

File Grammar Guide

BigWorld Technology 2.0. Released 2010.

Software designed and built in Australia by BigWorld.

**Level 2, Wentworth Park Grandstand, Wattle St
Glebe NSW 2037, Australia
www.bigworldtech.com**

Copyright © 1999-2010 BigWorld Pty Ltd. All rights reserved.

This document is proprietary commercial in confidence and access is restricted to authorised users. This document is protected by copyright laws of Australia, other countries and international treaties. Unauthorised use, reproduction or distribution of this document, or any portion of this document, may result in the imposition of civil and criminal penalties as provided by law.

Table of Contents

1. Overview	5
2. <code>alias.xml</code>	7
2.1. TypeDefinition section	7
3. BinSection files	9
4. ZipSection files	11
5. Brush files	13
6. <code>capabilities.xml</code>	17
7. <code>.cdata</code>	19
7.1. Contents	19
7.1.1. terrain2 resources	19
8. <code>.chunk</code>	25
8.1. TransformSection	30
9. <code>common_options.xml</code>	31
10. <code>dxenum.xml</code>	33
11. EffectMaterial section	35
12. <code>exporter.xml</code>	37
13. <code>filters.xml</code>	39
14. <code>flags.xml</code>	41
15. <code>.font</code>	43
16. <code>.gui</code>	47
16.1. SimpleGUIComponentSection	48
17. <code>gui.xml</code>	51
17.1. ItemSection	51
18. LightMapSection	55
19. <code>material_kinds.xml</code>	57
20. <code>.mfm</code>	59
21. <code>.model</code>	61
22. <code>modules.xml</code>	69
23. <code>.mvl</code>	71
24. <code>navgen_settings.xml</code>	73
25. <code>options.xml</code>	75
25.1. CAT	75
25.2. ModelEditor	75
25.3. ParticleEditor	88
25.4. WorldEditor	90
26. PackedSection files	121
27. <code>paths.xml</code>	123
28. <code>.ppchain</code>	125
29. <code>.primitives</code>	129
29.1. Vertex data section	129
29.2. Index data section	129
29.3. BSP data section	130
30. <code>shadows.xml</code>	133
31. <code>space.settings</code>	135
32. <code>.texanim</code>	137
33. <code>.texformat</code>	139
34. <code>texture_detail_levels.xml</code>	141
35. <code>ual_config.xml</code>	143
36. <code>.visual</code>	145
36.1. NodeSection	147
36.2. PortalSection	147
37. <code>visual_rules.xml</code>	149
38. <code>.xml</code>	151
38.1. <code><effect>.xml</code>	151

38.2. <enumeration>.xml	153
38.3. <flora>.xml	154
38.4. <graphics_settings>.xml	157
38.5. <light>.xml	158
38.6. <mouse_cursors>.xml	160
38.7. <particle>.xml	161
38.7.1. Sub system components	162
38.7.2. Particle renderers	168
38.7.3. Position/velocity vector generators	170
38.8. <sky>.xml	171
38.9. <fx>.xml	174
38.9.1. Light Section	175
38.9.2. Model Section	175
38.9.3. ParticleSystem Section	175
38.9.4. DummyModel Section	176
38.9.5. Entity Section	176
38.9.6. HardPoint Section	176
38.9.7. ModelRoot Section	176
38.9.8. Node Section	176
38.9.9. AddDecal Section	177
38.9.10. AlignModel Section	177
38.9.11. ClearParticles Section	177
38.9.12. CorrectMotionTriggeredParticles Section	177
38.9.13. Fade Section	178
38.9.14. Flicker Section	178
38.9.15. FlickeringLight Section	178
38.9.16. ForceParticle Section	179
38.9.17. PlayAction Section	179
38.9.18. RampTimeTriggeredParticles Section	179
38.9.19. RandomDelay Section	179
38.9.20. ResetTimeTriggeredParticles Section	180
38.9.21. SetBasis Section	180
38.9.22. SetColour Section	180
38.9.23. SetOrbitorPoint Section	181
38.9.24. SwarmTargets Section	181
38.10. <weather>.xml	181

Chapter 1. Overview

This document details the grammar of the files used by BigWorld Technology, both on the client and the server.

Each section describes, apart from the file's grammar, its purpose, and where applicable, references to further information.

The notation used in this document is as follows:

- **?** — File can have zero or one entry of this type.
- ***** — File can have zero or more entries of this type.
- **+** — File can have one or more entries of this type.
- **+n** — File can have n or more entries of this type.

This document describes the XML configuration files for the client engine and tools, and other files such as models, visuals, terrains, space settings, etc...

Note

For details on BigWorld terminology, see the document Glossary of Terms.

Chapter 2. alias.xml

For more details on alias.xml, see the document Server Programming Guide's section *Properties* → “Property Types” → “Alias of Data Types”.

Used to create alias for data types and located under `fantasydemo/res/scripts/entity_defs`, the grammar of `alias.xml` is described below:

```
<root>
  *<alias_name> TypeDefinition
    ?<Default> dflt_val </Default>
  </alias_name>
</root>
```

Grammar of `fantasydemo/res/scripts/entity_defs/alias.xml`

The list below describes the tags in `alias.xml`:

- **alias_name**

Alias for the type. This can be any string that is a valid XML tag.

- **dflt_val**

The default value must match the type specified in **TypeDefinition**.

For details, see the document Server Programming Guide's section *Properties* → “Default Values”.

- **TypeDefinition**

For details, see “TypeDefinition section” on page 7.

2.1. TypeDefinition section

The grammar for the **TypeDefinition** section is described below:

```
{ <primitive_type>1
| {ARRAY | TUPLE}2 <of> TypeDefinition </of>
| FIXED_DICT3
  <Properties>
    +<field_name>
      <Type> TypeDefinition </Type>
    </field_name>
  </Properties>
  ?<AllowNone> true|false </AllowNone>
  ?<implementedBy> custom_type4 </implementedBy>
| USER_TYPE <implementedBy> custom_type5 </implementedBy>
}
```

Grammar of `fantasydemo/res/scripts/entity_defs/alias.xml`

¹ See the document Server Programming Guide's section *Properties* → “Primitive Types”.

- 2 See the document Server Programming Guide's section *Properties* → “Composite Types” → “ARRAY and TUPLE Types”.
- 3 See the document Server Programming Guide's section *Properties* → “Composite Types” → “FIXED_DICT Data Type”.
- 4 See the document Server Programming Guide's section *Properties* → “Custom User Types”.
- 5 See the document Server Programming Guide's section *Properties* → “Custom User Types”.

The list below describes the tags in `alias.xml`:

- **ARRAY**

Alias for a composite ARRAY data type.

For details on ARRAYS, see the document Server Programming Guide's section *Properties* → “Composite Types” → “ARRAY and TUPLE Types”.

- **FIXED_DICT**

Alias for a composite FIXED_DICT data type.

For details on FIXED_DICT data types, see the document Server Programming Guide's section *Properties* → “Composite Types” → “FIXED_DICT Data Type”.

- ***primitive_type***

Alias for a primitive data type.

For a detailed list of primitive data types available to entity properties, see the document Server Programming Guide's section *Properties* → “Primitive Types”.

- **TUPLE**

Same as ARRAY.

- **USER_TYPE**

Alias for a custom user type.

For details on custom user types, see the document Server Programming Guide's section *Properties* → “Custom User Types”.

Chapter 3. BinSection files

A BinSection can be interpreted as a binary blob of data. It can also be viewed as containing other data sections.

Currently, the binary files `.anca`, and `.primitives` use this format (for details on `.primitive` files, see `.primitives` on page 129).

Described in BNF format, the file has the following format (the asterisk character — * — indicates that the previous section might appear zero or more times):

```
<bin_section> ::= <magic_number> <child_section_data> <index_table>
<child_section_data> ::= <binary_blob>*1
<index_table> ::= <data_section_entry>* <index_table_length>
<data_section_entry> ::= <blob_length><reserved_data><data_section_tag>
<data_section_tag> ::= <tag_length><tag_value>
```

BinSection file format in BNF grammar

¹ For each `<binary_blob>` there is a `<data_section_entry>`.

The list below describes the sections in the file:

- **<binary_blob>**

Binary data, padded to 4-byte boundary

- **<blob_length>**

4-byte little endian integer, containing the length of respective `<binary_blob>`.

- **<index_table_length>**

4-byte little endian integer, containing the length of section `<index_table>`, excluding this entry.

- **<magic_number>**

4-byte number 0x42A14E65 (65 4E A1 42 big endian).

- **<reserved_data>**

16-byte reserved data, containing the following fields:

- **preloadLen** — uint32 containing length of data when streamed in.
 - **version** — uint32 containing a version number.
 - **modified** — uint64 containing timestamp of last modification.
 - **<tag_length>**
- 4-byte little endian integer, containing length of the section's tag.
- **<tag_value>**

Section's tag, padded to 4-byte boundary

Chapter 4. ZipSection files

A ZipSection can be interpreted as a collection of data sections. When stored, the ZipSection uses the PKZIP file format (via the ZipFileSystem class) to group and compress multiple data sections, maintaining hierarchical structure. A ZipSection file can be opened with any standard .zip file viewer.

A ZipSection is designed for use as a developer friendly format for easy access to it's held data. Whereas a BinSection cannot be accessed outside the BigWorld framework, a ZipSection can be viewed, debugged and edited externally. A ZipSection can also be used to only load specific portions of larger files. Unlike the BinSection which is forced to load the entire data structure before decoding.

Note that data added to a ZipSection will automatically compressed when saved to disk.

Currently, the binary files .cdata, and .fxo use this format.

Chapter 5. Brush files

Brush files are used to store information used for terrain painting in **WorldEditor**.

They are described in the document Content Tools Reference Guide's sections *Terrain* → *Cloud shadows*.

The grammar is listed below:

```
<root>

  <type>                string                </type>
  <texture>              file                  </texture>
  <strength>             float                 </strength>
  <size>                 float                 </size>
  <uProjection>          float float float float </uProjection>
  <vProjection>          float float float float </vProjection>
  <opacity>              integer               </opacity>
  <uvLocked>             [true|false]         </uvLocked>

  <heightMask>           [true|false]
    <h1>                float                 </h1>
    <h2>                float                 </h2>
    <h3>                float                 </h3>
    <h4>                float                 </h4>
  </heightMask>

  <slopeMask>            [true|false]
    <s1>                float                 </s1>
    <s2>                float                 </s2>
    <s3>                float                 </s3>
    <s4>                float                 </s4>
  </slopeMask>

  <textureMask>          [true|false]
    <includeProj>        [true|false]         </includeProj>
    <uProjection>        float float float float </uProjection>
    <vProjection>        float float float float </vProjection>
    <texture>            file                  </texture>
    <invert>             [true|false]         </invert>
  </textureMask>

  <noiseMask>            [true|false]
    <minSat>             float                 </minSat>
    <maxSat>             float                 </maxSat>
    <minStrength>        float                 </minStrength>
    <maxStrength>        float                 </maxStrength>
    <noise>
      +<octave>
        <waveLength>    float                 </waveLength>
        <weight>         float                 </weight>
        <seed>           float                 </seed>
      </octave>
    </noise>
  </noiseMask>

</root>
```

Grammar of brush files

The list below describes the tags in brush files:

- **h1 (section heightMask)**

The minimum height in meters at which the height mask will be fully off. Heights below this value will not get painted. Heights between **h1** and **h2** are painted with a strength that varies from zero strength at **h1** meters to full strength at **h2**.

- **h2 (section heightMask)**

The minimum height in meters at which the height mask will be fully on. Heights between **h1** and **h2** are painted with a strength that varies from zero strength at **h1** meters to full strength at **h2**. Heights between **h2** and **h3** are painted with full strength.

- **h3 (section heightMask)**

The maximum height in meters at which the height mask will be fully on. Heights between **h2** and **h3** are painted with full strength. Heights between **h3** and **h4** are painted with a strength that varies from full strength at **h3** meters to zero strength at **h4**.

- **h4 (section heightMask)**

The maximum height in meters at which the height mask will be fully off. Heights between **h3** and **h4** are painted with a strength that goes from full strength at **h3** meters to zero strength at **h4**. Heights above **h4** will not get painted.

- **heightmask (section textureMask)**

This is used to determine whether the height mask is enabled.

- **includeProj**

This is used to determine whether the texture mask compares texture projections as well as the texture name when choosing which layers to replace.

- **invert (section textureMask)**

Invert the sense of the texture mask. If this is false then painting with the texture mask will replace the masked texture. If this is true then painting with the texture mask will replace everything except the masked texture.

- **opacity**

The opacity of the brush. The scale is 0 is completely transparent and 255 is completely solid.

- **minSat (section noiseMask)**

This is the minimum saturation value of the noise as a number between 0.0 and 1.0. All noise values below this value get set to the minimum strength (**minStrength**).

- **minimumStrength (section noiseMask)**

This is the minimum strength of the noise mask as a number between 0.0 and 1.0.

- **maxSat (section noiseMask)**

This is the maximum saturation value of the noise as a number between 0.0 and 1.0. All noise values above this value get set to the maximum strength (**maxStrength**).

- **maximumStrength (section noiseMask)**

This is the maximum strength of the noise mask as a number between 0.0 and 1.0.

- **noiseMask**

This determines whether the noise mask is enabled.

- **s1 (section slopeMask)**

The minimum angle in degrees at which the slope mask will be fully off. Slopes below this value will not get painted. Slopes between **s1** and **s2** are painted with a strength that varies from zero strength at **s1** degrees to full strength at **s2**.

- **s2 (section slopeMask)**

The minimum angle in degrees at which the slope mask will be fully on. Slopes between **s1** and **s2** are painted with a strength that varies from zero strength at **s1** degrees to full strength at **s2**. Slopes between **s2** and **s3** are painted with full strength.

- **s3 (section slopeMask)**

The maximum slope in degrees at which the slope mask will be fully on. Slopes between **s2** and **s3** are painted with full strength. Slopes between **s3** and **s4** are painted with a strength that varies from full strength at **s3** degrees to zero strength at **s4**.

- **s4 (section slopeMask)**

The maximum slope in degrees at which the slope mask will be fully off. Slopes between **s3** and **s4** are painted with a strength that varies from full strength at **s3** meters to zero strength at **s4**. Slopes above **s4** will not get painted.

- **seed (section noiseMask/noise/octave)**

The number used to seed the noise for this octave. Changing the seed allows for multiple copies of similar looking noise to be used.

- **slopeMask**

This is used to determine whether the slope mask is enabled.

- **strength**

The strength of the brush from 0.0 to 100.0.

- **size**

The size of the brush in meters.

- **texture**

- **Section root**

The name of the texture used to paint with.

- **Section terrainMask**

The name of the texture to mask against.

- **textureMask**

This determines whether texture masking is enabled.

- **type**

This identifies the brush type. Currently this is set to "TerrainPainting".

- **uProjection**

- **Section root**

The u-projection of the brush. This combined with the **vProjection** defines an orthogonal coordinate system from which the yaw, pitch, roll and u/v scales are derived.

- **Section terrainMask**

The u-projection of textures to mask against.

- **uProjection**

The u-projection of the brush. This combined with the **vProjection** defines an orthogonal coordinate system from which the yaw, pitch, roll and u/v scales are derived.

- **uvLocked**

If true then the u and v scales are locked to be the same.

- **vProjection**

- **Section root**

The v-projection of the brush. This combined with the **uProjection** defines an orthogonal coordinate system from which the yaw, pitch, roll and u/v scales are derived.

- **Section terrainMask**

The v-projection of textures to mask against.

- **waveLength (section noiseMask/noise/octave)**

The size of an octave of noise in meters.

- **weight (section noiseMask/noise/octave)**

The relative weight of an octave of noise. Octaves whose weights are larger will contribute more to the overall noise.

Chapter 6. capabilities.xml

For details on match triggers and capability flags, see:

- Document Client Programming Guide's section *Scripting* → “Functional components” → “Action Matcher”.
- Document Content Tools Reference Guide's section *ModelEditor* → “Actions panel” → “Action Triggers dialog box”.
- This chapter's *.model* on page 61 .

Used to create alias for entity capabilities (which are then used to trigger or cancel actions by Action Matcher), and located under `fantasydemo/res/scripts/common`, the grammar of `capabilities.xml` is described below:

```
<capabilities.xml>

  *<state> integer
    <name> string </name>
    <id> string </id>
  </state>

</capabilities.xml>
```

Grammar of `fantasydemo/res/scripts/common/capabilities.xml`

The list below describes the tags in `flags.xml`:

- **id**
Python-safe name of the capability, which can be used in scripts.
- **name**
Name of the capability.
- **state**
Number representing the capability bit, integer in the range 0 - 31.

Chapter 7. .cdata

Named as `<chunk>.cdata`, these files are defined under folder `<res>/spaces/<space>`, and contain binary terrain and lighting data.

For details on the information held by this and other chunk files, see the document Client Programming Guide's section *Chunks* → "Implementation files".

For details on this and other zip files' grammar, see *ZipSection files* on page 11 .

7.1. Contents

7.1.1. terrain2 resources

A `terrain2` section is contained in a chunk's `.cdata` file. It contains all the resources for the terrain in a chunk. The different types of terrain data are described in BNF format in the following chapter.

7.1.1.1. heights sections

The `heights` sections stores the height map for the terrain block. Multiple `heights` sections are stored in the block, one for each LOD level, each `heights` section stores data at half the resolution of the previous one. The `heights` sections are named as " `heights?` " where `?` is replaced by a number. The highest res height map is stored in a section named `heights` the second highest in a section called `heights1` all the way down to a map that stores `2x2` heights. This way if the height map resolution is `128x128`, 7 height maps are stored in the file (`heights`, `heights1`, ... `heights6`)

```
<heightMap> ::= <header><heightData>
<header> ::=
  <magic><width><height><compression><version><minHeight><maxHeight><padding>
```

- **<magic>**
uint32 0x00706d68 (string "hmp\0")
- **<width>**
uint32 containing the width of the data
- **<height>**
uint32 containing the height of the data
- **<compression>**
(unused) uint32 containing the compression type
- **<version>**
uint32 containing the version of the data, currently 4
- **<minHeight>**
float containing the minimum height of this block
- **<maxHeight>**
float containing the maximum height of this block

- **<padding>**

4 bytes of padding to make the header 16-byte aligned

- **<heightData>**

PNG compressed block of int32 storing the height in millimetres, dimensions = width * height from the header

7.1.1.2. layer sections

The layer sections store the texture layers for the terrain block. Multiple layer sections are stored in the terrain block. Each section describes one texture layer. The layer sections are named "layer ?" where ? is replaced by a number greater than 1. I.e if the block has 3 layers, three layer sections will be stored ("layer 1", "layer 2", "layer 3")

```
<textureLayer> ::= <header><textureName><blendData>
<header> ::=
  <magic><width><height><bpp><uProjection><vProjection><version><padding>
<textureName> ::= <length><string>
```

- **<magic>**

uint32 0x00646c62 (string bld/0")

- **<width>**

uint32 containing the width of the data

- **<height>**

uint32 containing the height of the data

- **<bpp>**

(unused) uint32 containing the size of the entries in the layer data

- **<uProjection>**

Vector4 containing the projection of the u coordinate of the texture layer

- **<vProjection>**

Vector4 containing the projection of the v coordinate of the texture layer

- **<version>**

uint32 containing the version of the data, currently 2

- **<padding>**

12 bytes of padding to make the header 16-byte aligned

- **<length>**

the length of the texturename string

- **<string>**

the name of the texture used by this layer

- **<blendData>**

png compressed block of uint8 defining the strength of this texture layer at each x/z position

7.1.1.3. normals & lodNormals sections

The normals section stores the high resolution normal map for the terrain block. The lodNormals section stores the LOD normals for the height block, the LOD normals are generally 1/16th of the size of the normals.

```
<normals> ::= <header><data>
<header> ::= <magic><version><padding>
```

- **<magic>**

uint32 0x006d726e (string "nrm/0")

- **<version>**

uint32 containing the version of the data, currently 1

- **<padding>**

8 bytes of padding to make the header 16-byte aligned

- **<data>**

png compressed block storing 2 signed bytes per entry for the x and z components of the normal the y component is calculate in the shader

7.1.1.4. holes section

The holes section stores the holemap for the terrain block, this section is only stored when a terrain block has holes in it.

```
<holes> ::= <header><data>
<header> ::= <magic><width><height><version>
```

- **<magic>**

uint32 0x006c6f68 (string "hol/0")

- **<width>**

uint32 containing the width of the data

- **<height>**

uint32 containing the height of the data

- **<version>**

uint32 containing the version of the data, currently 1

- **<data>**

The hole data stored in a bit field of width * height, each row in the data is rounded up to 1 byte. If a bit is set to 1 it denotes a hole in the map.

7.1.1.5. horizonShadows section

The horizonShadows section stores the horizon shadows for the terrain block.

```
<shadows> ::= <header><data>
<header> ::= <magic><width><height><bpp><version><padding>
```

- **<magic>**
uint32 0x00646873 (string "shd/0")
- **<width>**
uint32 containing the width of the data
- **<height>**
uint32 containing the height of the data
- **<bpp>**
(unused)uint32 containing the bits per entry in the data
- **<version>**
uint32 containing the version of the data, currently 1
- **<padding>**
12 bytes of padding to make the header 16-byte aligned
- **<data>**
The shadow data, (uint16,uint16) * width * height, the horizon shadow data stores two angles between which there is no occlusion from any terrain or objects.

7.1.1.6. lodTexture.dds section

The lodTexture.dds section stores the LOD texture for the terrain block. The LOD texture is a low resolution snapshot of all the texture layers blended together. The texture is stored in the DXT5 format. For more information about the dds texture format please refer to the DirectX documentation.

7.1.1.7. dominantTextures section

The dominantTextures section stores the dominant texture map. The dominant texture map stores the texture with the highest blend for each x/z location in the terrain block.

```
<dominant> ::= <header><texNames><data>
<header> ::=
  <magic><version><numTextures><texNameSize><width><height><padding>
```

- **<magic>**
uint32 0x0074616d (string "mat/0")
- **<version>**
uint32 containing the version of the data, currently 1

- **<numTextures>**
uint32 containing the number of textures referenced by the dominant texture map
- **<texNameSize>**
uint32 containing the size of the texture entries
- **<width>**
uint32 containing the width of the data
- **<height>**
uint32 containing the height of the data
- **<padding>**
8 bytes of padding to make the header 16-byte aligned
- **<texNames>**
numTextures entries of texNameSize size containing the names of the dominant textures referred to in this map. Texture names shorter than texNameSize are padded with 0
- **<data>**
stored as a compressed bin section. byte array of width * height, each entry is an index into the texture names which indexes the dominant texture at the x/z location of the entry

Chapter 8. .chunk

Named as `<chunk>o.chunk` (for outside chunks) and `<chunk>i.chunk` (for inside chunks), these files are defined under folder `<res>/spaces/<space>`.

Contains list of scene objects, texture sets, collision scene, etc...

The .chunk file format specification is illustrated below:

```
<fileName> ?label

?<terrain>
  <resource> [file.terrain|file.cdata/terrain] </resource>
</terrain>

?TransformSection 1

*<model> ?label
  +<resource> file.model </resource>
  ?<animation>
    <name> string </name>
    <frameRateMultiplier> float </frameRateMultiplier>
  </animation>
  TransformSection 2
  *<staticLighting>3
    <offset> int </offset>
    <size> int </size>
  </staticLighting>
</model>

*<entity>
  ?<id> string </id>
  <type> string </type>
  TransformSection 4
  ?<instantiation> integer </instantiation>
  ?<tag> string </tag>
  <properties>
    PropertiesList
  </properties>
</entity>

*<particles>
  <resource> file.xml </resource>
  TransformSection 5
</particles>

<boundingBox>
  <min> float float float </min>
  <max> float float float </max>
</boundingBox>

?AmbientLightSection 6 7

*DirectionalLightSection 8

*FlareSection 9

*OmniLightSection 10

*SpotLightSection 11
```

```

    *<water>
      <position>      float float float  </position>
      <orientation>   float              </orientation>
      <size>          float float float  </size>
      <tessellation>  float              </tessellation>
      <consistency>   float              </consistency>
    </water>

    ?<navmesh>
      <resource> file.cdata/navmesh  </resource>
    </navmesh>

    <waypointGenerationTime>
      <hi> integer </hi>
      <lo> integer </lo>
    </waypointGenerationTime>

  </fileName>

```

Grammar of chunk file

- 1 See "TransformSection" on page 30 .
- 2 See "TransformSection" on page 30 .
- 3 Inside chunks only.
- 4 See "TransformSection" on page 30 .
- 5 See "TransformSection" on page 30 .
- 6 Inside chunks only.
- 7 See "<light>.xml" on page 158 .
- 8 See "<light>.xml" on page 158 .
- 9 See "<light>.xml" on page 158 .
- 10 See "<light>.xml" on page 158 .
- 11 See "<light>.xml" on page 158 .

The list below describes the tags in the chunk file:

- **adjacentChunk (sections navPolySet/navPoly and waypointSet/waypoint)**

For details, see the document Server Overview's section Server Components → "CellApp" → "Navigation System", and Content Tools Reference Guide's section NavGen → "Navpolys, vertices and adjacency".

ID of the navpoly adjacent to this vertex.

- **AmbientLightSection**

For details, see "<light>.xml" on page 158 .

- **animation (section model)**

Tag for model animation section, which specifies animation file and frame rate multiplier.

- **boundingBox**

Tag for chunk's bounding box section, which specifies its point of origin and end.

- **consistency (section water)**

Consistency of the water: **0=fluid, 1=rigid**

- **DirectionalLightSection**

For details, see "<light>.xml" on page 158 .

- **entity**

Tag for entity section, which specifies settings for the models to be placed in the chunk, such as entity id, type, transform, entity-specific properties, etc....

- **FlareSection**

For details, see "*<light>.xml*" on page 158 .

- **frameRateMultiplier (section model/animation)**

The rate at which the animation will be sped up/slowed down.

- **girth (sections navPolySet and waypointSet)**

For details, see the document Server Overview's section Server Components → "CellApp" → "Navigation System", and Content Tools Reference Guide's section NavGen → "Navpolys, vertices and adjacency".

Girth for which the navigation mesh was generated. For details on the relation between entity girth and navigation mesh generation, see the document Content Tools Reference Guide, chapter *NavGen*, "Changing settings".

- **height (sections navPolySet and waypointSet)**

For details, see the document Server Overview's section Server Components → "CellApp" → "Navigation System", and Content Tools Reference Guide's section NavGen → "Navpolys, vertices and adjacency".

Vertical range of the navpoly prism.

- **hi (section waypointGenerationTime)**

The high 32 bits of the waypoint generation time.

- **id (section entity)**

ID of the entity in the chunk. This can be used by scripts to reference the entity.

- **instantiation (section entity)**

Where the entity should be instantiated: 0=server, 1=client

- **lighting (section model)**

The size and offset of the vertex colour used for the static lighting of this model.

- **lo (section waypointGenerationTime)**

The low 32 bits of the waypoint generation time.

- **max (section boundingBox)**

End point of the chunk's bounding box.

- **min (section boundingBox)**

Starting point of the chunk's bounding box.

- **minHeight (sections navPolySet and waypointSet)**

For details, see the document Server Overview's section Server Components → "CellApp" → "Navigation System", and Content Tools Reference Guide's section NavGen → "Navpolys, vertices and adjacency".

The minimum height of values in the set.

- **model**

Tag for model section, which specifies settings for the models to be placed in the chunk, such as resource file, animation file, transform, lighting, etc....

- **name (section model/animation)**

File containing the animation for the model.

The file will be located in the animations folder under the model file's folder, and the extension .animation will be automatically appended to it.

For example if `<model>` is `myFolder/example.model`, and `<animation>` is `myAnimation`, then the complete path/filename for the animation file is `<res>/myFolder/animations/myAnimation.animation`.

- **navmesh**

For details, see the document Server Overview's section Server Components → "CellApp" → "Navigation System", and Content Tools Reference Guide's section NavGen → "Navpolys, vertices and adjacency".

