



User documentation for JIRA Service Desk Server 3.0

Contents

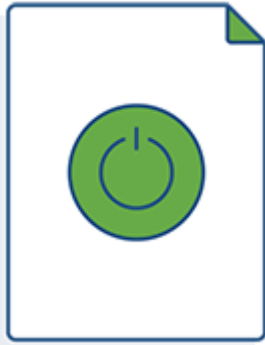
JIRA Service Desk Documentation	4
JIRA applications overview	4
Permissions overview	7
Installing JIRA Service Desk	13
Getting started with JIRA Service Desk	14
Getting started for service desk admins	16
Setting up your service desk	17
Creating service desk request types	18
Making queues for your service desk teams	21
Adding service desk agents	23
Customize your service desk channels	24
Bring your service desk to the next level	26
Introduce customers to your service desk	28
Getting started for service desk agents	29
Administering service desk projects	32
Automating your service desk	32
Configuring the customer portal	34
Configuring service desk notifications	36
Managing access to your service desk	37
Receiving requests by email	38
Managing the email channel	40
Troubleshooting issues with the email channel	41
Setting up queues for your team	44
Setting up request types	45
Troubleshooting issues with request types	50
Workflows	51
Setting up service desk users	51
Managing project role memberships	55
Using JIRA applications with HipChat	56
Default service desk project configuration	59
Working on service desk projects	61
Using service desk queues	61
Working with issues	62
Adding request participants	63
Attaching files and screenshots to issues	64
Creating issues and sub-tasks	65
Raising requests on behalf of customers	67
Editing and collaborating on issues	68
Linking issues	71
Editing multiple issues at the same time	76
Scheduling an issue	80
Moving an issue	81
Customizing the issues in a project	82
Logging work on issues	83
Searching for issues	86
Basic searching	89
Quick searching	91
Advanced searching	94
Advanced searching - functions reference	100
Advanced searching - fields reference	114
Advanced searching - keywords reference	139
Advanced searching - operators reference	141
Search syntax for text fields	150
Saving your search as a filter	155
Working with search results	159
Constructing cron expressions for a filter subscription	167

Configuring dashboards	169
Adding and customizing gadgets	171
Gadgets for JIRA applications	172
Managing your user profile	178
Allowing OAuth access	179
Requesting add-ons	182
Using keyboard shortcuts	183
Organizing work with components	184
Organizing work with versions	185
Setting up service desk reports	188
Setting up SLAs	188
Reporting on SLAs	194
Example: creating a basic SLA	196
Example: creating an SLA that doesn't track continuous time	196
Example: creating an SLA with multiple cycles	197
Managing SLA data	198
Serving customers with a knowledge base	199
JIRA Service Desk best practices	202
Best practices for designing the customer portal	202
Best practices for IT teams using JIRA Service Desk	204
Getting help with JIRA Service Desk	206

JIRA Service Desk Documentation

Put the power of the JIRA platform in the hands of your service desk team.

Find out more about JIRA Service Desk 3 here.



Getting started

- Getting started guide
- Installing JIRA Service Desk



Administering service desk

- Setting up service desk users
- Setting up service desk projects



Serving your customers

- Creating Service Level Agreements
- Setting up request types

Working on a service desk

- Searching for issues
- Working with issues
- Using service desk queues

Latest service desk news

- JIRA Service Desk Server release notes

Integrations

- Connecting to a knowledge base
- Using JIRA applications with HipChat


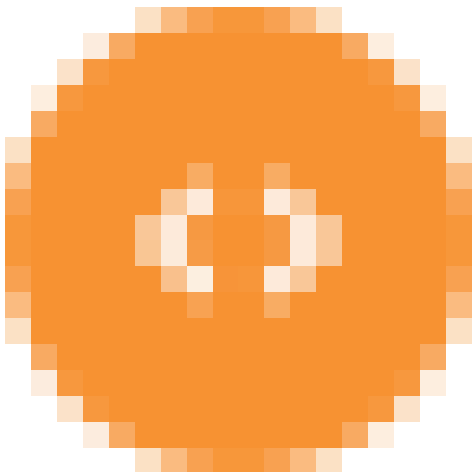

JIRA applications overview

The JIRA family of applications are built on the JIRA platform. JIRA Core is the default application of the JIRA platform, and will always be present in a JIRA instance. You may also choose to include other applications in your instance, such as JIRA Software and JIRA Service Desk. A user may require access to one, all, or any combination of these applications. If you're a JIRA administrator, you may want to read some more information on licensing and application access. See [Licensing and application access](#).

Application features and project types


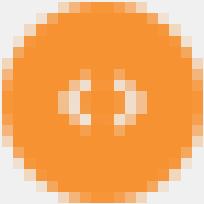

Each application delivers a tailored experience for its users, and has an associated project type which in turn offers application specific features. Below is a list of the project types, and their associated application specific features.




Application	Project type	Application specific feature set
-------------	--------------	----------------------------------

<p>JIRA Core</p>	 <p>Business projects</p>	<ul style="list-style-type: none"> • Available to all licensed users
<p>JIRA Software</p>	 <p>Software projects</p>	<ul style="list-style-type: none"> • Integration with development tools • Agile boards • Release hub for software versions
<p>JIRA Service Desk</p>	 <p>Service Desk projects</p>	<ul style="list-style-type: none"> • Service Level Agreements (SLAs) • A customisable 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.

A list of the applications, their user roles, and their project's application specific features is listed below:

			JIRA Core	JIRA Software	JIRA Service Desk
			JIRA-Core-user	JIRA-Software-user	JIRA-ServiceDesk-agent
Business Projects 	Project level	Create	✓	✓	✓
		View	✓	✓	✓
	Issue level	Create	✓	✓	✓
		View	✓	✓	✓
		Comment	✓	✓	✓
		Transition	✓	✓	✓
	JIRA Gadgets	View	✓	✓	✓
Software Projects 	Project level	Create	✗	✓	✗
		View	✓	✓	✓
	Issue level	Create	✓	✓	✓
		View	✓	✓	✓
		Comment	✓	✓	✓
		Transition	✓	✓	✓
		View Development Information	✗	✓	✗
		View Release information	✗	✓	✗
	Board level	Create	✗	✓	✗
		View	✗	✓	✗
	JIRA Software gadgets	View	✗	✓	✗
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](#).

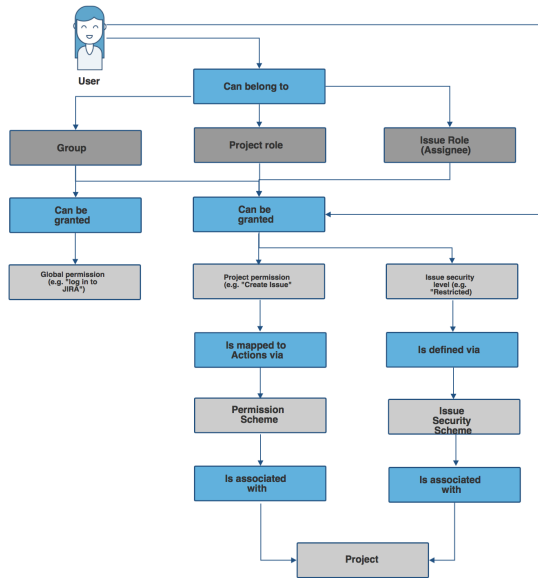
Types of permissions

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](#)

Installing JIRA Service Desk

If you are migrating from an existing JIRA instance with the JIRA Service Desk add-on, please check out our [Migration Hub](#) first.

Before you get your agents and customers started on a local instance of JIRA Service Desk Server, read the [JIRA Service Desk release notes](#) for the version that you are installing or upgrading to, then follow these instructions:

1. View the available JIRA Server applications [here](#).

2. Select the JIRA Service Desk package and choose the number of agents.
3. Download the installer.
4. Once the installer has downloaded, run it and follow the [Installing JIRA applications](#) steps.

If you experience any problems with your installation or you have any questions, contact [Support](#).

Get more out of your new JIRA Service Desk instance:

Connecting JIRA Service Desk to other Atlassian products enables a host of new integration features.

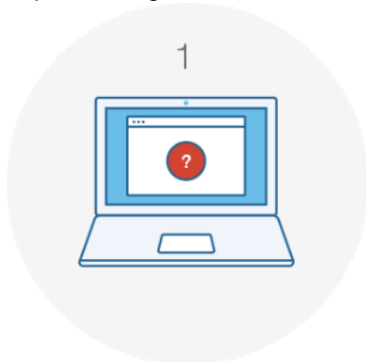
Learn more below:

- [Using JIRA applications with 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.
- [Using JIRA applications with HipChat](#) — HipChat is hosted group chat and video chat for companies and teams.

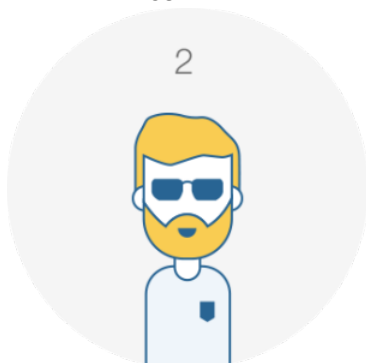
Getting started with JIRA Service Desk

JIRA Service Desk overview

JIRA Service Desk combines the productivity and power of the JIRA platform with an intuitive user experience that allows service teams to focus on the customer. Throughout this tutorial, we will reference the example of a new customer who uses JIRA Service Desk to send requests to his company's IT Team so he can settle into his new role. Here's how the customer and a service desk agent work together to resolve a request using JIRA Service Desk:



1 - Customer needs assistance and submits a request to JIRA Service Desk.



2 - Service desk agent picks up the issue.



3 - Customer and service desk agent discuss the problem.

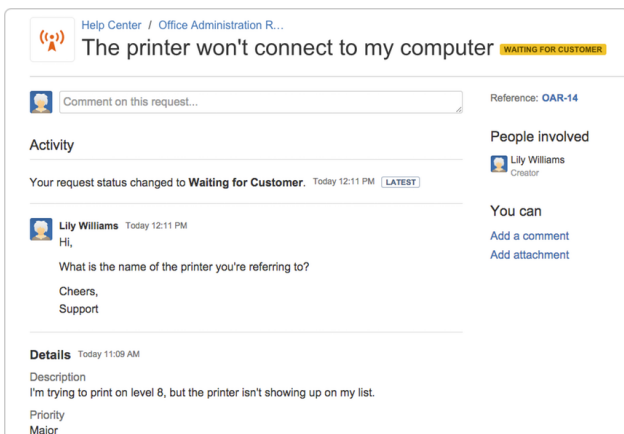


4 - The customer is satisfied and the service desk agent resolves the issue!

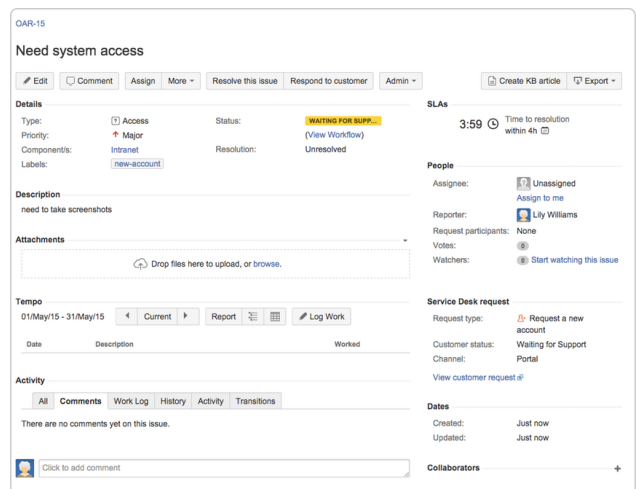
Request vs. issue

Your customers submit a JIRA Service Desk **request** through the customer portal or by email. These requests become issues that your agents work on internally in the JIRA Service Desk agent view.

How customers see a **request** in the customer portal:



How agents see an **issue** in the JIRA Service Desk agent view:



JIRA Service Desk roles

There are three main roles in JIRA Service Desk: administrator, agent, and customer. This guide focuses on the two licensed roles: administrators and agents. The administrator sets up and configures JIRA Service Desk projects. The agent works with the preconfigured service desk projects. Service desk customers are free and do not require a license. You can invite an unlimited number of customers to your service desk projects.

Admin

User with administrative rights for your service desk who can:

- Access all features in JIRA Service Desk
- Add and remove users to and from service desk projects
- Configure the customer portal, request types, queues, reports and SLA metrics
- Perform all tasks outlined in Admin and Agent tutorials

Agent

User who works on and resolves customer requests who can:

- Access the internal service desk interface
- View the customer portal, queues, reports and SLA metrics of assigned service desk projects
- Add, edit and delete customer-facing and private comments on issues
- Manage knowledge base content

Ready to dive into JIRA Service Desk?

Click on the admin or agent buttons below to proceed.

[I am a service desk admin](#)

[I am a service desk 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

Audience:

- Service desk administrators
- Team managers

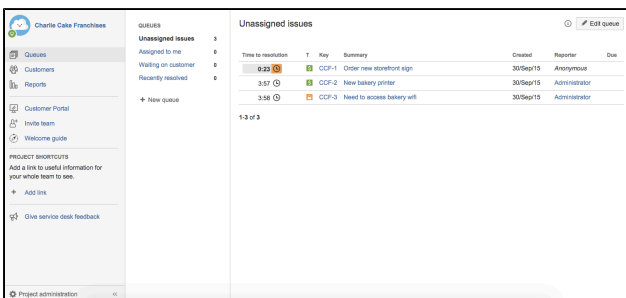
Time: 30 minutes

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 issues from the same tab:



Project administration

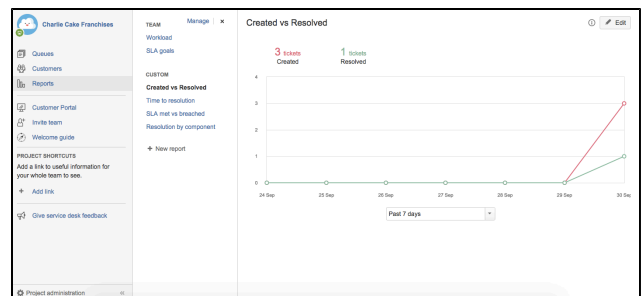


Here, you will administer request types, brand your customer portal, link your service desk to an email account, and manage users:

Reports



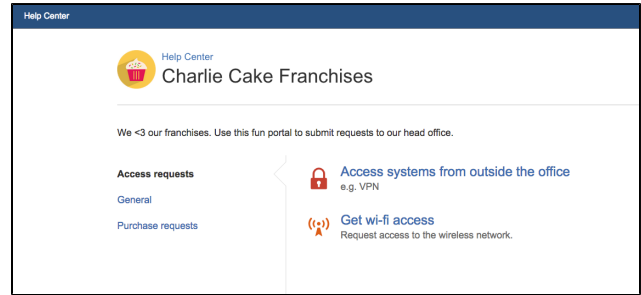
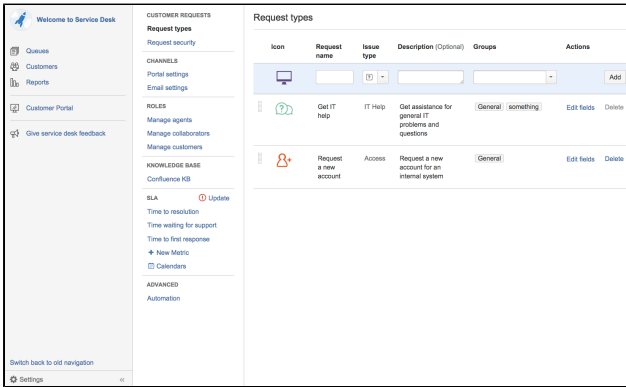
Use the Reports tab to view your team's workload. You can also set up custom reports to track your team's progress in more detail:



Customer portal



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



Now that you are familiar with your service desk workspace, you can set up your own JIRA Service Desksite 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

Let's get your service desk ready to use by setting you up with a JIRA Service Desk Cloud site. Cloud is our hosted offering and will allow you to set up your own site without installing a thing!

If you have an existing Cloud 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.

Sign up for a JIRA Service Desk site

Signing up for JIRA Service Desk Cloud will provide you with a fully-functional JIRA Service Desk site for one month.

1. Open [this link](#) in a new tab to view the signup page directly.
2. Follow the signup form steps to enter your site URL and admin username.
3. Once you have completed the signup process, grab a quick coffee (or tea, if that's your preference) — it will take a few minutes for

your JIRA Service Desk Cloud site to be created. You will receive an email when your site is ready.

Can't use Cloud?

If you cannot use JIRA Service Desk Cloud, instructions for installing JIRA Service Desk Server are available below.

- [Installing JIRA on Windows](#)
- [Installing JIRA on Linux](#)

Create a project

JIRA Service Desk comes with default project templates that you can use to suit your team's needs. Create a basic service desk project for teams that only need a few request types, or create an IT service desk for teams working with change and incident management processes. Let's get you set up with a basic service desk project.

1. Open the link just emailed to you to log in to your new site with the administrator credentials you selected.
2. Select **Projects > Create Project** from the the top navigation bar of your site.
3. Select "Basic Service Desk".
4. Name your project. In this example, we'll use the project name "Charlie Cake 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.
5. 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](#)

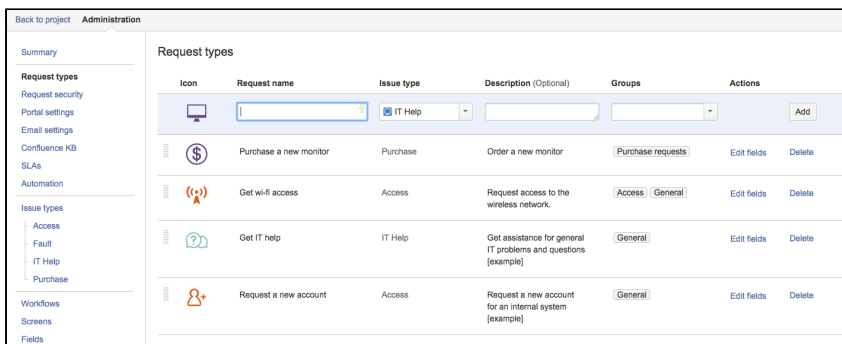
6. Bring your service desk to the next level
7. Introduce customers to your service desk

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 (e.g. 'Need a new monitor' vs 'Hardware Request');
- Break things down into smaller chunks (e.g. 'Help with printer configuration', 'Help with laptop problems', 'Help with software problems'); and
- Avoid specialist terminology (e.g. 'I need access to a system' vs 'Deploy SSH key').

Requests vs. Issues
Remember that customers submit *requests* to your service desk and your team picks up the corresponding *issues* to work on internally.

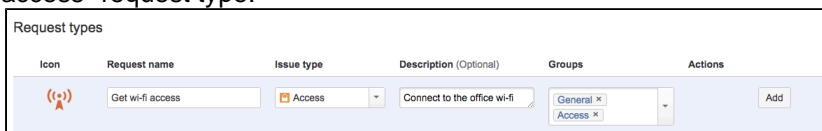
By the end of this step, your project's request type page should look something like this:



Create new request types

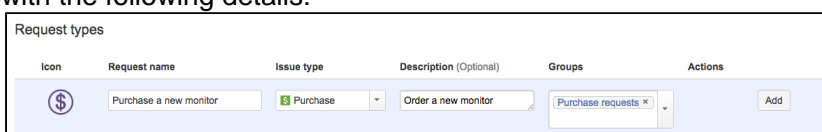
Let's go ahead and add two new request types, so you can familiarize yourself with the request type configuration options.

1. In your new service desk project, select **Project administration > 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 "Get wi-fi access" request type:



Select **Add** when finished.

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



Select **Add** when finished.

4. For the "Get wi-fi access" request type, select **Edit fields** to change which request form fields show up in the customer portal. These

simplified fields help customers understand what information they need to provide when submitting a request.

- The "Summary" field should already be displayed in the **Visible fields** section. Click the "Summary" display name to rename it to "What do you need?", and add more field help details as shown:

Select **Update** when finished.

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

- 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 label you can assign to each request type. Your request types are organized into tabs based on these group labels in the customer portal. To edit groups:

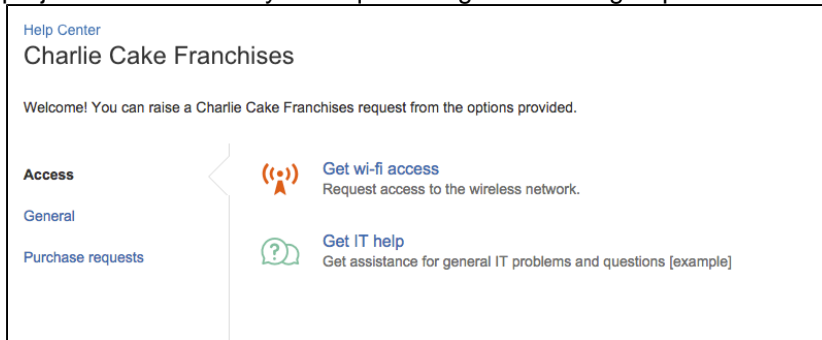
- Go back to the **Project administration > Request types** screen. You should see the following groups for each request type:

Request Type	Group
"Purchase a new monitor"	Purchase requests
"Get wi-fi access"	Access, General
"Get IT help"	General
"Request a new account"	General

- In the Groups column, hover over and click the existing groups edit them.
- Add the Access group to the "Request a new account" request type and select **Update**. This request type will now appear on two tabs in the customer portal, making it easier for customers to find.

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

- Go back to your project and open the customer portal link from your project sidebar to see your requests organized into groups:



Create a request from the customer portal

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

Excellent work! You now have four request types and another test issues 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

- Setting up your service desk
- Creating service desk request types
- Making queues for your service desk teams
- Adding service desk agents
- Customize your service desk channels
- Bring your service desk to the next level
- Introduce customers to your service desk

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 (e.g. "Unassigned issues"), but let's go ahead and create three new queues for your team:

1. From your service desk project sidebar, select **Queues**.
2. Select **New queue** and name your first new queue "Access requests".
3. Define the issues you want to appear in this queue by selecting the following drop-down menus: **Type** (select "Access"); **Status** (select "Waiting for IT Team"), and **Resolutions** (select "Unresolved"):

4. Select the following columns 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 (e.g. "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 purchases" for purchase requests that have been successfully resolved.

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

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.

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](#)

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

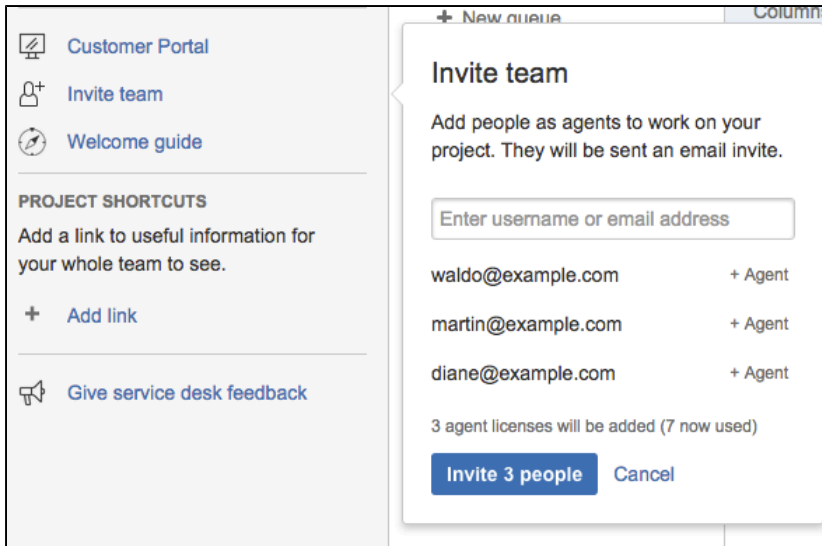
- Service Desk Customers who create requests via email or the customer portal
- Service Desk Team who view and respond to these requests

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

Add your agents

Project administrators can only 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**. Your updated agent license count is displayed:



3. These agents will be automatically added to the Service Desk Team role and assigned a JIRA Service Desk license.

Assign issues to agents

Your agents will generally work out of specific queues that have issues are automatically triaged into. 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 will appear in his personal queue.
4. Assign another test issue to Diane.

Add your customers

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

1. From your project sidebar, select **Customers**.
2. Select **Invite customers** in the top right corner and enter one or more email addresses.
3. When you select **Send invites**, invited customers will 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

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 administration > 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 for this Customer Portal**.
4. Save the sample image below and 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 administration > Email settings**.

