A through action snmp Commands

A through action snmp Commands

action add

To specify the action of adding values of two variables when an an Embedded Event Manager (EEM) applet is triggered, use the **action add** command in applet configuration mode. To undo the add action, use the **no** form of this command.

action *label* **add** {*long-integer*| *variable-name*} {*long-integer*| *variable-name*}

no action *label* **add**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
add	Adds the values of two variables.
variable-name	String value to be placed as the variable name.
long-integer	Long integer value to be added to a variable.

Command Default

By default, there is no change in the value of variables configured within an EEM applet.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

You can use this command to add the values of two variables. The result is stored in the variable named $_result. The value of the variable must be a long integer, else the action will fail.

Examples

The following example shows how to configure an EEM applet to add the values of two variables:

```
Router(config)#event manager applet one
Router(config-applet)#action 1.0 set $var1 10
Router(config-applet)#action 1.0 set $var2 20
Router(config-applet)#action 1.0 add $var1 $var2
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the Embedded Event Manager and enters applet configuration mode.

action append

To specify the action of appending the given string value to the current value of a variable when an Embedded Event Manager (EEM) applet is triggered, use the **action append** command in applet configuration mode. To undo the append action, use the **no** form of this command.

action *label* **append** *variable-name* [*variable-value*]

no action *label* **add**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
append	Appends the given string value to the current value of the variable specified.
variable-name	String value to be placed as the variable name.
variable-value	(Optional) Long integer value to be appended tothe value of the variable name specified.

Command Default

By default, there is no change in the value of variables configured within an EEM applet.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

You can use this command to append the given string value to the current value of variable. If the variable does not exist, it will be created and set to the given value.

Examples

The following example shows how to configure an EEM applet to append given string value to the current value of the variable specified:

```
Router(config)#event manager applet one
```

```
Router(config-applet)#action 1.0 set $var1 10
Router(config-applet)#action 1.0 append $var1 12
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the Embedded Event Manager and enters applet configuration mode.

action break

To specify the action of exiting from a loop of actions when an Embedded Event Manager (EEM) applet is triggered, use the **action break** command in applet configuration mode. To disable the break action, use the **no** form of this command.

action *label* **break**

no action *label* **break**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
break	Causes an immediate exit from a loop of actions.

Command Default

By default, there is no exit from a loop of actions configured within an EEM applet.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

You can use this command to skip all the actions down to the related end action.

Examples

The following example shows how to configure an EEM applet to break from a loop of actions:

```
Router(config)# event manager applet loop
Router(config-applet)# event none
Router(config-applet)# action 1 while 1 eq 1
Router(config-applet)# action 2 break
Router(config-applet)# action 3 end
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the Embedded Event Manager and enters applet configuration mode.

action cli

To specify the action of executing a Cisco IOS command-line interface (CLI) command when an Embedded Event Manager (EEM) applet is triggered, use the **action cli**command in applet configuration mode. To remove the action of executing a CLI command, use the **no** form of this command.

action *label* **cli command** *cli-string* [**pattern** *pattern-string*]

no action *label* **cli command** *cli-string*

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
command	Specifies the message to be sent to the Cisco IOS CLI.
cli-string	CLI command to be executed. If the string contains embedded blanks, enclose it in double quotation marks.
pattern	(Optional) Specifies the regular expression response pattern for the **command** *cli-string*only when the command string solicits input.
pattern-string	(Optional) Specifies the action to be specified with the **pattern** keyword. You are required to specify a regular expression *pattern-string* that will match the next solicited prompt.

Command Default

No CLI commands are executed when an EEM applet is triggered.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.3(14)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(18)SXF4	This command was integrated into Cisco IOS Release 12.2(18)SXF4 to support Software Modularity images only.

Release	Modification
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(18)SXF5	This command was integrated into Cisco IOS Release 12.2(18)SXF5.
12.2(33)SXH	The **pattern** keyword was added.

Usage Guidelines

Use the **action cli** command to specify the action of executing a Cisco IOS CLI command when an EEM applet is triggered. The **pattern** keyword is optional and is used only when the command string solicits input.

There are two types of Cisco IOS CLI commands:

- Normal--Those Cisco IOS CLI commands that produce output followed by a display of the normal router prompt. The **action cli** command ends when the normal router prompt is received.
- Solicited--Those Cisco IOS CLI commands that ask one or more questions before the normal router prompt is displayed, such as “confirm,” which has to be completed with a “yes” or a “no”input.

The **action cli**command ends when the solicited prompt as specified in the optional **pattern** keyword is received. You are required to specify a regular expression pattern that will match the next solicited prompt. Specifying an incorrect pattern will cause the **action cli** command to wait forever until the applet execution times out due to the maxrun timer expiration.

The vty lines are allocated from the pool of vty lines that are configured using the **line vty** CLI configuration command. EEM will use a vty line when a vty line is not being used by EEM and there are available vty lines. EEM will also use a vty line when EEM is already using a vty line and there are three or more vty lines available. Be aware that the connection will fail when fewer than three vty lines are available, preserving the remaining vty lines for Telnet use.

The table below shows the built-in variable that is set when the **action cli** command is run.

Table 1: EEM Built-in Variables for action cli Command

Built-in Variable	Description
$_cli_result	The result of the execution of the CLI command.

Examples

The following example shows how to specify an EEM applet to run when the Cisco IOS **interface loopback** CLI command is configured three times. The applet executes the **no shutdown** command to ensure that the loopback interfaces are operational.

```
Router(config)# event manager applet cli-match
Router(config-applet)# event cli command {.*interface loopback*} sync yes occurs 3
Router(config-applet)# action 1.0 cli command "no shutdown"
```

The following example shows how to specify an EEM applet to run when the **pattern** keyword specifies the *confirm* argument for the **clear counters Ethernet0/1** command.

