

100 PML Tips & Tricks

Help in programming on PML

AVEVA Plant and AVEVA Marine Solutions

001. Shortening time of Splashscreen displaying and using Pause command	3
002. Opening and automatic updating Association Manager form on system start	3
003. Processing all sessions at a selected date	3
004. Attributes for querying information on session comparing	3
005. Sample macro for name part replacing dialog	4
006. Writing attributes values for selected elements into array	4
007. Writing mouse-selected elements into array	4
008. Include paths to DLL	4
009. Setting digit number after the point	4
010. Example of use autozoom - 'Walk to Drawlist'	5
011. Setting focus on a gadget on opening a form	5
012. Writing all Drawlist elements into array	5
013. Displaying open file dialog	5
014. Opening save file dialog	5
015. Centring on CE	6
016. Calling command/macro on Design module start	6
017. Reading data from Excel using PML.NET functions	6
018. Displaying progress bar, showing process progress	6
019. How to suppress message about exit from pick mode after pressing Esc key	7
020. Turning on Model Editor using PML	7
021. Opening Members list form (old version of Hierarchy Explorer)	7
022. Finding out element creator (who created the first session with the element)	7
023. Collecting elements, which EXCLUDEVLY lie in given volume	8
024. Loading applications with command line	8
025. How to know which forms are opened	8
026. Excel loading function	8
027. How to query which sheets are opened in Draft	9
028. Closing all sheets in Draft	9
029. Getting coordinates of mouse click in sheet field	9
030. Finding out how long operation was executed	9
031. Difference between ALPHA LOG and ALPHA FILE	9
032. Calling command which is in DLL	9
033. Encoding connection type string into number	9
034. Using IFTRUE operator	9
035. Difference in element collection methods	10
036. Querying information about deleted elements	10
037. How to get coordinates of mouse click on element (not Origin point)	10
038. How to find out whether ISODRAFTMODE is on	10
039. How to move annotation element of drawing into explicit coordinates	11
040. How to find out coordinates of a centre and end-points of a tube	11
041. How to query position of object relative to drawing view	11
042. How to change period to comma using intelligent text	11
043. How to send text file to printer	11
044. How to find out computer and Windows user name	11
045. How to find out 3D coordinates by clicking on drawing	11
046. How to find out 2D coordinates by clicking on drawing	11
047. Commands for creating primitive elements on drawing and their attributes	11
048. How automatically calculate view scale	13
049. How to move content of view to centre after change size of view	14
050. How automatically to set size of view by 3D coordinates	14
051. How to collect elements of different owners	14
052. How to put combobox with images on form	14
053. How to add synonyms to commands	14
054. How to find out hierarchy depth	15
055. How to collect elements with given value of the first member attribute	16
056. How to find enhanced labels and run position alignment	16
057. Setting view direction for 3D view	16
058. Finding out screen resolution	16
059. How to calculate distance between two points (strait segment length)	16
060. How to find middle of a segment between two coordinates	17
061. What options can be used on starting of the system	17
062. How to display system elements of hierarchy	18
063. How to set default properties programmatically	18

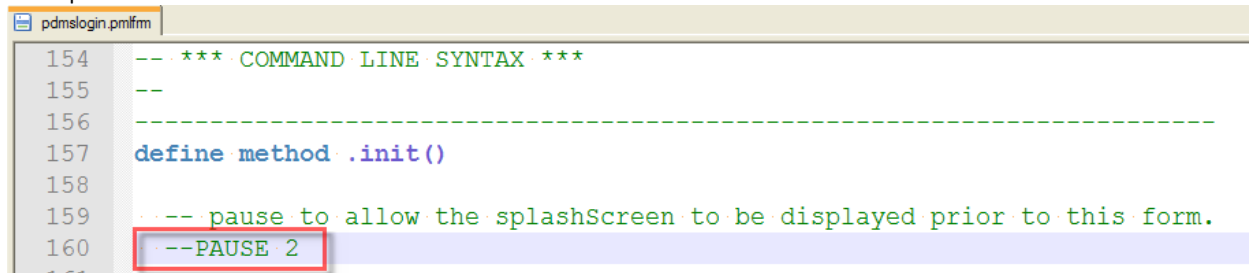
064. How to open PDF file on a given page	18
065. How to programmatically update Limits-Defined view of drawing after changing a Drawlist	18
066. How to open Folder Open Dialog.....	19
067. How to draw aid label in the head and in the tail of a branch	19
068. How to display colour numbers and corresponding names.....	19
070. How to automatically execute commands on changing current element.....	19
071. How to show information on all attributes of the system	20
072. How to query full description of attribute	20
073. How to get array of all available attributes in the system.....	21
074. How to select rectangle area on drawing	21
075. How to select labels on drawing using area selecting.....	21
076. How to calculate total length of tubes.....	21
077. How to query the title of a current form of a module or an application.....	22
078. How to call macro which sets defaults for layers.....	22
079. How to get list of global and local variables.....	22
080. How to determine whether a variable exists or not	22
081. How to do different types of sorting.....	22
082. How to check if value is set.....	23
083. How to pause macro execution and resume it later	24
085. How to handle error	24
086. How to go between labels inside code	24
087. How to determine whether some files are opened	25
088. How to convert text file to UTF-8.....	25
089. How to find out type of a variable	25
090. How to change ELBO to BEND and vice versa	25
091. How to set required settings when enabling Autoclash.....	25
092. How to display message in the bottom left corner of the program window	25
093. How to create dimension between elements.....	26
094. How to create dimension between sides of two elements	26
095. How to climb to needed element of hierarchy using loop	26
096. How to determine colour (Autocolour) of current element.....	27
097. How to execute .Net command	27
098. How to find out colour and translucency of element, which were set by Autocolour, or selected manually	28
099. How to get Drawlist and all current 3D-views.....	28
100. How quickly to add elements to view, which are referenced by given element.....	28

001. Shortening time of Splashscreen displaying and using Pause command

How to shorten system loading time and make login form show faster. (Versions 11.6 and below)

File: %pmlib%\monitor\pdmslogon.pmlfrm

It's required to comment next line



```

154  -- *** COMMAND LINE SYNTAX ***
155  --
156  -----
157  define method .init()
158
159  -- -- pause to allow the splashScreen to be displayed prior to this form.
160  -- PAUSE 2

```

PAUSE command allows to make a delay before executing next line of code. Delay is in seconds.

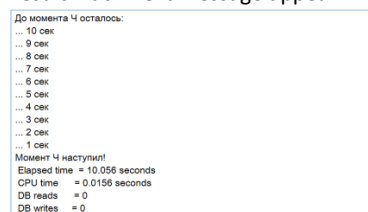
Example. Countdown:

```

clock init
$P before moment X is left:
do !x from 10 to 1 by -1
  $P ... $!x seconds
  PAUSE 1
Enddo
$P Moment X has come!
clock read

```

Result. Each next message appear in 1 second



```

До момента X осталось:
... 10 сек
... 9 сек
... 8 сек
... 7 сек
... 6 сек
... 5 сек
... 4 сек
... 3 сек
... 2 сек
... 1 сек
Момент X наступил!
Elapsed time = 10.056 seconds
CPU time = 0.0156 seconds
DB reads = 0
DB writes = 0

```

002. Opening and automatic updating Association Manager form on system start

Variant with file start from folder %pdmsui%\des\start

In the end of the file add the lines:

```

Show !!associationmanager
!!associationmanager.showAssocs( !!ASSOCIATIONMANAGER.REFRESH, |SELECT| )
15 show !!associationManager
16 !!ASSOCIATIONMANAGER.showAssocs( !!ASSOCIATIONMANAGER.REFRESH, |SELECT| )
17 $.
18

```

In the same way, you can run any other procedure or form

003. Processing all sessions at a selected date

!datetime = '16:45 20 Feb 2013'

```

var !sStr SESSIONS ON $!dateTime
q var !sStr

do !x from 1 to !sStr.Real()
  !session = !x
  !who = SESSU $!session
  HANDLE ANY
    $P Session with number $!session is not existed
  SKIP
  ENDHANDLE
  !when = SESSD $!session
  !comment = SESSC $!session
  $P Session: $!session User: $!who When: $!when Comment: $!comment
enddo

```

004. Attributes for querying information on session comparing

Set date or session number for comparing by command
SETCOMPDATE

You can query the following information in a particular session

Q ATTMDC – List of attributes modified since comparison date

Q ATTMOD(att) – True if specified attribute has been modified in this session

Q ATTMODC(att) – True if specified attribute has been modified since comparison date

Q ATTMODLIST(sess) – List of attributes modified since given session

Q ATTMODLISTC – List of attributes modified since comparison date

Q ELECRE(sess) – True if the element was created since given session

Q ELECREC – True if created since comparison date

Q ELEDEL(sess) – True if the element was deleted since the specified session

Q ELEDELC – True if the element was deleted since comparison date

Q ELEMODO(sess) – True if the element was modified since the specified session

Q ELEMODC – True if the element was modified since comparison date

005. Sample macro for name part replacing dialog

```
!what = !!Alert.Input('What to replace?','')
!with = !!Alert.Input('Replace with?','')
!wher = !!Alert.Input('In elements types (delimited with a space):','')

!splitw = !wher.Split()
do !x from 1 to !splitw.Size()

var !coll coll all $!splitw[$!x] for CE
do !y from 1 to !coll.Size()
  $!coll[$!y]
  !oldname = !CE.Name
  !newname = !oldname.Replace('$!what','$!with')
  NAME $!newname
enddo

enddo
```

006. Writing attributes values for selected elements into array

```
var !coll EVAL ( <attribute_name> ) FOR ALL ( <type> ) WITH ( <condition> ) FOR CE
```

Where

<attribute_name> - attribute name, e.g., TYPE or DESC or DTXR etc.