Tag for navigation mesh section, which specifies its resource file.

- **navPoly (section navPolySet)**

For details, see the document Server Overview's section Server Components → "CellApp" → "Navigation System", and Content Tools Reference Guide's section NavGen → "Navpolys, vertices and adjacency".

Tag for navpoly section, which specifies height, minimum height, its delimiting vertices, and optionally the vertex's adjacent chunk ID.

- **navPolySet**

For details, see the document Server Overview's section Server Components → "CellApp" → "Navigation System", and Content Tools Reference Guide's section NavGen → "Navpolys, vertices and adjacency".

Tag for section specifying group of navpolys.

- **OmniLightSection**

For details, see "`<light>.xml`" on page 158 .

- **orientation (section water)**

The yaw of the water.

- **particles**

Tag for particle section, which specifies settings for the particles to be placed in the chunks, such as resource file and transform.

- **properties (section entity)**

Tag for the entity-specific properties, containing **PropertiesList** section.

- **PropertiesList (section entity/properties)**

List of entity-specific properties, as specified by the entity definition file.

These are the persistent properties defined by the entity or the entity that it inherits from.

For details on entity definition files, see the document *Server Programming Guide's section Physical Entity Structure for Scripting* → “The Entity Definition File”.

- **resource**

- **Section model**

File containing the model. For details on format of model files, see `.model` on page 61. The path is relative to the game's resources folder (`<res>`).

- **Section navmesh**

*For details, see the document *Server Overview's section Server Components* → “CellApp” → “Navigation System”, and *Content Tools Reference Guide's section NavGen* → “Navpolys, vertices and adjacency”.*

Binary file containing navigation mesh data for the chunk. The path is relative to the chunk file's folder.

- **Section particles**

File containing the particle system. For details on format of particle system files, see “`<particle>.xml`” on page 161 . The path is relative to the game's resources folder (`<res>`).

- **Section terrain**

File containing terrain settings such as height map, overlay data, and textures used. The path is relative to the chunk file's folder.

- **SpotLightSection**

For details, see “`<light>.xml`” on page 158 .

- **tag (section entity)**

The tag of this instance, used to identify this particular instance of an entity.

- **terrain**

Tag for terrain section, which specifies settings for the terrain to be placed in the chunk, such as resource file.

- **tessellation (section water)**

The preferred distance between vertices in the water.

- **TransformSection (main section and entity, model, and particles sections)**

For details, see “TransformSection” on page 30 .

- **type (section entity)**

Type of the entity to be placed.

BigWorld uses this value to determine the path/name of the entity definition file: `<res>/scripts/entity_defs/<type>.def`

For details on entity definition files, see the document *Server Programming Guide's section Physical Entity Structure for Scripting* → “The Entity Definition File”.

- **vertex** (sections **navPolySet/navPoly** and **waypointSet/waypoint**)

For details, see the document *Server Overview's* section *Server Components* → “CellApp” → “Navigation System”, and *Content Tools Reference Guide's* section *NavGen* → “Navpolys, vertices and adjacency”.

XY position of the navpoly's vertex.

The third coordinate is used to store adjacency information for the edge formed between this vertex and the next — it is either the navpoly ID of the adjacent navpoly, or an encoding of an obstacle type.

- **water**

Tag for water section, which specifies its position, orientation, size, etc...

For details, see the document *Client Programming Guide's* section *Chunks* → “Sway items”.

- **waypoint** (section **waypointSet**)

Tag for waypoint section, which specifies height, minimum height, its delimiting vertices, and optionally the vertex's adjacent chunk ID.

- **waypointGenerationTime**

The filetime of the chunk the last time the waypoint was generated.

- **waypointSet**

Tag for section specifying group of waypoint.

8.1. TransformSection

The transform information contained in **TransformSection** depends on which of the following sections of the chunk file it appears:

- Main section
- `<model>`
- `<entity>`
- `<particles>`

The grammar for the TransformSection is described below:

```
<transform>
  <row0>    float    </row0>
  <row1>    float    </row1>
  <row2>    float    </row2>
  <row3>    float    </row3>
</transform>
```

Grammar of **TransformSection** in visual file

The list below describes the tags in **TransformSection**.

- **row0, row1, row2, row3**

Transform matrices to be applied to chunk, model, entity, or particle system.

Chapter 9. common_options.xml

The common_options.xml file was created to hold settings and options that are common to more than one BigWorld tool.

The common_options.xml file format specification is illustrated below:

```
<root>
  <cooperative>
    ?<mode> [AUTO|ON|OFF] </mode>

    <apps>
      *<app> string </app>
    </apps>
  </cooperative>
</root>
```

Grammar of common_options.xml file

The list below describes the tags in the common_options.xml file:

- **mode (section cooperative)**

Sets the way WorldEditor, ModelEditor and ParticleEditor should cooperate with each other and with other DirectX applications. Only three values are allowed:

- **AUTO:** It's the recommended and default state, and ensures the tools release their video memory when an application in the <apps> list below is running and the tool is in the background, but keeps DirectX resources if no other specified applications are running to improve performance.
- **ON:** Always releases DirectX resources when it is switched to the background, to allow maximum compatibility with other applications, but makes switching between applications slower.
- **OFF:** Never releases its DirectX resources, making application switching fast, but can result in problems and/or crashes when running multiple DirectX applications.

- **app (section cooperative/apps)**

Specifies one or more executables that WorldEditor, ParticleEditor and ModelEditor should cooperate with. DOS-style wildcards can be used so, for example, if you include modeeditor*.exe in the <apps> list, the tools will cooperate with modeeditor.exe, modeeditor_debug.exe, modeeditor_eval.exe, etc. Multiple applications can be specified by including more than one <app> tags, and applications don't have to be BigWorld tools. For example, you could include 3dsmax.exe and maya.exe to ensure smooth operation with those tools.

Chapter 10. dxenum.xml

For details, see “<enumeration>.xml” on page 153 .

Chapter 11. EffectMaterial section

Used by the EffectMaterial class, this section contains material information, and is part of the following file formats:

- **<material>.mfm** — See *.mfm* on page 59.
- **<item>.model** — See *.model* on page 61.
- **<item>.visual** — See *.visual* on page 145.

The format of the **EffectMaterial** section is illustrated below:

```
?<identifier>      string    </identifier>
+<fx>              file      </fx>
?<channel>         string    </channel>
?<mfm>             file      </mfm>
<materialKind>    integer    </materialKind>
<collisionFlags>  integer    </collisionFlags>

+<property>  string_property_name

  [ <Texture>  file                                </Texture>
    | <Vector4> float float float float          </Vector4>
    | <Float>   float                                </float>
    | <Bool>    [true|false]                        </bool>
    | <Int>     int
      ?<UIMin>  int  </UIMin>
      ?<UIMax>  int  </UIMax>
    </Int>
  ]

</property>
```

Grammar of the **EffectMaterial** section

The list below describes the tags in the **EffectMaterial** section:

- **Bool (section property)**
Generic material property requiring a bool value.
- **collisionFlags^A**
Specifies if the material will collide with models, or be affected by light and other effects.
- **Float (section property)**
Generic material property requiring a float value (*e.g.*, *diffuseLightExtraModulation*, *selfIllumination*, *parallaxAspectRatio*, *parallaxScale*, *glowRatio*, etc...)
- **fx**
Effect resource.
- **identifier**
Material name.
- **Int (section property)**

Generic material property requiring an int value (e.g., `alphaReference`, `lightDir`, etc...)

- **materialKind^A**

Material kind, used for sound and particle system This value is used to retrieve material information from file `bigworld/res/system/data/material_kinds.xml`. For details, see *material_kinds.xml* on page 57.

- **property^A**

Tag for miscellaneous material properties.

- **Texture (section property)**

Texture file to be assigned to the corresponding property (e.g., `diffuseMap`, `heightMap`, `specularMap`, `normalMap`, `glowMap`, etc...)

- **Vector4 (section property)**

Generic material property requiring 4 float values (e.g., `specularColour`, `vTransform`, `uTransform`, `materialSpecular`, etc...)

A — For details, see the document *Content Tools Reference Guide's* section *ModelEditor* → “Panel summary” → “Materials Settings panel”.

Chapter 12. exporter.xml

For details on our Maya exporter, see the document Content Tools Reference Guide's section *3ds Max and Maya Exporters* → “Maya”.

Located under `bigworld/tools/exporter/maya<version>`, the grammar of `exporter.xml` is listed below:

```
<root>

  ?<unitScale>    float          </unitScale>
  ?<exportMode>   integer        </exportMode>
  ?<allowScale>   [true|false]   </allowScale>
  ?<bumpMapped>   [true|false]   </bumpMapped>

</root>
```

Grammar of `bigworld/tools/exporter/maya<version>.exporter.xml`

The list below describes the tags in `bigworld/tools/exporter/maya<version>.exporter.xml`:

- **allowScale**

Determines if transforms should be exported with scale (if set to true), or if transforms should be normalised before export (if set to false).

- **bumpMapped**

Determines if additional information should be exported in order to allow object's use with normal mapping.

- **exportMode**

Mode in which the object was exported last time.

- **unitScale**

Scale used to translate Maya units to metres. Default is 0.1, which means that each Maya equals 0.1 metres.

Chapter 13. filters.xml

Used by WorldEditor to configure the filters available in Terrain Filtering panel, this file is located under `bigworld/tools/worldeditor/resources/data`.

From the pre-defined set of implemented filters, only the ones specified in this file will be listed in the **Terrain Filtering** panel's **Filters** list box (in the order in which they appear). For details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Filtering panel".

The grammar of `filters.xml` is listed below:

```
<filters.xml>

  *<filter>
    <name>          string      </name>
    <constant>       float       </constant>
    <kernel>         Vector3     </kernel>
    <kernel>         Vector3     </kernel>
    <kernel>         Vector3     </kernel>
    <strengthRatio>  float       </strengthRatio>
    <kernelSum>      float       </kernelSum>
    <included>       [true|false] </included>
  </filter>

</filters.xml>
```

Grammar of `filters.xml`

The list below describes the tags in `filters.xml`:

- **constant**

Base of strength, used in conjunction with the `strengthRatio` tag.

When refers to an actual filter, this value is used as strength directly (probably here it should be 0).

When refers to the terrain height brush (used in the **Terrain Height** panel^A), the final strength will be the product of constant and the value of **Terrain Height** panel's **Strength** field.

- **filter**

Section specifying parameters either for a filter (displayed in **Terrain Filtering** panel^B), or for the terrain height brush (used in **Terrain Height** panel^A).

- **included**

Determines if the filter will be listed in **Terrain Filtering** panel^B. Defaults to true.

- **index**

Deprecated tag — used to determine order in which filter was listed in **Terrain Filtering** panel^B.

- **kernel**

The 3 kernel tags (of type `Vector3`) compose an array of `float[9]`.

Each float value represents the strength to be applied to a cell enclosed by the brush.

- **kernelSum**

Normally the sum of the 9 kernel values.

The final height of a cell equals to:

$$(\text{nearby cell heights} \times \text{corresponding kernel} + \text{strength} \times \text{strengthRatio}) / \text{kernelSum}$$

- **name**

Name of the filter. The name is used to match the filter pre-defined in the source code, determining thus its working. If an invalid name is listed, then it will be displayed in **Terrain Filtering** panel's^B **Filters** list box, but will not be selectable.

Possible values are:

- **Extra Slow Smooth**
- **Slow Smooth**
- **Medium Smooth**
- **Fast Smooth**
- **Nice 'n' Smooth**
- **Sharpen**
- **Sharpen More**
- **Raise 20cm** (the shipped file sets the included tag for this filter to false — this section configures **Terrain Height** panel's^A brush).

- **strengthRatio**

Value to multiply by the **Terrain Height** panel's^A **Strength** field before recalculating new height.

So, if this ratio is 0, no strength is available

A — For details, see the document *Content Tools Reference Guide's* section *WorldEditor* → “Panel summary” → “Terrain Height panel”.

B — For details, see the document *Content Tools Reference Guide's* section *WorldEditor* → “Panel summary” → “Terrain Filtering panel”.

Chapter 14. flags.xml

Used by ModelEditor to display the material collision flags in the property view, this file is located under `bigworld/tools/WorldEditor/Resources`. For details on ModelEditor, see the document Content Tools Reference Guide's chapter *ModelEditor*.

The grammar of `flags.xml` is listed below:

```
<root>
  <collisionFlags>
    *<collision_flag> integer </collision_flag>
  </collisionFlags>
</root>
```

Grammar of `bigworld/tools/modeleditor/Resources/flags.xml`

The list below describes the tags in `flags.xml`:

- **collision_flag**

Option to be displayed in ModelEditor's **Material Settings** panel's **Collision Flag** drop-down list box .

The underlying integer value is saved in the visual file.

For details on the drop-down list box, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “Materials Settings panel”.

For details on visual file's grammar, see `.visual` on page 145 .

Chapter 15. .font

Mentioned in document Client Programming Guide's section *Graphical User Interface (GUI) → Fonts*.

Named `<font_name>_<font_size>.font` and located in the folder specified by section `<system>/<fontRoot>` in file `<res>/resources.xml`, a font file has the grammar described below:

```
<file_name>

+<creation>
  <sourceFont>      string          </sourceFont>
  <sourceFontSize>  integer         </sourceFontSize>
  <startChar>       [U+hex|integer] </startChar>
  <endChar>         [U+hex|integer] </endChar>
  <widestChar>      [U+hex|integer] </widestChar>
  <fixedWidth>      integer         </fixedWidth>
  <effectsMargin>   integer         </effectsMargin>
  <textureMargin>   integer         </textureMargin>
  <spaceProxyChar>  integer         </spaceProxyChar>
  <maxTextureWidth> integer         </maxTextureWidth>
  <dropShadow>     [true|false]    </dropShadow>
  <shadowAlpha>     integer         </shadowAlpha>
  <antialias>       [true|false]    </antialias>
  <bold>            [true|false]    </bold>
  *<secondary>
    <sourceFont>      string          </sourceFont>
    <unicodeRange>    U+hex-U+hex    </unicodeRange>
    <bold>            [true|false]    </bold>
    <antialias>       [true|false]    </antialias>
    <dropShadow>     [true|false]    </dropShadow>
  </secondary>
</creation>
*<generated>
  <map>              file           </map>
  <mapDimensions>    float float    </mapDimensions>
  <maxWidth>          integer        </maxWidth>
  <height>            integer        </height>
  <uvs>              string         </uvs>
  <widths>           string         </widths>
</generated>
</file_name>
```

Grammar of font file

The list below describes the tags in the font file:

- **creation**

Tag for section with font description.

- **antialias**

If set to true, then characters will be anti-aliased.

Defaults to true.

- **bold**

If set to true, then the `lfWeight` member of the `LOGFONT` structure will be set to `FW_BOLD`, otherwise it will be set to `FW_NORMAL`.

Defaults to false.

- **dropShadow**

If set to true, then BigWorld will automatically create a drop shadow in the font. This will help displaying text against any coloured background, as long as the text is drawn blended.

Defaults to false.

- **effectsMargin**

Number of pixels used by BigWorld font effects. For example, if you use dropShadow, then you should include 1 pixel for it.

Defaults to 0.

- **fixedWidth**

Width of all character in the font map.

Set this value to 0 if font has variable width (proportional spacing).

- **maxTextureWidth**

Maximum allowed width for the generated texture map.

- **maxWidth**

Maximum width assumed by any of the font characters.

- **shadowAlpha**

Alpha value to use when creating the shadow.

A value of 255 generates a fully opaque shadow. Lower values will create softer, and usually better looking, shadows.

Defaults to 255.

- **sourceFont**

This is set into the `lfFaceName` member of the Windows API `LOGFONT` structure.

- **sourceFontSize**

This is set into the `lfHeight` member of the Windows API `LOGFONT` structure.

- **spaceProxyChar**

ASCII code of the character to be used for calculating the width of a space character.

Defaults to 105 (character i).

- **startChar**

Unicode value of the first character to preload into the glyph cache.

Defaults to 0.

- **endChar**

Unicode value of the last character to preload into the glyph cache.

Defaults to 0.

- **textureMargin**

How many extra texels will be added in-between each glyph.

If a font is to be used with bilinear filtering turned on, it is best to include at least 1 texel worth of textureMargin, so the filtering will not sample from neighbouring glyphs.

Defaults to 0.

- **secondary**

Tag for section to define a secondary font to use when rendering a characters from a particular Unicode range. Multiple secondary fonts can be specified for different ranges.

- **sourceFont**

The face name of the font to use.

- **unicodeRange**

The range, in Unicode of characters, that this secondary font will apply.

- **bold**

If set to true, the font characters will be bold.

Defaults to the value of bold in the parent creation section.

- **antialias**

If set to true, the font characters will be antialiased.

Defaults to the value of antialias in the parent creation section.

- **dropShadow**

If set to true, then BigWorld will automatically create a drop shadow in the font. This will help displaying text against any coloured background, as long as the text is drawn blended. The value of shadowAlpha in the parent creation section will be used.

Defaults to the value of dropShadow in the parent creation section.

- **generated**

Tag for section with generated font metrics.

- **height**

Upon font generation, this tag will store the global font height.

- **map**

This tag will store the name of the texture containing the font map.

- **mapDimensions**

Upon font generation, this tag will store font map dimensions.

- **uvs**

List of UV coordinates for each of the characters in the font map.

- **widths**

List of widths for each of the characters in the font map.

- **maxWidth**

Maximum width assumed by any of the font characters.

Chapter 16. .gui

Used to persist GUI hierarchies, these files are located under `fantasydemo/res/gui`.

For a detailed description of each section (GUI class) and tag (GUI member), see the Client Python API's entry **Modules → GUI**.

The grammar of `<gui>.gui` files is listed below:

```
<file_name>

*<SimpleGUIComponent> integer
  SimpleGUIComponentSection
</SimpleGUIComponent>

*<GoboComponent> integer
  SimpleGUIComponentSection
</GoboComponent>

*<FrameGUIComponent> integer
  SimpleGUIComponentSection
    <edgeTextureName> folder/file </edgeTextureName>
    <cornerTextureName> folder/file </cornerTextureName>
  </FrameGUIComponent>

*<WindowGUIComponent> integer
  SimpleGUIComponentSection
    <scroll> integer </scroll>
    <minScroll> integer </minScroll>
    <maxScroll> integer </maxScroll>
  </WindowGUIComponent>

*<GraphGUIComponent> integer
  SimpleGUIComponentSection
    <nPoints> integer </nPoints>
    <minY> float </minY>
    <maxY> float </maxY>
    <frequency> float </frequency>
  </GraphGUIComponent>

*<TextGUIComponent> integer
  SimpleGUIComponentSection
    <label> string </label>
    <font> file1 </font>
    <pixelSnap> true|false </pixelSnap>
    <explicitSize> true|false </explicitSize>
  </TextGUIComponent>

*<BoundingBoxGUIComponent> integer
  SimpleGUIComponentSection
    ?<clipSpaceSource> true|false </clipSpaceSource>
    ?<clipToBox> true|false </clipToBox>
    ?<absoluteSubspace> integer </absoluteSubspace>
    ?<offsetSubspace> float float float </offsetSubspace>
    ?<alwaysDisplayChildren> true|false </alwaysDisplayChildren>
  </BoundingBoxGUIComponent>

*<ConsoleGUIComponent> integer
  SimpleGUIComponentSection
    <scale> float </scale>
    <lines>
```

```

        *<colour> float float float float </colour>
        *<line>   string                                </line>
    </lines>
</ConsoleGUIComponent>

*<AlphaGUIShader> integer
    <mode> integer </mode>
    <stop> float   </stop>
    <start> float   </start>
    <alpha> float   </alpha>
    <speed> float   </speed>
</AlphaGUIShader>

*<ClipGUIShader> integer
    <mode> integer </mode>
    <value> float   </value>
    <speed> float   </speed>
    <delay> float   </delay>
    <slant> float   </slant>
</ClipGUIShader>

*<ColourGUIShader> integer
    <start> float float float float </start>
    <middle> float float float float </middle>
    <end> float float float float </end>
    <value> float   </value>
    <speed> float   </speed>
</ColourGUIShader>

*<MatrixGUIShader> integer
    ?<target>
        <row0> float float float </row0>
        <row1> float float float </row1>
        <row2> float float float </row2>
        <row3> float float float </row3>
    </target>
    ?<eta> float </eta>
    ?<blend> true|false </blend>
</MatrixGUIShader>

</file_name>

```

Grammar of `fantasydemo/res/gui/<gui>.gui`

1 For details on grammar of font files, see `.font` on page 43 .

The list below describes the tags in file `<gui>.gui`:

- **AlphaGUIShader**, **BoundingBoxGUIComponent**, **ClipGUIShader**, **ColourGUIShader**, **ConsoleGUIComponent**, **FrameGUIComponent**, **GoboComponent**, **GraphGUIComponent**, **MatrixGUIShader**, **SimpleGUIComponent**, **TextGUIComponent**, **WindowGUIComponent**

Creates an object of the specified type.

For details on GUI components, see the document Client Programming Guide's section *Graphical User Interface (GUI)* → “C++ GUI support”.

16.1. SimpleGUIComponentSection

The grammar for the **SimpleGUIComponentSection** is described below:

```

<position>          float float float      </position>
<widthInClip>       true|false              </widthInClip>
<width>             float                  </width>
<heightInClip>      true|false              </heightInClip>
<height>            float                  </height>
<colour>            float float float float </colour>
<angle>             integer                 </angle>
<flip>              integer                 </flip>
<visible>           true|false              </visible>
<horizontalAnchor>  integer                 </horizontalAnchor>
<verticalAnchor>    integer                 </verticalAnchor>
<textureName>       folder/file             </textureName>
<materialFX>        integer                 </materialFX>
<filterType>        integer                 </filterType>
<tiled>             true|false              </tiled>
<tileWidth>         integer                 </tileWidth>
<tileHeight>        integer                 </tileHeight>
<script>            string                  </script>
?<children>
  +<attribute> integer </attribute>
</children>
?<shaders>
  +<attribute> integer </attribute>
</shaders>

```

Grammar of <gui>.gui's **SimpleGUIComponentSection**

For details on each of the tags, see the Client Python API, entry **Modules → GUI**.

Chapter 17. gui.xml

The GUI XML file is used to configure WorldEditor's menus and toolbars.

Defined in `bigworld/tools/worldeditor/resources/data`, the grammar of `gui.xml` is illustrated below:

```
<GUI>
  ItemSection
</GUI>
```

Grammar of `bigworld/tools/worldeditor/resources/data/gui.xml`

17.1. ItemSection

The grammar of **ItemSection** is illustrated below:

```
*<item>
  <name>          string          </name>
  <type>          [ ACTION | CHILD | CHOICE | EXPANDED_CHOICE | GROUP | SEPARATOR | TOGGLE ]
</type>
[ ItemSection ]
[ <width>         integer          </width>          ]
[ <displayName>   string           </displayName>   ]
[ <description>   string           </description>   ]
[ <shortcut>      string           </shortcut>      ]
[ <updater>       string           </updater>       ]
[ <commandID>     integer          </commandID>     ]
[ <action>        string           </action>        ]
[ <imageHot>      file:x_bgn,y_bgn,x_end,y_end </imageHot>      ]
[ <imageDisabled> file:x_bgn,y_bgn,x_end,y_end </imageDisabled> ]
[ <imageNormal>   file:x_bgn,y_bgn,x_end,y_end </imageNormal>   ]
[ <toolMode>      string           </toolMode>      ]
[ <transparency>  [ 0-255 ], [ 0-255 ], [ 0-255 ] </transparency> ]
</item>
```

Grammar of **ItemSection**

The list below describes the tags in **ItemSection**:

- **action**

C++ function, Python method, or expression updating value of a tag in `options.xml` to be called when the item is fired (*i.e.*, pressed, selected, etc...) — `options.xml` is located in `bigworld/tools/worldeditor`; for details on the file's grammar, see “WorldEditor” on page 90.

The possible values for this tag are described below:

- **Type: C++ function**

Defined in the tool's source code.

Example: `newSpace` (in `MainMenu`→`File`→`CreateNewSpace`)

- **Type: Python method**

Defined in any Python script located in `bigworld\tools\worldeditor\resources\scripts`.

Example: doQuickSave (in MainMenu→File→QuickSave)

Please note that if the method is defined in `UIExt.py`, then the method need not be appended by the module name. For example, to declare `UIAdapter.py`'s method `updateSelectionFilter` as the updater, you would have to specify the value `UIAdapter.updateSelectionFilter`.

- **Expression updating value of a tag in `options.xml`**

Defined inline (using C syntax)

Example: `render/terrain/wireFrame` = 1 (in
MainToolBar→TerrainWireframe→ShowTerrainWireframe.

- **commandID**

ID required by some UI item, like sub-menus or toolbar buttons.

Usually `GUIManager` can generate the command ID automatically, but sometimes a specific command ID might be needed for an UI item.

- **description**

Text to be displayed in the item's tooltip.

- **displayName**

Text to be displayed in the menu or menu item.

- **imageDisabled**

Image to be displayed for the toolbar button when it is disabled.

- **imageHot**

Image to be displayed for the toolbar button when the mouse hovers it.

- **imageNormal**

Image to be displayed for the toolbar button.

- **name**

Name of the item being configured (menu, menu item, toolbar group, or toolbar button).

- **shortcut**

Keyboard shortcut to activate the item.

- **toolMode**

Tool mode activated by the respective item.

For the list of available tool modes and their description, see the document *Content Tools Reference Guide's* section *WorldEditor* → "Panel summary".

- **transparency**

RGB value of the transparent color in the toolbar button images.

- **type**

Type of the item being configured.

The list below describes the available options

- **ACTION**

Indicates that the item has an action associated with it.

It must then define `action` tag.

Examples: **File** → **New Space** and **File** → **Open Space** menu items, **Save** and **Undo** toolbar buttons.

- **CHILD**

Indicates a sub-item of items of type `CHOICE`, `EXPANDED_CHOICE`, or `TOGGLE`.

Examples: **Tool mode** toolbar buttons^A (this toolbar group has type value of `CHOICE`), and **View** → **Status Bar** menu item's **ON** and **OFF** definitions (this menu item has a type value of `TOGGLE`).

- **CHOICE**

Indicates that the only one of the sub-items^B defined for this item may be active at any one time (*i.e.*, the sub-items will act as option buttons).

Examples: **Tool mode** toolbar buttons^A, and **Camera Speed** toolbar buttons.

- **EXPANDED_CHOICE**

Indicates that the sub-items^B defined for this items will be displayed as entries in a drop-down list box.

- **GROUP**

For menu items, indicates that the item has a sub-menu associated with it.

For toolbar items, indicates that the item defines other groups or toolbar buttons.

Examples: **File** menu, **File** → **Recent Files** menu item, and **Edit** toolbar group (with **Undo** and **Redo** buttons).

- **SEPARATOR**

For menu items, draws a horizontal line.

For toolbar items, draws a vertical line.

- **TOGGLE**

Indicates that the item will have an ON/OFF value associated to it (*i.e.*, the item will act as a check box).

It must define 2 sub-items: the first one specifying action and updater for when the item is ON, and the second one specifying `action` for when the item is OFF.

Examples: **View** → **Status Bar** and **View** → **Show Panels** menu items, and **Orthographic View** and **Player Preview Mode** toolbar buttons.

A — For the list of available tool modes and their description, see the document Content Tools Reference Guide's section WorldEditor → "Panel summary".

B — Sub-items are defined as having type tag set to `CHILD`.

- **updater**

Name of a C++ function, Python method, or expression enquiring value of a tag in options.xml that returns an integer value — if 0 is returned, then the item will be disabled, otherwise it will be enabled (for CHILD items of CHOICE items, if 0 is returned, then the item is cleared/unchecked, otherwise it is selected/checked).

options.xml is located in bigworld/tools/worldeditor — for details on this file's grammar, see “WorldEditor” on page 90)

The possible values for this tag are described below:

- **Type: C++ function**

Defined in the tool's source code.

