

1. Documentation	3
1.1 Structure 101	3
1.2 Structure User's Guide	3
1.2.1 Basic Concepts	4
1.2.2 JIRA Pages with Structure	5
1.2.2.1 Structure Board	5
1.2.2.2 Structure on the Issue Page	6
1.2.2.3 Structure Gadget	7
1.2.2.3.1 Using Structure Gadget in Confluence	8
1.2.2.3.2 Adding Structure Gadget to Confluence Configuration	10
1.2.2.4 Structure on the Project, Component and Version Pages	10
1.2.2.5 Structure on the Issue Navigator Page	11
1.2.3 Working with the Structure Widget	12
1.2.3.1 Structure Widget Overview	12
1.2.3.2 Navigating Structure	12
1.2.3.3 Structure Toolbar	13
1.2.3.4 Configuring Columns and View	14
1.2.3.5 Fixed Structure View	15
1.2.3.6 Searching and Filtering	16
1.2.3.6.1 Simple and JQL Search	17
1.2.3.6.2 Searching and Filtering Within Structure	18
1.2.3.6.3 Searching Outside Structure	19
1.2.3.7 Special Columns	19
1.2.3.7.1 Summary Column	20
1.2.3.7.2 Progress Column	20
1.2.3.7.3 Type-Priority (TP) Column	22
1.2.3.7.4 Flags Column	22
1.2.3.7.5 Multi-Selection Column	22
1.2.3.7.6 JIRA Actions Column	22
1.2.3.8 Changing Structure	23
1.2.3.8.1 Adding Issues to Structure	23
1.2.3.8.2 Moving Issues within Structure	23
1.2.3.8.3 Removing Issues from Structure	24
1.2.3.8.4 Bulk Operations	24
1.2.3.8.5 Using Drag-And-Drop	25
1.2.3.9 Working with Issues	26
1.2.3.9.1 Creating New Issues	26
1.2.3.9.2 Editing Issues	26
1.2.3.9.3 Using JIRA Actions	26
1.2.3.10 Secondary Issue Panels	27
1.2.3.10.1 JIRA Search Results	28
1.2.3.10.2 Issue Clipboard	28
1.2.3.10.3 Removed Issues	30
1.2.3.10.4 Pinned Issues	30
1.2.3.11 Printing Structure	31
1.2.3.12 Real-time collaboration	31
1.2.4 Using JQL to Select Issues in Structure	32
1.2.5 Managing Structures	32
1.2.5.1 Structure Details	33
1.2.5.2 Creating New Structures	33
1.2.5.3 Editing Structure Details	34
1.2.5.4 Structure Permissions	34
1.2.5.5 Copying a Structure	35
1.2.5.6 Deleting a Structure	35
1.2.6 Synchronization	36
1.2.6.1 Importing Structure	36
1.2.6.2 Exporting Structure	37
1.2.6.3 Installing Synchronizer	38
1.2.6.4 Removing Synchronizer	38
1.2.6.5 Turning Synchronizer On and Off	38
1.2.6.6 Running Resync	39
1.2.6.7 Synchronization and Permissions	40
1.2.6.8 Bundled Synchronizers	40
1.2.6.8.1 Sub-Tasks Synchronizer	40
1.2.6.8.2 Saved Filter Synchronizer	41
1.2.6.8.3 Links Synchronizer	41
1.2.6.8.4 GreenHopper Synchronizer	42
1.2.6.8.5 Status Rollup Synchronizer	44
1.3 Structure Administrator's Guide	46
1.3.1 Installing Structure	47
1.3.2 Setting Up Structure License	47
1.3.2.1 Structure License Parameters	48
1.3.2.2 When Structure is Available for Free	48
1.3.2.3 License Maintenance and Expiration	48
1.3.3 Selecting Structure-Enabled Projects	49

1.3.4 Restricting User Access to Structure	49
1.3.5 Changing Permission to Create New Structures	49
1.3.6 Structure Backup and Restore	50
1.3.6.1 Backing Up Structure	50
1.3.6.2 Restoring Structure from Backup	51
1.3.7 Structure Files Location	51
1.3.8 Disabling Feedback Button	52
1.3.9 Troubleshooting	52
1.3.9.1 Troubleshooting Synchronizers	52
1.3.10 System Requirements	53
1.3.11 Who Has Access to the Structure	53
1.4 Structure Developer's Guide	53
1.4.1 Accessing Structure from Your Plugin	54
1.4.1.1 Structure API Basics	55
1.4.1.2 Controlling Compatibility	56
1.4.1.3 Making Structure Dependency Optional	56
1.4.1.4 API Usage Samples	57
1.4.2 Extending Structure Functionality	58
1.4.2.1 Creating a New Synchronizer	58
1.4.3 Accessing Structure Data Remotely	58
1.4.4 Reference	58
1.4.4.1 Structure Java API Reference	59
1.4.4.1.1 Structure API Versions	59
1.4.4.2 Structure Plugin Module Types	59
1.4.4.2.1 Synchronizer Module	60
1.4.4.3 Structure REST API Reference	60
1.4.4.3.1 Structure Resource - GET	61
1.4.4.3.2 Forest Resource - GET	62
1.4.4.3.3 Forest Resource - POST	62
1.5 Structure FAQ	64
1.5.1 Issues Not Added to a Structure when Using Links Synchronizer or Import	64
1.5.2 Integration with Greenhopper	65
1.5.3 Using Subtasks and Structure	65
1.5.4 Difference from Sub-tasks	65
1.5.5 Why Use Structure Plugin?	65
1.6 Structure Troubleshooting	66
1.6.1 HAR Network Report	66
1.6.2 Performance Snapshot	67

# Documentation

Links to the available documentation collections:

- [Structure 101](#)
- [Structure User's Guide](#)
- [Structure Administrator's Guide](#)
- [Structure Developer's Guide](#)
- [Structure FAQ](#)
- [Structure Troubleshooting](#)

## Structure 101

**Structure 101**

### Installing Structure

1. Install the Plugin
  - a. JIRA 4.3 or later? Go to **Administration | Plugins | Install**, search for Structure and click **Install**.
  - b. Older JIRA Download from [Plugin Exchange](#) (make sure to pick up the correct version for your JIRA!)
2. Get Evaluation License
  - a. Check if you need a license by going to **Administration | Structure | License Details**.
  - b. If needed, click **Get Evaluation License** and install the received license.
  - c. See [Setting Up Structure License](#) for details.
3. Configure Enabled Projects
  - a. Open **Administration | Structure | Enabled Projects** and select at least one project.
  - b. See [Selecting Structure-Enabled Projects](#) for details.

### Getting Started

1. Open **Structure | Manage Structure** menu and click **Create Structure**. Create your own private structure to play with.
2. On the Manage Structure page, click on your structure name to go to the [Structure Board](#).
3. On the Structure Board, click [Search](#), then click **JQL** and type `assignee=currentUser() order by updated`. You can use any other query - see more about [Searching and Filtering](#).
4. When some issues are found by the search, click "More Issues in JIRA" button if it's not switched on, and then drag-and-drop several found issues into the Structure.
5. When you have several issues added to the structure, try to rearrange them to create a hierarchy. Hold **Shift** and drag issues up and down or left and right to place them at a different position in the hierarchy. You can also drag by the drag handle at the left side. See [Using Drag-And-Drop](#) to learn more.
6. Use **keyboard arrows** to navigate structure or **Ctrl+Arrow** to move issues up/down or indent/unindent.
7. Use [Structure Toolbar](#) for other actions like Copy/Paste.
8. Click on an issue's summary or hit "o" to open issue page - there's [structure on that page too](#).

## Structure User's Guide

This section contains information for Structure users.

Contents:

- [Basic Concepts](#)
- [JIRA Pages with Structure](#)
  - [Structure Board](#)
  - [Structure on the Issue Page](#)
  - [Structure Gadget](#)
    - [Using Structure Gadget in Confluence](#)
    - [Adding Structure Gadget to Confluence Configuration](#)
  - [Structure on the Project, Component and Version Pages](#)
  - [Structure on the Issue Navigator Page](#)
- [Working with the Structure Widget](#)
  - [Structure Widget Overview](#)
  - [Navigating Structure](#)

- Structure Toolbar
- Configuring Columns and View
- Fixed Structure View
- Searching and Filtering
  - Simple and JQL Search
  - Searching and Filtering Within Structure
  - Searching Outside Structure
- Special Columns
  - Summary Column
  - Progress Column
  - Type-Priority (TP) Column
  - Flags Column
  - Multi-Selection Column
  - JIRA Actions Column
- Changing Structure
  - Adding Issues to Structure
  - Moving Issues within Structure
  - Removing Issues from Structure
  - Bulk Operations
  - Using Drag-And-Drop
- Working with Issues
  - Creating New Issues
  - Editing Issues
  - Using JIRA Actions
- Secondary Issue Panels
  - JIRA Search Results
  - Issue Clipboard
    - Moving Issues Within A Structure
    - Copying Issues Between Structures
  - Removed Issues
  - Pinned Issues
- Printing Structure
- Real-time collaboration
- Using JQL to Select Issues in Structure
- Managing Structures
  - Structure Details
  - Creating New Structures
  - Editing Structure Details
  - Structure Permissions
  - Copying a Structure
  - Deleting a Structure
- Synchronization
  - Importing Structure
  - Exporting Structure
  - Installing Synchronizer
  - Removing Synchronizer
  - Turning Synchronizer On and Off
  - Running Resync
  - Synchronization and Permissions
  - Bundled Synchronizers
    - Sub-Tasks Synchronizer
    - Saved Filter Synchronizer
    - Links Synchronizer
    - GreenHopper Synchronizer
    - Status Rollup Synchronizer

## Basic Concepts

Structure plugin lets you organize issues into multiple hierarchical lists. To get started, you need to get acquainted with a few basic notions we're using throughout this manual and in the Structure plugin itself.

<b>Structure Plugin</b>	A really cool JIRA plugin that you are reading about.
<b>a structure</b>	A single hierarchical list of issues. By default, there's only one structure called "Global Structure", but you can create as many structures as needed.
<b>a structure contains an issue</b>	A structure is initially empty - it does not contain any issues. Issues can be added to a structure <a href="#">manually</a> , or they can be imported or added automatically by a <a href="#">synchronizer</a> . A single issue may be present in many structures at the same time, or not be present in any.

<b>structure widget</b>	The main tool for working with the structure - a grid with columns that shows the hierarchy and allows to work with it. That's what you saw on our <a href="#">live demo server</a> . Structure widget is present at <a href="#">several places in JIRA</a> .
<b>a sub-issue is under a parent issue</b>	When you place an issue under another issue in the hierarchy, you can say that it's a sub-issue and the other one is the parent issue. A sub-issue may contain sub-issues on its own (and so it's a sub-issue and parent issue at the same time). To make matters worse, an issue that's a sub-issue in one structure may not be a sub-issue in another structure.
<b>children issues</b>	Sometimes we call sub-issues "children issues", but "sub-issues" is the preferred term.
<b>JIRA has sub-tasks</b>	JIRA has a very limited hierarchy based on sub-tasks, and since Structure integrates nicely with the sub-tasks we need to refer to them as, well, "sub-tasks". So "sub-tasks" are JIRA issues of the specific issue types ("sub-task types"), while sub-issues are just any issues that happen to be placed under a parent issue in a structure.

## JIRA Pages with Structure

Issue structure is displayed by the **structure widget**, which is located in several places in JIRA - on a dedicated Structure Board, on the issue page, on the project, component and version pages. The widget displays a scrollable grid with the hierarchical list of issues and lets you work with the structure as well as with every issue.

Most functionality of the structure widget is the same on every page, however there are few specific things that the structure does on an issue page and on project/component/version pages. You can work with the structure on the page that's most convenient for you:

- [Structure Board](#)
- [Structure on the Issue Page](#)
- [Structure Gadget](#)
- [Structure on the Project, Component and Version Pages](#)
- [Structure on the Issue Navigator Page](#)

See also: [Working with the Structure Widget](#)


## Structure Board

Structure Board is a full-screen view with the structure widget and without anything else, so it's good for focusing your work on the structure.



Structure Board is opened when you click on the **Structure** top navigation menu in JIRA.

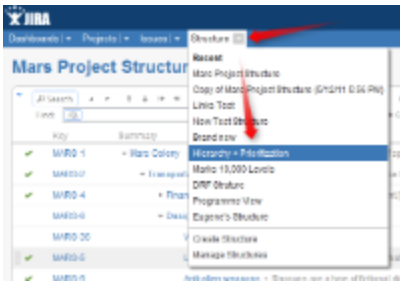


 You can press **g** and then quickly **s** on any JIRA page to open the structure board. (*Go Structure*)

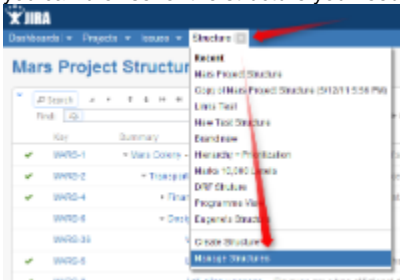
If you have several structures in JIRA, clicking **Structure** top navigation menu opens the most recently used structure.

You can open a specific, not current structure on the structure board using any of the following options:

- Click on the drop-down arrow beside **Structure** top navigation menu and choose one of the recently viewed structures. (Up to 10 structures are shown.)



- Click on the drop-down arrow beside **Structure** and select **Manage Structure** - this will take you to the Manage Structure page where you can browse for the structure you need, and then click on the structure name to open it.

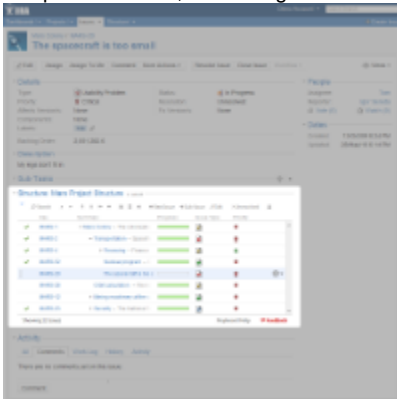


- If you know the ID of the structure you need opened, you can directly open an URL: <http://your.jira.address/secure/StructureBoard.jsps?s=structure-id>

Structure Board will try to accommodate structure widget nicely on your browser page to take advantage of most of the available space.

## Structure on the Issue Page

Structure widget is displayed on an issue page in case the issue belongs to a project [enabled for the Structure Plugin](#). The widget is presented as a separate section, located right above **Activity** section.



Structure section displays the name of the currently selected structure.

When you open an issue page with structure section, the viewed issue is automatically located and selected in the structure.

There are several specific features on the Issue Page that are not present on the Structure Board:

- Collapsible Structure Section
- Current Structure Selector
- Adjusted Time Tracking Section
- Pinned Current Issue Panel
- New Action: Open Structure

### Collapsible Structure Section

The section with the structure can be hidden, as any other section on the page. (*Available on JIRA 4.2 and later*)

Once you hide the structure section, it will remain hidden even if you open another issue page.



The *hidden* flag is stored in a browser cookie.

When the structure section is hidden, the issue hierarchy and the other structure data is not loaded from the server. Everything will be downloaded only when you open the structure section.

By default, structure widget in the Structure section is **fixed on the issue being viewed** - that is, it displays only parent issues and sub-issues of the viewed issue. You can un-fix the structure and view the whole hierarchy by using **Fix** button on the toolbar or a keyboard shortcut.

### Current Structure Selector

By default, structure section displays the current structure (selected previously by the structure selector or by opening a structure board with that structure).

When structure section is not hidden, there's **select** button near the section header and the name of the structure, which allows you to switch to a different structure without leaving the page.



Pressing the **select** button brings up a drop-down dialog with the recent structures and structures that contain the currently viewed issue. The structures in the list are marked in the following way:

- Structures that **contain the currently viewed issue** are marked with the **blue color**.
- **Currently viewed structure** is marked with **bold font**.

When you switch to another structure, the data is automatically reloaded and the selected structure becomes your current structure.

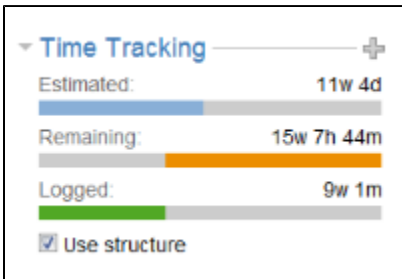
✔ When you click on the **Structure** menu in the top navigation bar, it opens the **Structure Board** with the current structure.

### Adjusted Time Tracking Section

Structure plugin automatically sums up time tracking information from the sub-issues and displays aggregate values in the time tracking section. Whenever any change is detected in the child issues, the time tracking information is refreshed.

You can turn off time tracking aggregation and revert to viewing Time Tracking panel as would JIRA present it without Structure plugin by turning off **Use Structure** check box.

When time tracking section is not present, it means that neither the current issue nor its sub-issues have any time tracking info.



### Pinned Current Issue Panel

When the currently selected structure does not contain the issue which page you are viewing, an automatic **Secondary Panel** is shown in the structure widget with this issue. It is called **Pinned Current Issue** panel.

Pinned Current Issue Panel allows you to use drag-and-drop or copy-paste to easily place the current issue at some position in the current structure. After you have added the current issue to the structure, the panel automatically disappears. You can also hide the panel by clicking on its "pin" icon in the top left corner of the structure widget.

### New Action: Open Structure

A new action is added to the issue view: **More Actions | Open Structure**. Selecting this menu opens **Structure Board** with the current structure, automatically turns on **Search Mode** and enters the issue key of the currently viewed issue as the search term.

As a result, the currently viewed issue is highlighted in the full-screen structure, and if it's not a part of the structure, you can add it there from the **JIRA Search Results** secondary panel.

## Structure Gadget

Structure provides a dashboard gadget that allows you to view and edit structure. The gadget may be also be imported into Confluence and included on a Confluence page.

### Adding Structure Gadget to Dashboard

Structure gadget is added as any other gadget: click **Add Gadget** button in the top right corner of the dashboard, find "Structure" and click **Add It Now**. You need to have change permissions on the dashboard (if you don't have permissions to change the dashboard, you can try to create a copy using **Tools | Copy Dashboard**).

✔ You can add several gadgets showing different structures to the same dashboard.

### Configuring the Gadget

The table below describes available configuration parameters for the Structure gadget.

Parameter	Default Value	Description
Structure		The structure to display. You can pick any structure that you can view.
Visible Rows	10	The maximum number of rows to display. This is how you control the maximum height of the gadget. If there are fewer visible rows, the gadget shrinks; if more, a vertical scroll bar appears. Pick any number between 2 and 50.
Allow Changes	Yes (checked)	If selected, gadget viewers can modify the structure right within the gadget, using keyboard shortcuts. (Of course, only if they have <i>Edit permissions</i> on the structure or issue being changed.) If this option is not selected, the gadget will only show the structure in read-only mode.

The screenshot shows the configuration interface for the Structure gadget. It has a blue header with the title 'Structure'. Below the header, there are three configuration options: 'Structure' with a dropdown menu showing 'Global Structure', 'Visible Rows' with a text input field containing '10', and 'Allow Changes' with a checked checkbox. Each option has a small explanatory text below it. At the bottom of the configuration area, there is a 'Save' button.

✔ Deselect **Allow Changes** to protect the structure from accidental changes, such as changes caused by drag-and-drop or hitting Delete key.

### Using the Gadget

The Structure gadget contains a stripped-down version of the standard Structure widget that you see on other pages. Because the screen space available to a gadget is usually limited, it lacks features like search and secondary panels. It also doesn't have a toolbar. However, most keyboard shortcuts are functional. If the gadget allows editing, you can rearrange issues with drag-and-drop; you can also move, create, edit, and delete issues using the keyboard.

If gadget is displayed in its "home" JIRA dashboard (not in Confluence or elsewhere), the last column lets you use action drop-down for the issues.

### Gadget View Configuration

When you initially configure and add Structure gadget to a dashboard, it has the default state - sub-issues are not shown (the hierarchy is collapsed), default columns are used.

You can [set up columns](#), collapse/expand various parts of the hierarchy and switch structure into the [Fixed Structure View](#).

✔ To switch Structure into Fixed View on a particular target issue, select it and press **Ctrl+**.



## Using Structure Gadget in Confluence

You can embed Structure Gadget in a Confluence page and view or edit structure in Confluence.


**i** Before you can use Structure Gadget on a Confluence page, your Confluence administrator must add Structure Gadget to Confluence Configuration. If you try to insert a macro and don't see *Structure* in the list, most likely the gadget is not configured.

**o** The displayed Structure gadget is not suitable for printing. Support for printable Structure gadget is coming next. So far, please use [Printable Page](#) to print a structure separately.

### How to Add Structure Gadget

1. When editing a page, click Insert/Edit Macro, and select **Structure**. Macro configuration dialog appears.

2. If **Login & approve** button is shown, you need to log in into JIRA first.
3. If **Structure plugin not available** message appears, then you currently don't have any visible structures. Probably you need to login.
4. **Configure gadget** - select the structure to be displayed and other parameters, then click **Save**.
5. Configure gadget appearance, for example, set **width** to **100%** and **border** to **not selected**.
6. Click **Insert** and you're done!

 **Galactic Empire**  
 1 Added by [admin](#), last edited by [admin](#) on Aug 29, 2011 ([view change](#))

## Battle Station Upgrade


Key	Summary	Progress	TP
MARS-89	<b>Battle Station Upgrade</b> - In order to conquer this Galaxy we need to	<div style="width: 100%;"><div style="width: 100%;"></div></div>	
MARS-90	Iteration 1	<div style="width: 100%;"><div style="width: 100%;"></div></div>	
MARS-91	<b>Need Better Catering</b> - We're so tired of the chicken clones!	<div style="width: 100%;"><div style="width: 100%;"></div></div>	
MARS-92	<b>Protect Exhaust Pipes</b> - Theoretically, someone could launch a	<div style="width: 100%;"><div style="width: 100%;"></div></div>	

Showing 8 issues

Galactic empires are a common trope used in science fiction and science fantasy, particularly in space opera. Many authors have either used a galaxy-spanning empire as background, or written about the growth or decline of such an empire. The capital of a galactic empire is frequently a core world or home world.