007. Writing mouse-selected elements into array

```
!graSel = object SELECTION()
!graSel.getCurrent()
!graSelL = !graSel.getSelection()
q var !graSelL
```

008. Include paths to DLL

1. Add new system variable (of any name) for libraries:

```
set CPML=\server\...\CPML\
```

2. In PML:

```
var !CPML EVAR CPML
!namespacePath = !CPML + 'library'
!execute = |IMPORT ' | + !namespacePath + |' |
$!execute
```

009. Setting digit number after the point

```
!var = 12.456
Q var !var.String('D0')
<STRING> '12'
```

```
Q var !var.String('D1')
<STRING> '12.5'
Q var !var.String('D2')
<STRING> '12.46'
Q var !var.String('D3')
<STRING> '12.456'
```

Here we use **String('Dn')** function, where n is desired number of digits after the point. Besides system round values according to basic arithmetic's rules.

010. Example of use autozoom - 'Walk to Drawlist'

```
VAR !COLLECTION COLLECT ALL FROM DRAWLIST
!BLOCK = OBJECT BLOCK('!COLLECTION[!EVALINDEX].DBREF()')
!LIST = !COLLECTION.EVALUATE(!BLOCK)
!VOLUME = OBJECT VOLUME(!LIST)
```

```
!LIM = OBJECT GPHVIEWS()
!LIM.LIMITS(!GPH3DDESIGN1.VIEW, !VOLUME)
```

Other variant:

```
!draw = object drawlist()
!glbDraw = !draw.Globaldrawlist().Members()
if (!glbDraw.size() EQ 0) then
  RETURN
endif
!RT = object array()
do !x from 1 to !glbDraw.Size()
  !RT[!x] = !glbDraw[!x].String()
enddo

!BLOCK = OBJECT BLOCK('!RT[!EVALINDEX].DBREF()')
!LIST = !RT.EVALUATE(!BLOCK)
!VOLUME = OBJECT VOLUME(!LIST)

!LIM = OBJECT GPHVIEWS()
!LIM.LIMITS(!GPH3DDESIGN1.VIEW, !VOLUME)
```

Alternatively, the simplest variant:

```
!volume = object VOLUME(!gphdrawlists.drawlists[1].members())
!gphViews.limits(!gphviews.views[1], !volume)
```

011. Setting focus on a gadget on opening a form

```
!this.keyboardfocus = !this.gadget_name
```

012. Writing all Drawlist elements into array

```
!IsaDrawlist = !gphDrawlists.drawlist(!GPH3DDESIGN1.VIEW)
!IsaDrawlistMembers = !IsaDrawlist.members()
!IsaDrawlistMembersNames = !IsaDrawlistMembers.evaluate(object BLOCK(| !IsaDrawlistMembers[!evalIndex].name|))
```

013. Displaying open file dialog

```
--declare .net namespace to use
using namespace 'Aveva.Pdms.Presentation'
Import 'pmlfilebrowser'
handle any
endhandle
--open
!browser = object PMLFILEBROWSER('LOAD')
!filePath = 'D:\'
!browser.show(!filePath, ", 'Open text file', false, 'Text files (*.txt) | *.txt', 2)
!FileName = !browser.file()
```

The name of the file will be returned to variable !FileName

014. Opening save file dialog

```
-- declare .net namespace to use
using namespace 'Aveva.Pdms.Presentation'
Import 'pmlfilebrowser'
handle any
endhandle
!browser = object PMLFILEBROWSER('SAVE')
!filePath = 'D:\'
!browser.show(!filePath, '', 'Сохранение текстового файла', false, 'Файлы txt (*.txt)|*.txt', 2)
!FileName = !browser.file()
```

The name of the file will be returned to variable !FileName

015. Centring on CE

```
!!gphAlwaysRotate(!gph3ddesign1.view, !!ce)
```

View centre is set at one of the corners of current element volume

016. Calling command/macro on Design module start

For starting Design module with already executed command, you should open file **start**, which is located in %PDMSUI%\des\admin directory in a text editor and add before the last line, which contains \$., a necessary command.

Example: Write a Design module start date and a user name into the log file:

```
VAR !getPdmsDflts SPLIT EVAR PDMSDFLTS
!logFile = object file(!getPdmsDflts[1] + '\enter-log.log')
VAR !currentData[1] CLOCK DATE
VAR !currentData[2] CLOCK TIME
VAR !currentData[3] LOGIN
!currentData[4] = '====='
!logFile.WriteFile('APPEND', !currentData)
```

```
59 VAR !getPdmsDflts SPLIT EVAR PDMSDFLTS
60 !logFile = object file(!getPdmsDflts[1] + '\enter-log.log')
61 VAR !currentData[1] CLOCK DATE
62 VAR !currentData[2] CLOCK TIME
63 VAR !currentData[3] LOGIN
64 !currentData[4] = '====='
65 !logFile.WriteFile('APPEND', !currentData)
66
67 $.
68
```

Also, for example, for starting Auto Clash on system start, it is required to add call of xmenuauto (pdmsui\des\clasher) file in appdesmain.pmlfrm (PMLLIB\design\forms).

Similarly, you can run commands for other modules using file **start** from modules' directories, e.g. **dra\admin\start** – for Draft module.

017. Reading data from Excel using PML.NET functions

```
using namespace 'Aveva.Pdms.Presentation'
Import 'GridControl'
handle any
endhandle
!filepath = 'c:\excel.xls'
!grid = object NETGRIDCONTROL()
!inds = object NETDATASOURCE('Grid Control Example', !filepath)
!grid.bindToDataSource(!inds)

!titles = !grid.getColumn(1)
!names = !grid.getColumn(2)
!icons = !grid.getColumn(3)
!descs = !grid.getColumn(4)
```

018. Displaying progress bar, showing process progress

```
--before the loop
!!FMSYS.setProgress(0)
```

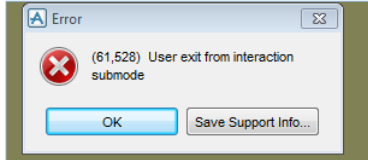
Inside the loop after each operation before `enddo`

```
!percent = 100 * $!x / !items.Size()
!!FMSYS.setProgress( !percent )
```

Where `!x` – index, `!items.Size()` – loop size

019. How to suppress message about exit from pick mode after pressing Esc key

If we run into pick mode with command `ID@` and after exit it pressing Esc key, we get a message



This message can stop program executing. For suppressing it we could use the following code:

```
prompt off
prompt load escape | Pick elements and press Esc|
ID @
HANDLE(61,528)
ENDHANDLE
--Actions
```

```
--For example
!name = !!CE.Name
```

Or picking in loop until pressing Esc:

```
DO !i from 1
prompt off
prompt load escape | Identify ELEMENT to be removed from drawing & press ESC|
ID @
HANDLE(61,528)
BREAK
ENDHANDLE
!NameArray[$!i] = !!CE.NAME
ENDDO
```

020. Turning on Model Editor using PML

```
--Get object
!modelEditor = object STATE()
--Turn on Model Editor
!modelEditor.modifymode(true)
--Turn off Model Editor
!modelEditor.modifymode(false)
```

021. Opening Members list form (old version of Hierarchy Explorer)

```
CALLIB UFORMS MEMB _CDCMEMBER FREE
```

022. Finding out element creator (who created the first session with the element)

Variant 1

```
--Query history of an element
VAR !allHistory HISTORY
Q var !allHistory
<STRING> '826 825 820 816'
```

--as we can see, the first session when element appeared is 816

```
--make array from a string
!historyArray = !allHistory.Split()
q var !historyArray
<ARRAY>
[1] <STRING> '826'
[2] <STRING> '825'
[3] <STRING> '820'
```

```
[4] <STRING> '816'
```

```
-- get the last element of the array (it is a number of the first session)
!firstSession = !historyArray[!historyArray.Size()]
```

```
--and determine session date and author of the session – actually data and author of element creation
VAR !CreateAuthor SESSUSER $!firstSession
VAR !CreateDate SESSDATE $!firstSession
```