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
```

You can choose to:

- 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

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

- [Service Level Agreements \(SLAs\)](#)
- [Service desk reports](#)
- [Knowledge base integration](#)

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 both the customer-facing request and the internal issue. Your agents can also view SLA goals by going to **Reports > 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 administration > SLAs.
2. Select **New Metric** to create a new SLA metric for your service desk project.
3. For more information, check out [Setting up service level agreements \(SLAs\)](#).

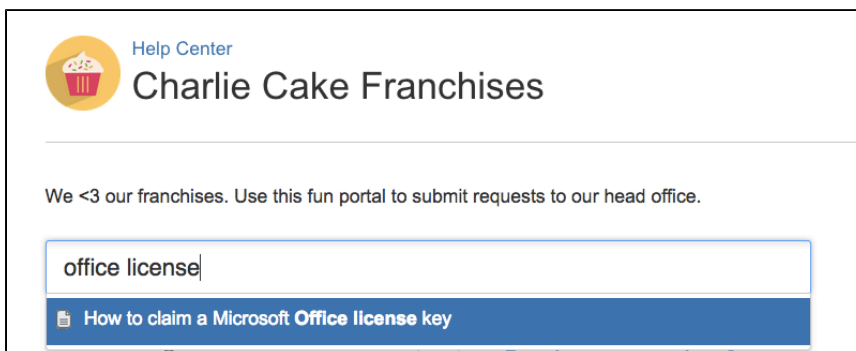
Track your team's success with reports

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](#).

Increase self-service with knowledge base integration

By connecting Confluence to your service desk project, you can help customers help themselves. Your customers can search for solutions in the self-service customer portal before they even create a request:



Your agents can also take advantage of knowledge base integration by selecting **Create KB article** directly from an issue and saving their customer responses as articles for future reference:

CCF-3 Return to queue

Need to access bakery wifi

Edit Comment Assign More Reopen issue Admin Create KB article Export

Details

Type: Access Status: **RESOLVED** SLAs: 3:59 Time to resolution within 4h

Priority: Medium (View Workflow)

Component/s: None Resolution: Done

Labels: None

KB articles will be a good resource for new agents in your service desk project and will help prevent existing agents from having to create the same response over again for related issues types.

1. In your service desk project, select **Project administration > Confluence KB**.
2. Choose "Link to a knowledge base" to select the Confluence application and space to link your service desk project to.
3. Check out [Serving customers with a knowledge base](#) to learn more.

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

Next

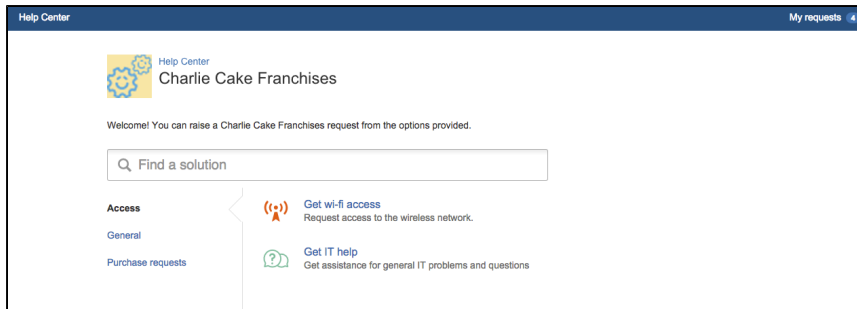
Introduce customers to your service desk

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

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 do, simply email them a request directly and keep the conversation going directly 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](#).

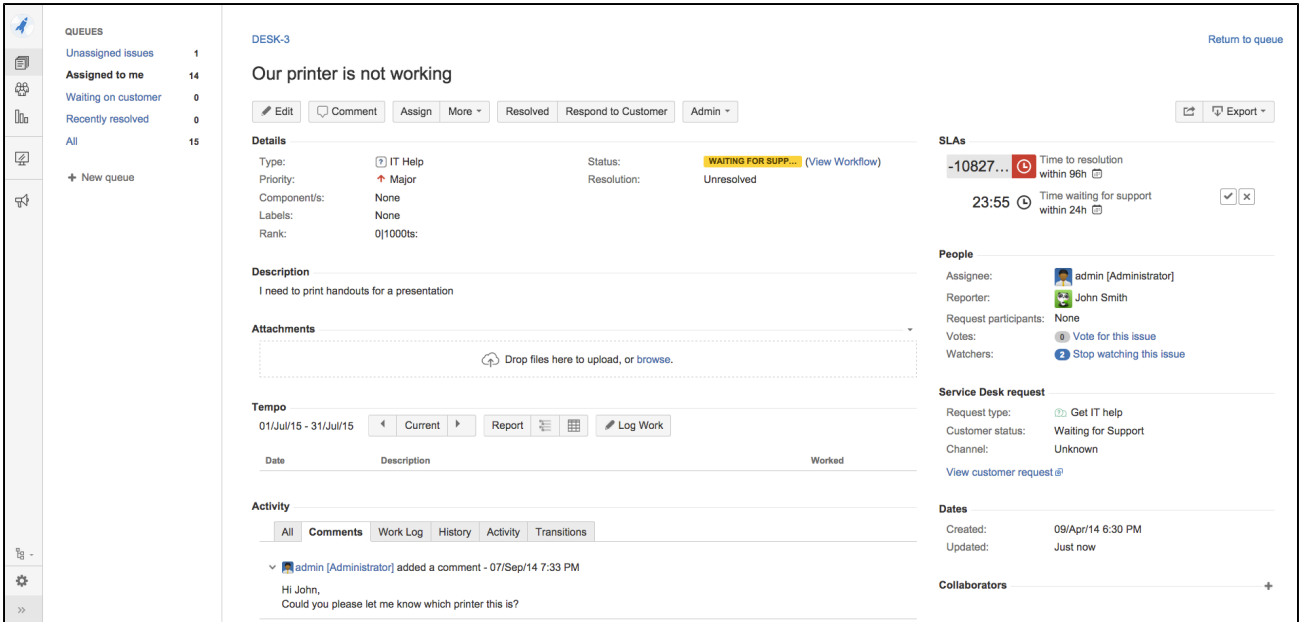
Getting started for service desk agents

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

[[Navigate your workspace](#)] [[Work on customer issues](#)] [[Capture knowledge](#)]

Navigate your workspace

Open JIRA Service Desk in your web browser. Take a few minutes to become familiar with the layout:



Queues



Queues display issues submitted by your customers. These issues appear in queues configured by your administrator.

Customers List



On the Customers tab, you can search for existing customers in your service desk project, invite new customers (if public signup is enabled), and see how many issues each customer has submitted.

Customer Portal



The customer portal link lets you see and interact with your service desk from a customer's perspective.

Reports



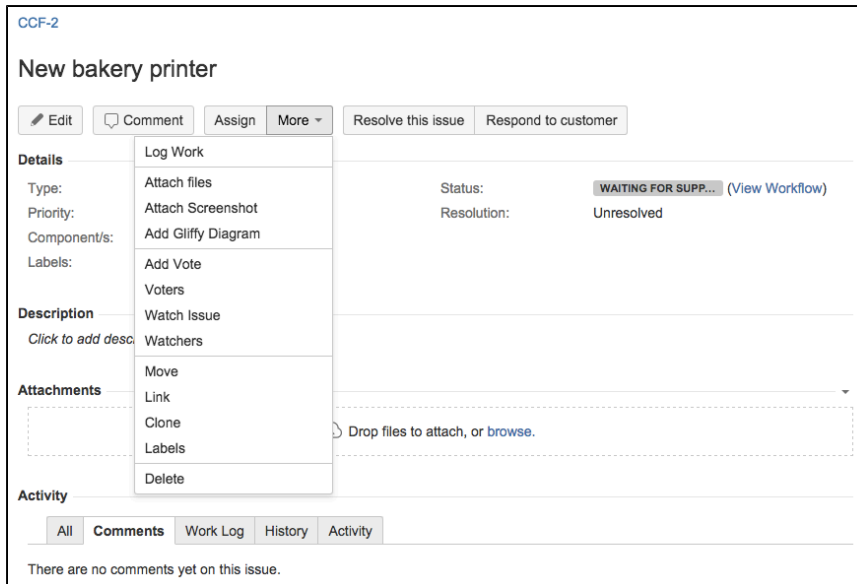
Reports display your team's workload and the Service Level Agreement (SLA) goals configured by your administrator. You can also view any preconfigured service desk reports on this tab.

Work on customer issues

Your administrator has already set up customized queues to help organize incoming customer requests. Please contact your administrator if you need to change a queue's configuration or add a new queue.

Open an issue

1. Select **Queues** and choose a queue to work from (e.g. Assigned to me).
2. Open a customer request by clicking the issue Summary or Key.
3. In addition to being able to edit and comment on a request, you can view a list of actions from the **More** menu. Hover over each action to display a brief explanation:



Leave a comment

1. Review the issue and perform the needed task (e.g. grant the customer wi-fi access). Then select **Respond to Customer**. You'll see a pre-populated customer greeting that you can edit with your full response and then preview. The message you type here will appear in the notification sent to the customer.
2. Use the **Internal comment** tab to write your own note or to include another colleague on the issue by using the "@" mention" feature (type @username) and writing your comment.

Attach a file or image

1. Copy and paste, or drag and drop, your attachment anywhere on the issue screen. You'll see the **Add attachments** dialogue pop up.
2. Add a comment and select **Share with customer** to send the attachment and comment to the customer, or **Add to issue only** to restrict the attachment and comment to internal users (e.g. other agents).

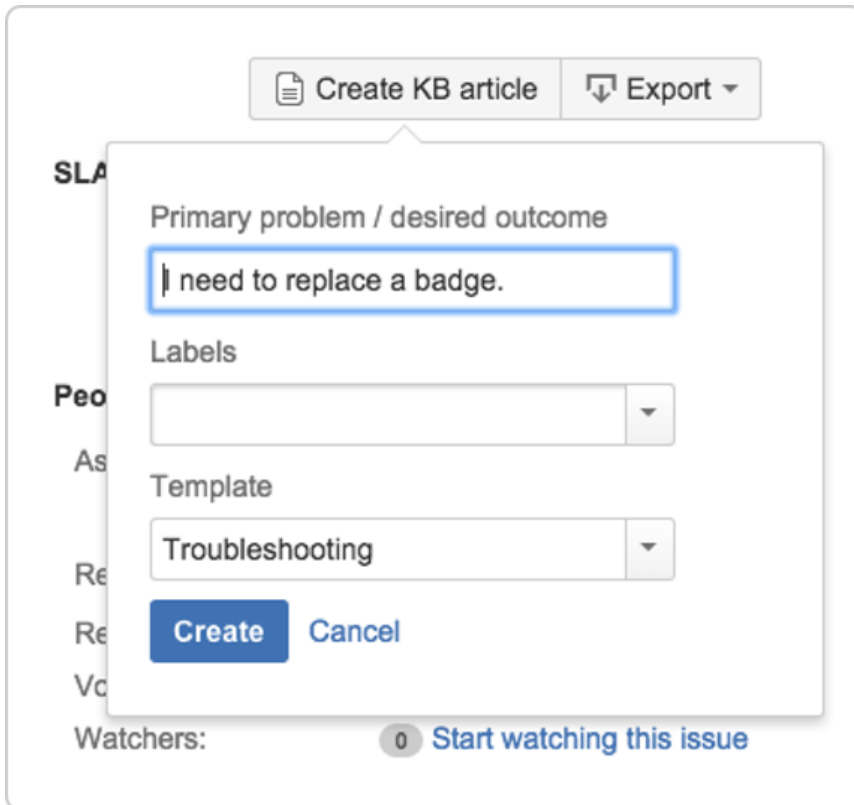
Resolve an issue

1. Once the customer's request is completed, select **Resolve this issue**.
2. Select a resolution (e.g. Done) and add any further details for the customer or your internal team. All participants on the issue will be notified of its resolved status.

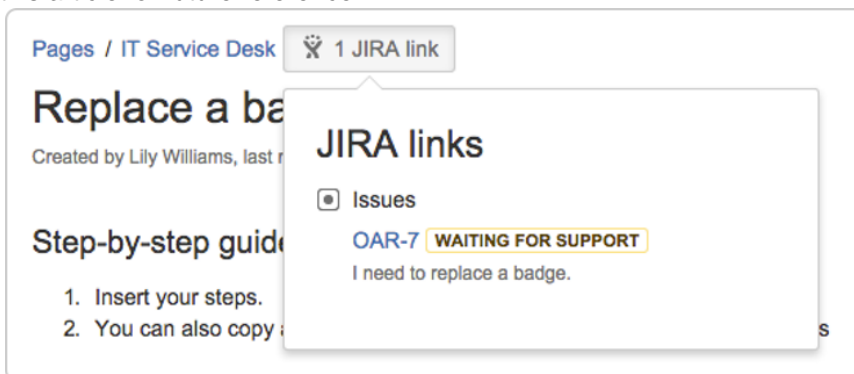
Capture knowledge

If your administrator has linked your service desk with a Confluence space, you can capture your response as a knowledge base article. You can then easily reference this article when responding to a similar issue in the future. KB articles will also appear in the customer portal, directing customers to relevant information before they even finish submitting their requests.

1. Click the **Create KB article** to enter the primary problem/desired outcome (or page title) and select the page template (How-To).



2. Fill out the How-To template and save the page in Confluence. You will see that your issue is linked to this article for future reference.



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.

- If you're new to JIRA Service Desk, check out [this tutorial](#)
- If you're familiar with JIRA Service Desk, use the search bar below to find any needed information

Search the topics in 'Administering service desk projects':

Automating your service desk

What is automation?

If you find your team stuck completing repetitive tasks or missing important request notifications, you can use automation to complete those tasks and send those alerts. Automation consists of rules that perform actions (e.g. alert agent) based on specific events (e.g. issue is created) and conditions (e.g. issue is high priority).

On this page:

- [What is automation ?](#)
- [How to use it](#)
- [Set up a rule](#)

How to use it

Your service desk project comes with preset rule templates that you can use to quickly set up automation. You can also create a custom rule (essentially a blank rule template). Here are some ways that automation can help your team and your customers, and the preset rule templates used to do so:

What automation does	Rule template used
Alert a member of your team when a customer submits an urgent request	Be aware of urgent issues
Alert your team lead when a serious issue is about to breach one of your SLAs	Keep on top of SLAs
Let your customers know when to expect a response from your team based on the priority of their ticket	Set customer expectations
When a customer comments on a ticket, transition it to "Waiting on Support"; when your team comments, transition the ticket to "Waiting on Customer"	Transition on comment
When a customer comments on a closed ticket, re-open it so your team can followup with more information	Re-open on customer comment

Set up a rule

1. In your service desk project, proceed to **Project administration > Automation** and select **Add your first automation rule** (or **New rule** if you have previously created one).
2. Select a preset or custom rule template from the list and then select **Next**.
3. Edit the rule name and description as needed. The rule name will appear on the main automation settings page, so changing the name will help you more easily reference what each rule does.
4. Fill in the WHEN, IF, THEN fields. Use **Tips for customizing this rule** for suggestions on what to enter in these fields.
5. Select **Options** to set the "Run as" user, or the user who will appear to perform the rule's action on the service desk ticket:

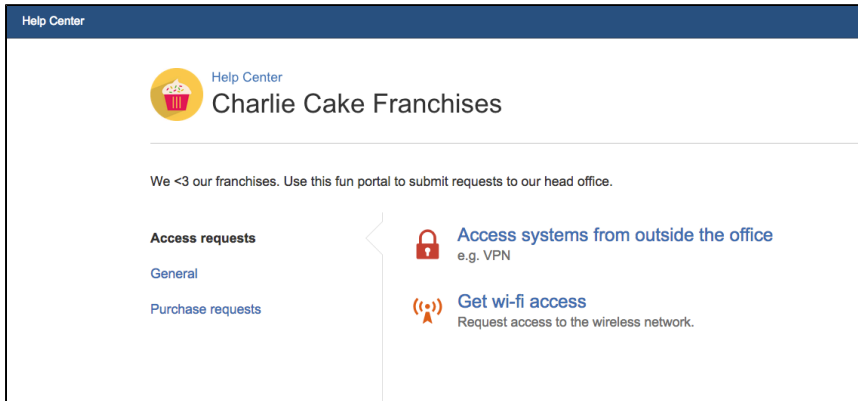


Note that the project default user can be set on the main automation settings page. Certain rules, such as those based on an SLA, cannot be run as the user who triggered the rule. You will simply be unable to select this option when that's the case.

6. In **Options**, check "Allow this rule to be triggered by other rules".
This option is useful if you have a rule that results in a comment from your team (e.g. Set customer expectations), and want that comment to trigger another rule that transitions the issue back to the customer (e.g. Transition on comment).
7. Select **Save** and you're done!

Configuring the customer portal

Your service desk project comes with a customizable customer-facing site (what we call the Customer Portal) where your service desk customers can submit and track requests. Customer Portals for multiple service desk projects can be accessed in the global help center. In your service desk project, select the **Customer Portal** link at any time to check out what your customers see when they want to contact your team:



On this page:

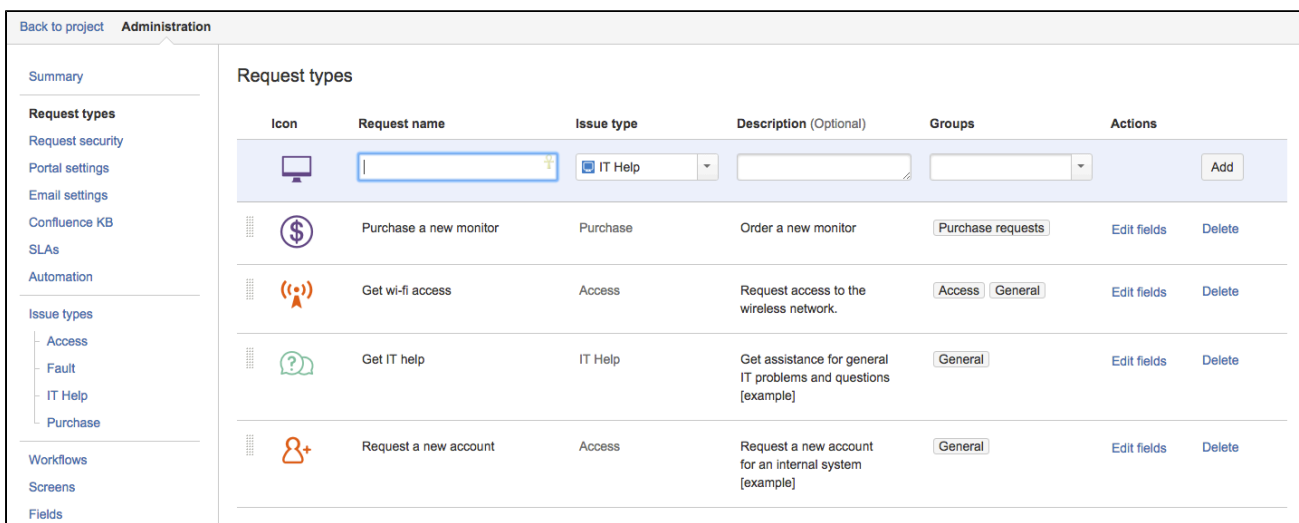
- [Setting up request types](#)
- [Branding your portal](#)
- [Managing access to your portal](#)
- [Viewing all portals in your Help Center](#)

If you have linked your project to a Confluence space, customers can also use this portal to search for help in relevant knowledge base articles before filing a request.

You must be a JIRA Service Desk Administrator or Project Administrator for this service desk project to make the following changes.

Setting up request types

You can create and edit the request types that customers fill out on the customer portal in **Project administration > Request types**:



Each service desk request type (e.g. Get wi-fi access) is based on an issue type (e.g. Access). You can organize your request types into groups (e.g. Access requests), which appear as tabs on the customer portal. To learn more about setting up request types and organizing request types into groups, check out [Setting up request types](#).

Branding your portal

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

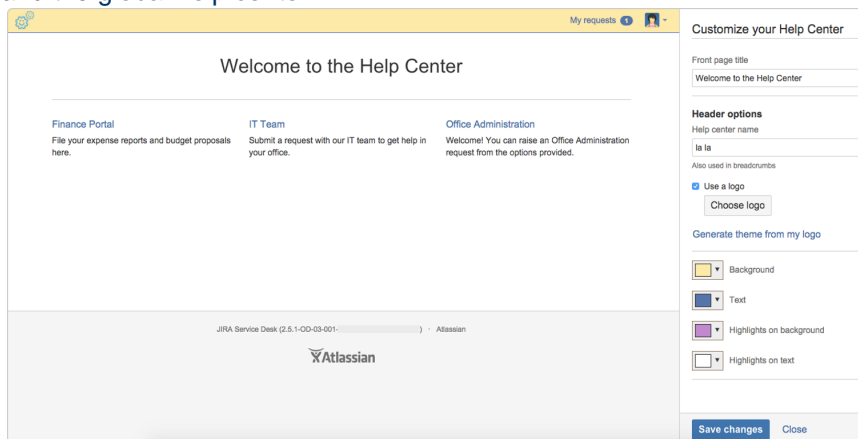
1. In **Project administration > Portal settings**, add a portal logo and a short description to familiarize customers with your service desk:



2. Follow the link (or select



> **Applications > JIRA Service Desk Configuration**) to brand your customer portal header in a live preview mode. The header and branding changes made here apply to all service desk project portals and the [global help center](#):



Managing access to your portal

You can open up access to your customer portal to allow new customers to create an account and submit a request for your team. If you don't want to have an open portal, you can restrict access to:

- Users who have an existing account for any other JIRA application, or service desk customers from all service desk projects
- 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](#).

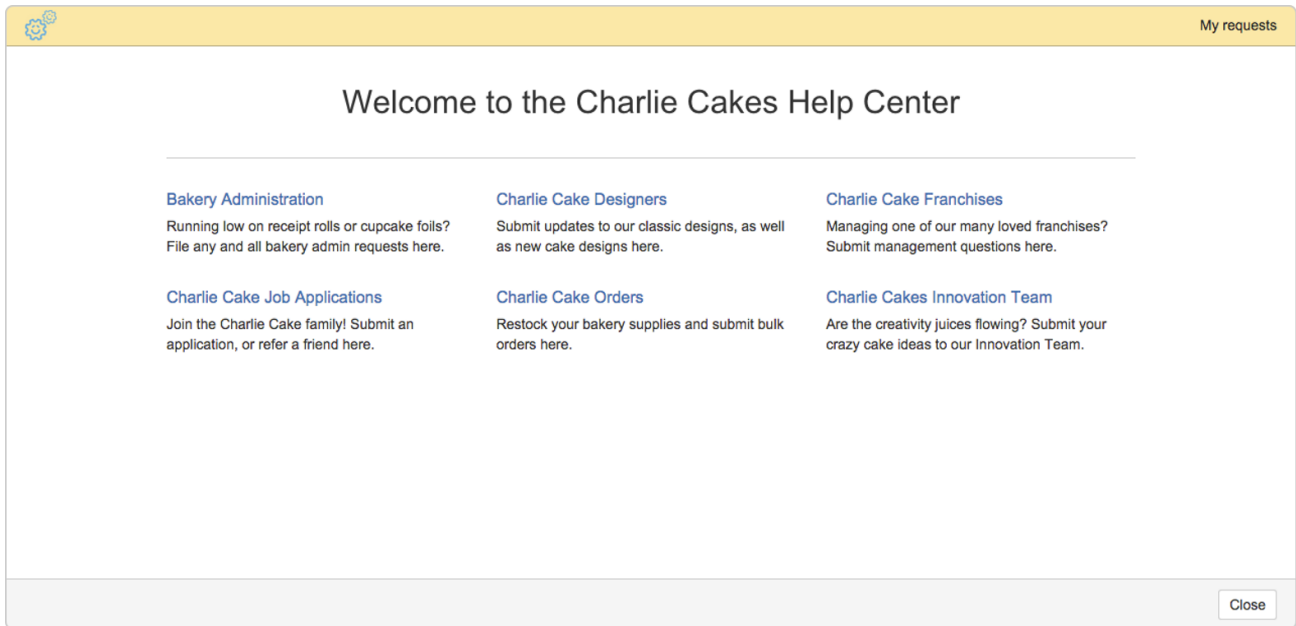
The customer portal integrates with [Atlassian Crowd](#), Atlassian's single sign-on (SSO), authentication, authorization, application provisioning, and identity management framework. For information about integrating with third-party SSO providers, please see [this page](#).

Viewing all portals in your Help Center

If your company uses multiple service desk projects (e.g. an IT service desk and an office administration service desk), you can simply 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_add`

ress>:<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, proceed to



> **Applications > JIRA Service Desk Configuration.**

Configuring service desk notifications

When customers submit a request through the customer portal or by email, they receive JIRA Service Desk notifications to keep them informed of the request's progress. Your team of agents and any internal users (e.g. JIRA Software developers) are updated through the default JIRA application notification scheme.


On this page:
<ul style="list-style-type: none"> • Who receives service desk notifications • Configuring service desk notifications • Setting the notification email type to HTML or plain text

Who receives service desk notifications

Your customers	Your team
<p>When a customer submits a request, they receive email notifications as the issue reporter when:</p> <ul style="list-style-type: none"> • the request is created • a comment is added to the request • another participant is added to the request • the request is resolved with a set resolution field • the request is reopened with a cleared resolution field 	<p>When agents work on an issue, they receive email notifications as part of the the project's notification scheme.</p>

Configuring service desk notifications

Two settings impact how notifications work for a service desk project:

- The system level **Notifications** setting controls the customer notifications for all service desk projects. This setting is enabled by default and can be disabled by going to  > **Applications** and selecting JIRA Service Desk **Configuration**.
- The project level **JIRA notification scheme** controls the agent notifications for a service desk project.
 - If the **Notifications** setting for JIRA Service Desk is enabled, agents will receive notifications as part of the default notification scheme. Customers will receive service desk notifications for all issues they're involved with.
 - If the **Notifications** setting for JIRA Service Desk is disabled, agents will continue to receive notifications as part of the default notification scheme; however, customers will only receive notifications if they are working on an issue in a public project.

Setting the notification email type to HTML or plain text

As an administrator, you can set the default email type for service desk notifications. 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 rely on software that requires plain text or use a plain text mail client, you can change your default setting to plain text and apply this change to new and existing customers.

1. Choose



> **System**. Scroll down to the **User Interface** section and choose **Default User Preferences**.

2. Select **Edit default values**.
3. 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:
4. Under **Operations**, select **Apply**.
5. Select **Update** to finish applying the email preference to all user accounts.

Managing access to your service desk

When you set up your service desk project, you can open up access to allow anyone to sign up for an account and raise a request for your team, or you can restrict your service desk project to a specific list of customers.

On this page:

- [Change your restricted access settings](#)
- [When to open your service desk](#)
- [When to restrict your service desk](#)

Change your restricted access settings

1. In your service desk project, go to **Project administration** > **Request security**.
2. Choose one of the options under "Who can raise requests". The security options are explained in more detail below.

When to open your service desk

As an example of when to open your service desk project, an IT service desk is usually open to all the employees in an organization, so everyone can access it and create requests. In this open service desk scenario, customers can create an account on the customer portal or email requests to your service desk email channel to have an account created automatically.

To open your service desk, your administrator must first enable [public signup](#). You can then select the first option under "Who can raise requests" on the request security page.

When to restrict your service desk

If public signup is not enabled, you have the option to partially restrict your service desk to service desk customers and JIRA application users from any type of project. This option prevents people from signing up themselves and emailing your linked email channel without an existing account.

To partially restrict your service desk, select the second option under "Who can raise requests" on the request security page.

In another example, for a service desk that handles contractors' leave requests, you might want to make it only available to your contractors so that the rest of your staff do not get confused about where to put in leave requests. Service desks like this one are restricted service desks and only customers you add to your project's Customers list can create requests.

To restrict your service desk, select the third option under "Who can raise requests" on the request security page.

Receiving requests by email

If your customers prefer to open and work on requests from the comfort of their email inboxes, you can enable email requests to receive all customer requests in JIRA Service Desk. Enabling email requests will help your team focus on your customers, instead of having to worry about missing requests or checking multiple inboxes.

Here's how it works:

- A customer emails a request to your linked service desk account
- An agent comments on the request in service desk, which sends the customer an email notification.
- The customer replies to these email notifications until the request is resolved. All customer replies are automatically added as comments on the corresponding issue in service desk.

On this page:

- [Before you start](#)
- [Adding an email account](#)
- [Choosing a request type](#)
- [Verifying your linked email account](#)
- [Preparing customers for email greatness](#)
- [Email usage notes](#)

Before you start

- Make sure you have the Administer Projects permission.
- Enable [public signup](#) or manually [add customers](#) to your service desk project to ensure that you receive new customer 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](#).

Adding an email account

Open your service desk project and proceed to **Project administration > Email settings**. Turn on email requests by selecting the **Turn it on** button. Easy enough.

- If you're using JIRA Service Desk Cloud (hosted by Atlassian), you will see a default email address based on your site's URL. You can use both this existing email address and a new linked email account for your service desk email channel.

Select **Add email account**. Choose your email service provider and enter the requested details before selecting **Next**.

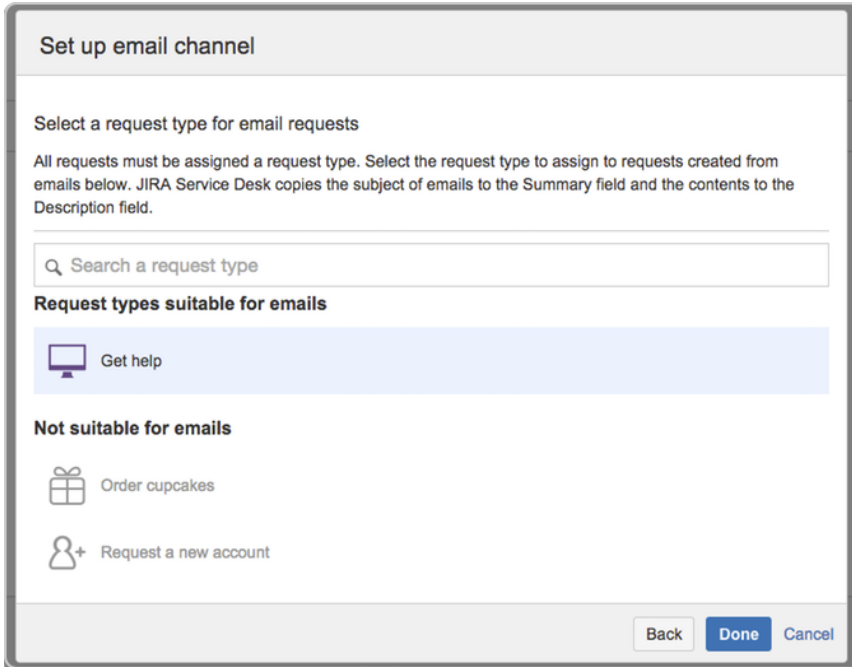
If you have two-step verification enabled for your Gmail account, you will need to set up an application-specific password.

Choosing a request type

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

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

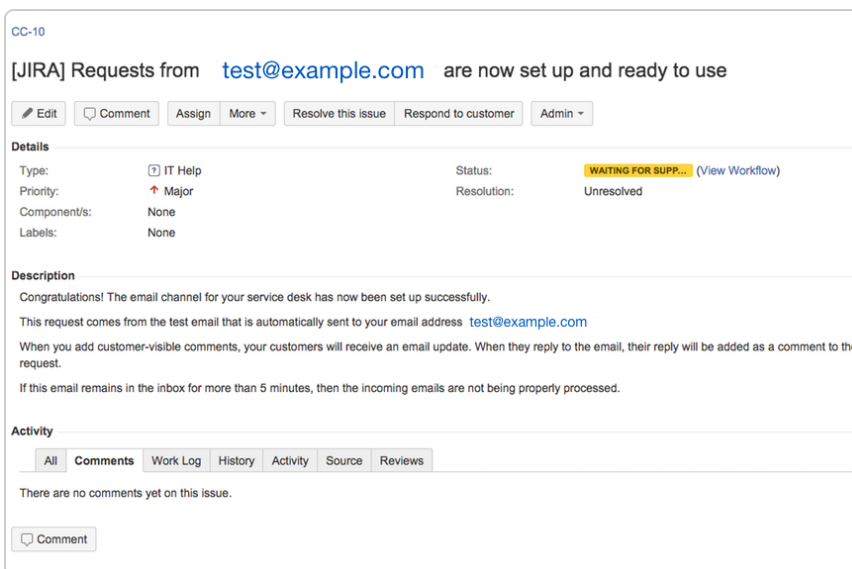
In order to use the email channel, you therefore need to have at least one request type in your project with the Summary and Description fields – we call these types of requests "suitable for emails". Associating email requests with a suitable service desk request type ensures that the emails are successfully filtered into your service desk queues. In this example, we have one suitable request type ("Get help"):



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

Verifying your linked email account

Once you have chosen a suitable request type, JIRA Service Desk will send a test email and create 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 will now appear as service desk requests in your project. For more information about what 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. Been received after your email account and service desk project have been successfully 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. Been received after your email account and service desk project have been successfully 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.

Preparing customers for email greatness

Before sharing your linked email account with your customers, you will want to confirm whether you have an [open or restricted service desk](#). New customer email requests can bounce if you have a restricted service desk, and that's no fun. Expand the statement that applies to you below to make sure your customers are ready to use your new email channel:

▼ [I have an open service desk...](#)

Great! New customers can create requests right away by emailing your linked service desk email account. A corresponding customer account will be created based on the new customer's email address. Customer accounts do not count towards your service desk license.

▼ [I have a restricted service desk...](#)