The best known to the general public today is probably the empire from Star Wars, which was formed in turn from the Galactic Republic.

From [http://en.wikipedia.org/wiki/Galactic\\_empire](http://en.wikipedia.org/wiki/Galactic_empire)




Item #	10188
Ages:	14+
Pieces	3803

## Adding Structure Gadget to Confluence Configuration

Adding JIRA gadgets to Confluence is covered by Atlassian documentation. Here's a list of references to get you started.

- Unless you'd like to see Structure as anonymous user, connect Confluence to JIRA using **Application Links**. You'll need to enable outgoing authentication from Confluence to JIRA.  
 Documentation: [Configuring Application Links](#)
  - Use **OAuth Authentication** to let the Confluence page viewer authenticate separately with JIRA. (Preferred)  
 Documentation: [Configuring OAuth for an Application Link](#)
  - Use **Trusted Applications** authentication if you'd like confluence users act in JIRA under the same usernames without additional authentication.  
 Documentation: [Trusted Application Authentication](#)

 Structure Gadget may allow modification of structure, updating and creating issues under the account that is used by Confluence to access JIRA. Make sure you understand how Trusted Applications work before allowing production structures to be accessed with this kind of authentication. Using OAuth is more secure because the end-user will never be able to do anything that they are not able to do directly in JIRA.

- Add Structure Gadget to the list of **External Gadgets**. Remember that you can copy the URL of the Gadget from the gadgets selection dialog, when you click **Add Gadget** on JIRA dashboard.  
 Documentation: [External Gadgets](#)
- Check on a sample page if you can include Structure macro and get data from JIRA.

Main article: [Adding JIRA Gadgets to a Confluence Page](#)

## Structure on the Project, Component and Version Pages

Structure widget is displayed in a separate tab on a Project page, and on the Component and Version pages as well - if the project is [enabled for Structure](#).

The selected Project (or Component / Version) defines the **scope** of the current structure view - which is used to filter the structure (see details below). For the sake of clarity, we describe the functionality of the structure on the Project page tab; it works in the same way on the Component page tab and Version page tab.

The widget on the project page is the standard fully functional structure widget, but it has several specific features not found on the [Structure Board](#):

- [Current Structure Selector](#)
- [Implicit Scope](#)

### **Current Structure Selector**


By default, structure tab displays the current structure (selected previously by the structure selector or by opening a structure board with that structure). There's a **select** button at the top of the page near the current structure name, which allows you to switch to a different structure

without leaving the page.



Pressing the **select** button brings up a drop-down dialog with the recent structures. The current structure is marked with **bold font** in the list.

When you switch to another structure, the data is automatically reloaded and the selected structure becomes your current structure.

 This works exactly like the Current Structure Selector on the issue page.

### Implicit Scope

When you open the Structure tab, the [search mode](#) is automatically turned on and the implicit search scope is used to filter out issues that don't belong to the current Project (or Component, or Version). This implicit scope is displayed with the *Project: project name* marker on the search panel.



If **More Issues** button is turned on, all issues from this project that are not part of the selected structure are displayed in the [Search Results Secondary Panel](#). This allows to quickly place other issues in the project in the structure.

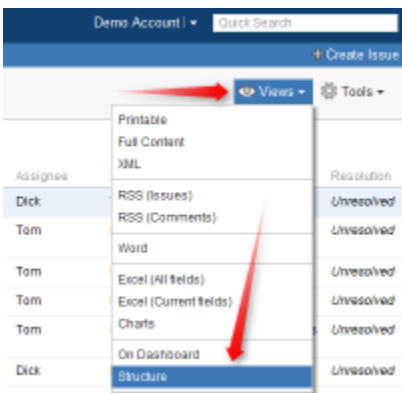


You still can use searching and filtering - but every search condition will be AND-ed with the scope condition. If you turn search mode off, full structure will be displayed.

## Structure on the Issue Navigator Page

hen you use JIRA search to find specific issues, you can open search results on the [Structure Board](#) using Structure's integration with the Issue Navigator page.

To do that, select **Views | Structure** menu, and that will get you to the Structure Board with the current structure, and with [search mode](#) automatically turned on, and the query used on the search page posted as the search term into the structure's search field.



The result is that you see the current structure, with issues from the search result highlighted (or if you have **Filter** turned on, the other issues are filtered out).



If some of the issues from the search result are not in the structure, you can see them if you turn on **More Issues** button.





Structure widget displays only issues from projects that are enabled for Structure. If search result contains issues from other projects, those issues are ignored.

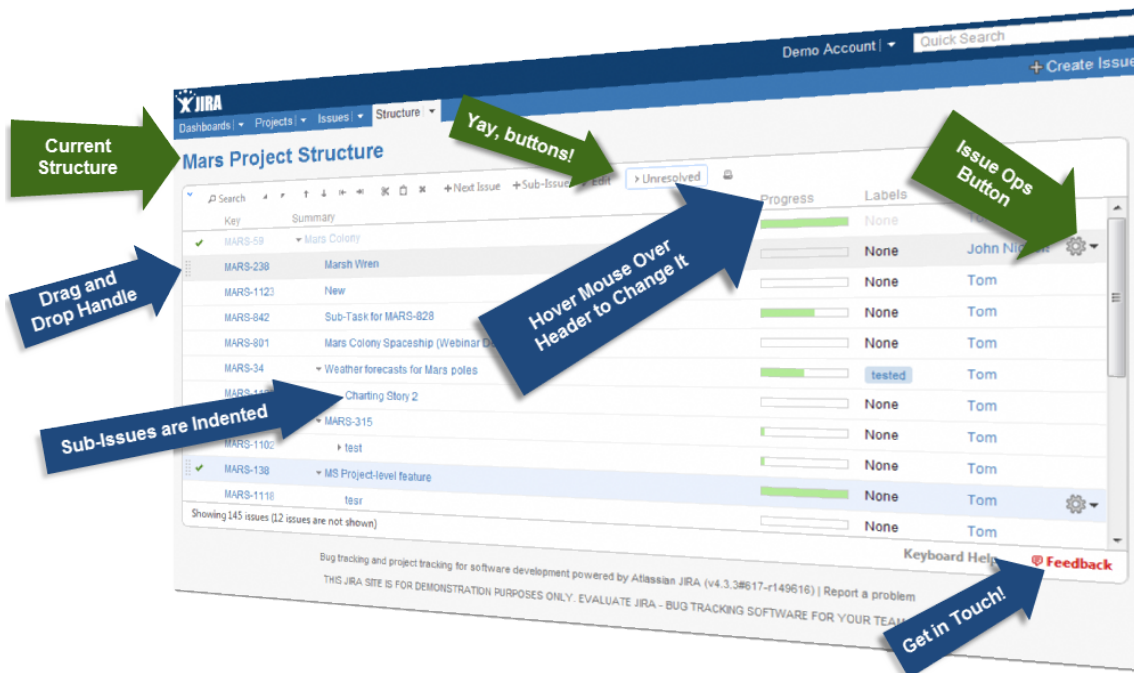
## Working with the Structure Widget

Structure widget is the main tool for working with a structure and issues therein. It's a component of the Structure plugin that is used on the Structure Board, on the Issue Page and in [other places](#) where structure is displayed.

The following sections describe how to use the structure widget in detail.

- Structure Widget Overview
- Navigating Structure
- Structure Toolbar
- Configuring Columns and View
- Fixed Structure View
- Searching and Filtering
- Special Columns
- Changing Structure
- Working with Issues
- Secondary Issue Panels
- Printing Structure
- Real-time collaboration

## Structure Widget Overview



Structure widget is a grid with adjustable columns that display the issues as a hierarchical list. Structure widget is displayed on the [Structure Board](#), [Issue Page](#) and in [other places](#) in JIRA.

Structure lets you navigate the hierarchy and search for specific issues.

Besides showing the Structure and allowing to navigate it, structure widget is the primary tool to change structure by rearranging issues in the hierarchy or using JIRA actions to work with every issue.



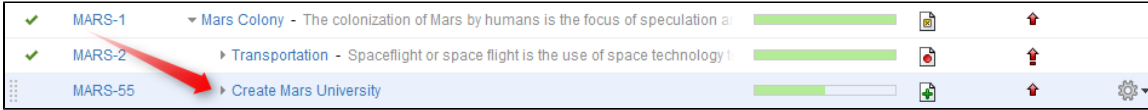
Next: [Navigating Structure](#)

## Navigating Structure

## Navigating with Mouse

You can select issues, scroll up and down, as you would do with any table. Clicking on a link would usually open the link, so if you'd like to just select an issue, click anywhere in the row except on the underlined links. The row is also selected when you click on the JIRA actions icon at the end of the row.

To show sub-issues or hide sub-issues of an issue, click the **Expander** button near the issue summary.



To expand or collapse the whole hierarchy, use **Expand All** or **Collapse All** buttons  on the toolbar.



If there are many issues in the structure, not everything is loaded from the server. As you scroll down or expand sub-issue lists, the data is loaded on demand, which means there might be a delay before the grid is filled with the data for the displayed issues.

## Navigating with Keyboard

You can use **arrow keys** to select next or previous issue in the list. Left and right arrows expand and collapse sub-issue list.

To expand all sub-issues, press the **Plus** keyboard button twice. To collapse all sub-issue lists, press twice the **Minus** keyboard button .

Using **Ctrl+Arrow Key** moves the selected issue up or down in the hierarchy or indents/out-dents it.

You can press **Alt+Down** to open JIRA actions menu for the selected issue.



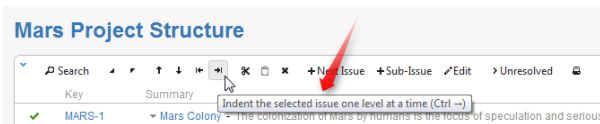
There are a lot more keyboard shortcuts that let you work with the Structure almost without touching the mouse. Press **Ctrl+?** to see the shortcuts cheat sheet or click **Keyboard** at the bottom of the structure widget.

## Structure Toolbar


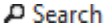



The structure toolbar provides access to the main functions of the structure widget.









Once you move your mouse pointer over the toolbar button, a tooltip with the description of the action is shown.



Below is the table describing the set of actions available through the toolbar.

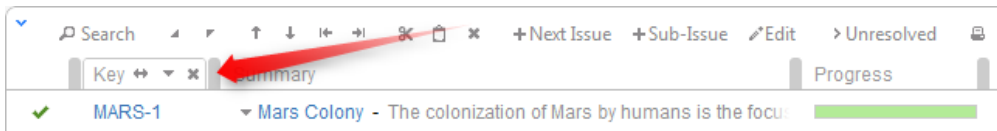
Icon	Action	More Information	Keyboard Shortcut
	Fix / Un-fix Structure on an issue.	<a href="#">Fixed Structure View</a>	<b>Ctrl+.</b>
	Turn Search on and off.	<a href="#">Searching and Filtering</a>	<b>Alt+ /</b>
	Expand/collapse the whole hierarchy.	<a href="#">Navigating Structure</a>	<b>++ / --</b>
	Without changing the issue's parent, move the issue up/down and place it before/after the previous child - if possible.	<a href="#">Moving Issues Within Structure</a>	<b>Ctrl+Up / Ctrl+Down</b>
	Unindent / Indent the issue one level, if possible.	<a href="#">Moving Issues Within Structure</a>	<b>Ctrl+Left / Ctrl+Right</b>

	Cut the selected issues to the <a href="#">Issue Clipboard</a> .	Issue Clipboard	<b>Ctrl+x</b> or <b>Command+x</b>
	Paste the issues from the <a href="#">Issue Clipboard</a> into the structure.	Issue Clipboard	<b>Ctrl+v</b> or <b>Command+v</b>
	Remove the currently selected issue from the structure.	Removing Issues	<b>Delete</b>
<b>+ Next Issue</b>	Create an issue following the currently selected issue on the same level.	Creating New Issues	<b>Enter</b>
<b>+ Sub-Issue</b>	Create a <a href="#">sub-issue</a> under the currently selected issue.	Creating New Issues	<b>Shift+Enter</b> / <b>Insert</b>
 <b>Edit</b>	Edit the current Issue's summary.	Editing Issues	<b>F2</b> / <b>ss</b>
<b>&gt; Unresolved</b>	Filter the structure to show unresolved issues only.	Searching and Filtering	<b>rr</b>
	Open a <a href="#">printable page</a> with the structure.	Printing Structure	

 You can hide/show toolbar clicking the arrow icon in the top left corner of the structure widget.

## Configuring Columns and View

To configure structure columns, position mouse pointer over the structure header for a second to have grid controls appear. These controls let you select which columns to show and how much space each column gets.



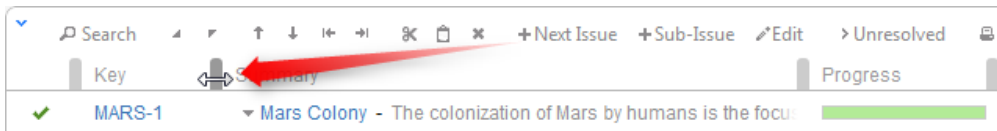
### Configuration Persistence


The configuration of the columns is saved in a browser cookie, so next time you open a page with the structure, you'll have the same configuration as the last time.


However, the configuration for the [Structure Board](#) and [Issue Page](#) are saved separately, so you might need to reconfigure structure widget once on the Structure Board and once on the Issue Page. Likewise, the configuration for the [project](#), [component](#) and [version tabs](#) is saved separately and needs separate configuration.

### Resizing Columns

To give more (or less) space to a column, grab the handle separating one column from another and drag it to the desired position. The adjacent column's width will also be adjusted.



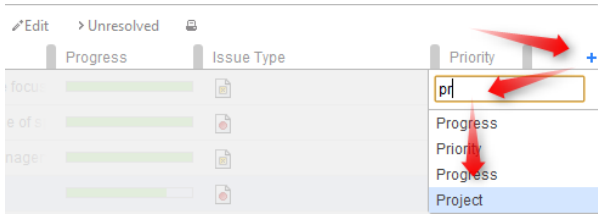
 Structure columns always take 100% of the horizontal space available to the structure widget, no more, no less. Most columns have fixed width, set up by the user and the main [Summary Column](#) takes all the remaining width.

 If the browser window is too narrow or the structure widget has too little horizontal space, the resizing may not work exactly as expected because it becomes problematic to accommodate all the columns with all the data. In that case, consider removing some columns or giving the browser window more width.

### Adding Columns

To add a column, click on the **+** button at the right corner of the table header. A drop-down with available columns appears. To select the desired column, you can:

- use the mouse to find a specific column, or
- use keyboard **arrow keys** to select the column and hit **Enter** when done, or
- start typing column name and get the column list filtered, then use arrow keys if needed and hit Enter when done.



To abort adding new column, hit **Escape**.

- ✓ Use keyboard shortcut **TT** to quickly open Add Column dialog (hit "t", then again "t").

### Removing Columns

To remove a column, click on the **x** mark in the column header.

- ℹ You cannot remove **Summary Column** or service columns.

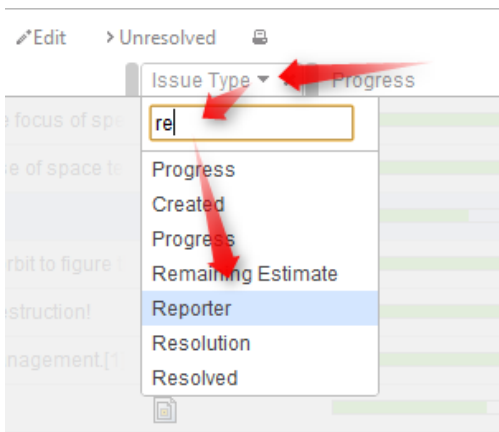
### Rearranging Columns

You can change position of a column by grabbing the column name with the mouse and dragging it to the left or right until the column position hint snaps into the desired location for that column.



### Changing Column Field

You can change the field displayed in a certain column by clicking **down** arrow in the column header and selecting a different field in the same way you select a new column.



The effect is the same as if you had removed this column, added another column and moved it into the same position as the old column.

### Fixed Structure View

You can view only a part of a structure that is related to a specific issue - that is called *fixing structure on an issue*. For example, [Structure Panel on the Issue Page](#) by default is fixed on the issue being displayed by the page.

The issue that the structure widget is fixed on is called **Target Issue**.

Key	Summary
PA-21	Root Issue
PA-22	Sub-Issue 1
PA-24	Sub-Sub-Issue 1.1
PA-25	Sub-Sub-Issue 1.2

Sub-Issue 1 is the target issue on this fixed structure view

### What is Displayed on a Fixed View

When the structure widget is fixed on a target issue, only the following issues are displayed:

- Target issue
- All parent issues of the target issue, up to the top-level issue - those are displayed above the grey line
- All sub-issues of the target issue, down to the deepest level

The issues that are "siblings" or located somewhere else in the hierarchy are not displayed.

✓ The issues that are not displayed when the structure is fixed are not just filtered out, they are not loaded from the server, which provides quicker page load time.

### Turning Fixed View On and Off

You can turn Fixed View on or off by clicking on the Pin button on the toolbar or by using **Ctrl+** keyboard shortcut.

⚠ On the Issue Page, you can only Fix Structure on the issue this page displays - you cannot Fix Structure on any other issue from the list. On the Structure Board, you can select any issue and Fix Structure on it.

### Limitations Imposed by the Fixed View

When you have Structure fixed on a target issue, you can't change the hierarchy from the target issue upwards. That is, you can add/move/delete sub-issues of the target issue, but you can't add issues to the target issue's parents or move target issue somewhere else.

On the screenshot example above, the following actions are available for the displayed issues:

Issue	Relation to the Target Issue	Possible Hierarchy Changes
Root Issue	Parent	None
Sub-Issue 1	Target Issue Itself	Add sub-issues, Delete from structure
Sub-Sub-Issue 1.1	Sub-Issue	Any changes except moving to the top level or away from under Target Issue

✓ Even though you can't move parent issues when the view is fixed, you still can select them, edit or apply JIRA operations.

### When Target Issue Is Missing from Structure

If it happens that the issue the structure is fixed on is missing from structure, the structure widget will not be able to display any data and will ask for your action:



In this case you have to either un-fix the view to see the whole structure, or add the issue at the end of the structure.

## Searching and Filtering



The Search feature provides several important functions:

- Find and highlight issues in your structure,
- Filter your structure so that it only displays specific issues,
- Find issues outside the structure and add them to the structure on the spot.

To access Search function, click on the **Search** button on the Structure Toolbar.



The Search area will appear below the toolbar. By default, the Search runs through the Summary field of Issues.

The search starts once you start entering the query, refining results as you keep typing. The latest search query is saved automatically, so even if you navigate to other pages, the search query will still be there when you return.

**i** If data changes on the server, search results are not automatically refreshed. You can make sure that you are viewing the up-to-date search results by clicking the **Go** button next to the **Find** field.

**✓** You can turn searching on by pressing **Alt+I**. You can cancel search and close the search panel by pressing **Escape**.

The Search function allows you to search through issues in [Simple and JQL](#) modes, [within](#) and [outside](#) the Structure. For more details on this functionality please refer to the corresponding sections.

## Simple and JQL Search

### Simple Search

Simple search mode is turned on when **JQL** button is off. Simple search mode lets you specify the following search conditions:

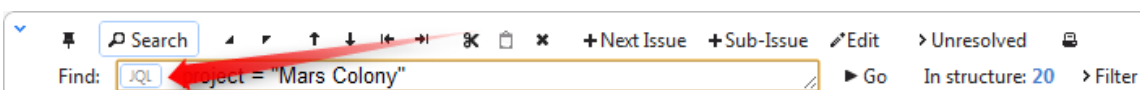
Condition Type	Example	How it works
Simple text	<i>structural hierarchy</i>	Look for issues that have <b>all</b> mentioned words in the <b>Summary</b> field. Each word in the search sentence must be present in the summary, or the summary must have a word that <b>begins with</b> the specified word. The words may come in any order.
Quoted excerpt	<i>"the quick brown fox"</i>	Look for the whole phrase in the summary (but see below about Lucene indexes).
Issue keys	<i>MARS-1, MARS-331</i>	If the text looks like one or more issue keys (delimited by comma or whitespace), search for exactly these issues.
All issues	*	Use single asterisk to search for "all issues". Only issues from the projects enabled for Structure are found.

**i** Structure relies on the JIRA search engine to run text searches. The engine is based on Lucene index which has a few peculiarities that may cause unexpected results. For example, short words may not be found. The result also depends on the Indexing Language specified in the JIRA General Configuration.

### JQL Search

When **JQL** button is on, the search condition is treated as JQL (JIRA Query Language) query. JQL lets you specify arbitrarily complex conditions to find very specific issues.

**✓** You can switch between Simple Search mode and JQL Search mode by clicking **JQL** button or pressing **Alt+j**.



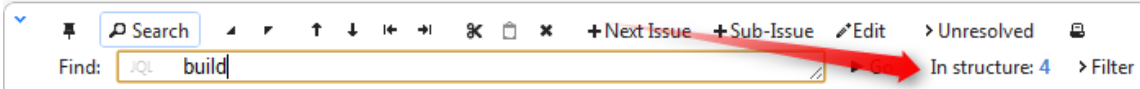
When the JQL mode is on, the usual JQL auto-complete suggests fields, operators and values as you type. Whenever you have a correct JQL in the search field, the JQL button will have blue border. When the JQL is incorrect or not complete, the JQL button will show red border.

More information on JQL is available in the [JIRA documentation](#).

## Searching and Filtering Within Structure

### Searching within the Structure

When the structure widget gets search results from the server, the number of matching issues found in the structure is displayed next to the **Find** field.



Structure widget grays out non-matching issues in the structure in order to highlight the matching issues.



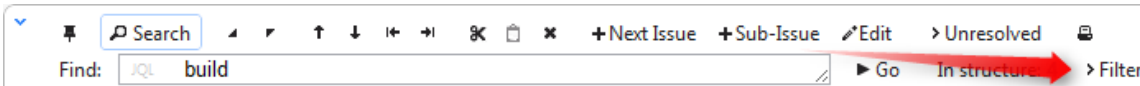
The selection is also moved to the first matching item, and if you press **Down/Up Arrow Keys** while the input focus is still in the search field, the selection will go to the next/previous matching issue in the structure.