Variant 2

```
-- Query history of an element and split it into array
VAR !allHistory history
!historyArray = !allHistory.Split()
```

```
--Query database name which contains a current element and create database object (DB)
VAR !CEDB DBNAME
!DBO = OBJECT DB(!CEDB)
```

```
-- get the last element of the array (it is number of the first session)
!firstSession = !historyArray[!historyArray.Size()]
```

```
--and determine author and date
!CreateAuthor = !DBO.Session($!firstSession).Author
!CreateDate = !DBO.Session($!firstSession).Date
```

Variant 3

```
--query attributes CRUSER, CRDATE, CRSESS
```

023. Collecting elements, which EXCLUDESEVLY lie in given volume

```
var !fullyInVolume collect all EXCLUSIVE within volume ce
do !x from 1 to !fullyInVolume.size()
  ADD $! fullyInVolume [$!x]
  ENHANCE $!fullyInVolume [$!x] COLOUR GREEN
Enddo
```

Or

```
var ! fullyInVolume collect all EXCLUSIVE within e1000 n1000 d100 to e10000 n 30000 u1000
do !x from 1 to !fullyInVolume.size()
  ADD $! fullyInVolume [$!x]
  ENHANCE $!fullyInVolume [$!x] COLOUR GREEN
Enddo
```

For including elements which partly lie in the given volume remove keyword EXCLUSIVE

024. Loading applications with command line

```
!!appxload('DES', 'GEN', false)    $* for application General
!!appxload('DES', 'EQUI', false)   $* for application Equipment
!!appxload('DES', 'PIPE', false)   $* for application Piping
!!appxload('DES', 'CABL', false)   $* for application Cable Tray
!!appxload('DES', 'HVACADV', false) $* for application HVAC Designer
!!appxload('DES', 'STLWRK', false) $* for application Beams/Columns
!!appxload('DES', 'PANEL', false)  $* for application Panels/Plates
!!appxload('DES', 'CIVIL', false)  $* for application Walls/Floors
!!appxload('DES', 'ACCESS', false) $* for application ASL Modeller
!!appxload('DES', 'DTMP', false)   $* for application Design Templates
```

025. How to know which forms are opened

```
q var !!fmsys.shownforms()
```

026. Excel loading function

```
!!EQUILOADCSV() → spreadsheetImport.pmlfrm
```

Example of Excel file. Name/Type are obligatory for new elements.

NAME	TYPE	OWNER	DESCRIPTION
/ENG_UDET_WORLD	UDETWL	/*	Engineering UDET Template world
/ENG_UDA_WORLD	UWRL	/*	Engineering UDA Template world
/ENG_DSX_WORLD	DSXWLD	/*	Engineering DSX Template world
/ENG_DBVW_WORLD	DBVWWL	/*	Engineering DBVIEW Template world
/ENG_STATUS_WORLD	STAWLD	/*	Engineering STATUS Template world

027. How to query which sheets are opened in Draft

```
q var !!appDraMain.sheet.rtext
```

028. Closing all sheets in Draft

```
!!cdrapplic.shee.clear()
!!appDraMain.updateGadgets()
```

029. Getting coordinates of mouse click in sheet field

```
prompt off
prompt load escape |Pick upper left corner of table position|
var !POS SHPO @
prompt on
```

030. Finding out how long operation was executed

```
CLOCK INIT
--operation
CLOCK READ
```

031. Difference between ALPHA LOG and ALPHA FILE

ALPHA LOG – logs all messages from command line, which are sent to command line and are written back
 ALPHA FILE – logs only messages outputted to command line

032. Calling command which is in DLL

Example – displaying attributes form

```
-- Show the PMLNet attributes form
import 'PDMSCOMMANDS'
handle(1000,0)
endhandle

using namespace 'Aveva.Pdms.Presentation.PDMSCommands'
!cm = object PMLNETCOMMANDMANAGER()
!cm.executeCommand('AVEVA.Attributes.Show')
```

033. Encoding connection type string into number

```
var !q HASH ( 'FAER' )
q var !q
```

```
<STRING> '889413'
```

Decoding

```
!coco = !!dehash(889413)
q var !coco
```

```
<STRING> 'FAER'
```

This operation could be required for getting real numeric value of WORD type

034. Using IFTRUE operator

Example – output into report **brwei** value or zero value if **brewi** couldn't be calculated.

```
( IFTRUE ( UNSET(CMPREF OF SPREF) , 0 , BRWE ))
```

IFTRUE checks the first argument, if it's true – then it will return the second argument, otherwise – the third argument.

035. Difference in element collection methods

There are different element collecting methods

PML1 - method (the simplest):

```
VAR !items COLLECT ALL ( <itemType> ) FOR CE
```

PML2 - method, using function:

```
!items = !!CollectAllFor('<itemType>', ||, CE)
```

PML2 - method, using object

```
!colObject = object collection()
!colObject.AddScope(CE)
!colObject.AddType('itemType')
!items = !colObject.Results()
```

What is the difference between them?

The first method collects all elements and put them into array of strings, where each element is reference number. The both other methods return array of DBref elements.

The first one is the fastest, but it's not object-oriented, at the same time the both other methods are more contemporary from the standpoint of programming. However these both methods use the first one.

036. Querying information about deleted elements

Sometimes it is necessary to analyse history of work in model for reading attributes of deleted elements. It can be done by the following actions: set comparing date, using command OLD collect elements of previous sessions and query their attributes:

```
var !db dbname
SETCOMPDATE FOR DB $!db TO SESSION sessno

var !atta old collect all ( type ) with ( deleted() ) for db $!db

do !x values !atta
  q old name of $!x
  q old ( pos of $!atta[$!x] )
enddo
```

037. How to get coordinates of mouse click on element (not Origin point)

To get coordinates of some place on the element where a user clicked you can use command PICK:

```
VAR !pickMe PICK
```

After the command system switches to mode of selecting element and after selecting is done make a query:

```
Q VAR !pickMe
```

It returns array with information about element and click position

```
<ARRAY>
[1] <STRING> '=19409/1108642'
[2] <STRING> 'Line D Through W 6135.652 S 83941.578 U 1700'
```

038. How to find out whether ISODRAFTMODE is on

```
VAR !isotype ISOTYPE
HANDLE (47,15)
--it means that ISOAFRAFTMODE is off
ELSEHANDLE none
-- exit from mode
EXIT
```

ENDHANDLE

039. How to move annotation element of drawing into explicit coordinates

XYPO XN YN

Where N – coordinates values

Coordinates of origin point are not affected.

040. How to find out coordinates of a centre and end-points of a tube

ITPOS OF TUBI – coordinates of a centre of a tube

P1POS – coordinates of the beginning of a tube

P2POS – coordinates if the end of a tube

041. How to query position of object relative to drawing view

Q VSIDE OF **ITEM_REFNO** – returns required coordinates

042. How to change period to comma using intelligent text

#POSU(P.:1),#POSU(P.2:)(C2:)(PM:1)

043. How to send text file to printer

It can be achieved by using system commands with NOTEPAD

```
VAR !COMMAND ' NOTEPAD /P ' + "'ПУТЬ_К_ФАЙЛУ'"
SYSCOM |$!COMMAND|
```

044. How to find out computer and Windows user name

VAR !HOST HOST – name of computer

VAR !USERNAME EVAR USERNAME – name of user

045. How to find out 3D coordinates by clicking on drawing

VAR !GET3DFROM2D ENUPOS OF @ – click within view field and get coordinates

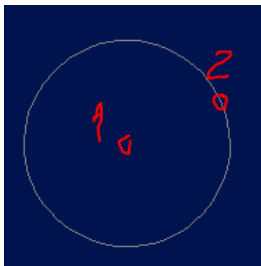
046. How to find out 2D coordinates by clicking on drawing

VAR !GET2D SHPOS @ – click within view field and get coordinates

047. Commands for creating primitive elements on drawing and their attributes

NEW CIRC DEF @ – creating circle by picking centre (1) and radius (2)

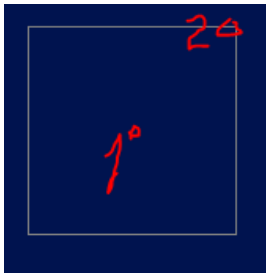
VAR !GETDIAM DIAM – diameter



NEW RECT ASDEF @ – creating square by picking centre (1) and corner (2)

