Documentation for JIRA Core 7.1
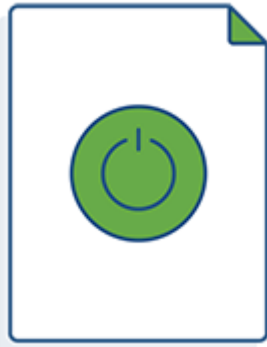
Contents

JIRA Core documentation

JIRA Core is a customizable workflow solution that simplifies any business process such as change management, approvals, asset management and more. Try it here.

Getting started	Guide for project admins	Guide for users
JIRA Core overview	Getting started as a project admin	Getting started as a user
Getting started with JIRA Core	Administering a project	Working in a project
Release notes		

JIRA Core overview

Plan, track and report any business project with JIRA Core, keepng your team organized, effective, and efficient.

What's trending?

Video	Blogs	Topics
Everything you need to know about JIRA Core	Say hello to JIRA Core!	Tips 'n tricks for JIRA Core
Say hello to JIRA Core! 2/2 the Q&A	How to set up business workflows	Building workflow: a working example
	Make issues work for your team	What makes up JIRA Core: the basics
	Project status at a glance	Ideas and use cases for JIRA Core
	Helping Marketing and Software teams work together	
	3 steps to Marketing zen	

What makes up JIRA Core?

JIRA Core is a JIRA application that provides you with a workflow management system that you can use for many things, including running projects, tracking assets, and basically anything that requires work moving

through a workflow. JIRA Core can be customized to suit your needs, and can also be extended and linked with other applications to provide you with the perfect solution to track all your work in one place.

Understanding the basics of JIRA Core will help you get the most out of your application. This page will explain the core terminology and functionality that makes JIRA Core the application you need to help you manage your work efficiently.

Ready to get to grips with the basic concepts? Read on!

Issues

Issues are the work packets in JIRA Core. Each issue can be further defined by assigning the issue an issue type. For example, if you're running a project in an office, issues could represent the tasks you need to do to complete that project. Each issue type would be a type of task, like administration task, filing task, or create document task. If you're using JIRA Core for asset tracking, an issue could represent an asset (or inventory item) and the issue type could be the types of assets (laptops, monitors, printers etc.). Issues then progress (or move) though JIRA Core via a associated workflow that dictates what can and can't (or more correctly, what should and shouldn't!) happen to that issue.

So how do I group issues? What if I want a construction project, and I also want to track my assets for that project in JIRA Core, how can I do that? Well, you use projects!

Projects

Projects are a way to group your issues, and apply a set of defaults. These defaults make sure all your issues have the information that they need to be progressed and tracked through your workflow. Each project can have an administrator, who is typically the project lead, and they're responsible for administering the project.

JIRA Core users may work with issues in one or several projects, and would complete the work required to progress the issues through their workflows.

Great! You now know issues are the work, and they're grouped in projects. But what's the workflow, and how does it effect the issues?

Workflows

Workflow dictates how an issue can be progressed in a project. The workflows can be as simple or as complex as you need them. Workflows are often modelled on existing processes, and are made up of statuses (or steps) and transitions (movements between statuses). When you create an issue, it will automatically be assigned a workflow and a status on that workflow. Where it can move to is defined by the transitions that exit that status. An example of the default workflow that ships with JIRA Core is shown below:

WORKFLOW

JIRA Applications

JIRA Core is one of several JIRA applications. Each application has its own distinct features that makes it suitable for servicing the market it's designed for. For example, JIRA Software is designed for agile software developers, and its features include agile boards, software specific project types, and better integration points with other developer tools.

You may have access to more than just the JIRA Core application, and you can view this list of applications and feature access to gain a better understanding of the power of JIRA!

So that's the basics. JIRA Core is functional straight 'out of the box', but if you think you'll need to know a little more, to get some ideas on what you could do, take a look at this guide to see what's possible with JIRA Core.

See what's possible with JIRA Core

JIRA Core comes ready to use right out of the box, and the project templates are a great way to get your first project up and running. In this section of the documentation, we'll take you through the project templates, and

give you some ideas of how you may want to customize your projects so that they suit your needs. We'll also give you some tips on how to get the most out of your projects and issues.

☀ JIRA Core

| **Sign up for a free trial** | Try me! |

Task management

The Task management template sets you up with a simple workflow, easy to understand issue types, and the right set of fields on your issues to help you get your work from To Do to Done in the quickest way possible.

Learn more about the default configuration and specific use cases...

Project management

The Project management template has a slightly more complex workflow, and is great for projects that require a little more work. Including an In Progress status allows you to work on tasks over longer periods and show work in progress.

Learn more about the default configuration and specific use cases...

Process management

 The Process management template ships with our most complex workflow, designed to give you an idea of what you can create with JIRA Core to mimic your own business process or work flow.

Learn more about the default configuration and specific use cases...

HR

If you're working in HR, you may want to have a look at a few ideas of how you can use JIRA Core for your projects, and configure them further to suit your needs.

Learn more about using JIRA Core for HR...

Marketing

If you're a marketing whiz and looking to move to the next level by managing your work more efficiently, making sure all your I's are dotted and T's crossed, have a look at how you can use JIRA Core to help you work smarter and more effectively.

Learn more about using JIRA Core for marketing...

Operations

Thinking about trying to set up some workflows for operations? Wondering how JIRA Core could help you keep things in one place while still letting you apply the workflow you need? Take a look at our operations example for some ideas.

Learn more about using JIRA Core for operations...

Finance

If you work in finance and struggle to keep tabs on everything required for reporting, budgeting, invoicing.... you know what you do better than we do! Have a look at how you could potentially leverage JIRA Core to make sure all your reports are done on time, all your invoices are issued and followed up, and you can check up on where you stand at any given time.

Learn more about using JIRA Core for finance...

Legal

Keeping tabs on everything you're working on can be hard, and when you also need to make sure things are reviewed and approved by the relevant parties, things can get harder to manage. JIRA Core can help you manage your work and keep tabs on it at each stage through the workflow.

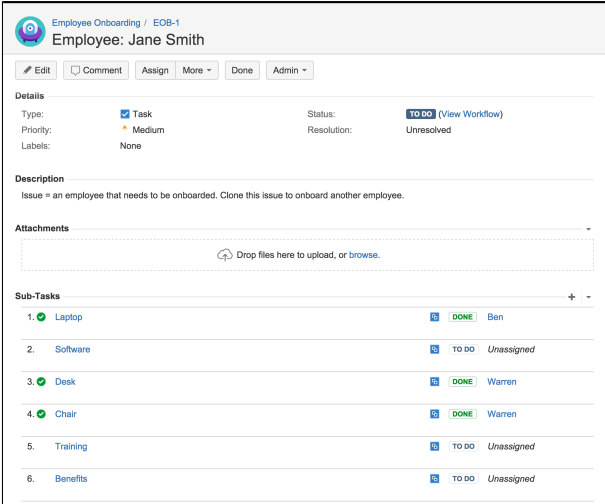Learn more about using JIRA Core for legal...

Using JIRA Core for HR projects

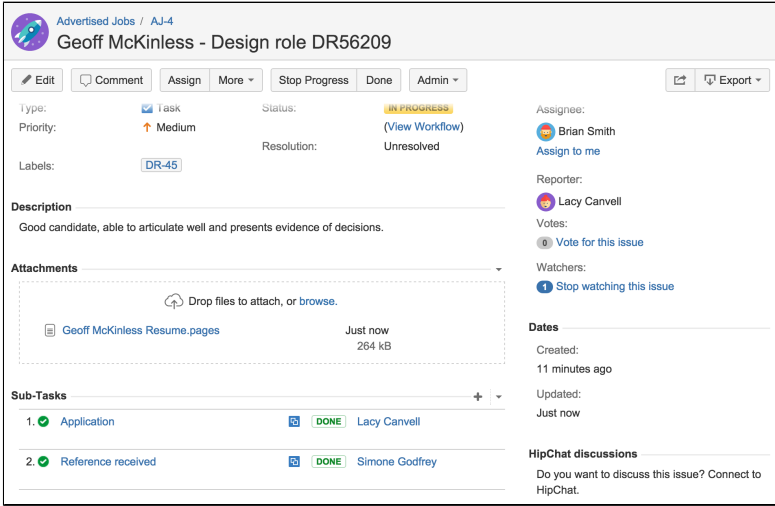 So you're thinking about using JIRA Core for your HR tasks? Maybe you're

thinking about using it to progress applications? Or maybe you'd like to make sure all your new staff are set up for success by setting up a flow for onboarding? We can help you get started, show you some tweaks you can make, short cuts you can take, and customizations you might want to think about to make sure your project reflects how **you** work, not how we think you should work! We'll list out a few scenarios here, and give you some suggestions on what you may want to do to make your project a success.

✖ JIRA Cor

| **Sign up for a free trial** | Try me! |

Let's look at onboarding...

Use case	**Onboarding** - setting up your new starters for success from day one, week one.
What Business project should I start with?	**Task Management** - This gives you a straightforward workflow of To do and Done, and accompanying issue types of Task and Sub-task.
How can I do this?	1. Create a task for each new employee 2. Create a sub-task for everything that needs to be done before the person starts. You may need to make sure a desk and chair is available, allocate a laptop, or make sure a building access card is granted. 3. Assign each sub-task to the relevant person responsible for making sure it's done. They can complete the sub-task when they've completed their task. 4. Once all these sub-tasks have been completed, you can mark the parent task as Done! Your new employee is ready to go!
How would this look?	
Can I make this easier?	You may be thinking, "Hold on a minute, I don't want to have to create a bunch of sub tasks every time I need to onboard someone!" Well, you don't. Set up a task with all the associated sub-tasks, and name it Onboarding. Whenever you need to onboard a new employee, find the task and use JIRA Core's clone functionality to make a brand new task with all the associated sub-tasks. By renaming the task you'll know which new employee each task belongs to. You can even assign the sub-tasks to the relevant people, so that when you clone the task, JIRA Core will make sure the relevant people are assigned to their tasks automatically.
Other customizations	You may decide that your onboarding should be more extensive, you may decide that instead of just getting the physical bits and pieces together for a new employee, you may want to also ensure that in their first week the get to have lunch with their team, sign off on all your statutory training, or get a building tour. You can customize your projects workflow to suit your needs, assigning the parent task to the person responsible for each step. You could decide to customize the fields available on the task, to make sure you record the information you need along the way (scores in training, asset numbers of any equipment that's been provided). You set up JIRA Core to mimic how you do onboarding. It can be as complex or as basic as you like!

Got the hang of how JIRA Core can be used? Let's take a look at another HR example, let's see how you could potentially set up a project to track job applications you receive.

Use case	**Application tracking** - You advertise for a role that's become available, and you start receiving applications in the mail, through your website, and via email.
What Business project should I start with?	**Project Management** - This gives you a workflow of To do, In progress and Done, and accompanying issue types of Task and Sub-task.
How can I do this?	1. Create a task for each candidate applying for a role. 2. Add any accompanying CV's, resumès or paperwork as an attachment to the task. 3. Depending on your hiring process, if the candidate is unsuccessful they may go directly to Done, or if you need to perform further tasks, like back ground checks, interviews, or wait for additional paperwork, the candidate may go to In progress. 4. Once you have a successful candidate, move them to Done and add a comment or label to say they've been successful.
How would this look?	
Can I make this easier?	Yes, you can! If you've integrated JIRA Core with your email system, you can make sure that incoming emails regarding job applications are automatically created as tasks. Any accompanying attachments on the email will also be added as attachments to the task. Ever wondered why some job advertisements ask for your name, the position you're applying for and job reference in the email subject line? It's because this is exactly what they're doing, taking those emails and automatically converting them to "something" they can track and process! Why not use JIRA Core to do it for you.
Other customizations	We know that processing a candidate for a role is never as easy as it sounds. Depending on the role, and any requirements you may need to do several interviews, background checks, make sure you've got the correct paperwork signed and returned... the list goes on! JIRA Core allows you to customise the workflow to suit this, and you can assign the task to the relevant person at each stage. Want to use the same project for all your job applications, but worried after a while it'll get too large with too many tasks open? Well you can help sort your issues by assigning them defined labels for each open role. This will help you search the tasks easily and pick out the tasks that relate to the role you're hiring for. You could also add components to the project, which are like a grouping for your tasks. You can use components to automatically assign someone to each task that's placed within that component.

These are just some ideas of how JIRA Core can help you with your process, and make it more visible and manageable.

Using JIRA Core for Marketing projects

You work in Marketing, and your company is starting to win more contracts.

There's a lot of work that needs done for each contract, and it can vary greatly! You're starting to lose visibility and need the ability to bring everything together and track the work that needs done, who's responsible, and when it's due by. You've heard of JIRA Core and you decide to give it a whirl... but where to start?!

Let's take a look at what you could consider doing...

Use case	**Marketing campaign** - you need assets, locations booked, websites created, approval to g live... the list goes on!
What Business project should I start with?	**Project Management** - This gives you a workflow of To do, In progress and Done, and accompanying issue types of Task and Sub-task.
How can I do this?	1. Create a new task for each item of work that needs done. Give it a descriptive title to make sure everyone knows at a glance what it's for. 2. Add sub-tasks for any additional work that may need done, making sure you break big chunks of work into more manageable pieces. 3. When the sub-tasks are done, you can complete the parent task. And your work is done
How would this look?	
Can I make this easier?	Don't forget you can add comments to your tasks, and mention people in the comments by using the "@" mention. This makes sure everyone is aware of the work and tasks that are needed. When using tasks and sub-tasks to track your work, it's also useful to remember yo can view the Activity and Statistics page for your project by clicking on the project name in your sidebar. These views give you a breakdown of the issues in your project so you always know where you stand. If you need more detailed info, you can also create you own dashboard with all the statistics you need, and add a link to the dashboard in your sidebar.
Other customizations	Marketing tasks can be very diverse and very specific in their requirements. You can customize your issues and make various fields 'required' to ensure they're always complete correctly. You can set the fields in various ways (drop-down, multi-select, free text etc.) to make sure whoever creates the tasks can only select from your pre-defined selections, or can complete the information as they see fit. Notifications can also be customized so that th right people are always updated when they need to be. Make sure the tasks and project are set up they way **you** work.

Getting the hang of how JIRA Core can be used? Talk to your administrator about more ways you can customize your project to make sure it suits how you work.

Using JIRA Core for Operations projects

We all know operations can cover a large and varied amount of work, and JIRA Core allows you to organize that work, no matter how simple or complex. JIRA Core workflows can be customized to allow you to move through various stages on a non-linear path. Each step in the workflow is a status, and you move between statuses via a transition. Your transition could bypass several statuses to get you directly from To do to Done, or it could take you down a path that means once you select any other status other than Done, you have to complete more statuses to progress to Done.

JIRA Cor

Sign up for a free trial

Try me!

Sound a little complicated? Let's look at an example, say you run a small manufacturing business making bespoke metal parts. Some metal parts are simple, and all they require is one process, such as simple cutting with a laser, to complete the part, and some metal parts are more complex, requiring a series of operations to produce the finished part. How could you control these in one project?

Use case	**Manufacturing** - metal fabrication
What Business project should I start with?	**Project Management** - This gives you a straightforward workflow of To do, In progress and Done, and accompanying issue types of Task and Sub-task.
How can I do this?	1. Create a task for each new piece of work that comes in. 2. Add a description that states what's needed in terms of raw material, and add any supporting drawings as attachments. 3. Create a sub-task for everything that needs to be done separately. You may need other parts cut and folded, or special parts ordered in. 4. Assign each sub-task to the relevant person responsible for making sure it's done. They can complete the sub-task when they've completed their task. Make sure you have these sub-tasks completed before advancing the main task. 5. Once all these sub-tasks have been completed, you can progress the parent task, and mark it as done once you have completed the work!
How would this look?	
Can I make this easier?	Well in this case, the flow is pretty straightforward. While work's in progress, the task is in progress, and when it's complete the work is done. Adding comments would be a great way keep track of the work, and where it is in the process. Assigning tasks and sub-tasks to the relevant parties helps make sure everyone knows when they have something to do!

Other customizations	JIRA Core's workflows can be configured to suit your process, so in this case you may want to track all the work on one task. You could add statuses of all the main stages in metal fabrication, like Cut, Laser, Punch, Fold... right through to Paint, Inspection and then Done. If any stage isn't required, you can simply transition the task straight through to the next required stage. You could also customize the issue types, making a different sub-task for each stage, as some parts could probably progress through fabrication independently. You can even set up separate workflows for different issue types in the project! It's all about how **you** work, and how **your** team likes to do things. JIRA Core just lets you do it and keep track of it.

Using JIRA Core for Finance projects

Finance teams can use JIRA Core to manage their processes and work, and making a few tweaks to a standard project template will make sure it suits the needs of you and your team. You could use a simple workflow when you need to generate invoices, or you could use a more complex process workflow for end of year financial reporting. Make JIRA Core your one stop shop for all your financial process needs.

✗ JIRA Cor

Sign up for a free trial	Try me!
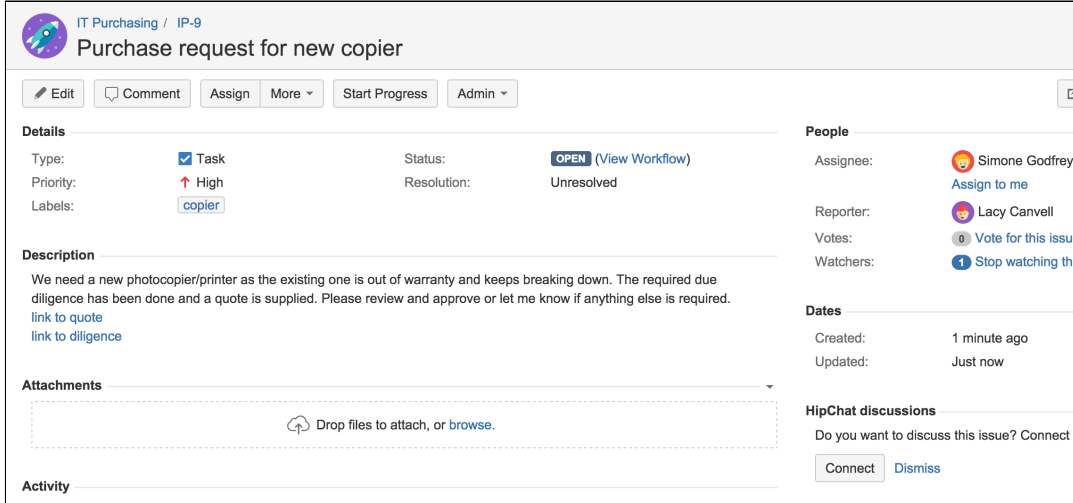
Let's take a look at a simple invoice workflow...

Use case	**Invoice tracking** - make sure you know when to generate your invoices, and when to chas them.
What Business project should I start with?	**Project Management** - This gives you a workflow of To do, In progress and Done, and accompanying issue types of Task and Sub-task.
How can I do this?	1. Create a task for each invoice that needs to be generated. 2. Once you've generated the invoice and sent it, move the task to In progress, and add a details that need to be recorded to the comments of the task. 3. You can set a due date for when the invoice should be paid, or a date for when you war chase it. 4. Once the invoice has been paid in full, you can move the task to Done!
How would this look?	Invoice issuing and receipts / IIR-25 **Issue invoice to Adamas Inc.** Edit Comment Assign More ▾ Stop Progress Done Admin ▾ **Details** Type: ☑ Task Status: **IN PROGRESS** (View Workflow) Priority: ↑ Medium Resolution: Unresolved Labels: None **Description** Issue statement for work done to date on website. **Attachments** Drop files to attach, or browse. 📄 Adamas Inc. invoice0934590.pdf 21 kB 2 minutes ago **Activity** All Comments Work Log History Activity **People** Assignee: Lacy Canvell Assign to me Reporter: Simone Godfrey Votes: 0 Vote for this issue Watchers: 1 Stop watching this issue **Dates** Due: 28/Oct/15 Created: 5 minutes ago Updated: 1 minute ago **HipChat discussions** Do you want to discuss this issue? Connect to HipCh Connect Dismiss

Can I make this easier?	If you need to make sure you record specific information for each invoice, such as the invoi[ce] number, add a field to the issue type Task called 'Invoice number'. This gives you a field yo[u] search by when filled in, so it's easy to find the issue related to the invoice number you hav[e] can also add any attachments or emails to the issue to record any correspondence. You ca[n] these along with any required comments. If you have one person who controls your invoicin[g] you can set them as the project lead to ensure they always get alerted when an issue is cre[ated]
Other customizations	While you can add a field to the issue type to make sure you have a specific field to record [the] invoice number in, you can go one step further and make that field required when you want [to] move the task from To do to In progress. That means without a valid invoice number being [in] issued, you can't move the task to In progress. You could do something similar to record wh[en] the payment has been made, you may decide you need to have a copy of the credit to your [credit] company account to close the task off as Done.

What about something a little bit more complex? Do you have a simple purchasing flow like Order item > Item received? Then you'll be happy with a simple workflow like Task management where a task is either To do or Done. But what if your process is more complex, like Item requested > Business case created > Business sent for review > Approval received > Purchase order created? Using JIRA Core's Process Management project will give you an initial workflow to get started, and when your process matures, you can mature your workflow to make sure it always does what you need.

Use case	**Purchasing** - Creating and submitting a business case for approval
What Business project should I start with?	**Process Management** - This gives you a workflow of Open > In Progress > Under Review [>] Approved > Done, and accompanying issue types of Task and Sub-task.
How can I do this?	1. Create a task for each new business case for approval. 2. Populate the task with all the relevant details, attaching any documents, spreadsheets [or] additional information that you think will support the business case. 3. Moving the task to Under Review and assign it to your approver (this could be your ma[nager,] maybe a member of the purchasing department). 4. Once approved, the final stage could be to either raise a purchase request, or possibly [pass] the data to another system to get the purchase completed. Either way, the issue is Don[e and] completed!
How would this look?	
Can I make this easier?	Yes! You can use a lot of JIRA Core's functionality, especially comments and mentioning ot[her] people (collaboration is alive and well!), to make sure you have all the information you nee[d to] complete the business case. Assigning the task to the relevant people at the right time as it [moves] through the workflow makes sure it'll always appear on their to do lists, and if notifications a[re on] they'll even receive an email to let them know it's ready for them.

Other customizations	You may decide to change the workflow. You may need to add in a second approver. You r want to consider assigning the task automatically to the correct person as it progresses thro workflow. You may consider adding different issue types for each type of business case tha accept, and assigning each issue type a specific, custom workflow (with the correct assigne status of course!). JIRA Core allows you to achieve this.

Think of anything else you may need to make sure you have everything recorded for your financials? Check out the Administering a project section of the docs for some ideas on other customizable areas of JIRA Core.

Task management

The task management project template sets you up with the most basic workflows, and is designed to allow you to create issues and get them completed with the minimal of fuss. Using JIRA Core's functionality you can involve the members of your team that you need to, and track your work using your built in reporting, your dashboard, and custom filters based on your searches.

XJIRA Cor

Sign up for a free trial   Try me!

Out of the box, your task management project comes with:

Workflow	To Do > Done
	WORKFLOW
	TO DO → DONE
Issue types	Task and sub-task. A sub-task must belong to a parent task.
	ISSUE TYPES
	☑ Task
	⊞ Sub-task
Issue fields	Summary, Issue Type, Reporter, Attachment, Due Date, Description, Assignee, Priority, Resolution and Labels
Resolutions	Done, Won't Do, Duplicate and Cannot Reproduce
Priorities	Highest, High, Medium, Low and Lowest

There's a lot you can do with a task management project, and with a few customizations, there's a lot more you can do to! Have a look at the example for a few ideas of how you can use it, and customize it to suit your needs.

Using JIRA Core for Legal projects
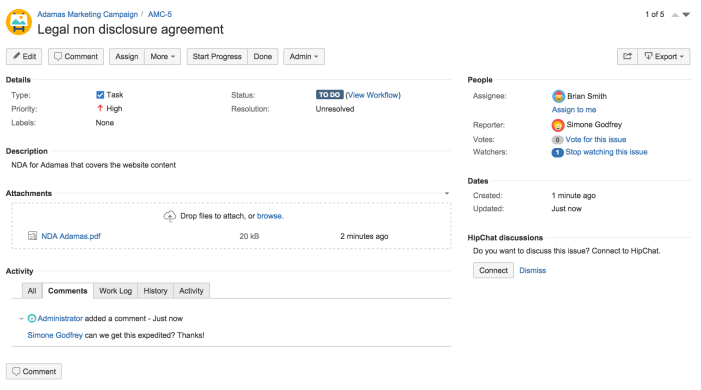
Whatever your business requirements, if there's a process or a workflow involved, JIRA Core can help you get things in a more orderly state. Your organization may have a legal team, who are responsible for legal documentation, licensing, non-disclosure agreements and all the things required to make sure your organization complies with all required local, national and international regulations. Keeping track of all the requests,

XJIRA Cor

Sign up for a free trial   Try me!

requirements, reviews and outstanding approvals can be quite tough, but by setting up a JIRA Core project and configuring it to how you work can help manage the work, and keep you on top of things.

Let's take a look at non disclosure agreements, and how you could set up a project to keep track of the work you're doing.
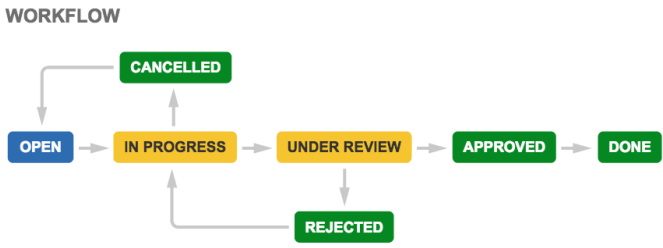
Use case	**Documentation requests** - get your documentation requests in one place, to be actioned when required.
What Business project should I start with?	**Project Management** - This gives you a workflow of To do, In progress and Done, and accompanying issue types of Task and Sub-task.
How can I do this?	1. Create a task for each document you need create. 2. Attach the document to the task. Remember to keep the document versioned in your word processing package! 3. Once the document is put together, get feedback in the comments section of the task. 4. When approved, transition the task to Done and finalise the document!
How would this look?	
Can I make this easier?	You may think there's something missing here, there's no review process? Well you can achieve this in 2 ways. You can modify the workflow to add an 'In review' status, or you could add a resolution to the 'Done' status of 'Rejected'. That means that the person who prepared the document would need to transition the task back to In progress to work on it again.
Other customizations	Legal documents are very important, and need to be accurate and applicable to each case. It's also important to make sure the correct person reviews and approves the documentation. Your workflow needs to make sure this happens. You could add a condition to the workflow transition that means only a specific person can transition it. That could be your approver. Remember, JIRA Core allows you to define your process. It can be as complex or as basic as you like!

Process management

The process management project is an example of a project with a slightly more complex workflow that you could set up in JIRA Core. The workflow shows an example of how you could create your work, progress it, have it reviewed, approved, and eventually completed. If you're interested in getting more from your workflows, you should review Working with workflows for some ideas on what you could do with your workflows to get the most out of JIRA Core.

