Regular expression renderer

This function has been **renamed** with the **JWT 3.0** release.

Find the new documentation at:

Regular expression renderer

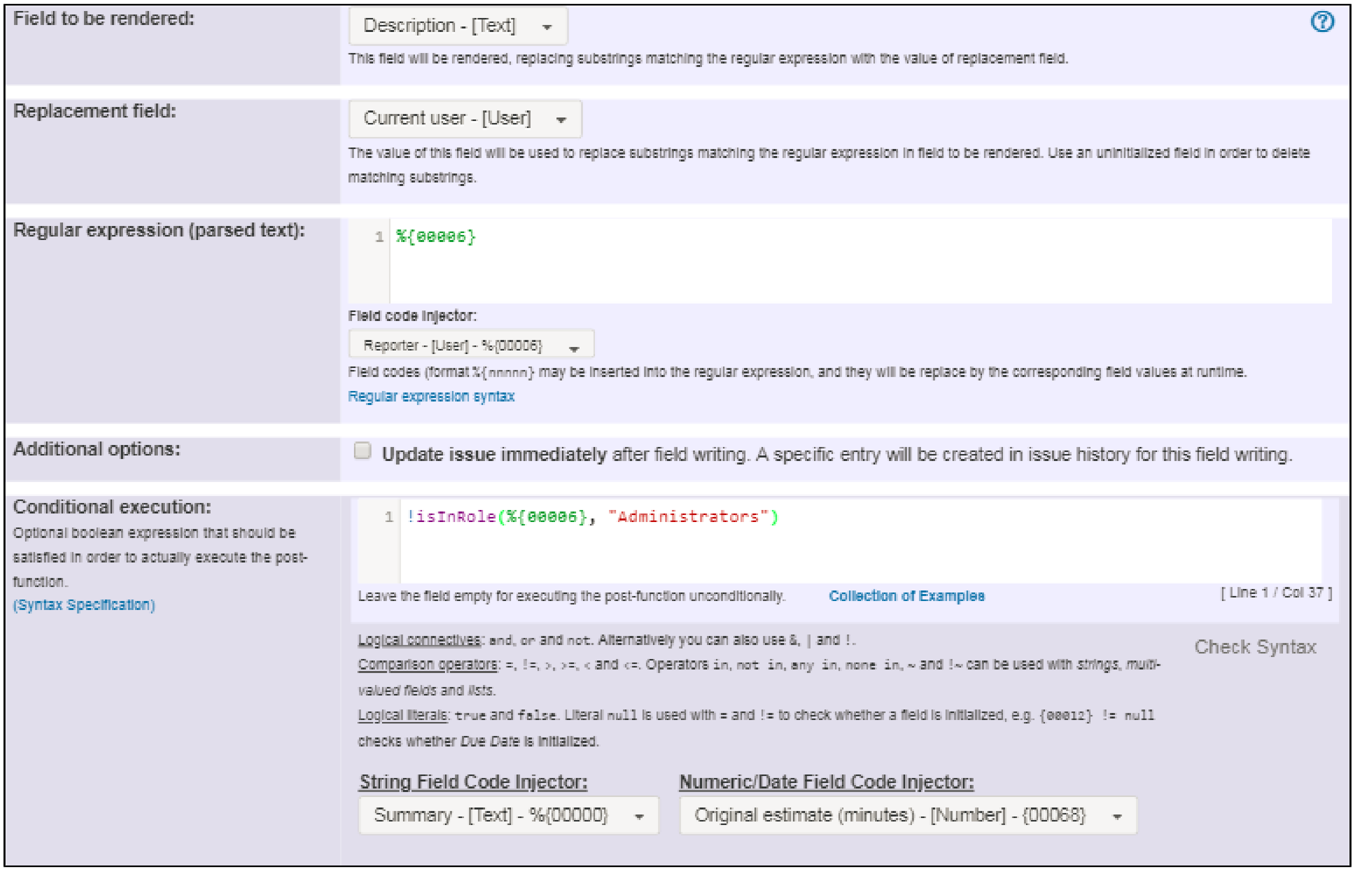
On this page

Purpose

This post-function can be used for replacing a substring in the content of another field. The replacing string is defined by the content of another field. You can implement this same behavior using more advanced features like **Copy parsed text to a field** or **Format field value** post-functions.