Example: updateUndo (in MainMenu→Edit→Undo)

- **Type: Python method**

Defined in any Python script located in bigworld\tools\worldeditor\resources\scripts.

Please note that if the method is defined in UIExt.py, then the method need not be appended by the module name. For example, to declare UIAdapter.py's method updateSelectionFilter as the updater, you would have to specify the value UIAdapter.updateSelectionFilter.

- **Type: Expression enquiring value of a tag in options.xml**

Defined inline (using C syntax)

Example: camera/ortho == 0 (in MainToolBar→Edit→ViewOrtho→ShowOrthoMode)

Chapter 18. LightMapSection

The flora and sky configuration files have a `light_map` section, specifying the settings for how the respective element should have light applied to it.

For more details, see the document Client Programming Guide's section *3D Engine (Moo)* → “Features” → “Lighting”.

The grammar for the **LightMapSection** is described below:

```
<light_map>

  <material>          file      <material>
  <width>             integer   <width>
  <height>            integer   <height>
  <timeToleranceInSecs> integer   <timeToleranceInSecs>
  ?<textureFeedName>  string    <textureFeedName>
  <effectTextureName> string    <effectTextureName>
  <effectTransformName> string   <effectTransformName>

</light_map>
```

Grammar of **LightMapSection** in `<flora>.xml` and `<sky>.xml`

The list below describes the tags in **LightMapSection**:

- **effectTextureName**

Name of the automatic **Effect** variable that this light map's Texture property will be exposed to the effect files as.

- **effectTransformName**

Name of the automatic **Effect** variable that this light map's World to Texture Transform will be exposed to the effect files as.

- **height**

Height of the light map in pixels. This should usually have the same value as `<width>`.

- **material**

Effect material file that the engine will use to draw the light map.

- **textureFeedName**

Texture name. Light maps are registered as texture feeds, and this entry specifies the name of the texture feed.

- **timeToleranceSecs**

Delay in seconds of game time between light map updates. Set this to 0 to update the light map for every frame.

- **width**

Width of the light map in pixels

Chapter 19. material_kinds.xml

Defines material kinds to be displayed in ModelEditor's Materials Settings panel's Material Kind drop-down list box — for details, see the Content Tools Reference Guide, section “Panel summary”, “Materials Settings panel”.

Located under `bigworld/res/system/data`, the grammar is listed below:

```
<root>

  *<kind>
    <id>      integer  </id>
    <desc>     string   </desc>
    <sound>    string   </sound>
    <help>     string   </help>
    <sfx>      file     </sfx>
    <weight>   weight   </weight>
    *<terrain> file     </terrain>
  </kind>

</root>
```

Grammar of `bigworld/res/system/data/material_kinds.xml`

The list below describes the tags in `bigworld/res/system/data/material_kinds.xml`:

- **desc**

Name to be displayed for the material kind.

- **sound**

Name of the sound tag to be triggered by footsteps.

- **help**

Help text to be displayed for the material kind.

- **id**

Unique numeric identifier for the material kind.

- **weight**

Weight for all of this material kind's terrain texture maps. When the dominant terrain texture is calculated for any XZ position in a terrain, the associated weighting for textures are taken into account. This influences which material kind is returned for a given location, and also influences the flora that grows at a given location. The material kind weight can be overridden on a per-terrain texture basis, see the terrain section below. If the weight is not specified, it defaults to 1.0

- **sfx**

Special effect file associated with the material.

This is currently not implemented.

- **terrain**

Textures associated with this type of material.

It is given by the texture file path, without extension.

Additionally, each terrain texture may have an associated weight. This overrides the material kind's weight (see above.) If the weight is not specified, it defaults to the material kind's weight.

Chapter 20. .mfm

Located under various sub-folders in the *<res>* tree, the grammar of the material file is illustrated below:

```
<file_name>  
  EffectMaterial ❶  
</file_name>
```

Grammar of *<material>.mfm*

❶ See *EffectMaterial* section on page 35 .

The list below describes the tags in the *<material>.mfm*:

- **EffectMaterial**

For details, see *EffectMaterial* section on page 35 .

Chapter 21. .model

Defined in various sub-folders under the resource tree *<res>* (for example, *<res>/environments*, *<res>/flora*, *<res>/sets/vehicles*, etc...), the .model file format specification is illustrated below:

```
<root>

?<parent>          file.model    </parent>
?<extent>          float         </extent>
[<nodefullVisual>  file.visual    </nodefullVisual>
|<nodelessVisual>  file.visual    </nodelessVisual>
|<billboardVisual> file          </billboardVisual>
]

  <!-- if nodeless or nodefull -->
?<batched>  false </batched>

  <!-- if billboard -->
[<source>
  +<model>    file.model    </model>
  ?<width>    float         </width>
  ?<height>   float         </height>
  *<dye>
    <matter>      string          </matter>
    <tint>         string          </tint>
    *<property_name> float float float float </property_name>
  </dye>
</source>
|
<boundingBox>
  ?<min>  float float float </min>
  ?<max>  float float float </max>
</boundingBox>
]

?<material>
  EffectMaterial 1
</material>

<!-- endif -->

*<animation>

  <!-- if nodefullVisual -->
?<name>      string          </name>
  <nodes>     string          </nodes>
?<firstFrame> integer        </firstFrame>
?<lastFrame>  integer        </lastFrame>
?<alpha>
  *<nodeName> float          </nodeName>
</alpha>
?<cognate>    string          </cognate>

  <!-- if nodelessVisual -->
  <name>       string          </name>
+<visual>     file.visual      </visual>

  <!-- if billboardVisual -->
  <name>       string          </name>
  <frameCount> integer        </frameCount>
```



```

    <! endif !>

    ?<frameRate>      float      </frameRate>

</animation>

*<action>

    <name>            string      </name>
    ?<animation>      string      </animation>
    ?<blendInTime>    float      </blendInTime>
    ?<blendOutTime>   float      </blendOutTime>
    ?<filler>         [true|false] </filler>
    ?<blended>        [true|false] </blended>
    ?<track>          integer     </track>
    ?<isMovement>     [true|false] </isMovement>
    ?<isCoordinated>  [true|false] </isCoordinated>
    ?<isImpacting>    [true|false] </isImpacting>

    ?<match>

        ?<trigger>

            ?<minEntitySpeed>    float      </minEntitySpeed>
            ?<maxEntitySpeed>    float      </maxEntitySpeed>
            ?<minEntityAux1>     float      </minEntityAux1>
            ?<maxEntityAux1>     float      </maxEntityAux1>
            ?<minModelYaw>       float      </minModelYaw>
            ?<maxModelYaw>       float      </maxModelYaw>
            ?<capsOn>            list_of_ints </capsOn>
            ?<capsOff>           list_of_ints </capsOff>
        </trigger>

        ?<cancel>

            ?<minEntitySpeed>    float      </minEntitySpeed>
            ?<maxEntitySpeed>    float      </maxEntitySpeed>
            ?<minEntityAux1>     float      </minEntityAux1>
            ?<maxEntityAux1>     float      </maxEntityAux1>
            ?<minModelYaw>       float      </minModelYaw>
            ?<maxModelYaw>       float      </maxModelYaw>
            ?<capsOn>            list_of_ints </capsOn>
            ?<capsOff>           list_of_ints </capsOff>
        </cancel>

        ?<scalePlaybackSpeed> [true|false] </scalePlaybackSpeed>
        ?<feetFollowDirection> [true|false] </feetFollowDirection>
        ?<oneShot>             [true|false] </oneShot>
        ?<promoteMotion>       [true|false] </promoteMotion>

    </match>

</action>

*<dye>
    <matter>      string </matter>
    <replaces>    string </replaces>

*<tint>
    <name>        string </name>
    <! if nodefullVisual or nodelessVisual !>
    <material>
        EffectMaterial 2
    </material>

```

```

    *<property>
      <name>      string          </name>
      ?<controls> integer          </controls>
      ?<mask>      integer          </mask>
      ?<future>    integer          </future>
      ?<default>   float float float float </default>
    </property>
    <!-- if billboardVisual -->
    *<dye>
      <matter>    string </matter>
      <tint>       string </tint>
      *PropertiesList
    </dye>
  </tint>
</dye>

</root>

```

Grammar of model file

- 1 For details, see *EffectMaterial* section on page 35 .
- 2 For details, see *EffectMaterial* section on page 35 .

The list below describes the tags in the model file:

- **action¹**
Tag for action section.
- **alpha (section animation) — If nodefullVisual**
Tag for animation layer data.
- **animation (section action)**
Name of animation played by action.
- **batched — If nodefullVisual or nodelessVisual**
Indicates that the model allows batch rendering. This can improve performance if there are several instances of a model appearing in the game.
- **billboardVisual**
Name of the texture file for the billboard.
- **blended (section action)**
Deprecated. Tag <track> should be used instead.
- **blendInTime (section action)**
Time to blend in the animation.
- **blendOutTime (section action)**
Time to blend out the animation.
- **boundingBox — If billboardVisual**
Minimum and maximum XYZ coordinates of the model's bounding box.

- **cancel¹ (section action/match)**

Group of conditions that must be met before Action Matcher cancels the action.

- **capsOff¹**

- **Section action/match/cancel**

User-defined flags that cannot be matched to cancel the action.

- **Section action/match/trigger**

User-defined flags that cannot be matched to trigger the action.

Group of conditions that must be met before Action Matcher cancels the action.

- **capsOn¹**

- **Section action/match/cancel**

User-defined flags that must be matched to cancel the action.

- **Section action/match/trigger**

User-defined flags that must be matched to trigger the action.

Group of conditions that must be met before Action Matcher cancels the action.

- **cognate (section animation) — If nodefullVisual**

Name of this animation's cognate animation.

Cognate animations are used when models need to coordinate their movements.

- **dye (section source) — If billboardVisual**

Dye to use for generating the billboard source model.

- **EffectMaterial (section material) — If billboardVisual**

For details, see *EffectMaterial* section on page 35 .

- **extent**

Maximum distance from camera in which model will still be drawn.

- **feetFollowDirection¹² (section action/match)**

Setting this flag to TRUE means that when the action is played the model should be turned so that the direction of motion of its entity should be matched up with the direction of motion of the action.

- **filler (section action)**

If set to TRUE, the animation is repeated automatically. Also known as 'loop'

- **firstFrame (section animation) — If nodefullVisual**

First frame to be played.

- **frameCount (section animation) — If billboardVisual**

Number of frames in billboard's animation.

- **frameRate (section animation)**

Preferred playback frame rate of animation.

- **height (section source) — If billboardVisual**

Height of billboard-type model.

- **isCoordinated¹² (section action)**

TRUE if the animation is part of a sequence that is to be coordinated with a corresponding action on another model, and this animation does not start at the origin.

- **isImpacting¹² (section action)**

TRUE if the movement in the root node of the animation should be promoted to impact on the model. Additionally, if isImpacting is true, and the model is owned by a client-controlled Entity (such as the player), then the translation part of the action will be promoted to impact the entity's position.

- **isMovement¹² (section action)**

TRUE if the animation contains built-in movement, such as a walk or run cycle.

The change in transform of the root node from the beginning to end is recorded, then proportionally subtracted from each frame. This means that if you turn this flag on, then a run animation that does not run on the spot will appear to.

This option allows you to also use the **scalePlaybackSpeed** option.

- **lastFrame² (section animation)**

Last frame to be played.

- **match¹ (section action)**

Group of tags specifying when Action Matcher should trigger or cancel an action.

- **material — If billboardVisual**

Tag for material section of model. Contains settings for shader, collision flags, map files, etc...

- **matter — If billboardVisual**

Name of model's matter.

- **max (section boundingBox)**

Maximum extents of model's bounding box

- **maxEntityAux1¹ (sections action/match/cancel and action/match/trigger)**

Maximum value of some arbitrary attribute that entity must have for the action to be cancelled or triggered.

- **maxEntitySpeed¹ (section action/match/cancel and action/match/trigger)**

Maximum matching speed to cancel or trigger the action.

- **maxEntityYaw¹** (section **action/match/cancel** and **action/match/trigger**)
Maximum yaw of the model in relation to the entity for the action to be cancelled or triggered.
- **min** (section **boundingBox**)
Minimum extents of model's bounding box
- **minEntityAux1¹** (section **action/match/cancel** and **action/match/trigger**)
Minimum value of some arbitrary attribute that entity must have for the action to be cancelled or triggered.
- **minEntitySpeed¹** (section **action/match/cancel** and **action/match/trigger**)
Minimum matching speed to cancel or trigger the action.
- **minEntityYaw¹** (section **action/match/cancel** and **action/match/trigger**)
Minimum yaw of the model in relation to the entity for the action to be cancelled or triggered.
- **model** (section **source**) — If **billboardVisual**
The billboard's source model.
- **name**
 - **Section animation**
Name of animation file containing animation data.
 - **Section action¹**
Name of action.
- **nodefullVisual**
Name of the visual file containing the model's geometry.
- **nodelessVisual**
Name of the visual file containing the model's geometry.
- **nodeName¹** (section **animation/alpha**)
Name of node, defining section with node's blend alpha information.
- **nodes** (section **animation**) — If **nodefullVisual**
Name of animation file containing raw keyframe data.
- **oneShot¹²** (section **action/match**)
Setting this to TRUE means that the Action Matcher will not continue playing that action past one cycle of it, if it is no longer triggered, but a <cancel> section was keeping it active.
- **parent**
Name of parent model file, from which it inherits actions and animations.
- **promoteMotion¹²** (section **action/match**)

This means that the motion in the action is promoted to apply to the model on which it is played. Note that this option must be turned on for **isImpacting** or **isMovement** actions to work correctly. Also, please note that this option will do nothing unless the action **isImpacting** or **isMovement**. Please see the comments in **isImpacting** or **isMovement** for more information about motion promotion.

- **property_name (section source/dye) — If billboardVisual**

Generic dye property.

- **scalePlaybackSpeed¹² (section action/match)**

Setting this flag to TRUE causes the speed of the animation to be scaled by the straight-line speed of the entity.

- **source — If billboardVisual**

Tag for source section. This section specified the source file, dimensions, dye, and bounding box for the billboard-type model.

- **tint (section source/dye) — If billboardVisual**

Name of model's tint for generating the billboard.

- **track¹² (section action)**

The track the action plays on. Actions playing on the same track interrupt each other. There can only be one action playing at a time on each track (unless it is an old action blending out).

- **trigger¹ (section action/match)**

Group of conditions that must be met before Action Matcher triggers the action.

- **umbraOccluder**

TRUE if the model is used as an umbra occluder, this only has an effect when Umbra is running in software mode.

- **visual (section animation) — If nodelessVisual**

Alternate visual used by this animation

- **width (section source) — If billboardVisual**

Width of billboard-type model.

Chapter 22. modules.xml

Used by client tools WorldEditor, ModelEditor, and ParticleEditor, and located under `bigworld/tools/<tool_folder>/resources/data`, this file is used to determine which modules should be run for the tool.

Its grammar is listed below:

```
<root>

  <startup>
    *<module>      string  </module>
  </startup>

  <Bases>
    *<section>      string  </section>
  </Bases>

  <Extensions>
    *<ext_name>
      [ ScriptedModule
        <script>
          <module> string  </module>
          <class>  string  </class>
        </script>
        |
        module_name
      ]
    </Extensions>

</root>
```

Grammar of `<tool_folder>/resources/data/modules.xml`

The list below describes the tags in `bigworld/tools/<tool_folder>/resources/data/modules.xml`:

- **class (section Extensions/ext_name/ ScriptedModule/script)**

Name of the Python class to use for this module.

- **Extensions**

Section containing all modules that can be used by the application.

- **ext_name (section Extensions)**

Section containing information for the module.

- **module (section startup)**

- **Section startup**

Module to use on startup of the application.

This tag is one of the `ext_name` tags in Extensions section.

- **Section Extensions/ext_name/ ScriptedModule/script**

Name (without extension) of the Python file to use for this module.

It is expected that the class specified in `class` tag is located in this file.

- **module_name** (section **Extensions/ext_name**)

Name of the module to use (referred to in `startup/module` tag)

- **script** (section **Extensions/ext_name/ ScriptedModule**)

Description of the Python class to use for this module.

- **ScriptedModule** (section **Extensions/ext_name**)

Used to denote that the description following it is for a Python script.

Chapter 23. .mvl

MVL files are specific to ModelEditor. They are only used to view models under different lighting conditions. You can create, load, save, and configure them as required, but they are only used in ModelEditor. WorldEditor uses XML files to describe lights.

Mentioned in document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “Materials Settings panel”.

The grammar for the MVL light configuration files is described below:

```
<root>
  <Lights>
    +<Light>
      ?<Type>      [Ambient|Directional|Omni|Spot|Empty] </Type>
      ?<Color>     float float float                  </Color>
      ?<Enabled>   [true|false]                      </Enabled>

      <!-- If Type != Ambient -->
      <Location> float float float                    </Location>

      <!-- If Type = Directional or Type = Spot -->
      ?<Orientation> float float float </Orientation>

      <!-- If Type = Omni or Type = Empty -->
      ?<Full_Radius> float </Full_Radius>
      ?<Falloff_Radius> float </Falloff_Radius>

      <!-- If Type = Spot -->
      ?<Cone_Size> float </ConeSize>

    </Light>
  </Lights>
</root>
```

Grammar of MVL light configuration file

The list below describes the tags in the MVL light configuration file:

- **Color (Available for Ambient, Directional, Omni, and Spot)**
Colour of the light,
- **Cone_Size (Available for Spot)**
Size of the cone of light generated by spot light.
- **Enabled (Available for Ambient, Directional, Omni, and Spot)**
Boolean specifying if light is activated.
- **Falloff_Radius (Available for Omni)**
Total area influenced by light.
Must be greater than Full_Radius.
- **Full_Radius (Available for Omni)**
Area over which the light will be at full intensity.

- **Light**

Section containing settings for a specific light.

- **Lights**

Section containing definition of one or more **Light** sections.

- **Location (Available for Directional, Omni, and Spot)**

Point of origin for the light.

- **Orientation (Available for Directional and Spot)**

Direction at which light will be projected.

- **Type**

Type of light being set. Possible values are:

- **Ambient** — Covers entire scene with omnidirectional light.
- **Directional** — Allows you to direct a wide area of radiance at a particular area or object.
- **Omni** — Radiates from a three-dimensional point, giving more control over the concentration of light in certain areas.
- **Spot** — Produces a small, concentrated, and directed source of light.
- **Empty** — Defaults to Omni.

Chapter 24. navgen_settings.xml

For details on NavGen, see the document Content Tools Reference Guide, section *NavGen* → “Changing settings”.

Located under `bigworld/tools/misc`, the grammar of `navgen_settings.xml` is listed below:

```
<navgen_settings.xml>

  <space>  folder/file  </space>

  +<girth>  float
    ?<always>          </always>
    ?<width>    float  </width>
    ?<height>   float  </height>
    ?<depth>    float  </depth>

    <bwlockd>          server:port  </bwlockd>
    <standalone>       [true|false]  </standalone>
    ?<floodResultPath> folder        </floodResultPath>

</navgen_settings.xml>
```

Grammar of `bigworld/tools/misc/navgen_settings.xml`

The list below describes the tags in `bigworld/tools/misc/navgen_settings.xml`:

- **always (section girth)**

Determines if this girth should always be included into waypoint. If set to false, it is up to the marker to determine whether to include a girth.

- **bwlockd^A**

WorldEditor lock server (bwlockd) server and port. Since NavGen also modifies terrain data, bwlockd server is used to avoid more than one world builder working on the same piece of data.

- **depth (section girth)**

Depth of the entity for which to generate navigation mesh.

- **floodResultPath**

Path where the pre-filter and post-filter TGA files are generated. For details, see the document Content Tools Reference Guide's section *NavGen* → “Generating the navmesh” → “Processing time” → “Adjacency map filtering (blue)”.

- **girth**

Tag for section defining entity girth for which to generate navigation mesh. Default is 0.5.

- **height (section girth)**

Height of the entity for which to generate navigation mesh.

- **space**

Folder containing the space to load. The path is relative to the game's resources folder (`<res>`).

- **standalone**^A

Determines if NavGen should connect to a bwlockd server.

- **width (section girth)**

Width of the entity for which to generate navigation mesh.

A — For details on bwlockd, see the document Content Tools Reference Guide's section Lock Server (BWLockD).

Chapter 25. options.xml

Different tools use a configuration file named `options.xml`. Their grammars are described in the following sections:

- **CAT** — See “CAT” on page 75 .
- **ModelEditor** — See “ModelEditor” on page 75 .
- **ParticleEditor** — See “ParticleEditor” on page 88 .
- **WorldEditor** — See “WorldEditor” on page 90 .

25.1. CAT

For details on the Client Access Tools, see the Client Programming Guide's section *Debugging* → “Client Access Tool (CAT)”.

Located under `bigworld/tools/cat`, the grammar of `options.xml` is listed below:

```
<options.xml>

  ?<additionalSearchPath>  list_of_folders  </additionalSearchPath>
  ?<scriptDirectory>      list_of_folders  </scriptDirectory>

</options.xml>
```

Grammar of `bigworld/tools/cat/options.xml`

25.2. ModelEditor

For details on the ModelEditor, see the document Content Tools Reference Guide's section *ModelEditor*.

Located under `bigworld/tools/modeleditor`, the grammar of `options.xml` is listed below:

```
<root>
  <help>
    <toolsReferenceGuide>  folder/file  </toolsReferenceGuide>
    <contentCreationManual> folder/file  </contentCreationManual>
    <shortcutsHtml>        folder/file  </shortcutsHtml>
  </help>

  <app>
    <maxTimeDelta>  float  </maxTimeDelta>
    <timeScale>     float  </timeScale>
  </app>

  <consoles>
    <darkenBackground> [true|false] </darkenBackground>
    <numMessageLines>  integer      </numMessageLines>
  </consoles>

  <space>  folder/file  </space>

  <ualConfigPath> folder/file </ualConfigPath>

  <camera>
    <speed>  {Slow|Medium|Fast|SuperFast}
    <{Slow|Medium|Fast|SuperFast}> integer
    <turbo>  integer  </turbo>
```

```

    </{Slow|Medium|Fast|SuperFast}>
  </speed>
  <invert>      [0|1]      </invert>
  <mode>        [0|1|2|3|4] </mode>
  <orbitSpeed>  float      </orbitSpeed>
  <rotDir>      [-1|1]     </rotDir>
</camera>

<input>
  <exclusive> [true|false] </exclusive>
</input>

<graphics>
  <bpp>        [16|32]     </bpp>
  <fullScreen> [true|false] </fullScreen>
  <height>      integer     </height>
  <shadows>     [true|false] </shadows>
  <width>       integer     </width>
</graphics>

<render>
  <environment>      [0|1]
    <drawFog>         [0|1] </drawFog>
    <drawShimmer>     [0|1] </drawShimmer>
    <drawBloom>       [0|1] </drawBloom>
    <drawClouds>      [0|1] </drawClouds>
    <drawSky>         [0|1] </drawSky>
    <drawSunAndMoon>  [0|1] </drawSunAndMoon>
    <drawDetailObjects> [0|1] </drawDetailObjects>
    <drawStars>       [0|1] </drawStars>
    <drawStaticSky>   [0|1] </drawStaticSky>
  </environment>
  <misc>
    <nearPlane> float </nearPlane>
    <farPlane>  float </farPlane>
  </misc>
  <scenery>
    <wireFrame> [0|1] </wireFrame>
    <drawWater>
      <refraction> [0|1] </refraction>
      <reflection> [0|1] </reflection>
      <simulation> [0|1] </simulation>
    </drawWater>
  </scenery>
  <lighting> [0|1|2] </lighting>
  <terrain>  [0|1]
    <wireFrame> [0|1] </wireFrame>
  </terrain>
</render>

<renderer>
  <shadows>
    <quality> [0|1|2] </quality>
  </shadows>
</renderer>

<precompileEffects> [0|1] </precompileEffects>

<settings>
  <animateZoom>      [0|1]      </animateZoom>
  <bkgColour>         float float float </bkgColour>
  <centreModel>      [0|1]      </centreModel>

```

```

<checkForSparkles> [0|1] </checkForSparkles>
<floorGrid> float </floorGrid>
<floorTexture> folder/file </floorTexture>
<floraDensity> [0|1|2|3|4] </floraDensity>
<gameTime> time </gameTime>
<sunAngle> angle </sunAngle>
<groundModel> [0|1] </groundModel>
<lastNewTintFX> [0|1] </lastNewTintFX>
<lockLodParents> [0|1] </lockLodParents>
<regenBBOOnLoad> [0|1] </regenBBOOnLoad>
<showActions> [0|1] </showActions>
<showAxes> [0|1] </showAxes>
<showBinormals> [0|1] </showBinormals>
<showBoundingBox> [0|1] </showBoundingBox>
<showBsp> [0|1] </showBsp>
<showCustomHull> [0|1] </showCustomHull>
<showEditorProxy> [0|1] </showEditorProxy>
<showHardPoints> [0|1] </showHardPoints>
<showLightModels> [0|1] </showLightModels>
<showModel> [0|1] </showModel>
<showNormals> [0|1] </showNormals>
<showOriginalAnim> [0|1] </showOriginalAnim>
<showPortals> [0|1] </showPortals>
<showShadowing> [0|1] </showShadowing>
<showSkeleton> [0|1] </showSkeleton>
<showWireframe> [0|1] </showWireframe>
<useCustomLighting> [0|1] </useCustomLighting>
<useFloor> [0|1] </useFloor>
<useTerrain> [0|1] </useTerrain>
<zoomDuration> float </zoomDuration>
<zoomOnLoad> [0|1] </zoomOnLoad>
</setting>

<startup>
  <lastLoadOK> [true|false] </lastLoadOK>
  <lastOrigin> float float float </lastOrigin>
  <lastView> matrix </lastView>
  <loadLastLights> [0|1] </loadLastLights>
  <loadLastModel> [0|1] </loadLastModel>
  <showSplashScreen> [0|1] </showSplashScreen>
</startup>

<messages>
  <assetMsgs> [0|1] </assetMsgs>
  <errorMsgs> [0|1] </errorMsgs>
  <infoMsgs> [0|1] </infoMsgs>
  <showDate> [0|1] </showDate>
  <showPriority> [0|1] </showPriority>
  <showTime> [0|1] </showTime>
  <noticeMsgs> [0|1] </noticeMsgs>
  <warningMsgs> [0|1] </warningMsgs>
</messages>

<fx>
  <file0> folder/file </file0>
  ...
  <file9> folder/file </file9>
</fx>

<lights>
  <file0> folder/file </file0>
  ...

```

```

    <file9> folder/file </file9>
  </lights>

  <models>
    <file0> folder/file </file0>
    ...
    <file9> folder/file </file9>
  </models>
</root>

```

Grammar of bigworld/tools/modeleditor/options.xml

The list below describes the tags in bigworld/tools/modeleditor/options.xml:

- **animateZoom (section settings)**

Determines whether the zoom to model's extents should be animated.

This value is set by the **ModelEditor Preferences** dialog box's **Animate Zoom To Extents** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Dialog boxes” → “Preferences dialog box”).

- **assetMsgs (section messages)**

Determines whether asset messages are displayed in the **Messages** panel (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “BigWorld Messages panel”).

- **bpp (section graphics)**

Specifies the bit depth to use for the display device.

- **centreModel (section settings)**

Centres the model's bounding box at the origin.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Centre Model** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “Display Settings panel”).

- **checkForSparkles (section settings)**

Overrides all other display settings, and displays the model against a white background with white ambient lighting.

This value is set by ModelEditor's **Display Settings** panel's **General** group box's **Check For Sparkles** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “Display Settings panel”).

- **contentCreationManual (section help)**

Path and filename of the Content Creation Manual — the path is relative to the executable's folder (bigworld/tools/modeleditor).

The document is accessed by pressing F1 or by selecting the **Help → Content Creation** menu item (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Menu items”).

- **darkenBackground (section consoles)**

Determines whether the console's background should be darkened, for better readability (for details on the consoles available in ModelEditor, see the document Content Tools Reference Guide's sections *Tools Consoles* and *ModelEditor* → “Keyboard shortcuts”).

