# POD File Format

# Specification

| | | |
|---|---|---|
| Filename | : | POD File Format.Specification |
| Version | : | PowerVR SDK REL_3.2@2607465a External Issue |
| Issue Date | : | 10 Sep 2013 |
| Author | : | Imagination Technologies Limited |

# Contents

# List of Figures

# 1. Introduction

## 1.1. Document Overview

The purpose of this document is to act as a specification for the POD file format.

## 1.2. Specification Version

POD specification version:

2.1

# 2. Format Description

## 2.1. File Overview

Each POD file is laid out as follows:

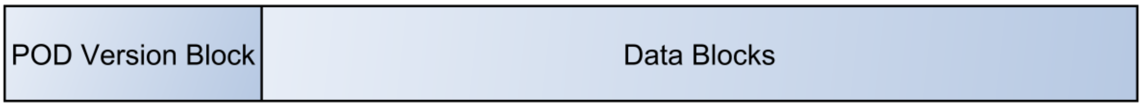

**Figure 2-1 POD File Layout**

Each block within the file takes the following format:

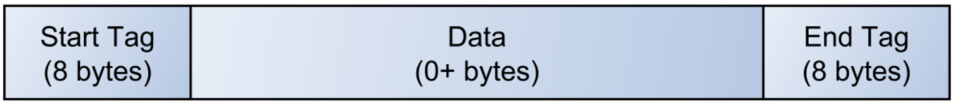

**Figure 2-2 Block Layout**

The ‘Start Tag’ and ‘End Tag’ share a structure, this structure is split into two DWORDs in the following format:

| DWORD | Bit | Symbol | Description |
|---|---|---|---|
| 0 | 31 | Start/End | Bit Value = 0 – This tag is the beginning of a block<br>Bit Value = 1 – This tag is the end of a block |
| | 0 – 30 | Identifier | Block Type Identifier |
| 1 | 0 – 31 | Length | The length of ‘Data’ in bytes. |

It is important to note that ‘Data’ may contain blocks, which may, in turn, contain further blocks, and so on and so forth. It should also be noted that a block that contains only further nested blocks between its ‘Start’ and ‘End’ tags will have a ‘Length’ of zero.

## 2.2. Reading POD

The algorithm for reading a POD file is follows:

```
While not at end-of-file
        Read 8 bytes from file
                If 'Identifier' is a valid 'Start Tag'
                        Read 'Length' number of bytes of 'Data'.
                                Handle ‘Data’
                        Go down a level in nested structure
                Else if 'Identifier' is valid 'End Tag':
                        Read 'Length' number of bytes of 'Data'.
                                Handle ‘Data’
                        Go up a level in nested structure
```

# 3. Block List

## 3.1. Main Blocks

### 3.1.1. Version

**Identifier**

- 1000

**Description**

A null terminated character string containing the following:

- "AB.POD.2.0"

### 3.1.2. Export Options

**Identifier**

- 1002

**Description**

A null terminated character array containing the options used to export the POD file. The contents of this string are implementation specific from exporter to exporter and are primarily used to allow an exporter to re-read the options used in an existing POD file.

### 3.1.3. History

**Identifier**

- 1003

**Description**

A null terminated character array containing the history of the POD file. The exact contents of this string are implementation specific from exporter to exporter. Its use is informational only.

### 3.1.4. Scene

**Identifier**

- 1001

**Description**

The overall description of the POD file scene.

**Data**

| Name | Description |
|---|---|
| Clear Colour | Clear colour of the scene<br>This item is stored as a 'Block', see Section 3.2.1 Clear Colour |
| Ambient Colour | Ambient colour of the scene<br>This item is stored as a 'Block', see Section 3.2.2 Ambient Colour |
| Num. Cameras | Number of cameras in the scene<br>This item is stored as a 'Block', see Section 3.2.3 Num. Cameras |
| Num. Lights | Number of lights in the scene<br>This item is stored as a 'Block', see Section 3.2.4 Num. Lights |
| Num. Meshes | Number of meshes in the mesh array<br>This item is stored as a 'Block', see Section 3.2.5 Num. Meshes |
| Num. Nodes | Number of nodes in the scene<br>This item is stored as a 'Block', see Section 3.2.6 Num. Nodes |
| Num. Mesh Nodes | The total number of meshes in the scene (this may be larger than 'Num. Meshes' as this number will include instanced meshes)<br>This item is stored as a 'Block', see Section 3.2.7 Num. Mesh Nodes |
| Num. Textures | Number of textures in the scene<br>This item is stored as a 'Block', see Section 3.2.8 Num. Textures |
| Num. Frames | Number of frames of animation in the scene<br>This item is stored as a 'Block', see Section 3.2.10 Num. Frames |
| Num. Materials | Number of materials in the scene<br>This item is stored as a 'Block', see Section 3.2.9 Num. Materials |
| Camera | Specifies all the information relating to a single camera within the scene.<br>This block may appear multiple times.<br>This item is stored as a 'Block', see Section 3.2.11 Camera |
| Light | Specifies all the information relating to a single light within the scene.<br>This block may appear multiple times.<br>This item is stored as a 'Block', see Section 3.2.12 Light |
| Mesh | Specifies all the information relating to a single mesh within the scene.<br>This block may appear multiple times.<br>This item is stored as a 'Block', see Section 3.2.13 Mesh |
| Node | Specifies all the information relating to a single node within the scene.<br>This block may appear multiple times.<br>This item is stored as a 'Block', see Section 3.2.14 Node |

| Name | Description |
|---|---|
| Texture | Specifies all the information relating to a single texture within the scene.<br>This block may appear multiple times.<br>This item is stored as a ‘Block’, see Section 3.2.15 Texture |
| Material | Specifies all the information relating to a single material within the scene.<br>This block may appear multiple times.<br>This item is stored as a ‘Block’, see Section 3.2.16 Material |
| Scene Flags | Specifies whether a number of flags are set within the POD file.<br>This item is stored as a ‘Block’, see Section 3.2.17 Scene Flags |
| FPS | Specifies the animation speed of the scene, in frames per second.<br>This item is stored as a ‘Block’, see Section 3.2.18 FPS |
| Scene User Data | Custom data added by the exporter.<br>This item is stored as a ‘Block’, see Section 3.2.19 Scene User Data |

## 3.2. Scene Blocks

### 3.2.1. Clear Colour

**Identifier**

- 2000

**Description**

The channel values of the scenes ‘clear colour’ in the order RGB.