- ✓ If you click somewhere else and the input focus is no longer in the search field, you still can navigate to the next or previous matching issue by pressing **]** or **[**.

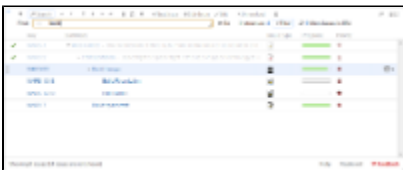
### Filtering Structure

If you wish to see only the issues that match the criteria specified in the search field, click the **Filter** button next to the issue count.

- ✓ You can use keyboard shortcut **Alt+f** to turn filtering on and off.



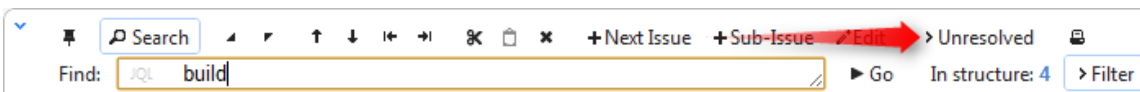
Once Filtering is turned on, you will only see the matching issues and their parent issues. (Parent issues of a matching issue are always shown to preserve the hierarchy view, even if they don't match the search criteria. Non-matching issues are grayed out.)



- ⚠ Filtering mode remains even if you navigate to another page.

### Showing Unresolved Issues Only

Structure toolbar has **Unresolved** button, which works as a shorthand for filtering using JQL: *Resolution is EMPTY*. Clicking **Unresolved** button would filter the structure in the same way Filtering would do.



You can turn on Unresolved button and use search or filtering at the same time.

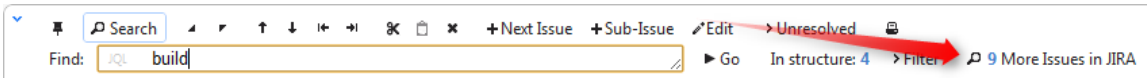


Press **RR** ("r", then quickly "r" once again) to turn Unresolved filter on and off.

## Searching Outside Structure

### Searching for Issues Outside Current Structure

If there are matching issues that do not belong to the structure, an additional **More Issues** button is displayed on the search panel, telling exactly how many matching issues are there in JIRA that are not in the structure.



You can turn **More Issues** mode on or off by clicking this button. When it is turned on, the extra issues are displayed in the **JIRA Search Results** secondary panel.



The search is run through all projects for which the Structure plugin is enabled (see [Selecting Structure-Enabled Projects](#)).



You can turn **More Issues** mode on and off with keyboard shortcut **Alt+m**.



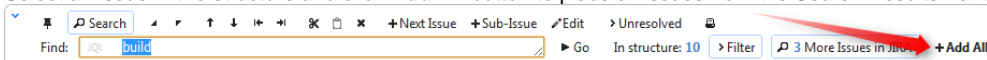
If the search results is large, only the first 1,000 issues are displayed in the Search Results Panel. If you add those issues to the structure, the next 1,000 issues are pulled. Use `ORDER BY JQL` phrase to sort the issues in the result to have the most important issues come first.

### Adding Issues to Structure

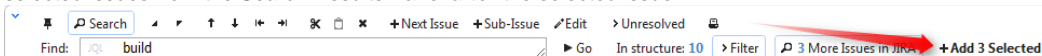
When you have issues found outside Structure with the **More Issues** mode on, you can add them from the Search Results secondary panel to the structure.

There are several options:

- Using **Drag-And-Drop**, move issues from the Search Results Panel to a specific location in the structure.
- Select several issues or all of them using check boxes and use drag-and-drop to move all the selected issues to the structure.
- Select an issue in the structure and click **Add All** button to place all issues from the Search Results Panel **after** the selected issue.



- Select an issue in the structure, then select several issues in the Search Results Panel and click **Add N Selected** button to place the selected issues from the Search Results Panel **after** the selected issue.



- While still having input focus (input caret) in the search criteria field, hit **Ctrl+Enter** to add all (or only selected) issues from the Search Results Panel after the currently selected issue in the structure. (Similar to Add All button.)
- Or hit **Ctrl+Shift+Enter** to add all (or only selected) issues from the Search Results Panel **under** the currently selected issue in the structure - the selected issue becomes the parent of the newly added issues.
- Use Cut/Paste actions and **Issue Clipboard**.



On the **Fixed Structure View**, issues can be added only as sub-issues of the target issue the view is fixed on. The result of using **Ctrl+Enter** or other means of adding issues to structure may be automatically adjusted to place the new issues as sub-issues of the target issue.

## Special Columns

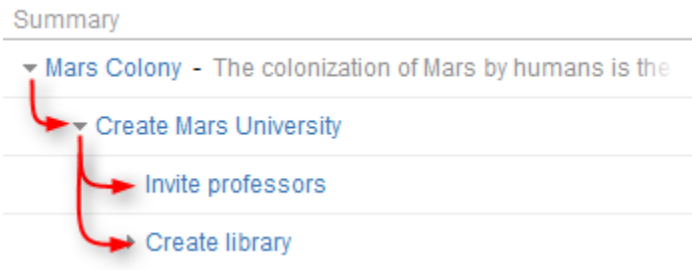
Some columns in the Structure widget are special. They either display structure-specific information or allow you to perform actions with the issues in the structure.

- **Summary Column** — The Summary column displays the issue summary and part of the issue description. Sub-issues have the text in the Summary column indented relative to their parent issue.

- **Progress Column** — The Progress column displays the aggregate issue progress, based on the issue's state, time tracking info and its sub-issues.
- **Type-Priority (TP) Column** — TP column displays icons for issue type and priority.
- **Flags Column** — The flags are the small icons displayed at the left side of the table to mark specific issue states.
- **Multi-Selection Column** — The Multi-Selection column lets you select more than one issue for operations like [drag-and-drop](#).
- **JIRA Actions Column** — The JIRA actions column displays the gear button that calls out the menu with available JIRA actions for the issue.

## Summary Column

The Summary column displays the issue summary and part of the issue description. Sub-issues have the text in the Summary column indented relative to their parent issue.



Summary can be edited right in the structure widget and it's the only field required for creating new issues.

**i** Summary column cannot be removed from the Structure grid because it displays the hierarchy.

## Progress Column

The Progress column displays the aggregate issue progress, based on the issue's state, time tracking info and its sub-issues.

**i** Progress is the custom Structure column, not available in the Issue Navigator or other standard JIRA views.

### How is Progress Calculated?

The progress is calculated based on the issue's Resolution field, time tracking data and the progress of sub-issues to give best estimate of the issue completion progress based on the extrapolation of the available data.

#### Calculating Progress for Issue Without Sub-Issues

If the issue does not have sub-issues:

- If the issue's Resolution field is not empty, the progress is 100%.
- Otherwise, if the issue has time tracking information, the progress is calculated proportionally to this issue completion%:  $(\text{Time Spent}) / (\text{Time Spent} + \text{Remaining Estimate})$
- Otherwise, the progress is 0%.

#### Calculating Progress for Issue with Sub-Issues

If the issue does have sub-issues:

- If the issue's Resolution field is not empty, the progress is 100% - regardless of the sub-issues progress.
- If the issue and its sub-issues do not have estimates or work logged (or if time tracking is turned off), the progress is calculated as the average from the sub-issues progresses.
- If time tracking is used and all issues have an estimate (either original estimate or remaining estimate) - the estimates and total work logged are summed up and the progress is calculated as the total completion%:  $(\text{Total Time Spent}) / (\text{Total Time Spent} + \text{Total Remaining Estimate})$ 
  - If a sub-issue does not have time tracking information, it is counted in as an average sub-issue, based on the mean total time (mean time spent + remaining estimate)

**✓** If the issue has both its own time tracking information and sub-issues with progress, its own progress value is counted as if was the progress of its another sub-issue.

**Examples**

1. Example without time estimates

Summary	Progress
▼ Top issue	<div style="width: 25%; background-color: green;"></div>
Sub-issue 1	<div style="width: 0%; background-color: green;"></div>
▼ Sub-issue 2	<div style="width: 50%; background-color: green;"></div>
✓ Sub-sub-issue 2.1	<div style="width: 100%; background-color: green;"></div>
Sub-sub-issue 2.2	<div style="width: 0%; background-color: green;"></div>

Issue	Explanation	Progress
Sub-sub-issue 2.1	This issue is resolved (indicated by the green mark) - so it is complete	100%
Sub-issue 2	It has two sub-issues with 100% and 0% progress, the total progress is average value	50%
Top issue	It has two sub-issues: sub-issue 1 is 0% done and sub-issue 2 is 50% done, the mean value is 25%.	25%

2. Example with time tracking information

Summary	Progress	Time Spent	Remaining
▼ Top issue	<div style="width: 60%; background-color: green;"></div>		
Sub-issue 1	<div style="width: 75%; background-color: green;"></div>	3 days	1 day
Sub-issue 2	<div style="width: 0%; background-color: green;"></div>		1 day

Issue	Explanation	Progress
Sub-issue 1	It has 3 days of work logged with 1 day remaining, so its progress is $\text{time spent} / \text{total time} = 3 / (3 + 1)$	75%
Sub-issue 2	This issue does not have any work logged, is not resolved and does not have sub-issues	0%
Top issue	The top issue has total time spent of 3 days (work logged on sub-issue 1) and 2 total days remaining (estimates on sub-issue 1 and sub-issue 2), so its progress $3 / (3 + 2)$ .	60%

3. More complex example

Summary	Progress	Time Spent	Remaining
▼ Top issue	<div style="width: 66%; background-color: green;"></div>		
Sub-issue 1	<div style="width: 75%; background-color: green;"></div>	3 days	1 day
▼ Sub-issue 2	<div style="width: 66%; background-color: green;"></div>		1 day
Sub-sub-issue 2.1	<div style="width: 66%; background-color: green;"></div>	2 days	1 day
Sub-sub-issue 2.2	<div style="width: 100%; background-color: green;"></div>	1 day	0 minutes
Sub-issue 3	<div style="width: 0%; background-color: green;"></div>		

Issue	Explanation	Progress
Sub-sub-issue 2.1	It has 2 days of work logged and 1 day remaining, the progress is $2 / (2 + 1)$	66%

Sub-sub-issue 2.2	This issue has 1 day of work logged and no work remaining - so even though it is not resolved, it's considered completed	100%
Sub-issue 2	It has total time spent of 3 days, and total remaining estimate of 2 days (the remaining time from sub-sub-issue 2.1 and its own 1 day, which is considered additional work, besides sub-issues). The progress is $3 / (3 + 2)$ .	60%
Sub-issue 1	This one has 3 days of work logged and 1 day remaining - the progress is $3 / (3 + 1)$	75%
Top issue	The progress of the <i>top issue</i> is calculated as follows. The obvious total time spent is 6 days, total remaining estimate is 3 days (count in all sub-issues on all levels). But there's also <i>sub-issue 3</i> , which does not have estimates or work logged, so it's estimated based on the average among the Top Issue's children issues - <i>sub-issue 1</i> and <i>sub-issue 2</i> : the average between total time of <i>sub-issue 1</i> ( $3 + 1 = 4$ days) and total time of <i>sub-issue 2</i> ( $3 + 2 = 5$ days) is 4.5 days. So <i>sub-issue 3</i> is treated as if it has total time 4.5 days (and given its 0% progress that's 0 days spent and 4.5 days remaining). That yields for the <i>top issue</i> : total time spent is 6 days, total remaining time is 7.5 days, and the progress is $6 / (6 + 7.5)$ , which gives 44% value.	44%

## Type-Priority (TP) Column



TP column displays icons for issue type and priority.

Its narrow width and short name allow to save horizontal space for other columns.

## Flags Column

The flags are the small icons displayed at the left side of the table to mark specific issue states.

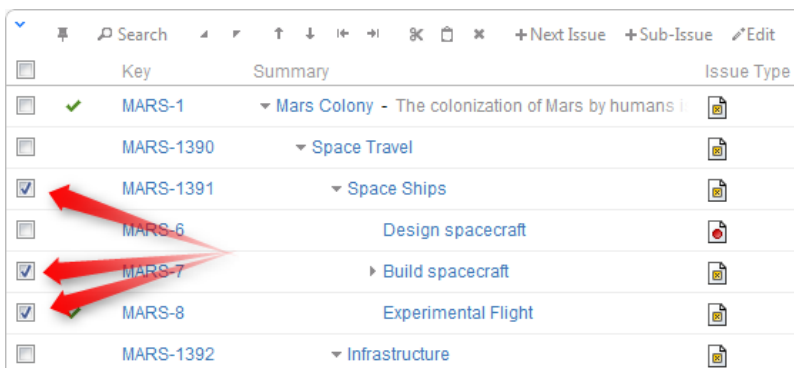
Structure displays the following flags:

	<b>Resolved</b> flag means that the issue's Resolution field is not empty. Such issue is considered completed and filtered out by the <a href="#">Unresolved</a> button.
	<b>Read-only</b> flag means that the current user does not have <b>Edit Issue</b> permission on this issue, so you cannot <a href="#">edit this issue</a> . Additionally, if the structure is configured to <a href="#">require Edit Issue permission on Parent Issue</a> , you cannot change or rearrange the immediate children of this issue.

## Multi-Selection Column

The Multi-Selection column lets you select more than one issue for operations like [drag-and-drop](#).

Multi-Selection column is normally hidden and displayed only when you either hit **Space** key to start multi-selection with the current issue or **Ctrl+A** (**Command+A** on Mac) to select all issues.



<input type="checkbox"/>	Key	Summary	Issue Type
<input type="checkbox"/>	MARS-1	Mars Colony - The colonization of Mars by humans	
<input type="checkbox"/>	MARS-1390	Space Travel	
<input checked="" type="checkbox"/>	MARS-1391	Space Ships	
<input type="checkbox"/>	MARS-6	Design spacecraft	
<input checked="" type="checkbox"/>	MARS-7	Build spacecraft	
<input checked="" type="checkbox"/>	MARS-8	Experimental Flight	
<input type="checkbox"/>	MARS-1392	Infrastructure	

Multi-Selection column is automatically shown on the [Secondary Issue Panels](#).

Multi-Selection column comprises checkboxes for every issue and a select-all check box in the table header. You can select check boxes with the mouse or hit **Space** to toggle the checkbox for the current issue and move selection forward.

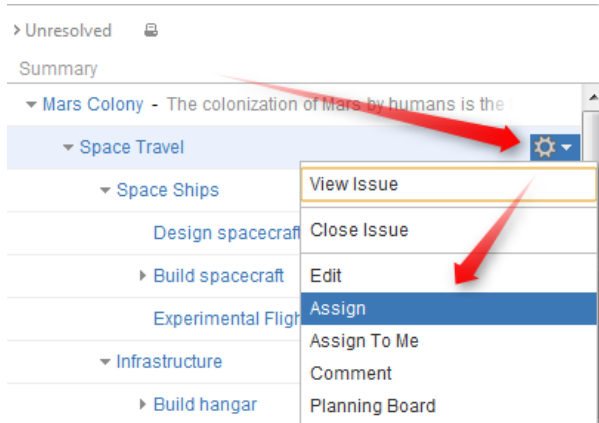


If you need to hide the Multi-Column Selection column in the structure panel, hit **Ctrl+A** to select all issues and then **Ctrl+A** again to clear all check boxes. (Or you can hit **Escape**, but it is shared with cancelling other application modes.)

## JIRA Actions Column

The JIRA actions column displays the gear button that calls out the menu with available JIRA actions for the issue.

This column works like the similar column on the JIRA's Issue Navigator page and lets you log work, apply workflow actions and use other JIRA actions available for the issue.



You can click the gear button and select the desired action with the mouse, or you can use keyboard shortcut **Alt+Down Arrow** to open the menu for the currently selected issue and then use **Up** and **Down** arrow keys and **Enter** key to select the action.

## Changing Structure

There are several basic operations you can do with a structure. They include:

- Adding issues that exist in JIRA to the structure;
- Moving issues within the structure;
- Removing issues from the structure.


There are several ways to make these changes. Some of these operations can be applied to a group of issues and some to individual issues only. See the respective subsections for more details.

 See also: [Creating New Issues](#)

## Adding Issues to Structure

You can add an issue to a structure both from the [Structure Board](#) and from the [Issue Page](#).

On the Structure Board, use [Search](#) to find the desired issues and add them to a structure using [drag-and-drop](#) or [copy/paste](#).

 When searching, make sure the [JIRA Search Results](#) secondary panel is switched on. Use **More Issues** button on the search panel.

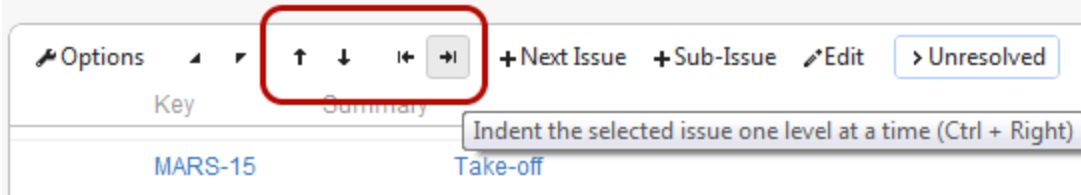
On an Issue Page, if the issue you are viewing is not in the current structure, the Structure section contains [Pinned Current Issue secondary panel](#) and you can drag it to the current structure from there. You can also [select a different structure using Structure Selector](#). Besides, you can use [Search](#) on the issue page as well and add other issues to the structure.

## Moving Issues within Structure

### Basic Moves


There are four basic operations that change structure. All of them are available on the toolbar, and they also can be done from keyboard. Hover mouse pointer over the operation button in the toolbar and a tooltip with the keyboard shortcut will appear.

# Structure



Operation	Keyboard Shortcut	What it does
Move Up	Ctrl + Up	Without changing the issue's parent, moves the issue up and places it before the previous child - if possible.
Move Down	Ctrl + Down	Without changing the issue's parent, moves the issue down and places it after the next child - if possible.
Level Up / Unindent	Ctrl + Left	Makes the issue follow its current parent. The new issue's parent is the previous parent's parent. (Confusing enough? Simply speaking, you're moving the issue one indent level to the left.)
Level Down / Indent	Ctrl + Right	Move the issue to be a sub-issue of its current preceding sibling. You guessed it, it's one indent level to the right.

When you move an issue that has sub-issues, the whole sub-tree is moved.


 When you make changes in the structure, they are uploaded to the server asynchronously, allowing you to continue working regardless of the network delay. You can do a rapid succession of the basic moves, for example, regardless of the time it takes to effect these changes on the server side. There's a **synchronizing** icon in the widget status bar that tells whether there are pending uploads or downloads.

## Moving an Issue to an Arbitrary Position


The basic moves can only adjust issue position one place at a time, so if you need to place an issue at a specific position not close to its current position, you can do that with [Drag-and-Drop](#) or [Cut & Paste](#). Cut & Paste also allows to copy hierarchy from one structure to another.

## Removing Issues from Structure

To remove an issue from the current structure, select this issue and press **Delete** button on the keyboard or click **Delete** button on the toolbar. The issue is removed with all its children issues.

The issue will be removed from the structure and placed into the [Removed Issues](#) secondary panel. You can access it by clicking the **Trash** icon  in the top right corner. To move deleted issues back into the structure, simply select them in the Removed Issues panel and use [drag-and-drop](#) or [cut/paste](#) to place the issues in the structure.

Once you leave the current page, the Removed Issues panel is emptied.

 Removing an issue from a structure does not delete the issue itself. It just removes it from the current structure.

## Bulk Operations

Currently there are two operations you can do in bulk mode, which in many cases do essentially the same thing:

- [Drag-And-Drop](#) lets you move a selection of issues within a structure or add them to a structure from the [secondary issue panels](#), such as Issue Clipboard or Search Results.
- [Cut and Paste](#) allow you to move issues both within a structure and between different structures.

To select multiple issues in a structure, highlight the first issue you want to select and press the **Space** key. A column with check boxes for issues selection will appear. Move around the structure using a keyboard and select other issues you need using the **Space** key or use the mouse to select the desired issues.


 See also: [Multi-Selection Column](#)

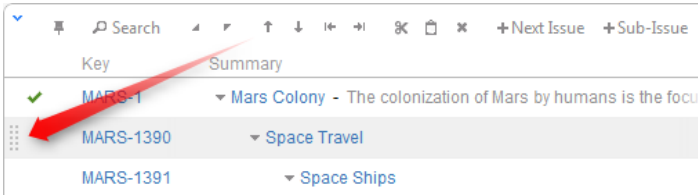


## Using Drag-And-Drop

Drag-and-drop feature allows you to quickly move issues or selections of issues within the structure or add them from the [secondary issue panels](#) to the structure.

### Basic Drag-And-Drop

To grab an issue, move your mouse pointer over the "handle" of the issue (the pointer will change to  when it's over the handle).



Then press and hold down the mouse button and start moving the issue.

- ✓ You can also start dragging by pressing and holding **Shift** on the keyboard and pressing mouse button anywhere on the row with issue (don't click on the links though).

As you move the issue over the grid, the structure will rearrange itself to show the possible positions for the dragged issues. Once the issue is in the correct place, release the mouse button and issue will be moved.

Drag direction	Effect
Up / Down	Moves issues up and down the hierarchy without changing the indentation level, if possible.
Left / Right	Changes indentation level of the moved issues, if possible - without moving them up or down.

- ✓ It may be hard to unindent an issue by using drag-and-drop if you grab it by the drag handle, since the drag handle is usually close to the edge of the screen. Hold **Shift** and drag by some other place on the issue row in that case.

- ✓ After dragging has started you can release **Shift** keyboard button.

### Dragging Multiple Issues

To move multiple issues, first select the desired issues using [Multi-Selection Column](#) (start with hitting **Space** button) and then move them using the "handle" of one of the selected issues. Holding **Shift** and dragging by any other place on the issue row also works.

