NAME

qcvm — A standalone QuakeC VM binary executor

SYNOPSIS

qcvm [**options**] [**parameters**] *program-file*

DESCRIPTION

qcvm is an executor for QuakeC VM binary files created using a QC compiler such as gmqcc(1) or fteqcc. It provides a small set of builtin functions, and by default executes the **main**() function if there is one. Some options useful for debugging are available as well.

OPTIONS

There are 2 types of options. Options for the executor, and parameter options used to add parameters which are passed to the main function on execution.

-h, **--help**
: Show a usage message and exit.

-trace
: Trace the execution. Each instruction will be printed to stdout before executing it.

-profile
: Perform some profiling. This is currently not really implemented, the option is available nonetheless.

-info
: Print information from the program's header instead of executing.

-disasm
: Disassemble the program by function instead of executing.

-disasm-func *function*
: Search for and disassemble the given function.

-printdefs
: List all entries from the program's defs-section. Effectively listing all the global variables of the program. This option disables execution.

-printfields
: List all entries from the program's fields-section. Listing all entity-fields declared in the program. This option disables execution.

-printfuns
: List functions and some information about their parameters. This option disables execution. With a verbosity level of 1, builtin numbers are printed. With a verbosity of 2, the function's sizes are printed as well. This takes a little longer since the size is found by searching for a `DONE` instruction in the code.

-v Increase verbosity level, can be used multiple times.

-vector *'x y z'*
: Append a vector parameter to be passed to **main**().

-float *number*
: Append a float parameter to be passed to **main**().

-string *'text'*
Append a string parameter to be passed to **main**().

BUILTINS

The following builtin functions are available:

1) `void print(string...) = #1;`
Print the passed strings to stdout. At most 8 strings are allowed.

2) `string ftos(float) = #2;`
Convert a float to a string.

3) `entity spawn() = #3;`
Spawn an entity.

4) `void remove(entity) = #4;`
Remove an entity.

5) `string vtos(vector) = #5;`
Convert a vector to a string.

6) `void error(string...) = #6;`
Print strings to stdout and then exit with an error (limited to 8 arguments)

7) `float vlen(vector) = #7;`
Get the length of a vector.

8) `string etos(entity) = #8;`
Get the entity ID as string.

9) `float stof(string) = #9;`
Convert a string to a float.

10) `string strcat(string, string) = #10;`
Concatenate two strings, returning a tempstring.

11) `float strcmp(string, string) = #11;`
12) `float strncmp(string, string, float) = #11;`
Compare two strings. Returns the same as the corresponding C functions.

12) `vector normalize(vector) = #12;`
Normalize a vector so its length is 1.

13) `float sqrt(float) = #13;`
Get a value's square root.

SEE ALSO

`gmqcc`(1)

AUTHOR

See <http://graphitemaster.github.com/gmqcc>.

BUGS

Please report bugs on <http://github.com/graphitemaster/gmqcc/issues>, or see <http://graphitemaster.github.com/gmqcc> on how to contact us.