MaXML 2.17: Getting Started

What is MaXML 2?

MaXML is a module created for BlitzMax, which provides you with the tools to manage, load, and save your application or game's data in the XML format. XML is an extremely versatile and widely accepted file format, and is used anywhere from word-processors to level designers.

One of the main disadvantages of using an XML parser in your project is the speed limitation; because of the need for your computer to read and translate the XML data into a format usable by your program, loading XML files with most parses is generally slow. MaXML 2 eliminates this problem by providing you with an extremely fast XML parser, which is carefully designed and streamlined to provide optimal performance. MaXML 2's compiler is capable of parsing most files at over 4.5 MB per second! (although this will vary depending on your processor and hard-drive speed).

Installing MaXML

Installing MaXML is almost no trouble at all. Simply place the folder named "maxml.mod" into the "mod/pub.mod/" folder under your BlitzMax installation. Next, run "docmods" (in the "bin" folder under your BlitzMax installation), which integrates MaXML's documentation with BlitzMax.

To ensure that MaXML is properly installed, try running the included "Example1.bmx" and "Example2.bmx".

Note: If you are using MaXML on a Mac or Linux system, you may need to rebuild mods.

MaXML Help

MaXML includes documentation which describes every feature it provides. If you are not familiar with XML terms, the documentation may at first seem confusing. If so, this section provides a simple description of the basics of XML.

An XML data structure is basically a collection of data items, called ***nodes***. Each node has a name, a value, and attributes (which also may have their own names and values). A typical XML node may look like this:

```
<house color="white" stories="2">Hotel</house>
```

In this example, a node called "house" is defined in XML syntax (you don't need to worry about learning XML syntax, since MaXML's loader/saver takes care of that for you). The **name** of the node is "house", and it's **value** is "Hotel". It contains two ***attributes***: color (white), and stories (2).

An XML file may contain many nodes, storing any amount of information you want. However, unlike many files (such as INI files), XML allows you to make categories, sub-categories, etc. to more easily manage information. For data categorization, XML allows nodes to "contain" other nodes. For example:

```
<house color="while" stories="2">
	<chimney color="red"/>
	<window location="front"/>
	<door type="normal"/>
</house>
```

The example above has 4 nodes: "house", "chimney", "window", and "door". The "chimney", "window", and "door" nodes are contained within the "house" node. A node that contains other nodes is called a ***parent*** node, and the nodes contained within it are called ***children***, almost as if an XML

structure was being used to store a family tree (and it can, if you want it to). For example, “chimney” is a child of “house”, and “house” is the parent node of “chimney”, “window”, and “door”. Two nodes that share the same parent are called ***siblings***. For example, the “chimney” and “door” nodes are siblings.

Even more detail can be added to the example above by adding a child node to the “door” node:

```
<house color=”while” stories=”2”>
      <chimney color=”red”/>
      <window size=”large” location=”front”/>
      <door type=”normal”>
            <doorknob color=”yellow”/>
            <window size=”small” location=”center”/>
      </door>
</house>
```

Remember: *any* node may contain a number of children. By adding child nodes with extra information about an object, you can provide any level of informational detail you want.

To get started using MaXML, try running the examples that are included. The examples demonstrate the basic use of MaXML to load/manipulate/save XML data structures.