VAR !GETXLEN XLEN – width

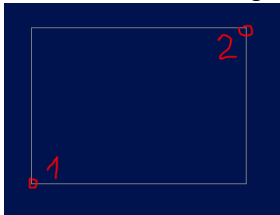
VAR !GETYLEN YLEN – height



NEW RECT DEF @ – creating rectangular by picking two opposite corners

VAR !GETXLEN XLEN – width

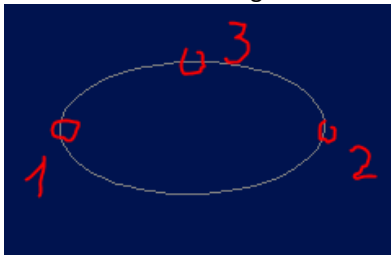
VAR !GETYLEN YLEN – height



NEW ELLI DEF @ – creating ellipse by picking start point (1), width (2) and height (3)

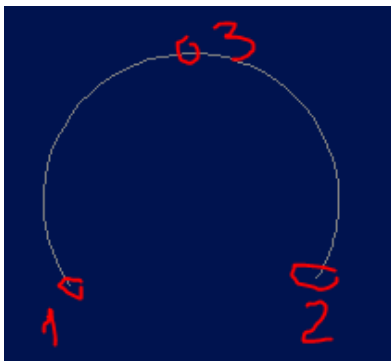
VAR !GETXLEN XLEN – width

VAR !GETYLEN YLEN – height



NEW ARC DEF @ – creating arc by picking start point (1), width (2) and radius (3)

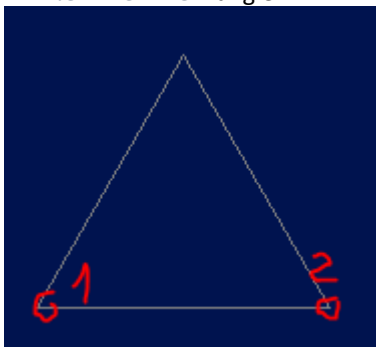
VAR !GETRADI RADI – radius



NEW ETRI DEF @ – creating equilateral triangle by picking start point (1) and end point (2) of base

VAR !GETLEN LENGTH – length of side

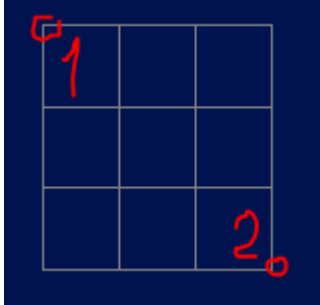
VAR !GETDEG ADEG – angle



NEW TABL DEF @ – creating table by picking two opposite corners

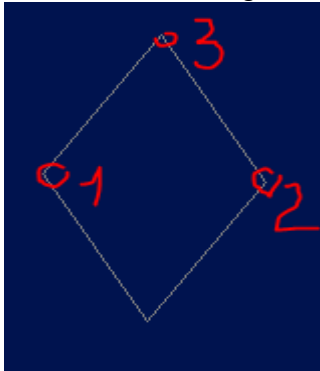
VAR !GETXLEN XLEN – width
VAR !GETYLEN YLEN – height

Number of rows is defined by **NROW** attribute and **NCOLUMN** for columns. Rows and columns have equal size proportional to table size and its' number, i.e. width if column would be equals to (**XLEN / NCOLUMN**), and height of row equals to (**YLEN / NROW**). Size of row and columns is changed dynamically by changing table size.



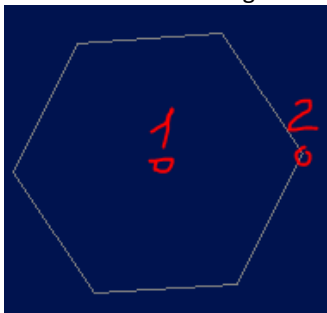
NEW DMND DEF @ – creating diamond by picking three points: start (1), width (2) and height (3)

VAR !GETXLEN XLEN – width
VAR !GETYLEN YLEN – height
VAR !GETDEG ADEG – angle



NEW HEXA DEF @ – creating hexagon by picking centre point (1) and vertex (2)

VAR !GETDIAM DIAM – diameter
VAR !GETDEG ADEG – angle



All primitive elements have origin point, that can be queried using:

VAR !GETORIGIN ORIGIN
Q VAR !GETORIGIN
<STRING> 'CENTRE X 366MM Y 195MM'
 So you can get the coordinates.

048. How automatically calculate view scale

AUTOSCALE – autoscale by content of view

or

AUTOSCALE LIMITS – autoscale using attribute LIM1 (usually for Limits-Defined views) of Drawlist of view.

049. How to move content of view to centre after change size of view

ONPOS 0 0 – reset position of centre of view content

AUTOSCALE – recalculate scale

UPDATE ALL – refresh view

050. How automatically to set size of view by 3D coordinates

AUTOSIZE FROM **POSITION1** TO **POSITION2** – from *the first 3D coordinate* to *the second* (two opposite corners)

Or you can use mouse to change size of existing view:

AUTOSIZE FROM @ TO @ - click at any to places in existing view and change its size (usually shrink it)

After changing size you can automatically change scale by – AUTOSCALE and refresh a picture by UPDATE ALL

051. How to collect elements of different owners

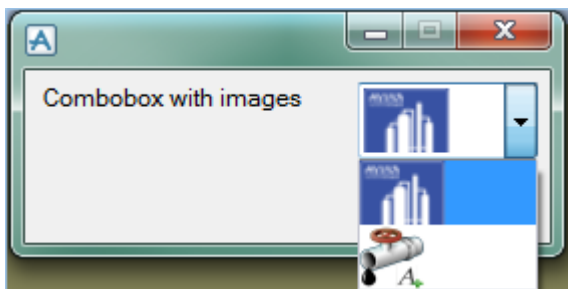
VAR !GETMEMBERS COLLECT (**ALL SITE MEM** **ALL ZONE MEM** **ALL EQUI MEM**) FOR CE – collect members of the given elements

052. How to put combobox with images on form

OPTION .TESTOPTION 'ЭТО СПИСОК С КАРТИНКАМИ' PIX WIDTH 64 HEIGHT 64

VAR LIST _TESTOPTION PAIRS |MYPICTURE.PNG| |1|
|MYPICTURE2.PNG| |2|

EXIT



053. How to add synonyms to commands

Synonym is a word or a set of words, which when inputted into command window execute a linked command. One of the most frequent variant of usage is executing set of commands from macro.

Synonyms are stored in file **VARs**, which is located in folder %PDMSUI%/module/ADMIN

Command \$\$ defines synonym word and after symbol “=” a linked command.

\$U is used for making non-overridable and non-deletable synonym.

Example: let's define synonym, which returns information about bolting on current element, if it is flange.

QB

Flange: =23584/5470|

--Content of bolting:

== 1 . 16 x Description: NUT Material: ASTM A193-B7/2H

== 2 . 16 x Description: WASHER Material: ASTM A193-B7/2H

== 3 . 8 x Description: LONG STUD BOLTS Material: ASTM A193-B7/2H

== Total weight of bolting (Stud+Nut+Washer) = 1.2192kg

To implement this do as follows:

1. Write macro and save it to file with the name arQB.pmlmac into folder %PMLLIB%

Here is the macro content:

```

if (!!CE.Type NEQ 'FLAN') then
  $P Works only on FLAN elements
  RETURN
Endif

!getCEName = !!CE.Namn
$P Flange: $!getCEName
$P --Content of bolting:
VAR !getTotal P1 BOLT TOTAL
do !x from 1 to !getTotal.Real()
  VAR !getDesc P1 BOLT $!x RTEXT
  if (!!getDesc.Matchwild('*-LEN-*')) then
    VAR !getBlen P1 BOLT $!x BLEN
    !getDesc = !getDesc.Replace('-LEN-', '$!getBlen')
  endif
  VAR !getMate P1 BOLT $!x XTEXT
  VAR !getQuan P1 BOLT $!x NOFF
  $P == $!x . $!getQuan x Description: $!getDesc Material: $!getMate
enddo

```

```
VAR !GetTotalIW P1 BOLT TOTW
```

```
$P == Total weight of bolting (Stud+Nut+Washer) = $!getTotalIW
```

