Strings

On this page

Fixed values

- Texts or strings need to be written in **double quotes**, e.g., "This is a string literal."
- Operator + is used for **concatenating string**. e.g., "This is" + " a string." = "This is a string." .
- The **Escape character** is "\" . This character can precede any of the following characters: ", **\, n, r, t, f** and **b** in order to invoke an alternative interpretation.
  For example, if you want to introduce a double quote in a string literal you should precede it with escape character \ as in "The man said: `\"Hello!\"."` , where we are using escape character \ to write string Hello! in double quotes.

Variable values (field values)

Text / String field values can be inserted in expressions using field codes with format **%{...somefield}**, or **%{...somefield.i}** for referencing concrete levels in cascading select fields (i = 0 for base level).

> **Pro tip**
>
> For checking if a field has a value you can use `%{...somefield} = null` or `%{...somefield} != null`.
> For a concrete level in a **Cascading Select** or **Multi-Cascading Select** field, you should use `%{...somefield.i} = null` or `%{...somefield.i} != null`.

Any field type has a string value, so you can also use **%{...somefield}** to insert string values of fields of types: **Number, Date, Date-Time** and **Priority**.

String Functions

Function	Input	Returned value
trim(string **s**)	STRING	Returns a copy of the STRING without leading and trailing blanks (space and tab characters). Example: `trim(" Hello World! ")` returns **"Hello World!"**.
substring(string **s**, number **beginIndex**, number **endIndex**)	STRING	Returns a substring of the STRING beginning at index **beginIndex** and ending at **endIndex - 1**. Thus the length of the substring is **endIndex-beginIndex**. Example: `substring("smiles", 1, 5)` returns "**mile**".
toUpperCase (string **s**)	STRING	Returns STRING with all its characters converted to upper case. Example: `toUpperCase("heLLo WORLD!")` returns "**HELLO WORLD!**".
toLowerCase (string **s**)	STRING	Returns STRING **s** with all its characters converted to lower case. Example: `toLowerCase("heLLo WORLD!")` returns "**hello world!**".
capitalizeWords(string **s**)	STRING	Capitalizes all the whitespace separated words in STRING. Example: `capitalizeWords("heLLo WORLD!")` returns "**HeLLo WORLD!**".
capitalizeWordsFully(string **s**)	STRING	Converts all the whitespace separated words in STRING into capitalized words, that is each word is made up of a titlecase character and then a series of lowercase characters. Example: `capitalizeWordsFully("heLLo WORLD!")` returns "**Hello World!**".

