

kixstrip

scripting@wanadoo.nl

Kixstrip 3.20

- Description
- Goals
- Important remarks
- Important restrictions
- Parameters
 - ?
 - /Block_Check
 - /Combine
 - /Debug
 - /Headers
 - /License
 - /Performance
 - /Print
 - /Progress
 - /Show_Errors
 - /Show_Structure
 - /Translate
 - /Tab
- Directives
 - ;(\$begin)
 - ;(\$debug line)
 - ;(\$debug off)
 - ;(\$debug on)
 - ;(\$end)
- List of possible errors
- List of possible screen output (example)
- List of possible summary report(s) (example)
- FAQ
 - *Can we solve possible warnings about **incomplete lines** automatically?*
 - *Which elements may slowdown your script?*
 - *What is the performance of kixstrip?*
 - *What to do when you want to reformat a 'combined script' with too much keywords on one line?*
 - *What to do by problems with kixstrip?*
 - *How can you get informed about new releases?*
 - *What is very different with other debug capabilities?*
 - *What are the general calls?*
- Examples
 - for **compressing** your KiXtart code
 - for **reformatting** your KiXtart code
 - for **debugging** your KiXtart code + how to debug two or more calling scripts
 - for **performance analysis** of your KiXtart code
 - for **progress analysis** of your KiXtart code
- Published versions
- Release notes
- Future/to-do activities

Kixstrip 3.20

Description

It is a tool with a lot of capabilities for KiXtart code. So it can:

- check your code on correctness.
- compress your code by removing unwanted statements and symbols and combine it to one line.
- reformat your code.

Also for developers it makes it possible:

- to debug your code.
- to do a performance or progress analyse.

Kixstrip 3.20

Goals

- **checking** your KiXtart code on correctness. Sometimes you are missing a single parenthesis, square bracket, single quotation or double quotation as symbol or your block structure aren't used correctly.
- **compress** a KiXtart script by removing unwanted comment statements, print-statements (= lines started with a "?" character), empty lines, tab-symbols and spaces. After that operation it will combine it to **one line**. Such line doesn't have any limitation in length, which is very unique. It can reduce the size of your script dramatically.
- **reformat** your KiXtart script in an identical way.
So:
 - it will handle block structures and make the right indentations.
 - it will translate KiXtart commands to uppercase and KiXtart macros and variables to lowercase.
(**remark:** the **kixref** tool handle such translation also for KiXtart functions)
- insert **debugging** code to your KiXtart code, which make it possible to show:
 - which lines were executed,
 - what was the time of execution,
 - what was the actual @error & @serror status after the previous line was executed.
- insert **performance** code to your KiXtart script which make it possible to see
 - how many lines were executed per second,
 - how many lines are available and were executed,
 - which part is using the greatest amount of statement calls. the result will be showed in terms of percentages.
(sometimes Do/Until, For/Next & While/Loop can increase it. If/Else/Endif & Select/Case/EndSelect can decrease it)
- insert **progress** code to your KiXtart script, which shows not only the **performance** information at the end, but also a progress bar during execution.

Kixstrip 3.20

Important remarks

- The message **no errors found** means not always: there are no problems (f.e. unreachable end of script, incomplete parameter usage by functions or undeclared/un-initialised variables). Those problems will be creating an unexpected result during execution and it will be handle by KiXtart itself.
- The **kixstrip** call with any parameter will return the help information.
- The last option may overwrite others. As user it isn't possible to make a combination which isn't legal to run.
f.e.
 - /debug, /progress & /performance are three different calls.
 - "kixstrip input.kix output.kix /Tab = 4 /NoTab" means no tabulation.At the end of a run the program will show, which options are/aren't actual used.
An example of call:

kixstrip input.kix output.kix /block_check /show_structure

```
Kixstrip 4.10 (vs 3.20e)      (c) MCA - scripting@wanadoo.nl - 2000, 2001, 2002
      19000 .E(19008)

Kixstrip 4.10 (vs 3.20e)      69.71 (sec)

input          19008 input.kix
output         19008 output.kix (skip: 0 blocks: 0 labels: 4)
block_check    block_structures=3996 labels=4
               tab=6 (default)

Summary KIXSTRIP:          block_structures
                          - do:until                [118:118]
                          - for|each:in|to:step|next [0|0:0|0:0|0]
                          - function:endfunction    [0:0]
                          - if:else:endif          [3849:1137:3849]
                          - select:case:endselect   [16:99:16]
                          - while:loop              [13:13]
                          3996 block_structures found.
Informative KIXSTRIP:    no UDF's found.
Informative KIXSTRIP:    no errors found.

active options:
/Block_Check /Headers /Print /Show_Structure /TAB=6 /Translate
inactive options:
/NoCombine /NoDebug /NoPerformance /NoProgress /NoShow_Errors
/NoLicense:          (not.specified)
```

- You get an incomplete code warning, when f.e. parentheses, brackets and quotations don't stay on the same line.
- Kixstrip363.exe / Kixstrip400.exe / Kixstrip410.exe are using same code. Only the KiXtart keywords-tables are different.
- A block starting with a line with symbol "**;(\$begin**)" and ending with a line with symbol "**;(\$end**)" will automatically removed.
- The options /debug, /progress & /performance are backwards compatible will all previous KiXtart releases.

Important restrictions

- filenames must be 8.3 format.
- destination file will automatically overwritten.
- comment statements starts with a ";" symbol will not be deleted by the option **"/Block_Check"**.
- print statements starts with a "?" symbol will not be deleted by the option **"/Print"**.
- option **"/debug"** can create an error, when a statement use two or more lines to complete a statement.

f.e.

```
WriteLine(1,"Welcome" +  
          @userid)
```

will be

```
? "-1-" WriteLine(1,"Welcome" +  
? "-2-"          @userid)
```

With the kixstrip directives **";(\$debug off)"** and **";(\$debug on)"** it can be bypassed.

- option **"/block_check"** will not reformat internal structures of a line. It will only translate some keywords (commands and macros). Translation can be prevented with the option **"/NoTranslate"**.
To reformat structure use first **"/NoCombine /Print"** as call and secondly **"/Block_Check"** call to get the new structure.
- support only new style of string/variable usage (see KiXtart 3.62 manual page 22 or KiXtart 4.00/4.01 manual page 25).
old style: "User \$user logon at @date"
new style: "User " + \$user + " logon at " + @date

Kixstrip 3.20

Parameters

- **kixstrip** (without)
in previous releases it returns some date information.

```
wed 30-may-2002 18:00:00 week 22 day 150
```

Now it shows help information.

- **kixstrip ?**
it shows help information

```
Kixstrip 4.10 (vs 3.20e)      (c) MCA - scripting@wanadoo.nl - 2000, 2001, 2002
-----
kixstrip  [input] [output]
          /Block_Check or /BC      /Combine          /Debug
          /Headers                 /License          /License=[var]
          /Performance             /Print           /Progress
          /Show_Errors or /SE      /Show_Structure or /SS /Translate
          /TAB=[num] (def: 6)
default: /Combine /Headers /License          (other options are negative)
other:   - options with prefix "/No" will be ignored. f.e. /NoDebug /NoTab
          - /Block_Check /Tab=2    (auto set: /NoCombine /NoLicense /Print      )
          /Debug                  (auto set: /NoLicense /Print /Show_Errors  )
          - /Block_Check          (reformat kixtart code inc. block structure )
          /Combine                 (combine lines to one line                )
          /Debug                  (insert "line numbers" for each printed line)
          /Headers                (insert summary report as footer          )
          /Performance            (insert "indicator" for each executed line)
          /Print                  (do not remove print statements           )
          /Progress               (insert "indicator" for each printed line )
          /Show_Errors            (insert error messages in output file     )
          /Show_Structure         (insert block structures in output file   )
          /Translate              (convert keywords to upper/lower-case     )
examples - kixstrip test.kix test.out /Block_Check /Debug
```

The default options are:

```
/Combine
/Headers
/License:
/NoPrint
```

MCA - scripting@wanadoo.nl - 2000, 2001, 2002

Kixstrip 3.20

- **/Block check, /BC**

it reformats your KiXtart script and make the right indentations for block structures. Also it will translate keywords to upper- and lowercase. KiXtart commands will be uppercase and KiXtart macros and variables will be lowercase. To prevent such translation uses option **"/NoTranslate"**.

The default options are:

```
/Block_Check  
/Headers  
/Print  
/TAB=6  
/Translate
```

Structures after selection of **"/Block_Check"** will be:

- DO
 <command parts>
UNTIL <logical part> <command parts>
 <command parts>
- IF <logical part> <command parts>
 <command parts>
ELSE
 <command parts>
ENDIF
 <command parts>
- SELECT
CASE
 <logical part> <command parts>
 <command parts>
CASE
 <logical part> <command parts>
 <command parts>
CASE
 1 <command parts>
 <command parts>
ENDSELECT
 <command part>
- WHILE <logical part> <command parts>
 <command parts>
LOOP
 <command part>
- FOR EACH <element> IN <group> <command parts>
 <command parts>
NEXT
 <command part>
- FOR <start> TO <stop> STEP <step> <command parts>
 <command parts>
NEXT
 <command part>

Kixstrip 3.20

- FUNCTION <name> (<argument1>, <argument2>, OPTIONAL <argument3>)
 <command parts>
 <command parts>
 ENDFUNCTION
 <command part>

Kixstrip 3.20

- **/Combine** (default option)
it removes unwanted comment statements, print-statements, empty lines, tab-symbols and spaces.
After that operation it will combine it to **one line**. Such line doesn't have any limitation in length, which is very unique.
Labels will always start on a new line.
Prevent remove of print-statements use the option "**/Print**".

The default options are:

/Combine

/Headers

/License:

/NoPrint

MCA - scripting@wanadoo.nl - 2000, 2001, 2002

Kixstrip 3.20

- **/Debug**

it inserts additional code to your KiXtart code, which make it possible to show:

- which lines were executed.
- what was the time of execution.
- what was the actual @error & @error status after the previous line was executed.
- what was the output of that line to your screen.

With the directives "**;\$debug on**", "**;\$debug line**" and "**;\$debug off**" it is possible

- to activate debugging mode. It is the default selection.
- to translate the specified line with debugging code.
- to deactivate debugging mode.

The default options are:

```
/Block_Check
/Debug
/Headers
/Print
/Show_Errors
/TAB=6
/Translate
```

The options /debug, /progress & /performance are backwards compatible will all previous KiXtart releases.