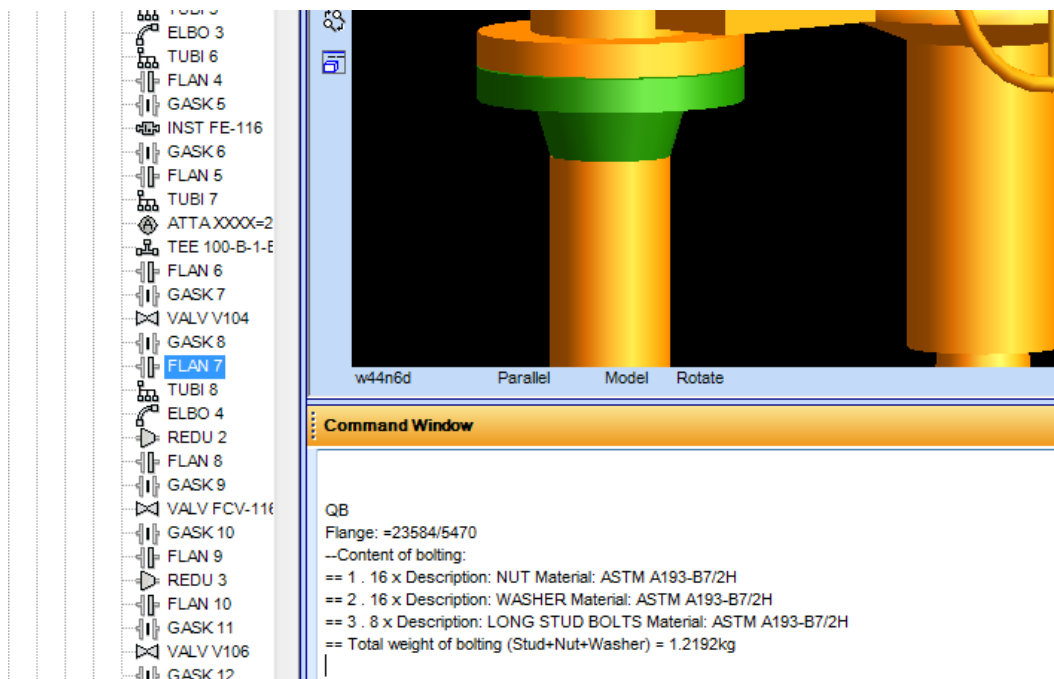
2. Open file VARS of module Design (%PDMSUI%/Des/Admin/) and add following lines to the end of it

```
$S QB = $M/%PMLLIB%\arQB.pmlmac
```

```
$U QB
```

3.Restart PDMS/Marine

Now if you select flange and type command **QB** in command window, it will execute the code of the macro (which is linked with the synonym)



It is possible to make synonym available for all users, to do this you should copy changed VARS file to your network folder %PDMSUI%/module/admin

NB. Code above does not handle any errors, e.g. if bolts do not have references to description or material.

054. How to find out hierarchy depth

VAR !depth DBDEPTH – query a depth of the current element, where WORLD has depth 0

Q var !!CE.ownlst – array of owners of the current element, at that the first element is the highest level element, i.e. main owner

055. How to collect elements with given value of the first member attribute

var !getItemS COLLECT ALL SPEC WITH (**TEXT OF FIRST MEM EQ 'INSU'**) – collects array of elements, whose **attribute of the first member** equals to **given value** (the given example collect all specification of isolation)

056. How to find enhanced labels and run position alignment

Var !getEnhancedLabels SPLIT ENHANCE – get array of labels enhanced by command ENHANCE

SPREAD LOCAL XOFF <**XOFF value**> YOFF <**YOFF value**> SELECT ALL FROM !getEnhancedLabels – positions labels locally using given offsets by axis.

If you want to calculate offsets for given length of leader line (radius) and angle, you can use the following expressions:

XOFF value = length of leader line * COS (angle)

YOFF value = length of leader line * SIN (angle)

SPREAD REMOTE OMIT <**OMIT value**> GAP <**GAP value**> MARGIN <**MARGIN value**> SELECT ALL FROM !getEnhancedLabels – positions labels remotely using the given values:

- prevent Labels being placed alongside the specified side – **OMIT value** (combination of LEFT RIGHT BOTTOM TOP)

- minimum gap between two labels – **GAP value**

- distance from border of view – **MARGIN value**

SPREAD REMOTE OMIT <**OMIT value**> GAP <**GAP value**> REPOSITION X <**X1 value**> Y <**Y1 value**> X <**X2 value**> Y <**Y2 value**> SELECT ALL FROM !getEnhancedLabels – positions labels remotely using given values:

- prevent Labels being placed alongside the specified side – **OMIT value** (combination of LEFT RIGHT BOTTOM TOP)

- minimum gap between two labels – **GAP value**

- rectangle corners around which labels will be positioned – **X1,X2,Y1,Y2 values**

057. Setting view direction for 3D view

!!gphViews.look (!!gph3Ddesign1.view, N90D) – plan North

!!gphViews.look (!!gph3Ddesign1.view, S90D) – plan South

!!gphViews.look (!!gph3Ddesign1.view, E90D) – plan East

!!gphViews.look (!!gph3Ddesign1.view, W90D) – plan West

!!gphViews.look (!!gph3Ddesign1.view, N) – look at North

!!gphViews.look (!!gph3Ddesign1.view, S) – look at South

!!gphViews.look (!!gph3Ddesign1.view, E) – look at East

!!gphViews.look (!!gph3Ddesign1.view, W) – look at West

!!gphViews.look (!!gph3Ddesign1.view, S45W35D) – ISO1

!!gphViews.look (!!gph3Ddesign1.view, W45N35D) – ISO2

!!gphViews.look (!!gph3Ddesign1.view, E45N35D) – ISO3

!!gphViews.look (!!gph3Ddesign1.view, S45E35D) – ISO4

058. Finding out screen resolution

!screenInfoFile = |c:\temp\screen_info.txt|

SYSCOM | WMIC DesktopMonitor get ScreenWidth, ScreenHeight > \$!screenInfoFile |

Next in PML open file C:\temp\screen.txt , read it and split the second line. The first value will be Height , the second – Width

!screenFileObj = object file(!screenInfoFile)

!screenHeight = !screenFileObj.ReadFile()[2].Split()[1].Trim()

!screenWidth = !screenFileObj.ReadFile()[2].Split()[2].Trim()

Q var !screenHeight

Q var !screenWidth

059. How to calculate distance between two points (strait segment length)

Variant 1. Using temporary PINs. Place them into corresponding coordinates and query distance between them using PML 1 commands:

--pick the first element, get its position, and after that place PIN there


```

ID @
!pos1 = !!CE.Pos.Wrt(WORLD)
PIN 1 AT $!pos1

-- pick the second element, get its position, and after that place PIN there
ID @
!pos2 = !!CE.Pos.Wrt(WORLD)
PIN 2 AT $!pos2

--query distance
VAR !getDistance CONST DIST PIN1 TO PIN2
Q var !getDistance

```

Variant 2. Using PML2 method Distance of object Position

```

-- pick the first element and get its position
ID @
!pos1 = !!CE.Pos.Wrt(WORLD)

-- pick the second element and get its position
ID @
!pos2 = !!CE.Pos.Wrt(WORLD)

--calculate distance between two coordinates using Distance method and convert real value to string with two digits after period
!getDistance = !pos1.Distance(!pos2).String('D2')
Q var !getDistance

```

060. How to find middle of a segment between two coordinates

Using PML2 method Midpoint of object Position:

```

-- pick the first and the second elements and get their positions
ID @
!pos1 = !!CE.Pos.Wrt(WORLD)
ID @
!pos2 = !!CE.Pos.Wrt(WORLD)
--construct aid line for visualization of segment
AID LINE NUM 1 $!pos1 TO $!pos2 LINESTYLE DOTTED
--calculate middle point of a segment
!getMiddle = !pos1.Midpoint(!pos2)
Q var !getMiddle
--add text label into the middle of the segment
AID TEXT 'V' AT $!getMiddle

```

061. What options can be used on starting of the system

In link properties, which points at batch file, you can specify different arguments, which activate starting options.

Target type: Windows batch file

Target location: PDMS12.1.SP4

Target:

Examples:

1. Automatic start of project with selected user, MDB and module

```
...pdms.bat DESIGN SAM SYSTEM/XXXXXX /SAMPLE
```

2. Automatic start of project with selected user, MDB, in non-graphic mode (command line mode) and macro executing, at that

If macro will manage switching of modules, then module should not be specified in arguments and system will log in into system database:

```
...pdms.bat TTY SAM SYSTEM/XXXXXX /SAMPLE $M/c:\myMacro.pmlmac
```

or

```
...pdms.bat BATCH SAM SYSTEM/XXXXXX /SAMPLE $M/c:\myMacro.pmlmac
```

or

```
...pdms.bat NOGRAPHICS SAM SYSTEM/XXXXXX /SAMPLE $M/c:\myMacro.pmlmac
```

If you need to enter into specific module, then you should add its name in the beginning:

```
...pdms.bat DESIGN TTY SAM SYSTEM/XXXXXX /SAMPLE $M/c:\myMacro.pmlmac
```

Or

```
...pdms.bat DRAFT BATCH SAM SYSTEM/XXXXXX /SAMPLE $M/c:\myMacro.pmlmac
```

Or

```
...pdms.bat PARAGON NOGRAPHICS SAM SYSTEM/XXXXXX /SAMPLE $M/c:\myMacro.pmlmac
```

