Rat Help Index

Rat - Reko Advanced Terminal emulator - version 1.1

The Windows program Rat is a terminal emulator. It can be used to connect your Windows equipped PC to a variety of UNIX based computers, to a Bull DPS6, DPS7 or DPS8 computer, to a DEC computer, and to a Prestel Videotex system. Rat has been designed to function in many different environments.

Some of the features in Rat are:

- Script language, scripts can be invoked using buttons , dialogs or from the host computer. Scripting will ease the user from many repetitive tasks.
- Extensive settings capabilities including colors, fonts, 3d effects, terminal types, national character translations, and a lot more.
- The ability to transfer files between the PC and the host computer in a variety of ways.
- Phonebooks for those often used telephone numbers.

Choose help subject

Show the main index page

Overview

Rat has evolved over a long period of time, many people have contributed with their ideas of how a good terminal emulator should work and look like. This means that Rat has a lot of features. This guide is not intended to cover all of the features, some of these will reveal themselves after that you have been using Rat for a while. You will probably find an answer to most of the questions you might have.

Rat is a terminal emulator, i.e. it is used to let your Windows PC act as if it is a terminal connected to a host. There are two ways in wich Rat can communicate with a host:

1. Using a serial connection, or
2. Using a network Winsocket connection.

Your systems administrator will be able to tell you which way you should use to connect to your host, or hosts. (See requirements).

When you open the Rat application, by double clicking on the Rat icon, a standard Windows window is displayed. It has a window frame with a title, it has menus and, perhaps, buttons. If this is the first time you started Rat, you will have the standard settings, it is possible that you have to change them. E.g. if you want to use another terminal type than the preset one.

Once you have the correct settings, things like pf keys, special keys (home, delete etc) will work like you expect them to on a regular dumb terminal. Since Rat is a Windows program, features like copy and paste; file transfer; buttons etc are added to make you work more effectively, and perhaps have some fun on the way.

After some time of using Rat, most people find that the do some repetetitive tasks, such as typing a certain combimation of characters. To make such things easier in life, Rat has a feature called scripting. Take some time to learn more about scripting since it will be a feature you most certainly will enjoy.

File transfer is another feature area you almost certainly will use. There are a number of ways to transfer data back and forth between your PC and a host. Copy and Paste is one method, another is to catch data sent by the host to either the clipboard or a file, the best way is to use one of the file transfer programs that Rat supports. (This depends on your host computer, contact REKO for more information).

The Phonebook is a database that can hold a small number of telephone numbers or names of locations that you frequently use a modem to call. You might not always find the phonebook that much useful, but it is there for you to use !

Requirements

Hardware requirements.

- A serial cable if you are using a serial communications link. The cable is used to connect the PC serial port to a modem, or direct connect to the host computer. Connect the serial cable to one of the serial ports labelled Com1 to Com4. Note that the serial cable is not supplied as part of the Rat package, also note that the host computer port should be an asynchronous RS232 connection, not current loop or synchronous.

- A modem and a telephone if you intend to dial up the host computer, or a cable from the host computer to your serial cable.

- A network card and a network cable if you intend to communicate with the host using a local network.

Software requirements.

- The RATWIN.EXE progrram, this is the Rat for Windows program.

- The RAT.HLP file. This is the help, and the one you are currently browsing.

- The RATSETUP.EXE program, this is the setup program used to install and/or update Rat on you computer.

- A Winsocket.DLL if you intend to communicate over a network. This software is developed by a number of developers, your systems adinistrator will be able to tell you which one you should use.

Installing Rat

It is very easy to install Rat, just do this:

o Mount the delivery diskette, the best thing would be to use a copy of the delivery diskette.

o Select "Archive | Run" in the program manager.

o Type "A:RATSETUP" if the diskette is mounted in the A: driver, or "B:RATSETUP" if the diskette is mounted in the B: driver. Then click OK.

The RATSETUP program will take some time to start, when its windows is displayed you will be asked to type the name of the directory you want Rat to be installed in. The preferred directory is "C:\RATWIN", use this directory if possible since future releases may depend on it.

You will be given the three choices, e.g.three buttons to click on.:

1) Use "Install" if you do not already have Rat installed
2) Use "Update" if Rat is already installed, and you only want to update Rat on your disk.
3) Use "Install icon only" if Rat is already on disk, but does not ave an icon in any program manager window.

You will se a window with the Rat icon in it if you installed, when Ratsetup has finished. See command line arguments for information on how to install rat icons for different settings.

Settings

Rat has extensive settings possibilities. This means Rat will work in a great variety of ways, whichever suits your specific environment best.

Settings can be done either by using the the dfferenet choices in the "**settings**" menu, or by using the SET script command, (see script commands.)

A setting you always must do is to set your communications way, e.g. if you want to use a serial or network connection, as well as where and how it is configured. Do this under the menu "**settings | line...**".