An example of call: **kixstrip input.kix output.kix /debug**

```
; NT/95 create description list of KiXtart errors - Kixtart 4.00

FUNCTION SetError($error_number) EXIT $error_number ENDFUNCTION
;
$tmp_directory=ExpandEnvironmentVars("%tmp%")
IF (Substr($tmp_directory,len($tmp_directory),1) <> "\")
    $tmp_directory=$tmp_directory+"\\"
ENDIF
;
$info_file=$tmp_directory+"kix-code.txt"
IF RedirectOutput($info_file,1)
ENDIF
? "KiXtart "+@kix+" - Summary Error Codes"
?
FOR $i=0 TO 9999
    SetError($i)
    IF (len(@serror) <> 0) AND (Instr(@serror,"retrieving error information for") = 0) AND
        (Instr(@serror," 13D") = 0)

        $result=@serror
        $pos=Instr($result,CHR(13)+CHR(10))
        WHILE ($pos <> 0)
            IF ($pos <> 0)
                $result=substr($result,1,$pos-1)+" "+substr($result,$pos+2,len($result)-$pos-1)
            ENDIF
            $pos=Instr($result,CHR(13)+CHR(10))
        LOOP
    IF (Substr($result,len($result),1)=CHR(10)) OR (Substr($result,len($result),1)=CHR(13))
        ? Substr("          ",1,7-len("@error"))+"@error "+Substr($result,1,len($result)-1)
    ELSE
        ? Substr("          ",1,7-len("@error"))+"@error "+$result
    ENDIF
ENDIF
NEXT
IF RedirectOutput("")
ENDIF
? "Informative KIX: create an actual description list of KiXtart "+@kix+" errors."
? "          file created '"+LCASE($info_file)+"'."
EXIT
```

Kixstrip 3.20

The output file will be:

```
CLS
COLOR C+/N
AT (1,1) " "
IF RedirectOutput("")
ENDIF
? "-" +LCASE(@day) + " " +@date+ " " +@time+ "- kixtart "+@kix+
" /3.04e script starting."
? "-"
? "-curdir: "+LCASE(@curdir)
? "-scriptdir: "+LCASE(@scriptdir)
IF (instr("-4.1x-", "-" +substr(@kix,1,3) + "x-") <> 0)
? "-scriptname: "+LCASE(@scriptname)
ENDIF
? "-startdir: "+LCASE(@startdir)
? "-"
? "-userid: "+LCASE(@userid) + "/" +LCASE(@wuserid)
? "-user priv: "+LCASE(@priv)
IF (instr("-4.xx-", "-" +substr(@kix,1,2) + "xx-") <> 0)
? "-version: inwin="+@inwin+"/dos="+@dos+"/productsuite="+
@productsuite+"/producttype="+@producttype+"/csd="+
LTRIM(RTRIM(@csd))
ELSE
? "-version: inwin="+@inwin+"/dos="+@dos
ENDIF
? "-"
? "- start-"+@time+ "- @error @serror "?
? "- 1-"+@time+ "- @error @serror "? ; NT/95 create description list of KiXtart errors - Kixtart 4.00
? "- 2-"+@time+ "- @error @serror "?
? "- 3-"+@time+ "- @error @serror "?
FUNCTION SetError($error_number)
EXIT $error_number
ENDFUNCTION
? "- 4-"+@time+ "- @error @serror "? ;
? "- 5-"+@time+ "- @error @serror "? $tmp_directory=ExpandEnvironmentVars("%tmp%")
? "- 6-"+@time+ "- @error @serror "? IF (substr($tmp_directory, len($tmp_directory), 1) <> "\")
? "- 7-"+@time+ "- @error @serror "? $tmp_directory=$tmp_directory + "\"
? "- 8-"+@time+ "- @error @serror "? ENDFUNCTION
? "- 9-"+@time+ "- @error @serror "? ;
? "- 10-"+@time+ "- @error @serror "? $info_file=$tmp_directory + "kix-code.txt"
? "- 11-"+@time+ "- @error @serror "? IF RedirectOutput($info_file, 1)
? "- 12-"+@time+ "- @error @serror "? ENDFUNCTION
? "- 13-"+@time+ "- @error @serror "? ? "KiXtart "+@kix+ " - Summary Error Codes"
? "- 14-"+@time+ "- @error @serror "? ?
? "- 15-"+@time+ "- @error @serror "? FOR $i=0 TO 9999
? "- 16-"+@time+ "- @error @serror "? SetError($i)
? "- 17-"+@time+ "- @error @serror "? IF (len(@serror) <> 0) AND
(Instr(@serror, "retrieving error information for") = 0) AND
(Instr(@serror, " ", 13D) = 0)
? "- 18-"+@time+ "- @error @serror "? $result=@serror
? "- 19-"+@time+ "- @error @serror "? $pos=Instr($result, CHR(13)+CHR(10))
? "- 20-"+@time+ "- @error @serror "? WHILE ($pos <> 0)
? "- 21-"+@time+ "- @error @serror "? IF ($pos <> 0)
? "- 22-"+@time+ "- @error @serror "? $result=substr($result, 1, $pos-1)
? "- 23-"+@time+ "- @error @serror "? " " +substr($result, $pos+2, len($result)-$pos-1)
? "- 24-"+@time+ "- @error @serror "? ENDFUNCTION
? "- 25-"+@time+ "- @error @serror "? $pos=Instr($result, CHR(13)+CHR(10))
? "- 26-"+@time+ "- @error @serror "? LOOP
? "- 27-"+@time+ "- @error @serror "? IF (Substr($result, len($result), 1) = CHR(10)) OR
(Substr($result, len($result), 1) = CHR(13))
? "- 28-"+@time+ "- @error @serror "? ? Substr(" ", 1, 7-len("@error")) +
"@error " +Substr($result, 1, len($result)-1)
? "- 29-"+@time+ "- @error @serror "? ELSE
? "- 30-"+@time+ "- @error @serror "? ? Substr(" ", 1, 7-len("@error")) + "@error " + $result
? "- 31-"+@time+ "- @error @serror "? ENDFUNCTION
? "- 32-"+@time+ "- @error @serror "? ENDFUNCTION
? "- 33-"+@time+ "- @error @serror "? NEXT
? "- 34-"+@time+ "- @error @serror "? IF RedirectOutput("")
? "- 35-"+@time+ "- @error @serror "? ENDFUNCTION
? "- 36-"+@time+ "- @error @serror "? ? "Informative KIX: create an actual description list of KiXtart "+@kix+
" errors."
? "- 37-"+@time+ "- @error @serror "? ? " file created '"+LCASE($info_file) + "'. "
? "- 38-"+@time+ "- @error @serror "? EXIT
? "- end-"+@time+ "- @error @serror "?
? "-"
? "- "+LCASE(@day) + " " +@date+ " " +@time+ "- kixtart "+@kix+
" /3.04e script ending."
? "-"
IF (instr("-4.1x-", "-" +substr(@kix,1,3) + "x-") <> 0)
? "- "+@cpu+ " (" +@mhz+ " Mhz, memory "+MemorySize()+ " MB)"
ENDIF
? "-"
; ($begin)
;
; wed 30-may-2002 18:00:00 (kix 4.10 vs 3.04e)
;
; Informative KIXSTRIP: no errors found (input=38 output=38 skip=0).
;
; Informative KIXSTRIP: 9 block_structures found.
; Informative KIXSTRIP: 1 UDF found.
; Informative KIXSTRIP: no labels found.
; Summary KIXSTRIP: BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET GETS GOSUB GOTO OLExxx PLAY
QUIT RETURN RUN SHELL SLEEP THEN USE
; Informative KIXSTRIP: 1 ENDFUNCTION
; Informative KIXSTRIP: 2 EXIT
; Informative KIXSTRIP: 1 FUNCTION
;
; ($end)
```

Kixstrip 3.20

- **/Headers** (default option)
it inserts a summary report as footer. Automatically a footer will be inserted when the program find possible problems.
With the options "**/Show_Errors**" and "**/Show_Structure**" you get also summary reports as footer (see later).

- **/License**
/License = [var]
it inserts a personal license note to the compressed KiXtart script file. With the environment variable **var** you can pass this personal note also.
The program will ask for user input when you only specifies the option **/License**. To suppress the license information in the output file you must specify the option **/NoLicense**.
The default license text is "*MCA - scripting@wanadoo.nl - 2000, 2001, 2002*".

Kixstrip 3.20

- **/Performance**

it inserts code to your KiXtart script which makes it possible to see:

- how many lines were executed per second.
- how many lines are available and were executed.
- which part is using the greatest amount of statement calls.

the result will be showed in terms of percentages. To make it possible we are using RESERVED variables. They start with the prefix "\$perf_".

The default options are:

```
/Block_Check
/Debug
/Headers
/Performance
/Print
/Show_Errors
/TAB = 6
/Translate
```

The options /debug, /progress & /performance are backwards compatible will all previous KiXtart releases.

What code will be inserted after selection of **"/Performance"**:

```
IF (instr("-3.6x-", "-" + substr(@kix,1,3) + "x-") = 0) AND (instr("-4.x-5.x-6.x-7.x-8.x-9.x-", "-" + substr(@kix,1,2) + "x-") = 0)
  IF MessageBox("sorry, your kixtart "+@kix+" release is too old." + CHR(13) + CHR(10) + CHR(13) + CHR(10) +
    " at least run KiXtart 3.60" + CHR(13) + CHR(10) + CHR(13) + CHR(10) +
    " please upgrade .", "KiXtart "+@kix+" info", 4112, 300)
  ENDIF
  EXIT
ENDIF
COLOR C/N
DIM $perf_hits[101]
$perf_cur= 1 gosub "perf_info" <your code>

$perf_cur= 501 gosub "perf_info" <your code>

;($begin)
;
;          sat 20-jul-2002 12:00:00      (kix 4.10 vs 3.20e)
;
;Informative KIXSTRIP:  no errors found (input=1 output=1 skip=0).
;
;Informative KIXSTRIP:  no block_structures found.
;Informative KIXSTRIP:  no UDF's found.
;Informative KIXSTRIP:  no labels found.
;Summary      KIXSTRIP:  BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET GETS GOSUB
                GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;
;($end)

:perf_report
IF (RedirectOutput("") = 0)
ENDIF
CLS
at(2,26) "Kixtart Performance Report "+@kix
box(3,5,5,75,"single")
$perf_column=6
DO
  at(4,$perf_column) " "
  $perf_column=$perf_column+1
UNTIL ($perf_column >= 75)
box(7,5,9,75,"single")
$perf_percentage=( (100*$perf_cur)/$perf_stop)
at(6,30) substr(" ",1,3-len("$perf_percentage")) $perf_percentage " %"
at(8,26) substr(" ",1,7-len("$perf_cur")) $perf_cur " .. " $perf_stop
at(8,49) "total lines: " $perf_total
$perf_stop_time=@time
$perf_start_stime=(3600*substr($perf_start_time,1,2)+60*substr($perf_start_time,4,2)+substr($perf_start_time,7,2))
$perf_stop_stime=(3600*substr($perf_stop_time,1,2)+60*substr($perf_stop_time,4,2)+substr($perf_stop_time,7,2))
$perf_elapse_time=$perf_stop_stime - $perf_start_stime
at(4,25) $perf_start_time " .. " $perf_stop_time
at(4,55) "elapse: " $perf_elapse_time " sec"
IF ($perf_elapse_time = 0)
  $perf_elapse_time=1
ENDIF
```

Kixstrip 3.20

```

at(6,49) "lines/second: " ($perf_total/$perf_elapse_time)
at(11,0)
$perf_info=" %% hits %% hits %% hits %% hits %% hits %% hits %% hits %% hits %% hits %% hits"
$perf_info
$perf_count1=1
DO
  $perf_info=""
  $perf_count2=1
  DO
    $perf_hits_pos=$perf_count1+$perf_count2-1
    $perf_hits_value=$perf_hits[$perf_hits_pos]
    $perf_info=$perf_info+substr(" ",1,3-len("$perf_hits_pos"))+$perf_hits_pos
    $perf_info=$perf_info+substr(" ",1,5-len("$perf_hits_value"))+$perf_hits_value
    $perf_count2=$perf_count2+1
  UNTIL ($perf_count2 > 10)
  at(11+$perf_count1/10+1,0) $perf_info
  $perf_count1=$perf_count1+10
UNTIL ($perf_count1 > 100)
at(22,0)
EXIT
:perf_info ; - performance -
$perf_stop=501
IF ($perf_cur <= 1)
  $perf_count1=0
  DO
    $perf_hits[$perf_count1]=0
    $perf_count1=$perf_count1+1
  UNTIL ($perf_count1 > 100)
  $perf_start_time=@time
  $perf_total=0
ENDIF
$perf_percentage=((100*$perf_cur)/$perf_stop)
$perf_hits[$perf_percentage]=$perf_hits[$perf_percentage]+1
$perf_total=$perf_total+1
RETURN

