OCAD 11 File Format

From OCAD 11 Wiki - English

Version: 2012-02-09

Contents

- 1 General
  - 1.1 Data types used
- 2 File Header
- 3 Symbols
  - 3.1 Base Symbol
  - 3.2 Point Symbol
  - 3.3 Line Symbol
  - 3.4 Area Symbol
  - 3.5 Text Symbol
  - 3.6 Line Text Symbol
  - 3.7 Rectangle Symbol
- 4 Objects
  - 4.1 Object Index Block
  - 4.2 OCAD Objet Index
  - 4.3 OCAD Object
- 5 Parameter Strings

General

This is a description of the file format of OCAD 11 files.

Be aware that this is an internal format and may change in future versions.

Data types used

OCAD is written in 32-bit Delphi and this description uses the names for the data types as they appear in Delphi. However the same data types are available in other development systems like C++.

Integer	32-bit signed integer
SmallInt	16-bit signed integer
Word	16-bit unsigned integer
WordBool	16-bit boolean
String[x]	Widestring. The first byte contains the number of characters followed by the characters. The string is not zero-terminated. The maximum number of characters is x. It occupies x + 1 bytes in the file.
Double	64-bit floating point number
TDPoly	A special data type (64-bit) used for all coordinates and text. It is defined as TDPoly = record x, y: integer; end; The lowest 8 Bits are used to mark special points: Marks for the x-coordinate: 1: this point is the first bezier curve point 2: this point is the second bezier curve point 4: for double lines: there is no left line between this point and the next point 8: this point is a area border line or a virtual line gap Marks for y-coordinate: 1: this point is a corner point 2: this point is the first point of a hole in an area 4: for double lines: there is no right line between this point and the next point 8: this point is a dash point The upper 24 bits contain the coordinate value measured in units of 0.01 mm.

Note: all file positions are in bytes starting from the beginning of the file.

File Header

OCAD files start with a file header.

```
TFileHeader = record              // size = 48 Byte
  OCADMark: SmallInt;             // 3245 (hex 0cad)
  FileType: Byte;                 // file type (0: normal map, 1: course setting project, 8: file is
  FileStatus: Byte;               // not used
  Version: SmallInt;              // 11
  Subversion: Byte;               // number of subversion (0 for 11.0, 1 for 11.1 etc.)
  SubSubversion: Byte;            // number of subsubversion (0 for 11.0.0, 1 for 11.0.1)
  FirstSymbolIndexBlk: integer;   // file position of the first symbol index block
  ObjectIndexBlock: integer;      // file position of the object index block -> TObjectIndexBlock
  OfflineSyncSerial: integer;     // serialNumber for offline work in Server mode
  Res1: integer;                  // not used
  Res2: longint;                  // not used
  Res3: longint;                  // not used
  FirstStringIndexBlk: longint;   // file position of the first string index block
```

```
  FileNamePos: integer;           // file position of the file name, used for temporary files only
  FileNameSize: integer;          // size of the file name, used for temporary files only
  Res4: integer;                  // not used
```

Symbols

Each Symbol Index Block contains the position of the next Symbol Index Block and the file position of 256 symbols.

```
TSymbolIndexBlock= record        // Size: 1028 Bytes
  NextSymbolIndexBlock: integer;
  SymbolPosition: array[0..255] of integer;
end;
```

Base Symbol

The different types of symbols are defined in different structures. There is an abstract type TBaseSym, which contains the fields common to all symbols types. It is used for programming reasons, but does not exist in real OCAD files.

```
TBaseSym = packed record
  Size: integer;                // Size of the symbol in bytes. This  depends on the type. Coordinate
  SymNum: integer;              // Symbol number. This is 1000 times the symbol number.
                                // for example:
                                //   203.145 is stored as 203145
  Otp: byte;                    // Object type
                                //   1: Point symbol
                                //   2: Line symbol or Line text symbol
                                //   3: Area symbol
                                //   4: Text symbol
                                //   6: Line text symbol
                                //   7: Rectangle symbol
  Flags: byte;                  // 1: rotatable symbol (not oriented to north)
                                // 4: belongs to favorites
  Selected: boolean;            // Symbol is selected in the symbol box
  Status: byte;                 // Status of the symbol
                                //   0: Normal
                                //   1: Protected
                                //   2: Hidden
                                // AND 16: selected
  PreferredDrawingTool: byte;   // Preferred drawing tool
                                //   0: off
                                //   1: Curve mode
                                //   2: Ellipse mode
                                //   3: Circle mode
                                //   4: Rectangular line mode
                                //   5: Rectangular area mode
                                //   6: Straight line mode
                                //   7: Freehand mode
                                //   8: Numeric mode
                                //   9: Stairway mode
  CsMode: byte;                 // Course setting mode
                                //   0: not used for course setting
                                //   1: course symbol
                                //   2: control description symbol
  CsObjType: byte;              // Course setting object type
                                //   0: Start symbol (Point symbol)
                                //   1: Control symbol (Point symbol)
                                //   2: Finish symbol (Point symbol)
                                //   3: Marked route (Line symbol)
                                //   4: Control description symbol (Point symbol)
                                //   5: Course title (Text symbol)
                                //   6: Start number (Text symbol)
                                //   7: Relay variant (Text symbol)
                                //   8: Text block for control description (Text symbol)
  CsCdFlags: byte;              // Course setting control description flags
                                //   a combination of the flags
                                //   64: available in column B
```

```
                                //   32: available in column C
                                //   16: available in column D
                                //   8: available in column E
                                //   4: available in column F
                                //   2: available in column G
                                //   1: available in column H
  Extent: integer;              // Extent how much the rendered symbols can reach outside the coordin
                                // this symbol. For a point object it tells how far away from the coo
                                // anything of the point symbol can appear
  FilePos: integer;             // Used internally. Value in the file is not defined.
  notUsed1: Byte;
  notUsed2: Byte;
  nColors: SmallInt;            // Number of colors of the symbol max. 14, -1: the number of colors i
  Colors: array[0..13] of SmallInt; // Colors of the symbol
  Description: array[0..63] of char;// Description text
  IconBits: array[0..483] of byte;  // Each byte represents a pixel of the icon in a 256 color palett
  SymbolTreeGroup: Array[0..63] of Word; // Group ID in the symbol tree, max 64 symbol groups
```