You also need to tell Rat what type of terminal you intend to emulate. Unix computers regularly use the Vt100 terminal type, a Bull computer might use the Vip7800 terminal type. Use menu "**settings | terminal types**" to set the correct terminal type.

The Settings menu is probably the most fun place since you will be able to tell Rat what screen colors you want to use; what fonts you want to use on different parts of the screen; if you want to have 3d effects on terminal graphics; if you want margins around the terminal area etc. Play around for a while and try the different viibile settings until you find a combination you like.

Some of the settings are more advanced in that you need an indepth understanding of how a Windows program works, or how a connection works. The standard settings should meet most needs, but there are times when some of these has to be changed. Most of the advanced settings are grouped under "**settings | terminal specific**".

You can use different settings when acccessing different hosts, save the settings by selecting "**File | Save as..**". Change to another setup by selecting "**File | Open..**", this will save you current setting and then switch the one you selected. Also see the Rat icon for information on how to choose different settings right from the start.

Scripts

Scripting is a way to make Rat work automatically, almost like using a program. Running a script is called "executing a script".

Scripts can be stored in either script files or as part of buttons. The choices in the "**script**" menu will give you the possibilty to execute script commands by typing them directly, or by executing a script file. The most common method is probably to execute a script by clicking on a button (see handling buttons).

First of all some rules: Each script line consists of one or more script commands, if you use several script commands in a line, the standing bar character ("|") should be used to separate the commands. Script commands can not span over several lines. The § character is used to tell Rat that the remainder of the line are remarks only, this is useful for command like SEND where number of arguments are not defined.

Upper and lowercase characters can usually be mixed, unless otherwise stated, i.e. Rat is case insensitive when executing script commands. You may insert spaces and tabs at your wish anywhere to increase readability.

Each command has a given syntax, or a predefined way that it should be written in. The syntax is defined at the description of each command in this help file. Any part of the command syntax enclosed within parantheses may be omitted, it is optional. If two or more text strings in the syntax definition are separated by a bar ("|") one of the text strings should be choosen.

Scripts arguments are much like spreadsheet cells. They can contain numerical values or text. Rat scripts can use arguments and argument commands to control the execution of scripts.

Script commands To show script commands groups
Script examples Examples to use in your own scripts

Handling buttons

Defining a button.

When you have decided that you want to execute a script by clicking on a button this is what you need to do:

o Select the area where you want the button to be placed on screen by using the mouse. The area will be inverted.

o Select "**edit | define as button**", this will bring up the define button dialog.

o Type the caption text you want to be placed inside the button in the button text field, if you want the button to be invoked when pressing a certain alt-key, type the key in the alt-key field.

o Click the "**script**" button, this will invoke the notepad.

o Enter all the script commands you want to execute when the button is clicked.

o Close the notepad, and click the "**OK**" button when the define button dialog is displayed.

o Execute the button script by clicking on the button or by typing the alt-key combination if you have such one defined.

Modifying a button.

If you need to modify button text or a button script do this.

o Select the button by holding down the ctrl-key and clicking on it at the same time, this will bring up the define button dialog.

OR

o Select "**edit | select buttons**", then click the button you want to change, this will bring up the define button dialog.

o Do any changes necessary, and click the "**OK**" button when you are done.

o Again select "**edit | select buttons**", if you used this method of selecting a button.

Command line, the Rat icon

When Rat is installed a Window and a Rat icon with the default setup file is installed for you. After some time you might find that you need to start Rat in different ways. Instead of always starting the default setup file and then selecting a new setup file from "**File | Open..**" you could create a new Rat icon that automatically uses the correct setup. Do this:

o Copy the existing rat icon by pressing ctrl and at the same time drag the existing icon to a new place.

o Open the information windows for the new icon by pressing alt+return, or by selecting "**Archive | Information**" in the program manager.

o Add the path to the settings file in the command line field, settings files must have a ".set" suffix. If there already is a settings path in the command line field, just remove it and insert the path for your new settings file.

o Select one of the many icons that Rat has, usng different icons for different settings will prehaps make things a bit easier for you.

o Click the "**OK**" button.

o Try the new icon by double clicking it, verify that you got the settings you wanted by selecting "**Help | Setup info**".

Some other command line arguments are:

o "xxx.but" the Button file "xxx" will be used automatically.
o "yyy.nat" the national character conversion file "yyy" will be used instead of the language defined in the setup.
o "do z--z" the script file "z--z" will automatically be excuted at startup.

There are some more commands, but they will not be covered here.

Script examples

Choose a script example that you need o either paste in to your own script, or just want learn more about.

