find issues that currently have or previously had the specified value for the specified field.

This operator has the following optional predicates:

- AFTER "date"
- BEFORE "date"

- BY "username"
- DURING ("date1", "date2")
- ON "date"

This operator will match the value name (e.g. "Resolved"), which was configured in your system *at the time that the field was changed*. This operator will also match the value ID associated with that value name too — that is, it will match "4" as well as "Resolved".

(Note: This operator can be used with the Assignee, Fix Version, Priority, Reporter, Resolution, and Status fields only.)

Examples

- Find issues that currently have or previously had a status of 'In Progress':

```
status WAS "In Progress"
```

- Find issues that were resolved by Joe Smith before 2nd February:

```
status WAS "Resolved" BY jsmith BEFORE "2011/02/02"
```

- Find issues that were resolved by Joe Smith during 2010:

```
status WAS "Resolved" BY jsmith DURING
("2010/01/01", "2011/01/01")
```

[^top of page](#)

WAS IN

The "WAS IN" operator is used to find issues that currently have or previously had any of multiple specified values for the specified field. The values are specified as a comma-delimited list, surrounded by parentheses.

Using "WAS IN" is equivalent to using multiple **WAS** statements, but is shorter and more convenient. That is, typing `status WAS IN ('Resolved', 'Closed')` is the same as typing `status WAS "Resolved" OR status WAS "Closed"`.

This operator has the following optional predicates:

- AFTER "date"
- BEFORE "date"
- BY "username"
- DURING ("date1", "date2")
- ON "date"

This operator will match the value name (e.g. "Resolved"), which was configured in your system *at the time that the field was changed*. This operator will also match the value ID associated with that value name too — that is, it will match "4" as well as "Resolved".

(Note: This operator can be used with the Assignee, Fix Version, Priority, Reporter, Resolution, and Status fields only.)

Examples

- Find all issues that currently have, or previously had, a status of 'Resolved' or 'In Progress':

```
status WAS IN ("Resolved", "In Progress")
```

[^top of page](#)

WAS NOT IN

The "WAS NOT IN" operator is used to search for issues where the value of the specified field has never been one of multiple specified values.

Using "WAS NOT IN" is equivalent to using multiple `WAS_NOT` statements, but is shorter and more convenient. That is, typing `status WAS NOT IN ("Resolved", "In Progress")` is the same as typing `status WAS NOT "Resolved" AND status WAS NOT "In Progress"`.

This operator has the following optional predicates:

- AFTER "date"
- BEFORE "date"
- BY "username"
- DURING ("date1", "date2")
- ON "date"

This operator will match the value name (e.g. "Resolved"), which was configured in your system *at the time that the field was changed*. This operator will also match the value ID associated with that value name, too — that is, it will match "4" as well as "Resolved".

(Note: This operator can be used with the Assignee, Fix Version, Priority, Reporter, Resolution, and Status fields only.)

Examples

- Find issues that have never had a status of 'Resolved' or 'In Progress':

```
status WAS NOT IN ("Resolved", "In Progress")
```

- Find issues that did not have a status of 'Resolved' or 'In Progress' before 2nd February:

```
status WAS NOT IN ("Resolved", "In Progress") BEFORE "2011/02/02"
```

[^top of page](#)

WAS NOT

The "WAS NOT" operator is used to find issues that have never had the specified value for the specified field.

This operator has the following optional predicates:

- AFTER "date"
- BEFORE "date"
- BY "username"
- DURING ("date1", "date2")
- ON "date"

This operator will match the value name (e.g. "Resolved"), which was configured in your system *at the time that the field was changed*. This operator will also match the value ID associated with that value name too — that is, it will match "4" as well as "Resolved".

(Note: This operator can be used with the Assignee, Fix Version, Priority, Reporter, Resolution, and Status fields only.)

Examples

- Find issues that do not have, and have never had a status of 'In Progress':

```
status WAS NOT "In Progress"
```

- Find issues that did not have a status of 'In Progress' before 2nd February:

```
status WAS NOT "In Progress" BEFORE "2011/02/02"
```

[^top of page](#)

CHANGED

The "CHANGED" operator is used to find issues that have a value that had changed for the specified field.

This operator has the following optional predicates:

- AFTER "date"
- BEFORE "date"
- BY "username"
- DURING ("date1", "date2")
- ON "date"
- FROM "oldvalue"
- TO "newvalue"

(Note: This operator can be used with the Assignee, Fix Version, Priority, Reporter, Resolution, and Status fields only.)

Examples

- Find issues whose assignee had changed:

```
assignee CHANGED
```

- Find issues whose status had changed from 'In Progress' back to 'Open':

```
status CHANGED FROM "In Progress" TO "Open"
```

- Find issues whose priority was changed by user 'freddo' after the start and before the end of the current week.

```
priority CHANGED BY freddo BEFORE endOfWeek() AFTER
startOfWeek()
```

[^top of page](#)

Advanced searching - functions reference

This page describes information about functions that are used for advanced searching.

A function in JQL appears as a word followed by parentheses, which may contain one or more explicit values or Jira fields. In a clause, a function is preceded by an [operator](#), which in turn is preceded by a [field](#). A function performs a calculation on either specific Jira data or the function's content in parentheses, such that only true results are retrieved by the function, and then again by the clause in which the function is used.

Unless specified in the search query, note that JQL searches do not return empty fields in results. To include empty fields (e.g. unassigned issues) when searching for issues that are not assigned

On this page:

- [approved\(\)](#)
- [approver\(\)](#)
- [breached\(\)](#)
- [cascadeOption\(\)](#)
- [closedSprints\(\)](#)
- [completed\(\)](#)

to the current user, you would enter (assignee != currentUser() OR assignee is EMPTY) to include unassigned issues in the list of results.

approved()

Only applicable if Jira Service Desk is installed and licensed.

Search for issues that required approval and have a final decision of approved.

Syntax	approved()
Supported fields	Custom fields of type Approval
Supported operators	=
Unsupported operators	~ , != , !~ , > , >= , < , <= IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find all issues that are approved: approval = approved()

[^ top of page](#)

- component sLeadByUser()
- currentLogin()
- currentUser()
- earliestUnreleasedVersion()
- elapsed()
- endOfDay()
- endOfMonth()
- endOfWeek()
- endOfYear()
- everbreached()
- issueHistory()
- issuesWithRemoteLinksByGlobalId()
- lastLogin()
- latestReleasedVersion()
- linkedIssues()
- membersOf()
- myApproval()
- myPending()
- now()
- openSprints()
- paused()
- pending()
- pendingBy()
- projectsLeadByUser()
- projectsWhereUserHasPermission()
- projectsWhereUserHasRole()
- releasedVersions()
- remaining()
- running()

- standardIssueTypes()
- startOfDay()
- startOfMonth()
- startOfWeek()
- startOfYear()
- subtaskIssueTypes()
- unreleasedVersions()
- updatedBy()
- votedIssues()
- watchedIssues()
- withinCalendarHours()

approver()

Only applicable if Jira Service Desk is installed and licensed.

Search for issues that require or required approval by the listed user/s. This uses an OR operator, and you must specify the username/s.

Syntax	<code>approver (user , user)</code>
Supported fields	Custom fields of type Approval
Supported operators	=
Unsupported operators	~ , != , !~ , > , >= , < , <= IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> • Find issues that require or required approval by John Smith: <code>approval = approver(jsmith)</code> • Find issues that require or required approval by John Smith or Sarah Khan: <code>approval = approver(jsmith,skhan)</code>

[^ top of page](#)

breached()

Only applicable if Jira Service Desk is installed and licensed.

Returns issues that whose most recent SLA has missed its goal.

Syntax	<code>breached ()</code>
Supported fields	SLA

Supported operators	<code>= , !=</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <= , IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Examples	<ul style="list-style-type: none"> Find issues where Time to First Response was breached: <code>"Time to First Response" = breached()</code>

[^ top of page](#)

cascadeOption()

Search for issues that match the selected values of a 'cascading select' custom field.

The *parentOption* parameter matches against the first tier of options in the cascading select field. The *childOption* parameter matches against the second tier of options in the cascading select field, and is optional.

The keyword "none" can be used to search for issues where either or both of the options have no value.

Syntax	<pre>cascadeOption(parentOption) cascadeOption(parentOption,childOption)</pre>
Supported fields	Custom fields of type 'Cascading Select'
Supported operators	<code>IN , NOT IN</code>
Unsupported operators	<code>= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Examples	<ul style="list-style-type: none"> Find issues where a custom field ("Location") has the value "USA" for the first tier and "New York" for the second tier: <code>location in cascadeOption("USA", "New York")</code> Find issues where a custom field ("Location") has the value "USA" for the first tier and any value (or no value) for the second tier: <code>location in cascadeOption("USA")</code> Find issues where a custom field ("Location") has the value "USA" for the first tier and no value for the second tier: <code>location in cascadeOption("USA" ,none)</code> Find issues where a custom field ("Location") has no value for the first tier and no value for the second tier: <code>location in cascadeOption(none)</code> Find issues where a custom field ("Referrer") has the value "none" for the first tier and "none" for the second tier: <code>referrer in cascadeOption("\"none\"" , "\"none\"")</code> Find issues where a custom field ("Referrer") has the value "none" for the first tier and no value for the second tier: <code>referrer in cascadeOption("\"none\"" ,none)</code>

[^ top of page](#)

closedSprints()

Search for issues that are assigned to a completed Sprint. Note, it is possible for an issue to belong to both a completed Sprint(s) and an incomplete Sprint(s). See also [openSprints\(\)](#).

Syntax	<code>closedSprints()</code>
---------------	------------------------------

Supported fields	Sprint
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find all issues that are assigned to a completed sprint: <code>sprint in closedSprints()</code>

[^ top of page](#)

completed()

Only applicable if Jira Service Desk is installed and licensed.

Returns issues that have an SLA that has completed at least one cycle.

Syntax	<code>completed()</code>
Supported fields	SLA
Supported operators	= , !=
Unsupported operators	= , ~ , !~ , > , >= , < , <= , IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues where Time to First Response has completed at least one cycle: <code>"Time to First Response" = completed()</code>

[^ top of page](#)

componentsLeadByUser()

Find issues in components that are led by a specific user. You can optionally specify a user, or if the user is omitted, the current user (i.e. you) will be used. Note that if you are not logged in to Jira, a user must be specified.

Syntax	<code>componentsLeadByUser()</code> <code>componentsLeadByUser(username)</code>
Supported fields	Component
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find open issues in components that are led by you: <code>component in componentsLeadByUser() AND status = Open</code> Find open issues in components that are led by Bill: <code>component in componentsLeadByUser(bill) AND status = Open</code>

[^ top of page](#)

currentLogin()

Perform searches based on the time at which the current user's session began. See also [lastLogin](#).

Syntax	<code>currentLogin()</code>
Supported fields	Created, Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	<code>= , != , > , >= , < , <=</code> <code>WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED*</code> <i>* Only in predicate</i>
Unsupported operators	<code>~ , !~ IS , IS NOT , IN , NOT IN</code>
Examples	<ul style="list-style-type: none"> Find issues that have been created during my current session: <code>created > currentLogin()</code>

[^ top of page](#)

currentUser()

Perform searches based on the currently logged-in user. Note, this function can only be used by logged-in users. So if you are creating a saved filter that you expect to be used by anonymous users, do not use this function.

Syntax	<code>currentUser()</code>
Supported fields	Assignee, Reporter, Voter, Watcher, custom fields of type User
Supported operators	<code>= , !=</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <= IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Examples	<ul style="list-style-type: none"> Find issues that are assigned to me: <code>assignee = currentUser()</code> Find issues that were reported to me but are not assigned to me: <code>reporter = currentUser() AND (assignee != currentUser() OR assignee is EMPTY)</code>

[^ top of page](#)

earliestUnreleasedVersion()

Perform searches based on the earliest unreleased version (i.e. next version that is due to be released) of a specified project. See also [unreleasedVersions](#). Note, the "earliest" is determined by the ordering assigned to the versions, not by actual Version Due Dates.

Syntax	<code>earliestUnreleasedVersion(project)</code>
Supported fields	AffectedVersion, FixVersion, custom fields of type Version
Supported operators	<code>IN , NOT IN</code>
Unsupported operators	<code>= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>

Examples	<ul style="list-style-type: none"> Find issues whose FixVersion is the earliest unreleased version of the ABC project: <code>fixVersion = earliestUnreleasedVersion(ABC)</code> Find issues that relate to the earliest unreleased version of the ABC project: <code>affectedVersion = earliestUnreleasedVersion(ABC)</code> or <code>fixVersion = earliestUnreleasedVersion(ABC)</code>
-----------------	--

[^ top of page](#)

elapsed()

Only applicable if Jira Service Desk is installed and licensed.

Returns issues whose SLA clock is at a certain point relative to a cycle's start event.

Syntax	<code>elapsed()</code>
Supported fields	SLA
Supported operators	<code>= , != , > , >= , < , <=</code>
Unsupported operators	<code>~ , IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Examples	<ul style="list-style-type: none"> Find issues that have been waiting for a first response for more than 1 hour: <code>"Time to First Response" > elapsed("1h")</code>

[^ top of page](#)

endOfDay()

Perform searches based on the end of the current day. See also [endOfWeek](#), [endOfMonth](#), and [endOfYear](#); and [startOfDay](#), [startOfWeek](#), [startOfMonth](#), and [startOfYear](#).

Syntax	<code>endOfDay()</code> <code>endOfDay("inc")</code> <i>where <code>inc</code> is an optional increment of <code>(+/-)nn(y M w d h m)</code>. If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. <code>endOfDay("+1")</code> is the same as <code>endOfDay("+1d")</code>. If the plus/minus (+/-) sign is omitted, plus is assumed.</i>
Supported fields	Created, Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	<code>= , != , > , >= , < , <=</code> <code>WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED*</code> <i>* Only in predicate</i>
Unsupported operators	<code>~ , !~ IS , IS NOT , IN , NOT IN</code>
Examples	<ul style="list-style-type: none"> Find issues due by the end of today: <code>due < endOfDay()</code> Find issues due by the end of tomorrow: <code>due < endOfDay("+1")</code>

[^ top of page](#)

endOfMonth()

Perform searches based on the end of the current month. See also [endOfDay](#), [endOfWeek](#), and [endOfYear](#); and [startOfDay](#), [startOfWeek](#), [startOfMonth](#), and [startOfYear](#).

Syntax	<pre>endOfMonth() endOfMonth("inc")</pre> <p>where <i>inc</i> is an optional increment of $(+/-)nn(y M w d h m)$. If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. <code>endOfMonth("+1")</code> is the same as <code>endOfMonth("+1M")</code>. If the plus/minus (+/-) sign is omitted, plus is assumed.</p>
Supported fields	Created, Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	<pre>= , != , > , >= , < , <= WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED* * Only in predicate</pre>
Unsupported operators	<pre>~ , !~ IS , IS NOT , IN , NOT IN</pre>
Examples	<ul style="list-style-type: none"> Find issues due by the end of this month: <code>due < endOfMonth()</code> Find issues due by the end of next month: <code>due < endOfMonth("+1")</code> Find issues due by the 15th of next month: <code>due < endOfMonth("+15d")</code>

[^ top of page](#)

`endOfWeek()`

Perform searches based on the end of the current week. See also [endOfDay](#), [endOfMonth](#), and [endOfYear](#); and [startOfDay](#), [startOfWeek](#), [startOfMonth](#), and [startOfYear](#).

For the `endOfWeek()` function, the result depends upon your locale. For example, in Europe, the first day of the week is generally considered to be Monday, while in the USA, it is considered to be Sunday.

Syntax	<pre>endOfWeek() endOfWeek("inc")</pre> <p>where <i>inc</i> is an optional increment of $(+/-)nn(y M w d h m)$. If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. <code>endOfWeek("+1")</code> is the same as <code>endOfWeek("+1w")</code>. If the plus/minus (+/-) sign is omitted, plus is assumed.</p>
Supported fields	Created, Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	<pre>= , != , > , >= , < , <= WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED* * Only in predicate</pre>
Unsupported operators	<pre>~ , !~ IS , IS NOT , IN , NOT IN</pre>
Examples	<ul style="list-style-type: none"> Find issues due by the end of this week: <code>due < endOfWeek()</code> Find issues due by the end of next week: <code>due < endOfWeek("+1")</code>

[^ top of page](#)**endOfYear()**

Perform searches based on the end of the current year. See also [startOfDay](#), [startOfWeek](#), and [startOfMonth](#); and [endOfDay](#), [endOfWeek](#), [endOfMonth](#), and [endOfYear](#).

Syntax	<pre>endOfYear()</pre> <pre>endOfYear("inc")</pre> <p>where <i>inc</i> is an optional increment of (+/-)nn(y M w d h m). If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. <code>endOfYear("+1")</code> is the same as <code>endOfYear("+1y")</code>. If the plus/minus (+/-) sign is omitted, plus is assumed.</p>
Supported fields	Created, Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	<pre>= , != , > , >= , < , <=</pre> <pre>WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED*</pre> <p>* Only in predicate</p>
Unsupported operators	<pre>~ , !~ IS , IS NOT , IN , NOT IN</pre>
Examples	<ul style="list-style-type: none"> Find issues due by the end of this year: <code>due < endOfYear()</code> Find issues due by the end of March next year: <code>due < endOfYear("+3M")</code>

[^ top of page](#)**everbreached()**

Only applicable if Jira Service Desk is installed and licensed.

Returns issues that have missed one of their SLA goals.

Syntax	<pre>elapsed()</pre>
Supported fields	SLA
Supported operators	<pre>= , !=</pre>
Unsupported operators	<pre>~ , > , >= , < , <= , IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</pre>
Examples	<ul style="list-style-type: none"> Find issues have missed their goal for Time to First Response: <code>"Time to First Response" = everbreached()</code>

[^ top of page](#)**issueHistory()**

Find issues that you have recently viewed, i.e. issues that are in the 'Recent Issues' section of the 'Issues' drop-down menu.

Note:

- `issueHistory()` returns up to 50 issues, whereas the 'Recent Issues' drop-down returns only 5.
- if you are not logged in to Jira, only issues from your current browser session will be included.

Syntax	<code>issueHistory()</code>
Supported fields	Issue
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues which I have recently viewed, that are assigned to me: <code>issue in issueHistory() AND assignee = currentUser()</code>

[^ top of page](#)

issuesWithRemoteLinksByGlobalId()

Perform searches based on issues that are associated with remote links that have any of the specified global ids.

Note:

- This function accepts 1 to 100 globalIds. Specifying 0 or more than 100 globalIds will result in errors.

Syntax	<code>issuesWithRemoteLinksByGlobalId()</code>
Supported fields	Issue
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues that are linked to remote links that have globalId "abc": <code>issue in issuesWithRemoteLinksByGlobalId(abc)</code> Find issues that are linked to remote links that have either globalId "abc" or "def": <code>issue in issuesWithRemoteLinksByGlobalId(abc, def)</code>

[^ top of page](#)

lastLogin()

Perform searches based on the time at which the current user's previous session began. See also [currentLogin](#).

Syntax	<code>lastLogin()</code>
Supported fields	Created. Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	= , != , > , >= , < , <= WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED* <i>* Only in predicate</i>
Unsupported operators	~ , !~ IS , IS NOT , IN , NOT IN
Examples	<ul style="list-style-type: none"> Find issues that have been created during my last session: <code>created > lastLogin()</code>

[^ top of page](#)

latestReleasedVersion()

Perform searches based on the latest released version (i.e. the most recent version that has been released) of a specified project. See also [releasedVersions\(\)](#). Note, the "latest" is determined by the ordering assigned to the versions, not by actual Version Due Dates.

Syntax	<code>latestReleasedVersion(project)</code>
Supported fields	AffectedVersion, FixVersion, custom fields of type Version
Supported operators	<code>= , !=</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <=</code> <code>IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Examples	<ul style="list-style-type: none"> Find issues whose FixVersion is the latest released version of the ABC project: <code>fixVersion = latestReleasedVersion(ABC)</code> Find issues that relate to the latest released version of the ABC project: <code>affectedVersion = latestReleasedVersion(ABC)</code> or <code>fixVersion = latestReleasedVersion(ABC)</code>

[^ top of page](#)

linkedIssues()

Perform searches based on issues that are linked to a specified issue. You can optionally restrict the search to links of a particular type. Note that LinkType is case-sensitive.

Syntax	<code>linkedIssues(issueKey)</code> <code>linkedIssues(issueKey,linkType)</code>
Supported fields	Issue
Supported operators	<code>IN , NOT IN</code>
Unsupported operators	<code>= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Examples	<ul style="list-style-type: none"> Find issues that are linked to a particular issue: <code>issue in linkedIssues(ABC-123)</code> Find issues that are linked to a particular issue via a particular type of link: <code>issue in linkedIssues(ABC-123,"is duplicated by")</code>

[^ top of page](#)

membersOf()

Perform searches based on the members of a particular group.

Syntax	<code>membersOf(Group)</code>
Supported fields	Assignee, Reporter, Voter, Watcher, custom fields of type User

Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> • Find issues where the Assignee is a member of the group "Jira-administrators": assignee in membersOf("Jira-administrators") • Search through multiple groups and a specific user: reporter in membersOf("Jira-administrators") or reporter in membersOf("Jira-core-users") or reporter=jsmith • Search for a particular group, but exclude a particular member or members: assignee in membersOf(QA) and assignee not in ("John Smith","Jill Jones") • Exclude members of a particular group: assignee not in membersOf(QA)

[^ top of page](#)

myApproval()

Only applicable if Jira Service Desk is installed and licensed.

Search for issues that require approval or have required approval by the current user.

Syntax	<code>myApproval()</code>
Supported fields	Custom fields of type Approval
Supported operators	=
Unsupported operators	~ , != , !~ , > , >= , < , <= IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> • Find all issues that require or have required my approval <code>approval = myApproval()</code>

[^ top of page](#)

myPending()

Only applicable if Jira Service Desk is installed and licensed.

Search for issues that require approval by the current user.

Syntax	<code>approved()</code>
Supported fields	Custom fields of type Approval
Supported operators	=
Unsupported operators	~ , != , !~ , > , >= , < , <= IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> • Find all issues that require my approval <code>approval = myPending()</code>

[^ top of page](#)

now()

Perform searches based on the current time.

Syntax	<code>now()</code>
Supported fields	Created, Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	<code>= , != , > , >= , < , <=</code> <code>WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED*</code> <i>* Only in predicate</i>
Unsupported operators	<code>~ , !~ IS , IS NOT , IN , NOT IN</code>
Examples	<ul style="list-style-type: none"> Find issues that are overdue: <code>duedate < now() and status not in (closed, resolved)</code>

[^ top of page](#)

openSprints()

Search for issues that are assigned to a Sprint that has not yet been completed. Note, it is possible for an issue to belong to both a completed Sprint(s) and an incomplete Sprint(s). See also [closedSprints\(\)](#).

Syntax	<code>openSprints()</code>
Supported fields	Sprint
Supported operators	<code>IN , NOT IN</code>
Unsupported operators	<code>= , != , ~ , !~ , > , >= , < , <=</code> <code>IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Examples	<ul style="list-style-type: none"> Find all issues that are assigned to a sprint that has not yet been completed: <code>sprint in openSprints()</code>

[^ top of page](#)

paused()

Only applicable if Jira Service Desk is installed and licensed.

Returns issues that have an SLA that is paused due to a condition.

To find issues that are paused because they are outside calendar hours, use [withincalendarhours\(\)](#).

Syntax	<code>paused()</code>
Supported fields	SLA
Supported operators	<code>= , !=</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <= , IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>

Examples	<ul style="list-style-type: none"> Find issues where Time to First Response is paused: "Time to First Response" = paused()
-----------------	---

[^ top of page](#)

pending()

Only applicable if Jira Service Desk is installed and licensed.

Search for issues that require approval.

Syntax	<code>pending()</code>
Supported fields	Custom fields of type Approval
Supported operators	=
Unsupported operators	~ , != , !~ , > , >= , < , <= IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find all issues that require approval: <code>approval = pending()</code>

[^ top of page](#)

pendingBy()

Only applicable if Jira Service Desk is installed and licensed.

Search for issues that require approval by the listed user/s. This uses an OR operator, and you must specify the username/s.

Syntax	<code>pendingBy(user1,user2)</code>
Supported fields	Custom fields of type Approval
Supported operators	=
Unsupported operators	~ , != , !~ , > , >= , < , <= IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues that require approval by John Smith: <code>approval = pendingBy(jsmith)</code> Find issues that require by John Smith or Sarah Khan: <code>approval = pendingBy(jsmith,skhan)</code>

[^ top of page](#)

projectsLeadByUser()

Find issues in projects that are led by a specific user. You can optionally specify a user, or if the user is omitted, the current user will be used. Note that if you are not logged in to Jira, a user must be specified.

Syntax	<code>projectsLeadByUser()</code> <code>projectsLeadByUser(username)</code>
---------------	--

Supported fields	Project
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT, WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find open issues in projects that are led by you: project in projectsLeadByUser() AND status = Open Find open issues in projects that are led by Bill: project in projectsLeadByUser(bill) AND status = Open

[^ top of page](#)

projectsWhereUserHasPermission()

Find issues in projects where you have a specific permission. Note, this function operates at the project level. This means that if a permission (e.g. "Edit Issues") is granted to the reporter of issues in a project, then you may see some issues returned where you are not the reporter, and therefore don't have the permission specified. Also note, this function is only available if you are logged in to Jira.

Syntax	projectsWhereUserHasPermission(permission) For the <code>permission</code> parameter, you can specify any of the permissions described on .
Supported fields	Project
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT, WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find open issues in projects where you have the "Resolve Issues" permission: project in projectsWhereUserHasPermission("Resolve Issues") AND status = Open

[^ top of page](#)

projectsWhereUserHasRole()

Find issues in projects where you have a specific role. Note, this function is only available if you are logged in to Jira.

Syntax	projectsWhereUserHasRole(rolename)
Supported fields	Project
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT, WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find open issues in projects where you have the "Developers" role: project in projectsWhereUserHasRole("Developers") AND status = Open

[^ top of page](#)

releasedVersions()

Perform searches based on the released versions (i.e. versions that your Jira administrator has released) of a specified project. You can also search on the released versions of all projects, by omitting the *project* parameter. See also [latestReleasedVersion\(\)](#).

Syntax	releasedVersions() releasedVersions(project)
Supported fields	AffectedVersion, FixVersion, custom fields of type Version
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues whose FixVersion is a released version of the ABC project: fixVersion in releasedVersions(ABC) Find issues that relate to released versions of the ABC project: (affectedVersion in releasedVersions(ABC)) or (fixVersion in releasedVersions(ABC))

[^ top of page](#)

remaining()

Only applicable if Jira Service Desk is installed and licensed.

Returns issues whose SLA clock is at a certain point relative to the goal.

Syntax	remaining()
Supported fields	SLA
Supported operators	= , != , > , >= , < , <=
Unsupported operators	~ , IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues that will breach Time to Resolution in the next two hours: "Time to Resolution" < remaining("2h")

[^ top of page](#)

running()

Only applicable if Jira Service Desk is installed and licensed.

Returns issues that have an SLA that is running, regardless of the calendar.

To find issues that are running based on calendar hours, use [withincalendarhours\(\)](#).

Syntax	running()
Supported fields	SLA
Supported operators	= , !=

Unsupported operators	<code>~ , !~ , > , >= , < , <= , IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Examples	<ul style="list-style-type: none"> Find issues where Time to First Response is running: "Time to First Response" = running()

[^ top of page](#)

standardIssueTypes()

Perform searches based on "standard" Issue Types, that is, search for issues that are not sub-tasks. See also [subtaskIssueTypes\(\)](#).

Syntax	<code>standardIssueTypes()</code>
Supported fields	Type
Supported operators	<code>IN , NOT IN</code>
Unsupported operators	<code>= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</code>
Examples	<ul style="list-style-type: none"> Find issues that are not subtasks (i.e. issues whose Issue Type is a standard issue type, not a subtask issue type): <code>issuetype in standardIssueTypes()</code>

[^ top of page](#)

startOfDay()

Perform searches based on the start of the current day. See also [startOfWeek](#), [startOfMonth](#), and [startOfYear](#); and [endOfDay](#), [endOfWeek](#), [endOfMonth](#), and [endOfYear](#).

Syntax	<code>startOfDay()</code> <code>startOfDay("inc")</code> <i>where inc is an optional increment of (+/-)nn(y M w d h m). If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. startOfDay("+1") is the same as startOfDay("+1d"). If the plus/minus (+/-) sign is omitted, plus is assumed.</i>
Supported fields	Created, Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	<code>= , != , > , >= , < , <=</code> <code>WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED*</code> <i>* Only in predicate</i>
Unsupported operators	<code>~ , !~ IS , IS NOT , IN , NOT IN</code>
Examples	<ul style="list-style-type: none"> Find new issues created since the start of today: <code>created > startOfDay()</code> Find new issues created since the start of yesterday: <code>created > startOfDay("-1")</code> Find new issues created in the last three days: <code>created > startOfDay("-3d")</code>

[^ top of page](#)

startOfMonth()

Perform searches based on the start of the current month. See also [startOfDay](#), [startOfWeek](#), and [startOfYear](#); and [endOfDay](#), [endOfWeek](#), [endOfMonth](#), and [endOfYear](#).