Email requests will not be processed if your customers don't have existing service desk accounts. Simply create [new customer accounts](#) (or send customer invitations) before telling new customers to email your service desk.

▼

Email usage notes

- Note that 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

Now that you have [set up your email channel](#), you can control when JIRA Service Desk connects to your mail server and filters relevant emails into your service desk projects. You can also view logging information directly to check on the status of your mail server connection.

Managing global mail settings

On this page:

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

There are two global mail settings - email puller and email processor - that are used only by JIRA Service Desk and do not impact any email settings you have set up for other JIRA applications. 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 (e.g. to remove auto-replies and spam) using information stored in the database.

You can access these settings by going to



> **System** > **Global mail settings**.

Managing the email channel for multiple service desk projects

JIRA administrators can get an overview of all the service desks in the system that use the email channel and the email accounts linked with them.

1. Choose



> **Applications**. Scroll down to the **JIRA Service Desk** section and choose **Email settings**.

From the **Email settings** page, you can also check the connection email processing statuses of each linked email account. Note that logging information older than 6 months is deleted daily.

1. Choose



> **Applications**. Scroll down to the **JIRA Service Desk** section and choose **Email settings**.

2. Under **Actions**, click **View log**.

3. Click the **Connectivity log** or **Processing log** tab to view the corresponding log details.

Troubleshooting issues with the email channel

This page contains information about the errors and problems that you might run into when setting up the email channel for your service desk.

Checking the connection

To troubleshooting 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 settings**.

2. Select **Test**.

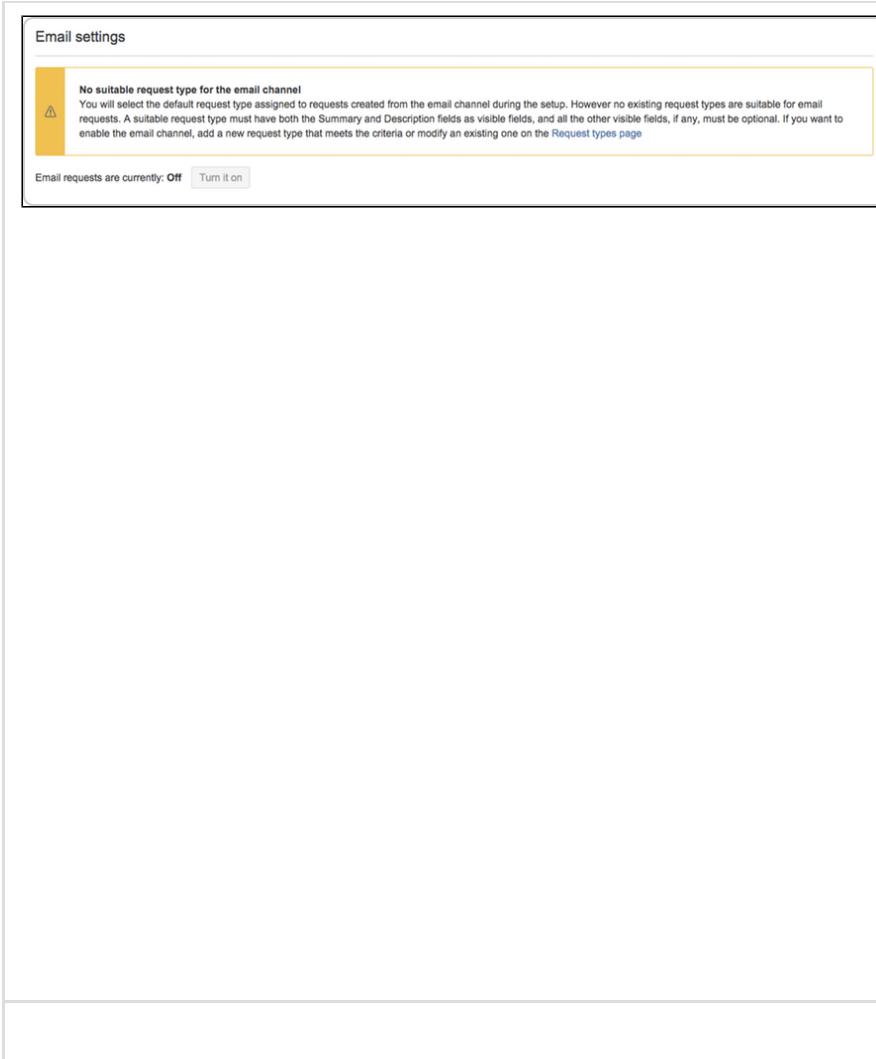
Resolving errors

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

Symptom	Description and resolution
---------	----------------------------

<p>Gmail</p> <p>Error message:</p> <p>Unfortunately JIRA Service Desk 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>JIRA Service Desk checks email accounts every minute. Gmail 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"> • Create an application-specific password for your JIRA Service Desk account in your Google Account settings. Details can be found in the Google Account Help page.
<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.
<p>Gmail accounts, POP3</p> <p>Requests are created from archived messages.</p>	<p>When JIRA Service Desk checks your email accounts for new messages, it looks in the inbox folder.</p> <p>Gmail uses labels to classify messages into categories and has these folders: Inbox, Sent Mail and Trash (or Deleted Items). 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 labels and therefore still brings them in as requests.</p> <p>To resolve this:</p> <ul style="list-style-type: none"> • Use IMAP.

<p>Customers send emails to create requests, but no requests are created and customers do not receive any notifications.</p>	<p>This problem could be due to one or more of the following causes:</p> <ul style="list-style-type: none">• 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 <i>managed</i> service desk.• The default request type for the email channel is unsuitable for the email channel. <p>▼ Learn more</p> <p>A suitable request type for the email channel must be selected in the Summary and the Description field as visible fields. Any other fields must be optional ones.</p> <p>To troubleshoot the issue and resolve it:</p> <ol style="list-style-type: none">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.
--	---



This message appears on the Email settings page and prevents you from turning on the email channel.

To resolve this:

- Head to **Project administration > Request types**
- Add a new request type (or choose an existing one)
- Select **Edit fields**
- Make sure both the Summary and Description fields are added and marked as **Required = Yes**. You also add an additional Attachment field with **Required = No**.
- Save the request type and head back to **Project administration > Email settings**

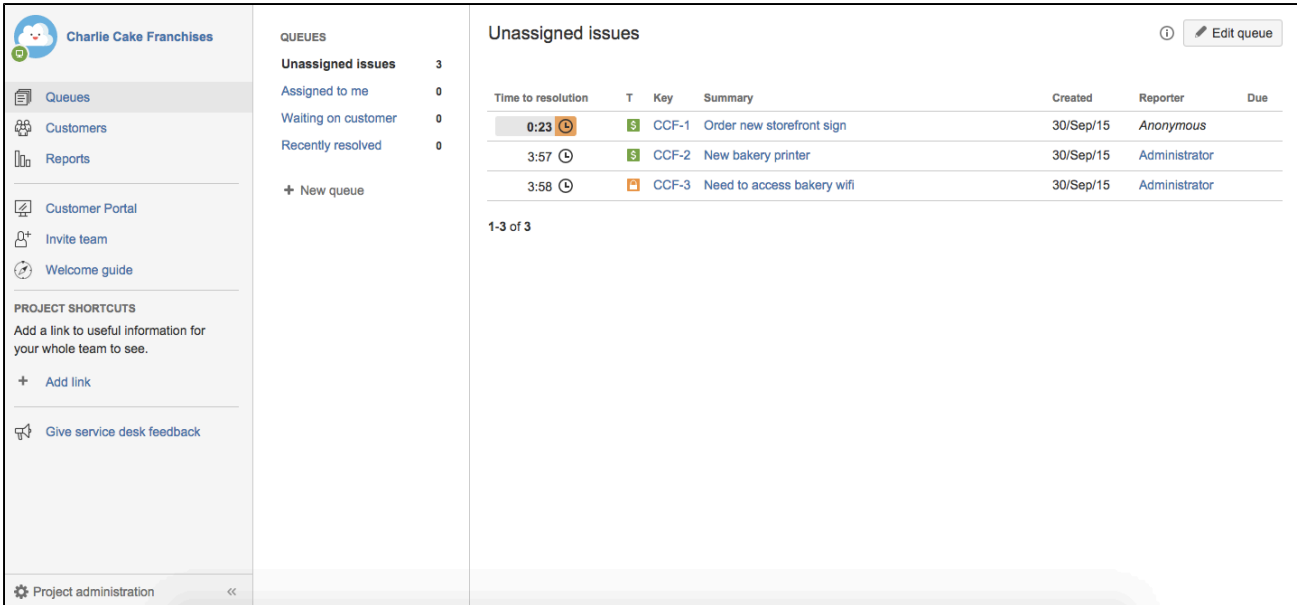
Setting up queues for your team

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.

On this page:

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

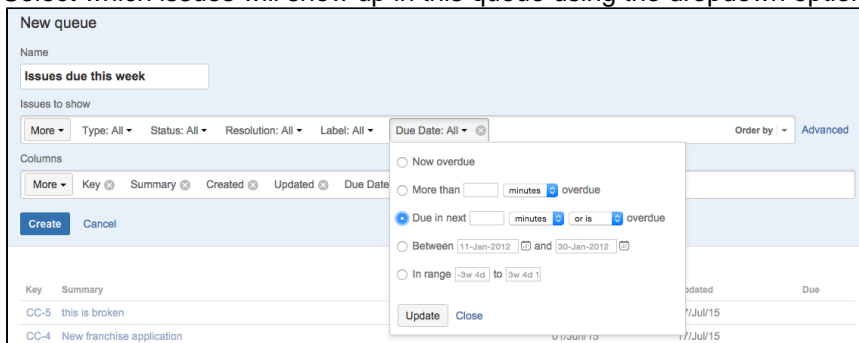
You can log in as a service desk administrator to configure queues on the aptly named **Queues** tab in your service desk project:



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 can create a new queue:

1. In the **Queues** tab of your service desk project, select **New queue**.
2. Name your queue using language your team will understand (e.g. Issues 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.

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.

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 must be a JIRA Service Desk Administrator or Project Administrator to configure request types and workflow statuses.

On this page:

- [Setting up request types](#)
- [Organizing request types into groups](#)
- [Customizing the fields on a request type](#)
- [Customizing the workflow statuses for a request type](#)
- [Hidden fields and unsupported fields](#)

Setting up request types

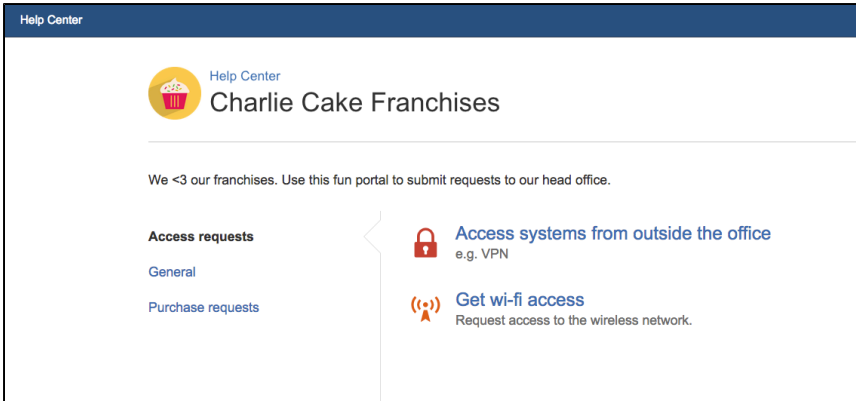
Each request type in a service desk is based on a JIRA issue type. Open **Project administration > Request types** to manage your project's request types:

The screenshot shows the 'Request types' configuration interface. It features a table with columns for 'Issue type', 'Request name', 'Description (Optional)', 'Groups', and 'Actions'. Two request types are visible: 'Access' (with a lock icon) and 'Access' (with an envelope icon). Callouts point to various elements: 'IT Help' in the issue type dropdown, the 'Request name' field, the 'Description' field, the 'Groups' dropdown, and the 'Edit form' and 'Delete' buttons in the actions column.

Note that a single issue type can be the basis for many different request types (for example, the "Purchase" issue type serves as the basis for both the "Request new hardware" and "Request new software" requests).

Organizing 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 (e.g. Access requests) in order for the groups to appear in the customer portal:



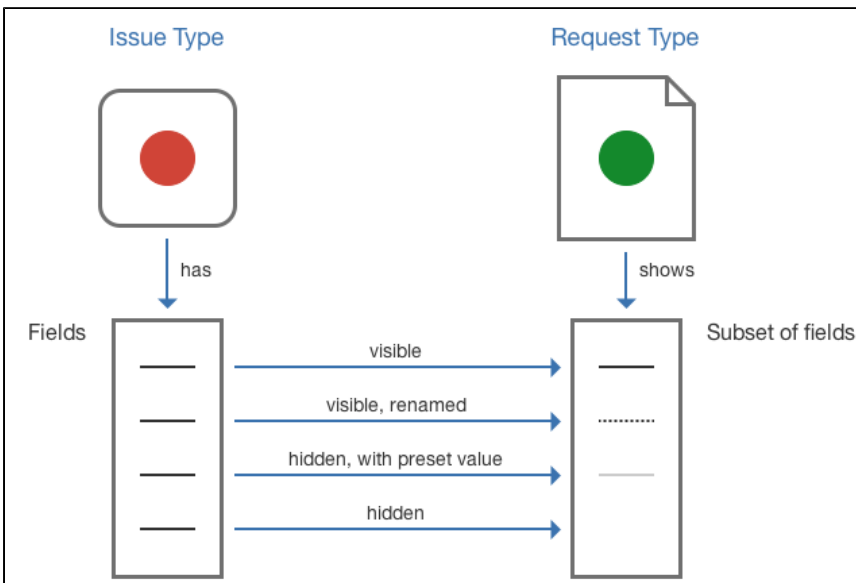
Administrators and Project Administrators can manage request type groups in **Project administration > Request types**. Simply select a request type and choose an existing group from the dropdown menu or create a new one. Note that request type groups are unique to a service desk project.

Tips:

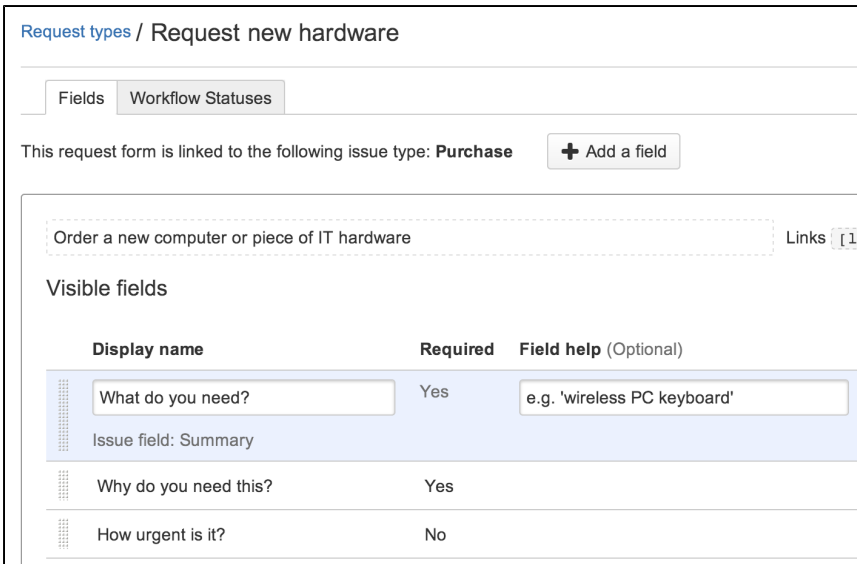
- You can drag and drop request types to re-arrange them on the customer portal.
- As groups are displayed in alphabetical order, you can prefix group names with numbers, e.g. *1 Access, 2 Service not working* to change the order in which they appear on the portal
- If you assign multiple groups to a single request type, the request type will appear on multiple tabs.

Customizing 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 field names to more customer friendly language. For example, the "Summary" field appears as "What do you need?" for customers.

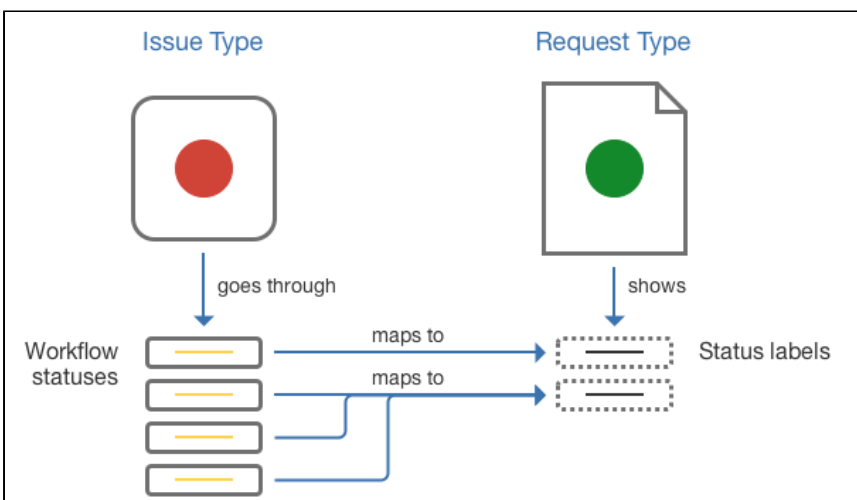


You can also keep fields hidden but available on the request type so that their value can be used for other processes. For more details about how different types of fields work in JIRA Service Desk, see [Hidden fields and unsupported fields](#).

If the issue type doesn't have the fields you need, you must [add a field](#) to the 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](#).

Customizing 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.

[Request types](#) / Request new hardware

Fields

Workflow Statuses

You can rename workflow statuses for this request type to make them more meaningful in the Customer Portal. If you set the same name for multiple statuses, these will appear as a single status to Customers. Customers will only be notified when there is a change in the status name.

Workflow status in JIRA	Status name to show customer
Resolved	<input style="width: 100%;" type="text" value="Finished!"/>
Waiting for Support	<input style="width: 100%;" type="text" value="Waiting for 3rd party"/>
Blocked Externally	<input style="width: 100%;" type="text" value="Waiting for 3rd party"/>
Waiting for Customer	<input style="width: 100%;" type="text" value="Need info from requester"/>
Untriaged	<input style="width: 100%;" type="text" value="Waiting for Support"/>

[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 [Configuring 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 administration > Workflows**.

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

These types of fields can't be added to a request type and won't appear 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 to be another one by going to Project administration > Email settings in your service desk. For more information about the email channel setup, see Receiving requests by email.</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.

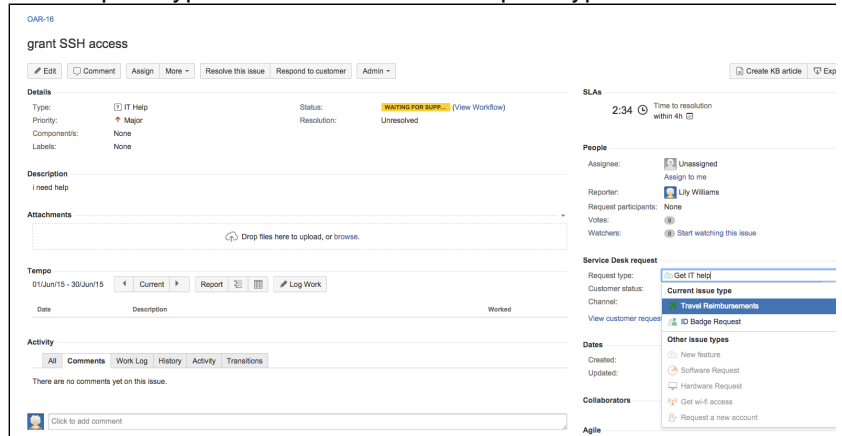
Request type displayed as "No match" in agent view.

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 desk 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 the third step, check **Change Customer Request Type** and select the request type that applies to this group of issues.

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 and transitions 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.

If you need to edit or create a more advanced workflow to match how your team or organization works, you can log in as a **JIRA Administrator** with global permission to access and create your workflow.

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 Managing workflow schemes 	Configuring workflow schemes
<ul style="list-style-type: none"> Import and export workflows Activate and deactivate workflows 	Managing your workflows
<ul style="list-style-type: none"> Adding custom events Configuring the initial status Working in text mode Configuring workflow triggers Using validators and custom fields Using XML to create a workflow Workflow properties 	Advanced workflow configuration

Setting up service desk users

In JIRA Service Desk, you can use groups to manage your agent license

count and and project roles to manage access to your service desk projects.

When you invite new agents to a service desk project, these users are automatically added to the Service Desk Team project role, and assigned a service desk agent license by being added to the jira-servicedesk-users group. When you invite customers to a service desk project, these users are not assigned a service desk license, but are similarly added to the Service Desk Customers project role and given restricted access to the customer portal.

To set up and manage users, you must be logged in as an **administrator**.

On this page:
<ul style="list-style-type: none"> • Overview of project roles • Managing service desk agents • Managing service desk customers • Involving JIRA Software and JIRA Core users • User management documentation

Overview of project roles

The following table summarizes the default service desk project roles and groups:

Project Role	Group	Details
Administrators	administrators	<p>Users who administer a service desk project. In addition to what the Service Desk Team can do, Administrators can:</p> <ul style="list-style-type: none"> • Add agents and customers to a service desk project • Remove agents and customers from a service desk • Configure request types, request security, and the email channel • Customize the customer portal • Create and edit reports and SLAs to track progress • Connect a Confluence knowledge base
Service Desk Team	jira-servicedesk-users	<p>Users who work on customer requests. Agents assigned to this project role can:</p> <ul style="list-style-type: none"> • View queues, reports, SLA goals, and the customers list • Create and edit issues in the service desk project view and the customer portal • Add, edit and delete customer-facing and private comments on issues • Manage content in a connected knowledge base <p>Other JIRA application users can be assigned to the Service Desk Team role, but they will have limited project access.</p>

Service Desk Customers	By default, customers cannot be added to groups and assigned a license	<p>Users who create requests through the customer portal or by email. Customers can't log in to JIRA applications and don't have access to the service desk project view used by administrators and agents. Customers can:</p> <ul style="list-style-type: none"> • Create, comment on, and track requests through the customer portal • Create and comment on requests via email • Add comments and attachment to requests • Add other participants to their own requests
------------------------	--	--

Managing service desk agents

Add agents to a service desk project

Easily invite new agents from your service desk project sidebar:

1. Select **Invite Team** and enter the agent's username (for existing system users) or email address (for new users). You'll see an updated JIRA Service Desk agent license count based on the number of new agents.
2. Select **Invite**.

Once invited, your new agents will receive an email with a link to your service desk project and be automatically added to the Service Desk Team project role and jira-servicedesk-users license group.

Remove agents

To remove an agent from your service desk project:

1. From your service desk project sidebar, go to **Project administration > Users and roles**.
2. Hover over the user or group you'd like to remove from the Service Desk Team project role and select



Note that removing an agent from your project does not revoke the agent's license. To free up an agent license, your administrator can proceed to



> **User management** and remove the agent's JIRA Service Desk application access.

Managing service desk customers

Add customers to a service desk project:

You can manually invite customers, or [open your service desk](#) to allow customers to create their own service desk accounts.

To manually invite customers:

1. From your service desk project sidebar, select **Customers** and then select **Invite customers**.
2. Enter your customer's email address or invite multiple customers. Your customers will receive their new account details by email and will be added to the Service Desk Customers project role.

If your administrators have already enabled [public signup](#), you can open up your service desk project to allow new customers to create accounts on the customer portal or send requests by email, which creates an account for them automatically. To open up your service desk:

1. From your service desk project sidebar, go to **Project administration > Request security**.
2. Select "Anyone can sign up for a customer account" and select **Save**.

Remove customers

Removing customers from your service desk project depends on your project's request security settings, which you can check in **Project administration > Request security**.

If your request security settings are set to "Only people on my customers list can raise a request", you can remove customers with these steps:

1. From your service desk project sidebar, go to **Project administration > Users and roles**.
2. Hover over the user or group you would like to remove from the Service Desk Customer role and select



If your request security settings are set to "Anyone can sign up for a customer account", or "Only people who have an account", your administrator will need to deactivate customer accounts by going to



> User management.

How restricted customer access works

The permissions assigned to customers are granted to the **Service Desk Customer - Portal Access** security type instead of the Service Desk Customers role. The Service Desk Customer - Portal Access security type gives people access to the customer portal only. This security type checks the Service Desk Customers role to determine which users are customers. The security type and the role essentially work hand in hand to make sure that customers get the permissions they need to use the customer portal and cannot access JIRA applications.

For example, if you want your customers to be able to create requests through your customer portal, grant the Create Issues permission to the Service Desk Customer - Portal Access security type, not the Service Desk Customers role.

Why does the security type have more permissions than what customers can do?

In the standard permission scheme, the Service Desk Customer - Portal Access security type has more permissions in place than the functionality available for customers to use. For example, the security type has the Edit Own Comments permission, but customers cannot do this on the Customer Portal. This is because JIRA Service Desk built the functions that we think are the most commonly used by service desk customers. We will evaluate the feature requests and expand the functions gradually. With the permissions in place now, future functionality additions to the Customer Portal will be easier because you will not have to modify permission schemes to make use of new functions in most cases. You can join the discussion on new features at our issue tracker: [844 issues](#).

Involving JIRA Software and JIRA Core users

JIRA application users without a JIRA Service Desk license (e.g. JIRA Software developers) can assist your team of agents on issues by working in the unlicensed view of a service desk project.

These users can:	<ul style="list-style-type: none"> • View issues, comments and attachments • Add and delete their own attachments • Add and delete their own internal comments • Watch and vote for issues
These users cannot:	<ul style="list-style-type: none"> • See service desk queues, reports, and customers list • Leave an external comment viewable by a service desk customer • Log work on a service desk issue • Transition a service desk issue • Be assigned to a service desk issue

As an example of how to involved other JIRA application users, Diane, a JIRA Service Desk agent, mentions Tony, a JIRA Software developer, on an issue to ask for advice on an exception in a customer's log file. Tony looks at the issue, views the attached log file, and posts an internal comment for Diane with his analysis. Diane then conveys the findings to the customer to resolve the issue.

Add JIRA Core or JIRA Software users to your service desk project

1. In your project, go to **Project administration > Users and roles**.
2. Select **Add users to a role**.
3. Search for the users you'd like to add and choose the Service Desk Team role.
4. Select **Add**.

JIRA Core and JIRA Software users in the Service Desk Team project roll will have restricted access to your service desk project and will only be able to see issues with which they're involved. Adding JIRA core and JIRA Software users to the Service Desk Team project role will not assign them to a service desk agent license.

Remove JIRA application users

1. In your project, go to **Project administration > Users and roles**.
2. To remove users from your service desk project only, simply hover over the user or group you wish to remove and select



To remove or deactivate users, your administrator will need to proceed to



> **User management**.

User management documentation

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

Documentation	Details
User management	Learn more about adding and removing users, managing users with groups, and managing access to JIRA applications.
Managing project roles	Learn more about adding and removing project roles, and managing project role membership.
Enabling public signup	Open up your service desk project to allow customers without an account to sign up on the Customer Portal and send requests by email.
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.

- [Viewing and editing project role members](#)

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



> 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.

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 JIRA issues and service desk requests directly in HipChat when someone on your team mentions them.

- [Connecting projects to rooms](#)
- [Invite users](#)
- [Discuss issues in rooms](#)
- [JIRA 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 logged in as an Administrator or a Project Administrator.
2. Choose



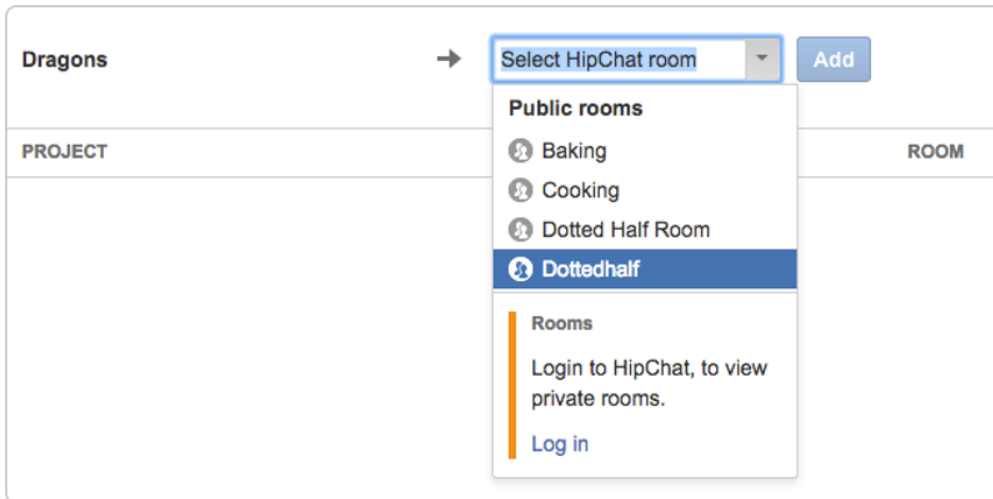
> Projects.

3. Select a project.