*** Example on how to use a script to call a certain number**

*

* The call will be made from a modem connected at com1, 2400 bps

* First set the correct settings

*

```
    set   com1 7data 1stop evenpar xon off Vip7200 2400bps
```

*

* Then instruct modem to call, first set it into ready position

*

```
    clear screen            § clear screen first
    send    atx1            §  use extended response codes
    sendcr                  §  Into ready position
    sendcr atdt12345678 §   call numbr 12345678 (insert correct)
```

*

* Now wait for a connect or no carrier text from the modem,

* issue an error message if we could not connect

*

```
    wait for CONNECT "NO CARRIER"       §   wait for connect, no carr
    ifnotsee "CONNECT" then Errormessage "No connection" | Stop
```

*

* Success, call got through, now continue ...

*

```
* Example on how to display and use results from a dialog
*
*    This example will show you how to display a dialog and
*    ask user for some input data
*
*    We want to use the dialog named "GetData", since that dialog has
*    one leadtext field (no 501) and one input field (no 201).
*
     DialogText 201 ''              § Empty the input field
     DialogText 501 'Choose telephone number to call'
*
*   Display dialog and get input data
*
     Do Dialog GetData
*
*   Check result of the dialog, input data will reside in argument
*   named dialog201
*
     ifnotok then stop              § exit if user did not press ok
     if '&dialog201' = '' then stop          § stop if empty field
*
*   User entered data, now continue by starting the callnr.scr script
*
     do script callnr.scr '&dialog201'
```

Script commands

Choose script command subjects you need to know more about.

Argument commands	commands to handle script arguments
Conditional commands	if commands
Clipboard commands	commands that let you use the clipboard
Execution break commands	commands used to control execution in script files
File commands	commands used to handle files with
Miscellaneous commands	commands for this and that
Screen commands	commands used to control screen appearance
Send data commands	send data to host commands

Miscellaneous commands

Below are the general script commands available.

* / §	Remark line, for comments
|	Separator for multiple commands on one line
Beep	Send a beep to the speaker
Disconnect	Close,disconnect, the com port and terminate
Reconnect	Disconnect, then open the same connection again
Set	Alter settings
Seton/Setoff	Set settings to on or off

File commands

Below are the commands used to do file handling with.

lfeof/lfnoteof	Check if no more or more data in input file

Execution break commands

Below are the commands used to break execution in script files.

Label	Label a certain line in a script file. Use with the Goto command
Goto	Go to a certain label in a script file
Exit	Exit from current script file
Stop	Stop script execution
Debug	Script debugging
Do script	Start execution of a script file

Screen commands

Below are commands used to control screen appearance.

Clear screen	Clear screen
Winmax	Maximize screen, so that it covers the entire screen
Winmin	Minimize screen, so that only is displayed as an icon

Clipboard commands

Below are the commands used to handle the clipboard commands.

Copy	Copy part of screen content to the clipboard
Paste	Paste data from the clipboard
Save	Save data from com port to clipboard
Nosave	Stop saving data from port to clipboard
Select	Select an area on screen
Deselect	Deselect a selected area

+**Conditional script commands**

Below are the conditional script commands available.

if .. then .. else	Compare two numeric or alphanumeric values and execute either "then" or "else" command
ifgot / ifnotgot	Check if a certain character has / has not been received
ifdata / if notdata	Check if data has / has not been received since last check
ifselect / ifnotselect	Check if an area has / has not been selected on screen
ifsee / ifnotsee	Check if a certain text string can / can not be seen on screen
ifok / ifnotok	Check if user pressed / did no press the OK button in previous dialog
ifcancel / ifnotcancel	Check if user pressed / did not press the Cancel button in prev dialog
iftrue / ifnotfalse	Check if an argument is true
iffalse / ifnottrue	Check if an argument is false
ifhave / ifexist	Check if the argument exists
ifnothave / ifnotexist	Check if argument does not exist
ifcontain / ifnotcontain	Check if argument contains / does not contain a certain string
ifhavefile / ifnothavefile	Check if a certain file exists / does not exists
ifinfile / ifnotinfile	Check if a certain file does / does not contain a certain string
ifsame / ifnotsame	Check if two arguments are / are not equal
Onstring	Execute a certain script command when a certain string is displayed

Send data commands

Below are the send data commands available

Send Send data
Sendcr Send data followed by a carriage return
Sendesc Send Escape followed by data and carriage return
Sendline Send data followed by carriage return and line feed
Sendcomm Send a sequence of special key characters
Sendscript Send an execute script command
Sendalarm Send alarm
Sendbell Send alarm
Sendid Send terminal emulation mode as a text string

See command key names for names of the commands and how to insert them into your send commands.

Script argument commands

Below are the commands used to handle script arguments.

Add	add a value to an argument
Concat	add a text to the end of an argument
Move	move a text or a value to an argument
Subtract	subtract a value from an argument
Strip	remove a certain text string from an argument

See standard arguments for information on what data Rat automatically holds for you.

Standard arguments

Below is a list containing the names of the standard arguments. When referencing to an arguments value, a "&" introducer should be used, example:

```
move &year to myyear
```

In this example the current year, in the form of "YYYY" will be moved to the "myyear" variable.