- **bkgColour (section settings)**

Colour to be displayed on the background.

- **drawBloom (section render/environment)**

Toggles the bloom full screen effect.

This value is set by ModelEditor's **Display Settings** panel's **General** group box's **Show Bloom** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **drawClouds (section render/environment)**

Toggles the rendering of clouds.

- **drawDetailObjects (section render/environment)**

Toggles the rendering of details object (*e.g.*, flora).

- **drawFog (section render/environment)**

Toggles the rendering of fog.

- **drawShimmer (section render/environment)**

Toggles the shimmer full screen effect.

This value is set by ModelEditor's **Display Settings** panel's **General** group box's **Show Shimmer** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **drawSky (section render/environment)**

Toggles the rendering of sky.

- **drawStars (section render/environment)**

Toggles the visibility of stars.

- **drawStaticSky (section render/environment)**

Toggles the visibility of static sky.

- **drawSunAndMoon (section render/environment)**

Toggles the rendering of sun and moon objects.

- **drawWater (section render/scenery)**

Toggles the rendering of water.

- **environment (section render)**

Toggles the visibility of all selected environment items.

- **errorMsgs (section messages)**

Determines whether error messages are displayed in the **Messages** panel (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "BigWorld Messages panel").

- **exclusive (section input)**

Determines whether ModelEditor should work in exclusive mode.

- **farPlane (section render/misc)**

Specifies the far plane distance. Objects further then this distance will not be rendered.

- **floorGrid (section settings)**

Specifies the size of the floor's grid.

- **floorTexture (section settings)**

Specifies which texture to use for the floor.

This value is set by ModelEditor's **Display Settings** panel's **Background** group box's **Choose Floor Texture** button (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **floraDensity (section settings)**

Specifies the density of the flora to be rendered alongside terrain.

This value is set by ModelEditor's **Display Settings** panel's **Background** group box's **Flora Density** drop-down list (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "BigWorld Messages panel" → "Display Settings panel").

- **fullscreen (section graphics)**

Specifies whether to display in fullscreen mode.

- **fx**

Lists the ten last most recently used .fx files.

- **gameTime (section settings)**

Specifies the current time of day in the world.

This value is set by ModelEditor's **Display Settings** panel's **Time of Day** slider and the **Lighting Setup** panel's **Game Lighting** slider (for details, see the document Content Tools Reference Guide's sections *ModelEditor* → "Panel summary" → "BigWorld Messages panel" and *ModelEditor* → "Panel summary" → "Lighting Setup panel").

- **sunAngle (section settings)**

Specifies the sun angle when using time of day lighting.

- **graphics**

Section specifying graphic card's options such as screen resolution, bit depth, fullscreen settings, etc...

- **groundModel (section settings)**

Positions the model's bounding box on the ground.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Lock to Ground** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "BigWorld Messages panel" → "Display Settings panel").

- **height (section graphics)**

Specifies the screen height resolution to use for the display device.

- **infoMsgs (section messages)**

Toggles whether information messages are displayed in the **Messages** panel (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "BigWorld Messages panel").

- **invert (section camera)**

Specifies whether to invert the camera orientation for the mouse right button's up and down movements.

This value is set by **ModelEditor Preferences** dialog box's **Invert Mouse** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Dialog boxes" → "Preferences dialog box").

- **lastLoadOk (section startup)**

Indicates whether ModelEditor ran successfully last time it was used. If this value is set to 0, then ModelEditor will not load the last loaded model and lighting setup on startup.

- **lastNewTintFX (section settings)**

Specifies that the last new Tint to be created in ModelEditor's **Materials Settings** panel was a Tint, and not a MFM (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Materials Settings panel").

- **lastOrigin (section startup)**

The position of the camera's origin saved from the last session.

- **lastView (section startup)**

The camera's view saved from the last session.

- **lighting (section render)**

Specifies how lighting should be rendered.

Possible values are:

- 0 — Standard
- 1 — Dynamic
- 2 — Specular

- **lights**

Lists the ten last most recently used lighting setup files.

- **loadLastLights (section settings)**

Specifies whether to automatically load on startup the light file used on last session.

This value is set by **ModelEditor Preferences** dialog box's **On Startup** group box's **Load Last Lights** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Dialog boxes" → "Preferences dialog box").

- **loadLastModel (section settings)**

Specifies whether to automatically load on startup the model file used on last session.

This value is set by **ModelEditor Preferences** dialog box's **On Startup** group box's **Load Last Model** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Dialog boxes” → “Preferences dialog box”).

- **lockLodParents (section settings)**

Specifies whether LOD parents of the loaded model should be read-only.

This value is set by **ModelEditor Preferences** dialog box's **Lock LOD Parents** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Dialog boxes” → “Preferences dialog box”).

- **maxTimeDelta (section app)**

Maximum number of seconds elapsed between the last frame and the current one when calculating frame time.

If the delta is bigger than `maxTimeDelta`, then it is set to `maxTimeDelta` for frame time calculation purposes.

- **mode (section camera)**

Determines which camera mode to use (for details on camera modes, see the document Content Tools Reference Guide's section *ModelEditor* → “Toolbar”).

- **models**

Lists the ten last most recently used model files. This list is used in the **File** → **Recent Models** menu item (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Menu items”).

- **nearPlane (section render/misc)**

Specifies the near plane distance. Objects closer then this distance will not be rendered.

- **noticeMsgs (section messages)**

Determines whether notice messages are displayed in the **Messages** panel (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “BigWorld Messages panel”).

- **numMessageLines (section consoles)**

Maximum number of short-lived error or warning messages displayed at the bottom of the viewport.

- **orbitSpeed (section camera)**

Determines the camera's speed of rotation when in **Orbit** camera mode (for details on camera modes, see the document Content Tools Reference Guide's section *ModelEditor* → “Toolbar”).

- **precompileEffects**

Determines whether `.fx` files should be compiled before ModelEditor first runs or should be built when first used.

- **quality (section renderer/shadows)**

Determines the shadow quality.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Shadowing Settings** drop-down list box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "BigWorld Messages panel" → "Display Settings panel").

- **reflection (section render/scenery/drawWater)**

Toggles water reflection.

- **regenBBOOnLoad (section settings)**

Specifies whether a visibility box should be automatically generated upon load for models that do not have one.

This value is set by **ModelEditor Preferences** dialog box's **On Model Load** group box's **Regenerate Visibility Box** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Dialog boxes" → "Preferences dialog box").

- **rotDir (section camera)**

Determines whether the camera will rotate clockwise(1) or anti-clockwise(-1) when in **Orbit** camera mode (for details on camera modes, see the document Content Tools Reference Guide's section *ModelEditor* → "Toolbar").

- **scenery (section render)**

Toggles the visibility of scenery items.

- **shadows (section graphics)**

Toggles the creation of a stencil buffer.


- **shortcutsHtml (section help)**

The location of the shortcuts HTML — the path is relative to the executable's folder (bigworld/tools/modeleditor).

Path and filename of the Content Creation Manual — the path is relative to the executable's folder bigworld/tools/modeleditor (the document accessed by selecting the **Help** → **Shortcuts** menu item — for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Menu items").

- **showActions (section settings)**

Specifies whether ModelEditor should graph on the viewport all actions being played on the model, with their respective blend weights.

This value is set by ModelEditor's **Actions** panel's  button (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "BigWorld Messages panel" → "Actions panel").

- **showAxes (section settings)**

Specifies whether to render the X-, Y-, and Z-axes.

This value is set by ModelEditor's **Display Settings** panel's **General** group box's **Show Axes** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **showBinormals (section settings)**

Specifies whether the binormals for each of the model's vertices should be displayed.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show Vertex Binormals** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **showBoundingBox (section settings)**

Specifies whether to render the model's bounding box (yellow) and visibility box (blue).

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show Bounding Boxes** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **showBsp (section settings)**

Specifies whether to render the model's BSP.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show BSP** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **showCustomHull (section settings)**

Specifies whether to display a shell's custom hull.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show Custom Hull** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **showDate (section messages)**

Determines whether the date of the issued messages should be displayed in the **Messages** panel.

This value is set by the **Messages** panel's **Show Date** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "BigWorld Messages panel").

- **showEditorProxy (section settings)**

Specifies whether to display the proxy model (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Object Properties panel").

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show Editor Proxy** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **showHardPoints (section settings)**

Specifies whether to display the model's hard points — each one will be rendered alongside its name and XYZ axes.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show Hard Points** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **showLightModels (section settings)**

Specifies whether to display gizmo representing the light source.

This value is set by ModelEditor's **Lighting Setup** panel's **Show Light Axes** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “Lighting Setup panel”).

- **showModel (section settings)**

Specifies whether to render the model.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show Model** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “Display Settings panel”).


- **showNormals (section settings)**

Specifies whether the normals for each of the model's vertices should be displayed.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show Vertex Normals** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “Display Settings panel”).

- **showOriginalAnim (section settings)**

Specifies whether to produce a non-compressed wireframe animation of the model, played along a fully-rendered compressed one.

This value is set by ModelEditor's **Animations** panel's **Compression Settings** button —  — (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “BigWorld Messages panel” “Animations panel”).

- **showPortals (section settings)**

Specifies whether to display the shell's portals.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show Portals** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “Display Settings panel”).

- **showPriority (section messages)**

Determines whether the priority of the issued messages should be displayed in the **Messages** panel.

This value is set by the **Messages** panel's **Show Priority** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “BigWorld Messages panel”).

- **showShadowing (section settings)**

Specifies whether to show shadows.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Shadowing Settings** drop-down list box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “BigWorld Messages panel” “Display Settings panel”).

- **showSkeleton (section settings)**

Specifies whether to display the model's skeleton.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show Skeleton** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **showSplashScreen (section startup)**

Specifies whether to display the application's splash screen.

This value is set by **ModelEditor Preferences** dialog box's **On Startup** group box's **Show Splash Screen** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Dialog boxes" → "Preferences dialog box").

- **showTime (section messages)**

Determines whether the time of the issued messages should be displayed in the **Messages** panel.

This value is set by the **Messages** panel's **Show Time** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "BigWorld Messages panel").

- **showWireframe (section settings)**

Specifies whether to display the model as a mesh.

This value is set by ModelEditor's **Display Settings** panel's **Model** group box's **Show Wireframe** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Panel summary" → "Display Settings panel").

- **simulation (section render/environment/drawWater)**

Toggles water simulation.

- **{Slow|Medium|Fast|SuperFast} (section camera/speed)**

Definition of normal speed (in *m/s*) for the 4 pre-defined camera speeds.

This value is set by clicking the **Camera Speed** toolbar buttons (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Toolbar"), or by the Ctrl+1, Ctrl+2, Ctrl+3, and Ctrl+4 keyboard shortcuts (for details, see the document Content Tools Reference Guide's section *ModelEditor* → "Keyboard shortcuts").

- **space**

Folder containing the space to load.

The path is relative to the specified resource folder (bigworld/res).

- **terrain (section render)**

Toggles the visibility of the terrain.

- **timeScale (section app)**

Number of seconds in real world time corresponding to each second in game time.

This value is used to calculate the time elapsed between last frame and current one.

- **toolsReferenceGuide (section help)**

Path and filename of the Content Tools Reference Guide — the path is relative to the executable's folder (bigworld/tools/modeleditor).

The document accessed by pressing F1 or by selecting the **Help → Tools Reference Guide** menu item — for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Menu items”).

- **turbo (section camera/speed/speed_label)**

Definition of turbo speed (in *m/s*) for the 4 pre-defined camera speeds.

The camera speed is set by clicking the **Camera Speed** toolbar buttons (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Toolbar”), or by the Ctrl+1, Ctrl+2, Ctrl+3, and Ctrl+4 keyboard shortcuts (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Keyboard shortcuts”).

- **ualConfigPath**

Path of the Asset Browser's configuration file — the path is relative to the executable's folder (bigworld/tools/modeleditor).

For details on this file, see the document Content Tools Reference Guide's section *Asset Browser* → “Customisation”).

- **useCustomLighting (section settings)**

Specifies whether to apply custom lighting to the model.

This value is set by ModelEditor's **Lighting Setup** panel's **Custom Lighting** option button (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “Lighting Setup panel”).

- **useFloor (section settings)**

Specifies whether to display a floor texture beneath the model.

This value is set by ModelEditor's **Display Settings** panel's **Background** group box's **Background Option** drop-down list box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “BigWorld Messages panel” → “Display Settings panel”).

- **useTerrain (section settings)**

Specifies whether to display terrain beneath the model.

This value is set by ModelEditor's **Display Settings** panel's **Background** group box's **Background Option** drop-down list box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “BigWorld Messages panel” → “Display Settings panel”).

- **warningMsgs (section messages)**

Determines whether warning messages should be displayed in the **Messages** panel (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Panel summary” → “BigWorld Messages panel”).

- **width (section graphics)**

Specifies the width resolution to use for the display device.

- **wireFrame**


- **Section render/scenery**

Toggles the display of scenery items as wireframe.

- **Section render/terrain**

Toggles the display of the terrain as wireframe.

- **zoomDuration (section settings)**

Determines the duration of the animation played when the **Zoom To Extents** toolbar button  is clicked (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Toolbar”).

- **zoomOnLoad (section settings)**

Specifies whether to automatically zoom to show the entire model when loading.

This value is set by **ModelEditor Preferences** dialog box's **On Model Load** group box's **Zoom To Extents** check box (for details, see the document Content Tools Reference Guide's section *ModelEditor* → “Dialog boxes” → “Preferences dialog box”).

25.3. ParticleEditor

For details on the ParticleEditor, see the document Content Tools Reference Guide's section *ParticleEditor*.

Located under bigworld/tools/particleeditor, the grammar of options.xml is listed below:

```
<root>
  <space>  folder/file  </space>
  <app>
    <maxTimeDelta>  float  </maxTimeDelta>
  </app>
  <romp>
    <watcherValues>
      <Client Settings/Time of Day>  time  </Client Settings/Time of Day>
      <Client Settings/Secs Per Hour>  int  </Client Settings/Secs Per Hour>
    </watcherValues>
  </romp>

  <camera>
    <speed>  float  </speed>
  </camera>

  <render>
    <misc>
      <drawFog>          [true|false]  </drawFog>
      <drawBloom>         [true|false]  </drawBloom>
      <drawShimmer>       [true|false]  </drawShimmer>
      <drawSunAndMoon>    [true|false]  </drawSunAndMoon>
      <drawClouds>        [true|false]  </drawClouds>
      <drawSky>           [true|false]  </drawSky>
    </misc>
  </render>

  <settings>
    <floorTexture>  folder/file  </floorTexture>
    <floorGrid>     float         </floorGrid>
  </settings>

  <defaults>
    <backgroundColour>  float float float float  </backgroundColour>
    <startDirectory>    folder                </startDirectory>
  </render>
```

```

        <blurTexture>    folder/file  </blurTexture>
        <spriteTexture>  folder/file  </spriteTexture>
        <ampTexture>     folder/file  </ampTexture>
        <meshVisual>     folder/file  </meshVisual>
        <trailTexture>   folder/file  </trailTexture>
    </renderer>
    <flareXML> folder/file          </flareXML>
    <bkgmode>  [Terrain|Floor|None] </drawScene>
</defaults>

<resourceGlue>
    <environment>
        <sunFlareXML> folder/file  </sunFlareXML>
    </environment>
</resourceGlue>

</root>

```

Grammar of bigworld/tools/particleeditor/options.xml

The list below describes the tags in bigworld/tools/particleeditor/options.xml:

- **ampTexture (section defaults/renderer)**

Default texture to be displayed if the **Amp** option button is selected in the **Renderer Properties** sub-panel.

- **backgroundColour (section defaults)**

Colour to be displayed on the background.

- **bkgMode (section defaults)**

Type of background to be displayed on the viewport.

- **blurTexture (section defaults/renderer)**

Default texture to be displayed if the **Blur** option button is selected in the **Renderer Properties** sub-panel.

- **Client Settings/ Secs per Hour (section romp/watcherValues)**

Number of real-time seconds in one game-time hour.

- **Client Settings/ Time of Day (section romp/watcherValues)**

Initial time of day in which to render the background. This determines the sun position and colouration.

- **drawBloom (section render/misc)**

Toggles the use of bloom effect.

- **drawClouds (section render/misc)**

Toggles rendering of clouds.

- **drawFog (section render/misc)**

Toggles rendering of fog.

- **drawShimmer (section render/misc)**

Toggles the use of shimmer effect.

- **drawSky (section render/misc)**

Toggles rendering of the sky.

- **drawSunAndMoon** (section **render/misc**)

Toggles rendering of the sun and the moon.

- **flareXML** (section **defaults**)

Default light configuration file to display when user adds a **Flare** sub-system component in the **Sub System Components** panel's list box

- **floorGrid** (section **settings**)

Size of the tiles of the background floor.

- **floorTexture** (section **settings**)

File containing the image to use as floor background.

- **maxTimeDelta** (section **app**)

Maximum amount of game time that can elapse between two frames.

- **meshVisual** (section **defaults/renderer**)

Default texture to be displayed if the **Mesh** option button is selected in the **Renderer Properties** sub-panel.

- **space**

Folder containing the space to load.

The path is relative to the resource folder (bigworld/res).

- **speed** (section **camera**)

Current camera speed.

- **startDirectory** (section **defaults**)

Folder which particle systems will be displayed in the **Select or Enter a Particle System** panel's drop-down list box.

- **sunFlareXML** (section **resourceGlue/environment**)

XML file to use for the sun flare effect when rendering in background mode.

- **trailTexture** (section **defaults/renderer**)

Default texture to be displayed if the **Trail** option button is selected in the **Renderer Properties** sub-panel.

25.4. WorldEditor

For details on the WorldEditor, see the document Content Tools Reference Guide's chapter *WorldEditor*.

Located under bigworld/tools/WorldEditor, the grammar of options.xml is listed below:

```
<root>

  <userTag>  string  </userTag>

  ?<versionControl>
    <batchLimit>                integer                </batchLimit>
```

```

    <enable> [true|false] </enable>
    <path> folder/file </path>
</versionControl>

<panels>
  <saveLayoutOnExit> [true|false] </saveLayoutOnExit>
</panels>

<help>
  <toolsReferenceGuide> folder/file </toolsReferenceGuide>
  <contentCreationManual> folder/file </contentCreationManual>
  <shortcutsHtml> folder/file </shortcutsHtml>
</help>

<undoredo>
  <limit> integer </limit>
</undoredo>

<snaps>
  <movement> float float float </movement>
  <angle> float </angle>
  <xyzEnabled> [0|1] </xyzEnabled>
  <itemSnapMode> [0|1] </itemSnapMode>
</snaps>

<shellSnaps>
  <movement> float float float </movement>
  <angle> float </angle>
</shellSnaps>

<space> folder
  *<folder> folder </folder>
</space>

<bwlockd>
  <use> [true|false] </use>
  <host> server:port </host>
</bwlockd>

<graphics>
  <graphicsPreferencesXML> folder/file </graphicsPreferencesXML>
  <shadows> [true|false] </shadows>
  <cameraHeight> integer </cameraHeight>
  <farclip> integer </farclip>
  <timeofday> integer </timeofday>
</graphics>

<input>
  <exclusive> false </exclusive>
  <legacyMouseWheel> [true|false] </legacyMouseWheel>
</input>

<consoles>
  <darkenBackground> [true|false] </darkenBackground>
  <numMessageLines> integer </numMessageLines>
</consoles>

<precompileEffects> [0|1] </precompileEffects>

<app>
  <maxTimeDelta> float </maxTimeDelta>
  <timeScale> float </timeScale>

```

```

</app>

<itemEditor>
  <browsePath>    folder/file <browsePath>
  <lastY>         float      </lastY>
</itemEditor>

<tools>
  <alphaTool>          folder/file    </alphaTool>
  <alphaToolSize>       integer       </alphaToolSize>
  <alphaToolStrength>   integer       </alphaToolStrength>
  <alphaToolGUISize>    float         </alphaToolGUISize>
  <chunkVisualisation>  folder/file    </chunkVisualisation>
  <showChunkVisualisation> [true|false] </showChunkVisualisation>
  <selectFilter>        selection_opt </selectFilter>
  <coordFilter>         coord_opt     </coordFilter>
  <optionsPage>         folder/file    </optionsPage>
</tools>

<romp>
  <watcherValues>
    <watcher_name> watcher_value </watcher_name>
  </watcherValues>
</romp>

<camera>
  <speed> {Slow|Medium|Fast|SuperFast}
    <{Slow|Medium|Fast|SuperFast}> integer
    <turbo> integer </turbo>
    </{Slow|Medium|Fast|SuperFast}>
  </speed>
  <ortho> [0|1] </ortho>
</camera>

<object>
  <bb_tab>
    <directory>    folder      </directory>
    ?<selectFilter> selection_opt </selectFilter>
  </bb_tab>
</object>

<dragOnSelect> [0|1] </dragOnSelect>
<drawBSP>      [0|1|2] </drawBSP>

<objects>
  <rememberSelectFilter> [0|1] </rememberSelectFilter>
  <readOnlyMode>         [0|1] </readOnlyMode>
  <materialOverrideMode> [0|1] </materialOverrideMode>
</objects>

<render>
  <hideOutsideObjects> [0|1] </hideOutsideObjects>
  <lighting>           [0|1|2] </lighting>
  <drawChunkPortals>   [true|false] </drawChunkPortals>

  <environment>        [0|1]
    <drawFog>           [0|1] </drawFog>
    <drawShimmer>       [0|1] </drawShimmer>
    <drawBloom>         [0|1] </drawBloom>
    <drawClouds>        [0|1] </drawClouds>
    <drawSky>           [0|1] </drawSky>
    <drawSunAndMoon>    [0|1] </drawSunAndMoon>

```

```

    <drawDetailObjects> [0|1] </drawDetailObjects>
    <drawStars> [0|1] </drawStars>
    <drawStaticSky> [0|1] </drawStaticSky>
</environment>

<scenery> [0|1]
    <wireFrame> [0|1] </wireFrame>
    <particle> [0|1] </particle>
    <drawWater>
        <reflection> [0|1] </reflection>
        <simulation> [0|1] </simulation>
    </drawWater>
    <shells> [0|1]
        <gameVisibility> [0|1] </gameVisibility>
    </shells>
</scenery>

<terrain> [0|1]
    <wireFrame> [0|1] </wireFrame>
    <LOD> [0|1] </LOD>
</terrain>

<lights> [0|1]
    <drawStatic> [0|1] </drawStatic>
    <drawDynamic> [0|1] </drawDynamic>
    <drawSpecular> [0|1] </drawSpecular>
</lights>

<misc> [0|1]
    <shadeReadOnlyAreas> [0|1] </shadeReadOnlyAreas>
    <drawHeavenAndEarth> [0|1] </drawHeavenAndEarth>
    <drawPatrolGraphs> [0|1] </drawPatrolGraphs>
    <drawEditorProxies> [0|1] </drawEditorProxies>
    <drawOverlayLocks> [0|1] </drawOverlayLocks>
</misc>

<project>
    <spaceMapAlpha> float </spaceMapAlpha>
</project>

<chunk>
    <vizMode> [0|1] </vizMode>
</chunk>

<numLayerWarning> [0|1] </numLayerWarning>

<height>
    <selTopLeft> float float </selTopLeft>
    <selBottomRight> float float </selBottomRight>
    <maskBottomRight> float float </maskBottomRight>
    <maskTopLeft> float float </maskTopLeft>
</height>

<maxFarPlane> float </maxFarPlane>

<drawChunkPortals> [true|false] </drawChunkPortals>
<drawVLO> [true|false] </drawVLO>
<drawChunkFlares> [true|false] </drawChunkFlares>
<drawChunkLights> [true|false] </drawChunkLights>
<drawParticleSystems> [true|false] </drawParticleSystems>
<drawEntities> [true|false] </drawEntities>

```

```

<links>
  <maxProcessingMillis>      integer      </maxProcessingMillis>
  <maxRecalcsPerFrame>       integer      </maxRecalcsPerFrame>
  <maxTimeBetweenRecalcs>    float        </maxTimeBetweenRecalcs>
  <maxTimeWaitForMoreChunks> float        </maxTimeWaitForMoreChunks>
  <minLinkLength>            float        </minLinkLength>
</links>

<gameObjects> [0|1]
  <drawEntities>            [0|1]      </drawEntities>
  <drawUserDataObjects>     [0|1]      </drawUserDataObjects>
</gameObjects>

</render>

<terrain>
  <texture>
    <maxsizelimit>          float        </maxsizelimit>
    <maxsize>               float        </maxsize>
    <minsizelimit>          float        </minsizelimit>
    <minsize>               float        </minsize>
    <size>                  float        </size>
    <strength>              float        </strength>
    <lodregentime>          float        </lodregentime>
  </texture>
  <height>
    <maxsizelimit>          float        </maxsizelimit>
    <maxsize>               float        </maxsize>
    <minsizelimit>          float        </minsizelimit>
    <minsize>               float        </minsize>
    <size>                  float        </size>
    <maxstrengthlimit>      float        </maxstrengthlimit>
    <maxstrength>           float        </maxstrength>
    <minstrengthlimit>      float        </minstrengthlimit>
    <minstrength>           float        </minstrength>
    <strength>              float        </strength>
    <maxheightlimit>        float        </maxheightlimit>
    <maxheight>             float        </maxheight>
    <minheightlimit>        float        </minheightlimit>
    <minheight>             float        </minheight>
    <height>                float        </height>
    <brushFalloff>          float        </brushFalloff>
    <relative>              [0|1]      </relative>
  </height>
  <filter>
    <maxsizelimit>          float        </maxsizelimit>
    <maxsize>               float        </maxsize>
    <minsizelimit>          float        </minsizelimit>
    <minsize>               float        </minsize>
    <size>                  float        </size>
    <index>                 integer      </index>
  </filter>
  <cutRepair>
    <maxsizelimit>          float        </maxsizelimit>
    <maxsize>               float        </maxsize>
    <minsizelimit>          float        </minsizelimit>
    <minsize>               float        </minsize>
    <size>                  float        </size>
    <brushMode>             [0|1]      </brushMode>
  </cutRepair>
</terrain>

```



```

<terrain2>
  <defaults>
    <heightMapSize>      integer      </heightMapSize>
    <normalMapSize>      integer      </normalMapSize>
    <holeMapSize>        integer      </holeMapSize>
    <shadowMapSize>      integer      </shadowMapSize>
    <blendMapSize>       integer      </blendMapSize>
    <heightMapUnits>     float        </heightMapUnits>
  </defaults>
  <blendsBuildInterval> integer      </blendsBuildInterval>
</terrain2>

<messages>
  <showDate>            [0|1]        </showDate>
  <showTime>            [0|1]        </showTime>
  <showPriority>        [0|1]        </showPriority>
  <errorMsgs>          [0|1]        </errorMsgs>
  <warningMsgs>        [0|1]        </warningMsgs>
  <noticeMsgs>         [0|1]        </noticeMsgs>
  <infoMsgs>           [0|1]        </infoMsgs>
  <assetMsgs>          [0|1]        </assetMsgs>
</messages>

<sendCrashDump>        [0|1]        </sendCrashDump>

<project>
  <normalFont>          folder/file   </normalFont>
  <boldFont>            folder/file   </boldFont>
</project>

<userTag>              string        </userTag>

<placementPreset>      string        </placementPreset>

<currentLanguage>      string        </currentLanguage>

<chunkTexture>
  <numLayerWarning>     integer      </numLayerWarning>
  <showtrack>          [0|1]        </showtrack>
  <autotrack>          [0|1]        </autotrack>
  <latch>              integer      </latch>
  <numLayerWarningShow> integer      </numLayerWarningShow>
</chunkTexture>

<ualConfigPath>        folder/file   </ualConfigPath>

<warningMemoryLoadLevel> integer      </warningMemoryLoadLevel>

<bspBoundingBox>      [0|1]        </bspBoundingBox>

<fullSave>
  <chunkNumberBetweenSave> integer      </chunkNumberBetweenSave>
  <safeMemoryCountInMB>   integer      </safeMemoryCountInMB>
  <stripeSize>            integer      </stripeSize>
</fullSave>

</root>

```

Grammar of bigworld/tools/worldeditor/options.xml

The list below describes the tags in bigworld/tools/worldeditor/options.xml:

- **activetexture (section terrain/texture)**

The last chosen texture in the **Asset Browser** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **alphaTool (section tools)**

Texture file used for terrain alpha tool.