4. In the Project Administration 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 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.

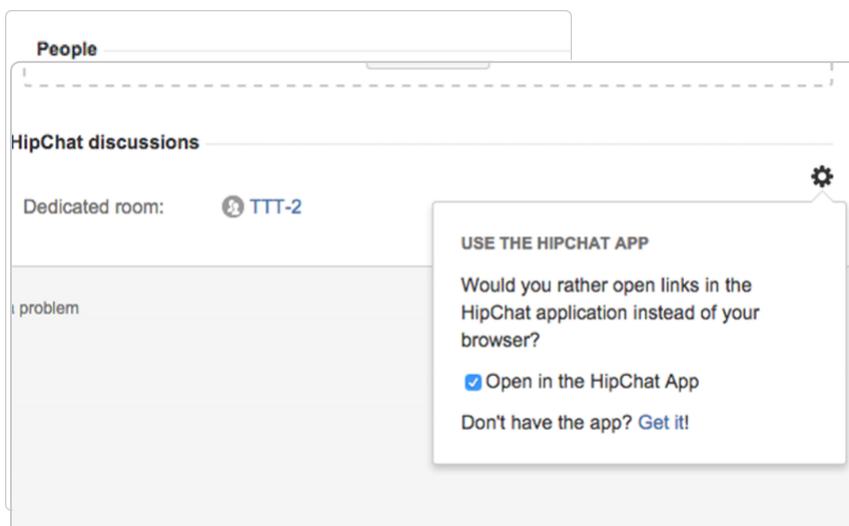
Invite users

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

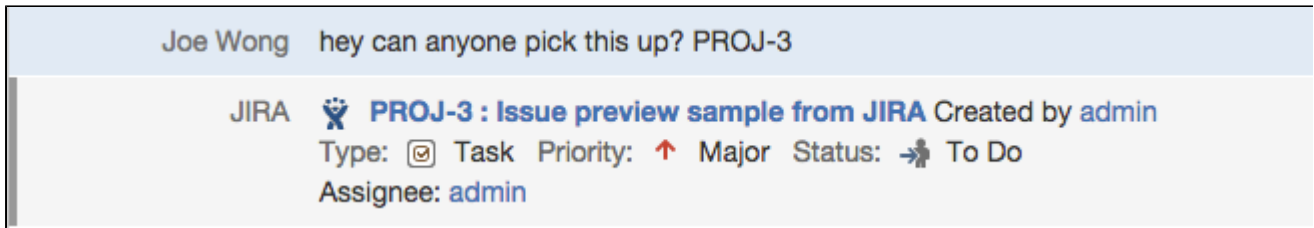
You can focus your discussion by creating or selecting a HipChat room to discuss a JIRA issue. When JIRA is integrated with HipChat and you are in the JIRA issue screen, you can select to "Create a room" or "Choose a room" in the **HipChat discussions** panel. This will associate the current issue and 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 JIRA issues screen, select the cog icon in the HipChat discussion to enable opening links in the application.

JIRA issue preview

With JIRA 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 a JIRA issue, without ever having to leave the discussion.



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 JIRA 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 Administration 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.**

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 administration 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_CSATENTRIES
- 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

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

- Working with issues
- Editing and collaborating on issues
- Attaching files and screenshots to issues

Tracking your work

- Keeping on top of SLAs
- Configuring dashboards

Serving your customers

- Raising requests on behalf of customers

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:

The screenshot shows the JIRA Service Desk interface for a project named 'Charlie Cake Franchises'. On the left is a sidebar with 'Queues' selected. The secondary sidebar shows a list of queues: 'Unassigned issues' (2), 'Assigned to me' (0), 'Waiting on customer' (0), and 'Recently resolved' (1). The main content area displays a table of unassigned issues.

Time to resolution	T	Key	Summary	Created	Reporter	Due
3:51	🕒	CCF-2	New bakery printer	30/Sep/15	Administrator	
3:51	🕒	CCF-3	Need to access bakery wifi	30/Sep/15	Administrator	

1-2 of 2

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:



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.

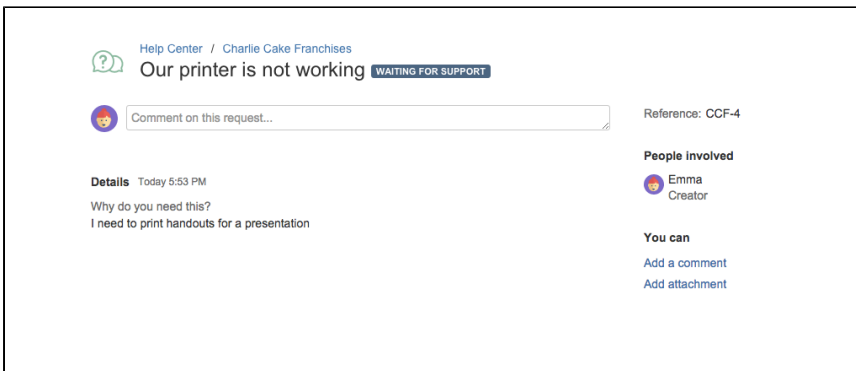
On this page:

- What is an issue?
- Next steps

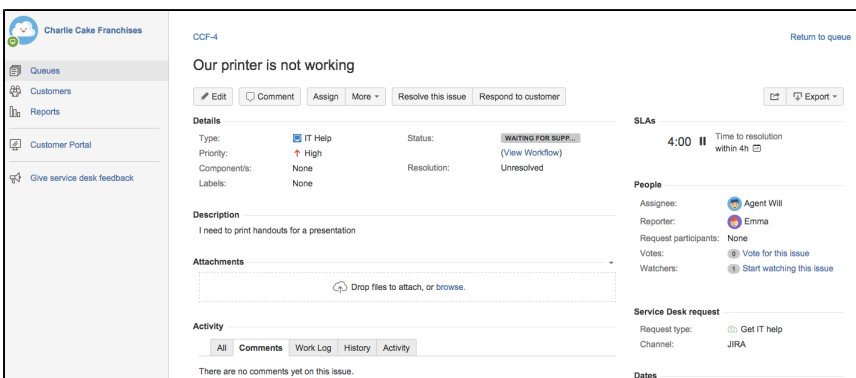
What is an issue?

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 represents a customer request (i.e. a helpdesk ticket), such as "Our printer is not working", which appears as follows in the customer portal:



As an agent, you then pick up the issue from the service desk agent view:



Next steps

Check out the following pages to reach issue ninja status:

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

Adding request participants

Customer requests are communications between the customer reporting an issue and the agent resolving that issue. To aid the resolution process, both agents and customers can add request participants. Request participants are additional JIRA Service Desk customers or other internal users who can be included in the resolution process.

On this page:
<ul style="list-style-type: none"> • Adding other customers • Adding internal users • About request participants

Adding other customers

Typically, you include other customers, to ask them for more information or to update them about the issue. Request participants can add comments and attachments to a request, and receive the same notifications from JIRA Service Desk as the reporter. Participants are able to see who else is involved in a request both on the Customer Portal and in email notifications. This makes it possible for them to work from their inbox. They can also add more participants, for example, other customers who may be experiencing the same issue and would like to be notified about the resolution.

To add request participants on an issue

1. Navigate to an issue.
2. In the **People** section of the issue, add users to the **Request participants** field. Note that you can only add existing customers to a service desk issue.

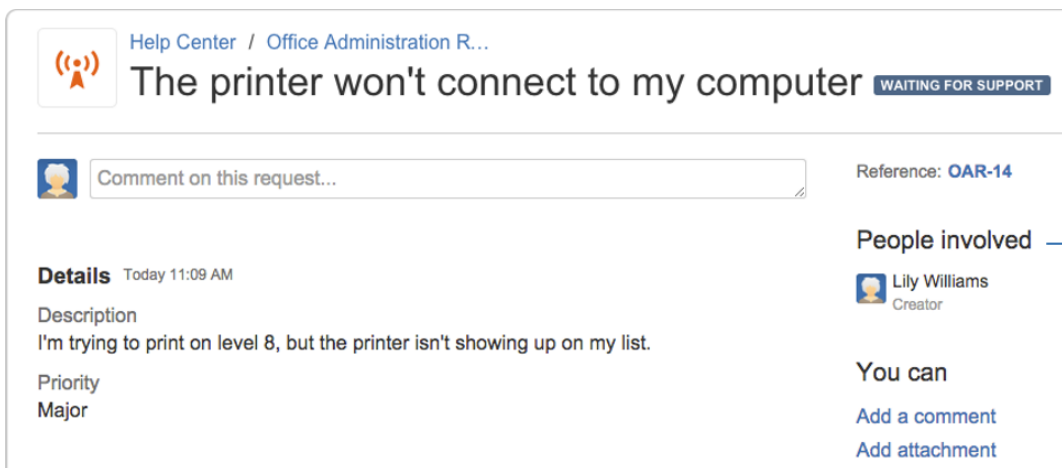
If customers need to add participants via the customer portal, they can do so by selecting **Add people**. Service desk administrators can enable or disable this functionality by going to **Project administration > Request security**.

Adding internal users

As an example, you can involve other agents or JIRA Software developers on an issue to analyze a bug that a customer has reported. To involve internal users, you can mention them in a comment or add them as a watcher. As watchers, they will be notified about the issue and can communicate any updates to you internally.

About request participants

Request participants receive an email notifying them that they have been added. All customers, including the reporter, will appear in the **People involved** section of the request both to the customer on the customer portal and to internal users in JIRA Service Desk.



Request participants
Lists participants on this request.

Note that involving people in a request does not bypass issue-level security. If issue-level security (e.g. restricting an issue to only be viewable by the reporter) has been applied, participants may not be able to access their requests. Service desk administrators can refer to the instructions in [Configuring Issue-level Security](#) to revise or delete an existing issue security scheme.

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 Internet Explorer versions 10+, you need to run Java version 1.7+ platform.
- 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 **More > Attach files**. 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.

When adding or editing a comment, you can also select



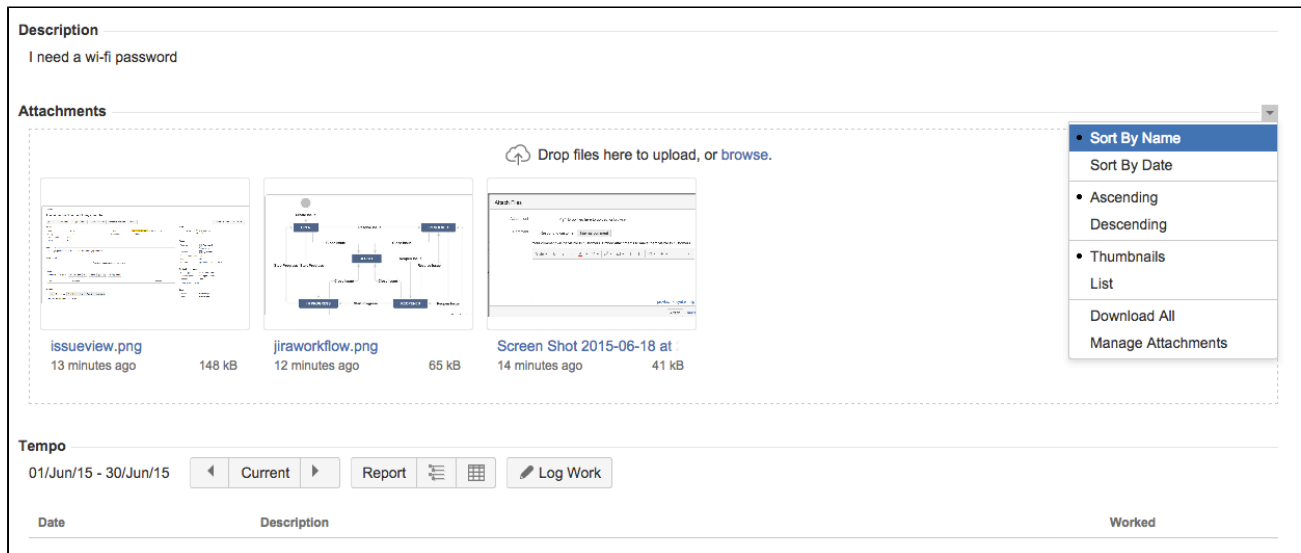
to add attachments. In this case, you'll see wiki markup added to the comment field. As soon as you share your comment, you'll see the file preview.

▼ 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



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. Open the issue and click **More > Attach Screenshot**.
3. Paste the image from your clipboard into the dialog using your system keyboard shortcut.
4. Enter a filename.
5. 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.

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, some of the fields in the new Create Issue dialog box may be pre-populated.
5. When you are satisfied with the content of your issue, click the **Create** button.

Cloning an issue

Cloning or copying an issue lets you quickly create a duplicate of an issue within the same project. The cloned issue is a replica of the original issue, containing the same information stored in the original issue — e.g. Summary, Affects Versions, Components, etc. The cloned issue can also be linked to the original issue.

1. Open the issue you wish to clone.
2. Select **More > Clone**. The **Clone Issue** screen will appear.
 - You can edit the clone issue's **Summary** if you wish.
 - If the issue contains links to other issue(s), you can select whether or not to include the links in the new cloned issue.
 - If the issue contains sub-tasks, you can select whether or not to create the sub-tasks in the new cloned issue.
 - If the issue contains attachments, you can select whether or not to include the attachments in the new cloned issue.
3. 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. An issue cannot be resolved until all its sub-tasks are completed and resolved. 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.

Note: 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.

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.

Raising requests on behalf of customers

Let's say you're helping a customer resolve an issue over the phone, and you need to followup 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](#)

Note that your service desk administrator must have [public signup](#) enabled if you want to raise requests on behalf of new customers, or invite new customers.

Raise a customer request

1. From your service desk project sidebar, select **Customer Portal**.
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's name:

4. Fill in the request details and select **Create**.

Your customer will be emailed a link to the new request – new customers will also receive an invitation to finish creating a service desk account – and you will be able to continue working on the issue from your service desk queue.

Invite a new customer

1. From your service desk project sidebar, select **Customers**.
2. Select **Invite customers** and enter your customer's email address.
3. Send your invitation and you're done!

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](#)
- [Formatting text with wiki markup](#)
- [Tracking issues with labels](#)
- [Watching and voting for issues](#)

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. 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.

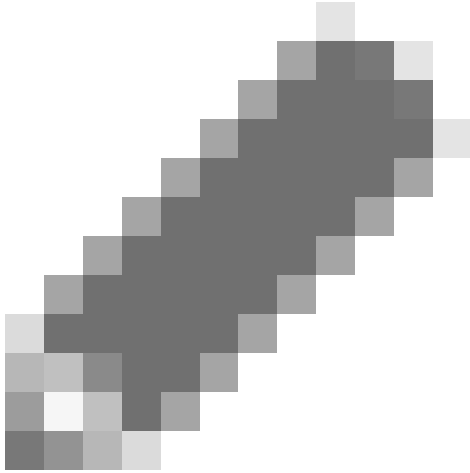



Commenting on issues

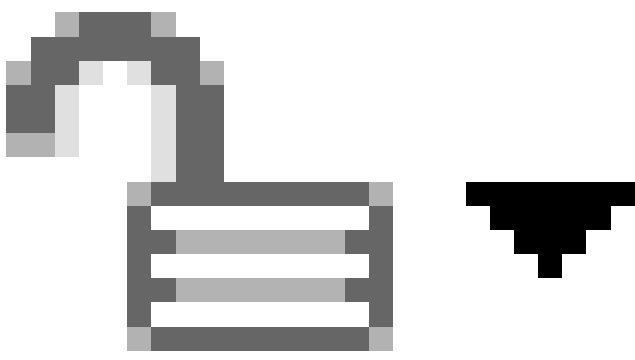

What permissions do you need?

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	Simply click Comment and select the internal (for other agents or collaborators) or external (for customers) tab.
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	<p>Select</p>  <p>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:</p> <div data-bbox="304 1473 1315 1585" style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p>  Susan Griffin added a comment - 15/Mar/13 2:36 PM - edited   </p> <p>I think this nerd is looking better. Thanks for fixing it up!</p> </div> <p>You can hover over 'edited' to see who edited the comment and when.</p>
Link to a comment	<p>Select the Permlink icon on the comment, which will be highlighted in pale blue. Copy the link from the Permlink icon and paste it into your email or chat message. The link will look something like this: http://jira.atlassian.com/browse/TST-123?focusedCommentId=94796#action_94796.</p>

<p>Restrict a comment</p>	<p>Apply viewing restrictions to a comment by selecting the open padlock icon</p>  <p>(or  Restricted to Users if restrictions already apply).</p>
---------------------------	---

Formatting text with wiki markup

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 or comment, you can expand the simple wiki editor toolbar to format your text and select **preview** to see how your formatted text will appear.

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:

Details			
Type:	Documentation SubTask	Status:	Open (View Workflow)
Priority:	Minor	Resolution:	Unresolved
Affects Version/s:	6.0	Fix Version/s:	6.0-OD10
Component/s:	None		
Labels:	doc		

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.

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 an association between a JIRA issue and a Confluence page.
- Link a JIRA issue to any other web page.

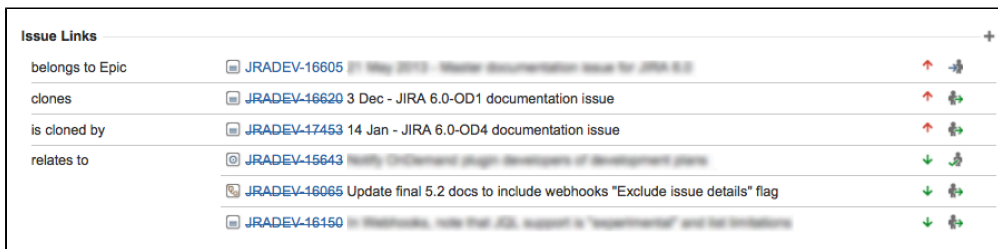
Your JIRA administrator can customize the types of links that you can create.

On this page:

- Creating a link to another issue on the same JIRA site
- Creating a link to an issue on another JIRA site
- 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-1~~.

To create links on JIRA 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.

i This feature is [not available in Atlassian Cloud](#).

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.
 - i 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 authorisation, 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.

i 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.
 - i** 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, discover that a reciprocal link from the remote JIRA issue back to your JIRA 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 authorisation step (at step 3), authenticate on the remote site with a user account which has this permission.

i If you are not prompted for authentication during authorisation, try clearing your browser's cookies first and repeat the procedure again.

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 authorisation, 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 JIRA 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 (see [screenshot above](#)).
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. For details, please see the documentation on [Advanced searching](#).

i 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.

Disabling Mail Notification 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.

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 **Watch Issues**, and select **Next**.
5. Review your bulk operation, and select **Confirm** when you are happy with the operation.

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 [simple 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.

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 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](#).

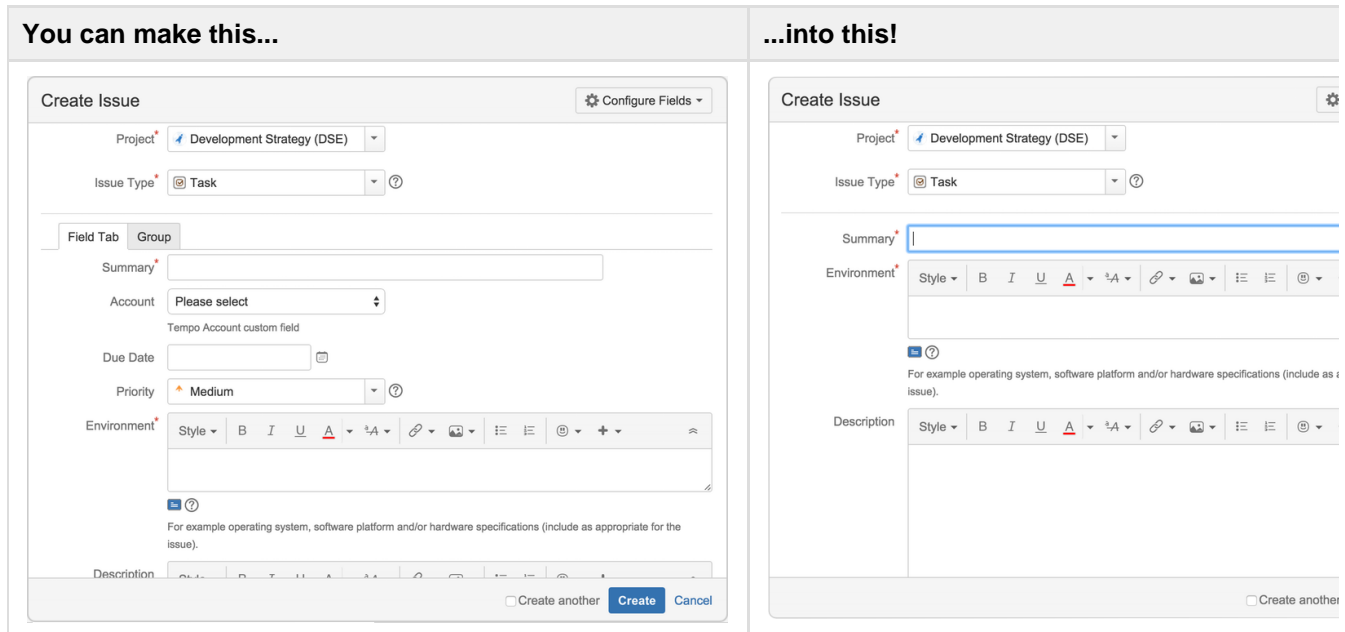
Customizing the issues in a project

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. 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
- 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.



To customize your issues, you need to be a JIRA administrator. You can review more conceptual information on [customizing issues](#) in the JIRA administrator's documentation.

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.

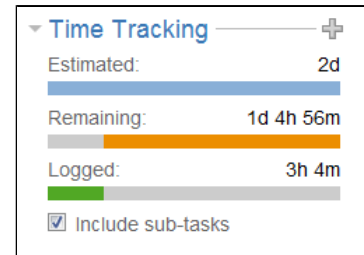
On this page:

- [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	<p>Amount of time anticipated to resolve the issue after completing this unit of work. You can adjust this value using the following options:</p> <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.

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.


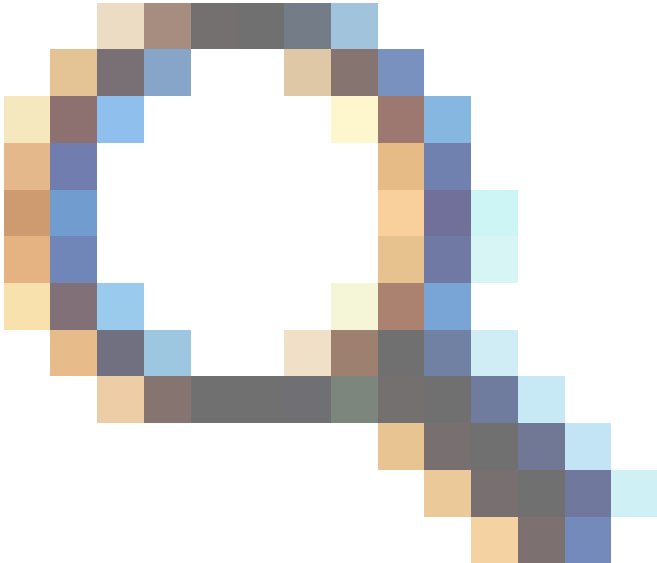
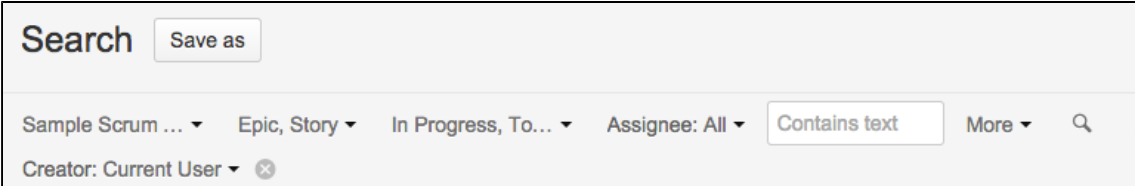
You'll find a step-by-step guide below that will show you how to run a search and use the search results. If you want more details on anything described on this page, see the related topics at the bottom of the page.

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

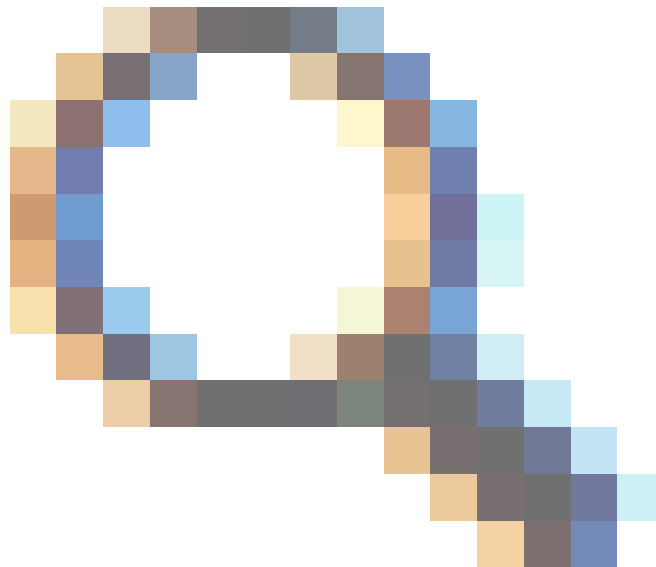
The first step in searching for issues is to define the criteria for your new search. You can define your search criteria in three different ways: using the quick search, using the basic search, or using the advanced search.

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 and press Enter</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 user-friendly interface that lets you define complex queries, without needing to know how to use 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 stru 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.

T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated	Due	Fix Version/s
	ANGRY-304	Red Angry Nerd is scary	Unassigned	Bartosz Gatz	↓	Open	Unresolved	21/Mar/13	21/Mar/13		
	ANGRY-299	My red nerd is not working properly	Unassigned	Susan Griffin	↓	Open	Unresolved	15/Mar/13	15/Mar/13		
	ANGRY-158	ANGRY-13 / give the red nerd a yellow hat	Unassigned	Rosie Jameson	↑	Open	Unresolved	16/Apr/12	10/Aug/12		1.2
	ANGRY-13	I don't like the red nerd	Unassigned	Edwin Wong	↑	In Progress	Unresolved	27/May/11	25/Feb/13		1.2
	ANGRY-306	Red Nerd should have his mouth open on impact	Unassigned	Bartosz Gatz	↓	Open	Unresolved	21/Mar/13	05/Apr/13	31/Mar/13	
	ANGRY-35	Bug on Atlassian - Angry Nerds - Red bird should bounce not flip	Roy Krishna	Edwin Wong	↑	Resolved	Fixed	03/Jun/11	20/Jul/12		1.2
	ANGRY-79	Fix nerd's hair-styling	Bryan Rollins	Edwin Wong	↓	Closed	Fixed	12/Jul/11	20/Jan/12		1.3
	ANGRY-70	Some graphical glitches	Christina Bang	Edwin Wong	↑	In Progress	Unresolved	05/Jun/11	18/Sep/12		1.2

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

The screenshot shows the JIRA Search interface. On the left, there is a 'FILTERS' sidebar with options like 'New filter', 'Find filters', 'My Open Issues', 'Reported by Me', 'Recently Viewed', and 'All Issues'. Below these are 'FAVORITE FILTERS' including '6.2-OD5 Docs', 'Enterprise issues ass...', 'JIRA documentation t...', 'm7 Ecosystem', and 'Open JRADEV issue...'. The main search area has a search bar with 'Sample Scrum ...', 'Epic, Story', 'In Progress, To...', 'Assignee: All', and 'Contains text'. Below the search bar, it shows 'Creator: Current User' and '1-4 of 4' results. The results table has columns: T, Key, Summary, Assignee, Reporter, P, Status, Created, and Updated. The first row is expanded to show details for 'SSP-38 Estimates epic'.

T	Key	Summary	Assignee	Reporter	P	Status	Created	Updated
	SSP-38	Estimates epic	Unassigned	Andrew Lui	↓	TO DO	03/Sep/14	23/Oct/14
	SSP-37	Filters epic	Unassigned	Andrew Lui	↓	TO DO	03/Sep/14	03/Sep/14
	SSP-36	Ranking epic	Unassigned	Andrew Lui	↓	TO DO	03/Sep/14	03/Sep/14
	SSP-35	A test story	Andrew Lui	Andrew Lui	↓	TO DO	05/Jun/14	03/Sep/14

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



icon).

▼ [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:

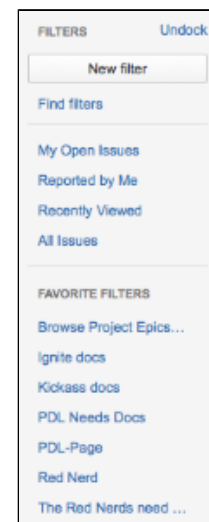
- the query contains an OR operator (note you can have an IN operator and it will be translated, e.g. `project in (A, B)`)
 - i.e. even though this query: `(project = JRA OR project = CONF)` is equivalent to this query: `(project in (JRA, CONF))`, only the second query will be translated.
 - the query 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 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 are interested in. Other times, you can't remember what the issue was, but you remember that it was an open issue, assigned to you. Quick search can help you in these scenarios.

Quick searching

The **Quick Search** box is located at the top right of your screen. To use it, just starting typing what you are looking for.



On this page:

- [Quick searching](#)
- [Understanding quick searching](#)
- [Searching JIRA issues from your browser's search box](#)
- [Next steps](#)

Understanding quick searching

Read the following topics to learn how to get the most out of quick searching: [Jumping to an issue](#) | [Smart querying](#) | [Free-text searching](#)

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 JIRA 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'.

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

<code>r:</code>	Find issues reported by you, another user or with no reporter, using the prefix <code>r:</code> followed by a specific reporter term, such as <code>me</code> , a username or <code>none</code> . <i>Note that there can be no spaces between "r:" and the specific reporter term.</i>	<code>r:me</code> — finds issues reported by you. <code>r:samuel</code> — finds issues reported by the user whose username is "samuel". <code>r:none</code> — finds issues with no reporter.
<code><project name></code> or <code><project key></code>	Find issues in a particular project.	<code>test project</code> <code>TST</code> <code>tst</code>
<code>overdue</code>	Find issues that were due before today.	<code>overdue</code>
<code>created:</code> <code>updated:</code> <code>due:</code>	Find issues with a particular Created, Updated, or Due Date using the prefixes <code>created:</code> , <code>updated:</code> , or <code>due:</code> , 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).	<code>created:today</code> <code>created:yesterday</code> <code>updated:-1w</code> — finds issues updated in the last week. <code>due:1w</code> — finds issues due in the next week. <code>due:-1d,1w</code> — finds issues due from yesterday to next week. <code>created:-1w,-30m</code> — finds issues created from one week ago, to 30 minutes ago. <code>created:-1d</code> <code>updated:-4h</code> — finds issues created in the last day, updated in the last 4 hours.
<code><priority></code>	Find issues with a particular Priority.	<code>blocker</code> <code>major</code> <code>trivial</code>
<code><issue type></code>	Find issues with a particular Issue Type. Note that you can also use plurals.	<code>bug</code> <code>task</code> <code>bugs</code> <code>tasks</code>
<code><resolution></code>	Find issues with a particular Resolution.	<code>fixed</code> <code>duplicate</code> <code>cannot reproduce</code>
<code>c:</code>	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>	<code>c:security</code> — 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...".

Free-text searching

You can search for any word within the issue(s) you are looking for, provided the word is in one of the following fields:

- Summary
- Description
- Comments

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](#).

Searching JIRA 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

Y	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated	Due	Fix Versions
	ANGRY-304	Red Angry Nerd is scary	Unassigned	Barloz Gatz	↓	Open	Unresolved	21/Mar/13	21/Mar/13		
	ANGRY-299	My red nerd is not working properly	Unassigned	Susan Griffin	↓	Open	Unresolved	15/Mar/13	15/Mar/13		
