Frames
The Labs.Com
2014/09/24
About Us ·
What's New ·
Services ·
PicArt
Programmer
HTML
Graphic Lab
PostScript ·
SVG Lab
3D Graphics
Blender
3DS Specifics ·
3DS File Forma.. ·
GIFMerge ·
GIFTool ·
The GIMP
Lense Flares ·
X11 Lab
Web Lab
Admin Lab
System Lab
OS Lab
Sound Lab
Video Lab
Sitemap ·

The Labs.Com ▪ Graphic Lab ▪ 3D Graphics ▪ Blender ▪ 3DS Specifics

Last update 1998/10/05

# 3DS Specifications

The Labs - Design & Functionality For The Net

## Autodesk 3D Studio File Format

# 3D Studio File Format (3ds).

## Autodesk Ltd.

Document Revision 0.93 - January 1997

3D-Studio File Format (.3ds)
Autodesk Ltd.

Document Revision 0.93 - January 1997

Rewritten by Martin van Velsen (email: vvelsen@ronix.ptf.hro.nl )
and Robin Fercoq ( 3ds-bin + mli )(email: robin@msrwww.fc-net.fr)
Based on documentation by Jim Pitts ( email: jim@micronetics.com )
Source update provided by:
Albert Szilvasy (email: szilvasy@almos.vein.hu )

A lot of the chunks are still undocumented if you know what they do please email me Martin van Velsen, Robin Feroq or Jimm Pitts. As I get more information on the file format, I will document it for everyone to see. I will post this regurlarly to alt.3d and alt.3d-studio and I can be contacted there if my email does not work.

(Also see the 3d-studio material .mli documentation by Robin Feroq.)

Disclaimer.
This document describes the file format of the 3ds files generated by 3d-studio by Autodesk. By using the information contained within, you agree not to hold any of the authors liable if, from its use, you f^Hmuck something up. OK?

Autodesk has at not yet released the offical specifications of the 3d-studio formats. You will therefor receive NO support from Autodesk or any company related to Autodesk concerning the nature and contents of the 3d-studio binary .3ds and .mli formats.

A warning beforehand. This docs describes the format of 3ds files produced by version 3.0 and higher of 3d-studio. You can find this version information at byte 29 in the binary file.

This document can be found on the regular newsgroups:
alt.3d and alt.3d-studio
It can also be found at: "http://www.mediatel.lu"

----------------------------------

Contents

1. Introduction

```
======================
```

The 3ds file format is made up of chunks. They describe what information is to follow and what it is made up of, it's ID and the location of the next block. If you don't understand a chunk you can quite simply skip it. The next chunk pointer is relative to the start of the current chunk and in bytes. The binary information in the 3ds file is written in a special kind of way. Namely the lest significant byte comes first in an int. For example: 4A 5C  (2 bytes in hex) would be 5C high byte and 4A low byte. In a long it is: 4A 5C 3B 8F where 5C4A is the low word and 8F 3B is the high word. And now for the chunks. A chunk is defined as:

```
start end size name
0     1   2    Chunk ID
2     5   4    Pointer to next chunk relative to the place where
               Chunk ID is, in other words the length of the chunk
```

Chunks have a hierarchy imposed on them that is identified byt it's ID. A 3ds file has the Primary chunk ID 4D4Dh. This is always the first chunk of the file. Within the primary chunk are the main chunks.