- ⚠ If you have multiple issues selected, but start dragging an issue that's not included in the multiple selection, only that issue is dragged.

### Cancelling Drag

If you need to cancel drag-and-drop without dropping issues at some random position, hit **Escape** keyboard button.

### Impossible Moves

If it's not possible to move the dragged issues onto the current position (for example, due to structure permission settings), an icon will be displayed at the top left corner of the dragged row.



## Scrolling Structure While Dragging

If you have a large structure, you may need to have Structure grid scrolled up or down while you're dragging issues. Just move the issues over the top or bottom edge of the structure widget and the issues will be scrolled up or down. The further you move the dragged issues, the faster the scrolling is.



Using **Cut & Paste** may be more effective than Drag-and-Drop if you need to move several issues to distant positions.

## Working with Issues

Structure lets you work with issues right in the structure widget.

- [Creating New Issues](#)
- [Editing Issues](#)
- [Using JIRA Actions](#)

## Creating New Issues

Structure plugin lets you create new issues right in the structure. The new issue becomes a "clone" of the selected issue - it has the same Project, Type and most other attributes, but you need to enter the summary for the new issue in the Summary field.

To create a new issue:

1. Select an existing issue in the structure.
2. Do one of the following:
  - a. Either click **New Issue** on the toolbar (you can also hit **Enter** on the keyboard);
  - b. or click **Sub-Issue** on the toolbar (you can also hit **Insert** or **Shift+Enter** on the keyboard).
3. Depending on the selected action, an editor for the new issue will be shown either as the successor of the selected issue or as its first sub-issue.
4. Enter new issue's summary and hit **Enter** to finish editing and create new issue on the server.
  - a. Hit **Escape** to cancel creating new issue while you are still editing the Summary.

After you've provided the summary and pressed **Enter**, the structure widget displays only the Summary field for a short moment as it takes some time to actually create an issue in JIRA. After the widget receives the confirmation from the server that the issue has been created, other columns for that issue are loaded.



While new issue is being uploaded to the server, you can start creating the next issue.

## Editing Issues

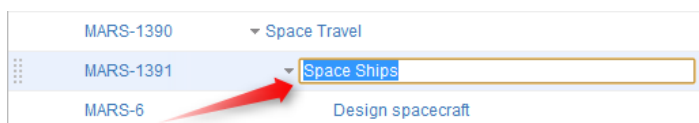
You can edit Summary of an issue in the structure widget.



You need Edit Issue permission on the issue to edit its Summary. If you don't have the permission, a **read-only flag** is shown.

To edit issue summary:

1. Select the issue in the structure widget.
2. Click **Edit** on the toolbar, or use a keyboard shortcut: **F2** or **ss** ("s", then quickly "s" again).
3. Change the Summary and hit **Enter**. To cancel editing, hit **Escape**.



## Using JIRA Actions

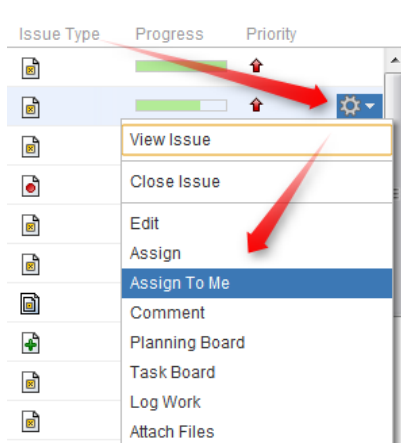
Structure widget lets you use JIRA actions available for the issues from the JIRA's Action & Operations Drop-Down and JIRA keyboard shortcuts for the most frequent actions.

### Using Actions Drop-Down

Structure widget has drop-down menu with actions and operations available for the selected issue - just as the JIRA's Issue Navigator.

To use an action:

1. Click on the **Gear Button** at the right side of the widget in the issue's row, or select the issue with the keyboard and hit **Alt+Down**.
2. Select the action with the mouse or use **Up/Down Arrow** keys and then **Enter** to select the action with the keyboard.



### Using JIRA Shortcuts

Most JIRA shortcuts that are available on the Issue Navigator page also work in the structure widget. Just select an issue and hit the shortcut.

- ✓ The most useful shortcut is "." (dot) - available since JIRA 4.2 - which lets you type in the name of the action you need performed.

Calling an action usually brings up a dialog or moves the browser to another page. Please pay attention to the dialog title or the window title to see that you're applying the action to the correct issue.

- ⊖ On the [Issue Page](#), keyboard shortcuts are always applied to the viewed issue - regardless of the selection in the structure!

## Secondary Issue Panels

Secondary Issue Panels are the auxiliary grids, which can open next to the structure main grid. Even though they serve slightly different purposes, they work very similarly. They display a set of issues, which you can add to the structure grid using the [drag-and-drop](#) or [cut & paste](#).

Structure has the following secondary panels:

- [Issue Clipboard](#) — Issue Clipboard secondary panel allows moving issues within the structure, and, more importantly, between different structures.
- [JIRA Search Results](#) — JIRA Search Results panel is displayed when the [Search](#) is on, there are issues in the search result that are not in the structure, and **More Issue** button is **turned on**.
- [Pinned Issues](#) — The **Pinned Issues** secondary pane is shown in the structure widget on the [Issue Page](#), if the currently selected issue is not present in the currently selected structure.
- [Removed Issues](#) — Removed Issues panel contains the issues that you have just removed from the current structure - just in case you need them back.

### Configuring Secondary Panel View

You can configure the columns displayed on the secondary panel in the same way it's done on the primary structure panel - see [Configuring Columns and View](#). The secondary panel configuration is the same for all secondary panels and is stored like the main panel's configuration.

### Resizing Secondary Panel

You can divide the horizontal space between a secondary panel and the main panel by dragging the separating border.

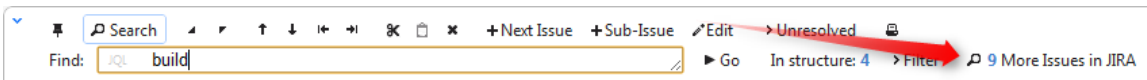
### Secondary Panels Are Read-Only

Although they seem to work in the same way the structure panel works, you cannot edit or create issues on the secondary panel, nor can you

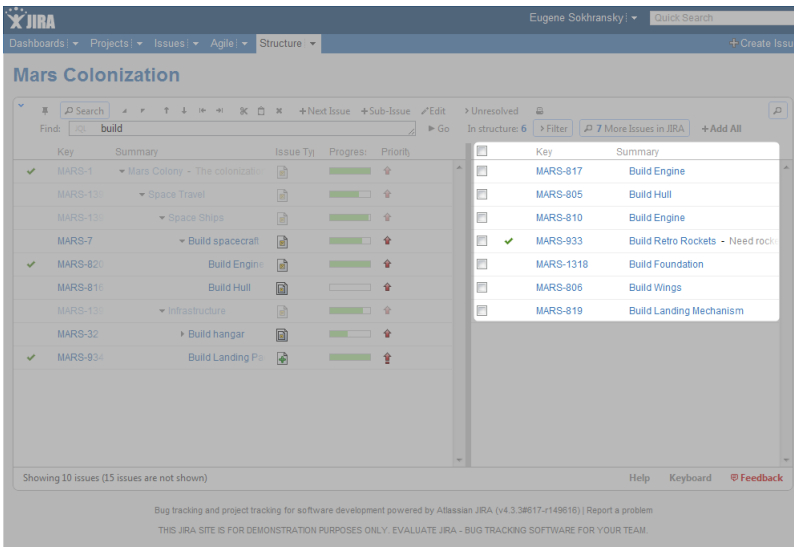
move issues around there.

## JIRA Search Results

JIRA Search Results panel is displayed when the **Search** is on, there are issues in the search result that are not in the structure, and **More Issue** button is **turned on**.



As you start typing your search query, apart from the highlighted/filtered issues belonging to the current structure, you can also see those that match your search criteria, but are not included in the structure. They will be displayed in the JIRA Search Results panel, which opens to the right of the main grid.



Besides moving issues from the secondary panel with drag-and-drop or cut/paste, you can use **Add All** button on the search toolbar or keyboard shortcuts specific to the search field.

**i** See [Searching Outside Structure](#) for the details.

**i** The search is run through all projects for which the Structure plugin is enabled (see [Selecting Structure-Enabled Projects](#)).

## Issue Clipboard

Issue Clipboard secondary panel allows moving issues within the structure, and, more importantly, between different structures.

There are two main scenarios for using the Issue Clipboard:

- **Moving Issues Within A Structure** — Instead of using **drag-and-drop** function to move the issues within a structure, you can use the **cut/paste** feature. This is especially convenient, if you have a large structure and, for example, need to move some issues from the top of the structure to the bottom or the other way around.
- **Copying Issues Between Structures** — The contents of the clipboard is preserved in the **current browser window**, which allows you to copy issues from one structure and paste them into another.

In general, you can select an issue or multiple issues and use the standard **Copy/Cut/Paste** keyboard shortcuts that you use for copying and pasting text.


**i** Please note that if you really have some text selected on the page, **Copy/Cut/Paste** shortcuts would operate on that text - you'll get a copy of the text in the system clipboard, and Structure clipboard will not be affected.

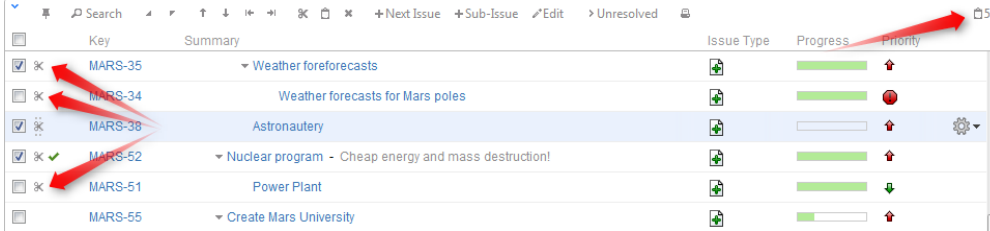
## Moving Issues Within A Structure


Instead of using **drag-and-drop** function to move the issues within a structure, you can use the cut/paste feature. This is especially convenient, if you have a large structure and, for example, need to move some issues from the top of the structure to the bottom or the other way around.


### Cut

First add the desired issues to the clipboard:

1. Select the issues you want to cut. Either select a single issue, or use **multiple selection**.
2. Click the **Cut** button on the structure toolbar (or press **Ctrl+x** or **Command+x**).
3. Selected issues will be put into the clipboard and marked with a small scissors icon ✂.
4. The **Clipboard** icon with the number of the cut issues  will appear in the top right corner of the structure widget.





 Note, that the cut issues are not removed from the structure.


 If the cut issue contains sub-issues, these sub-issues are cut with their parent.

### Paste

After you have cut the issues, you can now paste them back to any place in the structure:

1. If want to see the **Issue Clipboard** panel, you can open it clicking the **Clipboard** icon . However, this is not necessary to use the Cut/Paste function.
2. In the structure select the issue after which the issues from the clipboard should be placed.
3. Either click **Paste** button on the toolbar (or press **Ctrl+v** or **Command+v**) to place the issues **after** the selected issue at the same indentation level, or press **Ctrl+Shift+v** (or **Command+Shift+v** on Mac) to place the issues **under** the selected issue (as the children).

 If the cut issue contains sub-issues, these sub-issues are pasted with their parent.

 After Paste the clipboard is cleared.


### Copying Issues Between Structures

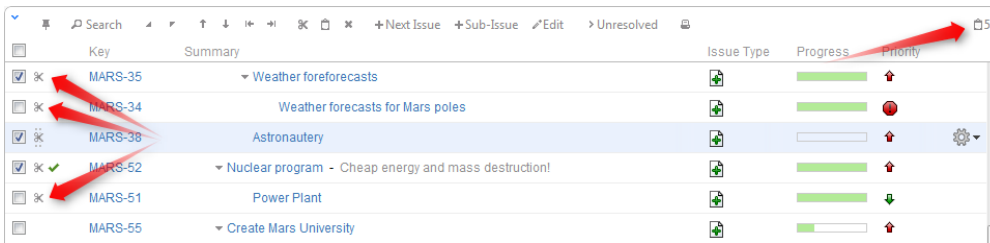
The contents of the clipboard is preserved in the **current browser window**, which allows you to copy issues from one structure and paste them into another.

To copy issues (with their sub-issues) from one structure to another do the following:

### Cut

First add the desired issues to the clipboard:

1. Open the structure to copy from.
2. Select the issues you want to cut. Either select a single issue, or use **multiple select**.
3. Click the **Cut** button on the structure toolbar (or press **Ctrl+x** or **Command+x**).
4. Selected issues will be added to the clipboard and marked with a small scissors icon ✂.
5. The **Clipboard** icon with the number of the cut issues  will appear in the top right corner of the structure widget.



**Note**, that the cut issues are not removed from the structure.

**Note** If the cut issue contains sub-issues, these sub-issues are cut with their parent.

### Paste

After you have cut the issues, you can now paste them to any other structure:

1. **In the same browser window**, switch to a desired structure (you can use Structure Board or [any other JIRA page with Structure](#)).
2. If want to see the **Issue Clipboard** panel, you can open it clicking the **Clipboard** icon . However, this is not necessary to use the Cut/Paste function.
3. In the structure grid select the issue after which the issues from the clipboard should be placed.
4. Either click **Paste** button on the toolbar (or press **Ctrl+v** or **Command+v**) to place the issues **after** the selected issue at the same indentation level, or press **Ctrl+Shift+v** (or **Command+Shift+v** on Mac) to place the issues **under** the selected issue (as the children).

When you paste issue hierarchy from a different structure, it's possible that the target structure already contains some of the issues. In this case, iterative **Merge** is performed, where issue is either moved to a new position if it is present in the structure or added if it is not present.

**Note** If you need to copy the same set of issues to several different structures, you can use drag-and-drop operation to move the issues from the secondary panel to the main grid instead of the Paste. In this case the issues will not be removed from the clipboard.

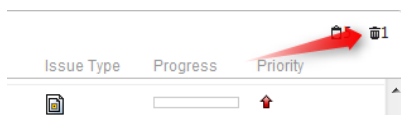
**Note** If the cut issue contains sub-issues, these sub-issues are pasted with their parent.

### Removed Issues

Removed Issues panel contains the issues that you have just removed from the current structure - just in case you need them back.

Once you remove an issue from a structure it is saved to the **Removed Issues** secondary panel. If you have removed an issue by mistake, you can open Removed Issues panel and reinsert it back into the structure.

The Trash icon in the top right corner of the structure widget indicates if there are any issues in the Removed Issues panel. To show or hide the panel, click the Trash icon.

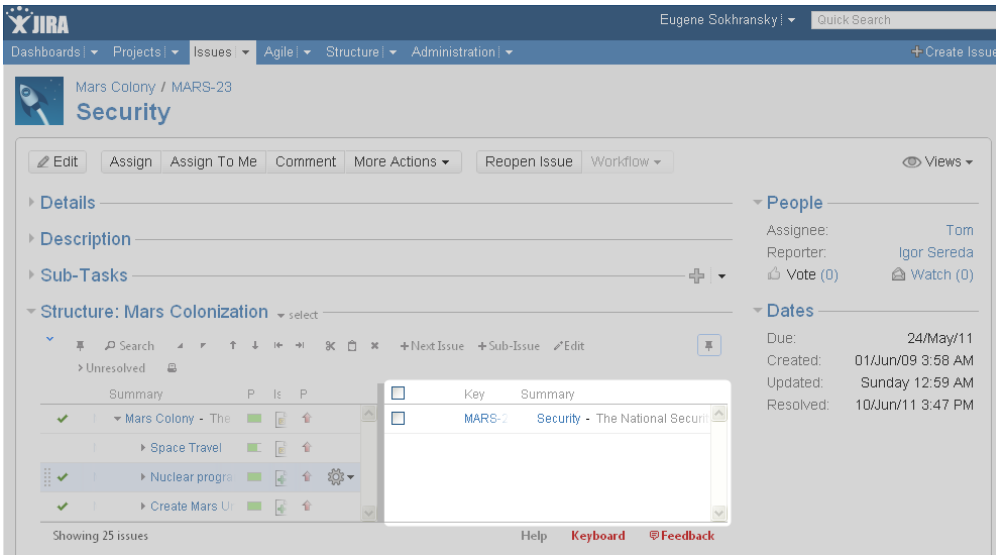


To add issues back to the structure, select the desired issues in the Removed Issues panel and move them to the desired location in the structure using **drag-and-drop** or **cut and paste** operations.

**Note** Contrary to the **Issue Clipboard**, the contents of the Removed Issues panel does not survive page reload. So if you navigate to a different page, you will no longer be able to view the removed issues.

### Pinned Issues

The **Pinned Issues** secondary pane is shown in the structure widget on the **Issue Page**, if the currently selected issue is not present in the currently selected structure.



With this secondary pane you can immediately see that the issue is not in the structure and, if necessary, add it to the structure on the spot using drag-and-drop.

To hide/show pinned issues, click the Pin icon in the top right corner of the structure widget.

## Printing Structure

Printable page lets you print the current structure from the browser.

Click the **Printer** button on the structure toolbar and the structure spreadsheet will open in a separate browser window or tab. The view fully copies the structure widget appearance - you can see the same issues as in the structure. For example, if some sub-issues are hidden, you will not see them on the printable page either.

Mars Colonization

Print Font size: 8 A a

Key	Summary	Progress	Issue Type	Key	Summary	Progress	T	P
MARS-1	Mars Colony - The colonization of	100%		MARS-1	Mars Colony	100%		
MARS-1390	Space Travel	72%		MARS-1390	Space Travel	72%		
MARS-1391	Space Ships	97%		MARS-1391	Space Ships	97%		
MARS-6	Design spacecraft	100%		MARS-6	Design spacecraft	100%		
MARS-7	Build spacecraft	80%		MARS-7	Build spacecraft	80%		
MARS-8	Experimental Flight	100%		MARS-8	Experimental Flight	100%		
MARS-1392	Infrastructure	83%		MARS-1392	Infrastructure	83%		
MARS-28	Orbit calculation - We n	100%		MARS-28	Orbit calculation	100%		
MARS-35	Weather forecasts	100%		MARS-35	Weather forecasts	100%		
MARS-38	Astronautery	0%		MARS-38	Astronautery	0%		
MARS-52	Nuclear program - Cheap er	100%		MARS-52	Nuclear program	100%		
MARS-55	Create Mars University	100%		MARS-55	Create Mars University	100%		

The columns displayed on the printable page will be the same as in the structure widget, however, the widths of the columns will be set by the browser. To change the columns on the printable page, change them in the structure widget and click the Printer button again.

Summary column on the printable page displays only the summary field, without issue description. If you'd like to print description, add a separate Description column to the structure widget.

Depending on the number of columns, and the amount of the texts, it may be necessary to adjust font size before printing.



It's a good idea to print a single sample page to decide whether font size needs changing.

When ready to print, click Print button on the printable page or use your browser's Print menu.


## Real-time collaboration

Structure widget is a real-time collaboration tool.

The issue hierarchy displayed in the widget is kept up-to-date with the JIRA server, so if someone else changes the structure on the server, you will see the web page update within several seconds.

In the same fashion, the values of the issue are maintained up-to-date: if someone edits an issue or otherwise changes it, the structure widget will update the displayed fields within a few seconds. A value that has been changed is highlighted for a second with a flashing yellow background.

This feature lets you collaborate with other people who may work with the same structure on different computers.

 Structure keeps data up-to-date by polling the server with short requests every few seconds when the application is ready. If structure widget detects that the browser is inactive it will reduce polling frequency to conserve network traffic.

## Using JQL to Select Issues in Structure

Structure adds `structure()` JQL function that lets you search for issues that are added to a structure, or for all sub-issues of a certain issue in a structure.

In order to search for structure issues, you need to use [Advanced Searching](#) in JIRA and JQL.


### *structure()* JQL function

To specify a structure condition in JQL, use the following format:

```
issue in structure(structureNameopt, parentIssueopt)
```


Function arguments:

<b>structureName</b>	<i>Optional</i>	The name of the structure. If you omit the structure name, the default <i>Global Structure</i> will be searched – unless it was removed from JIRA. Remember to enclose the name in double quotes ("") if it contains spaces or non-letters.
<b>parentIssue</b>	<i>Optional</i>	Use this parameter to select only part of the structure that consists of the specified parent issues and all its sub-issues of any level. If the issue specified is not added to the structure, an empty set is returned.

 You can use structure ID instead of the structure name. You can see structure ID in the URL of the Structure Board if you open **Manage Structure** page and click on the structure.

### Examples

- `issue in structure()` — issues that are added to the Global Structure
- `issue in structure("My Personal Structure")` — issues that are added to structure called "My Personal Structure"
- `issue in structure(101)` — issues that are added to structure with ID 101
- `issue in structure(PROJECT-123)` — issues that are added to the Global Structure, and are located under PROJECT-123 issue at some level, including the PROJECT-123 issue itself.
- `issue in structure("Structure X", PROJECT-123)` — issues that are added to the structure named "Structure X", and are located under PROJECT-123 issue at some level, including the PROJECT-123 issue itself.

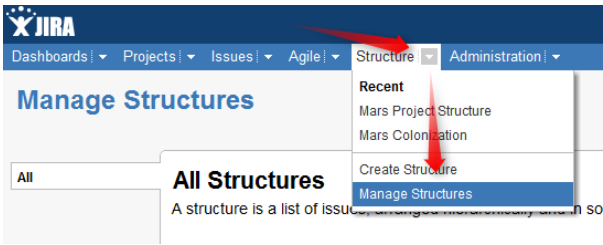
 If a user does not have [access to structure](#), they will not be able to create new queries with the `structure()` function and existing queries will have `structure()` function return an empty set. However, the user will still see `structure()` function offered in the JQL completion drop-down.

## Managing Structures

Structure Plugin lets you have several independent structures in JIRA. Manage Structures page lets you view, create, and delete structures as well as change their parameters.

To open Manage Structures page, go to **Structure** menu in the top navigation bar, and then select **Manage Structures**.





