Token Manipulation

Token in AdiIRC scripting are lists of strings that is separated by a single unique character.

Example

The numerical representation of space char in ASCII table is **32**, and **44** is for comma.

```
//set -e %q = which came first, the chicken or the egg?

; Will print 2, because it has a single comma "splitting" the text in two parts.
//echo -ag $numtok(%q, 44)

; Will print 8, because in this case are 8 strings separated by space char.
//echo -ag $numtok(%q, 32)
```

To help you to manipulate list of tokens, there is an extensive set of identifiers and commands.

$addtok

Added in 1.9.0

$addtok(text,token,C)

Adds a token to the end of text but only if it is not already in text.

$addtok *is case-insensitive, see* $addtokcs *for case-sensitive version.*

Parameters

text - The text to tokenize.
token - Token to add.
C - The ASCII value to tokenize by.

Example

```
; Returns a.b.c.d
//echo -ag $addtok(a.b.c,d,46)

; Returns a.b.c.d
//echo -ag $addtok(a.b.c.d,c,46)
```

$addtokcs

Added in 1.9.0

$addtokcs(text,token,C)

Adds a token to the end of text but only if it is not already in text.

$addtokcs *is case-sensitive, see* $addtok *for case-insensitive version.*

Parameters

text - The text to tokenize.
token - Token to add.
C - The ASCII value to tokenize by.

Example

```
; Returns a.b.c.D
//echo -ag $addtokcs(a.b.c,D,46)

; Returns a.b.c.d
//echo -ag $addtokcs(a.b.c.d,c,46)
```

$deltok

Added in 1.9.0

$deltok(text,N-N2,C)

Deletes the Nth token from text.

N-N2 can be used to get a range of tokens, both numbers can be negative.
N- can be used to get all tokens from position N.

Parameters

text - The text to tokenize.
N-N2 - The range of tokens or Nth token to delete.
C - The ASCII value to tokenize by.

Example

```
; Delete the 3rd token, result is 'a.b.d'
//echo -ag $deltok(a.b.c.d,3,46)

; Delete the second to third token, result is 'a.d'
//echo -ag $deltok(a.b.c.d,2-3,46)

; Delete all tokens from the second to end.
//echo -ag $deltok(a.b.c.d,2-,46)
```

$findtok

Added in 1.9.0

$findtok(text,token,N,C)

Returns the position of the Nth matching token in text.

$findtok *is case-insensitive, see* $findtokcs *for case-sensitive version.*

Parameters

text - The text to tokenize.
token - The token to find.
N - The Nth token to find.
C - The ASCII value to tokenize by.

Example

```
; Find position where token 'c' starts.
//echo -ag $findtok(a.b.c.d,c,1,46)
```

$findtokcs

Added in 1.9.0

$findtokcs(text,token,N,C)

Returns the position of the Nth matching token in text.

$findtokcs *is case-sensitive, see* $findtok *for case-insensitive version.*

Parameters

text - The text to tokenize.
token - The token to find.
N - The Nth token to find.
C - The ASCII value to tokenize by.

Example

```
; Find position where token 'C' starts.
//echo -ag $findtokcs(a.b.C.d,c,1,46)
```

$gettok

Added in 1.9.0

$gettok(text,N,C)

Returns the Nth token in text.

Same as $token.

N-N2 can be used to get a range of tokens, both numbers can be negative.
N- can be used to get all tokens from position N.

Parameters

text - The text to tokenize.
N - The Nth token to get.
C - The ASCII value to tokenize by.

Example

```
; Print the 3rd token in 'a.b.c.d.e'.
//echo -ag $gettok(a.b.c.d.e,3,46)

; Print the second to forth token in 'a.b.c.d.e'.
//echo -ag $gettok(a.b.c.d.e,2-4,46)

; Print all tokens from the second position.
//echo -ag $gettok(a.b.c.d.e,2-,46)
```

$instok

Added in 1.9.0

$instok(text,token,N,C)

Inserts token into the Nth position in text, even if it already exists in text.

N can be a negative value.

Parameters

text - Text to add the token to.
token - Token to insert.
N - The Nth position to insert at.
C - The ASCII value to tokenize by.

Example

```
; Insert token 'c' at token position '3'.
//echo -ag $instok(a.b.d,c,3,46)
```

$istok

Added in 1.9.0

$istok(text,token,C)

Returns $true if token exists in text, otherwise returns $false.