Sign up for a free trial | Try me!
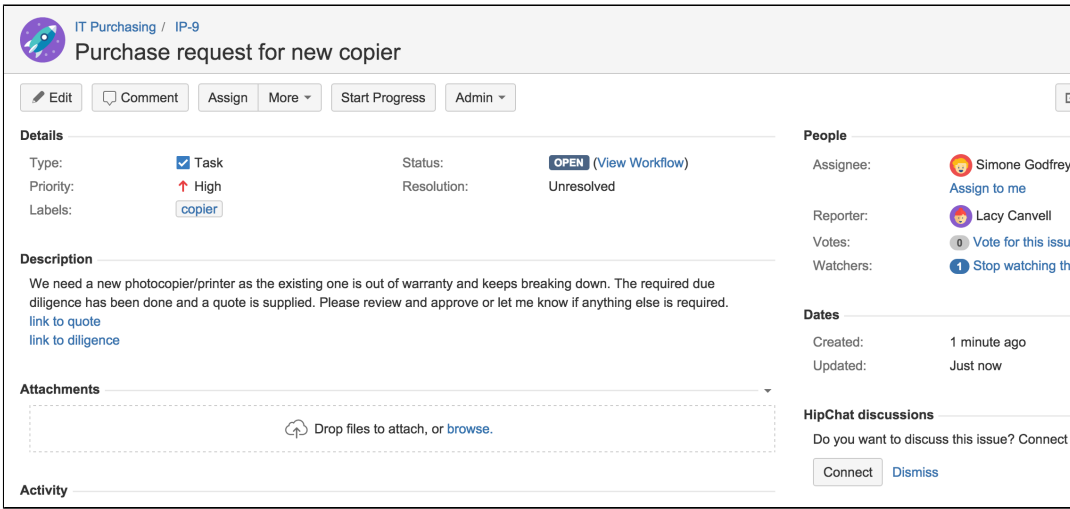
Your process management project is set up by default like this:

Workflow	In a perfect world, your work would move from Open > In Progress > Under Review > Approved > Done, but there's options for the work to be returned to previous statuses via Rejected and Cancelled. **WORKFLOW** CANCELLED OPEN → IN PROGRESS → UNDER REVIEW → APPROVED → DONE REJECTED
Issue types	Task and sub-task. A sub-task must belong to a parent task. **ISSUE TYPES** ☑ Task ⊞ Sub-task
Issue fields	Summary, Issue Type, Reporter, Attachment, Due Date, Original Estimate, Remaining Estimate, Description, Assignee, Priority, Resolution and Labels
Resolutions	Done, Won't Do, Duplicate and Cannot Reproduce
Priorities	Highest, High, Medium, Low and Lowest

Processes can be as complicated or as simple as you need them to be, and JIRA Core allows you to achieve just that. For more ideas on what you could do, check our example for some pointers.

Use case	**Purchasing** - Creating and submitting a business case for approval
What Business project should I start with?	**Process Management** - This gives you a workflow of Open > In Progress > Under Review Approved > Done, and accompanying issue types of Task and Sub-task.
How can I do this?	1. Create a task for each new business case for approval. 2. Populate the task with all the relevant details, attaching any documents, spreadsheets additional information that you think will support the business case. 3. Moving the task to Under Review and assign it to your approver (this could be your man maybe a member of the purchasing department). 4. Once approved, the final stage could be to either raise a purchase request, or possibly the data to another system to get the purchase completed. Either way, the issue is Don completed!

How would this look?	IT Purchasing / IP-9
	Purchase request for new copier
	Edit Comment Assign More ▾ Start Progress Admin ▾
	Details
	Type: ☑ Task Status: **OPEN** (View Workflow)
	Priority: ↑ High Resolution: Unresolved
	Labels: copier
	Description
	We need a new photocopier/printer as the existing one is out of warranty and keeps breaking down. The required due diligence has been done and a quote is supplied. Please review and approve or let me know if anything else is required.
	link to quote
	link to diligence
	Attachments
	Drop files to attach, or browse.
	Activity
	People
	Assignee: Simone Godfrey
	Assign to me
	Reporter: Lacy Canvell
	Votes: 0 Vote for this issue
	Watchers: 1 Stop watching this
	Dates
	Created: 1 minute ago
	Updated: Just now
	HipChat discussions
	Do you want to discuss this issue? Connect
	Connect Dismiss
Can I make this easier?	Yes! You can use a lot of JIRA Core's functionality, especially comments and mentioning of people (collaboration is alive and well!), to make sure you have all the information you need complete the business case. Assigning the task to the relevant people at the right time as it through the workflow makes sure it'll always appear on their to do lists, and if notifications a they'll even receive an email to let them know it's ready for them.
Other customizations	You may decide to change the workflow. You may need to add in a second approver. You want to consider assigning the task automatically to the correct person as it progresses thro workflow. You may consider adding different issue types for each type of business case tha accept, and assigning each issue type a specific, custom workflow (with the correct assigne status of course!). JIRA Core allows you to achieve this.

Project management

The project management project offers a slightly more robust workflow, allowing you to work on bigger items by offering a workflow that has an In Progress status. This allows you to start work on a task, and keep working in it until it's complete. Track your work in JIRA Core by using our built in reporting, your dashboard, and custom filters based on your searches.

✯ JIRA Cor
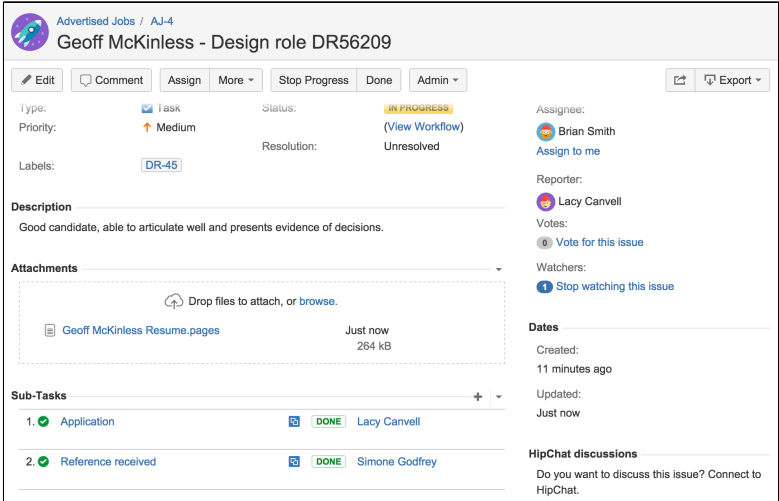
Sign up for a free trial | Try me!
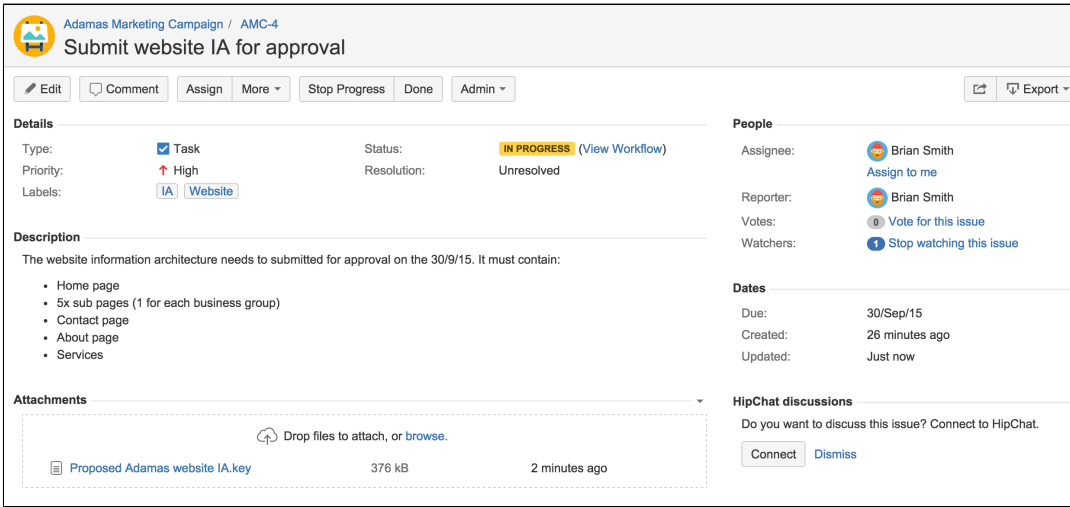
Workflow	To Do > In Progress > Done
	WORKFLOW
	TO DO → IN PROGRESS → DONE
Issue types	Task and sub-task. A sub-task must belong to a parent task.
	ISSUE TYPES
	☑ Task
	⊞ Sub-task
Issue fields	Summary, Issue type, Reporter, Attachment, Due Date, Original Estimate, Remaining Estimate, Description, Assignee, Priority, Resolution and Labels

Resolutions	Done, Won't Do, Duplicate and Cannot Reproduce
Priorities	Highest, High, Medium, Low and Lowest

Feel like project management will suit your needs, but think you may need a little more? Have a look at some of our examples for some tips and tricks on what you can do with the project management project.

Use case	**Application tracking** - You advertise for a role that's become available, and you start receiving applications in the mail, through your website, and via email.
What Business project should I start with?	**Project Management** - This gives you a workflow of To do, In progress and Done, and accompanying issue types of Task and Sub-task.
How can I do this?	1. Create a task for each candidate applying for a role. 2. Add any accompanying CV's, resumès or paperwork as an attachment to the task. 3. Depending on your hiring process, if the candidate is unsuccessful they may go directly to Done, or if you need to perform further tasks, like back ground checks, interviews, or wait for additional paperwork, the candidate may go to In progress. 4. Once you have a successful candidate, move them to Done and add a comment or label to say they've been successful.
How would this look?	
Can I make this easier?	Yes, you can! If you've integrated JIRA Core with your email system, you can make sure that incoming emails regarding job applications are automatically created as tasks. Any accompanying attachments on the email will also be added as attachments to the task. Ever wondered why some job advertisements ask for your name, the position you're applying for and job reference in the email subject line? It's because this is exactly what they're doing, taking those emails and automatically converting them to "something" they can track and process! Why not use JIRA Core to do it for you.
Other customizations	We know that processing a candidate for a role is never as easy as it sounds. Depending on the role, and any requirements you may need to do several interviews, background checks, make sure you've got the correct paperwork signed and returned... the list goes on! JIRA Core allows you to customise the workflow to suit this, and you can assign the task to the relevant person at each stage. Want to use the same project for all your job applications, but worried after a while it'll get too large with too many tasks open? Well you can help sort your issues by assigning them defined labels for each open role. This will help you search the tasks easily and pick out the tasks that relate to the role you're hiring for. You could also add components to the project, which are like a grouping for your tasks. You can use components to automatically assign someone to each task that's placed within that component.

Maybe you're in marketing? Have a look at how you could maybe use JIRA Core to manage your work.

Use case	**Marketing campaign** - you need assets, locations booked, websites created, approval to g live... the list goes on!
What Business project should I start with?	**Project Management** - This gives you a workflow of To do, In progress and Done, and accompanying issue types of Task and Sub-task.
How can I do this?	1. Create a new task for each item of work that needs done. Give it a descriptive title to make sure everyone knows at a glance what it's for. 2. Add sub-tasks for any additional work that may need done, making sure you break big chunks of work into more manageable pieces. 3. When the sub-tasks are done, you can complete the parent task. And your work is done
How would this look?	
Can I make this easier?	Don't forget you can add comments to your tasks, and mention people in the comments by using the "@" mention. This makes sure everyone is aware of the work and tasks that are needed. When using tasks and sub-tasks to track your work, it's also useful to remember yo can view the Activity and Statistics page for your project by clicking on the project name in your sidebar. These views give you a breakdown of the issues in your project so you always know where you stand. If you need more detailed info, you can also create you own dashboard with all the statistics you need, and add a link to the dashboard in your sidebar.
Other customizations	Marketing tasks can be very diverse and very specific in their requirements. You can customize your issues and make various fields 'required' to ensure they're always complete correctly. You can set the fields in various ways (drop-down, multi-select, free text etc.) to make sure whoever creates the tasks can only select from your pre-defined selections, or can complete the information as they see fit. Notifications can also be customized so that th right people are always updated when they need to be. Make sure the tasks and project are set up they way **you** work.

Tips and tricks

JIRA Core is an extremely flexible workflow management system. You can customize and configure JIRA Core to suit the needs of you and your team. If you haven't done so already, you should take a quick look at the JIRA Core overview page to gain an understanding of what JIRA Core can do, and how it does it. I'm sure you have more questions on how to do certain things, how to configure elements of JIRA Core and what exactly is possible. A lot of the configuration and how to information is in the JIRA application administration documentation. The content on this page is some tips and tricks, including links to more information, that should get you up and running and more productive faster!

⌄ How can I add a status to my workflow and extend it?

Add a status to your workflow, like adding a "Review" status for documentation updates, is essentially changing the workflow. To do this you should read the Working with workflows documentation, which will

step you through adding a status, making sure you add transitions so that you can get to your new status, and give you more information on what you can do to that status to make it even more powerful. For instance, you may want to make sure that when you do move an issue to "Review", it's automatically assigned to a specific person. This can be achieved by adding a post function that allows you to update a specific field on an issue. Make the field the Assignee field, and select the specific person you want the issue assigned to. Bingo! The issue will now be assigned to the person when it's transitioned to that status. Take a look at Atlassian Answers, it's full of useful information.

˅ What if I want a parallel workflow, where I can have more than one person work on an issue at a time?

JIRA Core has one Assignee field, and this allows you to assign an issue to one person at a time. This is intentional, as it means one person is responsible for that piece of work. It stops the cases occurring where either two people are working on the same thing, or neither work on it as they think the other person is working on it! That being said, there's a few ways to achieve the concept above. You could split the issue into sub-tasks, and assign each sub-task to a different person. Another slightly more complex option is to add another field to the issue called a "user picker" field, which allows you to select other users. The Assignee of the issue is still responsible for the work, but it does mean other people can be added to the issue and will show as working on it. See the tip on assigning issues to a group below for more information!

˅ So I can add other fields to an issue? How would I do that?

You can add additional fields to JIRA Core's default issues, in fact you can even add additional issue types if you like. All this is covered in the JIRA application administration documentation, just search for Adding a custom field. That'll walk you through which fields you can add to an issue, and how.

˅ Issue fields sound pretty interesting, is there any way to restrict access to them?

Issue field security is something that a lot of organizations request, but it's not something we support in JIRA

Core. You can read more about that decision on this issue **JRA-1330** - Field level security permissions CLOSED .

You can set the security level on an issue however, visit the JIRA application administration and search for Setting security on an issue for more information on that topic.

˅ How can I copy an issue?

If you'd like to copy an issue, it's possible by using the Clone functionality. A cloned issue will create a link to the issue it's cloned from, and vice versa. You can read up on more details about issues and cloning them here.

˅ How can I review all my information in one place?

Dashboards! JIRA Core comes with a variety of gadgets that you can configure to show your information how you need to see it. You can add these gadgets to your dashboard, and make that dashboard your homepage in JIRA Core when you first log in. That way the first thing you'll see is the information that matters to you, and you'll know what you need to do.

˅ How do I find my work? And how do I find issues that I need to?

JIRA Core offers a powerful search function, and you can even save these searches as filters for use at a later stage. You can even share these filters with your team, so you're all looking at the same information. Keep on top of things together!

˅ So I use search and I still struggle to find issues I want to check. Is there any easy way to tag issues?

Yes there is! JIRA Core comes with the ability to add a label to an issue, and labels can be used in searches. In fact, the label is 'clickable' which means that when you click the label, you'll be taken to a search that shows all issues with that label. You can then refine that search if needed.

A word of caution though, check with your project administrator as to your organization's policy on using labels. There may be a policy in place regarding how this field should be used, as some organizations use it as part of a process.

˅ Can I assign an issue to a group, so that someone in that group will work on it?

JIRA Core is designed so that issues must be assigned to a single individual to prevent tasks from being overlooked. A team lead or manager should assign issues out to individuals, or your users will pick from a list of issues that they have the option to take on.

However, if you want to configure JIRA Core to allow issues to be assigned to multiple users there are a few option for doing so:

- Managing Issues via a Queue

- Managing Issues via Group Ownership
- Managing Issues via a User Account
- Managing Issue via Sub-Tasks

It is easy to still setup a queue the a group can pick from, or affiliate an issue with group in addition to having it assigned to an individual within that group:

Managing Issues via a Queue

You can configure your JIRA Core project to assign issues to an 'Unassigned' "queue" by default, which your users can then pick issues from.

To do this, set up the following:

1. Configure your JIRA Core project to allow the 'default assignee' to be 'Unassigned'.
2. Ensure that 'Allow unassigned issues' is set to ON in your General Configuration settings (**Administra tion > Global Settings > General Configuration**).
3. Set any issues that you want to be in the queue to be 'Unassigned'.
4. Create a dashboard page with a filter that lists all 'Unassigned' issues, share the dashboard page and request that interested members of the group display the shared page on their dashboards.

Managing Issues via Group Ownership

You can add a custom field to store which users and groups should be associated with a given issue. This is particularly useful for projects where a team owns all issues of a particular type.

To do this, set up the following:

1. Add a group picker custom field to your issues.
2. Configure an email notification in your project's notification scheme to be sent to the 'Group Custom Field Value'.

An issue can now be "assigned" to the group by selecting the appropriate group in the group picker. An email notification will be sent to the group.

ⓘ *Another option is to add a user picker custom field rather than a group picker, and assign multiple users to an issue. However, you will then have both the JIRA Core default user field and custom user field for your assignees.*

Managing Issues via a User Account

You can create a JIRA Core user account to represent a group of people (e.g. 'developers') and assign issues to this user.

To do this, set up the following:

1. Create a JIRA Core user to represent the group.
2. *(Optional)* Create an email mailing list for this group (not a JIRA Core function) and set the mailing list email as the JIRA Core user's email address.
3. Create a dashboard page showing issues assigned to this user, share the dashboard page and request that interested members of the group display the shared page on their dashboards.

An issue can now be assigned the new "user" representing the group and your users can track the issues on their dashboards. If you have set up a mailing list, your users will also be notified by email.

Managing Issue via Sub-Tasks

If you have a task managed by different users then you are able to break the combined task into individual subtasks with their own single assignees.

⌄ It sounds like issues could benefit from having some defined groups to make them easier to manage. Is this

possible?

Yes it is. JIRA Core comes with two methods of grouping issues, Versions and Components. You can read more on these in the Administering a projectsection of the documentation. Essentially versions and components allow you to group issues, and the main distinction is that with a component you can assign a user as a default assignee when the component is added to the issue, and versions can have a start and end date.

How do I build the workflow I want?

So you've got to the stage where you need a little more from your workflows than just "To Do", "In Progress" and "Done"? Great! JIRA Core lets you customize and configure the workflow of your project so that you can move work through it the way you need it to. You need be a JIRA administrator to change a workflow, and we have all the information you need on changing a workflow in the JIRA administrator documentation, but what you probably want to know is what is possible with my workflow? What exactly can I do? Am I confined to just adding steps? We'll take you through some steps to creating a great workflow, and give you some information on what's possible.

On this page:
- What's in a workflow?
- How do I create the workflow I need?
- Make every transition count
- Building your workflow
- Tips and tricks

What's in a workflow?

First, let's take a look at what defines a workflow. A workflow has four unique components: statuses, transitions, assignees, and resolutions. As an example, libraries have well-known workflows surrounding materials they lend out to the public. Each item could be stored in JIRA Core as an issue, and administered in the following simple workflow:
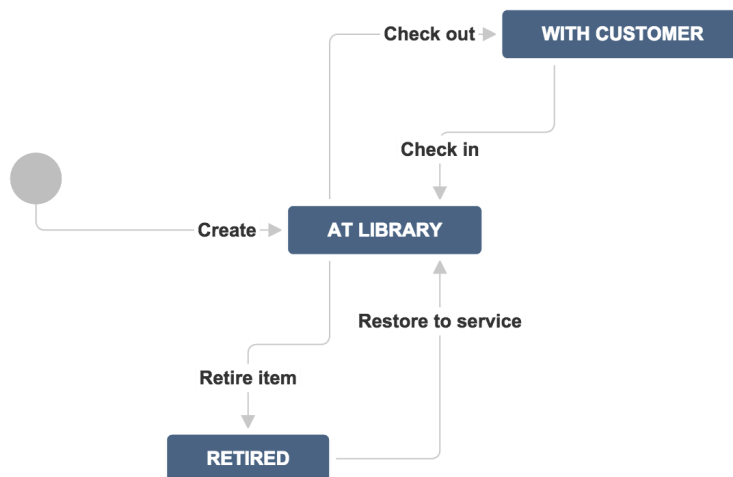
Statuses: where

Statuses represent the position of the issue within a workflow. In the library example, a book can be in one of three places: at the library, with a customer, or retired from inventory. Every status should be a unique state in your workflow.

Transitions: how

Transitions are the bridges between statuses; the way a particular issue moves from status to status. At the library, books are loaned out by librarians, returned by customers, and evaluated by library staff to see if they're fit for circulation. Note that customers cannot evaluate if an item is fit for inventory – that must be decided by the library staff. With JIRA Core, you're able to set workflow permissions that allow only the right people to transition an issue.

Assignees: who

Workflows guide how people work together. In JIRA Core the assignee dictates the responsible party for any given issue. In the library example, we would set the assignee to the borrower every time a book is loaned out, so that the library knows which customer has that book. When the book is returned from inventory, we would then remove the assignee when we transition the book back to inventory.

Resolutions: why

Resolutions explain why an issue transitions from an open status to a closed status. The act of retiring a book removes it from inventory, therefore, we can put a resolution on that book. In this example, we might use resolutions such as *damaged*, *out of date* or *lost*. The two biggest mistakes new JIRA Core administrators make are:

- **Confusing resolution with status** – Status describes where an item is in the workflow whereas resolution explains why an item is no longer in flight. In order to effectively use these features to retire a book from circulation, our librarian should create a **status** called *retired*, and options for **resolution** t hat include things like *lost*, *damaged*, or *out of date*. Searching for everything that's retired (or resolved) – with the ability to search for *why* it's retired via the resolution options gives much better reporting metrics. That's why JIRA Software uses **resolution** to know if the issue is considered inactive by the organization. The best way to check to see if you've got it right is to use the created versus resolved gadget. You can have a workflow between when an issue is tagged with a resolution and officially closed. Many organizations verify resolved issues to ensure they're resolved for the right reasons.

- **Setting resolution correctly** – You must set a resolution when an issue moves from an open state to a closed state. Likewise, resolution needs to be removed when an issue gets reopened. At the library, if the librarians decide to put the book back into circulation, they would need to remove the resolution from that book. For example, a book that is able to be checked out should not have a resolution of *lost*.

How do I create the workflow I need?

Building the workflow you need can be challenging. Remember, it's all about building what you and your team need to work more efficiently and effectively. That may involve you and one other person, or it could involve several departments and many people. Here's a few pointers to get you going.

Gather all of your stakeholders

Workflow is about scaling culture, and culture is about people. Whenever it comes time to build a process around a set of people, identify all of the stakeholders in that workflow. For example, if you're trying to build a workflow between product management, software development, and support, ensure you have one representative from each organization in the meeting. As the person designing the workflow, spend time talking with each stakeholder about what's important to them. Before the meeting starts, draw a draft workflow on the whiteboard, then walk through each case in the meeting and gather stakeholder feedback. Workflows are touchy as they govern how people work, so be patient, and your investment will pay off later.

Use a whiteboard in low fidelity so it's easy to get feedback and make changes at this stage.

Keep the workflow simple: less is more

Most of the time, stakeholders want to have statuses for each part of the workflow. Generally, that's a good thing, but remember: each status adds more transitions and complexity to the workflow. Workflow is designed to make things simple and scalable. Whenever you're adding a new status to a workflow, ensure that the stakeholder has no other option but to grow the workflow. Let's look at two examples.

- For many news papers, articles needs to be reviewed and this is an important part of the editorial process. Jane, the editorial manager, wants to add a specific status called *article review* so that it's clear to the team what issues are still being written, and which issues are waiting review. Reviewing articles is distinctly different from writing articles. It makes sense to add a new state as the review state will indicate an article has been written, and a different person (assignee) is now responsible for review.
- Bill, the editor, wants to add a new status called *rejected* for all issues that don't pass review by his team. I'd advise against doing this, as the writing team can simply reopen any issue that fails review. An alternative option is to move the issue into a *resolved* state with the resolution *rejected*.

As a JIRA Core administrator you owe it to yourself and your users to keep workflow simple.

Make every transition count

JIRA Core has a number of options administrators can employ when transitioning issues between states.

- **Conditions** – Conditions control who can perform a transition. For example, only someone in the library can retire a book from inventory.
- **Validators** – Validators ensure that a transition can happen given the state of the issue. For example, if a book is to be retired because it's allegedly damaged by a customer, we'd like to ensure the book has been checked out at least once to verify that claim.
- **Post functions** – Post functions execute actions on issues after both conditions and validators pass. The most common post function is to clear the resolution when reopening an issue. In the library example, we would use a post function to clear the reason a book was retired when it got put back into service. JIRA Core keeps a detailed history of issues, so we could look up when the issue was retired and why.
- **Assignees** – Whenever a transition occurs, always ask who the new owner of the issue is. Ensure that in every transition there's a default action to route the issue to the right place. Users can override the default action as well.
- **Properties** – JIRA Core recognizes some properties on transitions. The most common one is to limit resolutions displayed to the user on a given transition. For example, we might want the resolution *scrat ched* to show up for disk media when retiring, but not for books.

Building your workflow

Check out the JIRA administrator documentation on workflows for all the information on how to build the workflow you want. Once you've built your workflow in JIRA Core, you should take the time to ensure the workflow is robust and doesn't have any dead ends, or unexpected behaviours. You'll want to share the workflow with all of your stakeholders for a final round of feedback. You should also enable the option in JIRA Core to easily show where the current issue is in the workflow. All users have to do is click *view workflow*.

Tips and tricks

There's a few things you should check, and a few tricks you can use you get the most out of your workflows:

- If you intend to change the workflow you're using for your project, make sure that the workflow isn't being used by any other projects! Any changes made to a shared workflow will impact all the projects using it.
- If you already have a workflow that works for you, and you want to use it in another project, you can either create the new project and select "Create with shared configuration", or once you've created the project you can change the default workflow to the existing workflow that you want.
- Once you edit your workflow, remember to publish it! Publishing the workflow effectively makes it active on your project.

Installing JIRA Core

JIRA Core can be customized to suit your needs, and can also be extended and linked with other applications to provide you with the perfect solution to track all your work in one place.You can install JIRA Core Server anytime by downloading a trial version, selecting the server installation type that suits your needs, and choosing the number of users that you want the installation for.

Once you have downloaded the installer, follow our Installation guide to get up and running.

Download

If you experience any trouble with your installation or you have any question, please contact Support .
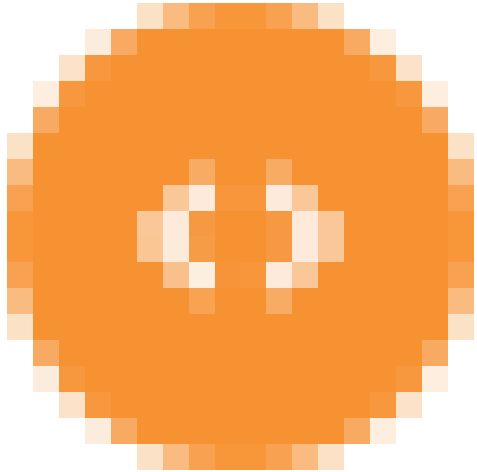
JIRA applications overview

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.
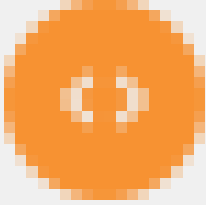
Application	Project type	Application specific feature set
JIRA Core	 Business projects	• Available to all licensed users

JIRA Software		
	Software projects	• Integration with development tools • Agile boards • Release hub for software versions
JIRA Service Desk		
	Service Desk projects	• 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	✗	✗	✓

Using JIRA applications with Confluence

What is Confluence?