Syntax	<pre>startOfMonth() startOfMonth("inc")</pre> <p>where <i>inc</i> is an optional increment of (+/-)nn(y M w d h m). If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. <code>startOfMonth("+1")</code> is the same as <code>startOfMonth("+1M")</code>. If the plus/minus (+/-) sign is omitted, plus is assumed.</p>
Supported fields	Created, Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	<p>= , != , > , >= , < , <=</p> <p>WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED*</p> <p>* Only in predicate</p>
Unsupported operators	~ , !~ IS , IS NOT , IN , NOT IN
Examples	<ul style="list-style-type: none"> Find new issues created since the start of this month: <code>created > startOfMonth()</code> Find new issues created since the start of last month: <code>created > startOfMonth("-1")</code> Find new issues created since the 15th of this month: <code>created > startOfMonth("+14d")</code>

[^ top of page](#)

startOfWeek()

Perform searches based on the start of the current week. See also [startOfDay](#), [startOfMonth](#), and [startOfYear](#); and [endOfDay](#), [endOfWeek](#), [endOfMonth](#), and [endOfYear](#). For the `startOfWeek()` function, the result depends upon your locale. For example, in Europe, the first day of the week is generally considered to be Monday, while in the USA, it is considered to be Sunday.

Syntax	<pre>startOfWeek() startOfWeek("inc")</pre> <p>where <i>inc</i> is an optional increment of (+/-)nn(y M w d h m). If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. <code>startOfWeek("+1")</code> is the same as <code>startOfWeek("+1w")</code>. If the plus/minus (+/-) sign is omitted, plus is assumed.</p>
Supported fields	Created, Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	<p>= , != , > , >= , < , <=</p> <p>WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED*</p> <p>* Only in predicate</p>
Unsupported operators	~ , !~ IS , IS NOT , IN , NOT IN
Examples	<ul style="list-style-type: none"> Find new issues since the start of this week: <code>created > startOfWeek()</code> Find new issues since the start of last week: <code>created > startOfWeek("-1")</code>

[^ top of page](#)**startOfYear()**

Perform searches based on the start of the current year. See also [startOfDay](#), [startOfWeek](#) and [startOfMonth](#); and [endOfDay](#), [endOfWeek](#), [endOfMonth](#) and [endOfYear](#).

Syntax	<pre>startOfYear()</pre> <pre>startOfYear("inc")</pre> <p>where <i>inc</i> is an optional increment of $(+/-)nn(y M w d h m)$. If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. <code>endOfYear("+1")</code> is the same as <code>endOfYear("+1y")</code>. If the plus/minus (+/-) sign is omitted, plus is assumed.</p>
Supported fields	Created, Due, Resolved, Updated, custom fields of type Date/Time
Supported operators	<p>= , != , > , >= , < , <=</p> <p>WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED*</p> <p>* Only in predicate</p>
Unsupported operators	~ , !~ IS , IS NOT , IN , NOT IN
Examples	<ul style="list-style-type: none"> Find new issues since the start of this year: <code>created > startOfYear()</code> Find new issues since the start of last year: <code>created > startOfYear("-1")</code>

[^ top of page](#)**subtaskIssueTypes()**

Perform searches based on issues that are sub-tasks. See also [standardIssueTypes\(\)](#).

Syntax	<code>subtaskIssueTypes()</code>
Supported fields	Type
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues that are subtasks (i.e. issues whose Issue Type is a subtask issue type): <code>issuetype in subtaskIssueTypes()</code>

[^ top of page](#)**unreleasedVersions()**

Perform searches based on the unreleased versions (i.e. versions that your Jira administrator has not yet released) of a specified project. You can also search on the unreleased versions of all projects, by omitting the *project* parameter. See also [earliestUnreleasedVersion\(\)](#).

Syntax	<pre>unreleasedVersions() unreleasedVersions(project)</pre>
Supported fields	AffectedVersion, FixVersion, custom fields of type Version
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues whose FixVersion is an unreleased version of the ABC project: <code>fixVersion in unreleasedVersions(ABC)</code> Find issues that relate to unreleased versions of the ABC project: <code>affectedVersion in unreleasedVersions(ABC)</code>

[^ top of page](#)

updatedBy()

Search for issues that were updated by a specific user, optionally within the specified time range. An update in this case includes creating an issue, updating any of the issue's fields, creating or deleting a comment, or editing a comment (only the last edit).

For the time range, use one of the following formats:

"YYYY/MM/dd"

"YYYY-MM-dd"

Or use "w" (weeks), or "d" (days) to specify a date relative to the current time. Unlike some other functions, `updatedBy` doesn't support values smaller than a day, and will always round them up to 1 day.

Syntax	<pre>updatedBy(user) updatedBy(user, dateFrom) updatedBy(user, dateFrom, dateTo)</pre>
Supported fields	Issuekey, and its aliases (id, issue, key)
Supported operators	IN , NOT IN
Unsupported operators	=, ~ , != , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues that were updated by John Smith: <code>issuekey IN updatedBy(jsmith)</code> Find issues that were updated by John Smith within the last 8 days: <code>issuekey IN updatedBy(jsmith, "-8d")</code> Find issues updated between June and September 2018: <code>issuekey IN updatedBy(jsmith, "2018/06/01", "2018/08/31")</code> If you try to find issues updated in the last hour, like in the following example, the time will be rounded up to 1 day, as smaller values aren't supported: <code>issuekey IN updatedBy(jsmith, "-1h")</code>

[^ top of page](#)

votedIssues()

Perform searches based on issues for which you have voted. Also, see the Voter field. Note, this function can only be used by logged-in users.

Syntax	<code>votedIssues()</code>
Supported fields	Issue
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues that you have voted for: <code>issue in votedIssues()</code>

[^ top of page](#)

watchedIssues()

Perform searches based on issues that you are watching. Also, see the Watcher field. Note that this function can only be used by logged-in users.

Syntax	<code>watchedIssues()</code>
Supported fields	Issue
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	<ul style="list-style-type: none"> Find issues that you are watching: <code>issue in watchedIssues()</code>

[^ top of page](#)

withinCalendarHours()

Only applicable if Jira Service Desk is installed and licensed.

Returns issues that have an SLA that is running according to the SLA calendar.

For example, say your project has two SLAs that count Time to First Response. Some issues with this SLA use a 9am-1pm calendar, and others use a 9am-5pm calendar. If an agent starts work at 3pm, they probably want to work on issues from the 9am-5pm agreement first. They can use `withincalendarhours()` to find all the issues where Time to First Response is running at 3pm.

Syntax	<code>withinCalendarHours()</code>
Supported fields	SLA
Supported operators	= , !=
Unsupported operators	~ , !~ , > , >= , < , <= , IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED

Examples

- Find issues where Time to First Response is within calendar hours:
"Time to First Response" = withinCalendarHours()

[^ top of page](#)

Search syntax for text fields

This page provides information on the syntax for searching text fields, which can be done in the quick search, basic search, and advanced search.

Text searches can be done in the advanced search when the **CONTAINS (~) operator** is used, e.g. `summary~"windows"`. It can also be done in quick search and basic search when searching on supported fields.

Acknowledgments: Jira uses Apache Lucene for text indexing, which provides a rich query language. Much of the information on this page is derived from the [Query Parser Syntax](#) page of the Lucene documentation.

On this page:

- [Query terms](#)
- [Term modifiers](#)
- [Boosting a term: ^](#)
- [Boolean operators](#)
- [Grouping](#)
- [Escaping special characters: \ or \\](#)
- [Reserved words](#)
- [Word stemming](#)
- [Limitations](#)
- [Next steps](#)

Query terms

A query is broken up into **terms** and **operators**. There are two types of terms: **Single Terms** and **Phrases**.

A **Single Term** is a single word, such as "test" or "hello".

A **Phrase** is a group of words surrounded by double quotes, such as "hello dolly".

Multiple terms can be combined together with Boolean operators to form a more complex query (see below). If you combine multiple terms without specifying any Boolean operators, they will be joined using AND operators.

Note: All query terms in Jira are not case sensitive.

Term modifiers

Jira supports modifying query terms to provide a wide range of searching options.

[Exact searches \(phrases\)](#) | [Wildcard searches: ? and *](#) | [Fuzzy searches: ~](#) | [Proximity searches](#)

Exact searches (phrases)

To find exact matches for **phrases**, for example *Jira Software*, you need to enclose the whole phrase in quote-marks (""). Otherwise, the search will return all issues that contain both words in no particular order - this would include *Jira Software*, but also *Jira is the best software!*.

If you're using advanced search, you need to additionally escape each of the quote-marks with a backslash (\). For details, see the examples below or find your field in [Advanced search - field reference](#).

Examples

- **Basic search:** Find all issues that contain the phrase *Jira Software*:

```
Just type "Jira Software" into the search field.
```

- **Advanced search:** Find all issues that contain the words *Jira* and *Software*, in no particular order.

```
text ~ "Jira Software"
```

- **Advanced search:** Find all issues that contain the phrase *Jira Software*.

```
text ~ "\"Jira Software\""
```

- **Advanced search:** Find all issues that contain the URL `https://atlassian.com`:

```
text ~ "\"https://atlassian.com\""
```

As you can see in the two preceding examples, the query contains two pairs of quote-marks. The external ones are needed to meet the JQL rules and aren't related to your search query. The same pair of quote-marks would be automatically added by Jira in the basic search after running your search.

Using special characters to create phrases

In previous versions of Jira, you could use some special characters to combine **terms** into **phrases**, for example *Jira+Software* or *Jira/Software*. This is no longer the case, as the mechanism used for searching has changed and the special characters surrounding **terms** are ignored.

Wildcard searches: ? and *

Jira supports single and multiple character wildcard searches.

To perform a single character wildcard search, use the "?" symbol.

To perform a multiple character wildcard search, use the "*" symbol.

Wildcard characters need to be enclosed in quote-marks, as they are reserved characters in advanced search. Use quotations, e.g. `summary ~ "cha?k and che*"`

The single character wildcard search looks for terms that match that with the single character replaced. For example, to search for "text" or "test", you can use the search:

```
te?t
```

Multiple character wildcard searches looks for 0 or more characters. For example, to search for Windows, Win95, or WindowsNT, you can use the search:

```
win*
```

You can also use the wildcard searches in the middle of a term. For example, to search for Win95 or Windows95, you can use the search:

```
wi*95
```

You cannot use a * or ? symbol as the first character of a search. The feature request for this is [JRA-6218](#).

Fuzzy searches: ~

Jira supports fuzzy searches. To do a fuzzy search, use the tilde, "~", symbol at the end of a single word term. For example, to search for a term similar in spelling to "roam", use the fuzzy search:

```
roam~
```

This search will find terms like foam and roams.

Note: Terms found by the fuzzy search will automatically get a boost factor of 0.2.

Proximity searches

Jira supports finding words that are within a specific distance away. To do a proximity search, use the tilde, "~", symbol at the end of a phrase. For example, to search for "atlassian" and "Jira" within 10 words of each other in a document, use the search:

```
"atlassian Jira"~10
```

Boosting a term: ^

Jira provides the relevance level of matching documents based on the terms found. To boost a term, use the caret, "^", symbol with a boost factor (a number) at the end of the term you are searching. The higher the boost factor, the more relevant the term will be.

Boosting allows you to control the relevance of a document by boosting its term. For example, if you are searching for

```
atlassian Jira
```

and you want the term "atlassian" to be more relevant, boost it using the ^ symbol along with the boost factor next to the term. You would type:

```
atlassian^4 Jira
```

This will make documents with the term atlassian appear more relevant. You can also boost Phrase Terms, as in the example:

```
"atlassian Jira"^4 querying
```

By default, the boost factor is 1. Although, the boost factor must be positive, it can be less than 1 (i.e. 0.2).

Boolean operators

Boolean operators allow terms to be combined through logic operators. Jira supports AND, "+", OR, NOT and "-" as Boolean operators.

Boolean operators must be ALL CAPS.

OR | AND | Required term: + | NOT | Excluded term: -

OR

The OR operator is the default conjunction operator. This means that if there is no Boolean operator between two terms, the OR operator is used. The OR operator links two terms, and finds a matching document if either of the terms exist in a document. This is equivalent to a union using sets. The symbol `||` can be used in place of the word OR.

To search for documents that contain either "atlassian Jira" or just "confluence", use the query:

```
"atlassian Jira" || confluence
```

or

```
"atlassian Jira" OR confluence
```

AND

The AND operator matches documents where both terms exist anywhere in the text of a single document. This is equivalent to an intersection using sets. The symbol `&&` can be used in place of the word AND.

To search for documents that contain "atlassian Jira" and "issue tracking", use the query:

```
"atlassian Jira" AND "issue tracking"
```

Required term: +

The "+" or required operator requires that the term after the "+" symbol exists somewhere in the field of a single document.

To search for documents that must contain "Jira" and may contain "atlassian", use the query:

```
+Jira atlassian
```

NOT

The NOT operator excludes documents that contain the term after NOT. This is equivalent to a difference using sets. The symbol `!` can be used in place of the word NOT.

To search for documents that contain "atlassian Jira" but not "japan", use the query:

```
"atlassian Jira" NOT "japan"
```

Note: The NOT operator cannot be used with just one term. For example, the following search will return no results:

```
NOT "atlassian Jira"
```

Usage of the **NOT** operator over multiple fields may return results that include the specified excluded term. This is due to the fact that the search query is executed over each field in turn, and the result set for each field is combined to form the final result set. Hence, an issue that matches the search

query based on one field, but fails based on another field will be included in the search result set.

Excluded term: -

The "-" or prohibit operator excludes documents that contain the term after the "-" symbol.

To search for documents that contain "atlassian Jira" but not "japan", use the query:

```
"atlassian Jira" -japan
```

Grouping

Jira supports using parentheses to group clauses to form sub queries. This can be very useful if you want to control the boolean logic for a query.

To search for bugs and either atlassian or Jira, use the query:

```
bugs AND (atlassian OR Jira)
```

This eliminates any confusion and makes sure that bugs must exist, and either term atlassian or Jira may exist.

Do not use the grouping character '(' at the start of a search query, as this will result in an error. For example, "(atlassian OR Jira) AND bugs" will not work.

Escaping special characters: \ or \\

Jira supports the ability to search issues for special characters by escaping them in your query syntax. The current list of such characters is:

```
+ - & | ! ( ) { } [ ] ^ ~ * ? \ :
```

To escape these characters, type a backslash character '\' before the special character (or if using advanced searching, type two backslashes '\\\' before the special character).

For example, to search for (1+1) in either a basic or quick search, use the query:

```
\(1\+1\)
```

and to search for [example] in the summary of an advanced search (in Jira Query Language or JQL), use the query:

```
summary ~ "\\[example\\]"
```

Please note: If you are using advanced searching, see [Reserved characters](#) for more information about how these characters and others are escaped in Jira Query Language.

Reserved words

To keep the search index size and search performance optimal in Jira, the following English *reserved words* (also known as 'stop words') are ignored from the search index and hence, Jira's text search features:

"a", "and", "are", "as", "at", "be", "but", "by", "for", "if", "in", "into",

"is", "it", "no", "not", "of", "on", "or", "such", "that", "the", "their", "then", "there", "these", "they", "this", "to", "was", "will", "with"

Be aware that this can sometimes lead to unexpected results. For example, suppose one issue contains the text phrase "VSX will crash" and another issue contains the phrase "VSX will not crash". A text search for "VSX will crash" will return both of these issues. This is because the words `will` and `not` are part of the reserved words list.

i Your Jira administrator can make Jira index these reserved words (so that Jira will find issues based on the presence of these words) by changing the **Indexing Language** to **Other** (under **Administration > System > General Configuration**).

Word stemming

Since Jira cannot search for issues containing parts of words (see [below](#)), word 'stemming' allows you to retrieve issues from a search based on the 'root' (or 'stem') forms of words instead of requiring an exact match with specific forms of these words. The number of issues retrieved from a search based on a stemmed word is typically larger, since any other issues containing words that are stemmed back to the same root will also be retrieved in the search results.

For example, if you search for issues using the query term 'customize' on the Summary field, Jira stems this word to its root form 'custom', and will retrieve all issues whose Summary field also contains any word that can be stemmed back to 'custom'. Hence, the following query:

```
summary ~ "customize"
```

will retrieve issues whose Summary field contains the following words:

- customized
- customizing
- customs
- customer
- etc.

i Please Note:

- Your Jira administrator can disable word stemming (so that Jira will find issues based on exact matches with words) by changing the **Indexing Language** to **Other** (under **Administration > System > General Configuration**).
- Word stemming applies to *all* Jira fields (as well as text fields).
- When Jira indexes its fields, any words that are 'stemmed' are stored in Jira's search index in root form only.

Limitations

Please note that the following limitations apply to Jira's search:

Whole words only

Jira cannot search for issues containing parts of words but on whole words only. The exception to this are words which are [stemmed](#).

This limitation can also be overcome using [fuzzy searches](#).

Next steps

Read the following related topics:

- [Searching for issues](#)
- [Quick searching](#)
- [Basic searching](#)
- [Advanced searching](#)

Saving your search as a filter

Jira's powerful [issue search](#) functionality is enhanced by the ability to save searches, called *filters* in Jira, for later use. You can do the following with Jira filters:

- Share and email search results with your colleagues, as well as people outside of your organization
- Create lists of [favorite filters](#)
- Have search results [emailed to you](#) according to your preferred schedule
- View and export the search results in various formats (RSS, Excel, etc)
- Display the search results in a report format
- Display the search results in a [dashboard gadget](#)

On this page:

- [Saving a search as a filter](#)
- [Running a filter](#)
- [Managing your existing filters](#)
- [Managing other user's shared filters](#)
- [Next steps](#)

Screenshot: *Issue filter results in detail view (click to view full size image)*

The screenshot shows the Jira Service Desk interface. At the top, there's a header for 'Teams in Space' with options to 'Save' and 'Details'. Below this, there's a search bar with filters like 'Teams in Space', 'Epic, Story', 'To Do, in progress', and 'Assignee: All'. A search query 'Contains text' is entered, and a 'Search' button is visible. The main content area is split into two columns. The left column shows a list of issues with their IDs and titles. The right column shows the detailed view of the selected issue, 'UI should allow users to book on large or personal space craft'. This view includes fields for Type (Story), Status (IN PROGRESS), Priority (High), Resolution (Unresolved), and a description of the issue. The description states: 'Currently, LocalTransportControler makes an assumption that all the participants in the group are on the same itinerary. Many of our local travel providers limit reservations to 4 people. The UI should walk the user through booking a large travel vendor for the group and allow certain users to opt out for a personal space craft.'

Saving a search as a filter

1. Define and run your search.
2. Click the **Save as** link above the search results. The **Save Filter** dialog is displayed.
3. Enter a name for the new filter and click **Submit**. Your filter is created.

Your new filter will be added to your favorite filters and shared, according to the sharing preference in your user profile. If you haven't specified a preference, then the global default will be applied, which is 'Private' unless changed by your Jira administrator.

Running a filter

1. Choose **Issues > Search for issues**.
2. Choose any filter from the list on the left:
 - System filter — **My Open Issues, Reported by Me, Recently Viewed, All Issues**
 - Favorite filters (listed alphabetically)
 - **Find filters** lets you search for any filter that's been shared, which you can then subscribe to (adding it to your **Favorite Filters**).
3. After selecting a filter, the search results are displayed. The search criteria for the filter are also displayed and can be changed.

*Note, if you run the **Recently Viewed** system filter, this will switch you to the advanced search, as*

the basic search cannot represent the *ORDER BY* clause in this filter.

Managing your existing filters

Click **Issues > Manage filters** to manage your filters.

Manage Filters			
Favorite My Popular Search	My Filters ⓘ Filters are issue searches that have been saved for re-use. This page shows all filters that you own.		
	Name	Shared With	Subscriptions
	★ Current sprint	• Private filter	None - Subscribe ⚙️
	☆ Due this week (TCYO)	• Project: Core	None - Subscribe ⚙️
	★ Epics	• Private filter	None - Subscribe ⚙️
	☆ Filter for EXP board	• Project: Space Exploration	None - Subscribe ⚙️
	★ Teams in Space	• Private filter	None - Subscribe ⚙️
	★ High-priority issues	• Private filter • Project: JIRA	None - Subscribe ⚙️
	★ Ignite docs	• Private filter	None - Subscribe ⚙️

The **Manage Filters** page allows you to view and configure filters that you have created, as well as work with filters that other users have shared with you. See the following topics for more information:

- [Searching for a filter](#)
- [Updating a filter](#)
- [Deleting a filter](#)
- [Cloning a filter](#)
- [Adding a filter as a favorite](#)
- [Sharing a filter](#)
- [Defining a filter-specific column order](#)
- [Subscribing to a filter](#)

Searching for a filter

You can find and run any filters that you have created or that have been shared by other users.

1. Click the **Search** tab on the 'Manage Filters' page.
2. Enter your search criteria and click **Search** to run the search.
3. Your search results are displayed on the same page. Click the name of any issue filter to run it.

*Tip: If the filter has been added as a favorite by many users, you may also be able locate it on the **Popular** tab of the **Manage Filters** page.*

Updating a filter

You can update the name, description, sharing, favorite of any filters that you created, or have permission to edit. If you want to edit a filter for which you only have the *view* permission, either [clone](#) (aka copy) the shared filter, or ask your Jira administrator to [change the filter's ownership](#).

Update the filter's details:

1. Click the **My** tab on the 'Manage Filters' page.
2. Locate the filter you wish to update, click the **cog icon > Edit**.
3. The **Edit Current Filter** page displays, where you can update the filter details as required.
4. Click **Save** to save your changes.

Update the filter's search criteria:

1. Click the **My** tab on the 'Manage Filters' page.
2. Locate the filter you want to update and run it.
3. Update the search criteria as desired, and rerun the query to ensure the update is valid. You will see the word *Edited* displayed next to your filter name.
4. Click **Save** to overwrite the current filter with the updated search criteria. If you want discard your changes instead, click the arrow next to the save button, and select **Discard changes**.

Deleting a filter

1. Click the **My** tab on the 'Manage Filters' page.
2. Locate the filter you wish to delete, click the **cog icon > Delete**.

Cloning a filter

You can clone any filter – which is just a way of making a copy that you own – that was either created by you or shared with you.

1. Locate the filter you wish to clone and run it.
2. Update the search criteria as desired. Click the arrow next to the **Save** button, and select **Save > Save as** to create a new filter from the existing filter.

Adding a filter as a favorite

Filters that you've created or that have been shared by others can be added to your favorite filters. Favorite filters are listed in the menu under **Issues > Filters**, and in the left panel of the issue navigator.

1. Locate the filter you wish to add as a favorite.
2. Click the star icon next to the filter name to add it to your favorites.

Sharing a filter

Filters that you have created or have permission to edit can be shared with other users, user groups, projects, and project roles. They can also be shared globally. You can choose whether you want to share the filter with the permission to edit, or only to view. Any filter that is shared is visible to users who have the 'Jira Administrators' global permission. See [Managing other users' shared filters](#) below.

1. Click the **My** tab on the 'Manage Filters' page.
2. Locate the filter you wish to share, click the **cog icon > Edit**.
3. Update the **Add Viewers** and **Add Editors** fields by selecting the user, group, project, or project role that you want to share the filter with, and clicking **Add**. Note that you can only share filters with groups/roles of which you are a member.
 - [Why can't I see the filter's sharing configuration?](#)
You need the Create Shared Object global permission to configure sharing for a filter. Contact your Jira administrator to obtain this permission.
4. Click **Save** to save your changes.

*Tip: You can also share your filter by running it, then clicking **Details > Edit Permissions**.*

Defining a filter-specific column order

You can add a defined column order to a saved filter, which displays the filter results according to the saved column order. Otherwise, the results are displayed according to your personal column order (if you have set this) or the system default.

*Tip: To display your configured column order in a filter subscription, select 'HTML' for the 'Outgoing email format' in your **User Profile**. If you receive text emails from Jira, you won't be able to see your configured column order.*

To add a column layout to a saved filter:

1. Click the **My** tab on the 'Manage Filters' page.
2. Locate the filter you wish to update; click the filter's name to display the results. Be sure you are viewing the filter in the **List** view so that you see the columns.
3. Configure the column order as desired by clicking on the column name and dragging it to the new position. Your changes are saved and will be displayed the next time you view this filter.

To remove a filter's saved column layout:

1. Click the **My** tab on the 'Manage Filters' page.
2. Locate the filter you wish to update; click the filter's name to display the results. Be sure you are viewing the filter in the **List** view so that you see the columns.
3. Click the **Columns** option on the top right of the displayed columns, and select **Restore Defaults** in the displayed window.

Exporting column ordered issues

When the results of a saved filter are exported to Excel, the column order and choice of columns are those that were saved with the filter. Even if a user has configured a personal column order for the results on the screen, the **saved configuration** is used for the Excel export. To export using your own configuration, save a copy of the filter along with your configuration, and then export the results to Excel.

Subscribing to a filter

See [Working with search results](#).

Managing other user's shared filters

A **shared filter** is a filter whose creator has shared that filter with other users. Refer to [Sharing a filter](#) above for details. When a shared filter is created by a user, that user:

- Initially 'owns' the shared filter.
- Being the owner, can edit and modify the shared filter.

If you have the **Jira Administrators** global permission, you can manage shared filters that were created by other users. For instructions, see [Managing shared filters](#).

Next steps

Read the following related topics:

- [Searching for issues](#)
- [Basic searching](#)
- [Advanced searching](#)
- [Working with search results](#)

Working with search results

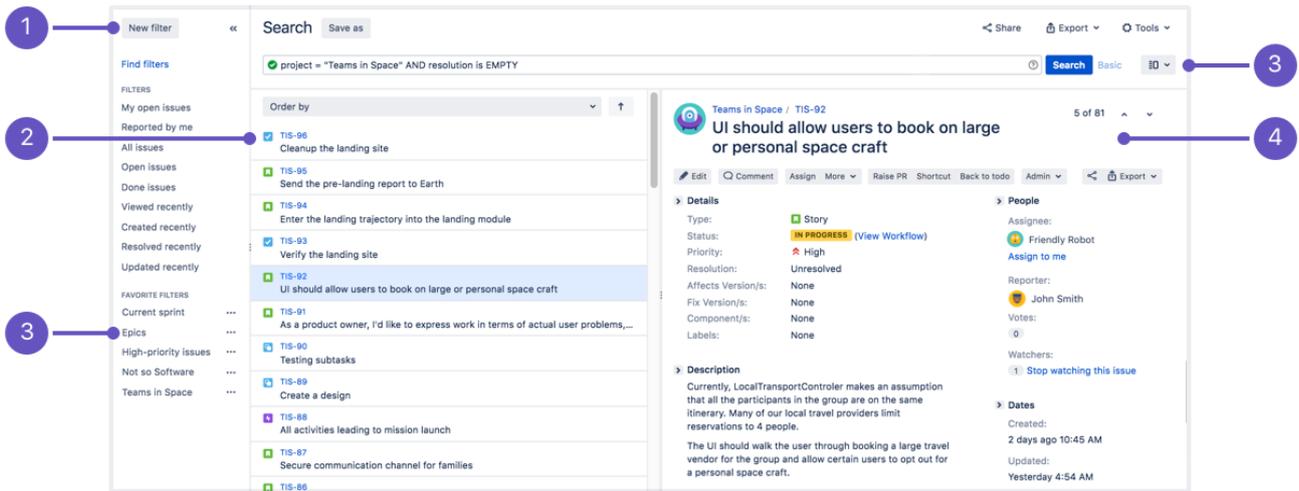
Once you have run a search, your search results will be displayed in the issue navigator. You may want to triage the entire list of issues or may be looking for just one. This page will show you what you can do with your search results, from changing what you see in the issue navigator to modifying the issues.

On this page:

- [Changing your view of the search results](#)
- [Working with individual issues](#)
- [Sharing your search results](#)
- [Displaying your search results in Confluence](#)
- [Displaying your search results as a chart](#)
- [Exporting your search results](#)
- [Printable views](#)
- [Subscribing to your search results](#)
- [Bulk modifying issues in your search results](#)
- [Next steps](#)

The following screenshot provides an overview of the key features of the issue navigator.

Screenshot: Issue navigator (Detail view)



1. **Filter panel:** Click << to collapse the filter panel so you can have more space in the detail view.
2. **Issue:** Select an issue from this panel to see the details in the detail view window.
3. **Filters:** Select a filter to see all the matching issues in the panel to the immediate right.
4. **Views:** Click to switch between the detail view and list view.
5. **Detail view:** Check out all the details about the selected issue in this detail view.

Changing your view of the search results

List view or Detail view	<p>Click the  dropdown to switch between List view and Detail view for your search results.</p> <ul style="list-style-type: none"> • List view: Shows your search results as a list of issues. This view is easiest to scan and is best when you only need to know a few details about each issue. • Detail view: Shows your search results as a list of issues, with the right panel showing the details of the currently selected issue. This view is best when you need more information about the individual issues, or you want to quickly edit issues as you go (via inline edit for certain fields).
Change the sort order	<p>Click the column name. If you click the same column name more than once, the sort order will switch between ascending and descending. Note:</p> <ul style="list-style-type: none"> • You cannot sort by the 'Images' column nor the sub-task aggregate columns (i.e. all columns beginning with "). • If you sort the search results for an advanced search, an 'ORDER BY' clause will be added/updated for your JQL query to reflect the order of issues in your search results.

Columns — show/hide and move	<p>You can create different column configurations for yourself and for specific filters. To switch between different column configurations, click Columns and select one of the following tabs:</p> <ul style="list-style-type: none"> • My Defaults: This is your default column configuration for search results. • Filter: This is enabled if you are viewing the search results for a filter. It will override your default column configuration. • System (shows if you are a Jira administrator): This is the column configuration that applies to all users. It will be overridden by a user's default column configuration and filter-specific column configurations. <p>You can also modify any of these configurations. Make sure you have switched the desired configuration, then do the following:</p> <ul style="list-style-type: none"> • Show/hide columns: Click Columns, choose the desired columns, then click Done. • Move a column: Click the column name and drag it to the desired position. <p>∨ Why can't I add a column to my column configuration?</p> <p>If you cannot find a column, please make sure that you haven't run in to any of the following restrictions:</p> <ul style="list-style-type: none"> • You can only see columns for issue fields that have not been hidden and that you have permissions to see. • It is possible to add any of the existing custom fields to the column list, as long as the fields are visible, and you have the right permissions. • Some custom fields, even if selected, do not appear in the Issue Navigator for all issues. For example, project-specific custom fields will be shown only if the filter has been restricted to that project only. Issue type custom fields will only appear if the filter has been restricted to that issue type.
--	--

Working with individual issues

You can action individual issues in your search results, directly from the issue navigator. Note that the list of issues will remain constant even if you change an issue, so that it doesn't meet the original search criteria. The advantage of this is that you have a constant set of search results that you can work from when triaging issues.