	ANGRY-158	ANGRY-13 / give the red nerd a yellow hat	Unassigned	Rose Jameson	↑	Open	Unresolved	16/Apr/12	10/Aug/12		1.2
	ANGRY-13	I don't like the red nerd	Unassigned	Edwin Wong	↔	In Progress	Unresolved	27/May/11	25/Feb/13		1.2
	ANGRY-306	Red Nerd should have his mouth open on impact	Unassigned	Barloz Gatz	↓	Open	Unresolved	21/Mar/13	05/Apr/13	31/Mar/13	
	ANGRY-305	Bug on Atlassian - Angry Needs - Red bird should bounce not flip	Ray Krishna	Edwin Wong	↑	Resolved	Fixed	03/Jun/11	20/Jul/12		1.2
	ANGRY-79	Fix nerd's hair-styling	Bryan Rollins	Edwin Wong	↓	Closed	Fixed	12/Jul/11	20/Jan/12		1.3
	ANGRY-70	Some graphical glitches	Christina Bang	Edwin Wong	↔	In Progress	Unresolved	05/Jun/11	18/Sep/12		1.2

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



icon).

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



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
--	-------------	-----------

Fields	A field in JQL is a word that represents a JIRA field (or a custom field that has already been defined in JIRA).	Fields reference page ▼ Show list of fields <ul style="list-style-type: none">• Affected version• Assignee• Attachments• Category• Comment• Component• Created• Creator• Custom field• Description• Due• Environment• Epic link• Filter• Fix version• Issue key• Last viewed• Level• Original estimate• Parent• Priority• Project• Remaining estimate• Reporter• Resolution• Resolved• Sprint• Status• Summary• Text• Time spent• Type• Updated• Voter• Votes• Watcher• Watchers• Work ratio
---------------	--	---

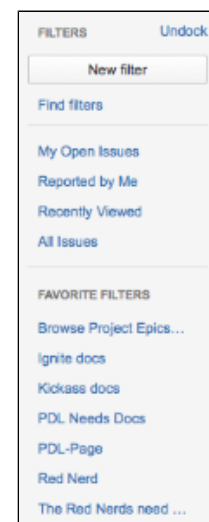
<p>Operators</p>	<p>An operator in JQL is one or more symbols or words that compare the value of a field on its left with one or more values (or functions) on its right, such that only true results are retrieved by the clause. Some operators may use the NOT keyword.</p>	<p>Advanced searching - operators reference</p> <ul style="list-style-type: none"> ▼ Show list of operators <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
<p>Keywords</p>	<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</p> <ul style="list-style-type: none"> ▼ Show list of keywords <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"> • Show list of functions • cascadeOption() • closedSprints() • componentsLeadByUser() • currentLogin() • currentUser() • earliestUnreleasedVersion() • endOfDay() • endOfMonth() • endOfWeek() • endOfYear() • issueHistory() • issuesWithRemoteLinksByGlobalId() • lastLogin() • latestReleasedVersion() • linkedIssues() • membersOf() • now() • openSprints() • projectsLeadByUser() • projectsWhereUserHasPermission() • projectsWhereUserHasRole() • releasedVersions() • standardIssueTypes() • startOfDay() • startOfMonth() • startOfWeek() • startOfYear() • subtaskIssueTypes() • unreleasedVersions() • votedIssues() • watchedIssues()
-------------------------	--	---

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 - 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.

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	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT, WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED

List of functions:

- [cascadeOption\(\)](#)
- [closedSprints\(\)](#)
- [componentsLeadByUser\(\)](#)
- [currentLogin\(\)](#)
- [currentUser\(\)](#)
- [earliestUnreleasedVersion\(\)](#)
- [endOfDay\(\)](#)
- [endOfMonth\(\)](#)
- [endOfWeek\(\)](#)
- [endOfYear\(\)](#)
- [issueHistory\(\)](#)
- [issuesWithRemoteLinksByGlobalId\(\)](#)
- [lastLogin\(\)](#)
- [latestReleasedVersion\(\)](#)
- [linkedIssues\(\)](#)
- [membersOf\(\)](#)
- [now\(\)](#)
- [openSprints\(\)](#)
- [projectsLeadByUser\(\)](#)
- [projectsWhereUserHasPermission\(\)](#)
- [projectsWhereUserHasRole\(\)](#)
- [releasedVersions\(\)](#)
- [standardIssueTypes\(\)](#)
- [startOfDay\(\)](#)
- [startOfMonth\(\)](#)
- [startOfWeek\(\)](#)
- [startOfYear\(\)](#)
- [subtaskIssueTypes\(\)](#)
- [unreleasedVersions\(\)](#)
- [votedIssues\(\)](#)
- [watchedIssues\(\)](#)

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](#)**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	= , != , > , >= , < , <= 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 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	= , !=

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 are assigned to me: assignee = currentUser() Find issues that were reported to me but are not assigned to me: reporter = currentUser() and assignee != currentUser()

[^ 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	earliestUnreleasedVersion(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 the earliest unreleased version of the ABC project: fixVersion = earliestUnreleasedVersion(ABC) Find issues that relate to the earliest unreleased version of the ABC project: affectedVersion = earliestUnreleasedVersion(ABC) or fixVersion = earliestUnreleasedVersion(ABC)

[^ 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	<p>endOfDay()</p> <p>endOfDay("inc")</p> <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. endOfDay("+1") is the same as endOfDay("+1d"). 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 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	<p><code>=</code> , <code>!=</code> , <code>></code> , <code>>=</code> , <code><</code> , <code><=</code> <code>WAS*</code> , <code>WAS IN*</code> , <code>WAS NOT*</code> , <code>WAS NOT IN*</code> , <code>CHANGED*</code> * Only in predicate</p>
Unsupported operators	<code>~</code> , <code>!~</code> <code>IS</code> , <code>IS NOT</code> , <code>IN</code> , <code>NOT IN</code>
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	<p><code>=</code> , <code>!=</code> , <code>></code> , <code>>=</code> , <code><</code> , <code><=</code> <code>WAS*</code> , <code>WAS IN*</code> , <code>WAS NOT*</code> , <code>WAS NOT IN*</code> , <code>CHANGED*</code> * Only in predicate</p>

Unsupported operators	~ , !~ IS , IS NOT , IN , NOT IN
Examples	<ul style="list-style-type: none"> Find issues due by the end of this week: due < endOfWeek() Find issues due by the end of next week: due < endOfWeek("+1")

[^ 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. endOfYear("+1") is the same as endOfYear("+1y"). 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	= , != , > , >= , < , <= WAS* , WAS IN* , WAS NOT* , WAS NOT IN* , CHANGED* * Only in predicate
Unsupported operators	~ , !~ IS , IS NOT , IN , NOT IN
Examples	<ul style="list-style-type: none"> Find issues due by the end of this year: due < endOfYear() Find issues due by the end of March next year: due < endOfYear("+3M")

[^ 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	issueHistory()
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	<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 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	<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 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	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 the latest released version of the ABC project: fixVersion = latestReleasedVersion(ABC) Find issues that relate to the latest released version of the ABC project: affectedVersion = latestReleasedVersion(ABC) or fixVersion = latestReleasedVersion(ABC)

[^ 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	linkedIssues(issueKey) linkedIssues(issueKey,linkType)
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 a particular issue: issue in linkedIssues(ABC-123) Find issues that are linked to a particular issue via a particular type of link: issue in linkedIssues(ABC-123,"is duplicated by")

[^ top of page](#)

membersOf()

Perform searches based on the members of a particular group.

Syntax	membersOf(Group)
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](#)

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](#)

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: <code>project in projectsLeadByUser() AND status = Open</code> Find open issues in projects that are led by Bill: <code>project in projectsLeadByUser(bill) AND status = Open</code>

[^ 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	<code>projectsWhereUserHasPermission(permission)</code> 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: <code>project in projectsWhereUserHasPermission("Resolve Issues") AND status = Open</code>

[^ 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	<code>projectsWhereUserHasRole(rolename)</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 where you have the "Developers" role: <pre>project in projectsWhereUserHasRole("Developers") AND status = Open</pre>
-----------------	---

[^ 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	<pre>releasedVersions() releasedVersions(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 a released version of the ABC project: <pre>fixVersion in releasedVersions(ABC)</pre> Find issues that relate to released versions of the ABC project: <pre>(affectedVersion in releasedVersions(ABC)) or (fixVersion in releasedVersions(ABC))</pre>

[^ 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	<pre>standardIssueTypes()</pre>
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 not subtasks (i.e. issues whose Issue Type is a standard issue type, not a subtask issue type): <pre>issuetype in standardIssueTypes()</pre>

[^ 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	<pre>startOfDay()</pre> <pre>startOfDay("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>startOfDay("+1")</code> is the same as <code>startOfDay("+1d")</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 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()</pre> <pre>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	<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 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()</pre> <pre>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	<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 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	<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 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	<code>unreleasedVersions()</code> <code>unreleasedVersions(project)</code>
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](#)

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](#)

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
Auto-complete	Yes

List of Fields:

- [Affected version](#)
- [Assignee](#)
- [Attachments](#)
- [Category](#)
- [Comment](#)
- [Component](#)
- [Created](#)
- [Creator](#)
- [Custom field](#)
- [Description](#)
- [Due](#)
- [Environment](#)
- [Epic link](#)
- [Filter](#)
- [Fix version](#)
- [Issue key](#)
- [Last viewed](#)
- [Level](#)
- [Original estimate](#)
- [Parent](#)
- [Priority](#)
- [Project](#)
- [Remaining estimate](#)
- [Reporter](#)
- [Resolution](#)
- [Resolved](#)
- [Sprint](#)
- [Status](#)
- [Summary](#)
- [Text](#)
- [Time spent](#)
- [Type](#)
- [Updated](#)

Supported operators	<p>= , != , > , >= , < , <=</p> <p>IS , IS NOT , IN , NOT IN</p> <p><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></p>
Unsupported operators	<p>~ , !~</p> <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" <i>Note that full-stops are reserved characters and need to be surrounded by quote-marks.</i> Find issues with an AffectedVersion of "Big Ted": affectedVersion = "Big Ted" Find issues with an AffectedVersion ID of 10350: affectedVersion = 10350

- Voter
- Votes
- Watcher
- Watchers
- Work ratio

[^ 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
Field Type	USER
Auto-complete	Yes
Supported operators	<p>= , !=</p> <p>IS , IS NOT , IN , NOT IN , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</p> <p><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></p>
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"> 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	<code>=, !=, ~, !~, >, >=, <, <=</code> 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	<code>=, !=</code> <code>IS, IS NOT, IN, NOT IN</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <= WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues that belong to projects in the "Alphabet Projects" Category: <code>category = "Alphabet Projects"</code>

[^ top of page](#)

Comment

Search for issues that have a comment that contains particular text. [JIRA text-search syntax](#) can be used.

Syntax	<code>comment</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 contains text that matches "My PC is quite old" (i.e. a "fuzzy" match): <code>comment ~ "My PC is quite old"</code> Find issues where a comment contains the exact phrase "My PC is quite old": <code>comment ~ "\"My PC is quite old\""</code>

[^ 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	<code>component</code>
Field Type	COMPONENT
Auto-complete	Yes
Supported operators	<code>= , !=</code> <code>IS , IS NOT , IN , NOT IN</code>

Unsupported operators	~ , !~ , > , >= , < , <= WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Supported functions	When used with the IN and NOT IN operators, component supports: <ul style="list-style-type: none"> • componentsLeadByUser()
Examples	<ul style="list-style-type: none"> • Find issues in the "Comp1" or "Comp2" component: component in (Comp1, Comp2) • Find issues in the "Comp1" and "Comp2" components: component in (Comp1) and component in (Comp2) or component = Comp1 and component = Comp2 • Find issues in the component with ID 20500: component = 20500

[^ 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	created
Alias	createdDate
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

<p>Supported functions</p>	<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>
<p>Examples</p>	<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.

<p>Syntax</p>	<p><code>creator</code></p>
<p>Field Type</p>	<p>USER</p>
<p>Auto-complete</p>	<p>Yes</p>
<p>Supported operators</p>	<p><code>= , !=</code> <code>IS , IS NOT , IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN</code></p>
<p>Unsupported operators</p>	<p><code>~ , !~ , > , >= , < , <=</code> <code>CHANGED</code></p>
<p>Supported functions</p>	<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, check-box and radio button fields
Supported operators	<p><i>Different types of custom field support different operators.</i></p> <p><i>For the default custom field types, see the following:</i></p>
Supported operators: number and date fields	<p>= , != , > , >= , < , <=</p> <p>IS , IS NOT , IN , NOT IN</p>
Unsupported operators: number and date fields	<p>~ , !~</p> <p>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</p>
Supported operators: picker, select, check-box and radio button fields	<p>= , !=</p> <p>IS , IS NOT , IN , NOT IN</p>
Unsupported operators: picker, select, check-box and radio button fields	<p>~ , !~ , > , >= , < , <=</p> <p>WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED</p>

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. For the default custom field types, the following functions are supported:</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](#)

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](#)

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	= , != , > , >= , < , <= 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 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> <p><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></p>
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"> • <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>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](#)

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	<code>lastViewed</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 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	parent
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	priority
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": priority = High Find issues with a priority ID of 10000: priority = 10000

[^ 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	project
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"> projectsLeadByUser() projectsWhereUserHasPermission() projectsWhereUserHasRole()
Examples	<ul style="list-style-type: none"> Find issues that belong to the Project that has the name "ABC Project": project = "ABC Project" Find issues that belong to the project that has the key "ABC": project = "ABC" Find issues that belong to the project that has the ID "1234": project = 1234

[^ 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	remainingEstimate
Alias	timeEstimate
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](#)

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	resolution
Field Type	RESOLUTION
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 resolution of "Cannot Reproduce" or "Won't Fix": resolution in ("Cannot Reproduce", "Won't Fix") Find issues with a resolution ID of 5: resolution = 5 Find issues that do not have a resolution: resolution = unresolved

[^ 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	resolved
Alias	resolutionDate
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 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](#)

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	<code>sprint</code>
Field Type	NUMBER
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	<ul style="list-style-type: none"> • <code>openSprints()</code> • <code>closedSprints()</code>

Examples	<ul style="list-style-type: none"> Find issues that belong to sprint 999: <code>sprint = 999</code> Find issues that belong to sprint "February 1": <code>sprint = "February 1"</code> Find issues that belong to either "February 1", "February 2" or "February 3": <code>sprint in ("February 1", "February 2", "February 3")</code> Find issues that are assigned to a sprint: <code>sprint is not empty</code>
-----------------	--

[^ 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	<code>status</code>
Field Type	STATUS
Auto-complete	Yes
Supported operators	<code>= , !=</code> <code>IS , IS NOT, IN , NOT IN , WAS, WAS IN, WAS NOT, WAS NOT IN</code> <code>, CHANGED</code>
Unsupported operators	<code>~ , !~ , > , >= , < , <=</code>
Supported functions	None
Examples	<ul style="list-style-type: none"> Find issues with a status of "Open": <code>status = Open</code> Find issues with a status ID of 1: <code>status = 1</code> Find issues that currently have, or previously had, a status of "Open": <code>status WAS Open</code>

[^ top of page](#)

Summary

Search for issues where the summary contains particular text. [JIRA text-search syntax](#) can be used.

Syntax	<code>summary</code>
Field Type	TEXT
Auto-complete	No
Supported operators	<code>~ , !~</code> <code>IS , IS NOT</code>

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	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 were last updated before 12th December 2010: updated < "2010/12/12" • Find issues that were last updated on or before 12th December 2010: updated < "2010/12/13" • Find all issues that were last updated before 2.00pm on 31st December 2010: updated < "2010/12/31 14:00" • Find issues that were last updated more than two weeks ago: updated < "-2w" • Find issues that were last updated on 15 January 2011: updated > "2011/01/15" and updated < "2011/01/16" • Find issues that were last updated in January 2011: updated > "2011/01/01" and updated < "2011/02/01"

[^ 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	voter
Field Type	USER
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"> membersOf() When used with the EQUALS and NOT EQUALS operators, this field supports: <ul style="list-style-type: none"> currentUser()
Examples	<ul style="list-style-type: none"> Search for issues that you have voted for: voter = currentUser() Search for issues that the user "jsmith" has voted for: voter = "jsmith" Search for issues for which a member of the group "jira-administrators" has voted: voter in membersOf("jira-administrators")

[^ top of page](#)

Votes

Search for issues with a specified number of votes.

Syntax	votes
Field Type	NUMBER
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 all issues that have 12 or more votes: votes >= 12

[^ 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	watcher
---------------	---------

Field Type	USER
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"> membersOf() When used with the EQUALS and NOT EQUALS operators, this field supports: <ul style="list-style-type: none"> currentUser()
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	watchers
Field Type	NUMBER
Auto-complete	No
Supported operators	= , != , > , >= , < , <= IN , NOT IN
Unsupported operators	~ , !~ IS , IS NOT , 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"> membersOf() When used with the EQUALS and NOT EQUALS operators, this field supports: <ul style="list-style-type: none"> currentUser()
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 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**

$\text{workRatio} = \text{timeSpent} / \text{originalEstimate}) \times 100$

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.

List of Keywords:

- 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.

List of Operators:

- [EQUALS: =](#)
- [NOT EQUALS: !=](#)

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"
```