```

What will you see as a Kixtart performance report:

```

Kixtart Performance Report 4.10
-----
12:00:00 .. 12:00:12          elapse: 12 sec
-----
100 %          lines/second: 407
-----
1203 .. 1203          total lines: 4889
-----

```

```

% hits % hits % hits % hits % hits % hits % hits % hits % hits
1 12 2 12 3 12 4 12 5 7 6 10 7 1 8 10 9 6 10 5
11 0 12 0 13 0 14 4 15 2 16 7 17 4 18 1 19 8 20 9
21 4 22 12 23 10 24 10 25 12 26 2 27 7 28 7 29 12 30 10
31 12 32 12 33 149 34 69 35 90 36 9 37 3 38 7 39 5 40 7
41 8 42 4 43 5 44 8 45 9 46 9 47 7 48 4 49 12 50 4
51 3 52 12 53 9 54 12 55 12 56 12 57 12 58 12 59 111 60 48
61 86 62 12 63 12 64 12 65 82 66 2010 67 556 68 9 69 134 70 10
71 10 72 12 73 45 74 48 75 8 76 12 77 12 78 12 79 64 80 276
81 302 82 8 83 20 84 16 85 7 86 12 87 12 88 19 89 32 90 21
91 16 92 34 93 0 94 35 95 11 96 8 97 9 98 0 99 8100 1

```

it means: f.e. your script is 1000 lines long, 50% means the group of lines 501 till 510 which part of your script was running most and so you can trace exceptional cases. Sometimes Do/Until, For/Next & While/Loop can increase this value and the block structures If/Else/Endif & Select/Case/EndSelect can decrease it.

- **/Print**

it will not remove print-statements by the usage of option **"/Combine"**.

Kixstrip 3.20

- **/Progress**

it inserts code to your KiXtart script which makes it possible to see "how is the progress of your script during execution" and "what is your performance" (= **/Performance**).

The progress indicator let you see:

- how much is your script completed.
- which parts of your script were skipped.
- at which line you are leaving your script.

"How is the progress of your script during execution" doesn't mean:

- we know how long an external command will run.
- we know how long f.e. a COPY command can take.

"How is the progress of your script during execution" means:

- it is an indicator which shows the current position of the script.

To make it possible we are using RESERVED variables. They start with the prefix "**\$perf_**".

remark: *this additional code can slowdown your script.*

The default options are:

```
/Block_Check
/Debug
/Headers
/Print
/Progress
/Show_Errors
/TAB=6
/Translate
```

The options /debug, /progress & /performance are backwards compatible with all previous KiXtart releases.

What code will be inserted after selection of **"/Progress"**:

```
IF (instr("-3.6x-", "-" + substr(@kix,1,3) + "x-") = 0) AND (instr("-4.x-5.x-6.x-7.x-8.x-9.x-", "-" + substr(@kix,1,2) + "x-") = 0)
  IF MessageBox("sorry, your kixtart "+@kix+" release is too old."+CHR(13)+CHR(10)+CHR(13)+CHR(10)+CHR(10)+
    " at least run KiXtart 3.60"+CHR(13)+CHR(10)+CHR(13)+CHR(10)+
    " please upgrade .","KiXtart "+@kix+" info",4112,300)
  ENDIF
  EXIT
ENDIF
COLOR C+/N
DIM $perf_hits[101]
$perf_cur= 1 gosub "perf_info" <your code>

$perf_cur= 501 gosub "perf_info" <your code>

;($begin)
;
; sat 20-jul-2002 12:00:00 (kix 4.10 vs 3.20e)
;
;Informative KIXSTRIP: no errors found (input=1 output=1 skip=0).
;
;Informative KIXSTRIP: no block_structures found.
;Informative KIXSTRIP: no UDF's found.
;Informative KIXSTRIP: no labels found.
;Summary KIXSTRIP: BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET GETS GOSUB
; GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;
;($end)

:perf_report
IF (RedirectOutput("") = 0)
ENDIF
CLS
at(2,26) "Kixtart Performance Report "+@kix
box(3,5,5,75,"single")
$perf_column=6
DO
  at(4,$perf_column) " "
  $perf_column=$perf_column+1
UNTIL ($perf_column >= 75)
box(7,5,9,75,"single")
$perf_percentage=((100*$perf_cur)/$perf_stop)
at(6,30) substr(" ",1,3-len("$perf_percentage")) $perf_percentage " %"
```

Kixstrip 3.20

```

at(8,26) substr("      ",1,7-len("$perf_cur")) $perf_cur " .. " $perf_stop
at(8,49) "total lines: " $perf_total
$perf_stop_time=@time
$perf_start_stime=(3600*substr($perf_start_time,1,2)+60*substr($perf_start_time,4,2)+substr($perf_start_time,7,2))
$perf_stop_stime=(3600*substr($perf_stop_time,1,2)+60*substr($perf_stop_time,4,2)+substr($perf_stop_time,7,2))
$perf_elapse_time=$perf_stop_stime - $perf_start_stime
at(4,25) $perf_start_time " .. " $perf_stop_time
at(4,55) "elapse: " $perf_elapse_time " sec"
IF ($perf_elapse_time = 0)
    $perf_elapse_time=1
ENDIF
at(6,49) "lines/second: " ($perf_total/$perf_elapse_time)
at(11,0)
$perf_info=" %% hits %% hits %% hits %% hits %% hits %% hits %% hits %% hits %% hits %% hits"
$perf_info
$perf_count1=1
DO
    $perf_info=""
    $perf_count2=1
    DO
        $perf_hits_pos=$perf_count1+$perf_count2-1
        $perf_hits_value=$perf_hits[$perf_hits_pos]
        $perf_info=$perf_info+substr(" ",1,3-len("$perf_hits_pos"))+$perf_hits_pos
        $perf_info=$perf_info+substr(" ",1,5-len("$perf_hits_value"))+$perf_hits_value
        $perf_count2=$perf_count2+1
    UNTIL ($perf_count2 > 10)
    at(11+$perf_count1/10+1,0) $perf_info
    $perf_count1=$perf_count1+10
UNTIL ($perf_count1 > 100)
at(22,0)
EXIT
:perf_info ; - progress -
$perf_stop=501
IF ($perf_cur <= 1)
    CLS
    $perf_count1=0
    DO
        $perf_hits[$perf_count1]=0
        $perf_count1=$perf_count1+1
    UNTIL ($perf_count1 > 100)
    at(9,26) "Kixtart Progress Report "+@kix
    box(10,5,12,75,"single")
    box(14,5,16,75,"single")
    $perf_column=6
    $perf_start_time=@time
    $perf_total=0
ENDIF
$perf_percentage=((100*$perf_cur)/$perf_stop)
$perf_hits[$perf_percentage]=$perf_hits[$perf_percentage]+1
IF ($perf_column > 74)
    $perf_column=6
ENDIF
$perf_column=(6+((74-6)*$perf_percentage)/100)
$perf_total=$perf_total+1
at(11,$perf_column) "##"
at(13,30) substr(" ",1,3-len("$perf_percentage")) $perf_percentage " %"
at(15,26) substr(" ",1,7-len("$perf_cur")) $perf_cur " .. " $perf_stop
at(22,0)
RETURN

```

What will you see as a KiXtart progress report:

```

Kixtart Performance Report 4.10
-----
12:00:00 .. 12:00:12          elapse: 12 sec
-----
100 %                        lines/second: 407
-----
1203 .. 1203                 total lines: 4889
-----

```

```

% hits % hits % hits % hits % hits % hits % hits % hits % hits % hits
1  12 2 12 3 12 4 12 5 7 6 10 7 1 8 10 9 6 10 5
11  0 12 0 13 0 14 4 15 2 16 7 17 4 18 1 19 8 20 9
21  4 22 12 23 10 24 10 25 12 26 2 27 7 28 7 29 12 30 10
31 12 32 12 33 149 34 69 35 90 36 9 37 3 38 7 39 5 40 7
41  8 42 4 43 5 44 8 45 9 46 9 47 7 48 4 49 12 50 4
51  3 52 12 53 9 54 12 55 12 56 12 57 12 58 12 59 111 60 48
61 86 62 12 63 12 64 12 65 82 66 2010 67 556 68 9 69 134 70 10
71 10 72 12 73 45 74 48 75 8 76 12 77 12 78 12 79 64 80 276
81 302 82 8 83 20 84 16 85 7 86 12 87 12 88 19 89 32 90 21
91 16 92 34 93 0 94 35 95 11 96 8 97 9 98 0 99 8100 1

```

it means: f.e. your script is 1000 lines long, 50% means the group of lines 501 till 510 which part of your script was running most and so you can trace exceptional cases.

Sometimes Do/Until, For/Next & While/Loop can increase this value and the block structures If/Else/Endif & Select/Case/EndSelect can decrease it.

Kixstrip 3.20

screen output will be

```
Kixstrip 4.10 (vs 3.20e)      (c) MCA - scripting@wanadoo.nl - 2000, 2001, 2002

1 .E(14)

Warning KIXSTRIP: 11 line incomplete "double quotation".
Warning KIXSTRIP: 10 line incomplete "right parenthesis".
Warning KIXSTRIP: 10, 11 line incompleted.
Warning KIXSTRIP: 10 "@user" line contains unknown function.

Kixstrip 4.10 (vs 3.20e)      0.05 (sec)

input          14 input.kix
output         14 output.kix (skip: 0 blocks: 0 labels: 0)
block_check   -ERROR- block_errors=1
                block_structures=3 functions=1 labels=0
                tab=6 (default)

Warning KIXSTRIP:  1 error in block structure. missing statement(s).
                  - do:until          [0:0]
                  - for|each:in|to:step|next [0|0:0|0:0|0]
                  - function:endfunction [1:1]
-ERROR-          - if:else:endif        [2:2:1]
                  - select:case:endselect [0:0:0]
                  - while:loop         [0:0]
                  3 block_structures found.
Warning KIXSTRIP:  1 UDF found.
Warning KIXSTRIP:  2 lines are incompleted.
                  rerun program with option "/block_check /show_errors".
Warning KIXSTRIP:  1 line contains unknown function.
Warning KIXSTRIP:  some lines contains errors or possible errors.

active options:
/Block_Check /Headers /Print /Show_Errors /TAB=6 /Translate
inactive options:
/NoCombine /NoDebug /NoPerformance /NoProgress /NoShow_Structure
/NoLicense:      (not.specified)
```

Kixstrip 3.20

- **/Show Structure, /SS**

it inserts a summary report about your block structure and other relevant KiXtart elements as footer.

