Types and constants for ADA programming language

```
--****************************************************************************
--                                          (C) IpeSoft s.r.o. (Ltd.) ZILINA
--   PROJECT :  D2000
--   FILE    :  Imp_def.ads
--
--   DESCRIPTION : types and constants for import of graphic formats
--
--****************************************************************************
with system; use system;

package Imp_Def is

  --****************************************************************************
  -- Constants of gr.object's types
  --****************************************************************************
  cLine    : constant :=  0;  -- Line
  cPLine   : constant :=  1;  -- Multiline
  cDLine   : constant :=  2;  -- Disjointed multiline
  cArc     : constant :=  3;  -- Arc
  c3Arc    : constant :=  4;  -- 3-point arc
  cBox     : constant :=  5;  -- Rectangle
  cPAngle  : constant :=  6;  -- Polygon
  cCircle  : constant :=  7;  -- Circle
  cPiArc   : constant :=  8;  -- Circle sector
  cElipse  : constant := 11;  -- Ellipse
  cText    : constant := 12;  -- Text
  cGroup   : constant := 32;  -- Group of objects

  -- extra types
  cLineCombined   : constant := -1; -- it combines more lines with the same attributes to one object
                                    -- either PolyLine or PolyLineDisjoint
  cPLineAdd       : constant := -2; -- it adds a set of parameters to the Polyline
                                    -- (if it has not been created, create a new one)

  cFontStyle      : constant := -3; -- it creates a text style + automatic creating

                                    -- when creating some texts - since the ver. 5.00
                                    -- (one style only can correspond with the given name)

  --****************************************************************************
  -- the constants that define the permitted types for some parameters
  ----------------------------------------------------------------------------
  -- label in comments
  -- (*) - default parameter
  --****************************************************************************

  --****************************************************************************
  -- color definition, implicitly the colors from logical palette which is used during import
  --****************************************************************************
  type TColorArr  is ARRAY (0.. 15) OF integer;  -- bright..dark


  -- color base
  CLR_BASE_WHITE   : constant :=  0;  W : constant := CLR_BASE_WHITE;
  CLR_BASE_YELLOW  : constant := 16;  Y : constant := CLR_BASE_YELLOW;
  CLR_BASE_CYAN    : constant := 32;  C : constant := CLR_BASE_CYAN;
  CLR_BASE_GREEN   : constant := 48;  G : constant := CLR_BASE_GREEN;
  CLR_BASE_RED     : constant := 64;  R : constant := CLR_BASE_RED;
  CLR_BASE_PINK    : constant := 80;  P : constant := CLR_BASE_PINK;
  CLR_BASE_BLUE    : constant := 96;  B : constant := CLR_BASE_BLUE;

  -- color index
  CLR_WHITE    : constant TColorArr := (W+0,W+1,W+2,W+3,W+4,W+5,6,W+7,W+8,W+9,W+10,W+11,W+12,W+13,W+14,W+15);
  CLR_YELLOW   : constant TColorArr := (Y+0,Y+1,Y+2,Y+3,Y+4,Y+5,6,Y+7,Y+8,Y+9,Y+10,Y+11,Y+12,Y+13,Y+14,Y+15);
  CLR_CYAN     : constant TColorArr := (C+0,C+1,C+2,C+3,C+4,C+5,6,C+7,C+8,C+9,C+10,C+11,C+12,C+13,C+14,C+15);
  CLR_GREEN    : constant TColorArr := (G+0,G+1,G+2,G+3,G+4,G+5,6,G+7,G+8,G+9,G+10,G+11,G+12,G+13,G+14,G+15);
  CLR_RED      : constant TColorArr := (W+0,W+1,W+2,W+3,W+4,W+5,6,W+7,W+8,W+9,W+10,W+11,W+12,W+13,W+14,W+15);
  CLR_PINK     : constant TColorArr := (P+0,P+1,P+2,P+3,P+4,P+5,6,P+7,P+8,P+9,P+10,P+11,P+12,P+13,P+14,P+15);
  CLR_BLUE     : constant TColorArr := (B+0,B+1,B+2,B+3,B+4,B+5,6,B+7,B+8,B+9,B+10,B+11,B+12,B+13,B+14,B+15);

  -- some colors
  i_CLR_WHITE         : constant integer :=  0;
  i_CLR_PALEGRAYLIGHT : constant integer :=  2;
  i_CLR_PALEGRAY      : constant integer :=  4;
  i_CLR_DARKGRAY      : constant integer := 10;
  i_CLR_DARKGRAYDARK  : constant integer := 12;
  i_CLR_BLACK         : constant integer := 15;
```

```
i_CLR_YELLOW    : constant integer := 22;  i_CLR_DARKYELLOW: constant integer := 26;
i_CLR_CYAN      : constant integer := 38;  i_CLR_DARKCYAN  : constant integer := 42;
i_CLR_GREEN     : constant integer := 54;  i_CLR_DARKGREEN : constant integer := 58;
i_CLR_RED       : constant integer := 70;  i_CLR_DARKRED   : constant integer := 74;
i_CLR_PINK      : constant integer := 86;  i_CLR_DARKPINK  : constant integer := 90;
i_CLR_BLUE      : constant integer :=102;  i_CLR_DARKBLUE  : constant integer :=106;


--*****************************************************************************
-- line style
-- lineStyles - type TPenStyle is (Solid, Alternate, Dash, Dot, DashDot, DashDotDot, Invisible);
--*****************************************************************************
tLS_Solid       : constant := 0;   -- -----------  (*)
tLS_Alternate   : constant := 1;   -- . . . . . . - only thin, do not use
tLS_Dash        : constant := 2;   -- - - - - - -
tLS_Dot         : constant := 3;   -- . . . . . .