**Data**

| Name | Data Type | Description |
|---|---|---|
| Red Channel | Float/Fixed | A 4 byte float/fixed describing the value of the red channel in the range of [0 - 1]. |
| Green Channel | Float/Fixed | A 4 byte float/fixed describing the value of the green channel in the range of [0 - 1]. |
| Blue Channel | Float/Fixed | A 4 byte float/fixed describing the value of the blue channel in the range of [0 - 1]. |

### 3.2.2. Ambient Colour

**Identifier**

- 2001

**Description**

The channel values of the scene’s ambient colour in the order RGB.

**Data**

| Name | Data Type | Description |
|---|---|---|
| Red Channel | Float/Fixed | A 4 byte float/fixed describing the value of the red channel in the range of [0 - 1]. |
| Green Channel | Float/Fixed | A 4 byte float/fixed describing the value of the green channel in the range of [0 - 1]. |
| Blue Channel | Float/Fixed | A 4 byte float/fixed describing the value of the blue channel in the range of [0 - 1]. |

### 3.2.3. Num. Cameras

**Identifier**

- 2002

**Description**

The number of cameras in the scene.

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Cameras | unsigned 32bit Integer | 4 bytes, stored as an unsigned 32bit integer, representing the number of cameras within the scene. |

### 3.2.4. Num. Lights

**Identifier**

- 2003

**Description**

The number of lights in the scene.

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Lights | unsigned 32bit Integer | 4 bytes, stored as an unsigned 32bit integer, representing the number of lights within the scene. |

### 3.2.5. Num. Meshes

**Identifier**

- 2004

**Description**

The number of meshes in the scene.

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Meshes | unsigned 32bit Integer | 4 bytes, stored as an unsigned 32bit integer, representing the number of meshes within the scene. |

### 3.2.6. Num. Nodes

**Identifier**

- 2005

**Description**

The number of nodes in the scene.

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Nodes | unsigned 32bit Integer | 4 bytes, stored as an unsigned 32bit integer, representing the number of nodes within the scene. |

### 3.2.7. Num. Mesh Nodes

**Identifier**

- 2006

**Description**

The total number of meshes in the scene (this may be larger than ‘Num. Meshes’ as this number will include instanced meshes).

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Mesh Nodes | unsigned 32bit Integer | 4 bytes, stored as an unsigned 32bit integer, representing the total number of meshes in the scene (this may be larger than ‘Num. Meshes’ as this number will include instanced meshes) |

### 3.2.8. Num. Textures

**Identifier**

- 2007

**Description**

The number of textures in the scene.

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Textures | unsigned 32bit Integer | 4 bytes, stored as an unsigned 32bit integer, representing the number of textures within the scene. |

### 3.2.9. Num. Materials

**Identifier**

- 2008

**Description**

The number of materials in the scene.

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Lights | unsigned 32bit Integer | 4 bytes, stored as an unsigned 32bit integer, representing the number of materials within the scene. |

### 3.2.10. Num. Frames

**Identifier**

- 2009

**Description**

The number of frame of animation for the scene.

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Frames | unsigned 32bit Integer | 4 bytes, stored as an unsigned 32bit integer, representing the number of number of frames of animation for the scene. |

### 3.2.11. Camera

**Identifier**

- 2010

**Description**

Contains all the information pertaining to a single camera within the scene.

**Data**

| Name | Description |
|---|---|
| Target Object Index | The index into the node array of the object the camera should target.<br>This item is stored as a 'Block', see Section 3.8.1 Target Object Index |
| Field of View | The FOV of the camera.<br>This item is stored as a 'Block', see Section 3.8.2 Field of View |
| Far Plane | The location of the far plane for the camera.<br>This item is stored as a 'Block', see Section 3.8.3 Far Plane |
| Near Plane | The location of the near plane for the camera.<br>This item is stored as a 'Block', see Section 3.8.4 Near Plane |
| FOV Animation | The FOV for each frame of animation, for use with FOV animation.<br>This item is stored as a 'Block', see Section 3.8.5 FOV Animation |

### 3.2.12. Light

**Identifier**

- 2011

**Description**

Contains all the information pertaining to a single light within the scene.

**Data**

| Name | Description |
|---|---|
| Target Object Index | The index into the node array of the object the light should target.<br>This item is stored as a 'Block', see Section 3.7.1 Target Object Index |
| Light Colour | The colour of the light.<br>This item is stored as a 'Block', see Section 3.7.2 Light Colour |
| Light Type | The type of the light (e.g. Point, Directional, Spot etc.)<br>This item is stored as a 'Block', see Section 3.7.3 Light Type |
| Constant Attenuation | The constant attenuation of the light.<br>This item is stored as a 'Block', see Section 3.7.4 Constant Attenuation |
| Linear Attenuation | The linear attenuation of the light.<br>This item is stored as a 'Block', see Section 3.7.5 Linear Attenuation |
| Quadratic Attenuation | The quadratic attenuation of the light.<br>This item is stored as a 'Block', see Section 3.7.6 Quadratic Attenuation |
| Falloff Angle | The falloff angle of the light (in radians).<br>This item is stored as a 'Block', see Section 3.7.7 Falloff Angle |
| Falloff Exponent | The falloff exponent of the light.<br>This item is stored as a 'Block', see Section 3.7.8 Falloff Exponent |

### 3.2.13. Mesh

**Identifier**

- 2012

**Description**

Contains all the information pertaining to a single mesh within the scene.

**Data**

| Name | Description |
|---|---|
| Num. Vertices | The number of vertices in the mesh.<br>This item is stored as a 'Block', see Section 3.6.1 Num. Vertices |
| Num. Faces | The number of triangles in the mesh.<br>This item is stored as a 'Block', see Section 3.6.2 Num. Faces |
| Num. UVW Channels | The number of texture coordinate channels in the mesh.<br>This item is stored as a 'Block', see Section 3.6.3 Num. UVW Channels |
| Vertex Index List | The list of vertex indices for the faces in an indexed mesh.<br>This item is stored as a 'Block', see Section 3.6.4 Vertex Index List |
| Strip Length | A list, one entry per strip, of the number of triangles within each strip.<br>This item is stored as a 'Block', see Section 3.6.5 Strip Length |
| Num. Strips | The total number of strips.<br>This item is stored as a 'Block', see Section 3.6.6 Num. Strips |
| Vertex List | The list of vertices, when data is interleaved this will contain and offset into the 'Interleaved Data List' and a stride for moving from element to element.<br>This item is stored as a 'Block', see Section 3.6.7 Vertex List |
| Normal List | The list of normals, when data is interleaved this will contain and offset into the 'Interleaved Data List' and a stride for moving from element to element.<br>This item is stored as a 'Block', see Section 3.6.8 Normal List |
| Tangent List | The list of tangents, when data is interleaved this will contain and offset into the 'Interleaved Data List' and a stride for moving from element to element.<br>This item is stored as a 'Block', see Section 3.6.9 Tangent List |
| Binormal List | The list of binormals, when data is interleaved this will contain and offset into the 'Interleaved Data List' and a stride for moving from element to element.<br>This item is stored as a 'Block', see Section 3.6.10 Binormal List |
| UVW List | The list of texture coordinates, when data is interleaved this will contain and offset into the 'Interleaved Data List' and a stride for moving from element to element.<br>This will appear a number of times equal to 'Num. UVW Channels'.<br>This item is stored as a 'Block', see Section 3.6.11 UVW List |
| Vertex Colour List | A list of colours per vertex.<br>This item is stored as a 'Block', see Section 3.6.12 Vertex Colour List |
| Bone Index List | A list of indices into the Bone Batch Index List detailing which matrices should affect which vertex, with a number of indices per vertex equal to the number of bones.<br>This item is stored as a 'Block', see Section 3.6.13 Bone Index List |
| Bone Weights | The weight for each bone reference in the 'Bone Index List'.<br>This item is stored as a 'Block', see Section 3.6.14 Bone Weights |