View an issue	<p>Click the key or summary of the issue.</p> <ul style="list-style-type: none"> • If you are in List view, you will be redirected to the issue (leaving the search results page). • If you are in Detail view, the issue details will display in the right panel.
Action an issue	<p>To action an issue (e.g. edit it, transition it, log work on it, etc):</p> <ul style="list-style-type: none"> • If you are in List view, click the cog icon and select from the options. • If you are in Detail view, select the issue and action it the issue via the details panel. <p>You can also select an issue and action it via keyboard shortcuts in either views. <i>Tip: use the 'j' and 'k' keys to select the previous/next issue in the issue navigator.</i></p>

Sharing your search results

Click **Share** in the issue navigator to email a link to a search result or shared filter.

- Recipients will receive an email with a link to the search result and the content of the **Note** field (if specified). The subject of the email will state that you (using your username) shared the issue.
- If you share the results of a filter, rather than an ad-hoc search, recipients will receive a link to the filter. Note, if the recipient does not have permission to view the filter, they will receive a link to the search results instead.

Displaying your search results in Confluence

If your Jira applications are connected to Confluence, you can display your search results on a Confluence page using the Jira issues macro. For instructions, see [Jira issues macro](#).

Displaying your search results as a chart

Click **Export > Dashboard charts**. Choose the desired chart from the dialog that is displayed, then click **Save to Dashboard**.

The chart will be added to your dashboard. For more information on what each chart shows, see [Reporting](#).

Exporting your search results

CSV	<p>Click Export > CSV (All fields) or Export > CSV (Current fields), and choose a delimiter to separate the values.</p> <p>The CSV file will contain a header row with every applicable issue field, comment, and attachment in your search result.</p> <ul style="list-style-type: none"> • CSV (All fields): this will include every issue field, comment and attachment. The header row may contain multiple values of "Comment" and/or "Attachment" if your issues have multiple comments and/or attachments. • CSV (Current fields): this will include only issue fields that are currently displayed. <p>Note, large exports (e.g. many hundreds of issues) are not recommended. You can change the number of issues that are exported, by changing the value of the tempMax parameter in the URL.</p> <p>If you're making a lot of exports, your Jira admin can disable the extra dialog that asks about delimiters. In this case, comma will be used as the default delimiter. Learn more</p>
HTML	<p>Click Export > HTML (All fields) or Export > HTML (Current fields).</p> <p>The HTML file will contain a header row with a value for every applicable issue field in your search result.</p> <ul style="list-style-type: none"> • HTML (All fields): this will create an HTML file for every issue field (excluding comments). This will only show the custom fields that are available for all of the issues in the search results. For example, if a field is only available for one project and multiple projects are in the search results then that field will not appear in the HTML file. The same goes for fields that are only available for certain issue types. • HTML (Current fields): this will create an HTML file for the issue fields that are currently displayed. <p>Note, large exports (e.g. many hundreds of issues) are not recommended. You can change the number of issues that are exported, by changing the value of the tempMax parameter in the URL.</p>
XML	<p>Click Export > XML.</p> <p>You can use the URL of the XML view in a Confluence Jira issues macro. However, you can also use the JQL or the URL of the issue search, which are easier to get.</p> <p>To restrict which issue fields are returned in the XML export, specify the <code>field</code> parameter in your URL. For example, to include only the Issue key and Summary, add <code>&field=key&field=summary</code> to the URL. If the <code>field</code> parameter is not specified, the XML output will include <i>all</i> the issue fields. Otherwise, if one or more <code>field</code> parameters are specified, the XML output will contain only the Issue key plus your chosen field(s). See the "List of fields for field parameter" below.</p>
Word	<p>Click Export > Word.</p> <p>The export will include the Description, Comments, and all other issue data, not just the issue fields that are currently configured in your Issue Navigator. Note, large exports (e.g. hundreds of issues) are not recommended.</p>

List of fields for field parameter (XML exports):

▼ Show me...

Value	Sample XML output
title	<pre><title>[TEST-4] This is a test</title></pre>
link	<pre><link>https://extranet.atlassian.com:443/J:</pre>
project (or pid)	<pre><project id="10330" key="TST">Test</project></pre>
description	<pre><description>This is a detailed description</description></pre>
environment	<pre><environment>Sydney network</environment></pre>
key	<pre><key id="22574">TEST-4</key></pre>
summary	<pre><summary>This is a test</summary></pre>
type (or issuetype)	<pre><type id="3" iconUrl="https://extranet.atla:</pre>
parent	<pre><parent id="22620">TEST-5</parent></pre>
priority	<pre><priority id="4" iconUrl="https://extranet.atlassian.com:44:</pre>
status	<pre><status id="5" iconUrl="https://extranet.atlassian.com:44:</pre>

resolution	<pre><resolution id="1">Fixed</resolution></pre>
labels	<pre><labels> <label>focus</label> </labels></pre>
assignee	<pre><assignee username="jsmith">John Smith</as:</pre>
reporter	<pre><assignee username="jsmith">John Smith</as:</pre>
security	<pre><security id="10021">Private</security></pre>
created	<pre><created>Mon, 1 Sep 2008 17:30:03 -0500 (CI</pre>
updated	<pre><updated>Mon, 1 Sep 2008 17:30:03 -0500 (CI</pre>
resolved (or resolutiondate)	<pre><resolved>Mon, 1 Sep 2008 17:30:03 -0500 ((</pre>
due (or duedate)	<pre><due>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>.</pre>
version (or versions)	<pre><version>2.4.7</version></pre>
fixfor (or fixVersions)	<pre><fixVersion>2.6</fixVersion></pre>
component (or components)	<pre><component>Documentation</component></pre>

votes	<pre><votes>1</votes></pre>
comments (or comment)	<pre><comments> <comment id="39270" author="jsmith" cre familiar</comment> <comment id="39273" author="jbrown" cre too</comment> </comments></pre>
attachments (or attachment)	<pre><attachments> <attachment id="30318" name="Issue Navig created="Mon, 9 Feb 2009 13:32:58 -0600 (C <attachment id="30323" name="Windows XP created="Tue, 10 Feb 2009 00:30:11 -0600 (C </attachments></pre>
timeoriginalestimate	<pre><timeoriginalestimate seconds="600">10 min</pre>
timeestimate	<pre><timeestimate seconds="300">5 minutes</time</pre>
timespent	<pre><timespent seconds="300">5 minutes</timespe</pre>
aggregatetimeoriginalestimate	<pre><aggregatetimeoriginalestimate seconds="360</pre>
aggregatetimeestimate	<pre><aggregatetimerremainingestimate seconds="18</pre>
aggregatetimespent	<pre><aggregatetimespent seconds="18000">5 hour;</pre>

timetracking	<pre> <timeoriginalestimate seconds="600">10 min <timeestimate seconds="300">5 minutes</time <timespent seconds="300">5 minutes</timesp <aggregatetimeoriginalestimate seconds="360 <aggregatetimerremainingestimate seconds="18000" <aggregatetimespent seconds="18000">5 hour: </pre>
issuelinks	<pre> <issuelinks> <issuelinktype id="10020"> <name>Duplicate</name> <inwardlinks description="is duplic <issuelink> <issuekey id="22477">INTSY: </issuelink> </inwardlinks> </issuelinktype> </issuelinks> </pre>
subtasks (or subtask)	<pre> <subtasks> <subtask id="22623">TEST-8</subtask> </subtasks> </pre>
customfield_xxxxx	<pre> <customfields> <customfield id="customfield_10112" key="com.atlassian.Jira.plugin.system.custo <customfieldname>Department</custo <customfieldvalues> <customfieldvalue>Adminstratio </customfieldvalues> </customfield> </customfields> </pre>

allcustom	<pre> <customfields> <customfield id="customfield_10112" key="com.atlassian.Jira.plugin.system.custo <customfieldname>Department</custor <customfieldvalues> <customfieldvalue>Adminstratio </customfieldvalues> </customfield> <customfield id="customfield_10111" key="com.atlassian.Jira.plugin.system.custo <customfieldname>Expenditure Type<, <customfieldvalues> <customfieldvalue>Operating</c </customfieldvalues> </customfield> </customfields> </pre>
------------------	--

Printable views

Printable	<p>Click Export > Printable.</p> <p>Creates a view of your search results in your browser that can be printed 'Landscape'. This view only contains issue Type, Key, Summary, Assignee, Reporter, Priority, Status, Resolution, Created date, Updated date, and Due date.</p>
Full content	<p>Click Export > Full content.</p> <p>Creates a view of your search results in your browser that can be printed. This view contains all issue fields, comments, and a list of attachments (there is no preview) for every issue returned by your search.</p>

Subscribing to your search results

A subscription provides you with a periodic notification for all issues returned by the search. If you want to be notified when a particular issue changes, you should watch the issue instead.

Email	<p>Your search must be saved as a filter, if you want to create an email subscription for it. You can create a subscription of any frequency for yourself and/or other users. Note, only the first 200 results of a filter are sent.</p> <ol style="list-style-type: none"> 1. Run the filter that you want to subscribe to, then click Details (next to filter name). 2. Fill in the 'Filter Subscription form' and click Subscribe. <p>More information:</p> <ul style="list-style-type: none"> • If you choose 'Advanced' for your Schedule, see this page for help on constructing Cron expressions. • You can choose to specify a group as a recipient, however you can only select a group that you are a member of: <ul style="list-style-type: none"> • You must have the 'Manage Group Filter Subscriptions' global permission. • Be aware that the emailed filter results will be specific to each recipient. For example, if the filter uses the <code>currentUser()</code> function, the search results will be evaluated with the recipient as the current user. This does not apply to distribution lists (group email aliases). • Be careful about sharing a subscription with a group with many members, as it can take a long time to generate the emails to be sent, since the search needs to be executed for each user (as per the previous point).
RSS	<p>Click Export > RSS (Issues) or Export > RSS (Comments). The URL of the page that shows can be used in your feed reader.</p> <p>Tips:</p> <ul style="list-style-type: none"> • You can change the number of issues that are returned, by changing the value of the temp Max parameter in the URL. • If you only want to receive current comments in an RSS feed, use the Date Updated field when doing a search. For example, to only receive comments created in the last week, add the Date Update field and set it to updated within the last 1 week. • You may need to log into your Jira applications to view restricted data in your feed. If so, you can add os_authType=basic to the feed URL (e.g. <code>http://mycompany.com/anypage?os_authType=basic</code>) to show a login dialog when viewing the feed.

Bulk modifying issues in your search results

Bulk operations let you action multiple issues at once. These actions include transitioning issues, deleting issues, moving issues, and watching/unwatching issues.

Click **Tools > Bulk Change: all <N> issue(s)** and follow the 'Bulk Operation' wizard.

For more information, see [Editing multiple issues at the same time](#).

Next steps

Read the following related topics:

- [Searching for issues](#)
- [Constructing cron expressions for a filter subscription](#)

Constructing cron expressions for a filter subscription

This page describes how to construct a cron expression. Cron expressions can be used when creating a subscription to a filter, as described in [Working with search results](#).

A cron expression gives you more control over the frequency, compared to the default schedules. For example, you could define a cron expression to notify you at 8:15 am on the second Friday of every month.

Constructing a cron expression

A cron expression is a string of fields separated by spaces. The following table displays the fields of a cron expression, *in the order that they must be specified (from left to right)*:

	Second	Minute	Hour	Day-of-month	Month	Day-of-week	Year (optional)
Allowed values	0-59	0-59	0-23	1-31	1-12 or JAN-DEC	1-7 or SUN-SAT	1970-2099
Allowed special characters	, - * /	, - * /	, - * /	, - * / ? L W C	, - * /	, - * / ? L C #	, - * /

Note, cron expressions are not case-sensitive.

Here is an example:

```
0 15 8 ? JAN MON 2014
```

This literally translates to 0 second, 15 minute, 8 hour, any day of the month, January, 2014.

In plain English, this represents 8:15am on every Monday during January of 2014. Note, the ? character means "no particular value". In this example, we've set the Day-of-month to no particular value. We don't need to specify it, as we've specified a Day-of-week value. Read more about special characters in the next section.

More examples of cron expressions are explained in the [Examples section](#) at the bottom of this page.

Special characters

Special character	Usage
,	Specifies a list of values. For example, in the Day-of-week field, 'MON,WED,FRI' means 'every Monday, Wednesday, and Friday'.
-	Specifies a range of values. For example, in the Day-of-week field, 'MON-FRI' means 'every Monday, Tuesday, Wednesday, Thursday and Friday'.
*	Specifies all possible values. For example, in the Hour field, '*' means 'every hour of the day'.
/	Specifies increments to the given value. For example, in the Minute field, '0/15' means 'every 15 minutes during the hour, starting at minute zero'.
?	Specifies no particular value. This is useful when you need to specify a value for one of the two fields Day-of-month or Day-of-week , but not the other.
L	Specifies the last possible value; this has different meanings depending on context. In the Day-of-week field, 'L' on its own means 'the last day of every week' (i.e. 'every Saturday'), or if used after another value, means 'the last xxx day of the month' (e.g. 'SATL' and '7L' both mean 'the last Saturday of the month'). In the Day-of-month field, 'L' on its own means 'the last day of the month', or 'LW' means 'the last weekday of the month'.

W	Specifies the weekday (Monday-Friday) nearest the given day of the month. For example, '1W' means 'the nearest weekday to the 1st of the month' (note that if the 1st is a Saturday, the email will be sent on the nearest weekday <i>within the same month</i> , i.e. on Monday 3rd). 'W' can only be used when the day-of-month is a single day, not a range or list of days.
#	Specifies the nth occurrence of a given day of the week. For example, 'TUES#2' (or '3#2') means 'the second Tuesday of the month'.

Examples

0 15 8 ? * *	Every day at 8:15 pm.
0 15 8 * * ?	Every day at 8:15 am.
0 * 14 * * ?	Every minute starting at 2:00 pm and ending at 2:59 pm, every day.
0 0/5 14 * * ?	Every 5 minutes starting at 2:00 pm and ending at 2:55 pm, every day.
0 0/5 14,18 * * ?	Every 5 minutes starting at 2:00 pm and ending at 2:55 pm, AND every 5 minutes starting at 6:00 pm and ending at 6:55 pm, every day.
0 0-5 14 * * ?	Every minute starting at 2:00 pm and ending at 2:05 pm, every day.
0 0/10 * * * ? *	Every 10 minutes, forever.
0 10,44 14 ? 3 WED	2:10 pm and 2:44 pm every Wednesday in the month of March.
0 15 8 ? * MON-FRI	8:15 am every Monday, Tuesday, Wednesday, Thursday, and Friday.
0 15 8 15 * ?	8:15 am on the 15th day of every month.
0 15 8 L * ?	8:15 am on the last day of every month.
0 15 8 LW * ?	8:15 am on the last weekday of every month.
0 15 8 ? * 6L	8:15 am on the last Friday of every month.
0 15 8 ? * 6#2	8:15 am on the second Friday of every month.
0 15 8 ? * 6#2 2007-2009	8:15 am on the second Friday of every month during the years 2007, 2008, and 2009.

Managing your user profile

You can manage your Jira settings (e.g. your password, email address, or the format in which you would like to receive email notifications) in your user profile. Your user profile also displays recent work in the Activity Stream, and contains useful shortcuts to issues you have been working on or reported.

To manage your user profile:

Choose **your user name** at top right of the screen, then choose **Profile**.

On this page:

- Editing your user details
- Changing your avatar
- Choosing your homepage
- Managing email notifications
- Managing your user preferences
- Managing service desk preferences
- Managing your OAuth and login tokens

Editing your user details

If your instance is using an external user management system like Crowd, these options may not be available to you.

In the **Details** section on the **Summary** page, click the edit icon



at the top-right of the section to edit your display name, email address, and password. If your Jira administrator has configured the user directory with external password management, the **Change Password** link will not be available.

Changing your avatar

Select



or your current avatar to change the image that appears next to your name in Jira. If your administrator has enabled [Gravatar for user avatars](#), your Gravatar (i.e. the Gravatar associated with the email address in your user profile) will automatically be set as your user avatar. If Gravatar has been enabled, you will not be able to choose Jira -specific user avatars and vice versa. using [Gravatar.com](#). If Gravatar has been disabled, you can choose your user avatar from the ones pre-packaged with Jira or upload your own.

- Your cropped image is resized to 48x48 pixels before it is saved as your new custom user avatar.
- A separate 16x16 pixel version of your custom user avatar will be generated for use in comments.
- Custom user avatars can only be selected by the user who uploaded them.

Choosing your homepage

Your Jira home page is the Jira page you are presented with immediately after you log in.

You can configure the following Jira pages as your Jira home page:

- The Dashboard
 - The Issue Navigator
 - The Rapid Board (available if you're using Jira Software)
1. Click on your **profile** icon at the top right of the screen.
 2. Select the appropriate home page option within the **My Jira Home** section:
 - Dashboard
 - Issue Navigator
 - Rapid Board (available if you're using Jira Software)
 3. (Optional) To verify that your Jira home page has been reset, log out and log back in to Jira again. You should be taken directly to the Jira home page you selected in the previous step.



Your page will be reloaded the Jira home page you selected.

Managing email notifications

In the **Preferences** section on the **Summary** page, click the **edit** icon



at the top-right of the section to open the **Updated User Preferences** dialog box. You can then manage the following:

- Change the **Email Type** to change the format (plain text or HTML) in which Jira sends its outgoing email notifications.
- In **My Changes**, Choose between making Jira send you email notifications about issue updates made by either both you and other people (**Notify me**) or other people only (i.e. **Do not notify me**).

Managing your user preferences

The global defaults for most of the user preferences below can be set by your Jira administrator; however, you can override these default settings by changing the following:

- The **Page Size**, or number of issues displayed on each Issue Navigator page
- Your preferred **language** from the drop-down list. If you don't see your preferred language in the list, see [Translating Jira](#) for more information.
- Your **time zone** specified in your profile doesn't match the time zone of the computer you are working on, Jira will ask if you want to update this selected time zone setting. All time fields in Jira will now be displayed in your preferred time zone.
- Choose the default **Sharing** setting for when you create new filters and dashboards, which can be either shared with all other users (**Public**) or restricted.
- Choose to enable or disable Jira's keyboard shortcuts feature.
- Choose between allowing Jira to make you an **autowatcher** of any issue that you create or comment on.

Managing service desk preferences

Service desk agents can enable or disable the **Pre-populated commenting** field by editing their user profiles. This setting can help save time by pre-filling conversation greeting text when agents comment on customer issues. When enabled, the text **Hi <Reporter_name>**, and **– <Agent_name>** appears in the comment field and in the email notification sent to customers.

Managing your OAuth and login tokens

An OAuth access token is issued by Jira to give [gadgets](#) access to restricted data on an external, OAuth-compliant web application or website (also known as a "consumer"). Check out [Allowing OAuth access](#) for recommendations on when to issue or revoke OAuth access tokens.

If you are accessing your Jira applications in a public environment, you can clear you login tokens by clicking the **Clear all Tokens** link in the Details section of your Profile.

Allowing OAuth access

About OAuth access tokens

OAuth access tokens allow you to:

- Use a Jira gadget on an external, OAuth-compliant web application or website (also known as a 'consumer')
- Grant the gadget access to the same Jira data that you can access.

Before you begin

Your Jira administrator must link your Jira instance and the consumer using an application link and OAuth. For example, if you want to add a Jira gadget to your Bamboo homepage, then your Jira administrator must first approve Bamboo as an OAuth consumer.

On this page:

- [About OAuth access tokens](#)
- [Before you begin](#)
- [Issuing OAuth access tokens](#)
- [Revoking OAuth access tokens](#)

Issuing OAuth access tokens

To allow a gadget to access the same Jira data that you can, Jira issues it an OAuth access token. The OAuth token is unique to the gadget.

1. When you use a Jira gadget on a consumer (such as Bamboo) and this gadget requires access to your Jira data, you will be prompted to log in to Jira if you have not already done so.
2. After you log in to Jira, you will be prompted with a **Request for Access** message.
3. To issue the OAuth token and grant the gadget access to your Jira data, click **Allow**. The gadget can access your Jira data until you revoke the token.
4. To view tokens you have issued, go to your **Profile > Tools > View OAuth Access Tokens**:

Authorized Applications			
The following applications are using your account to access JIRA data			
 Bamboo on server-gdn-bamboo.internal.atlassian.com	Approved on Apr 26, 2016 at 7:09 AM	READ AND WRITE ACCESS	Revoke Access
 Confluence	Approved on Apr 22, 2016 at 5:53 AM	READ AND WRITE ACCESS	Revoke Access
 JDOG - JIRA Team Dogfood on jdog.jira-dev.com	Approved on Apr 27, 2016 at 12:43 AM	READ AND WRITE ACCESS	Revoke Access
 Stash on stash.atlassian.com	Approved on May 12, 2016 at 6:10 AM	READ AND WRITE ACCESS	Revoke Access
 Atlassian JIRA Extranet - Special Projects on extranet.atlassian.com	Approved on Jul 15, 2016 at 12:01 AM	READ AND WRITE ACCESS	Revoke Access
 Atlassian JIRA on jira.atlassian.com	Approved on May 30, 2016 at 1:06 AM	READ AND WRITE ACCESS	Revoke Access

Revoking OAuth access tokens

You can revoke an OAuth access token to deny a Jira gadget access to your Jira data. When you revoke access, the gadget can only access public data on your Jira instance.

1. To view tokens you have issued, go to your **Profile > Tools > View OAuth Access Tokens**
2. Next to the application whose OAuth access you wish to revoke, click **Revoke Access**.
3. You may be prompted to confirm this action. If so, click **OK**.
4. The gadget's access token is revoked and the Jira gadget can only access public Jira data.

Requesting apps

The [Atlassian Marketplace](#) website offers hundreds of apps that administrators can install to enhance and extend your Jira applications. If the app request feature is enabled for your instance, you can submit requests for Marketplace apps directly to your administrator.

The 'Atlassian Marketplace for Jira' page presents an integrated view of the Marketplace website from within the Jira user interface. The page offers the same features as the Marketplace website, such as app search and category filtering, but tailors the browsing experience to Jira application users.

This in-product view of the Marketplace gives day-to-day users of the Atlassian applications, not just administrators, an easy way to discover the apps that can help them work. When you find an app of interest, you can submit a request with just a few clicks.

Submitting an app request

1. From anywhere in the application, open your profile menu and choose **Atlassian Marketplace**.
2. In the Atlassian Marketplace page, use the search box to find apps or use the category menus to browse or filter by apps by type, popularity, price or other criteria. You can see what your fellow users have requested by choosing the **Most Requested** filter.
3. When you find an app that interests you, click **Request** to generate a request for your administrator.
4. Optionally, type a personal message to your administrators in the text box. This message is visible to administrators in the details view for the app.
5. When ready, click **Submit Request**.
6. Click **Close** to dismiss the 'Success!' message dialog box.

At this point, a notification appears in the interface your administrators use to administer apps. Also your request message will appear in the app details view, visible from the administrator's 'Find New apps' page. From there, your administrator can purchase the app, try it out or dismiss requests.

Updating an app request

After submitting the request, you can update your message at any time. Click the **Update Request** button next to the listing in the 'Atlassian Marketplace' page to modify the message to your administrator.

The administrator is not notified of the update. However, your updated message will appear, as you have modified it in the details view for the app immediately.

Using keyboard shortcuts

Keyboard shortcuts are a great way for you to speed up editing, navigating, and for performing actions without having to take your fingers off the keyboard.

Some keyboard shortcuts require additional permissions or applications, and depend on how your Jira administrator(s) have configured permissions for your user account and which applications are installed.

On this page:

- [View keyboard shortcuts](#)
- [Enabling and disabling keyboard shortcuts](#)

View keyboard shortcuts

- Choose  at top right of the screen, then choose **Keyboard shortcuts**.
- When viewing a page, press **Shift + /**.

The Keyboard Shortcuts dialog is displayed and shows commands for the operating system and browser that you are using. The dialog is divided into sections for the following information:

- **Global shortcuts** - shortcuts that can be used when you are in any part of Jira
 - **Navigating issues** - shortcuts for navigating through issues
 - **Issue actions** - shortcuts for working with issues
 - **App specific** - any application-specific shortcuts. These shortcuts only work in the listed application.
- ▼ [More about the Keyboard Shortcuts dialog...](#)

If you have other Jira applications installed, you may have additional keyboard shortcuts available. For example, if you have Jira Software installed, you will see a series of additional keyboard shortcuts in the lower-right of this dialog box (and some additional **Global** keyboard shortcuts specific to Jira Software in the upper-left section). However, the keyboard shortcuts in the **Agile Shortcuts** section only function in Jira Software, and not in a Jira context.

Enabling and disabling keyboard shortcuts

Keyboard shortcuts are enabled by default. However, you can disable them on a per-user basis in the Keyboard Shortcuts dialog box.

1. Ensure you are logged in and open the Keyboard Shortcuts dialog box (see [above](#)).
2. At the bottom of the Keyboard Shortcuts dialog box, click **Disable Keyboard Shortcuts** or **Enable Keyboard Shortcuts**.

You can also disable or re-enable keyboard shortcuts by editing the Preferences section of your user profile. See [Managing your user profile](#) for more information.

Modifier keys

Some keyboard shortcuts require modifier keys to be pressed simultaneously, along with a single 'action' key. Modifier keys may differ, depending on your combination of operating system and web browser. The following table identifies the modifier keys for some supported web browsers and operating systems:

Web Browser	Mac OS X	Windows	Linux/Solaris	Notes
Firefox	Ctrl	Alt + Shift	Alt + Shift	In Firefox, it is possible to customize 'Modifier key shortcuts'. Please read Mozilla's documentation for more information.
Internet Explorer		Alt		Typing a 'Modifier key shortcut' that leads to a link requires you to press the 'Enter' to complete the action.
Safari	Ctrl + Alt/Option	Ctrl		
Chrome	Ctrl + Alt/Option	Alt + Shift	Alt + Shift	

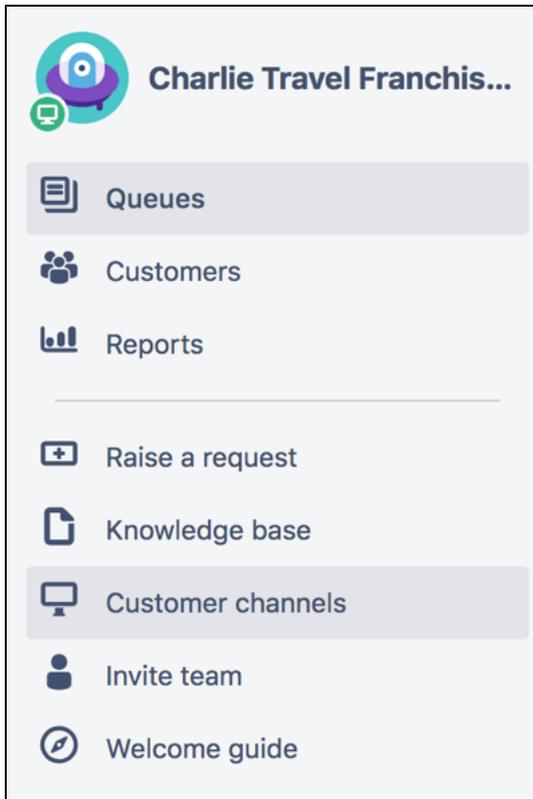
Adding announcements

Your service desk project comes with a customizable customer-facing site called the *customer portal* that customers can use to raise and track requests. If you link your project to a Confluence knowledge base space, customers can also self-service issues by searching for relevant articles. You can customize what customers can do on the portal, and change the look of the portal.

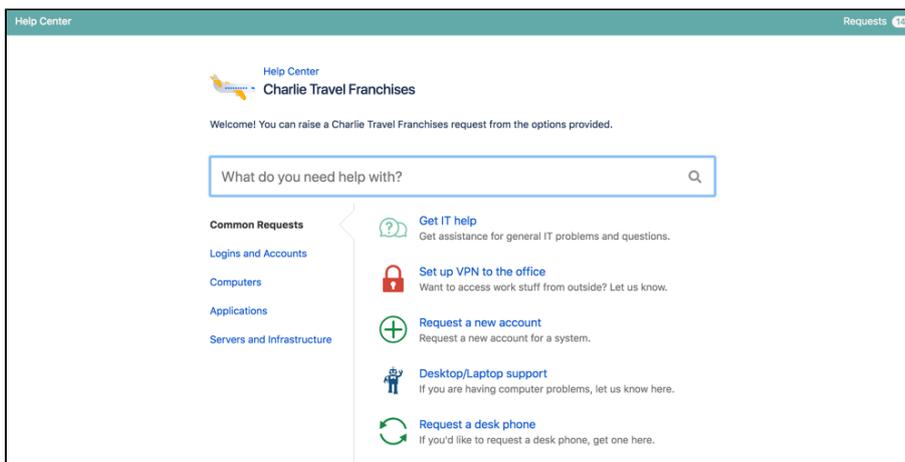
To access the customer portal, click **Customer channels** in the project sidebar:

On this page:

- [Set up request types](#)
- [Add transitions](#)
- [Manage access to your portal](#)
- [View all portals in your help center](#)
- [Brand your portal](#)
- [Add announcements](#)



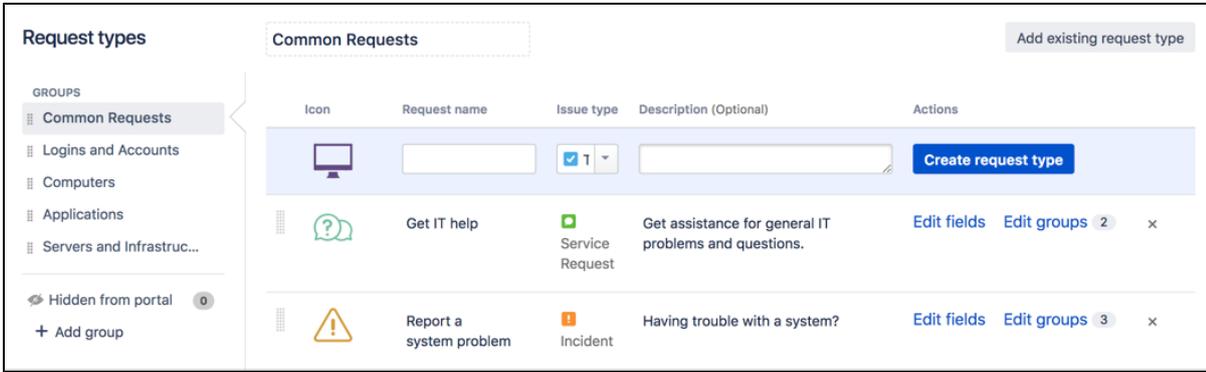
The portal displays the request types that customers can raise. You can also change the logo, name, and welcome message to reflect your brand:



You must be a **Jira Service Desk administrator** or **project administrator** for this service desk project, to make the following changes.

