

# ZScripts Framework Server Scripts MSI/API Documentation

July 3, 2013

**NOTE: This document only documents the MSI (Module Scripting Interface), not end modules (e.g., `ban_command`) that aren't intended to be required by other modules.**

## Part I

### CORE (main)

**loadModule(modname)** Takes modname, loads the module.

**unloadModule(modname)** Unloads modname.

**reloadModule(modname)** Reloads modname

**registerHandler(handlername, object, propname=handlername)** Registers object[propname] as handlername in the script events.

**log(message)** Logs the message. Note: Overridden by logs.js.

**broadcast(m)** Deprecated.

**modules:** Contains all the modules loaded.

## Part II

### text

**escapeHTML(text)** Escapes html in text

**stripHTML(text)** Converts html to plaintext (works so-so.)

## Part III

### util

**bind(object, function)** Binds function to object, returns the bind.

**shuffle(array)** Shuffles array in place.

**inspect(variant)** Returns a human readable representation of variant. Plaintext and '\n' separated.

## Part IV

### time

**conversionFactors[factor]** Object key is e.g., "day", output is number of milliseconds per. Some are approximations (e.g. month)

**diffToStr(dif)** Converts number of milliseconds into a string length of time (e.g., 62000 -> "1 hour, 2 seconds" or something.

**strToDiff(str)** Converts diff string to number of miliseconds. May or may not work depending on the input formatting.

## Part V

# theme

**scriptHTML** Internal

**scriptText** Internal

**warnHTML** Internal

**gameHTML** Internal

**INFO** Constant for formatAs (default)

**WARN** Constant for formatAs

**CRITICAL** Constant for formatAs

**GAME** Constant for formatAs

**TOUR** Constant for formatAs

**formatAs(text, const)** format text as const, -1 for raw

## Part VI

# com

**message(users, msg, type, html, chans, servercode)** Sends msg to all users formatted as type (see theme, use constant or -1), in any of chans (unless chan is null, then all chans.) If servercode is truthy then append it to the logs prefixed with servercode.

**broadcast(msg, type, html, chans)** Like message but for all users.

## Part VII

# io

**openDB(dbname)** Opens database dbname, returns dbname. Must not already be open.

**closeDB(dbname)** Closes DB dbname.

**flushDB(dbname)** Writes out DB to file (semi-slow with large databases [> ~100,000 keypair])

**commitDB(dbname)** Commits changes to DB to file (/usually/ faster)

**backupDB(dbname)** Makes a backup of DB (semi-slow with big databases [> ~100,000 keypair])

**purgeDB(dbname)** Unimplemented.

**markDB(dbname)** Mark dbname as changed. May or may not do anything. Might break your code if you only use it sometimes but not never or always.

**readConfig(cfgname, defaultConfig)** Read config.

**writeConfig(cfgname)** Write config (DEPRECATED)

## Part VIII

# logs

**DEBUG** Debug constant

**INFO** Information constant

**WARN** Warning constant

**ERROR** Error constant

**CRITICAL** Critical constant

**logMessage(level, msg)** Logs the message.

## Part IX

# sched

**at(time, function)** Runs function at time. (will be deprecated in future but not yet, haven't coded alternative yet)

## Part X

# profile

**database** Object that keeps profiles. This is an I/O database opened with the I/O module.

**relationaldatabase** Object discarded on reload that holds relational database information.

**profileID(src)** Gets profile id for src.

**lastName(profid)** Last name of profile ID.

**profileNames(prof)** Names of prof.

**profileIPs(prof)**

**profileUpdateInfo(prof, src)** Updates profile information for src.

**registerPlayer(src)** Registers src a profile.

... Complete docs later, this module is not very important and big.

## Part XI

# security

**checkUsers()**

**checkUser(src)** kicks banned users.

**profsMuted(prof)** Boolean if profile is muted.

**profsBanned(prof)** Boolean if profile is banned.

**getMute(prof)** Gets mute object for profile.

**setMute(prof, muteObj)** Sets mute object on profile.

**getBan(prof)** Gets ban object for profile.

**setBan(prof, banObj)** Sets ban object on profile.

**checkMute(prof)** Checks users mute for expiry

**checkBan(prof)** Checks ban for expiry

**removeMute(prof)** Removes mute on prof

**removeBan(prof)** Removes ban on prof

## Part XII

# commands

**registerCommand(name, obj, prop=name)** Registers name from obj[prop]. Obj must be a module.

**unregisterCommand(name)** Usually you don't need to use this, the command is remove at the unload event.

**issueCommand(...)** Don't use.

**Command Objects are of the following form:**

**perm(src, cmd, chan)** A permission function.

**code(src, cmd, chan)** Executes the command.

**desc** Describes the command

**options** Object with key descriptions of the command's options.

## Part XIII

# chat

**registerCapture(src, func, module)** Registers the capture for the module to user src.

**registerFilter(filter, module)** Registers the filter for the module.

## Part XIV

# less

**less(src, msg, html)** Sends msg to src. html says if it's an HTML message. User is presented with a special interface to read a very long message.