$istok *is case-insensitive, see* $istokcs *for a case-sensitive version.*

Parameters

text - The text to tokenize.
token - The token to check.
C - The ASCII value to tokenize by.

Example

```
; Check if 'b' is a token in 'a.b.c.d'
//echo -ag $istok(a.b.c.d, b, 46)
```

$istokcs

Added in 1.9.0

$istokcs(text,token,C)

Returns $true if token exists in text, otherwise returns $false.

$istokcs *is case-sensitive, see* $istok *for a case-insensitive version.*

Parameters

text - The text to tokenize.
token - The token to check.
C - The ASCII value to tokenize by.

Example

```
; Check if 'B' is a token in 'a.B.c.d'
//echo -ag $istokcs(a.B.c.d, B, 46)
```

$matchtok

Added in 1.9.0

$matchtok(text,string,N,C)

Returns tokens that contain the specified string.

$matchtok *is case-insensitive, see* $matchtokcs *for case-sensitive version.*

Parameters

text - The text to tokenize.
string - Search string.
N - If N is 0, returns number of matches, otherwise returns the Nth match.

C - The ASCII value to tokenize by.

Example

```
; Returns number of matches
//echo -ag $matchtok(one two three, e, 0, 32)

; Returns the second match
//echo -ag $matchtok(one two three, e, 2, 32)
```

$matchtokcs

Added in 1.9.0

$matchtokcs(text,string,N,C)

Returns tokens that contain the specified string.

$matchtokcs *is case-sensitive, see* $matchtok *for case-insensitive version.*

Parameters

text - The text to tokenize.
string - Search string.
N - If N is 0, returns number of matches, otherwise returns the Nth match.
C - The ASCII value to tokenize by.

Example

```
; Returns number of matches
//echo -ag $matchtokcs(onE two thrEe, E, 0, 32)

; Returns the second match
//echo -ag $matchtokcs(onE two thrEe, E, 2, 32)
```

$numtok

Added in 1.9.0

$numtok(text,C)

Returns number of tokens in text.

Parameters

text - The text to tokenize.
C - The ASCII value to tokenize by.

Example

```
; Print number of tokens in text
//echo -ag $numtok(a.b.c.d.e, 46)
```

$puttok

Added in 1.9.0

$puttok(text,token,N,C)

Overwrites the Nth token in text with a new token.

N-N2 can be used to get a range of tokens, both numbers can be negative.

N- can be used to get all tokens from position N.

Parameters

text - The text to tokenize.
token - Token to put.
N - The Nth token to replace.
C - The ASCII value to tokenize by.

Example

```
; Replace the second token with 'e'.
//echo -ag $puttok(a.b.c.d,e,2,46)

; Print the second to third token to 'e'.
//echo -ag $puttok(a.b.c.d,e,2-3,46)

; Replace all tokens from position 2 to 'e'.
//echo -ag $puttok(a.b.c.d,e,2-,46)
```

$remtok

Added in 1.9.0

$remtok(text,token,N,C)

Removes the Nth matching token from text.

$remtok *is case-insensitive, see* $remtokcs *for case-sensitive version.*

Parameters

text - The text to tokenize.
token - Token to remove.
N - If N = 0, all matching tokens, otherwise the Nth token.
C - The ASCII value to tokenize by.

Example

```
; Remove the first 'b' token.
//echo -ag $remtok(a.b.c.d,b,1,46)

; Remove all 'b' tokens.
//echo -ag $remtok(a.b.b.b,b,0,46)
```

$remtokcs

Added in 1.9.0

$remtokcs(text,token,N,C)

Removes the Nth matching token from text.

$remtokcs *is case-sensitive, see* $remtok *for case-insensitive version.*

Parameters

text - The text to tokenize.
token - Token to remove.
N - If N = 0, all matching tokens, otherwise the Nth token.
C - The ASCII value to tokenize by.

Example

```
; Remove the first 'B' token.
//echo -ag $remtokcs(a.b.B.c.d,B,1,46)

; Remove all 'B' tokens.
//echo -ag $remtokcs(a.b.B.B,b,0,46)
```

$reptok

Added in 1.9.0

$reptok(text,token,new,[N],C)

Replaces the Nth matching token in text with a new token.