Set up request types

You can customize the types of requests that customers raise from the portal. To create and manage request types, visit **Project settings > Request types**.



Service desk projects include several request types that address common IT help scenarios. The request types are organized into groups to help customers find what they need on the portal. For example, you can add a "Common Requests" group to help customers get assistance with the most commonly reported issues like system problems or IT support.

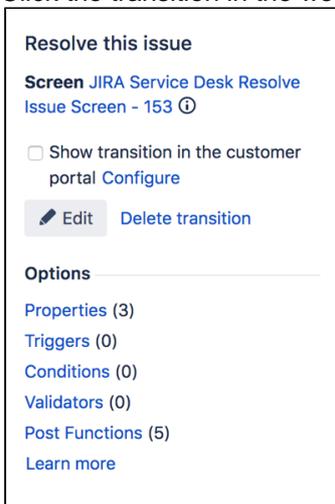
To learn more about customizing request types, check out [Setting up request types](#).

Add transitions

You can show transitions on the customer portal so that customers can transition their requests. For example, say an agent shares a knowledge base article with a customer. If the article solves the customer's problem, then the customer can resolve the request.

To add a transition to the portal, you edit an existing transition in the workflow.

1. In your service desk project, click **Project settings > Workflows**.
2. Click  next to the workflow that contains the transition you want to add to the portal.
3. Click **Diagram** to open the diagram view.
4. Click the transition in the workflow, and then select *Show transition on the customer portal*.



Customer transitions behave slightly differently than other workflow transitions:

- Screens don't display on the customer portal. When you add a transition to the portal, you can set a resolution for requests that customers transition.
- When an issue is transitioned from the portal, it bypasses any validators that are defined for the transition.

If it seems like the portal transition isn't working properly, make sure there isn't an automation rule in conflict with the transition.

To learn more about workflows and transitions, see the [advanced workflow](#) configuration page.

Manage access to your portal

You can open your customer portal to allow new customers to create an account and submit a request to your team. If you don't want to have an open portal, you can restrict access to:

- Customers who have an existing account for any other Jira product or service desk project
- Customers who appear specifically on your service desk project's customer list

For more information about opening or restricting your portal, see [Managing access to your service desk](#).

The customer portal integrates with [Atlassian Crowd](#), Atlassian's single sign-on (SSO), authentication, authorization, application provisioning, and identity management framework. For information about integrating with third-party SSO providers, see this [page](#).

View all portals in your help center

If your company uses multiple service desk projects (for example, an IT service desk and an office administration service desk), you can provide your customers with a single URL to find a list of all the customer portals they have access to and the requests created in each one:

```
http://<computer_name_or_IP_address>:<HTTP_port_number>/Jira/servicedesk/customer/portals
```

The URL you provide will send customers to what we call the Help Center. The Help Center displays all customer portals generated by service desk projects in a single instance of Jira Service Desk, as well as the header you [previously branded](#).

To make any changes to the header, or to update the name of your Help Center, go to



> **Applications > Jira Service Desk Configuration.**

Brand your portal

You can customize your customer portal to reflect your team and company's brand with the following two steps:

1. In **Project settings > Portal settings**, add a portal logo and a short description to familiarize customers with your service desk:

Portal settings

Look and feel
[View and customize](#) the look and feel of your Help Center.

Name
Charlie Travel Franchises

Introduction text (optional)
Welcome! You can raise a Charlie Travel Franchises request from the options provided.

Logo
 Use a custom logo for this Customer Portal

Images are resized to 64 pixels (height)
 Choose logo

Save logo

Announcements
[View and change](#) your Customer Portal announcement.

Can agents add announcements to this portal?

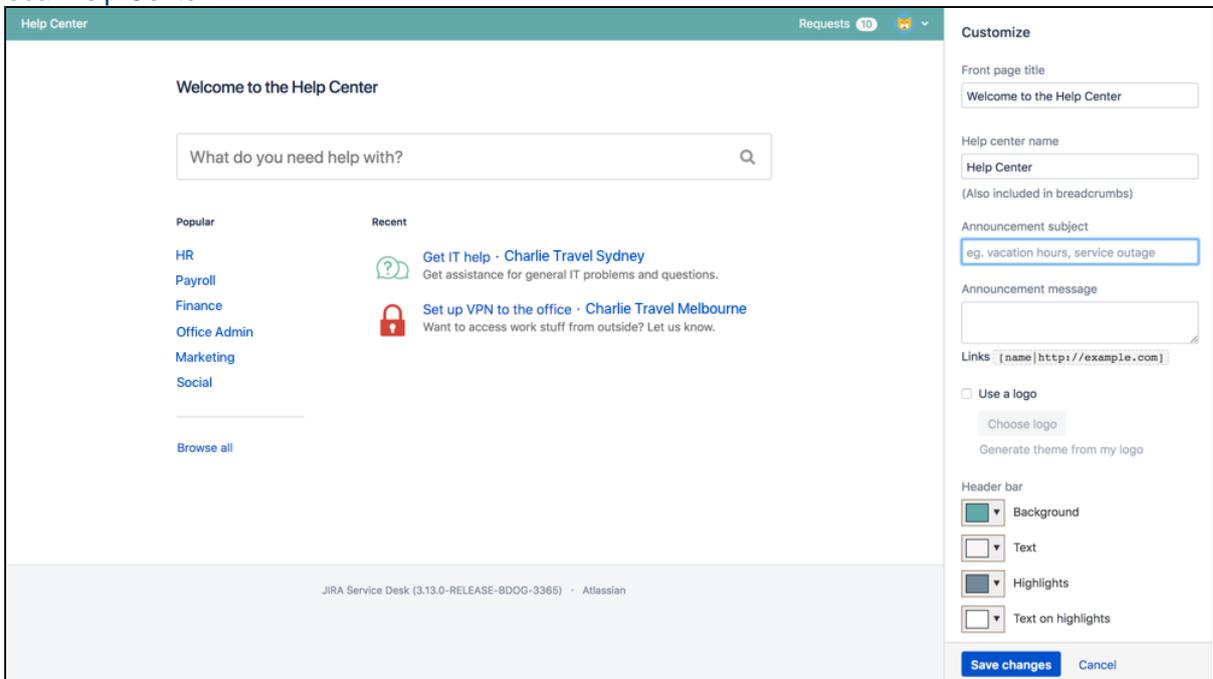
Yes, agents can add announcements to this portal
 No, only JIRA or Project administrators can add announcements to this portal

Agents are not allowed to change the Help Center announcement. You can change this setting on the [JIRA Service Desk configuration page](#).

2. Follow the **View and customize** link (or select



> **Applications > Jira Service Desk Configuration**) to brand your Help Center in a live preview mode. The header and branding changes made here apply to all service desk project portals and the global Help Center:



Add announcements

Do you want to communicate a system outage or post a message about operating hours? Announcements are a convenient way to alert your customers about outages or events. You or your agents can add announcements to the Help Center and portals.

Add announcements to the Help Center

1. Go to



> **Applications** > **Jira Service Desk Configuration** > **Help Center**.

2. Complete the **Announcement subject** and **message**. The message displays on the Help Center, and on the **My Projects** page.

Add announcements to the portal

1. In the project view for a specific service desk project, click **Project administration** > **Portal settings**.
2. On the **Portal settings** page, click **View and change** your Customer Portal announcement, and add the message you want the customers of that particular project to see.

Want to add a human touch to your service desk? Share a productivity tip or just say hello.

Allow agents to add announcements

To allow agents to add announcements to the Help Center, Jira administrators can go to



> **Applications > Jira Service Desk Configuration**. In the Help Center setting, choose **Yes**.

To allow agents to add announcements to the portal, project administrators can go to **Project administration > Portal settings**. In the announcements section, choose **Yes**.

Adding customers

As an agent, you can add customers to your service desk. If your administrator grants you permission, you can also manage organizations in the project.

- **Customers** raise requests in your service desk.
- **Organizations** are groups of customers that are shared across projects.

On this page:

- Add customers
- Add organizations
- Add customers to an organization
- Remove customers from an organization
- Remove organizations from a project

Add customers

Add customers to a project via **Customers**



> **Add customers**. Customers on this list can raise requests in the project, via the portal or email.

Customers are automatically added to the list if your service desk is open to users with Jira accounts, or allows customers to create their own accounts.

Add organizations

Add a new or existing organization to a project via **Customers**



> **Add organizations**. Organizations that you add display on the **Customers** list, and their members can raise requests in the project via the portal or email.

Add customers to an organization

1. Go to the **Customers**



2. Select the organization in the list.
3. Click **Add customers**. The customers you add can raise requests in all projects that use the organization.

Because the organization can be used in multiple projects, the customers are not added to the **Customers** role for the project.

Remove customers from an organization

1. Select an organization in the **Customers** list.
2. Find the customer you want to remove, and then click **X** next to their name.

Customers you remove lose access to projects that use the organization, unless they have access through another organization, or have the **Customers** role for the projects.

Remove organizations from a project

1. Select an organization in the **Customers** list.
2. Select **Remove from project**.

When you remove an organization from a project, its members lose access to the project unless they have the **Customers** role for the project.

Adding request participants

Request participants are Jira Service Desk customers who watch requests. You might add request participants so they can provide more information about a request, or to keep them in the loop about a request's progress.

On this page:

- [About participants](#)
- [Add participants to an issue](#)
- [Add participants in the portal](#)
- [Add participants via email](#)
- [Add watchers](#)

About participants

Request participants are customers and organizations who can view, comment, and receive notifications about the request. By default, request participants will receive notifications about the request they have been added to and can turn off notifications, at any time, in the customer portal or email. If a request is shared with an organization rather than an individual, the customers within that organization must opt in for each request they would like to receive notifications about.

Both agents and customers can add and remove request participants. Who they can add as participants depends on the project's [Customer permissions](#).

Request participants follow issue-level security schemes. For example, if an administrator customizes requests so that only reporters can view them, then request participants won't be able to view the request. Administrators can refer to the instructions in [Configuring Issue-level Security](#) to update the issue security scheme.

Add participants to an issue

Agents can add request participants in the issue view under **People > Request participants** or **People > Organizations**. When you add a request participant, they receive an email notifying them that they are participating in the request.

The following image shows the request participants in an issue:

> People

Assignee:  Karen Bywater

Reporter:  Mitch Davis

Request participants:  Alana Grant

Organizations:  Charlie Travel

Votes: 0 [Vote for this issue](#)

Watchers: 1 [Start watching this issue](#)

Add participants in the portal

In the customer portal, agents and customers can add participants by selecting **Share** (



). Participants receive an email notifying them that they are participating in the request.

Don't notify me

Share

Share this request

Type name, email address, or organization

Share [Cancel](#)

If a customer is in an organization, they can also share the request when they raise it. By default, the request is shared with the customer's organization unless they select **Private**. If the customer is in more than one organization, the request is private by default.

Add participants via email

If you create or respond to a request via email, add a request participant's email address in the TO or CC fields. The participant receives an email notifying them that they have been added.

Customers can add individual participants if they have permission to share requests. If the customer is in one organization, their request is shared automatically. Customers who are in more than one organization can't share requests with an organization via email.

Add watchers

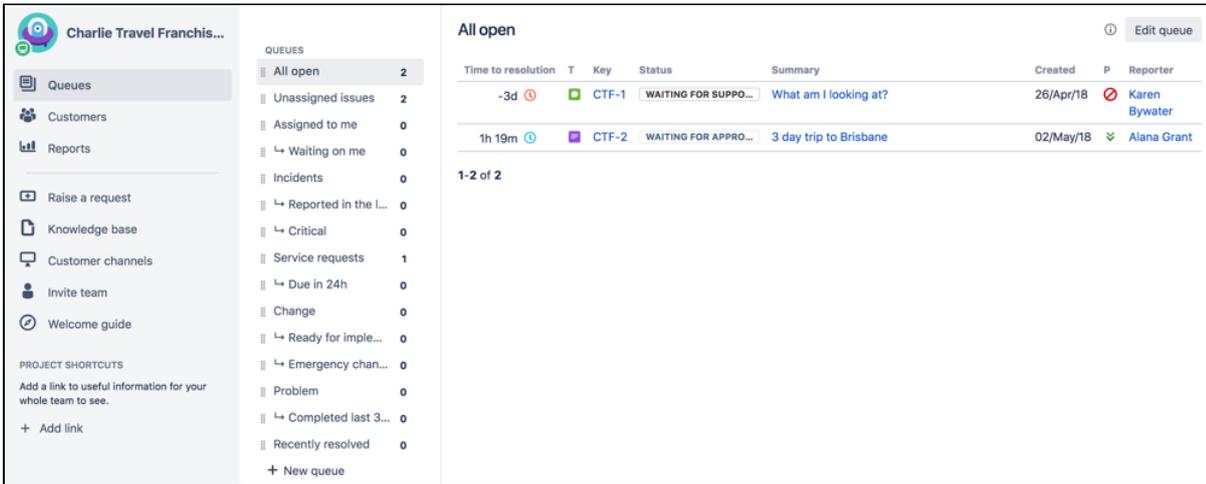
You can also involve other agents or Jira users to get help with an issue. For example, you might want Jira Software developers to help analyze a bug that a customer has reported. To involve internal users, add them as watchers. As watchers, they're notified about internal activity on an issue, and can communicate with you via internal comments.

Using service desk queues

Customer requests become issues that you can view and work on in queues. Jira Service Desk comes with

default queues that your administrator can update to automatically triage issues for your team. As an agent, you can see how many issues are in each queue, and switch between queues to work on the right issues at the right time.

You can easily navigate to your service desk queues at any time by selecting **Queues** from your project sidebar.



Switching queues

When you select **Queues** from your project sidebar for the first time, the secondary sidebar menu will open automatically. This sidebar displays all queues in your service desk project, as well as the number of issues in each queue. Simply select the name of the queue you wish to work from to view its issues.

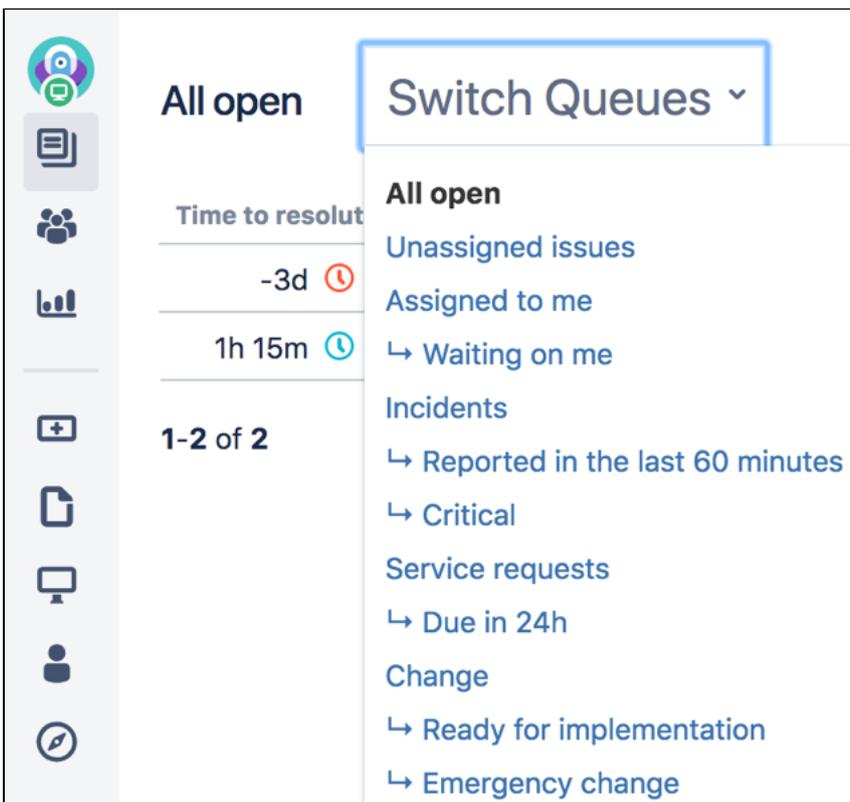
To expand the view of a single queue, you can minimize your project sidebar by selecting



and minimize your queue sidebar by selecting



in the sidebar's upper right corner. When the queue sidebar is collapsed, a Switch queue dropdown will appear, which you can use to view a different queue or to reopen the queue sidebar:



Raising requests on behalf of customers

Let's say you're helping a customer resolve an issue over the phone, and you need to follow up with additional information. You can use the customer portal to quickly enter your customer's name, fill in the issue details, and submit the service desk request.

If you don't need to create a request, but simply want to invite customers to your service desk so they know how to get help, you can skip ahead to [Invite a new customer](#).

On this page:

- [Raise a customer request](#)
- [Invite a new customer](#)

Raise a customer request

1. From your service desk project sidebar, select



2. Select the request type that matches your customer's need.
3. In the **Raise this request on behalf of** field, enter a new customer's email address, or search for an existing customer:

 A screenshot of the Jira Service Desk interface. At the top, it says "Help Center / Charlie Travel Franchises" and "Get IT help" with a question mark icon. Below that, there is a section titled "Raise this request on behalf of" with a dropdown menu showing "Alana Grant" and a person icon.

4. Fill in the request details. If the customer is in an organization, you can choose to share the request with that organization.
5. Select **Create**. Your customer is emailed a link to the new request. New customers also receive an invitation to finish creating a service desk account.

Invite a new customer

1. From your service desk project sidebar, select **Customers**.
2. Select **Add customers** and enter your customer's email address.
3. If they belong to an organization, select it from the **Add to organization** dropdown.
4. Select **Add**, and you're done!

Organizing work with versions

Versions are points-in-time for a project. They help you organize your work by giving you milestones to aim for. You can then assign the issues in your project to a specific version, and build up the work you need to do to complete that version.

On this page:

- Managing a project's versions
- Add a new version
- Release a version
- Archive a version
- Delete a version
- Merge multiple versions
- Reschedule a version

You need to have the project-specific **Administer Projects** project permission or the **Jira Administrator** global permission to be able to:

- Add — create a new version against which issues can be aligned.
- Release — mark a version as released.
- Archive — hide an old version from the Releases report, and in the user interface.
- Delete — remove a version. You must choose an action for any issues with that version.
- Merge — combine multiple versions into one.
- Reschedule — re-arrange the order of versions.

Once a version has been created for a project, the 'Affects version' and 'Fix version' fields will become available for your issues. If you cannot see these fields on your issue, your project may not have any version yet, or the fields are hidden from view.

Managing a project's versions

The easiest way to manage a project's versions is through the Versions page.

1. Choose



> **Projects**, and click the name of the project.

2. Choose **Versions** in the sidebar. The **Versions** page is displayed, showing a list of versions.

Screenshot: The 'Versions' page

Name	Description	Start date	Release date
2.1	Bug fix and feature polish		27/Mar/13
3.0	5 New Levels, Android Support		17/Feb/13
2.0	UI cleanup		27/Mar/13
1.3	The Phantom Nerd		29/Jul/11
1.2	Return of the Nerds		15/Jun/11
1.1	The Empire Strikes Nerds		08/Jun/11
1.0	A New Hope for Nerds		01/Apr/11

Add a new version

1. The Add Version form is located at the top of the 'Versions' page.
2. Enter the name for the version. The name can be:
 - simple numeric, e.g. "2.1", or
 - complicated numeric, e.g. "2.1.3", or
 - a word, such as the project's internal code-name, e.g. "Memphis".
3. Optional details such as the version description (text not HTML), start date and release date (i.e. the planned release date for a version) can be also be specified. These can be changed later if required.
4. Click the **Add** button. You can drag the new version to a different position by hovering over the 'drag' icon
 
 at the left of the version name.

Release a version

1. On the 'Versions' page, hover over the relevant version to display the cog icon, then select **Release** from the drop-down menu.
2. If there are any issues set with this version as their 'Fix For' version, Jira allows you to choose to change the 'Fix For' version if you wish. Otherwise, the operation will complete without modifying these issues.

To revert the release of a version, simply select **Unrelease** from the drop-down menu.

Archive a version

1. On the 'Versions' page, hover over the relevant version to display the cog icon, then select **Archive** from the drop-down menu.
2. The version list indicates the version 'archived' status with a semi-transparent icon. No further changes can be made to this version unless it is un-archived. Also it is not possible to remove any existing archived versions from an issue's affected and fix version fields or add any new archived versions.

To revert the archive of a version, simply select **Unarchive** from the drop-down menu.

Delete a version

1. On the 'Versions' page, hover over the relevant version to display the cog icon, then select **Delete** from the drop-down menu.
2. This will bring you to the 'Delete Version: <Version>' confirmation page. From here, you can specify the actions to be taken for issues associated with the version to be deleted. You can either associate these issues with another version, or simply remove references to the version to be deleted.

Merge multiple versions

Merging multiple versions allows you to move the issues from one or more versions to another version.

1. On the 'Versions' page, click the **Merge** link at the top right of the page.
2. The 'Merge Versions' popup will be displayed. On this page are two select lists — both listing all un-archived versions.

In the 'Merging From Versions' select list, choose the version(s) whose issues you wish to move. Versions selected on this list will be removed from the system. All issues associated with these versions will be updated to reflect the new version selected in the 'Merge To Version' select list. It is only possible to select one version to merge to.
3. Click the **Merge** button. If you are shown a confirmation page, click **Merge** again to complete the operation.

Reschedule a version

Recheduling a version changes its place in the order of versions.

- On the 'Versions' page, click the



icon for the relevant version, and drag it to its new position in the version order.

Organizing work with components

Components are used to organize or group customer requests in a service desk project. You could set up a component for systems that your teams are responsible for (e.g. company intranet), and then set a default assignee so that any customer request about that system is assigned to the agent who manages it.

You need to have the project-specific **Administer Projects** permission or the **Jira Administrator** global permission to be able to:

- Add — create a new component against which issues can be aligned
- Edit — change a components details
- Delete — remove a component

Once a component has been created for a project, the 'Component' field becomes available for your issues. If you cannot see this field on your issue, your project may not have any components yet, or the field is hidden from view.

On this page:

- [Managing a project's components](#)
- [Adding a new component](#)
- [Selecting a default assignee](#)
- [Editing a component's details](#)
- [Deleting a component](#)

Managing a project's components

The easiest way to manage a project's components is through the Components page.

1. Choose



> **Projects**, and click the name of the project.

2. Choose **Components** in the sidebar. The **Components** page is displayed, showing a list of components and each component's details. From here, you can manage the project's components as described below.

Adding a new component

1. The Add Component form is located at the top of the 'Components' screen.
2. Enter the **Name** for the component. Optionally, enter a **Description** and select a **Component Lead** and **Default Assignee** (see [options](#) below).
3. Click **Add**.

Selecting a default assignee

You can optionally set a default assignee for a component. This will override the project's default assignee for issues in that component. If an issue has multiple components, and the default assignees of components clash, the assignee will be set to the default assignee of the component that is first alphabetically.

Default assignee option	Description	Notes

Project Default	Issues matching this component will have the assignee set to the same default assignee as the parent project.	
Project Lead	The assignee will be set to the project leader.	If the project leader is not permitted to be assigned to issues in the permission scheme, this option will be disabled and will say "Project Lead is not allowed to be assigned issues".
Component Lead	The assignee will be set to the component leader.	If the component leader is not permitted to be assigned to issues in the permission scheme, this option will be disabled and will say "Component Lead is not allowed to be assigned issues". The Component Lead option will also not be available if the component does not have a lead assigned to the component. Instead, under this option, it will say "Component does not have a lead".
Unassigned	The assignee of the issue will not be set on the creation of this issue.	This option will only be available if "Allow unassigned issues" is enabled in the general configuration.

Editing a component's details

1. On the 'Components' screen, open the menu in the **Actions** column for the component you want to edit, and select **Edit**.
2. Edit the component's **Name**, **Description**, **Lead**, and **Default Assignee** in the **Edit component** dialog.
3. Click the **Save** button to save your changes.

Searching for a component

If you need to find a component in a long list, it's easiest to search for it. Start typing text into the search box that you know the component contains, and your list will automatically be filtered for you.

Deleting a component

1. On the 'Components' screen, open the menu in the **Actions** column for the component you want to delete, and select **Delete**.
2. You will be prompted to associate any issues assigned to this component with another component if you wish.
3. Click the **Delete** button to delete the component.

Workflows

All Jira projects contain issues that your team can view, work on, and transition through stages of work — from creation to completion. The path that your issues take is called a workflow. Each Jira workflow is composed of a set of statuses (the state your work can be in) and transitions (how your work moves between statuses) that your issue moves through during its lifecycle, and typically represents work processes within your organization.

In addition, Jira uses workflow schemes to define the relationship between issue types and workflows. Workflow schemes are associated with a project, and make it possible to use a different workflow for different combinations of project and issue types.

Jira administrators and project administrators have different permissions when it comes to workflows.

Project administrators

As a project administrator, you can only edit a workflow that belongs to your project if:

- you have the *Extended project administration* permission, which is enabled by default (you can check that in **Project settings > Permissions**),
- the workflow isn't shared with any other projects (it's only available in your project),
- the workflow isn't the default Jira workflow (no-one can edit these workflows).

If the workflow is shared with another project, you'll see that information when you view the workflow. You'll also see how many issue types share the workflow, and would be affected by any changes you may make. You can make the following changes to the workflow:

- Add a status (the statuses must already exist in the Jira instance, you can't create, edit or remove statuses),
- Delete a status (the statuses must not be used by any of the project's issues),
- Create, add, edit or delete transitions (you can't select or update a screen used by the transition, or edit or view a transition's properties, conditions, validators or post-functions).

To view a workflow

1. Select **Projects** and choose the project whose workflow/s you want to view.
2. Select **Project settings** in the sidebar.
3. Select **Workflows** to see the list of workflows and issue types they're associated with.
 - Click a workflow to display it as diagram. If you're able to edit the workflow, you'll see an **Edit** button. If the workflow is shared with another project or issue type/s, that information will be available, and you can view it by clicking the relevant link.
 - Additionally, you can view a workflow in a simple, text form by clicking **View as text** next to the workflow's name.

To edit a workflow

1. When viewing a workflow, select **Edit**.
2. You can add a status or transition by clicking the relevant button. You can edit existing transitions by selecting them.
3. **Publish** your workflow to make it active.

If you don't publish the workflow, it'll remain as a draft until such time as you publish it, or discard it.

If you have a draft workflow present on your project, and you want to see the original workflow that's currently active, select **Project settings > Workflows**, and click a workflow.

Jira administrators

As a Jira administrator, you can complete the actions listed in the table below. The actions you have available are more extensive, and the documentation links will direct you to the Administrator documentation set.

What you can do...	Documentation
<ul style="list-style-type: none"> • Edit existing workflows • Create new workflows • Configure existing workflows 	Working with workflows
<ul style="list-style-type: none"> • Add a workflow scheme • Configure a workflow scheme • Manage workflow schemes 	Configuring workflow schemes
<ul style="list-style-type: none"> • Import and export workflows • Activate and deactivate workflows 	Managing your workflows

- Add custom events
- Configure the initial status
- Work in text mode
- Configure workflow triggers
- Use validators and custom fields
- Use XML to create a workflow
- Workflow properties

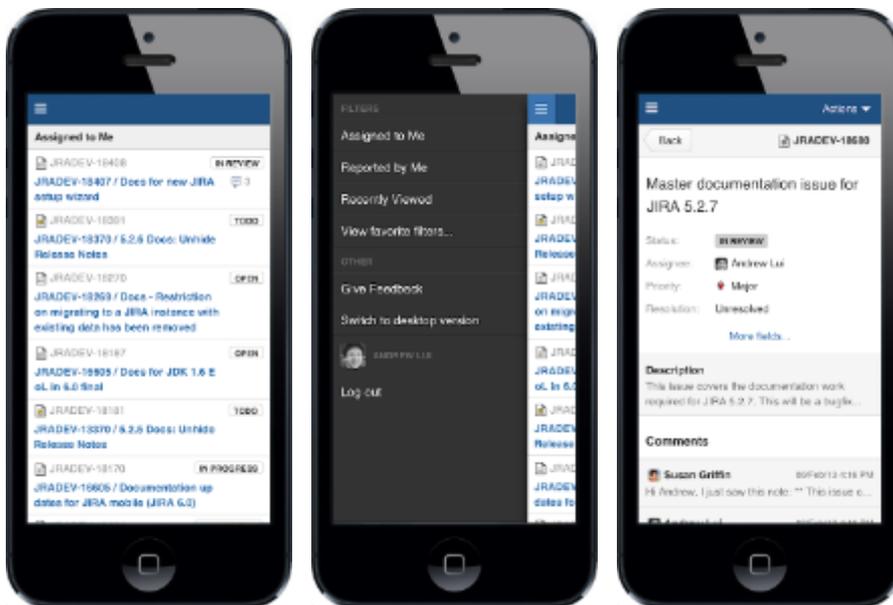
[Advanced workflow configuration](#)

Using Jira on a mobile device

When you view a Jira page on a mobile device, such as an iPhone or an Android phone, Jira will display an optimized version of the page. Jira chooses the mobile or desktop interface based on your device.