| Name | Description |
|---|---|
| Bone Batch Index List | A list of indices into the 'Node' list, each 'Node' representing the transformations associated with a single bone. (Read via 'Bone Index List').<br>This item is stored as a 'Block', see Section 3.6.15 Bone Batch Index List |
| Num. Bone Indices per Batch | A number of integers equal to 'Num. Bone Batches' that state how many bones exist within each batch.<br>This item is stored as a 'Block', see Section 3.6.16 Num. Bone Indices per Batch |
| Bone Offset per Batch | A number of integers equal to 'Num. Bone Batches' that state the offset into the 'Vertex Index List' for each sub-mesh that uses the given bone batch.<br>This item is stored as a 'Block', see Section 3.6.17 Bone Offset per Batch |
| Max. Num. Bones per Batch | The maximum number bones any given bone batch can contain.<br>This item is stored as a 'Block', see Section 3.6.18 Max. Num. Bones per Batch |
| Num. Bone Batches | The total number of bone batches used in the mesh.<br>This item is stored as a 'Block', see Section 3.6.19 Num. Bone Batches |
| Unpack Matrix | A matrix used for unpacking scaled vertex data.<br>This item is stored as a 'Block', see Section 3.6.20 Unpack Matrix |
| Interleaved Data List | The list of all vertex data interleaved, read using the offsets and strides mentioned above.<br>This item is stored as a 'Block', see Section 3.6.21 Interleaved Data List |

### 3.2.14. Node

**Identifier**

- 2013

**Description**

Contains all the information pertaining to a single node within the scene.

**Data**

| Name | Description |
|---|---|
| Node Index | The index of the node into the mesh, light, or camera array, as appropriate.<br>This item is stored as a 'Block', see Section 3.5.1 Node Index |
| Node Name | The name of the object.<br>This item is stored as a 'Block', see Section 3.5.2 Node Name |
| Material Index | The index of the material used on this mesh, if the node is a mesh.<br>This item is stored as a 'Block', see Section 3.5.3 Material Index |
| Parent Index | The index of this objects parent in the node array.<br>This item is stored as a 'Block', see Section 3.5.4 Parent Index |
| Animation Flags | A flag variable that is used to determine the forms of animation the node contains, if any.<br>This item is stored as a 'Block', see Section 3.5.5 Animation Flags |
| Animation Position Index | A list of indices into 'Animation Position', one per frame, used for indexing animation.<br>This item is stored as a 'Block', see Section 3.5.6 Animation Position Index |
| Animation Position | A list of position animations, in the form of three floats (x, y, z order), per frame when not indexed or applied in the order given by 'Animation Position Index' when indexed.<br>This item is stored as a 'Block', see Section 3.5.7 Animation Position |
| Animation Rotation Index | A list of indices into 'Animation Rotation', one per frame, used for indexing animation.<br>This item is stored as a 'Block', see Section 3.5.8 Animation Rotation Index |
| Animation Rotation | A list of rotation animations, in the form of a quaternion, per frame when not indexed or applied in the order given by 'Animation Rotation Index' when indexed.<br>This item is stored as a 'Block', see Section 3.5.9 Animation Rotation |
| Animation Scale Index | A list of indices into 'Animation Scale', one per frame, used for indexing animation.<br>This item is stored as a 'Block', see Section 3.5.10 Animation Scale Index |
| Animation Scale | A list of scaling animations, in the form of seven floats (x, y, z, x-axis, y-axis, z-axis, stretch rotation), per frame when not indexed or applied in the order given by 'Animation Scale Index' when indexed. X-Axis, Y-Axis, Z-Axis and Stretch Rotation are used to convert the object into the axes the scaling is performed in.<br>This item is stored as a 'Block', see Section 3.5.11 Animation Scale |
| Animation Matrix Index | A list of indices into 'Animation Matrix', one per frame, used for indexing animation.<br>This item is stored as a 'Block', see Section 3.5.12 Animation Matrix Index |

| Name | Description |
|---|---|
| Animation Matrix | A list of matrix animations, in the form of sixteen floats (4x4), per frame when not indexed or applied in the order given by ‘Animation Matrix Index’ when indexed.<br>Matrices are stored ‘Row Major’ in memory, and used ‘Column Major’ mathematically.<br>This item is stored as a ‘Block’, see Section 3.5.13 Animation Matrix |
| Node User Data | Custom data added by the exporter.<br>This item is stored as a ‘Block’, see Section 3.5.14 Node User Data |

### 3.2.15. Texture

**Identifier**

- 2014

**Description**

Contains all the information pertaining to a single texture within the scene.

**Data**

| Name | Description |
|---|---|
| Texture Name | The filename of the texture.<br>This item is stored as a 'Block', see Section 3.4.1 Texture Name |

### 3.2.16. Material

**Identifier**

- 2015

**Description**

Contains all the information pertaining to a single material within the scene.