to give you a preview and a reference to the hierarchy of chunks, below is a diagram to show the diffrent chunk ID's and their place in the file. The chunks are given a name because below the diagram is a list which defines the names to the actual chunk id's. This makes it easier to put it in some source code ( how convenient that some sample code is included )

```
MAIN3DS  (0x4D4D)
|
+--EDIT3DS  (0x3D3D)
|  |
|  +--EDIT_MATERIAL (0xAFFF)
|  |  |
|  |  +--MAT_NAME01 (0xA000) (See mli Doc)
|  |
|  +--EDIT_CONFIG1  (0x0100)
|  +--EDIT_CONFIG2  (0x3E3D)
|  +--EDIT_VIEW_P1  (0x7012)
|  |  |
|  |  +--TOP            (0x0001)
|  |  +--BOTTOM         (0x0002)
|  |  +--LEFT           (0x0003)
|  |  +--RIGHT          (0x0004)
|  |  +--FRONT          (0x0005)
|  |  +--BACK           (0x0006)
|  |  +--USER           (0x0007)
|  |  +--CAMERA         (0xFFFF)
|  |  +--LIGHT          (0x0009)
|  |  +--DISABLED       (0x0010)
|  |  +--BOGUS          (0x0011)
|  |
|  +--EDIT_VIEW_P2  (0x7011)
|  |  |
|  |  +--TOP            (0x0001)
|  |  +--BOTTOM         (0x0002)
|  |  +--LEFT           (0x0003)
|  |  +--RIGHT          (0x0004)
|  |  +--FRONT          (0x0005)
|  |  +--BACK           (0x0006)
|  |  +--USER           (0x0007)
|  |  +--CAMERA         (0xFFFF)
|  |  +--LIGHT          (0x0009)
|  |  +--DISABLED       (0x0010)
|  |  +--BOGUS          (0x0011)
|  |
|  +--EDIT_VIEW_P3  (0x7020)
|  +--EDIT_VIEW1    (0x7001)
|  +--EDIT_BACKGR   (0x1200)
|  +--EDIT_AMBIENT  (0x2100)
|  +--EDIT_OBJECT   (0x4000)
|  |  |
|  |  +--OBJ_TRIMESH   (0x4100)
|  |  |  |
|  |  |  +--TRI_VERTEXL          (0x4110)
|  |  |  +--TRI_VERTEXOPTIONS    (0x4111)
|  |  |  +--TRI_MAPPINGCOORS     (0x4140)
|  |  |  +--TRI_MAPPINGSTANDARD  (0x4170)
|  |  |  +--TRI_FACEL1           (0x4120)
|  |  |  |  |
|  |  |  |  +--TRI_SMOOTH            (0x4150)
|  |  |  |  +--TRI_MATERIAL          (0x4130)
|  |  |  |
|  |  |  +--TRI_LOCAL            (0x4160)
|  |  |  +--TRI_VISIBLE          (0x4165)
|  |  |
|  |  +--OBJ_LIGHT    (0x4600)
|  |  |  |
|  |  |  +--LIT_OFF              (0x4620)
|  |  |  +--LIT_SPOT             (0x4610)
|  |  |  +--LIT_UNKNWN01         (0x465A)
|  |  |
|  |  +--OBJ_CAMERA    (0x4700)
|  |  |  |
|  |  |  +--CAM_UNKNWN01         (0x4710)
|  |  |  +--CAM_UNKNWN02         (0x4720)
|  |  |
|  |  +--OBJ_UNKNWN01 (0x4710)
|  |  +--OBJ_UNKNWN02 (0x4720)
|  |
|  +--EDIT_UNKNW01  (0x1100)
|  +--EDIT_UNKNW02  (0x1201)
|  +--EDIT_UNKNW03  (0x1300)
|  +--EDIT_UNKNW04  (0x1400)
|  +--EDIT_UNKNW05  (0x1420)
|  +--EDIT_UNKNW06  (0x1450)
|  +--EDIT_UNKNW07  (0x1500)
|  +--EDIT_UNKNW08  (0x2200)
|  +--EDIT_UNKNW09  (0x2201)
|  +--EDIT_UNKNW10  (0x2210)
|  +--EDIT_UNKNW11  (0x2300)
|  +--EDIT_UNKNW12  (0x2302)
|  +--EDIT_UNKNW13  (0x2000)
|  +--EDIT_UNKNW14  (0xAFFF)
|
+--KEYF3DS (0xB000)
   |
   +--KEYF_UNKNWN01 (0xB00A)
   +--............. (0x7001) ( viewport, same as editor )
   +--KEYF_FRAMES   (0xB008)
   +--KEYF_UNKNWN02 (0xB009)
   +--KEYF_OBJDES   (0xB002)
      |
      +--KEYF_OBJHIERARCH  (0xB010)
      +--KEYF_OBJDUMMYNAME (0xB011)
      +--KEYF_OBJUNKNWN01  (0xB013)
      +--KEYF_OBJUNKNWN02  (0xB014)
      +--KEYF_OBJUNKNWN03  (0xB015)
      +--KEYF_OBJPIVOT     (0xB020)
      +--KEYF_OBJUNKNWN04  (0xB021)
      +--KEYF_OBJUNKNWN05  (0xB022)
```

