User guide

TopoL for Windows

New functions of ToPas and DDE

TopoL Software s.r.o.

Information in this document is subject to change without any prior notice. While every precaution has been taken in the preparation of this manual, authors and TopoL Software s.r.o. company assume no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained herein.
No part of this manual shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, desktop publishing, recording or otherwise, for any purpose without written permission from TopoL Software s.r.o. company.

© 2000 TopoL software s.r.o.. All rights reserved.

Microsoft and MS-DOS are registered trademarks and Windows is trademark of Microsoft Corporation.
All other product and brand names are trademarks, registered trademarks, or service marks of their respective owners.

Content

NEW FUNCTIONS OF TOPAS AND DDE 5

OBJECT ATTRIBUTES 6

REPAINT 6
PAINT(COLOR, SYMBOL: LONGINT) 6
XCOORD, YCOORD, ZCOORD: LIST OF DOUBLE 6
XLIST, YLIST: LIST OF DOUBLE; 6
ACCURACY: LONGINT 6
SETACCURACY(ACC: LONGINT): LONGINT 6

POINT ATTRIBUTES 7

SETANGLE(ANG: DOUBLE): DOUBLE 7

POLYLINES ATTRIBUTES 8

TOPOLOGICAL: BOOLEAN 8
NUMFREETAILS 8
FREETAILFIRST 8
FREETAILLAST 8

AREA ATTRIBUTES 9

NUMEDGES: LONGINT 9
AREEDGE(ORDER_NUM_LIN: LONGINT): LONGINT 9

SPATIAL FUNCTIONS 10

CONTAINSXY(OBJECTTYPE: CHAR; XLIST, YLIST: LIST OF DOUBLE): BOOLEAN 10
ECONTAINSXY(OBJECTTYPE: CHAR; XLIST, YLIST: LIST OF DOUBLE): BOOLEAN 10
CONTAINEDXY(OBJECTTYPE: CHAR; XLIST, YLIST: LIST OF DOUBLE): BOOLEAN 10
ECONTAINEDXY(OBJECTTYPE: CHAR; XLIST, YLIST: LIST OF DOUBLE): BOOLEAN 10
MEETXY(OBJECTTYPE: CHAR; XLIST, YLIST: LIST OF DOUBLE): BOOLEAN 10
OVERLAPXY(OBJECTTYPE: CHAR; XLIST, YLIST: LIST OF DOUBLE): BOOLEAN 11
EQUALXY(OBJECTTYPE: CHAR; XLIST, YLIST: LIST OF DOUBLE): BOOLEAN 11
TOUCHXY(OBJECTTYPE: CHAR; XLIST, YLIST: LIST OF DOUBLE): BOOLEAN 11

OTHER FUNCTIONS 12

FUNCTION CALCVALUE(TEXT_FILE_NAME: STRING): STRING ... 12
DESCRIPTION, PARAMETER 12
ABORTINFO(STR: STRING) 12
AREUNIT: LONGINT 12
AREUNITSTR: STRING 12

DDE FUNCTIONS 13

FUNCTION DDEINITIATE(SERVER, TOPIC: STRING): LONGINT; .. 13
FUNCTION DDEREQUEST(CHANNEL: LONGINT; ITEM: STRING): STRING; 13
FUNCTION DDEPOKE(CHANNEL: LONGINT; ITEM: STRING; DATA: STRING): BOOLEAN; 13
FUNCTION DDEEXECUTE(CHANNEL: LONGINT; MACRO: STRING; WAITFORTOPOL: BOOLEAN): BOOLEAN; 13
FUNCTION DDETERMINATE(CHANNEL: LONGINT): BOOLEAN; ... 13
FUNCTION DDESELFCONNECT(START: BOOLEAN): BOOLEAN .. 13

RUNNING OTHER FUNCTIONS 14

FUNCTION ISTASKRUNNING(NAME: STRING): BOOLEAN; 14
FUNCTION TASKEXECUTE(CMDLINE: STRING; CMDSHOW: LONGINT): BOOLEAN; 14

New functions of ToPas and DDE

New function of ToPas and DDE where implemented in version 5 of TopoL. There are these new functions of ToPas :

· New key words calcvalue (it enables to define the procedure), description and parameter (this enable to set up the parameter in displayed dialog during the evaluation of DIN files. For example different parameter for each selected area)

· New attributes for displaying of objects Repaint and Paint. They enable to redraw the object during the evaluation of DIN files.