More about managing structures:

- [Structure Details](#)
- [Creating New Structures](#)
- [Editing Structure Details](#)
- [Structure Permissions](#)
- [Copying a Structure](#)
- [Deleting a Structure](#)

## Structure Details

Every structure has the following parameters:

<b>Name</b> ( <i>required</i> )	Name is used to identify the structure in the drop-downs like the <i>Structure</i> menu in the top navigation bar.
<b>Description</b>	Used to describe the meaning of the structure to the users.
<b>Owner</b>	The user who has created the structure. You can't change the owner.
<b>Permissions</b>	Define who can view, edit or configure the structure. See <a href="#">Structure Permissions</a> for details.
<b>Require Edit Issue Permission flag</b>	When <i>Require Edit Issue permission on parent issue to rearrange sub-issues</i> flag is set, additional permission constraints are applied to figure out what changes is the user allowed to make. See <a href="#">Structure Permissions</a> for details.

**Edit Structure** [Help!](#)

Name \*

Description

Owner User: eugene

Permissions User permission level is calculated by applying rules from this list, from top to bottom. The last matching rule takes precedence. Structure owner and JIRA administrators always have Control permissions.

1. By default, permission level is **None**

Add Rule  to  for  [+](#) Add

Options  Require Edit Issue permission on parent issue to rearrange sub-issues

You can specify structure details when [Creating New Structures](#) and when [Editing Structure Details](#).

## Creating New Structures

To create a new structure, use **Structure | Create Structure** menu or click **Create Structure** button on the Manage Structures page.

You need to specify at least the name of the new structure, and optionally description, permissions and other parameters. See [Structure Details](#) for description of the structure parameters.

When you create a new structure, you become the owner of the structure. Structure owner always has full access to the structure - see [Structure Permissions](#).



Only logged in users who have access to the Structure Plugin are allowed to create new structures. See [Who Has Access to the Structure](#).

## Editing Structure Details

To edit [details](#) of a structure:

1. Open Manage Structure page by using **Structure | Manage Structures** menu.
2. Locate the structure you need to change and click on **Configure** link in the **Operations** column.



If you do not see **Configure** link, then you probably do not have Control permission on that structure.

## Structure Permissions

Every structure has a list of permission rules, which defines who is allowed to see, edit or configure the structure.

### Access Levels

Each user has one of the following access levels to a structure:

<b>None</b>	The user does not see the structure at all and does not know that it exists.
<b>View</b>	The user can view the structure but cannot make changes.
<b>Edit</b>	The user can view the structure and can rearrange issues in the structure, add issues to the structure and remove issues from the structure.
<b>Control</b>	The user can view, edit and configure the structure - including changing structure permission rules and configuring synchronizers.

### Default Access

By default, all users have **None** access level.

The structure's owner and JIRA administrators always have **Control** access level.

Therefore, if you create a new structure and do not specify any permission rules, it will be a private structure that only you and JIRA administrators will be able to see and modify.

### Permission Rules

Users who have **Control** permission on a structure can define permission rules by [Editing Structure Details](#).

Permission rules list is an ordered list that's used to calculate the access level for a given user. Each rule has a **condition** that is matched against the user, and **access level** which is set if the condition matches. The conditions are applied from top to bottom, and the **last matching rule has precedence**.

The following conditions are supported by permission rules:

<b>Anyone</b>	Matches any user, including anonymous (not logged in). This condition can be used to set a default permission for everyone.
<b>Group(G)</b>	Matches users that belong to the group G.
<b>Project Role(R,P)</b>	Matches users that have role R in project P.

Additionally, there is a special rule type **Apply Permissions From**, which works by going through the permission rules from a different structure. You can apply permission rules only from structures with Control access level for you.

### Examples

- Anyone can view, developers can edit, only the owner and admins can control:

1. View for Anyone
2. Edit for jira-developers (Group)

- Any logged in user can edit, except for the users from structure-noaccess group, who can't even view the structure. Project administrators are allowed to control the structure:

1. Edit for jira-users (Group)
2. None for structure-noaccess (Group)
3. Control for Administrators of Mars Colony (Project Role)

- Incorrect configuration: everyone is given View access level

1. Control for jira-developers (Group)
2. Edit for jira-users (Group)
3. View for Anyone

Although the configuration looks ok at first glance, remember that **the last matching rule has precedence**. So regardless of whether the user is part of jira-developers or jira-users group, their access level will be set to View by the last rule.

### **Edit Issue JIRA Permission and Editing Structure**

If you set *Require Edit Issue Permission on Parent Issue* flag on the [Structure Details](#) page, additional per-issue permissions checks will be performed to decide whether the user is allowed to change the structure.

If the flag is on, the user must have Edit Issue permission on a parent issue to adjust its sub-issues. In other words, direct sub-issues (or children issues) are treated as if they are part of the parent issue, and therefore adding sub-issues, removing sub-issues and rearranging sub-issues is actually changing the parent issue - for which the Edit Issue permission is required.



The user must also have **Edit** access level to the structure to be able to make changes at all.

Note the following:

- Top-level issues do not have a parent issue, and therefore are not affected by this flag: the user can add/rearrange issues at the top level of the structure if they have Edit access level.
- If issue A has sub-issue B, and B has sub-issue C, then to be able to move or remove C from the structure, the user needs Edit Issue permission on B - not on A. In other words, the Edit Issue permission is required only for the direct parent issue.

## **Copying a Structure**

You can create a copy of the structure that will contain the same hierarchy of issues as the structure being copied.

To create a copy:

1. Open **Manage Structures** page using top navigation **Structure** menu.
2. Find the structure you'd like to copy and click **Copy** link in the Operations column.



If you don't see Copy link in the Operations column, then you possibly don't have permissions to create new structures.

3. A copy of the structure will be immediately created. You will be taken to Configure page for the copied structure where you can modify structure name, description and permissions.



You become an Owner of the copied structure



Permission rules for the copied structure are empty (private structure). Add permission rules to share this copy. See [Structure Permissions](#).

## **Deleting a Structure**

When you delete a structure, the following information is deleted:

- The hierarchical list of issues from the structure

- Structure details - name, description, permissions
- Synchronizers installed into structure

The issues that the structure contains are not affected in any way. They remain in JIRA and still can be part of another structure.



If there's any synchronizer installed for the structure you delete, it will not have a chance to react. So, if removing an issue from the structure should cause synchronization (such as removal of the links, done by the [Links Synchronizer](#)), you might need to first manually delete issues from the Structure to let the synchronizer do its job, and then delete the structure itself.

To delete a structure:

1. Open **Manage Structures** page using top navigation **Structure** menu.
2. Find the structure you'd like to delete and click **Delete** link in the Operations column.



You need **Control** access level to be able to delete a structure. See [Structure Permissions](#).

3. Review the structure you are about to delete and confirm the operation. **There's no Undo!**

## Synchronization

Synchronization lets you keep Structure issue hierarchy in sync with some other issue properties. For example, you can enforce the rule that JIRA sub-tasks should always be placed under their parent in the structure, or that there should be an issue link from parent issue to each sub-issue.

Synchronization can also be run once to perform a one-time update of the structure (Import) or one-time update of the issues based on the structure (Export).

Synchronization is extendable system that allows JIRA plugins provide their own synchronizers. The following synchronizers are supplied with the Structure plugin:

- [Sub-Tasks Synchronizer](#) places JIRA sub-tasks under their parent issues in the structure
- [Links Synchronizer](#) makes sure that sub-issues are linked to their parent issues with a specific link type, and it also can be used to reconstruct structure from links
- [Saved Filter Synchronizer](#) populates structure with issues that pass a saved filter, it also can be used to remove issues from structure when they no longer pass the filter
- [GreenHopper Synchronizer](#) works to sync GreenHopper ranking of issues with their position in the structure and to make it easier to assign stories to epics by using Structure
- [Status Rollup Synchronizer](#) allows to make propagate issue status upwards, for example, make parent issue Resolved if all sub-issue are resolved.

One-time synchronization works when you run [Export](#) or [Import](#), or when you run a [Resync](#). Automatic synchronization runs in the background and listens for the updates in the structure and beyond.



Please be careful: synchronization may cause massive changes on the issues. For example, if you install GreenHopper synchronizer and then add issues to the structure in some random order, those issues' ranks will be changed according to that order almost immediately! We plan to add "synchronization undo" in the future version to reverse possibly uncalled bulk changes. But for now, please make sure you have daily backups and carefully read how a synchronizer works before installing it

In order to install a synchronizer you need to have **Control** permissions on a structure and have necessary permissions on the JIRA issues.

## Importing Structure

When you **Import** a structure, you get a set of issues which should be in the structure and/or information on how they should be arranged in a hierarchical list. Then this information is applied to an existing structure.

For example, you can use a [Saved Filter Synchronizer](#) to add All Open Issues to a structure (or the results of whatever Saved Filter you have), or [GreenHopper Synchronizer](#) to rearrange issues in the structure according to their rank and epic in GreenHopper.




To run Import, you must have **Control** permissions on the structure and additional permissions may be required by a specific synchronizer.


To import hierarchy into a structure:

1. Open **Manage Structures** page using top navigation Structure menu.


2. Select the structure you'd like to import into and click **Import** link.


 If you don't see **Import** link in the Operations column, then you possibly don't have Control permissions for this structure.

3. Select a synchronizer from the drop-down list and proceed to configure import parameters.

 If there are no synchronizers in the drop-down list, then either none are currently installed or none of the installed synchronizers support import into a structure.

4. Enter synchronizer parameters. Each synchronizer has its own parameters, so please refer to [specific synchronizer documentation](#). If you're not yet acquainted with how this specific synchronizer works, please read the Rules section on the parameters page.
5. Click **Run Import**. When you start import, synchronizer will analyze data and possible update the whole structure.
6. After you click Run Import and confirm the operation, a job status page is shown. When the job is marked as Finished, the synchronization is done and you can view the results by opening the affected structure.


 When import is run, it runs under your user name and with your permissions. So if you don't have enough permissions to read values somewhere else or to view issues you'd like to import, you may not see the expected result.

 **Import** and **Export** are actually a one-time **Resync**. Export is resync from Structure and import is resync into Structure. If you need to run export or import periodically, you can set up a synchronizer with all the parameters but without enabling it - so no synchronization happens in the background. When you need to export or import, you can open Synchronization Settings page for the structure and run Resync. Just make sure you've selected the correct Resync direction!

## Exporting Structure


When you **Export** a structure, you use the hierarchy from the structure to create, update or delete other issue attributes or make any other changes based on the hierarchy that a specific synchronizer provides.

For example, you can use [Links Synchronizer](#) to create issue links between sub-issues and their parents.


 To run Export, you must have **Control** permissions on the Structure, and you will likely need some additional permissions, depending on which synchronizer you're going to use. For example, you have to have *Link Issues* permission when working with the Links synchronizer.

To export hierarchy from a structure:


1. Open **Manage Structure** page using top navigation Structure menu.
2. Select the structure you'd like to export from and click **Export** link.

 If you don't see **Export** link in the **Operations** column, then you possibly don't have **Control** permissions on this structure.

3. Select a synchronizer from the drop-down list and proceed to configure export parameters.

 If there are no synchronizers in the drop-down list, then either none are currently installed or none of the installed synchronizers support exporting from a structure.

4. Enter synchronizer parameters. Each synchronizer has its own parameters, so please refer to [specific synchronizer documentation](#). If you're not yet acquainted with how this specific synchronizer works, please read the *Rules* section on the parameters page.
5. Click **Run Export**. When you start export, the synchronizer will read the current structure and apply it to whatever it syncs with.
6. After you have clicked Run Export and confirmed the operation, a job status page will be present. When the job is marked *Finished*, the synchronization is done and you can inspect the results.

 When export is run, it runs under your user name and with your permissions. So if you don't have enough permissions to make a certain change in JIRA, the synchronizer will skip that change (a warning will be printed out in the server logs).



**Import** and **Export** are actually a one-time **Resync**. Export is resync from Structure and import is resync into Structure. If you need to run export or import periodically, you can set up a synchronizer with all the parameters but without enabling it - so no synchronization happens in the background. When you need to export or import, you can open Synchronization Settings page for the structure and run Resync. Just make sure you've selected the correct Resync direction!

## Installing Synchronizer

When you install a synchronizer on a structure, you make this structure automatically sync with something else. For example, after you have installed and enabled a **Links Synchronizer**, any changes someone makes to the structure will cause issue links to be created or deleted to match those changes. Or when you have installed and enabled **Saved Filter Synchronizer** in *Add* mode, creating or changing an issue that causes it to pass the selected saved filter will cause this issue to be added to the structure.

When you install a synchronizer, you define its parameters. It's not possible to edit those parameters later, but you can remove the synchronizer and install a new one with different parameters.

Please note that after a synchronizer is installed, it's not working yet - it must be **Enabled** to start monitoring the changes.

To install a new synchronizer:

1. Open **Manage Structure** page using top navigation Structure menu.
2. Find the structure you'd like to sync. The **Sync With** column shows currently installed synchronizers. Click on the **Settings** link in that column.



If you don't see **Settings** link in the **Sync With** column, then you possibly don't have **Control** permissions on this structure.

3. Synchronization settings page shows detailed information about each installed synchronizer and lets you work with them. To proceed with the installation of a new synchronizer, select the type of the synchronization and click **Configure and Install Synchronizer**.
4. Enter synchronization parameters. Each synchronizer has its own parameters, please refer to the [specific synchronizer documentation](#).



If you're not acquainted with how this synchronizer works, please make sure to read the *Rules* section at the top of the page. Especially text in red.

5. Press **Create** button and the synchronizer gets installed. However, it's not enabled yet.
  - a. Before synchronization is enabled, you might want to run Resync to bring the current state of the structure and JIRA to the same page. In that case, press **Resync and Enable** button after the synchronization is installed, or later use the same link on the synchronization settings page.
  - b. If you need to enable synchronization without resyncing first, press **Enable without Resyncing**.
  - c. You can enable and resync the synchronizer later from the synchronization settings page. Press **Done** if you don't need to enable the newly installed synchronizer now.



**Import** and **Export** are actually a one-time **Resync**. Export is resync from Structure and import is resync into Structure. If you need to run export or import periodically, you can set up a synchronizer with all the parameters but without enabling it - so no synchronization happens in the background. When you need to export or import, you can open Synchronization Settings page for the structure and run Resync. Just make sure you've selected the correct Resync direction!

## Removing Synchronizer

You can remove an installed synchronizer at any time if you have **Control** permissions on the structure.

1. Open **Manage Structure** page using top navigation Structure menu.
2. Find the structure you need to remove a synchronizer from. You can look at *Sync With* column to see which synchronizers are installed in a structure. Click **Settings** link for the selected structure.
3. Find the synchronizer in the list and use **Delete** link to remove it.

Note that if the synchronizer is currently performing an incremental sync or resync, it will be allowed to finish.


## Turning Synchronizer On and Off

A synchronizer is disabled by default and it's usually explicitly enabled after it is installed into a structure, probably with a resync. The following list summarizes the possible states of a synchronizer:

- **Disabled** - the background incremental synchronization is not running. You can run [resync](#) to do a one-time full sync.
- **Enabled** - the background incremental synchronization is running. When a change is detected, synchronization applies the change to the other part of the synchronous link as soon as possible, typically within several seconds.
- **Not Available** - the synchronizer is installed but it cannot run any synchronization. The possible reasons are changes in JIRA configuration or lack of permissions from the user.

To **disable** an active synchronizer:

1. Open synchronization settings page for the structure.
2. Find the synchronizer and click **Disable** link.

 If the synchronizer is currently running a sync, it will be allowed to finish.

To **enable** an inactive synchronizer:

1. Open synchronization settings page for the structure.
2. Find the synchronizer and click **Enable** link.
3. Alternatively, you can click **Resync and Enable** to [resync](#) and enable immediately after resync finishes.

## Running Resync

A resync, or full resynchronization, is a one-time process activated manually by the user to bring Structure and some other aspect of issues to the same page. Resync typically scans all issues that may be affected - contrary to the incremental synchronization, which inspects only issues that have been changed.


For example, running resync on a Saved Filter synchronizer (in *Add* mode) runs the related Saved Filter and makes sure all issues from the result set are in the structure. When the same synchronizer is working in the background, it checks only those issues that have been changed.

### Resync Directions

Resync is also different from incremental synchronization in that it has a direction. The incremental sync tries to apply changes on *both* sides to the other side, if possible, depending on where the change has happened: with GreenHopper synchronizer, if you change the rank (issue position in backlog on the Planning Board), its position in the structure is also changed; and if you change the position in the structure, GreenHopper's rank is changed. However, when applying Resync, you need to choose which side of the data is to be taken as the final version and which is to be updated.

Resync can be run:


- *from Structure*, which means that the issue hierarchy in the Structure is the final data and the synchronizer should update whatever it syncs with. This is what happens when you [export from a structure](#).
- *into Structure*, which means that the issue hierarchy is going to be updated (or issues possibly added or removed), and whatever the synchronizer syncs with has the final say. This is what happens when you [import into a structure](#).

 A synchronizer may support resyncing in only one direction. For example, Saved Filter synchronizer, which adds issues from a saved filter result, can only sync *into* Structure.

### Running Resync

To run a resync:

1. Open **Manage Structure** page using top navigation Structure
2. Click **Settings** link in the *Sync With* column for your structure
3. On the synchronization settings page, find the synchronizer you'd like to Resync, and either
  - a. Click **Resync**
  - b. Click **Resync and Enable** if the synchronizer is disabled and you'd like to enable it immediately after Resync finishes
4. Select a direction for the Resync. For example, *GreenHopper Structure* means that the data will be taken from GreenHopper and the structure will be rearranged. If a direction is not supported by the synchronizer, it will be disabled.
5. Click **Start Resync**.

 Resyncing in a wrong direction may lead to data loss! Please make sure you understand that you're doing the correct thing and confirm running the resync when a confirmation dialog appears.

6. The job status page that appears will tell you when the Resync has finished.



If the synchronizer is currently running an incremental synchronization, the resync will wait until it finishes.

## Synchronization and Permissions



**IMPORTANT!** Please read.

When synchronizer makes changes to bring structure and some other aspect of issues to the same page, it should do that on behalf of a certain user - for the sake of permissions and logging. This user is the one who has created the synchronizer, and it is displayed in *Run as User* column on the synchronization settings page.

Due to asynchronous nature of the synchronization, the changes that cause sync may be effected by a different user or users. **However, when sync runs, the updates will be made on behalf of the user who installed the synchronizer!**

This is really important to understand. Consider the following settings:

- You create a Structure and you set up structure [permissions](#) so that anyone can edit the structure.
- You have Link Issues permissions on a project and you install Links synchronizer to have children issues linked to their parent issue.

Now, anyone can edit the structure - add issues there, remove issues from there and rearrange the issues in the structure. **Every change of the structure will lead to adding and removing links between the affected issues on your behalf - even if the user who changes the structure does not have Link Issues permission!**

So when using synchronizer, Structure edit permissions implicitly grant limited permissions to make changes according to the synchronizer's algorithm, as well as issue permissions implicitly grant limited permissions to edit the structure.

## Bundled Synchronizers

There are several bundled synchronizers coming with the Structure. Other synchronizers can be provided by other JIRA plugins.

- [Sub-Tasks Synchronizer](#) — Sub-Tasks Synchronizer lets you have sub-tasks automatically placed under their respective parent issues in the structure.
- [Saved Filter Synchronizer](#) — Saved Filter Synchronizer lets you automatically add issues from a Saved Filter result to the structure or remove issues from the structure that no longer are in the Saved Filter result.
- [Links Synchronizer](#) — Links Synchronizer maintains issue links between parent issue and children issues.
- [GreenHopper Synchronizer](#) — GreenHopper Synchronizer lets you automatically synchronize GreenHopper Rank with the position of issues in the structure, and Epic/Theme field with the position of stories under epics in the structure.
- [Status Rollup Synchronizer](#) — Status Rollup synchronizer automatically aggregates statuses of the sub-issues and updates the status of the parent issue. For example, it can make parent issue *Resolved* if all sub-issues are *Resolved*.

## Sub-Tasks Synchronizer

Sub-Tasks Synchronizer lets you have sub-tasks automatically placed under their respective parent issues in the structure.



This synchronizer is available only when Sub-Tasks are enabled in your JIRA and you have at least one Sub-task issue type defined.

### Sub-Tasks Synchronizer Parameters

You can select which sub-task issue types the synchronizer works with. Issues of other issue types will not be affected.

This synchronizer supports only Import / Resync into Structure ([more about resync](#)).

### Sub-Tasks Synchronizer Rules

- When there's a sub-task (of one of the selected types) and its parent issue is in the structure, the sub-task is also added to the structure and placed under its parent task.
- The parent issue must be in the structure already - the synchronizer does not add parent AND sub-task, neither does it add parent for the sub-tasks already added.



You can add parent issues to structure manually, or use Saved Filter synchronization to add parent issues (and probably sub-tasks) automatically.



- If a sub-task is already in the structure, and is located under a different parent (or at the top level), it will be moved under its *subtask parent* (with all sub-issues that it may have).
- Changes in structure are not synced back to sub-tasks: if you place an issue under another issue, it will not become a sub-task.
  - If you move a sub-task away from its parent task, it will soon be moved back by the synchronizer.

## Saved Filter Synchronizer

Saved Filter Synchronizer lets you automatically add issues from a Saved Filter result to the structure or remove issues from the structure that no longer are in the Saved Filter result.

This powerful synchronizer lets you control the contents of the structure with a saved filter. You can create Saved Filters using JIRA's Issue Navigator page.