An example of such report:

```
;( $begin)
;
;                               sat 20-jul-2002 12:00:00      (kix 4.10 vs 3.20e)
;
;Informative KIXSTRIP:  no errors found (input=19008 output=19008 skip=0).
;
;Summary      KIXSTRIP:  block structures
;              - do:until           [118:118]
;              - for|each:in|to:step|next [0|0:0|0:0|0]
;              - function:endfunction  [0:0]
;              - if:else:endif        [3849:1137:3849]
;              - select:case:endselect [16:99:16]
;              - while:loop          [13:13]
;Informative KIXSTRIP: 3996 block_structures found.
;Informative KIXSTRIP: no UDF's found.
;Informative KIXSTRIP: 4 labels found.
;Summary      KIXSTRIP:  BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET GETS GOSUB
;              GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;Informative KIXSTRIP: 10 BREAK
;Informative KIXSTRIP: 3 GOTO
;Informative KIXSTRIP: 3 PLAY
;Informative KIXSTRIP: 11 RUN
;Informative KIXSTRIP: 173 SHELL
;Informative KIXSTRIP: 3 USE
;
;($end)
```

The screen output will be:

```
Kixstrip 4.10 (vs 3.20e)      (c) MCA - scripting@wanadoo.nl - 2000, 2001, 2002

19000 .E(19008)

Kixstrip 4.10 (vs 3.20e)      69.71 (sec)

input      19008 input.kix
output    19008 output.kix (skip: 0 blocks: 0 labels: 4)
block_check  block_structures=3996 labels=4
            tab=6 (default)

Summary KIXSTRIP:  block structures
                  - do:until           [118:118]
                  - for|each:in|to:step|next [0|0:0|0:0|0]
                  - function:endfunction  [0:0]
                  - if:else:endif        [3849:1137:3849]
                  - select:case:endselect [16:99:16]
                  - while:loop          [13:13]
            3996 block_structures found.
Informative KIXSTRIP:  no UDF's found.
Informative KIXSTRIP:  no errors found.

active options:
/Block_Check /Headers /Print /Show_Structure /TAB=6 /Translate
inactive options:
/NoCombine /NoDebug /NoPerformance /NoProgress /NoShow_Errors
/NoLicense:      (not.specified)
```

- **/Translate**

it translates keywords to upper- and lowercase. Commands and boolean operators will be uppercase. Macros and variables will be lowercase.

- **/Tab = [num]**

it will insert the specified amount of spaces by reformatting your KiXtart code. Kixstrip will replace all tab-symbols by spaces.

Kixstrip 3.20

Remark:

all possible parameters have also a deactivation version. The parameters have as prefix **"/No"**.
f.e. /NoDebug, /NoHeaders and /NoTab.

Directives

- **;\$begin**
A block starting with a line with symbol "**;\$begin**" and ending with a line with symbol "**;\$end**" will automatically removed.
- **;\$end**
- **;\$debug line**
With the directives "**;\$debug on**", "**;\$debug line**" and "**;\$debug off**" it is possible
 - to activate debugging mode. It is the default selection.
 - to translate the specified line with debugging code.
 - to deactivate debugging mode.
- **;\$debug off**
- **;\$debug on**

List of possible errors

- Success KIXSTRIP
- Informative KIXSTRIP:
- Warning KIXSTRIP:
- Error KIXSTRIP:
- Fatal KIXSTRIP:

- line incomplete "single quotation".
- line incomplete "double quotation".
- line incomplete "left square brackets".
- line incomplete "right square brackets".
- line incomplete "left parenthesis".
- line incomplete "right parenthesis".
- block "do/until" incompleted. starting point of block structure missing.
- block "for/.../next" incompleted. starting point of block structure missing.
- block "for to step/next" incompleted. starting point of block structure missing.
- block "function/endfunction" incompleted. starting point of block structure missing.
- block "if/else/endif" incompleted. starting point of block structure missing.
- block "select/case/endselect" incompleted. starting point of block structure missing.
- block "while/loop" incompleted. starting point of block structure missing.
- block incompleted.
- line incompleted.
- line contains unknown function.
- line contains too much keywords. rerun program with output as input file.

Kixstrip 3.20

List of possible screen output (example)

```
Kixstrip 4.10 (vs 3.20e)      (c) MCA - scripting@wanadoo.nl - 2000, 2001, 2002

      1 .E(14)

Warning KIXSTRIP: 11 line incomplete "double quotation".
Warning KIXSTRIP: 10 line incomplete "right parenthesis".
Warning KIXSTRIP: 10, 11 line incompleted.
Warning KIXSTRIP: 10 "@user" line contains unknown function.

Kixstrip 4.10 (vs 3.04e)                                0.05 (sec)

input          14 input.kix
output         14 output.kix (skip: 0 blocks: 0 labels: 0)
block_check   -ERROR- block_errors=1
                block_structures=3 functions=1 labels=0
                tab=6 (default)

Warning KIXSTRIP:      1 error in block structure. missing statement(s).
                    - do:until          [0:0]
                    - for|each:in|to:step|next [0|0:0|0:0|0]
                    - function:endfunction [1:1]
                    -ERROR- - if:else:endif [2:2:1]
                    - select:case:endselect [0:0:0]
                    - while:loop          [0:0]
                    3 block_structures found.
Warning KIXSTRIP:      1 UDF found.
Warning KIXSTRIP:      2 lines are incompleted.
                    rerun program with option "/block_check /show_errors".
Warning KIXSTRIP:      1 line contains unknown function.
Warning KIXSTRIP: some lines contains errors or possible errors.

active options:
 /Block_Check /Headers /Print /Show_Errors /TAB=6 /Translate
inactive options:
 /NoCombine /NoDebug /NoPerformance /NoProgress /NoShow_Structure
 /NoLicense:      (not.specified)
```

Kixstrip 3.20

List of possible summary report(s) (example)

```
;( $begin)
;
;                               sat 20-jul-2002 12:00:00      (kix 4.10 vs 3.20e)
;
;Informative KIXSTRIP:      input=14 output=14 skip=0
;
;Warning      KIXSTRIP:      1 error in block structure. missing statement(s).
;              - do:until          [0:0]
;              - for|each:in|to:step|next [0|0:0|0:0|0]
;              - function:endfunction [1:1]
;              -ERROR- - if:else:endif [2:2:1]
;              - select:case:endselect [0:0:0]
;              - while:loop          [0:0]
;Warning      KIXSTRIP:      2 lines are incompletd.
;              rerun program with option "/block_check /show_errors".
;Warning      KIXSTRIP:      some lines contains errors or possible errors.
;              1 line incomplete "double quotation".
;              1 line incomplete "right parenthesis".
;              2 line incompletd.
;              1 line contains unknown function.
;Informative KIXSTRIP:      3 block_structures found.
;Informative KIXSTRIP:      1 UDF found.
;Informative KIXSTRIP:      no labels found.
;Summary      KIXSTRIP:      BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION
;              GET GETS GOSUB GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;Informative KIXSTRIP:      1 ENDFUNCTION
;Informative KIXSTRIP:      1 EXIT
;Informative KIXSTRIP:      1 FUNCTION
;
;($end)
;( $begin)
;
;Warning KIXSTRIP: 11 line incomplete "double quotation".
;Warning KIXSTRIP: 10 line incomplete "right parenthesis".
;Warning KIXSTRIP: 10, 11 line incompletd.
;Warning KIXSTRIP: 10 "@user" line contains unknown function.
;
;($end)
```

It contains also a summary of commands and functions, which may slowdown your script.

Kixstrip 3.20

FAQ

- *Can we solve possible warnings about **incomplete lines** automatically?*
YES
 - the call "kixstrip input.kix output.kix /Print" will combine all statements to one or more lines. Splitting lines will be combined automatically.
remark: all comment will be removed.
 - with the call "kixstrip input.kix output.kix /Block_Check" it recreates your structure and you will not have any warning about incomplete lines.
- *Which elements may slowdown your script?*
 - handling files, mappings and other external activities (f.e. shell + run commands). Sometime it is waiting for a slow response of your hardware or network or you task cost on normal circumstances also much time.
 - reading information from your registry slows only a script down when you are reading very great amount of registry keys.
 - too much usage of SLEEP commands. Minimum wait time for each call is one second.
 - reading information from your file(s).
- *What is the performance of kixstrip?*
it is very fast

- input	32.888 lines	(size 1.127.878 bytes)
- output	137 lines	(size 599.822 bytes)
	135 seconds	(= 222 lines/second)
- reduction	53,181 %	

it was running on a Pentium III 1000 MHz with 512 Mbytes memory.
- *What to do when you want to reformat a 'combined script' with too much keywords on one line?*
Rerun the output again with kixstrip till the message has been gone. After reaching the limit the un-interpreted words are only copied to your output file.
- *What to do by problems with kixstrip?*
 - save your kixtart code in original form.
 - run kixstrip with necessary options.
 - save the output of your kixstrip run
f.e. kixstrip input.kix output.kix [options] > output_kixstrip.txt
 - send
 - original script
 - new script
 - kixstrip output
 - description of your problemto our address scripting@wanadoo.nl.
We prefer a ZIP file.
- *How can you get informed about new releases?*
To get informed about changes to our site. Please send us a mail. You can use our **Mail – member of site** request which can be find on all pages of our site.
- *What is very different with other debug capabilities?*
It runs without necessary user interventions. It isn't necessary to modify your script. Kixstrip defines as default debugging file %tmp%\kixdebug.txt and it redefines the RedirectOutput statements in your script.

Kixstrip 3.20

- *What are the general calls?*
 - for **checking** your KiXtart code
`kixstrip input.kix output.kix /block_checking /show_structure`
 - for **compressing** your KiXtart code
`kixstrip input.kix output.kix`
`kixstrip input.kix output.kix /print`
 - for **reformatting** your KiXtart code
`kixstrip input.kix output.kix /block_check`
`kixstrip input.kix output.kix /block_check /tab=3`
 - for **debugging** your KiXtart code
`kixstrip input.kix output.kix /debug`
 - for **performance analysis** of your KiXtart code
`kixstrip input.kix output.kix /performance`
 - for **progress analysis** of your KiXtart code
`kixstrip input.kix output.kix /progress`

Kixstrip 3.20

Examples: for checking your KiXtart code

kixstrip input.kix output.kix /block_checking

```
;
; NT/95 calculates os - Kixtart 3.62, 3.63, 4.0x, 4.10
;
IF (@inwin = 1)
    $nt_mode="yes"
ELSE
    $nt_mode="no"
ENDIF
;
$os=""
@dos=@dos
SELECT
CASE
    ($nt_mode = "yes") AND (@dos = "5.1") ; - Windows XP -
    $os="XP"
CASE
    ($nt_mode = "yes") AND (@dos = "5.0") ; - Windows 2000 -
    $os="W2K"
CASE
    ($nt_mode = "yes") ; - Windows NT -
    $os="NT4"
CASE
    ($nt_mode <> "yes") AND (@dos = "4.90") ; - Windows ME -
    $os="ME"
CASE
    ($nt_mode <> "yes") AND (@dos = "4.10") ; - Windows 98 -
    $os="W98"
CASE
    ($nt_mode <> "yes") AND (@dos = "4.0") ; - Windows 95 -
    $os="W95"
CASE
    1
    $os="???" ; - undetermined -
ENDSELECT
$os=LTRIM(RTRIM(substr($os+" ",1,3)))
;
? "$$os          "+$os

;($begin)
;
;          thu 25-jul-2002 12:00:00          (kix 4.10 vs 3.20e)
;
;Informative KIXSTRIP:  no errors found (input=31 output=31 skip=0).
;
;Informative KIXSTRIP:  2 block_structures found.
;Informative KIXSTRIP:  no UDF's found.
;Informative KIXSTRIP:  no labels found.
;Summary      KIXSTRIP:  BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET
                GETS GOSUB GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;
;($end)
```