3. Automatic start of project with selected user, MDB and module in read only mode

```
...pdms.bat DESIGN -readOnly:True SAM SYSTEM/XXXXXX /SAMPLE
```

4. Automatic start of project with selected user, MDB and module in integrate schematics and engineering mode (3D Schematic Integrator licence will be used)

```
...pdms.bat DESIGN INTEGRATEDMODE SAM SYSTEM/XXXXXX /SAMPLE
```

5. Automatic start of project with selected user, MDB and module without console window

```
...pdms.bat DESIGN NOCONSOLE SAM SYSTEM/XXXXXX /SAMPLE
```

062. How to display system elements of hierarchy

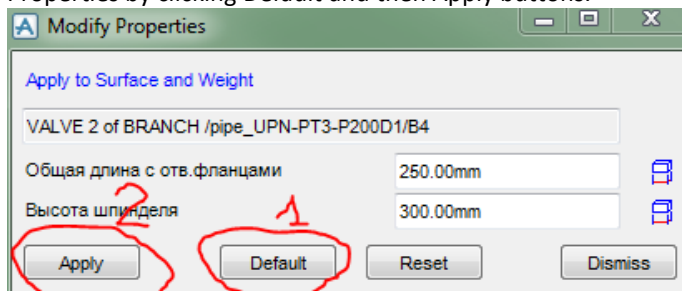
\$T8+ - enables displaying of elements, which are not intended to be created by user

\$T8- - disables displaying of elements, which are not intended to be created by user

**It is necessary to refresh hierarchy after that operation.
For example, collapse and expand World element**

063. How to set default properties programmatically

You need to set default values for the elements which have links to edited properties. It is usually done in form Modify Properties by clicking Default and then Apply buttons.



You can programmatically emulate this by the following code:

```
do !x from 1 to 5
!!comPropCntrl.data[$!x].initEditForm(|DEFAULT|, true)
HANDLE ANY
ENDHANDLE
!!comPropCntrl.data[$!x].okEditForm()
HANDLE ANY
ENDHANDLE
Enddo
```

NB. The cycle is necessary to catch the number of Data array cell. This number is connected to the number of Modify-Properties form and it occasionally changes.

064. How to open PDF file on a given page

--specify path to Acrobat Reader

```
!acropath = 'C:\Program Files (x86)\Adobe\Reader 11.0\Reader\AcroRd32.exe'
```

--specify path to file

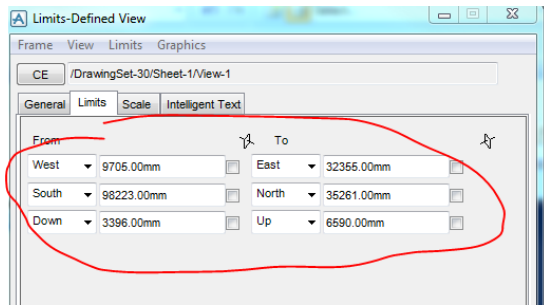
```
!filepath = 'C:\AVEVA-PROJECTS\RDP12\RDPDATA\AVEVA Plant Manuals 12.1.SP4\Administrator Command Reference Manual.pdf'
```

--execute system command with arguments

```
SYSCOM | ""$!acropath" /A "page=39" "$!filepath" "&|
```

065. How to programmatically update Limits-Defined view of drawing after changing a Drawlist

After changing content of Drawlist, you need programmatically update Limits-Defined view:



It can be achieved by running the following command on the required View:
CALLDRG ULIMITS GETLIMITS _CDROVIEW DRAW FALSE

066. How to open Folder Open Dialog

```
import 'PMLFileBrowser'
handle any
endhandle
using namespace 'Aveva.Pdms.Presentation'
!folder = object PMLFolderBrowser()
!folder.Show('Select a folder',true)
!FolderPath = !folder.selectedPath()
q var !FolderPath
```

067. How to draw aid label in the head and in the tail of a branch

```
--read coordinates of the head and the tail of a current brunch
!getHpos = !!CE.Hpos
!getTpos = !!CE.Tpos
--draw aid labels at these coordinates
AID TEXT 'Head' AT $!getHpos
AID TEXT 'Tail' AT $!getTpos
```

068. How to display colour numbers and corresponding names

```
do !x from 1 to 365
  VAR !getCol COL $!x
  $P Colout number $!x = $!getCol
Enddo
```

There are only 365 colour variants.

Another variant to query full information about a colour is using the object COLOUR

```
!getColour = object colour(colour number)
```

```
!q = object colour(2)
q var !q

<COLOUR> red
BLUE <REAL> Unset
CODE <REAL> 2
GREEN <REAL> Unset
NAME <STRING> 'red'
RED <REAL> Unset
TYPE <STRING> 'SPECIFIC'
```

070. How to automatically execute commands on changing current element

It is necessary to do some actions on changing a current element.

Example:

On changing current element its type will be automatically written in the form field.

```
kill !!testForm
setup form !! testForm
TRACK |PADDCE| call |!this.trackce()|
text .text1 'CE type' width 10 is string
exit

-----
define method .trackce()
!getType = !!CE.Type
!this.text1.val = !getType
Endmethod
```

Here, after CE has changed, `trackce()` method commands are run

Changing of CE is controlled by command `TRACK`, and the database type is controlled by the following string arguments

Database types for TRACK command:

- SYSTCE – system databases (module Admin)
- PADDCE – PADD databases (Draft)
- DESICE – Design databases
- CATACE – Catalogue databases
- ISODCE – Isodraft databases
- SCHECE – Schematics databases
- PROPCE – Properties databases
- MANUCE – Manufacturing databases

071. How to show information on all attributes of the system

You can see detailed information about attributes with the help of the special form, which can be displayed by the command:

```
SHOW !!attlib
```

This form reads attlib.dat file, which contains all system attributes, from installation root. Editing the file is not permitted.

072. How to query full description of attribute

How to get attribute type, size, full name and etc.

```
VAR !data ATTDEF attribute NAME RPTX TYPE DEFI SIZE VISI QSET UNIT
```

Where *attribute* is a name of analyzed attribute, and arguments written with green are necessary characteristics of the attribute.

In !data is returned array, which size and order of elements depend on which characteristics and their order were supplied.

Example of query:

```
VAR !data ATTDEF POS NAME RPTX TYPE DEFI SIZE VISI QSET UNIT
```

```
Q var !data
```

```
<ARRAY>
```

```
[1] <STRING> 'POS'
[2] <STRING> 'POSITION'
[3] <STRING> '8'
[4] <STRING> '5'
[5] <STRING> '3'
[6] <STRING> 'true'
[7] <STRING> 'true'
[8] <STRING> 'DIST'
```

List of characteristics:

NAME – name of attribute

RPTX – full description of attribute (Report Text)

TYPE – type of attribute value

- 0 = All
- 1 = Integer
- 2 = Real
- 3 = Logical
- 4 = Text
- 5 = Reference
- 6 = Word
- 7 = Orientation
- 8 = Position
- 9 = Direction

DEFI – system definition of attribute

- 1 = DDL

2 = Dynamic
 3 = DDL or Dynamic
 4 = Pseudo
 5 = DDL or Pseudo
 6 = Dynamic or Pseudo
 7 = DDL or Dynamic or Pseudo

SIZE – size of value

VISI – visibility

QSET – quality to be set

UNIT – units of measure

Example: you need to know if an attribute is of Reference type

```
--query a single characteristic (TYPE) of attribute
VAR !checkAttType ATTDEF CREF TYPE
--check returned value
If (!checkAttType EQ '5') then
  $P Attribute of Reference type
Endif
```

073. How to get array of all available attributes in the system

Attributes data is stored in file attlib.dat.

```
--search and read the file (for searching build-in function !!searchPaths is used)
!getAttFile = !!searchPaths('attlib.dat')
!getAttList = !getAttFile[1].file.readFile()
```

After that operation array **!getAttList** will contain all attributes of the system (unordered)

074. How to select rectangle area on drawing

```
VAR !getWindow RSHPOS @
```

The coordinates of corners of rectangle area selected by user will be saved in variable **!getWindow**

075. How to select labels on drawing using area selecting

```
--select area
```

```
VAR !getWindow RSHPOS @
```

```
--split to coordinates values
```

```
var !LeftX PART(|$!getWindow|,2)
var !RightX PART(|$!getWindow|,6)
var !LeftY part(|$!getWindow|,4)
var !RightY part(|$!getWindow|,8)
```

```
--check how clicks were done (left to right or right to left)
```

```
if (($!LeftX) GT ($!RightX)) then
  var !TMP |$!LeftX|
  var !LeftX |$!RightX|
  var !RightX |$!TMP|
endif
```

```
if (($!LeftY) gt ($!RightY)) then
  var !TMP |$!LeftY|
  var !LeftY |$!RightY|
  var !RightY |$!TMP|
endif
```

```
--select labels
```

```
ENHANCE ALL (GLAB SLAB) WITH (xyps[1] GE ($!LeftX) AND xyps[1] LE ($!RightX) AND xyps[2] GE ($!LeftY) AND xyps[2] LE ($!RightY)) FOR VIEW
```

```
--get array of selected labels
```

```
VAR !getEnhanced SPLIT ENHANCE
```