- 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

[^top of page](#)

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: `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 text fields only, 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

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 contains the word "win" (or simple derivatives of that word, such as "wins"):

```
summary ~ win
```

- 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\""
```

[^top of page](#)

DOES NOT CONTAIN: !~

The "!~" operator is used to search for issues where the value of the specified field is not a "fuzzy" match for the specified value. For use with text fields only, 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

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
```

[^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](#)

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.

[Wildcard searches: ? and *](#) | [Fuzzy searches: ~](#) | [Proximity searches](#)

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 [JIRA-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 a 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", "s", "such", "t", "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

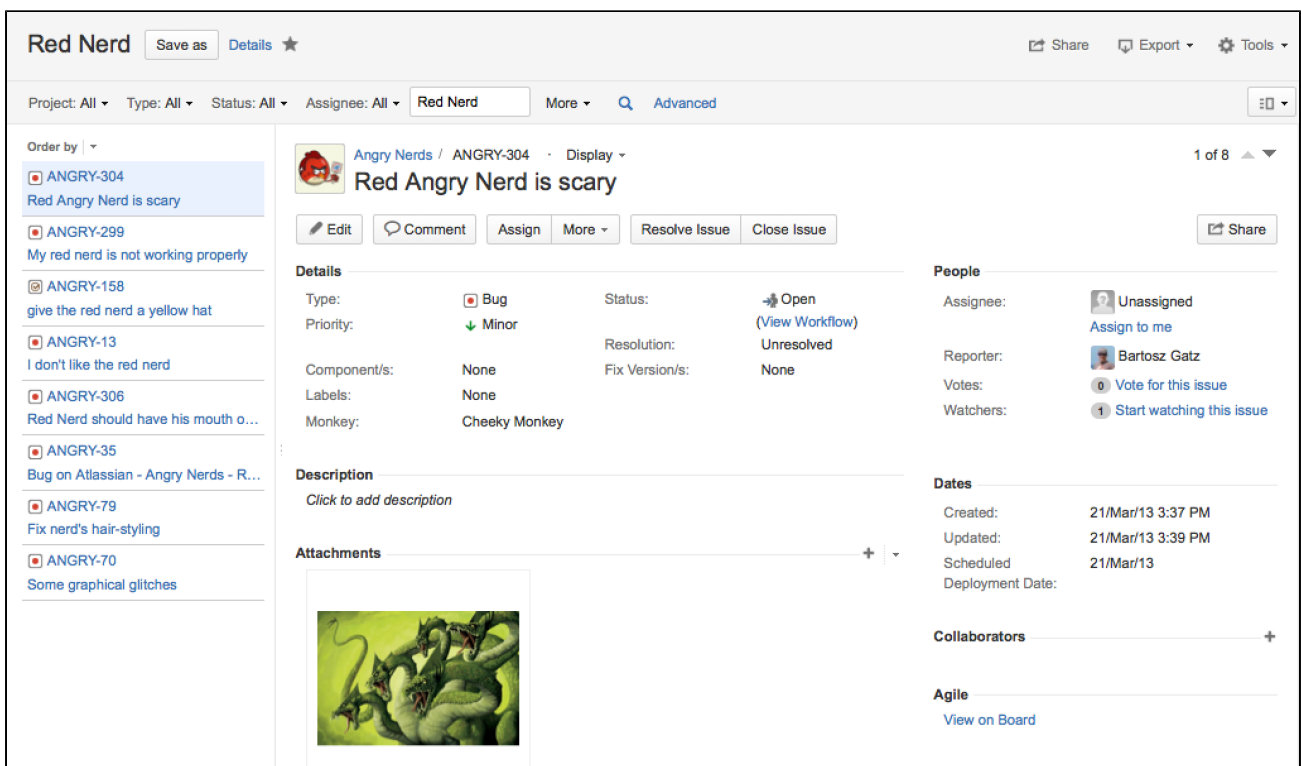
JIRA's powerful [issue search](#) functionality is enhanced by the ability to save searches, called *filters* in JIRA, for later use. You can do the following with JIRA filters:

- 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)



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.

Name	Shared With	Subscriptions
★ Browse Project Epics for ADG	• Shared with all users	None - Subscribe
★ Ignite docs	• Private filter	None - Subscribe
☆ JIRA OnDemand Feature Tracking	• Shared with all users	None - Subscribe
★ Kickass docs	• Private filter	None - Subscribe
★ PDL Needs Docs	• Shared with all users	None - Subscribe
★ PDL-Page	• Private filter	None - Subscribe
★ Red Nerd	• Private filter	None - Subscribe
★ The Red Nerds need modifications	• Shared with all users	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 have created. If you want to edit a filter that was shared with you, 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 can be shared with other users via user groups, projects, and project roles. They can also be shared globally. 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 Shares** field by selecting the 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.


The following screenshot provides an overview of the key features of the issue navigator.

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
- Subscribing to your search results
- Bulk modifying issues in your search results
- Next steps

Screenshot: Issue navigator (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 issue macro. For instructions, see [JIRA issues macro](#).

Displaying your search results as a chart

Click **Export > Charts**. Choose the desired chart from the dialog that is displayed, then click **Save to Dashboard**.

The chart will be added to your dashboard.

Exporting your search results

Excel	<p>Click Export > Excel (All fields) or Export > Excel (Current fields).</p> <ul style="list-style-type: none"> • Excel (All fields): this will create a spreadsheet column for every issue field (excluding comments). <i>Note: This will only show the custom fields that are available for all of the issues in the search results. For example, a field that is only available for one project when your search results has issues from multiple projects.</i> • Excel (Current fields): this will create a spreadsheet column 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>
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>
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>

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/jira/brc</pre>
project (or pid)	<pre><project id="10330" key="TST">Test</project></pre>
description	<pre><description>This is a detailed description of th</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.atlassian.</pre>
parent	<pre><parent id="22620">TEST-5</parent></pre>
priority	<pre><priority id="4" iconUrl="https://extranet.atlassian.com:443/jira/</pre>
status	<pre><status id="5" iconUrl="https://extranet.atlassian.com:443/jira/</pre>
resolution	<pre><resolution id="1">Fixed</resolution></pre>

labels	<pre><labels> <label>focus</label> </labels></pre>
assignee	<pre><assignee username="jsmith">John Smith</assignee></pre>
reporter	<pre><assignee username="jsmith">John Smith</assignee></pre>
security	<pre><security id="10021">Private</security></pre>
created	<pre><created>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/cr</pre>
updated	<pre><updated>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/up</pre>
resolved (Or resolutiondate)	<pre><resolved>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/r</pre>
due (Or duedate)	<pre><due>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/due></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" created=" familiar</comment> <comment id="39273" author="jbrown" created=" too</comment> </comments></pre>
attachments (or attachment)	<pre><attachments> <attachment id="30318" name="Issue Navigator - created="Mon, 9 Feb 2009 13:32:58 -0600 (CST)"/> <attachment id="30323" name="Windows XP (with created="Tue, 10 Feb 2009 00:30:11 -0600 (CST)"/> </attachments></pre>
timeoriginalestimate	<pre><timeoriginalestimate seconds="600">10 minutes</t</pre>
timeestimate	<pre><timeestimate seconds="300">5 minutes</timeestima</pre>
timespent	<pre><timespent seconds="300">5 minutes</timespent></pre>
aggregatetimeoriginalestimate	<pre><aggregatetimeoriginalestimate seconds="36000">10</pre>
aggregatetimeestimate	<pre><aggregatetimerremainingestimate seconds="18000">5</pre>
aggregatetimespent	<pre><aggregatetimespent seconds="18000">5 hours</aggr</pre>
timetracking	<pre><timeoriginalestimate seconds="600">10 minutes</t <timeestimate seconds="300">5 minutes</timeestima <timespent seconds="300">5 minutes</timespent> <aggregatetimeoriginalestimate seconds="36000">10 <aggregatetimerremainingestimate seconds="18000">5 <aggregatetimespent seconds="18000">5 hours</aggr</pre>

issuelinks	<pre><issuelinks> <issuelinktype id="10020"> <name>Duplicate</name> <inwardlinks description="is duplicated k <issuelink> <issuekey id="22477">INTSYS-1009< </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.customfield <customfieldname>Department</customfieldnr <customfieldvalues> <customfieldvalue>Adminstration</cust </customfieldvalues> </customfield> </customfields></pre>
allcustom	<pre><customfields> <customfield id="customfield_10112" key="com.atlassian.jira.plugin.system.customfield <customfieldname>Department</customfieldnr <customfieldvalues> <customfieldvalue>Adminstration</cust </customfieldvalues> </customfield> <customfield id="customfield_10111" key="com.atlassian.jira.plugin.system.customfield <customfieldname>Expenditure Type</custom <customfieldvalues> <customfieldvalue>Operating</customfi </customfieldvalues> </customfield> </customfields></pre>

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. • If you want to specify a group as a recipient: <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 tempMax 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 [Modifying multiple \(bulk\) issues](#).

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.

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.

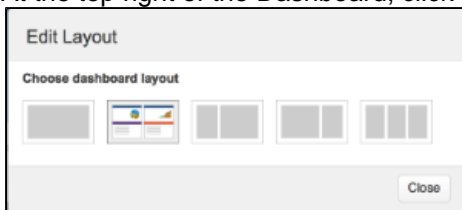
1. At the top right of the Dashboard, click the **Tools** menu.
2. Select either **Create Dashboard** to create a blank dashboard, or **Copy Dashboard** to create a copy of the dashboard you are currently viewing.
3. Name and describe your dashboard.
4. Fill out the rest of the fields as applicable.
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 (e.g. three columns instead of two):

1. At the top right of the Dashboard, click the '**Edit Layout**' link. A selection of layouts will be displayed:



2. Select your preferred layout.

Managing gadgets

To get the most out of your dashboard, including adding, rearranging, removing, and configuring gadgets, see [add and customize some 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.

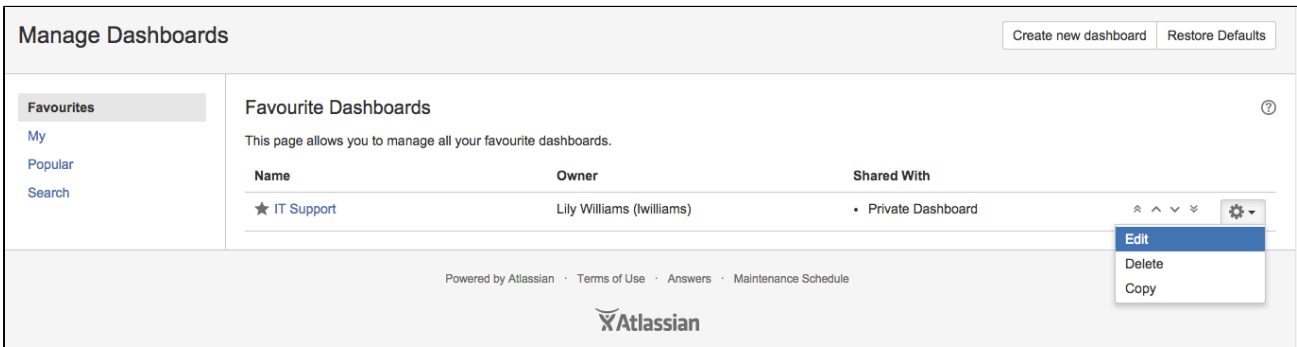
Sharing and editing your dashboard

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/Share > Add sharing permissions**.
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.



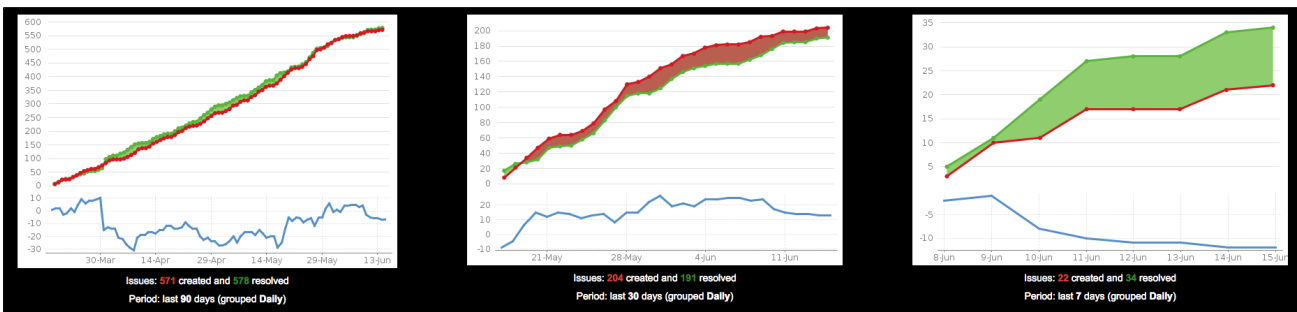
Note on dashboard permissions

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, click **Tools > 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 **Dashboard** link in the header.
2. On the dashboard, Click **Add Gadget**.
3. 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 gadgets to 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!
<i>Sprint Burndown Gadget</i>				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.

<p>Sprint Health Gadget</p>				<p>Seeing a summary of the issues in a sprint in a handy color-coded bar graph.</p> <p>▼ Notes...</p> <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'.
<p>Version Report</p>				<p>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.</p>
<p>Agile Wallboard Gadget</p>				<p>Know how you're tracking with an agile board displayed on your wallboard (or dashboard).</p>
<p>Assigned to Me</p>				<p>Quickly see all the unresolved issues assigned to you.</p>
<p>Average Age Chart</p>				<p>Want to know the average age of unresolved issues? This gadget tells you just that.</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 (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>Average Number of Times in Status</p>				<p>Displays the average number of times issues have been in a status.</p>
<p>Average Time in Status</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, e.g. 'Bamboo Charts Gadget from http://172.20.5.83:8085', 'Bamboo Charts Gadget from http://172.19.6.93:8085', etc. When you add this gadget to your JIRA dashboard, you may see a message similar to this: <div style="border-left: 1px solid #ccc; padding-left: 10px; margin-left: 20px;"> <p><i>The website (container) you have placed this gadget on is unauthorized. Please contact your system administrator to have it approved.</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 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> <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, e.g. 'Bamboo Charts Gadget from http://172.20.5.83:8085', 'Bamboo Charts Gadget from http://172.19.6.93:8085', etc. When you add this gadget to your JIRA dashboard, you may see a message similar to this: <div style="border-left: 1px solid #ccc; padding-left: 10px; margin-left: 20px;"> <p><i>The website (container) you have placed this gadget on is unauthorized. Please contact your system administrator to have it approved.</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 once-off authentication before your gadget will display correctly.</p>

<p>Bamboo Plans</p>				<p>Seeing a list of all plans on a particular Bamboo server and each plan's current status.</p> <p> Notes... <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, e.g. 'Bamboo Charts Gadget from http://172.20.5.83:8085', 'Bamboo Charts Gadget from http://172.19.6.93:8085', etc. When you add this gadget to your JIRA dashboard, you may see a message similar to this: <div style="border-left: 1px solid #ccc; padding-left: 10px; margin-left: 20px;"> <p><i>The website (container) you have placed this gadget on is unauthorized. Please contact your system administrator to have it approved.</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 once-off authentication before your gadget will display correctly.</p> </p>
<p>Clover Coverage</p>				<p>Seeing the Clover coverage of plans from a particular Bamboo server.</p> <p> Notes... <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, e.g. 'Bamboo Charts Gadget from http://172.20.5.83:8085', 'Bamboo Charts Gadget from http://172.19.6.93:8085', etc. When you add this gadget to your JIRA dashboard, you may see a message similar to this: <div style="border-left: 1px solid #ccc; padding-left: 10px; margin-left: 20px;"> <p><i>The website (container) you have placed this gadget on is unauthorized. Please contact your system administrator to have it approved.</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 once-off authentication before your gadget will display correctly.</p> </p>
<p>Created vs. Resolved Chart</p>				<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... <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. </p>

Crucible Charts	✓	✓	✓	Seeing statistical summaries of your code reviews. Notes... <ul style="list-style-type: none"> Your JIRA administrator must have configured the FishEye plugin 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	✓	✓	✓	Get two charts about your repo in one: lines of code and commit activity. Notes... <ul style="list-style-type: none"> Your JIRA administrator must have configured the FishEye plugin on your JIRA server, if you want to add the Crucible Charts gadget to your dashboard (not applicable to JIRA Cloud).
Introduction	✓	✓	✓	Say hello to users with a configurable message on the dashboard. 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.
Issue Statistics	✓	✓	✓	See the issues returned from a specified project or saved filter (grouped by a specified field).
Issues In Progress	✓	✓	✓	Time to work! See all issues that are currently in progress and assigned to you.
JIRA Issues Calendar	✓	✓	✓	Generating a calendar-based view of due dates for issues and versions Notes... <ul style="list-style-type: none"> The JIRA Calendar plugin is required for this gadget to be available.
JIRA Road Map		✓		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.
Labels Gadget	✓	✓	✓	Use this gadget to see a list of all the labels used in a given project.