· New functions XCoord, YCoord and ZCoord which returns the list of coordinates of any graphical object.

· New functions for spatial analyses ContainsXY, EContainsXY, ContainedXY, EContainedXY, MeetXY, OverlapXY, EqualXY, TouchXY.

· New basic attributes of areas NumEdges, AreEdge.

· Additional attributes and functions of points SetAngle, SetAccuracy, Accuracy.

· Additional attributes and functions of polylines Topological, NumFreeTails, FreeTailFirst, FreeTailLast, SetAccuracy, Accuracy.

· The user can connect any other Windows application using DDE (DDEInitiate, DDERequest, DDEPoke, DDEExecute, DDETerminate, DDESelfConnect)

· The user run any other application (IsTaskRunning, TaskExecute).

· The user can change the title of the abort dialog which can be displayed during the evaluation (AbortInfo).

Object attributes

RePaint

- redraws the object
- example

[p,1]RePaint

Paint(color, symbol: longInt)

- paints the object by specified symbol and color
- example

[p,1]Paint(255, 1)

XCoord, YCoord, ZCoord: list of double

- return the list of coordinates of any graphical object
- example

var

XList, Ylist: list of double;

begin

  Xlist := [l,1]Xcoord;

  Ylist := [l,1]Ycoord;

end.

Accuracy: longInt

- returns the accuracy attribute within interval 0..9
- defined for points and polylines

SetAccuracy(acc: longInt): longInt

- sets up the accuracy attribute within interval 0..9 and returns this value
- defined for points and polylines

Point attributes

SetAngle(ang: double): double

- sets up the angle attribute and returns this value

Polylines attributes

Topological: boolean

- return if the polyline is topological or not

NumFreeTails

- returns the number of free tiles of the polyline

FreeTailFirst

- returns if the first point of the polyline is free or not

FreeTailLast

- returns if the last point of the polyline is free or not

Area attributes

NumEdges: longInt

- returns the number of edges creating the area

AreEdge(order_num_lin: longint): longInt

- returns the number of polyline which create the area
- parameter order_num_lin defines the number of the edge

Spatial functions

ContainsXY(ObjectType: char; Xlist, Ylist: list of double): boolean

- returnes true if the object contains object the specified by type and list of coordinates

- it is not defined for points, the other object can be point, in that case the list of coordinates can be only one value of double type

- for polylines the parameter ObjectType can not be P for areas

- example

```
var
  XList, Ylist: list of double;
  Contai: boolean;
begin
  Xlist := [l,1]Xcoord;
  Ylist := [l,1]Ycoord;

  Contai := [p,1]contains(l, Xlist, Ylist)
end.
```

EContainsXY(ObjectType: char; Xlist, Ylist: list of double): boolean

- returnes true if the object entirely contains the object specified by type and list of coordinates

- it is not defined for points, the other object can be point, in that case the list of coordinates can be only one value of double type

- for polylines the parameter ObjectType can not be P for areas

ContainedXY(ObjectType: char; Xlist, Ylist: list of double): boolean

- returnes true if the object is contained in the object specified by type and list of coordinates

- the parameter ObjectType can not be B for points

- for areas the parameter ObjectType can not be L for polylines

EcontainedXY(ObjectType: char; Xlist, Ylist: list of double): boolean

- returnes true if the object is entirely contained in the object specified by type and list of coordinates