The Jira mobile interface is designed for viewing and interacting with issues on the go. If you need full access to Jira, you can always switch to the Jira desktop interface via the mobile menu (shown in the screenshots below).

What does Jira look like on a mobile device?



What can you do in Jira on a mobile device?

The Jira mobile interface has been designed to give users quick access to their issues on the go. This includes;

- Viewing issues, comments, attachments, issue links and your favorite filters.
- Performing basic operations like adding comments, watching or voting on issues and assigning issues to users.

If you need to create or modify issues on the go, you can still do so by switching to the desktop interface via the mobile menu (shown in the screenshots above).

Frequently asked questions

- [What mobile devices are supported?](#)
- [Do I need to install an app to view Jira on a mobile device?](#)
- [Can I access my Jira Cloud site via a mobile device?](#)
- [Why can't I view my custom field in Jira on my mobile?](#)

What mobile devices are supported?

See [Supported Platforms](#) for details of supported mobile devices.

Do I need to install an app to view Jira on a mobile device?

Currently, you can only use Jira mobile app for Jira cloud instances. If you run a JIRA server instance, you can view Jira on a mobile device using a web interface (optimized for mobile devices). Simply browse to your Jira server's URL using your mobile browser to bring up the mobile interface for Jira.

Can I access my Jira Cloud site via a mobile device?

Yes, just enter the URL of your Jira Cloud site in your mobile web browser.

Why can't I view my custom field in Jira on my mobile?

The Jira Mobile interface will show custom fields in the issue details screen. Custom fields that have their own custom field renderer will not display on the Jira Mobile interface. You will need to switch to the desktop interface to view these fields.

Can I disable Jira mobile for my Cloud site?

You can disable Jira mobile for your Cloud site, so that users will only be able to access the desktop view of Jira on their mobile device.

Jira mobile is implemented as an app in Jira, so you can disable it by disabling the app. For instructions on disabling apps, see [Managing apps](#). Note, Jira mobile is a system app.

Configuring dashboards

Your dashboard is the main display you see when you log in to your project. You can create multiple dashboards for different projects, or multiple dashboards for one big project. Each project has a default dashboard, or you can create a personal dashboard and add gadgets to keep track of assignments and issues you're working on. Dashboards are designed to display gadgets that help you organize your projects, assignments, and achievements in different charts.

You can see all dashboards by selecting the **Dashboards** drop-down from your Jira application header.

On this page:

- [About the default dashboard](#)
- [Creating a dashboard](#)
- [Managing dashboards and permissions](#)
- [Sharing and editing your dashboard](#)
- [Adding favorite dashboards](#)
- [Note on dashboard permissions](#)
- [Setting up a Wallboard](#)

About the default dashboard

The gadgets on the default dashboard can be reordered and switched between the left and right columns. Additional gadgets can also be added, while some gadgets can be configured. The layout of the dashboard (e.g. number of columns) can also be configured.

All changes made to the default dashboard will also change the dashboards of all users currently using the default dashboard. However, gadgets that users do not have permissions to see will not be displayed to them. For example, the 'Administration' gadget, although it may exist in the default dashboard configuration, will not be visible to non-admin users.

Creating a dashboard

You can easily create and customize your own dashboard to display the information you need. Note that only administrators can customize the default dashboard for your project.

1. At the top right of the Dashboard, select



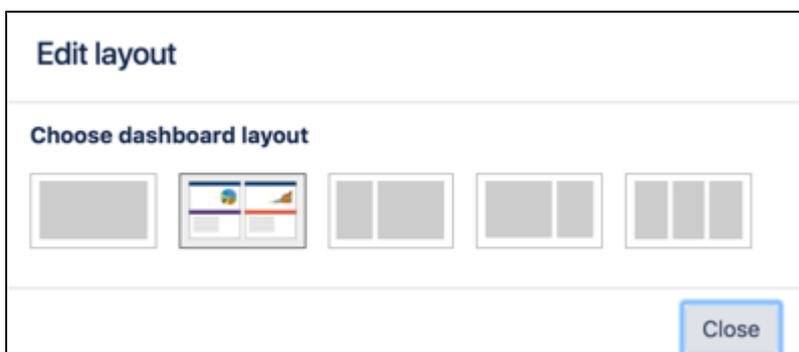
2. Select either **Create Dashboard** to create a blank dashboard, or **Copy Dashboard** to create a copy of the dashboard you are currently viewing.
3. Name and describe your dashboard.
4. Fill out the rest of the fields as applicable.
5. Click **Add**.

By default, sharing is set to private if you have not specified a personal preference. You can adjust this setting in the sharing preferences in your [user profile](#), and change dashboard permissions at any time in the Manage Dashboards page.

Choosing a dashboard layout

To choose a different layout for your dashboard page (for example, three columns instead of two):

1. At the top right of the Dashboard, click **Edit layout**. A selection of layouts will be displayed:



2. Choose your preferred layout.

Managing gadgets

To get the most out of your dashboard, including adding, rearranging, removing, and configuring gadgets, see [Adding and customizing gadgets](#).

Managing dashboards and permissions

You can edit, delete, copy, mark favorites, and share your dashboards from the Manage Dashboards page.

1. Select **Dashboards > Manage Dashboards**.
2. Choose the dashboard.

Sharing and editing your dashboard

You can edit the details for your dashboard, and restrict or share with other users according to the permissions that are set. In addition, you can see all the dashboards you've created, any public dashboards, and any shared dashboards.

1. Click



> **Edit**. (If you're viewing the dashboard, go to

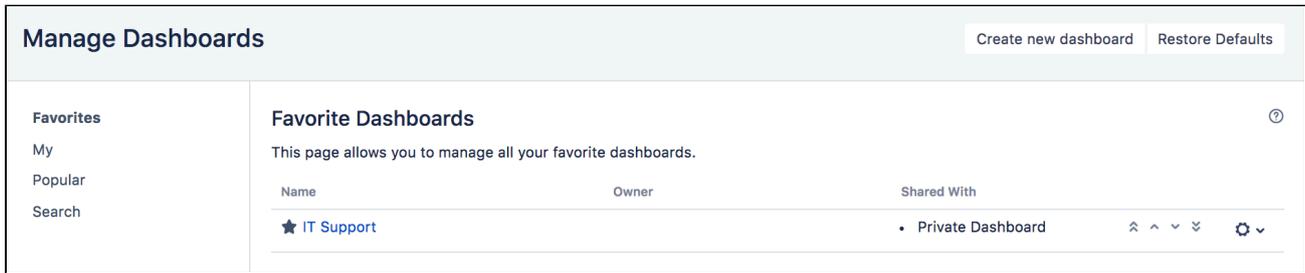


> **Edit/Share Dashboard**).

2. Edit the settings.

Adding favorite dashboards

If you find a dashboard you like, click the star icon next to its name to add it to your favorite dashboards list. You can also add the default dashboard to your favorites list so it's easily available to you.



Note on dashboard permissions

Jira administrators, as set in global permissions, can manage their users' shared dashboards in the **Shared dashboards** menu. Administrators can also change the ownership of a dashboard if the creator is unable to maintain the dashboard or its gadgets. See [Managing shared dashboards](#) for more information.

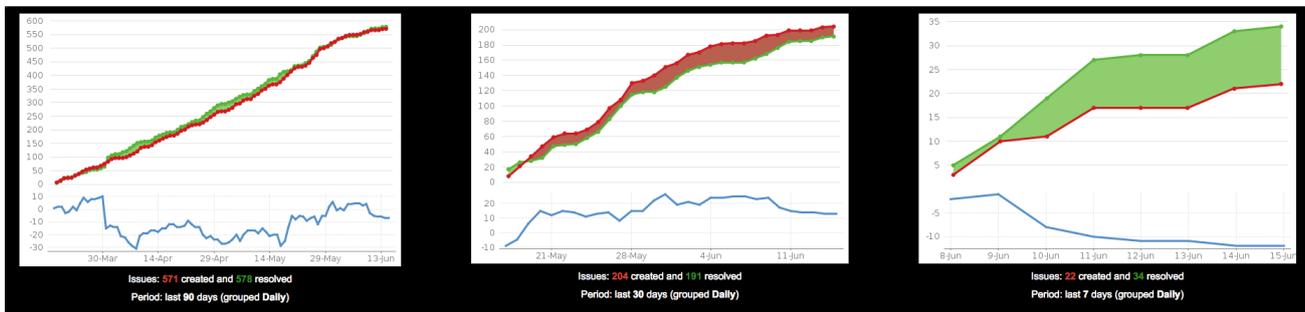
Setting up a Wallboard

Turn any Jira application dashboard into a wallboard by plugging your computer into a TV monitor. The Wallboard is a dashboard **gadget** that acts as an information radiator to provide instant visual insight into project progress and team accomplishments. With your favorite dashboard selected, select



> **View as Wallboard**. The dashboard will appear against a black background, and will rotate gadgets if the user enables the slideshow option.

The Wallboard below shows the same **Created vs. Resolved Issues** gadgets and data above.



Adding and customizing gadgets

Adding a gadget to a dashboard

You can add gadgets to your own personal dashboard(s). To add a gadget to the default dashboard for your Jira application, you must be a Jira admin.

Some applications allow dashboards that are shared by groups of people. If you have permission to update a shared dashboard, the other people sharing the dashboard will see your changes, too.

1. Go to the dashboard by selecting the **Dashboards** link in the header.
2. If you don't already have a dashboard, select **Manage Dashboards** from the dropdown, then **Create new dashboard**.
3. Once your dashboard is created, on the dashboard, select **Add Gadget**.
4. Use the gadget wizard to choose the gadgets you want to add. You can see a list of these gadgets in [Gadgets for Jira applications](#).

For more information about managing dashboards, see [Configuring dashboards](#).

Customizing how gadgets look

There are a few ways you can customize the view of gadgets in a dashboard:

To	Do this
Expand or collapse gadgets	Use the  button in the gadget header.
Expand a gadget to take up the entire dashboard	Use the  button in the gadget header. ▾ Notes... This view often provides more functionality than is available in the standard view of the gadget. Only some gadgets provide the maximized or canvas view. The canvas view setting is stored in a cookie, and is not saved to the dashboard server.
Rearrange gadgets	Use the  button in the gadget header.
Customize the gadget frames Delete a gadget	Use the  button in the gadget header.

Custom gadgets

You need administrator privileges to add a gadget to the list of available gadgets. If you have permission to add and remove gadgets from the directory itself, you will see the '**Add Gadget to Directory**' and '**Remove**' buttons on the 'Add Gadget' screen. This functionality is only available for the Server version of applications; if you would like to add an Atlassian gadget to a directory in your Cloud site, please contact Atlassian Support.

Gadgets for Jira applications

Gadgets let you customize the information that appears on dashboards in Jira applications (or on your wallboards, if you use dashboards for that purpose). This page lists all of the gadgets available for Jira applications and which ones they're available for.

Gadget	Jira Core	Jira Software	Jira Service Desk	Use it to
Activity Stream				See the activity in your instance: it's like a Facebook feed for your instance!

Sprint Burndown Gadget				See the burndown for a given sprint in a handy line chart. Notes... <ul style="list-style-type: none">• The vertical axis represents your configured estimation statistic.• The gadget will only display sprints that have not been completed.
Sprint Health Gadget				Seeing a summary of the issues in a sprint in a handy color-coded bar graph. Notes... <ul style="list-style-type: none">• The colors in this gadget match the colors in your column configuration .

- The work completed is calculated based on the estimation statistic used for your board. This is reflected by the green part of the progress bar. For example, if you have 50 story points in a sprint and you have 3 issues with 10 story points that have been resolved, the 'Work complete' will be 20% (i.e. 10 out of 50 story points).
- The gadget won't reflect the progress from work logged in the 'Remaining Estimate' and 'Time Spent' fields in Jira.

- Adding or removing an issue from a sprint, after it has started is considered a change of scope. The percentage is calculated using the statistic that is configured for the board. For example, if you started a sprint with 50 story points and add an issue with 5 story points, the Sprint Health gadget would show a 10% scope change.
 - If you add/remove issues that don't have estimates, the scope change will not be altered.
 - If you're using Time Tracking, Scope Change will not be shown.
 - The "blocker" field counts all blockers that are in 'To Do' or 'In Progress'.
- See [Configuring estimation and tracking](#) for more information.

Version Report				Track the projected release date for a version. This helps you monitor whether the version will release on time, so you can take action if work is falling behind.
Agile Wallboard Gadget				Know how you're tracking with an agile board displayed on your wallboard (or dashboard).
Assigned to Me				Quickly see all the unresolved issues assigned to you.
Average Age Chart				<p>Want to know the average age of unresolved issues? This gadget tells you just that.</p> <p> Notes... <ul style="list-style-type: none"> The report is based on your choice of project or issue filter, and your chosen units of time (i.e. hours, days, weeks, months, quarters or years). For the purposes of this gadget, an issue is defined as unresolved if it has no value in the system resolution field. The age of an issue is the difference between the current date and the created date of the issue. </p>

<p>Average Number of Times in Status*</p> <p><i>*With installation of the Jira Charting App</i></p>	✓	✓	✓	<p>Displays the average number of times issues have been in a status.</p>
<p>Average Time in Status*</p> <p><i>*With installation of the Jira Charting App</i></p>	✓	✓	✓	<p>Displays the average number of days issues have spent in status.</p>
<p>Bamboo Charts</p>	✓	✓	✓	<p>Checking out Bamboo plan stats in your dashboard.</p> <p>▼ Notes...</p> <ul style="list-style-type: none"> Your Jira administrator must have configured the Bamboo plugin on your Jira server, if you want to add the Bamboo Charts gadget to your dashboard. If you have added multiple Bamboo servers in Jira there will be one Bamboo Charts gadget available per server.

				<ul style="list-style-type: none"> When you add this gadget to your Jira dashboard, you may see a message similar to this: <div style="border: 1px solid #ccc; padding: 5px; margin: 5px 0;"> <p>The webs</p> </div> <p>To fix this problem, you will need to configure your Bamboo site to allow Jira to draw information from it via gadgets on the Jira dashboard. To do this, your Jira administrator first needs to define your Jira site as an OAuth consumer in Bamboo. You will then be required to perform a once-off authentication before your gadget will display correctly.</p>
<p>Bamboo Plan Summary Chart</p>				<p>Seeing a graphical summary of a Bamboo build plan.</p> <p>▼ Notes...</p>

- Your Jira administrator must have configured the Bamboo plugin on your Jira server, if you want to add the Bamboo Charts gadget to your dashboard. If you have added multiple Bamboo servers in Jira there will be one Bamboo Charts gadget available per server.

				<ul style="list-style-type: none"> When you add this gadget to your Jira dashboard, you may see a message similar to this: <div style="border: 1px solid #ccc; padding: 5px; margin: 5px 0;"> <i>The webs</i> </div> <p>To fix this problem, you will need to configure your Bamboo site to allow Jira to draw information from it via gadgets on the Jira dashboard. To do this, your Jira administrator first needs to define your Jira site as an OAuth consumer in Bamboo. You will then be required to perform a one-off authentication before your gadget will display correctly.</p>
Bamboo Plans				<p>Seeing a list of all plans on a particular Bamboo server and each plan's current status.</p> <p>Notes...</p>

- Your Jira administrator must have configured the Bamboo plugin on your Jira server, if you want to add the Bamboo Charts gadget to your dashboard. If you have added multiple Bamboo servers in Jira there will be one Bamboo Charts gadget available per server.

				<ul style="list-style-type: none"> When you add this gadget to your Jira dashboard, you may see a message similar to this: <div style="border: 1px solid #ccc; padding: 5px; margin: 5px 0;"> <p><i>The webs</i></p> </div> <p>To fix this problem, you will need to configure your Bamboo site to allow Jira to draw information from it via gadgets on the Jira dashboard. To do this, your Jira administrator first needs to define your Jira site as an OAuth consumer in Bamboo. You will then be required to perform a one-off authentication before your gadget will display correctly.</p>
<p>Bubble Chart</p>				<p>Visually track the correlation of issues in a project or filter during a configured period, based on the following details:</p>

- number of days the issues have been open
 - number of comments the issues have
 - number of participants or votes the issues have
- ▼ [Notes...](#)
- The horizontal axis represents the number of days the issues have stayed open, while the vertical axis represents the number of comments the issues have.
 - The bubble colors also indicate the correlation between days open and number of comments – with the color green indicating low values and the color red indicating high values.
 - Only the first 200 matching open issues are displayed on the Bubble Chart.
 - You can configure the following settings for the Bubble Chart:

- The **period** during which the issue comments are considered recent
- The basis of the **bubble size**, either **participants** or **votes**
- **Automatic refresh** of Bubble Chart data every 15 minutes
- **Relative coloring** to distinguish issues receiving more comments from issues receiving fewer comments

				<ul style="list-style-type: none"> • Logarithmic scale (default is linear scale) to distribute the bubbles from each other accordingly. We recommend that you use the logarithmic scale if your Bubble Chart contains a large range of data.
Clover Coverage	✔	✔	✔	<p>Seeing the Clover coverage of plans from a particular Bamboo server.</p> <p>▼ Notes...</p> <ul style="list-style-type: none"> • Your Jira administrator must have configured the Bamboo plugin on your Jira server, if you want to add the Bamboo Charts gadget to your dashboard. If you have added multiple Bamboo servers in Jira there will be one Bamboo Charts gadget available per server.

- When you add this gadget to your Jira dashboard, you may see a message similar to this:

The webs

To fix this problem, you will need to configure your Bamboo site to allow Jira to draw information from it via gadgets on the Jira dashboard. To do this, your Jira administrator first needs to [define your Jira site as an OAuth consumer in Bamboo](#). You will then be required to perform a once-off authentication before your gadget will display correctly.

Created vs. Resolved Chart				<p>Checking your progress by seeing the number of issues created vs number of issues resolved over a given period of time.</p> <p>▼ Notes...</p> <ul style="list-style-type: none">• The chart is based on your choice of project or issue filter, and the chart can either be cumulative or not.• An issue is marked as resolved in a period if it has a resolution date in that period.• The resolution date is the last date that the Resolution field was set to any non-empty value.
-----------------------------------	---	---	---	--

Crucible Charts				<p>Seeing statistical summaries of your code reviews.</p> <p>▼ Notes...</p> <ul style="list-style-type: none"> Your Jira administrator must have configured the FishEye app on your Jira server, if you want to add the Crucible Charts gadget to your dashboard (not applicable to Jira Cloud).
Days Remaining in Sprint Gadget				Countdown! See how many working days you have before the current sprint ends.
Favorite Filters				See a list of all the issue filters that have currently been added by you as a favorite filter.
Filter Results				Seeing the results of a specified issue filter on the dashboard.
FishEye Charts				Chart LOC data from a FishEye repository.

FishEye Recent Changesets				<p>Get two charts about your repo in one: lines of code and commit activity.</p> <p> Notes... <ul style="list-style-type: none"> Your Jira administrator must have configured the FishEye app on your Jira server, if you want to add the Crucible Charts gadget to your dashboard (not applicable to Jira Cloud). </p>
Introduction				<p>Say hello to users with a configurable message on the dashboard.</p> <p> Notes... <ul style="list-style-type: none"> The text/html displayed in the introduction gadget is configured by your Jira administrator, through the Jira configuration page. </p>
Issue Statistics				<p>See the issues returned from a specified project or saved filter (grouped by a specified field).</p>
Issues In Progress				<p>Time to work! See all issues that are currently in progress and assigned to you.</p>

Jira Issues Calendar				<p>Generating a calendar-based view of due dates for issues and versions</p> <p> Notes... <ul style="list-style-type: none"> The Jira Calendar app is required for this gadget to be available. </p>
Jira Road Map				<p>See which versions are due for release in a given period, as well as a summary of the progress made towards completing the issues in the versions.</p>
Labels Gadget				<p>Use this gadget to see a list of all the labels used in a given project.</p>
Pie Chart				<p>See the issues returned from a specified project or issue filter, grouped by a specified field.</p>
Quick Links				<p>Link to frequently-used searches and operations.</p>
Recently Created Chart				<p>See the rate at which issues are being created, as well as how many of those created issues are resolved - all in a bar chart.</p>

<p>Resolution Time</p>				<p>Check trends in the average time taken to resolve issues.</p> <p>▼ Notes...</p> <ul style="list-style-type: none"> • The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years). • The 'Resolution Time' is the difference between an issue's Resolution Date and Created date. • If a Resolution Date is not set, the issue won't be counted in this gadget. • The Resolution Date is the last date that the system Resolution field was set to any non-empty value.
<p>Text</p>				<p>Display your specified HTML text on the dashboard.</p> <p>▼ Notes...</p>

- This gadget is only available if your Jira administrator has enabled it. It is disabled by default because it is a potential security risk, as it can contain arbitrary HTML which could potentially make your Jira system vulnerable to XSS attacks.
- To enable the text gadget:
Choose  **> Manage apps**. The 'Find apps' screen shows apps available via the [Atlassian Marketplace](#). Choose **Manage apps** to view the apps currently installed on your Jira site. Enable the **Text** module in the **Atlassian Jira - Plugins - Gadgets Plugin** (You need to select the System apps from the drop-down).

				<ul style="list-style-type: none"> If you cannot enable the text gadget, please contact Atlasian Support for assistance.
Test Sessions	✓	✓	✓	View a list of test sessions.
Time Since Chart	✓	✓	✓	<p>See a bar chart showing the number of issues for which your chosen field (e.g. 'Created', 'Updated', 'Due', 'Resolved', or a custom field) was set on a given date.</p> <p>▼ Notes...</p> <ul style="list-style-type: none"> 'Resolved' here is the system Resolution Date field, which is the last date that the system Resolution field was set to any non-empty value. The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years).
Time to First Response* <i>*With installation of the Jira Charting App</i>	✓	✓	✓	Displays the number of hours taken to respond to issues for a project or filter.

Two Dimensional Filter Statistics	✓	✓	✓	See data based on a specified issue filter (For example, you could create a filter to retrieve all open issues in a particular project. You can then configure the gadget to display the statistical data on this collection of issues, in a table with configurable axes.
Voted Issues	✓	✓	✓	See all the issues you've voted for.
Watched Issues	✓	✓	✓	Seeing all the issues you're watching.
Workload Pie Chart* <i>*With installation of the Jira Charting App</i>	✓	✓	✓	Displays the matching issues for a project or filter as a pie chart.

Set up a knowledge base for self-service

Follow this step by step guide for linking Jira Service Desk to a Confluence knowledge base, so that customers can help themselves and agents can share their expertise.

- 1. [Install Confluence](#)
- 2. [Get the right number of Confluence licenses](#)
- 3. [Link Jira Service Desk to Confluence](#)
- 4. [Learn about knowledge base settings and permissions](#)
- 5. [Link your project to a Confluence space](#)
- 6. [Write and search for knowledge base articles](#)



1. Install Confluence

User: **JIRA ADMINS**

You'll need **Confluence 5.10 and above** and **Jira Service Desk 3.1 and above** to access knowledge base features. Get the latest versions of each for all the bells and whistles.

Steps

1. [Install](#) the latest version of Confluence.
2. Have the **same user base** in both Jira and Confluence by one of the following methods:
 - [Connect to Crowd or Jira](#) for user management
 - [Connect to an internal Directory](#) with LDAP authentication
 - [Connect to an LDAP directory](#)
3. Check that you're an **administrator** in both Jira and Confluence.

Good to know

- Read [Use Jira applications and Confluence together](#) to check version compatibility between products.

Having the same user base in both Jira and Confluence means you won't have to manually create and maintain customer accounts.



2. Get the right number of Confluence licenses

User: **JIRA ADMINS**

Your **service desk agents will need a Confluence license** to create and edit knowledge base articles. However, service desk customers won't need a Confluence license to view articles.

Good to know

- Read [Confluence licensing and pricing](#) to learn more.



3. Link Jira Service Desk to Confluence

User: **JIRA ADMINS**

Linking two applications allows you to share information and access one application's functions and resources from within the other.

Steps

1. Go to **Jira Administration**



> **Applications > Application links.**

2. Enter the URL of your Confluence site and select **Create new link**.
3. Check **The servers have the same set of users**, to configure using OAuth (with impersonation) authentication.
4. If you're *not* an admin on both servers you won't be able to set up a 2-way (reciprocal) application link. If you want to go ahead and create a 1-way link anyway, clear the **I am an administrator on both instances** checkbox.
5. Use the wizard to finish configuring the link.

Good to know

- Read [Linking to another application](#) to learn how to set up a reciprocal link from Confluence.
- Read this [troubleshooting article](#) if you get into difficulty linking Jira to Confluence.



4. Learn about knowledge base settings and permissions

The permissions you set in Confluence determine whether (or not) your agents can create articles and if your customers can view them.

Read [Knowledge base settings and permissions](#) to decide what you need.



5. Link your project to a Confluence space

User: **PROJECT ADMINS**

When you link a Confluence space to your service desk project, agents can search for solutions and create new articles for common requests. Customers can then use the articles to self-service problems.

There are two ways to link your service desk to a Confluence Space:

Link to an existing space

1. Go to **Knowledge base**



> **Link existing space.**

2. Select the space you want from the dropdown.
3. Choose the link **Define who can view knowledge base articles** to set permissions.

You can also unlink or change spaces in **Project settings**



> **Knowledge base.**

Link to a new space

1. Check you have *Confluence admin* and *Create space* permissions.
 2. Go to **Knowledge base**
- 
- .
3. Select **Create new space.**
 4. Congratulations, you have created a new Confluence space.

Good to know

- Read [Use Confluence as a knowledge base](#) to learn how to set up a knowledge base blueprint.



6. Write and search for knowledge base articles

User: **SERVICE DESK AGENTS**

Once you've linked to a Confluence space, agents are ready to start writing and sharing knowledge with customers and teammates.

Read [Write and search for knowledge base articles](#) to learn how to do this.

Knowledge base settings and permissions

Now that you've linked your Jira Service Desk project to a Confluence space, you'll need to set your agents and customers up with the right permissions.

On this page:

- Who needs a Confluence license
- Choose who can view articles
- Check if agents can create articles
- Learn more about Confluence permissions

Who needs a Confluence license

If you want your service desk agents to create, comment on, and search the spaces on your Confluence site, they'll need a Confluence license. Your service desk customers however, don't need a Confluence license to view knowledge base articles.

To check if someone has a license:

1. Go to **Jira Administration**



> **User management.**

2. Select the user from the list.
3. If the **Confluence** box under **Application access** is checked, the user has a Confluence license.

Jennifer Evans

Email address	jevans-sd-demo@example.com
Username	jevans-sd-demo
Application access	<input checked="" type="checkbox"/> JIRA Service Desk <input type="checkbox"/> JIRA Software <input checked="" type="checkbox"/> JIRA Core <small>JIRA Core is included with other JIRA products</small> <input checked="" type="checkbox"/> Confluence

Choose who can view articles

Go to **Project settings**



> **Knowledge base > Access > Viewing.**

Access

Viewing

Define who can view knowledge base articles through the portal and in the linked Confluence space.

All active users and customers can access the knowledge base without a Confluence license.

Only licensed users who have access to the space

This setting determines who can view articles via the help center or a link your team shares. You have two options to choose from:

Option 1. All active users and customers

Users who don't have a Confluence license can view knowledge base articles via the customer portal and help center.

Choose this if:

- You want your team to write articles and share them with customers.

Limitations:

- If your service desk site and Confluence site have separate user bases, you'll need to create a Confluence user account for each Service Desk customer. If you don't want the customer to use a Confluence license, don't assign the Confluence user to a group.

Option 2. Only licensed users

Users who don't have a Confluence license can't read knowledge base articles unless you allow anonymous access in the knowledge base space.

Choose this if:

- You only want to use your knowledge base for internal articles.

Limitations:

- Anonymous access is not compatible with SSO using 2-legged OAuth.

Data privacy

If you choose the **All active users and customers setting**, you'll see the following message in the Confluence space permission screen:

Permissions Restricted Pages

 Any active user can view this space

Any active user can view pages in this space, including users who don't have a Confluence license. This was enabled through JIRA Service Desk so people can view knowledge articles when raising a help request.

[Edit this permission](#)

This permission overrides all existing space permissions. *Any logged in Confluence user* will be able to see the space (regardless of their group membership).

You can disable this permission at any time, but it can only be re-enabled from Jira Service Desk.

Login process

If you aren't using SSO or something that passes login information between instances, then users will need to log in to Confluence using their *Jira Service Desk credentials* to view articles.

Check if agents can create articles

Your agents will need a Confluence license to create and edit articles, and permission to create articles in the Confluence space.

To check if an agent can create articles:

1. Go to **Project settings**



> **Knowledge base** > **Access** > **Authoring**.

2. Select the **space permissions** link, this will take you to your space permission's page.
3. Check that the agent (or a group they are a member of) has the **Add page** permission.

Learn more about Confluence permissions

There are three levels of permissions in Confluence that determine who can view and create articles:

- **Global permissions** are **site-wide** and can be assigned by a system or Confluence admin.
- **Space permissions** are **space specific** and are managed by the space admin.
- **Page restrictions** affect both **viewing and editing** and are managed by space admin.

If you, your agents or your customers are having difficulty viewing articles, check the permissions with your Confluence admin.

Write and search for knowledge base articles

Your knowledge base is there for customers, even when you're working on other requests, at home for the day, or away on holiday.