The path is relative to executable's folder (bigworld/tools/worldeditor).

- **alphaToolGUISize (section tools)**

Size of **Terrain Texturing** panel's **Textures In Chunk** field in relation to WorldEditor's window (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **alphaToolSize (section tools)**

Default size of the terrain alpha tool.

- **alphaToolStrength (section tools)**

Default strength of the terrain alpha tool.

- **angle**

- **Section shellSnaps**

Number of degrees by which to rotate the selected items.

Items can be rotated via the mouse wheel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Mouse controls") or the rotation gizmo (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Gizmos").

- **Section snaps**

Number of degrees by which to rotate the selected items (with the exception of shells) using the mouse wheel.

This value is set by the **Object** panel's **Object Grip Snaps** group box's **Angle** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Object panel").

Shell rotation is set by tag shellSnaps/angle.

- **assetMsgs (section messages)**

Determines whether asset messages are displayed in the Messages panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "BigWorld Messages panel").

- **autoTrack (section chunkTexture)**

Toggles whether the **Chunk Textures** panel automatically tracks the chunk under the cursor, or requires that the user explicitly set the current chunk (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Chunk Textures panel").

- **batchLimit (section versionControl)**

The maximum number of files that can be committed as a single batch.

- **blendMapSize (section terrain2/defaults)**

Default size (in pixels per chunk) for the terrain texture layer resolution (which is applied to the terrain via the Terrain Texturing panel — for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

The values has to be a power of 2.

- **blendsBuildInterval (section terrain2)**

WorldEditor uses this value, expressed in milliseconds, to determine how frequently it should rebuild terrain layer information for advanced terrain blocks. At most, only one terrain block will be processed. The default value is 100 milliseconds, meaning that it will process 10 blocks per second. A value of 0 results in processing a block every second. The default value of 100 milliseconds is the recommended value.

- **boldFont (section project)**

Bold font used in the **Project** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Chunk Textures panel”).

- **browsePath (section itemEditor)**

Last item selected in the Asset Browser panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Chunk Textures panel”).

- **brushFalloff (section terrain/height)**

Current mode of the terrain height brush in the **Terrain Height** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”).

The possible values are:

- **0** - Flat falloff.
- **1** - Linear falloff.
- **2** - Curved falloff.

- **brushMode (section terrain/cutRepair)**

Current mode of the terrain mesh cut/repair panel in the **Terrain Mesh Cut/Repair** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”).

The possible values are:

- **0** - make holes.
- **1** - repair holes.

- **bspBoundingBox**


Determines whether a SpeedTree's BSP bounding box is used when selecting the tree, or whether the tree's bounding box is used.

- **cameraHeight (section graphics)**

Height of the camera during player walkthrough mode.

This value is set by WorldEditor's **General Options** panel's **Camera Height** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “General Options panel”).



The walkthrough mode is toggled by the **Player Preview Mode** toolbar button —  — (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Toolbar”).

- **chunkNumberBetweenSave (section fullSave)**

Number of chunks to save as a group when doing a full-save operation.

- **chunkVisualisation (section tools)**

Texture file used to draw around the outside of chunks.

The path is relative to the executable's folder (bigworld/tools/worldeditor).

- **contentCreationManual (section help)**

Path and filename of the Content Creation Manual — the path is relative to the executable's folder (bigworld/tools/worldeditor).

The document accessed by pressing F1 or by selecting the **Help** → **Content Creation** menu item — for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Menu items”).

- **coordFilter (section tools)**

Value of the **Coordinate System** toolbar drop-down list box. The possible values are:

- World
- Local
- View

This value is set by the **Object** pane's **Coordinate System** group box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Object panel”).

- **currentLanguage**

Current language used by WorldEditor.

This value is set by selecting the Languages menu item (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Menu items”).

- **darkenBackground (section consoles)**

Determines whether the console's background should be darkened, for better readability (for details on the consoles available in WorldEditor, see the document Content Tools Reference Guide's sections *Tools Consoles* and *WorldEditor* → “Keyboard shortcuts”).

- **directory (section object/Entity)**

Default folder for entities in the **Asset Browser** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Asset Browser panel”).

- **directory (section object/Lights)**

Default folder for lights in the **Asset Browser** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Asset Browser panel”).

- **directory (section object/Particles)**

Default folder for particles in the **Asset Browser** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Asset Browser panel”).

- **directory (section object/Prefabs)**

Default location for prefabs in the **Asset Browser** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Asset Browser panel”).

- **directory (section object/Shell)**

Default location for shells in the **Asset Browser** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Asset Browser panel”).

- **directory (section object/SpeedTree)**

Default location for SpeedTrees in the **Asset Browser** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Asset Browser panel”).

For details on SpeedTree, see the document Third-Party Integrations's section *SpeedTree*).

- **directory (section terrain/textures)**

Default folder on the **Terrain Texturing** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

- **dragOnSelect**

Determines if the selected object(s) can be moved via the mouse or only via the movement gizmo.

This value is set by the **Object** panel's **Drag On Select** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Object panel”).

- **drawBloom (section render/environment)**

Toggles the bloom full screen effect — this value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment→Bloom** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

- **drawBSP**

The kind of BSPs should be displayed.

Possible values are:

- **0** — Normal
- **1** — Custom
- **2** — All

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Scenery→BSP** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

- **drawChunkFlares (section render)**

Toggles the rendering of chunk flares.

- **drawChunkLights (section render)**

Toggles the rendering of lights.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Lights** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawChunkPortals (section render)**

Toggles the rendering of portals of chunks/shells.

- **drawClouds (section render/environment)**

Toggles the rendering of clouds.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment→Clouds** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawDetailObjects (section render/environment)**

Toggles the rendering of details object (*e.g.*, flora).

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment→Detail Objects** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawDynamic (section render/lights)**

Toggles the visibility of dynamic lights.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Lights→Dynamic Light Models** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawEditorProxies (section render/misc)**

Toggles the display of proxy models defined for models in ModelEditor.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Editor Proxies** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawEntities (section render/gameObjects)**

Toggles the rendering of entities.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Game Objects→Entities** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawFog (section render/environment)**

Toggles the rendering of fog.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment→Draw Fog** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawHeavenAndEarth (section render/misc)**

Toggles the rendering of the heaven and earth portals used by chunks.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Heaven And Earth Portals** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawShimmer (section render/environment)**

Toggles the shimmer full screen effect.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment→Shimmer** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawSky (section render/environment)**

Toggles the rendering of sky.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment→Sky** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawSpecular (section render/lights)**

Toggles the visibility of specular lights.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Lights→Specular Light Models** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawStars (section render/environment)**

Toggles the visibility of stars.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment→Stars** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawStatic (section render/lights)**

Toggles the visibility of static lights.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Lights→Static Light Models** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawStaticSky (section render/environment)**

Toggles the visibility of static sky.

- **drawSunAndMoon (section render/environment)**

Toggles the rendering of sun and moon objects.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment→Sun And Moon** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawUserDataObjects (section render/gameObjects)**

Toggles the rendering of user data objects and their links.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Game Objects→User Data Objects** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **drawVLO (section render)**

Toggles the rendering of Very Large Objects.

- **drawWater (section render/scenery)**

Toggles the rendering of water.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment→Water** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **enable (section versionControl)**

Determines whether the version control functionality is enabled.

- **enableDynamicUpdating**

This value is set by WorldEditor's **General Options** panel's **Lighting** group box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **environment (section render)**

Toggles the visibility of all selected environment items.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment** group box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **errorMsgs (section messages)**

Determines whether error messages are displayed in the **Messages** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **exclusive (section input)**

Determines whether WorldEditor should work in exclusive mode.

- **farclip (section graphics)**

Far plane size in metres.

This value is set in the **General Options** panel's **Far Plane** slider (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "General Options panel").

- **gameVisibility (section render/scenery/shells)**

Toggles the use of the portal visibility algorithm by the shell.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Shell→Game Visibility** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "General Options panel").

- **gameObjects (section render)**

Toggles the visibility of game objects.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Game Objects** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

- **graphics**

Section specifying graphic card's options such as camera height while player is in walkthrough mode, far plane, etc...

- **graphicsPreferencesXML (section graphics)**

The location of the graphics options XML files.

- **height (section height)**

The last used absolute or relative height in metres.


- **heightMapSize (section terrain2/defaults)**

The default resolution of the height map in pixels per chunk.

This needs to be a power of two.

- **hideOutsideObjects (section render)**

Determines whether outside objects should be rendered.

This value is set by the **Hide/show outside objects** toolbar button —  — (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Toolbar”).

- **holeMapSize (section terrain2/defaults)**

The default size of the hole map in pixels per chunk.

This value does not have to be a power of two.

- **host (section bwlockd)**

IP address and port of the lock server (for details, see the document Content Tools Reference Guide's chapter *Lock Server (BWLockD)*).

- **importmask (section terrain/texture)**

Toggles the terrain painting import mask used in **Terrain Texturing** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

- **importstrength (section terrain/texture)**

Strength of the brush when blending texture onto terrain area underneath brush icon.

This value is set by WorldEditor's **Terrain Texturing** panel's **Mask** group box's **Strength** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

- **index (section terrain/filter)**

Index of the current terrain filter.

The indices are part of the filter definition file, which is located in `resources/data/filters.xml` (under WorldEditor's executable folder).

- **infoMsgs (section messages)**

Toggles whether information messages are displayed in the **Messages** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "BigWorld Messages panel").

- **itemSnapMode (section snaps)**

Determines the type of item snap (if any).

This value is set by WorldEditor's Item **Snap Mode** toolbar buttons (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Toolbar").

Possible values are:

- **0** — Free positioning (also set via toolbar button).
- **1** — Terrain lock (also set by toolbar button).
- **2** — Obstacle lock (also set by toolbar button).


- **lastY (section itemEditor)**

Internal flag, used to record the last Y value when moving objects.

- **latch (section chunkTexture)**

Toggles whether the **Chunk Textures** panel is automatically displayed when activating the **Terrain Texturing** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

This value is set by WorldEditor's **Chunk Textures** panel's **Show When The Terrain Texturing Tool Is Activated** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Chunk Textures panel").

The **Terrain Texturing** panel is activated by the **Terrain Texture Tool** toolbar button —  — (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary"). .

- **legacyMouseWheel (section input)**

Determines whether the mouse wheel should change the camera speed (when this value is set to `true`) or zoom the viewport in and out (when this value is `false`).

- **lighting (section render)**

Specifies how lighting should be rendered.

Possible values are:

- **0** — Standard
- **1** — Dynamic
- **2** — Specular

This value is also set in the **General Options** panel's **Lighting** group box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “General Options panel”).

- **lights (section render)**

Toggles the visibility of all selected light types — this value is set by WorldEditor's **General Options** panel's **Show** list box's **Lights** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

- **limit (section undoredo)**

Number of undo/redo action saved in memory.

If this value is not set, then the number of actions will be limited by the available memory.

- **LOD (section render/terrain)**

Toggles terrain LODing — this value is set by WorldEditor's **General Options** panel's **Show** list box's **Terrain→Terrain Level Of Detail** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

- **lodregentime (section render/terrain)**

The time in seconds between regenerating terrain texture LODs.

- **maskBottomRight (section render/height)**

Bottom-right coordinates for the terrain painting import mask used in the **Terrain Texturing** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

The coordinates are in chunks with the bottom left-chunk being (0 , 0).

- **materialOverrideMode (section objects)**

Toggles the handling of all materials' properties as editable, regardless of the value of its `worldBuilderEditable` flag.

For details on this flag, see the document Client Programming Guide's section *3D Engine (Moo)* → “EffectMaterial” → “Artist-editable/tweakable variables”.

- **maxFarPlane (section render)**

The maximum distance in meters that the far-plane can be set to — this value is set by the **General Options** panel's **Far Plane** field — for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “General Options panel”).

- **maxCBLsPerFrame (section render/links)**

There is a cost associated with loading, unloading, creating, deleting, and editing UDOs. The user can control the size of the jobs performed per frame by editing this variable.

Controls the number of chunk back links (an incoming link from an unloaded UDO) processed per frame.

- **maxCLsPerFrame (section render/links)**

There is a cost associated with loading, unloading, creating, deleting, and editing UDOs. The user can control the size of the jobs performed per frame by editing this variable.

Controls the number of chunk links (an outgoing link to an unloaded UDO) processed per frame.

- **maxheight** (section **terrain/height**)

Maximum value that can currently be assigned to the **Terrain Height** panel's **Explicit Height** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”).

Note

Note that **maxheightlimit** specifies the value that can ever be set as a maximum to **Terrain Height** panel's **Explicit Height** field via the **Set Slider Limits** dialog box.

- **maxheightlimit** (section **terrain/height**)

Maximum value that can be assigned to the **Terrain Height** panel's **Explicit Height** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”) via the **Set Slider Limits** dialog box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Dialog boxes” → “Set Slider Limits dialog box”).

Note

Note that **maxheight** specifies the maximum value that currently can be assigned to **Terrain Height** panel's **Explicit Height** field.

- **maxProcessingMillis** (section **render/links**)

There is a cost associated with loading, unloading, creating, deleting and editing User Data Objects that are linked. The user can control the time spent per frame updating links by editing this variable. The recommended value is 2, which allocates 2 milliseconds per frame for this task.

- **maxRecalcsPerFrame** (section **render/links**)

There is a cost associated with recalculating the geometry of a link so that it conforms to the terrain as chunks are loaded in. The user can control the how often these recalculations are performed by editing this variable.

Controls the number of links recalculated per frame.

- **minLinkLength** (section **render/links**)

This is the minimum length for links. If a link is shorter than this value, the user will get error messages indicating that this link can cause problems if used in navigation.

- **maxsize**

- **Section terrain/cutRepair**

Maximum value that can currently be assigned to the **Terrain Mesh Cut/Repair** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Mesh Cut/Repair panel”).

- **Section terrain/filter**

Maximum value that can currently be assigned to the **Terrain Filtering** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Filtering panel”).

- **Section terrain/height**

Maximum value that can currently be assigned to the **Terrain Height** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Height panel").

- **Section terrain/texture**

Maximum value that can currently be assigned to the **Terrain Texturing** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

Note

Note that this tag is different from `maxsizelimit`, which specifies the biggest value that can ever be set as a maximum to the respective panel's **Size** field via the **Set Slider Limits** dialog box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Dialog boxes" → "Set Slider Limits dialog box").

- **maxsizelimit**

- **Section terrain/cutRepair**

Maximum value that can be assigned to the **Terrain Mesh Cut/Repair** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Mesh Cut/Repair panel") via the **Set Slider Limits** dialog box.

- **Section terrain/filter**

Maximum value that can be assigned to the **Terrain Filtering** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Filtering panel") via the **Set Slider Limits** dialog box.

- **Section terrain/height**

Maximum value that can be assigned to the **Terrain Height** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Height panel") via the **Set Slider Limits** dialog box.

- **Section terrain/texture**

Maximum value that can be assigned to the **Terrain Texturing** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel") via the **Set Slider Limits** dialog box.

For details on the **Set Slider Limits** dialog box, see the document Content Tools Reference Guide's section *WorldEditor* → "Dialog boxes" → "Set Slider Limits dialog box").

Note

Note that this tag is different from `maxsize`, which specifies the maximum value that currently can be assigned to the respective panel's **Size** field.

- **maxstrength**

- **Section terrain/height**

Maximum value that can currently be assigned to the **Terrain Height** panel's **Strength** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”).

- **Section terrain/texture**

Maximum value that can currently be assigned to the **Terrain Texturing** panel's **Strength** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”).

Note

Note that this tag is different from `maxstrengthlimit`, which specifies the biggest value that can ever be set as a maximum to the respective panel's **Strength** field via the **Set Slider Limits** dialog box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Dialog boxes” → “Set Slider Limits dialog box”).

- **maxstrengthlimit (section terrain/height)**

Maximum value that can be assigned to the **Terrain Height** panel's **Strength** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”) via the **Set Slider Limits** dialog box.

For details on the **Set Slider Limits** dialog box, see the document Content Tools Reference Guide's section *WorldEditor* → “Dialog boxes” → “Set Slider Limits dialog box”).

Note

Note that this tag is different from `maxstrength`, which specifies the maximum value that currently can be assigned to the respective panel's **Strength** field.

- **maxTimeBetweenRecalcs (section render/links)**

There is a cost associated with recalculating the geometry of a link so that it conforms to the terrain as chunks are loaded in. The user can control the how often these recalculations are performed by editing this variable.

Controls the maximum time in seconds between updates.

- **maxTimeDelta (section app)**

Maximum number of seconds elapsed between the last frame and the current one when calculating frame time.

If the delta is bigger than `maxTimeDelta`, then it is set to `maxTimeDelta` for frame time calculation purposes.

- **maxTimeWaitForMoreChunks (section render/links)**

There is a cost associated with recalculating the geometry of a link so that it conforms to the terrain as chunks are loaded in. The user can control the how often these recalculations are performed by editing this variable.

Controls the maximum time in seconds WorldEditor will wait for additional chunks to be loaded before recalculating the links.

- **minheight** (section **terrain/height**)

Minimum value that can currently be assigned to the **Terrain Height** panel's **Explicit Height** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”).

Note

Note that this tag is different from **minheightlimit**, which specifies the lowest value that can ever be set as a minimum to **Terrain Height** panel's **Explicit Height** field via the **Set Slider Limits** dialog box.

- **minheightlimit** (section **terrain/height**)

Minimum value that can be assigned to the **Terrain Height** panel's **Explicit Height** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”) via the **Set Slider Limits** dialog box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Dialog boxes” → “Set Slider Limits dialog box”).

Note

Note that this tag is different from **minheight**, which specifies the minimum value that currently can be assigned to **Terrain Height** panel's **Explicit Height** field.

- **minsize**

- **Section terrain/cutRepair**

Minimum value that can currently be assigned to the **Terrain Mesh Cut/Repair** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Mesh Cut/Repair panel”).

- **Section terrain/filter**

Minimum value that can currently be assigned to the **Terrain Filtering** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Filtering panel”).

- **Section terrain/height**

Minimum value that can currently be assigned to the **Terrain Height** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”).

- **Section terrain/texture**

Minimum value that can currently be assigned to the **Terrain Texturing** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

Note

Note that this tag is different from `minsize`, which specifies the minimum value that can ever be set as a minimum to the respective panel's **Size** field via the **Set Slider Limits** dialog box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Dialog boxes" → "Set Slider Limits dialog box").

- **minsize**

- **Section terrain/cutRepair**

Minimum value that can be assigned to the **Terrain Mesh Cut/Repair** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Mesh Cut/Repair panel") via the **Set Slider Limits** dialog box.

- **Section terrain/filter**

Minimum value that can be assigned to the **Terrain Filtering** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Filtering panel") via the **Set Slider Limits** dialog box.

- **Section terrain/height**

Minimum value that can be assigned to the **Terrain Height** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Height panel") via the **Set Slider Limits** dialog box.

- **Section terrain/texture**

Minimum value that can be assigned to the **Terrain Texturing** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel") via the **Set Slider Limits** dialog box.

For details on the **Set Slider Limits** dialog box, see the document Content Tools Reference Guide's section *WorldEditor* → "Dialog boxes" → "Set Slider Limits dialog box").

Note

Note that this tag is different from `minsize`, which specifies the minimum value that currently can be assigned to **Terrain Mesh Cut/Repair** panel's **Size** field.

- **minstrength (section terrain/height)**

Minimum value that can currently be assigned to the **Terrain Height** panel's **Strength** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Height panel").

Note

Note that this tag is different from `minstrengthlimit`, which specifies the lowest value that can ever be set as a minimum to the respective panel's **Strength** field via the **Set Slider Limits** dialog box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Dialog boxes” → “Set Slider Limits dialog box”).

- **minstrengthlimit**

- **Section terrain/height**

Minimum value that can be assigned to the **Terrain Height** panel's **Strength** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Height panel”) via the **Set Slider Limits** dialog box.

- **Section terrain/texture**

Minimum value that can be assigned to the **Terrain Texturing** panel's **Strength** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”) via the **Set Slider Limits** dialog box.

For details on the **Set Slider Limits** dialog box, see the document Content Tools Reference Guide's section *WorldEditor* → “Dialog boxes” → “Set Slider Limits dialog box”).

Note

Note that this tag is different from `minstrength`, which specifies the minimum value that currently can be assigned to the respective panel's **Strength** field.

- **misc (section render)**

Toggles the visibility of all selected miscellaneous items.

- **movement**

- **Section shellSnaps**

Value (in metres) for snapping shells along the respective axis.

- **Section snaps**

Value (in metres) for snapping items (with the exception of shells) along the respective axis. Shell snap is set by tag `shellSnaps/movement`.

- **normalFont (section project)**

Normal font used in the **Project** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Chunk Textures panel”).

- **normalMapSize (section terrain2/defaults)**

Default size of the normal map in pixels per chunk.

This value has to be a power of two.

- **noticeMsgs (section messages)**

Determines whether notice messages are displayed in the Messages panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "BigWorld Messages panel").

- **numLayerWarning (section chunkTexture)**

Maximum number of textures that chunks might have before a warning message is displayed in the offending chunks.

This value is set by the **Chunk Textures** panel's **Maximum Textures Per Chunk Warning** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Chunk Textures panel").

- **numLayerWarningShow (section chunkTexture)**

Determines whether WorldEditor should display the "maximum texture per chunk" warning.

This value is set by the **Chunk Textures** panel's button   buttons (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Chunk Textures panel").

- **numMessageLines (section consoles)**

Maximum number of short-lived error or warning messages displayed at the bottom of the viewport.

- **ortho (section camera)**

Toggles camera's orthographic view.

This value is set by the **Orthographic View** toolbar button —  — (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Toolbar").

- **optionsPage (section tools)**

Location of the internal data file used when generating the **General Options** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

The path is relative to the executable's folder (bigworld/tools/worldeditor).

- **particle (section scenery)**

Toggles the visibility of particle systems — this value is set by WorldEditor's **General Options** panel's **Show** list box's **Scenery→Particles** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **path (section versionControl)**

Path and file name of the file to be used by WorldEditor's version control stub. It can be an executable or any file with an associated default program, like .py file with Python installed.

The default value is resources/scripts/svn_stub.exe.

For details, see the document Content Tools Reference Guide's section *Lock Server (BWLockD)* → "Using a version control system stub".

- **placementPreset**

Method of object placement (regarding randomisation of rotation and scale).

This value is set by the **Object** panel's **Placement Control** group box's **Setting** drop-down list box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Object panel").

- **precompileEffects**

Determines whether .fx files should be compiled before WorldEditor first runs or should be built when first used.

- **readOnlyMode (section objects)**

Determines if changes to the scene can be saved.

- **reflection (section render/scenery/drawWater)**

Toggles water reflection — this value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment**→**Water**→**Reflection** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **relative (section terrain/height)**

Determines the mode of the terrain height brush — 0 activates the *relative* mode, and 1 activates the *absolute* mode.

This value is set by the **Terrain Height** panel's **Mode** group of radio buttons (**Absolute** and **Relative**) — for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Height panel".

- **safeMemoryCountInMB (section fullSave)**

Amount of memory (in MB) that should be allocated during testing low-memory conditions when doing a full save.

- **saveLayoutOnExit (section panels)**

Determines whether the layout of the panels should be saved on exit.

For details on the panel system, see the Content Tools Reference Guide's section *Panel System*.

- **scenery (section render)**

Toggles the visibility of scenery item — this value is set by WorldEditor's **General Options** panel's **Show** list box's **Scenery** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **selectFilter (section tools)**

Type of elements of the world that will be affected by a select action.

This value is set by the **Object** panel's **Selection Filter** drop-down list box — for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Object panel".

Possible values are:

- **All Entities**
- **All Except Terrain and Shells**

- **Shells + Contents**
- **All Lights**
- **Omni Lights**
- **Ambient Lights**
- **Directional Lights**
- **Spot Lights**
- **Models**
- **Entities**
- **Clusters and Markers**
- **Particles**
- **Waypoint Stations**
- **Terrains**
- **Sound**
- **Water**
- **Portals**

▪ **selBottomRight (section render/height)**

Bottom-right coordinate for importing, used in the **Terrain Import/Export** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Import/Export panel").

The coordinates are in chunks, with the bottom left-chunk being (0 , 0).

▪ **selTopLeft (section render/height)**

Top-left coordinate for importing, used in the **Terrain Import/Export** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Import/Export panel").

The coordinates are in chunks, with the bottom left-chunk being (0 , 0).

▪ **sendCrashDump**

Determines whether dumps should be sent to BigWorld in the event of a WorldEditor crash.

▪ **shadeReadOnlyAreas (section render/misc)**

Toggles the shading in red of read-only areas.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Shade Read-Only Areas** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

For more details on the lock server, see the document Content Tools Reference Guide's section *Lock Server (BWLockD)*.

- **shadows (section graphics)**

Toggles the creation of a stencil buffer.

- **shadowMapSize (terrain2/defaults)**

Default size of the shadow map.

This value has to be a power of 2.

- **shells (section scenery)**

Toggles the visibility of shells in the world.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Shell** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Terrain Texturing panel”).

- **shellSnaps**

Tag for section specifying snap and rotation settings for shells.

The values for the tags in this sections are set by WorldEditor's **Object** panel's **Object Grid Snap** group box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Object panel”).

- **shortcutsHtml (section help)**

The location of the shortcuts HTML — the path is relative to the executable's folder (bigworld/tools/worldeditor).

Path and filename of the Content Creation Manual — the path is relative to the executable's folder bigworld/tools/worldeditor (the document accessed by selecting the **Help** → **Shortcuts** menu item — for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Menu items”).

- **showChunkVisualisation (section tools)**

Determines the type of chunk visualisation mode.

Possible values are:

- **0** - No visualisation
- **1** - Chunk visualisation
- **2** - Vertex visualisation
- **3** - Mesh visualisation

This value is set by clicking the **Chunk Visualisation Mode** toolbar buttons (for details, see the document Content Tools Reference Guide's sections *WorldEditor* → “Toolbar”) or by pressing F6 (see the document Content Tools Reference Guide's sections *WorldEditor* → “Keyboard shortcuts”).

For details on the chunk visualisation modes, see *WorldEditor* → “Chunk visualisation modes”).

- **showDate (section messages)**

Determines whether the date of the issued messages should be displayed in the **Messages** panel.

This value is set by the **Messages** panel's **Show Date** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “BigWorld Messages panel”).

- **showPriority (section messages)**

Determines whether the priority of the issued messages should be displayed in the **Messages** panel.

This value is set by the **Messages** panel's **Show Priority** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "BigWorld Messages panel").

- **showTime (section messages)**

Determines whether the time of the issued messages should be displayed in the **Messages** panel.

This value is set by the **BigWorld Messages** panel's **Show Time** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "BigWorld Messages panel").

- **showTrack (section messages)**

Determines if WorldEditor should outline the chunk under the cursor.

This value is set by the **Chunk Textures** panel's  buttons (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Chunk Textures panel").

- **simulation (section render/scenery/drawWater)**

Toggles water simulation — this value is set by WorldEditor's **General Options** panel's **Show** list box's **Environment→Water→Simulation** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **size**

- **Section terrain/cutRepair**

Size of the terrain mesh cut/repair brush in metres.

This value is set by the **Terrain Mesh Cut/Repair** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Mesh Cut/Repair panel").

- **Section terrain/filter**

Size of the terrain filter brush in metres.

This value is set by the **Terrain Filtering** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Filtering panel").

- **Section terrain/height**

Size of the terrain height brush in metres.

This value is set by the **Terrain Height** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Height panel").