076. How to calculate total length of tubes

It is necessary to calculate overall length of tubes beneath current element

```

--collect tubes elements
VAR !tubis COLLECT ALL TUBI FOR CE
--define variable which will increase while length is being calculated
!totalLength = 0
--query if BENDs should be included
!IsbendsNeeded = !!Alert.Confirm('Do include bends?')
--run summing on all tubes elements length
do !x from 1 to !tubis.Size()
    !totalLength = !totalLength + !tubis[!x].Dbref().Itle
    HANDLE ANY
ENDHANDLE
enddo
--add length of bends if YES was clicked
if (!IsBendsNeeded EQ 'YES') then
    !text = ' , including bends, '
    VAR !bends COLLECT ALL BEND FOR CE
    do !x from 1 to !tubis.Size()
        !totalLength = !totalLength + !bends[!x].Dbref().Fitlen
        HANDLE ANY
    ENDHANDLE
    enddo
else
    !text = ' , without bends, '
endif
--display message
!!Alert.Message('Overall length of tubes $!text = ' + !totalLength.String('D3'))

```

077. How to query the title of a current form of a module or an application

```

Q VAR !!appcntrl.formtitle
Q VAR !!appcntrl.currentapp

```

078. How to call macro which sets defaults for layers

In the process of VIEW creation in module Draft some settings (i.e. attributes) of LAYers, such as FONTs, colours etc., are set as defaults values from corresponding defaults file. Each layer has its own file. The files are located in folder %PDMSDFLTs% and have names which begin with dra-gen -*lay, where * - three character code of the layer, e.g. dim or lab (for more information see. «100 Secrets of setting Draft»). When layer is created these files are executed line by line.

File which is responsible for reading the settings is:

%PDMSUI%\dra\gen\usetlayprp

And it is called as:

CALLDRG USETLAYPRP \$<\$!PURP\$>

Where **!PURP** – code of layer type (DIM , LAB ...)

Code for running default setting

CALLDEF \$!FILE

Where **!FILE** variable is set as the name of the appropriate file with default settings

079. How to get list of global and local variables

Q VAR LOC – shows the list of local variables for current session

Q VAR GLO - shows the list of global variables

080. How to determine whether a variable exists or not

VAR !checkVar DEFINED (!someVar)

Queries existence of variable with name **!someVar**. If it exists then the variable **!checkVar** will be returned as **true**, otherwise as **false**

081. How to do different types of sorting

Let's have the following array !array

```

[1] <STRING> 'Camel'
[2] <STRING> 'Zebra'

```

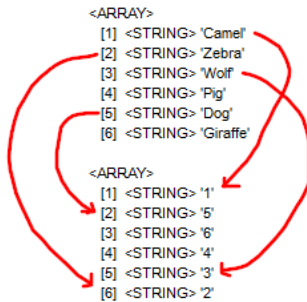
```
[3] <STRING> 'Wolf'
[4] <STRING> 'Pig'
[5] <STRING> 'Dog'
[6] <STRING> 'Giraffe'
```

Next we show several methods of sorting

1. Sorting of element indices according to alphabetical order of elements

```
VAR !SortIndex SORT !ARRAY
```

Result:

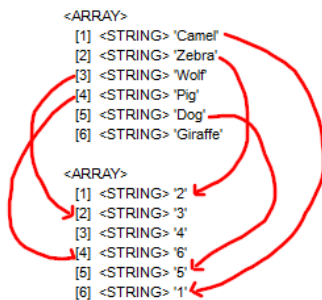


As you can see in array `!SortIndex` were put indices of elements of array `!Array` according to alphabetical order

2. Sorting of element indices according to descending alphabetical order of elements

```
VAR !SortIndex SORT !ARRAY DESCENDING
```

Result:



3. Sorting of elements values in ascending alphabetic order

```
!array.Sort()
```

```
<ARRAY>
[1] <STRING> 'Camel'
[2] <STRING> 'Dog'
[3] <STRING> 'Giraffe'
[4] <STRING> 'Pig'
[5] <STRING> 'Wolf'
[6] <STRING> 'Zebra'
```

It should be considered that method `sort()` sorts original array and after sorting is done you cannot get original order back. Therefore if you need to save original array you should define additional variable and assign value of original array to it:

```
!newArray = !array
!newArray.Sort()
```

Then the original array will remain as it was, and the new one will be sorted.

082. How to check if value is set

VAR !checkUnset UNSET (**attribute**)

Where **attribute** – name of attribute, e.g. Desc

If value is not set, then 'true' will be returned.

083. How to pause macro execution and resume it later

Sometimes it is necessary to pause execution of macro, then to do some actions in system and to resume after it. To do so you should use following commands:

--here is the first part of macro before making a pause

\$M-

--and here is the second one, which is continue to execute after resume

Command **\$M-** inside macro gets a signal to the system to pause execution and allows a user to make some actions. To resume macro execution from the line where it was stopped use command **\$M+**

085. How to handle error

Usually errors stop code execution, but sometimes it is necessary to resume work of macro even if an error occurs.

Example:

-- create new element with a given name

NEW EQUI /EquiName

--handle error, which occurs if such a name already exists

HANDLE (41,12)

\$P Name already exists

DELETE EQUI

--if there is no error, then execute another code

ELSEHANDLE NONE

\$P Created

ENDHANDLE

To handle any error you can use following code:

HANDLE ANY

--code of handling error

ENDHANDLE

086. How to go between labels inside code

Sometimes it is necessary to do quick jump to another part of code, which is marked by a special label. Example: show data input window to user, check inputted data, and if data do not agree with some condition, then it should be returned to data input window.

--Set start label

LABEL /startagain

--show data input window

!inputD = !!Alert.Input('Enter number between 1 and 10','1')

--check that user has inputted a number

!isReal = !inputD.Real()

HANDLE ANY

!!Alert.Message('It is not number. Repeat input...')

--return to start label

GOLABEL /startagain

ENDHANDLE

--next, if it is a number, then check that it is between 1 and 10

if (!inputD.Real() LT 1 OR !inputD.Real() GT 10) then

!!Alert.Message('Inputted number is not in required range. Repeat input...')

-- return to start label

GOLABEL /startagain

endif

--execute code if all conditions are complied

!!Alert.Message('You have entered \$!inputD ')

087. How to determine whether some files are opened

VAR !checkOpenFiles OPENF

If there are some opened (and not closed) files, then their names will be returned.

088. How to convert text file to UTF-8

```
!FileToConvert = 'c:\temp\textfile.txt'
!ConverterPath = 'C:\AVEVA\Plant\PDMS12.1.SP4\Transc.exe'
SYSCOM 'CMD /C $!ConverterPath 0 "$!FileToConvert" 65001 -s ' & '
Convert to UTF-8
```

```
!FileToConvert = 'c:\temp\textfile.txt'
!ConverterPath = 'C:\AVEVA\Plant\PDMS12.1.SP4\Transc.exe'
SYSCOM 'CMD /C $!ConverterPath 0 "$!FileToConvert" 65001 -s -n & '
Convert UTF-8 without BOM
```

089. How to find out type of a variable

Q VAR !var.ObjectType()

090. How to change ELBO to BEND and vice versa

CHANGETYPE TO BEND

CHANGETYPE TO ELBO

At that you should change Spref and reconnect element.