```
Router(config)# event manager applet cli-match
Router(config-applet)# action 1.0 cli command "enable"
Router(config-applet)# action 2.0 cli command "clear counters Ethernet0/1" pattern "confirm"
```

```
Router(config-applet)# action 3.0 cli command "y"
!
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the Embedded Event Manager and enters applet configuration mode.

action comment

To specify the action of adding comments to an applet when an Embedded Event Manager (EEM) applet is triggered, use the **action comment** command in applet configuration mode. To disable the comment, use the **no** form of this command.

action *label* **comment** *string*

no action *label* **comment**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
comment	Adds comments to an applet.
string	Series of characters, including embedded spaces, to be placed as the comment.

Command Default

By default, there are no comments added to an applet.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

You can use this command to add comments to applets. This results in a no-op when the applet is run.

Examples

The following example shows how to add comments to an applet:

```
Router(config)#event manager applet one
Router(config-applet)#action 1.0 comment keyvalue
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the Embedded Event Manager and enters applet configuration mode.

action context retrieve

To specify the action of retrieving variables identified by a given set of context name keys, when an Embedded Event Manager (EEM) applet is triggered, use the **action context retrieve**command in applet configuration mode. To undo the retrieve action, use the **no** form of this command.

action *label* **context retrieve key** *key-name* **variable** *variable-name-pattern*

no action *label* **context retrieve**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
context retrieve	Used to retrieve variables identified by the given context name keys.
key *key-name*	Provides the context name key.
variable *variable-name-pattern*	Provides description of the variable.

Command Default

By default, no variables specified by a given set of context name keys are retrieved.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

You can use this command to retrieve the variable(s) identified by a given set of context name keys. Information that is retrieved is automatically deleted from the context database.

The information for the variable specified in the command is retrieved, only if a variable with the same name was saved in the corresponding context save call, using the **action context save**command.

Examples

The following example shows how to configure an EEM applet to retrieve variables identified by a given set of context name keys:

```
Router(config)#event manager applet one
Router(config-applet)#action 1.0 context retrieve key pki-72a variable var1
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the Embedded Event Manager and enters applet configuration mode.
action context save	This command is used to save information across multiple policy triggers.

action context save

To specify the action of saving information across multiple policy triggers, when an Embedded Event Manager (EEM) applet is triggered, use the **action context save**command in applet configuration mode. To remove the saved the information, use the **no** form of this command.

action *label* **context save key** *key-name* **variable** *variable-name-pattern*

no action *label* **context save**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
context save	Used to save information across multiple policy triggers.
key *key-name*	Provides the context name key.
variable *variable-name-pattern*	Provides description of the variable.

Command Default

By default, no infomration in saved across multiple policy triggers.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.

Usage Guidelines

You can use the **action context save**command to save information across multiple policy triggers. The command saves variables that match the given pattern with the context name key as identification. Saved information can be retrieved by a different applet using the **action context retrieve**command.

Once the saved information is retrieved, it is automatically deleted from the context database. To save the same information from the applet that retrieved it, you must run the **action context save**command on that applet again.

Examples

The following example shows how to configure an EEM applet to save information across multiple policy triggers:

```
Router(config)#event manager applet one
Router(config-applet)#action 1.0 context save key pki-72a variable var1
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the Embedded Event Manager and enters applet configuration mode.
action context retrieve	Retrieves variables identified by the given context name keys.

action else

To identify the beginning of an else conditional action block in an if/else conditional action block when an Embedded Event Manager (EEM) applet is triggered, use the **action else** command in applet configuration mode. To remove the else conditional action block, use the **no** form of this command.

action *label* **else**

no action *label* **else**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

Command Default

If the command is not entered within applet configuration mode, the respective applet is not registered when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

Use the **action else**command to identify the else conditional action block. If a statement is not associated with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit time of the configuration.

To provide a consistent user interface for the customers between the Tool Command Language (Tcl) and the CLI applet-based EEM policies, the following criteria are followed:

- Event specification criteria are written in Tcl in the Tcl-based implementation.
- Event specification data are written using the CLI applet-submode configuration statements in the applet-based implementation.

Some of the keywords appear to be longer than necessary or hyphenated in the applet-based implementation because the Tcl-based implementation was developed and deployed first.

To enter applet configuration mode, use the **event manager applet** *applet-name* command after entering global configuration mode. In applet configuration mode the config prompt changes to (config-applet)#. The applet configuration mode supports three types of configuration statements:

- **event** --Specifies the event criteria that causes this applet to run.
- **action** --Performs a built-in action.
- **set** --Sets an applet variable (currently _exit_status is the only variable supported).

Examples

The following example shows how to identify the beginning of an else action block:

```
Router(config)# event manager applet action
Router(config-applet)# action label if $var eq 0
Router(config-applet)# action label2 else
Router(config-applet)# end
```

Related Commands

Command	Description
action elseif	Identifies the beginning of an elseif conditional action block when an EEM applet is triggered.
action if	Identifies the beginning of an if conditional action block when an EEM applet is triggered.

action end

To identify the end of a conditional action block in the if/else and while conditional action block when an Embedded Event Manager (EEM) applet is triggered, use the **action end** command in applet configuration mode. To remove the end conditional action block, use the **no** form of this command.

action *label* **end**

no action *label* **end**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

Command Default

If the commands are not specified within applet configuration mode, the respective applet is removed when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.

Usage Guidelines

Use the **action end**command to identify the end of a conditional action block in the if/else and while conditional action block.

To provide a consistent user interface for the customers between the Tool Command Language (Tcl) and the CLI applet-based EEM policies, the following criteria are followed:

- Event specification criteria are written in Tcl in the Tcl-based implementation.
- Event specification data are written using the CLI applet submode configuration statements in the applet-based implementation.

Some of the keywords appear to be longer than necessary or hyphenated in the applet-based implementation because the Tcl-based implementation was developed and deployed first.

To enter applet configuration mode, use the **event manager applet** *applet-name* command after entering global configuration mode. In applet configuration mode the config prompt changes to (config-applet)#. The applet configuration mode supports three types of configuration statements:

- **event** --Specifies the event criteria that causes this applet to run.

- **action** --Performs a built-in action.
- **set** --Sets an applet variable (currently _exit_status is the only variable supported).

Examples

The following example shows hoe to identify the end of a conditional action block:

```
Router(config)# event manager applet action
Router(config-applet)# event none
Router(config-applet)# action 1.0 set x "5"
Router(config-applet)# action 2.0 if $x lt 10
Router(config-applet)# action 3.0  puts "$x is less than 10"
Router(config-applet)# action 4.0 end
```

Related Commands

Command	Description
action else	Identifies the beginning of the else conditional action block when an EEM applet is triggered.
action if	Identifies the beginning of an if conditional action block when an EEM applet is triggered.

action foreach

To specify the iteration of an input string using the delimiter as a tokenizing pattern, use the **action foreach** command in applet configuration mode. To remove iteration of the input string, use the **no** form of this command.

action *label* **foreach** [*string-iterator*] [*string-input*] [*string-delimiter*]

no action *label* **foreach**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string-iterator	(Optional) Series of characters that acts as an iterator. If the string contains embedded blanks, enclose it in double quotation marks.
string-input	(Optional) Series of characters that acts as an input. If the string contains embedded blanks, enclose it in double quotation marks.
string-delimiter	(Optional) Series of characters that acts as a delimiter. If the string contains embedded blanks, enclose it in double quotation marks. The default delimiter is whitespace.

Command Default

If the commands are not specified within applet configuration mode, the respective applet is removed when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

Use the **action foreach** command to iterate an input string using the delimiter as a tokenizing pattern. The delimiter is a regular expression pattern string. The token found in each iteration is assigned to the given

iterator variable. All arithmetic calculations are performed as long integers without any checks for overflow. If a statement is not associated with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit time of the configuration.

To provide a consistent user interface for the customers between the Tool Command Language (Tcl) and the CLI applet-based EEM policies, the following criteria are followed:

- The event specification criteria are written in Tcl in the Tcl-based implementation.
- The event specification data are written using the CLI applet submode configuration statements in the applet-based implementation.

Some of the keywords appear to be longer than necessary or hyphenated in the applet-based implementation because the Tcl-based implementation was developed and deployed first.

To enter applet configuration mode, use the **event manager applet** *applet-name*command after entering global configuration mode. In applet configuration mode the config prompt changes to (config-applet)#. The applet configuration mode supports three types of configuration statements:

- **event** --Specifies the event criteria that causes this applet to run.
- **action** --Performs a built-in action.
- **set** --Sets an applet variable (currently _exit_status is the only variable supported).

Examples

The following example shows how to iterate an input string using the delimiter as a tokenizing pattern:

```
Router(config)# event manager applet action
Router(config-applet)# event none
Router(config-applet)# action 1 foreach _iterator "red blue green orange"
Router(config-applet)# action 2  puts "iterator is $_iterator"
Router(config-applet)# action 3 end
Router# event manager run action
iterator is red
iterator is blue
iterator is green
iterator is orange
Router#
```

action gets

To get an input from the local tty in a synchronous applet and store the value in the given variable when an Embedded Event Manager (EEM) applet is triggered, use the **action gets** command in applet configuration mode. To cancel the process of receiving an input from the local tty, use the **no** form of this command.

action *label* **gets** *variable*

no action *label* **gets**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
variable	Variable word that stores the input value from the synchronous applet.

Command Default

If the commands are not specified within applet configuration mode, the respective applet is removed when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

Use the **action gets** command to get an input from the local tty in a synchronous applet and store the value in the given variable. This command is not operational for asynchronous applets. The applet continues without any error but does not set the variable. All arithmetic calculations are performed as long integers without any checks for overflow. If a statement is not associated with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit time of the configuration.

To provide a consistent user interface for the customers between the Tool Command Language (Tcl) and the CLI applet-based EEM policies, the following criteria are followed:

- Event specification criteria are written in Tcl in the Tcl-based implementation.

- Event specification data are written using the CLI applet submode configuration statements in the applet-based implementation.

Some of the keywords appear to be longer than necessary or hyphenated in the applet based implementation because the Tcl-based implementation was developed and deployed first.

To enter applet configuration mode, use the **event manager applet** *applet-name* command after entering the global configuration mode. In applet configuration mode the config prompt changes to (config-applet)#. The applet configuration mode supports three types of configuration statements:

- **event** --Specifies the event criteria that causes this applet to run.
- **action** --Performs a built-in action.
- **set** --Sets an applet variable (currently _exit_status is the only variable supported).

Examples

The following example shows how to get the input from the local tty in a synchronous applet and store the value:

```
Router(config)# event manager applet action
Router(config-applet)# event none
Router(config-applet)# action label2 gets input
Router(config-applet)# action label3 syslog msg “Input entered was \”$input\“”
```

Related Commands

Command	Description
action puts	Prints data directly to the local tty in a synchronous applet when an EEM applet is triggered.

action if

To identify the beginning of an if conditional block when an Embedded Event Manager (EEM) applet is triggered, use the **action if** command in applet configuration mode. To remove the if conditional action block, use the **no** form of this command.

action *label* **if** [*string-op-1*] {**eq**| **gt**| **ge**| **lt**| **le**| **ne**} [*string-op-2*]

no action *label* **if**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string-op-1	(Optional) Sequence of characters that will replace the range of characters in the string.
eq	Equal To keyword used for comparing two strings.
gt	Greater Than keyword used for comparing two strings.
ge	Greater Than or Equal To keyword used for comparing two strings.
lt	Less Than keyword used for comparing two strings.
le	Less Than or Equal To keyword used for comparing two strings.
ne	Not Equal To keyword used for comparing two strings.
string-op-2	(Optional) Sequence of characters that will replace the range of characters in the string.

Command Default

If the commands are not specified within applet configuration mode, the respective applet is removed when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

Use the **action if**command to identify the beginning of the if conditional action block. All arithmetic calculations are performed as long integers without any checks for overflow. If the *goto label* option is used, the if functionality will not identify the beginning of a action block, but will signify the applet to jump to the given label if the condition is true.

If a statement is not associated with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit time of the configuration.

To provide a consistent user interface for the customers between the Tool Command Language (Tcl) and the CLI applet-based EEM policies, the following criteria are followed:

- Event specification criteria are written in Tcl in the Tcl-based implementation.
- Event specification data are written using the CLI applet submode configuration statements in the applet-based implementation.

Some of the keywords appear to be longer than necessary or hyphenated in the applet based implementation because the Tcl based implementation was developed and deployed first.

To enter applet configuration mode, use the **event manager applet** *applet-name* command after entering global configuration mode. In applet configuration mode the config prompt changes to (config-applet)#. The applet configuration mode supports three types of configuration statements:

- **event** --Specifies the event criteria that causes this applet to run.
- **action** --Performs a built-in action.
- **set** --Sets an applet variable (currently _exit_status is the only variable supported).

Examples

The following example shows how to identify the beginning of an if conditional block:

```
Router(config)# event manager applet action
Router(config-applet)# event none
Router(config-applet)# action 1.0 set x "5"
Router(config-applet)# action 2.0 if $x lt 10
Router(config-applet)# action 3.0  puts "$x is less than 10"
Router(config-applet)# action 4.0 end
Router# event manager run action
5 is less than 10
Router#
```

Related Commands

Command	Description
action elseif	Identifies the beginning of the else conditional action block in the else/if conditional block when an EEM applet is triggered.
action ifgoto	Signifies the applet to jump to the given label if the condition is true when an EEM applet is triggered.

action ifgoto

To instruct the applet to jump to a given label if the specified condition is true when an Embedded Event Manager (EEM) applet is triggered, use the **action ifgoto** command in applet configuration mode. To cancel the process of applet jump, use the **no** form of this command.

action *label-1* **if** [*string-op-1*] {**eq**| **gt**| **ge**| **lt**| **le**| **ne**} [*string-op-2*] **goto** *label-2*

no action *label* **ifgoto**

Syntax Description

label-1	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string-op-1	(Optional) Sequence of characters that will replace the range of characters in the string.
eq	Equal To keyword used for comparing two strings.
gt	Greater Than keyword used for comparing two strings.
ge	Greater Than Or Equal To keyword used for comparing two strings.
lt	Less Than keyword used for comparing two strings.
le	Less Than Or Equal To keyword used for comparing two strings.
ne	Not Equal To keyword used for comparing two strings.
string-op-2	(Optional) Sequence of characters that will replace the range of characters in the string.
label-2	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.

Command Default

If the command is not specified within applet configuration mode, the respective applet is removed when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

Use the **action ifgoto**command to signify the applet to jump to a given label if the specified condition is true. If the **goto** *label*option is used, the **action if** command will not identify the beginning of an action block. Goto actions are supported only within the if/goto format. To simulate a goto without if, use a test that is always true. All arithmetic calculations are performed as long integers without any checks for overflow. If a statement is not associated with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit time of the configuration.

To provide a consistent user interface for the customers between the Tool Command Language (Tcl) and the CLI applet-based EEM policies, the following criteria are followed:

- Event specification criteria are written in Tcl in the Tcl-based implementation.
- Event specification data is written using the CLI applet submode configuration statements in the applet-based implementation.

Some of the keywords appear to be longer than necessary or hyphenated in the applet-based implementation because the Tcl-based implementation was developed and deployed first.

To enter applet configuration mode, use the **event manager applet** *applet-name* command in the global configuration mode. In applet configuration mode, the config prompt changes to (config-applet)#. Applet configuration mode supports three types of configuration statements:

- **event** --Specifies the event criteria that cause this applet to run.
- **action** --Performs a built-in action.
- **set** --Sets an applet variable (currently _exit_status is the only variable supported).

Examples

The following example shows how to instruct the applet to jump to a given label:

```
Router(config)# event manager applet action
Router(config-applet)# event none
Router(config-applet)# action 1 set x "5"
Router(config-applet)# action 2 if $x lt 10 goto 4
Router(config-applet)# action 3 puts "skipping this"
Router(config-applet)# action 4 puts "jumped to action 4"
Router(config-applet)# action 5 end
Router# event manager run action
jumped to action 4
```

Related Commands

Command	Description
action else	Identifies the beginning of the else conditional action block when an EEM applet is triggered.
action if	Identifies the beginning of an if conditional action block when an EEM applet is triggered.

action increment

To specify the action of incrementing the value of a variable, when an Embedded Event Manager (EEM) applet is triggered, use the **action increment** command in applet configuration mode. To remove the action from the applet, use the **no** form of this command.

action *label* **increment** *variable-name long-integer*

no action *label* **increment**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
increment	Increments the value of the variable with the long integer specified.
variable-name	String value to be placed as the variable name.
long-integer	Long integer value by which the variable is incremented.

Command Default

By default, there is no change in the value of variables configured within an EEM applet.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

You can use this command to increment the value of a variable. The value of the variable must be a long integer, else the action will fail.

Examples

The following example shows how to configure an EEM applet to increment the value of a variable:

```
Router(config)#event manager applet one
Router(config-applet)#action 1.0 set varname 20
```

```
Router(config-applet)#action 1.0 increment varname 12
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the Embedded Event Manager and enters applet configuration mode.

action info type interface-names

To obtain interface names when an Embedded Event Manager (EEM) applet is triggered, use the **action info type interface-names**command in applet configuration mode. To disable the action of obtaining interface names, use the **no** form of this command.

action *label* **info type interface-names** [**include** *string-operator*| **exclude** *string-operator*| **regexp** *regular-expression*]

no action *label* **info type**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
include	(Optional) Includes all interface names that contain the string pattern.
exclude	(Optional) Excludes all interface names that contain the string pattern.
string-operator	(Optional) String pattern for including or excluding the interface names.
regexp	(Optional) Obtains all the interfaces that match the specified regular expression.
regular-expression	(Optional) Regular expression pattern. For example, [^abc].

Command Default

All the current interface names are obtained from the database.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

The **action info type interface-names**command obtains the current interface names and stores them as a space-separated list in the $_info_interface_names built-in variable.

Examples

The following example shows how to specify that interface names that include “eth” are obtained:

```
Router# configure terminal
Router(config)# event manager applet interface-app
Router(config-applet)# action 1.2 info type interface-names include eth
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the EEM and enters applet configuration mode.

action info type snmp getid

To retrieve the individual variables from a Simple Network Management Protocol (SNMP) entity during the SNMP get operation, use the **action info type snmp getid**command in applet configuration mode. To disable the retrieving of individual variables from SNMP, use the **no** form of this command.

action *label* **info type snmp getid** *oid-value* [**community** *community-string*] [**ipaddr** *ip-address*]

no action *label* **info type**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
getid	Retrieves SNMP variables.
oid-value	Object ID value of the data element, in SNMP dotted notation. An object identifier is expressed as a series of integers or text strings. For example, the object name for the interfaces MIB can be expressed as 1.3.6.1.2.1.2 or iso.internet.mgmt.mib-2.interfaces. An OID is defined as a type in the associated MIB, CISCO-EMBEDDED-EVENT-MGR-MIB, and each type has an object value. Monitoring of some OID types is supported. The following types are valid: • INTEGER_TYPE • COUNTER_TYPE • GAUGE_TYPE • TIME_TICKS_TYPE • COUNTER_64_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE
community	(Optional) Specifies the community string to access the SNMP entity.

community-string	(Optional) SNMP community string. Community string functions like passwords to access the SNMP entity. The string can consist of 1 to 32 alphanumeric characters and can be set to any of the following types of community strings: • **ro** --Sets the read-only access to the SNMP entity. The default value for this community string is **public**. • **rw** --Sets read-write access to the SNMP entity. The default value for this community string is **private**.
ipaddr	(Optional) Specifies the IP address of the SNMP entity.
ip-address	(Optional) IP address of Network Management System (NMS) from which the objects are retrieved for SNMP get and set operations.

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

The table below shows the built-in variables in which the variables retrieved from the SNMP get operation are stored.

Table 2: EEM Built-in Variables for action info Command

Built-in Variable	Description
$_info_snmp_sysname_oid	The OID value of the sysName variable.
$_ info_snmp_sysname_value	The value string for the sysName variable.
$_info_snmp_syslocation_oid	The OID value of the sysLocation variable.
$_info_snmp_syslocation_value	The value string for the sysLocation variable.
$_info_snmp_sysdescr_oid	The OID value of the sysDescr variable.
$_info_snmp_sysdescr_value	The value string for the sysDescr variable.

Built-in Variable	Description
$_info_snmp_sysobjectid_oid	The OID value of the sysObjectID variable.
$_info_snmp_sysobjectid_value	The value string for the sysObjectID variable.
$_info_snmp_sysuptime_oid	The OID value of the sysUptime variable.
$_info_snmp_sysuptime_value	The value string for the sysUptime variable.
$_info_snmp_syscontact_oid	The OID value of the sysContact variable.
$_info_snmp_syscontact_value	The value string for the sysContact variable.

Examples

The following example shows how to retrieve the sysDescr.0 variable from an SNMP entity:

```
Router(config)# event manager applet
Router(config-applet)# action 1.3 info type snmp getid 1.3.6.1.2.1.1.1.0 community public
ipaddr 172.17.16.69
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the EEM and enters applet configuration mode.
snmp-server community	Sets the community access string to enable access to the SNMP entity.

action info type snmp inform

To send Simple Network Management Protocol (SNMP) inform requests when an Embedded Event Manager (EEM) applet is triggered, use the **action info type snmp inform**command in applet configuration mode. To disable the sending of SNMP inform requests, use the **no** form of this command.

action *label* **info type snmp inform trap-oid** *trap-oid-value* **trap-var** *trap-variable* **community** *community-string* **ipaddr** *ip-address*

no action *label* **info type**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
trap-oid	Specifies the object identifier of the object generating the SNMP trap.
trap-oid-value	The OID value of the object generating the SNMP trap.
trap-var	Specifies the variable associated with the instance of the object generating the trap.
trap-variable	The variable value of the object generating SNMP trap.
community	Specifies the community string to access the SNMP entity.
community-string	SNMP community string. Community string functions like passwords to access the SNMP entity. The string can consist of 1 to 32 alphanumeric characters and can be set to any of the following: • **ro** --Sets the read-only access to the SNMP entity. The default value for this community string is **public**. • **rw** --Sets read-write access to the SNMP entity. The default value for this community string is **private**.
ipaddr	Specifies the IP address of the SNMP entity.

ip-address	IP address of Network Management System (NMS) from which the objects are retrieved for SNMP get and set operations.

Command Default

No SNMP inform requests are sent by default.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

SNMP inform requests are the SNMP notifications that alert the SNMP manager to a network condition and request for confirmation of receipt from the SNMP manager.