**Data**

| Name | Description |
|---|---|
| Material Name | The name of the material.<br>This item is stored as a 'Block', see Section 3.3.1 Material Name |
| Diffuse Texture Index | The index of the diffuse texture into the scenes texture list.<br>This item is stored as a 'Block', see Section 3.3.2 Diffuse Texture Index |
| Ambient Texture Index | The index of the ambient texture into the scenes texture list.<br>This item is stored as a 'Block', see Section 3.3.3 Ambient Texture Index |
| Specular Colour Texture Index | The index of the specular colour texture into the scenes texture list.<br>This item is stored as a 'Block', see Section 3.3.4 Specular Colour Texture Index |
| Specular Level Texture Index | The index of the specular level texture into the scenes texture list.<br>This item is stored as a 'Block', see Section 3.3.5 Specular Level Texture Index |
| Bump Map Texture Index | The index of the bump map texture into the scenes texture list.<br>This item is stored as a 'Block', see Section 3.3.6 Bump Map Texture Index |
| Emissive Texture Index | The index of the emissive texture into the scenes texture list.<br>This item is stored as a 'Block', see Section 3.3.7 Emissive Texture Index |
| Glossiness Texture Index | The index of the glossiness texture into the scenes texture list.<br>This item is stored as a 'Block', see Section 3.3.8 Glossiness Texture Index |
| Opacity Texture Index | The index of the opacity texture into the scenes texture list.<br>This item is stored as a 'Block', see Section 3.3.9 Opacity Texture Index |
| Reflection Texture Index | The index of the reflection texture into the scenes texture list.<br>This item is stored as a 'Block', see Section 3.3.10 Reflection Texture Index |
| Refraction Texture Index | The index of the refraction texture into the scenes texture list.<br>This item is stored as a 'Block', see Section 3.3.11 Refraction Texture Index |
| Material Opacity | The opacity of the material.<br>This item is stored as a 'Block', see Section 3.3.12 Material Opacity |
| Ambient Colour | The ambient colour of the material.<br>This item is stored as a 'Block', see Section 3.3.13 Ambient Colour |
| Diffuse Colour | The diffuse colour of the material.<br>This item is stored as a 'Block', see Section 3.3.14 Diffuse Colour |
| Specular Colour | The specular colour of the material.<br>This item is stored as a 'Block', see Section 3.3.15 Specular Colour |
| Shininess | The shininess of the material.<br>This item is stored as a 'Block', see Section 3.3.16 Shininess |

| Name | Description |
|---|---|
| Effect File Name | The name of the effect file used by the material.<br>This item is stored as a ‘Block’, see Section 3.3.17 Effect File Name |
| Effect Name | The name of the effect within the file ‘Effect File Name’<br>This item is stored as a ‘Block’, see Section 3.3.18 Effect Name |
| Blending RGB Source Value | The first RGB data source, with an optional pre-blend operation.<br>This item is stored as a ‘Block’, see Section 3.3.19 Blending RGB Source Value |
| Blending Alpha Source Value | The first alpha data source, with an optional pre-blend operation.<br>This item is stored as a ‘Block’, see Section 3.3.20 Blending Alpha Source Value |
| Blending RGB Destination Value | The second RGB data source, with an optional pre-blend operation.<br>This item is stored as a ‘Block’, see Section 3.3.21 Blending RGB Destination Value |
| Blending Alpha Destination Value | The second alpha data source, with an optional pre-blend operation.<br>This item is stored as a ‘Block’, see Section 3.3.22 Blending Alpha Destination Value |
| Blending RGB Operation | The blending operation defining how the materials RGB data sources should be combined.<br>This item is stored as a ‘Block’, see Section 3.3.23 Blending RGB Operation |
| Blending Alpha Operation | The blending operation defining how the materials alpha data sources should be combined.<br>This item is stored as a ‘Block’, see Section 3.3.24 Blending Alpha Operation |
| Blending RGBA Colour | An RGBA colour used with some blend operations.<br>This item is stored as a ‘Block’, see Section 3.3.25 Blending RGBA Colour |
| Blending Factor Array | A factor value for used with some blend operations.<br>This item is stored as a ‘Block’, see Section 3.3.26 Blending Factor Array |
| Material Flags | Specifies whether a number of flags are set within the POD file.<br>This item is stored as a ‘Block’, see Section 3.3.27 Material Flags |
| Material User Data | Custom data added by the exporter.<br>This item is stored as a ‘Block’, see Section 3.3.28 Material User Data |

### 3.2.17. Scene Flags

**Identifier**

- 2016

**Description**

Specifies whether a number of flags are set within the POD file.

**Data**

| Name | Data Type | Description |
|---|---|---|
| Scene Flags | unsigned 32bit integer | Specifies whether a number of flags are set within the POD file.<br>The values are as follows:<br>• 0x00000001 – The Fixed 16.16 data type is used |

### 3.2.18. FPS

**Identifier**

- 2017

**Description**

Specifies the animation speed of the scene, in frames per second.

**Data**

| Name | Data Type | Description |
|---|---|---|
| FPS | unsigned 32bit integer | Specifies the animation speed of the scene, in frames per second. |

### 3.2.19. Scene User Data

**Identifier**

- 2018

**Description**

Custom data added by the exporter.

**Data**

| Name | Data Type | Description |
|---|---|---|
| User Data | Variable | Custom data added by the exporter. The format of the data is undefined. |

### 3.2.20. Units

**Identifier**

- 2019

**Description**

Specifies the number of metres a single unit of length represents (e.g. a vertex at an x, y, z coordinate of 0, 0, 0 moving to 0, 0, 1 has moved 1 unit in the Z-axis)

**Data**

| Name | Data Type | Description |
|---|---|---|
| User Data | Variable | Custom data added by the exporter. The format of the data is undefined. |

## 3.3. Material Blocks

### 3.3.1. Material Name

**Identifier**

- 3000

**Data**

| Name | Data Type | Description |
|---|---|---|
| Material Name | null terminated character array | The name of the material. |

### 3.3.2. Diffuse Texture Index

**Identifier**

- 3001

**Data**

| Name | Data Type | Description |
|---|---|---|
| Diffuse Texture Index | signed 32bit integer | An unsigned 32bit integer representing the index of the diffuse texture into the scenes texture list. |

### 3.3.3. Ambient Texture Index

**Identifier**

- 3009

**Data**

| Name | Data Type | Description |
|---|---|---|
| Ambient Texture Index | signed 32bit integer | An unsigned 32bit integer representing the index of the ambient texture into the scenes texture list. |

### 3.3.4. Specular Colour Texture Index

**Identifier**

- 3010

**Data**

| Name | Data Type | Description |
|---|---|---|
| Specular Colour Texture Index | signed 32bit integer | An unsigned 32bit integer representing the index of the specular colour texture into the scenes texture list. |

### 3.3.5. Specular Level Texture Index

**Identifier**

- 3011

**Data**

| Name | Data Type | Description |
|---|---|---|
| Specular Level Texture Index | signed 32bit integer | An unsigned 32bit integer representing the index of the specular level texture into the scenes texture list. |

### 3.3.6. Bump Map Texture Index

**Identifier**

- 3012

**Data**

| Name | Data Type | Description |
|---|---|---|
| Bump Map Texture Index | signed 32bit integer | An unsigned 32bit integer representing the index of the bump map texture into the scenes texture list. |

### 3.3.7. Emissive Texture Index

**Identifier**

- 3013

**Data**

| Name | Data Type | Description |
|---|---|---|
| Emissive Texture Index | signed 32bit integer | An unsigned 32bit integer representing the index of the emissive texture into the scenes texture list. |