- **Section terrain/texture**

Size of the terrain texture brush in metres.

This value is set by the **Terrain Texturing** panel's **Size** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **{Slow|Medium|Fast|SuperFast} (section camera/speed)**

Definition of normal speed (in m/s) for the 4 pre-defined camera speeds.

This value is set by clicking the **Camera Speed** toolbar buttons (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Toolbar"), or by the Ctrl+1, Ctrl+2, Ctrl+3, and Ctrl+4 keyboard shortcuts (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Keyboard shortcuts").

- **snaps**

Tag for section specifying snap and rotation settings for items (with the exception of shells, which are set by section `shellSnaps`).

- **spaceMapAlpha (section render/project)**

Level of transparency applied to the world's hand drawn map layer that is rendered over the world's game map.

This value is set by **Project** panel's **Hand Drawn Map vs. Calculated Map** slider (for details, see the Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Project panel").

- **standalone**

Determines if the Lock Server (`bwlockd`) should be used (for details, see the document Content Tools Reference Guide's chapter *Lock Server (BWLockD)*).

- **strength**

- **Section terrain/height**

Strength of the terrain height brush.

This value is set by the **Terrain Height** panel's **Strength** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Height panel").

- **section terrain/texture**

Strength of the terrain painting brush.

This value is set by the **Terrain Texturing** panel's **Strength** field (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **stripeSize (section fullSave)**

Number of chunks in a single strip to use when doing a full save.

- **terrain (section render)**

Toggles the visibility of the terrain — this value is set by WorldEditor's **General Options** panel's **Show** list box's **Terrain** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **texture (section terrain/channeln)**

Texture file used in that channel.

The section contains the history of texture files used in the channel.

- **timeofday (section graphics)**

Initial time of the day in the world.

This value is set in **Environment Options** panel's **Time Of Day** slider (for details, see the Content Tools Reference Guide's section *WorldEditor* → “Panel summary” → “Environment Options panel”).

- **timeScale (section app)**

Number of seconds in real world time corresponding to each second in game time.

This value is used to calculate the time elapsed between last frame and current one.

- **toolsReferenceGuide (section help)**

Path and filename of the Content Tools Reference Guide — the path is relative to the executable's folder (bigworld/tools/worldeditor).

The document accessed by pressing F1 or by selecting the **Help** → **Tools Reference Guide** menu item — for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Menu items”).

- **turbo (section camera/speed/speed_label)**

Definition of turbo speed (in m/s) for the 4 pre-defined camera speeds.

The camera speed is set by clicking the **Camera Speed** toolbar buttons (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Toolbar”), or by the Ctrl+1, Ctrl+2, Ctrl+3, and Ctrl+4 keyboard shortcuts (for details, see the document Content Tools Reference Guide's section *WorldEditor* → “Keyboard shortcuts”).

- **ualConfigPath**

Path of the Asset Browser's configuration file — the path is relative to the executable's folder (bigworld/tools/worldeditor).

For details on this file, see the document Content Tools Reference Guide's section *Asset Browser* → “Customisation”).

- **use (section bwlockd)**

Determines whether the Lock Server (bwlockd) should be used within WorldEditor (for details, see the document Content Tools Reference Guide's chapter *Lock Server (BWLockD)*).

- **userTag**

Name to use by the Lock Server (bwlockd) for shells when multiple users are working on a same space, in order to avoid name collision (for details, see the document Content Tools Reference Guide's chapter *Lock Server (BWLockD)*).

- **vizMode (section render/terrain)**

Determines the terrain visualisation mode.

The possible values are:

- **0** — Displays the terrain as normal.
- **1** — Displays the boundary of chunks.
- **2** — Displays the vertices of the terrain heights or blends (depending on which mode is active).
- **3** — Displays the mesh of the terrain heights or blends (depending on which mode is active).

- **warningMemoryLoadLevel**

Memory load level (as a percentage) of used *vs.* total memory at which point a dialog will be displayed.

The dialog box will contain a warning about the low-memory condition, along with options such as clearing the undo/redo buffer to try to recover more memory.

- **warningMsgs (section messages)**

Determines whether warning messages should be displayed in the **Messages** panel (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **watcher_name (section romp/watcherValues)**

Specifies the value for the watcher.

- **wireFrame**

- **Section render/scenery**

Toggles the display of scenery items as wireframe.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Scenery→Wireframe** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **Section render/terrain**

Toggles the display of the terrain as wireframe.

This value is set by WorldEditor's **General Options** panel's **Show** list box's **Terrain→Wireframe** check box (for details, see the document Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Terrain Texturing panel").

- **xyzEnabled (section snaps)**

Toggles item snap mode.

This value is set in the **Object** panel's **Object Grid Snaps** group box's **Enabled** check box (for details, see the Content Tools Reference Guide's section *WorldEditor* → "Panel summary" → "Object panel").

Possible values are:

- **0** - Item snap disabled
- **1** - Item snap enabled

Chapter 26. PackedSection files

PackedSection files are **read-only** data sections described in BNF format. The PackedSection file has the following format (the asterisk character — * — indicates that the previous section might appear zero or more times):

```
<packed_section> ::= <magic_number> <version> <string_table> <data_section>
<string_table> ::= <null_terminated_string>* '\0'
<data_section> ::= <num_children> <data_pos> <child_record>* <bin_data>
<child_record> ::= <key_pos> <data_pos>
<bin_data> ::= <bin_data_for_this_section> <bin_data_for_children>
<bin_data> ::= <bin_data_for_this_section>
```

PackedSection file format in BNF grammar

The list below describes the sections in the file:

- **<magic_number>**

4-byte number 0x42A14E45.

- **<version>**

int8 number.

- **<string_table>**

Sequence of null-terminated strings, followed by an empty string.

In the section <key_pos>, these strings are referred to by their index in the table.

- **<data_section>**

Size of this section's data is indicated by the first <data_pos>.

In other words, all data from the start of <bin_data> through to the offset given by the first <data_pos> is the data associated with this section.

<data_section> without children are just raw binary data for that type. For example, a float is four bytes of the float, a Vector3 is 3 consecutive floats, a Matrix 12 float.

- **<num_children>**

int number.

- **<data_pos>**

int32 number, representing the offset relative to the start of section <bin_data> (and not as relative to the start of the file).

The type of each section's data is indicated in the high bits of the following <data_pos> section (and not in its own). This is the reason for the final <data_pos> after <child_record>. For the precise number of bits used for indicating types and the constant mappings for the various supported types, see file `bigworld/src/lib/resmgr/packed_section.hpp`.

- **<bin_data>**

Binary data block of the data for this section, and the data for each child section concatenated together.

- **<child_record>**

Data starts at the <data_pos> of the previous record (or the <data_pos> of the <data_section> if this is the first record), and ends at the <data_pos> of this record.

- **<key_pos>**

int16 number, representing the index (and not a byte offset) in section <string_table> (relative to the start of the file).

Index count starts at 0.

In **PackedSection** files, integers are slightly optimised (*e.g.*, if value is zero, then no data will be stored; if value fits in an int8, then 1 byte will be stored, and so on).

Chapter 27. paths.xml

The following folders contain versions of paths.xml:

- **bigworld/tools/exporter/3dsmax<version>**
- **bigworld/tools/exporter/maya<version>**
- **bigworld/tools/misc**
- **bigworld/tools/modeleditor**
- **bigworld/tools/particleeditor**
- **bigworld/tools/worldeditor**
- **fantasydemo/game**

The grammar of paths.xml is listed below:

```
<root>
  <Paths>
    *<Path>  folders  </Path>
  </Paths>
</root>
```

Grammar of paths.xml

The list below describes the tags in paths.xml:

- **Path**

Specifies (either with drive letter or not) the folder or Zip file to be added to path list.

Chapter 28. .ppchain

PPChain files are saved from the WorldEditor's Post Processing editor panel. They describe an entire chain, or a partial chain. By default they are located in the `<res>/system/post_processing/chains` folder. A .ppchain file has the grammar described below:

```
<file_name>

+<Effect>
  +<bypass>          Vector4          </bypass>

  *<PyPhase>
    <name>            string            </name>
    <clearRenderTarget> [true|false]    </clearRenderTarget>
    <material>        material section  </material>
    <renderTarget>    string            </renderTarget>
    <filterQuad>
      [ <PyFilterQuad>
        *<filterTap> Vector4          </filterTap>
        </PyFilterQuad>

        | <PyPointSpriteTransferMesh> </PyPointSpriteTransferMesh>

        | <PyTransferQuad> </PyTransferQuad>

        | <PyVisualTransferMesh>
          <resourceID> string          </resourceID>
        </PyVisualTransferMesh>
      ]
    </filterQuad>
  </PyPhase>

  *<PyCopyBackBuffer>
    <name>            string            </name>
    <renderTarget>    string            </renderTarget>
  </PyCopyBackBuffer>

  *<PlayerFader>
    <name>            string            </name>
    <renderTarget>    string            </renderTarget>
  </PlayerFader>

</Effect>

</file_name>
```

Grammar of ppchain file

The list below describes the tags in the ppchain file:

- **Effect**

A .ppchain file consists solely of a list of Effects.

- **bypass**

Set to (0,0,0,0) to bypass an effect, or any non-negative value to enable. This bypass value is often overridden for dynamic control by Python game scripts.

- **PyPhase**

Tag for PyPhase style phase section. A PyPhase is a generic post-processing phase that is used in the majority of circumstances.

- **name**

String identifier for this phase.

- **renderTarget**

The string identifier of the render target to which this phase will render.

- **clearRenderTarget**

Whether or not the render target should be cleared before rendering this phase.

- **material**

Effect Material section, see *EffectMaterial* section on page 35

- **filterQuad**

The filter quad geometry that will be used to render with. See available Filter Quad sections below

- **PyCopyBackBuffer**

Tag for PyCopyBackBuffer style phase section. This type of phase performs a StretchRect copy of the device's back buffer into the render target. When making a copy of the device's back buffer it is preferable to use this phase rather than a generic transfer phase, as the StretchRect operation resolves any hardware antialiasing at this time. Additionally, this phase knows when the back buffer surface has not yet been modified by another phase and will temporarily bypass the operation in this case.

- **name**

String identifier for this phase.

- **renderTarget**

The string identifier of the render target to which this phase will render.

- **PlayerFader**

Tag for PlayerFader style phase section. The PlayerFader works in conjunction with the CursorCamera to fade out the player model when the camera moves forward and intersects with the player's bounding box due to the camera colliding with scene geometry.

- **name**

String identifier for this phase.

- **renderTarget**

The string identifier of the render target to which this phase will render.

- **PyFilterQuad**

Tag for PyFilterQuad style filter quad. This filter quad has an arbitrary number of filter taps, and draws in multiple passes doing 4 taps per pass.

- **filterTap**

Vector4 (uOffset, vOffset, weight, unused).

- **PyPointSpriteTransferMesh**

Tag for PyPointSpriteTransferMesh style filter quad.

- **PyTransferQuad**

Tag for PyTransferQuad style filter quad.

- **PyVisualTransferMesh**

Tag for PyVisualTransferMesh style filter quad.

- **resourceID**

The string identifier of the .visual file with which this phase will render.

Chapter 29. .primitives

Defined in various sub-folders under the resource tree <res> (for example, <res>/environments, <res>/flora, <res>/sets/vehicles, etc...), the .primitives files use **BinSection** (for details on grammar of BinSection files, see *BinSection files* on page 9) to have discrete bits of binary data saved in one file using the BigWorld file system.

The primitive file can contain vertex data, index data, and BSP data.

29.1. Vertex data section

The section with vertex data contains a small header, followed by the raw vertex data.

Described in BNF format, the file has the following format (the asterisk character — * — indicates that the previous section might appear zero or more times.):

```
<vertex_format>
<number_of_vertices>
<raw_vertex_data>
```

.primitives file format in BNF grammar — Vertex data section

The list below describes the sections in the file:

- **<number_of_vertices>**

Number of vertices.

- **<raw_vertex_data>**

Actual vertex data, with size equal to `sizeof vertex * <number_of_vertices>`.

- **<vertex_format>**

64-byte field, containing the name of the vertex format (e.g., xyznuv, xyznuvtb, etc...).

29.2. Index data section

The section with index data contains a small header and the raw index data, followed by the primitive groups. The primitive groups define the batches with different materials in the triangle list (these are the same primitive groups referenced from the .visual file — for details on grammar of .visual files, see *.visual* on page 145).

Described in BNF format, the file has the following format (the asterisk character — * — indicates that the previous section might appear zero or more times):

```
<index_format>
<number_of_indices>
<number_of_primitive_groups>
<raw_index_data>
<raw_primitive_data> ::= <primitive_group_data>* 1
<primitive_group_data> ::=
  <start_idx><num_of_prmtvs><start_vrtx><num_of_vrtcs>
```

.primitives file format in BNF grammar — Index data section

1 `<raw_primitive_data>` contains `<number_of_primitive_groups>` sections
`<primitive_group_data>`.

The list below describes the sections in the file:

- **`<index_format>`**

64-byte field, containing the type of index list.

Possible values are `list` (which mean a 16-bit index list), and `list32` (which means a 32-bit index list).

- **`<num_of_primitives>`**

Number of triangles rendered in this group.

- **`<num_of_vrtcs>`**

Number of vertices used by the triangles in this group.

- **`<number_of_indices>`**

Integer number containing the number of indices.

- **`<number_of_primitive_groups>`**

Integer number containing the number of primitive groups.

- **`<raw_index_data>`**

Actual index data, with size equal to $2 \times \text{<number_of_vertices>}$ (if `<index_format>` is `list`), or $4 \times \text{<number_of_vertices>}$ (if `<index_format>` is `list32`).

- **`<raw_primitive_data>`**

Actual primitive groups data, with size equal to $16 \times \text{<number_of_primitive_groups>}$.

This section contains `<number_of_primitive_groups>` sections `<primitive_group_data>`.

- **`<start_idx>`**

First index used by this group of triangles.

- **`<start_vrtx>`**

First vertex used by the triangles in this group.

29.3. BSP data section

The section with BSP data contains a small header, followed by the raw BSP data.

Described in BNF format, the file has the following format (the asterisk character — * — indicates that the previous section might appear zero or more times):

```
<bsp_file> ::= <header> <triangle>* <node>* <user_data>*
<header> ::= <magic_number> <num_triangles> <max_triangles> <num_nodes>
<triangle> ::= <Vector3> <Vector3> <Vector3>
<node> ::= <node_flags> <plane_eq> <num_indices> <triangle_index>*
<node_flags> ::= <reserved> <is_partitioned> <has_front> <has_back>
<plane_eq> ::= <normal><d>
<normal> ::= <Vector3>
```

```

<user_data> ::= <user_data_key> <user_data_blob>
<user_data_blob> ::= <blob_size> <byte>*

```

.primitives file format in BNF grammar — BSP data section

The list below describes the sections in the file:

- **<blob_size>**
4-byte unsigned integer containing the number of bytes in the subsequent blob of binary data.
- **<d>**
Float value equal to the dot product of <plane_eq> and <normal>.
- **<has_back>**
1-bit flag indicating if node has a back child.
- **<has_front>**
1-bit flag indicating if node has a front child.
- **<is_partitioned>**
1-bit flag indicating if all triangles lie on the node's plane
- **<magic_number>**
4-byte number 0x00505342 containing version number in the last byte
- **<max_triangles>**
4-byte unsigned integer containing maximum size of all <triangle_index> lists.
- **<normal>**
Normal to plane's front.
- **<num_indices>**
2-byte unsigned integer containing the number of <triangle_index> sections in current <node> section.
- **<num_nodes>**
4-byte unsigned integer containing the number of nodes in <node>.
- **<num_triangles>**
4-byte unsigned integer containing the number of triangles in <triangle>.
- **<reserved>**
5-bit reserved number 10100.
- **<triangle_index>**
2-byte unsigned integer containing the index to the <triangle> section.
- **<triangle>**

XYZ position of the triangle's vertices.

- **<user_data_key>**

4-byte unsigned integer containing user-defined key associated with data.

- **<Vector3>**

Vector composed of 3 float values.

Chapter 30. shadows.xml

Located under `bigworld/res/system/data`, this file is used by the 3D game engine to configure various shadowing capabilities.

The grammar is listed below:

```
<shadows.xml>

  <enabled>    true|false </enabled>
  <intensity>  float      </intensity>
  <distance>   float      </distance>
  <fadeStart>  float      </fadeStart>
  <bufferSize> integer    </bufferSize>
  <maxCount>   integer    </maxCount>

</shadows.xml>
```

Grammar of shadow configuration file

The list below describes the tags in the shadow configuration file:

- **bufferSize**

Size of the shadow texture.

- **distance**

Shadow cut-off distance.

- **enabled**

Toggles the availability of shadows in the engine.

- **fadeStart**

Distance from which shadow will start to fade.

It will fade until reaches distance.

- **intensity**

Maximum shadow intensity.

Value ranges from 0 (fully transparent) to 1 (fully opaque).

- **maxCount**

Maximum number of simultaneous dynamic shadows casters in a scene.

This is the default value, but it is possible to set a smaller amount via graphics settings. The options available are derived by bit shifting this value towards zero.

Chapter 31. space.settings

Contains environment settings, bounding rectangle of grid squares, etc...

Mentioned in document Client Programming Guide's section *Chunks*, "Implementation files".

The grammar for `space.settings` is described below:

```
<root>
  <timeOfDay>      file  </timeOfDay>
  <skyGradientDome> file
  ?<farPlane>      float </farPlane>
  </skyGradientDome>
  ?<seas>
    ?<seaLevel>      float          </seaLevel>
    ?<wavePeriod>     float          </wavePeriod>
    ?<waveExtent>     float          </waveExtent>
    ?<tidePeriod>      float          </tidePeriod>
    ?<tideExtent>      float          </tideExtent>
    ?<surfaceTopColour> float float float float </surfaceTopColour>
    ?<surfaceBotColour> float float float float </surfaceBotColour>
    ?<underwaterColour> float float float float </underwaterColour>
  </seas>
  ?<farPlane>        float          </farPlane>
  ?<singleDir>        [true|false]   </singleDir>
  ?<startPosition>    float float float </startPosition>
  ?<startDirection>   float float float </startDirection>
  ?<bounds>
    <minX> integer </minX>
    <minY> integer </minY>
    <maxX> integer </maxX>
    <minY> integer </minY>
  </bounds>
</root>
```

Grammar of `space.settings`

The list below describes the tags in `space.settings`:

- **bounds**

The bounds in X- and Z-axis - the Y here is actually Z-axis in world space.

The unit is the grid size.

- **farPlane**

Space's far plane.

- **maxX (section bounds)**

Max grid bounds in X

- **maxY (section bounds)**

Max grid bounds in Z

- **minX (section bounds)**

Min grid bounds in X

- **minY (section bounds)**

Min grid bounds in Z

- **singleDir**

If this is set to true, all chunk files will reside in one folder. If this is set to false, all chunk files will be put into different folders (for large space)

- **skyGradientDome**

Sky configuration file.^A

The path is relative to game's resource folder — `<res>`.

- **startDirection**

The direction a player faced when he starts game

- **startPosition**

The position a player in when he starts game

- **timeOfDay**

Sky configuration file.^A

The path is relative to game's resource folder — `<res>`.

A — For details, see the document Client Programming Guide's section 3D Engine (Moo), "Features", "Lighting", "Light maps", "Sky light map". For details on file's grammar, see "<sky>.xml" on page 171 .

Chapter 32. .texanim

Animated textures can be applied to material via ModelEditor. For details, see the document Content Creation Manual's lesson **Create and Apply Animated Texture Maps → Applying animated texture map to a material**.

The grammar of `<texture>.texanim` files is listed below:

```
<root>

  <frames>  +[a-z]          </frames>
  <fps>      integer        </fps>
  +<texture> folder/file    </texture>

</root>
```

Grammar of `<texture>.texanim`

The list below describes the tags in file `<mouse_cursors>.xml`:

- **frames**

Order of textures to be played in a loop for the animated texture.

- **fps**

Number of frames per second to be displayed.

- **texture**

Texture to be displayed in a specific frame.

Frames are labelled from a to z, and can be played in any order (specified in frames tag), or repeated as part of a loop.

If no frames tag is specified, then the textures are replayed in the order in which they appear in the file.

Chapter 33. .texformat

Located under various folders under the <res> tree, texture_detail_levels.xml and .texformat files are divided into filename matching criteria, and conversion rules.

For more details, see the document Client Programming Guide's section *3D Engine (Moo)* → “Textures” → “Texture detail levels/compression”.

For details on how SpeedTree uses .texformat files to implement LOD, see the document Third-Party Integrations's section *SpeedTree* → “Level of detail”.

The grammar for the texture detail level configuration file is described below:

```
<root>
+<detailLevel>
  *<prefix>          string      </prefix>
  *<postfix>         string      </postfix>
  *<contains>        string      </contains>
  ?<maxDim>          integer     </maxDim>
  ?<minDim>          integer     </minDim>
  ?<reduceDim>       integer     </reduceDim>
  ?<format>          string      </format>
  ?<mipCount>        integer     </mipCount>
  ?<horizontalMips>  true|false  </horizontalMips>
  ?<noFilter>        true|false  </noFilter>
  ?<mipSize>         integer     <mipSize>
  ?<lodMode>         integer     <lodMode>
</detailLevel>
</root>
```

Grammar of texture detail level configuration file

The list below describes the tags in the texture detail level configuration file:

- **contains**

Filename match criterion.

Filename substring that must be matched to apply the conversion rule.

- **detailLevel**

Tag for detail level conversion rules.

- **format**

Conversion rule.

Indicates is mipmaps are stored along the horizontal axis. If set to true and mipCount is non-zero, then mipmaps are stored along the horizontal axis of the texture; otherwise, they are stored along vertical axis.

- **horizontalMips**

Conversion rule.

Indicates is mipmaps are stored along the horizontal axis.

If set to true and mipCount is non-zero, then mipmaps are stored along the horizontal axis of the texture; otherwise, they are stored along vertical axis.

- **maxDim**

Conversion rule.

The maximum width/height dimension that the converted texture might have.

- **minDim**

Conversion rule.

The minimum width/height dimension that the converted texture might have.

- **mipCount**

Conversion rule.

Number of mipmaps stored when precomputed mipmaps are stored in a single source texture.

- **mipSize**

Conversion rule.

Size of the topmost level of the mipmap, along the axis determined by the horizontalMips value.

- **noFilter**

Instructs Moo what filter to use when auto-generating mipmaps: if true, then point filtering will be used; if false, then box filtering will be used.

- **postfix**

Filename match criterion.

Filename postfix that must be matched for the conversion rule to be applied.

- **prefix**

Filename match criterion.

Path/filename prefix that must be matched to apply the conversion rule.

- **reduceDim**

Conversion rule.

The number of times to halve the dimensions of the texture.

- **lodMode**

Texture Quality Setting Modifier.

This modifier tweaks how the texture responds to the texture quality setting which varies from 0 (highest) to 2 (lowest). How this modifier tweaks the quality setting is described in the following table.

	texture quality setting		
lodMode	0	1	2
0 (disabled)	0	0	0
1 (normal)	0	1	2
2 (low bias)	0	1	1
3 (high bias)	0	0	1

Chapter 34. texture_detail_levels.xml

For details, see *.texformat* on page 139 .

Chapter 35. ual_config.xml

This file configures various aspects of the appearance, layout, and working of the Asset Browser on the BigWorld tools that use it (WorldEditor, ModelEditor, and ParticleEditor), and is located under `bigworld/tools/<tool_folder>/resources/ual`.

Mentioned in document Content Tools Reference Guide's section *Asset Browser*, “Customisation”, the tags in this file are documented in `bigworld/tools/worldeditor/resources/ual/ual_config.xml`.

Chapter 36. .visual

Defined in various sub-folders under the resource tree *<res>* (for example, *<res>/environments*, *<res>/flora*, *<res>/sets/vehicles*, etc...), the .visual file format specification is illustrated below:

```
<root>

  NodeSection 1

    <renderSet>
      <treatAsWorldSpaceObject> [true|false] </treatAsWorldSpaceObject>
    +<node>                                string </node>
      <geometry>
        <vertices>          object.vertices </vertices>
        <primitive>         object.indices </primitive>
      +<primitiveGroup> integer
        <material>
          EffectMaterial 2
        </material>
      </primitiveGroup>
    </geometry>
  </renderSet>

  <boundingBox>
    <min>  float float float </min>
    <max>  float float float </max>
  </boundingBox>

  *PortalSection 3

  *<boundary>
    <normal> float float float </normal>
    <d>      float </d>
    ?PortalSection 4
  </boundary>

</root>
```

Grammar of visual file

- 1 See "NodeSection" on page 147 .
- 2 See *EffectMaterial* section on page 35 .
- 3 See "PortalSection" on page 147 .
- 4 See "PortalSection" on page 147 .

The list below describes the tags in the visual file:

- **boundary**

Section describing the planes surrounding the visual, much like the boundingBox section.

These planes define a convex hull around the visual.

- **boundingBox**

Minimum and maximum XYZ coordinates of the model's bounding box.

If visual has skinned vertices or animation, then this section is the visual's initial pose's bounding box.

- **d (section boundary)**

Distance along the normal between boundary and origin.

- **EffectMaterial**

For details, see *EffectMaterial* section on page 35 .

- **geometry (section renderSet)**

List of geometries used by this renderSet.

- **material (section renderSet/geometry/primitiveGroup)**

Properties for the primitive group's material, such as shader, collision flags, maps, etc...

- **max (section boundingBox)**

Maximum extents of model's bounding box

- **min (section boundingBox)**

Minimum extents of model's bounding box

- **node (section renderSet)**

Node referenced by this renderSet.

- **NodeSection**

For details, see "NodeSection" on page 147 .

- **normal (section boundary)**

Normal of the plane surrounding the visual.

- **PortalSection (main section, and boundary)**

For details, see "PortalSection" on page 147 .

- **primitive (section renderSet/geometry)**

The name of the primitives used by this geometry (loaded from `.primitives` file — for details on this file's grammar, see `.primitives` on page 129).

- **primitiveGroup (section renderSet/geometry)**

Settings for primitive group.

Primitive group is a group of triangles that share the same material properties.

- **renderSet**

Section for one or more objects with same transforms.

- **treatAsWorldSpaceObject (section renderSet)**

Determines if lighting and other position-based properties are passed to shader in world space.

- **vertices (section renderSet/geometry)**

The name of the vertices used by this geometry (loaded from `.primitives` file — for details on this file's grammar, see `.primitives` on page 129).

36.1. NodeSection

The grammar for the **NodeSection** is described below:

```
<node>

  <identifier> string </identifier>

  <transform>
    <row0> float float float </row0>
    <row1> float float float </row1>
    <row2> float float float </row2>
    <row3> float float float </row3>
  </transform>

  *NodeSection

</node>
```

Grammar of **NodeSection** in visual file

The list below describes the tags in **NodeSection**:

- **identifier**

Node's unique ID.

The root node must be called Scene Root.

- **row0, row1, row2, row3**

Node's transform matrices.

- **transform**

3x4 matrix defining node's transform relative to the parent node.

36.2. PortalSection

Portals describe the line of sight into neighbouring chunk, and should be present only in shells.

The grammar for the **PortalSection** is described below:

```
<portal> [heaven|invasive|Empty]

  <uAxis> float float float </uAxis>
  +3<point> float float float </point>

</portal>
```

Grammar of **PortalSection** in visual file

The list below describes the tags in **PortalSection**:

- **point**

XYZ coordinates of one of portal's corners.

- **type**

Describes the type of portal being defined.

Can have the following values:

- **heaven**

Outside lighting will be applied to shell, and from inside it you will be able to see outside chunks.

- **invasive**

Portal will have same properties as a heaven portal, but shell will also be visible from outside chunks.

- **Empty**

Portal will be purely a connecting one. This type can only be linked to other connecting portals.

- **uAxis**

The cross vector for the coordinate system of the portal points.

The up vector is calculated by taking the cross product of the boundary's normal and the `uAxis`.