Point Symbol

Point symbols are stored with this structure:

```
TPointSym = packed record
  Size: integer;                        // see TBaseSym
  SymNum: integer;                      // see TBaseSym
  Otp: byte;                            // 1
  Flags: byte;                          // see TBaseSym
  Selected: boolean;                    // see TBaseSym
  Status: byte;                         // see TBaseSym
  PreferredDrawingTool: byte;           // see TBaseSym
  CsMode: byte;                         // see TBaseSym
  CsObjType: byte;                      // see TBaseSym
  CsCDFlags: byte;                      // see TBaseSym
  Extent: integer;                      // see TBaseSym
  FilePos: integer;                     // see TBaseSym
  Group: SmallInt;                      // see TBaseSym
  nColors: SmallInt;                    // see TBaseSym
  Colors: array[0..13] of SmallInt;     // see TBaseSym
  Description: array[0..63] of char;    // see TBaseSym
  IconBits: array[0..483] of byte;      // see TBaseSym
  SymbolTreeGroup: Array[0..63] of Word; // see TBaseSym
  DataSize: word;                       // number of coordinates (each 8 bytes) which follow this st
                                        // counts as 2 Coordinates (16 bytes)
  Reserved: SmallInt;
end;
```

After this structure follow the symbol elements which build that Point symbol. These symbol elements are stored in the following structure. Note that these symbol elements are stored in a different way than ordinary map objects.

```
TSymElt = record
  stType: SmallInt;                 // type of the symbol element
                                    //   1: line
                                    //   2: area
                                    //   3: circle
                                    //   4: dot (filled circle)
  stFlags: word;                    // Flags
                                    //   1: line with round ends
  stColor: SmallInt;                // color of the object. This is the number which appears in
                                    // the colors dialog box
  stLineWidth: SmallInt;            // line width for lines and circles unit 0.01 mm
  stDiameter: SmallInt;             // Diameter for circles and dots. The line width is included
                                    // one time in this dimension for circles.
  stnPoly: SmallInt;                // number of coordinates
  stRes1: SmallInt;                 // Not used
  stRes2: SmallInt;                 // Not used
  stPoly: array[0..32767] of TPoint;  // coordinates of the symbol element
end;
```

If there are several objects, they just follow each other (only the coordinates used are stored). To determine the number of objects the DataSize variable must be used.

Line Symbol

Line symbols are stored in the following structure. In the explanation the terms used in the Line Symbol dialog box are shown.

```
TLineSym = packed record
  Size: integer;                          // see TBaseSym
  SymNum: integer;                        // see TBaseSym
  Otp: byte;                              // 2
  Flags: byte;                            // see TBaseSym
  Selected: boolean;                      // see TBaseSym
  Status: byte;                           // see TBaseSym
  PreferredDrawingTool: byte;             // see TBaseSym
  CsMode: byte;                           // see TBaseSym
  CsObjType: byte;                        // see TBaseSym
  CsCDFlags: byte;                        // see TBaseSym
  Extent: integer;                        // see TBaseSym
  FilePos: integer;                       // see TBaseSym
  Group: SmallInt;                        // see TBaseSym
  nColors: SmallInt;                      // see TBaseSym
  Colors: array[0..13] of SmallInt;       // see TBaseSym
  Description: array[0..63] of char;      // see TBaseSym
  IconBits: array[0..483] of byte;        // see TBaseSym
  SymbolTreeGroup: Array[0..63] of Word; // see TBaseSym
  LineColor: SmallInt;                    // Line color
  LineWidth: SmallInt;                    // Line width
  LineStyle: SmallInt;                    // Line style
                                          //   0: bevel joins/flat caps
                                          //   1: round joins/round caps
                                          //   4: miter joins/flat caps
  DistFromStart: SmallInt;                // Distance from start
  DistToEnd: SmallInt;                    // Distance to the end
  MainLength: SmallInt;                   // Main length a
  EndLength: SmallInt;                    // End length b
  MainGap: SmallInt;                      // Main gap C
  SecGap: SmallInt;                       // Gap D
  EndGap: SmallInt;                       // Gap E
  MinSym: SmallInt;                       // -1: at least 0 gaps/symbols
                                          //  0: at least 1 gap/symbol
                                          //  1: at least 2 gaps/symbols
                                          //  etc.
  nPrimSym: SmallInt;                     // No. of symbols
  PrimSymDist: SmallInt;                  // Distance
  DblMode: word;                          // Mode (Double line page)
  DblFlags: word;                         // Double line flags
                                          //    1: Fill color on
                                          //    2: Background color on
  DblFillColor: SmallInt;                 // Fill color
  DblLeftColor: SmallInt;                 // Left line/Color
  DblRightColor: SmallInt;                // Right line/Color
  DblWidth: SmallInt;                     // Width
  DblLeftWidth: SmallInt;                 // Left line/Line width
  DblRightWidth: SmallInt;                // Right line/Line width
  DblLength: SmallInt;                    // Dashed/Distance a
  DblGap: SmallInt;                       // Dashed/Gap
  DblBackgroundColor: SmallInt;           // Reserved
  DblRes: array[0..1] of SmallInt;        // Reserved
  DecMode: word;                          // Decrease mode
                                          //   0: off
                                          //   1: decreasing towards the end
                                          //   2: decreasing towards both ends
  DecLast: SmallInt;                      // Last symbol
  DecRes: SmallInt;                       // Reserved
  FrColor: SmallInt;                      // Color of the framing line
  FrWidth: SmallInt;                      // Line width of the framing line
  FrStyle: SmallInt;                      // Line style of the framing line
                                          //   0: bevel joins/flat caps
                                          //   1: round joins/round caps
                                          //   4: miter joins/flat caps
                                          //  PointedEnd := LineStyle and 2 > 0;
```

```
  PrimDSize: word;                      // number or coordinates (8 bytes) for the Main symbol A whi
                                        // Each symbol header counts as 2 coordinates (16 bytes).
  SecDSize: word;                       // number or coordinates (8 bytes) for the Secondary symbol
                                        // Each symbol header counts as 2 coordinates (16 bytes).
  CornerDSize: word;                    // number or coordinates (8 bytes) for the Corner symbol whi
                                        // Each symbol header counts as 2 coordinates (16 bytes).
  StartDSize: word;                     // number or coordinates (8 bytes) for the Start symbol C wh
                                        // Each symbol header counts as 2 coordinates (16 bytes).
  EndDSize: word;                       // number or coordinates (8 bytes) for the End symbol D whic
                                        // Each symbol header counts as 2 coordinates (16 bytes).
  UseSymbolFlags: Byte;                 // 1 = end symbol, 2 = start symbol, 4 = corner symbol, 8 =
  Reserved: Byte;
end;
```