### 3.3.8. Glossiness Texture Index

**Identifier**

- 3014

**Data**

| Name | Data Type | Description |
|---|---|---|
| Glossiness Texture Index | signed 32bit integer | An unsigned 32bit integer representing the index of the glossiness texture into the scenes texture list. |

### 3.3.9. Opacity Texture Index

**Identifier**

- 3015

**Data**

| Name | Data Type | Description |
|---|---|---|
| Opacity Texture Index | signed 32bit integer | An unsigned 32bit integer representing the index of the opacity texture into the scenes texture list. |

### 3.3.10. Reflection Texture Index

**Identifier**

- 3016

**Data**

| Name | Data Type | Description |
|---|---|---|
| Reflection Texture Index | signed 32bit integer | An unsigned 32bit integer representing the index of the reflection texture into the scenes texture list. |

### 3.3.11. Refraction Texture Index

**Identifier**

- 3017

**Data**

| Name | Data Type | Description |
|---|---|---|
| Refraction Texture Index | signed 32bit integer | An unsigned 32bit integer representing the index of the refraction texture into the scenes texture list. |

### 3.3.12. Material Opacity

**Identifier**

- 3002

**Data**

| Name | Data Type | Description |
|---|---|---|
| Material Opacity | Float/Fixed | The opacity of the material. |

### 3.3.13. Ambient Colour

**Identifier**

- 3003

**Data**

| Name | Data Type | Description |
|---|---|---|
| Ambient Colour | Float/Fixed | The ambient colour of the material, three channels, in the order RGB. |

### 3.3.14. Diffuse Colour

**Identifier**

- 3004

**Data**

| Name | Data Type | Description |
|---|---|---|
| Diffuse Colour | Float/Fixed | The diffuse colour of the material, three channels, in the order RGB. |

### 3.3.15. Specular Colour

**Identifier**

- 3005

**Data**

| Name | Data Type | Description |
|---|---|---|
| Specular Colour | Float/Fixed | The specular colour of the material, three channels, in the order RGB. |

### 3.3.16. Shininess

**Identifier**

- 3006

**Data**

| Name | Data Type | Description |
|---|---|---|
| Shininess | Float/Fixed | The shininess of the material. |

### 3.3.17. Effect File Name

**Identifier**

- 3007

**Data**

| Name | Data Type | Description |
|---|---|---|
| Effect File Name | null terminated character array | The name of the effect file used by the material. |

### 3.3.18. Effect Name

**Identifier**

- 3008

**Data**

| Name | Data Type | Description |
|---|---|---|
| Effect Name | null terminated character array | The name of the effect in the effect file used by the material. |

### 3.3.19. Blending RGB Source Value

**Identifier**

- 3018

**Data**

| Name | Data Type | Description |
|---|---|---|
| Blending RGB Source Value | unsigned 32bit integer | The first RGB data source, with an optional pre-blend operation. The following values are valid operations:<br>• 0x0300 - SRC_COLOR<br>• 0x0301 - ONE_MINUS_SRC_COLOR |

### 3.3.20. Blending Alpha Source Value

**Identifier**

- 3019

**Data**

| Name | Data Type | Description |
|---|---|---|
| Blending Alpha Source Value | unsigned 32bit integer | The first alpha data source, with an optional pre-blend operation. The following values are valid operations:<br>• 0x0302 - SRC_ALPHA<br>• 0x0303 - ONE_MINUS_ALPHA |

### 3.3.21. Blending RGB Destination Value

**Identifier**

- 3020

**Data**

| Name | Data Type | Description |
|---|---|---|
| Blending RGB Destination Value | unsigned 32bit integer | The second RGB data source, with an optional pre-blend operation. The following values are valid operations:<br>• 0x0306 - DST_COLOR<br>• 0x0307 - ONE_MINUS_DST_COLOR |

### 3.3.22. Blending Alpha Destination Value

**Identifier**

- 3021

**Data**

| Name | Data Type | Description |
|---|---|---|
| Blending RGB Destination Value | unsigned 32bit integer | The second RGB data source, with an optional pre-blend operation. The following values are valid operations:<br>• 0x0304 - DST_ALPHA<br>• 0x0305 - ONE_MINUS_DST_ALPHA |

### 3.3.23. Blending RGB Operation

**Identifier**

- 3022

**Data**

| Name | Data Type | Description |
|---|---|---|
| Blending RGB Operation | unsigned 32bit integer | The blending operation defining how the materials RGB data sources should be combined. Valid values are:<br>• 0 – ZERO<br>• 1 – ONE<br>• 2 – BLEND_FACTOR<br>• 3 – ONE_MINUS_BLEND_FACTOR<br>• 0x8001 – CONSTANT_COLOUR<br>• 0x8002 – ONE_MINUS_CONSTANT_COLOUR<br>• 0x8006 – ADD<br>• 0x8007 – MIN<br>• 0x8008 – MAX<br>• 0x800a – SUBTRACT<br>• 0x800b – REVERSE_SUBTRACT |

### 3.3.24. Blending Alpha Operation

**Identifier**

- 3023

**Data**

| Name | Data Type | Description |
|---|---|---|
| Blending Alpha Operation | unsigned 32bit integer | The blending operation defining how the materials alpha data sources should be combined. Valid values are:<br>• 0 – ZERO<br>• 1 – ONE<br>• 2 – BLEND_FACTOR<br>• 3 – ONE_MINUS_BLEND_FACTOR<br>• 0x8003 – CONSTANT_ALPHA<br>• 0x8004 – ONE_MINUS_CONSTANT_ALPHA<br>• 0x8006 – ADD<br>• 0x8007 – MIN<br>• 0x8008 – MAX<br>• 0x800a – SUBTRACT<br>• 0x800b – REVERSE_SUBTRACT |

### 3.3.25. Blending RGBA Colour

**Identifier**

- 3024

**Data**

| Name | Data Type | Description |
|---|---|---|
| Blending RGBA Colour | Float/Fixed | An RGBA colour used with some blend operations, in the form of four floats in the order RGBA. |

### 3.3.26. Blending Factor Array

**Identifier**

- 3025

**Data**

| Name | Data Type | Description |
|---|---|---|
| Blending Factor Array | Float/Fixed | A list of blend factors, one per colour in 'Blending RGBA Colour' used for some blending operations. |

### 3.3.27. Material Flags

**Identifier**

- 3026

**Data**

| Name | Data Type | Description |
|---|---|---|
| Material Flags | unsigned 32bit integer | Specifies whether a number of flags are set within the POD file.. The values are as follows:<br>• 0x01 – Blending enabled<br>• 0x00 – Blending disabled |