### Saved Filter Synchronizer Parameters

<b>Filter</b>	A Saved Filter to sync with. The drop-down box offers to choose from the Saved Filters owned by you or favorited by you.
<b>Add</b>	Turns on <b>Add Mode</b> : the synchronizer will make sure that all issues from the saved filter's result are present in the Structure.
<b>Place added issue at the top level</b>	The newly discovered issues from the Saved Filter result are placed at the top level, at the end of the structure.
<b>Place added issue as a sub-issue of ...</b>	You can enter issue key (like PROJECT-123) of an issue that will serve as the parent for the newly discovered issues from the Saved Filter. They will be placed as children of this issue, at the end of the current children list. Note that if this issue is not present in the structure, the issues won't be added at all.
<b>Remove</b>	Turns on <b>Remove Mode</b> : the synchronizer will remove issues from the structure when they no longer are present in the saved search result. However, if an issue to be removed contains sub-issues that should stay in the structure, it will not be removed.
<b>Remove only from where added issues are placed</b>	Additional flag to remove issues only if they are either at the top level or under the issue where they were initially placed by the synchronizer. So if you move an automatically added issue somewhere else, it will not be removed even if it is no longer present in the search result.

This synchronizer supports only Import / Resync into Structure ([more about resync](#)).



If the saved filter is deleted later, or if you lose permissions to run it, the synchronizer will not work.



No matter how synchronizers are configured, they will only affect issues from the projects that are [enabled for synchronization](#).



**CAREFUL!** Please be careful when turning on Remove mode and installing another synchronizer into the same structure. It is possible to set up the structure synchronizers in a way to make them cycle: some other synchronizer, like Sub-tasks synchronizer, would add an issue to the structure and then Saved Filter synchronizer in Remove or Add/Remove mode would remove that issue, and so forth.

### Saved Filter Synchronizer Rules

- Synchronizer adds issues from a saved filter's result to structure and/or removes issues that no longer are in the filter's result.
- Whenever an issue changes, a query is run to see if it matches the filter. On resync, all issues are checked.
- With **Add** mode on, an issue will be added to the structure if it matches the filter - even if the user has manually removed it from there. If the issue is already in the structure, it will not be affected.
- With **Remove** mode on, an issue will be removed from the structure if it does not match the filter - even if the user has manually added it before.

## Links Synchronizer

Links Synchronizer maintains issue links between parent issue and children issues.

You can use this synchronizer to replicate the hierarchy in the structure with issue links, or to import a hierarchy that was previously created with links.



Links synchronizer is available only when Links are enabled and there's at least one link type.

#### Links Synchronizer Parameters

<b>Link Type</b>	The type of the link to sync with. Links of other types will be ignored.
<b>Exclusive Mode</b>	The exclusive mode allows the synchronizer to add issues to the structure if they have a link of the selected type, as well as remove links from issues that are not in the structure.
<b>Link Direction</b>	Defines which side of the link is the parent issue and which is sub-issue.

This synchronizer supports both Import and Export / Resync into/from Structure ([more about resync](#)). However, the incremental synchronization only works in the direction *from* Structure (creating links won't lead to adding / placing issues in the structure unless you Resync).



No matter how synchronizers are configured, they will only affect issues from the projects that are [enabled for synchronization](#).



**CAREFUL!** Please be careful when using this synchronizer with **Exclusive** mode turned on, because Resyncing *from* Structure would delete all the existing links of the selected type between issues that are not in the corresponding positions in the structure.

#### Links Synchronizer Rules

- When synchronizer is enabled, any changes in the structure will be reflected by creating and removing links of the selected type. Links created or removed by the user will not be automatically reflected in the structure.
- Links created and removed by the synchronizer are not recorded in the issue history, and issue update time is not changed (due to performance reasons).
- Use Resync (*from* Structure to Links) or Export to update the links between the issues currently in the structure.
  - If Exclusive Mode is turned on, all other links of the selected type will be removed (limited by projects enabled for Structure).
- Use Resync (*from* Links *into* Structure) or Import to rearrange the issues in the structure according to the existing links.
  - If Exclusive Mode is turned on, all issues that have links of the selected type will be added to this structure. Otherwise, only issues that are already in the structure are affected.
- Links that violate hierarchy restrictions will be ignored. The restrictions are:
  - A sub-issue may have only one parent issue.
  - There should be no sub-issue cycles.

## GreenHopper Synchronizer

GreenHopper Synchronizer lets you automatically synchronize GreenHopper Rank with the position of issues in the structure, and Epic/Theme field with the position of stories under epics in the structure. It should be used if you'd like to use both GreenHopper and Structure plugins in the same project.



GreenHopper synchronizer is available only for JIRA 4.2 or later and only when GreenHopper plugin is installed.




GreenHopper 5.8 introduced *Global Rank* field, which can be used to manage a multi-project backlog. GreenHopper synchronizer in Structure lets you select multiple projects to sync with when you are using Global Rank.

#### GreenHopper Synchronizer Parameters

<b>Project</b>	A project that GreenHopper is used in. The structure may contain issues from other projects, they will not be affected. <b>GreenHopper 5.8 or later:</b> Multiple projects may be selected. The issues from all selected projects will be synchronized using the same Global Rank field.
<b>Auto-add Subtasks</b>	When turned on, sub-tasks will be automatically added to the structure and forced to stay under their respective parent issues, like they do on GreenHopper's Planning Board. This works similarly to <a href="#">Sub-Tasks Synchronizer</a> .
<b>Rank Field</b>	The field of type "GreenHopper Rank" that holds the rank (backlog order) for the selected Project. If you do not wish to synchronize rank, select <i>Don't synchronize</i> .
<b>Epic Field</b>	The field of type "Labels" that is used to hold the key of the Epic that the story belongs to. Typically named "Epic/Theme". If you do not wish to synchronize Epics content, select <i>Don't synchronize</i> .

<b>Epic Type</b>	Relevant only if an Epic Field is selected. Defines an issue type that is treated as Epic - typically named "Epic". All issues placed under an issue of this type in the structure will be updated to have Epic Field point to that issue.
------------------	--

This synchronizer supports both Import and Export / Resync into/from Structure ([more about resync](#)). Incremental synchronization watches both structure changes and GreenHopper changes and applies the change to the other side.

 **CAREFUL!** Please be careful when using this synchronizer, especially when you add multiple issues to the Structure, as this may lead to massive updates in the GreenHopper ranks without undo.

**GreenHopper Synchronizer Rules**

**Common Rules:**


- Issues that do not belong to the synchronized project and issues that are assigned to Fix Versions that have been released are not affected.
- This synchronizer does not add issues to the structure (with two exceptions, explained below). You can use Saved Filter synchronizer together with GreenHopper synchronizer to automatically add and position issues.

**Sub-Tasks Synchronization:**

- With **Auto-Add Subtasks** mode on, sub-tasks are added to the structure if their parent is there in the structure.
- The sub-tasks are forced to stay under their parent, so if you move a subtask somewhere else, it will jump back under the parent again. You can rearrange the order of the sub-tasks, which will be sync'ed to the GreenHopper Rank if the Rank Field is configured.

**Rank Synchronization:**

- Repositioning issues in the structure causes Rank change and the repositioning issues on the Planning Board.
- Rearranging issues on the GreenHopper's Planning Board causes the issues to be rearranged in the structure.
- When issues are repositioned in the structure according to Rank, they are never moved under a different parent issue.

 This restricts the possible rank changes in GreenHopper - you can only move an issue to the position of another issue that is under the same parent issue in the structure, otherwise the issue will "jump back" later.

**Epic Synchronization:**

- Placing an issue under an Epic in the structure will cause its Epic/Theme field to change to that Epic.
  - It does not matter at what level of depth is the sub-issue. A sub-sub-sub-issue of an Epic issue will also have its Epic/Theme field updated.
- If you manually change Epic field to point to a different Epic, the issue will be repositioned under that Epic in the structure.
  - An issue that has the Epic/Theme field pointed to an Epic in the structure will be automatically added to the structure.

**How to Add Issues to Structure Sync'ed with GreenHopper**

When GreenHopper synchronizer is enabled, it automatically updates GreenHopper order in background when any Structure change happens. So if you carelessly add issues from the sync'ed project to the structure in some random order, their ranks will be updated according to that order.

**To add issues to the structure without breaking the existing backlog order:**

- If adding manually on the Structure Widget, use JQL search and add *order by Rank* clause at the end of the query. Use the rank field that is used by the synchronizer.
- Select the position of the added issues carefully (best with drag-and-drop or copy/paste) - the order is likely to change unless you place issues under another issue without any other sub-issues (see *Syncing Partial Orders* below).
- If using Saved Filter synchronizer to add issues, add *order by Rank* clause to the Saved Filter's query. However, the new issues that are added with the Saved Filter synchronizer will appear at the end of the structure and so will have the latest ranking.

**Syncing Partial Orders**

GreenHopper's Planning Board is flat (except for sub-tasks), and the Structure is hierarchical - so it is not possible to precisely rearrange Structure to have all issues come in the same order as they do on the Planning Board, without changing issue parents or making the Structure also flat.

Henceforth, the Structure syncs subsets of the issues in the hierarchy with GreenHopper rank. For example, consider the following Structure:

A	
	B
	C
D	

	E
	F

It is not possible to rearrange the sub-issues so that they come in the following order: B, E, C, F - although this is possible on the Planning Board. Instead, the structure will synchronize sub-sets of the issues in the Structure with GreenHopper. The following sub-sets will be synchronized separately:

- A, D - top-level issues: A must come before D on the Planning Board
- B, C - sub-issues of A are sync'd separately, so B must come before C on the Planning Board
- E, F - ditto for the sub-issues of D

### Status Rollup Synchronizer

Status Rollup synchronizer automatically aggregates statuses of the sub-issues and updates the status of the parent issue. For example, it can make parent issue *Resolved* if all sub-issues are *Resolved*.

#### Status Rollup Synchronizer Parameters

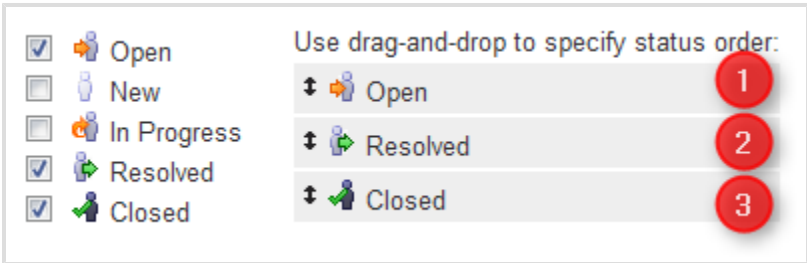
<b>Enabled Projects</b>	Only issues belonging to the selected projects are <b>changed</b> . It does not matter what project sub-issues belong to, as long as their parent belongs to the enabled project — every sub-issue counts with its status.
<b>Enabled Issue Types</b>	Same with types — you can select issues of which types may be changed by the synchronizer, and like with the enabled projects, only the parent issue type is checked.
<b>Statuses Rolled Up</b>	The selection and order of statuses that are used to calculate parent issue status. Parent issue status is set to the <i>earliest</i> status among its sub-issues. If a sub-issue has a status not selected in this parameter, the parent issue is not changed.
<b>Allowed Transitions</b>	For every status, you can select which transitions can the synchronizer make to move an issue to that status.
<b>Resolution</b>	Value to set to the <i>Resolution</i> field when workflow transition requires it. By default, a current or default value for Resolution is used.

The synchronizer is normally installed, resynced and used in the Incremental mode, tracking changes to issues and structure and updating issues. The synchronizer supports Exporting from Structure, changing statuses of the issues in the structure on one-time basis.

#### How Status Rollup Synchronizer Works

The synchronizer tracks updates to issues and to structure, and tries to make sure that the status of the parent issue corresponds to the aggregate status of its direct children.

When you configure Status Rollup, the most important parameter is the selected Statuses and their order:



**i** Statuses that are not selected in the parameters are not recognized by the synchronizer. If a sub-issue has one of the unselected statuses, the synchronizer does not change the parent issue.

The order of the selected statuses should correspond to *earliest-to-latest* order of phases of the workflow. For example, the screenshot above shows configuration where *Open* is followed by *Resolved*, which is followed by *Closed*. With that configuration, once all sub-issues of an issue are *Resolved*, the synchronizer will try to make the issue *Resolved* too. Once all sub-issues are *Closed*, the issue will be made *Closed*. But if at least one sub-issue happens to be *Open*, the issue status will be set to *Open* — because it is the earliest status in the specified order.

Summary	Status
Parent Issue	Open
Sub-Issue 1	Resolved
Sub-Issue 2	Open
Sub-sub-issue 2.1	Open
Sub-sub-issue 2.2	Resolved
Sub-Issue 3	Closed
Sub-sub-issue 3.1	Closed

Issue status is set to the earliest status its direct sub-issues have

On the screenshot above:

- All **sub-sub-issues** and **sub-issue 1** do not have sub-issues of their own, so the synchronizer does not change their status.
- **Sub-issue 3** has a single sub-issue, which has status **Closed** — so since all of its sub-issues are closed, it should be **Closed** too.
- **Sub-issue 2** has one **Open** sub-issue and one **Resolved** sub-issue — it should be **Open** because Open status comes before Resolved in the order specified earlier.
- **Parent Issue** has sub-issues that have statuses **Open**, **Resolved** and **Closed** — so it should be **Open** for the same reason. Once all sub-issues are **Resolved**, Parent Issue will be automatically **Resolved**. Once all sub-issues are **Closed**, Parent Issue will automatically be **Closed**.

Remember, that whenever one of the sub-issues gets a status not listed in the synchronizer configuration, the synchronizer just skips the issue. For example, if we change the status of **Sub-issue 2** above to **In Progress**, **Parent Issue** will not be updated. If we then change the status of **Sub-issue 2** to **Resolved**, **Parent Issue** status will be updated to **Resolved**.

### How Status is Changed

JIRA allows status to be changed only through a workflow transition, so the only way Status Rollup synchronizer can set the desired status on an issue is to apply a workflow transition. Therefore, when you select a status, you also need to select which transitions is synchronizer allowed to make.

The screenshot shows a configuration window for allowed transitions. A red arrow points to the 'Open' status icon, labeled 'Desired Issue Status'. Another red arrow points to the 'Open' status icon, labeled 'Possible Statuses of Issues Before Transition'. Below this, a list of transitions is shown with checkboxes:
 

- Any possible
- Stop Progress (from In Progress - jira)
- Stop Progress (from In Progress - products-4)
- Confirm (from New - products-4)
- Reopen Issue (from Resolved, Closed - products-4)

 Red arrows point to the text 'Any possible' and 'Reopen Issue' with labels 'Transition Name' and 'Workflow Name' respectively.

So what the synchronizer does is:

1. See what status does the issue currently have;
2. Calculate what status should it have, based on the statuses of sub-issues;
3. Find workflow transitions that can transfer the issue from the current status to the required status;
4. Check which of those transitions are allowed by the configuration;
5. Try to apply matching transition number one, if it fails - try the next one, and so on.



Note that all transitions are done under the account of the user who has installed the synchronizer.

### Why Can a Workflow Transition Fail

It's not guaranteed that the synchronizer will be able to change the Status, because workflows are too flexible and there are many reasons that a given transition, which you have allowed in the configuration, can fail to execute. Here's some of the possible causes:

- You (the user who has installed the synchronizer) do not have the required permissions to make the transition;
- You are not the Assignee of the issue — required for In Progress status;
- Some other pre-condition defined in the workflow fails;
- Workflow transition requires a field to be set on an issue that has no default value.

As described above, it's possible that there are several possible transitions from one status to another. The synchronizer will try all of them unless one of them succeeds.



If the synchronizer fails to update the status, a warning message will be written into the server logs (subject to logging configuration).

### Changing Resolution

You can set up a specific *Resolution* value to be set whenever a transition involves changing the resolution. If you don't specify this parameter, the default resolution or already existing resolution will be used.



In order to tell which issues have been automatically moved to a status like Resolved or Closed, you can set up a special resolution like *Auto-Resolved*.

### Manually Changing Status of an Issue That Has Sub-Issues

Even if an issue has sub-issues and is subject to Status Rollup, you can manually change its status. Although the synchronizer will **not** be forced to recalculate the status of that issue immediately, it will recalculate the status if any of the sub-issues change – probably reversing your change, if it finds an allowed transition.



If you'd like the synchronizer to only move issues *forward*, that is, from *Open* to *Resolved*, but not vice versa, you can configure the allowed transitions accordingly.

## Structure Administrator's Guide

This section contains information for JIRA administrators about installing and configuring Structure plugin.



Quick steps to get Structure working:

1. [Installing Structure](#)
2. [Setting Up Structure License](#)
3. [Selecting Structure-Enabled Projects](#)

Contents:

- [Installing Structure](#)
- [Setting Up Structure License](#)
  - [Structure License Parameters](#)
  - [When Structure is Available for Free](#)
  - [License Maintenance and Expiration](#)
- [Selecting Structure-Enabled Projects](#)
- [Restricting User Access to Structure](#)
- [Changing Permission to Create New Structures](#)
- [Structure Backup and Restore](#)
  - [Backing Up Structure](#)

- Restoring Structure from Backup
- Structure Files Location
- Disabling Feedback Button
- Troubleshooting
  - Troubleshooting Synchronizers
- System Requirements
- Who Has Access to the Structure

## Installing Structure

Structure is installed like the most other plugins. With JIRA 4.3 or later, or when you have Atlassian Universal Plugin Manager installed, you can locate the plugin at the Plugin Exchange and install from there.

Alternatively, you can download the plugin JAR manually from [Plugin Exchange](#) and place it into `plugins/installed-plugins` subdirectory under your JIRA home, then restart JIRA.

**Next:** [Set up Structure license key](#)

## Setting Up Structure License

Unless your JIRA runs on one of the [free licenses](#), Structure requires a license key to work. You can get a free no-obligation 30-day evaluation license key for your JIRA server in a few seconds.



Structure license must be set up in order for Structure features to become visible. Please install a license (get evaluation license, if needed) right after you have installed Structure plugin.



You need the *JIRA System Administrators* global permission to see and modify Structure license details.



In some cases, Structure plugin will not need a license key.

To **check whether you need a license:**

1. Navigate to **Administration | Structure | License Details**.
2. See section **Current License** - if there's no license there or if the license is expired, then you need to get an evaluation license or purchase a commercial license.
  - a. If Current License section says that you have Free License, then your JIRA must be qualifying for automatic free license and no further action is needed from you. See [When Structure is Available for Free](#).

### Getting an Evaluation License

To get a free 30-day unlimited-users evaluation license, follow **Get Evaluation License** link on the structure license page, or open [evaluation license request page](#) directly. In latter case please enter your JIRA server ID to get a correct license.

### Getting a Commercial License

You can purchase a commercial license at <http://almworks.com/structure/purchase.html>

### Installing a License Key

To install a license key:

1. Navigate to **Administration | Structure | License Details**.
2. Copy and paste the key to the **Install License** section.
3. Click **Install License**.
4. The installed [license parameters](#) will be displayed in the **Current License** section.
  - a. If your license doesn't fit the server, an error is displayed.

### Uninstalling a License Key

To uninstall a license key, just click **Uninstall License** on the structure license page. If there's no such button there, then no license is installed.

**Next:** Select which projects are enabled for Structure

## Structure License Parameters

The following parameters are displayed in the **Current License** section when you install a Structure license.

Parameter	Meaning
<b>License Type</b>	Commercial or Evaluation, may be others
<b>Licensee</b>	Organization authorized to use the license
<b>Serial Number</b>	A unique number assigned to the license
<b>Expires</b>	If present, the license is not perpetual: it will expire at the specified date. After that date passes, the Structure plugin will not be available unless the license key is changed.
<b>Maintenance Expires</b>	If present, the license key can only work with the versions of the Structure plugin released prior to the specified date. If you need to use a newer version of the Structure, you need to renew maintenance.
<b>User Limit</b>	This is the maximum number of users allowed by JIRA that are supported by this license key. The license that JIRA runs on must allow this number or fewer users.
<b>Server ID</b>	Although not shown in the license table, most licenses are tied to a specific JIRA server ID and would not install on a server with a different ID. If you need to move a license key to a different server, please contact support.

## When Structure is Available for Free

Structure plugin automatically installs a free license in case your JIRA runs on one of the following free licenses:

- Free license for **open-source** projects;
- Free license for a **non-profit** organization;
- Free **community** license;
- Free **demonstration** license;
- Free **developer** license.

The clauses from the Atlassian EULA that govern the use of those free licenses also apply to using Structure on JIRA servers where these licenses are installed.

## License Maintenance and Expiration

### **Commercial License**

Your commercial license for the Structure plugin (including Starter licenses) typically has no expiration date, so it's good to use forever. However, it has *Maintenance Expiration Date* which limits which versions of the plugin can be used with that license – you can only use the versions released prior to that date.

To use versions released later, you need to purchase maintenance renewal, which extends your maintenance expiration date one year forward – independently of the date of purchase.

Example:

Date license purchased	2012-01-01
License expiration date	None
Maintenance expiration date	2013-01-01
Products and terms allowed by the license	All versions released prior to 2013-01-01 can be used indefinitely
Maintenance renewal purchased	2012-12-10 (doesn't matter)
Renewed license maintenance expiration date	2014-01-01



Renewed terms	All versions released prior to 2014-01-01 can be used indefinitely
---------------	--

## Evaluation License

Evaluation and temporary licenses have expiration date, after which they just stop working – they allow to use the product before the specified date.

Make sure you renew evaluation or get another license key before expiration.

### License expiration and maintenance expiration warnings

JIRA administrators will see a warning message at the top of some pages when current Structure license nears expiration. (The message can be hidden and won't show again in the same browser.)

## Selecting Structure-Enabled Projects

Structure can be enabled for any selection of the JIRA projects, or for none of them. (In the latter case noone can use Structure.)



By default, Structure is not enabled for any project. To start working or evaluating Structure, enable it for at least one project.

To select which projects are enabled for Structure:

1. Navigate to **Administration | Structure | Configuration**.
2. Click **Enable/Disable Structure in Projects**.
3. Select whether Structure should be available for **all projects** or for **selected projects**.
4. In the latter case, change the projects list in the **Selected Projects** list by selecting one or more projects and using **Enable** and **Disable** buttons.
5. Click **Apply** when done.
6. In case you have disabled some projects that are already used in a structure (a structure contains issues from that project), you'll be given a warning. You can opt to **Proceed with Changes** or cancel.
  - a. If you proceed and disable a project that has issues in some structures, those structures will appear to the users without those issues.
  - b. If you later enable that project back - the issues will reappear where they were (all structure changes taken into account).