Area Symbol

Area symbols are stored in the following structure. In the explanation the terms used in the Area Symbol dialog box are shown. The unit of all dimensions is 0.01 mm.

```
TAreaSym = packed record
  Size: integer;                        // see TBaseSym
  SymNum: integer;                      // see TBaseSym
  Otp: byte;                            // 3
  Flags: byte;                          // see TBaseSym
  Selected: boolean;                    // see TBaseSym
  Status: byte;                         // see TBaseSym
  PreferredDrawingTool: byte;           // see TBaseSym
  CsMode: byte;                         // see TBaseSym
  CsObjType: byte;                      // see TBaseSym
  CsCDFlags: byte;                      // see TBaseSym
  Extent: integer;                      // see TBaseSym
  FilePos: integer;                     // see TBaseSym
  Group: SmallInt;                      // see TBaseSym
  nColors: SmallInt;                    // see TBaseSym
  Colors: array[0..13] of SmallInt;     // see TBaseSym
  Description: array[0..63] of char;    // see TBaseSym
  IconBits: array[0..483] of byte;      // see TBaseSym
  SymbolTreeGroup: Array[0..63] of Word; // see TBaseSym
  BorderSym: integer;                   // Symbol for border line  activated if BorderOn is true
  FillColor: SmallInt;                  // Fill color activated if FillOn is true
  HatchMode: SmallInt;                  // Hatch mode
                                        //   0: None
                                        //   1: Single hatch
                                        //   2: Cross hatch
  HatchColor: SmallInt;                 // Color (Hatch page)
  HatchLineWidth: SmallInt;             // Line width
  HatchDist: SmallInt;                  // Distance
  HatchAngle1: SmallInt;                // Angle 1
  HatchAngle2: SmallInt;                // Angle 2
  FillOn: boolean;                      // Fill is activated
  BorderOn: boolean;                    // Border line is activated
  StructMode: SmallInt;                 // Structure
                                        //   0: None
                                        //   1: aligned rows
                                        //   2: shifted rows
  StructWidth: SmallInt;                // Width
  StructHeight: SmallInt;               // Height
  StructAngle: SmallInt;                // Angle
  StructRes: SmallInt;                  // Reserved
  DataSize: word;                       // number of coordinates (each 8 bytes) which follow this st
                                        // counts as 2 Coordinates (16 bytes)
end;
```

Text Symbol

Text symbols are stored in the following structure. In the explanation the terms used in the Text Symbol dialog box are shown. The unit of all dimensions is 0.01 mm, except for the font sizes which are measured in 0.1 typographical points.

```
TTextSym = packed record
  Size: integer;                          // see TBaseSym
  SymNum: integer;                        // see TBaseSym
  Otp: byte;                              // 4
  Flags: byte;                            // see TBaseSym
  Selected: boolean;                      // see TBaseSym
  Status: byte;                           // see TBaseSym
  PreferredDrawingTool: byte;             // see TBaseSym
  CsMode: byte;                           // see TBaseSym
  CsObjType: byte;                        // see TBaseSym
  CsCDFlags: byte;                        // see TBaseSym
  Extent: integer;                        // see TBaseSym
  FilePos: integer;                       // see TBaseSym
  Group: SmallInt;                        // see TBaseSym
  nColors: SmallInt;                      // see TBaseSym
  Colors: array[0..13] of SmallInt;       // see TBaseSym
  Description: array[0..63] of char;      // see TBaseSym
  IconBits: array[0..483] of byte;        // see TBaseSym
  SymbolTreeGroup: Array[0..63] of Word; // see TBaseSym
  FontName: string[31];                   // TrueType font
  FontColor: SmallInt;                    // Color
  FontSize: SmallInt;                     // 10 times the size in pt
  Weight: SmallInt;                       // Bold as used in the Windows GDI
                                          //   400: normal
                                          //   700: bold
  Italic: boolean;                        // true if Italic is checked
  Res1: byte;                             // not used
  CharSpace: SmallInt;                    // Char. spacing
  WordSpace: SmallInt;                    // Word spacing
  Alignment: SmallInt;                    // Alignment
                                          //   0: Bottom Left
                                          //   1: Bottom Center
                                          //   2: Bottom Right
                                          //   3: Bottom Justified
                                          //   4: Middle Left
                                          //   5: Middle Center
                                          //   6: Middle Right
                                          //   7: only in LText!
                                          //   8: Top Left
                                          //   9: Top Center
                                          //  10: Top Right
                                          //  11: only in LText!
  LineSpace: SmallInt;                    // Line spacing
  ParaSpace: SmallInt;                    // Space after Paragraph
  IndentFirst: SmallInt;                  // Indent first line
  IndentOther: SmallInt;                  // Indent other lines
  nTabs: SmallInt;                        // number of tabulators for text symbol
  Tabs: array[0..31] of longint;          // Tabulators
  LBOn: wordbool;                         // true if Line below On is checked
  LBColor: SmallInt;                      // Line color (Line below)
  LBWidth: SmallInt;                      // Line width (Line below)
  LBDist: SmallInt;                       // Distance from text
  Res2: SmallInt;
  FrMode: byte;                           // Framing mode
                                          //   0: no framing
                                          //   1: shadow framing
                                          //   2: line framing
                                          //   3: rectangle framing
  FrLineStyle: byte;                      // Framing line style
                                          //   0: default OCAD 8 Miter
                                          //   2: ps_Join_Bevel
                                          //   1: ps_Join_Round
                                          //   4: ps_Join_Miter
  PointSymOn: boolean;                    // Point symbol is activated
  PointSymNumber: integer;                // Point symbol for text symbol activated if PointSymOn is t
  Res3: string[18];                       // not used
  FrLeft: SmallInt;                       // Left border for rectangle framing
  FrBottom: SmallInt;                     // Bottom border for rectangle framing
  FrRight: SmallInt;                      // Right border for rectangle framing
  FrTop: SmallInt;                        // Top border for rectangle framing
  FrColor: SmallInt;                      // Framing color
  FrWidth: SmallInt;                      // Framing width for line framing
  Res4: SmallInt;                         // not used
  Res5: wordbool;                         // not used
  FrOfX: SmallInt;                        // Horizontal offset for shadow framing
  FrOfY: SmallInt;                        // Vertical offset for shadow framing
end;
```

