Examples of String List expressions

On this page

- Examples of String List expressions
- String List Literals
- · Reading Fields from Issue Lists
- Reading Multi-valued Fields as String Lists
- Reading Multi-valued Fields from Issue Lists
- · Parsing a String into a String List

Examples of String List expressions

This page presents a collection of string list expressions valid for the Expression Parser.

String list is one of the data types that can be returned by an expression in Jira Workflow Toolbox.

String List Literals

The syntax of a string list literal is a comma separated list of strings in double quotes written inside brackets. Examples:

Expression	Example of Returned Value	Notes
[]	Empty list.	-
["red", "blue", "green"]	String list with 3 strings.	-
["Ada Lovelace", "John von Neuman", "Alan Turing"]	String list with 3 strings.	-

Reading Fields from Issue Lists

When we read field values from issues using field codes for string values (format %{nnnnn}), the returned data type is a string list.

Expression	Example of Returned Value	Notes
fieldValue(%{00003}, subtasks())	List of usernames of the assignees of the current issue's subtasks. It may contain repeated usernames.	%{00003} = Assignee
<pre>distinct(fieldValue(%{00003}, subtasks()))</pre>	List of usernames of the assignees of the current issue's subtasks. It will not contain repeated usernames, because of the usage of function distinct().	%{00003} = Assignee
<pre>fieldValue(%{00017}, linkedIssues("blocks, is cloned by"))</pre>	List of priorities of those issues linked to current issue through issue link types "blocks " and "is cloned by".	%{00017} = Priority

You will find very useful these examples of Issue List expressions in order to select the issues you need to read fields from.

Reading Multi-valued Fields as String Lists

Fields codes of multi-valued fields like Affected versions, Fixed versions, Components, Labels, Attachments, Watchers, Request Participants, Multi-Select List, Check List, Multi-User Picker, Multi-Group Picker and Multi-Version Picker custom fields, return a **string** with a **comma separated list of values**. We can convert this string into a string list by means of function toStringList().

Expression	Example of Returned Value	Notes
toStringList(% {00074})	If Fixed versions field code (i.e., $%\{00074\}$) returned "1.1, 1.2, 1.3", the expression would return ["1.1", "1.2", "1.3"]	%{00074} = Fixed versions

toStringList(% {00094})	String list with the components currently selected in field Components .	%{00094} = Components
-------------------------	---	-----------------------

Reading Multi-valued Fields from Issue Lists

Expression	Example of Returned Value	Notes
<pre>distinct(toStringList(toString (fieldValue(%{00094}, subtasks())), ","))</pre>	Returns a string list with all the components present in the sub-tasks of current issue. The list obtained doesn't contain duplicates, thanks to function distinct()	%{00094} = Components

Parsing a String into a String List

We can use the following functions in order to parse a string into a string list:

Function	Explanation
toStringList(string s, string se parators) : string list	INPUT: String with a list of tokens separated by one or more characters. OUTPUT: A string list with tokens in argument s separated by characters in argument separators. Example: toStringList("red, orange, yellow; green; blue; purple", ",;") returns the following string list: ["red", "orange", "yellow", "green", "blue", "purple"].
findPattern(string s, string reg exp) : string list	Returns a string list with all substrings in argument s matching regular expression in string argument regexp . Example: findPattern("Between 1900 and 2000 world population increase from 1.5 to 6.1 billions.", "\\d+(\\.\\d+)?") returns ["1900", "2000", "1.5", "6.1"].
findPatternIgnoreCase(string s , string regexp) : string list	Returns a string list with all substrings in argument s matching regular expression in string argument regexp . Evaluation of the regular expression is carried out in ignoring case mode. Example: findPatternIgnoreCase("Grass is Green and Sky is Blue.", "red green blue") returns ["Green", "Blue"].