Which projects are enabled for the Structure affects [Who Has Access to the Structure](#)

## Restricting User Access to Structure

By default, Structure is accessible to anyone who has *Browse* permission on [structure-enabled projects](#). You can further restrict this access level to one or more user groups.

To select who can use Structure:

1. Navigate to **Administration | Structure | Configuration**.
2. Click **Select Structure Users**.
3. Select whether Structure should be available to **anyone** or to **selected user groups**.
4. In the latter case, change the **Selected Groups** list by selecting one or more user groups and using **Add** and **Remove** buttons.
5. Click **Apply** when done.



Which projects are enabled for the Structure also affects [Who Has Access to the Structure](#).



When Structure is enabled for **anyone**, even anonymous visitors will have access to Structure. To make Structure accessible to only logged in users, restrict access to **jira-users** group.


## Changing Permission to Create New Structures

By default, any logged-in user with [access to Structure](#) can create new structures of their own. However, you can restrict this ability to one or more user groups.

To select who can create new structures:


1. Navigate to **Administration | Structure | Configuration**.
2. Click **Select Who Can Create Structures**.
3. Select whether new structures can be created by **all Structure users** or by **selected user groups**.
4. In the latter case, change the **Selected Groups** list by selecting one or more user groups and using **Add** and **Remove** buttons.
5. Click **Apply** when done.


 The user also needs [general access to Structure](#) to be able to create new structures.

 Users who have *JIRA Administrators* global permission are always allowed to create new structures.

## Structure Backup and Restore

Structure data is backed up and restored separately from other JIRA data. Structure data includes structures, hierarchies, synchronizers - everything added to JIRA by the Structure plugin. Structure backup does not include issue data (except for some issue attributes that are added just in case).

 Structure is backed up separately due to technical limitations of the supported versions of JIRA platform. In the future we plan to merge Structure backup with the main JIRA backup.

 You need the *JIRA System Administrators* global permission to back up and restore Structure data.

### **Proper Backup and Restore Sequence**

To back up your JIRA with structure:

1. Perform the [standard JIRA backup](#).
2. [Back up Structure](#).

Of course you can back up structure separately if all you need to copy is the hierarchies and structure configuration.

To restore your JIRA with structure:

1. [Restore JIRA data](#) first.
2. [Restore structure from backup](#).

Structure backup files refer to issues by their IDs, so JIRA needs to have those issues restored before structure restore is commenced.

### **File-Based Backup**

You can also back up and restore the whole `structure` directory in the JIRA home (see [Structure Files Location](#)) manually - but only when JIRA is not running or when Structure plugin is disabled. When restoring structure data backed up this way, make sure you're using the same version of the Structure Plugin that was there when the backup was created.

Detailed Instructions:

- [Backing Up Structure](#)
- [Restoring Structure from Backup](#)

## Backing Up Structure

Backing up Structure saves the existing structures, their configuration and issue hierarchies. Structure backup does not save the issues themselves or other JIRA data - see [Structure Backup and Restore](#).

To back up Structure:

1. Navigate to **Administration | Structure | Backup Structure**.
2. Enter the name for the backup file. If you omit file extension, either `.xml` or `.zip` will be added to it.



You cannot specify directory for the backup file. Backup is always done to the *export* sub-directory under JIRA home.

3. **Backup as ZIP** option is turned on by default, but you can un-check it to get plain XML backup file.
4. Click **Backup**
5. If the file already exists, you will be given an option to overwrite the file or cancel the operation.

## Restoring Structure from Backup

Restoring structure from backup brings back the structures created at the moment of backup. However, if Structure database is not empty, the restore can be used to merge existing data with the data taken from backup.



Restoring structure will not affect issues in any way. The issues that make up the hierarchy should already exist in JIRA. If you do full restore, then you need to run the standard JIRA data restore first - see [Structure Backup and Restore](#).

To restore the structure from backup:

1. Navigate to **Administration | Structure | Restore Structure**.
2. Enter the full path to the structure backup file (either *.xml* or *.zip*).
3. In case current Structure database is not empty, you'll see the additional option: **Clear Structure Data Before Restoring from Backup**. If you leave it unchecked, the data from the backup will be merged with the existing structure data. If you check the Clear option, the current structure data will be wiped before restoring from backup.
4. Click **Restore**.
5. If you chose to clear the current database, confirm the operation. It cannot be undone!

After the structure has been restored, open **Structure | Manage Structure** page to see if the structures are there.



You also can restore structure data from backup files made with the earlier versions of the Structure plugin.

### How Existing Structure Data is Merged with the Data from Backup

1. Structures are merged by the structure ID. If a structure in backup and a structure in the database have the same ID, they are merged.
  - a. Structure merged by consequently adding issues mentioned in the backup structure (referred to by numeric issue ID) to the existing structure at the correct positions. If such issue already exists in the database structure, it is not added but rather moved to the required position - with all sub-issues, if there are any.
2. Synchronizers are merged by the synchronized ID. If a synchronizer with given ID already exists, it's not changed. If there's no synchronizer with such ID, it is installed.

## Structure Files Location

### Structure Files

`$JIRA_HOME/structure`

Structure keeps most of its data in the `structure` sub-directory under the [JIRA home directory](#).

### Database

Structure uses embedded Apache Derby database engine to store its data in `structure/db` directory.

### Undo Logs for Synchronizers

When [synchronizers](#) change issues in background - either when [running Export or Resync](#) or when running incremental synchronization - they store the undo information in special log files in `structure/syncundo` directory. The log files are kept for some time (~ 30 days by default) and then removed.

In case an incorrectly configured synchronizer ruins issue data (for example, Links synchronizer is able to remove all links of specific type when told to do so), the logs from `syncundo` sub-directory may be used to recover the data.

## Backing up Structure Files

When JIRA is not running or when Structure plugin is disabled, you can manually back up structure directory (separately or as a part of JIRA home backup). Likewise, you can manually restore the `structure` subdirectory if Structure plugin is not running - but be careful when downgrading to previous version of the Structure. As a rule, the database is not forward-compatible, so you can't downgrade and keep the database unless the versions of the Structure happen to have the same database schema.

You can also back up Structure into XML by using [Structure Backup](#) menu on the administrator's page.

## Disabling Feedback Button

A red **Feedback** button is displayed at the bottom of the Structure widget, allowing users to immediately open [Structure forum at UserVoice.com](#) to post their ideas about how to make Structure better for them.

However, this is a link that leads outside JIRA, so if your security policy prohibits such links in JIRA or if you generally don't like this idea, you can remove the Feedback link by doing the following:

- Open **Administration | Plugins** menu.
- Locate **Structure** among the installed plugins.
- On JIRA 4.3 or later, click on **manage plugin modules**
- Locate the following module: *Link to the UserVoice feedback forum at the bottom of the Structure widget (uservice-link)*
- Click **Disable** to disable that module

You can enable the Feedback button in the same way.



We suggest that you keep Feedback button turned on. It's just a link that helps everyone.

## Troubleshooting

This section lists common problems and methods of solving them:

- [Troubleshooting Synchronizers](#)

### Troubleshooting Synchronizers

[Structure synchronizers](#) work in background and can lead to changes in the structures or issue data that might be hard to trace. Complex configuration rules don't make things better, so it's important for JIRA admin to be able to track which synchronizers are doing what and what has caused a particular change a user is complaining about.

#### Log Files

You can reconfigure your JIRA logging to give structure synchronizers log messages lower logging level to get detailed reports about what's going on, and probably direct messages from the synchronizers into a separate log files.

The package name that all bundled synchronizers log under is `com.almworks.jira.structure.ext.sync`. You can add the following lines to `log4j.properties` file to have debug messages from synchronizers show on the console and/or in the log file (depending on their respective log levels):

```
log4j.logger.com.almworks.jira.structure.ext.sync = DEBUG, console, filelog
log4j.additivity.com.almworks.jira.structure.ext.sync = false
```

Or, you can set up a separate log file for synchronizer actions:

```
log4j.appender.structure-sync=com.atlassian.jira.logging.MultiTenantJiraHomeAppender
log4j.appender.structure-sync.File=structure-sync.log
log4j.appender.structure-sync.Threshold=TRACE
log4j.appender.structure-sync.MaxFileSize=20480KB
log4j.appender.structure-sync.MaxBackupIndex=1
log4j.appender.structure-sync.layout=org.apache.log4j.PatternLayout
log4j.appender.structure-sync.layout.ConversionPattern=%d %t %p %X{jira.username} [%c{4}] %m%n

log4j.logger.com.almworks.jira.structure.ext.sync = DEBUG, structure-sync, console
log4j.additivity.com.almworks.jira.structure.ext.sync = false
```

## Synchronization Undo Files

Structure keeps synchronization undo files for possible script processing in case massive changes are erroneously effected on issue data. The undo files contain actions that should be taken, in reverse order, to bring issue database to the original / previous state. However, undo files do not contain changes for the structures - only for the issue data.

See [Structure Files Location](#) for information about where syncundo files are located.

In some cases undo files may be preferred to the logs as they more clearly indicate the changes.

## System Requirements

The following are additions to [JIRA Requirements](#) imposed by the Structure.

### Browsers

Structure Plugin is compatible with the following browsers:

Browser	Versions	Known to NOT work
Mozilla Firefox	3.6+	3.5
Chrome	5+	
Internet Explorer	7+	6
Safari	5+	
Opera	10+	

### Server Requirements

- Structure needs at least additional 100MB of disk space on the server.
- It needs JIRA process to have write permissions to JIRA home directory to create *structure* sub-directory automatically.
  - Alternatively, you can create *structure* sub-directory under JIRA home manually and give JIRA process permissions to write to it.

### Non-conforming systems

With regards to systems that don't conform to JIRA requirements and Structure requirements: while we sometimes know that a specific configuration doesn't work, more often it's grey area so feel free to try and let us know.

## Who Has Access to the Structure

Structure is visible only to specific users. Those users who do not have access to the Structure, will not see *Structure* menu and other user interface elements, provided by the Structure plugin.

A user has access to Structure if all of the following conditions are met:

- The user has **Browse** permission on at least one of the projects that are [enabled for Structure](#).
- Structure is [enabled for this user](#):
  - Either Structure is enabled for everyone,
  - Or the user belongs to at least one of the enabled groups.



Users who have *JIRA Administrators* global permission always have access to Structure.

## Structure Developer's Guide

## Structure for Developers

Structure Plugin contains APIs that allow you to access the hierarchical lists of issues from other plugins and applications. Here are the typical use cases:

### **Custom Development**

You customize JIRA for your customer or employer, and you need to integrate Structure with some other in-house system – see [section about integrating plugins](#) and [Java API reference](#).

### **Plugin Integration**

You have your own great JIRA plugin, or plan to create one, and you'd like to use the issue hierarchy provided by Structure – see [Accessing Structure from Your Plugin](#).

### **Extending Structure**

You'd like to extend Structure, adding functionality to the plugin itself – read [documentation about extending Structure functionality with additional plugins](#). (You can also [extend the functionality of the Structure plugin itself](#).)

### **Remote Access**

You need to get or change issue hierarchy remotely from some automated scripts or a client application – read about [Accessing Structure Data Remotely](#) and [Structure REST API](#).

## Structure Developer Documentation

- [Accessing Structure from Your Plugin](#)
  - [Structure API Basics](#)
  - [Controlling Compatibility](#)
  - [Making Structure Dependency Optional](#)
  - [API Usage Samples](#)
- [Extending Structure Functionality](#)
  - [Creating a New Synchronizer](#)
- [Accessing Structure Data Remotely](#)
- [Reference](#)
  - [Structure Java API Reference](#)
    - [Structure API Versions](#)
  - [Structure Plugin Module Types](#)
    - [Synchronizer Module](#)
  - [Structure REST API Reference](#)
    - [Structure Resource - GET](#)
    - [Forest Resource - GET](#)
    - [Forest Resource - POST](#)

## Accessing Structure from Your Plugin

Structure provides a Java API that lets other plugins interact with the Structure data. The API is accessed through a few services that you can have injected into your components.

To start using Structure in your plugin:

### **1. Add dependency to your `pom.xml`**


Figure out the [version of the API](#) that you need - it may depend on your JIRA and Structure plugin version.

To use API classes, add the following dependency:

```
<dependency>
  <groupId>com.almworks.jira.structure</groupId>
  <artifactId>structure-api</artifactId>
  <version>2.0.0</version>
  <scope>provided</scope>
</dependency>
```

## 2. Import an interface

In your `atlassian-plugin.xml`, use `<component-import>` module to import the interfaces that you need.

 You can import `StructureServices` and get all other interfaces from there.

```
<component-import key="structure-services"
  interface="com.almworks.jira.structure.api.StructureServices"/>
```

## 3. Have Structure API service injected into your component.

```
public class MyClass {
  private final StructureManager structureManager;

  public MyClass(StructureServices structureServices) {
    structureManager = structureServices.getStructureManager();
  }

  ...
}
```

**Next:** [Learn API Basics](#)


# Structure API Basics

## Classes to Start With

Use ...	to ...
<code>StructureManager</code> and <code>Forest</code>	create and manipulate structures – add issues to the hierarchy, move issues within hierarchy, delete issues from the hierarchy, listen to structure events

The API uses an open source library `Integers`, which provides primitive-type collections with `java.util`-like interfaces. When working with `Forest`, you will typically use `LongList` and `LongArray` (an implementation of `LongList`).

See [API Usage Samples](#) to get the idea how to work with those interfaces.

 The dependency on the `Integers` library is added automatically when your add dependency on the API. Same goes for another dependency on the small JetBrains annotations library, that provides `@Nullable` and `@NotNull` annotations.

## More Power

Use ...	to ...
<code>StructureSynchronizer</code>	create a new synchronizer and declare it in a <a href="#">Synchronizer Module</a>
<code>StructureSyncManager</code>	manage synchronizers

StructureBackupManager	backup structure to a file and restore it back
ProgressCalculator	calculate progress (as shown by the Progress column in the structure widget)
StructureJobManager	schedule asynchronous jobs
IssueEventBridge	listen for aggregated issue events

Next: consider [Controlling Compatibility](#) and [Making Structure Dependency Optional](#)

## Controlling Compatibility

### Why Declare Compatible Versions

Structure Java API will change with time, and it is a good practice to ensure that your plugin uses the correct version of the API.

[Structure API Versions](#) page explains how version numbers change based on how compatibility is affected. Say, you develop your code using Structure API version 3.4.5 – your code will work with any version of the API starting from 3.4 and up to, but not including version 4.0.

So what happens if your code is run on JIRA with Structure that provides an incompatible API? It may break, or it may work. The exact answer depends on which parts of the API you use and what are the differences. But if the code breaks, it may not break outright – it may seem to work at first, until it tries to use a method that's not there, for example.

To make your code fail fast, you can declare dependency on a specific range of versions of the Structure API. In that case, if the version of the API is different, your plugin will fail to load and the user will immediately know that there's a problem.

### Importing Specific Range of API Versions

You can declare dependency on the specific range of the API versions via OSGi bundle instructions added to your `pom.xml` or `atlassian-plugin.xml`. Figure out the compatible OSGi versions range from the [API versions](#) table and modify your `pom.xml` to contain the following:

```
<plugin>
  <groupId>com.atlassian.maven.plugins</groupId>
  <artifactId>maven-jira-plugin</artifactId>
  <version>3.6</version>
  <extensions>true</extensions>
  <configuration>
    <instructions>
      <Import-Package>
        com.almworks.jira.structure*;version="[3.4,4)",
        com.almworks.integers*;version="0",
        org.jetbrains.annotations;version="0"
      </Import-Package>
    </instructions>
  </configuration>
</plugin>
```

Here we are declaring the acceptable range of versions for the Structure classes, taken from the example above. We don't much care about the versions of Integers and Annotations libraries, so `version="0"` will match any version of those packages.



You may have other `Import-Package` instructions to declare dependency rules for other packages, and you may have other instructions besides `Import-Package` as well. See the [API Usage Samples](#) for a more complete example.

**Next:** [Making Structure Dependency Optional](#)

## Making Structure Dependency Optional

If you are integrating your plugin with Structure, or when you generally write code that uses Structure API but also should work when Structure Plugin is not present, you need to declare that dependencies are optional and isolate dependencies in the code.

### 1. Declare Optional Dependency



Since your plugin must first be loaded as an OSGi bundle, it should declare dependencies from the Structure API packages as optional.

Modify `<Import-Package>` declaration in your `pom.xml` or `atlassian-plugin.xml` and add `resolution:=optional` classifier. (Add `Import-Package` to control API compatibility if you don't have this declaration yet.)

```
<Import-Package>
  com.almworks.jira.structure*;version="[3.4,4)";resolution:=optional,
  com.almworks.integers*;version="0";resolution:=optional,
  org.jetbrains.annotations;version="0";resolution:=optional
</Import-Package>
```

## 2. Isolate Dependencies in the Code

So once you have declared the optional resolution of the Structure API classes, your bundle will load - but if your code tries to access a class from the Structure API, you'll get a `NoClassDefFoundError`. To avoid that, you need to isolate the dependency on Structure API classes - typically in some wrapper classes.



This is also a point to make design decisions. So your code can use Structure when it's present, and can work independently when Structure is not there. Are there any abstractions that address both of these situation? What are the concepts that are realized through Structure API and through some other means when Structure is not available?

Here's a sample wrapper for the Structure API that provides `ForestAccessor` wrapper (whatever it does) when Structure is available and null otherwise.

```
public class StructureAccessor {
  public static boolean isStructurePresent() {
    if (!ComponentAccessor.getPluginAccessor().isPluginEnabled("com.almworks.jira.structure")) {
      return false;
    }
    try {
      Class.forName("com.almworks.jira.structure.api.StructureManager");
    } catch (Exception e) {
      return false;
    }
    return true;
  }


  public static ForestAccessor getForest(long structureId, User user) {
    if (!isStructurePresent()) return null;
    StructureManager structureManager;
    try {
      structureManager = ComponentManager.getOSGiComponentInstanceOfType(StructureManager.class);
    } catch (Exception e) {
      return null;
    }
    // check user permissions
    if (!structureManager.isAccessible(structureId, user, PermissionLevel.VIEW, false)) {
      return null;
    }
    Forest forest = null;
    try {
      forest = structureManager.getForest(structureId, user, false);
    } catch (StructureException e) {
      return null;
    }
    return new ForestAccessor(forest);
  }
}
```

## API Usage Samples

Use the sample plugins to learn by example. Download the source bundle from this page and use it with the latest API version.

**Download**

Name	Size	Creator	Creation Date	Comment
ZIP Archive structure-api-examples.zip	17 kB	Igor Sereda	Oct 22, 2011 04:55	

 The provided code is not production-quality and not supported. It is provided as a sample of how one can use Structure API. The sample code is in public domain – feel free to copy, modify and base your work on it.

### Sample Plugins List

Sample Plugin	Description
simple-plugin	Very basic demo of using <code>StructureManager</code> .
scheduled-sync	A plugin that allows to schedule periodical synchronization <a href="#">Resync</a> - uses <code>StructureManager</code> , <code>StructureSyncManager</code> and <code>StructureJobManager</code> .

## Extending Structure Functionality

You can add functionality to the Structure Plugin itself by creating a plugin that uses one of the available Structure extension points.


- [Creating a New Synchronizer](#) — Structure comes with a number of bundled [synchronizers](#), but you can add another synchronizer to the system, allowing Structure users to install it on structures and run export / import.

### Creating a New Synchronizer

Structure comes with a number of bundled [synchronizers](#), but you can add another synchronizer to the system, allowing Structure users to install it on structures and run export / import.

#### 1. Implement `StructureSynchronizer`

Create your implementation of [StructureSynchronizer](#) interface.

 Use [AbstractSynchronizer](#) or [AbstractIssueListeningSynchronizer](#) as the base class.

#### 2. Define `structure-synchronizer` Module

Add [Synchronizer Module](#) module to your `atlassian-plugin.xml`, referring to your implementation of the `StructureSynchronizer`.

#### 3. Test Thoroughly

Test how your synchronizer works when other synchronizers are also installed onto the same structure.

## Accessing Structure Data Remotely

Structure plugin provides REST API, which is primarily used by the [structure widget](#). The same API can be used to access the hierarchical data remotely from an automation script or another user agent application.

See details in the [REST API Reference](#).

## Reference

### Structure Developer Reference

- [Structure Java API Reference](#)
  - [Structure API Versions](#)

- Structure Plugin Module Types
  - Synchronizer Module
- Structure REST API Reference
  - Structure Resource - GET
  - Forest Resource - GET
  - Forest Resource - POST

## Structure Java API Reference

Structure API Reference for the latest version: <http://almworks.com/structure/javadoc/latest>

You can download javadocs from the Maven repositories into your IDE.

Check out information about [Structure API Versions](#) to select the correct API artifact, and you can also download Javadoc JARs there.

## Structure API Versions

Current Versions

Version	Supported JIRA Versions	Supported Structure Versions	OSGi Import Version	Release Date
<a href="#">2.0.0 Javadocs</a>	JIRA 4.3 – JIRA 4.4.*	1.3.*	"[2.0,3)"	2011-10-22
<a href="#">1.0.0 Javadocs</a>	JIRA 4.2.*	1.3.*	"[1.0,2)"	2011-10-22

[Latest Version Javadocs](#)

To see how to include the API jar into your project dependencies, read about [Accessing Structure from Your Plugin](#).

## Version Compatibility

Versioning of the API artifact follows these generally accepted rules:

- Major version is increased when the client code – your code – might not compile with the new version.
- Minor version is increased when new methods are added to the API (so your code might break if you downgrade to a lower minor version).
- Micro version is changed when there's no impact on the compatibility.

## Getting Versions

The API jars can be downloaded from the public Maven repositories. This is the recommended way.