Pie Chart	✓	✓	✓	See the issues returned from a specified project or issue filter, grouped by a specified field.
Quick Links	✓	✓	✓	Link to frequently-used searches and operations.
Recently Created Chart	✓	✓	✓	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.
Resolution Time	✓	✓	✓	<p>Check trends in the average time taken to resolve issues.</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 (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>Display your specified HTML text on the dashboard.</p> <p> Notes... <ul style="list-style-type: none"> 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 > Add-ons. The 'Find add-ons' screen shows add-ons available via the Atlassian Marketplace. Choose Manage add-ons to view the plugins currently installed on your JIRA site. Enable the Text module in the Atlassian JIRA - Plugins - Gadgets Plugin (You need to select the System add-ons from the drop-down). If you cannot enable the text gadget, please contact Atlassian Support for assistance. </p>
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... <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). </p>
Time to First Response	✓	✓	✓	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				Displays the matching issues for a project or filter as a pie chart.

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 your OAuth and login tokens](#)

Editing your user details

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)
-  Your page will be reloaded the JIRA home page you selected.
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.

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 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 your 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 this gadget access to JIRA data which is restricted or privy to your JIRA user account.

On this page:

- [About OAuth access tokens](#)
- [Before you begin](#)
- [Issuing OAuth access tokens](#)
- [Revoking OAuth access tokens](#)

Before you begin

Your JIRA administrator must establish an OAuth relationship with this external web application or instance by approving it as an OAuth consumer. For example, if you want to add a JIRA gadget to your Bamboo homepage and allow this gadget to access your restricted JIRA data, then your JIRA administrator must first approve Bamboo as an OAuth consumer.

The JIRA gadget on the 'consumer' is granted access to your JIRA data via an 'OAuth access token', which acts as a type of 'key'. As long as the consumer is in possession of this access token, the JIRA gadget will be able to access JIRA data that is both publicly available and privy to your JIRA user account. You can revoke this access token at any time from your JIRA user account, otherwise, all access tokens expire after seven days. Once the access token is revoked or has expired, the JIRA gadget will only have access to publicly available data on your JIRA instance.

An OAuth access token will only appear in your user profile if the following conditions have been met:

1. Your JIRA Administrator has established an application link using OAuth between your JIRA instance and the consumer. JIRA Administrators should refer to [Using AppLinks to link to other applications](#).
2. You have accessed a JIRA gadget on a consumer and have allowed this gadget access to your JIRA data. See [Issuing OAuth access tokens](#) below for details on this process.

Screenshot: Viewing your OAuth Access Tokens

Consumer	Consumer Description	Issued on	Expires on	Actions
Activity Stream	Atlassian RefImpl at http://localhost:8080/dashboards	02/10/2009	09/10/2009	Revoke OAuth Access Token
Created vs Resolved Chart	Atlassian RefImpl at http://localhost:8080/dashboards	02/10/2009	09/10/2009	Revoke OAuth Access Token

Issuing OAuth access tokens

An OAuth access token is issued by JIRA to provide one of its gadgets on a consumer, access to your JIRA data (that is, data which is restricted to your JIRA user account).

1. When you are using a JIRA gadget on a consumer (such as Bamboo) and this gadget requires access to your JIRA data, you will first be prompted to log in to JIRA (if you have not already done so).
2. Once you have logged in to JIRA, you will be prompted with a '**Request for Access**' message:

Screenshot: Request for Access Message

Request for Access

The application **Bamboo** would like to access your **Atlassian JIRA** account on your behalf. If you trust this application and would like to allow it access, click the 'Approve Access' button. An example of such access is a gadget running on another server.

By approving this request for access, you are allowing the application to **read** and **update** data using your username. The application will not have access to your password.

You can revoke this access at any time by going to the OAuth Access Tokens section of your user profile. [Learn more.](#)

Approve Access
Deny Access

At this point, JIRA is preparing to issue the JIRA gadget (on the consumer) with an OAuth access token.

3. To grant the gadget access to your JIRA data, click the '**Approve Access**' button. The consumer application will receive the OAuth access token from your JIRA instance. This access token is specific to this gadget and as long as the token resides with the gadget, your gadget will have access to your JIRA data.

Revoking OAuth access tokens

You can revoke an OAuth access token to deny a JIRA gadget on a consumer access to JIRA data which is restricted to your JIRA user account. You can only revoke OAuth access tokens that [you have allowed JIRA to issue previously](#).

1. Choose **your user name** at top right of the screen, then choose **Profile**.
2. Click the '**Tools**' menu and select the '**View OAuth Access Tokens**' menu item.
3. The '**OAuth Access Tokens**' page will be displayed.

Screenshot: Viewing your OAuth Access Tokens

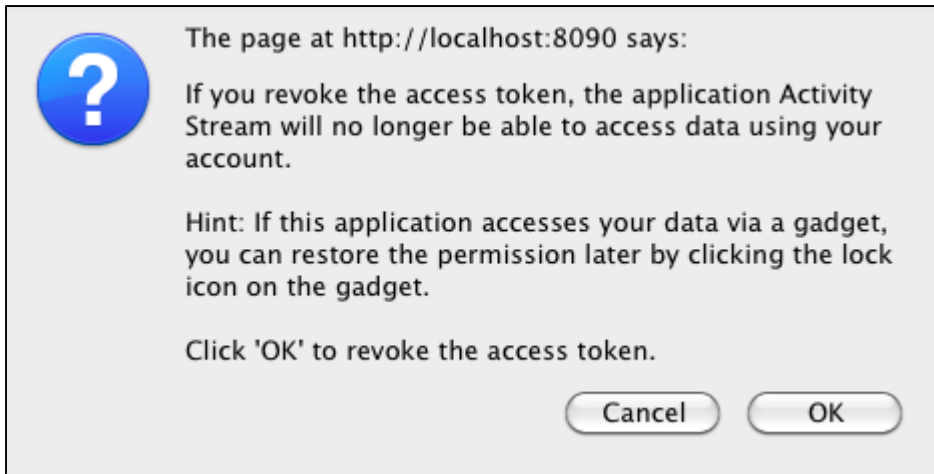
OAuth Access Tokens

You have allowed the following gadgets/applications to access JIRA data using your account:

Consumer	Consumer Description	Issued on	Expires on	Actions
Activity Stream	Atlassian RefImpl at http://localhost:8080/dashboards	02/10/2009	09/10/2009	Revoke OAuth Access Token
Created vs Resolved Chart	Atlassian RefImpl at http://localhost:8080/dashboards	02/10/2009	09/10/2009	Revoke OAuth Access Token

Your list of OAuth access tokens is presented in a tabular format, with each access token presented in separate rows and each property of these tokens presented in a separate columns. Refer to the [OAuth access token table details](#) section below for more information about this table.

4. Locate the JIRA gadget and its associated consumer application whose OAuth access token you wish to revoke and click its 'Revoke OAuth Access Token' link in the 'Actions' column.
5. You may be prompted to confirm this action. If so, click the '**OK**' button.



The gadget's access token is revoked and the JIRA gadget on the consumer will only have access to publicly available JIRA data.

OAuth access token table details

Column name	Description
Consumer	The name of the JIRA gadget that was added on the consumer.
Consumer Description	A description of this consumer application. This information would have been obtained from the consumer's own OAuth settings when an OAuth relationship was established between JIRA and that consumer. ⓘ If the consumer is another Atlassian application, this information is obtained from the Consumer Info tab's 'Description' field of the OAuth Administration settings. The application's administrator can customize this Consumer Info detail.
Issued On	The date on which the OAuth access token was issued to the consumer by JIRA. This would have occurred immediately after you approved this gadget access to your JIRA data (privy to your JIRA user account).
Expires On	The date when the OAuth access token expires. This is seven days after the 'Issued On' date. When this date is reached, the access token will be automatically removed from this list.
Actions	The functionality for revoking the access token .

Requesting add-ons

The [Atlassian Marketplace](#) website offers hundreds of add-ons that administrators can install to enhance and extend your JIRA applications. If the add-on request feature is enabled for your instance, you can submit requests for Marketplace add-ons 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 add-on 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 add-ons that can help them work. When you find an add-on of interest, you can submit a request with just a few clicks.

Submitting an add-on 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 add-ons or use the category menus to browse or filter by add-ons 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 add-on 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 add-on
- 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 add-ons. Also your request message will appear in the add-on details view, visible from the administrator's 'Find New Add-ons' page. From there, your administrator can purchase the add-on, try it out or dismiss requests.

Updating an add-on 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 add-on 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	

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
- Select a default assignee — choose who is automatically assigned issues with that component assigned
- 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.

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).

Selecting a default assignee

You can optionally set a default assignee for a component. This will override the project's default assignee for

On this page:

- [Managing a project's components](#)
- [Adding a new component](#)
- [Selecting a default assignee](#)
- [Editing a component's details](#)
- [Deleting a component](#)

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, hover over the relevant component to display the pencil icon.
2. Edit the component's **Name**, **Description**, **Lead**, and **Default Assignee**.
3. Click the **Update** button to save your changes.

Deleting a component

1. On the 'Components' screen, hover over the relevant component to display the **Delete** button.
2. You will be prompted to associate these issues with another component if you wish.

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.

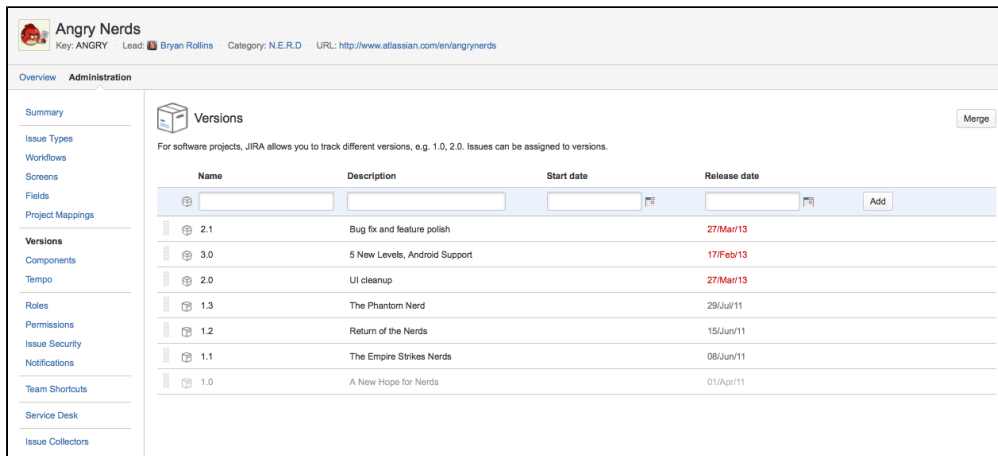
1. Choose




> **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



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.

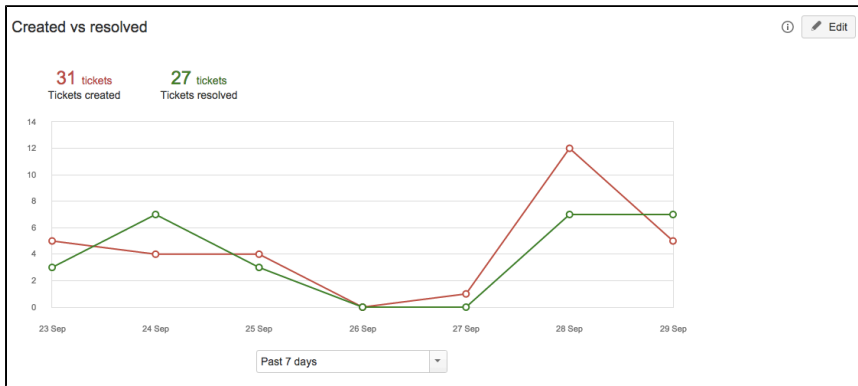
- On the 'Versions' page, click the



icon for the relevant version, and drag it to its new position in the version order.

Setting up service desk reports

JIRA Service Desk provides powerful realtime reporting functionality so you can see your team's performance metrics:



On this page:

- What makes up a service desk report
- Create a new report


You can also create your own custom reports to query any combination of performance data. Your team members have access to a read-only version of the **Reports** tab so they can also see the data you're tracking. To create and edit reports, make sure you're logged in as an administrator.

What makes up a service desk report

Each report consists of one or more series. A report series is composed of a predefined series (e.g. issues created), or a time metric series (e.g. time to resolution). For each series, you need to select a label and color to easily identify the data points in this series on your report, and define which issues should be counted with a [JQL statement](#).

Create a new report

In your service desk project sidebar, select **Reports**. You will see a list of default reports for your service desk project. In this example, we'll create a report that shows the breakdown of all high priority issues based on the issue type:

- Select **New report**
- Choose a report name that you and your team will understand (e.g. High priority issues)
- Select **Add a series** and fill in the following details:
 - Series** = Created (to display all created issues that fit the series criteria)
 - Label** = IT Help issues
 - Color** = 
 - JQL filter** = type = "IT Help" AND priority = Highest OR priority = High
- Select **Add** and save your report.
- You can add additional series to capture high priority issues of other types (e.g. type = "Fault" or type = "Access")

See [Reporting on SLAs](#) for detailed information on how to run reports on SLA progress or status.

Setting up SLAs

JIRA Service Desk provides powerful built-in Service Level Agreement (or SLA) management so you can

track your team's progress against agreements you've set for customers. An SLA is made up of two settings:

- A time metric, which lets you define how time will be measured for this SLA; and,
- A goal for selected issues, which lets you define a target for the time metric. Different sets of issues can have different goals.

On this page:

- [Setting up SLA time metrics](#)
- [Setting up SLA goals](#)
- [Creating SLA Calendars](#)
- [How your team sees SLAs](#)
- [Managing SLA data](#)
- [SLA usage notes](#)

JIRA Service Desk comes with a few pre-configured SLA metrics to cover some of the most common IT requirements; however you can modify or create custom SLA metrics to reflect the SLAs you use in your business. JIRA Service Desk also provides robust reporting tools that you can use to track your team's performance against your SLAs. Check out [Reporting on SLAs](#) for tips on tracking your SLAs.

You must be logged in as a JIRA Administrator or Project Administrator to configure SLAs.

Setting up SLA time metrics

You can think of the time metric as a stopwatch that tracks time between two points in an issue's life-cycle. JIRA Service Desk lets you control exactly when time is tracked, letting you start, pause, and stop based on selected issue conditions. For example:

- In an SLA that guarantees issues are resolved in a certain amount of time, time starts when the issue is created and stops when the issue is resolved. You can choose to exclude time spent waiting on a customer by pausing the SLA on the Waiting for Customer status.
- In an SLA that guarantees a fast response time from your team, time starts when the issue status is Waiting for Support and stops when the status is Waiting for Customer. Each time the issue meets the start condition, a new cycle of the SLA will begin.

From your service desk project sidebar, select **Project administration > SLAs > New Metric** to fill in the following conditions:

Condition	Description
Metric name	This name (e.g. Time to resolution) will appear to agents in the SLA section of issues, and should indicate what you're tracking. Note that you can't change the name of an SLA metric after you've saved it.
Start	Time starts being counted against the SLA whenever the issue has this condition (e.g. Issue Created)
Pause on	Time doesn't get counted against the SLA whenever the issue has this condition (e.g. Status: Waiting for Customer)
Stop	Time stops being counted against the SLA whenever the issue has this condition (e.g. Resolution: Set)

Here's an example of the New Metric form:

Time to resolution

Save
Cancel

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

Start
Begin counting time when

Clear selected items

- Issue Created
- Resolution: Cleared
- Assignee: From Unassigned
- Assignee: To Unassigned
- Assignee: Changed
- Entered Status: Resolved
- Entered Status: Waiting for Cu...
- Entered Status: Waiting for Su...

→

Pause on
Time is not counted during

- Assignee: Set
- Assignee: Not Set
- Status: Resolved
- Status: Waiting for Customer
- Status: Waiting for Support
- Resolution: Set
- Resolution: Not Set

→

Stop
Finish counting time when

- Resolution: Set
- Assignee: From Unassigned
- Assignee: To Unassigned
- Assignee: Changed
- Entered Status: Resolved
- Entered Status: Waiting for Cu...
- Entered Status: Waiting for Su...
- Resolution: Cleared
- Comment: By Customer

Notice that you can set multiple conditions for the start, stop, and pause time. Check out [Example: creating an SLA with multiple cycles](#) for an in-depth look at how you can use this functionality.

Setting up SLA goals

While the time conditions on an SLA specify what your team considers to be trackable time, the goal section of the SLA metric lets you set the amount of time that's allowed for different scenarios. SLA goals can be in hours or in time increments less than an hour. For example, an SLA that guarantees issues are resolved in a certain amount of time could have the following goals:

- Blocker issues are resolved within 24 hours and Critical issues are resolved within 36 hours
- Blocker issues created by a member of the Build Engineering team are resolved within 12 hours, while Blocker issues created by a member of the Accounting team are resolved within 36 hours.

In the New Metric or Edit Metric screen, you can 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 (e.g. 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 (e.g. 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 this list:

Created in 2015 by Atlassian. Licensed under a [Creative Commons Attribution 2.5 Australia License](#).

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)"/>	24/7 Calendar (Default)	<input type="button" value="Add"/>
priority = Critical	36h	24/7 Calendar (Default)	<input type="button" value="Delete"/>
priority = Blocker	24h	24/7 Calendar (Default)	<input type="button" value="Delete"/>
All remaining issues	48h	24/7 Calendar (Default)	

Creating SLA Calendars

By default, SLAs are measured against 24/7 working hours; however, you can use SLA Calendars to specify your team's working hours. For example, SLA calendars let you exclude lunch breaks, holidays, or weekends from the time that affects the SLA metrics.

Once you have saved your new SLA metric, select the **Calendars** button to add a calendar with your team's work schedule. The Sample 9-5 Calendar shown here can be edited to allow for a 1-hour lunch break that does not count towards the SLA goal. Simply delete the 09:00-17:00 line item and replace it with two line items (before and after lunch) for the same day:

Calendars

Sample 9-5 Calendar

Time zone

Work week

Monday 13 : 00 to 17 : 00

Monday	08:00	12:00	<input type="button" value="Delete"/>
	13:00	17:00	<input type="button" value="Delete"/>
Tuesday	09:00	17:00	<input type="button" value="Delete"/>
Wednesday	09:00	17:00	<input type="button" value="Delete"/>
Thursday	09:00	17:00	<input type="button" value="Delete"/>
Friday	09:00	17:00	<input type="button" value="Delete"/>

Save your calendar and open an SLA metric to associate the calendar with one of your metric goals. Note that 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 your team sees SLAs

Your team members can see a read-only version of the **SLA** tab so they can view how the SLA is configured. In the detail view of issues, the **SLA** section lists even more detail about the SLA that the issue is being measured against:



Review the following sections for more detail on what the SLA tracker conventions indicate.

Ongoing SLAs

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

	<p>SLA has greater than 1 hour remaining.</p>
	<p>SLA has less than 1 hour remaining. If the SLA goal is one hour, the SLA has 30 minutes remaining.</p>
	<p>SLA has less than 30 minutes remaining. If the SLA goal is one hour, the SLA has 15 minutes remaining.</p>
	<p>SLA has breached the target. The amount of time past the goal is shown as a negative number.</p>
	<p>SLA has been paused.</p>

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.

	<p>SLA completed successfully.</p>
	<p>SLA completed unsuccessfully (it breached the target)</p>

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.)

SLAs

- 0:44 Time waiting for support
within 24 hours
Yesterday 5:36 PM - Yesterday 5:37
- 72:41 Time to resolution PM
within 96 hours 59 minutes of 1 hour
- 0:59 ✓ Time to response
within 1 hour

SLA sorting

When you view a list of issues (in a queue or elsewhere), you can sort them by their SLA resolution 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.

Managing SLA data

When new SLA metric names are created, new custom fields are created in JIRA to store them. The type of these custom fields is SLA Custom Field Type. As a JIRA administrator, you have the following options to manage the SLA custom fields.

Allow project administrators to create SLA metric names

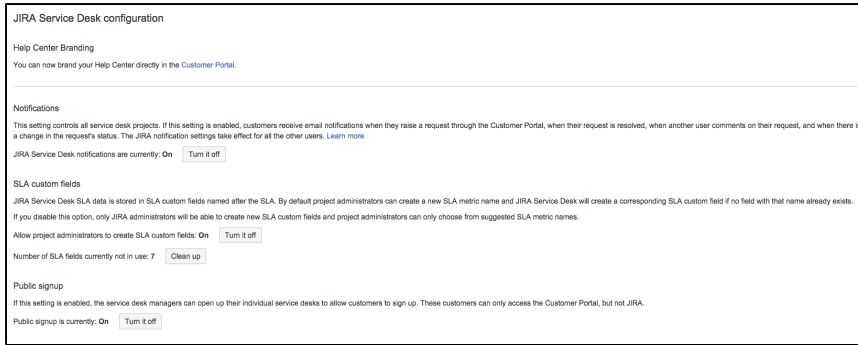
New metric names create new custom fields. You can restrict the creation of them to only be available for JIRA administrators.

1. Choose



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

2. Use the **Allow project administrators to create SLA custom fields** option. If the setting is disabled, service desk administrators can only select from existing metric names when creating SLAs.



Clean up unused custom fields

You can find out if there are SLA custom fields that are not used by any SLA metrics and clean them up with one simple click. On the configuration page, the **Number of SLA fields currently not in use** menu shows the number of unused custom fields if any. To delete them, click the **Clean up** button.

SLA usage notes

- Having the same user assigned to both the reporter and assignee roles may cause your SLA to work incorrectly.
- If you edit an existing SLA, JIRA Service Desk will re-index all the existing issues in the project; the re-indexing will ensure that the SLA status on the open issues reflects any changed criteria. All the historical SLA data for elapsed time will be recalculated to measure against the new metrics. Note that the SLA status is only recalculated for open issues and 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.
- 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. In other words, 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.
- Setting up a goal to be dependent on a different SLA is not recommended.

Reporting on SLAs

JIRA Service Desk provides robust [reporting tools](#) that you can use to track your team's performance against your SLAs. This page lists the SLA-specific JQL conditions you can use to query the SLA data in your service desk, as well as examples for creating some common JQL queries on SLAs.

- [State conditions](#)
- [Duration conditions](#)
- [Common SLA queries](#)

State conditions

State conditions are JQL functions used with operators = or != . For example:

```
"Time to resolution" = breached() or "Time to resolution" != breached()
```

Success/fail functions

Function name	with =	with !=
breached()	Gives all issues whose SLA last cycle (completed or ongoing) has breached (target goal failed)	Gives all issues whose SLA last cycle has not breached (for completed) or not breached yet (for ongoing cycles)

everBreached()	Gives all issues whose SLA has any cycle (current or past) that has ever breached.	Gives all issues whose SLA has all cycles (past or present) successful or not breached yet (if ongoing).
----------------	--	--

SLA state functions

This state addresses the last SLA cycle. This cycle can be completed (the stop event is reached) or ongoing (the stop event is not reached yet). When the cycle is ongoing, the cycle can be running or paused (if pause condition is true).

SLAs that have no cycles yet (the cycle has never been started) are not returned by these conditions.

Function name	with =	with !=
completed()	Gives all issues whose SLA last cycle is completed	Gives all issues whose SLA last cycle is not completed
running()	Gives all issues whose SLA last cycle is ongoing and not paused	Gives all issues whose SLA last cycle is not running (i.e. completed or paused)
paused()	Gives all issues whose SLA last cycle is ongoing and paused	Gives all issues whose SLA last cycle is not paused (i.e. completed or running)

Duration conditions

Conditions on duration are JQL functions used with operators <, <=, >, >=.

The '=' and '!=' operators are not supported.

These functions only apply to SLAs whose last cycle is ongoing (running or paused). Completed SLAs or SLAs without cycles will not be returned.

Example:

```
"Time to resolution" < elapsed(2h) or "Time to resolution" < remaining("2h 30m")
```

There are two duration conditions:

Function name	Description
elapsed()	Gives issues whose SLA last cycle match condition on elapsed time since start event.
remaining()	<p>This function gives issues whose SLA last cycle match condition on remaining time before SLA breaches current goal target duration.</p> <p>This function is implicit, meaning that</p> <pre>"Time to resolution" > 5h</pre> <p>is the same as</p> <pre>"Time to resolution" > remaining(5h)</pre>

Common SLA queries

This table lists some examples of common SLA queries; the conditions you use for your own reports will vary depending on the way your JIRA project is set up.

To find out	Query
All issues that are about to break SLAs	"Time to first response" < 1h and "Time to first response" != breached()
Issues that have plenty of time until they are due	"Time to first response" > 40h
Issues that have at least one breached SLA cycle	"Time to response" = everBreached()
The order of issues based on an SLA metric	project = SIS ORDER BY "Time to resolution"

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 critical 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 ✎ Edit

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

Start → **Pause on** → **Stop**

Begin counting time when → (Optional) Time is not counted during → Finish counting time when

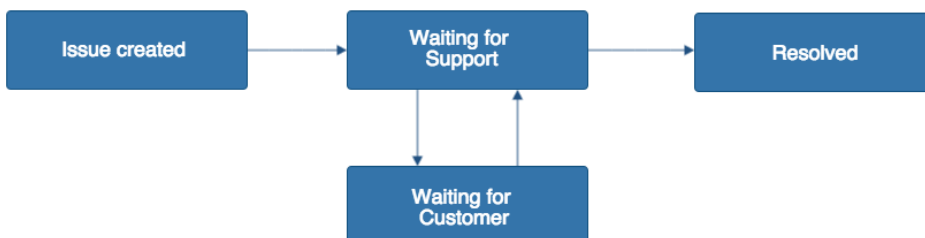
Issue Created → [] → Resolution: Set

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	Success this week
priority = Blocker or priority = Critical and labels = 24H	24 hrs	24/7 Calendar (Default)	
priority = Blocker or priority = Critical	24 hrs	9-5 working days	
All remaining issues	No target	9-5 working days	

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

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

Start → **Pause on** → **Stop**
 Begin counting time when → (Optional) Time is not counted during → Finish counting time when

Issue Created → 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	Success this week
All remaining issues	No target	

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 Access issues 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 Edit

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

Start → **Pause on** → **Stop**
 Begin counting time when → (Optional) Time is not counted during → Finish counting time when

Entered Status: Waiting for Support → [] → Resolution: Set

Issue Created → [] → Entered Status: Waiting for Customer

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	Success this week
issuetype = "Access"	2 hrs	
All remaining issues	24 hrs	

Example issue workflow



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

Managing SLA data

When new SLA metric names are created, new custom fields are created in JIRA to store them. The type of these custom fields is SLA CustomField Type. As a JIRA administrator, you have the following options to manage the SLA custom fields.