Examples

The following example shows how to send an SNMP inform request:

```
Router(config)# event manager applet
Router(config-applet)# action 1.4 info type snmp inform trap-oid 1.3.6.1.4.1.1.226.0.2.1
trap-var sysUpTime community public ipaddr 172.69.16.2
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the EEM and enters applet configuration mode.
snmp-server community	Sets the community access string to enable access to the SNMP entity.
snmp-server enable traps	Enables all SNMP notification types available on your system.

action info type snmp oid

To specify the type of Simple Network Management Protocol (SNMP) get operation and the object to retrieve during the SNMP set operation, when an Embedded Event Manager (EEM) applet is triggered, use the **action info type snmp oid**command in applet configuration mode. To disable this function, use the **no** form of this command.

action *label* **info type snmp oid** *oid-value* {**get-type** {**exact**| **next**} [**community** *community-string*]| **set-type** *oid-type oid-type-value* **community** *community-string*} [**ipaddr** *ip-address*]

no action *label* **info type**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
oid	Requests the value of the SNMP object as specified by the SNMP object identifier (OID).
oid-value	Object ID value of the data element, in SNMP dotted notation. An object identifier is expressed as a series of integers or text strings. For example, the object name for the interfaces MIB can be expressed as 1.3.6.1.2.1.2 or iso.internet.mgmt.mib-2.interfaces. An OID is defined as a type in the associated MIB, CISCO-EMBEDDED-EVENT-MGR-MIB, and each type has an object value. Monitoring of some OID types is supported. The following types are valid: • INTEGER_TYPE • COUNTER_TYPE • GAUGE_TYPE • TIME_TICKS_TYPE • COUNTER_64_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE

get-type	Specifies the type of SNMP get operation to apply to the object ID specified by the *oid-value* argument. • **exact** --(Optional) Retrieves the object ID specified by the *oid-value* argument. • **next** --(Optional) Retrieves the object ID that is the alphanumeric successor to the object ID specified by the *oid-value* argument.
community	Specifies the community string to access the SNMP entity.
community-string	SNMP community string. Community string functions like passwords to access the SNMP entity. The string can consist of 1 to 32 alphanumeric characters and can be set to any of the following: • **ro** --Sets the read-only access to the SNMP entity. The default value for this community string is **public**. • **rw** --Sets read-write access to the SNMP entity. The default value for this community string is **private**.
set-type	Specifies the type of object to retrieve during the SNMP set operation. To perform a set operation, you need to specify the OID, OID type, and value.

oid-type	The type of OID. The following values are valid: • **counter32** --A 32-bit number with a minimum value of 0. When the maximum value is reached, the counter resets to 0. • **gauge** --A 32-bit number with a minimum value of 0. For example, the interface speed on a router is measured using a gauge object type. • **integer** --A 32-bit number used to specify a numbered type within the context of a managed object. For example, to set the operational status of a router interface, 1 represents up and 2 represents down. • **ipv4** --IP version 4 address. • **octet string** --An octet string in hexadecimal notation used to represent physical addresses. • **string** --An octet string in text notation used to represent text strings. • **unsigned32** --A 32-bit number used to represent decimal value.
oid-type-value	Integer or text string value of the OID type specified for the SNMP set operation. The valid values for each OID type are: • **counter** --Integer value in the range from 0 to 4294967295. • **gauge--** Integer value in the range from 0 to 4294967295. • **integer--** Integer value in the range from 0 to 4294967295. • **ipv4--** IPv4 address in dotted decimal notation. • **octet string** --Text string. • **string--** Text string. • **unassigned32--** Unsigned integer value in the range from 0 to 4294967295.
ipaddr	(Optional) Specifies the IP address of the SNMP entity.
ip-address	(Optional) IP address of Network Management System (NMS) from which the objects are retrieved for SNMP get and set operations.

Command Default No requests for SNMP set or get operations are sent when the EEM applet is triggered.

Command Modes Applet configuration (config-applet)

Command History

Release	Modification
12.3(14)T	This command was introduced.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(18)SXF4	This command was integrated into Cisco IOS Release 12.2(18)SXF4 to support Software Modularity images only.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(18)SXF5	This command was integrated into Cisco IOS Release 12.2(18)SXF5.
12.4(22)T	The **set-type**, **community**,and **ipaddr** keywords were added.

Usage Guidelines The SNMP set operation sets individual variables in the SNMP entity, whereas the SNMP get operation retrieves individual variables from the SNMP entity.

The table below shows the built-in variables in which the results of SNMP get and set operations are stored.

Table 3: EEM Built-in Variables for action info Command

Built-in Variable	Description
$_info_snmp_oid	The SNMP object ID.
$_info_snmp_value	The value string of the associated SNMP data element.

Examples The following example shows how to retrieve individual variables of an object from the SNMP entity:

```
Router(config)# event manager applet
Router(config-applet)# action 1.3 info type snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 get-type
 exact community public ipaddr 172.17.16.69
Router(config-applet)#
```

The following example shows how to set an individual variable in the SNMP entity:

```
Router(config)# event manager applet
Router(config-applet)# action 1.4 info type snmp oid 1.3.6.1.4.1.9.9.48.1.1.1.6.1 set-type
```

```
 integer 42220 sysName.0 community public ipaddr 172.17.16.69
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the EEM and enters applet configuration mode.
snmp-server community	Sets the community access string to enable access to the SNMP entity.

action info type snmp trap

To send Simple Network Management Protocol (SNMP) trap requests when an Embedded Event Manager (EEM) applet is triggered, use the **action info type snmp trap**command in applet configuration mode. To disable the sending of SNMP trap requests, use the **no** form of this command.