091. How to set required settings when enabling Autoclash

You can enable Autoclash by the following command

CALLCL XAUTO TRUE

At that macro **XAUTO** (PDMSUI\des\clasher) is called, which calls **DCLASHER** by the command **CALLCL DCLASHER \$!SYSTEM \$!FORCE**

Where !SYSTEM = true, !FORCE = false

Command for switching on Autoclash:

AUTOCLASH ON (the command is in set of DESCLASH commands)

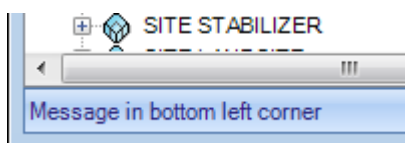
It is called from **XAUTO**

To add own settings or call other commands when enabling Autoclash, edit configuration file **XAUTO**

```
46 $* Turn AUTOClash ON
47 if(!MODE.upcase() eq (TRUE)) then
48
49
50 $* Set Flags is in Clash Mode
51 ... DESCLASH
52 ... AUTOCLASH ON
53
54 !newv = !tarGetNewLimitsOfExpandedVolume(500)
55 ... REM OBST ALL
56 ... OBST ce
57 ... LIMITS $!newv
58
59 ... REPORT GRAPHICS ON
60 ... CLASHLIST DISPLAY
61 ... handle (60 71) & $!TarGetNewLimitsOfExpandedVolume
```

092. How to display message in the bottom left corner of the program window

PROMPT 'Message in bottom left corner'



To hide it use:

PROMPT DISMISS

093. How to create dimension between elements

If you have VIEW with direction Down, i.e. Plan North:

Horizontal dimension between two elements

NEW LDIM FROM /item1 TO /item2

Vertical dimension between two elements

NEW LDIM DIR N FROM /item1 TO /item2

Similarly, dimension between two P-points of elements (if they exist)

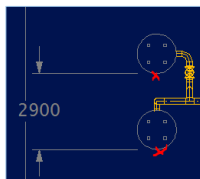
NEW LDIM DIR N FROM P1 Of /item3 TO P2 Of /item4

094. How to create dimension between sides of two elements

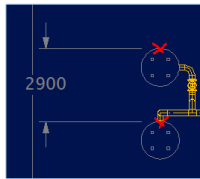
If you have VIEW with direction Down, i.e. Plan North:

Vertical directed dimension:

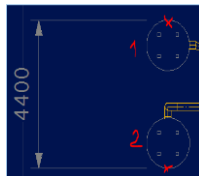
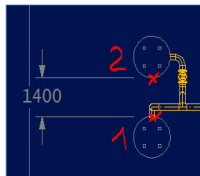
NEW LDIM DIR N FROM BEFORE ID@ TO BEFORE ID@



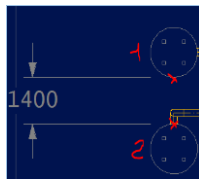
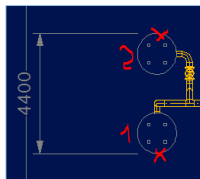
NEW LDIM DIR N FROM AFTER ID@ TO AFTER ID@



NEW LDIM DIR N FROM AFTER ID@ TO BEFORE ID@



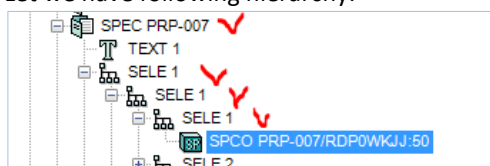
NEW LDIM DIR N FROM BEFORE ID@ TO AFTER ID@



Similarly for horizontal directed dimension.

095. How to climb to needed element of hierarchy using loop

Let we have following hierarchy:



We need to climb from SPCO level to SPEC level, at the same time collecting some attributes of SPCO elements of the way.

Example (we assume that CE = SPCO)

```

if (!!CE.Type NEQ 'SPCO') then
  !!Alert.Message('Go to SPCO')
  RETURN
endif

--remember CE
!ceName = !!CE.Name

--define array for storing attributes
!outputArray = object array()
!outputArray[1] = 'SPCO даёт ответ=' + !!CE.Tanswer
do
  --go to owner
  OWNER
  if (!!CE.Type EQ 'SELE') then
    --query attributes
    if (!!ce.Answer EQ 0) then
      !Answer = !!CE.Tanswer
    else
      !Answer = !!CE.Answer.String()
      if (!!CE.Maxanswer NEQ 0 AND !!CE.Maxanswer NEQ !!CE.Answer) then
        !Answer = !Answer + '-' + !!CE.Maxanswer.String()
      endif
    endif
  endif

  --append data to array
  !outputArray.Append('SELE answers=' + !Answer + ' and asks=' + !!CE.Quest)
endif

--if we are on SPEC level, then it is the last row in array and we exit from loop
if (!!CE.Type EQ 'SPEC') then
  !outputArray.Append('SPEC asks=' + !!CE.Quest)
  BREAK
endif
enddo

--invert array
!outputInvert = !outputArray
!outputInvert.Invert()

--output selection history
$IceName
$P Selected SPCO has following selection criteria:
do !x from 1 to !outputInvert.Size()
  $P $!x == $!outputInvert[$!x]
Enddo

```

096. How to determine colour (Autocolour) of current element

```
VAR !getRuleColour AUTOCOLOUR FOR CE
```

The command returns colour of a current element, if it was found in Autocolour Rules.

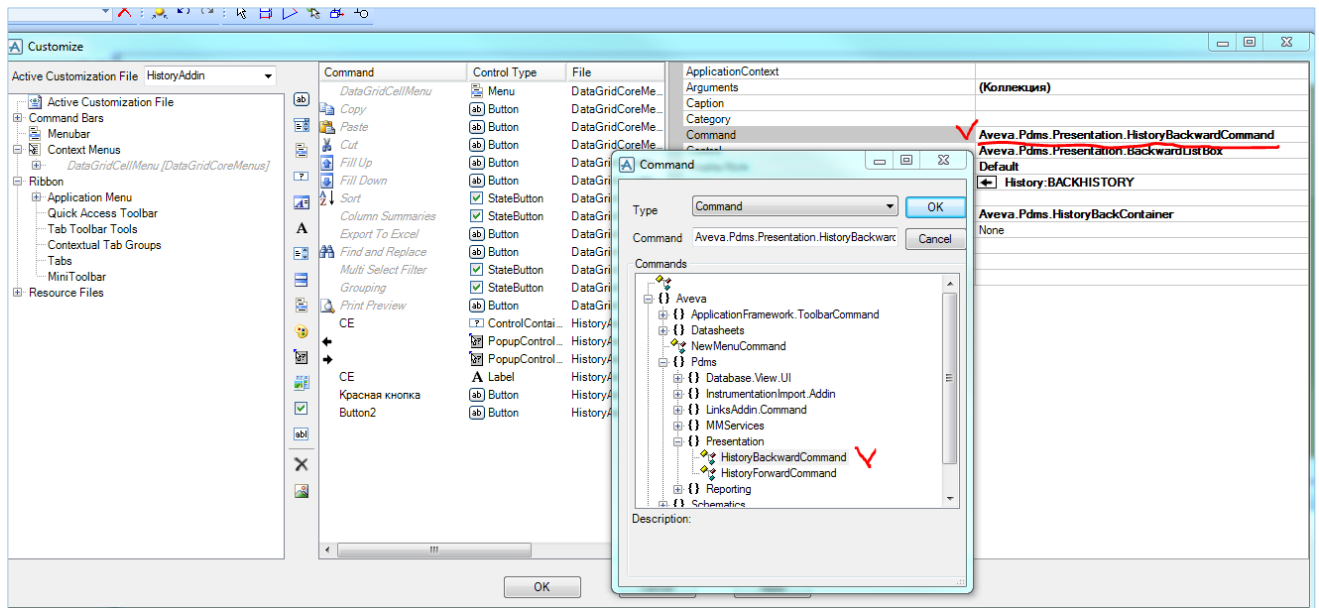
097. How to execute .Net command

```

import 'PDMSCommands'
handle (1000,0)
endhandle
using namespace 'Aveva.Pdms.Presentation.PDMSCommands'
!commandManager = object PMLNETCOMMANDMANAGER()
!commandManager.executeCommand('Aveva.Pdms.Presentation.HistoryBackwardCommand')

```

You can see the list of **build-in .Net commands** if you open Customisation form (right click in toolbar and select Customization...) and selecting any Command look at its Command attribute:



098. How to find out colour and translucency of element, which were set by Autocolour, or selected manually

```
--get Drawlist
!getDrawlist = !!gphDrawlists.drawlist(!!GPH3DDESIGN1.VIEW)
```

```
--query colour. item_id is Dbref
!getColour = !getDrawlist.colour(item_id)
```

If !getColour returns 0, then the element is not added to view

NB. For getting translucency you can use the following code:

```
!getTranslucency = !getDrawlist.translucency(item_id)
```

099. How to get Drawlist and all current 3D-views

```
q var !gphDrawlists.drawlist(1).Members() – array of members of Drawlist
q var !gphviews.views – array of names of opened 3D-views (name of View-gadget of 3D view window)
```

100. How quickly to add elements to view, which are referenced by given element

As it is known there are a lot of elements, which can reference to each other, for example:

- Nozzle (NOZZ) references to connected branch (BRAN)
- Attachment element ATTA can reference to hanger (HANG)
- Etc.

So how we can add to view elements, which are referenced by given element?

Example:

We have a branch consisting of several hangers, which are not added to a view. How to add them to a view?

Solution: it is required to analyze Cref attribute – basic reference attribute of ATTA, which is used for referencing on hangers (HANG)

```
--first of all find all Cref of all ATTA
VAR !getCrefs EVAL ( NAME OF CREF ) FOR ALL ATTA WITH ( CREF NEQ Nulref ) FOR CE
--and add all founded elements by one command
ADD ALL FROM !getCref
```