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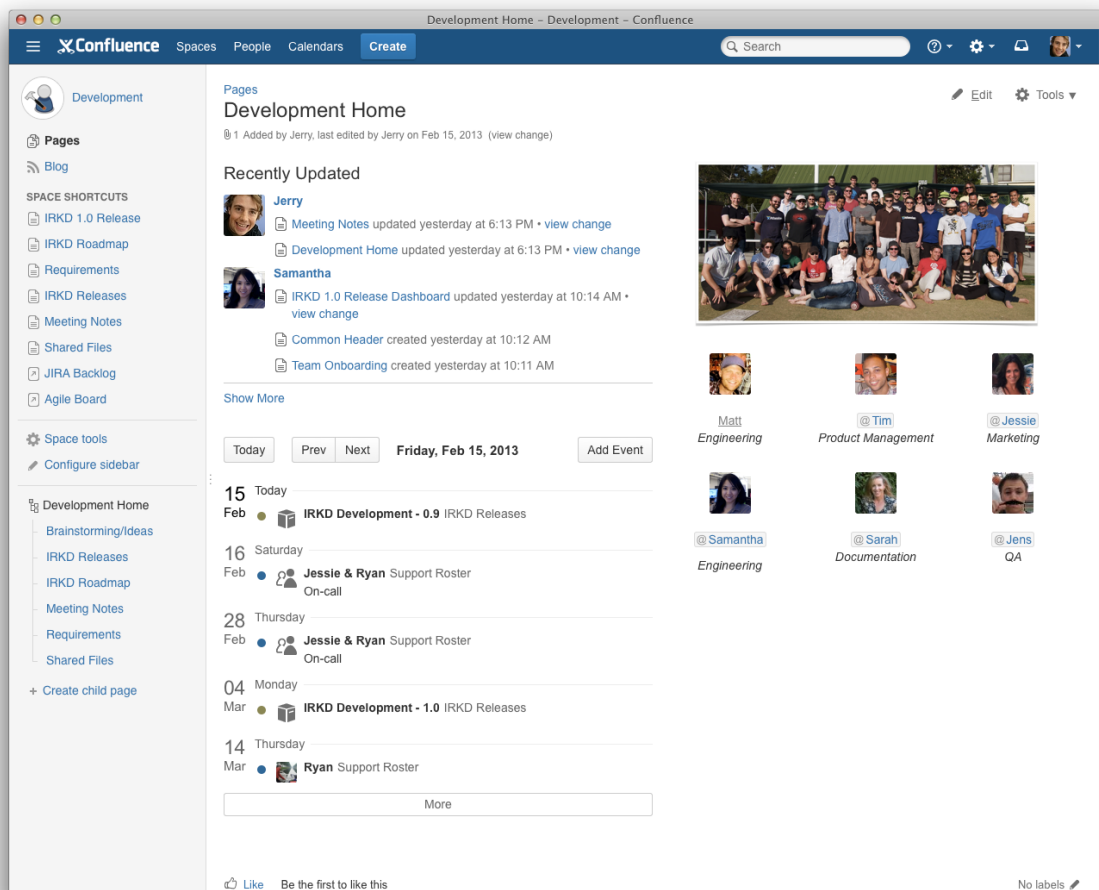
Why use Confluence with JIRA?

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

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

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

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

Confluence features for JIRA users

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

Define product requirements

Many of our customers write product requirement documents using Confluence to plan new produc The Product Requirements Blueprint helps devel teams collaboratively create, discuss, and organi product requirements. It's easy to link your produ requirements in Confluence to issues in JIRA.

For more information, see Blueprints.
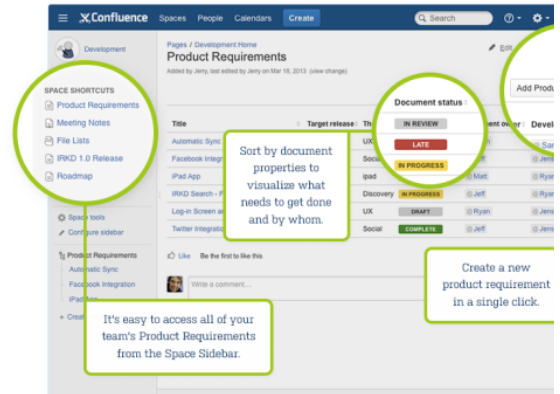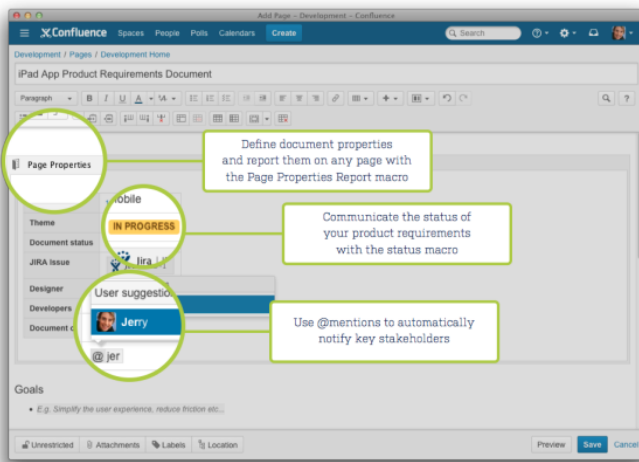
Team Calendars: Your Birds-Eye View of JIRA

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

- **Timeline Calendar:** View plans 3 months ahead of time.
- **JQL Support:** Track your versions, issues, and agile sprints.
- **Date Ranges:** Visualize issues over time to understand upcoming workload.
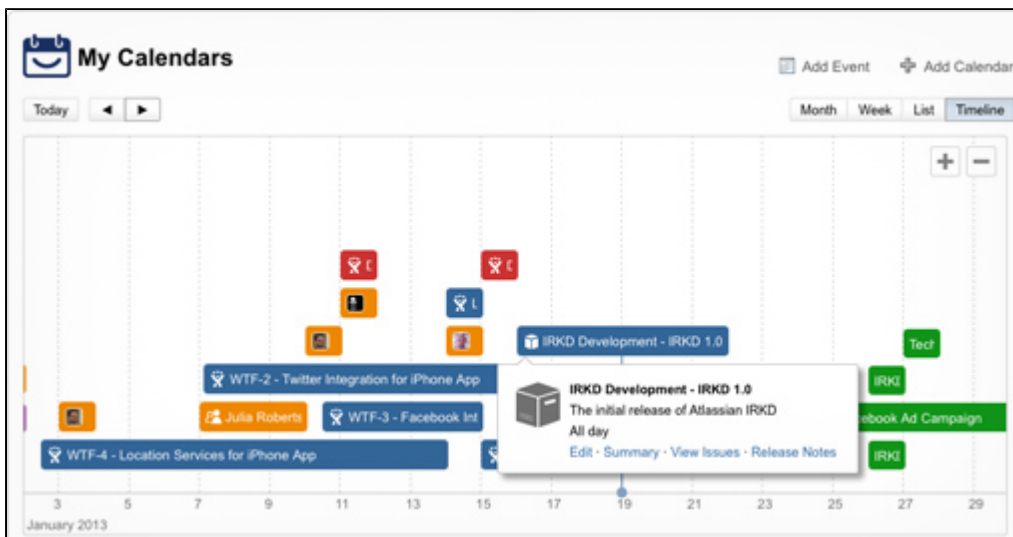
To install this feature, please visit Atlassian Marketplace.

Report on issues

Insert issues on any Confluence page using the JIRA Issues macro

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

- Display a table of issues on your page, based on the results of a search using JIRA Query Language

(JQL) syntax or a JIRA URL.
- Display a single issue from the JIRA site, or a subset of selected issues from your JIRA search results.
- Display a count of issues from the JIRA site.
- Create a new issue on the JIRA site and display that issue on your page.

Autoconvert pasted issue links

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

Automatic links

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

Gadgets

You can embed a Confluence activity stream or a Confluence page in JIRA's dashboard. Likewise, JIRA gadgets can be rendered on a Confluence page. See these topics for information on how to set up these gadgets:

- Add a Confluence Gadget to a JIRA Dashboard
- Add JIRA Gadgets to a Confluence Page

Using JIRA applications with HipChat

Integrating JIRA applications and HipChat gives you and your team the following collaboration power:

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

Connecting projects to rooms

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

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

   ⚙

   > Projects.
3. Select a project.
4. In the Project 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 in connect JIRA to a private room in HipChat you will need to authoriize HipChat from the **HipChat Integration** setup screen.
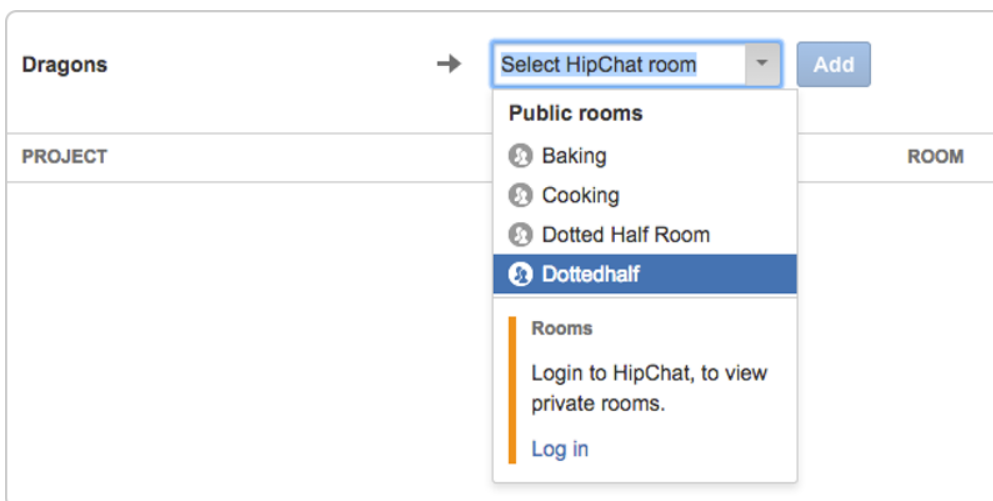
Once you have authorized JIRA, all of the private rooms that you are a member of will be displayed in the

room selector drop-down menu. When your JIRA project and room are integrated, everyone in the private room will be able to see the notifications that are sent to that room.

Invite users

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

You can remove this access by following the instructions in Removing OAuth Permissions.

Discuss issues in rooms

You can focus your discussion by creating or selecting a HipChat room to discuss an issue. When JIRA is integrated with HipChat and you are in the issue screen, you can select to "Create a room" or "Choose a room" in the **HipChat discussions** panel. This will associate the current issue and the room and any changes to the issue will send a notification to that room.
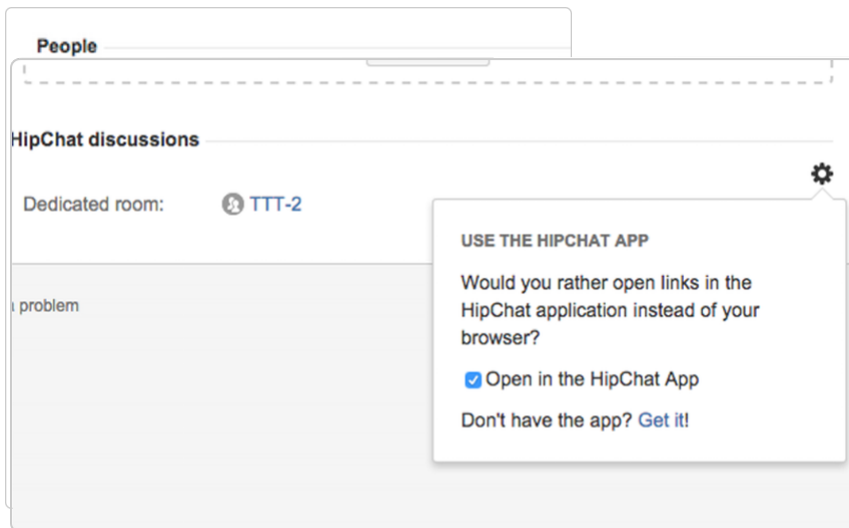
You can also select to have links open in your HipChat App (OSX only) when you select a link. In the issues screen, select the cog icon in the HipChat discussion to enable opening links in the application.

Issue preview

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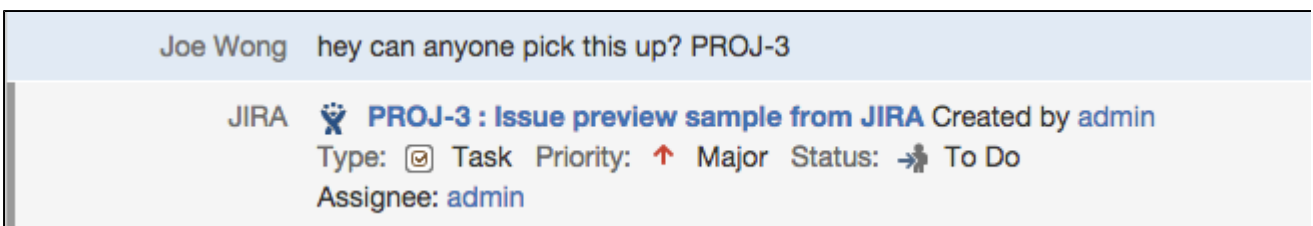
Connectivity requirements for JIRA and/or HipChat Server customers

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

A note on JIRA permissions

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

Configuring issue previews

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

As a JIRA Administrator

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

As a Project Administrator

1. You must be a logged in as a **Project Administrator**.
2. Choose
    ⚙
    > Projects.
3. Select a project.
4. In the Project 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**.

Using JIRA applications with Portfolio for JIRA

Portfolio for JIRA provides a single, accurate view for planing and managing initiatives across multiple teams and projects with ease. See our guide to how JIRA and Portfolio for JIRA work together.

*   Plan top-level business initiatives and break them down into lower level deliverables for development teams
*   Track investments across strategic themes to ensure that those investments align with business priorities
*   Generate realistic delivery forecasts with automatic scheduling algorithms
*   View the progress of any initiative based on real-time, accurate data from JIRA
*   Drive accurate and realistic capacity planning by defining teams and allocating work based on skills and availability
*   Scope releases in a matter of clicks
*   Make fast prioritization and tradeoff decisions to instantly see the impact of plan changes
*   Minimize productivity loss by easily reacting to the ever-changing needs of your business in real time

#	Title	Release	Team	Estimate	Design	Implementation	Initiative	Theme
1	Shiny web frontend	1.0, 1.5	The Avengers, Th…	140	31	109	Project "Stratos"	
	I want to sign up and login	1.0	The Avengers	25	3	22		
	I want to share dashboards with colleagues	1.0	The Avengers	20	2	18		
	I want to see charts in a dashboard	1.0	The Avengers	22	7	15		
	I want to create my own dashboards	1.0	The Avengers	8	1	7		
	I want it to work on a tablet	1.0	The Beatles	25	8	17		
	Show some whistles	1.5	The Avengers	20	5	15		
	Create some bells	1.5	The Avengers	20	5	15		
2	A usable API	1.0	The Avengers	35	7	28	Project "Stratos"	

Shiny web frontend — Something with cloud
A usable API
Blazing-fast server
Mobile app
Monitoring
Something with big data
Security and access control

05/Aug/14 - Start 29/Sep/14 - Phoenix (1.0) 22/Dec/14 - Project X (1.5) 16/Mar/15 - Later

Plan automatically

- Set priorities, estimates, and target dates to instantly see when you can ship releases based on your commitments
- Automatically optimize your plan and get suggestions on ideal resource allocation to create a realistic forecast
- Account for dependencies, resources, parallel vs. sequential activities and the realistic number of people that can work on a single item to create an optimized roadmap
- Use themes to categorize your backlog items by strategic focus areas, value streams, or investment categories to see relative resource allocation between themes

#	Title	Release	Team	Members	Estimate	Design	UI/UX Design
8	Something with cloud	Later	The Avengers, Th…	Black Widow, George, John	20	20	15
	Go Mobile!	1.0, Later	The Beatles, The …	Bodie, Cowley, Doyle, George	85	30	15
6	Mobile app	1.0, Later	The Beatles, The …	Bodie, Cowley, Doyle, George	85	30	15
	Explore the basics		The Beatles		50	25	10
		Later	The Professionals	Bodie, Cowley, Doyle	22.5	2.5	2.5
	Enable Social sharing of smart things	1.0	The Beatles	George	12.5	2.5	2.5
	Project "Stratos"	1.0, Later	The Avengers, Th…	Black Widow, Cowley, George…	142	37	12.5
5	Security and access control	1.0	The Avengers, Th…	Hawkeye, Hulk, John	10	2.5	2.5
4	A usable API	1.0, Later	The Avengers, Th…	Black Widow, Hawkeye, John	22	7	
	REST API - Allow to submit data	1.0	The Avengers	Black Widow, Hawkeye	7	2	

1 missing skill in team 'The Beatles'. Technical Design

Avoid bottlenecks

- Identify and avoid bottlenecks and potential holdups by accounting for dependencies across teams and projects
- Model skills and define who can do which type of work to avoid unrealistic resource loads
- Automatically account for team member availability, including time off, training and inter-team commitments
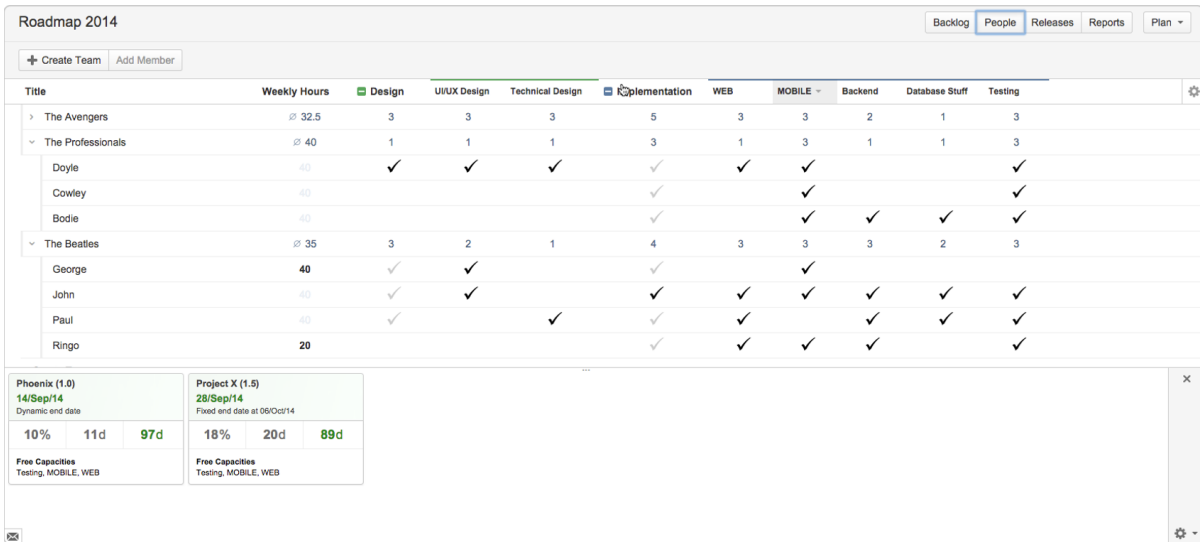
Keep up to date

- Keep the long term plan in sync with reality by adjusting delivery dates, team member resources and dependencies using up to date data
- Changes are displayed in real-time across all projects, giving you a comprehensive view of your entire roadmap
- Having an up-to-date schedule gives your team a transparent understanding of what's next, lending clarity to your decision-making
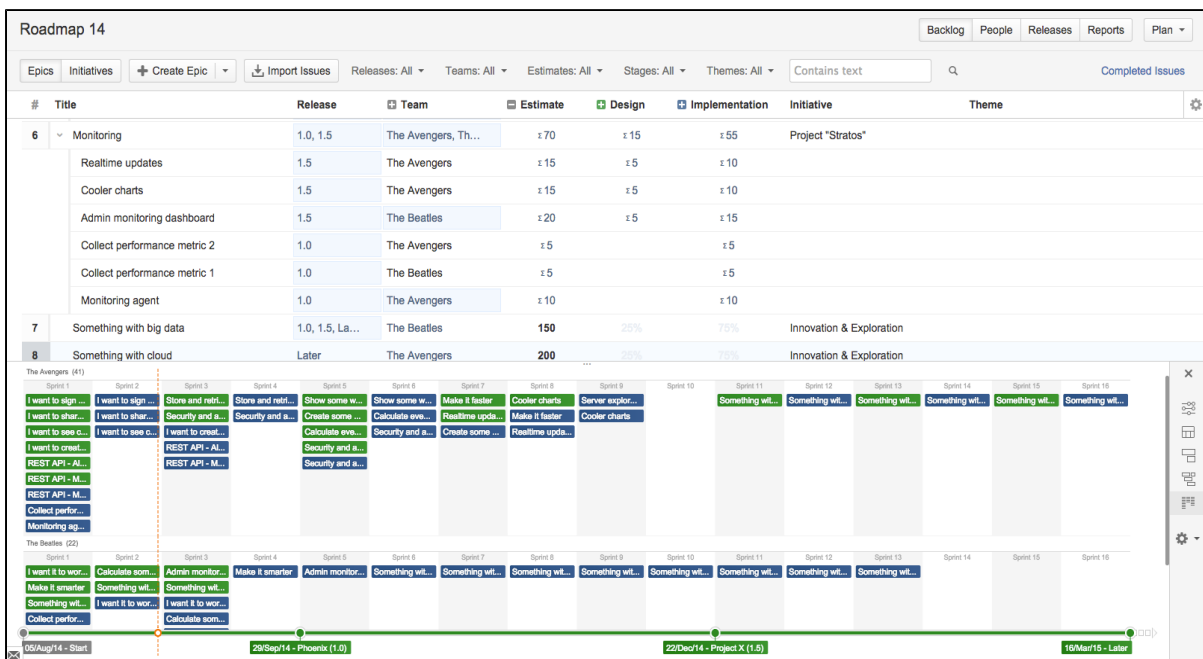
Make realistic commitments

- Confidently make commitments for scope and ship dates by using your more reliable forecasts
- Adjust dependencies to quickly check the impact across all of your projects, for example, if a critical feature is taking longer than expected
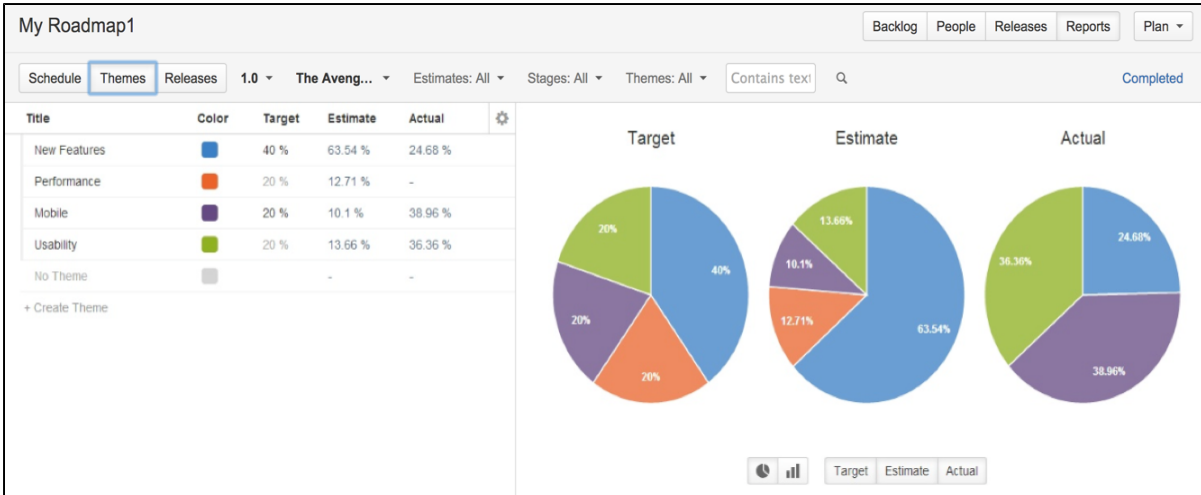
Model changes

- Quickly visualize what different scenarios and decisions mean to your projects
- Evaluate new change requests by seeing the impact on scope, dates, resources and cost commitments
- Model different scenarios without affecting the underlying data in your JIRA projects

You can find more information about managing your portfolio here: Portfolio for JIRA.

Getting started with JIRA Core

Welcome to the JIRA Core Getting started tutorials! We know you want to get up and running with JIRA Core as quickly as possible, so we've designed our Getting started tutorials to take you through the basic concepts and tasks you'll need to know in JIRA Core.

This is a hands-on tutorial, and we'll be taking you through the basics of JIRA Core, and its three main user roles.

- If you're evaluating JIRA Core, we suggest you do all three tutorials.
- If you've been invited to JIRA Core, we suggest you do the tutorial that best suits your user role.

So what is JIRA Core?

JIRA Core is a workflow management system which allows you to set up unique processes that suit the way you work. At the heart of all systems are workflows, moving packets of work from A to B. JIRA Core allows you to make your workflow as easy or as complex as you need, giving you the freedom to concentrate on the work, not the process. JIRA Core can be used in a variety of ways. For example, you could use it to run a

t-shirt business, setting up workflows that can manage your internal processes such as your design process, your sales process, your manufacturing process, and even controlling your stock. The beauty of JIRA Core is that the only constraints on your workflow are your processes!

What types of JIRA Core users are there?

JIRA Core users can be grouped into three types:

Administrators	Project administrators	Users
Administrators are responsible for the configuration of JIRA Core. They control the initial set-up, the look and feel, and the configuration of project workflows, issues, and generally what the end user will see and be able to interact with.	Project administrators are responsible for configuring their projects. They can administer projects, change the look and feel, and make various configuration changes to the project. A project administrator can't create a project unless they're explicitly given this permission by an administrator.	Users are responsible for working in specific JIRA Core projects. Users are given access to a project's issues, and, depending on their permissions, work on the issue by commenting on it, transitioning it through its workflow, and closing it when complete.

Now that you have a basic understanding of JIRA Core and its users, select a tutorial and let's get started:

I'm an administrator »   I'm a project administrator »   I'm a user »

Getting started as an administrator

For the purpose of the tutorials, we'll assume you're going to use JIRA Core Cloud, as it's the quickest to set up. If you want to install a JIRA Core Server instance, please refer to the installation guides on the first step of the JIRA administrator guide.

JIRA Core administrators should complete this tutorial to understand the basic concepts of managing a JIRA Core Cloud site.

Here's what you can count on learning:

1. How to set up a JIRA Core Cloud site
2. How to add users to your site
3. How to create a project and customize it
4. How to manage permissions

By the end of this tutorial, you'll have a fully functioning JIRA Core Cloud site, several users with different access permissions, and you'll have created your first project.

Audience Administrators **Time** 45 minutes (including JIRA Core Cloud provisioning)

Ready to dive in and get your hands dirty? Get started and learn how to set up your site.

Let's get started »

Setting up your site

1. Setting up your site
2. Creating a project
3. Adding new users
4. Managing permissions

You'll need your own JIRA Core Cloud site for this tutorial. Let's get you set up with a JIRA Core Cloud site. JIRA Core Cloud is our hosted JIRA Core offering, and you'll be up and running with your own JIRA Core site in a few minutes without installing a thing! If you have a JIRA Core Cloud site already, you can skip this step and go straight to step 2 by clicking the Next button below.

Sign up for JIRA Core Cloud

Signing up for JIRA Core Cloud will provide you with a fully-functional JIRA Core Cloud site. You'll be able to use this site to evaluate JIRA Core for your team's needs.

1. Open this link in a new tab to view the Atlassian Cloud site signup page.
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 about 10 minutes for your JIRA Core Cloud site to be created.

When your site is ready, access it via your browser using the site address that you entered when signing up (e.g. https://mytrial.atlassian.net). Log in to your new site using the administrator credentials that you entered when signing up (e.g. jsmith).

Set up your new site

In this step, you will be changing the site logo and the color scheme for your JIRA Core Cloud site. You can change the color scheme, logo, date format, and more, to enhance the branding of your organization or make it consistent with other systems.