replaceAll(string **s**, string **regexp**, string **replacement**)	STRING REGEX	Returns a copy of **s** where each substring matching the given **regular expression** **regexp** has been replaced with the given **replacement** string. Example: `replaceAll(" Hello World ", "\\s", "")` returns "**HelloWorld**" .
replaceFirst(string **s**, string **regexp**, string **replacement**)	STRING REGEX	Returns a copy of STRING where the first substring matching the given **regular expression** **regexp** has been replaced with the given **replacement** string. Example: `replaceFirst("Hello World", "l", "_")` returns "**He_lo World**" .
matches(string **s**, string **regexp**)	STRING REGEX	Returns a BOOLOEAN value **true** if string s matches **regular expression** **regexp**, otherwise returns **false**. Example: `matches("readme.txt", ".*\\.txt$")` returns **true** .
findPattern(string **s**, string **regexp**)	STRING REGEX	Returns a STRING [] 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 [] REGEX	Returns a STRING [] 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"]** .
findModify(string **s**, string **regexp**, string **replacement_expression**)	STRING REGEX	Returns a STRING like **s**, but where all substrings matching **regexp** have been replaced with the result of evaluating **replacement_expression** against each these substrings. Argument **text_expression** is an expression that returns a **string**, where **^%** represents each of the matching substrings, and **^** represents the order of appearance beginning with **1**. Example: `findModify("The cure for boredom is curiosity.", "[aeiou]", modulus(^, 2) = 1 ? toUpperCase(^%) : ^%)` returns **"ThE curE for bOredOm is cUriOsity."** .
findReplaceAll(string **s**, string **find**, string **replacement**)	STRING	Returns a STRING with content of argument **s** where every ocurrence of substring find has been replaced with string **replacement**. Example: `findReplaceAll("Goodbye my love, hello my friend.", "my", "your")` returns "**Goodbye your love, hello your friend.**" .
findReplaceAllIgnoreCase(string **s**, string **find**, string **replacement**)	STRING	Returns a STRING with content of argument **s** where every ocurrence of substring **find**, ignoring the case, has been replaced with string **replacement**. Example: `findReplaceAllIgnoreCase("Hello my love, hello my friend.", "hello", "Goodbye")` returns "**Goodbye my love, Goodbye my friend.**" .
findReplaceFirst(string **s**, string **find**, string **replacement**)	STRING	Returns a STRING with content of argument **s** where first ocurrence of substring **find** has been replaced with string **replacement**. Example: `findReplaceFirst("Goodbye my love, hello my friend.", "my", "your")` returns "**Goodbye your love, hello my friend.**" .
findReplaceFirstIgnoreCase(string **s**, string **find**, string **replacement**)	STRING	Returns a STRING with content of argument **s** where first ocurrence of substring **find**, ignoring the case, has been replaced with string **replacement**. Example: `findReplaceFirstIgnoreCase("Goodbye my love, hello my friend.", "My", "your")` returns "**Goodbye your love, hello my friend.**" .
length(string **s**)	STRING	Returns a NUMBER with the length of **s**. Example: `length("Star Wars")` returns **9** .

getAscii(number **code**)	NUMBER	Returns a STRING containing the symbol corresponding to a extended ASCII **code** (0 <= **code** <= 255). Example: `getAscii(65)` returns "**A**" .
similarity(string **s1**, string **s2**)	STRING	Returns a NUMBER value between 0 and 100 representing the percentage of similarity between two strings based on the **Jaro Winkler similarity algorithm** . **100** represents **full equivalence**, and **0** represents **zero similarity** between both string arguments. Examples: `similarity("Automation Toolbox for Jira", "Automation Toolbox for Jira")` returns **100** `similarity("Automation Toolbox for Jira", "Jira WorflowTolbox")` returns **97** `similarity("My Gym. Childrens Fitness", "My Gym Children's Fitness Center")` returns **92** `similarity("D N H Enterprises Inc", "D & H Enterprises, Inc.")` returns **91** `similarity("ABC Corporation", "ABC Corp'")` returns **92** `similarity("Hello World!", "Bye bye World!")` returns **69** `similarity("I caught a lizard", "This is my giraffe")` returns **51**
escapeHTML (string **s**)	STRING	Escapes the characters in a STRING using HTML entities. Example: `escapeHTML("<Français>")` returns "**< Français>**" .
unescapeHTML(string **s**)	STRING	Unescapes STRING containing entity escapes to a string containing the actual Unicode characters corresponding to the escapes. Example: `unescapeHTML(""bread" & "butter"")` returns **"\"bread\" & \"butter\""** .
wikiToHTML(string **s**)	STRING	Renders rich text wiki content of STRING into HTML. Example: `wikiToHTML("+Hello *world*!+")` return "** <ins>Hello world!</ins> **" .
htmlToTxt(string **s**)	STRING	Renders HTML content of STRING into plain text by removing all the html tags. Example: `wikiToHTML("<p>Hello <b>world</b>!</p>")`return "**Hello world!**" .

Examples

Input	Output
`"Hello" + " " + "world" + "."`	**Hello world.**
`trim(%{...summary})`	**Summary** of an issue **without leading** and **trailing blanks**
`%{...description} + "\nLAST USER: " + toUpperCase(%{...currentUser})`	**Description** of an issue and a new line with string "**LAST USER:** " and the **name of current user** in upper case.