Here are some reasons you might write knowledge base articles:

- **You get a lot of similar requests.** Like "How do I access office wifi?". When customers search for 'wifi' in the help center, they'll find your article. If they send you a request, you can share the article rather than walking them through the steps in a comment.
- **You're upgrading a system.** You can write a step by step upgrade guide and link to it from an announcement in the help center.
- **You're troubleshooting the same issue with customers.** If you normally walk people through a series of steps to diagnose and troubleshoot a problem, such as a broken printer, a troubleshooting guide will save you time.

On this page:

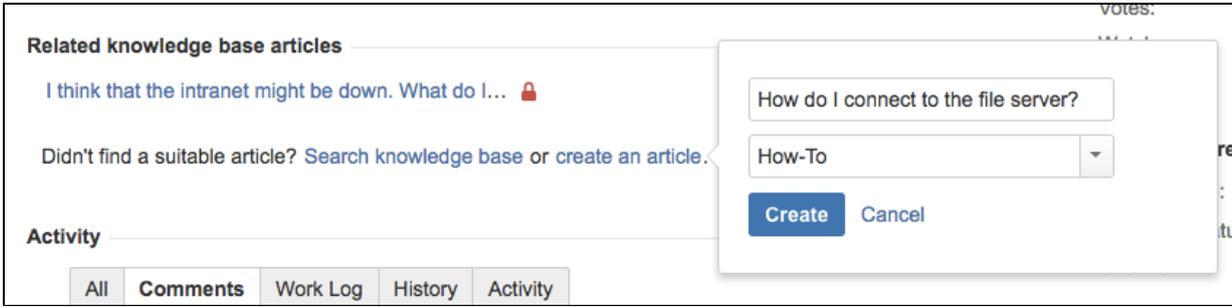
- [How to write articles](#)
- [How to share articles](#)
- [How the search works](#)

How to write articles

After you set up your knowledge base space, the issue view displays a panel called **Related knowledge base articles**:

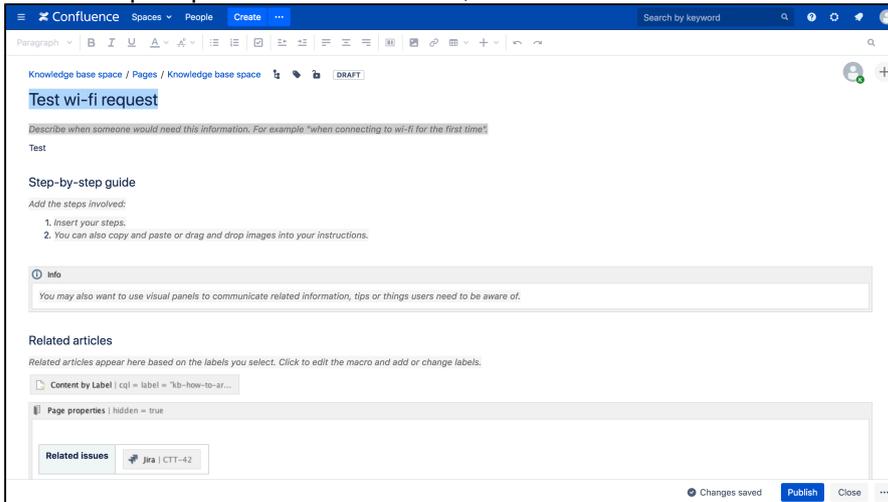
.

If you don't see a relevant article, you can create a new one from the issue. You'll need the **Add page** permission in the Confluence space. Customers can read any article that doesn't have a red padlock.



To create a new article:

1. Click the **create an article** link.
2. By default, the issue summary becomes the new article's title, you can edit this.
3. Choose either the **How-To** or **Troubleshooting** template.
4. Follow the prompts to create the article, then **Publish**.



5. Check how it displays in the issue view.

Page labels are essential in knowledge base spaces. These are used to add topics to your articles, and allows your knowledge base to become self-organizing over time.

You can also create articles from the sidebar, then **Knowledge base**



> **Create article.**

How to share articles

There are two ways you can share articles with customers:

- Hover over an article to share it as a comment on a request:



- Share links from a knowledge base article that you access via **Knowledge base**



in the sidebar.

When customers click the link from a request or their email, the article opens in the help center:

How the search works

The knowledge base recommends articles to customers and agents using keywords it pulls from the **Summary** field of a request.

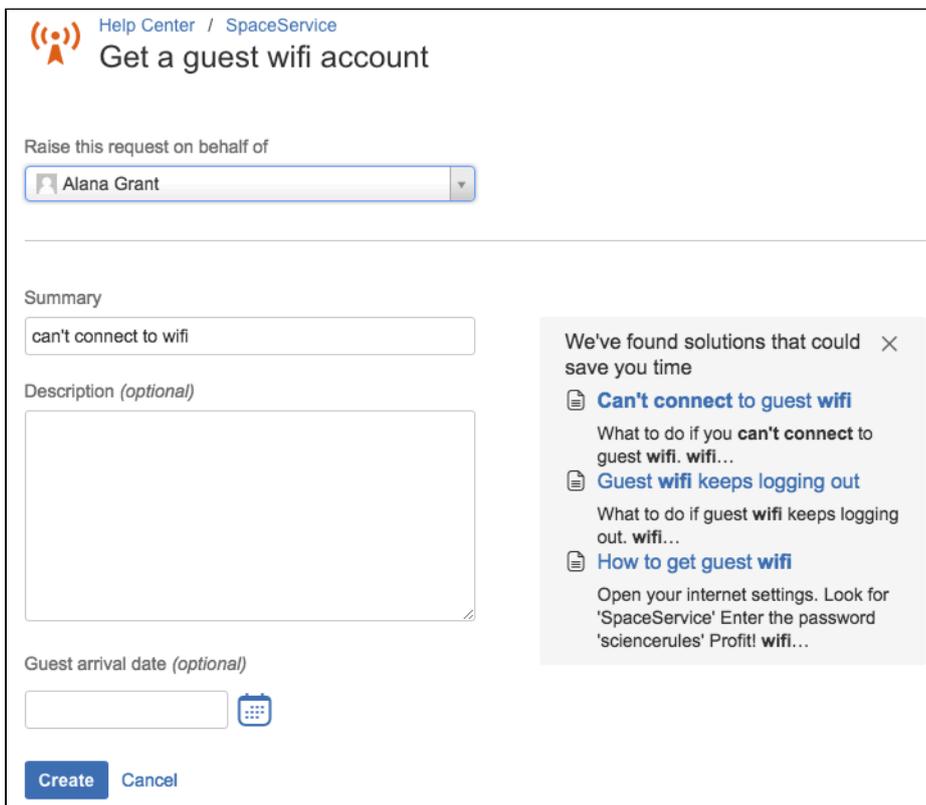
Help Center and Customer Portal search results

When a customer types something in the search field on the Help Center or Customer Portal, the results return the following:

- Related knowledge base articles. These help customers resolve their own issues, without needing to raise a request.
- Related request types. These are recommended to customers, so they know which request type to fill out based on their issue.

Example results when a customer types "Product" in the Help Center search field

To make it even easier for customers to find what they need, you can recommend articles when they fill out a particular request type.



*Recommended articles for the **Get a guest wifi account** request type*

In the above example, the request type is restricted to the label 'wifi'. When the customer starts typing words in the **Summary** field, articles in the knowledge base that have the label 'wifi' are recommended.

To recommend articles for certain request types, go to **Project settings > Knowledge base > Auto-search on request types**, then in the **Search KB** column, choose **Yes**:

Auto-search on request forms

Knowledge base articles can be automatically presented to customers as they fill in a request form. This automatic search can be restricted by label.

Request form	Search KB	Restrict to articles with labels
Technical support	Yes	Technical_support

You can restrict the search to articles with a certain label. To make life easier for yourself, use a label similar to the name of the request form.

You can't use labels as a way of restricting access to an article. For example, if you create a knowledge base article and add the label "display", and then create another article and don't add the label "display" – the page without the label will still display in the search. Adding labels are a way to filter related articles, not assign viewing permissions.

If you need to restrict an article so that only certain users can view it, do this through [knowledge base settings and permissions](#).

Important things to note:

- The knowledge base search and the request type search are independent of each other.
- The knowledge base search will display a maximum of 3 results.
- The request type search will display a maximum of 5 results, and:
 - The primary search picks up words that the customer is typing in the **Summary** field, and looks for them in the request type's **Name** and **Description**.
 - The secondary search uses JQL queries to find relevant request types based on the request's **Summary** field.

Issue view search results

When agents view a request, the knowledge base recommends related articles that they can reference or share with customers. Your service desk will search the Confluence space you have linked for any articles that contain keywords in the request's **Summary** field.

Important things to note:

- The knowledge base search will display a maximum of 3 results.
- The search ignores all other fields from the request (only the summary field is used).
- Check that your request types have a visible **Summary** field. Read [Setting up request types](#).
- The search skips articles that the agent doesn't have permission to view.
- The search and ranking follows the same rules as the [Confluence search](#). In short, Confluence identifies all articles that contain keywords matching those in the request's **Summary** field, and sorts them by word frequency.
- Agents can only share articles that the customer has permission to see.

Change the article title to include more keywords from the request summary field, if you don't see an article that you think you should.

Agents can also search for articles via the **Knowledge base** in the project sidebar.

Using the Help Center

You can direct customers to your Help Center, so they don't have to remember whether they need to submit a request for a new laptop in the IT Service Desk. Simply searching for "new laptop" in the Help Center will display the correct request type automatically.

Customers can use the Help Center to:

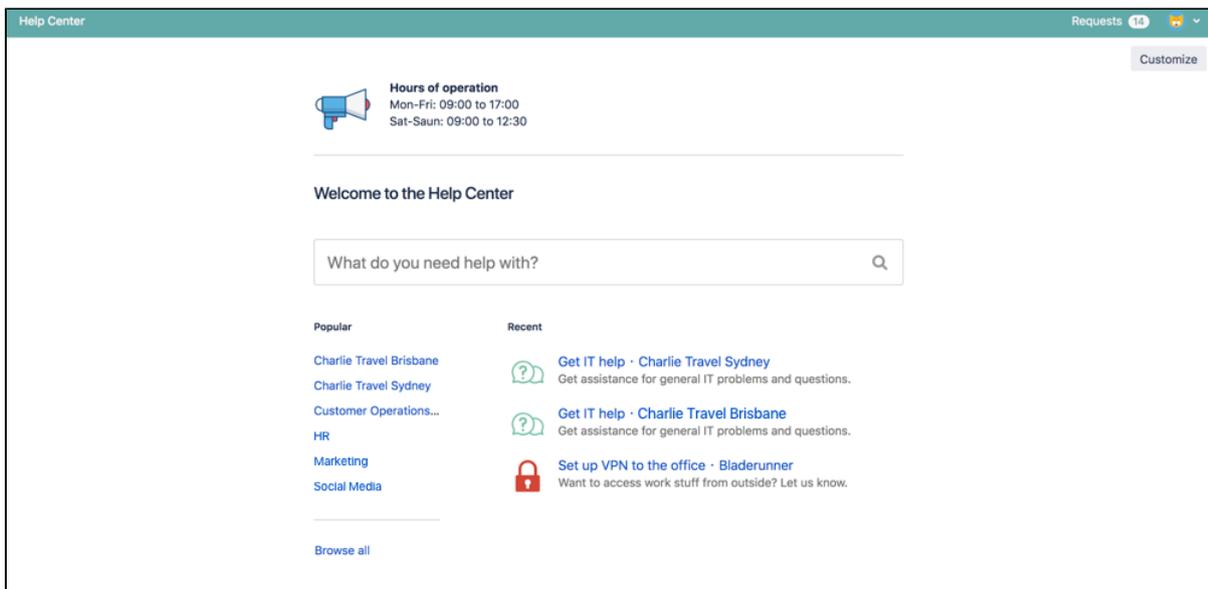
- View popular service desks
- Browse an individual service desk
- Search for request types and knowledge base articles
- See requests they have raised across all service desks

On this page:

- Branding the Help Center
- Sharing the Help Center with customers
- Searching across multiple customer portals
- How smart search works

To help your customers get the most out your Help Center, we recommend branding your Help Center and integrating your service desk projects with a knowledge base.

Here's a quick look at the Help Center layout:



Customers will only be able to see popular service desks and search across service desks they already have access to. The list of popular service desks is generated automatically based on the number of requests raised and cannot be set manually. Recent request types displayed are unique to each customer. Note that customers who have not yet raised requests will not see any recent request types in the Help Center.

Learn more about managing customer access to service desk projects [here](#).

Branding the Help Center

Jira administrators can brand the global center with your company logo and color scheme. If you are logged in as an administrator, go to



> **Applications > Jira Service Desk Configuration** to customize your Help Center with the help of a live preview. Your changes will be applied to the Help Center and to the header of all customer portals. For more information about managing a project-specific customer portal, check out [Configuring the customer portal](#).

Sharing the Help Center with customers

You can provide your customers with the following Help Center URL, so they don't have to remember the URL for each customer portal they have access to:

`http://<computer_name_or_IP_address>:<HTTP_port_number>/Jira/servicedesk/customer/portals`

Searching across multiple customer portals

Customers can search for request types and knowledge articles in a specific customer portal or use the global Help Center search to find request types and knowledge articles in portals they have access to:



Customers can read articles directly in the Help Center and provide feedback to your team by marking articles as helpful or not helpful. If customers still need to contact your team after looking through related knowledge articles, they can choose one of the suggested request types or browse all customer portals they have access to.

How smart search works

The improved smart search algorithm learns from past searches and request types raised, so if customers have previously raised hardware requests for a laptop and monitor, they can search for "laptop" or "monitor" in the future to find the same hardware request type.

Your team can also help improve search results by updating the request type field when a customer has, for example, searched for "new laptop" and raised an software request instead of the needed hardware request.

The Help Center smart search has been built to be language-agnostic and can therefore learn from search words or phrases entered in any language. As customers enter more searches and raise more requests, the search algorithm gets smarter regardless of the language used.

Collecting customer satisfaction (CSAT) feedback

Measuring customer satisfaction can help you better understand your customers and improve service levels.

Jira Service Desk provides a simple, built-in mechanism to collect customer feedback. Key features include:

- Simple customer workflow to provide feedback on resolved issues
- Customer satisfaction scores are visible within resolved issues and on agent queues for resolved issues
- Single-click to view customer satisfaction report for a service desk project
- Easily create and view custom reports and trend graphs based on satisfaction scores.

Customer feedback data can be used to identify strengths and weaknesses in the service quality, engage and motivate the team to improve satisfaction scores, and provide mentoring and training where required.

Enabling the customer satisfaction feature

This feature is enabled by default for new service desk projects; however, it must be enabled for each existing service desk project. To enable customer satisfaction settings for an existing project:

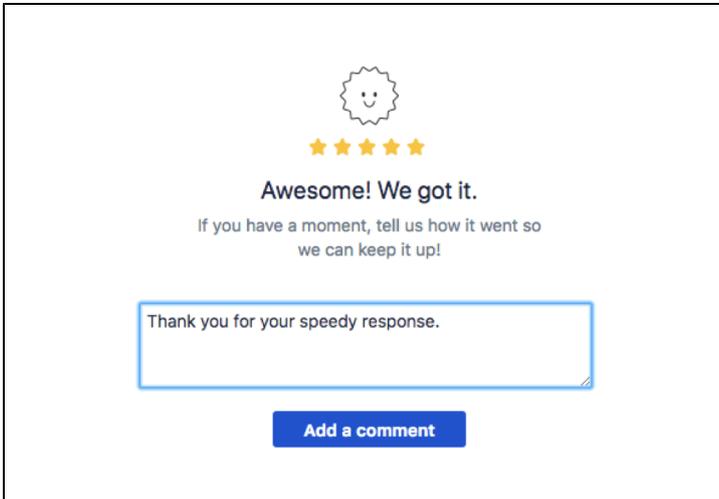
1. Log in as a service desk project administrator.
2. Select the service desk project you wish to configure.

On this page:

- [Enabling the customer satisfaction feature](#)
- [Viewing and reporting on customer feedback](#)

3. Select **Project settings > Satisfaction settings** .
4. Enable **Collect customer satisfaction feedback**.
5. Optionally, edit the **Question** phrase to suit your service desk environment. This phrase appears in the resolved issue notification message that customers see.

When you enable satisfaction settings for a service desk project, resolved issue notifications will contain a satisfaction rating scale. Customers can click the rating scale to indicate their level of satisfaction. A confirmation page is displayed on the customer portal, where they can change the rating, and optionally provide any additional comments that they would like to convey to the team.



Viewing and reporting on customer feedback

Customer satisfaction scores and comments are displayed in the issue view for resolved issues. Agents can also view the satisfaction scores on their own recently resolved queues.

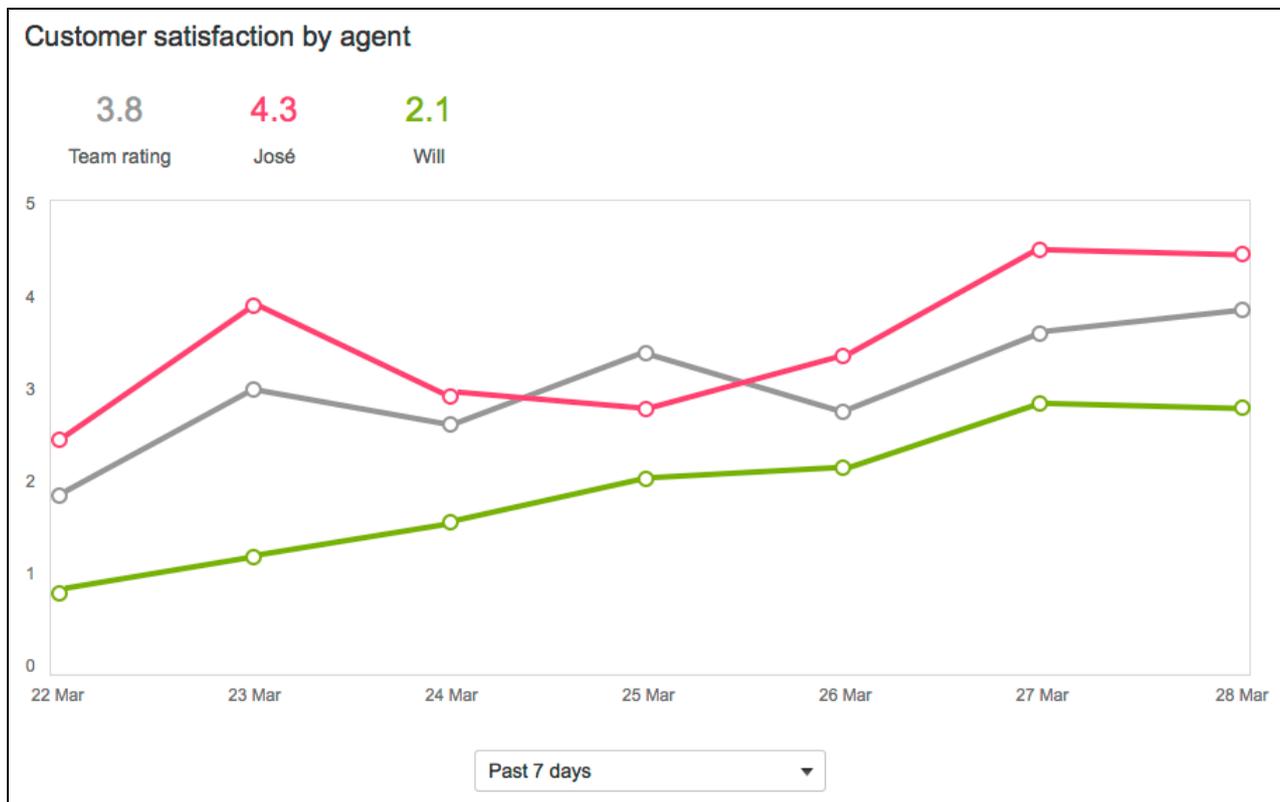
Service desk project administrators and agents can view the default Satisfaction report, which displays the average customer satisfaction scores for the team.

Customer satisfaction				
2.6 Average rating				
Rating	Comment	Key	Agent	Received
★★★★★	Thanks for coming down to my desk to help me print in A3	IT-1384	Will Turner	Yesterday
★★★★☆		IT-997	José Vela	2 days ago
★☆☆☆☆	I had to wait 3 days for a solution. I replied on Wednesday but nobody got back to me until this morning.	IT-1216	Mia Kennedy	3 days ago

1-3 of 3

Past 7 days

A service desk project administrator can also create and view [custom reports](#) analyzing satisfaction trends. Agents can also view any custom satisfaction reports created for their service desk projects.

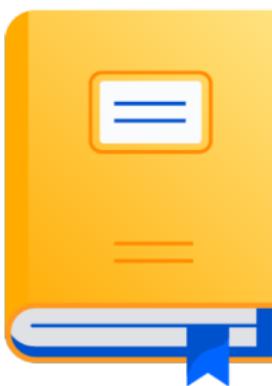


Useful examples of custom reports include:

- A trend graph of the average satisfaction rating for a specific period to view changes in service levels.
- Satisfaction scores based on the type of service request. This would identify issues for which the team could provide knowledge articles.
- Satisfaction scores for an individual agent compared to team scores to help identify agents who would benefit from further training.

Jira Service Desk best practices

Check out the following best practice articles:



- Best practices for designing the customer portal
- Best practices for IT teams using Jira Service Desk
- Best practices for software teams using Jira Service Desk

Best practices for designing the customer portal

Every service desk project comes with a preconfigured customer portal that your customers use to interact with your service team. Here are some best practices on how to design an easy-to-use customer portal that will help both your team and your customers work more efficiently.

On this page:

- Brand your Help Center
- Brand your customer portals
- Help customers find the request types they need
- Group related request types
- Set up a knowledge base

Brand your Help Center

Customers can use the global Help Center to search for request types and knowledge base articles across all customer portals they have access to.

Brand your Help Center by:

- Uploading your company logo and your service desk can automatically generate a matching theme for your customer portals and global Help Center header.
- Naming your customer portals and global Help Center, so your customers can easily identify your team's service desk.

The screenshot shows the Jira Service Desk Help Center interface. The main content area displays a search bar with the placeholder text "What do you need help with?". Below the search bar, there are two columns: "Popular" and "Recent". The "Popular" column lists categories like HR, Payroll, Finance, Office Admin, Marketing, and Social. The "Recent" column shows two articles: "Get IT help · Charlie Travel Sydney" and "Set up VPN to the office · Charlie Travel Melbourne".

The "Customize" sidebar on the right contains the following settings:

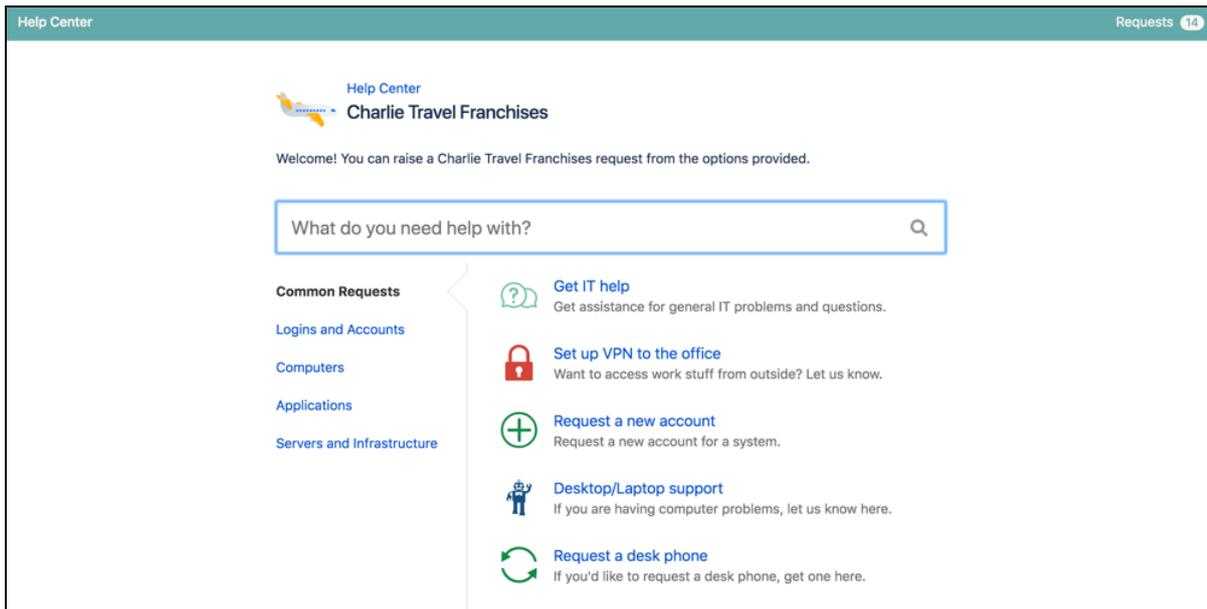
- Front page title:** Welcome to the Help Center
- Help center name:** Help Center (Also included in breadcrumbs)
- Announcement subject:** eg. vacation hours, service outage
- Announcement message:** (Empty text area)
- Links:** [name|http://example.com]
- Use a logo
 - Choose logo
 - Generate theme from my logo
- Header bar:**
 - Background
 - Text
 - Highlights
 - Text on highlights

At the bottom of the sidebar, there are "Save changes" and "Cancel" buttons.

Brand your customer portals

For each project's customer portal, you can customize the name, welcome message, and logo to let

customers know which portal to use to contact a specific team in your organization. Note that the Help Center name and header appear across all your project's customer portals.



Check out [Configuring the customer portal](#) to learn more.

Help customers find the request types they need

- Name request types in language that's familiar to your customers, and use keywords they'll recognize. For example, name a request 'Access to a system' instead of 'VPN access'.
- Use different icons for different request types, so customers can easily identify request types in the customer portal.
- Add contextual help (for example, specify photo dimensions and format for the attachment field) with the **Field help** field.
- Use examples in your request type descriptions (for example, 'If you need a software license such as Microsoft Office, raise a request here').
- Link to existing information that might be helpful for customers in the request type description. For example, if you have already have a list of available Microsoft Office license numbers on your Intranet, simply add a link to the page in the request type description and instruct customers to claim a license from that page without needing to open a request.

See [Setting up request types](#) for more information on naming request types.

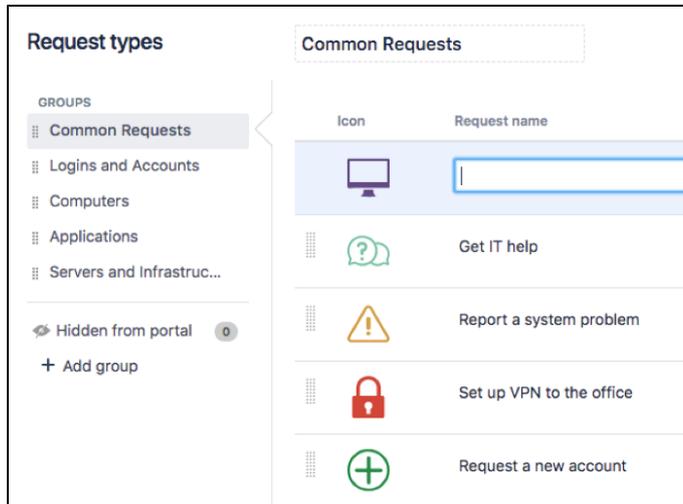
Group related request types

If you have a large number of request types, say more than 7, we recommend grouping some of them together to help customers find what they need. Grouped request types appear as tabs in your customer portal.

To add groups, select **+Add group** from the sidebar. While viewing a group, select **Add existing request type** to add your request types to it.

If you don't want your customers to use a certain request type, you can remove the request type from all groups, or move it into the hidden from portal group.

Agents can use hidden request types to organize work internally. Note that removing a request type won't affect requests that have already been created.

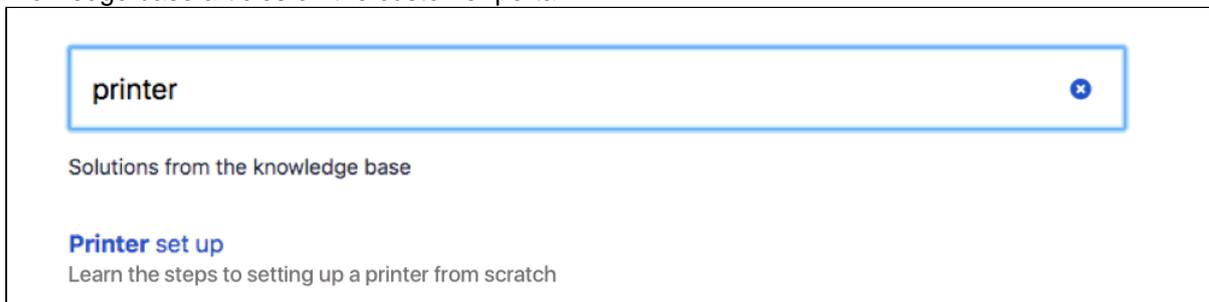


Groups appear as tabs in your customer portal. You can control the order in which groups appear by dragging and dropping them in **Project settings > Request types**.

See [Setting up request types](#) for more information on managing request types and groups.

Set up a knowledge base

- After using your service desk for a while, your team will probably have accumulated a large amount of information that can be provided to your customers so that they can solve some problems before even opening requests. At this point, you can consider integrating Confluence's knowledge base capabilities with Jira Service Desk.
- Connect your service desk project to a Confluence space so customers can search for relevant knowledge base articles on the customer portal:



For information about how to achieve this, see [Set up a knowledge base for self-service](#).

Best practices for IT teams using Jira Service Desk

Information Technology Infrastructure Library (ITIL) is a framework for ITSM (IT service management). ITIL recommendations have been an industry standard for 20 years. Adopting these practices takes time. You can seek formal training on how to make ITIL recommendations work best for your business.

If you're interested in more general information about ITSM or other guides, [check out our ITSM resources](#).