Line Text Symbol

Line Text symbols are stored in the following structure. In the explanation the terms used in the Line Text Symbol dialog box are shown. The unit of all dimensions is 0.01 mm, except for the font sizes which are measured in 0.1 typographical points.

```
TLTextSym = packed record
  Size: integer;                          // see TBaseSym
  SymNum: integer;                        // see TBaseSym
  Otp: byte;                              // 6
  Flags: byte;                            // see TBaseSym
  Selected: boolean;                      // see TBaseSym
  Status: byte;                           // see TBaseSym
  PreferredDrawingTool: byte;             // see TBaseSym
  CsMode: byte;                           // see TBaseSym
  CsObjType: byte;                        // see TBaseSym
  CsCDFlags: byte;                        // see TBaseSym
  Extent: integer;                        // see TBaseSym
  FilePos: integer;                       // see TBaseSym
  Group: SmallInt;                        // see TBaseSym
  nColors: SmallInt;                      // see TBaseSym
  Colors: array[0..13] of SmallInt;       // see TBaseSym
  Description: array[0..63] of char;      // see TBaseSym
  IconBits: array[0..483] of byte;        // see TBaseSym
  SymbolTreeGroup: Array[0..63] of Word; // see TBaseSym
  FontName: string[31];                   // TrueType font
  FontColor: SmallInt;                    // Color
  FontSize: SmallInt;                     // 10 times the value entered in Size
  Weight: SmallInt;                       // Bold as used in the Windows GDI
                                          //   400: normal
                                          //   700: bold
  Italic: boolean;                        // true if Italic is checked
  Res1: byte;                             // not used
  CharSpace: SmallInt;                    // Char. spacing
  WordSpace: SmallInt;                    // Word spacing
  Alignment: SmallInt;                    // Alignment     --> constant.pas
                                          //   0: Bottom Left
                                          //   1: Bottom Center
                                          //   2: Bottom Right
                                          //   3: Bottom All line
                                          //   4: Middle Left
                                          //   5: Middle Center
                                          //   6: Middle Right
                                          //   7: Middle All line
                                          //   8: Top Left
                                          //   9: Top Center
                                          //  10: Top Right
                                          //  11: Top All line
  FrMode: byte;                           // Framing mode
                                          //   0: no framing
                                          //   1: shadow framing
                                          //   2: line framing
  FrLineStyle: byte;                      // Framing line style
                                          //   0: default OCAD 8 and 9.0 Miter
                                          //   2: ps_Join_Bevel
                                          //   1: ps_Join_Round
                                          //   4: ps_Join_Miter
  Res2: string[31];                       // not used
  FrColor: SmallInt;                      // Framing color
  FrWidth: SmallInt;                      // Framing width for line framing
  Res3: SmallInt;                         // not used
  Res4: wordbool;                         // not used
  FrOfX: SmallInt;                        // Horizontal offset for shadow framing
  FrOfY: SmallInt;                        // Vertical offset for shadow framing
end;
```

Rectangle Symbol

Rectangle symbols are stored in the following structure. In the explanation the terms used in the Rectangle Symbol dialog box are shown. The unit of all dimensions is 0.01 mm.

```
TRectSym = packed record
  Size: integer;                          // see TBaseSym
  SymNum: integer;                        // see TBaseSym
  Otp: byte;                              // 7
  Flags: byte;                            // see TBaseSym
  Selected: boolean;                      // see TBaseSym
  Status: byte;                           // see TBaseSym
  PreferredDrawingTool: byte;             // see TBaseSym
  CsMode: byte;                           // see TBaseSym
  CsObjType: byte;                        // see TBaseSym
  CsCDFlags: byte;                        // see TBaseSym
  Extent: integer;                        // see TBaseSym
  FilePos: integer;                       // see TBaseSym
  Group: SmallInt;                        // see TBaseSym
  nColors: SmallInt;                      // see TBaseSym
  Colors: array[0..13] of SmallInt;       // see TBaseSym
  Description: array[0..63] of char;      // see TBaseSym
  IconBits: array[0..483] of byte;        // see TBaseSym
  SymbolTreeGroup: Array[0..63] of Word; // see TBaseSym
  LineColor: SmallInt;                    // Line color
  LineWidth: SmallInt;                    // Line width
  Radius: SmallInt;                       // Corner radius
  GridFlags: word;                        // A combination of the flags
                                          //   1: Grid On
                                          //   2: Numbered from the bottom
  CellWidth: SmallInt;                    // Cell width
  CellHeight: SmallInt;                   // Cell height
  ResGridLineColor: SmallInt;             // Reserved
  ResGridLineWidth: SmallInt;             // Reserved
  UnnumCells: SmallInt;                   // Unnumbered Cells
  UnnumText: string[3];                   // Text in unnumbered Cells
  Res1: SmallInt;                         // not used
  Res2: string[31];                       // not used
  ResFontColor: SmallInt;                 // Reserved
  FontSize: SmallInt;                     // font size
  Res3: SmallInt;                         // not used
  Res4: wordbool;                         // not used
  Res5: SmallInt;                         // not used
  Res6: SmallInt;                         // not used
end;
```

Objects

Each Object Index Block contains the position of the next Object Index Block and the file position and other information of 256 objects.