tLS_DashDot     : constant := 4;   -- - . - . - .
tLS_DashDotDot  : constant := 5;   -- - . . - . .
tLS_Invisible   : constant := 6;   -- unsupported

--*****************************************************************************
-- line end
-- lineEnd - type TLineEnd  is (Flat,  Square, Round);
--*****************************************************************************
tLE_Flat        : constant := 0;   -- flat (*)
tLE_Square      : constant := 1;   -- square
tLE_Round       : constant := 2;   -- rounded

--*****************************************************************************
-- line join
-- lineJoin - type TLineJoin is (Bevel, Round,  Miter);
--*****************************************************************************
tLJ_Bevel       : constant := 0;   -- ending at the endpoint (*)
tLJ_Round       : constant := 1;   -- rounded ending behind the endpoint
tLJ_Miter       : constant := 2;   -- square ending behind the endpoint


--*****************************************************************************
-- the patterns in Windows
-- brushStyle - type TBrushStyle is (Solid, Hollow, BDiagonal, Cross, DiagCross,
--                  FDiagonal,  Horizontal, Vertical);
--*****************************************************************************
tBS_Solid       : constant := 0;   -- solid pattern
tBS_Hollow      : constant := 1;   -- hollow pattern (*)
tBS_BDiagonal   : constant := 2;
tBS_Cross       : constant := 3;
tBS_DiagCross   : constant := 4;
tBS_FDiagonal   : constant := 5;
tBS_CHorizontal : constant := 6;
tBS_Vertical    : constant := 7;


--*****************************************************************************
-- a text position
-- cTextPos - type tTextPos   is (tpAtPos,tpInBox,tpIntoBox);
--*****************************************************************************
tTP_ATPOS       : constant := 0;   -- defined position of the text (*)
tTP_INBLOCK     : constant := 1;   -- text placed in a rectangle
tTP_INTOBLOCK   : constant := 2;   -- text fills the rectangle

--*****************************************************************************
-- horizontal alignment of text in rectangle
-- cTextCenterH -horizontal center
--*****************************************************************************
tTHC_LEFT       : constant := 0;   -- left
tTHC_MIDDLE     : constant := 1;   -- center (*)
tTHC_RIGHT      : constant := 2;   -- right

--*****************************************************************************
-- vertical alignment of text in rectangle
-- cTextCenterV - vertical center
--*****************************************************************************
tTVC_TOP        : constant := 0;   -- top
tTVC_MIDDLE     : constant := 1;   -- middle (*)
tTVC_BOTTOM     : constant := 2;   -- bottom
```

```
  --*****************************************************************************
  -- parameters of graphic objects - param's type constants
  -----------------------------------------------------------------------------
  -- comments for the implementation
  --   1. the parameters must be placed in such procedures as they are mentioned in the comment behind the
parameter,
  --       this ensures the parameters will be accepted
  --   2. some of the parameters will not be executed
  --   3. you can set maximum 1000 position points, however version V4.5 accepts at the most first 1+30
  --      if there are more points for polyline, more objects of this type are generated
  --   4. a color index for import index to the default color palette
  --   5. RGB color - from the version V5.0, the colors fill the table with 128 elements; the object Palette will
be created from this table,
  --      the colors in object will be saved in the form of indexes into this table
  --*****************************************************************************

  -----------------------------------------------------------------------------
  -- positions and sizes
  -----------------------------------------------------------------------------
  cPosXY               : constant :=1;  -- 2 * int/float - point position
  cPosDXY              : constant :=2;  -- 2 * int/float - a distance from the previous point

  -----------------------------------------------------------------------------
  -- line params
  -----------------------------------------------------------------------------
  cLineColorRGB        : constant :=10;  -- int  - RGB color of object - from the version V5.0
  cLineColorIdx        : constant :=11;  -- int  - color index in the local palette
  cLineStyle           : constant :=12;  -- int  - line style - lineStyles
  cLineWidth           : constant :=13;  -- int  - line width 1..