We recommend starting with processes that are essential to your business goals. Jira Service Desk provides workflows for IT teams in our IT Service Desk project template. We created the following ITIL workflow add-ons, available in the Atlassian Marketplace:

- [Change management for Jira Service Desk](#)
- [Jira Service Desk incident management](#)
- [Problem management for Jira Service Desk](#)
- [Service requests for Jira Service Desk](#)

Jira Service Desk is configurable. We recommend you start with an ITIL framework and then adjust to your specific business needs.

What follows is an overview of some best practices for your IT service desk. This guide covers:

- [Fulfilling service requests with your IT service desk](#)
- [Managing changes with your IT service desk](#)
- [Managing incidents with your IT service desk](#)
- [Managing problems with your IT service desk](#)
- [Calculating priority automatically](#)

On this page

- [Use the IT Service Desk template](#)
- [Provide a robust service catalog](#)
- [Help your customers serve themselves with a knowledge base](#)
- [Start with ITIL recommended workflows and adapt to your needs](#)

Use the IT Service Desk template

Only Jira administrators can create projects.

To create a project using the IT Service Desk template:

1. Select **Projects > Create Project**.
2. Choose the **IT Service Desk** template and select **Next**.
3. Name your project.
4. Select **Submit**.

Provide a robust service catalog

Using simple forms, your customers can do a lot of the early leg work for your service desk. Jira Service Desk provides them with clear and concise options for requesting help. A robust service catalog makes sure that service requests are prioritized and get to the correct service agent. And, your agents get the information needed to fulfill the request before beginning work.

The IT Service Desk template comes with a service catalog of common requests. We call these forms "request types" and you can customize them to suit your needs.

To start, identify the most common and urgent IT tasks for your service desk. Doublecheck that there's a corresponding request type for each of these.

Go to **Project settings > Request types** to view or edit your service catalog. The entries there appear in your customer portal. [Read more about request types](#).

Help your customers serve themselves with a knowledge base

Mature IT service desks solve common problems without ever seeing a ticket. We recommend providing your customers a knowledge base.

Link your Jira Service Desk site to a Confluence knowledge base. Keep a record of known solutions and two important things happen:

1. Your customers find and view relevant articles when they search the customer portal. They may find the answers they need without ever raising a request.
2. Your service agents can find relevant articles when working on issues. This saves them time hunting down answers or workarounds to common IT requests.

Learn more at [Set up a knowledge base for self-service](#).

Start with ITIL recommended workflows and adapt to your needs

ITIL recommendations are a framework, a set of ITSM best practices meant for you to adapt from and grow.

In Jira Service Desk, we associate your service catalog with workflows by assigning a request type to an underlying *issue type*. Our recommendations for IT teams use four ITIL-inspired workflows. To learn more about how these issue types and workflows work to streamline your service desk, check out our guides for handling:

- [Fulfilling service requests with your IT service desk](#)
- [Managing changes with your IT service desk](#)
- [Managing incidents with your IT service desk](#)
- [Managing problems with your IT service desk](#)
- [Calculating priority automatically](#)

These allow you to have many customer request forms that follow the same workflow. For example, new hardware requests and password resets use the same service request workflow.

Our default workflows make your service desk effective, out of the box. You can customize them as you go, scaling to the needs of your business.

Fulfilling service requests with your IT service desk

The scope of service requests in Information Technology Infrastructure Library (ITIL) is large. Tasks can range from resetting a password to onboarding a new hire. Service requests include customer comments, complaints, or other requests for information.

The IT Service Desk template comes with a few pre-built service request types. We set these up to help your service desk agents handle common service requests.

The service request fulfillment process:

- manages customer expectations
- speeds up request resolution
- standardizes any approval processes

Effective service request management reduces the bureaucracy and cost of maintaining IT services.

This page describes some best practices for fulfilling service requests using Jira Service Desk. You may seek formal training in how to make ITIL recommendation work best for your business.

On this page

- [Service request fulfillment process](#)

- Setup for service request fulfilment in Jira Service Desk
 - Configure the workflow and fields with the service request workflow add-on
 - Service request fulfillment workflow
- Tips for creating service request forms on your portal
- Default form fields for service requests
 - Extra form fields recommended by ITIL

- Learn more about IT service management (ITSM)

Service request fulfillment process

The information needed to capture and resolve service requests varies. But, you can standardize the process for fulfilling these requests.

The IT Service Desk template associates certain requests with a service request fulfillment workflow. We set up the workflow to complement the following service request fulfillment process. Use it as a jumping off point for your service desk.

The service request fulfillment process, in brief:

1. A customer requests help from your service catalog or via email.
2. The service desk assesses the request alongside pre-defined approval and qualification processes. If needed, they send the request for financial or business approval.
3. A service desk agent works to fulfill the service request, or forwards the request to someone who can.
4. After resolving the request, the service desk closes the ticket. The agent consults the customer to make sure they are satisfied.

Setup for service request fulfillment in Jira Service Desk

Configure the workflow and fields with the service request workflow add-on

We used the ITIL framework for change management to build a workflow for Jira Service Desk: <https://marketplace.atlassian.com/plugins/com.atlassian.servicedesk.servicerequest/server/overview>.

You can use this add-on as a template to help you build your own service request fulfillment process.

To use the workflow from the Marketplace:

1. Log in as a user that has the Jira administrator global permission, and follow the instructions listed here to [import a workflow](#).
2. To add the workflow fields to your change issues, activate the screen by following the instructions here: [Defining a screen](#).

Service request fulfillment workflow



Tips for creating service request forms on your portal

- Begin with the most common requests. Choose ones that are simple and easy to fulfill. This delivers immediate value to customers. It allows the service desk team to learn as they build out future phases of the service catalog.
- Document all the requirements for a service request before adding it to the catalog. These include question data, the approval process, fulfilment procedures, the fulfilment team, process owners, SLAs, reports, and so on. This allows the IT team to better manage the request type over time.
- Capture the data needed to start fulfilling the request. But, don't overload the customer with too many questions.
- Work with stakeholders to standardize the approval process, where possible. For example, pre-approve all requests for new monitors. Or, assign software approvals to the customer's manager.
- Document any knowledge base information that might allow customers to service their own request. Record this in a linked Confluence space. If you do, customers can view articles while they search your portal. [Read more about creating a knowledge base.](#)
- Review your team's performance in fulfilling requests. Adjust your SLAs, requirements, and training to improve customer satisfaction.
- Create reports to help manage the lifecycle of a service request offering. These trends can uncover forms that are no longer needed, too complex, or insufficient.

[Read more about creating reports.](#)

Default form fields for service requests

Jira Service Desk allows you to customize the fields of information collected from customers. Additionally, you can customize the fields of information used by your agents. Jira Service Desk does this through issue type fields and screens. Fields help agents fulfill the request, discuss with vendors, and categorize requests.

By default, we include the following fields in your agents' view of a service request:

Field name	Description
Summary	A short description of the request.
Reporter	The person who submitted the request.
Component/s	Segments of your IT infrastructure that relate to the request. For example, "Billing services" or "VPN server". These are used for labeling, categorization, and reporting.
Attachment	Files or images added to the request.
Description	A long, detailed description of the request.
Linked Issues	A list of other requests that affect or are effected by the request. If your business uses other Atlassian products, this list may include linked development issues.
Assignee	The service desk agent assigned to fulfill the request.
Priority	The importance of the request's resolution, usually in regards to your business needs and goals. Sometimes, priority is calculated by impact and urgency.
Labels	A list of additional custom labels used for categorizing or querying records.
Request participants	A list of extra customers or vendors who take part in resolving the request.
Approvers	A list of business or financial contacts responsible for approving the service request.
Organizations	A list of customer or vendor groups interested in the request's resolution.

Extra form fields recommended by ITIL

ITIL recommends a few more fields for their in-depth processes. The IT Service Desk template doesn't include these by default. This is because IT teams who use Jira Service Desk don't often use these fields. If needed, you can include these fields or add custom fields. [Find out more about fields in Jira.](#)

ITIL also recommends including the following fields:

Field name	Description
Impact	The effect of the service request, usually in regards to service level agreements.
Urgency	The time available before the business feels the service request's impact.
Pending reason	A short description or code that indicates why the service request is not progressing.
Product categorization	A category of IT asset or system that the request effects.
Operational categorization	A category of action or function required to fulfill the request.

Learn more about IT service management (ITSM)

Get more tips and tricks for successful [ITSM](#), view case studies, and learn how to take your service desk to the next level.

[Check out the ITSM resources on IT Unplugged.](#)

Managing changes with your IT service desk

Effective service desks plan and control changes, and they understand their impact to their business. An Information Technology Infrastructure Library (ITIL) change management workflows aims to make your change efforts successful.

The IT Service Desk template comes with a change management workflow. This workflow ensures you record, assess, approve, and implement change requests. We recommend you start with our default workflow and adapt it to your business needs.

If done well, a change management process:

- stabilizes your IT services
- makes IT services reliable and predictable
- adapts IT services to evolving business needs

You can lessen risk, outages, and defects. And, you can prevent duplicating efforts from failed changes.

This page describes some best practices for managing changes using Jira Service Desk. You may seek formal training in how to make ITIL recommendation work best for your business.

On this page

- [Change management process](#)
- [Setup for change management in Jira Service Desk](#)
 - [Configure the workflow and fields with the Change Management workflow add-on](#)
 - [Change management workflow](#)
- [Coordinate changes with a calendar](#)
- [Default form fields for change requests](#)
- [Learn more about IT service management \(ITSM\)](#)

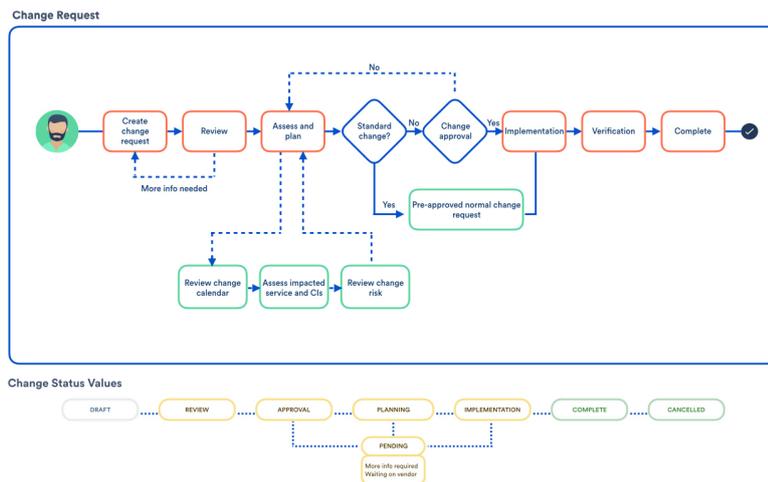
[Change management process](#)

The IT Service Desk template connects certain requests to a change workflow. We set up the workflow to complement the following change management process. Use the workflow to transition change request records alongside these ITIL recommended activities:

- reviews
- planning
- approvals
- implementations

We recommend you start with this workflow and adapt it to your needs over time.

The ITIL change management process, in brief:



1. An internal IT member requests a change. They note details like the affected systems, possible risks, and expected implementation.
2. The change manager or peers determine if the change will be successful. They may ask for more information in this step.
3. After review, the team plans how to put the change in place. They record details about:
 - the expected outcomes
 - resources
 - timeline
 - testing
 - ways to roll back the change
4. Depending on the type of change and risk, a change approval board (CAB) may need to review the plan.
5. The team works to implement the change, documenting their procedures and results.
6. The change manager reviews and closes the implemented change. They note whether it was successful, timely, accurately estimated, within budget, and other details.

Setup for change management in Jira Service Desk

Configure the workflow and fields with the Change Management workflow add-on

We used the ITIL framework for change management to build a workflow for Jira Service Desk: <https://marketplace.atlassian.com/plugins/com.atlassian.servicedesk.change/server/overview>.

You can use this add-on as a template to help you build your own change management process.

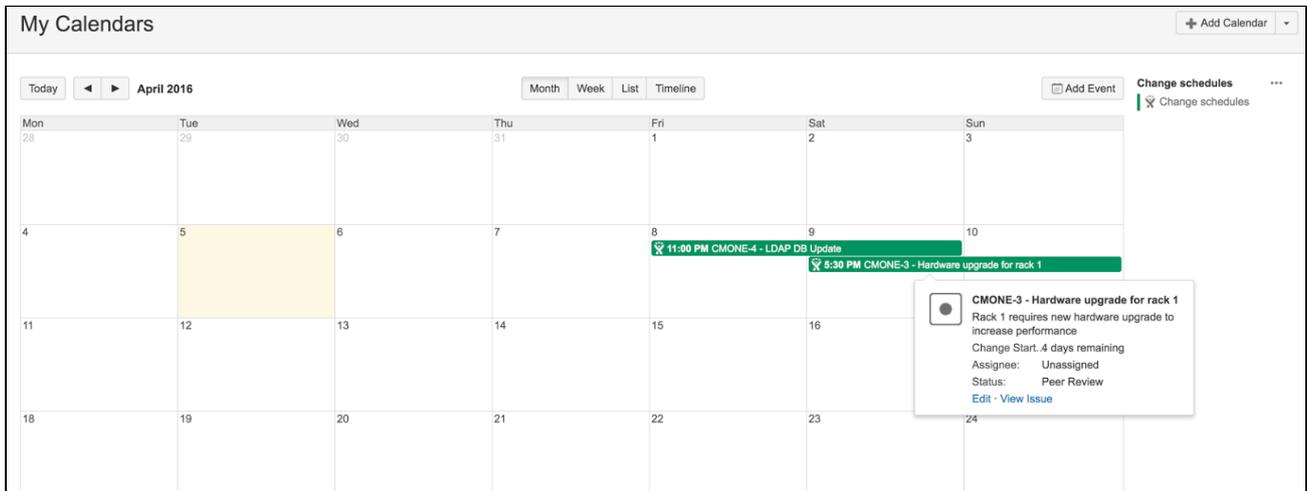
To use the workflow from the Marketplace:

1. Log in as a user that has the Jira administrator global permission, and follow the instructions listed here to [import a workflow](#).
2. To add the workflow fields to your change issues, activate the screen by following the instructions here: [Defining a screen](#).

Change management workflow

Coordinate changes with a calendar

With Confluence, you can schedule changes. Use Team Calendars to track changes, and schedule change to coincide with business events. [Try Team Calendars for Confluence for free.](#)

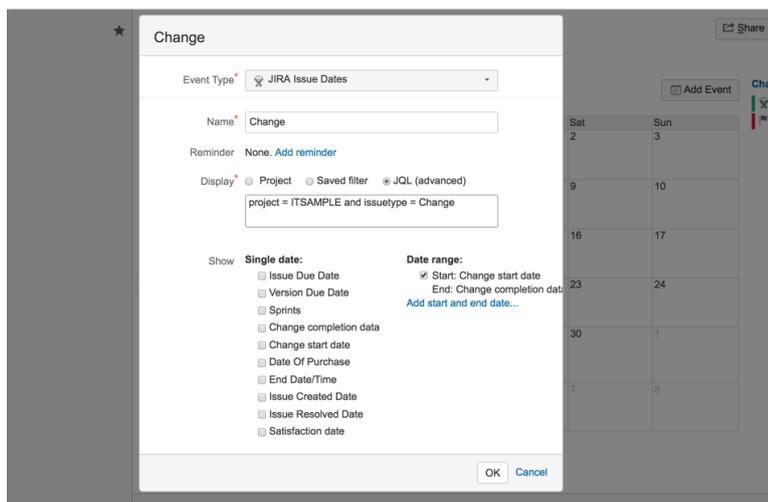


To set up a change calendar with Team Calendars for Confluence:

1. In Confluence, go to your team's space.
2. From the sidebar, select **Calendars**.
3. Select **Add calendar**.
4. Viewing the calendar, select **Add event**.
5. Select the **Event Type** drop down and choose **Jira Issue Dates**.
6. Under **Display**, select the **JQL (advanced)** and enter the following:

```
project = "Your IT service desk project name" AND issuetype= Change
```

7. Under **Date range**, select **Add start and end date...** Select **Change start date** as the start date. Select **Change completion date** as the end date.
8. Select **OK** .



The calendar automatically picks up the start and end dates of change requests from your service desk . Then, it plots them on the calendar.

Default form fields for change requests

Jira Service Desk allows you to customize the fields of information collected from customers. Additionally, you can customize the fields of information used by your agents. Jira Service Desk does this through issue type fields and screens. Fields help agents assess, approve, and categorize the request for reporting or querying.

By default, we include the following fields in your agent's view of a change request. If needed, you can add custom fields. [Find out more about fields in Jira.](#)

Field name	Description
Summary	A short description of the request.
Reporter	The person who submitted the request.
Component/s	Segments of your IT infrastructure that relate to the request. For example, "Billing services" or "VPN server". These are used for labeling, categorization, and reporting.
Attachment	Files or images added to the request.
Description	A long, detailed description of the request.
Linked Issues	A list of other requests that affect or are effected by the request. If your business uses other Atlassian products, this list may include linked development issues.
Assignee	The team member assigned to work on the request.
Priority	The importance of the request's resolution, usually in regards to your business needs and goals. Sometimes, priority is calculated by impact and urgency.
Labels	A list of extra custom labels used for categorizing or querying records.
Request participants	A list of extra customers who take part in the request, for example, people from other teams, or vendors. Read more about participants.
Approvers	A list of people responsible for approving the request, usually business, financial or technical contacts .
Organizations	A list of customer groups interested in the request's resolution.
Impact	The effect of the change, usually in regards to service level agreements.
Urgency	The time available before the business feels the request's impact.
Change type	The category of the change (usually <i>standard</i> , <i>normal</i> , or <i>emergency</i>). For example, a standard change does not require action from change managers. A normal change does.
Change reason	A short description or code that indicates why the reporter needs the change.
Change risk	The risk of implementing the change determined by the change advisory board . Usually based on complexity, scope, testing, recovery, timing, etc.
Change start date	The scheduled date the change's implementation .
Change completion date	The date the change's implementation is complete.

Change advisory board (CAB)	A list of individuals responsible for assessing, approving and scheduling the change.
Pending reason	A short description or code that indicates why the change is not progressing.

Learn more about IT service management (ITSM)

Get more tips and tricks for successful [ITSM](#), view case studies, and learn how to take your service desk to the next level.

[Check out the ITSM resources on IT Unplugged.](#)

Managing incidents with your IT service desk

An incident model helps service desks investigate, record, and resolve service interruptions or outages. An Information Technology Infrastructure Library (ITIL) incident management workflow aims to reduce downtime and negative impacts.

Incident management focuses on short-term solutions. To manage reoccurring incidents or underlying problems, see [Managing problems with your IT service desk](#).

The IT Service Desk template comes with an incident management workflow. This workflow ensures that you log, diagnose, and resolve incidents. We recommend you start with this workflow and adapt it to your business needs.

When managed well, incident records can identify:

- missing service requirements
- potential improvements
- future team member training

This page describes some best practices for managing incidents using Jira Service Desk. You may seek formal training in how to make ITIL recommendation work best for your business.

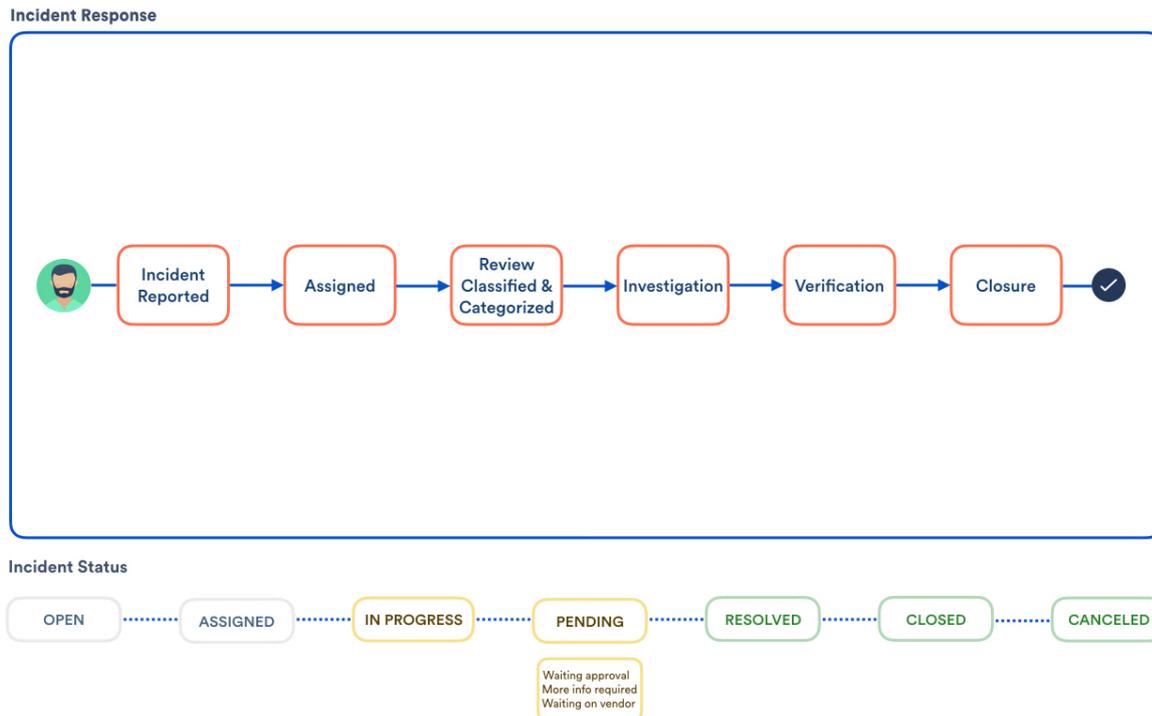
On this page

- Incident management process
- Set up incident management in Jira Service Desk
 - Configure the workflow and fields with the Incident Management workflow add-on
 - Incident management workflow
- Default form fields for incident reports
- Learn more about IT service management (ITSM)

Incident management process

The IT Service Desk template associates certain requests with an incident workflow. We set up the workflow to complement the following incident management process. We recommend you start with this workflow and adapt it to your specific needs over time.

The ITIL incident management process, in brief:



1. Service end users, monitoring systems, or internal IT members report interruptions.
2. The service desk describes and logs the incident. They link together all reports related to the service interruption.
3. The service desk records the date and time, reporter name, and a unique ID for the incident. Jira Service Desk does this automatically.
4. A service desk agent labels the incidents with appropriate categorization. The team uses these categories during post-incident reviews and for reporting.
5. A service desk agent prioritizes the incident based on impact and urgency.
6. The team diagnoses the incident, the services effected, and possible solutions. Agents communicate with incident reporters to help complete this diagnosis.
7. If needed, the service desk team escalates the incident to second-line support representatives. These are the people who works regularly on the effected systems.
8. The service desk resolves the service interruption and verifies that the fix is successful. The resolution is fully documented for future reference.
9. The service desk closes the incident.
10. Team members should carry out post-incident reviews for major incidents. These investigations can help determine:
 - a. missing requirements
 - b. potential changes to service level agreements
 - c. potential service improvements or focus areas

Set up incident management in Jira Service Desk

Configure the workflow and fields with the Incident Management workflow add-on

We used the ITIL framework to build the following workflow add-on for incident management: <https://marketplace.atlassian.com/plugins/com.atlassian.servicedesk.incident/server/overview>.

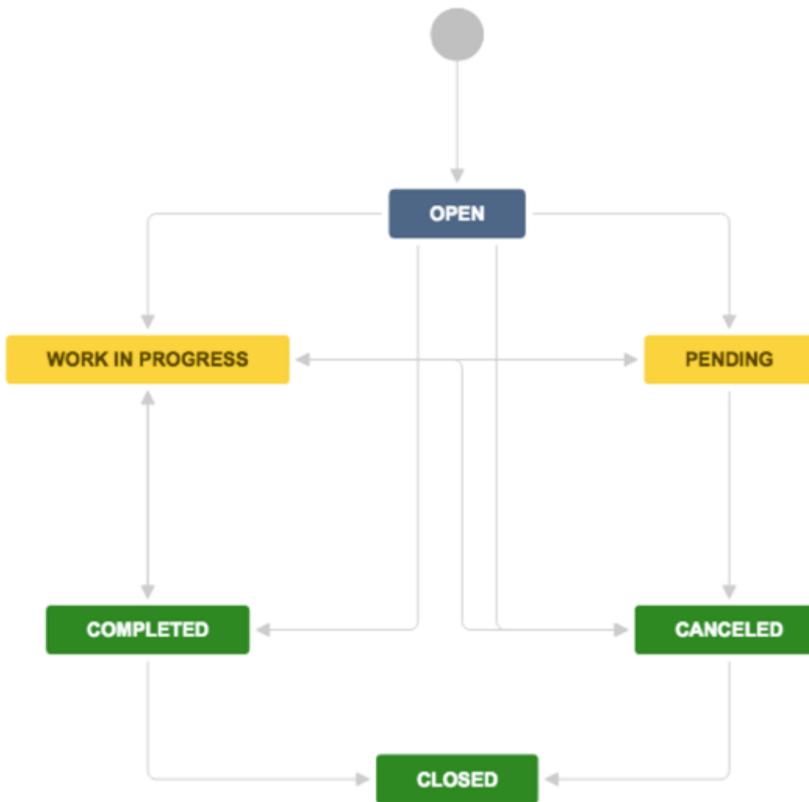
You can use this workflow as a template for your own incident management process.

To use the workflow from the Marketplace:

1. Log in as a user that has the Jira administrator global permission, and follow the instructions listed here to [import a workflow](#).
2. To add the workflow fields to your incidents, activate the screen by following the instructions here: [Defi](#)

ning a screen.

Incident management workflow



Default form fields for incident reports

Jira Service Desk allows you to customize the fields of information collected from customers. Additionally, you can customize the fields of information used by your agents. Jira Service Desk does this through issue type fields and screens. Fields help agents investigate, assess, and categorize the incident for reporting or querying.

By default, we include the following fields in your agents' view of an incident. If needed, you can add custom fields. [Find out more about fields in Jira.](#)

Field name	Description
Summary	A short description of the request.
Reporter	The person who submitted the request.
Component/s	Segments of your IT infrastructure that relate to the request. For example, "Billing services" or "VPN server". These are used for labeling, categorization, and reporting.
Attachment	Files or images added to the request.
Description	A long, detailed description of the request.
Linked Issues	A list of other requests that affect or are effected by the request. If your business uses other Atlassian products, this list may include linked development issues.

Assignee	The team member assigned to work on the request.
Priority	The importance of the request's resolution, usually in regards to your business needs and goals. Sometimes, priority is calculated by impact and urgency.
Labels	A list of additional custom labels used for categorizing or querying records.
Request participants	A list of extra customers who take part in the request, for example, people from other teams or vendors. Read more about participants.
Approvers	A list of people responsible for approving the request, usually business, financial or technical contacts .
Organizations	A list of customer groups interested in the request's resolution.
Impact	The effect of the incident, usually in regards to service level agreements.
Urgency	The time available before the business feels the incident's impact.
Pending reason	A short description or code that indicates why the incident is not progressing.
Product categorization	A category of IT asset or system that the request effects.
Operational categorization	A category of action or function required to fulfill the request.
Source	The asset or system where the incident originated.

Learn more about IT service management (ITSM)

Get more tips and tricks for successful [ITSM](#), view case studies, and learn how to take your service desk to the next level.

[Check out the ITSM resources on IT Unplugged.](#)

Managing problems with your IT service desk

Information Technology Infrastructure Library (ITIL) distinguishes between incidents and problems. Incident management serves to quickly restore services or broken experiences. For more information on incidents, see [incident management](#).

Problem management seeks to prevent incidents from happening again. Problems are typically reported by internal IT members and not their customers.

An ITIL problem management workflow aims to investigate, record, and prevent IT infrastructure problems. The IT Service Desk template comes with a built-in workflow for handling problems. We recommend you start with the template's default workflow and adapt it to your business needs. When correctly managed, problem records prompt agents to detail known errors and workarounds in your knowledge base. These documents:

- help service agents resolve issues and restore services
- reduce downtime
- increase the quality and trust of your IT infrastructure

This page describes best practices for managing problems using Jira Service Desk. You may seek formal training in how to make ITIL recommendation work best for your business.

On this page

- Problem management process
- Set up problem management in Jira Service Desk
 - Configure the workflow and fields with the problem management workflow add-on
 - Problem management workflow
- Default form fields for problem reports
- Learn more about IT service management (ITSM)

Problem management process

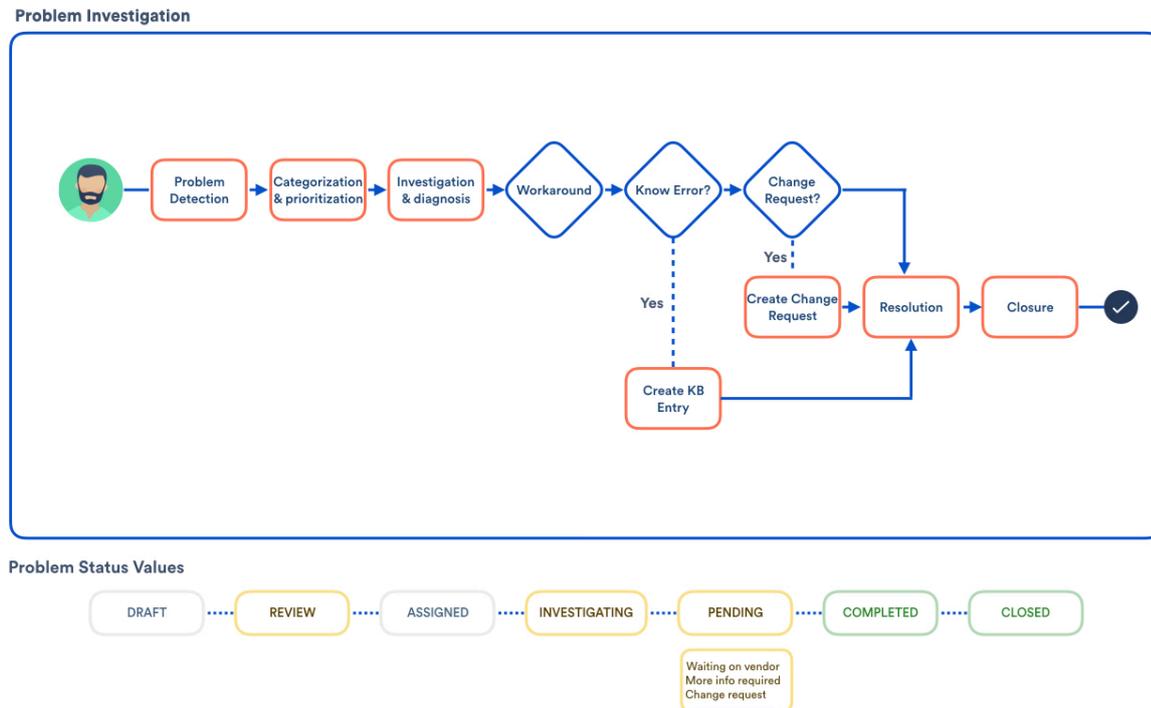
The IT Service Desk template comes with a problem-management workflow. We set up this workflow to complement the following problem management process. Use the workflow to transition problem registers

alongside these ITIL recommended activities:

- problem investigation
- identification of workarounds
- recording of known errors

We recommend you start with our default workflow and adapt it to your specific needs over time.

