Setting up JIRA filters and dashboards

Working with filters

A JIRA filter is a saved query that can be used on its own, or more commonly, to populate a Kanban board or dashboard gadget. JIRA filters are written in JQL (JIRA Query Language).

You can create a filter from any query, built using Basic or Advanced Search for example, and then share this with ***whatever user group is applicable in your instance*** so that it can be viewed by others (needed in order for your dashboard to display to everyone!).

You can also use an existing filter as a starting point, then edit the query and save with a new name (and remember to share with ***the relevant user group***).

Common JQL expressions

assignee

Issues that are assigned to a user. Possible uses include:

Use	Description
`assignee = currentUser()`	Issues that are assigned to the current user (you). **Note:** If you use assignee = currentUser() in a filter, and someone else runs the filter, they'll see their own issues instead. A workaround is to use your own username instead (for example, assignee = chris.butta).
`assignee != currentUser()`	Issues that are **not** assigned to the current user (you).
`assignee = username`	Issues that are assigned to username. If you type the first few letters of a username, JIRA will suggest a list of matching usernames.

ORDER BY

The order to present issues in. You can order issues by any field, such as assignee, updated, resolution, etc.

Sorting order can be ASC (ascending) or DESC (descending).

project

Issues in a given project. Possible uses include:

Use	Description
`project = "Project Name"`	Issues within a given JIRA project.
`project in ("Project Name 1", "Project Name 2")`	Issues within any of the given JIRA projects. This can contain an unlimited number of projects, separated by commas.
`project not in ("Project Name 1", "Project Name 2")`	Issues **not** within any of the given JIRA projects. This can contain an unlimited number of projects, separated by commas.

Note: The Project Name can be either:

- The project key (the letters in the ticket name - for example, the DOC in DOC-1987).
- The long form project name (for example, "Documentation").

resolution

Issues that have been closed with a given resolution. By default, open tickets have resolution = Unresolved.

status

Issues that are in a given status. Possible uses include:

Use	Description
`status = "Value Name"`	Issue status is Value Name.
`status in ("Value Name 1", "Value Name 2")`	Issue status can be any of the given values. This can contain an unlimited number of values, separated by commas.
`status not in ("Value Name 1", "Value Name 2")`	Issue status is **not** any of the given values. This can contain an unlimited number of values, separated by commas.

"Value Name" can contain any valid workflow status, including custom ones, such as "In Tech Review" for the DOC project.

statusCategory

Syntax:

```
statusCategory = "Value Name"
```

Issues that are in a given status. This is less flexible than the status expression above. Possible values only include "To Do", "In Progress", and Done.

Note: For values that consist of two or more words, use quotes. Values that consist of a single word don't need quotes (but you can use them anyway).

watcher

Issues that are being watched by a user. Possible uses and syntax are the same as assignee above.

Example filters

Filter syntax

To connect expressions within filters, use the AND or OR operators:

- a AND b means the filter will find issues for which both a and b are true.
- a OR b means the filter will find issues for which either a or b is true. They can both be true, but they can't both be false.

Examples

```
assignee = currentUser() AND resolution = Unresolved AND status = "In Review" ORDER BY updated DESC
```

Breakdown:

Expression	Description
`assignee = currentUser()`	Issues assigned to the current user.
`resolution = Unresolved`	Issues that are still open (unresolved).
`status = "In Review"`	Issues that are in the In Review status.
`ORDER BY updated DESC`	Issues will be sorted by the date they were last updated, starting with the ones that were updated most recently.

You can then display these filter results in a gadget and pull relevant information, such as the editor assigned to review each ticket

---

```
assignee = currentUser() AND resolution = unresolved AND project in ("Doc Tools | Development", "Docs Projects") ORDER BY duedate DESC
```

Breakdown:

Expression	Description
`assignee = currentUser()`	Issues assigned to the current user.
`resolution = Unresolved`	Issues that are still open (unresolved).
`project in ("Doc Tools | Development", "Docs Projects")`	Issues in the "Doc Tools | Development" and "Docs Projects" projects.
`ORDER BY duedate DESC`	Issues will be sorted by the due date, in descending order.

Working with dashboards

Adding Filter Results gadgets to your JIRA dashboard

To add a Filter Results gadget to your JIRA dashboard, follow these steps:

1. In the top-right corner of your dashboard, click the **Add Gadget** button.
2. JIRA will inform you that more gadgets are available. Click **Load all gadgets**.
3. Scroll down to the **Filter Results** gadget or search for it from the search box on the left.
4. Click the **Add Gadget** button to the right of the **Filter Results** gadget.

Jira will add the gadget to your dashboard in the background. To configure it, close the **Add Gadget** window.

Note: If you want other users to see your filter results, you must share your filter with the **jira-users** group. To do this, follow these steps:

1. Access the **Filter** view (click the JIRA icon in the top-left corner of the screen, then click **Filters**).
2. Select your filter from the list.
3. On the right of the filter name, click **Details**, then select **Edit Permissions**.
4. In the **Access** section of the **Edit Filter** window, add the **jira-users** group.
5. Click **Save.**