Example: Replace reporter with current user if he does not belong to the administrator project role

If reporter doesn't belong to role **Administrators**, then replace the username of the **Reporter** in the issue **Description** with the one of the **Current User**:



Field to be rendered: Description - [Text]

This field will be rendered, replacing substrings matching the regular expression with the value of replacement field.

Replacement field: Current user - [User]

The value of this field will be used to replace substrings matching the regular expression in field to be rendered. Use an uninitialized field in order to delete matching substrings.

Regular expression (parsed text):

```
%{00006}
```

Field code injector: Reporter - [User] - %{00006}

Field codes (format %{nnnnn} may be inserted into the regular expression, and they will be replace by the corresponding field values at runtime.

Regular expression syntax

Additional options: □ **Update issue immediately** after field writing. A specific entry will be created in issue history for this field writing.

Conditional execution:

Optional boolean expression that should be satisfied in order to actually execute the post-function.

(Syntax Specification)

```
!isInRole(%{00006}, "Administrators")
```

Leave the field empty for executing the post-function unconditionally. Collection of Examples [Line 1 / Col 37]

Logical connectives: and, or and not. Alternatively you can also use &, | and !.

Comparison operators: =, !=, >, >=, < and <=. Operators in, not in, any in, none in, ~ and !~ can be used with *strings, multi-valued fields* and *lists*.

Logical literals: true and false. Literal null is used with = and != to check whether a field is initialized, e.g. {00012} != null checks whether *Due Date* is initialized.

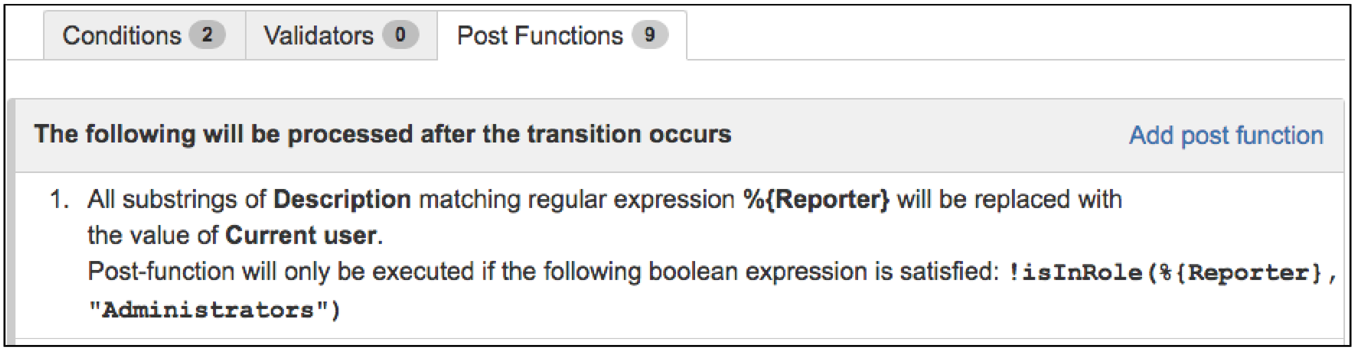
String Field Code Injector: Summary - [Text] - %{00000}

Numeric/Date Field Code Injector: Original estimate (minutes) - [Number] - {00068}

Note that:

- **%{00006}** is field code for **Reporter**

Once configured, the post-function looks like this:



The following will be processed after the transition occurs

1. All substrings of **Description** matching regular expression **%{Reporter}** will be replaced with the value of **Current user**.
   Post-function will only be executed if the following boolean expression is satisfied: `!isInRole(%{Reporter}, "Administrators")`

Usage Examples

Page: Add watchers from a part of the issue summary: "Summary_text - watcher1, watcher2, watcher3, ..."

Related Features

- **Copy parsed text to a field**: with this post-function much more complex text formating operation can be implemented. To do it use **advanced parsing mode**.
- **Parse field for extracting data**
- **Format field value**