The ITIL problem management process, in brief:



1. Incident trends, vendors, or technical support staff report problems to the service desk.
2. A service desk team member records the details of the problem and links all related incidents.
3. A service desk agent labels the problem with appropriate categorization. They may reuse the labels of the incidents linked to the problem. The team uses these categories during review and for reporting.
4. A service desk agent prioritizes the problem. They base priority on the frequency of related incidents and their impact.
5. The service desk team determines the root cause of the problem.
6. The service desk team records the workarounds used to resolve related incidents. These workarounds to reduce service interruptions until the service desk fully resolves the problem.
7. The service desk team adds known errors to their knowledge base. They include symptoms of related incidents and relevant workarounds.
8. The service desk team proposes a change to the infrastructure to resolve the problem.
9. The service desk closes the problem.
10. Team members should carry out in-depth reviews of major problems.

Set up problem management in Jira Service Desk

Configure the workflow and fields with the problem management workflow add-on

We used the ITIL framework to build the following workflow add-on for problem management: <https://marketplace.atlassian.com/plugins/com.atlassian.servicedesk.problem> .

You can use this workflow as a template for your own problem management process.

To use the workflow from the Marketplace:

1. Log in as a user that has the Jira administrator global permission, and follow the instructions listed here to [import a workflow](#).

- To add the workflow fields to your incidents, activate the screen by following the instructions here: [Defining a screen](#).

Problem management workflow

Default form fields for problem reports

Jira Service Desk allows you to customize the fields of information collected from customers. Additionally, you can customize the fields of information used by your agents. Jira Service Desk does this through issue type fields and screens. Fields help agents investigate, assess, and categorize the problems for reporting or querying.

By default, we include the following fields in your agents' view of a problem. If needed, you can add custom fields. [Find out more about fields in Jira](#).

Field name	Description
Summary	A short description of the request.
Reporter	The person who submitted the request.
Component/s	Segments of your IT infrastructure that relate to the request. For example, "Billing services" or "VPN server". These are used for labeling, categorization, and reporting.
Attachment	Files or images added to the request.
Description	A long, detailed description of the request.
Linked Issues	A list of other requests that affect or are effected by the request. If your business uses other Atlassian products, this list may include linked development issues.
Assignee	The team member assigned to work on the request.
Priority	The importance of the request's resolution, usually in regards to your business needs and goals. Sometimes, priority is calculated by impact and urgency.
Labels	A list of additional custom labels used for categorizing or querying records.
Request participants	A list of extra customers who take part in the request, for example, people from other teams, or vendors. Read more about participants .
Approvers	A list of people responsible for approving the request, usually business, financial or technical contacts .
Organizations	A list of customer groups interested in the request's resolution.
Impact	The effect of the problem, usually in regards to service level agreements.
Urgency	The time available before the business feels the problem's impact.
Source	The asset or system where the problem originated.
Investigation reason	The trigger for prompting an investigation. For example, reoccurring incidents, non-routine incidents, or other.
Pending reason	A short description or code that indicates why the problem is not progressing.

Product categorization	A category of IT asset or system that the request effects.
Operational categorization	A category of action or function required to fulfill the request.
Root cause	The original cause of the incidents related to the problem.
Workaround	The detailed description of a temporary, known solution to restore a service. If you have Confluence, we recommend you document your workaround in your knowledge base.

Learn more about IT service management (ITSM)

Get more tips and tricks for successful [ITSM](#), view case studies, and learn how to take your service desk to the next level.

[Check out the ITSM resources on IT Unplugged.](#)

Calculating priority automatically

Jira Service Desk comes with some powerful automation tools. IT teams can set up their service desk to calculate a request's priority automatically.

Removing manual processes gives time back to your team. Your team spends less time triaging and prioritizing requests. And, they spend more time resolving IT service tasks. Calculating the correct priority helps put requests into the correct SLA.

Some IT teams use an impact-urgency matrix to determine the priority of an issue. This page walks through an example for defining this matrix. Then, it discusses using these decisions to automate how priority calculation with these fields.

At the end, you should know a bit more about automation and how it can help you remove all sorts of manual processes. Automation frees up your agents' time and makes your service desk more efficient.

Create a priority matrix using impact and urgency values

Work with your team to determine how your service desk prioritizes incidents. Below is a sample matrix for how our team thinks about priority. Yours may differ depending on your resources and other factors.

Here's an example matrix:

Impact	Urgency			
	Critical	High	Medium	Low
Extensive / Widespread	Highest priority	Highest priority	High priority	Medium priority
Significant / Large	Highest priority	High priority	Medium priority	Low priority
Moderate / Limited	High priority	Medium priority	Low priority	Lowest priority
Minor / Localized	Medium priority	Low priority	Lowest priority	Lowest priority

Note that the priority values listed above are just examples for this tutorial. You can also create priorities that are specific to your service desk. See [Defining priority field values](#) and [Associating priorities with projects](#) for more details.

Before you begin

Make sure you have the following two [custom fields](#) of the type **Select List (single choice)**, each containing the values listed in the previous table:

- **Urgency:** Critical, High, Medium, Low
- **Impact:** Extensive, Significant, Moderate, Minor

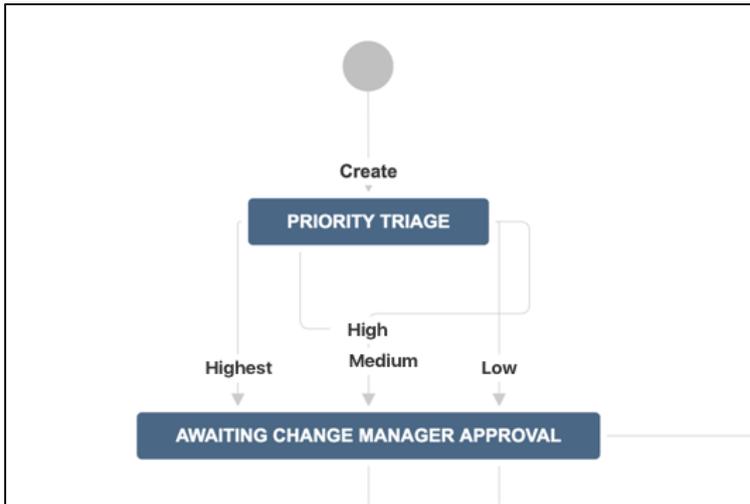
Automating the calculation

After Step 1, the workflow transitions a change to **Priority triage** status as soon as an issue is created. You will also have 4 transitions to set the **Priority** to a value.

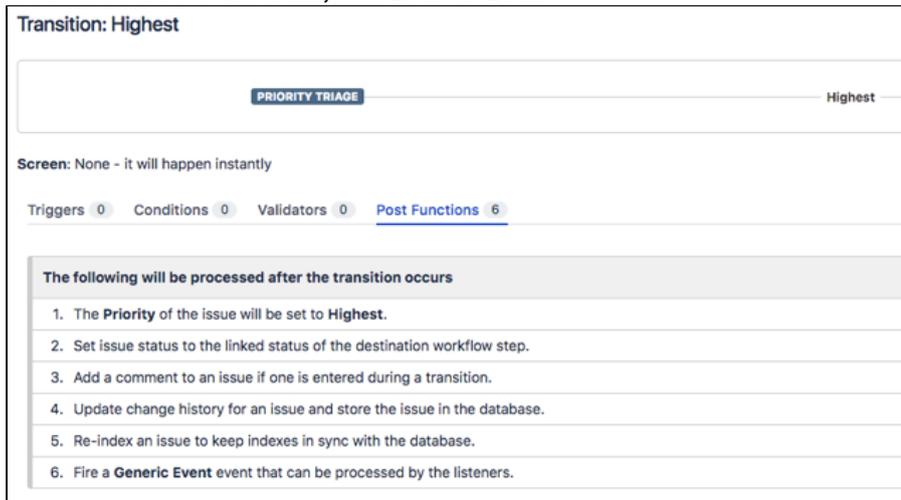
With Step 2, you will set up an automation rule that checks the value of the Urgency and Impact fields and fires off the corresponding transition according to the matrix.

Step 1: Configure the workflow

1. Go to the workflow that is used by your **Change** issue type.
2. Between the **Create** transition and your first status, add a new status and name it **Priority Triage**.
3. Add the following four transitions from this status:
 - a. Highest
 - b. High
 - c. Medium
 - d. Low



4. In each of these transitions, **add a post function**.
 - a. Select the **Update Issue Field** post function.
 - b. In the post function, update the **Priority** field to match the transition, for example, the *Highest* transition will have a post function that changes the **Priority** field to **Highest**. Similarly, the **High** transition will have this post function to set the **Priority** field as **High**, and the **Medium** transition to set the value to **Medium**, and **Low** to **Low**.



5. Publish the workflow.

Step 2: Configure the automation rule

After the post function is set, let's create an automation rule that triggers the appropriate transition depending on the urgency and impact selected during request creation.

The rules should follow the following pattern:

- **When:** Issue created
- **If, or Else if:** Specify the urgency and impact value pair according to the matrix, for example:

```
status = "Priority triage" AND Impact = "Extensive / Widespread"
AND Urgency = Highest
```

- **Then:** Transition issue, and select the transition that matches the value pair according to the matrix.

Note: If your **Urgency** or **Impact** value is optional on the request type form or issue create screen, then there might be cases where these fields are empty. In this case, make sure that you add a **Else if** condition that caters for this scenario, for example:

- **Else if:**

```
status = "Priority triage" AND Impact is empty OR Urgency is
EMPTY
```

- **Then:** Transition issue, **Low**.

Learn more about IT service management (ITSM)

Get more tips and tricks for successful [ITSM](#), view case studies, and learn how to take your service desk to the next level.

[Check out the ITSM resources on IT Unplugged.](#)

Best practices for software teams using Jira Service Desk

This guide helps teams using server versions of Jira Service Desk and Jira Software.

Software teams that develop with Jira Software can give technical support and get customer feedback using Jira Service Desk. Teams using Jira Software can view and comment on Jira Service Desk bug reports *without any additional licensing or billing cost*. Read more about [collaborating on Jira Service Desk](#).

You can use this guide and Jira Service Desk's customer service project template to:

- collect user bugs and feedback via a built-in web portal
- give users technical support
- help users with licensing and billing

Jira Service Desk is highly configurable, and the template is just a starting point. Over time, you can fine-tune the template to your specific business needs.

This guide covers how to:

- [Get set up for customer service](#)
- [Collect effective bug reports from customers](#)
- [Customize Jira Service Desk's bug report workflow](#)
- [Collaborate with other Jira teams on Jira Service Desk issues](#)
- [Escalate Jira Service Desk issues to other Jira teams](#)

Collaborate with other Jira teams on Jira Service Desk issues

Software teams can get context from your service desk agent's conversations with customers. You can set up your service desk to allow Jira Software teams to view and comment on Jira Service Desk issues.

By default, Jira Service Desk agents can:

- view Jira Software issues
- comment on Jira Software issues

- transition Jira Software issues

By default, Jira Software and Jira Core users:

- *can't* view Jira Service Desk requests in the customer portal
- *can't* view Jira Service Desk issues in a Jira Service Desk project
- *can't* comment on Jira Service Desk requests in the customer portal
- *can't* comment on Jira Service Desk issues in a Jira Service Desk project
- *can't* view links to Jira Service Desk issues in Jira Software or Jira Core project

If you plan to work with other Jira teams, you should change your project's permissions. We recommend giving all logged-in users permission to view and comment on service desk projects.

The table below contains a breakdown of three different types of permissions. These permissions make collaboration between service desk agents and software development teams possible:

	Request participants	View permissions on Jira Service Desk projects	Comment permission on Jira Service Desk projects
Permissions	These users can: <ul style="list-style-type: none"> • view requests in the portal • add public comments in the portal • add attachments in the portal • transition the request through customer visible statuses in the portal 	These users can: <ul style="list-style-type: none"> • view issues in service desk projects • view linked service desk issue information in Jira Software projects 	These users can: <ul style="list-style-type: none"> • add internal comments on service desk project issues
Effect on Jira Service Desk licensing and billing	None	None	None

The easiest way to collaborate (out of the box)

Jira Service Desk agents can add other Jira team members as request participants. Then, Jira Software or Jira Core members can interact with customers in the web portal.

Request participants can:

- view requests in the portal
- add public comments in the portal
- add attachments in the portal
- transition the request through customer visible statuses in the portal

Agents can add request participants to service desk issues. Look for the **People** section in the issue and select the **Request participants** field to add members from other Jira teams.

[Read more about request participants.](#)

Modify your permissions to let other teams view and comment on service desk issues

To view and edit the permission scheme for your service desk project, go to **Project settings** (



) > **Permissions**.

You need to be a Jira admin to change project permission schemes. Changes to permission schemes affect all projects that share that scheme. Be careful!

[Read more about configuring project permission schemes.](#)

Give developers permission to view service desk issue

To let all Jira Server users on your site view issues on your service desk:

1. On the project permission scheme page, select **Actions > Edit permissions**.
2. Under **Project Permissions**, select **Edit** in the **Browse Projects** entry.
3. In the **Granted** to selection, choose **Application access**.
4. From the dropdown, select **Any logged in user**.
5. Select **Grant**.

This allows other Jira users to:

- view service desk issues
- view links to service desk requests in their own projects

Give developers permission to comment on service desk issues

To let all Jira Server users on your site comment internally on service desk issues:

1. On the project permission scheme page, select **Actions > Edit permissions**.
2. Under **Comment Permissions**, select **Edit** in the **Add comments** entry.
3. In the **Granted to** selection, choose **Application access**.
4. From the dropdown, select **Any logged in user**.
5. Select **Grant**.

Agents, not customers, will see these comments. Only Jira Service Desk agents and admins can comment directly to customers. Other Jira users may comment publicly if the agent adds them as request participants.

Share your development teams' custom fields

Bug issues in Jira Software and Jira Service Desk can stay in sync if they share custom fields and screens. We suggest you replicate the issue fields and screens from your development teams' bug issues in your Jira Service Desk request types.

Work with your development teams' and Jira administrators to share custom fields between your request types and development projects.

[Read more about custom fields.](#)

Collect effective bug reports from customers

Jira Software is an issue and bug tracker that helps developers plan, build and ship their work. Jira Service Desk is a simple way for customers to send your team bugs and feedback about shipped software.

Your users are your best friends. Their bug reports can help you:

- resolve issues before other users encounter them
- investigate and fix problems that slip through QA or automated testing
- catch issues from platforms you don't prioritize during testing
- show your users you care about quality and their experience

Issues can happen in any part of software development, not just the code. If your designers, documentation writers, product managers, or other team members track their work in Jira, you can escalate bugs related to their work directly to their teams, too.

The bug reporting and resolution process

The information needed to replicate and resolve bugs varies. But, you can standardize your process for collecting bug reports.

The customer service template associates certain requests with a bug report workflow. This workflow complements the bug report process. Use it as a jumping off point for your service desk.

Requests that follow a bug report process have the same workflow in Jira Service Desk and Jira Software. But, the teams' processes for handling bug reports have significant differences.

Bug process for agents working in Jira Service Desk	Bug process for developers working in Jira Software
<ol style="list-style-type: none"> 1. A user reports a problem with their software or service. 2. A service desk agent investigates to see if they can provide any known solutions to the customer: <ul style="list-style-type: none"> • If the problem has a known solution, the service desk agent works with the customer to fix the problem. • The problem may be new or the service desk agent can't find an answer from a knowledge base or development point of contact. In this case, the service desk agent escalates the bug report to the development team for fixing. Read more about escalating issues to other Jira teams. 3. The service desk agent liaises between the customer and development team to collect any extra information needed to fix the bug. They communicate the development team's progress. 4. The service desk agent verifies with the customer that they fixed the problem. Then, the agent resolves the customer's bug report. 	<ol style="list-style-type: none"> 1. The development team receives a bug report. 2. The project manager determines the bug's priority. They assign the issue to a developer. 3. The developer investigates the bug and either verifies or rejects the report. 4. The developer fixes the problem and transitions the issue to a QA tester. 5. The tester verifies they fixed problem and resolves the ticket. They notify the service desk agent that development is complete. They may provide extra information, like when the fix will reach the customer.

Collect specific information from users with custom fields

Work with your development team and Jira administrators to share a custom field set. Define what custom fields you want to collect to aid developers fixing bugs. Your Jira administrator can maintain these fields in a single screen scheme. They can apply the scheme to both development and service desk projects. [Read more about custom fields and screen schemes.](#)

Developers note incomplete information as the biggest blocker to investigating and fixing bugs. The most common information that developers use are:

- the steps to reproduce
- observed and expected behavior
- screenshots

You might want to collect other information to categorize, report, or automate actions related to the bug report. For example:

- operating system
- version
- component
- URL
- user agent string

The more information you can collect, the easier diagnosing the problem will be. Your development team will thank you.

By default, the **Report a bug** request type comes with these fields:

- Summary
- Symptom
- Attachment

To add pre-created custom fields to your request types:

1. In your service desk project, select **Project settings** () > **Request types**.
2. In the **Report a bug** request type entry, select **Edit fields**.
3. Select **Add a field**.

[Read more about adding custom fields.](#)

Tips for creating bug report forms on your portal

- You can add help and instructions to your bug report request type. Encourage your customers to report each problem on a separate request. This helps tracking bugs in reports and development sprints.
- Use natural language when asking for information. For example, if you add a field to collect expected behavior, ask your customer to report these in plain terms: "What did you expect to happen?"
- Check in with your development teams every so often. Ask if there's any more information they need to squash bugs in your software.
- Check in with your Jira administrators every so often. Ask if there's any changes to screens that you should reproduce in your request types.

Customize Jira Service Desk's bug report workflow

When a customer reports a bug using the portal, Jira Service Desk assigns the request to your organization's **Bug** workflow. By default, this workflow is the same for both Jira Service Desk and Jira Software issues.

This workflow follows the basic process above. Jira administrators can customize it to adapt to the needs of your customers, developers, and service desk agents.

Customize your bug report and resolution workflow

Jira's default **Bug** workflow is a good starting point for most teams. As your organization grows, you may want to update your workflow to suit the specific needs of your business. For example, you can add steps for approvals or QA verification.

You need to be a Jira admin to make changes to workflows. Changes to workflows affect all projects that use the workflow. Be careful!

To edit the bug workflow:

1. In your service desk project, select **Project settings** () > **Workflows**.
2. Select the edit icon () next to the entry titled **<Project key>: Jira Service Desk default workflow**.
3. Use the workflow editor to add or remove steps and transitions.

[Read more about working with workflows.](#)

Add a "waiting on development" step to the request's workflow

You may want to add a stage to your bug workflow for when a development team investigates the issue. This status helps your service team track which bugs are being handled by a development team. And, it informs customers when the issue is being looked at by developers.

To add a new step in the workflow:

1. Follow the above steps to customize the **<Project key>: Jira Service Desk default workflow**.
2. Select **+ Add status**.
3. Give the status a name. For example, *Waiting on development team* and select Add.

4. Give the status a description. This helps administrators who manage workflows across Jira Server apps.
5. Switch the category to **In progress**.
6. Select **Create**.

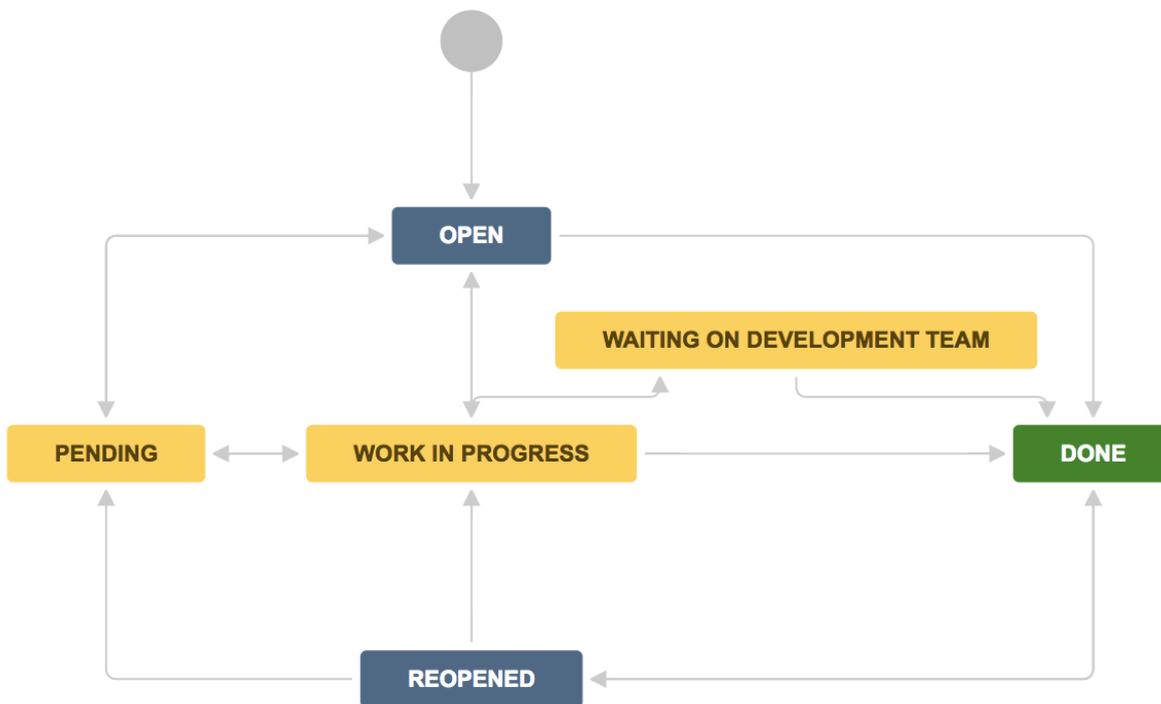
To add an incoming "escalate to development team" transition:

1. Select the **Waiting on development team** status.
2. Select **Add Transition**.
3. In the **From status** drop down, select **Work in progress**.
4. In the **To status** drop down, select **Waiting on development team**.
5. In the **Name** field, provide the call to action. For example, *Escalate to development team*.
6. In the **Screen** drop down, select **Workflow Screen**.
7. Select **Add**.

To add a transition to allow the agent to mark the request as done:

1. Select **+ Add transition**.
2. In the From status drop down, select **Waiting on development team**.
3. In the To status drop down, select **Done**.
4. In the **Name** field, provide the call to action. For example, *Mark as close*.
5. In the **Screen** drop down, select **Resolve Issue Screen**.

Publish your draft workflow.



Escalate Jira Service Desk issues to other Jira teams

As agents investigate bug reports, they may need to escalate issues to the development team. For example, the agent may not find a fix from the knowledge base or a development contact. In this case, they may escalate the issue to a development team to fix in the software.

The development team should use an issue on their software board or backlog to track the fix. To make this easier, Jira Service Desk agents can create these issues for them. Here's how the process works:

1. A service desk agent receives a bug report from a customer.
2. The agent verifies that the bug needs a developer to fix the issue.
3. The service desk agent creates a linked issue in the development team's Jira Software project.
4. The development team lead prioritizes and assigns the software issue to a developer to fix.
5. The developer works in their Jira Software project while fixing the issue.

6. The developer resolves the issue and Jira Service Desk automatically updates the service agent about the fix.

Agents can pass on feature requests or improvement suggestions to project planning teams. They can also create issues with public-facing websites for marketing or documentation teams that use Jira Core.

Create linked issues

Service desk agents can escalate a Jira Service Desk issue to a Jira Software or Jira Core project by creating a linked issue. To create a linked issue:

1. View the issue.
2. Select the **More** option.
3. Select **Create linked issue**.
4. Choose the appropriate project where the issue needs to be escalated to.
5. Select **Create**.

Linked issues created this way use a "causes" link by default. This means:

- The service desk issue is "caused by" the linked development issue.
- The development issue "causes" the linked service desk issue.

Jira Software and Jira Core teams get the most use of this technique when they can view and comment on issues in your service desk project. [Read more about collaborating with other Jira teams on Jira Service Desk issues.](#)

Automatically update Jira Service Desk agents about the progress of a linked issue

Service desk projects come with an automation rule that updates agents on the status of linked issues. The rule adds an internal comment on the service desk issue whenever another team transitions a "caused by" linked issue.

So, when a developer looks into a bug and resolves it as "fixed", they notify the service desk agent in the process. Then, the agent can follow up with the customer to make sure the fix works for them.

We enable the **Update when a linked issue changes** automation rule by default. To view or edit this rule:

In your service desk project, select **Project settings** (



) > **Automation**.

[Read more about automation.](#)

Get set up for customer service

Create a customer service project

You need to be a Jira administrator to do this step.

To create a project using the customer service template:

1. Select **Projects > Create project**.
2. Choose the **Customer service** template and hit **Next**.
3. Name your project.
4. Select **Submit**.

Your new project comes with a service catalog, recommended workflows, and basic reports. If you bundle Jira Service Desk with Confluence, we also create a Confluence knowledge base. We recommend using this to record known solutions, FAQs and other articles that your customers can use to serve themselves. [Try Confluence for free.](#)

Create forms to collect the right information from your users

Our web forms make sure users give you all the details you need to give support or collect feedback. If you get all the information up front, you can:

- prioritize requests
- get them to the right agent
- give development teams the information they need to investigate and fix bugs quickly

We call these customizable web forms *request types*. To start, find the most common and urgent requests that users have sent your way, then make sure there's a corresponding request type for each of these.

To view or edit your request forms:

1. In your service desk project, select **Project settings** (



) > **Request types**. The forms there display in your web portal.

The customer service template comes with the following request types:

- Technical support
- Licensing and billing questions
- Product trial questions
- Other questions
- Report a bug
- Suggest a new feature
- Suggest improvement

[Read more about request types.](#)

Fill a knowledge base so customers can help themselves

A knowledge base stores how-tos, FAQs, and other short articles. Customers can use these to solve problems without contacting support. And, agents can use these to share knowledge and solve requests faster.

The easiest way to add a knowledge base to your service desk is to link your project to a Confluence knowledge base space. When you add a knowledge base to your service desk, it benefits both your users and your agents:

1. Customers search and view relevant articles when they search the customer portal. They may find the answers they need without ever raising a request.
2. Agents find relevant articles when they work on issues. This saves time finding and writing common answers or onboarding new team members.

Learn more at [Set up a knowledge base for self-service.](#)

Getting help with Jira Service Desk

How can we help you?

We have a number of [help resources](#) available. You can get your problem resolved faster by using the appropriate resource.

I don't know how to do something

Something isn't working

I don't like the way something works

Something else?

I don't know how to do something

1. Search the [Atlassian Community](#).
2. Raise a [support request](#)*

Something isn't working

1. Check the 'Jira Software' knowledge base at <https://confluence.atlassian.com/display/JiraKB>.
2. Search the [Atlassian Community](#).
3. Raise a [support request](#)*

If you've identified a bug but don't need further assistance, raise a [bug report \(https://Jira.atlassian.com\)](https://Jira.atlassian.com).

I don't like the way something works

- Raise a [suggestion \(https://Jira.atlassian.com\)](https://Jira.atlassian.com).

Something else?

- If you need help with something else, raise a [support request](#)*

* *Tip: If you are the **Jira system administrator**, you have a number of additional support tools available. See [Raising support requests as an administrator](#) for details.*

About our help resources

Atlassian Support

Our support team handles support requests that are raised in our [support system](#). You need to log in using your [Atlassian account](#) before you can raise support requests.

For information on our general support policies, including support availability, SLAs, bugfixes, and more, see [Atlassian Support Offerings](#). Note, you'll find anything security-related at [Security @ Atlassian](#).

Tip:

If you are a Jira system administrator, you have a number of additional support tools available. These include the ability to raise a support request from within your Jira applications, create zip files of key Jira application information, and more. For details, see the [Administering Jira Applications](#) documentation.

Atlassian Community

[Atlassian Community](#) is our official application forum. Atlassian staff and Atlassian users contribute questions and answers to this site.

You may be able to find an answer immediately on the Atlassian Community, instead of having to raise a support request. This is also your best avenue for help if:

- you are using an unsupported instance or an unsupported platform,
- you are trying to perform an unsupported operation, or
- you are developing an add-on for Jira Service Desk.

You can also have a look at the [most popular Jira Service Desk answers](#) and the [most popular Jira answers](#).

Jira Service Desk knowledge base

If there are known issues with a version after it has been released, the problems will be documented as articles in our [knowledge base](#).

Atlassian issue tracker

Our official [issue tracker](#) records our backlog of bugs, suggestions, and other changes. This is open for the public to see. If you log in with your Atlassian account, you will be able to create issues, comment on issues, vote on issues, watch issues, and more.

Tip:

Before you create an issue, search the existing issues to see if a similar issue has already been created.