A chunk type which you can find all through the file are the color chunks which are called:

COL_RGB<br>
COL_TRU<br>
COL_UNK

## 2. Chunks anyone ?

Now for the actual numbers, as you will see I used the define, because I ripped that part right out of my code. Be carefull however because there are a lot of new chunk types which were not documented in the original paper by Jim Pitts.

```
//>------ Primary chunk

#define MAIN3DS       0x4D4D

//>------ Main Chunks

#define EDIT3DS       0x3D3D  // this is the start of the editor config
#define KEYF3DS       0xB000  // this is the start of the keyframer config

//>------ sub defines of EDIT3DS
```

```
#define EDIT_MATERIAL 0xAFFF
#define EDIT_CONFIG1  0x0100
#define EDIT_CONFIG2  0x3E3D
#define EDIT_VIEW_P1  0x7012
#define EDIT_VIEW_P2  0x7011
#define EDIT_VIEW_P3  0x7020
#define EDIT_VIEW1    0x7001
#define EDIT_BACKGR   0x1200
#define EDIT_AMBIENT  0x2100
#define EDIT_OBJECT   0x4000

#define EDIT_UNKNW01  0x1100
#define EDIT_UNKNW02  0x1201
#define EDIT_UNKNW03  0x1300
#define EDIT_UNKNW04  0x1400
#define EDIT_UNKNW05  0x1420
#define EDIT_UNKNW06  0x1450
#define EDIT_UNKNW07  0x1500
#define EDIT_UNKNW08  0x2200
#define EDIT_UNKNW09  0x2201
#define EDIT_UNKNW10  0x2210
#define EDIT_UNKNW11  0x2300
#define EDIT_UNKNW12  0x2302
#define EDIT_UNKNW13  0x3000
#define EDIT_UNKNW14  0xAFFF

//>------ sub defines of EDIT_OBJECT
#define OBJ_TRIMESH   0x4100
#define OBJ_LIGHT     0x4600
#define OBJ_CAMERA    0x4700

#define OBJ_UNKNWN01  0x4010
#define OBJ_UNKNWN02  0x4012 //>>---- Could be shadow

//>------ sub defines of OBJ_CAMERA
#define CAM_UNKNWN01  0x4710
#define CAM_UNKNWN02  0x4720

//>------ sub defines of OBJ_LIGHT
#define LIT_OFF       0x4620
#define LIT_SPOT      0x4610
#define LIT_UNKNWN01  0x465A

//>------ sub defines of OBJ_TRIMESH
#define TRI_VERTEXL   0x4110
#define TRI_FACEL2    0x4111
#define TRI_FACEL1    0x4120
#define TRI_SMOOTH    0x4150
#define TRI_LOCAL     0x4160
#define TRI_VISIBLE   0x4165

//>>------ sub defs of KEYF3DS

#define KEYF_UNKNWN01 0xB009
#define KEYF_UNKNWN02 0xB00A
#define KEYF_FRAMES   0xB008
#define KEYF_OBJDES   0xB002

//>>------  these define the different color chunk types
#define COL_RGB  0x0010
#define COL_TRU  0x0011
#define COL_UNK  0x0013

//>>------ defines for viewport chunks

#define TOP           0x0001
#define BOTTOM        0x0002
#define LEFT          0x0003
#define RIGHT         0x0004
#define FRONT         0x0005
#define BACK          0x0006
#define USER          0x0007
#define CAMERA        0x0008 // 0xFFFF is the actual code read from file
#define LIGHT         0x0009
#define DISABLED      0x0010
#define BOGUS         0x0011
```

## 3. 3D Editor Chunks

===========================

So far for the quick stuff now the more detailed info.

* Main chunks

The main chunk ( the primary chunk of 0x4D4D that is ) is actually the complete file. So the size of this chunk is the size of the file minus the main chunk header.