1. Select

   ⚙▾

   > System.
2. Scroll down to the User Interface section and select **Look and feel**.
3. Choose a logo image file or URL to upload. Note that this logo will appear in the top left corner of your site.
4. Select **Upload Logo**. Your logo will be uploaded and the the color scheme for your site will be automatically updated to match the logo color scheme. You can manually change your color scheme in the **Colors** section of this page.
5. Scroll down to the **Date/Time Formats** section to select your preferred format. We'll keep the other look and feel settings the same for now.

> **Can't use JIRA Core Cloud?**
>
> Instructions for installing a JIRA Core Server instance on your own server are available below.
>
> - Installing JIRA on Windows
> - Installing JIRA on Linux
> - Installing JIRA on Mac OS X *(note: Mac OS X isn't a supported platform, and is recommended for evaluations only)*

Success! You now have a new JIRA Core Cloud site set up. Now, let's create a project for your team.

Next

Creating a project

1. Setting up your site
2. Creating a project
3. Adding new users
4. Managing permissions

A JIRA Core project is a container that holds issues. Issues can be viewed as the packets of work required within a project. To create issues, you must have an available project to contain them. JIRA Core comes with several default project types with preconfigured workflows and issue types, so you can quickly get your project up and running. In this step of the tutorial, you will use the project management template to help your team plan, organize, and collaborate on their work.

Note that creating and configuring a project is done by an administrator. A project administrator controls user access to the project, and can only configure certain aspects of the look and feel of the project. You should still be logged in to JIRA Core as an administrator from the previous step. If not, log into your administrator account.

Create a project

When creating a project, you will need to give it a name, a key, and a project lead. The title can be as descriptive as you want, and the key should be something meaningful. The project lead is usually the project manager, but can effectively be any user you select when creating the project.

1. Select **Projects** > **Create Project** and choose **Project Management**.
2. Enter **Dragon Design Tees** as the project name. Note that JIRA Core creates a Project key for you, but you can overwrite this if you want to. By default, you should already be listed as the Project Lead.
3. If prompted to create a link to another application, such as a Confluence space, leave that option unchecked for now.
4. Select **Submit** to create your new project.

> **About project keys**
>
> Each project has a unique *name* (e.g.**Dragon Fire Tee**) and a unique *key* (e.g.**DFT**). The project key becomes the first part of that project's *issue keys,* e.g. **DFT-1, DFT-2,** etc.

Customize your project

In this step, you will be customizing the your project avatar and project details to help your team identify the project more easily. These customizations are helpful if you have several projects in your JIRA Core Cloud site. If you have navigated away from your project, simply go to **Projects** > **Dragon Design Tees**.

1. In the bottom left corner, select

   ⚙

   to open the project administration menu.
2. Select **Edit Project** in the upper right corner.
3. Click the **Project Avatar** image.
4. Select an available icon or upload an image.
5. Enter a URL and Description for your project to make it easier for your team to identify. Note that these fields are optional and only for display.
6. Click **Update** to save your changes.

Congratulations! You've now created and customized your first project. Next, we'll add users to your project and look at how you can set up and restrict access to projects.

Next

Adding new users
1. Setting up your site
2. Creating a project
3. Adding new users
4. Managing permissions

Working alone isn't much fun, so let's add some test users to your JIRA Core Cloud site. You can add users directly, or allow new users to sign up themselves. In this step in the tutorial, you'll add three users directly to your site.

Add a few users

You will be adding three users: **Jason, Kate** and **Emma**. You can add more or choose your own usernames
if you like, but please note that we will be referring to these usernames later in the tutorials. You can always
disable or delete any users you set up.

If you've logged out of your new Cloud site, log in with the administrator account you created.

1. Navigate to the **User Management** screen by selecting

   ⚙ ▾

   > User Management.
2. Choose **Create User** to add a new user. Specify the username as **jason**. Set the rest of the fields to
   whatever you want. You're going to be creating a couple more users, so check the **Create another** ch
   eck box before selecting **Create user**.
3. Now create two more users, with the usernames **emma** and **kate**, following the same process outlined
   above.

You should have a screen that looks something like this:

Full name	**Username**	**Email address**	**Last session**	**Action**
Emma	emma	emma@example.com	Invitation sent 10 days ag…	Resend
Administrator	ixadmin	admin@example.com	18 Sep 2015 5:23PM	Deactivate
Jason	jason	jason@example.com	Invitation sent 10 days ag…	Resend
Kate	kate	kate@example.com	Invitation sent 10 days ag…	Resend
System Administrator SYSADMIN	sysadmin	noreply@atlassian.com	18 Sep 2015 5:00PM	Deactivate

> Note that usernames are **not** case sensitive. Emma can enter her username as Emma, emma, or
> even EmMA to log into JIRA Core. Passwords, on the other hand, are case sensitive.

**Well done! You've added three new users to your Cloud site. Next, you'll learn how to manage access
to your project with site and project permissions.**

Next

Managing permissions

1. Setting up your site
2. Creating a project
3. Adding new users
4. Managing permissions

You won't want every user in your team to have the same level of access to JIRA Core. For example, you
may want to restrict who can administer JIRA Core, or prevent users from viewing a project. In this step, you
will learn about the different permissions in JIRA Core and set permissions for a new project.

Overview of roles, groups, and users

A role is a project-specific set of groups and/or individual users. In our example of the design project in the
t-shirt business, all product managers need to be able to assign work (issues) across all projects, while
senior designers need to be able to assign work on specific design projects. In JIRA Core, you can define a

product manager role that includes all product managers. You can then define a set of permissions with the 'Assign issue' permission for this role, and apply this set of permissions to all projects. Individual senior designers can be added to the product manager role on each project, as needed.

Overview of global and project permissions

Global permissions cover a small set of functions that affect all projects in JIRA Core (for example, permission to administer JIRA Core). They can only be assigned to groups:
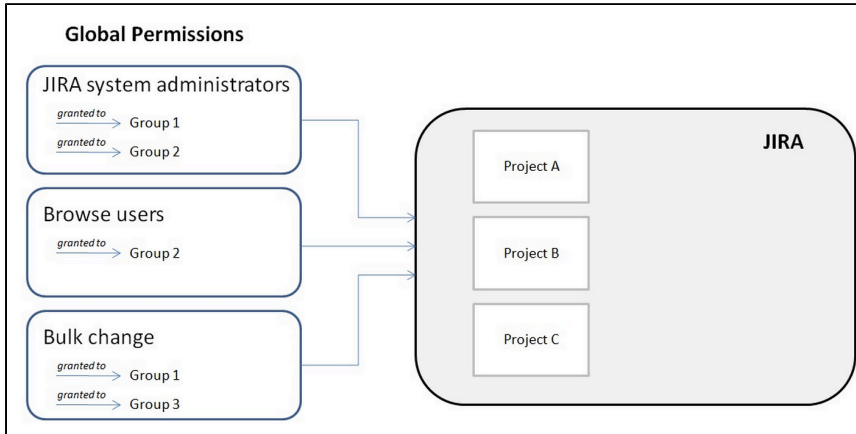
Project permissions cover a set of more granular functions that affect a single project in JIRA Core. For example, permission to create issues in a project. They can be assigned to groups, users and project-specific roles:
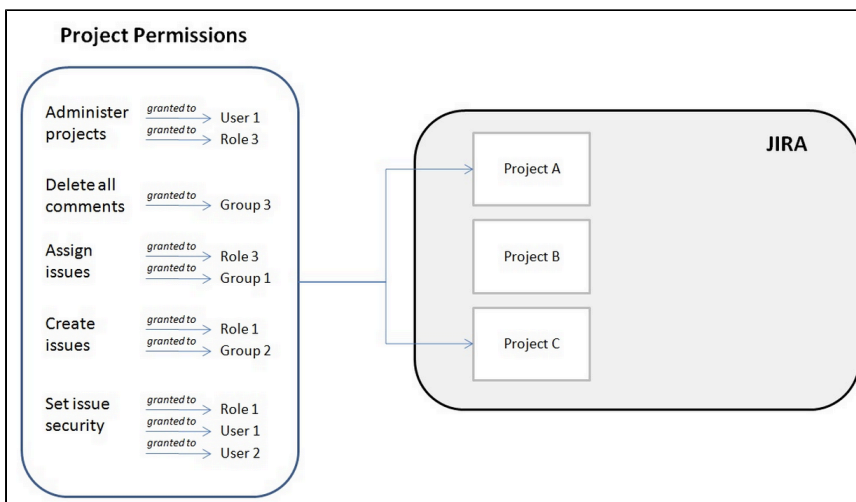
Now let's put this into practice! You're going to go through the tasks involved to use project permissions to hide a new, secret t-shirt design project from some of your users.

Create a new project role

This project role will only contain users that you want to view a particular project. We will assign permissions to this role in the next step.

1. Select

   > **JIRA administration** > **System**.
2. In the security section, select **Project roles**.
3. Below the existing project roles, add another project role named "Review". Leave the Description field blank for now and select **Add project role**.
4. Select **Manage Default Members** and then, under Default Users, select **Edit** to add yourself and **Jason** to the Review project role. Do not add Kate or Emma.

Configure a new permission scheme

The 'Browse Projects' permission controls whether a user can browse a project, i.e. whether they can view the project. Let's assign this permission to your new project role.

1. Select

   ⚙ ▾

   > **Issues** > **Permission Schemes**.
2. Copy the **Default Permission Scheme**.
3. Edit the copied permission scheme and change the name to **Confidential Permission Scheme**. Select **Update**.
4. Select **Permissions** for the **Confidential Permission Scheme**. For the **Browse Projects** permission:
   * Choose **Delete** for 'Application Role (Any logged in user)'.
   * Choose **Add**, select **Review** in the **Project Role** drop-down, and select **Add**.

Associate the scheme with a new project

For the last step, let's associate the permission scheme with your new project.

1. Select **Project** > **Create Project** and choose **Task Management**.
2. Name the project **Top Secret Tee** and **Submit**.
3. Navigate to Project administration by selecting

   ⚙

   > **Permissions** from the menu on the left.
4. On the Project permissions screen, select **Actions** > **Use a different scheme**.
5. Set the Scheme to **Confidential Permission Scheme** and select **Associate**.

The only users that will be able to browse your new project are Jason and yourself. Note that default members are only added to a role for new projects. You can also use this approach to restrict users from creating issues, adding comments, closing issues, etc, in a project.

Well done! You created a project permissions scheme and applied it to a project.

You've now completed the Administrator Getting started tutorial. We suggest you complete the Project Administrator tutorial as well, so you have a better understanding of how your team will be using JIRA Core. So put on your project administrator hat and let's get started!

I'm a project administrator »

Getting started as a project administrator

JIRA Core project administrators manage specific business projects that they have been assigned to in JIRA Core. Project administrators can customize their projects, add and remove users, and perform certain configuration tasks.

This tutorial will teach you how to manage your project. Here's what you can count on learning:

Audience

Project Administrators, Administrators

Time

10 minutes

1. How to customize your project
2. How to add users to your project

Ready to exercise those product administrator muscles? Get started by customizing your project.

Let's get started »

Customizing your project

1. Customizing your project
2. Adding users to your project

As a project administrator, you have the ability to edit project role memberships, project versions and certain project details (project name, project description, project lead and URL). In this step, you'll customize your project by changing the name, the URL and the avatar.

If you're continuing from the Getting started as an administrator tutorial, you'll already have a project called Dragon Design Tees. For this tutorial, use the Dragon Design Tees project or another project to which you have been given project administrator access.

Access your project's administration page

1. Select **Projects** > **Dragon Design Tees** (or the project you wish to administer).
2. Select

   ⚙

   to open the Project Administration page:
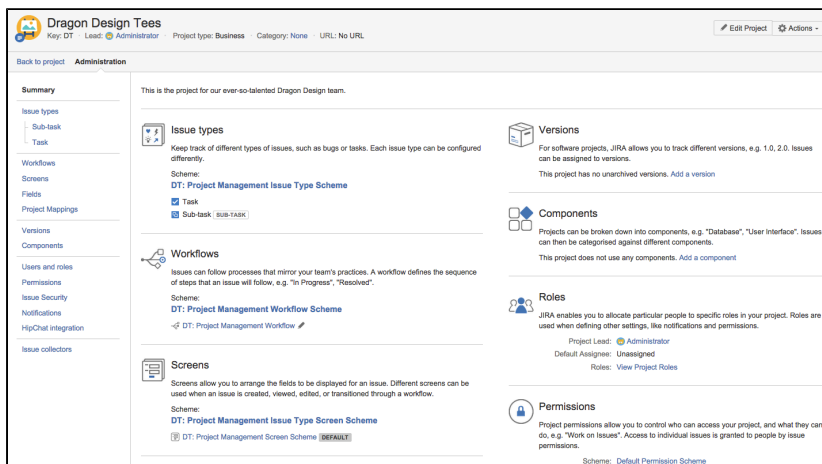
Accessing your projects

You can also access your projects by selecting

⚙ ▾

> **Projects**. You will see a list of the projects you administer, and can select one you want to work on.

Customize your project

Customizing your project will make it stand out, and make it easier for your teams to recognize if they're working on multiple projects.

1. On the Project Administration > Summary page, select **Edit project**.
2. Click **Select image** to change your project avatar.
3. Select an image from those displayed, or upload an image specific to your team.
4. Note that you can also change the project name and description in this Edit Project dialogue.
5. Select **Update** when you have made all changes.

Your changes will appear to all users who can access the project.

Success! You've customized your project avatar. Now, let's add a user to your project.

Next

Adding users to your project

1. Customizing your project
2. Adding users to your project

As a project administrator you can add groups or individual users to various roles in your project. Roles are project specific, so adding a group or individual user will only affect your project. You can also use the same process to remove access from groups or users. It's important to note that you can only add **existing** groups and users, you need to be a JIRA Core administrator to create the groups or users.

Viewing and changing project access

In this step, you will add one of your users (Emma) to the Administrators role, so she can help manage your project.

1. In your project, select **Project administration** > **Users and roles**.
2. Select Add users to role.
3. Search for Emma. You can add multiple users and groups, and delete those you have accidentally added in this Add users dialogue.
4. Select the Administrators role and select **Add**.

Emma will now have administrator access to your project. You can use the same process to add users or groups to the Administrators role. If you would like to give users restricted access to your project (so they can only view issues they're assigned to, for example), you can ask your administrator to create a new project role.

Well done! You have added Emma as an administrator on your project.

You've now completed the Project Administrator Getting Started guide. We suggest that you continue on to the Getting started as a user guide, so you have a better understanding of how your team can use JIRA Core.

Getting started as a user

1. Getting started as an administrator
2. Getting started as a project administrator
3. Getting started as a user

This tutorial shows you how to work with JIRA Core on a day to day basis. Here's what you can count on learning:

1. How to access your projects
2. How to work with issues
3. How to search for issues

By the end of this tutorial, you'll have a good understanding of how you can work with JIRA Core, and be able to navigate around a project and search for issues.

Audience
General users, project administrators, administrators
Time
20 minutes

Ready to get going? Let's begin.

Let's get started »

Accessing a project

1. Accessing a project
2. Creating and working with issues
3. Searching for issues and filtering

Business projects can be viewed as containers that hold issues. Each project has an associated workflow, and this workflow is applied to all issues held in the project. Start working on your assigned tasks by

accessing and viewing the project in which they're held.

Accessing a project

When you are given access to a project and you log in for the first time, you'll see the project listed in the **Projects** dropdown menu. Let's check it out.

1. Select **Projects** > **View All Projects**. A list of all projects you have access to will display.
2. Select **Dragon Design Tees**. You'll land on the project summary page, which displays recent activity and lets you easily track the status of issues in this project.
3. If you're participating in multiple projects, you can easily see which one you are currently working on, by selecting **Projects** and looking under Current Projects.

Viewing a project's issues

You can view and filter the issues in your project so that you see, for example, only issues that are assigned to you.

1. In your project, select **Issues** from the project sidebar. From here, you'll be able to preview issues and select preconfigured filters to change which issues are displayed.

Well done! You now know how to browse to a project, and view the project details. Next, you'll create and work with some of your own issues.

Next

Creating and working with issues

1. Accessing a project
2. Creating and working with issues
3. Searching for issues and filtering

An issue is the most basic entity in JIRA Core. Depending on your team and its needs, issues can represent different things. In the Dragon Design Tees project example, each issue would represent the work needed to create and complete new t-shirt designs.

Create your first issue

Let's create an issue to track the creation of a new t-shirt design.

1. Choose **Create** in the JIRA Core header.
2. Fill out the fields using the sample data is shown below. Only the fields with * are mandatory.
   - Project: *Dragon Design Tees*
   - Issue Type: *Task*
   - Summary: *Create a contractor resources spreadsheet*
   - Description: *The spreadsheet must contain a breakdown of all contractors and specific skill sets.*
   - Leave all other fields blank or at their default values.
3. Choose **Create** to create your new issue. A confirmation message will display for a few seconds. Make note of the issue key of your new issue (e.g. DT-1).

> **Need to create multiple issues?**
>
> Checking the "**Create another**" check-box on the 'Create Issue' dialog will keep the dialog open after you click Create.

Create a two more issues to familiarize yourself with the process.

Editing an issue

You can easily edit your issue to add more information, attach new files or screenshots, and more. Inline editing is the quickest way to edit an issue. To access all issue fields, including blank fields that don't appear when inline editing, you can use the Edit Issue dialogue.

1. In your project, select **Issues** from the project sidebar and open the first issue you created.

2. Hover over the Priority field. Click

to edit the field.
3. Change the Priority to "Critical" and click anywhere outside the field to save your change.

The issue will be updated immediately. Now, let's see how to start working on an issue.

Progressing the issue

Every issue has a lifecycle. In JIRA Core, the lifecycle of an issue is managed by a workflow. A workflow consists of the issue statuses (e.g. To Do) and the transitions between each status (e.g. Start progress). In this step, you'll start working on your issue by transitioning it from the To Do to In Progress statuses.

1. Open the first issue you created (e.g. DT-1). The issue should be in the To Do status.
2. Select **Start Progress**. The status of your issue will be changed to **In Progress**.
3. You can then choose to stop progress, or close the issue by marking it as Done and selecting a resolution.

Notice that resolving an issue prompts you to enter more information, whereas starting progress on the issue did not. Some issue transitions have screens associated with them in the workflow associated with your project. Remember, a project can only have one workflow associated with it.

Assign the issue to another user

You can better manage your work by assigning issues to other members of your team.

1. Open one of the issues you have created (e.g. DT-2).
2. Choose **Assign** on the issue.
3. Type **emma** in the **Assignee** field and select her as the assignee from the drop-down list that appears.
4. Type (don't copy and paste) the following text in the **Comment** field, then choose **Assign**.

```
Hi @emma, I've assigned this issue to you to work
on this week.
```

> You will notice a few things when you enter the comment:
> - When you start typing after the @ symbol, you will be prompted to choose a user: emma, in this example. Emma will be sent an email notification that links to the issue, when you save. This feature is called **mentioning a user**.
> - On choosing **Assign**:
>   - The issue will be assigned to Emma with a comment added to it.
>   - A link will be automatically created to the issue in the issue comment.

Awesome! You've created some issues, edited one, and assigned another to a member of your team. You're really getting the hang of working with issues. Now, you'll learn how to best search for the issues you need to work on.

Next

Searching for issues and filtering
1. Accessing a project

> **About workflows**
>
> JIRA Platform ships with default workflows. The workflows can be changed and customized, but this can only be done by an Administrator.

Creating issues is important, and its equally as important to be able to search and manage multiple issues to ensure you and your team work together well. In this step, we will show you how to work with multiple issues. You will learn how to use different search techniques to find issues. We will also show you how to share search results with your team and report on issues.

Create some more issues

Before you start, you are going to need a few more issues. Create a few more in your JIRA Core project, using the sample data below.

- **Issue Type** = `Task` and **Summary** = `Brainstorm new designs`
- **Issue Type** = `Task` and **Summary** = `Approve new t-shirt design`
- **Issue Type** = `Task`, **Summary** = `Send mockup to printers` an d **Assignee** = `kate`

> **Want to create issues in bulk?**
>
> If you're familiar with the CSV format, you can create a CSV file to import issues in bulk. This can be handy if you're handling a lot of data.

Search for issues

In this example, we are going tackle a common scenario: searching for all unresolved issues assigned to you. You might regularly run a search like this to check your backlog of work.

1. From your JIRA Core Cloud site header, select **Issues** > **Search for Issues**. You should see issues from your demo project and any other projects you have access to.
2. Set **Assignee** = **Current User** in the search criteria.

   ℹ️ *Notice that the search results refresh when you select new criteria.*
3. Choose **More** > type **Resolution** then select it.
4. Set **Resolution** = **Unresolved**. The search results will show the issues that are unresolved and assigned to you.

If you are thinking that it would be handy to be able to rerun this search, we have got you covered! Hover over the

»

icon in the top left and choose **My Open Issues**. Keep this screen open for the next step.

Save your search

If you run a search with the same criteria frequently, you may want to save it as a **filter**. This lets you run the search again with a single click, rather than selecting the same criteria every time. For example, you may use a filter to review your open tasks for the day.

In this step, you will search for all tasks assigned to Kate in the Dragon Design Tees project, and then save this search as a filter.

1. Select **Issues** > **Search for Issues** to start a new search.
2. Set **Project = Dragon Design Tees** and **Assignee = kate** as the criteria. You should see at least one issue.
3. Select **Save As** (above the search criteria), enter **Kate Dragon Design issues** as the **Filter Name** and save it.

That's it! Hover over the

»

icon in the top left. You can see your new filter under the **Favorite Filters** s ection. Just click it to run it. Let's now look at some of the ways that you can

use your new issue filter.

Share your search results

Getting your team on the same page is easy with shared filters. You could share a filter with your team that shows the unresolved stories for a development iteration, or the critical issues in a support backlog.

Here are two ways that you can share search results:

Email the search results

Run the desired filter, then choose **Share**. Enter the users that you want to share the filter with and they will be emailed a link to your filter (if you have email notifications set up).

Share the search results via a dashboard

The dashboard is the screen that all JIRA Core users see when they first log in. You can show a filter's results on a dashboard and share it with other users.

1. Choose **Dashboards** > **Manage Dashboards**, then choose **Create new dashboard**.
2. Name your dashboard **Dragon Design Tees** and choose the **+Add** b utton next to **Add Shares** to share it with everyone.
3. Leave the other fields and choose **Add**.
4. Choose **Dragon Design Tees** in the **Favorite Dashboards** section to configure it.
5. Choose **add a new gadget** to open the 'Gadget Directory'.
6. Enter filter results in the search box and choose **Add It Now**.
7. Enter **Kate Dragon Design issues** in the **Saved filter** field and choose **Save**.

Other users can now add this dashboard by choosing it as a favorite.

That's it! You've now performed some searches and filters, and learned to to save and share them with your team. You're ready to jump right in and experience the full power of JIRA Core!

You've now completed the Getting started as a user tutorial. For more information on using JIRA Core, continue to the JIRA Core documentation home.

Administering a project

Working in JIRA Core is all about working with issues in projects. A JIRA Core project is a collection of issues, and issues are the basic packets of work that need to be done. A project allows you to apply a process to each issue (via an associated workflow) within your project. You could choose to create a project to track leave requests, or a project to create and monitor a marketing campaign, JIRA Core allows you to modify your project to suit your needs.

Project details	A project must have a name, a project key and a project lead. These details help identify the project and the issues within it. *Learn more: Editing a project's details*
Project access	Access to your project is controlled through project role membership. You can assign individual users or groups to the project roles. *Learn more: Managing project role memberships*
Versions	A version is a way to group issues within a project. Versions have a start and end date, and can be used to effectively group issues (and therefore your work) into deliverables by date *Learn more: Organizing work with versions*
Components	A component is another way to group issues within a project. Components don't have dates assigned to them, but they can have a component lead

who could be a subject matter expert, or a team lead for that area of the project.

Learn more: Organizing work with components

Workflows

Each project has at least one workflow that can be applied to it's issues. A workflow controls how an issue progresses, from creating the issue to closing the issue.

Learn more: Workflows

Issues

Issues are the packets of work that need to be done, and each issue contains information held in fields. You can customize your issues by changing the fields, to make sure you always have the information required to complete your work.

Learn more: Customizing the issues in a project

Editing a project's details

You are able to edit the project name, project description, project avatar and project URL of a project that you have the Project Administrator permission for.

Note that when you change the name of project, this change will also be reflected in any saved filters that contained the project's name, so they **won't** be affected by the change.

Editing a project

1. Navigate to the administration page for the project:
   - Choose

     ⚙

     > Projects, or
   - Navigate to the desired project's summary and click the **Project Administration** button at the bottom of the project navigation sidebar.
2. Select **Edit project**.
3. Make your edits.
4. Select **Update**.

Managing project role memberships

You can use project roles to easily associate users and groups with a particular project. For example, you may want to send notifications to a specific set of users associated with your project, and by adding them all to a project role, you can then use that project role to control who receives the notifications. You can also use project roles to restrict how much access certain users or groups have. Unlike groups, which have the same membership throughout your application, project roles have specific members for each project.

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

Viewing and editing project role members

1. Log in as a project administrator and open your project.
2. Select

   ⚙

   > **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.

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
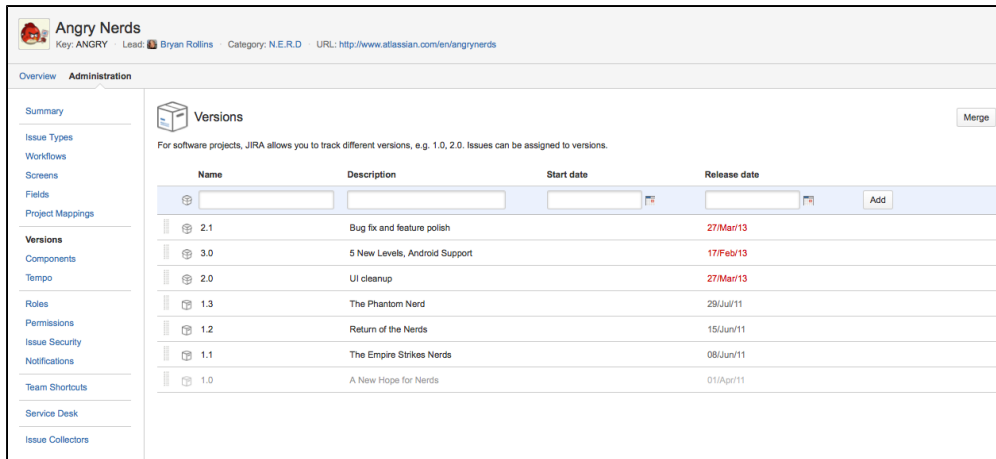
Add a new version

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 pl anned 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** fr om 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** fro m 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** fro m 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. Versi ons selected on this list will be removed from the system. All issues associated with these versions will be updated to reflect the new version selected in the 'Merge To Version' select list. It is only possible to select one version to merge to.
3. Click the **Merge** button. If you are shown a confirmation page, click **Merge** again to complete the operation.

Reschedule a version

Recheduling a version changes its place in the order of versions.

- On the 'Versions' page, click the

  ⋮⋮

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

Organizing work with components

Components are sub-sections of a project. They are used to group issues within a project into smaller, more manageable groups. You can set a default assignee for a component, and this will override the project's default assignee for issues in that component.

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.

On this page:

- Managing a project's component s
- Adding a new component
- Selecting a default assignee
- Editing a component 's details
- Deleting a component

Managing a project's components

The easiest way to manage a project's components is through the Components page.

1. Choose
   ⚙
    > Projects, and click the name of the project.
2. Choose **Components** in the sidebar. The **Components** page is displayed, showing a list of components and each component's details. From here, you can manage the project's components as described below.