Kixstrip 3.20

kixstrip input.kix output.kix /block_checking /show_structure

```
;
; NT/95 calculates os - Kixtart 3.62, 3.63, 4.0x, 4.10
;
IF (@inwin = 1)
    $nt_mode="yes"
ELSE
    $nt_mode="no"
ENDIF
;
$os=""
@dos=@dos
SELECT
CASE
    ($nt_mode = "yes") AND (@dos = "5.1") ; - Windows XP -
    $os="XP"
CASE
    ($nt_mode = "yes") AND (@dos = "5.0") ; - Windows 2000 -
    $os="W2K"
CASE
    ($nt_mode = "yes") ; - Windows NT -
    $os="NT4"
CASE
    ($nt_mode <> "yes") AND (@dos = "4.90") ; - Windows ME -
    $os="ME"
CASE
    ($nt_mode <> "yes") AND (@dos = "4.10") ; - Windows 98 -
    $os="W98"
CASE
    ($nt_mode <> "yes") AND (@dos = "4.0") ; - Windows 95 -
    $os="W95"
CASE
    1
    $os="???" ; - undetermined -
ENDSELECT
$os=LTRIM(RTRIM(substr($os+" ",1,3)))
;
? "$$os          "+$os

;($begin)
;
;          thu 25-jul-2002 12:00:00          (kix 4.10 vs 3.20e)
;
;Informative KIXSTRIP:  no errors found (input=31 output=31 skip=0).
;
;Summary      KIXSTRIP:  block structures
;
;              - do:until [0:0]
;              - for|each:in|to:step|next [0|0:0|0:0|0]
;              - function:endfunction [0:0]
;              - if:else:endif [1:1:1]
;              - select:case:endselect [1:7:1]
;              - while:loop [0:0]
;Informative KIXSTRIP:  2 block_structures found.
;Informative KIXSTRIP:  no UDF's found.
;Informative KIXSTRIP:  no labels found.
;Summary      KIXSTRIP:  BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET
                    GETS GOSUB GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;
;($end)
```

Kixstrip 3.20

Examples: for compressing your KiXtart code

kixstrip input.kix output.kix

```
;$begin
;
;      Kixtart 4.10 - 3.20e
;
; (c) MCA - scripting@wanadoo.nl - 2000, 2001, 2002
;
;      The software product is protected by copyright laws and
;      international copyright treaties, as well as other in-
;      tellectual property laws and treaties.
;      The SOFTWARE PRODUCT is licensed, not sold.
;
;      If you want to use the software for any (new) purpose
;      you needs a license agreement, which can only be deli-
;      vered by creator and owner of this software.
;
;($end)
if (@inwin=1) $nt_mode="yes" else $nt_mode="no" endif $os="" @dos=@dos
select case ($nt_mode="yes") AND (@dos="5.1") $os="XP" case ($nt_mode="yes") AND (@dos="5.0")
$os="W2K" case ($nt_mode="yes") $os="NT4" case ($nt_mode<>"yes") AND (@dos="4.90")
$os="ME" case ($nt_mode<>"yes") AND (@dos="4.10") $os="W98" case ($nt_mode<>"yes")
AND (@dos="4.0") $os="W95" case 1 $os="???" endselect
$os=LTRIM(RTRIM(substr($os+" ",1,3)))
;$begin
;
;      thu 25-jul-2002 12:00:00      (kix 4.10 vs 3.20e)
;
;Informative KIXSTRIP:  no errors found (input=31 output=24 skip=7).
;
;Informative KIXSTRIP:  2 block_structures found.
;Informative KIXSTRIP:  no UDF's found.
;Informative KIXSTRIP:  no labels found.
;Summary      KIXSTRIP:  BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET
GETS GOSUB GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;
;($end)
```

kixstrip input.kix output.kix /nolicense /print

```
if (@inwin=1) $nt_mode="yes" else $nt_mode="no" endif $os="" @dos=@dos
select case ($nt_mode="yes") AND (@dos="5.1") $os="XP" case ($nt_mode="yes") AND (@dos="5.0")
$os="W2K" case ($nt_mode="yes") $os="NT4" case ($nt_mode<>"yes") AND (@dos="4.90")
$os="ME" case ($nt_mode<>"yes") AND (@dos="4.10") $os="W98" case ($nt_mode<>"yes")
AND (@dos="4.0") $os="W95" case 1 $os="???" endselect
$os=LTRIM(RTRIM(substr($os+" ",1,3))) ? "$os      "+$os
;$begin
;
;      thu 25-jul-2002 12:00:00      (kix 4.10 vs 3.20e)
;
;Informative KIXSTRIP:  no errors found (input=31 output=25 skip=6).
;
;Informative KIXSTRIP:  2 block_structures found.
;Informative KIXSTRIP:  no UDF's found.
;Informative KIXSTRIP:  no labels found.
;Summary      KIXSTRIP:  BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET
GETS GOSUB GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;
;($end)
```

Kixstrip 3.20

kixstrip input.kix output.kix /noheaders /nolicense /print

```
if (@inwin=1) $nt_mode="yes" else $nt_mode="no" endif $os="" @dos=@dos
select case ($nt_mode="yes") AND (@dos="5.1") $os="XP" case ($nt_mode="yes") AND (@dos="5.0")
$os="W2K" case ($nt_mode="yes") $os="NT4" case ($nt_mode<>"yes") AND (@dos="4.90")
$os="ME" case ($nt_mode<>"yes") AND (@dos="4.10") $os="W98" case ($nt_mode<>"yes")
AND (@dos="4.0") $os="W95" case 1 $os="???" endselect
$os=LTRIM(RTRIM(substr($os+" ",1,3))) ? "$$os "+$os
```

Kixstrip 3.20

Examples: for reformatting your KiXtart code

kixstrip input.kix output.kix /block_check /show_structure /tab=3

```
;
; NT/95 create description list of KiXtart errors - Kixtart 4.00
;
; (c) scripting@wanadoo.nl - 2001
;
; vs 1.01 - program
;
FUNCTION SetError($error_number)
    EXIT $error_number
ENDFUNCTION
;
$tmp_directory=ExpandEnvironmentVars("%tmp%")
IF (substr($tmp_directory,len($tmp_directory),1) <> "\")
    $tmp_directory=$tmp_directory+"\\"
ENDIF
;
$info_file=$tmp_directory+"kix-code.txt"
IF RedirectOutput($info_file,1)
ENDIF
? "KiXtart "+@kix+" - Summary Error Codes"
?
FOR $i=0 TO 9999
    SetError($i)
    IF (len(@serror) <> 0) AND (Instr(@serror,"retrieving error information for") = 0) AND
        (Instr(@serror,", 13D") = 0)
        $result=@serror
        $pos=Instr($result,CHR(13)+CHR(10))
        WHILE ($pos <> 0)
            IF ($pos <> 0)
                $result=substr($result,1,$pos-1)+" "+substr($result,$pos+2,len($result)-$pos-1)
            ENDIF
            $pos=Instr($result,CHR(13)+CHR(10))
        LOOP
        IF (Substr($result,len($result),1) = CHR(10)) OR (Substr($result,len($result),1) = CHR(13))
            ? Substr("          ",1,7-len("@error"))+"@error "+Substr($result,1,len($result)-1)
        ELSE
            ? Substr("          ",1,7-len("@error"))+"@error "+$result
        ENDIF
    ENDIF
NEXT
IF RedirectOutput("")
ENDIF
? "Informative KIX: create an actual description list of KiXtart "+@kix+" errors."
? "          file created '"+LCASE($info_file)+"'."
EXIT

;($begin)
;
;          thu 25-jul-2002 12:00:00          (kix 4.10 vs 3.20e)
;
;Informative KIXSTRIP:  no errors found (input=46 output=46 skip=0).
;
;Summary      KIXSTRIP:  block structures
;                - do:until          [0:0]
;                - for|each:in|to:step|next [1|0:0|1:0|1]
;                - function:endfunction [1:1]
;                - if:else:endif       [6:1:6]
;                - select:case:endselect [0:0:0]
;                - while:loop          [1:1]
;Informative KIXSTRIP:  9 block_structures found.
;Informative KIXSTRIP:  1 UDF found.
;Informative KIXSTRIP:  no labels found.
;Summary      KIXSTRIP:  BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET
;                GETS GOSUB GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;Informative KIXSTRIP:  1 ENDFUNCTION
;Informative KIXSTRIP:  2 EXIT
;Informative KIXSTRIP:  1 FUNCTION
;
;($end)
```

Kixstrip 3.20

Examples: for **debugging** your KiXtart code

kixstrip input.kix output.kix /debug

(without directives)

```
;CLS
IF (instr("-3.0x-3.1x-3.2x-3.3x-", "-" + substr(@kix,1,3) + "x-") <> 0)
  IF MessageBox("sorry, your kixtart "+@kix+
    " release is too old." + CHR(13) + CHR(10) + CHR(13) + CHR(10) + " please upgrade.",
    "KiXtart "+@kix+" info", 4112, 300)
  ENDF
EXIT
ENDIF
COLOR C+/N
;AT (1,1) " "

$_debug_file="kixdebug.txt" ; - %tmp% directory -
IF (len($_debug_file) <> 0)
  IF (substr("%tmp%", len("%tmp%"), 1) = "\")
    $_debug_file="%tmp%" + $_debug_file
  ELSE
    $_debug_file="%tmp%\\" + $_debug_file
  ENDF
ENDIF
;$_debug_file="c:\kixdebug.txt"
IF RedirectOutput($_debug_file)
ENDIF

? "-"+LCASE(@day)+" "+@date+" "+@time+"- kixtart "+@kix+"/3.20e script starting"
IF (Val(substr(@kix,1,1)) >= 4)
  IF (len(@scriptname) <> 0)
    " (" + LCASE(@scriptname) + ")"
  ENDF
ENDIF
? "-"
IF ("$_debug_already_starting" <> "yes")
  ? "-curdir:      "+LCASE(@curdir)
  ? "-scriptdir:   "+LCASE(@scriptdir)
  IF (instr("-4.1x-", "-" + substr(@kix,1,3) + "x-") <> 0)
    ? "-scriptname:  "+LCASE(@scriptname)
  ENDF
  ? "-startdir:    "+LCASE(@startdir)
  ? "-"
  ? "-userid:      "+LCASE(@userid) + "/" + LCASE(@wuserid)
  ? "-user priv:   "+LCASE(@priv)
  IF (instr("-4.xx-", "-" + substr(@kix,1,2) + "xx-") <> 0)
    ? "-version:     inwin="+@inwin+"/dos="+@dos+"/productsuite="+
      @productsuite+"/producttype="+@producttype+"/csd="+LTRIM(RTRIM(@csd))
  ELSE
    ? "-version:     inwin="+@inwin+"/dos="+@dos
  ENDF
  ? "-"
  IF (instr("-4.1x-", "-" + substr(@kix,1,3) + "x-") <> 0)
    ? "-"+@cpu+" (memory "+MemorySize()+" MB)"
    ? "-"
  ENDF
ENDIF
ENDIF

$_debug_temp_name="" ; -format: yyyyymmdd_hhmmss.sss_999 scriptname-
IF (instr("-3.6x-", "-" + substr(@kix,1,3) + "x-") <> 0) OR (Val(substr(@kix,1,1)) >= 4)
  IF (instr("-4.xx-", "-" + substr(@kix,1,2) + "xx-") <> 0)
    $_debug_temp_name=@msecs
  SELECT
  CASE (len($_debug_temp_name) = 1)
    $_debug_temp_name="00" + $_debug_temp_name
  CASE (len($_debug_temp_name) = 2)
    $_debug_temp_name="0" + $_debug_temp_name
  ENDSELECT
  $_debug_temp_name="." + $_debug_temp_name
  IF Srnd(@msecs)
  ENDF
  SLEEP 0.050
ELSE
  IF Srnd((-1) * 32767 / (substr(@time, 7, 2) + 1))
  ENDF
ENDIF
```