### 3.3.28. Material User Data

**Identifier**

- 3027

**Data**

| Name | Data Type | Description |
|---|---|---|
| User Data | Variable | Custom data added by the exporter. The format of the data is undefined. |

## 3.4. Texture Blocks

### 3.4.1. Texture Name

**Identifier**

- 4000

**Data**

| Name | Data Type | Description |
|---|---|---|
| Texture Name | null terminated character array | The name of the texture file (file path not included). |

## 3.5. Node Blocks

### 3.5.1. Node Index

**Identifier**

- 5000

**Data**

| Name | Data Type | Description |
|---|---|---|
| Node Index | signed 32bit integer | The index of the node into the mesh, light, or camera array, as appropriate. |

### 3.5.2. Node Name

**Identifier**

- 5001

**Data**

| Name | Data Type | Description |
|---|---|---|
| Node Name | null terminated character array | The name of the object. |

### 3.5.3. Material Index

**Identifier**

- 5002

**Data**

| Name | Data Type | Description |
|---|---|---|
| Material Index | signed 32bit integer | The index of the material used on this mesh, if the node is a mesh. |

### 3.5.4. Parent Index

**Identifier**

- 5003

**Data**

| Name | Data Type | Description |
|---|---|---|
| Parent Index | signed 32bit integer | The index of this objects parent in the node array. |

### 3.5.5. Animation Flags

**Identifier**

- 5012

**Data**

| Name | Data Type | Description |
|---|---|---|
| Animation Flags | unsigned 32bit integer | A series of flags that determine which forms of animation are present in the node. Valid flags are:<br>• 0x01 – Position Animation<br>• 0x02 – Rotation Animation<br>• 0x04 – Scale Animation<br>• 0x08 – Matrix Animation |

### 3.5.6. Animation Position Index

**Identifier**

- 5013

**Data**

| Name | Data Type | Description |
|---|---|---|
| Animation Position Index | signed 32bit integer array | A list of indices into 'Animation Position', one per frame, used for indexing animation. |

### 3.5.7. Animation Position

**Identifier**

- 5007

**Data**

| Name | Data Type | Description |
|---|---|---|
| Animation Position | Float/Fixed array | A list of position animations, in the form of three floats (XYZ order), one per frame when not indexed; or applied in the order given by 'Animation Position Index' when indexed, with a maximum number of entries equal to the maximum value within the index. |

### 3.5.8. Animation Rotation Index

**Identifier**

- 5014

**Data**

| Name | Data Type | Description |
|---|---|---|
| Animation Rotation Index | signed 32bit integer array | A list of indices into 'Animation Rotation', one per frame, used for indexing animation. |

### 3.5.9. Animation Rotation

**Identifier**

- 5008

**Data**

| Name | Data Type | Description |
|---|---|---|
| Animation Rotation | Float/Fixed array | A list of rotation animations, in the form of a quaternion, one per frame when not indexed; or applied in the order given by 'Animation Rotation Index' when indexed, with a maximum number of entries equal to the maximum value within the index. |

### 3.5.10. Animation Scale Index

**Identifier**

- 5015

**Data**

| Name | Data Type | Description |
|---|---|---|
| Animation Scale Index | signed 32bit integer array | A list of indices into 'Animation Scale', one per frame, used for indexing animation. |

### 3.5.11. Animation Scale

**Identifier**

- 5009

**Data**

| Name | Data Type | Description |
|---|---|---|
| Animation Scale | Float/Fixed array | A list of rotation animations, in the form of seven floats (x, y, z, x-axis, y-axis, z-axis, and stretch rotation), one per frame when not indexed; or applied in the order given by 'Animation Scale Index' when indexed, with a maximum number of entries equal to the maximum value within the index.<br>X-Axis, Y-Axis, Z-Axis and Stretch Rotation are used to convert the object into the axes the scaling is performed in. |

### 3.5.12. Animation Matrix Index

**Identifier**

- 5016

**Data**

| Name | Data Type | Description |
|---|---|---|
| Animation Matrix Index | signed 32bit integer array | A list of indices into 'Animation Matrix', one per frame, used for indexing animation. |

### 3.5.13. Animation Matrix

**Identifier**

- 5010

**Data**

| Name | Data Type | Description |
|---|---|---|
| Animation Matrix | Float/Fixed array | A list of matrix animations, in the form of sixteen floats (4x4), one per frame when not indexed; or applied in the order given by 'Animation Matrix Index' when indexed, with a maximum number of entries equal to the maximum value within the index.<br>Matrices are stored 'Row Major' in memory, and used 'Column Major' mathematically. |

### 3.5.14. Node User Data

**Identifier**

- 5017

**Description**

Custom data added by the exporter.

**Data**

| Name | Data Type | Description |
|---|---|---|
| User Data | Variable | Custom data added by the exporter. The format of the data is undefined. |

## 3.6. Mesh Blocks

### 3.6.1. Num. Vertices

**Identifier**

- 6000

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Vertices | unsigned 32bit Integer | The number of vertices in the mesh. |

### 3.6.2. Num. Faces

**Identifier**

- 6001

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Faces | unsigned 32bit Integer | The number of faces in the mesh, more specifically, the number of triangles in the mesh. |

### 3.6.3. Num. UVW Channels

**Identifier**

- 6002

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. UVW Channels | unsigned 32bit Integer | The number of texture coordinate channels in the mesh. |

### 3.6.4. Vertex Index List

**Identifier**

- 6003

**Data**

| Name | Description |
|---|---|
| Vertex Index List | The list of vertex indices for the faces in an indexed mesh, in the form of a POD Data block see Section 3.9 POD Data Block. |

### 3.6.5. Strip Length

**Identifier**

- 6004

**Data**

| Name | Data Type | Description |
|---|---|---|
| Strip Length | unsigned 32bit integer array | A list, one entry per strip, of the number of triangles within each strip. |

### 3.6.6. Num. Strips

**Identifier**

- 6005

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Strip | unsigned 32bit integer | The total number of strips. |

### 3.6.7. Vertex List

**Identifier**

- 6006

**Data**

| Name | Description |
|---|---|
| Vertex List | The list of vertices within the mesh, in the form of a POD Data Block, see Section 3.9 POD Data Block. |

### 3.6.8. Normal List

**Identifier**

- 6007

**Data**

| Name | Description |
|---|---|
| Normal List | The list of normals within the mesh, in the form of a POD Data Block, see Section 3.9 POD Data Block. |

### 3.6.9. Tangent List

**Identifier**

- 6008

**Data**

| Name | Description |
|---|---|
| Tangent List | The list of tangents within the mesh, in the form of a POD Data Block, see Section 3.9 POD Data Block. |

### 3.6.10. Binormal List

**Identifier**

- 6009

**Data**

| Name | Description |
|---|---|
| Binormal List | The list of binormals within the mesh, in the form of a POD Data Block, see Section 3.9 POD Data Block. |