Adding a new component

1. The Add Component form is located at the top of the 'Components' screen.
2. Enter the **Name** for the component. Optionally, enter a **Description** and select a **Component Lead** a nd **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 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.

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.Each project in JIRA

requires a workflow. If you want to keep things simple you can use the default built-in workflows in JIRA that consist of moving an issue through the basic statuses. Your workflow will depend on which Project types you are using and what workflows have been created in your JIRA application.

What you can do...	Documentation
- Edit existing workflows - Create new workflows - Configure existing workflows	Working with workflows
- Add a workflow scheme - Configure a workflow scheme - Managing workflow schemes	Configuring workflow schemes
- Import and export workflows - Activate and deactivate workflows	Managing your workflows
- 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

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 Core allows you to customize issues by changing the configuration and behavior of these fields to suit your team's needs.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.

You can make this...	...into this!

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.

Working in a project

A project in JIRA Core is a collection of issues. Your team could use a project to coordinate the development of a product, track a project, manage a help desk, and more, depending on your requirements. A project can also be configured and customized to suit the needs of you and your team.

This section of the documentation covers working in a project that's already been set up for you.

User profile	Working in a project, and JIRA Core in general means you can log in. And if you can log in, you have a profile that you can edit to make sure it's configured how you want it to be configured.
	Learn more about how to manage your profile
Viewing a project	Getting to grips with how your project looks and is set up will help you work more effectively and efficiently.
	Learn more about how to view a project and its contents
Working with issues	Issues are the packets of work that need done in a project. Knowing what you can do with them, and how to do it, is the basis of all work in JIRA Core.
	Learn more about working with issues
Searching for issues	You can't work on issues if you can't find them. Working out what the most effective way to search for issues is, and how to use the powerful advanced search function, will help you always be able to locate your issues when you need them.
	Learn more about searching
Reporting	Reporting helps you manage and track your work.
	Learn more about reporting in JIRA Core
Gadgets	Gadgets are nifty ways of displaying data about your project and your issues. Setting them up and using the right ones makes sure you always have your stats at hand.
	Learn more about gadgets
Dashboards	You can set up dashboards, and share them with team members, so that everyone is viewing the same information on your projects and issues. You can add gadgets to dashboards, and create wallboards which allow you to display your stats anywhere you can connect to.
	Learn more about dashboards and wallboards

Managing your user profile

You can manage your JIRA settings (e.g. your password, email address, or the format in which you would like to receive email notifications) in your user profile. Your user profile also displays recent work in the Activity Stream, and contains useful shortcuts to issues you have been working on or reported.

To manage your user profile:

Choose **your user name** at top right of the screen, then choose **Profile**.

On this page:

- Editing your user details
- Changing your avatar
- Choosing your homepage
- Managing email notifications
- Managing your user preferences
- Managing service desk preferences
- Managing your OAuth and login tokens

Editing your user details

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)

    ℹ️ 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 service desk preferences

Service desk agents can enable or disable the **Pre-populated commenting** field by editing their user profiles. This setting can help save time by pre-filling conversation greeting text when agents comment on customer issues. When enabled the following text appears in the comment field and in the email notification sent to customers:

Managing your OAuth and login tokens

An OAuth access token is issued by JIRA to give gadgets access to restricted data on an external, OAuth-compliant web application or website (also known as a "consumer"). Check out Allowing OAuth access for recommendations on when to issue or revoke OAuth access tokens.

If you are accessing your JIRA applications in a public environment, you can clear you login tokens by clicking the **Clear all Tokens** link in the Details section of your Profile.

Allowing OAuth access

About OAuth access tokens

OAuth access tokens allow you to:

- Use a JIRA gadget on an external, OAuth-compliant web application or website (also known as a 'consumer')
- Grant this gadget access to JIRA data which is restricted or privy to your JIRA user account.

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.

On this page:

- About OAuth access tokens
- Before you begin
- Issuing OAuth access tokens
- Revoking OAuth access tokens

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

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

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.
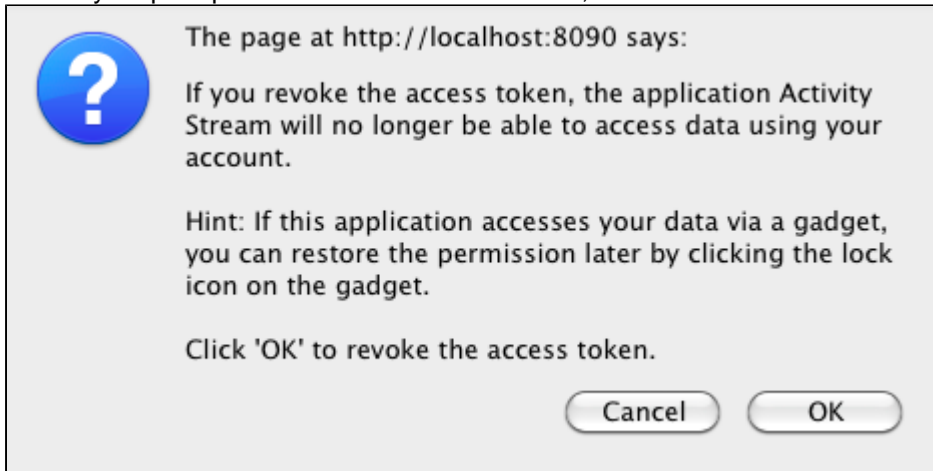   *Screenshot: Viewing your OAuth Access Tokens*

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 OAut h 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 page at http://localhost:8090 says:
>
> If you revoke the access token, the application Activity Stream will no longer be able to access data using your account.
>
> Hint: If this application accesses your data via a gadget, you can restore the permission later by clicking the lock icon on the gadget.
>
> Click 'OK' to revoke the access token.
>
> [Cancel]   [OK]

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. ℹ️ If the consumer is another Atlassian application, this information is obtained from the Con sumer 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	

Viewing a project

When you select a project to view for the first time, you're taken to the project Activity page. If you've viewed the project before, you'll be taken to the last screen you visited on it. The Activity page provides a summary of your project activity, and further navigation which allows you to 'dig down' into further detail. The sidebar provides navigation to all the issues in the project, and the reports available. If you're a project administrator, you can also add unique shortcuts to information and other webpages to the sidebar, which are accessible to all users with access to the project.
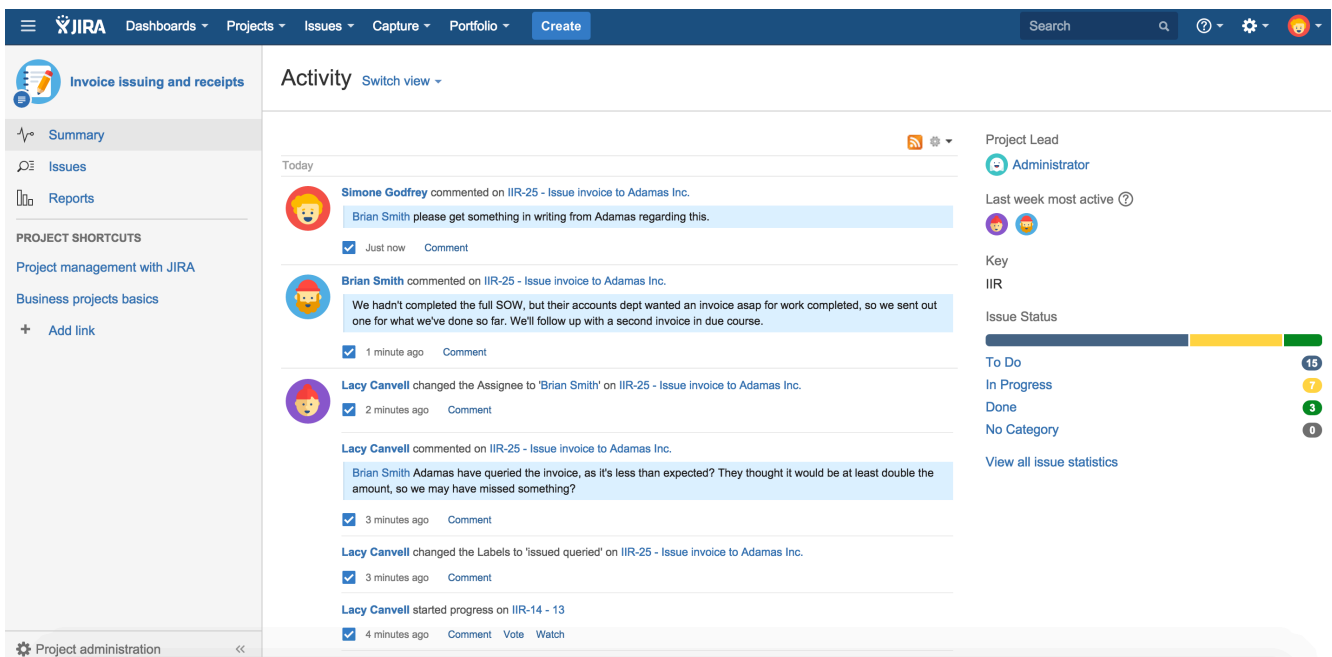
You can access the project Activity page by selecting the **Project** drop-down, and selecting your project from the list. If your project is not listed, select **View all projects** to search for your project. Once you're viewing the project, click the **Summary** link to view the Activity page.

From the project summary screen, you can view the following by selecting the link in the project navigation sidebar:

- **Project:**
    - Summary (as shown above)— Shows recent activity in your project, the project lead, the most active users within your project, and the project key.
    - Issues— Takes you to the issue navigator, which shows a list or detailed view of issues in your project.
    - Reports – Shows reports on statistics for particular people, versions, issues or other fields within issues.
    - Components* — Shows a summary of all components for a given project.
    - Versions* — Shows a snapshot of all versions within a project, and can be filtered by released and unreleased.

* Versions and Components are only available if your project administrator has created versions or components within the project.

- **Project Shortcuts:**
    - Project shortcuts can be added to your project navigation page to any online resources your team may want to access. These shortcut links are available to everyone who has access to the project.

Viewing a project's versions

JIRA's **Versions** page shows a summary of all versions in a project. These can be further filtered by released versions and unreleased versions, and you can search by text contained within a version. Versions can only be displayed if your project administrator has created versions for your project. When versions have been created, you can assign the versions to your issues via the **Affects Version/s** and **Fix Version/s** fields

The Versions page is a visual representation of how your versions are progressing. The data represented is taken from the Fix Version/s field, and shows the status of the issues assigned to that version. Selecting the version link in the Version column will display further detail on that version, as well as giving you the option to view release notes, and to release that version (providing you have the correct permission to do so).

To view a project's releases:

1. On the top navigation bar, click the white triangle next to **Projects**. The projects drop-down will display. ✅ **Tip:** You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project you wish to browse. If the project is not displayed in the drop-down, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA instance, and select your project from there.
3. Click **Versions** in the project navigation sidebar. A list of versions for your project is displayed.

Viewing a project's components

JIRA's **Components** page shows a summary of all components (if any have been created) in a project. You can search for components by text contained within the component's name or description.

To browse a project's components,

1. On the top navigation bar, click the white triangle next to **Projects**. The projects drop-down will display. ✅ **Tip:** You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project you wish to browse. If the project is not displayed in the drop-down, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA instance, and select your project from there.
3. Click **Components** on the left of the page. A list of components for your project will display (see screenshot below).
    - Click the name of a component to view all the issues related to the component.

Viewing a project's issues

The Activity page allows you to access the issues contained in that project via the project navigation sidebar. The issues are displayed as a list, and have several filters you can select to view the issues relevant to you.

To view a project's Issues,

1. On the top navigation bar, click the white triangle next to **Projects**. The projects drop-down will display.

> ✅ **Tip:** You can access your current project directly by simply clicking the **Projects** link instead of the triangle.

2. Click the project you wish to browse. If the project is not displayed in the drop-down, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA instance, and select your project from there.
3. Click the **Issues** tab in the project navigation sidebar. An embedded issue navigator is shown for all the issues in your project.
   Tip: To view an expanded view of an issue, click the expand (

   ↙↗

) icon on the issue or any issue link in the issue (description or comment).

Project shortcuts

You can add shortcuts to the project navigation sidebar to web pages and online information that you think your team may find useful. When you add a link, it will be available to all users who can access the project. The shortcuts can be accessed when the sidebar is collapsed by hovering over the shortcut icon, however to add, edit or delete a shortcut the sidebar must be expanded.

To add, delete or edit a link, you need to have the JIRA Administrator permission or the Project Administrator permission. When adding or editing a link you also need to have the full web address, and you can optionally give it a label that will show in the project navigation sidebar. Labeling the shortcuts makes it easier to identify where the link will take you.

Add a shortcut

1. On the top navigation bar, click the white triangle next to **Projects**. The projects drop-down will display.
   ✅ **Tip:** You can access your current project directly by simply clicking the **Projects** link instead of the triangle.
2. Click the project you wish to browse. If the project is not displayed in the drop-down, click **View All Projects**, which allows you to view a list of all accessible projects on your JIRA instance, and select your project from there.
3. Select **Add link** in the project navigation sidebar.
4. Enter the web address for your shortcut and an optional label, then select **Add**. Your shortcut will display in the project navigation sidebar.

> Note that you must enter a valid web address that JIRA recognizes. These must be prefixed with a valid URI. The valid URI's are listed below.

Edit a shortcut

1. Access the project navigation sidebar as you did above, and locate the shortcut you'd like to edit.
2. Select the drop-down arrow to the right of the shortcut.
3. Select **Edit**.
4. Edit the link details, and select **Save**. Your shortcut has been edited.

Delete a shortcut

1. Access the project navigation sidebar as you did above, and locate the shortcut you'd like to delete.
2. Select the drop-down arrow to the right of the shortcut.
3. Select **Delete**. Confirm the deletion. Your shortcut has been deleted and removed from the project navigation sidebar.

List of Valid URI's

"http://", "https://", "mailto:", "skype:", "callto:", "facetime:", "git:", "irc:", "irc6:", "news:", "nntp:", "feed:", "cvs:", "svn:", "mvn:", "ssh:", "itms:", "notes:", "smb:", "hipchat:"

Working with issues

Need help working with issues? JIRA Coreallows you to create issues quickly, assign them to the right person, and get working on them in

seconds! On this page, you'll find a quick overview for everything that you can do with an issue, as well as links to pages with more detail.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.

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 Core, an issue is essentially a packet of work. It could be a small task like "Remember to order pizza for charity night", or a large chunk of hard work like "Build bridging wall between house and garage"! It all depends on your project, and how you and your team decide to break down your work into issues.

An issue is broken down into several key areas. Here's an example of an issue to build a wall, and we've pointed out the key areas for you.

<screenshot when tied down>

Next steps

Check out the following pages to reach issue ninja status:

- Creating issues and sub-tasks
- Attaching files and screenshots to issues
- Editing and collaborating on issues
- Logging work on issues

Attaching files and screenshots to issues

To share information with your team, you can attach documents, images, and screenshots to your JIRA application issues.

Multiple files can be attached to an issue. Click an image thumbnail to open a preview, and if there is more than one image in the gallery, navigate to the next image preview by clicking the right arrow in the preview.

Before you begin

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

∨ JIRA administrator set permissions

- You can attach files and screenshots if your JIRA administrator has file attachments enabled.
- You need the **Create Attachments** permission in the appropriate projects.
- The screenshot feature only works with Windows or Mac client. If you use another operating

system, you can attach a screenshot using the file attachment feature. For Linux users, please see our article for enabling this feature.
- If your JIRA admin has disabled thumbnails in JIRA's attachment settings, the image files will appear as a list.
- If your JIRA admin has disabled ZIP support in JIRA's attachment settings, the attachments feature will not be available. You must download the zip file to your computer before accessing its individual files.
- To remove attachments from an issue, you need one of the following the project permissions in that issue's project:
  - **Delete Own Attachments** — to delete files that you have added to the issue.
  - **Delete All Attachments** — to delete files that anyone has added to the issue.

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

Adding attachments

You can add file and image attachments to any issue. To add an attachment when you first create an issue, copy a file from your computer and paste it directly in the **Create Issue** dialog. To add attachments to an existing issue, open the issue and follow these steps:

1. Click **More >Attach files**.
2. Add a file(s).
3. Click **Attach** or **Open**.

You can also drag and drop files onto an issue to attach them.

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

Sorting and managing attachments

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

🗑

.

The selected criteria will be lost once you log out.

Accessing ZIP file contents

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

Capturing and attaching screenshots

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

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

Creating issues and sub-tasks

The building blocks of any project are issues. Issues act as the packets of work that travel through their respective workflows within their projects, until the work is completed. An issue may also have sub-tasks that can be assigned and tracked individually, as well as issue level security to restrict the issue to select members of your team. On this page, you'll learn more about creating and converting issues and sub-tasks, and setting issue level security. If you are looking to import multiple issues (and sub-tasks) using a CSV file, you can find the import process explained in more detail here.

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

Before you begin

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

Creating an issue

1. Click **Create** at the top of the screen to open the **Create Issue** dialog box.
2. Select the relevant **Project** and **Issue Type** in the **Create Issue** dialog box.

3. Type a **Summary** for the issue and complete any appropriate fields — at least the required ones that are marked by an asterisk.
   If you want to access fields that are not shown in this dialog box, or you want to hide existing fields:
   a. Click the **Configure Fields** button at the top right of the screen.
   b. Click **Custom** and select the fields you want to show or hide by selecting or clearing the relevant check boxes respectively, or click **All** to show all fields.
      When you next create an issue, these selected fields will be displayed.
4. Optional: To create a series of similar issues – with the same **Project** and **Issue Type** – select the **Create another** checkbox at the bottom of the dialog. Depending on your configuration and the values you may have specified when creating previous issues, some of the fields in the new Create Issue dialog box may be pre-populated. Make sure you check they're all correct before creating the next issue.
5. When you are satisfied with the content of your issue, click the **Create** button.

Cloning an issue

Cloning 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 most of the same details stored in the original issue — e.g. Summary, Affects Versions, Components, etc. However, some details aren't cloned — e.g. Work Log, Comments, Issue history, and Links to Confluence pages. Statuses and resolutions aren't cloned as well; the statuses return to the first step of the corresponding workflow, and the resolutions are cleared. 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.

Creating issues using the CSV importer

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

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

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

On this page:

- Preparing your CSV file
- Running the CSV file import wizard
- Tips for importing CSV data into issue fields

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

1. Preparing your CSV file

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

Preparing your CSV file

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

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

CSV file requirements

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

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

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

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

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

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

    For example, this is valid:

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

    ... but this is not valid:

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

Encapsulating JIRA data structure in your CSV file

Capturing data that spans multiple lines

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

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

Treating special characters literally

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

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

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

Aggregating multiple values into single issue fields

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

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

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

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

Importing attachments

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

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

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

Importing issues into multiple projects

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

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

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

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

Importing worklog entries

Your CSV file can contain worklog entries. For example:

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

To track time spent, you need to use seconds.

Importing to multi select custom fields

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

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

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

Importing cascading choice custom fields

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

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

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

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

JRA-34202 - Allow CSV import to support Multi-Level Cascading Select plugin fields
OPEN

).

Running the CSV file import wizard

Before you begin: If your JIRA installation has existing data, you should back it up.

1. Select **Issues > Import Issues from CSV** to open the **Bulk Create Setup** page. (If you do not have the option **Import issues from CSV**, your JIRA Admin must update the JIRA Importers plugin to version 6.2.3 or above.)
2. On the **Setup** page, select your **CSV Source File**.
   Leave the **Use an existing configuration file** checkbox cleared if you do not have a configuration file, or if you want to create a new configuration file. Configuration files specify a mapping between column names in your CSV file's header row and fields in your installation.
   - If you select this option, you will be asked to specify an **Existing Configuration File**.
   - If you do not select this option, then at the end of the CSV file import wizard, JIRA will ask you if you want create a configuration file that you can use for subsequent CSV imports.
3. Click the **Next** button to proceed to the **Settings** step of the CSV file import wizard. Complete the required fields.
   - If your CSV file uses a different separator character other than a comma, specify that character in the **CSV Delimiter** field. If the separator is a 'Tab', this can be entered using the format '**/t**'.
4. Click the **Next** button to proceed to the **Map fields** step of the CSV file import wizard. Here, you can map the column headers of your CSV file to the fields in your selected project. If you want to select specific JIRA field values to map specific CSV values to, tick the checkbox for **Map field value**.
   ℹ️ **Note:** You must map a CSV field to the issue's summary field. This ensures the issues created have a summary.
5. Click the **Next** button to proceed to the **Map values** step of the CSV file import wizard. On this step of the import wizard, you can select which specific CSV field values you want to map to which specific issue field value. For example, your issue types you may have a CSV field value of "Feature Request", which you may want to map to the issue type field value "New Feature".
   ℹ️ **Please note:**

- Any fields whose **Map field value** checkboxes were selected in the previous step of the CSV file import wizard will be presented on this page.
- Leave a field cleared or clear any content within it if you wish to import the value 'as is'.
- If you are importing a username-based CSV field (e.g. **Reporter** or **Assignee**) and you do not select the **Map field value** checkbox for this field in the previous step of the CSV file import wizard, then the importer will automatically map imported usernames from the CSV file to (lowercase) JIRA usernames.
  ⓘ Regardless of whether or not you select the **Map field value** checkbox, JIRA will automatically create usernames based on the data in your CSV file if they have not already been defined in JIRA.

6. Click the **Begin Import** button when you are ready to begin importing your CSV data into JIRA. The importer will display updates as the import progresses, then a success message when the import is complete.
7. If you're confident your import is correctly set up, click the **Begin Import** button. Your import will begin and once complete you will be informed of any errors. If you'd like to check your import first, click the **V alidate** button and JIRA will validate your import and inform you of any expected errors or warnings. You can then go back and correct these before running your full import.

ⓘ **Note:**

- If you experience problems with the import (or you are curious), click the **download a detailed log** link to reveal detailed information about the CSV file import process.
- If you need to import another CSV file with the same (or similar) settings to what you used through this procedure, click the **save the configuration** link to download a CSV configuration file, which you can use at the first step of the CSV file import wizard.

Congratulations, you have successfully imported your CSV data into JIRA! If you have any questions or encounter any problems, please contact Atlassian support.

Tips for importing CSV data into issue fields

Below are some helpful tips when importing data from your CSV file into specific issue fields:

Issue Field	Import Notes
Project	CSV data is imported on a per-project basis. You can either specify an existing project(s) as the target, or the importer will automatically create a new project(s) for you at time of import.
Summary	This is the only required field.
Component(s)	You can import issues with multiple components by entering each component in a separate column.
Affects Version(s)	You can import issues with multiple 'Affects Versions' by entering each version in a separate column.
Fix Version(s)	You can import issues with multiple 'Fix Versions' by entering each version in a separate column.
Comment Body	You can import issues with multiple comments by entering each comment in a separate column.
Due Date	Please use the date format specified on the second step of the CSV import wizard.
Issue Type	If not specified in your CSV file, imported issues will be given the default (i.e. first) Issue Type, as specified in your JIRA instance. For more information, see Defining issue type field values. You can also create new values on-the-fly during the import process.
Labels	You can import issues with multiple labels by entering each label in a separate column.

Priority	If not specified in your CSV file, imported issues will be given the default (i.e. first) Priority as specified in your JIRA instance. For more information, see Defining priority field values. You can also create new values on-the-fly during the import process.
Original Estimate	The value of this field needs to be specified as number of seconds.
Remaining Estimate	The value of this field needs to be specified as number of seconds.
Time Spent	The value of this field needs to be specified as number of seconds.
Users	You can choose to have the importer automatically create JIRA users for any values of the Assignee or Reporter field. • Users will be created as active accounts in JIRA. Users will need to get their passwords emailed to them the first time they log into JIRA. • Users with no real name will get the portion of their email address (login name) before the "@" character as their Full Name in JIRA. • If you are using External User Management, the import process will not be able to create users; instead, the importer will give you a list of any new users that need to be created. You will need to create the users in your external user repository before commencing the import. • If you have a user-limited license (e.g. personal license), and the number of required users is larger than the limit, then the import will be stopped. A page will be displayed showing a list of users that can't be created. • If Assignee and Reporter are not mapped, then no usernames are created.
Other fields	If you wish to import any other fields, you can choose to map them to specific JIRA custom field(s). If your custom fields don't exist yet in JIRA, the importer can automatically create them for you. If your custom field is a date field, please use the date format specified on the second step of the CSV import wizard.

Editing and collaborating on issues

Complete your work more efficiently with these tips and tricks for editing and collaborating on 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 mentioning other people
- 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.

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.
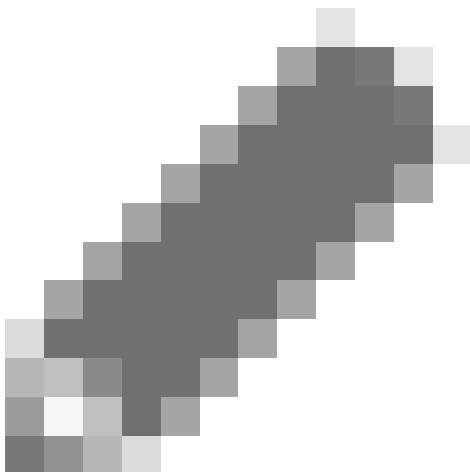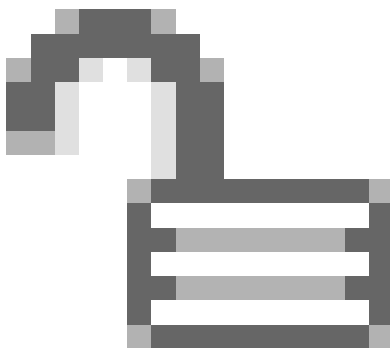
Commenting on issues
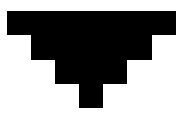
> **What permissions do you need?**
> To add comments to an issue, i.e. to see the **Comment** button, you must have both of the following project permissions for the issue's relevant project:
>
> * **Browse Project** permission to view the issue to be commented on
> * **Add Comments** permission to 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**
Delete a comment	On the comment you wish to delete, select the trashcan icon located on the comment. Confirm that you want to remove this comment from the issue by selecting **Delete** when prompted.

Edit a comment	Select
	located on the comment, and edit the text or restrictions (**Viewable by...**) as needed. When you save your revised comment, you'll see 'edited' displayed to indicate that the comment has been edited:
	⌄ Susan Griffin added a comment - 15/Mar/13 2:36 PM - *edited* ‹› ✎ 🗑 I think this nerd is looking better. Thanks for fixing it up!
	You can hover over 'edited' to see who edited the comment and when.
Link to a comment	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.
Restrict a comment	Apply viewing restrictions to a comment by selecting the open padlock icon
	(or 🔒 ▼ Restricted to Users if restrictions already apply).

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, comment, or any rich-text field, you can expand the simple wiki editor toolbar to format your text and select **preview** to see how your formatted text will appear. Note that your JIRA administrator can enable, disable and configure the which allows you to use wiki markup, so your options may vary slightly.

Tracking issues with labels

Labeling helps you categorize and search for an issue. When viewing an issue, select **More > Labels** to add or remove labels, which will appear in the Details section:
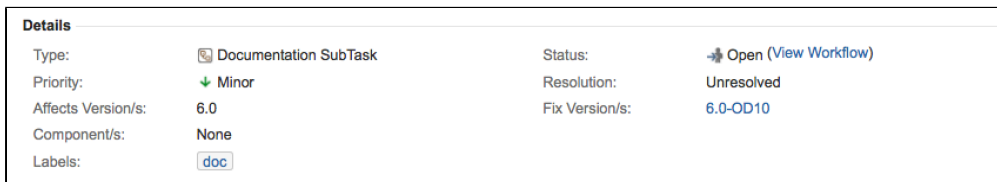
You can click a label (e.g. **doc** in the above screenshot) to jump to the Issue Navigator and see a list of all issues that have this label. You can also add the Labels Gadget to your dashboard to quickly find issues with labels relevant to you and your team.