If you can't download API jars from Maven repository for any reason, you can download them from this page and install into your local Maven repository:

```
mvn install:install-file -DgroupId=com.almworks.jira.structure -DartifactId=structure-api
-Dversion=2.0.0 -Dpackaging=jar -Dfile=structure-api-2.0.0.jar
```

Name	Size	Creator	Creation Date	Comment
Java Source structure-api-2.0.0-sources.jar	72 kB	Igor Sereda	Oct 21, 2011 17:46	
Java Source structure-api-2.0.0.jar	93 kB	Igor Sereda	Oct 21, 2011 17:46	
Java Source structure-api-2.0.0-javadoc.jar	445 kB	Igor Sereda	Oct 21, 2011 17:46	
Java Source structure-api-1.0.0-sources.jar	72 kB	Igor Sereda	Oct 21, 2011 17:45	
Java Source structure-api-1.0.0.jar	92 kB	Igor Sereda	Oct 21, 2011 17:45	
Java Source structure-api-1.0.0-javadoc.jar	444 kB	Igor Sereda	Oct 21, 2011 17:45	

## Structure Plugin Module Types

The following module types are added by the Structure plugin:

- `structure-synchronizer` defines a new synchronization type

## Synchronizer Module

Synchronizer module allows you to plug additional synchronizers into Structure.

### Module description sample

Here's a template of a synchronizer module declaration, and explanation of the parameters follows.

```
<structure-synchronizer key="module-key" order="100"
  class="com.company.your.plugin.sync.SyncClass">
  <label key="label.i18n.key">Name of Synchronizer</label>
  <description key="description.i18n.key">Description of Synchronizer</description>
  <rules key="rules.i18n.key">Large text to be shown at the top of synchronizer's configuration
  page.</rules>

  <resource type="velocity" name="form" location="/templates/myplugin/sync-form.vm"/>

  <inbound>
    <resync-description key="inbound.resync.i18n.key"/>
  </inbound>
  <outbound>
    <resync-description key="outbound.resync.i18n.key"/>
  </outbound>
</structure-synchronizer>
```

Element	Required	Description
structure-synchronizer	Yes	The module descriptor
structure-synchronizer/@key	Yes	Unique module key within the plugin
structure-synchronizer/@order	Yes	Order of the synchronizer among other synchronizers, whenever a list of synchronizers is present
structure-synchronizer/@class	Yes	The class that implements the synchronizer. Must implement <a href="#">StructureSynchronizer</a> - better extend <a href="#">AbstractSynchronizer</a> or <a href="#">AbstractIssueListeningSynchronizer</a>
structure-synchronizer/label	Yes	The name of the synchronizer
structure-synchronizer/description	No	Description of the synchronizer
structure-synchronizer/rules	No	Possibly large text, that is shown at the top of synchronizer configuration page.
structure-synchronizer/resource[@name="form"]	Yes	A velocity template that contains form for the synchronizer parameters.
structure-synchronizer/inbound	Either inbound, outbound or both	If present, it means that Resync can run "from outside into Structure"
structure-synchronizer/inbound/resync-description		
structure-synchronizer/outbound	Either inbound, outbound or both	If present, it means that Resync can run "from Structure to outside"
structure-synchronizer/outbound/resync-description		

## Structure REST API Reference

Structure REST API is yet under active development. The functionality available through REST is not complete, yet it allows to work with the structures.

The API is also not stable, although we're not seeing major changes coming to the main resources.

## General Notes

### REST Resource Address

Structure REST API resources have the URL

```
BASEURL/rest/structure/1.0/NAME
```

where BASEURL is the base JIRA address (`http://localhost:2990/jira` being standard base URL for development environment) and NAME is the name of the resource.

### Authentication

Authentication is done via standard JIRA authentication engine and supported by cookies. When accessing REST API from a remote application, you may need to set up the session first by calling JIRA authentication REST resource. (You don't need to do that if you access Structure REST API from a JavaScript on a page from the same JIRA instance.)

Most read operations are available to non-authenticated access (subject to permission checks for the anonymous user). Most mutation operations are available to authenticated users only.

### REST Resources and Operations

- [GET /structure/id](#) - get information about a structure
- [GET /structure/id/forest](#) - get a forest
- [POST /structure/id/forest](#) - make changes to a forest

### Structure Resource - GET

#### Request

```
GET $baseUrl/rest/structure/1.0/structure/$id
```

Returns basic data about a structure.

#### Parameters:

\$id	the ID of the structure
------	-------------------------

#### Response

```
{
  "id" : 109,
  "name" : "My Pretty Structure",
  "description" : "This is a structure that I like very much.",
  "readOnly" : true,
  "editRequiresParentIssuePermission" : true,
  "error" : "Something terrible happened"
}
```

#### Response Fields:

id	the ID of the structure
name	the name of the structure
description	the description on the structure

readOnly	true if the user has only READ access level to the structure (not present in the response otherwise)
editRequiresParentIssuePermission	true if the <a href="#">Require Edit Issue Permission on Parent Issue</a> flag is set on this structure
error	error text, if any error happened - if the user has no access to the structure, for example

## Forest Resource - GET

### Request

```
GET $baseUrl/rest/structure/1.0/structure/$id/forest
GET $baseUrl/rest/structure/1.0/structure/$id/forest?root=$root
```

Returns the hierarchical issue list (forest) of the specified structure.

### Parameters:

\$id	<i>required</i>	the ID of the structure
\$root	<i>optional</i>	the ID of the pinned issue for the <a href="#">Fixed Structure View</a>

### Response

```
{
  "structure":100,
  "version":2981,
  "root":10001,
  "formula":"10009:0,10008:1,10006:2,10002:3,10001:4,10062:4,10070:3,10071:2,10041:2"
}
```

### Response Fields:

structure	the ID of the structure
version	the version of the forest
root	the pinned issue if the fixed structure view has been requested (absent otherwise)
formula	a string representation of the issue forest

### Forest Formula

The forest of issues is represented by a string formula, which consists of pairs "*issue ID:depth*", delimited by a comma. Each pair represents a row in the fully expanded view of the forest.

```
ForestFormula ::= NodeSequence
NodeSequence ::= Node [ "," NodeSequence ]
Node ::= IssueID ":" Depth
IssueID ::= Integer
Depth ::= Integer
```

## Forest Resource - POST

### Request

```
POST $baseUrl/rest/structure/1.0/structure/$id/forest
```

```
{
  "base":0,
  "root":0,
  "actions":[
    {
      "action":"move",
      "issue":13212,
      "under":10037,
      "after":13210
    },
    {
      "action":"delete",
      "issue":13213
    },
    {
      "action":"add",
      "issue":10410,
      "under":10092,
      "after":0
    }
  ]
}
```

Updates the hierarchical issue list (forest) by applying the specified actions.

#### Parameters:

\$id	<i>required</i>	The ID of the structure.
base	<i>required</i>	The base version for the update reply from the server <i>Set to 0. Detailed explanation of this parameter will be published later.</i>
root	<i>required</i>	The ID of the pinned issue for the <a href="#">Fixed Structure View</a> , used only to create the reply.
actions	<i>required</i>	Array of actions to be applied to the forest, must contain at least one action. The actions are applied in the specified order.

#### Possible actions

The `actions` array may contain the following JSON objects:

##### **Move action**

action	"move"
issue	the ID of the issue to move
under	the new parent for the moved issue, or 0 to put it at the top level
after	the sibling issue (issue under the same parent) that must come before the moved issue, or 0 to place the issue as the first child issue

If the moved issue contains sub-issues, the whole sub-tree will be moved.

This action object corresponds to the `moveSubtree` method of the [ForestAccessor](#) interface.

##### **Add action**

action	"add"
issue	the ID of the issue to add
under	the new parent for the moved issue, or 0 to put it at the top level
after	the sibling issue (issue under the same parent) that must come before the moved issue, or 0 to place the issue as the first child issue

The new issue will be added to the structure at the position specified by (`under`, `after`) coordinates.

This action object corresponds to the `addIssue` method of the [ForestAccessor](#) interface.

#### Delete action

action	"delete"
issue	the ID of the issue to delete

The issue will be removed from the forest. (The issue itself will not be changed.)

If the removed issue contains sub-issues, the whole sub-tree will be removed.

This action object corresponds to the `removeSubtree` method of the [ForestAccessor](#) interface.

#### Response

```
{
  "update": {
    ...
  },
  "errors": [
    "error text"
  ]
}
```

#### Response Fields:

update	the object containing update to the client's structure specified by parameters <code>root</code> and <code>base</code> , absent if <code>base</code> is 0
errors	array of the error strings, absent if no errors

## Structure FAQ

### Frequently Asked Questions

- [Issues Not Added to a Structure when Using Links Synchronizer or Import](#)
- [Integration with Greenhopper](#)
- [Using Subtasks and Structure](#)
- [Difference from Sub-tasks](#)
- [Why Use Structure Plugin?](#)

## Issues Not Added to a Structure when Using Links Synchronizer or Import

### Question

I'm trying to use Links synchronizer (Import) with link type X but the issues are not added to the structure.

### Answer

Unless **Exclusive Mode** option is not turned on, Links synchronizer does not add issues to the structure.

- If you'd like to add **all** issues that have a link of type X to the structure, run Import with Exclusive Mode turned on.



Note that if you install a synchronizer (rather than run an Import) with **Exclusive Mode** flag, it will continuously work both ways - removing issues from the structure will cause links to be removed. If you run a [Resync](#) and choose direction from Structure to Links, then all links of type X between issues that are not in the structure (but from projects enabled for Structure) will be **deleted**. If you Resync an empty structure to links with Exclusive Mode on, you'll effectively remove all links of that type.

- If you'd like to add **some** of the linked issues to the structure, you need to first add them via [Search](#) or [Saved Filter Synchronizer](#), and



then run Import **without** Exclusive Mode flag, or install and resync a synchronizer.

See also: [Links Synchronizer](#)

## Integration with Greenhopper

### Question

We're using GreenHopper - are there any conflicts with Structure? Can we see the Structure's hierarchy in GreenHopper?

### Answer

You can use Greenhopper and Structure side by side. Structure plugin stores its data in a separate place, so it will not conflict with any other plug-in.

By default, Structure's hierarchy and issue order are independent from GreenHopper, but you can install a [GreenHopper Synchronizer](#) to have GreenHopper Rank synchronized with the position in the Structure and GreenHopper Epic/Theme field synchronized with the stories being under epics in the Structure.

GreenHopper is not yet able to display the hierarchy as Structure does, but with the Rank synchronization the issues will be positioned in the same order they are positioned in Structure.

See also: [GreenHopper Synchronizer](#)

## Using Subtasks and Structure

### Question

Should I disable sub-tasks to use Structure?

### Answer

Not necessarily. While Structure plugin can be a good replacement for sub-tasks, they can be used in parallel — for example, if you want to try Structure on a single project without affecting other JIRA users.

Structure treats sub-tasks as any other issues. You can also install a [Sub-Tasks Synchronizer](#), which makes sure that JIRA sub-tasks are positioned under their JIRA parent issues.

## Difference from Sub-tasks

### Question

How is issue hierarchy provided by Structure plug-in different from the standard sub-tasks?

### Answer

Sub-tasks have several major limitations:

- sub-tasks are only a one-level hierarchy;
- sub-tasks are separate issue types;
- sub-tasks always inherit project and security level from their parent task.

None of these limitations are present in Structure. At the same time, Structure plugin provides all the features that sub-tasks have, and more.

See also: [Structure Widget Overview](#)

## Why Use Structure Plugin?

### Question

Why use Structure plugin?

## Answer

Structure plugin makes Atlassian JIRA a perfect tool for planning, mind mapping or task decomposition. Project managers can make extensive plans and monitor progress using JIRA issues. Project team members can break down tasks into smaller pieces for more precise estimation and reporting.

See also: [Why Structure?](#)

## Structure Troubleshooting

- [HAR Network Report](#)
- [Performance Snapshot](#)

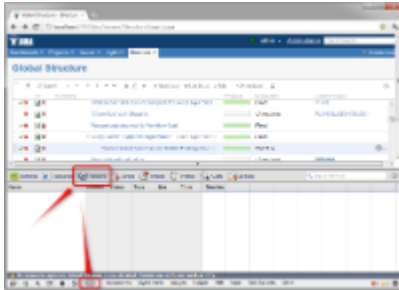
## HAR Network Report

HAR Network Report is something we (ALM Works Support) may ask you to collect, to help us understand a tricky problem that we could not reproduce.

**i** HAR stands for **HTTP Archive Format**, a text-based format for the log of network communications between a user agent (the browser) and a web server. You can also use this report with a [HAR Viewer](#) for in-depth analysis of your JIRA page load performance. (Be aware though that with an online viewer you may transfer sensitive or security-related information to a third party.)

### Collecting HAR Report with Google Chrome

1. Open a new Chrome window and navigate to the page where the problem happens.
2. Press **Ctrl+Shift+I** or use menu **Wrench | Tools | Developer Tools** to open a section with developer tools. Switch to the **Network** tab there. Make sure **All** tab is selected below.

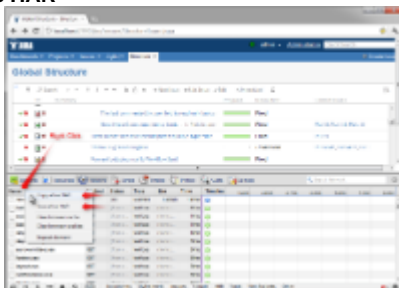


3. Reload the page by using **Ctrl+R** or clicking the Reload button. This will make Network tab log all network exchange during page load.



The network tab will start collecting information on network exchange automatically after it's opened. If you know that the problem is not related to the initial page load, you may skip this step to avoid adding extra data to the log. If unsure, reload the page to collect the full report.

4. Reproduce the problem being analyzed.
5. After the problem has been reproduced, **right-click** on the **Name** column in the Network tab and choose either **Save all as HAR** or **Copy all as HAR**



6. Paste the report into an e-mail to our support, or attach the saved .HAR file.

# Performance Snapshot

Performance Profile allows ALM Works support team to analyze performance-related problems on your JIRA server without direct access to it.

We are using Java Profiler product called [YourKit](#). In order to collect the profile, you'll need to download freely distributed "agent" library, connect it to your JIRA instance and capture a performance snapshot. You will need to purchase a license from YourKit only if you want to analyze the captured profile yourself.



No special knowledge is required to collect the performance profile, but being familiar with using the command-line on the server that runs JIRA helps.

## 1. Download Profiling Agent

Download the ZIP with profiling agent from here: [jira-profiler-v1-yjp956.zip](#) md5sum e3ea2b72ef4b22584c641425275050d0

Unpack the downloaded ZIP file into the directory where you have JIRA installed (**not** JIRA home!). This will create `<jira_install>/profiler` directory under your JIRA installation path.



You can unpack the profiler into any other directory, but this instructions and our scripts assume that the profiler is unpacked into JIRA install dir.

If you will be able to restart JIRA before profiling, this is all you need — you can proceed to [restarting JIRA with Profiling](#).

### 1.1. Additional Download to Profile JIRA Without Restart

If you need to profile JIRA without restarting it first (and assuming it is not already started with a profiler agent), you will need to download full distribution of the YourKit Java Profiler:

1. Open <http://yourkit.com/download/index.jsp>
2. Click on **ZIP Archive** type of download - **NOT** the installer! ZIP archive is typically downloaded under "Solaris" section - it is the correct link even if you run JIRA on Windows.
3. License key is not required for our purpose! Do not request evaluation license. (Unless you intend to do an evaluation of YourKit, of course.)
4. Unpack the downloaded ZIP into `<jira_install>/profiler` – this is the directory created at step 1. Unpacking will create a sub-directory there - for example, `<jira_install>/profiler/yjp-9.5.6`.

## 2. Restart JIRA with Profiling



If you need to profile without restart, skip this step.



The following instruction is provided for a standalone JIRA installation.

To restart JIRA with profiling, you need to pass additional options to Java that runs JIRA. This is done by editing `<jira_install>\bin\setenv.bat` on Windows or `<jira_install>/bin/setenv.sh` on a Unix-based OS and pointing Java to a profiler agent that you have unpacked at step 1.

1. Find out which profiler agent to use.
  - a. Look into `<jira_install>/profiler/bin` directory. Typically there will be two sub-directories for your operating system: 32-bit and 64-bit. The bitness must match the bitness of JVM that runs JIRA. You can verify which Java your JIRA runs on if you open **Administration | System Info** in JIRA and look for "Java VM". If it mentions "64-Bit", then JIRA runs on a 64-bit Java.
  - b. Note the name of the subdirectory under `profiler` directory that corresponds to the bitness of target JVM: it may be `win64` or `linux-x86-32` or something like that.
2. Edit `setenv` script:
  - a. On Windows, set or append the following parameters to JVM\_SUPPORT\_RECOMMENDED\_ARGS in `<jira_install>\bin\setenv.bat` (following is a single long line):

```
set
JVM_SUPPORT_RECOMMENDED_ARGS=-agentlib:%~dp0.\profiler\bin\win64\yjpgagent=port=10001,onlylc
-XX:MaxPermSize=500m
```

- b. On other OS, set or append the following parameters to `JVM_SUPPORT_RECOMMENDED_ARGS` in `<jira_install>/bin/setenv.sh` (following is a single long line):

```
JVM_SUPPORT_RECOMMENDED_ARGS="-agentpath:`dirname  
\"$0\"`/../../profiler/bin/linux-x86-64/libyjpagent.so=port=10001,onlylocal,dir=`dirname  
\"$0\"`/../../profiler/snapshots,delay=20000 -XX:MaxPermSize=500m"
```

3. Note that in the lines above, you should change **win64** or **linux-x86-64** to the name of the directory where the correct profiler agent for your OS/Java is located.
4. You may also need to change **port=10001** to make profiling agent listen on some other TCP port - in case port 10001 is already taken.
5. Stop JIRA and start it again.
6. Watch `<jira_install>/logs/catalina.out` for YourKit message like *[YourKit Java Profiler 9.5.6] Loaded*.



Use Copy & Paste to copy the parameters and then edit them in the `setenv.sh`



If the parameters are set incorrectly, JIRA start may fail. Verify that you have specified the agent directory correctly. Also verify that `<jira_install>` directory path does not contain spaces.



Profiler agent will use directory `<jira_install>/profiler/snapshots` to write performance snapshots - it must be write-accessible to the JIRA process.

Now you can proceed to [Running Profiling Session](#).

### 3. Attach Profiler Agent to JIRA without Restarting



If you have restarted JIRA with profiling, skip this step.



If possible, restart JIRA with profiling instead of attaching profiler agent on the fly.

You will need the full distribution of YourKit downloaded at step 1.1. You will need to run a Java program as specified below - with the same version of Java that JIRA runs on. We assume that it is in your `PATH` variable in the command-line, but if it's not - you need to specify a full path to java.

1. Find out the process ID of the process that runs JIRA. You can use `jps` command from the Java distribution.
2. Find out the location of JDK (Java Development Kit). If you don't have JDK installed (only JRE), this procedure won't work. Typically JDK home is stored in the command-line environment variable `JAVA_HOME`.
3. Change current directory to `<jira_install>/profiler/yjp-9.5.6`. (You may have a different version of yjp.)
4. Run the following command, substituting JIRA process ID instead of **PID**.
  - a. On Windows:

```
java -cp lib\yjp.jar;%JAVA_HOME%\lib\tools.jar com.yourkit.Main -attach PID  
port=10001,onlylocal,dir=<jira_install>\profiler\snapshots
```

Replace `<jira_install>` with the full path of the JIRA installation folder.

- b. On other OS:

```
java -cp lib/yjp.jar:$JAVA_HOME/lib/tools.jar com.yourkit.Main -attach PID  
port=10001,onlylocal,dir=`pwd`/../../snapshots
```

The command should output something like this:

```
Attaching to process 60108 using options port=10001,onlylocal,dir=..\snapshots
The profiler agent has attached. Waiting while it initializes...
The agent is loaded and is listening on port 10001.
You can connect to it from the profiler UI.
```

## 4. Running Profiling Session

To successfully run a profiling session, you need to have JIRA running with a profiling agent, as explained above. The agent does not add much overhead when being idle — it sits there waiting for your commands to start a profiling session.

### 4.1. General Procedure

The profiling session is controlled by sending commands to the profiling agent (within the JIRA process). The program that is used to send the commands is `yjp-controller-api-redis.jar`, located in `<jira_install>/profiler`. The common format for running this program is:

```
java -jar yjp-controller-api-redis.jar localhost 10001 <command>
```

The `<command>` is replaced with some actual command, and if you changed the default port of the agent from 10001 to something else, you need to specify that port number here instead of 10001. This command should be run from `<jira_install>/profiler` directory.



We are assuming that `java` is on your PATH. If not the case, use the full path to `java` executable.

### 4.2. CPU Performance Analysis

If JIRA is unresponsive or burns CPU extensively, you can run CPU analysis session.

1. Start session with the following command:

```
java -jar yjp-controller-api-redis.jar localhost 10001 start-cpu-sampling
```

2. Let JIRA work for some time. If needed, take a specific action that causes the problem to manifest.
3. Stop session and record a snapshot:

```
java -jar yjp-controller-api-redis.jar localhost 10001 capture-performance-snapshot
```

## 5. Sending the Snapshots to Support Team

By default, snapshots are written into `<jira_install>/profiler/snapshots` directory. Locate it and create a ZIP archive of all relevant snapshot files. If the ZIP is less than 10 Megabytes, it's ok to send it to us by e-mail.

If the ZIPPed snapshot is 10 MB or larger, you need to use FTP to send it over to us:

1. Use any FTP client (`ftp` or `lftp` from the command line).
2. Connect to host `f.almworks.com`
3. Use login name `almftp` and password `almftp`
4. Upload files to the root folder.
5. After the upload is finished, please send us an e-mail with a notification that you have uploaded the snapshots.



You will not be able to list or download files from that FTP, and your FTP client may show errors about that. That's ok and should not prevent you from uploading snapshots.

## 6. After Profiling Session

You may want to continue running JIRA with the profiling agent loaded, since it does not product much overhead. Make sure you have stopped all the monitoring.

For a safer / cleaner environment, you can restart JIRA with the profiling options in `setenv` script commented out.