Object Index Block

```
TObjectIndexBlock = record   // Size: 10296 Bytes
  NextObjectIndexBlock: integer;
  Table: array[0..255] of TObjectIndex;
end;
```

OCAD Objet Index

```
TObjectIndex = packed record // Size: 40 Byte
  rc: LRect;                 // bounding box (lower left and upper right)
  Pos: integer;              // file position of the object -> TOcadObject
  Len: integer;              // number of coordinate pairs, the size of the object in the file is the
                             // Note: this is reserved space in the file, the effective length of the
  Sym: integer;              // -4 = layout vector object
                             // -3 = image object eg AI object
                             // -2 = graphic object
                             // -1 = imported, no symbol assigned or symbol number
  ObjType: byte;             // 1 = Point object
                             // 2 = Line object
                             // 3 = Area object
```

```
                            // 4 = Unformatted text
                            // 5 = Formatted text
                            // 6 = Line text
                            // 7 = Rectangle object
  EncryptedMode: byte;      // 0 = normal
                            // 1 = encoded object (TOcadObject)
  Status: byte;             // 0 = deleted (not undo) (eg from symbol editor or course setting)
                            // 1 = normal
                            // 2 = hidden
                            // 3 = deleted for undo
  ViewType: byte;           // 0 = normal object
                            // 1 = course setting object
                            // 2 = modified preview object
                            // 3 = unmodified preview object
                            // 4 = temporary object (symbol editor or control description)
  Color: SmallInt;          // Color number for symbolized objects or color of graphic objects, -1 f
  Group: SmallInt;          // Group number of grouped objects
  ImpLayer: SmallInt;       // Layer number of imported objects, 0 means no layer number
  LayoutFont: byte;         // Index of Ocd.Layout.FontAttributesList
  Res2: byte;               // reserved
end;
```

OCAD Object

```
TOcadObject = class(TObject)
  Sym: integer;                     // symbol number
                                    // -4 = layout vector object
                                    // -3 = from PDF, AI
                                    // -2 =
  Otp: byte;                        // object typ
  _Customer: byte;                  //
  Ang: SmallInt;                    // Angle, unit is 0.1 degrees, used for
                                    // - point object
                                    // - area objects with structure
                                    // - unformatted text objects
                                    // - rectangle objects
  nItem: integer;                   // number of coordinates in the Poly array
  nText: SmallInt;                  // number of characters in the Poly, array used for storing text
                                    // nText is > 0 for
                                    // - line text objects
                                    // - text objects
                                    // for all other objects it is 0
  Mark: Byte;                       // Used for Marked property
  SnappingMark: Byte;               // Used for Snapping marked property
  Col: integer;                     // Color number for symbolized objects or color of graphic object
  LineWidth: SmallInt;              //
  DiamFlags: SmallInt;              //
  ServerObjectId: integer;          // added for server objects
  Height: integer;                  // Height [1/256 mm]
  _Date: double;                    // not used
  Poly: TDPoly;                     // array[0..] coordinates of the object followed by a zero-termin
                                    // if nText > 0 TCord is explained at the beginning of this descr
end;
```

Parameter Strings

The Parameter Strings contain all the information about the setup structure, background maps,course setting and database connection.

Similar to the symbols and objects there are String Index Blocks which contain the basic information for 256 Parameter Strings and the file position of the strings.

TStringIndexBlock contains the basic information for 256 strings

```
TStringIndexBlock = record
  NextIndexBlock: integer;            // file position of the next StringIndexBlock, 0 if this is the
```

```
  Table: array[0..255] of TStringIndex;
end;
```

TStIndex contains the basic information for 1 string:

```
TStringIndex = packed record
  Pos: integer;                       // file position of string
  Len: integer;                       // length reversed for the string
  RecType: integer;                   // string typ number, if < 0 then deleted string
  ObjIndex: integer;                  // number of the object
end;
```

StringIndexBlk in the FileHeader points to the first StringIndexBlock.

The strings (null terminated) have the following structure:

- first field: all characters until the first tab (character 9). The first field can be missing (the string starts with a tab).
- tab (character 9)
- code: this is the first character after the tab
- value: all characters until the next tab
- tab
- code
- value
- ...