Watching and voting for issues

> **What permissions do you need?**
> To view other users watching or voting for an issue, you need the **View Voters and Watchers** and **Manage Watcher List** project permissions.

If your administrator has set up the needed notification scheme, you can select **Start watching this issue** to be automatically notified of issue updates. You can also click the number of watchers on the issue to add other JIRA users as watchers.

If your administrator has enabled the voting on issues, you can select **Vote for this issue** to encourage the responsible team to resolve or complete the issue.

Linking issues

Issue linking allows you to create an association between two existing issues on either the same or different JIRA servers. For example:

- An issue may *relate to* another.
- An issue may *duplicate* another.
- An issue may *block* another.

Issue linking also allows you to:

- Create a new linked issue from an existing issue in a service desk or business project.
- Create an association between an issue and a Confluence page.
- Link an issue to any other web page.

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

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 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.
   > ⓘ 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.
     > ⓘ 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 applicati on links* between your JIRA site and the other JIRA site containing the issue(s) you want to link to.

ⓘ 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.
   > ⓘ **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 **Se rver** 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 **Fi nd 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, you discover that a reciprocal link from the remote issue back to your issue has not been created, then your JIRA system administrator has most likely created only a one-way link from your JIRA site to the remote JIRA site.

✅ **Solution:** Ask your JIRA system administrator to configure *fully reciprocal application links* between your JIRA site and the remote JIRA site.

❌ **Problem:** If you attempted to create a reciprocal link but received the following message:

'**A reciprocal link from issue 'XYZ-123' back to this issue was not created as the remote JIRA server returned the following error: No Link Issue Permission for issue 'XYZ-123'.**' (where 'XYZ-123' is the issue key on the remote JIRA site),

then a reciprocal link on the remote JIRA site will not have been created, because the user account through which you authenticated on the remote JIRA site (at step 3 above) does not have the Link Issues project permission.

✅ **Solution:**

- Ask the JIRA project administrator(s) on the remote JIRA site to grant your user account the Link Issues project permission for the relevant project(s) to which you need to create issue links.
- Alternatively, if the application link between your JIRA site and the remote JIRA site use OAuth authentication and you suspect you may have authenticated on the remote site with another user account that does not have the Link Issues project permission, repeat the procedure above but during the 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.

Create a new linked issue from an existing issue in a service desk or business project

> To create a linked issue, you need to have Create issue and Linked Issues permissions in the destination project(s).

To create a linked issue:

1. Open the issue from which you wish to create the linked JIRA issue.
2. In the Issue screen, select **More > Create linked issue** to display the **Create Linked Issue** dialog box
3. ✅ Keyboard Shortcut: **'.' +** start typing **Create linked issue.** The newly created linked issue contains the same Project, Issue Type, and Summary information stored in the original issue. It is also linked to the service desk issue, in this case CCF-3.

4. Select the destination **Project** in which the new linked issue is to be created.
5. Select the correct Issue Type for the new linked issue.
6. In the **Linked issues** field, specify issue(s) to be linked to your new linked issue.
7. Edit the linked issue **Summary.**
8. Edit the **Description** and describe why you are linking these issues.
9. Select the **Copy attachments** checkbox to include any attachments from the original issue.
10. Select the **Copy links** checkbox to include any URLs from the original issue.
11. Click the **Create** button at the bottom of the dialog.

Your linked issue has now been created.

Creating a link to a Confluence page

> This feature is only supported in Confluence versions 4.0 or later.

⚠️ To create this type of link, your JIRA system administrator needs to have configured an *application link* be
tween 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.
   ℹ️ 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+Demon
     stration+Space
     ```

   - Click the **search for a page** link. The **Link** dialog box is replaced by the **Find a Confluence**

page dialog box.

ℹ️ 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 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. See Advanced searching for more information.

ℹ️ 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.

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 **C ancel** 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.

      ⓘ **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 possible[1]. 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	• 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	• 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	• 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	• 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	• 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	• 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	• Current user has 'edit issue' permission for all the selected issues
Change Priority	• 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	• 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	• 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	The 'Change Custom Fields' operation is available only if: • 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	• Your workflow must allow editing of closed issues
Change Sprint	You need to specify the sprint ID. • 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 Core to track and review, and inform teams about issues dates. The powerful scheduling feature allows you to perform fixed and relative date searches based on specific due dates as well as arbitrary search periods. You can also perform advanced searches using JIRA Query Language.

Scheduling an issue

To schedule an issue, populate its **Due** date field. This can be done either when creating an issue, or at a later stage by editing the issue.

> To enable Issue Scheduling, at least one group or project role must be given the Schedule Issues permission by your JIRA administrator. Only users with the Schedule Issues permission can populate the **Due** date field.

Searching by due date

You can use either basic search or advanced search to search for issues by their Due Date.

Using simple search

You can search for issues using the search form in Issue Navigator (see Searching for issues). There are two ways to search for issues based on the **Due** date field. The first way is using fixed date values, the second is

using periods that are relative to the current date.

Fixed date searches

There are two text fields in the search form that allow searching based on the **Due** date field.

- To search for all issues that are due after a certain date, enter the date in the Due After text field. For example, to find all issues that are due after 1st June 2010, enter 1-6-2010 in the Due After field. You can also use the Calendar popup to select a date by clicking the calendar icon to the right of the field.

- To search for issues that are due before a certain date, enter the date in the Due Before text field. For example, to find all issues that are due before 1st July 2010, enter 1-7-2010 in the Due Before field.

To search for issues that are due between two dates, populate both the Due After and the Due Before fields.

Relative period search

It is possible to perform a search that is relative to the time when it is run. For example, it is possible to do a search for issues that are due seven days from now. To do this, enter 7d in the Due Date To text field of the Issue Navigator. If the search is saved and run the next day, the issues that are due in seven days from the time that the search is run will be retrieved. Thus, this search will find all issues that are due within a week every time it is run.

The values that are entered in the Due Date From and Due Date To fields have to conform to a special syntax (described below). However, it is also possible to use the Due Date popup by clicking the icon to the right of the Due Date To text field to specify the search period.

Due Date Popup

Use the Due Date popup to do the following:

- To search for issues that are overdue at the time of the search, select the first radio button, and click **OK**.
- To search for issues that are overdue by more than a certain number of days, populate the text field in the second row, and click **OK**.
- To search for issues that are due in the next certain amount of days, and are not overdue at the time of the search, populate the text field in the third row with the number of days, and choose **and not** from the select box in the third row. Select the third radio button, and click **OK**.
- To search for issues that are due in the next certain amount of days, and are overdue at the time of the search, populate the text field in the third row with the number of days, and choose **and** from the select box in the third row. Select the third radio button, and click **OK**.
- The fourth row of the popup is used for arbitrary period searches. Use the **to** text field to specify the upper bound of the search, and the **from** text field to specify the lower bound of the search. A blank text field means no bound. Populating the text fields in the fourth row actually has the same effect as populating the Due Date From and Due Date To text boxes. The syntax is described below.

Relative Period Search Syntax

The Due Date From and Due Date To fields use a special syntax to denote time period bounds. The syntax uses numbers and abbreviations that follow the numbers to represent what the numbers actually mean. The abbreviations are "w" for weeks, "d" for days, "h" for hours, and "m" for minutes. For example, to specify 10 days in the future, use "10d" or "1w and 3d". To specify a period bound in the past, prefix the value with the "-" sign. For example, to specify 2 days, 4 hours, and 3 minutes ago, use "-2d 4h 3m".

Using advanced search

You can also use JIRA Query Language (JQL) to search for issues by due date — see Advanced searching, and particularly the documentation on the Due field.

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 this 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.

Logging work on issues

JIRA Core provides the flexibility to set your estimation and tracking statistics differently depending on your team's needs. Time Tracking features projects schedule planning and time expectations management thanks to its reports, which show original and current time estimates for all the issues, and whether they are ahead or behind the original schedule. Teams often need to be able to estimate how long a product will take to deliver in order to have conviction that the work will be completed. You will be able to easily track the tracking time info by just watching the issue time tracking color bars.

Here's how time tracking appears on an issue:

- The Estimated field displays the amount of time originally anticipated to resolve the issue
- The Remaining field displays the amount of time currently anticipated to resolve the issue
- The Logged field displays the amount of time logged working on the issue so far
- Choosing to include sub-tasks displays the aggregated time of an issue and all its sub-tasks
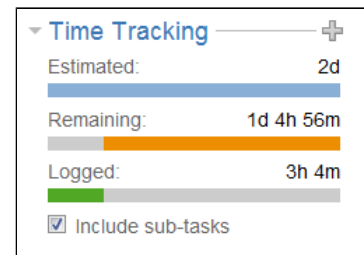
When you log time for the first time, the time spent is subtracted from the original estimate, and the resulting value is automatically presented in the remaining estimate. When subsequent work is logged, any time spent is subtracted from the remaining estimate.

Before you begin

- Make sure your JIRA administrator has enabled the Time Tracking feature.
- Make sure you have the Work on Issues, Delete Work Logs, and Edit Work Logs project permissions.

> Note that anyone with the Browse Project permission can view time tracking information on an issue.

Setting a time estimate for an issue

Teams can set a time estimate for an issue in order to calculate how long it will take to solve the issue.

1. Open the issue and select **Edit**.
2. Scroll down the Edit issue window to fill in the following time tracking fields:

Field	Description
Original Estimate	Amount of time you believe is required to solve the issue. If you want to change original estimate values once they have logged work time, ask your JIRA administrator to disable legacy mode on time tracking.

Remaining Estimate	Amount of time you believe is required to solve the issue in its current state.

If the JIRA time tracking feature is in legacy mode, you will only see the original estimate field if work has not been logged. Once work time has been logged, you will only see the remaining estimate field.

> **Tips:**
> - You can specify additional time units after a time value 'X', such as Xw, Xd, Xh, or Xm, to represent weeks (w), days (d), hours (h), and minutes (m), respectively. If you type a number without specifying a time unit (e.g. if you type '2' instead of '2h'), the default time unit that your JIRA administrator specified will apply.
> - Default conversion rates are 1w = 5d and 1d = 8h.

3. Select **Update**.

When work is first logged against the issue, the **Time Spent** is subtracted from the **Original Estimate**, and the resulting value is automatically presented in the **Remaining Estimate**. When subsequent work is logged, any **Time Spent** is subtracted from the **Remaining Estimate.**

Additionally, once work has been logged on an issue, various reports based on the time tracking information become available.

Logging work on an issue

Once you have started to work on a specific issue, you can log your work by following these steps:

1. Select the issue you want to log time on.
2. Go to **More > Log Work**.
3. Fill in the following **Log Work** fields, and select **Log**:

Log Work field	Description
Time spent	The amount of time spent on the issue. This is the aggregate amount of time that has been logged against this issue.
Date started	Date and time when you started this unit of work.
Remaining estimated	Amount of time anticipated to resolve the issue after completing this unit of work. You can adjust this value using the following options: • **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	Type a description related to the achieved work.
	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: • 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 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 issue that you are looking for? This page will show you how to search for issues in JIRA. Any user can search for issues, although they will only see issue results from projects where they can view issues (i.e. 'Browse Project' permission).

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	The quick search is the fastest way to define search criteria. However, it is less precise than othe complex queries (e.g. `project = JIRA AND status = Open AND priority = High`). It your search criteria is not complex, for example, you know the project key and some key words fo
	To use the quick search: Enter your search criteria in the search box in the header and press E *Tip: If you know the issue key or project key, enter it before other search terms, e.g. "JRA help lir*

Basic search	The basic search is more precise than the quick search, but easier to use than the advanced sea[...] user-friendly interface that lets you define complex queries, without needing to know how to use J[...] searching).

To use the basic search: Navigate to **Issues** (in header) > **Search for issues**, then enter your s[...]
*Tip: If the advanced search is shown instead of the basic search, click **Basic** next to the*
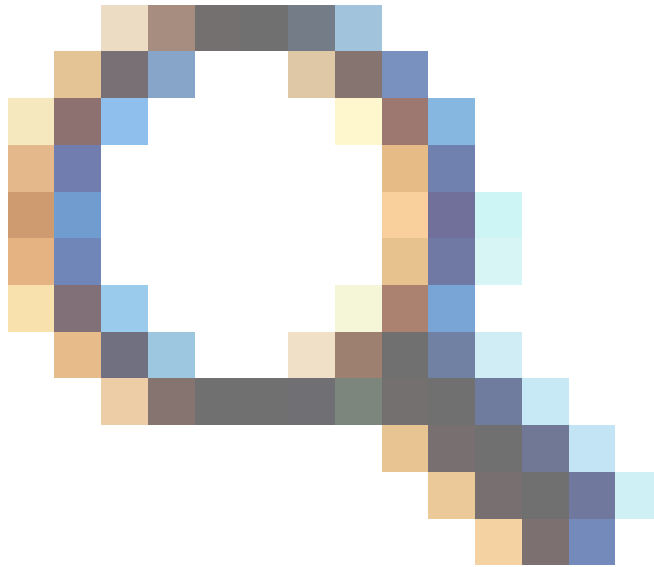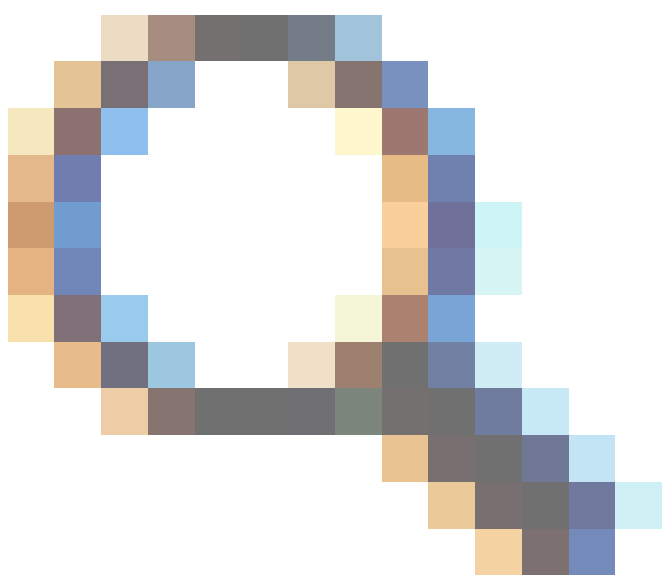

icon.
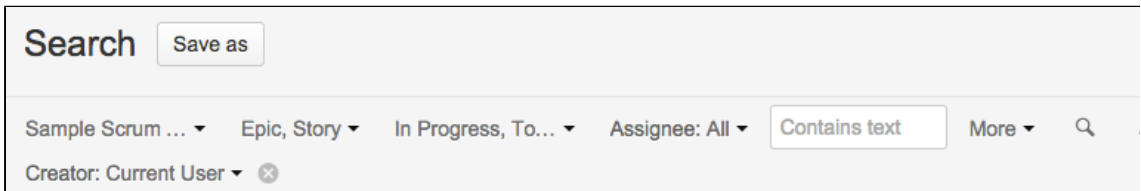
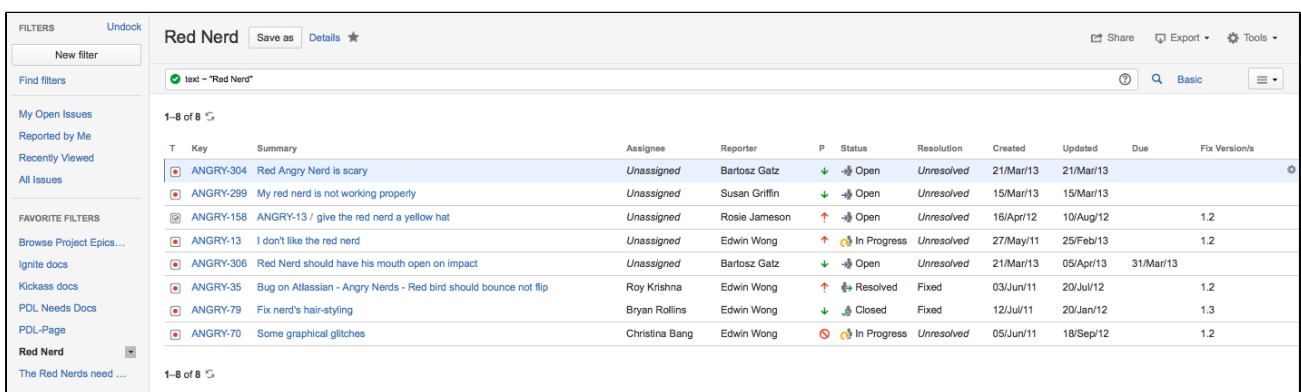| **Advanced search** | The advanced search is the most powerful of the three search methods. You can specify criteria in the other searches (e.g. ORDER BY clause). However, you need to know how to construct struc the JIRA Query Language (JQL) to use this feature. |
| | **To use the advanced search:** 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.

Search   Save as   ↗ Share

✓ project = SSP AND issuetype in (Epic, Story) AND status in ("In Progress", "To Do") AND creator in (currentUser())

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.

FILTERS	Undock	Red Nerd Save as Details ★							↗ Share ⬇ Export ▾ ⚙ Tools ▾
New filter		✓ text ~ "Red Nerd"							② 🔍 Basic ≡ ▾
Find filters		1–8 of 8 ↺							
My Open Issues		T	Key	Summary	Assignee	Reporter	P	Status	Resolution	Created	Updated	Due	Fix Version/s
Reported by Me			ANGRY-304	Red Angry Nerd is scary	Unassigned	Bartosz Gatz	↓ Open	Unresolved	21/Mar/13	21/Mar/13			⚙
Recently Viewed			ANGRY-299	My red nerd is not working properly	Unassigned	Susan Griffin	↓ Open	Unresolved	15/Mar/13	15/Mar/13		
All Issues			ANGRY-158	ANGRY-13 / give the red nerd a yellow hat	Unassigned	Rosie Jameson	↑ Open	Unresolved	16/Apr/12	10/Aug/12		1.2
FAVORITE FILTERS			ANGRY-13	I don't like the red nerd	Unassigned	Edwin Wong	↑ In Progress	Unresolved	27/May/11	25/Feb/13		1.2
Browse Project Epics...			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	
Ignite docs			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
Kickass docs			ANGRY-79	Fix nerd's hair-styling	Bryan Rollins	Edwin Wong	↓ Closed	Fixed	12/Jul/11	20/Jan/12		1.3
PDL Needs Docs			ANGRY-70	Some graphical glitches	Christina Bang	Edwin Wong	⊘ In Progress	Unresolved	05/Jun/11	18/Sep/12		1.2
PDL-Page		1–8 of 8 ↺							
Red Nerd ▾									
The Red Nerds need ...									

3. Working with the search results

You've got the search results displaying the way that you want. Now you can work with the actual issues in the search results. The issue navigator lets you action individual issues, as well as the entire set of issues returned by your search.

Individual issues:

- **View the issue:** Click the issue key or name.
- **Action individual issues**: Click the cog icon next to the issue row and select an option.

All issues in the search results:

- **Export the search results to different formats, like Excel and XML:** Click **Export** and select the desired format.
- **Share the search results:** Click **Share**, then enter the recipient's details.
- **Create an RSS feed:** Click **Export > RSS (Issues)** or **RSS (Comments)**.
- **Bulk modify issues in search results**: Click **Tools** and select **all <*n*> issue(s)** under **Bulk Change**.

4. Save your search

If you frequently run the same search, you can save the search criteria as a filter. This saves you from having to manually redefine the search criteria every time. JIRA applications also include a number of predefined system filters for common queries, such as 'My Open Issues', 'Reported by Me', 'Recently Viewed', and 'All Issues'.

To save your search as a filter: On the search results page, click **Save as** and enter a name for the filter. Your new filter will be shown in the left panel with your other favorite filters, filters shared with you, and the system filters. To run a filter, just click it.

Next steps

Read the following related topics:

- Quick searching
- Basic searching
- Advanced searching
- Saving your search as a filter
- Working with search results

Basic searching

The basic search provides a user-friendly interface that lets you define complex queries, without needing to know how to use JQL (advanced searching).

- If you don't have complex search criteria, you may want to use quick search instead.
- If you are comfortable with the JIRA Query Language (JQL), you may want to use advanced search instead. This search is more powerful than than the basic search.

On this page:
- Basic searching
- Running a saved search
- Troubleshooting
- Next steps

Screenshot: Basic search

Basic searching

1. Choose **Issues** > **Search for issues**.
   - If there are existing search criteria, click the **New filter** button to reset the search criteria.
   - If the advanced search is shown instead of the basic search, click **Basic** (next to the 🔍 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 special characters and modifiers in your search text, such as wildcards and logical operators. See Search syntax for text fields.
3. The search results will automatically update in the issue navigator, unless your administrator has disabled automatic updates of search results. If so, you will need to click the **Update** button on the field drop-down after every change.

Running a saved search

Saved searches (also known as filters) are shown in the left panel, when using basic search. If the left panel is not showing, hover your mouse over the left side of the screen to display it.

To run a filter, e.g. **My Open Issues**, simply click it. The search criteria for the basic search will be set, and the search results will be displayed.

Note, clicking the **Recently Viewed** filter will switch you to the advanced search, as the basic search cannot represent the ORDER BY clause in this filter.
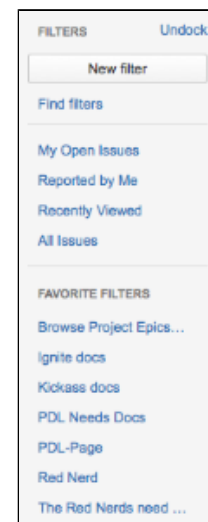
Troubleshooting

˅ Why can't I find the field I want to choose?

Some fields are only valid for a particular *project/issue type context*. For these fields, you must select the applicable project/issue type. Otherwise, the field is not available for selection.

˅ Why are the field criteria displaying in grey text?

Some fields are only valid for a particular *project/issue type context*. If you choose a field in your search, then remove all projects/issue types that reference the field, then the field is invalid. The invalid field does not apply to your search and displays in grey text.

˅ Why is there a red exclamation mark in my field?

Some field values are only valid for a particular *project/issue type context*. For example, you may have configured a project to use a status *In QA Review* in its workflow. If you select this project and status in your search, then change the search to filter for a project that doesn't use *In QA Review*, the status will be invalid and ignored in the search.

˅ Why don't my search results automatically update?

Your search results will always update automatically whenever any fields are changed, provided that your administrator has not disabled automatic updates of search results. Ask your administrator whether they have disabled automatic updates of search results.

Next steps

Read the following related topics:

- Searching for issues
- Advanced searching
- Saving your search as a filter
- Working with search results — find out how to use the issue navigator, export your search results, bulk modify issues, and share your search results.

Quick searching

Sometimes, you just want to be able to get to the particular issue that you 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.

On this page:
- Quick searching
- Understanding quick searching
- Searching issues from your browser's search box
- Next steps

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.

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 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`

`r:`	Find issues reported by you, another user or with no reporter, using the prefix *r:* followed by a specific reporter term, such as *me*, a username or *none*. *Note that there can be no spaces between "r:" and the specific reporter term.*	`r:me` — finds issues reported by you. `r:samuel` — finds issues reported by the user whose username is "samuel". `r:none` — finds issues with no reporter.

`<project name>` or `<project key>`	Find issues in a particular project.	`test project`
		`TST`
		`tst`

`overdue`	Find issues that were due before today.	`overdue`

`created:`
`updated:`
`due:`
	Find issues with a particular Created, Updated, or Due Date using the prefixes *created:*, *updated:*, or *due:*, respectively. For the date range, you can use *today*, *to morrow*, *yesterday*, a single date range (e.g. '-1w'), or two date ranges (e.g. '-1w,1w'). Note that date ranges cannot have spaces in them. Valid date/time abbreviations are: 'w' (week), 'd' (day), 'h' (hour), 'm' (minute).	`created:today`
		`created:yesterday`
		`updated:-1w` — finds issues updated in the last week.
		`due:1w` — finds issues due in the next week.
		`due:-1d,1w` — finds issues due from yesterday to next week.
		`created:-1w,-30m` — finds issues created from one week ago, to 30 minutes ago.
		
		`created:-1d`
		`updated:-4h` — finds issues created in the last day, updated in the last 4 hours.

`<priority>`	Find issues with a particular Priority.	`blocker`
		`major`
		`trivial`

`<issue type>`	Find issues with a particular Issue Type. Note that you can also use plurals.	`bug`
		`task`
		`bugs`
		`tasks`

`<resolution>`	Find issues with a particular Resolution.	`fixed`
		`duplicate`
		`cannot reproduce`

`c:`	Find issues with a particular Component(s). You can search across multiple components. *Note that there can be no spaces between "c:" and the component name.*	`c:security` — finds issues with a component whose name contains the word "security".

`v:`	Find issues with a particular Affects Version(s). To find all issues belonging to a 'major' version, use the wildcard symbol `'*'`. *Note that there can be no spaces between "v:" and the version name.*	`v:3.0` — finds issues that match the following versions (for example): • 3.0 • 3.0 eap • 3.0 beta ...but will not match against the following versions (for example): • 3.0.1 • 3.0.0.4 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.
`ff:`	Find issues with a particular Fix For Version(s). Same usage as `v:` (above).	
`*`	Wildcard symbol `'*'`. Can be used with `v:` and `ff:`.	`v:3.2*` — finds any issue whose version number is (for example): • 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 you wildcards, e.g. `"win*8"`.

For more information on free-text searching, see Search syntax for text fields.

Searching issues from your browser's search box

If you are using Firefox or Internet Explorer 8 (or later), you can add your JIRA instance as a search engine/provider via the drop-down menu next to the browser's search box. Once you add your JIRA instance

as a search engine/provider in your browser, you can use it at any time to conduct a Quick Search for issues in that JIRA instance.

OpenSearch

JIRA supports this browser search feature as part of the autodiscovery part of the OpenSearch stand ard, 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

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 | Perf orming 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

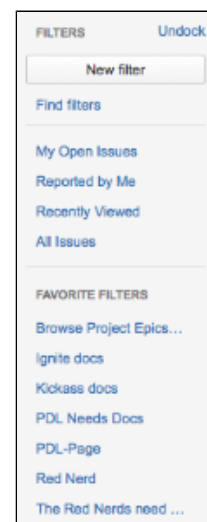| Fields	A field in JQL is a word that represents a JIRA field (or a custom field that has already been defined in JIRA).	Fields reference page ⌄ Show list of fields • Affected version • Assignee • Attachments • Category • Comment • Component • Created • Creator • Custom field • Customer Request Type • Description • Due • Environment • Epic link • Filter • Fix version • Issue key • Labels • 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

Operators	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.	Operators reference page
		⌄ Show list of operators
		  • EQUALS: =
		  • NOT EQUALS: !=
		  • GREATER THAN: >
		  • GREATER THAN EQUALS: >=
		  • LESS THAN: <
		  • LESS THAN EQUALS: <=
		  • IN
		  • NOT IN
		  • CONTAINS: ~
		  • DOES NOT CONTAIN: !~
		  • IS
		  • IS NOT
		  • WAS
		  • WAS IN
		  • WAS NOT IN
		  • WAS NOT
		  • CHANGED

Keywords	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.
		Keywords reference page
		⌄ Show list of keywords
		  • AND
		  • OR
		  • NOT
		  • EMPTY
		  • NULL
		  • ORDER BY

Functions	A function in JQL appears as a word followed by parentheses, which may contain one or more explicit values or JIRA fields. 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.	Functions reference page ⌄ 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 - 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	`affectedVersion`
Field Type	VERSION
Auto-complete	Yes
Supported operators	`= , != , > , >= , < , <=` `IS, IS NOT, IN, NOT IN` *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.*
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: • releasedVersions() • latestReleasedVersion() • unreleasedVersions() • earliestUnreleasedVersion()

List of Fields:
- Affected version
- Assignee
- Attachments
- Category
- Comment
- Component
- Created
- Creator
- Custom field
- Customer Request Type
- Description
- Due
- Environment
- Epic link
- Filter
- Fix version
- Issue key
- Labels
- 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

| Examples | Find issues with an AffectedVersion of 3.14:
`affectedVersion = "3.14"`
Note that full-stops are reserved chara cters and need to be surrounded by quote-marks.
Find issues with an AffectedVersion of "Big Ted":
`affectedVersion = "Big Ted"`
Find issues with an AffectedVersion ID of 10350:
`affectedVersion = 10350` |

^ 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	`= , !=`
	`IS, IS NOT, IN, NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED`
	
	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.

