toxic.3d version 1.0

Interactive 3d engine for Flash 4/5

Documentation release: 1.0

By Karsten Schmidt (toxi@toxi.co.uk)
Original code based on scripts by Thomas Noller (thomas@method.com)

This version of the engine is open source and freely available from www.toxi.co.uk

Note: If you want to use this engine for commercial purposes please obtain a license from the above web site. If you're interested in helping to develop this project further, please contact me before you start coding. This will simplify coordination and incorporation of new features.

Please send bug reports and comments to flash3d@toxi.co.uk

Last update: 26/11/00 10:18 PM

1 toxic.3d engine description

This documentation gives a brief overview of all current features and properties. It's purpose is mainly for reference only – unfortunately it is not a tutorial for building your own applications, you'll have to get your head around it yourself… If you're unsure about how to realise certain functions, please refer to the demo file.

In the current version, all movie clips with object definitions have to be placed in the 3d.engine's timeline. Object instances have to be named "object1"…"objectn" . Each of these objects requires another clip to label its points. These instances have to be named "items1"…"itemsn". These "item" clips also contain the buttons used for navigational purposes.

1.1 public methods

start3D (ObjectID)

Calling this funtion initialises the engine with the object number objectID.
See frame "start" in main timeline. Properties used:

`maxz`	- maximum Z coordinate before clipping occurs.
`d`	- viewing distance
`zoom_max`	- maximum zoom factor
`z_in_factor`	- zooming in speed
`z_out_factor`	- zooming out speed
`zoom_active`	- zooming status (true/false)

StartMorph ()

Begin the morphing process.
(See frame "links" in the clip "toxic.items" of the demo)

ZoomInNow ()

Possible rollover action which makes the object zooming in.
To zoom out again set the properties `zoom_active=true` and `zoom_speed=z_out_factor`.

1.2 private methods

These methods are used by the engine's public methods and shouldn't be used "outside" the engine by your own code.

`setActiveObject`

`initMorph`

`morphPoints`

`initPointAnimation`

`animatePoints`

`getQuadrant`

2 3d Object definition

toxic.3d objects are stored individually in movie clips which only contain a set of action script variables to define the properties of objects. To make such an object available to the toxic.3d engine it must be placed and correctly named within the toxic.3d movie clip. For easier identification place each object on an individual layer.

Naming convention for the object instance: clip "object" & objectID, ie. "object3"

2.1 Object properties

2.1.1 Corner points definition of an object

Each object is defined by a set of x; y ;z coordinates with an origin in (0;0;0)

`-point_count`

Total number of points in one 3d object.

`-xn, yn, zn`

3d coordinates of each point

2.1.2 Vertices definition

Due the way corner points are defined, vertice drawing in this version is not based on actual 3d planes of the object. Instead they are organised in a number of bundles consisting of:

`-v_base_n`

Base point ID of bundle n. All vertices in this bundle are using this point as starting point.

`-v_count_n`

Number of vertices in bundle `n`.

`-v_linkn_m`

Point ID of the current vertice `m` in bundle n.
ie. `v_link2_8`=5 means the 8th vertice in bundle 2 will be drawn from `v_base_2` to point 5.

`-v_style_n`

drawing style for vertices of bundle n. Pre-defined styles are:

0 – solid white hairline
1 – red dots near corner points
2 – white dotted line with interruption near centre
3 – turquoise dots near corner points
4 – white dotted 90 degrees arc

`-v_count`

Total number of bundles of vertices in an object.

2.1.3 Object Animation Setup

The engine allows objects to be self animated, morphing to another shape and back. The number of points in this new shape has to stay the same (Though you can simulate objects with less(!) points by merging several points into the same coordinates)

Plaese note: *There's only one definition of vertices, so be careful that the alternate shape will still comply with this setup!*

`-is_animated`

True or False property. If set True the following paremeters have to be defined too:

`-ani_time`

Number of frames it takes to interpolate coordinates between the two shapes (key frames) of this object. When one key frame is reached, the direction of the interpolation will automatically invert. Animation is always starting from the original point definition.

`-axn, ayn, azn`

3d coordinates of each point of the alternative shape. Here you only have to define the points where coordinates differ from the original shape.

2.1.4 Object morphing setup

-`morph_target`

ID of target object

-`m1, m2 ... mn`

morph points in target object
i.e. `m1=7` means point 1 in current object will move to the coordinates of point 7 in target object. this has to be specified for all points in the current object! If the target object has more points than current object, variables `mstartn` have to be defined.

`-mstartn`

This has to be defined for all missing points (compared with the target object) in the current object. The value assigned to `mstart` is the ID number of another defined point in the object. Therefore in `mstartn=x : n>point_count; x<=point_count` of the current object. Specifying `mstart` is essential to make the morphing process look better. Points will disappear during the morphing, if not specified. I.e. the current object has 8, the target object has 12 points. Mstart has to be specified for points 9 – 12. Mstart9=2 would create an additional 9th point at the coordinates of point 2.

Note: For all points where `mstart` is defined, calling `startMorph` will automatically create morph target points (`mn`) with the same index. I.e. For `mstart9` a `m9=9` will be created by `startMorph`.