Argument name	.. contains
x	the current cursor X position, i.e. cursor column
y	the current cursor Y position, i.e. cursor line
screenline	how many lines there are on the screen
screenchar	how many columens there are on the screen
liney	the text in the line where the cursor is located
linenn	the text in the line nn
lastchar	the last character read from host
selectx1	the left column of a screen selection if have one
selectx2	the right column of a screen selection if have one
selecty1	the top line of a screen selection if hav one
selecty2	the bottom line of a screen selection if hav one
OK	the item vaue for the OK button in dialogs (1)
Cancel	the item value for the Cancel button in dialogs (2)
Dialog	the item value for the button clicked in the last dialog
Dialog1 .. Dialog10	the text entered in the edit field 1 up to 10 in the last dialog
Dialognn	the text entered in the edit field with item number nn,, other than 1--10
time	the current time in the form "hour:min:sec"
date	the current date in the form "yyyy-mm-dd"
year	the current year in the form "yyyy"
yyyy	the current year in the form "yyyy"
yy	the current year in the form "yy"
month	the current month in the form "mm"
mo	the current month in the form "mm"
day	the current day in the month in the form "dd"
dd	the current day in the month in the form "dd"
hour	the current hour in the form "hh"
hh	the current hour in the form "hh"
min	the current minute in the form "mm"
mi	the current minute in the form "mm"
sec	the current second in the form "ss"
ss	the current second in the form "ss"
1 .. 9	the command line argument #1 .. #9

COPY command

This command will copy data from screen to clipboard.

--> Syntax : **COPY (ADD|KEEP)**

If no area on screen has been selected, the whole screen will be copied to the clipboard. If "add" or "keep" is used, the data will be added after the data already residing in the clipboard, otherwise all data residing in clipboard will be replaced.

Example :

```
deselect
copy add
```

The entire screen will be copied to the clipboard.

PASTE command

This command will paste data from clipboard, and send it to the host.

--> Syntax : **PASTE (LF)**

If text resides in the clipboard it will be pasted and sent to the host. If "LF" is defined, each carriage return in the clipboard will be followed by a Line Feed character. This is sometimes useful for some Vt100 terminal applications. Stop an ongoing "paste" by pressing a key on the keyboard.

Example :

paste

All text data in clipboard will be sent to the host computer.

SELECT command

This command will select a rectangular area on the screen, just as if you used the mouse to select.

--> Syntax : **SELECT (ALL) (LINE line1 ((TO) line2)) (ROW row1 ((TO) row2))**

"All" means the whole screen will be selected. "Line" and "Row" are used to select certain lines and rows. If "Line2" is not given then selection will only be "line1", if "Row2" is not given then selection will only be "row1".

Example :

```
select line 1 row 10 to 15
```

Select a rectangular area covering line 1, with rows 10 to 15.

DESELECT command

This command will deselect any selection on the screen.

--> Syntax : **DESELECT**

Example :

deselect

ADD command

Add a value to one or more user defined arguments.

--> Syntax : **ADD value (TO) argument1 (argument2 ..)**

"Value", which should be numeric, will be added to "argument1" and "argument2" etc. Any argument that does not exist will be created, and given a value of zero.

Note that normally you should not use the "&" character before the argument name. If you do Rat will first insert the content of the argument into the script line BEFORE, not WHEN the command is executed.

Example :

```
add 1   to   my_variable
```

1 is added to the user defined argument "my_variable".

CONCAT command

Add a text string to the end of one or more user defined arguments.

--> Syntax : **CONCAT (UPPERCASE|UCASE) text (TO) argument1 (argument2 ..)**

"Text" will be added to "argument1" and "argument2" etc. "Text" will first be converted to uppercase if "Uppercase" or "Ucase" is given. Any argument that does not exist will be created, and given a content of an empty string.

Note that normally you should not use the "&" character before the argument name. If you do Rat will first insert the content of the argument into the script line BEFORE, not WHEN the command is executed.

Example :

```
concat ".SCR"  to my_file1  my_file2
```

The text ".SCR" is added to the end of the user defined arguments "my_file1" and "my_file2".

MOVE command

Replace the contents of one or more user defined arguments wirh a defined string.

--> Syntax : **MOVE (UPPERCASE|UCASE) text (TO) argument1 (argument2 ..)**

"Text" will be moved to "argument1" and "argument2" etc. "Text" will first be converted to uppercase if "Uppercase" or "Ucase" is given. Any argument that does not exist will be created, and given a content of an empty string or zero. This will probably be the way that you define an argument that you want to use.

Note that normally you should not use the "&" character before the argument name. If you do Rat will first insert the content of the argument into the script line BEFORE, not WHEN the command is executed.

Example :

```
move   "\windows\write.exe"   to my_command
```

Move the text "\windows\write.exe" to the user defined argument "my_command".

```
move   &my_arg1   to my_arg2
```

Move the contents of the user defined argument "my_arg1" to user defined argument "my_arg2".