Some of String Types (number < 1000) may have multiple instances of the same type. They have to be stored as lists.

```
si_CsObject = 1;
  // First = Code
  // Y = Type (s = start, c = control, m = marked route, f = finish, d = control description, n = cou
  //           v = variation code, t = text block)
  // b = Symbol for field B (Trail-O, Macr-O, Micr-O)
  // c = Symbol for field C
  // d = Symbol for field D
  // e = Symbol for field E
  // f = Symbol for field F
  // g = Symbol for field G
  // h = Symbol for field H
  // m = f: Funnel tapes
  // o = t: Text control description object; '': iof symbols
  // p = Control description corner: 0=top left; 1=bottom left
  // s = Size information
  // t = Text for text description and text block
  // u = elevation user [double]
  // v = is elevation user used [boolean]
```

```
si_Course = 2;
  // First = CourseName
  // C = Climb
  // E = Extra distance
  // F = from start number (auto created class)
  // H = ClassName for control description (imported from event software)
  // K = Combination (imported from event software)
  // M = Map file
  // R = number of runners/teams (auto created class)
  // S = Map scale
  // T = to start number (auto created class)
  // Y = Course type (s = relay, o = one-man relay)
  // L = Number of Legs (used for relay and one-man relay)
  // s = start
  // c = control
```

```
  // m = marked route
  // k = mandatory crossing point
  // w = mandatory passage through out of bounds area
  // g = map change
  // f = finish
  // l = leg variation starts
  // b = branch of a leg variation starts
  // p = end of a leg variation
  // r = relay variation starts
  // v = branch of a relay variation starts
  // q = end of a relay variation
  // e = used internally only (control number)
  // i = used internally only (back to begin of variation)
  // j = used internally only (line to end of variation)
  // n = course name object
  // u = start number object
  // t = text block for control description
  // o = other object
```

```
si_CsClass = 3;
  // First = class name
  // c = Course name
  // f = FromNumber (relay)
  // r = Number of runners
  // t = ToNumber (relay)
```

```
si_DataSet = 4;
  // First = dataset name
  // e = dBase file
  // d = ODBC data source
  // u = database user name (encrypted)
  // p = database password (encrypted)
  // t = table
  // k = key field
  // y = symbol field
  // x = text field
  // f = size field
  // l = length unit
  // a = area unit
  // c = decimals
  // h = easting field
  // v = northing field
  // g = angle field
```

```
si_DbObject = 5;
  // First = key
  // d = Dataset
```

```
si_OimFile = 6;
```

```
si_PrevObj = 7;
  // First = course name;
  // o = object
  // d = description (eg object name)
  // f = from, startpoint of line
  // t = to, end point of line
```

```
si_BackgroundMap = 8;
  // a = angle omega [double, 8]
  // b = angle phi [double, 8]
  // d = dim
  // o = render with spot colors (only for raster background maps)
  // p = assigned to spot color (only for raster background maps)
  // q = subtract from spot color (0 = normal, 1 = subtract)
  // r = visible in background favorites (0 = hidden, 1 = visible)
  // s = visible in normal, spot color and draft mode (0 = hidden, 1 = visible)
  // t = transparent
  // x = offset x [double, 6]
  // y = offset y [double, 6]
```

```
  // u = pixel size x [double, 10]
  // v = pixel size y [double, 10]
  // i = is infrared image (0=undefined, 1=32bit-infrared, 2=32bit RGB)
  // m = is WMS online background map
  // k = WMS server name
  // l = WMS layer name
  // n = WMS layer scale range
```

```
si_Color = 9;
  // First = name
  // n = number
  // c = cyan
  // m = magenta
  // y = yellow
  // k = black
  // o = overprint
  // t = transparency
  // s = spot color separation name
  // p = percentage in the spot color separation
```

```
si_SpotColor = 10;
  // First = name
  // v = visible
  // n = number
  // f = frequency (lpi)
  // a = angle
  // c = cyan
  // m = magenta
  // y = yellow
  // k = black
```

```
si_FileInfo_OCAD10 = 11; //replace in OCAD 11 by si_MapNotes = 1061
```

```
si_Zoom = 12;
  // x = offset x
  // y = offset y
  // z = zoom
```

```
si_ImpLayer = 13;
  // First = Name
  // n = layer number
```

```
si_OimFind = 14;
  // First = Name
  // c = Condition
  // d = Dataset
  // f = From zoom
  // h = Hint field
  // n = Name field
  // o = Hotspot type
  // p = Pointer type
  // s = Show hotspots
  // l = List names(in listbox)
  // t = To zoom
  // u = URL field
  // x = Prefix
  // y = Postfix
  // z = Target
  // r = Pointer color red
  // g = Pointer color green
  // b = Pointer color blue
  // R = Hotspot color red
  // G = Hotspot color green
  // B = Hotspot color blue
```

```
si_SymTree = 15;
  // First = name
```

```
  // g = group id
  // v = visible
  // e = is node expanded (boolean)
  // i = Level in Tree
```

```
si_CryptInfo = 16;
  // Description is in TCryptInfo class
```

```
si_Bookmark = 18;
  // First = bookmark name
  // d = description
  // x = offset x
  // y = offset y
  // z = zoom
```

```
si_Selection = 19;
  // First = selection name
  // n = number
  // o = object ids
```

```
si_GpsAdjustPar = 21;
  // m = Gps adjusted mode 1=true; 0=false
  // n = number of GPS adjustment points --> GpsAdjustPoint
  // a = GPS angle
```

```
si_GpsAdjustPoints = 22;
  // First = name
  // x = offset x on map
  // y = offset y on map
  // h = Longitude
  // v = Latitude
  // c = checked (true/false)
```

```
si_Group = 23;
  // First = Name
  // n = group number
```

```
si_RecentDocs = 24;
  // First = file name
```

```
si_CsAutoCdAllocationTable = 25;
  // First = Map symbol number (STRING)
  // a = Control description symbol 0 (STRING)
  // b = Control description symbol 1 (STRING)
  // c = Control description symbol 2 (STRING)
  // d = Control description symbol 3 (STRING)
  // e = Control description symbol 4 (STRING)
  // f = Control description symbol 5 (STRING)
  // g = Control description dragged direction (STRING)
  // h = Control description click (STRING)
```

```
si_RulerGuidesList = 26;
  // h = horizontal guide (0=vertical, 1=horizontal)
  // c = x or y ccordiante (INTEGER, OCAD coordinate)
```

```
si_LayoutObjects = 27;
  // First = path for images; description for vector objects
  // r = type: raster image object = 1; vector object = 0
  // s = visible (0 = hidden, 1 = visible)
  // a = angle omega
  // b = angle phi
  // d = dim
```