### 3.6.11. UVW List

**Identifier**

- 6010

**Data**

| Name | Description |
|---|---|
| UVW List | The list of UVWs within the mesh, in the form of a POD Data Block see, Section 3.9 POD Data Block.<br>This block may appear multiple times, once per set of UVW mappings. |

### 3.6.12. Vertex Colour List

**Identifier**

- 6011

**Data**

| Name | Description |
|---|---|
| Vertex Colour List | The list of vertex colours for each vertex within the mesh, in the form of a POD Data Block, see Section 3.9 POD Data Block. |

### 3.6.13. Bone Index List

**Identifier**

- 6012

**Data**

| Name | Description |
|---|---|
| Bone Index List | A list of indices into the 'Bone Batch Index List' detailing which bones should affect which vertex, in the form of a POD Data Block, see Section 3.9 POD Data Block.<br>The total number of indices is equal to the highest number of bones affecting any vertex within the mesh ('Max. Num. Bones per Batch'), multiplied by the number of vertices:<br>$num.indices = num.bones_{max} * num.vertices$<br>Each vertex has an equal number of indices; indices that are not relevant to a given vertex have the weight that matches the index in question set to zero. |

### 3.6.14. Bone Weights

**Identifier**

- 6013

**Data**

| Name | Description |
|---|---|
| Bone Weights | The weight for each bone reference in the 'Bone Index List' stored as a POD Data Block, see Section 3.9 POD Data Block.<br>The total number of weights is equal to the total number of indices and is in the same order. |

### 3.6.15. Bone Batch Index List

**Identifier**

- 6015

**Data**

| Name | Data Type | Description |
|---|---|---|
| Bone Batch Index List | unsigned 32 bit integer array | A list of indices into the 'Node' list, each indexed 'Node' representing the transformations associated with a single bone. (Read via 'Bone Index List').<br>Each batch within the bone batch index list will be a number of elements long equal to the value of 'Max. Num. Bones per Batch'.<br>For example, if one bone batch contains eight elements (the maximum number of bones per batch), and another three, the three element array will be padded with zeros to eight elements, giving a list of indices 16 elements long.<br>A number of elements from each batch should be read equal to the value in 'Num. Bone Indices per Batch' for that batch. In the above example, the 'Num. Bone Indices per Batch' would contain [8, 3]. Eight indices would be read from the first batch within the list, and three from the second.<br>Finally, there are a number of batches in the 'Bone Batch Index List' equal to the value of 'Num. Bone Batches'. |

### 3.6.16. Num. Bone Indices per Batch

**Identifier**

- 6016

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Bone Indices per Batch | unsigned 32bit integer array | A list of integers, each integer representing the number of indices in each batch in the 'Bone Batch Index List'. |

### 3.6.17. Bone Offset per Batch

**Identifier**

- 6017

**Data**

| Name | Data Type | Description |
|---|---|---|
| Bone Offset per Batch | unsigned 32bit integer array | A list of integers, each integer representing the offset into the 'Vertex List', or 'Vertex Index List' of the data is indexed, the batch starts at.<br>For example, if the list contained [0, 799] the first bone batch would influence vertices 0-798; the second bone batch would influence vertices 799 onwards. |

### 3.6.18. Max. Num. Bones per Batch

**Identifier**

- 6018

**Data**

| Name | Data Type | Description |
|---|---|---|
| Max. Num. Bones per Batch | unsigned 32bit integer | An unsigned 32bit integer representing the maximum number of bones per bone batch. |

### 3.6.19. Num. Bone Batches

**Identifier**

- 6019

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Bone Batches | unsigned 32bit integer | An unsigned 32bit integer representing the number of bone batches in the 'Bone Batch Index List'. |

### 3.6.20. Unpack Matrix

**Identifier**

- 6020

**Data**

| Name | Data Type | Description |
|---|---|---|
| Unpack Matrix | signed 32bit float | A matrix used for unpacking the data found in the 'Vertex List'.<br>If this matrix is not the identity matrix, and the 'Vertex List' contain data in a non-float data type, then that data has been scaled to make better use of the precision of the given data type. Where this is true, vertices must be 'unpacked' using the 'Unpack Matrix' before any other transformations are applied.<br>Using 'Unpack Matrix' with the 'Fixed Point' data type will not function correctly. |

### 3.6.21. Interleaved Data List

**Identifier**

- 6020

**Data**

| Name | Data Type | Description |
|---|---|---|
| Interleaved Data List | byte array | The list of all vertex data, interleaved on a per-vertex basis, as described in Section 4.3 Interleaved Data. |

## 3.7. Light Blocks

### 3.7.1. Target Object Index

**Identifier**

- 7000

**Data**

| Name | Data Type | Description |
|---|---|---|
| Target Object Index | unsigned 32bit integer | The index into the 'Node' list of the object whose position the light should use as its target. |

### 3.7.2. Light Colour

**Identifier**

- 7001

**Data**

| Name | Data Type | Description |
|---|---|---|
| Light Colour | Float/Fixed array | A three element list of the values of the colour channels of the light, in the order RGB. |

### 3.7.3. Light Type

**Identifier**

- 7002

**Data**

| Name | Data Type | Description |
|---|---|---|
| Light Type | unsigned 32bit integer | An unsigned 32bit integer representing the type of lights.<br>The following values are valid:<br>• 0 – Point Light<br>• 1 – Directional Light<br>• 2 – Spot Light |

### 3.7.4. Constant Attenuation

**Identifier**

- 7003

**Data**

| Name | Data Type | Description |
|---|---|---|
| Constant Attenuation | signed 32bit float | A signed 32bit float representing the constant attenuation of the light, only valid if the light is a spot light. |

### 3.7.5. Linear Attenuation

**Identifier**

- 7004

**Data**

| Name | Data Type | Description |
|---|---|---|
| Linear Attenuation | signed 32bit float | A signed 32bit float representing the linear attenuation of the light, only valid if the light is a spot light. |

### 3.7.6. Quadratic Attenuation

**Identifier**

- 7005

**Data**

| Name | Data Type | Description |
|---|---|---|
| Quadratic Attenuation | signed 32bit float | A signed 32bit float representing the linear attenuation of the light, only valid if the light is a spot light. |

### 3.7.7. Falloff Angle

**Identifier**

- 7006

**Data**

| Name | Data Type | Description |
|---|---|---|
| Falloff Angle | signed 32bit float | A signed 32bit float representing the falloff angle of the light, only valid if the light is a spot light. |

### 3.7.8. Falloff Exponent

**Identifier**

- 7007

**Data**

| Name | Data Type | Description |
|---|---|---|
| Falloff Exponent | signed 32bit float | A signed 32bit float representing the falloff exponent of the light, only valid if the light is a spot light. |