There are two more main chunks, the 3d-editor chunk and the keyframer chunk:

| id | |
|---|---|
| 3D3D | Start of Editor data ( this is also the place where the objects are ) |
| B000 | Start of Keyframer data |

Directly after a Main chunk is another chunk. This could be any other type of chunk allowable within its main chunks scope. ( see diagram )

* Subchunks of 3D3D

| id | Description |
|---|---|
| 0100 | Part of configuration |
| 1100 | unknown |
| 1200 | Background Color |
| 1201 | unknown |
| 1300 | unknown |
| 1400 | unknown |
| 1420 | unknown |
| 1450 | unknown |
| 1500 | unknown |
| 2100 | Ambient Color Block |
| 2200 | fog ? |
| 2201 | fog ? |
| 2210 | fog ? |
| 2300 | unknown |
| 3000 | unknown |
| 3D3E | Editor configuration main block |
| 4000 | Definition of an Object |
| AFFF | Start of material list |

* Subchunks of AFFF - Start of material list
* A000 - material name
- This chunk contains the name of the material which is an ASCIIZ string

(
More material chunks are explained in the doc about 3d-studio .mli files. The chunk types mentioned in that doc are exactly the same as in the .3ds file
)

* Subchunks of 3D3E - Editor configuration

| id | Description |
|---|---|
| 7001 | Start of viewport indicator |
| 7011 | Viewport definition ( type 2 ) |
| 7012 | Viewport definition ( type 1 ) |
| 7020 | Viewport definition ( type 3 ) |

The 3D3E chunk is a curious one because it contains a lot of redundant data. ( or so it seems ) The most important chunk is 7020. this chunk describes the 4 viewports wich are active in the editor. I assume that U are using the 4 normal viewport layout, because I have not tested it with other configurations. The editor confid will contain 5x chunk 7020 and 5x chunk 7011. only the first 4 7020 chunks are important for how the viewports look like. I guess that the other chunks only give additional info, but I am not sure. The things you are looking for in these chunks is at byte: 6 & 7 ( as offset from the first 6 bytes chunk header and pointer ) these bytes ( unsigned int ) contain the info at to what view is used, with the following id's:

| id | Description |
|---|---|
| 0001 | Top |
| 0002 | Bottom |
| 0003 | Left |
| 0004 | Right |
| 0005 | Front |
| 0006 | Back |
| 0007 | User |
| FFFF | Camera |

```
0009   Light
0010   Disabled

* Subchunks of 4000 - Object description Block

- first item of Subchunk 4000 is an ASCIIZ string of the objects name.
  ASCIIZ means a string of charakters ended by a zero.

Remember an Object can be a Camera a Light or a mesh
 id      Description
4010     unknown
4012     shadow ?
4100     Triangular Polygon List ( Contains only subchunks )
4600     Light
4700     Camera

( Mapping:

  These chunks are optional. They stand just after the vertex list when
  the object is mapped. )

* Subchunks of 4100 - Triangular Polygon List

 id      Description
4110     Vertex List
4111     Vertex Options
4120     Face List
4130     Face Material
4140     Mapping Coordinates
4150     Face smoothing group
4160     Translation Matrix
4165     Object visible/invisble
4170     Standard Mapping

* 4110 - Vertex List

  start end size type          name
  0     1    2   unsigned int  Total vertices in object
  2     5    4   float         X-value
  6     9    4   float         Y-value
 10    13    4   float         Z-value

bytes 2..13 are repeated times the total amount of vertices in the object

* 4111 - Vertex Options

First 2 bytes: number of vertices.

Then a short int for each vertex:

bit 0-7    0
bit 8-10   x
bit 11-12  0
bit 13   vertex selected in selection 3
bit 14   vertex selected in selection 2
bit 15   vertex selected in selection 1

bit 8-10 are just like random. From a save to another of the same scene
it may change.

Other bits (0-7 and 11-12) have effects on visibility of vertex.

The 4111 chunk can be deleted without much influence, 3ds will still load
the file all right.


* 4120 - Face list

start end size type          name
0     1    2   unsigned int  total polygons in object (numpoly)

2     3    2   unsigned int  number of vertex A
4     5    2   unsigned int  number of vertex B
6     7    2   unsigned int  number of vertex C
8     9    2   unsigned int  face info (*)

repeats 'numpoly' times for each polygon.

The first three ints are the three vertices of the face.
0 stands for the first vertex defined in the vertex list.
The order has a purpose: to give the direction for the normal
```

```
 of each face.
 If you turn a screw (standard screw) in the way the vertices
 indicate you will find the normal.
 If vertices given in order are A B C:

               C
               ^
               |
        A----->B

 This means unscrewing => the normal points out of the screen.

 (*) this number is is a binary number which expands to 3 values.
     for example 0x0006 would expand to 110 binary. The value should be
     read as 1 1 0 .This value can be found in 3d-studio ascii files as
     AB:1 BC:1 AC:0 .Which probably indicated the order of the vertices.
     For example AB:1 would be a normal line from A to B. But AB:0 would
     mean a line from B to A.

     bit 0       AC visibility
     bit 1       BC visibility
     bit 2       AB visibility
     bit 3       Mapping (if there is mapping for this face)
     bit 4-8   0 (not used ?)
     bit 9-10  x (chaotic ???)
     bit 11-12 0 (not used ?)
     bit 13      face selected in selection 3
     bit 14      face selected in selection 2
     bit 15      face selected in selection 1

 * 4130 - Face Material Chunk

   If the object is all default material there is no 4130 chunk.
   In fact, there is one 4130 chunk for each material present on the object.

   Each 4130 face material chunks begins with an asciiz of a material,
   then after the null character is a short int that gives the number
   of faces of the object concerned by this material, then there is the
   list itself of these faces. 0000 means the first face of the (4120)
   face list.

   ************ Read the Doc on MLI files for more info on **************
   ************ Mapping and Materials                      **************

 * 4140 Mapping coordinates.

   First 2 bytes: number of vertices.

   Then, for each vertex 2 floats that give the mapping coordinates.
   That is, if a point is at the center of the map it will have 0.5 0.5
   as mapping coordinates.

 * 4150 - Face Smoothing Group

   nfaces*4bytes

   If read as long int, the nth bit indicate if the face belongs or not
   to the nth smoothing group.

 * 4160 Local axis

   Local axis info.
   The three first blocks of three floats are the definition
   (in the absolute axis) of the local axis X Y Z of the object.
   And the last block of three floats is the local center of the object.


 * 4170 Standard mapping

   First 2 bytes: type of mapping
   0 => plannar or specific (in this case, like mapping from the lofter,
                             the information of this chunk is irrelevant)
   1 => cylindrical
   2 => spherical

   then come 21 floats that describe the mapping.

* 4600 - Light

   start end size type   name
   0      3    4  float  Light pos X
```

| | | | | |
|---|---|---|---|---|
| 4 | 7 | 4 | float | Light pos Y |
| 8 | 11 | 4 | float | Light pos Z |

after this structure check for more chunks.

| id | Description ( full description later ) |
|---|---|
| 0010 | RGB color |
| 0011 | 24 bit color |
| 4610 | Light is a Spot light |
| 4620 | Light is off/on ( Boolean ) |

* 4610 - Spot Light

| start | end | size | type | name |
|---|---|---|---|---|
| 0 | 3 | 4 | float | Target pos X |
| 4 | 7 | 4 | float | Target pos X |
| 8 | 11 | 4 | float | Target pos X |
| 12 | 15 | 4 | float | Hotspot |
| 16 | 19 | 4 | float | Falloff |

* 0010 - RGB Color

| start | end | size | type | name |
|---|---|---|---|---|
| 0 | 3 | 4 | float | Red |
| 4 | 7 | 4 | float | Green |
| 8 | 11 | 4 | float | Blue |

* 0011 - RGB Color - 24 bit

| start | end | size | type | name |
|---|---|---|---|---|
| 0 | 1 | 1 | byte | Red |
| 1 | 1 | 1 | byte | Green |
| 2 | 2 | 1 | byte | Blue |

* 4700 - Camera

Describes the details of the camera in the scene

| start | end | size | type | name |
|---|---|---|---|---|
| 0 | 3 | 4 | float | Camera pos X |
| 4 | 7 | 4 | float | Camera pos Y |
| 8 | 11 | 4 | float | Camera pos Z |
| 12 | 15 | 4 | float | Camera target X |
| 16 | 19 | 4 | float | Camera target X |
| 20 | 23 | 4 | float | Camera target X |
| 24 | 27 | 4 | float | Camera bank ( rotation angle ) |
| 28 | 31 | 4 | float | Camera lens |