action *label* **info type snmp trap enterprise-oid** *enterprise-oid-value* **generic-trapnum** *generic-trap-number* **specific-trapnum** *specific-trap-number* **trap-oid** *trap-oid-value* **trap-var** *trap-variable*

no action *label* **info type**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
trap	Sends SNMP trap requests.
enterprise-oid	Specifies the enterprise OID value of the object.
enterprise-oid-value	Enterprise OID value of the object generating the SNMP trap. The OID value is enterprise specific and is expressed as a series of integers or text strings.
generic-trapnum	Specifies the generic SNMP trap number.
generic-trap-number	The generic trap number. The following generic traps and trap numbers are valid: • coldStart (0) • warmStart (1) • linkDown (2) • linkUp (3) • authenticationFaliure(4) • egpNeighborLoss(5) • enterpriseSpecific (6)
specific-trapnum	Specifies the enterprise-specific trap if the generic trap number is not set to 6.
specific-trap-number	The number associated with the trap specific to an enterprise event.

trap-oid	Specifies the object identifier of the object generating the SNMP trap.
trap-oid-value	The OID value of the object generating the SNMP trap.
trap-var	Specifies the variable associated with the instance of the object generating the trap.
trap-variable	The variable value of the object generating SNMP trap.

Command Default

No SNMP trap requests are sent by default.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

Traps are SNMP notifications that alert the SNMP manager or the NMS to a network condition. Unlike SNMP inform requests, traps do not request the receipt from the SNMP manager.

Examples

The following example shows how to send an SNMP trap request:

```
Router(config)# event manager applet
Router(config-applet)# action 1.4 info type snmp trap enterprise-oid 1.3.6.1.4.1.1
generic-trapnum 4 specific-trapnum 7 trap-oid 1.3.6.1.4.1.1.226.0.2.1 trap-var sysUpTime
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the EEM and enters applet configuration mode.
snmp-server enable traps	Enables all SNMP notification types available on your system.

action info type snmp var

To create a variable for a Simple Network Management Protocol (SNMP) object identifier (OID) and its value from an Embedded Event Manager (EEM) applet, use the **action info type snmp var**command in applet configuration mode. To remove the variable, use the **no** form of this command.

action *label* **info type snmp var** *variable-name* **oid** *oid-value oid-type oid-type-value*

no action *label* **info type**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
var	Specifies the SNMP variable or the object instance of the SNMP MIB object.
variable-name	Name of the SNMP variable. For example, sysDescr.0.
oid	Requests the value of the SNMP object as specified by the SNMP OID.
oid-value	Object ID value of the data element, in SNMP dotted notation. An object identifier is expressed as a series of integers or text strings. For example, the object name for the interfaces MIB can be expressed as 1.3.6.1.2.1.2 or iso.internet.mgmt.mib-2.interfaces. An OID is defined as a type in the associated MIB, CISCO-EMBEDDED-EVENT-MGR-MIB, and each type has an object value. Monitoring of some OID types is supported. The following types are valid: • INTEGER_TYPE • COUNTER_TYPE • GAUGE_TYPE • TIME_TICKS_TYPE • COUNTER_64_TYPE • OCTET_PRIM_TYPE • OPAQUE_PRIM_TYPE

oid-type	The type of OID. The following values are valid: • **counter32** --A 32-bit number with a minimum value of 0. When the maximum value is reached, the counter resets to 0. • **gauge** --A 32-bit number with a minimum value of 0. For example, the interface speed on a router is measured using a gauge object type. • **integer** --A 32-bit number used to specify a numbered type within the context of a managed object. For example, to set the operational status of a router interface, 1 represents up and 2 represents down. • **ipv4** --IP version 4 address. • **octet string** --An octet string in hex notation used to represent physical addresses. • **string** --An octet string in text notation used to represent text strings. • **unsigned32** --A 32-bit number used to represent decimal value.
oid-type-value	Integer or text string value of the OID type specified for creating a variable. The valid values for each OID type are: • **counter** --Integer value in the range from 0 to 4294967295. • **gauge--** Integer value in the range from 0 to 4294967295. • **integer--** Integer value in the range from 0 to 4294967295. • **ipv4--** IPv4 address in dotted decimal notation. • **octet string** --Text string. • **string--** Text string. • **unassigned32--** Unsigned integer value in the range from 0 to 4294967295.

Command Default No variables are created by default when an EEM applet is triggered.

Command Modes Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

A variable is identified by its OID and its instance. The instance is generally specified by appending a .0 to its OID. For example, sysDescr.0.

Examples

The following example shows how to create a variable for an object identifier:

```
Router(config)# event manager applet
Router(config-applet)# action 1.3 info type snmp var sysDescr.0 oid
1.3.6.1.4.1.9.9.48.1.1.1.6.1 integer 4220
Router(config-applet)#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the EEM and enters applet configuration mode.

action multiply

To specify the action of multiplying the variable value with a specified given integer value when an Embedded Event Manager (EEM) applet is triggered, use the **action multiply** command in applet configuration mode. To remove the calculation process, use the **no** form of this command.

action *label* **multiply** [*long-integer-1*| *variable-name-1*] [*long-integer-2*| *variable-name-2*]

no action *label* **multiply**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
long-integer-1	(Optional) First integer value for the multiplication.
variable-name-1	(Optional) First variable name for the multiplication. The value stored in the multiplier variable-name must be a long integer value or else the action will fail.
long-integer-2	(Optional) Second integer value for the multiplication.
variable-name-2	(Optional) Second variable name for the multiplication. The value stored in the multiplier variable-name must be a long integer value or else the action will fail.

