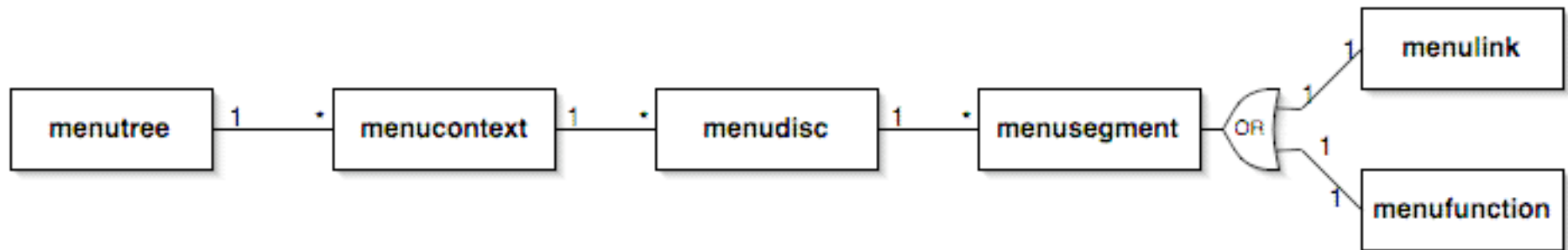


## Circular Menus.

### Structure.

The menu system looks after one or more menu trees.

Each menu tree contains one or more contexts and each context one or more menu discs. Each disc contains a number of segments, each segment has either a function attached or a menu disc name which is used to activate (show) another menu disc. Currently a menu disc can have 2, 4, 6 or 8 segments. Other segment amounts are possible but have not been tested.



### Parameters.

Each menu element has a name, these names need to be unique within each menu element type: it is not a problem to have the same name for a context as well as a tree as well as a disc. Do not have the same name for multiple discs or contexts within a single tree.

Unless specified each element parameter is required.

## **<menutree>**

Root node of the menu tree.

**name**     Name of the menu tree.

## **<menustandardsizes>**

Sets 2 floating point sizes, one for the inner hole of the disc, and one for the outer radius of the disc. These sizes are the default values for menu discs, they are used if a menu disc has no sizes of it's own specified. This element must be specified as the first element in the menu tree.

**inner**     Floating point value for radius of inner hole.

**outer**     Floating point value for radius of outer edge.

## **<menucontext>**

The outer container of a set of menu discs. A menu tree can have multiple contexts.

**name**     Name of the context, needs to be unique within a tree.

## **<menudisc>**

Contains all information needed to create a menu disc. If the inner and outer sizes are not specified the disc will be created with the sizes specified in **<menustandard sizes>**, this allows the creation of differently sized discs within single tree. All other parameters are required.

**inner** Floating point value for radius of inner hole.

**outer** Floating point value for radius of outer edge.

**name** Name of the disc, needs to be unique within disc.

**segments** Number of segments for this disc, currently only 2, 4, 6 and 8 have been tested to work.

**texture** A file and path to a texture or, with minor modification, a texture name reference.

## **<menusegment>**

Contains the data for a segment within a disc. Segments are not named.

**off** Indicates if this segment starts turned on or off.

**infotext** Simple description of the segment's function.

## **<menulink>**

Contains a link for the segment. The link is a menu disc name inside the current context.

**name** Name of a valid menu disc.

## **<menufunction>**

Segment function that calls a command. This could potentially also hold a language identifier to allow different script languages, a different parser could be used for each.

**command**     A script function that is passed along to a script command parser for execution.

## **Menu System**

Each of the XML elements translates to a C++ class. There is one class that is not set-up using XML and that is the "Menu System" class. CMenuSystem (refer to Doxygen created HTML) is a controlling class for the menus. Using CMenuSystem trees are registered, contexts and trees are activated and deactivated, tree searching methods using the names given to the objects.

There is also a "check for quit" function that can be called by the owner of the menu system, it's a hack used to stop the menu system from requiring "links" into the system in which the menus are used.

I short: it's the central point of contact for the menu system.