$reptok *is case-insensitive, see* $reptokcs *for case-sensitive version.*

Parameters

text - The text to tokenize.
token - Token to match.
new - New token to replace with.
[N] - If N = 0, all matching tokens, otherwise the Nth match. (Optional, default is 1)
C - The ASCII value to tokenize by.

Example

```
; Replace the first token matching 'b' with 'e'
//echo -ag $reptok(a.b.c.d,b,e,1,46)

; Replace all tokens matching 'b' with 'e'
//echo -ag $reptok(b.b.b.c,b,e,0,46)
```

$reptokcs

Added in 1.9.0

$reptokcs(text,token,new,[N],C)

Replaces the Nth matching token in text with a new token.

$reptokcs *is case-sensitive, see* $reptok *for case-insensitive version.*

Parameters

text - The text to tokenize.
token - Token to match.
new - New token to replace with.
[N] - If N = 0, all matching tokens, otherwise the Nth match. (Optional, default is 1)
C - The ASCII value to tokenize by.

Example

```
; Replace the first token matching 'B' with 'E'
//echo -ag $reptokcs(a.B.b.c.d,B,E,1,46)

; Replace all tokens matching 'B' with 'e'
//echo -ag $reptokcs(B.B.B.b,B,e,0,46)
```

$sorttok

Added in 1.9.0

$sorttok(text,C,ncra)

Sorts the tokens in text.

$sorttok *is case-insensitive, see* $sorttokcs *for case-sensitive version.*

Parameters

text - The text to tokenize.
C - The ASCII value to tokenize by.
ncra - n = numeric sort, c = channel nick prefix sort, r = reverse sort, a = alphanumeric sort. (Default is an alphabetic sort)

Example

```
; Sort the tokens alphanumeric
//echo -ag $sorttok(e.d.c.b.a,46)

; Sort the tokens numeric, reversed
//echo -ag $sorttok(1.3.5.2.4,46,nr)
```

$sorttokcs

Added in 1.9.0

$sorttokcs(text,C,ncra)

Sorts the tokens in text.

$sorttokcs *is case-sensitive, see* $sorttok *for case-insensitive version.*

Parameters

text - The text to tokenize.
C - The ASCII value to tokenize by.
ncra - n = numeric sort, c = channel nick prefix sort, r = reverse sort, a = alphanumeric sort. (Default is an alphabetic sort)

Example

```
; Sort the tokens alphanumeric
//echo -ag $sorttokcs(e.D.c.b.a,46)
```

$wildtok

Added in 1.9.0

$wildtok(text,wildstring,N,C)

Returns the Nth token that matches the wildcard string.

$wildtok *is case-insensitive, see* $wildtokcs *for case-sensitive version.*

Parameters

text - The text to tokenize.
wildstring - String to search.
N - If N = 0, return number of matching tokens, otherwise the Nth token.
C - The ASCII value to tokenize by.

Example

```
; Prints the number of matches from the wildstring 't*'.
//echo -ag $wildtok(one two three, t*, 0, 32)

; Prints the first match from the wildstring 't*e'.
```

```
//echo -ag $wildtok(one two three, t*e, 1, 32)
```

$wildtokcs

Added in 1.9.0

$wildtokcs(text,wildstring,N,C)

Returns the Nth token that matches the wildcard string.

$wildtokcs *is case-sensitive, see* $wildtok *for case-insensitive version.*

Parameters

text - The text to tokenize.
wildstring - String to search.
N - If N = 0, return number of matching tokens, otherwise the Nth token.
C - The ASCII value to tokenize by.

Example

```
; Prints the number of matches from the wildstring 'T*'.
//echo -ag $wildtokcs(one two Three, T*, 0, 32)

; Prints the first match from the wildstring 'T*e'.
//echo -ag $wildtokcs(one two Three, T*e, 1, 32)
```

Tokenize

Added in 1.9.0

/tokenize <C> <text>

Fills the $1 $2 ... $N identifiers with tokens in <text> separated by character <c>.

Parameters

<C> - The ASCII value to separate by.
<text> - The text to separate.

Example

```
; Separate the text "one:two:three" by ASCII value 58 (:)
//tokenize 58 one:two:three | echo -a Result of $!2: $2
```

$*

Added in 1.9.0

$*

This is a special identifier which can be used to loop a script line for each token in $1-.

Example

```
; Create a loopme alias.
alias loopme {
  echo -ag Current token is $*
}

; Call the loopme alias.
/loopme a b c d e f g
```