```
  // t = transparent
  // x = offset x
  // y = offset y
  // u = pixel size x
  // v = pixel size y
  // i = is infrared image (0=undefined, 1=32bit-infrared, 2=32bit RGB)
  // n = ActObjectIndex
```

```
si_LayoutFontAttributes = 28;
  // First = font name
  // s = font size
```

```
si_PrintAndExportRectangleList = 29;
  // First = Name
  // b = bottom [integer, ocad unit]
  // l = left [integer, ocad unit]
  // r = right [integer, ocad unit]
  // t = top [integer, ocad unit]
```

```
si_DisplayPar = 1024;
  // f = Show symbol favorites
  // g = selected symbol group
  // s = selected symbol
  // t = Show symbol tree
  // h = horizontal splitter (pixel from top)
  // v = vertical splitter (pixel from right)
  // b = horizontal splitter for background map panel (pixel from top)
  // i = display mode for unsymbolized objects (0=normal, 2=hidden)
  // j = display mode for graphic objects (0=normal, 2=hidden)
  // k = display mode for image objects (0=normal, 1=protect, 2=hidden)
  // l = display mode for layout objects (0=normal, 2=hidden)
```

```
si_OimPar = 1025;
  // First = OpenLayers
  // a = Antialiasing  [boolean]
  // b = Border width [integer]
  // A = Map Title [string]
  // C = Map Subtitle [string]
  // E = Editlayer enable [boolean]
  // F = Searchbox enable [boolean]
  // m = Do not create tiles [boolean]
  // H = Show layer selector [boolean]
  // I = Show overview map [boolean]
  // i = Show overview map maximized [ boolean]
  // K = Show mouse coordinates [boolean]
  // P = Show Permalink [boolean]
  // J = Font size in pt [integer]
  // L = Subheader fontsize in pt [integer]
  // R = Site background color red [integer: 0..255]
  // G = Site background color green [integer: 0..255]
  // B = Site background color blue  [integer: 0..255]
  // M = Border color [string] as hex e.g.: 00FF00 must be exact 6 values
  // N = Header font color [string] as hex exapmle see border color
  // O = Header background color [string] as hex
  // T = Subheader font color [string] as hex
  // Q = Subheader background color [string] as hex
  // X = Filename of last Safeplace [string]
  // f = Generate map from zoomlevel [integer]
  // t = Generate map to zoomlevel [integer]
```

```
si_PrintPar = 1026;
  // a = print scale
  // l = landscape
  // c = print (0 = color map, 1 = spot color separation)
  // g = print screen grid
  // d = screen grid color
  // i = intensity
  // w = additional width for lines and dots
  // r = Range (0 entire map, 1 part of map, 2 one page)
  // L = part of map left
  // B = part of map bottom
```

```
  // R = part of map right
  // T = part of map top
  // x = horizontal overlap
  // y = vertical overlap
  // b = print black   (course setting only)
  // m = Mirror        (course setting only)
  // s = HorScal       (course setting only)
  // t = VerScal       (course setting only)
  // p = reference point for part of map setup ([0..8])
```

```
si_CdPrintPar = 1027;
  // t = Title
  // s = Size
```

```
si_DefaultBackgroundMapsPar = 1028;
  // s = default scale
```

```
si_EpsPar = 1029;
  // r = EPS resolution
```

```
si_ViewPar = 1030;
  // b = draft mode IGN for ocad-map [int: 0..100]
  // c = draft mode IGN background maps [int: 0..100]
  // d = hidden background maps
  // m = draft mode for ocad-map [int: 0..100]
  // t = draft mode background maps [int: 0..100]
  // v = view mode (0=normal, 1=spot colors, 2=draft, 3=draft IGN)
  // x = offset x
  // y = offset y
  // z = zoom
  // h = hatch areas (0 = normal, 1 = hatched)
  // k = keyline mode (0 = normal, 1 = keyline)
  // i = map visible
  // l = hidden layout
  // OLD: a, p, s
```

```
si_CoursePar = 1031;
  // a = create classes automatically
  // b = background for control descriptions
  // c = numbering (0 = number, 1 = number and code, 2 = code only)
  // d = control descriptions for all controls
  // e = event title
  // f = control frequencies (0 = number and frequencies, 1 = frequencies only)
  // h = thicker horizonal line in control description (0=not, 1=every third, 2=every fourth)
  // i = maximal length of control description (number of rows)
  // l = distance to connection line
  // n = distance to number
  // o = lock for cs object positions (0=unlocked, 1=locked) (BOOL)
  // p = fullstop after control number
  // q = lock for cs courses (0=unlocked, 1=locked) (BOOL)
  // r = export relay combinations
  // s = cell size control description
  // t = course title
  // w = write number of start to control description
```

```
si_TiffPar = 1032;
  // c = compression (1 = no compression, 2 = CCITT, 4 = FaxG4, 5 = LZW)
```

```
si_TilesPar = 1033;
  // w = width
  // h = height
```

```
si_DbPar = 1034;
  // d = dataset
  // l = last code
  // n = create new record
```

```
si_ExportPar = 1035;
  // First = Format (AI, BMP, DXF, GIF, JPEG, OIM, PDF, Shape, SVG, TIFF)
  // a = Anti-Aliasing
  // b = combined spot colors
  // c = color format (0 = 24 bit, 1 = 256 colors, 2 = grayscale, 3 = 1 bit, 4 = halftone screen)
  // g = Raster (geo-referenced)export mode 0 = resolution, 1 = pixel size
  // i = pixel size (in Meter)
  // l = color correction (0 off, 1 on)
  // o = spot color separations (0=CMYK, 1=spot colors)
  // p = part of map (0 off, 1 on)
  // q = JPEG quality (INTEGER, [0..100])
  // r = resolution
  // s = scale
  // t = tiles (0 off, 1 on)
  // w = world file
  // z = compressed svg file (0 off, 1 on)
```

```
si_CsExpTextPar = 1037;
  // C = classes (0 = courses, 1 = classes)
  // L = export climbing (0 = off, 1 = on)
```

```
si_CsExpStatPar = 1038;
  // C = classes (0 = courses, 1 = classes)
  // a = separator 1
  // b = tab 1
  // c = separator 2
  // d = tab 2
  // e = separator 3
  // f = tab 3
```

```
si_ScalePar = 1039;
  // a = real world angle
  // b = additional local horizontal offset in m
  // c = additional local vertical offset in m
  // d = grid distance for real world  in m [double, 6]
  // g = grid distance for paper coordinates in mm [double, 6]
  // i = grid and zone
  // m = map scale
  // r = real world cord (0 = paper, 1 = real world)
  // x = real world offset x
  // y = real world offset y
```

```
si_DbCreateObjPar = 1040;
  // c = condition
  // d = dataset
  // t = text field
  // m = unit of measure (m = m, km = km)
  // u = horizontal offset
  // v = vertical offset
  // x = horizontal field
  // y = vertical field
```

```
si_SelectedSpotColors = 1041;
```

```
si_XmlScriptPar = 1042;
  // f = last used file
```

```