Chapter 37. visual_rules.xml

Used by 3ds Max and Maya visual exporters to validate the files before exporting, the set of rules is defined in `bigworld/tools/res/visual_rules.xml`. For more details, see the document *Content Tools Reference Guide's chapter 3ds Max and Maya Exporters*.

The grammar of `visual_rules.xml` is listed below:

```
<visual_rules.xml>

  *<rule>
    <identifier>          string          </identifier>
    ?<parent>             string          </parent>
    ?<exportAs>           normal|static|static with nodes </exportAs>
    ?<path>               folder          </path>
    ?<filespec>           string          </filespec>
    ?<minSize>            float float float </minSize>
    ?<maxSize>            float float float </maxSize>
    ?<maxNodesPerRenderSet> integer        </maxNodesPerRenderSet>
    ?<maxTriangles>       integer        </maxTriangles>
    ?<minTriangles>       integer        </minTriangles>
    ?<portals>            true|false      </portals>
    ?<portalSnap>         float float float </portalSnap>
    ?<portalDistance>     float          </portalDistance>
    ?<portalOffset>       float          </portalOffset>
    *<hardPoint>          string          </hardPoint>
    ?<checkUnknownHardPoints> true|false  </checkUnknownHardPoints>
  </rule>

</visual_rules.xml>
```

Grammar of `bigworld/tools/res/visual_rules.xml`

The list below describes the tags in file `visual_rules.xml`:

- **checkUnknownHardPoints**

Specified if the hard point not specified in the set of `<hardPoint>` tags should be flagged as error.

- **exportAs**

Specifies how to save the visual. Can have one of the following values: `normal`, `static`, or `static with nodes`.

- **fileSpec**

Matching criteria (using wildcards) specifying the visuals for which this rule applies to. This option is not inherited via the tag parent.

- **hardPoint**

Name of the hard point that the visual must have. The set of hard points to check against is the one formed by all hard points mentioned in all parents. This tag may be defined multiple times for the same rule.

- **identifier**

Name of the rule.

- **maxNodesPerRenderSet^A**

Maximum nodes allowed per renderset, *i.e.*, how many bones each geometric element is allowed to use for skinning.

- **maxSize^A**

Maximum size of the visual in metres.

- **maxTriangles^A**

Maximum number of triangles in the visual.

- **minSize^A**

Minimum size of the visual in metres.

- **minTriangles^A**

Minimum number of triangles in the visual.

- **parent**

Name of the rule from which to inherit options.

For each option, the plug-in will search up the hierarchy until it finds a match for it.

The options `path` and `fileSpec` are not inherited.

- **path**

Resource folder which visuals this rule applies to.

This option is not inherited via the tag `parent`.

- **portalDistance^A**

Number of which the distance between the origin and the portal must be a multiple of.

- **portalOffset^A**

Number of which the centre of the portal must be a multiple of on the portal plane.

- **portals**

Specifies if the visual may have portals.

- **portalSnap^A**

Number of which the bounding box of the portal must be a multiple of.

A value must be specified for each dimension.

A — A value of zero indicates that the option should not be checked

Chapter 38. .xml

Various BigWorld files use the XML format, and this chapter describes the grammar of the following ones:

- Lens effect configuration files — For details, see “<effect>.xml” on page 151 .
- Enumeration files — For details, see “<enumeration>.xml” on page 153 .
- Flora configuration files — For details, see “<flora>.xml” on page 154 .
- Graphics settings configuration files — For details, see “<graphics_settings>.xml” on page 157 .
- Light configuration files — For details, see “<light>.xml” on page 158 .
- Mouse cursor configuration files — For details, see “<mouse_cursors>.xml” on page 160 .
- Particle system files — For details, see “<particle>.xml” on page 161 .
- Sky configuration files — For details, see “<sky>.xml” on page 171 .
- Special Effects files — For details, see “<fx>.xml” on page 174 .
- Weather System configuration files — For details, see “<weather>.xml” on page 181 .

38.1. <effect>.xml

Used by the client engine to configure lens effects, these files are located under `fantasydemo/res/environments/fx`.

The grammar of <effect>.xml files is listed below:

```
<root>
  <maxDistance> float </maxDistance>
  <area> float </area>
  <fadeSpeed> float </fadeSpeed>

  *<Flare>
    <occlusionLevel> float </occlusionLevel>
    <type> folder/file </type>
    <size> float </size>
    <width> float </width>
    <height> float </height>
    <depth> float </depth>
    <rgba> float float float float </rgba>

    ?<secondaries>
      *<Flare>
        <type> folder/file </type>
        <size> float </size>
        <depth> float </depth>
        <rgba> float float float float </rgba>
      </Flare>
    </secondaries>

  </Flare>
</root>
```

Grammar of `fantasydemo/res/environments/fx/<effect>.xml`

The list below describes the tags in `<effect>.xml`:

- **area**

Radius in pixels of the flare that will be used for occlusion calculations.

By default, this value is 1, as used for point lens effects. Area lens effects such as the sun require a larger value.

- **depth**

Distance along the vector from the light source to the centre of the screen.

If set to 1.0, the flare will be displayed directly on the light source. If set to anything less than 1.0, flare will be moved somewhere along the vector.

Values below 0.0 are valid, and will place the flare on the other side of the pivot point (0.0, the centre of the screen).

- **fadeSpeed**

Speed in which the lens effect's flares will fade in/out. Larger numbers result in slower fade in/out speeds.

- **Flare**

The flares to use for the lens effect.

If multiple flares are specified, then the flare which `occlusionLevel` most closely matches the current visibility will be rendered on the screen.

- **height**

size sets both height and width, therefore set either size or both height and width.

Clip Coordinates range from (-1,-1) to (1,1) so a size of 2.0 represents the entire screen.

Height of the lens flare in clip coordinates. Clip coordinates range from (-1,-1) to (1,1), and so a height of 2.0 represents the entire screen.

- **maxDistance**

Distance up to which flare will still be seen. This value also applies to secondary flares.

- **occlusionLevel**

Value between 0.0 and 1.0 specifying the percentage of the lens effect that needs to be visible in order for this flare to be rendered.

All `occlusionLevel` values between flares must be unique — flares with duplicate `occlusionLevel` values are not loaded.

- **rgba**

Colour value to tint the lens flare's texture map. This colour is multiplied by the values in the texture.

- **secondaries**

Tag for section specifying secondary flares. Secondary flares are displayed whenever the primary flare is displayed. Set the `depth` value for secondary flares to something other than 1.0 for the best results (although a value of 1.0 is still valid).

- **size**

size sets both height and width, therefore set either size or both height and width.

Clip Coordinates range from (-1,-1) to (1,1) so a size of 2.0 represents the entire screen.

Size (height and width) of the lens flare in clip coordinates.

- **type**

Material generated by ModelEditor used to describe the flare. For details on grammar of materials, see *.mfm* on page 59 .

- **width**

size sets both height and width, therefore set either size or both height and width.

Clip Coordinates range from (-1,-1) to (1,1) so a size of 2.0 represents the entire screen.

Width of the lens flare in clip coordinates.

38.2. <enumeration>.xml

BigWorld's enumeration files (*dxenum.xml* and *<enumeration>.xml*) provide descriptions for enumerated values defined in DirectX, so that they can be used in source files (such as FX files) and BigWorld tools (such as ModelEditor).

BigWorld uses the enumeration file specified in *<res>/resources.xml*'s section *system/dxenum* (for details on this file, see the document Client Programming Guide's section *Overview* → “Configuration files” → “File resources.xml”). It is initially set to *system/data/dxenum.xml*, which is the file shipped with BigWorld Technology.

For details on the implementation of enumeration for material properties in WorldEditor and ModelEditor, see the document Client Programming Guide's section *3D Engine (Moo)* → “Textures” → “Texture detail levels/compression”.

The grammar for the enumeration file is described below:

```
<root>

  +<enum_name>
    +<ENTRY>
      <NAME>    string  </NAME>
      <VALUE>   number  </VALUE>
      <DESC>    string  </DESC>
    </ENTRY>
  </enum_name>

  *<enum_alias>
    <ALIAS>  enum_name  </ALIAS>
  </enum_alias>

</root>
```

Grammar of enumeration file

The list below describes the tags in the enumeration file:

- **ALIAS**

Name of the enumeration which entries the current *enum_alias* will use.

Value must be the `enum_name` of another enumeration specified in the file.

- **DESC**

Description of the entry.

- **ENTRY**

Tag for section specifying a value entry for the enumeration.

- ***enum_alias***

Name of an enumeration for which the entries have already been specified as part of another enumeration.

- ***enum_name***

Name by which to refer to the enumeration.

- **NAME**

Name by which to refer to the enumeration entry.

For example, FALSE for VALUE of 1, or BLENDFACTOR for VALUE of 14.

- **VALUE**

Numeric value of the entry.

38.3. <flora>.xml

Flora configuration files have their location set by the `floraXML` tag in `<res>/resources.xml`.

The XML flora file defines among other things, the light map to be used, plus the ecotypes and noise generation functions.

For details on flora light maps, see the document Client Programming Guide's section *3D Engine (Moo)* → “Features” → “Lighting” → “Flora light map”.

For more details on `<res>/resources.xml`, see the document Client Programming Guide's section *Overview* → “Configuration files” → “File `resources.xml`”.

The format of this file is described below:

```
<flora.xml>

  ?LightMapSection    1

    <vb_size>         integer  </vb_size>
    <texture_width>    integer  </texture_width>
    <texture-height>   integer  </texture_height>

    <ecotypes>

      ?<empty>
        *<texture>    file      </texture>
        ?<sound_tag>   string    </sound_tag>
      </empty>

      *<ecotype>

        *<texture>     file      </texture>
```

```

    *<visual>      file
      ?<flex>      float </flex>
    </visual>

    [GeneratorSection | VisualSection]

    </ecotype>
  </ecotypes>
</flora.xml>

```

Grammar of the flora configuration file

1 See *LightMapSection* on page 55 .

The grammar for the **GeneratorSection** is described below:

```

<generator> [empty | visual1 | chooseMax]

  <!-- If empty -->
  <empty> </empty>

  <!-- If visual -->
  VisualSection

  <!-- If chooseMax -->
  * [NoiseSection | RandomSection | FixedSection]

</generator>

```

Grammar of flora configuration file's **GeneratorSection**

1 Please note that the keyword `visual` may appear as a value for the `generator` section (see the grammar above), or as a regular tag (see the grammar for **VisualSection** below)

The grammar for the **VisualSection** is described below:

```

*<visual>1 file
  ?<flex> float </flex>
</visual>

```

Grammar of flora configuration file's **VisualSection**

The grammar for the **NoiseSection** is described below:

```

*<noise>
  <frequency> float </frequency>
  GeneratorSection
</noise>

```

Grammar of flora configuration file's **NoiseSection**

The grammar for the **RandomSection** is described below:

```

<random> float_btwn_0_and_1 </random>

```

Grammar of flora configuration file's **RandomSection**

The grammar for the **FixedSection** is described below:

```
<fixed> float_btwn_0_and_1 </fixed>
```

Grammar of flora configuration file's **FixedSection**

The list below describes the tags in the flora configuration file:

- **chooseMax**

One of the possible values for tag generator.

If this type is specified, then it will define one more value sub-generators (possible values: noise, random, and fixed).

Section specifying functions that generate values between 0 and 1.

The one that generates the highest value will be used.

- **ecotype**

Arbitrary name identifying the ecotype.

More than one texture can be associated with the ecotype.

An ecotype might have a `sound_tag` associated with it, as well as a detail object (returned by generator section).

The ecotype of a part of the terrain is determined by the texture with the highest blend applied to it.

- **ecotypes**

List of ecotypes.

- **fixed**

One of the possible value sub-generators that can be specified if generator is chooseMax.

Value would normally be between 0 and 1, although higher values can be specified to make sure that this generator is always chosen.

- **flex**

Amount of flexion to be applied for visual, in relation to standard shader amount.

- **frequency**

Perlin noise frequency. Value must be between 0 and 1.

- **generator**

Specifies the parameters for choosing the detail objects to be randomly placed and oriented over the ecotype.

Script `bigworld/tools/WorldEditor/resources/scripts/ Ecotypes.py` can be used to specify the rules of placement (i.e., how steep should the terrain be before the object is not placed, etc...).

- **LightMapSection**

For details, see *LightMapSection* on page 55 .

- **noise**

One of the possible value sub-generators that can be specified if generator is chooseMax.

Higher output value produces larger and fewer patches.

- **random**

One of the possible value sub-generators that can be specified if generator is chooseMax.

This returns a random value between 0 and 1.

- **texture**

One of a list of terrain textures that will produce the given ecotype.

Note that it should be a DDS file, as this is the type of file stored in the terrain data.

Script `bigworld/tools/WorldEditor/resources/scripts/ Ecotypes.py` uses these entries to match terrain textures with ecotypes.

- **texture_height**

Maximum height of flora visuals.

- **texture_width**

Maximum width of flora visuals.

- **vb_size**

Size of the vertex buffer (in bytes).

- **visual**

- **As tag**

File containing information about one of a list of flora visuals that is to be used for this ecotype.

Ecotypes may contain more than one visual. In this case, visuals are used on a rotating basis, starting at the top of the list.

For more details, see `.visual` on page 145 .

- **As value for generator section**

One of the possible value sub-generators that can be specified if generator is chooseMax.

Specifies that visuals will be used as flora objects. It may contain a list of visuals, which are used on a rotating basis, starting at the top of the list.

38.4. <graphics_settings>.xml

The graphics settings configuration files have their location set by the `graphicsSettingsXML` tag in `resources.xml` (for details, see the document Client Programming Guide's section *Overview* → “Configuration files” → “File `resources.xml`”).

The XML graphics settings file hosts the customisable options of the client's graphics settings feature — for details, see the document Client Programming Guide's section *3D Engine (Moo)* → “Graphics settings”.

The grammar of this `<graphics_settings>.xml` is described below:


```

<graphics_settings.xml>

  <flora>
    +<option>
      <label> string </label>
      <value> float </value>
    </option>
  </flora>

  <farPlane>
    +<option>
      <label> string </label>
      <value> float </value>
    </option>
  </farPlane>

</graphics_settings.xml>

```

Grammar of `<graphics_settings>.xml` configuration file

The list below describes the tags in `<graphics_settings>.xml`:

- **flora**

Section for setting flora density labels and distance from camera.

For details, see the document Client Programming Guide's section *3D Engine (Moo)* → “Graphics settings” → “Customising options” → “FLORA_DENSITY”.

- **farPlane**

Section for setting far plane labels and distance from camera.

For details, see the document Client Programming Guide's section *3D Engine (Moo)* → “Graphics settings” → “Customising options” → “FAR_PLANE”.

38.5. `<light>.xml`

Mentioned in:

- Document Content Tools Reference Guide's section *WorldEditor* → “Assets” → “Asset Browser panel”.
- Document Content Creation Manual's lesson **Add Lights to the World**.

The grammar for the XML light configuration files for ambient lights is described below:

```

<fileName>
  <ambientLight>
    <colour>      float float float </colour>
    ?<multiplier> float          </multiplier>
  </ambientLight>
</fileName>

```

Grammar of XML light configuration file — Ambient lights

The grammar for the XML light configuration files for directional lights is described below:

```

<fileName>

```

```

<directionalLight>
  <colour>      float float float  </colour>
  <direction>   float float float  </direction>
  ?<dynamic>    [true|false]       </dynamic>
  ?<static>     [true|false]       </static>
  ?<specular>   [true|false]       </specular>
  ?<multiplier> float              </multiplier>
</directionalLight>
</fileName>

```

Grammar of XML light configuration file — Directional lights

The grammar for the XML light configuration files for flares is described below:

```

<fileName>
  <flare>
    <resource>  file                </resource>
    <position>  float float float  </position>
    <colour>    float float float  </colour>
  </ambientLight>
</fileName>

```

Grammar of XML light configuration file — Flares

The grammar for the XML light configuration files for omni lights is described below:

```

<fileName>
  <omniLight>
    <colour>      float float float  </colour>
    <position>    float float float  </position>
    <innerRadius>  float              </innerRadius>
    <outerRadius>  float              </outerRadius>
    ?<dynamic>    [true|false]       </dynamic>
    ?<static>     [true|false]       </static>
    ?<specular>   [true|false]       </specular>
    ?<multiplier> float              </multiplier>
  </omniLight>
</fileName>

```

Grammar of XML light configuration file — Omni lights

The grammar for the XML light configuration files for spot lights is described below:

```

<fileName>
  <spotLight>
    <colour>      float float float  </colour>
    <position>    float float float  </position>
    <direction>   float float float  </direction>
    <innerRadius>  float              </innerRadius>
    <outerRadius>  float              </outerRadius>
    <cosConeAngle> float              </cosConeAngle>
    ?<dynamic>    [true|false]       </dynamic>
    ?<static>     [true|false]       </static>
    ?<specular>   [true|false]       </specular>
    ?<multiplier> float              </multiplier>
  </spotLight>
</fileName>

```

Grammar of XML light configuration file — Spot lights

The list below describes the tags in the MVL light configuration file:

- **colour** (Available for **ambientLight**, **directionalLight**, **flare**, **omniLight**, and **spotlight**)

Colour of the light.

- **cosConeAngle** (Available for **spotlight**)

Cosine of the angle of light's cone.

- **direction** (Available for **directionalLight** and **spotlight**)

Direction at which light will be projected.

- **dynamic** (Available for **directionalLight**, **omniLight**, and **spotlight**)

Specifies that light should be applied to dynamic, i.e., moving objects.

- **innerRadius** (Available for **omniLight**, and **spotlight**)

Area over which the light will be at full intensity.

- **outerRadius** (Available for **omniLight**, and **spotlight**)

Total area influenced by light.

- **position** (Available for **flare**, **omniLight**, and **spotlight**)

XYZ point of origin for the light.

- **resource** (Available for **flare**)

Configuration file for the flare effect.

- **specular** (Available for **directionalLight**, **omniLight**, and **spotlight**)

Boolean specifying whether light should be applied to materials with a specular map specified.

- **static** (Available for **directionalLight**, **omniLight**, and **spotlight**)

Boolean specifying whether light should be applied to static, i.e., scene objects.

38.6. <mouse_cursors>.xml

Used by the client engine to define the shapes of mouse cursor available from within the game, the mouse cursor definition file can be specified in tag `gui/cursorDefinitions` in `res/resources.xml`, but defaults to `res/gui/mouse_cursor.xml` (the tags are documented in the file itself — for details, see Client Programming Guide's section *Overview* → “Configuration files” → “File `resources.xml`”).

For details on how to control the behaviour and appearance of mouse cursor via game scripts, see the document Client Programming Guide's section *Graphical User Interface (GUI)* → “Mouse cursor”.

The grammar of `<mouse_cursors>.xml` files is listed below:

```
<mouse_cursors>

  *<cursor_name>
```

```

        <hotspot>  integer integer  </hotspot>
        <texture>  folder/file      </texture>
    </cursor_name>

</mouse_cursors>

```

Grammar of <mouse_cursors>.xml

The list below describes the tags in file <mouse_cursors>.xml:

- **cursor_name**

Arbitrary value identifying the type of cursor.

For example, FantasyDemo's mouse cursor definition file (res/gui/ mouse_cursors.xml) is shipped with the following cursor names:

- **arrow**
- **drag**
- **no**
- **point**
- **hotspot**

XY location of the cursor icon's hotspot in the respective texture.

- **texture**

Image file to be associated to the icon type.

38.7. <particle>.xml

Generated by ParticleEditor (for more details, see Content Tools Reference Guide's chapter *ParticleEditor*), the grammar for <particle>.xml files is described below:

```

<file_name>

+<sub_system>
    <serialiseVersionData>  integer          </serialiseVersionData>
    <capacity>              integer          </capacity>
    <windFactor_>           float            </windFactor_>
    <explicitTransform_>    [true|false]     </explicitTransform_>
    <explicitPosition_>     float float float </explicitPosition_>
    <explicitDirection_>    float float float </explicitDirection_>
    <localOffset_>          float float float </localOffset_>
    <maxLod_>               float            </maxLod_>
    <fixedFrameRate_>       float            </fixedFrameRate_>

    <Actions>
        *Source              1
        *Sink                2
        *Barrier             3
        *Force               4
        *Stream              5
        *Jitter              6
        *Scaler              7
        *TintShader          8

```

```

        *NodeClamp          9
        *Orbitor            10
        *Flare              11
        *Collide            12
        *MatrixSwarm        13
        *Magnet             14
        *Splat              15
    </Actions>
    <Renderer>
        { Amp               16
          Blur              17
          Mesh              18
          Sprite            19
          Trail             20
        }
    </Renderer>

</file_name>

```

Grammar of <particle>.xml

- 1 See "SourceComponent" on page 163 .
- 2 See "SinkComponent" on page 164 .
- 3 See "BarrierComponent" on page 164 .
- 4 See "ForceComponent" on page 165 .
- 5 See "StreamComponent" on page 165 .
- 6 See "JitterComponent" on page 165 .
- 7 See "ScalerComponent" on page 166 .
- 8 See "TintShaderComponent" on page 166 .
- 9 See "NodeClampComponent" on page 166 .
- 10 See "OrbitorComponent" on page 167 .
- 11 See "FlareComponent" on page 167 .
- 12 See "CollideComponent" on page 167 .
- 13 See "MatrixSwarmComponent" on page 167 .
- 14 See "MagnetComponent" on page 168 .
- 15 See "SplatComponent" on page 168 .
- 16 See "AmpParticleRenderer" on page 168 .
- 17 See "BlurParticleRenderer" on page 169 .
- 18 See "MeshParticleRenderer" on page 169 .
- 19 See "SpriteParticleRenderer" on page 169 .
- 20 See "TrailParticleRenderer" on page 169 .

38.7.1. Sub system components

The sub-sections below describe the 15 possible sub system components of the particle system (they are described in the order in which they appear in ParticleEditor's **Add Component** drop-down list box):

- **SourceComponent** — See "SourceComponent" on page 163 .
- **SinkComponent** — See "SinkComponent" on page 164 .
- **BarrierComponent** — See "BarrierComponent" on page 164 .
- **ForceComponent** — See "ForceComponent" on page 165 .
- **StreamComponent** — See "StreamComponent" on page 165 .
- **JitterComponent** — See "JitterComponent" on page 165 .
- **ScalerComponent** — See "ScalerComponent" on page 166 .

- **TintShaderComponent** — See “TintShaderComponent” on page 166 .
- **NodeClampComponent** — See “NodeClampComponent” on page 166 .
- **OrbitorComponent** — See “OrbitorComponent” on page 167 .
- **FlareComponent** — See “FlareComponent” on page 167 .
- **CollideComponent** — See “CollideComponent” on page 167 .
- **MatrixSwarmComponent** — See “MatrixSwarmComponent” on page 167 .
- **MagnetComponent** — See “MagnetComponent” on page 168 .
- **SplatComponent** — See “SplatComponent” on page 168 .

38.7.1.1. SourceComponent

The grammar for the **SourceComponent** is described below:

```

<Source>
  <delay_>      float  </delay_>
  <minimumAge_> float  </minimumAge_>
  <name_>       string </name_>
  <pPositionSrc>
    {BoxVectorGenerator      | 1
     CylinderVectorGenerator | 2
     LineVectorGenerator     | 3
     PointVectorGenerator    | 4
     SphereVectorGenerator   | 5
    }
  </pPositionSrc>
  <pVelocitySrc>
    {BoxVectorGenerator      | 6
     CylinderVectorGenerator | 7
     LineVectorGenerator     | 8
     PointVectorGenerator    | 9
     SphereVectorGenerator   | 10
    }
  </pVelocitySrc>
  <motionTriggered_> [true|false] </motionTriggered_>
  <timeTriggered_>  [true|false] </timeTriggered_>
  <grounded_>       [true|false] </grounded_>
  <dropDistance_>   float </dropDistance_>
  <rate_>           float </rate_>
  <sensitivity_>    float </sensitivity_>
  <activePeriod_>   float </activePeriod_>
  <sleepPeriod_>    float </sleepPeriod_>
  <sleepPeriodMax_> float </sleepPeriodMax_>
  <minimumSize_>    float </minimumSize_>
  <maximumSize_>    float </maximumSize_>
  <forcedUnitSize_> integer </forcedUnitSize_>
  <allowedTimeInSeconds_> float </allowedTimeInSeconds_>
  <initialRotation_> float float </initialRotation_>
  <randomInitialRotation_> float float </randomInitialRotation_>
  <initialColour_>  float float float float </initialColour_>
  <randomSpin_>     [true|false] </randomSpin_>
  <minSpin_>        float </minSpin_>
  <maxSpin_>        float </maxSpin_>
  <ignoreRotation_> [true|false] </ignoreRotation_>
  <inheritVelocity_> float </inheritVelocity_>
</Source>

```

Grammar of **SourceComponent** in `<particle>.xml`

- 1 See "BoxVectorGenerator" on page 170 .
- 2 See "CylinderVectorGenerator" on page 170 .
- 3 See "LineVectorGenerator" on page 171 .
- 4 See "PointVectorGenerator" on page 171 .
- 5 See "SphereVectorGenerator" on page 171 .
- 6 See "BoxVectorGenerator" on page 170 .
- 7 See "CylinderVectorGenerator" on page 170 .
- 8 See "LineVectorGenerator" on page 171 .
- 9 See "PointVectorGenerator" on page 171 .
- 10 See "SphereVectorGenerator" on page 171 .

38.7.1.2. SinkComponent

The grammar for the **SinkComponent** is described below:

```
<Sink>
  <delay_>      float  </delay_>
  <minimumAge_> float  </minimumAge_>
  <name_>        string </name_>
  <maximumAge_> float  </maximumAge_>
  <minimumSpeed_> float </minimumSpeed_>
</Sink>
```

Grammar of **SinkComponent** in `<particle>.xml`

38.7.1.3. BarrierComponent

The grammar for the **BarrierComponent** is described below:

```
<Barrier>
  <delay_>      float  </delay_>
  <minimumAge_> float  </minimumAge_>
  <name_>        string </name_>
  <shape_>       [1|2|3] </shape_> 1
  <reaction_>    [0|1|2] </reaction_> 2
  <vecA_>        float float float </vecA_>
  <vecB_>        float float float </vecB_>
  <radius_>      float  </radius_>
</Barrier>
```

Grammar of **BarrierComponent** in `<particle>.xml`

1 Possible values:

- 1 — Vertical cylinder
- 2 — Box
- 3 — Sphere

13 Possible values:

- 0 — Bounce
- 1 — Remove
- 2 — Allow

38.7.1.4. ForceComponent

The grammar for the **ForceComponent** is described below:

```
<Force>
  <delay_>      float          </delay_>
  <minimumAge_> float          </minimumAge_>
  <name_>        string         </name_>
  <vector_>      float float float </vector_>
</Force>
```

Grammar of **ForceComponent** in *<particle>.xml*

38.7.1.5. StreamComponent

The grammar for the **StreamComponent** is described below:

```
<Stream>
  <delay_>      float          </delay_>
  <minimumAge_> float          </minimumAge_>
  <name_>        string         </name_>
  <vector_>      float float float </vector_>
  <halfLife_>    float          </halfLife_>
</Stream>
```

Grammar of **StreamComponent** in *<particle>.xml*

38.7.1.6. JitterComponent

The grammar for the **JitterComponent** is described below:

```
<Jitter>
  <delay_>      float          </delay_>
  <minimumAge_> float          </minimumAge_>
  <name_>        string         </name_>
  <affectPosition_> [true|false] </affectPosition_>
  <affectVelocity_> [true|false] </affectVelocity_>
  <pPositionSrc>
    {BoxVectorGenerator      | 1
     CylinderVectorGenerator | 2
     LineVectorGenerator     | 3
     PointVectorGenerator    | 4
     SphereVectorGenerator   | 5
    }
  </pPositionSrc>
  <pVelocitySrc>
    {BoxVectorGenerator      | 6
     CylinderVectorGenerator | 7
     LineVectorGenerator     | 8
     PointVectorGenerator    | 9
     SphereVectorGenerator   | 10
    }
  </pVelocitySrc>
</Jitter>
```

Grammar of **JitterComponent** in *<particle>.xml*

1 See “BoxVectorGenerator” on page 170 .

- 2 See “CylinderVectorGenerator” on page 170 .
- 3 See “LineVectorGenerator” on page 171 .
- 4 See “PointVectorGenerator” on page 171 .
- 5 See “SphereVectorGenerator” on page 171 .
- 6 See “BoxVectorGenerator” on page 170 .
- 7 See “CylinderVectorGenerator” on page 170 .
- 8 See “LineVectorGenerator” on page 171 .
- 9 See “PointVectorGenerator” on page 171 .
- 10 See “SphereVectorGenerator” on page 171 .