  cLineEnd             : constant :=14;  -- int  - end of thick line - lineEnd
  cLineJoin            : constant :=15;  -- int  - polyline join - lineJoin

  -----------------------------------------------------------------------------
  -- fill params
  -----------------------------------------------------------------------------
  cFillColorRGB        : constant :=30;  -- int - RGB color of object - from the version V5.0
  cFillColorIdx        : constant :=31;  -- int - color index in the local palette
  cFillPattern         : constant :=32;  -- int  - line style - brushStyle

  -----------------------------------------------------------------------------
  -- circles
  -----------------------------------------------------------------------------
  cCircleRadial        : constant :=50;  -- int/float - radius
  --
  cCircleAngleDegStart : constant :=56;  -- int/float - degree
  cCircleAngleDegEnd   : constant :=57;  -- int/float - degree
  cCircleAngleDegSize  : constant :=58;  -- int/float - degree
  --
  cCircleAngleRadStart : constant :=59;  -- int/float - radiant
  cCircleAngleRadEnd   : constant :=60;  -- int/float - radiant
  cCircleAngleRadSize  : constant :=61;  -- int/float - degree

  -----------------------------------------------------------------------------
  -- text params
  -----------------------------------------------------------------------------
  cTextText            : constant :=70;  -- text
  cTextColorRGB        : constant :=71;  -- int - RGB color of object - from the version V5.0
  cTextColorIdx        : constant :=72;  -- int - color index in the local palette
  --
  cTextPos             : constant :=73;  -- int - position in a rectangle, tTP_ATPOS,..
  cTextCenterH         : constant :=76;  -- int - horizontal alignment in rectangle
  cTextCenterV         : constant :=77;  -- int - vertical alignment in rectangle

  --*****************************************************************************
  -- creating of the text style
  -- text style is an object with parameter CFont or when creating a text, there are these conditions
  --  * font will not be created unless all the parameters are set
  --  * if cFontStyleName style exists, the new one will not be created, the existing one will be used
  --  * if the text style is developed when creating the text, it will be used
  --  * if cFontStyleName is set when creating the text, the text will be drawn by this style (if it exists)
  --*****************************************************************************
  cFontStyleName       : constant :=200;  --  string  - font style name
  cFontName            : constant :=201;  --  string  - font name
  cFontSize            : constant :=202;  --  Integer - font size
  cFontBold            : constant :=203;  --  Boolean - attribute for bold
  cFontItalic          : constant :=204;  --  Boolean - attribute for italic
  cFontUnderline       : constant :=205;  --  Boolean - attribute for underline
  cFontStrikeOut       : constant :=206;  --  Boolean - attribute for strikeout
  cFontCharSet         : constant :=207;  --  Integer - character set (Western, Central European)
```

```
  --****************************************************************************
  -- types of obj. actions
  --****************************************************************************
  type tObjAction  is (closeFigure, -- close polyline, a change to polygon
                       closeObject, -- close object
                       closeGroup,  -- close object and group
                       closeAll);   -- close object and all groups
                                    -- called internal, after ending the import

  --****************************************************************************
  -- access to functions - definitions
  --****************************************************************************

  type tCreateObj    is access procedure (objType : integer);
  type tObjAtion     is access procedure (action  : tObjAction := closeObject);


  type tset_string   is access procedure (param   : integer; val :string);
  type tset_boolean  is access procedure (param   : integer; val :boolean);
  type tset_integer  is access procedure (param   : integer; val :integer);
  type tset_float    is access procedure (param   : integer; val :long_float);
  type tset_integer2 is access procedure (param   : integer; val1,val2 : integer);
  type tset_float2   is access procedure (param   : integer; val1,val2 : long_float);

  -- info text that occurs during import
  type tShowInfo     is access procedure (text : string);

  --****************************************************************************
  -- access to functions
  --****************************************************************************
  createObj    : tCreateObj    := null;
  objAction    : tObjAtion     := null;


  set_string   : tset_string   := null;
  set_boolean  : tset_boolean  := null;
  set_integer  : tset_integer  := null;
  set_float    : tset_float    := null;
  set_integer2 : tset_integer2 := null;
  set_float2   : tset_float2   := null;

  -- info text that occurs during import
  ShowInfo     : tShowInfo     := null;

  --****************************************************************************
  -- access to import functions
  --****************************************************************************

  pShowInfo     : constant := 0;
  pcreateObj    : constant := 1;
  pobjAction    : constant := 2;
  pset_string   : constant := 3;
  pset_boolean  : constant := 4;
  pset_integer  : constant := 5;
  pset_float    : constant := 6;
  pset_integer2 : constant := 7;
  pset_float2   : constant := 8;

  -- pointer to 1.char of null terminating string
  maxResStr     : constant := 10000;

  -- import name and file type
  type tGetFileType     is access procedure (description,extension: out address);

  -- initialization of call-back procedures
  type tImportConnect   is access procedure (procType:integer; procAddr : address);


  -- load autocad dxf file
  type tImportFile      is access procedure (FileName:address;x,y:INTEGER;resStr:address);

end Imp_Def;
--****************************************************************************

-------------------------------------------------------------------------------
-- Revisions History --
-------------------------------------------------------------------------------
--  0.00     28.02.01 - creating of module
-------------------------------------------------------------------------------
```

Related pages:

Import of vector formats into pictures of D2000 System