Unsupported operators	`~ , !~ , > , >= , < , <=`
Supported functions	When used with the **IN** and **NOT IN** operators, this field supports: - membersOf() When used with the **EQUALS** and **NOT EQUALS** operators, this field supports: - currentUser()

Examples	- Find issues that are assigned to John Smith:
	`assignee = "John Smith"`
	or
	`assignee = jsmith` - Find issues that are currently assigned, or were previously assigned, to John Smith:
	`assignee WAS "John Smith"`
	or
	`assignee WAS jsmith` - Find issues that are assigned by the user with email address "bob@mycompany.com":
	`assignee = "bob@mycompany.com"`
	
	Note that full-stops and "@" symbols are reserved characters and need to be surrounded by quote-marks.

^ top of page

Attachments

Search for issues that have or do not have attachments.

Syntax	`attachments`
Field Type	ATTACHMENT
Auto-complete	Yes
Supported operators	**IS, IS NOT**
Unsupported operators	**=, != , ~ , !~ , > , >= , < , <= IN, NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED**
Supported functions	None
Examples	- Search for issues that have attachments: `attachments IS NOT EMPTY` - Search for issues that do not have attachments: `attachments IS EMPTY`

^ top of page

Category

Search for issues that belong to projects in a particular category.

Syntax	`category`
Field Type	CATEGORY
Auto-complete	Yes
Supported operators	**=, !=** **IS, IS NOT, IN, NOT IN**
Unsupported operators	**~ , !~ , > , >= , < , <= WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED**
Supported functions	None
Examples	Find issues that belong to projects in the "Alphabet Projects" Category: `category = "Alphabet Projects"`

^ top of page

Comment

Search for issues that have a comment that contains particular text. JIRA text-search syntax can be used.

Syntax	`comment`
Field Type	TEXT
Auto-complete	No
Supported operators	**~ , !~**

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

Supported functions	None
Examples	- Find issues where a comment contains text that matches "My PC is quite old" (i.e. a "fuzzy" match: `comment ~ "My PC is quite old"` - Find issues where a comment contains the exact phrase "My PC is quite old": `comment ~ "\"My PC is quite old\""`

^ top of page

Component

Search for issues that belong to a particular component(s) of a project. You can search by component name or component ID (i.e. the number that JIRA automatically allocates to a component).

Note, it is safer to *search by component ID than by component name*. Different projects may have components with the same name, so searching by component name may return issues from multiple projects. It is also possible for your JIRA administrator to change the name of a component, which could break any saved filters that rely on that name. Component IDs, however, are unique and cannot be changed.

Syntax	`component`
Field Type	COMPONENT
Auto-complete	Yes
Supported operators	`= , !=` `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, component supports: componentsLeadByUser()

Examples	- 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`
Supported functions	When used with the **EQUALS**, **NOT EQUALS**, **GREATER THAN**, **GREATER THAN EQUALS**, **LESS THAN** or **LESS THAN EQUALS** operators, this field supports: • currentLogin() • lastLogin() • now() • startOfDay() • startOfWeek() • startOfMonth() • startOfYear() • endOfDay() • endOfWeek() • endOfMonth() • endOfYear()

Examples	• Find all issues created before 12th December 2010:
	`created < "2010/12/12"`
	• Find all issues created on or before 12th December 2010:
	`created <= "2010/12/13"`
	• Find all issues created on 12th December 2010 before 2:00pm:
	`created > "2010/12/12" and created < "2010/12/12 14:00"`
	• Find issues created less than one day ago:
	`created > "-1d"`
	• Find issues created in January 2011:
	`created > "2011/01/01" and created < "2011/02/01"`
	• Find issues created on 15 January 2011:
	`created > "2011/01/15" and created < "2011/01/16"`

^ top of page

Creator

Search for issues that were created by a particular user. You can search by the user's full name, ID, or email address.

Syntax	`creator`
Field Type	USER

Auto-complete	Yes
Supported operators	`= , !=` `IS , IS NOT , IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN`
Unsupported operators	`~ , !~ , > , >= , < , <=` `CHANGED`
Supported functions	When used with the **IN** and **NOT IN** operators, this field supports: • membersOf() When used with the **EQUALS** and **NOT EQUALS** operators, this field supports: • currentUser()

Examples	• Search for issues that were created by Jill Jones:
	`creator = "Jill Jones"`
	`or`
	`creator = "jjones"`
	• Search for issues that were created by the user with email address "bob@mycompany.com":
	`creator = "bob@mycompany.com"`
	
	(Note that full-stops and "@" symbols are reserved characters, so the email address needs to be surrounded by quote-marks.)

^ 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	*Depends on the custom field's configuration* *Note, JIRA text-search syntax can be used with custom fields of type 'Text'.*
Auto-complete	Yes, for custom fields of type picker, group picker, select, checkbox and radio button fields
Supported operators	*Different types of custom field support different operators.*
Supported operators: number and date fields	`= , != , > , >= , < . <=` `IS , IS NOT , IN , NOT IN`

Unsupported operators: number and date fields	`~ , !~` `WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED`
Supported operators: picker, select, checkbox and radio button fields	`= , !=` `IS , IS NOT , IN , NOT IN`
Unsupported operators: picker, select, checkbox and radio button fields	`~ , !~ , > , >= , < . <=` `WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED`
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	*Different types of custom fields support different functions.*
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: • 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: • releasedVersions() • latestReleasedVersion() • unreleasedVersions() • earliestUnreleasedVersion()

| Examples | 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` |

Customer Request Type

Only applicable if JIRA Service Desk is installed and licensed.

Search for Issues matching a specific Customer Request Type in a service desk project. You can search for a Customer Request Type either by name or description as configured in the Request Type configuration screen.

Syntax	`"Customer Request Type"`
Field Type	Custom field
Auto-complete	Yes

Supported operators	`= , !=`
	
	`IN , NOT IN`

Unsupported operators	`~ , !~ , > , >= , < , <=`
	`IS , IS NOT, WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED`
	
	Note that the Lucene value for Customer Request Type, is `portal-key/request-type-key`. While the portal key cannot be changed after a service desk portal is created, the project key can be changed. The Request Type key cannot be changed once the Request Type is created.

Supported functions	None
Examples	- Find issues where Customer Request Type is **Request a new account** in projects that the user has access to: `"Customer Request Type" = "Request a new account"` - Find issues where the Customer Request Type is **Request a new account** in **SimpleDesk project**, where the right operand is a selected Lucene value from the auto-complete suggestion list. `"Customer Request Type" = "sd/system-access"` - Find issues where Customer Request Type is either **Request a new account** or **Get IT Help**. `"Customer Request Type" IN ("Request a new account", "Get IT Help")`

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	- 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	When used with the **EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS**, **LESS THAN** or **LESS THAN EQUALS** operators, this field supports:
	- currentLogin() - lastLogin() - now() - startOfDay() - startOfWeek() - startOfMonth() - startOfYear() - endOfDay() - endOfWeek() - endOfMonth() - endOfYear()
Examples	- Find all issues due before 31st December 2010: `due < "2010/12/31"` - Find all issues due on or before 31st December 2010: `due <= "2011/01/01"` - Find all issues due tomorrow: `due = "1d"` - Find all issues due in January 2011: `due >= "2011/01/01" and due <= "2011/01/31"` - Find all issues due on 15 January 2011: `due = "2011/01/15"`

^ top of page

Environment

Search for issues where the environment contains particular text. JIRA text-search syntax can be used.

Syntax	`environment`
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	- Find issues where the environment contains text that matches "Third floor" (i.e. a "fuzzy" match): `environment ~ "Third floor"` - Find issues where the environment contains the exact phrase "Third floor": `environment ~ "\"Third floor\""`

^ 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	`"epic link"`
Field Type	Epic Link Relationship
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 **IN** or **NOT IN** operators, `epic link` supports: • issueHistory() • linkedIssues() • votedIssues() • watchedIssues()

Examples	• Find issues that belong to epic "Jupiter", where "Jupiter has the issue key ANERDS-31:
	`"epic link" = ANERDS-31`
	`or`
	`"epic link" = Jupiter`

^ 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	`filter`
Aliases	`request , savedFilter , searchRequest`
Field Type	Filter
Auto-complete	Yes
Supported operators	`= , !=` `IN , NOT IN`
Unsupported operators	`~ , !~ , > , >= , < , <=` `IS , IS NOT, WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED`
Supported functions	None

| Examples | • Search the results of the filter "My Saved Filter" (which has an ID of 12000) for issues assigned to the user jsmith:
`filter = "My Saved Filter" and assignee = jsmith`
`or`
`filter = 12000 and assignee = jsmith` |

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	`fixVersion`
Field Type	VERSION
Auto-complete	Yes
Supported operators	**= , != , > , >= , < , <=** **IS , IS NOT, IN , NOT IN, WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED** *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.*
Unsupported operators	**~ , !~**
Supported functions	When used with the **IN** and **NOT IN** operators, this field supports: • releasedVersions() • latestReleasedVersion() • unreleasedVersions() • earliestUnreleasedVersion()

Examples	• Find issues with a Fix Version of 3.14 or 4.2:
	`fixVersion in ("3.14", "4.2")`
	
	(Note that full-stops are reserved characters, so they need to be surrounded by quote-marks.)
	• Find issues with a Fix Version of "Little Ted":
	`fixVersion = "Little Ted"`
	• Find issues with a Fix Version ID of 10001:
	`fixVersion = 10001`

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	`issueKey`
Aliases	`id , issue , key`

Field Type	ISSUE
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 **IN** or **NOT IN** operators, `issueKey` supports: • issueHistory() • linkedIssues() • votedIssues() • watchedIssues()
Examples	• Find the issue with key "ABC-123": `issueKey = ABC-123`

^ top of page

Labels

Search for issues tagged with a label or list of labels. You can also search for issues without any labels to easily identify which issues need to be tagged so they show up in the relevant sprints, queues or reports.

Syntax	`labels`
Field Type	LABEL
Auto-complete	Yes
Supported operators	`= , !=, IS, IS NOT, IN, NOT IN` *We recommend using IS or IS NOT to search for a single label, and IN or NOT IN to search for a list of labels.*
Unsupported operators	`~ , !~ , , > , >= , < , <=` `WAS, WAS IN, WAS NOT, WAS NOT IN, CHANGED`
Supported functions	None

Examples	• Find issues with an existing label:
	  `labels = "x"`
	• Find issues without a specified label, including issues without a label:
	  `labels not in ("x") or labels is EMPTY`

Last viewed

Search for issues that were last viewed on, before, or after a particular date (or date range). Note that if a time-component is not specified, midnight will be assumed. Please note that the search results will be relative to your configured time zone (which is by default the JIRA server's time zone).

Use one of the following formats:

```
"yyyy/MM/dd HH:mm"
"yyyy-MM-dd HH:mm"
"yyyy/MM/dd"
"yyyy-MM-dd"
```

Or use `"w"` (weeks), `"d"` (days), `"h"` (hours) or `"m"` (minutes) to specify a date relative to the current time.

The default is `"m"` (minutes). Be sure to use quote-marks (`"`); if you omit the quote-marks, the number you supply will be interpreted as milliseconds after epoch (1970-1-1).

Syntax	`lastViewed`
Field Type	DATE
Auto-complete	No
Supported operators	`= , != , > , >= , < , <=` `IS , IS NOT, IN , NOT IN`
Unsupported operators	`~ , !~` `WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED`
Supported functions	When used with the **EQUALS**, **NOT EQUALS**, **GREATER THAN**, **GREATER THAN EQUALS**, **LESS THAN** or **LESS THAN EQUALS** operators, this field supports: • currentLogin() • lastLogin() • now() • startOfDay() • startOfWeek() • startOfMonth() • startOfYear() • endOfDay() • endOfWeek() • endOfMonth() • endOfYear()

Examples	• Find all issues last viewed before 12th December 2010:
	  `lastViewed < "2010/12/12"`
	• Find all issues last viewed on or before 12th December 2010:
	  `lastViewed <= "2010/12/13"`
	• Find all issues last viewed on 12th December 2010 before 2:00pm:
	  `lastViewed > "2010/12/12" and created < "2010/12/12 14:00"`
	• Find issues last viewed less than one day ago:
	  `lastViewed > "-1d"`
	• Find issues last viewed in January 2011:
	  `lastViewed > "2011/01/01" and created < "2011/02/01"`
	• Find issues last viewed on 15 January 2011:
	  `lastViewed > "2011/01/15" and created < "2011/01/16"`

^ 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	`level`
Field Type	SECURITY LEVEL
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	• Search for issues with a security level of "Really High" or "level1":
	`level in ("Really High", level1)`
	• Search for issues with a security level ID of 123:
	`level = 123`

^ 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	`originalEstimate`
Alias	`timeOriginalEstimate`
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	• Find issues with an original estimate of 1 hour:
	`originalEstimate = 1h`
	• Find issues with an original estimate of more than 2 days:
	`originalEstimate > 2d`

^ 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	• Find issues that are sub-tasks of issue TEST-1234: `parent = TEST-1234`

^ 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	• 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, `project` supports: • projectsLeadByUser() • projectsWhereUserHasPermission() • projectsWhereUserHasRole()

Examples	• 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	• 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	When used with the **IN** and **NOT IN** operators, this field supports: • membersOf() When used with the **EQUALS** and **NOT EQUALS** operators, this field supports: • currentUser()

Examples	• 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"`
	
	(Note that full-stops and "@" symbols are reserved characters, so the email address needs to be surrounded by quote-marks.)

^ 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	• 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	When used with the EQUALS, NOT EQUALS, GREATER THAN, GREATER THAN EQUALS, LESS THAN or LESS THAN EQUALS operators, this field supports: • currentLogin() • lastLogin() • now() • startOfDay() • startOfWeek() • startOfMonth() • startOfYear() • endOfDay() • endOfWeek() • endOfMonth() • endOfYear()

Examples	• Find all issues that were resolved before 31st December 2010:
	`resolved <= "2010/12/31"`
	• Find all issues that were resolved before 2.00pm on 31st December 2010:
	`resolved < "2010/12/31 14:00"`
	• Find all issues that were resolved on or before 31st December 2010:
	`resolved <= "2011/01/01"`
	• Find issues that were resolved in January 2011:
	`resolved > "2011/01/01" and resolved < "2011/02/01"`
	• Find issues that were resolved on 15 January 2011:
	`resolved > "2011/01/15" and resolved < "2011/01/16"`
	• Find issues that were resolved in the last hour:
	resolved > -1h

^ top of page

Sprint

Search for issues that are assigned to a particular sprint. This works for active sprints and future sprints. The search is based on either the sprint name or the sprint ID (i.e. the number that JIRA automatically allocates to a sprint).

If you have multiple sprints with similar (or identical) names, you can simply search by using the sprint name — or even just part of it. The possible matches will be shown in the autocomplete drop-down, with the sprint dates shown to help you distinguish between them. (The sprint ID will also be shown, in brackets).

Syntax	`sprint`
Field Type	NUMBER
Auto-complete	Yes
Supported operators	`= , !=` `IS , IS NOT, IN , NOT IN`
Unsupported operators	`~ , !~ , > , >= , < , <=` `WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED`
Supported functions	• openSprints() • closedSprints()

Examples	• Find issues that belong to sprint 999:
	`sprint = 999`
	• Find issues that belong to sprint "February 1":
	`sprint = "February 1"`
	• Find issues that belong to either "February 1", "February 2" or "February 3":
	`sprint in ("February 1","February 2","February 3")`
	• Find issues that are assigned to a sprint:
	sprint is not empty

^ top of page

Status

Search for issues that have a particular status. You can search by status name or status ID (i.e. the number that JIRA automatically allocates to a status).

Note:

- It is safer to search by status ID than status name. It is possible for your JIRA administrator to change the name of a status, which could break any saved filter that rely on that name. Status IDs, however, are unique and cannot be changed.
- The WAS, WAS NOT, WAS IN and WAS NOT IN operators can only be used with the name, not the ID.

Syntax	`status`
Field Type	STATUS
Auto-complete	Yes

Supported operators	`= , !=`
	`IS , IS NOT, IN , NOT IN , WAS, WAS IN, WAS NOT, WAS NOT IN`
	`, CHANGED`

Unsupported operators	`~ , !~ , > , >= , < , <=`
Supported functions	None
Examples	• Find issues with a status of "Open": status = Open • Find issues with a status ID of 1: `status = 1` • Find issues that currently have, or previously had, a status of "Open": status WAS Open

Summary

Search for issues where the summary contains particular text. JIRA text-search syntax can be used.

Syntax	`summary`
Field Type	TEXT
Auto-complete	No
Supported operators	`~ , !~` `IS , IS NOT`
Unsupported operators	`= , != , > , >= , < , <=` `IN , NOT IN , WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED`
Supported functions	None

Examples	• 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\""`

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	• 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\""`

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	• Find issues where the time spent is more than 5 days: `timeSpent > 5d`

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	• Find issues with an issue type of "Bug":
	`type = Bug`
	• Find issues with an issue typeof "Bug" or "Improvement":
	`issueType in (Bug,Improvement)`
	• Find issues with an issue type ID of 2:
	`issueType = 2`

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	`updatedDate`
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: • currentLogin() • lastLogin() • now() • startOfDay() • startOfWeek() • startOfMonth() • startOfYear() • endOfDay() • endOfWeek() • endOfMonth() • endOfYear()

| Examples | 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 > "20011/01/01" 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: - membersOf() When used with the **EQUALS** and **NOT EQUALS** operators, this field supports: - currentUser()
Examples	- 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	• Find all issues that have 12 or more votes: `votes >= 12`

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: • membersOf() When used with the **EQUALS** and **NOT EQUALS** operators, this field supports: • currentUser()

Examples	• Search for issues that you are watching:
	`watcher = currentUser()`
	• Search for issues that the user "jsmith" is watching:
	`watcher = "jsmith"`
	• Search for issues that are being watched by a member of the group "jira-administrators":
	`watcher in membersOf("jira-administrators")`

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: • membersOf() When used with the **EQUALS** and **NOT EQUALS** operators, this field supports: • currentUser()
Examples	• Find all issues that are being watched by more than 3 people: `watchers > 3`

^ top of page

Work ratio

Only available if time-tracking has been enabled by your JIRA administrator.

Search for issues where the work ratio has a particular value. Work ratio is calculated as follows: **workRatio = timeSpent / originalEstimate) x 100**

Syntax	`workRatio`
Field Type	NUMBER
Auto-complete	No
Supported operators	= , != , > , >= , < , <= IS , IS NOT , IN , NOT IN
Unsupported operators	~ , !~ WAS, WAS IN, WAS NOT, WAS NOT IN , CHANGED
Supported functions	None
Examples	• 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.

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.

List of Operators:
- 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

EQUALS: =

The "=" operator is used to search for issues where the value of the specified field exactly matches the specified value. (Note: cannot be used with text fields; see the CONTAINS operator instead.)

To find issues where the value of a specified field exactly matches *multiple* values, use multiple "=" statements with the AND operator.
Examples

- Find all issues that were created by jsmith:

```
reporter = jsmith
```

- Find all issues that were created by John Smith:

```
reporter = "John Smith"
```

^top of page

NOT EQUALS: !=

The "!=" operator is used to search for issues where the value of the specified field does not match the specified value. (Note: cannot be used with text fields; see the DOES NOT MATCH ("!~") operator instead.)

Note that typing `field != value` is the same as typing `NOT field = value`, and that `field != EMPTY` is the same as `field IS_NOT EMPTY`.

The "!=" operator will not match a field that has no value (i.e. a field that is empty). For example, `component`

`!= fred` will only match issues that have a component **and** the component is not "fred". To find issues that have a component other than "fred" **or have no component**, you would need to type: `component != fred or component is empty`.

Examples

- Find all issues that are assigned to any user except jsmith:

```
not assignee = jsmith
```

  or:

```
assignee != jsmith
```

- Find all issues that are not assigned to jsmith:

```
assignee != jsmith or assignee is empty
```

- Find all issues that were reported by me but are not assigned to me:

```
reporter = currentUser() and assignee != currentUser()
```

- Find all issues where the Reporter or Assignee is anyone except John Smith:

```
assignee != "John Smith" or reporter != "John Smith"
```

- Find all issues that are not unassigned:

```
assignee is not empty
```

  or

```
assignee != null
```

^top of page

GREATER THAN: >

The ">" operator is used to search for issues where the value of the specified field is greater than the specified value.

Note that the ">" operator can only be used with fields that support ordering (e.g. date fields and version fields), and cannot be used with text fields. To see a field's supported operators, check the individual field reference.

Examples

- Find all issues with more than 4 votes:

```
votes > 4
```

- Find all overdue issues:

```
duedate < now() and resolution is empty
```

- Find all issues where priority is higher than "Normal":

```
priority > normal
```

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"
```

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
```

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"
```

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 report er = "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")
```

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, `assi gnee 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 JIR A text-search syntax.
Examples

- Find all issues where the Summary contains the word "win" (or simple derivatives of that word, such as "wins"):

```
summary ~ win
```

- 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\""
```

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
```

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
```

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
```

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")
```

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()
```

Advanced searching - functions reference

This page describes information about functions that are used for advanced searching.

A function in JQL appears as a word followed by parentheses, which may contain one or more explicit values or JIRA fields. In a clause, a function is preceded by an operator, which in turn is preceded by a field. A function performs a calculation on either specific JIRA data or the function's content in parentheses, such that only true results are retrieved by the function, and then again by the clause in which the function is used.

> Unless specified in the search query, note that JQL searches do not return empty fields in results. To include empty fields (e.g. unassigned issues) when searching for issues that are not assigned to the current user, you would enter (assignee != currentUser() OR assignee is EMPTY) to include unassigned issues in the list of results.

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()

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	`cascadeOption(parentOption)` `cascadeOption(parentOption,childOption)`
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`

| Examples | Find issues where a custom field ("Location") has the value "USA" for the first tier and "New York" for the second tier:
`location in cascadeOption("USA","New York")`
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:
`location in cascadeOption( "USA" )`
Find issues where a custom field ("Location") has the value "USA" for the first tier and no value for the second tier:
`location in cascadeOption( "USA" ,none)`
Find issues where a custom field ("Location") has no value for the first tier and no value for the second tier:
`location in cascadeOption(none)`
Find issues where a custom field ("Referrer") has the value "none" for the first tier and "none" for the second tier:
`referrer in cascadeOption( "\"none\"" , "\"none\"" )`
Find issues where a custom field ("Referrer") has the value "none" for the first tier and no value for the second tier:
`referrer in cascadeOption( "\"none\"" ,none)` |

^ 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	`closedSprints()`
Supported fields	Sprint
Supported operators	`IN , NOT IN`
Unsupported operators	`= , != , ~ , !~ , > , >= , < , <=` `IS , IS NOT, WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED`
Examples	Find all issues that are assigned to a completed sprint: `sprint in closedSprints()`

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	`componentsLeadByUser()`
	
	`componentsLeadByUser(username)`

Supported fields	Component
Supported operators	`IN , NOT IN`
Unsupported operators	`= , != , ~ , !~ , > , >= , < , <= IS , IS NOT, WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED`

Examples	• Find open issues in components that are led by you:
	`component in componentsLeadByUser() AND status = Open`
	• Find open issues in components that are led by Bill:
	`component in componentsLeadByUser(bill) AND status = Open`

currentLogin()

Perform searches based on the time at which the current user's session began. See also lastLogin.

Syntax	`currentLogin()`
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	• Find issues that have been created during my current session: `created > currentLogin()`

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	`currentUser()`
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	- 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() OR assignee is EMPTY)`

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	- Find issues whose FixVersion is the earliest unreleased version of the ABC project: `fixVersion = earliestUnreleasedVersion(ABC)` - Find issues that relate to the earlist unreleased version of the ABC project: `affectedVersion = earliestUnreleasedVersion(ABC) or fixVersion = earliestUnreleasedVersion(ABC)`

endOfDay()

Perform searches based on the end of the current day. See also endOfWeek, endOfMonth, and endOfYear; and startOfDay, startOfWeek, startOfMonth, and startOfYear.

Syntax	`endOfDay()`
	
	`endOfDay("inc")`
	
	where `inc` is an optional increment of `(+/-)nn(y|M|w|d|h|m)`. If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. endOfDay("+1") is the same as endOfDay("+1d"). If the plus/minus (+/-) sign is omitted, plus is assumed.

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	• Find issues due by the end of today:
	`due < endOfDay()`
	• Find issues due by the end of tomorrow:
	`due < endOfDay("+1")`

endOfMonth()

Perform searches based on the end of the current month. See also endOfDay, endOfWeek, and endOfYear; and startOfDay, startOfWeek, startOfMonth, and startOfYear.

Syntax	`endOfMonth()`
	
	`endOfMonth("inc")`
	
	where `inc` is an optional increment of `(+/-)nn(y|M|w|d|h|m)`. If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. endOfMonth("+1") is the same as endOfMonth("+1M"). If the plus/minus (+/-) sign is omitted, plus is assumed.

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	• Find issues due by the end of this month:
	`due < endOfMonth()`
	• Find issues due by the end of next month:
	`due < endOfMonth("+1")`
	• Find issues due by the 15th of next month:
	`due < endOfMonth("+15d")`

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	`endOfWeek()`
	
	`endOfWeek("inc")`
	
	where `inc` is an optional increment of `(+/-)nn(y|M|w|d|h|m)`. If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. endOfWeek("+1") is the same as endOfWeek("+1w"). If the plus/minus (+/-) sign is omitted, plus is assumed.

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	• Find issues due by the end of this week:
	`due < endOfWeek()`
	• Find issues due by the end of next week:
	`due < endOfWeek("+1")`

endOfYear()

Perform searches based on the end of the current year. See also startOfDay, startOfWeek, and startOfMonth ; and endOfDay, endOfWeek, endOfMonth, and endOfYear.

Syntax	`endOfYear()`
	
	`endOfYear("inc")`
	
	where `inc` is an optional increment of `(+/-)nn(y│M│w│d│h│m)`. *If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. endOfYear("+1") is the same as endOfYear("+1y"). If the plus/minus (+/-) sign is omitted, plus is assumed.*

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	• Find issues due by the end of this year:
	`due < endOfYear()`
	• Find issues due by the end of March next year:
	`due < endOfYear("+3M")`

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 | • Find issues which I have recently viewed, that are assigned to me:
`issue in issueHistory() AND assignee = currentUser()` |

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	`issuesWithRemoteLinksByGlobalId()`
Supported fields	Issue
Supported operators	**IN , NOT IN**
Unsupported operators	**= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN ,** **WAS NOT , WAS NOT IN , CHANGED**

Examples	• Find issues that are linked to remote links that have globalId "abc":
	`issue in issuesWithRemoteLinksByGlobalId(abc)`
	• Find issues that are linked to remote links that have either globalId "abc" or "def":
	`issue in issuesWithRemoteLinksByGlobalId(abc, def)`

lastLogin()

Perform searches based on the time at which the current user's previous session began. See also currentLogin.