Related pages

- [Setting up SLAs](#)

Learn more about JIRA custom fields and how to [add a field](#).

1. Set whether project administrators can create new SLA metric names

New metric names create new custom fields. You can restrict the creation of them to only be available for JIRA administrators.

1. Choose



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

2. Use the **Allow project administrators to create SLA custom fields** option. If the setting is disabled, service desk administrators can only select from existing metric names when creating SLAs.

2. Clean up unused custom fields

You can find out if there are SLA custom fields that are not used by any SLA metrics and clean them up with one simple click.

On the configuration page, the **Number of SLA fields currently not in use** menu shows the number of unused custom fields if any. To delete them, click the **Clean up** button.

JIRA Service Desk configuration

Help Center Branding
You can now brand your Help Center directly in the Customer Portal.

Notifications
This setting controls all service desk projects. If this setting is enabled, customers receive email notifications when they raise a request through the Customer Portal, when their request is resolved, when another user comments on their request, and when there is a change in the request's status. The JIRA notification settings take effect for all the other users. [Learn more](#)

JIRA Service Desk notifications are currently: On Turn it off

SLA custom fields
JIRA Service Desk SLA data is stored in SLA custom fields named after the SLA. By default project administrators can create a new SLA metric name and JIRA Service Desk will create a corresponding SLA custom field if no field with that name already exists. If you disable this option, only JIRA administrators will be able to create new SLA custom fields and project administrators can only choose from suggested SLA metric names.

Allow project administrators to create SLA custom fields: On Turn it off

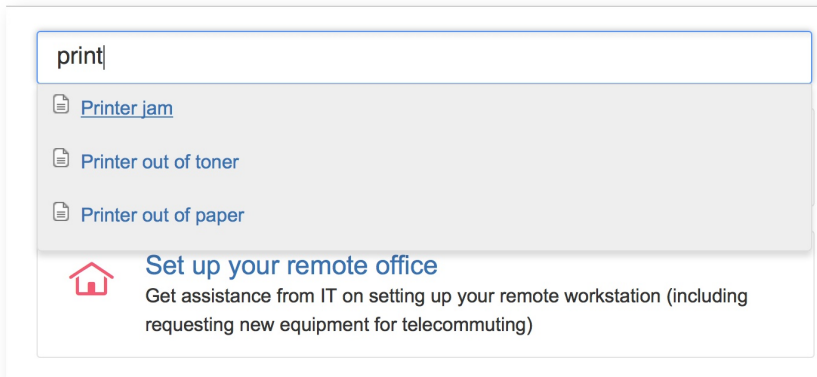
Number of SLA fields currently not in use: 7

Public signup
If this setting is enabled, the service desk managers can open up their individual service desks to allow customers to sign up. These customers can only access the Customer Portal, but not JIRA.

Public signup is currently: On Turn it off

Serving customers with a knowledge base

If you use Confluence, you can integrate its knowledge base capabilities with JIRA Service Desk to help customers find solutions on their own:



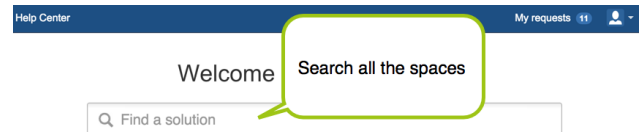
To integrate a Confluence space with your service desk project, log in as an administrator and go to **Project administration > Confluence KB**.

On this page:

- Searching on the global portal
- Restricting the topic search
- Creating knowledge base topics from an issue
- Integrating JIRA Service Desk with Confluence
- Troubleshooting issues with 2-Legged OAuth

Searching on the global portal

A search box appears on the landing page if you've connected Confluence with any service desk. Your customers can easily search for anything they need to find out across all the spaces of Confluence. If multiple Confluence servers are connected to service desks, only the **primary** one will be used by the search box on the landing page.



Restricting the topic search

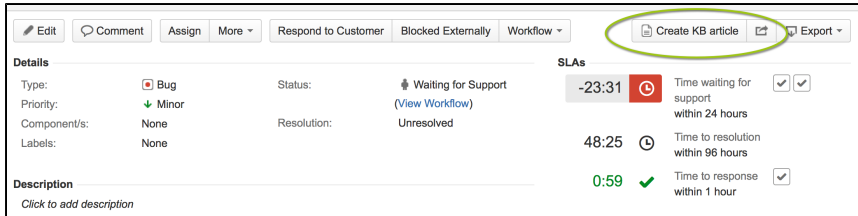
You can control how Confluence suggests topics for each request type using the **Request form search** section. You can control this in two ways:

- **Prevent Confluence from suggesting pages** - Select **No** in the **Search KB** column for the request type. For example, you might not want the "Get access to a system" request type to suggest pages since users have to request access through the Customer Portal.
- **Limit the pages that will be suggested** - In the **Restrict to articles with label** column, enter the labels that must be applied to pages in order for them to appear in the suggested page list. For example, you might want to only include pages with the label "purchasing" to appear when customers enter a "Request new software" request.

Tip: If you add label restrictions to a request type, these labels will also appear as the default labels for knowledge base articles [created from JIRA Service Desk](#) for issues based on that request type.

Creating knowledge base topics from an issue

If you link a Confluence space with a service desk, your team can create a knowledge base article based on an issue from the view issue page in JIRA Service Desk. Service team members can choose whether to create the article with a How-To article or Troubleshooting article blueprint; those blueprints can help your knowledge base expand with cleanly organized topics.



The issue title and description are automatically added to the new Confluence page as its title and body text. (Any images in the issue aren't copied over to Confluence.) If you've set up [label restrictions](#) on the request type the issue was based on, those labels are automatically suggested for the article.

Service desk members must have the [Add page permission](#) in the Confluence space to create a knowledge base article from an issue in JIRA.

Integrating JIRA Service Desk with Confluence

- You can connect JIRA Service Desk with Confluence 5.6 or later.
- If you use an installed version of JIRA and Confluence, they must be linked via an [application link](#) using OAuth.
- You must have [permission](#) to view a space in Confluence in order to select it as your knowledge base. If you don't have this permission, check with your Confluence administrator.
- In order to use the topic search from the Customer Portal, customers must be Confluence users with [permission](#) to view the space the or Confluence space must allow anonymous access.
 - If the Confluence space is set up to allow anonymous viewing, any user can search the service desk when they're putting in requests (in other words, they don't have to be Confluence users).
 - However, if the space viewing is restricted to certain users, customers must have the same username in Confluence in order to search the space from the service desk.

Notes on setting up a 2-legged OAuth application link

An application link lets two applications ask each other for data. One application consumes data from another application that produces data.

For JIRA Service Desk, the application that consumes data is JIRA and the application that produces data is Confluence. When configuring the application link for JIRA Service Desk, the incoming authentication tab inside Confluence must be correctly configured.

When configuring 2-Legged OAuth, it is important to understand that the two applications actually maintain independent sets of configuration information. The configuration in the incoming authentication tab of the application that produces data is different the configuration in outgoing authentication of the application that consumes data

- Confluence needs to correctly configure either "Allow user impersonation through 2-Legged OAuth" or specify a user in "Execute as" in the incoming authentication tab.
- JIRA only needs to enable 2-Legged OAuth on the outgoing configuration tab.

About 2-Legged OAuth (2LO)

Application links that use 2-Legged OAuth accomplish this communication in one of two ways. One way to handle this communication is to trust all requests made over the application link and grant those requests access to everything. The second way to have requests made over the application link is to pretend that each request is being made by an actual user. That pretend user is used to determine what permissions should be applied to the request. The user the application link picks depends on how the application link is configured. JIRA Service Desk uses the second way when making requests between applications.

- To enable a 2LO application link you need to enable OAuth and select "Allow 2-Legged OAuth" in both incoming authentication in Confluence and the outgoing authentication in JIRA.
- When you select 2LO, make sure that you also specify the user that is used by the application link when making requests.

How to choose the user that you want searches to be performed as

- If your applications have the same users and you would like 2-Legged OAuth to pretend your users are making the request as themselves: In the incoming authentication tab of the application that will serve the data, select "Allow user impersonation through 2-Legged OAuth". This must be checked in the incoming authentication tab of the application that will serve the data as it can only be configured by that application. Your users must have the same usernames in JIRA and Confluence.
- If your users do not have the same usernames in JIRA and Confluence, you can create a special user with correct permissions: In the incoming authentication tab of the application that will serve the data, add the user name you want to use in the text field beside "Execute as" to have all searches made over the link be performed as though that user made the request. The user name must be entered in the text field beside "Execute as" in the incoming authentication tab of the application that will serve the data as it can only be configured by that application.
Note: This setting only opens up the search restrictions for users who do not have the [permission](#) to view the space. When they open the Confluence pages by clicking the search results, they still do not get to see the pages due to the lack of the permission.
- If you want to retrieve data that is available to anonymous users, do not select "Allow user impersonation through 2-Legged OAuth" and do not add a user name in the text field beside "Execute as". The user that 2-Legged OAuth uses, in this case, is the anonymous user.

Troubleshooting issues with 2-Legged OAuth

You might see the following errors when connecting JIRA Service Desk to a Confluence knowledge base:

There was an error contacting Confluence. A possible cause of this could be an invalid Application Link. Another possible cause could be that the current user does not have access to Confluence. Please check that a valid Application Link to Confluence is set up and that you have access to Confluence and have the appropriate permissions for this action.

or

Client must be authenticated to access this resource.

These errors occur when the application link attempts to make a request in Confluence as a user that does not have permission to do so. In this case JIRA Service Desk is attempting to make a space search in Confluence. That search is being performed by the application link as a particular user and that user does not have permission to do the search.

To resolve the errors:

▼ Check if 2-Legged OAuth is configured correctly in Confluence

1. Open application link configuration in Confluence. You cannot make the required changes from inside JIRA applications when using 2-Legged OAuth.
2. Select incoming authentication.
3. Is "Allow 2-Legged OAuth" checked? If Confluence does not have 2-Legged OAuth enabled, requests made by JIRA applications that attempt to use 2-Legged OAuth are not processed.
4. Is "Allow user impersonation through 2-Legged OAuth" checked? In order for JIRA application users to search as though they were making the search in Confluence with their Confluence user permission, you must select this setting.
5. If "Allow user impersonation through 2-Legged OAuth" is not checked, is there a user name in the text field beside "Execute as"? When 2-Legged OAuth is enabled but "Allow user impersonation through 2-Legged OAuth" is not checked, all searches are performed as the user entered in the "Execute as" text field. If no user is specified in that field, all searches will be performed as an anonymous user.

▼ Check if 2-Legged OAuth is enabled in JIRA applications

1. Open application link configuration in JIRA, and then select outgoing authentication. Is "Allow 2-Legged OAuth" checked? If JIRA does not have "Allow 2-Legged OAuth" checked, then 2-Legged OAuth is not configured in JIRA applications.
2. If the incoming authentication tab only displays a single check box "Allow 2-Legged OAuth" and no

other check box it may suffer from a bug. This bug does not correctly enable 2-legged OAuth requests sent from JIRA applications when the application link is first created. You can properly enable 2-legged OAuth by doing the following:

- a. Uncheck "Allow 2-Legged OAuth" in the incoming authentication tab
- b. Click the "Update" button
- c. Check "Allow 2-Legged OAuth"
- d. Click the "Update" button

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 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.

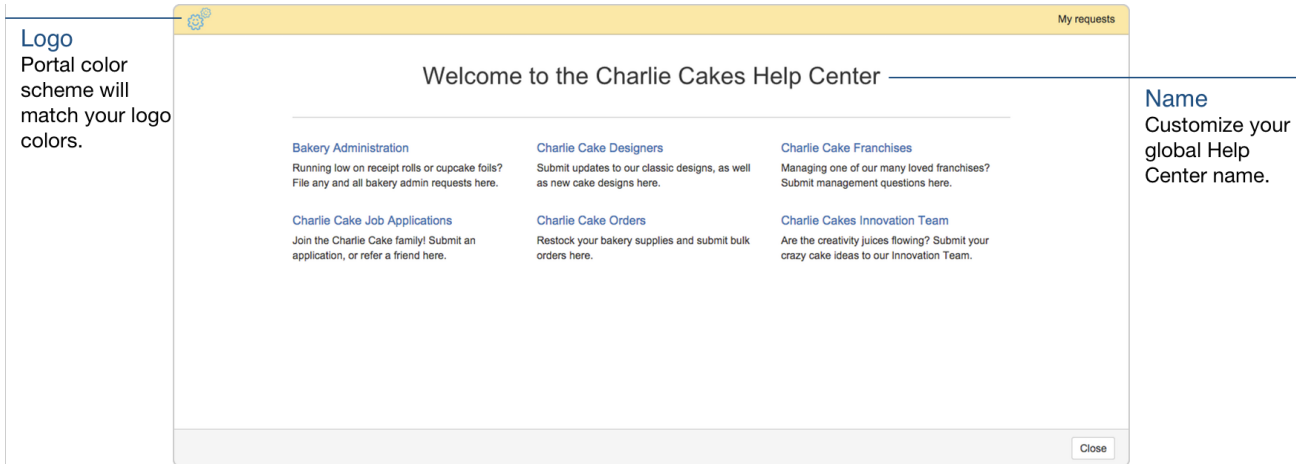
On this page:

- [Brand your customer portal](#)
- [Help customers find the request types they need](#)
- [Group related request types](#)
- [Set up a knowledge base](#)

Brand your customer portal

- Upload your company logo and your service desk can automatically generate a matching theme for your customer portals and global Help Center header.

- Name your customer portals and global Help Center, so your customers can easily identify your team's service desk.



Check out [Configuring the customer portal](#) to learn more.

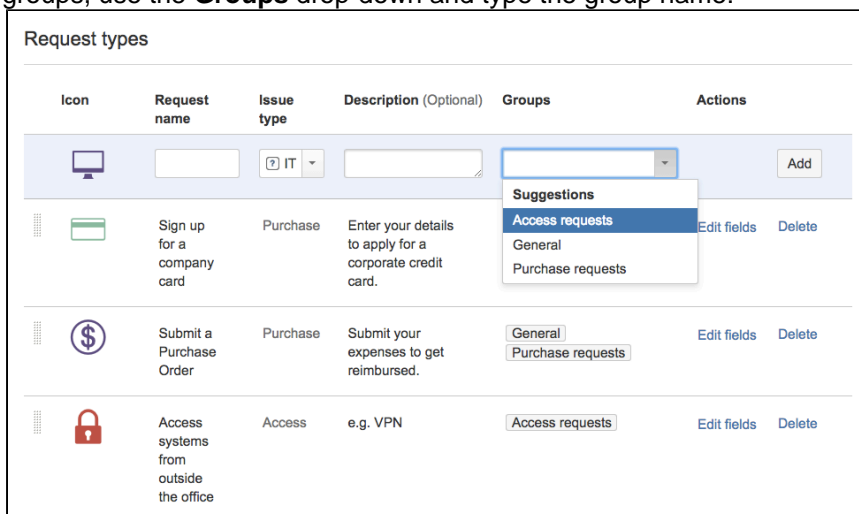
Help customers find the request types they need

- 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 (e.g. specify photo dimensions and format for the attachment field) with the **Field help** field.
- Use examples in your request type descriptions (e.g. '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.

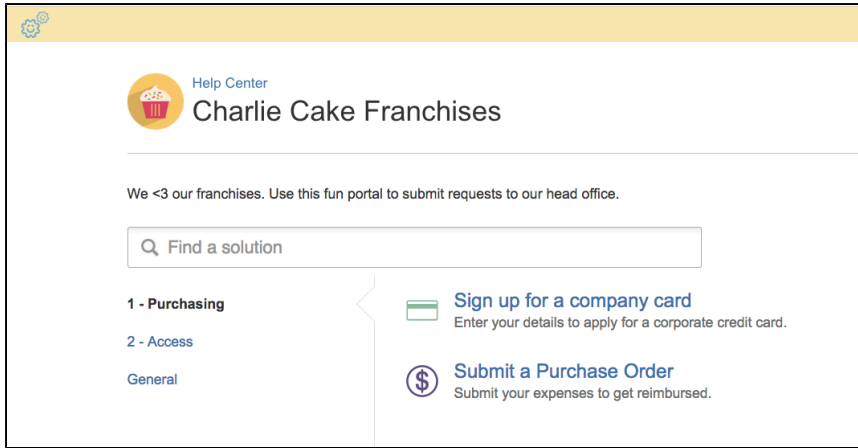
See [Setting up request types](#) for more information on naming request types.

Group related request types

- If you have a large number of request types, e.g. more than 7, customers will probably need to scroll to get to some of them. In this case, consider grouping some request types together. To set up groups, use the **Groups** drop-down and type the group name:



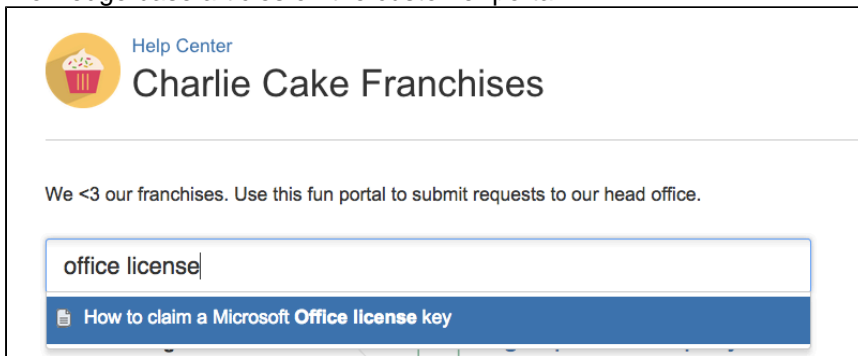
- You can also control the order in which request type groups appear on the customer portal by adding a number to the group name. Groups appear as tabs on the left-hand side of your customer portal:



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 [Serving customers with a knowledge base](#)

Best practices for IT teams using JIRA Service Desk

Overview

This best practice guide is for IT teams using JIRA Service Desk. By the end of this guide, you will understand how to use JIRA Service Desk for change and incident management, and have a working IT Service Desk project. Note that this guide references the latest version of JIRA Service Desk Cloud.

On this page:

- [Overview](#)
- [Before you begin](#)
- [Set up a project](#)
- [Set up your service catalogue](#)
- [Assign component leads](#)
- [Edit your workflow](#)
- [Success!](#)

Before you begin

- Sign up for a JIRA Service Desk Cloud [evaluation](#), or use an existing JIRA Service Desk Cloud

- instance
- Log in as an administrator

Set up a project

In this step, you'll create a new IT Service Desk project.

1. Sign into JIRA Service Desk as an administrator.
2. Select **Projects > Create Project**.
3. Select the IT Service Desk project type and hit **Next**.
4. Name your project "Team Help Desk" (which will automatically fill in the project key field) and select **Submit**.

Your new project comes with preconfigured request types, queues, workflows, and reports to help you and your team manage incidents, problems, and changes.

Set up your service catalogue

Your new project comes prepackaged with Service Request, Change, Incident, and Problem issue types. The Service Request, Change, and Incident issue types can be associated with request types that customers see on the customer portal. Request types are a lightweight way to set up a service catalogue in JIRA Service Desk. In this step, you'll view the default request types and learn how to customize the request type fields and workflow statuses.

1. In your new Team Help Desk project, select **Project administration > Request types**. You'll see the default request types and groups. Groups are essentially labels that let you organize the tabs and order in which request types appear on the customer portal.
2. To edit the name and description of a request type, simply click the corresponding request type name or description field and select **Update**.
3. For the "Report a system problem" request type, select **Edit fields**. Visible fields appear on the request type form that customers fill out, while hidden fields are only viewable to your team. Click the "Select a system" visible field and note that this issue field displays all components in your project. We'll review components in the next step.
4. On the Fields tab, select **Add a field** to add the Linked Issues field and select **Apply**. You will be prompted to set a preset value. Set the preset value to "Linked issue relates to", so agents and other internal users can see all other issues marked as related to this one.
5. On the Workflow Statuses tab, you can rename the default workflow status name to something more customer friendly (e.g. rename Resolved to Done). You can also select **view workflow** to view and edit the workflow this request type users (in this case, the Incident workflow for Incident issue types). We'll review more about workflows in a bit.

Assign component leads

In addition to the default request types and workflows, JIRA Service Desk comes with default components. You can use components to group issues related to the same system or application, and assign component leads so each system manager stays informed. In this step, you'll view the default components in your project and see how an issue tagged with a component is created and managed.

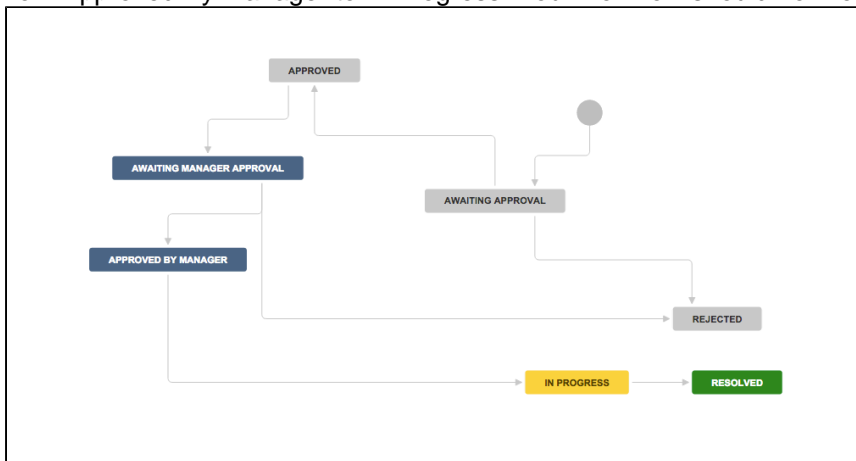
1. In your service desk project, select **Project administration > Components**. You'll see the default list of components. Simply click the component name and description to edit these fields. You can also select a component lead (e.g. the team lead who manages a system) and a default assignee (e.g. the team lead or project lead who will automatically be assigned to any issue tagged with that component).
2. Select **Back to project** and open the Customer Portal link from your project sidebar.
3. You can now create a test request from a customer's perspective. Select the "Report a system problem" request type.
4. Fill in the issue summary and description (e.g. Printers broken on level 3). Click the Select a system field to see a list of available options and choose "Printers".
5. Select **Create** to finish creating your test request, and then close the customer portal to return to your service desk project view.
6. From your project sidebar, select **Queues** and open the test issue you just created. You'll notice

"Printers" in the issue component field. When you click the "Printers" component, you'll be taken to the Search Issue screen which displays all issues in this project with the same component.

Edit your workflow

Now that you're a bit more familiar with how to service desk request types and components, let's look at editing the change request workflow to require two managers to approve the request before work can start.

1. In your service desk project, select **Project administration > Change** (under Issue Types). You'll see the default workflow used for all change requests in this project. In this default workflow, a change request is created and then awaits approval. If the request is approved, the request is transitioned to in progress, and then resolved when work is completed.
2. Select **Edit Workflow** and select **Add status**. Name your new status "Awaiting Manager Approval" and leave the category as To Do.
3. Add another status named "Approved By Manager" and leave the category as To Do.
4. Select **Add transition**, and make the transition start from the new Awaiting Manager Approval status and go to the Rejected status. Name your transition "Reject by manager". Select **Add**.
5. Add a new transition from the Approved status to the Awaiting Manager Approval status, and name this transition "Request manager approval".
6. Create one more new transition from the Awaiting Manager Approval status to the Approved By Manager status, and name this transition "Approve by manager".
7. Drag the Start progress transition so that instead of leading from Approved to In Progress, it leads from Approved By Manager to In Progress. Your workflow should now look something like this:



8. Select **Publish** to finalize your workflow changes.

Your change request workflow now requires two approvals before work can start. Note that you can [create a Managers project role](#), and add a [condition](#) (e.g. User is in project role) to the Reject by manager and Approve by manager transitions. This condition will only allow users in the Managers project role to action these transitions. Check out [Managing your workflows](#) for more information on adding statuses and transitions to an existing workflow.

Success!

You now have a working service desk project with a customizable service catalogue, components to better track issues specific to systems your team managers, and a new workflow for managing change requests. Let us know how you use JIRA Service Desk for change or incident management by creating a ticket in the Feedback project in our [public issue collector](#).

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 [Atlassian Answers](#).
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 [Atlassian Answers](#).
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>)

I don't like the way something works

Raise a suggestion. (<https://jira.atlassian.com>)

Something else?

If you need help with something else, raise a support request.

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 Answers

[Atlassian Answers](#) 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 Atlassian Answers, 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](#).

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.