Command Default

If the command is not entered within applet configuration mode, the respective applet is not registered when you exit the configuration.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

Use the **action multiply**command to multiply the value of the variable with a given integer value. All arithmetic calculations are performed as long integers without any checks for overflow. If a statement is not associated

with this applet, events are still triggered without any action or result. A warning message stating that no statements are associated with this applet is displayed at the exit time of the configuration. All the results of the **action multiply** command are stored in $_result.

To provide a consistent user interface for the customers between the Tool Command Language (Tcl) and the CLI applet-based EEM policies, the following criteria are followed:

- Event specification criteria are written in Tcl in the Tcl-based implementation.
- Event specification data are written using the CLI applet submode configuration statements in the applet-based implementation.

Some of the keywords appear to be longer than necessary or hyphenated in the applet-based implementation because the Tcl-based implementation was developed and deployed first.

To enter applet configuration mode, use the **event manager applet** *applet-name* command after entering global configuration mode. In applet configuration mode the config prompt changes to (config-applet)# . Applet configuration mode supports three types of configuration statements:

- **event** --Specifies the event criteria that causes this applet to run.
- **action** --Performs a built-in action.
- **set** --Sets an applet variable (currently _exit_status is the only variable supported).

Examples

The following example shows how to multiply the stored variable value.

```
Router(config)# event manager applet action
Router(config-applet)# action label2 multiply 23 25
```

Related Commands

Command	Description
action add	Adds the value of the variable by the given value when an EEM applet is triggered.
action divide	Divides the value of the variable by the given value when an EEM applet is triggered.
action subtract	Subtracts the value of the variable by the given value when an EEM applet is triggered.

action puts

To enable the action of printing data directly to the local tty when an Embedded Event Manager (EEM) applet is triggered, use the **action puts**command in applet configuration mode. To disable this function, use the **no** form of this command.

action *label* **puts [nonewline]** *string*

no action *label* **puts**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
nonewline	(Optional) Suppresses the display of the new line character.
string	Sequence of characters. If the string contains embedded blanks, enclose it in double quotation marks.

Command Default

Data is not printed to the local tty.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

The **action puts** command applies to synchronous events. The output of this command for a synchronous applet is directly displayed to the tty, bypassing the syslog. This command defaults to the syslog for asynchronous events. The **nonewline** keyword suppresses the display of the new line character. The output of the **action puts** command for an asynchronous applet is directed to the logger.

Examples

The following example shows how to print data directly to the local tty:

```
Router(config-applet)# event manager applet puts
```

```
Router(config-applet)# event none
Router(config-applet)# action 1 regexp "(.*) (.*) (.*)" "one two three" _match _sub1
Router(config-applet)# action 2 puts "match is $_match"
Router(config-applet)# action 3 puts "submatch 1 is $_sub1"
Router# event manager run puts
match is one two three
submatch 1 is one
Router#
```

Related Commands

Command	Description
action gets	Gets input from the local tty and stores the value in the given variable.
event manager applet	Registers an event applet with the EEM and enters applet configuration mode.

action regexp

To match a regular expression pattern on an input string when an Embedded Event Manager (EEM) applet is triggered, use the **action regexp**command in applet configuration mode. To disable this function, use the **no** form of this command.

action *label* **regexp** *string-pattern string-input* [*string-match* [*string-submatch1*] [*string-submatch2*] [*string-submatch3*]]

no action *label* **regexp**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
string-pattern	The sequence of characters to be used for regular expression pattern matching.
string-input	The sequence of characters to be used as input.
string-match	(Optional) The variable name to store the entire match.
string-submatch	(Optional) The variable name to store any submatches that are present. A maximum of three submatch strings can be specified.

Command Default

No regular expression patterns are matched.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

The *string-pattern* argument is a regular expression. If some part of the string matches the pattern, it returns 1; otherwise it returns 0. The optional *string-match* and *string-submatch* arguments store the results of the match.

The table below shows the built-in variable in which the results of the **action regexp**command are stored.

Table 4: EEM Built-in Variables for action regexp Command

Built-in Variable	Description
$_regexp_result	The result of the regular expression pattern matching is stored in this variable.

Examples

The following example shows how to define a regular expression match:

```
Router(config-applet)# event manager applet regexp
Router(config-applet)# event none
Router(config-applet)# action 1 regexp "(.*) (.*) (.*)" "one two three" _match _sub1
Router(config-applet)# action 2 puts "match is $_match"
Router(config-applet)# action 3 puts "submatch 1 is $_sub1"
Router# event manager run regexp
match is one two three
submatch 1 is one
Router#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the EEM and enters applet configuration mode.

action set (EEM)

To set the value of a variable when an Embedded Event Manager (EEM) applet is triggered, use the **action set**command in applet configuration mode. To remove the value of an EEM applet variable, use the **no** form of this command.

action *label* **set** *variable-name variable-value*

no action *label* **set**

Syntax Description

label	Unique identifier that can be any string value. Actions are sorted and run in ascending alphanumeric key sequence using the label as the sort key. If the string contains embedded blanks, enclose it in double quotation marks.
variable-name	Name assigned to the variable to be set.
variable-value	Value of the variable.

Command Default

No variable value is set.

Command Modes

Applet configuration (config-applet)

Command History

Release	Modification
12.4(22)T	This command was introduced. This command replaces the **set** (EEM) command.
12.2(33)SRE	This command was integrated into Cisco IOS Release 12.2(33)SRE.

Usage Guidelines

Use the **action set** command to set the value of a variable when an EEM applet is triggered.

Examples

The following example shows how to set the value of a variable:

```
Router(config-applet)# event manager applet set
Router(config-applet)# event none
Router(config-applet)# action 1 set str “this is some text”
Router(config-applet)# action 2 string range “$str” 0 6
Router(config-applet)# action 3 puts “$_string_result”
Router# event manager run set
“this is”
Router#
```

Related Commands

Command	Description
event manager applet	Registers an event applet with the EEM and enters applet configuration mode.