SUBTRACT command

Subtract a value from one or more user defined arguments.

--> Syntax : **SUBTRACT value (FROM) argument1 (argument2 ..)**

"Value", which should be numeric, will be subtracted from "argument1" and "argument2" etc. Any argument that does not exist will first be created, and given a value of zero.

Note that normally you should not use the "&" character before the argument name. If you do Rat will first insert the content of the argument into the script line BEFORE, not WHEN the command is executed.

Example :

```
subtract  1  my_variable
```

1 is subtracted from the user defined argument "my_variable".

STRIP command

Remove a text string from one or more user defined arguments.

--> Syntax : **STRIP /ALL) (UPPERCASE|UCASE) text (FROM) argument1 (argument2 ..)**

First or "all" ocurrences of "text", will be removed from "argument1" and "argument2" etc. "Text" will first be converted to uppercase if "Uppercase" or "Ucase" is given. Any argument that does not exist will first be created, and given a a content of an empty string.

Note that normally you should not use the "&" character before the argument name. If you do Rat will first insert the content of the argument into the script line BEFORE, not WHEN the command is executed.

Example :

```
strip all "_" from name1
```

All "_" characters will be removed from user defined argument "name1".

BEEP command

Use this command to sound the internal speaker.

--> Syntax : **BEEP (number-of-beeps)**

Yoy may sound the speaker just once, or a number of times by inserting a value ("number-of-beeps"). Remember not to use large number of beeps since all activitity stops during the beep period.

Example :

```
beep 5
```

The internal speaker will beep 5 times.

*** and § commands**

Use these commands to tell Rat that the line should be treated as a remark line.

--> Syntax : ***** **remarks..**

--> Syntax : **§** **remarks..**

Note that "§" may be used to signal that the rest of a line is a comment (remark), as opposed to the "*" that may only be used at the beginning of script lines.

Example :

```
* Author : CE     created : March 31, 1992
```

A comment line containing revision information.

```
send "abc"   §   this text will not be sent
```

The text "abc" will be sent, but "this text will not be sent" will not be sent.

| (STANDING BAR) command separator

Use this command as a separator if you need to place several script commands on one line.

--> Syntax : **comnmand | next command..**

Example:

```
if &a = 1 then  move 2 to b | move 3 to c  | move 4 to d
```

If the variable "a" contains the value "1!, then place "2" in the variable "b", "3" in "c" etc...

DISCONNECT command

Used this command to close the connection and terminate the session.

--> Syntax : **DISCONNECT**

This is the same as pressing alt+F4. Do not confuse it with the STOP command (that command will only stop the current script execution).

Example :

```
disconnect
```

RECONNECT command

Use this command when you want to hang up the connection, then connect again.

--> Syntax : **RECONNECT**

This is command is sometimes useful when you want to be sure that a modem connection really is closed.

Example :

reconnect

CLEAR command

This command will clear or reinitialize things.

--> Syntax : **CLEAR target...**

Only one target can be cleared in the same command. Valid targets are :

Screen	Removes all text from the screen
Buttons	Removes all buttons from the screen
Icons	Removes all icons from the screen
Menus	Removes all user defined menus
Onstrings	Removes all active Onstrings
Read	Zeroise the character count table used for the IFGOT and IFGOTNOT commands.

Example :

```
clear   screen
clear   menus
clear   buttons
```

WINMIN command

This command will minimize the screen, i.e. the same as choosing "Minimize" from the menu.

--> Syntax : **WINMIN**

Use "ctl-ESC" if you need to enlarge the screen again.

Example :

```
Winmin
```

WINMAX command

This command will maximize the screen, i.e. the same as choosing "Maximize" from the menu.

--> Syntax : **WINMAX**

Example :

Winmax

LABEL command

To mark a position in a script file.

--> Syntax : **LABEL name...**

Use the Goto command to jump to a certain position in a script file. It is possible to use the same "name" several times, but Rat will always search for the label from the beginning of the script file. Upper lower case characters can be mixed, Rat will ignore the case when looking for labels.

Example :

```
Label Loop
...
        goto loop
```

First a label named "loop" is defined, later after some execution has been done (...), the "goto" command will cause Rat to continue exection at the "Loop" line.

GOTO / GO TO commands

To continue execution at a certain position in a script file.

--> Syntax : **GOTO labelname...**

--> Syntax : **GO TO labelname...**

Use the Label command to mark a certain position in a script file. Rat will always search for the label from the beginning of the script file, case will be ignored when looking for labels. If the label is not found in the current script file, the current script file will be exited, and search will start in calling script file if such exist. So remember to double check label names !

Example :

```
if &a = 1 then goto Ais1
```

Continue execution at the label named "Ais1" if user defined argument has a value of 1.

EXIT command

To discontinue, i.e. stop, execution of a script file.