Kixstrip 3.20

```

        SLEEP 1
    ENDIF
    $_debug_temp_name=substr(@date,1,4)+substr(@date,6,2)+substr(@date,9,2)+"_" +
    substr(@time,1,2)+substr(@time,4,2)+substr(@time,7,2)+$_debug_temp_name+"_" + Rnd()
    IF (len($_debug_temp_name) < 25)
        $_debug_temp_name=substr($_debug_temp_name+"",1,25)
    ENDIF
    IF (instr("-4.1x-", "-" + substr(@kix,1,3) + "x-") <> 0)
        IF (len(@scriptname) < 12)
            $_debug_temp_name=$_debug_temp_name+" "+substr(@scriptname+"",1,12)
        ELSE
            $_debug_temp_name=$_debug_temp_name+" "+@scriptname
        ENDIF
    ENDIF
ENDIF

IF ("$_debug_already_starting" <> "yes")
    ? "-debug file: " + $_debug_file
    ? "-debug name: " + LCASE($_debug_temp_name)
ELSE
    IF (len($_debug_temp_name) <> 0)
        ? "-debug: " + LTRIM(RTRIM(LCASE($_debug_temp_name))) + " -> " + $_debug_file
    ELSE
        ? "-debug-file: " + $_debug_file
    ENDIF
ENDIF

IF (instr("-3.6x-", "-" + substr(@kix,1,3) + "x-") = 0) AND (Val(substr(@kix,1,1)) < 4)
    GOTO _debug_starting_point
ENDIF
DIM $_debug_name ; -create local variable-
: _debug_starting_point
$_debug_name=LCASE($_debug_temp_name)
$_debug_already_starting="yes"
? "-"
? "----- start-" + @time + "-" + $_debug_name + "-" @error @serror"?

? "-
? "- 1-" + @time + "-" + $_debug_name + "-" @error @serror"? ;
? "- 2-" + @time + "-" + $_debug_name + "-" @error @serror"? ; NT/95 calculates os
? "- 3-" + @time + "-" + $_debug_name + "-" @error @serror"? ;
? "- 4-" + @time + "-" + $_debug_name + "-" @error @serror"? IF (@inwin = 1)
? "- 5-" + @time + "-" + $_debug_name + "-" @error @serror"? $nt_mode="yes"
? "- 6-" + @time + "-" + $_debug_name + "-" @error @serror"? ELSE
? "- 7-" + @time + "-" + $_debug_name + "-" @error @serror"? $nt_mode="no"
? "- 8-" + @time + "-" + $_debug_name + "-" @error @serror"? ENDIF
? "- 9-" + @time + "-" + $_debug_name + "-" @error @serror"? ;
? "- 10-" + @time + "-" + $_debug_name + "-" @error @serror"? $os=""
? "- 11-" + @time + "-" + $_debug_name + "-" @error @serror"? @dos=@dos
? "- 12-" + @time + "-" + $_debug_name + "-" @error @serror"? SELECT
? "- 13-" + @time + "-" + $_debug_name + "-" @error @serror"? CASE
? "- ($nt_mode = "yes") AND (@dos = "5.1") ; - Windows XP -
? "- $os="XP"
? "- 14-" + @time + "-" + $_debug_name + "-" @error @serror"? CASE
? "- ($nt_mode = "yes") AND (@dos = "5.0") ; - Windows 2000 -
? "- $os="W2K"
? "- 15-" + @time + "-" + $_debug_name + "-" @error @serror"? CASE
? "- ($nt_mode = "yes") ; - Windows NT -
? "- $os="NT4"
? "- 16-" + @time + "-" + $_debug_name + "-" @error @serror"? CASE
? "- ($nt_mode <> "yes") AND (@dos = "4.90") ; - Windows ME -
? "- $os="ME"
? "- 17-" + @time + "-" + $_debug_name + "-" @error @serror"? CASE
? "- ($nt_mode <> "yes") AND (@dos = "4.10") ; - Windows 98 -
? "- $os="W98"
? "- 18-" + @time + "-" + $_debug_name + "-" @error @serror"? CASE
? "- ($nt_mode <> "yes") AND (@dos = "4.0") ; - Windows 95 -
? "- $os="W95"
? "- 19-" + @time + "-" + $_debug_name + "-" @error @serror"? CASE
? "- $os=""
? "- 20-" + @time + "-" + $_debug_name + "-" @error @serror"? $os="???" ; - undetermined -
? "- 21-" + @time + "-" + $_debug_name + "-" @error @serror"? ENDSELECT
? "- 22-" + @time + "-" + $_debug_name + "-" @error @serror"? $os=LTRIM(RTRIM(substr($os+"",1,3)))
? "- 23-" + @time + "-" + $_debug_name + "-" @error @serror"? ;
? "- 24-" + @time + "-" + $_debug_name + "-" @error @serror"? ? "$$os ""+$os
? "- 25-" + @time + "-" + $_debug_name + "-" @error @serror"?

? "----- end-" + @time + "-" + $_debug_name + "-" @error @serror"

```

Kixstrip 3.20

```
? "-"
? "-" + LCASE(@day) + " " + @date + " " + @time + "-" kixstart "+@kix+"/3.20e script ending"
IF (Val(substr(@kix,1,1)) >= 4)
  IF (len(@scriptname) <> 0)
    " (" + LCASE(@scriptname) + ")"
  ENDIF
ENDIF
? "-"
IF RedirectOutput("CON")
ENDIF
COLOR C+/N
?
? "Informative KIX "+@kix+": "+" debug info see "+CHR(34)+$_debug_file+CHR(34)
IF (Val(substr(@kix,1,1)) >= 4)
  IF (len(@scriptname) <> 0)
    " (" + LCASE(@scriptname) + ")"
  ENDIF
ENDIF
IF RedirectOutput($_debug_file)
ENDIF
; ($begin)
;
;                               thu 25-jul-2002 12:00:00           (kix 4.10 vs 3.20e)
;
; Informative KIXSTRIP:   no errors found (input=31 output=31 skip=0).
;
; Informative KIXSTRIP:   2 block_structures found.
; Informative KIXSTRIP:   no UDF's found.
; Informative KIXSTRIP:   no labels found.
; Summary      KIXSTRIP:   BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET
;                               GETS GOSUB GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;
; ($end)
```

Kixstrip 3.20

The output of kix32 run is

```
-thursday 2002/07/25 21:07:41- kixtart 4.10/3.20e script starting (example.kix)
-
-curdir:      d:\
-scriptdir:   d:
-scriptname:  example.kix
-startdir:    c:\windows
-
-userid:      MCA/MCA
-user priv:   guest
-version:     inwin=2/dos=4.90/productsuite=/producttype=Windows Me/csd=
-
-Intel Pentium III (memory 511 MB)
-
-debug file:  c:\temp\kixdebug.txt
-debug name:  20020725_210741.530_1769 example.kix
-
----- start-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-      1-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-      2-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-      3-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-      4-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-      7-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-      8-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-      9-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-     10-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-     11-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
1
-     12-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-     20-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-     21-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-     28-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-     29-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-     30-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
$os
      ME
-     31-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
----- end-21:07:41-20020725_210741.530_1769 example.kix - 0 The operation completed successfully.
-
-thursday 2002/07/25 21:07:41- kixtart 4.10/3.20e script ending (example.kix)
-
```

Kixstrip 3.20

how to debug two or more calling scripts

our scripts are

script1.scr

```
? "script 1 - kix "+@kix $nul=RedirectOutput("")
call "c:\script2.kix"
sleep 1
call "c:\script2.kix"
? "script 1 - completed."
```

script2.scr

```
? "script 2 - kix "+@kix $nul=RedirectOutput("")
```

our kixstrip calls are

```
kixstrip script1.scr script1.kix /debug
kixstrip script2.scr script2.kix /debug
```

our call without any modification to those files will be

```
kix32.exe script1.kix
```

Kixstrip 3.20

for **KiXtart 3.47** run the output file **%tmp%\kixdebug.txt** will contain

```
-friday 2002/07/12 04:12:01- kixtart 3.47/3.20e script starting
-
-curdir:      c:
-scriptdir:
-startdir:   c:\
-
-userid:     MCA/MCA
-user_priv:  guest
-version:    inwin=2/dos=4.90
-
-debug file: c:\temp\kixdebug.txt
-debug name:
-
----- start-04:12:01-- 0 The operation completed successfully.
-
-          1-04:12:01-- 0 The operation completed successfully.

script 1 - kix 3.47
-          2-04:12:01-- 0 The operation completed successfully.

-friday 2002/07/12 04:12:01- kixtart 3.47/3.20e script starting
-
-debug-file: c:\temp\kixdebug.txt
-
----- start-04:12:01-- 0 The operation completed successfully.
-
-          1-04:12:01-- 0 The operation completed successfully.

script 2 - kix 3.47
-          2-04:12:01-- 0 The operation completed successfully.

----- end-04:12:01-- 0 The operation completed successfully.
-
-friday 2002/07/12 04:12:01- kixtart 3.47/3.20e script ending
-
-          3-04:12:01-- 0 The operation completed successfully.
-
-          4-04:12:02-- 0 The operation completed successfully.

-friday 2002/07/12 04:12:02- kixtart 3.47/3.20e script starting
-
-debug-file: c:\temp\kixdebug.txt
-
----- start-04:12:03-- 0 The operation completed successfully.
-
-          1-04:12:03-- 0 The operation completed successfully.

script 2 - kix 3.47
-          2-04:12:03-- 0 The operation completed successfully.

----- end-04:12:03-- 0 The operation completed successfully.
-
-friday 2002/07/12 04:12:03- kixtart 3.47/3.20e script ending
-
-          5-04:12:03-- 0 The operation completed successfully.

script 1 - completed.
----- end-04:12:03-- 0 The operation completed successfully.
-
-friday 2002/07/12 04:12:03- kixtart 3.47/3.20e script ending
-
```

Kixstrip 3.20

for **KiXtart 4.10** run the output file **%tmp%\kixdebug.txt** will contain

```
-friday 2002/07/12 04:12:30- kixtart 4.10/3.20e script starting (script1.kix)
-
-curdir:      c:
-scriptdir:   c:\
-scriptname:  script1.kix
-startdir:    c:\
-
-userid:      MCA/MCA
-user_priv:   guest
-version:     inwin=2/dos=4.90/productsuite=/producttype=Windows Me/csd=
-
-Intel Pentium III (memory 511 MB)
-
-debug file:  c:\temp\kixdebug.txt
-debug name:  20020725_041230.150_528  script1.kix
-
----- start-04:12:30-20020725_041230.150_528  script1.kix - 0 The operation completed successfully.
-
-      1-04:12:30-20020725_041230.150_528  script1.kix - 0 The operation completed successfully.
script 1 - kix 4.10
-      2-04:12:30-20020725_041230.150_528  script1.kix - 0 The operation completed successfully.
-
-friday 2002/07/12 04:12:30- kixtart 4.10/3.20e script starting (script2.kix)
-
-debug: 20020725_041230.260_887  script2.kix -> c:\temp\kixdebug.txt
-
----- start-04:12:30-20020725_041230.260_887  script2.kix - 0 The operation completed successfully.
-
-      1-04:12:30-20020725_041230.260_887  script2.kix - 0 The operation completed successfully.
script 2 - kix 4.10
-      2-04:12:30-20020725_041230.260_887  script2.kix - 0 The operation completed successfully.
----- end-04:12:30-20020725_041230.260_887  script2.kix - 0 The operation completed successfully.
-
-friday 2002/07/12 04:12:30- kixtart 4.10/3.20e script ending (script2.kix)
-
-      3-04:12:30-20020725_041230.150_528  script1.kix - 0 The operation completed successfully.
-
-      4-04:12:31-20020725_041230.150_528  script1.kix - 0 The operation completed successfully.
-
-friday 2002/07/12 04:12:31- kixtart 4.10/3.20e script starting (script2.kix)
-
-debug: 20020725_041231.300_1018  script2.kix -> c:\temp\kixdebug.txt
-
----- start-04:12:31-20020725_041231.300_1018  script2.kix - 0 The operation completed successfully.
-
-      1-04:12:31-20020725_041231.300_1018  script2.kix - 0 The operation completed successfully.
script 2 - kix 4.10
-      2-04:12:31-20020725_041231.300_1018  script2.kix - 0 The operation completed successfully.
----- end-04:12:31-20020725_041231.300_1018  script2.kix - 0 The operation completed successfully.
-
-friday 2002/07/12 04:12:31- kixtart 4.10/3.20e script ending (script2.kix)
-
-      5-04:12:31-20020725_041230.150_528  script1.kix - 0 The operation completed successfully.
script 1 - completed.
----- end-04:12:31-20020725_041230.150_528  script1.kix - 0 The operation completed successfully.
-
-friday 2002/07/12 04:12:31- kixtart 4.10/3.20e script ending (script1.kix)
-
```