## 4. Keyframer Chunks

* Keyframer chunk

| id | Description |
|---|---|
| B00A | unknown |
| 7001 | See first description of this chunk |
| B008 | Frames |
| B009 | unknown |
| B002 | Start object description |

* B008 - Frame information

simple structure describing frame info

| start | end | size | type | name |
|---|---|---|---|---|
| 0 | 3 | 4 | unsigned long | start frame |
| 4 | 7 | 4 | unsigned long | end frame |

*B002 - Start of Object info

Subhunks

| id | Description |
|---|---|

| | |
|---|---|
| B010 | Name & Hierarchy |
| B011* | Name Dummy Object |
| B013 | unknown |
| B014* | unknown |
| B015 | unknown |
| B020 | Objects pivot point ? |
| B021 | unknown |
| B022 | unknown |

* B010 - Name & Hierarchy descriptor

| start | end | size | type | name |
|---|---|---|---|---|
| 0 | ? | ? | ASCIIZ | Object name |
| ? | ? | 2 | unsigned int | unknown |
| ? | ? | 2 | unsigned int | unknown |
| ? | ? | 2 | unsigned int | Hierarchy of Object |

The object hierarchy is a bit complex but works like this.
Each Object in the scene is given a number to identify its
order in the tree. Also each object is orddered in the 3ds
file as it would appear in the tree.
The root object is given the number -1 ( FFFF ).
As the file is read a counter of the object number is kept.
Is the counter increments the objects are children of the
previous objects. But when the pattern is broken by a number
what will be less than the current counter the hierarchy returns
to that level.

for example.

| object<br>name | hierarchy |
|---|---|
| A | -1 |
| B | 0 |
| C | 1 |
| D | 2 |
| E | 1 |
| F | 4 |
| G | 5 |
| H | 1 |
| I | 7 |
| J | 8 |
| K | 0 |
| L | 10 |
| M | 11 |
| N | 0 |
| O | 13 |
| P | 14 |

This example is taken
from 50pman.3ds

I would really reccomend
having a look at one of the
examples with the hierarchy
numbers to help work it out.

```
                     A
     +-----------------+----------------+
     B                 K                N
+----+----+            |                |
C    E    H            L                O
|    |    |            |                |
D    F    I            M                P
     |    |
     G    J
```

Still not done with this chunk yet !
If the object name is $$$DUMMY then it is a dummy object
and therefore you should expect a few extra chunks.

* B011 - Dummy objects name.

Names a dummy object. ASCIIZ string.

* B020 - Pivot Point ?

The Objects pivot point. Not quite sure what the first five floats do yet
( ideas ?).

| start | end | size | type | name |
|---|---|---|---|---|
| 0 | 3 | 4 | float | unknown |
| 4 | 7 | 4 | float | unknown |
| 8 | 11 | 4 | float | unknown |
| 12 | 16 | 4 | float | unknown |
| 16 | 19 | 4 | float | unknown |
| 20 | 23 | 4 | float | unknown |

```
24   27    4  float  Pivot Y
28   32    4  float  Pivot X
```

```
                    5. CODE
          =============================
```

```
!!!!!!!!!!! The source code is not UP-TO-DATE it is written after rev 0.9
of this Doc It will be updated when I can find the time ( or maybe YOU can)
If the code looks like it has been written by a a twelve year old, then
looks do decieve, I like very simple and easy to read source code. All that
matters is that it does the trick.

3dsbin-read.c by Martin van Velsen.
```

Blender 3DS File Format

Hipocrisy of the finest:

"I agree that no single company can create all the hardware and software. Openness is central because it's the foundation of choice."
-- *Steve Balmer (Microsoft) blaming Apple regarding iPhone, February 18, 2009*

"Things work better when hardware and software are considered together, [..]. We control it all, we design it all, and we manufacture it all ourselves."
-- *Steve Balmer announcing Windows 8 Tablet, June 19, 2012*

Last update 1998/10/05

Top of Page

The Labs.Com