## 3.8. Camera Blocks

### 3.8.1. Target Object Index

**Identifier**

- 8000

**Data**

| Name | Data Type | Description |
|---|---|---|
| Target Object Index | unsigned 32bit integer | The index into the 'Node' list of the object whose position the camera should use as its target. |

### 3.8.2. Field of View

**Identifier**

- 8001

**Data**

| Name | Data Type | Description |
|---|---|---|
| Field of View | Float/Fixed | The field of view value of the camera. |

### 3.8.3. Far Plane

**Identifier**

- 8002

**Data**

| Name | Data Type | Description |
|---|---|---|
| Far Plane | Float/Fixed | The position of the far plane in relation to the camera. |

### 3.8.4. Near Plane

**Identifier**

- 8003

**Data**

| Name | Data Type | Description |
|---|---|---|
| Near Plane | Float/Fixed | The position of the near plane in relation to the camera. |

### 3.8.5. FOV Animation

**Identifier**

- 8004

**Data**

| Name | Data Type | Description |
|---|---|---|
| FOV Animation | Float/Fixed array | An array of Float/Fixed values, each representing the FOV of the camera during each frame of animation. |

## 3.9. POD Data Block

### 3.9.1. Data Type

**Identifier**

- 9000

**Data**

| Name | Data Type | Description |
|---|---|---|
| Data Type | unsigned 32bit integer | An unsigned 32bit integer representing the data type of the elements in 'Data'.<br>Valid values are:<br>• 0 – None<br>• 1 – signed 32bit float<br>• 2 – unsigned 32bit integer<br>• 3 – unsigned short<br>• 4 – four, single byte integer values representing colour channels in the order RGBA<br>• 5 – four, single byte integer values representing colour channels in the order ARGB<br>• 6 – a 4 byte value representing a D3DCOLOR (see msdn.microsoft.com)<br>• 7 – a 4 byte value representing UBYTE4<br>• 8 – a 4 byte value representing a DEC3N<br>• 9 – a 4 byte value representing a fixed point value in the format 16.16<br>• 10 – unsigned byte<br>• 11 – short<br>• 12 – normalised short<br>• 13 – byte<br>• 14 – normalised byte<br>• 15 – unsigned normalised byte<br>• 16 – unsigned normalised short<br>• 17 – unsigned integer |

### 3.9.2. Num. Components

**Identifier**

- 9001

**Data**

| Name | Data Type | Description |
|---|---|---|
| Num. Components | unsigned 32bit integer | The number of components per item held in 'Data'. For example, if 'Data' contained a list of vertex positions consisting of three floats, 'Num. Components' would be '3', four floats would give a value of '4' etc. |

### 3.9.3. Stride

**Identifier**

- 9002

**Data**

| Name | Data Type | Description |
|---|---|---|
| Stride | unsigned 32bit integer | The distance, in bytes, from one array member to the next. |

### 3.9.4. Data

**Identifier**

- 9003

**Data**

| Name | Data Type | Description |
|---|---|---|
| Data | Variable array/Byte | A list of elements, of type determined from the value described in Section 3.9.1 Data Type above.<br>If the data for a given block is interleaved, 'Data' will instead contain a byte representing the offset into the 'Interleaved Data List' of the first element of the block in question, as an unsigned 8bit integer value.<br>For example, if 'Data' would represent normal data for a vertex, but that normal data is interleaved, 'Data' will contain the offset into the 'Interleaved Data List' of the first vertex's normal data, from that point onwards, the normal data for each vertex can be read by moving forward by the value of 'Stride'. |

# 4. Important Notes

## 4.1. Block Type Identifiers

Each 'Block Type Identifier' is an unsigned 32bit integer, however, as the most significant bit of the integer is reserved for determining if a tag is a 'Start Tag' or an 'End Tag' it must be masked; the 'Start Tag' and 'End Tag' masks are as follows:

- 0x00000000 – 'Start Tag' mask
- 0x80000000 – 'End Tag' mask

The 'Identifier' section within each entry of the Block List gives the value prior to masking.

## 4.2. Indices

Several blocks within the POD format reference an index; this index refers to the position (counting from zero) of an element within a list or similar data structure. The consequence of this is that the ordering of objects within the file must be maintained, or translated, post-loading for these indices to have meaning. It should also be noted that indices can be set to '-1', in this instance the index does not refer to any element. For example, a camera that is not following an object may have its 'Target Object Index' set to '-1'.

## 4.3. Interleaved Data

Meshes within POD files may contain interleaved vertex data; in this instance, the arrays of vertex positions, UVW Channels, normal data etc. (see Section 3.6 Mesh Blocks) are repurposed. The POD Data Blocks that normally contain the vertex positions, UVW Channels, normal data etc. will instead contain the position of the first element of the appropriate type within the interleaved data array and a stride. It is possible to read a particular data type for a given element from the interleaved data array by calculating the offset as follows:

$$position_n = position_{initial} + n(stride)$$

It is possible to check for interleaving by checking the size and contents of 'Interleaved Data List', if the block has a size and contents then the mesh in question is interleaved.

If a mesh is interleaved, the following will be interleaved if present:

- Vertex Data
- Normal Data
- Tangent Data
- Binormal Data
- UVW Data
- Vertex Colours
- Bone Indices
- Bone Weights

## 4.4. Float/Fixed

Some elements of a POD file may use either floating point or fixed point data types. These are referenced in the 'Block List' as 'Float/Fixed'. 'Float' should be used by default, unless overwritten by the 'Scene Flags' block.

## 4.5. Existence of Blocks

Only the existence of the ‘Version’ block is guaranteed. Nesting of blocks must be maintained as described in the ‘Block List’. The existence of child blocks within a parent block is guaranteed if the child block is required for the parent block to function. For example, a ‘Spot’ ‘Light’ is guaranteed to include all the attenuation blocks, while a camera ‘Camera’ that is not following an object may not contain a ‘Target Object Index’ block.

## 4.6. Node Ordering

Nodes will appear in the following order:

1. Meshes
2. Lights
3. Cameras
4. Everything else

This is important to remember as the only way to be certain that a node’s index references, for example, a camera, is to know that all meshes and lights have passed.

# 5. Contact Details

For further support contact:

devtech@imgtec.com

PowerVR Developer Technology
Imagination Technologies Limited
Home Park Estate
Kings Langley
Herts, WD4 8LZ
United Kingdom

Tel: +44 (0) 1923 260511
Fax: +44 (0) 1923 277463

Alternatively, you can use the PowerVR Insider forums:

www.imgtec.com/forum

For more information about PowerVR or Imagination Technologies Limited visit our web pages at:

www.imgtec.com