38.7.1.7. ScalerComponent

The grammar for the **ScalerComponent** is described below:

```
<Scaler>
  <delay_>      float  </delay_>
  <minimumAge_> float  </minimumAge_>
  <name_>        string </name_>
  <size_>        float  </size_>
  <rate_>        float  </rate_>
</Scaler>
```

Grammar of **ScalerComponent** in `<particle>.xml`

38.7.1.8. TintShaderComponent

The grammar for the **TintShaderComponent** is described below:

```
<TintShader>
  <delay_>      float          </delay_>
  <minimumAge_> float          </minimumAge_>
  <name_>        string         </name_>
  <repeat_>      [true|false]   </repeat_>
  <period_>      float          </period_>
  <tints_>
    *<Tint>
      <time>     float          </time>
      <color>    float float float </color>
    </Tint>
  </tints_>
</TintShader>
```

Grammar of **TintShaderComponent** in `<particle>.xml`

38.7.1.9. NodeClampComponent

The grammar for the **NodeClampComponent** is described below:

```
<NodeClamp>
  <delay_>      float          </delay_>
  <minimumAge_> float          </minimumAge_>
  <name_>        string         </name_>
  <fullyClamp_> [true|false]   </fullyClamp_>
</NodeClamp>
```

Grammar of **NodeClampComponent** in `<particle>.xml`

38.7.1.10. OrbitorComponent

The grammar for the **OrbitorComponent** is described below:

```
<Orbitor>
  <delay_>          float          </delay_>
  <minimumAge_>     float          </minimumAge_>
  <name_>           string         </name_>
  <point_>          float float float </point_>
  <angularVelocity_> float         </angularVelocity_>
  <affectVelocity_> [true|false]   </affectVelocity_>
</Orbitor>
```

Grammar of **OrbitorComponent** in `<particle>.xml`

38.7.1.11. FlareComponent

The grammar for the **FlareComponent** is described below:

```
<Flare>
  <delay_>          float          </delay_>
  <minimumAge_>     float          </minimumAge_>
  <name_>           string         </name_>
  <flareName_>      folder/file    </flareName_>
  <flareStep_>      integer        </flareStep_>
  <colourize_>      [true|false]   </colourize_>
  <useParticleSize_> [true|false] </useParticleSize_>
</Flare>
```

Grammar of **FlareComponent** in `<particle>.xml`

38.7.1.12. CollideComponent

The grammar for the **CollideComponent** is described below:

```
<Collide>
  <delay_>          float          </delay_>
  <minimumAge_>     float          </minimumAge_>
  <name_>           string         </name_>
  <spriteBased_>    [true|false]   </spriteBased_>
  <elasticity_>      float          </elasticity_>
  <minAddedRotation_> float        </minAddedRotation_>
  <maxAddedRotation_> float        </maxAddedRotation_>
  <entityID_>       integer        </entityID_>
  <soundTag_>       string         </soundTag_>
  <colourize_>      [true|false]   </colourize_>
  <useParticleSize_> [true|false] </useParticleSize_>
</Collide>
```

Grammar of **CollideComponent** in `<particle>.xml`

38.7.1.13. MatrixSwarmComponent

The grammar for the **MatrixSwarmComponent** is described below:

```
<MatrixSwarm>
```

```

    <delay_>         float  </delay_>
    <minimumAge_>    float  </minimumAge_>
    <name_>           string </name_>
  </MatrixSwarm>

```

Grammar of **MatrixSwarmComponent** in `<particle>.xml`

38.7.1.14. MagnetComponent

The grammar for the **MagnetComponent** is described below:

```

<Magnet>
  <delay_>         float  </delay_>
  <minimumAge_>    float  </minimumAge_>
  <name_>           string </name_>
  <strength_>      float  </strength_>
  <minDist_>       float  </minDist_>
</Magnet>

```

Grammar of **MagnetComponent** in `<particle>.xml`

38.7.1.15. SplatComponent

The grammar for the **SplatComponent** is described below:

```

<Splat>
  <delay_>         float  </delay_>
  <minimumAge_>    float  </minimumAge_>
  <name_>           string </name_>
</Splat>

```

Grammar of **SplatComponent** in `<particle>.xml+`

38.7.2. Particle renderers

The sub-sections below describe the 5 possible vector generator for position and velocity:

- **AmpParticleRenderer** — See “AmpParticleRenderer” on page 168 .
- **BlurParticleRenderer** — See “BlurParticleRenderer” on page 169 .
- **SpriteParticleRenderer** — See “SpriteParticleRenderer” on page 169 .
- **MeshParticleRenderer** — See “MeshParticleRenderer” on page 169 .
- **TrailParticleRenderer** — See “TrailParticleRenderer” on page 169 .
- **VisualParticleRenderer** — See “VisualParticleRenderer” on page 170 .

38.7.2.1. AmpParticleRenderer

The grammar for the **AmpParticleRenderer** is described below:

```

<AmpParticleRenderer>
  <viewDependent_> [true|false] </viewDependent_>
  <local_>          [true|false] </local_>
  <textureName_>    folder/file  </textureName_>

```

```

<width_>          float      </width_>
<height_>         float      </height_>
<steps_>          integer     </steps_>
<variation_>      float      </variation_>
<circular_>       [true|false] </circular_>
</AmpParticleRenderer>

```

Grammar of **AmpParticleRenderer** in `<particle>.xml`

38.7.2.2. BlurParticleRenderer

The grammar for the **BlurParticleRenderer** is described below:

```

<BlurParticleRenderer>
  <viewDependent_> [true|false] </viewDependent_>
  <local_>          [true|false] </local_>
  <textureName_>    folder/file  </textureName_>
  <width_>          float        </width_>
  <time_>           float        </time_>
</BlurParticleRenderer>

```

Grammar of **BlurParticleRenderer** in `<particle>.xml`

38.7.2.3. MeshParticleRenderer

The grammar for the **MeshParticleRenderer** is described below:

```

<MeshParticleRenderer>
  <viewDependent_> [true|false] </viewDependent_>
  <local_>          [true|false] </local_>
  <visualName_>     folder/file  </visualName_>
  <doubleSided_>    [true|false] </doubleSided_>
  <materialFX_>     integer       </materialFX_>
  <sortType_>       integer       </sortType_>
</MeshParticleRenderer>

```

Grammar of **MeshParticleRenderer** in `<particle>.xml`

38.7.2.4. SpriteParticleRenderer

The grammar for the **SpriteParticleRenderer** is described below:

```

<SpriteParticleRenderer>
  <viewDependent_> [true|false] </viewDependent_>
  <local_>          [true|false] </local_>
  <textureName_>    folder/file  </textureName_>
  <materialFX_>     integer       </materialFX_>
  <frameCount_>     integer       </frameCount_>
  <frameRate_>      float        </frameRate_>
</SpriteParticleRenderer>

```

Grammar of **SpriteParticleRenderer** in `<particle>.xml`

38.7.2.5. TrailParticleRenderer

The grammar for the **TrailParticleRenderer** is described below:

```

<TrailParticleRenderer>
  <viewDependent_> [true|false] </viewDependent_>
  <local_> [true|false] </local_>
  <textureName_> folder/file </textureName_>
  <width_> float </width_>
  <skip_> integer </skip_>
  <steps_> integer </steps_>
</TrailParticleRenderer>

```

Grammar of **TrailParticleRenderer** in *<particle>.xml*

38.7.2.6. VisualParticleRenderer

The grammar for the **VisualParticleRenderer** is described below:

```

<VisualParticleRenderer> <viewDependent_> [true|false] </viewDependent_>
  <local_> [true|false] </local_>
  <visualName_> folder/file </visualName_>
</VisualParticleRenderer>

```

Grammar of **VisualParticleRenderer** in *<particle>.xml*

38.7.3. Position/velocity vector generators

The sub-sections below describe the 5 possible vector generator for position and velocity:

- **BoxVectorGenerator** — See “BoxVectorGenerator” on page 170 .
- **CylinderVectorGenerator** — See “CylinderVectorGenerator” on page 170 .
- **LineVectorGenerator** — See “LineVectorGenerator” on page 171 .
- **PointVectorGenerator** — See “PointVectorGenerator” on page 171 .
- **SphereVectorGenerator** — See “SphereVectorGenerator” on page 171 .

38.7.3.1. BoxVectorGenerator

The grammar for the **BoxVectorGenerator** is described below:

```

<BoxVectorGenerator>
  <nameID_> BoxVectorGenerator </nameID_>
  <corner_> float float float </corner_>
  <opposite_> float float float </opposite_>
</BoxVectorGenerator>

```

Grammar of **BoxVectorGenerator** in *<particle>.xml*

38.7.3.2. CylinderVectorGenerator

The grammar for the **CylinderVectorGenerator** is described below:

```

<CylinderVectorGenerator>
  <nameID_> CylinderVectorGenerator </nameID_>
  <origin_> float float float </origin_>

```

```

<direction_> float float float      </direction_>
<maxRadius_> float                  </maxRadius_>
<minRadius_> float                  </minRadius_>
<basisU_>    float float float      </basisU_>
<basisV_>    float float float      </basisV_>
</CylinderVectorGenerator>

```

Grammar of **CylinderVectorGenerator** in `<particle>.xml`

38.7.3.3. LineVectorGenerator

The grammar for the **LineVectorGenerator** is described below:

```

<LineVectorGenerator>
  <nameID_>      LineVectorGenerator </nameID_>
  <origin_>      float float float   </origin_>
  <direction_>   float float float   </direction_>
</LineVectorGenerator>

```

Grammar of **LineVectorGenerator** in `<particle>.xml`

38.7.3.4. PointVectorGenerator

The grammar for the **PointVectorGenerator** is described below:

```

<PointVectorGenerator>
  <nameID_>      PointVectorGenerator </nameID_>
  <position_>    float float float    </position_>
</PointVectorGenerator>

```

Grammar of **PointVectorGenerator** in `<particle>.xml`

38.7.3.5. SphereVectorGenerator

The grammar for the **SphereVectorGenerator** is described below:

```

<SphereVectorGenerator>
  <nameID_>      SphereVectorGenerator </nameID_>
  <centre_>      float float float    </centre_>
  <maxRadius_>   float                 </maxRadius_>
  <minRadius_>   float                 </minRadius_>
</SphereVectorGenerator>

```

Grammar of **SphereVectorGenerator** in `<particle>.xml`

38.8. <sky>.xml

Sky configuration files are used by spaces, and can be located in any folder of your choice, as their location is set by the `timeOfDay` tag in `<res>/spaces/<space>/space.settings` (for details on this file's grammar, see `space.settings` on page 135).

Mentioned in the documentClient Programming Guide's sections *Terrain* → *Cloud shadows* → “Tweaking” and *3D Engine (Moo)* → “Features” → “Lighting” → “Light maps” → “Sky light map”.

The grammar is listed below:

```

<root>

  <texture>          file      </texture>
  <mieAmount>         float     </mieAmount>
  <turbidityOffset>   float     </turbidityOffset>
  <turbidityFactor>   float     </turbidityFactor>
  <vertexHeightEffect> float    </vertexHeightEffect>
  <sunHeightEffect>   float     </sunHeightEffect>
  <power>             float     </power>
  <farPlane>          integer   </farPlane>

  ?LightMapSection 1

  <day_night_cycle>
    <angle>           float     </angle>
    <moonAngle>       float     </moonAngle>
    <hourLength>       float     </hourLength>
    <startTime>        float     </startTime>

    +<lightKey>
      <time>           float     </time>
      <colour>         float float float </colour>
    </lightKey>

    +<ambientKey>
      <time>           float     </time>
      <colour>         float float float </colour>
    </ambientKey>

</root>

```

Grammar of sky configuration file

1 See *LightMapSection* on page 55 .

The list below describes the tags in the sky configuration file:

- **ambientKey**

Section specifying colour that should be used to evenly light all models at the specified time.

The ambient lighting is an approximation of global bounce lighting, and hence comes from all directions (is directionless)

- **angle**

Angle of the sun (in degrees).

It determines the sun's path across de sky.

- **colour**

Base colour that the respective light source (sun or ambient) should produce.

- **day_night_cycle**

Section containing settings that configure the lighting at various times of the day, as well as how quickly hours pass.

- **farPlane**

Camera far plane for all spaces using this sky definition file.

This will be overridden on a per-space basis if there is a `farplane` entry in the `space.settings` file (for details, see *space.settings* on page 135).

- **hourLength**

Number of real seconds that a game hour will last.

- **lightKey**

Section specifying colour that the sun should produce at the specified time.

- **LightMapSection**

For details, see *LightMapSection* on page 55 .

- **mieAmount^A**

Amount of mie (forward) scattering in the sky. Set this to larger values to increase the amount of light that is added to the sky when looking in the direction of the sun.

- **moonAngle**

Angle of the moon (in degrees).

It determines the moon's path across the sky.

- **power**

Exponent for the mie (forward) scattering in the sky.

Set this to a higher value to create a more focused corona around the sun.

- **sunHeightEffect^A**

Unit-less value that adds mie (forward) scattering as the sun rises in the sky.

- **texture**

File containing texture, which displays the colour gradient of the sky over time.

Vertical line displays the sky gradient at a particular time.

Horizontal line displays a point in the sky along the day.

- **time**

Time (in 24-hour format) at which the respective light should be used.

- **turbidityFactor^A**

Overall multiplier for mie (forward) scattering in the sky.

Set this to a higher value to represent more particulate matter in the sky (and thus see more forward scattering of light in the direction of the sun).

- **turbidityOffset^A**

Overall offset for mie (forward) scattering in the sky.

▪ **vertexHeightEffect**^A

Unit-less value that adds mie (forward) scattering depending on the height of the vertex in the sky. Higher values produce more scattering on the horizon.

A — Advanced setting. Should only be modified by advanced users.

38.9. <fx>.xml

FX files are used by the python FX module, and can be located in any folder of your choice. FX files are based around Actors. For each actor, you should specify one Joint. And for each actor, you can specify any number of Events. Each Actor, Joint or Event section requires the name of the actor to be specified as its section name as a string.

The grammar is listed below:

```
<file_name>
  +<Actor> string
    { Light | 1
      Model | 2
      ParticleSystem } 3
  </Actor>

  +<Joint> string
    { DummyModel | 4
      Entity | 5
      HardPoint | 6
      ModelRoot | 7
      Node } 8
  </Joint>

  *<Event> string
    { AddDecal | 9
      AlignModel | 10
      ClearParticles | 11
      CorrectMotionTriggeredParticles | 12
      Fade | 13
      Flicker | 14
      FlickeringLight | 15
      ForceParticle | 16
      PlayAction | 17
      PlaySound | 18
      RampTimeTriggeredParticles | 19
      RandomDelay | 20
      ResetTimeTriggeredParticles | 21
      SetBasis | 22
      SetColour | 23
      SetOrbitorPoint | 24
      SwarmTargets } 25
  </Event>
</file_name>
```

Grammar of FX file

The list below describes the tags in the FX file:

- ¹ See “Light Section” on page 175 .
- ² See “Model Section” on page 175 .
- ³ See “ParticleSystem Section” on page 175 .

- 4 See "DummyModel Section" on page 176 .
- 5 See "Entity Section" on page 176 .
- 6 See "HardPoint Section" on page 176 .
- 7 See "ModelRoot Section" on page 176 .
- 8 See "Node Section" on page 176 .
- 9 See "AddDecal Section" on page 177 .
- 10 See "AlignModel Section" on page 177 .
- 11 See "ClearParticles Section" on page 177 .
- 12 See "CorrectMotionTriggeredParticles Section" on page 177 .
- 13 See "Fade Section" on page 178 .
- 14 See "Flicker Section" on page 178 .
- 15 See "FlickeringLight Section" on page 178 .
- 16 See "ForceParticle Section" on page 179 .
- 17 See "PlayAction Section" on page 179 .
- 19 See "RampTimeTriggeredParticles Section" on page 179 .
- 20 See "RandomDelay Section" on page 179 .
- 21 See "ResetTimeTriggeredParticles Section" on page 180 .
- 22 See "SetBasis Section" on page 180 .
- 23 See "SetColour Section" on page 180 .
- 24 See "SetOrbitorPoint Section" on page 181 .
- 25 See "SwarmTargets Section" on page 181 .

38.9.1. Light Section

The **Light Actor** creates a **PyChunkLight**. It allows specification of the following :

```
<Light>
  <innerRadius> float </innerRadius>
  <outerRadius> float </outerRadius>
  <colour> Vector4 </colour>
</Light>
```

- **innerRadius**

Inner Radius of the light, in metres.

- **outerRadius**

Outer Radius of the light, in metres.

- **colour**

Colour of the light source in RGBA form.

38.9.2. Model Section

The **Model Actor** creates a **PyModel**. It allows specification of the following :

```
<Model> string </Model>
```

- **Model**

Resource ID of the model to load.

38.9.3. ParticleSystem Section

The **ParticleSystem Actor** creates a **PyMetaParticleSystem**. It allows specification of the following :

```
<ParticleSystem> string </ParticleSystem>
```

- **Particle System**

Resource ID of the particle system to load.

38.9.4. DummyModel Section

The **DummyModel Joint** attaches the actor to a dummy model, which follows the player but is not attached to them. This is useful because even if the player changes model, or becomes invisible, effects attached to a DummyModel will still be visible. It allows specification of the following :

```
<DummyModel> </DummyModel>
```

38.9.5. Entity Section

The **Entity Joint** adds the actor as a secondary model of an entity. If the actor is a Model, it creates a Motor that makes it follow the entity. If the actor is a ParticleSystem, its position is set to where the source is at time of attachment. It allows specification of the following :

```
<Entity> </Entity>
```

38.9.6. HardPoint Section

The **HardPoint Joint** attaches the actor to a hardpoint. It uses the hardpoint feature, meaning that both the actor and the source must have the corresponding hard point. It allows specification of the following :

```
<HardPoint> string </HardPoint>
```

- **HardPoint**

Hard Point name, without the HP_ prefix.

38.9.7. ModelRoot Section

The **ModelRoot Joint** attaches the actor to the source model's root node. It allows specification of the following :

```
<ModelRoot> </ModelRoot>
```

38.9.8. Node Section

The **Node Actor** attaches the actor directly to the specified node on the source model. It allows specification of the following :

```
<Node> string </Node>
```

- **Node**

Node name.

38.9.9. AddDecal Section

The **AddDecal Event** adds a decal to the world, at the position of the source when the event is fired. It allows specification of the following :

```
<AddDecal> string
    <size> float </size>
    <extent> Vector3 </extent>
</AddDecal>
```

- **size**

width and depth of the decal, in metres.

- **extent**

extent of the collision ray. The actor's position is used as the midpoint for the collision ray when choosing where to put the decal in the world.

38.9.10. AlignModel Section

The **AlignModel Event** will position and align a model actor at runtime according to the basis tuple (dir,pos) passed into the Effect's variable arguments dictionary. It allows specification of the following :

```
<AlignModel> </AlignModel>
```

38.9.11. ClearParticles Section

The **ClearParticles Event** clears all particle containers, resetting them to their initial state. It allows specification of the following :

```
<ClearParticles>
    *<systemName> string </systemName>
</ClearParticles>
```

- **systemName**

Optional list of names of sub-systems within the MetaParticleSystem actor to apply the event to.

38.9.12. CorrectMotionTriggeredParticles Section

The **CorrectMotionTriggeredParticles Event** resets the motion trigger flag on all applicable sub-systems. This allows motion triggered particle systems to be re-used. It works around a problem caused by the motion trigger using its last known world position to calculate the distance the particle system has moved since the last frame. It allows specification of the following :

```
<CorrectMotionTriggeredParticles>
    *<systemName> string </systemName>
</CorrectMotionTriggeredParticles>
```

- **systemName**

Optional list of names of sub-systems within the MetaParticleSystem actor to apply the event to.

38.9.13. Fade Section

The **Fade Event** fades out a light actor over the specified time. It allows specification of the following :

```
<Fade>
  <time> float </time>
</Fade>
```

- **time**

Specifies the time over which the light will fade out.

38.9.14. Flicker Section

The **Flicker Event** flickers the colour of a light actor. It allows specification of the following :

```
<Flicker>
  <amplitude> float </amplitude>
  <speed> float </speed>
</Flicker>
```

- **amplitude**

Multiplies the amplitude of the noise that drives the flicker.

- **speed**

Multiplies the frequency of the noise that drives the flicker.

38.9.15. FlickeringLight Section

The **FlickeringLight Event** creates a canned flickering light effect at the position of the source's model's root node. It lasts for the duration of the effect, or the duration of a specified actor in the effect, fading out over the last 50 percent. It allows specification of the following :

```
<FlickeringLight>
  <innerRadius> float </innerRadius>
  <outerRadius> float </outerRadius>
  <colour> Vector4 </colour>
  ?<actorForTiming> string </actorForTiming>
</FlickeringLight>
```

- **innerRadius**

Inner Radius of the light, in metres.

- **outerRadius**

Outer Radius of the light, in metres.

- **colour**

Colour of the light source in RGBA form.

- **actorForTiming**

[Optional] Specify the name of another actor in the Effect to use as the duration of the FlickeringLight event. If not specified, the total duration of the rest of the effect will be used.

38.9.16. ForceParticle Section

The **ForceParticle Event** spawns a single unit of particles. The unit size is specified with the particle system, in Particle Editor. It allows specification of the following :

```
<ForceParticle>
  *<systemName> string </systemName>
</ForceParticle>
```

- **systemName**

Optional list of names of sub-systems within the MetaParticleSystem actor to apply the event to.

38.9.17. PlayAction Section

The **PlayAction Event** plays, in sequence, a list of Actions on a Model. It allows specification of the following :

```
<PlayAction>
  +<Action> string </Action>
</PlayAction>
```

- **Action**

List of action names to play sequentially on the actor.

38.9.18. RampTimeTriggeredParticles Section

The **RampTimeTriggeredParticles Event** ramps up/down the generation rate of time triggered particles, to fade them in/out. It also gives them a specified duration, which allows time triggered particles to behave correctly in a OneShot effect scenario. It allows specification of the following :

```
<RampTimeTriggeredParticles>
  <Duration> float </Duration>
  <FadeTime> float </FadeTime>
</RampTimeTriggeredParticles>
```

- **Duration**

Total duration of the event, i.e. how long the particles will be visible for.

- **FadeTime**

Time in seconds, measured from the end of the event, when the particles should be turned off.

38.9.19. RandomDelay Section

The **RandomDelay Event** waits for a random amount of time, then spawns a list of events at once. It allows specification of the following :

```
<RandomDelay>
  <MinDelay> float </MinDelay>
```

```

    <MaxDelay> float </MaxDelay>
    +<Event> EventSection </Event>
</RandomDelay>

```

- **MinDelay**

Minimum time in seconds for the random delay.

- **MaxDelay**

Maximum time in seconds for the random delay.

- **Event**

List of Events to spawn at once, after the delay has been observed.

38.9.20. ResetTimeTriggeredParticles Section

The **ResetTimeTriggeredParticles Event** resets the time trigger of a particle system actor. This is useful for effects that are reused. It means that when reused, the particles will begin "from the beginning", instead of sometime through their normal cycle. It allows specification of the following :

```

<ResetTimeTriggeredParticles>
    *<systemName> string </systemName>
</ResetTimeTriggeredParticles>

```

- **systemName**

Optional list of names of sub-systems within the MetaParticleSystem actor to apply the event to.

38.9.21. SetBasis Section

The **SetBasis Event** positions and aligns a particle system in the world, based on the basis tuple (dir,pos) passed in at run-time to the effect's variable arguments dictionary. It allows specification of the following :

```

<SetBasis>
    *<systemName> string </systemName>
</SetBasis>

```

- **systemName**

Optional list of names of sub-systems within the MetaParticleSystem actor to apply the event to.

38.9.22. SetColour Section

The **SetColour Event** sets a colour modulator on the TintShader action of a particle system actor, using a Vector4Provider passed in at run-time to the effect's variable arguments dictionary. It allows specification of the following :

```

<SetColour>
    *<systemName> string </systemName>
</SetColour>

```

- **systemName**

Optional list of names of sub-systems within the MetaParticleSystem actor to apply the event to.

38.9.23. SetOrbitorPoint Section

The **SetOrbitorPoint Event** sets the world position of the orbitor location for Particle System actors using the Orbitor action, using the location of the effect source when the event is initiated. It allows specification of the following :

```
<SetOrbitorPoint>
  *<systemName> string </systemName>
</SetOrbitorPoint>
```

- **systemName**

Optional list of names of sub-systems within the MetaParticleSystem actor to apply the event to.

38.9.24. SwarmTargets Section

The **SwarmTargets Event** sets a list of target nodes on a particle system containing the MatrixSwarm action. It allows specification of the following :

```
<SwarmTargets>
  +<Node> string </Node>
  *<systemName> string </systemName>
</SwarmTargets>
```

- **Node**

One or more node names on the target model to be set on the MatrixSwarm particle system actions as their target.

systemName

Optional list of names of sub-systems within the MetaParticleSystem actor to apply the event to.

38.10. <weather>.xml

The weather XML file specifies a list of weather systems available for an entire game. The number of different weather systems is up to you. Since the file is simply a list of systems, and each tag name is the user's name for the weather section, there's no point documenting the grammar of the overall file. Instead we will look at what goes within each weather system section.

The grammar is listed below:

```
<weather.xml>
  *< WeatherSystemName >
    +<skyBox> string </skyBox>
    ?<sfx> string </sfx>
    ?<rain> float </rain>
    ?<sun> Vector4 </sun>
    ?<ambient> Vector4 </ambient>
    ?<fog> Vector4 </fog>
    ?<windSpeed> Vector2 </windSpeed>
    ?<windGustiness> float </windGustiness>
    ?<bloom>
      <attenuation> Vector4 </attenuation>
      <numPasses> int </numPasses>
      <power> float </power>
      <width> int </width>
```



```

        </bloom>
    </weather.xml>

```

The list below describes the tags in the Weather System section:

- **skyBox**

resource ID of a .model file. The model file should use one of the environment .fx file in order to draw and sort correctly in the scene. You may specify any number of sky box models. These will be drawn in the order they are specified in the XML file.

- **sfx**

resource ID of a Special Effect file. This will be attached to the player when the weather system is summoned.

- **rain**

Amount of rain, from 0.0 to 1.0.

- **sun**

Colour that multiplies with the dynamic time-of-day colour of the sunlight. In RGBA form.

- **ambient**

Colour that multiplies with the dynamic time-of-day ambient colour. In RGBA form.

- **fog**

Colour that multiplies with the dynamic time-of-day colours of fog. The fourth value is the fog density, where 1.0 is the normal amount of fog, and any higher value is thicker fog. The colours are in RGB form.

- **windSpeed**

Wind velocity, as x/z metres per second in world-space.

- **windGustiness**

Wind gustiness. The gustiness randomly adjusts the wind velocity over time. A value of 0 means the wind will always be exactly the windSpeed.

- **bloom**

Section describing bloom settings.

- **attenuation**

Per-pass colour attenuation of the bloom filter. In normalised RGBA form; e.g (1,1,1,1) means no colour attenuation.

- **numPasses**

Number of gaussian blur passes the bloom filter will perform.

- **power**

Mathematical power of the scaling function, for shader 2.0 cards and above.

- **width**

Texel width multiplier of the bloom filter kernel. Larger values produce a larger bloom area, but beware of creating gaps or holes in the effect.