- the parameter ObjectType can not be B for points

- for areas the parameter ObjectType can not be L for polylines

MeetXY(ObjectType: char; Xlist, Ylist: list of double): boolean

- returnes true if the object is touched by the object specified by type and list of coordinates

OverlapXY(ObjectType: char; Xlist, Ylist: list of double): boolean

- returnes true if the object and the object specified by type and list of coordinates overlap each other

EqualXY(ObjectType: char; Xlist, Ylist: list of double): boolean

- returnes true if the object is equal to the object specified by type and list of coordinates
- the parameter ObjectType has to be of the same type as the object

TouchXY(ObjectType: char; Xlist, Ylist: list of double): boolean

- returns true if the object and the object specified by type and list of coordinates have at least one common point

Other functions

function CalcValue(text_file_name: string): string

- enables to define the procedure

- the procedure has to be defined in ASCII file where the ResultValue variable is defined and used. This is the example of that macro text file.

```
valuemacro Popis;


var
  ResultValue: String;
begin
  ResultValue := @ParCis:C;
  if @Podlomeni <> 0 then
    ResultValue := ResultValue + '/' + @Podlomeni:C;
end.
```

Description, parameter

- this enable to set up the parameter in displayed dialog during the evaluation of DIN files. For example different parameter for each selected area

- this is the example of declaration part of the DIN file

```
parameter
koef : double = 1.0 description ‘Coefitient ‘;
```

AbortInfo(str: string)

- changes the title of the abort dialog which can be displayed during the evaluation of DIN file

AreUnit: longint

- returns the unit of the area size in m2

- can be 10000 or 1

AreUnitStr: string

- returns the unit of the area size

- can be ha or m2

DDE functions

function DDEInitiate(server, topic: string): longInt;

- returns the id of the channel or the error code

function DDERequest(channel: longInt; item: string): string;

- returns the string with the data about the item

function DDEPoke(channel: longInt; item: string; data: string): boolean;

- sets up the data in string for item
- returns the boolean variable if the function was successful

function DDEExecute(channel: longInt; macro: string; WaitForTopol: boolean): boolean;

- executes the macro
- boolean WaitForTopol variable means that if the system is connected to itself the execution of the ToPas program waits for the end of the execution of the DDE macro command
- returns the boolean variable if the function was successful

function DDETerminate(channel: longInt): boolean;

- terminates the channel
- returns the boolean variable if the function was successful

function DDESelfConnect(start: boolean): boolean

- It is better to use this function if one uses some DDE TopoL functions which changes the data. In that case one has to call this function with parameter TRUE before opening of the channel. After that one should not use any function to access the internal or database attributes. They can be again accessed after this function is called with parameter FALSE which should be called after terminating of the channel.
- returns the boolean variable if the function was successful

Running other functions

function IsTaskRunning(name: string): boolean;

- returns the boolean variable if the application name is running or not
- name can be maximum 8 characters of length

function TaskExecute(cmdline: string; cmdshow: longint): boolean;

- returns the boolean variable if the application was executed
- parameters cmdline and cmdshow corresponds to the parameters of Windows API function WinExec
- cmdshow can be from the range 0..9

0 sw_Hide – Hides the window and passes activation to another window

1 sw_ShowNormal – Activates and displays a window. If the window is minimized or maximized, Windows restores it to its original size and position (same as SW_RESTORE).

2 sw_ShowMinimized – Activates a window and displays it as an icon.

3 sw_ShowMaximized – Activates a window and displays it as a maximized window.

4 sw_ShowNoActive – Displays a window in its most recent size and position. The window that is currently active remains active.

5 sw_Show – Activates a window and displays it in its current size and position.

6 sw_Minimize – Minimizes the specified window and activates the top-level window in the system's list.

7 sw_ShowMinNoActive – Displays a window as an icon. The window that is currently active remains active.

8 sw_ShowNA – Displays a window in its current state. The window that is currently active remains active.

9 sw_Restore – Activates and displays a window. If the window is minimized or maximized, Windows restores it to its original size and position (same as SW_SHOWNORMAL).