[9.12] Use one *Request* (in a Dialog box) for more than one variable

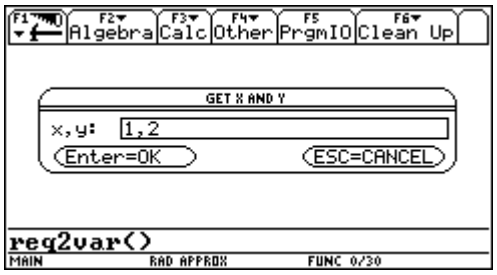
The *Request* command is used in Dialog boxes to get input from the user. However, there is a limit to the number of *Request* commands allowed in a Dialog box. In addition, data is naturally entered in pairs or triplets, such as point coordinates. By using one *Request* command to get two variables from the user, you reduce the number of *Request* commands, and the user can enter the data in a more natural way.

The two expressions entered by the user are separated by a comma. This program demonstrates the technique:

```
req2var()
Prgm
©Demo, get two variables in one Request
©14oct00 dburkett@infinet.com

local s,x,y,xy

©Set default values for x and y
1→x
2→y

©Combine x and y for default prompt
string(x)&","&string(y)→xy

©End up here if user omits comma between x and y
lbl tryagain

©Prompt for x and y
dialog
 title "GET X AND Y"
 request "x,y",xy               ©x and y will be in string xy
enddlog
if ok=0:return                  ©Let user quit by pressing [ESC]

©Find the comma position in xy
instring(xy,",")→s

©Test for comma
if s=0 then
 dialog                         ©Display error message if no comma found
  title "ERROR"
  text "You must put a comma"
  text "between x and y"
 enddlog
 if ok=0:return                 ©Let user quit by pressing [ESC],
goto tryagain                   ©or try again by pressing [ENTER]
endif

©Convert x and y strings to expressions
expr(left(xy,s-1))→x
expr(right(xy,dim(xy)-s))→y

©Just for a demo, display x and y. Your application probably won't do this
dialog
 title "SHOW X AND Y"
 text "x is "&string(x)
 text "y is "&string(y)
enddlog

EndPrgm
```

Note that the copy of this program in the tlcode.zip file does not include all the comments.

When this program runs, this Dialog box is shown:



Note that the defaults of 1 and 2 are shown. The user presses [ENTER] to accept these defaults, or enters the new values, separated by a comma. Or, the user can press [ESC] to quit the program.

If the user forgets to enter the comma separating the two values, an error message dialog box is shown. The user can press [ESC] to quit the program at this point, or press [ENTER] to try again.

This idea can be extended to more than two variables. The program *req3var()* shows how to get three variables from one *Request* command.

```
req3var()
Prgm
©Demo, get three variables in one Request
©14oct00 dburkett@infinet.com

local s1,s2,x,y,z,xyz

©Set default values for x, y and z
1→x
2→y
3→z

©Combine x, y and z for default prompt
string(x)&","&string(y)&","&string(z)→xyz

©End up here if user omits commas between x, y and z
lbl tryagain

©Prompt for x,y and z
dialog
 title "GET X, Y AND Z"
 request "x,y,z",xyz                    ©x, y and z will be in string xyz
enddlog
if ok=0:return                          ©Let user quit by pressing [ESC]

©Find the comma positions in xyz
instring(xyz,",")→s1                    ©s1 is the position of the first comma
instring(xyz,",",s1+1)→s2               ©s2 is the position of the second comma

©Test for commas
if s1=0 or s2=0 then
 dialog                                 ©Display error message if both commas
  title "ERROR"                         ©not found
  text "You must put commas"
```

```
  text "between x, y and z"
 enddlog
 if ok=0:return                          ©Let user quit by pressing [ESC],
 goto tryagain                           ©or try again by pressing [ENTER]
endif

©Convert x,y and z strings to expressions
expr(left(xyz,s1-1))→x
expr(mid(xyz,s1+1,s2-s1-1))→y
expr(right(xyz,dim(xyz)-s2))→z

©Just for a demo, display x, y and z. Your application probably won't do this
dialog
 title "SHOW X, Y AND Z"
 text "x is "&string(x)
 text "y is "&string(y)
 text "z is "&string(z)
enddlog

EndPrgm
```

If you need to use this method more than once in a dialog box, you can reduce the total code size by with a subroutine which converts a string of parameters to a list. These two functions, *rq2v()* and *rq3v()*, perform the conversion. Use *rq2v()* for two parameters:

```
rq2v(st)
Func
©Convert ("a,b") to {a,b}
©31mar01/dburkett@infinet.com

local s

instring(st,",")→s

if s=0 then
 return "ERR"
else
 return {expr(left(st,s-1)),expr(right(st,dim(st)-s))}
endif

EndFunc
```

Use *rq3v()* for three parameters:

```
rq3v(st)
Func
©Convert ("a,b,c") to {a,b,c}
©31mar01/dburkett@infinet.com

local s1,s2

instring(st,",")→s1
instring(st,",",s1+1)→s2

if s1=0 or s2=0 then
 return "ERR"
else
 return
{expr(left(st,s1-1)),expr(mid(st,s1+1,s2-s1-1)),expr(right(st,dim(st)-s2))}
endif

EndFunc
```

Both of these functions return *"ERR"* if the commas are missing. This example shows how to use both functions.

```
©Initialize prompt variables
"0,0"→ab
"0,0,0"→cde

©Prompt for input variables
dialog
 request "a,b",ab
 request "c,d,e",cde
enddlog

©Extract a, b variables
util\rq2v(ab)→res
if res="ERR" then
 (... handle error here ...)
else
 res[1]→a
 res[2]→b
endif

©Extract c, d, e variables
util\rq3v(cde)→res
if res="ERR" then
  (... handle error here ...)
else
 res[1]→c
 res[2]→d
 res[3]→e
endif
```

ab and *cde* are strings that are returned from the *Request* commands in the dialog box. These are passed directly to *rq2v()* and *rq3v()*. The returned lists are saved in variable *res*. For example, if the users enters

1,2

at the *ab* prompt, and

3,4,5

at the *cde* prompt, then *ab* = "1,2" and *cde* = "3,4,5". The *res* result variable will be, respectively, {1,2} and {3,4,5}. During the variable extraction sections, the individual elements of *res* are stored to the appropriate variable. Note that both extraction sections test for the error condition. The actual error handling depends on your program. Typically, you would display an error message and give the user another chance to correctly enter the variables.

You can use this method to get numbers and expressions from the user, but not multiple lists or matrices, because the programs do not distinguish between the commas that separate the expressions and the commas within the expressions.