Syntax	`lastLogin()`
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	• Find issues that have been created during my last session: `created > lastLogin()`

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	`latestReleasedVersion(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	- 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	- 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	- 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-administators") 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	`now()`
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	Find issues that are overdue: `duedate < now() and status not in (closed, resolved)`

^ 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	`openSprints()`
Supported fields	Sprint
Supported operators	**IN , NOT IN**
Unsupported operators	**= , != , ~ , !~ , > , >= , < , <=** **IS , IS NOT, WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED**
Examples	Find all issues that are assigned to a sprint that has not yet been completed: `sprint in openSprints()`

^ 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	projectsLeadByUser() projectsLeadByUser(username)
Supported fields	Project
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT, WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED
Examples	• Find open issues in projects that are led by you: 　project in projectsLeadByUser() AND status = Open • Find open issues in projects that are led by Bill: 　project in projectsLeadByUser(bill) AND status = Open

^ top of page

projectsWhereUserHasPermission()

Find issues in projects where you have a specific permission. Note, this function operates at the project level. This means that if a permission (e.g. "Edit Issues") is granted to the reporter of issues in a project, then you may see some issues returned where you are not the reporter, and therefore don't have the permission specified. Also note, this function is only available if you are logged in to JIRA.

Syntax	projectsWhereUserHasPermission(permission) For the permission 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	• Find open issues in projects where you have the "Resolve Issues" permission: 　project in projectsWhereUserHasPermission("Resolve Issues") 　AND status = Open

^ top of page

projectsWhereUserHasRole()

Find issues in projects where you have a specific role. Note, this function is only available if you are logged in to JIRA.

Syntax	projectsWhereUserHasRole(rolename)
Supported fields	Project
Supported operators	IN , NOT IN
Unsupported operators	= , != , ~ , !~ , > , >= , < , <= IS , IS NOT, WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED

Examples	• Find open issues in projects where you have the "Developers" role:
	`project in projectsWhereUserHasRole("Developers") AND`
	`status = Open`

releasedVersions()

Perform searches based on the released versions (i.e. versions that your JIRA administrator has released) of a specified project. You can also search on the released versions of all projects, by omitting the *project* parameter. See also latestReleasedVersion().

Syntax	`releasedVersions()`
	
	`releasedVersions(project)`

Supported fields	AffectedVersion, FixVersion, custom fields of type Version
Supported operators	`IN , NOT IN`
Unsupported operators	`= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN ,` `WAS NOT , WAS NOT IN , CHANGED`

Examples	• Find issues whose FixVersion is a released version of the ABC project:
	`fixVersion in releasedVersions(ABC)`
	• Find issues that relate to released versions of the ABC project:
	`(affectedVersion in releasedVersions(ABC)) or (fixVersion in`
	`releasedVersions(ABC))`

standardIssueTypes()

Perform searches based on "standard" Issue Types, that is, search for issues that are not sub-tasks. See also subtaskIssueTypes().

Syntax	`standardIssueTypes()`
Supported fields	Type
Supported operators	`IN , NOT IN`
Unsupported operators	`= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN ,` `WAS NOT , WAS NOT IN , CHANGED`
Examples	• Find issues that are not subtasks (i.e. issues whose Issue Type is a standard issue type, not a subtask issue type): `issuetype in standardIssueTypes()`

startOfDay()

Perform searches based on the start of the current day. See also startOfWeek, startOfMonth, and startOfYear; and endOfDay, endOfWeek, endOfMonth, and endOfYear.

Syntax	`startOfDay()`
	
	`startOfDay("inc")`
	
	where `inc` is an optional increment of `(+/-)nn(y|M|w|d|h|m)`. *If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. startOfDay("+1") is the same as startOfDay("+1d"). If the plus/minus (+/-) sign is omitted, plus is assumed.*

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	• Find new issues created since the start of today:
	`created > startOfDay()`
	• Find new issues created since the start of yesterday:
	`created > startOfDay("-1")`
	• Find new issues created in the last three days:
	`created > startOfDay("-3d")`

^ 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	`startOfMonth()`
	
	`startOfMonth("inc")`
	
	where `inc` is an optional increment of `(+/-)nn(y|M|w|d|h|m)`. *If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. startOfMonth("+1") is the same as startOfMonth("+1M"). If the plus/minus (+/-) sign is omitted, plus is assumed.*

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	• Find new issues created since the start of this month:
	`created > startOfMonth()`
	• Find new issues created since the start of last month:
	`created > startOfMonth("-1")`
	• Find new issues created since the 15th of this month:
	`created > startOfMonth("+14d")`

^ 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	`startOfWeek()`
	
	 `startOfWeek("inc")`
	
	 where *inc* is an optional increment of `(+/-)nn(y|M|w|d|h|m)`. *If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. startOfWeek("+1") is the same as startOfWeek("+1w"). If the plus/minus (+/-) sign is omitted, plus is assumed.*

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	• Find new issues since the start of this week:
	 `created > startOfWeek()`
	 • Find new issues since the start of last week:
	 `created > startOfWeek("-1")`

^ 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	`startOfYear()`
	
	 `startOfYear("inc")`
	
	 where *inc* is an optional increment of `(+/-)nn(y|M|w|d|h|m)`. *If the time unit qualifier is omitted, it defaults to the natural period of the function, e.g. endOfYear("+1") is the same as endOfYear("+1y"). If the plus/minus (+/-) sign is omitted, plus is assumed.*

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	• Find new issues since the start of this year:
	 `created > startOfYear()`
	 • Find new issues since the start of last year:
	 `created > startOfYear("-1")`

^ top of page

subtaskIssueTypes()

Perform searches based on issues that are sub-tasks. See also standardIssueTypes().

Syntax	`subtaskIssueTypes()`
Supported fields	Type
Supported operators	**IN , NOT IN**
Unsupported operators	**= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED**
Examples	• Find issues that are subtasks (i.e. issues whose Issue Type is a subtask issue type): `issuetype in subtaskIssueTypes()`

^ 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	`unreleasedVersions()`
	
	`unreleasedVersions(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	• Find issues whose FixVersion is an unreleased version of the ABC project:
	`fixVersion in unreleasedVersions(ABC)`
	• Find issues that relate to unreleased versions of the ABC project:
	`affectedVersion in unreleasedVersions(ABC)`

^ 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	`votedIssues()`
Supported fields	Issue
Supported operators	**IN , NOT IN**
Unsupported operators	**= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED**

Examples	• Find issues that you have voted for: `issue in votedIssues()`

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	`watchedIssues()`
Supported fields	Issue
Supported operators	**IN , NOT IN**
Unsupported operators	**= , != , ~ , !~ , > , >= , < , <= IS , IS NOT , WAS , WAS IN , WAS NOT , WAS NOT IN , CHANGED**
Examples	• Find issues that you are watching: `issue in watchedIssues()`

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 JRA-6218.

Fuzzy searches: ~

JIRA supports fuzzy searches. To do a fuzzy search, use the tilde, "~", symbol at the end of a single word term. For example, to search for a term similar in spelling to "`roam`", use the fuzzy search:

```
roam~
```

This search will find terms like foam and roams.

Note: Terms found by the fuzzy search will automatically get a boost factor of 0.2.

Proximity searches

JIRA supports finding words that are within a specific distance away. To do a proximity search, use the tilde, "~", symbol at the end of a phrase. For example, to search for "`atlassian`" and "`jira`" within 10 words of each other in a document, use the search:

```
"atlassian jira"~10
```

Boosting a term: ^

JIRA provides the relevance level of matching documents based on the terms found. To boost a term, use the caret, "^", symbol with a boost factor (a number) at the end of the term you are searching. The higher the boost factor, the more relevant the term will be.

Boosting allows you to control the relevance of a document by boosting its term. For example, if you are searching for

```
atlassian jira
```

and you want the term "atlassian" to be more relevant, boost it using the ^ symbol along with the boost factor next to the term. You would type:

```
atlassian^4 jira
```

This will make documents with the term atlassian appear more relevant. You can also boost Phrase Terms, as in the example:

```
"atlassian jira"^4 querying
```

By default, the boost factor is 1. Although, the boost factor must be positive, it can be less than 1 (i.e. 0.2).

Boolean operators

Boolean operators allow terms to be combined through logic operators. JIRA supports AND, "+", OR, NOT and "-" as Boolean operators.

> Boolean operators must be ALL CAPS.

OR | AND | Required term: + | NOT | Excluded term: -

OR

The OR operator is the default conjunction operator. This means that if there is no Boolean operator between two terms, the OR operator is used. The OR operator links two terms, and finds a matching document if either of the terms exist in a document. This is equivalent to a union using sets. The symbol || can be used in place of the word OR.

To search for documents that contain either "atlassian jira" or just "confluence", use the query:

```
"atlassian jira" || confluence
```

or

```
"atlassian jira" OR confluence
```

AND

The AND operator matches documents where both terms exist anywhere in the text of a single document. This is equivalent to an intersection using sets. The symbol && can be used in place of the word AND.

To search for documents that contain "atlassian jira" and "issue tracking", use the query:

```
"atlassian jira" AND "issue tracking"
```

Required term: +

The "+" or required operator requires that the term after the "+" symbol exists somewhere in 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` m ay 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.

ℹ️ **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.

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** ta b 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.

Screenshot: Issue navigator (Detail view)

Changing your view of the search results

List view or Detail view	Click the ⬛ ▾ dropdown to switch between List view and Detail view for your search results. • **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	Click the column name. If you click the same column name more than once, the sort order will switch between ascending and descending. Note: • 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	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:
	
	• **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.
	
	You can also modify any of these configurations. Make sure you have switched the desired configuration, then do the following:
	
	• 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.
	⌄ Why can't I add a column to my column configuration?
	
	  If you cannot find a column, please make sure that you haven't run in to any of the following restrictions:
	
	  • 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	Click the key or summary of the issue. • If you are in List view, you will 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	To action an issue (e.g. edit it, transition it, log work on it, etc): • 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. You can also select an issue and action it via keyboard shortcuts in either views. *Tip: use the 'j' and 'k' keys to select the previous/next issue in the issue navigator.*

Sharing your search results

Click **Share** in the issue navigator to email a link to a search result or shared filter.

- Recipients will receive an email with a link to the search result and the content of the **Note** field (if specified). The subject of the email will state that you (using your username) shared the issue.
- If you share the results of a filter, rather than an ad-hoc search, recipients will receive a link to the filter. Note, if the recipient does not have permission to view the filter, they will receive a link to the search results instead.

Displaying your search results in Confluence

If your JIRA applications are connected to Confluence, you can display your search results on a Confluence page using the JIRA issues macro. For instructions, see JIRA issues macro.

Displaying your search results as a chart

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

The chart will be added to your dashboard. For more information on what each chart shows, see Reporting.

Exporting your search results

Excel	Click **Export** > **Excel (All fields)** or **Export** > **Excel (Current fields)**. • **Excel (All fields)**: this will create a spreadsheet column for every issue field (excluding comments). *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.* • **Excel (Current fields)**: this will create a spreadsheet column for the issue fields that are currently displayed. 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.
Word	Click **Export** > **Word.** 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.
XML	Click **Export** > **XML**. 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. To restrict which issue fields are returned in the XML export, specify the `field` parameter in your URL. For example, to include only the **Issue key** and **Summary,** add `&field=key&field=summary` to the URL. If the `field` parameter is not specified, the XML output will include *all* the issue fields. Otherwise, if one or more `field` 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.

List of fields for field parameter (XML exports):
  ⌄ Show me...

Value	Sample XML output
title	`<title>[TEST-4] This is a test</title>`
link	`<link>https://extranet.atlassian.com:443/jira/brc`
project (or **pid**)	`<project id="10330" key="TST">Test</project>`
description	`<description>This is a detailed description of th`
environment	`<environment>Sydney network<environment>`
key	`<key id="22574">TEST-4</key>`
summary	`<summary>This is a test<summary>`
type (or **issuetype**)	`<type id="3" iconUrl="https://extranet.atlassian.`
parent	`<parent id="22620">TEST-5</parent>`
priority	`<priority id="4"` `iconUrl="https://extranet.atlassian.com:443/jira/`
status	`<status id="5"` `iconUrl="https://extranet.atlassian.com:443/jira/`
resolution	`<resolution id="1">Fixed</resolution>`

labels	
	``` <labels> <label>focus</label> <labels> ```
assignee	``` <assignee username="jsmith">John Smith</assignee> ```
reporter	``` <assignee username="jsmith">John Smith</assignee> ```
security	``` <security id="10021">Private</security> ```
created	``` <created>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/cr ```
updated	``` <updated>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/up ```
resolved (or **resolutiondate**)	``` <resolved>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/r ```
due (or **duedate**)	``` <due>Mon, 1 Sep 2008 17:30:03 -0500 (CDT)>/due> ```
version (or **versions**)	``` <version>2.4.7</version> ```
fixfor (or **fixVersions**)	``` <fixVersion>2.6</fixVersion> ```
component (or **components**)	``` <component>Documentation</component> ```
votes	``` <votes>1</votes> ```

| comments (or comment) | ```xml
<comments>
    <comment id="39270" author="jsmith" created="
familiar</comment>
    <comment id="39273" author="jbrown" created="
too</comment>
</comments>
``` |
| --- | --- |
| attachments (or attachment) | ```xml
<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>
``` |
| timeoriginalestimate | ```xml
<timeoriginalestimate seconds="600">10 minutes</t
``` |
| timeestimate | ```xml
<timeestimate seconds="300">5 minutes</timeestima
``` |
| timespent | ```xml
<timespent seconds="300">5 minutes</timespent>
``` |
| aggregatetimeoriginalestimate | ```xml
<aggregatetimeoriginalestimate seconds="36000">10
``` |
| aggregatetimeestimate | ```xml
<aggregatetimeremainingestimate seconds="18000">5
``` |
| aggregatetimespent | ```xml
<aggregatetimespent seconds="18000">5 hours</aggr
``` |
| timetracking | ```xml
<timeoriginalestimate seconds="600">10 minutes</t
<timeestimate seconds="300">5 minutes</timeestima
<timespent seconds="300">5 minutes</timespent>
<aggregatetimeoriginalestimate seconds="36000">10
<aggregatetimeremainingestimate seconds="18000">5
<aggregatetimespent seconds="18000">5 hours</aggr
``` |

| **issuelinks** | ```
<issuelinks>
    <issuelinktype id="10020">
        <name>Duplicate</name>
        <inwardlinks description="is duplicated b
            <issuelink>
                <issuekey id="22477">INTSYS-1009<
            </issuelink>
        </inwardlinks>
    </issuelinktype>
</issuelinks>
``` |
| **subtasks** (or **subtask**) | ```
<subtasks>
    <subtask id="22623">TEST-8</subtask>
</subtasks>
``` |
| **customfield_xxxxx** | ```
<customfields>
    <customfield id="customfield_10112"
key="com.atlassian.jira.plugin.system.customfield
        <customfieldname>Department</customfieldn
        <customfieldvalues>
            <customfieldvalue>Adminstration</cust
        </customfieldvalues>
    </customfield>
</customfields>
``` |
| **allcustom** | ```
<customfields>
    <customfield id="customfield_10112"
key="com.atlassian.jira.plugin.system.customfield
        <customfieldname>Department</customfieldn
        <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>
``` |

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	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.
	
	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**.
	
	More information:
	
	• 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:
	  • 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 `currentUser()` 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	Click **Export** > **RSS (Issues)** or **Export** > **RSS (Comments)**. The URL of the page that shows can be used in your feed reader. Tips: • 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. `http://mycompany.com/anypage?os_authType=basic`) to show a login dialog when viewing the feed.

Bulk modifying issues in your search results

Bulk operations let you action multiple issues at once. These actions include transitioning issues, deleting issues, moving issues, and watching/unwatching issues.

Click **Tools** > Bulk Change: **all <N> issue(s)** and follow the 'Bulk Operation' wizard.

For more information, see Editing multiple issues at the same time.

Next steps

Read the following related topics:

• Searching for issues
• Constructing cron expressions for a filter subscription

Constructing cron expressions for a filter subscription

This page describes how to construct a cron expression. Cron expressions can be used when creating a subscription to a filter, as described in Working with search results.

A cron expression gives you more control over the frequency, compared to the default schedules. For example, you could define a cron expression to notify you at 8:15 am on the second Friday of every month.

Constructing a cron expression

A cron expression is a string of fields separated by spaces. The following table displays the fields of a cron expression, *in the order that they must be specified (from left to right)*:

	Second	Minute	Hour	Day-of-month	Month	Day-of-week	Year (optional)
Allowed values	`0-59`	`0-59`	`0-23`	`1-31`	`1-12 or JAN-DEC`	`1-7 or SUN-SAT`	`1970-2099`
Allowed special characters	**, - * /**	**, - * /**	**, - * /**	**, - * / ? L W C**	**, - * /**	**, - * / ? L C #**	**, - * /**

Note, cron expressions are not case-sensitive.

Here is an example:

```
0 15 8 ? JAN MON 2014
```

This literally translates to 0 second, 15 minute, 8 hour, any day of the month, January, 2014.

In plain English, this represents 8:15am on every Monday during January of 2014. Note, the ? character means "no particular value". In this example, we've set the Day-of-month to no particular value. We don't need to specify it, as we've specified a Day-of-week value. Read more about special characters in the next section.

More examples of cron expressions are explained in the Examples section at the bottom of this page.

Special characters

Special character	Usage
,	Specifies a list of values. For example, in the **Day-of-week** field, 'MON,WED,FRI' means 'every Monday, Wednesday, and Friday'.
-	Specifies a range of values. For example, in the **Day-of-week** field, 'MON-FRI' means 'every Monday, Tuesday, Wednesday, Thursday and Friday'.
*	Specifies all possible values. For example, in the **Hour** field, '*' means 'every hour of the day'.
/	Specifies increments to the given value. For example, in the **Minute** field, '0/15' means 'every 15 minutes during the hour, starting at minute zero'.
?	Specifies no particular value. This is useful when you need to specify a value for one of the two fields **Day-of-month** or **Day-of-week,** but not the other.
L	Specifies the last possible value; this has different meanings depending on context. In the **Day-of-week** field, 'L' on its own means 'the last day of every week' (i.e. 'every Saturday'), or if used after another value, means 'the last xxx day of the month' (e.g. 'SATL' and '7L' both mean 'the last Saturday of the month). In the **Day-of-month** field, 'L' on its own means 'the last day of the month', or 'LW' means 'the last weekday of the month'.
W	Specifies the weekday (Monday-Friday) nearest the given day of the month. For example, '1W' means 'the nearest weekday to the 1st of the month' (note that if the 1st is a Saturday, the email will be sent on the nearest weekday *within the same month,* 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.

Reporting

JIRA Core provides a range of reports that show statistics for particular people, projects, versions, or information about issues.

The documentation in this section will help you configure and use the reports in JIRA Core.

Search the topics in 'Reporting':

Generating a report

To generate a report:

1. Navigate to the desired project and click **Reports**.
2. Select a report from the list. See the 'Reports' section below for information about each report.

Reports

Chart	Purpose
Average Age Report	Shows the average age of unresolved issues for a project or filter. This helps you see whether your backlog is being kept up to date.

Created vs Resolved Issues Report	Maps created issues versus resolved issues over a period of time. This helps you understand whether your overall backlog is growing or shrinking. ⌄ Notes... • Viewing the chart — Areas in red show periods where more issues were created than resolved. Areas in green show periods where more were resolved than created.
Pie Chart Report	Shows a pie chart of issues for a project or filter grouped by a specified field. This helps you see the breakdown of a set of issues, at a glance. For example, you could create a chart to show issues grouped by Assignee for a particular version in a project (using a filter).
Recently Created Issues Report	Shows the number of issues created over a period of time for a project or filter, and how many were resolved. This helps you understand if your team is keeping up with incoming work. ⌄ Notes... • Viewing the chart — The green portion of the bar shows the created issues that are resolved. The red portion shows created but unresolved issues as yet.
Resolution Time Report	Shows the length of time taken to resolve a set of issues for a project or filter. This helps you identify trends and incidents that you can investigate further.
Single Level Group By Report	Shows issues grouped by a particular field for a filter. This helps you group search results by a field, and see the overall status of each group. For example, you could view the issues in a version of a project, grouped by Assignee.
Time Since Issues Report	For a date field and project or filter, maps the issues against the date that the field was set. This can help you track how many issues were created, updated, etc over a period of time.

Time Tracking Report *	Shows time tracking information on issues for a particular version of a project.
	
	⌄ Notes...
	
	The table in the report shows the issues within the version:
	
	• There are four time tracking fields as follows:
	  • **Original Estimate** - The original estimate of the total amount of time it would take to complete this issue.
	  • **Estimated Time Remaining** - The current estimate of the remaining amount of time it would take to complete this issue.
	  • **Time Spent** - The amount of time spent on the issue. This is the aggregate amount of time that has been logged against this issue.
	  • **Accuracy**- The accuracy of the original estimate compared to the current estimate for the issue. It is the difference between the sum of the **Time Spent** and **Estimated Time Remaining** fields, and the **Original Estimate** field.
	• If sub-tasks are enabled, the *''* column at the right of the field shows the aggregate time tracking information for each 'parent' issue (i.e. the sum of the issue's own values, plus those of its sub-tasks).
	• The last line of the table shows the aggregate time tracking information for the whole version.
	
	The report also includes two bar-graphs (above the table), which represent the aggregate time tracking information for the version:
	
	• The first bar-graph **('Progress')** shows the percentage of completed issues (green) and incomplete issues (orange) in this version:
	
	Progress: 40% [green/orange bar]
	
	• The second bar-graph **('Accuracy'** -blue) shows the accuracy of the original estimates.
	
	The length of the **Accuracy** bar compared to the **Progress** bar indicates whether the issues in this version are ahead of or behind schedule. There are three cases:
	
	1. *The issues are on schedule with the original estimate.*
	The **Accuracy** bar is completely blue and is the same length as the **Progress** bar above it.
	
	Progress: 40% [green/orange bar]
	Accuracy: 0% [blue bar]
	
	2. *The issues are behind the original estimate (i.e. will take longer than originally estimated).*
	The **Progress** graph is longer than the **Accuracy** graph. The blue region represents the original estimated time, and the light-grey region is the amount of time by which issues are behind.
	
	Progress: 42% [green/orange bar]
	Accuracy: -4% [blue bar]
	
	3. *The issues are ahead of the original estimate (i.e. will take less time than originally estimated).*
	The **Accuracy** graph is longer than the **Progress** graph. The blue bar represents the original estimated time, and the light-grey region represents the amount of time by which the original estimates were overestimated.
	
	Progress: 47% [green/orange bar]
	Accuracy: 8% [blue bar]

User Workload Report *	Shows how much work a user has been allocated, and how long it should take. For a specified user, you'll be able to see the number of unresolved issues assigned to the specified user, and the remaining workload, on a per-project basis.

| Version Workload Report * | Shows how much outstanding work there is (per user and per issue) before a given version is complete.

For the specified version, you'll be able to see a list of unresolved issues assigned to each user, each user's workload, and a summary of the total remaining workload for the version. |
| Workload Pie Chart Report * | Shows the relative workload for assignees of all issues for a project or filter. |

* Only available if your JIRA administrator has enabled time tracking.

Reports available in Confluence

If you have connected JIRA to Confluence, you can create the following reports in Confluence.

Chart	Purpose
Change Log	Displays a list of issues from JIRA. This list can be static or dynamic, automatically updating as the status of your issues change in JIRA.
Status Report	The Status Report displays the progress of a JIRA project and fix version in pie charts by status, priority, component, and issue type. The Status Report uses the JIRA Chart macro, and is dynamic.

Other reports

- Additional reports (e.g. Gantt Chart Report, Timesheet Report, JIRA SQL Plugin) are available for download from the Atlassian Marketplace.
- JIRA administrators can also create new reports with the plugin API — see our Tutorial - Creating a JIRA report. If you don't want to build a plugin yourself, Atlassian Experts are available for custom projects.
- Issue filters can be exported to Microsoft Excel, where they can be further manipulated into charts and reports. See Working with search results.

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 your 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.

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 Adding and customizing gadgets.

Managing dashboards and permissions

You can edit, delete, copy, mark favorites, and share your dashboards from the Manage Dashboards page.

1. Select **Dashboards** > **Manage Dashboards**.
2. Choose the dashboard.

Sharing and editing your dashboard

You can edit the details for your dashboard, and restrict or share with other users according to the permissions that are set. In addition, you can see all the dashboards you've created, any public dashboards, and any shared dashboards.

1. Click

   ⚙ ▾

   > **Edit/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 Gad gets 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!
Sprint Burndown Gadget		✔		See the burndown for a given sprint in a handy line chart.
Sprint Health Gadget		✔		Seeing a summary of the issues in a sprint in a handy color-coded bar graph. ⌄ Notes... • 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'.
Version Report		✔		Track the projected release date for a version. This helps you monitor whether the version will release on time, so you can take action if work is falling behind.
Agile Wallboard Gadget		✔		Know how you're tracking with an agile board displayed on your wallboard (or dashboard).
Assigned to Me	✔	✔	✔	Quickly see all the unresolved issues assigned to you.

Average Age Chart	✅	✅	✅	Want to know the average age of unresolved issues? This gadget tells you just that.
				˅ Notes...
				  • 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.

Average Number of Times in Status	✅	✅	✅	Displays the average number of times issues have been in a status.
Average Time in Status	✅	✅	✅	Displays the average number of days issues have spent in status.

Bamboo Charts	✅	✅	✅	Checking out Bamboo plan stats in your dashboard.
				˅ Notes...
				  • 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:
				    The website (container) you have placed this gadget on is unauthorized. Please contact your system administrator to have it approved.
				    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.

Bamboo Plan Summary Chart	✅	✅	✅	Seeing a graphical summary of a Bamboo build plan. ⌄ Notes... • 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: > *The website (container) you have placed this gadget on is unauthorized. Please contact your system administrator to have it approved.* 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.
Bamboo Plans	✅	✅	✅	Seeing a list of all plans on a particular Bamboo server and each plan's current status. ⌄ Notes... • 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: > *The website (container) you have placed this gadget on is unauthorized. Please contact your system administrator to have it approved.* 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.

Clover Coverage	✅	✅	✅	Seeing the Clover coverage of plans from a particular Bamboo server.
				⌄ Notes...
				  • 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:
				    The website (container) you have placed this gadget on is unauthorized. Please contact your system administrator to have it approved.
				    To fix this problem, you will need to configure your Bamboo site to allow JIRA to draw information from it via gadgets on the JIRA dashboard. To do this, your JIRA administrator first needs to define your JIRA site as an OAuth consumer in Bamboo. You will then be required to perform a once-off authentication before your gadget will display correctly.

Created vs. Resolved Chart	✅	✅	✅	Checking your progress by seeing the number of issues created vs number of issues resolved over a given period of time.
				⌄ Notes...
				  • The chart is based on your choice of project or issue filter, and the chart can either be cumulative or not.
				  • An issue is marked as resolved in a period if it has a resolution date in that period.
				  • The resolution date is the last date that the **Resolution** field was set to any non-empty value.

Crucible Charts	✅	✅	✅	Seeing statistical summaries of your code reviews. ⌄ Notes... • 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... • 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... • 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... • 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	✅	✅	✅	Check trends in the average time taken to resolve issues. ⌄ Notes... • 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.

Text	✅	✅	✅	Display your specified HTML text on the dashboard.
				⌄ Notes...
				• 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.
Test Sessions	✅	✅	✅	View a list of test sessions.
Time Since Chart	✅	✅	✅	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.
				⌄ Notes...
				• 'Resolved' here is the system Resolution Date field, which is the last date that the system Resolution field was set to any non-empty value.
				• The report is based on your choice of project or issue filter, and your chosen units of time (ie. hours, days, weeks, months, quarters or years).
Time to First Response	✅	✅	✅	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.