--> Syntax : **EXIT**

If the command resides in a script file the file will be closed and execution will continue at the command after the do script command that started the script file. If the command was issued at the highest level, i.e. via a button script or from the script menu, all script execution will stop.

Example :

```
ifnothave &1 then exit
```

Discontinue execution if caller did not supply script file argument number 1.

STOP command

To discontinue, i.e. stop, all current script execution.

--> Syntax : **STOP**

All currently opened script files will be closed, and all waiting script commands will be flushed.

Example :

```
if &hour > 20 then stop
```

Stop all script exeuction if time is after 8 o'clock in the evening.

DEBUG commands

To turn debugging mode on or off.

--> Syntax : **DEBUG ON**

--> Syntax : **DEBUG OFF**

If debugging mode is turned on, each script command will be displayed in a debug dialog before it is executed. The debug dialog allows you to change the script command, stop debugging, stop script execution etc.

Example :

Debug on

IF .. THEN .. ELSE command

This command will compare two values, numerically or alphabetically, and execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IF argument1 operator argument2 (THEN command1) (ELSE command2)**

Rat will compare argument1 with argument2 using operator. If comparison is true the THEN command will be executed, if present. If comparison is false the ELSE command will be executed if present. Comparisons can be either numerical or alphabetical, Valid operators are :

>	True if arg1 is numerically greater than arg2
<	True if arg1 is numerically less than arg2
<= , =<	True if arg1 is numerically less than or equal to arg2
>= , =>	True if arg1 is numerically greater than or equal to arg2
= , ==	True if arg1 is numerically or alphabetically equal to arg2
<>	True if arg1 is numerically or alphabetically different from arg2

Example :

```
if 10 > 20 then do script a.scr
```

Execute script a.scr if 10 is greater than 20, (who knows if it always will !)

```
if &myrecord = 'END' then goto A else beep
```

Goto label A if internal area "myrecord" contains the text "END" , otherwise beep the internal speaker.

IFGOT / IFNOTGOT .. commands

This command will check a table of received characters and see whether a certain character has been received or not on the serial line, then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFGOT character (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTGOT character (THEN command1) (ELSE command2)**

The "character" can either be a a character or the numerical representation of a character. I.e. a "010" means the LF character. Use the CLEAR READ command to initialise the count table.

Example :

```
ifgot "a" then beep
```

Beep if the character "a" has been received.

```
ifnotgot 013 then goto loop
```

Goto label Loop if carriage return (character 013) has not been received yet.

IFDATA / IFNOTDATA .. commands

This command will check if any data has been received or not on the serial line, then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFDATA (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTDATA (THEN command1) (ELSE command2)**

The internal data received counter will be reinitialised after each check, this means that you do not have to do a clear command to initialise it, but it also means that the value in the counter is zeroised every time you check it.

Example :

```
ifdata then sendcr "Welcome"
```

IFSELECT / IFNOTSELECT .. commands

This command will check if user has selected an area on the screen or not on, then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFSELECT (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTSELECT (THEN command1) (ELSE command2)**

The user uses the mouse or the SELECT script command to select an area on screen,.

Example :

ifselect then copy

Copy a screen area if it has been selected.

IFSEE / IFNOTSEE .. commands

The command checks whether a certain string is visible on the screen or not, then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFSEE string (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTSEE string (THEN command1) (ELSE command2)**

Remember to enclose "string" wihtin quotes if if contains embedded blanks.

Do not confuse this command with the Onstring command.

Example :

```
ifsee "welcome home" then sendcr "My name"
```

Send "My name<cr>" if "welcome home" id displayed on the screen.

```
ifnotgot 013 then goto loop
```

Goto label Loop if carriage return (character 013) has not been received yet.

ONSTRING commands

The command defines a script command to be executed when a certain text string is visible on screen. Rat will not wait for the string to appear on screen, execution will continue with the next command without any delay.

--> Syntax : **ONSTRING string script command..**

"script command" will be executed when (and if) "string" is displayed anywhere in the terminal area. Remember to enclose "string" within quotes if it contains embedded blanks. "Script command"may consist of several script commands if commands are separated by the standing bar character ("|"). The search for "string" is case insensitive, i.e.

There may be up to 10 different onstrings active at the same time, this may degrade performance though since Rat has to check the screen contents as often as possible when data has arrived from the host computer.

Once "script command" has been filed for execution, the onstring command will be removed.

Do not confuse this command with the Ifsee and Wait for commands.

Example :

```
onstring "NO CARRIER" stop
```

Execute the "stop" command when "NO CARRIER" is displayed on screen.

```
onstring "Logon" sendcr "L ME" | delay 1 s | sendcr "PASSWRD"
```

When "Logon" is displayed send "L ME"<cr>, wait 1 second then send "PASSWRD"<cr>.