si_BackupPar = 1043;
  // p = path
```

```
si_ExportPartOfMapPar = 1044;
  // b = boundary (0=rectangular boundries, 1=selected object for boundries)
  // s = export with selected object (0=false, 1=true)
  // r = reference point for part of map setup ([0..8])
  // l = export database links (0=false, 1=true)
```

```
si_DemPar = 1045;
  // First = file
  // l = DEM loaded
  // f = Frame visible
  // i = last used import folder
```

```
si_GpsImportFilePar = 1046;
  // a = assign symbol (true/false)
  // t = symbol number for tracks
  // w = symbol number for waypoint
```

```
si_ImportXyz = 1047;
  // n = point symbol number
```

```
si_RelayCoursesDialog = 1048;
  // First = last exported or printed course (STRING)
  // l = legs (-1=All) (INT)
  // p = variants selected (0=start numbers, 1=variants) (BOOL)
  // s = start number (-1=All) (STRING)
  // v = variants index (-1=All) (INT)
```

```
si_CsAutoControlDescription = 1049;
  // First = ocd background map file name for auto control description (STRING)
  // a = auto control description (0=off, 1=off) (BOOL)
```

```
si_GpxExportPar = 1050;
  // First = description
  // a = author
  // k = keywords
  // r = routes or tracks (true = routes, false = tracks)
```

```
si_KmlInfo = 1051;
  // Description in ExpKml.pas
```

```
si_GpsRouteAnalyzer = 1052;
  // First = Project file
```

```
si_CoordinateSystemPar = 1053;
  //First = description
  // d = datumId
  // e = ellipsoidId
  // x = ellipsoid axis
  // t = ellipsoid flattening
  // r = projectionId
  // m = false easting
  // n = false northing
  // f = scale factor
  // c = central meridian
  // o = longitude of origin
  // a = latitude of origin
  // s = standard parallel 1
  // p = standard parallel 2
  // l = location
  // z = azimuth
  // g = EPSG code
```

```
si_GraticulePar = 1054;
  // h = horizontal distance
  // v = vertical distance
  // l = symbol number for grid lines
  // t = symbol number for text labels
```

```
si_GraticuleNameIndexPar = 1055;
  // s = style
  // x = index origin of longitude
  // y = index origin of latitude
  // h = horizontal distance
  // v = vertical distance
```

```
si_KmzExportPar = 1056;
  // first = Name: string
  // t = Tiles: integer (0 = no tiles, 1 = Garmin Custom Maps optimized, 2 = tiles)
```

```
si_LegendPar = 1057;
  // u = show only Used Symbols
  // v = show also hidden symbols
  // h = icon Height
  // w = icon Height
  // y = line Spacing100
  // p = show Point Symbols
  // l = show Point Symbols
  // a = show Point Symbols
  // r = show Point Symbols
  // t = show Point Symbols
  // z = show Point Symbols
  // d = description Symbol
  // s = show Symbol Number
```

```
si_RulersPar = 1058;
  // no First
  // s = show (0=false, 1=true)
  // x = ruler origin x (INTEGER, OCAD coordinate)
  // y = ruler origin y (INTEGER, OCAD coordinate)
  // m = move also ruler guides (0=false, 1=true)
```

```
si_RulerGuidesPar = 1059;
  // no First
  // s = show (0=false, 1=true)
  // l = lock (0=false, 1=true)
```

```
si_DbOptions = 1060;
  // c = create record when cutting object (0=false, 1=true)
  // d = delete record when deleting object (0=false, 1=true)
```

```
si_MapNotes = 1061;
  // First = text
```

```
si_SendFileByEmail = 1062;
  // First = subject
  // t = to
  // d = add loaded DEM (0=false, 1=true)
  // a = add loaded databases (0=false, 1=true)
  // b = add loaded databases (0=false, 1=true)
  // l = add loaded databases (0=false, 1=true)
```

```
si_MapGridPar = 1063;
  // a = angle
  // e = easting offset
  // n = northing offset
  // h = horizontal (easting) distance
```

```
  // v = vertical (northing) distance
  // x = create easting grid lines
  // y = create northing grid lines
  // j = create vertices at grid junctions
  // l = symbol number for grid lines
  // t = symbol number for text labels
```

```
si_DemSlopePar = 1064;
  // m = Method: 0 = continuous; 1 = black/white
  // c = Slope for black pixels in continuous method
  // b = Slope for black pixels in black/white method
  // l = load slope map as background map
```

```
si_DemProfilePar = 1065;
  // First = profile template path
  // a = scale option (0=auto-scale, 1=used defined scale)
  // c = scale
  // d = distance unit (0=km, 1=m)
  // g = show grid (0=false, 1=true)
  // m = time unit (0=h:mm)
  // r = round time steps (0=no round, 5=round to 5 minutes, 10=round to 10 minutes)
  // t = show text at the bottom of the profile (0=false, 1=true)
  // x = length resolution
  // z = elevation resolution
  // k = elevation factor
```

```
si_DemHillshadingPar = 1066;
  // m = Method: 0 = simple; 1 = combined
  // a = azimut
  // d = declination
  // e = exaggeration
  // l = load hillshading as background map
```

```
si_DemHypsometricMapPar = 1067;
  // m = Method: 0 = grayscale; 1 = colored
  // l = load hillshading as background map
```

```
si_DemClassifyVegetationPar = 1068;
  // m = Method: 0 = grayscale; 1 = colored
  // l = load hillshading as background map
```

```
si_ShapeExportPar = 1069;
  // a = item 'area objects' selected (0=false, 1=true)
  // c = create projection file
  // d = dataset ['all' = All Objects or name of dataset]
  // l = item 'line objects' selected (0=false, 1=true)
  // p = item 'point objects' selected (0=false, 1=true)
  // r = replace word wrap by tilde (0=false, 1=true)
  // t = item 'text objects' selected (0=false, 1=true)
  // u = UTF-8 Encoding (0=false, 1=true)
```

```
si_DxfExportPar = 1070;
  // a = convert text from ANSI to OEM (0=false, 1=true)
  // o = convert text from OEM to Unicode (0=false, 1=true)
  // s = only objects with selected symbol (0=false, 1=true)
  // l = add symbol description in DXF layer name (0=false, 1=true)
  // e = export OCAD Curves as DXF splines (0=false, 1=true)
  // c = coordinates (0=m, 1=mm)
```

```
si_DemImportLasPar = 1071;
  // u = unclassified
  // g = ground
  // l = low  vegetation
  // m = mean vegetation
  // h = high vegetation
  // b = buildings
```

```
  // w = water class
  // o = other class
  // r = Return option
```

```
si_MapRoutingPar = 1072;
  // f = from (0=Location, 1=Coordiante)
  // l = from location
  // e = from easting
  // n = from northing
  // T = to (0=Location, 1=Coordiante)
  // L = to location
  // E = to easting
  // N = to northing
  // d = add driving directions (0=false, 1=true)
```

Retrieved from "http://www.ocad.com/en.wiki/index.php?title=OCAD_11_File_Format&oldid=4761"

- This page was last modified on 12 June 2012, at 13:46.