Kixstrip 3.20

Examples: for performance analysis of your KiXtart code

kixstrip input.kix output.kix /performance

```
IF (instr("-3.6x-", "-" + substr(@kix,1,3) + "x-") = 0) AND
  (instr("-4.x-5.x-6.x-7.x-8.x-9.x-", "-" + substr(@kix,1,2) + "x-") = 0)
  IF MessageBox("sorry, your kixtart "+@kix+" release is too old." + CHR(13) + CHR(10) + CHR(13) + CHR(10) +
    " at least run KiXtart 3.60" + CHR(13) + CHR(10) + CHR(13) + CHR(10) +
    " please upgrade .", "KiXtart "+@kix+" info", 4112, 300)
  ENDIF
  EXIT
ENDIF
COLOR C+/N
DIM $perf_hits[101]
$perf_cur= 1 gosub "perf_info" ;
$perf_cur= 2 gosub "perf_info" ; NT/95 calculates os - Kixtart 3.62, 3.63, 4.0x, 4.10
$perf_cur= 3 gosub "perf_info" ;
$perf_cur= 4 gosub "perf_info" IF (@inwin = 1)
$perf_cur= 5 gosub "perf_info" $nt_mode="yes"
$perf_cur= 6 gosub "perf_info" ELSE
$perf_cur= 7 gosub "perf_info" $nt_mode="no"
$perf_cur= 8 gosub "perf_info" ENDIF
$perf_cur= 9 gosub "perf_info" ;
$perf_cur= 10 gosub "perf_info" $os=""
$perf_cur= 11 gosub "perf_info" @dos=@dos
$perf_cur= 12 gosub "perf_info" SELECT
$perf_cur= 13 gosub "perf_info" CASE
    ($nt_mode = "yes") AND (@dos = "5.1") ; - Windows XP -
$perf_cur= 14 gosub "perf_info" $os="XP"
$perf_cur= 15 gosub "perf_info" CASE
    ($nt_mode = "yes") AND (@dos = "5.0") ; - Windows 2000 -
$perf_cur= 16 gosub "perf_info" $os="W2K"
$perf_cur= 17 gosub "perf_info" CASE
    ($nt_mode = "yes") ; - Windows NT -
$perf_cur= 18 gosub "perf_info" $os="NT4"
$perf_cur= 19 gosub "perf_info" CASE
    ($nt_mode <> "yes") AND (@dos = "4.90") ; - Windows ME -
$perf_cur= 20 gosub "perf_info" $os="ME"
$perf_cur= 21 gosub "perf_info" CASE
    ($nt_mode <> "yes") AND (@dos = "4.10") ; - Windows 98 -
$perf_cur= 22 gosub "perf_info" $os="W98"
$perf_cur= 23 gosub "perf_info" CASE
    ($nt_mode <> "yes") AND (@dos = "4.0") ; - Windows 95 -
$perf_cur= 24 gosub "perf_info" $os="W95"
$perf_cur= 25 gosub "perf_info" CASE
    1
$perf_cur= 26 gosub "perf_info" $os="???" ; - undetermined -
$perf_cur= 27 gosub "perf_info" ENDSELECT
$perf_cur= 28 gosub "perf_info" $os=LTRIM(RTRIM(substr($os+" ",1,3)))
$perf_cur= 29 gosub "perf_info" ;
$perf_cur= 30 gosub "perf_info" ? "$$os "$os
$perf_cur= 31 gosub "perf_info"

; ($begin)
;
;           thu 25-jul-2002 12:00:00           (kix 4.10 vs 3.20e)
;
; Informative KIXSTRIP: no errors found (input=31 output=31 skip=0).
;
; Informative KIXSTRIP: 2 block_structures found.
; Informative KIXSTRIP: no UDF's found.
; Informative KIXSTRIP: no labels found.
; Summary KIXSTRIP: BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET
;           GETS GOSUB GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;
; ($end)
```

Kixstrip 3.20

```
:perf_report
IF (RedirectOutput("") = 0)
ENDIF
CLS
at(2,26) "Kixtart Performance Report "+@kix
box(3,5,5,75,"single")
$perf_column=6
DO
  at(4,$perf_column) " "
  $perf_column=$perf_column+1
UNTIL ($perf_column >= 75)
box(7,5,9,75,"single")
$perf_percentage=((100*$perf_cur)/$perf_stop)
at(6,30) substr(" ",1,3-len("$perf_percentage")) $perf_percentage " %"
at(8,26) substr(" ",1,7-len("$perf_cur")) $perf_cur " .. " $perf_stop
at(8,49) "total lines: " $perf_total
$perf_stop_time=@time
$perf_start_stime=(3600*substr($perf_start_time,1,2)+60*substr($perf_start_time,4,2)+
  substr($perf_start_time,7,2))
$perf_stop_stime=(3600*substr($perf_stop_time,1,2)+60*substr($perf_stop_time,4,2)+
  substr($perf_stop_time,7,2))
$perf_elapse_time=$perf_stop_stime - $perf_start_stime
at(4,25) $perf_start_time " .. " $perf_stop_time
at(4,55) "elapse: " $perf_elapse_time " sec"
IF ($perf_elapse_time = 0)
  $perf_elapse_time=1
ENDIF
at(6,49) "lines/second: " ($perf_total/$perf_elapse_time)
at(11,0)
$perf_info=" %% hits %% hits %% hits %% hits %% hits %% hits %% hits %% hits %% hits %%
hits"
$perf_info
$perf_count1=1
DO
  $perf_info=""
  $perf_count2=1
  DO
    $perf_hits_pos=$perf_count1+$perf_count2-1
    $perf_hits_value=$perf_hits[$perf_hits_pos]
    $perf_info=$perf_info+substr(" ",1,3-len("$perf_hits_pos"))+$perf_hits_pos
    $perf_info=$perf_info+substr(" ",1,5-len("$perf_hits_value"))+$perf_hits_value
    $perf_count2=$perf_count2+1
  UNTIL ($perf_count2 > 10)
  at(11+$perf_count1/10+1,0) $perf_info
  $perf_count1=$perf_count1+10
UNTIL ($perf_count1 > 100)
at(22,0)
EXIT
:perf_info ; - performance -
$perf_stop=31
IF ($perf_cur <= 1)
  $perf_count1=0
  DO
    $perf_hits[$perf_count1]=0
    $perf_count1=$perf_count1+1
  UNTIL ($perf_count1 > 100)
  $perf_start_time=@time
  $perf_total=0
ENDIF
$perf_percentage=((100*$perf_cur)/$perf_stop)
$perf_hits[$perf_percentage]=$perf_hits[$perf_percentage]+1
$perf_total=$perf_total+1
RETURN
```

Kixstrip 3.20

Examples: for **progress analysis** of your KiXtart code

kixstrip input.kix output.kix /progress

```
IF (instr("-3.6x-", "-" + substr(@kix, 1, 3) + "x-") = 0) AND
  (instr("-4.x-5.x-6.x-7.x-8.x-9.x-", "-" + substr(@kix, 1, 2) + "x-") = 0)
  IF MessageBox("sorry, your kixtart "+@kix+" release is too old." + CHR(13) + CHR(10) + CHR(13) + CHR(10) +
    " at least run KiXtart 3.60" + CHR(13) + CHR(10) + CHR(13) + CHR(10) +
    " please upgrade .", "KiXtart "+@kix+" info", 4112, 300)
  ENDIF
  EXIT
ENDIF
COLOR C+/N
DIM $perf_hits[101]
$perf_cur= 1 gosub "perf_info" ;
$perf_cur= 2 gosub "perf_info" ; NT/95 calculates os - Kixtart 3.62, 3.63, 4.0x, 4.10
$perf_cur= 3 gosub "perf_info" ;
$perf_cur= 4 gosub "perf_info" IF (@inwin = 1)
$perf_cur= 5 gosub "perf_info" $nt_mode="yes"
$perf_cur= 6 gosub "perf_info" ELSE
$perf_cur= 7 gosub "perf_info" $nt_mode="no"
$perf_cur= 8 gosub "perf_info" ENDIF
$perf_cur= 9 gosub "perf_info" ;
$perf_cur= 10 gosub "perf_info" $os=""
$perf_cur= 11 gosub "perf_info" @dos=@dos
$perf_cur= 12 gosub "perf_info" SELECT
$perf_cur= 13 gosub "perf_info" CASE
    ($nt_mode = "yes") AND (@dos = "5.1") ; - Windows XP -
    $os="XP"
$perf_cur= 14 gosub "perf_info" CASE
    ($nt_mode = "yes") AND (@dos = "5.0") ; - Windows 2000 -
    $os="W2K"
$perf_cur= 15 gosub "perf_info" CASE
    ($nt_mode = "yes") ; - Windows NT -
    $os="NT4"
$perf_cur= 16 gosub "perf_info" CASE
    ($nt_mode <> "yes") AND (@dos = "4.90") ; - Windows ME -
    $os="ME"
$perf_cur= 17 gosub "perf_info" CASE
    ($nt_mode <> "yes") AND (@dos = "4.10") ; - Windows 98 -
    $os="W98"
$perf_cur= 18 gosub "perf_info" CASE
    ($nt_mode <> "yes") AND (@dos = "4.0") ; - Windows 95 -
    $os="W95"
$perf_cur= 19 gosub "perf_info" CASE
    1
    $os="???" ; - undetermined -
$perf_cur= 20 gosub "perf_info" ENDSELECT
$perf_cur= 21 gosub "perf_info" $os=LTRIM(RTRIM(substr($os+" ", 1, 3)))
$perf_cur= 22 gosub "perf_info" ;
$perf_cur= 23 gosub "perf_info" ? "$$os "$os
$perf_cur= 24 gosub "perf_info"

; ($begin)
;
;          thu 25-jul-2002 12:00:00          (kix 4.10 vs 3.20e)
;
; Informative KIXSTRIP: no errors found (input=31 output=31 skip=0).
;
; Informative KIXSTRIP: 2 block_structures found.
; Informative KIXSTRIP: no UDF's found.
; Informative KIXSTRIP: no labels found.
; Summary KIXSTRIP: BREAK CALL DEBUG DISPLAY ENDFUNCTION EXECUTE EXIT FUNCTION GET
; GETS GOSUB GOTO OLExxx PLAY QUIT RETURN RUN SHELL SLEEP THEN USE
;
; ($end)
```