IFOK / IFNOTOK .. commands

This command will check if user pressed the OK button or not last time a user initiated dialog was displayed, then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFOK (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTOK (THEN command1) (ELSE command2)**

This will only handle dialogs that was started using the DO DIALOG script command.
Example :

```
ifok then clear menus
```

IFCANCEL/ IFNOTCANCEL .. commands

This command will check if user pressed the Cancel button or not last time a user initiated dialog was displayed, then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFCANCEL (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTCANCEL (THEN command1) (ELSE command2)**

This will only handle dialogs that was started using the DO DIALOG script command.
Example :

```
ifcancel then exit
```

IFTRUE / IFNOTFALSE .. commands

This command will check a n argument to see whether it is true or not , then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFTRUE argument (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTFALSE argument (THEN command1) (ELSE command2)**

The command will probably be used when checking the contents of a user defined argument or an internal argument. True means that the argument has the value "TRUE".

Example :

```
iftrue &myvariable then beep
```

Beep if the user argument "myvariable" contains the string "TRUE".

IFFALSE / IFNOTTRUE .. commands

This command will check an argument to see whether it is false or not , then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFFALSE argument (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTTRUE argument (THEN command1) (ELSE command2)**

The command will probably be used when checking the contents of a user defined argument or an internal argument. False means that the argument has the value "FALSE".

Example :

iffalse &keyclick then set keyclick to true

Set keyclick to true if it is false.

IFHAVE / IFEXIST .. commands

This command will check whether an argument is present or not , then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFHAVE argument (THEN command1) (ELSE command2)**

--> Syntax : **IFEXIST argument (THEN command1) (ELSE command2)**

The command will probably be used when checking the contents of a user defined argument or a script file argument. The comparison is true if the argument is present.

Example :

```
ifexist &1 then goto have_arg_1
```

Goto Label have_arg_1 if caller supplied script file argument 1.

IFNOTHAVE / IFNOTEXIST .. commands

This command will check whether an argument is missing or not , then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFNOTHAVE argument (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTEXIST argument (THEN command1) (ELSE command2)**

The command will probably be used when checking the contents of a user defined argument or a script file argument. The comparison is true if the argument is missing.

Example :

```
ifnotexist &1 then stop
```

Stop execution if caller did not supply script file argument 1.

IFCONTAIN / IFNOTCONTAIN .. commands

This command will check whether a certain string can be found in another string , then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFCONTAIN source searched (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTCONTAIN source searched (THEN command1) (ELSE command2)**

Rat will check if the "searched" string can befound anywhere in the "source" string. Use it to check either data entered in dialogs, data sent from the host computer, script file arguments, file records etc.

Example :

```
ifcontain &my_variable "abc" then beep
```

Beep if the user defined argument "my_variable" contains the string "abc".

IFHAVEFILE/ IFNOTHAVEFILE .. commands

This command will check whether a certain file exists or not, then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFHAVEFILE file (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTHAVEFILE file (THEN command1) (ELSE command2)**

If the "file" exists in another directory the directory characters has to be used. Note that the current directory can be changed using the Do program/Do Dos script commands.

Example :

```
ifhavefile "c:\winword\winword.exe" then do dos \winword\winword
```

Start the Word for Windows program if it exists.

IFINFILE / IFNOTINFILE .. commands

This command will check whether a certain string can be found in a certain file or not , then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFINFILE file searched (THEN command1) (ELSE command2)**

--> Syntax : **IFNOTINFILE file searched (THEN command1) (ELSE command2)**

Rat will check if the "searched" string can befound anywhere in the "file". If the "file" exists in another directory the directory characters has to be used. Note that the current directory can be changed using the Do program/Do Dos script commands.

Example :

```
ifinfile "\windows\my.fil" "user:REKO" then beep
```

Beep if the string "user:REKO" exists anywhere in the file "\windows\my.fil".

IFSAME / IFEQUAL / IFNOTSAME / IFNOTEQUAL .. commands

This command will check whether two strings are alphabetically equal or not , then execute either the THEN command or ELSE command depending on result of comparison.

--> Syntax : **IFSAME string1 string2 (THEN command1) (ELSE command2)**
IFEQUAL string1 string2 (THEN command1) (ELSE command2)

--> Syntax : **IFNOTSAME string1 string2 (THEN command1) (ELSE command2)**
IFNOTEQUAL string1 string2 (THEN command1) (ELSE command2)

Note that the comparison will be alphabetical and not numerical, use the If string1 = string2 if numerical comparison is necessary. In a numerical comparison "1" will be equal to "001", but not in an alphabetical comparison.

Example :

```
ifsame &the_area "success" then do script success.scr
```

Start the script file "success.scr" if the user defined argument is the string "success".

SEND data commands

Use this command to send strings of data to the com port.

