

Parse summary for setting issue priority

On this page

- [Features used to implement the example](#)
- [Example: Parse summary for setting issue priority](#)
- [Other examples of that function](#)
- [Related Usage Examples](#)

Features used to implement the example

- [Parse field for extracting data](#)

Example: Parse summary for setting issue priority

Let's see a configuration that allows setting priority of an issue when we write the priority name between brackets in the issue summary. This can be particularly useful for issues created by email.

We insert [Parse field for extracting data](#) post-function in "Create Issue" transition (after "Creates de issue originally" post-function) using the following configuration:

Source field: Field containing the text that will be parsed for extracting certain value(s).	Summary - [Text]
Leading delimiter: Text literal or pattern expressed as a regular expression that should be matched immediately before the value to be extracted. By default, delimiters are not included in returned values.	[☒ text literal ☐ text literal (ignoring case) ☐ regular expression ☐ regular expression (ignoring case) Field code injector: Summary - [Text] - %{00000} Dynamic patterns can be created through insertion of field codes that will be replaced with the corresponding field values.
Format of the value to be extracted: Format of the expected value to be extracted. Some pattern types require a specification of the format to be entered.	☐ any text between leading and trailing marks (i.e., no pattern) ☐ number ☐ url ☐ email Number, url and email formats are filtered by regular expressions that match the vast majority of possible values, but there might be valid values that don't match these regular expressions. If avoiding any undesired value rejection is a priority, any text should be selected. ☐ regular expression ☒ regular expression (ignoring case) ☐ date-time (specifying date-time formats: Date and Time Patterns) Format types regular expression, regular expression (ignoring case) and date-time require a format specification to be entered. <u>Format specification:</u> trivial minor major critical blocker Field code injector: Summary - [Text] - %{00000} Dynamic patterns can be created through insertion of field codes that will be replaced with the corresponding field values.
Trailing delimiter: Text literal or pattern expressed as a regular expression that should be matched immediately after the value to be extracted. By default, delimiters are not included in returned values.	] ☒ text literal ☐ text literal (ignoring case) ☐ regular expression ☐ regular expression (ignoring case) Field code injector: Summary - [Text] - %{00000} Dynamic patterns can be created through insertion of field codes that will be replaced with the corresponding field values.
Target field: Field that will be set with the extracted value(s).	Priority - [Issue priority] ☐ Don't overwrite target field if it's already set. ☐ Include the leading delimiter in the output. ☐ Include the trailing delimiter in the output.
Ocurrences to be extracted: Values to be extracted in case that more than one data matching extraction patterns are found.	☒ first occurrence in source field ☐ last occurrence in source field ☐ all occurrences (values are returned as a comma separated list of values)
Conditional execution: Optional boolean expression that should be satisfied in order to actually execute the post-function. (Syntax Specification)	1 Collection of Examples [Line 1 / Col 1] Leave the field empty for executing the post-function unconditionally. <u>Logical connectives:</u> and, or and not. Alternatively you can also use &, and !. <u>Comparison operators:</u> =, !=, >, >=, < and <=. Operators in, not in, any in, none in, ~ and != can be used with <i>strings</i>, <i>multi-valued fields</i> and <i>lists</i>. <u>Logical literals:</u> true and false. Literal null is used with = and != to check whether a field is initialized, e.g. {00012} != null checks whether <i>Due Date</i> is initialized. String Field Code Injector: Summary - [Text] - %{00000} Numeric/Date Field Code Injector: Original estimate (minutes) - [Number] - {00068} Check Syntax
Run as: Select the user that will be used to execute this feature. JIRA will apply restrictions according to the permissions, project roles and groups of the selected user. Current user User defined by a field. Input a specific user.	

Note that:

- Value format is regular expression: trivial|minor|major|critical|blocker
- Leading delimiter is regular expression: [

- Trailing delimiter is regular expression:]

Once configured, post-function "Create Issue" looks like this:

The following will be processed after the transition occurs

Add post function

1. Write into **Priority** a piece of information extracted from **Summary**.
Format of value to be extracted is string matching the following **regular expression ignoring case**:

trivial|minor|major|critical|blocker

Leading delimiter is the following **string literal**:

[

Trailing delimiter is the following **string literal**:

]

In case of multiple matches extract only **first** occurrence.
This feature will be run as user in field **Current user**.

An example of the result of the execution of this post-function:

Human Resources / HR 79

Example [critical]

Edit Comment Assign More

Start Progress Resolve Issue Workflow Admin

Export

Details

Type: Bug

Priority: Critical

Affects Version/s: None

Component/s: None

Labels: None

Status: OPEN (View Workflow)

Resolution: Unresolved

Fix Version/s: None

People

Assignee: Administrator

Reporter: Administrator

Votes: 0

Watchers: 1 Stop watching this issue

Dates

Created: Just now

Updated: Just now

Agile

View on Board

HipChat discussions

Do you want to discuss this issue? Connect to HipChat.

Connect Dismiss

Description

Click to add description

Activity

All Comments Work Log History Activity

There are no comments yet on this issue.

Comment

Other examples of that function

Page: [Parse description for creating issue links](#)
Page: [Parse summary for setting "Due date"](#)
Page: [Parse summary for setting issue priority](#)
Page: [Parsing text from last comment and appending it to issue's summary](#)

Related Usage Examples

- [Creating a Jira Service Desk internal comment](#)
 - [example](#)
 - [post-function](#)
- [Limit the number of hours a user can log per day](#)
 - [example](#)
 - [validator](#)
 - [post-function](#)
 - [work-log](#)
- [Using project properties to calculate custom sequence numbers](#)

- example
 - post-function
 - calculated-field
 - project-properties
- Set a date based on current date
 - example
 - post-function
- Setting the priority depending on the multiplication of custom fields
 - example
 - calculated-field
 - post-function
- Parse Email addresses to watchers list
 - example
 - post-function
- Set the assignee based on a condition
 - example
 - post-function
- Create a dynamic set of sub-tasks based on checkbox selection with unique summaries
 - example
 - post-function
 - custom-field
 - sub-task
- Create a static set of sub-tasks with unique summaries
 - example
 - post-function
- Triage Jira Service Desk email requests (Move issues)
 - example
 - post-function
 - move
 - transition-issue
- Moving story to "In Progress" when one of its sub-tasks is moved to "In Progress" (Transition issues)
 - example
 - post-function
 - transition
- Transition sub-tasks when parent is transitioned
 - example
 - post-function
 - sub-task
 - transition
 - outdated
- Transition only a sub-task among several ones
 - example
 - post-function
 - sub-task
 - transition
 - outdated
- Moving sub-tasks to "Open" status when parent issue moves to "In Progress"
 - example
 - post-function
 - sub-task
 - transition
 - outdated
- Moving story to "Ready for QA" once all its sub-tasks are in "Ready for QA" status
 - example
 - post-function
 - sub-task
 - transition
 - outdated