Kixstrip 3.20

```
:perf_report
IF (RedirectOutput("") = 0)
ENDIF
CLS
at(2,26) "Kixtart Performance Report "+@kix
box(3,5,5,75,"single")
$perf_column=6
DO
  at(4,$perf_column) " "
  $perf_column=$perf_column+1
UNTIL ($perf_column >= 75)
box(7,5,9,75,"single")
$perf_percentage=((100*$perf_cur)/$perf_stop)
at(6,30) substr(" ",1,3-len("$perf_percentage")) $perf_percentage " %"
at(8,26) substr(" ",1,7-len("$perf_cur")) $perf_cur " .. " $perf_stop
at(8,49) "total lines: " $perf_total
$perf_stop_time=@time
$perf_start_stime=(3600*substr($perf_start_time,1,2)+60*substr($perf_start_time,4,2)+
  substr($perf_start_time,7,2))
$perf_stop_stime=(3600*substr($perf_stop_time,1,2)+60*substr($perf_stop_time,4,2)+
  substr($perf_stop_time,7,2))
$perf_elapse_time=$perf_stop_stime - $perf_start_stime
at(4,25) $perf_start_time " .. " $perf_stop_time
at(4,55) "elapse: " $perf_elapse_time " sec"
IF ($perf_elapse_time = 0)
  $perf_elapse_time=1
ENDIF
at(6,49) "lines/second: " ($perf_total/$perf_elapse_time)
at(11,0)
$perf_info=" %% hits %% hits %% hits %% hits %% hits %% hits %% hits %% hits %% hits %%"
hits"
$perf_info
$perf_count1=1
DO
  $perf_info=""
  $perf_count2=1
  DO
    $perf_hits_pos=$perf_count1+$perf_count2-1
    $perf_hits_value=$perf_hits[$perf_hits_pos]
    $perf_info=$perf_info+substr(" ",1,3-len("$perf_hits_pos"))+$perf_hits_pos
    $perf_info=$perf_info+substr(" ",1,5-len("$perf_hits_value"))+$perf_hits_value
    $perf_count2=$perf_count2+1
  UNTIL ($perf_count2 > 10)
  at(11+$perf_count1/10+1,0) $perf_info
  $perf_count1=$perf_count1+10
UNTIL ($perf_count1 > 100)
at(22,0)
EXIT
:perf_info ; - progress -
$perf_stop=31
IF ($perf_cur <= 1)
  CLS
  $perf_count1=0
  DO
    $perf_hits[$perf_count1]=0
    $perf_count1=$perf_count1+1
  UNTIL ($perf_count1 > 100)
  at(9,26) "Kixtart Progress Report "+@kix
  box(10,5,12,75,"single")
  box(14,5,16,75,"single")
  $perf_column=6
  $perf_start_time=@time
  $perf_total=0
ENDIF
$perf_percentage=((100*$perf_cur)/$perf_stop)
$perf_hits[$perf_percentage]=$perf_hits[$perf_percentage]+1
IF ($perf_column > 74)
  $perf_column=6
ENDIF
$perf_column=(6+((74-6)*$perf_percentage)/100)
$perf_total=$perf_total+1
at(11,$perf_column) "##"
at(13,30) substr(" ",1,3-len("$perf_percentage")) $perf_percentage " %"
at(15,26) substr(" ",1,7-len("$perf_cur")) $perf_cur " .. " $perf_stop
at(22,0)
RETURN
```

Kixstrip 3.20

Published versions

- 1.35 30.07.2000
- 1.44 5.08.2000
- 1.45 10.08.2000
- 1.70 15.08.2000
- 1.71 5.09.2000
- 1.73 10.09.2000
- 1.74 10.09.2000
- 1.75 10.09.2000
- 1.76 25.09.2000
- 1.77 25.10.2000
- 2.01 25.03.2001
- 2.02 10.04.2001
- 2.12 10.04.2001
- 2.15 15.04.2001
- 2.17 1.05.2001
- 2.23 5.05.2001
- 2.25 15.06.2001
- 2.26 10.07.2001
- 2.27 10.07.2001
- 2.28 20.07.2001
- 2.29 20.07.2001
- 2.33 15.11.2001
- 3.03 10.04.2002
- 3.04 10.04.2002
- 3.13 05.07.2002
- 3.18 10.07.2002
- 3.19 20.07.2002
- 3.20 30.07.2002

Kixstrip 3.20

Release notes

- **1.35** release version
- **1.44** add
 - options `"/Show_Errors"` + `"/Debug"`.
 - strip also unused spaces.
 - show which options are active.
 - information about possible errors.change
 - warnings will become a print-statement instead of comment-statement when we are using the option `"/Combine"`.
- **1.45** change
 - minor.
- **1.70** add
 - options `"/Headers"` + `"/Translate"`.
 - replace all TAB characters with spaces.
 - strip also unused "TAB" characters.
 - show information about block structures and counters for special KiXtart keywords.
 - show user information after reading and checking your script completely.
 - show a progress bar during processing used files + elapse time after completion.
 - check for illegal/unknown macro names.
 - remove header and footer after a rerun of the kixstrip program. Special elements are `;"($begin)"` and `;"($end)"`.
 - check for KiXtart block structure keywords.change
 - enhancement in error handling.
 - same error messages are only printed once.
 - selection of option `"/Headers"` will automatically insert a summary report.add & remove
 - option `"/Comment"`.
- **1.71** add
 - option `"/Progress"`.
- **1.73** add
 - option `"/Performance"`.
- **1.74** add
 - information about options `"/Performance"` + `"/Progress"`.
- **1.75** change
- **1.76** add
 - show "active" + "inactive" options to your user-screen.
- **1.77** update
 - KiXtart 3.63
- **2.01** fix
 - range check error when a keyword starts in first position.add
 - show also KiXtart + program version in summary report.
- **2.02** fix
 - range check error by printing error messages. Redesign this part.info
 - last compatible version with KiXtart 3.63 (remark: by version 2.25 we are using same code with different KiXtart keyword tables).

Kixstrip 3.20

- **2.12** update
 - KiXtart 4.00 beta 1
- add
 - for/next + function/endfunction block structures.
 - option `"/Show_Structure"`.
 - show special counters information for:
 - then (= unwanted/illegal keyword)
 - OLExxx (which are no longer supported since KiXtart 4.00 release)
- change
 - enhancement in error + counter information.
- **2.15** add
 - show information about lines read + printed in summary report.
- change
 - enhancement in block structures analyse.
- **2.17** update
 - KiXtart 4.00 beta 2
- **2.23** add
 - @time extension to option `"/Debug"`.
- change
 - extend number of keywords in one line to 2048 elements.
- fix
 - handle parameters of USE and COPY KiXtart commands in a correct way.
 - a "divide by zero" error by the options `"/Performance"` + `"/Progress"`, when script runs within zero seconds.
- **2.25** change
 - KiXtart 3.63 + 4.00 are using same code. Only the KiXtart keyword tables are different.
- **2.26** update
 - KiXtart 4.00 RC1 (build 41)
- **2.27** add
 - @error + @serror extension to option `"/Debug"`.
- remove
 - debug header code from options `"/Performance"` + `"/Progress"`.
- **2.28** update
 - KiXtart 4.00 RC2 (build 45)
- change
 - user information "function found" to "UDF found".
- **2.29** fix
 - for KiXtart problem with * and / options. Problem influence the options `"/Performance"` + `"/Progress"`.
- **2.33** update
 - KiXtart 4.00 (build 62) + 4.01 (build 64)
- add
 - improvement of handling special keywords (and, file, list, not, off, on, optional, or & preserve)
 - shows kixstrip version also in copyright note
- change
 - standardize of generated `"/Performance"` and `"/Progress"` code

Kixstrip 3.20

- **3.03** update
 - KiXtart 4.01 (build 64)
 - KiXtart 4.02 (build 71)
 - KiXtart 4.10 beta 1 (build 84)
- Change
 - debug header has been extended with "@startdir", "@scriptdir", "@curdir", "@inwin", "@dos", "@userid/@wuserid", "@productsuite", "@producttype", "@cpu", "mhz" and "MemorySize()".
 - debug code is also using start and last line part.
- fix
 - replace all TABs by SPACES
- **3.04** fix
 - new "/Debug" code wasn't compliant with KiXtart 3.6x release.
- **3.13** add
 - extensions made to option "/Debug". It shows now
 - which lines were executed,
 - what was the time of execution,
 - what was the KiXtart error status by execution of each line,
 - actual environment information about os, user, etc.,
 - unique name based on @date, @msecs, @scriptname, @time and random number,
 - when was debugging starting and ending.
 - other extensions made to option "/Debug". Now
 - it creates automatically the debug information file "%tmp%\kixdebug.txt".
 - it isn't necessary to change your "RedirectOutput" statements. Automatically it will be reset to specified debug information file.
 - with kixstrip directives ";\$debug off)" and ";\$debug on)" you can active and deactivate debugging code.
 - with kixstrip directive ";\$debug line)" you insert debugging code only once.
 - Kixstrip without options will show "help" instead of "date" information
- **3.18** add
 - **down/up-wards compatibility** of options "/Debug", "/Performance" and "/Progress" for all KiXtart releases.
 - extensions made to option "/Debug". Now
 - it shows only once complete information block,
 - it shows only unique name when useful. The required local variable declaration exists from KiXtart 3.60 release,
 - the statements AT and @mhz are removed because of unexpected side effects,
 - the statement CLS is removed.
- fix
 - remove unwanted change of your "RedirectOutput" statements, when "/Debug" isn't specified
- update
 - KiXtart 4.10 final release (build 99)
- **3.19** fix
 - minor change for "/Performance" and "/Progress"
- **3.20** fix
 - minor change for "/Debug"

Kixstrip 3.20

To-do activities

- **add:** `/Monitor` will use your registry as another way to monitor progress of your script. As user you have the capability to extend monitor-part in an easy way. Extend with f.e. the variables to monitor, the actual error status, last executed line.
- **add:** `/Translate` will also convert KiXtart functions. with `"kixstrip show /translate!"` it will show the user the actual translation table.
- create a batch file to handle long filenames
f.e. first copy long filename to a short temp file / handle kixstrip command with temp files / copy output of kixstrip command back to a (new) long filename
- parameter-counting of KiXtart functions.
- handle split statements over two or more lines in a correct way.
- possibility to handle internal macros and variables in a string.
- reset (better: correct) `@error` status each time a line will be executed by `/Debug` variant. important: sometimes these value will be used by original code. a reset will influence the script in another way.
- **add** colour indicator by option `/Progress`.
- **add** translate variables to mnemonics (f.e. `$<nr>`).
(possible is that kixref an alternative is to handle this)
- **add** `;(include filename)`
will replace this line by the contents of specified file. Result will be
 `;(include filename)`
 `;(begin)`
 [compressed kixtart code]
 `;(end)`
(what when this include statement between `;(begin)` + `;(end)` structures)

Kixstrip 3.20

Alternative

- We can check very fast some piece of code without using kixstrip tool. It isn't same or more intelligent as the tool, but it can give you a nice result.

```
goto check_my_script
;
; your script
;
:check_my_script ? "end of script found"
```

Development

- **kixstrip**
 - **/block_check**
 - **/debug**
 - **/show_errors**
- **kixref**
 - **/translate**
 - **show /translate**
 - **/warnings**

Distribution

- **kixstrip** (/NoLicense /Print)
- **compress**
- **codec**
- **kixcrypt / wkixcrypt**
- **bat2exec**
- **secure21.zip**

related <http://kixtart.org> topics