--> Syntax : **SEND data**

All data in the line will be sent, remember to use quotes if you have embedded blanks. Data can be sent in different ways depending on the command. The allowed commands are :

Send	Plain send of data
Sendcr	Send data followed by a carriage return
Sendline	Send data followed by a carriage return and line feed
Sendesc	Send escape followed by data and carriage return
Sendcomm	Send a sequence of special key characters
Sendscript	Send an execute script command
Sendalarm	Send alarm
Sendbell	Send alarm
Sendid	Send terminal emulation mode as a text string

Examples:

```
send   abc   123
```

send "abc 123", note : one space between the words since quotes were not used

```
sendcr   atdt123456
```

send "atdt123456" followed by a carriage return

```
sendcomm   cr,down,down
```

Send the special key sequences for keys cr (carriage return), and down arrow twice.

See command key names for names of the commands.

Command key names

The command keys will be useful in script send commands where you want to send a sequence of special keys or a pf key to the host computer. Below is a list of the allowed special key names.

You must enter a special key introducer ("«") before a sequence of special keys or pf keys. This is due to the fact that Rat has to be able to differ between ordinary text and a special key name, the sequence must be terminated by the "»" character. Separate the key names with a comma, example:

send «cr,down,down,cr,pf1»

The special key introducer is entered this way:

Hold down the alt-key and type 174 on the numerical keyboard.

The special key terminator is entered this way:

Hold down the alt-key and type 175 on the numerical keyboard.

Note that the special key names below are case insensitive. Also note that if there is a programmable setting for the key, then the script command defined for the key will be executed instead of sending the standard special key sequence defined for the terminal type.

Special key name	**...will send to host**
pf1 .. pf12	f1 .. f12
p1 .. p12	f1 .. f12
f1 .. f12	f1 .. f12
sf1 .. sf12	shift+f1 -- shift+f12
spf1 .. spf12	shift+f1 -- shift+f12
ctl-a .. ctl-z	ctl-A .. ctl-Z
ctla .. ctlz	ctl-A .. ctl-Z
^a .. ^z	ctl-A .. ctl-Z
Üa .. Üz	ctl-A .. ctl-Z
alt-a .. alt-z	alt-a .. alt-z
alta .. altz	alt-a .. alt-z
Brk	break
Bs	backspace
Bell	a bell (ctl-G)
Cr	Cr
Return	Cr
OK	Cr
esc	escape
Cancel	escape
tab	tab
xmit	xmit (enter)
enter	xmit (enter)
lf	line feed
ff	form feed
eol	eol
eop	eop
home	home
clr	clear
del	del

down	down arrow
up	up arrow
left	left arrow
right	right arrow
reset	reset
backtab	shift+tab
print	print
sysreq	sys req
att	attention
wait	will delay all execution for one second
wt	will delay all execution for one second

SET command

The settings command is used to alter different Rat settings.

--> Syntax : **SET settings...**

Several different settings can be set in the same SET command. Note that the "/" character below is used to define one of two ways to print the name of the settings, either in short or normal form. Valid settings are :

Vip7200 / 7200	emulate the Bull Vip7200 terminal
Vip7800 / 7800	emulate the Bull Vip7800 terminal
Dku7001 / 7001 | 7001	emulate the Bull Dku7001 terminal
Dku7002 / 7002	emulate the Bull Dku7002 terminal
Dku7102 / 7102	emulate the Bull Dku7002 terminal
Vt52	emulate the Digital Vt52 terminal
Vt100	emulate the Digital Vt100 terminal
Videotex / Vtex	emulate the Prestel Videotext terminal
Tty	emulate a TTY terminal
Com1 | Com2 | Com3 | Com4	defines which Com port to use
n-nbps	defines line speed
1stop, 2stop	defines number of stop bits
nopar / noparity	defines no parity to be used
even / evenpar / evenparity	defines even parity to be used
odd / oddpar / oddparity	defines odd parity to be used
7data, / 8data	defines number of data bits
xon on | off	defines xon should/should not be used
LA / Language x--x	defines which character set (language) to use, supported languages can be found in the Settings;Language menu
KeyClick / KC on | off	set key click on/off, i.e. a click will sound each time a key is pressed
ButtonClick / BC on | off	set button click on/off, i.e. a click will sound each time a button is pressed

Examples :

```
Set Vt100 com1 1200bps 8data 1stop nopar xon off language US_ASCII
```

This command will set Vt100 emulation, use com1 with 8 data bits, 1 stop bit, no parity , no xon and character set will be US Ascii.

SETON / SETOFF commands

This is a shortcut version of the SET command. It is used to set settings on or off.

--> Syntax : **SETON settings...**

--> Syntax : **SETOFF settings...**

Several different settings can be set on or off in the same SET command. Valid settings are :

Xon	turns xon on or off
KeyClick / KC	set key click on/off, i.e. a click will/will not sound each time a key is pressed
ButtonClick / BC	set button click on/off, i.e. a click will/will not sound each time a button is pressed

Examples :

```
SetOn keyclick
Setoff xon
```

These commands will turn keyclick on, and xon off.