NAME

runtests.pl − run one or more test cases

SYNOPSIS

runtests.pl [options] [test number] [!test number] [key word] [!key word]

DESCRIPTION

runtests.pl runs one, several or all the existing test cases in curl's test suite. It is often called from the root Makefile of the curl package with 'make test'.

TEST NUMBER

If no test case number is given, all existing tests that the script can find will be considered for running. You can specify single test cases to run, space-separated, like "1 3 5 7 11", and you can specify a range like "45 to 67". You can also specify only the tests you don't want to run by listing the numbers with a leading exclamation point, like "!66".

It is also possible to specify tests to skip based on a key word describing the test. These are specified with a leading exclamation point and the key word or phrase, like "!HTTP NTLM auth". Likewise, tests to run can be specified simply by specifying the unadorned key words, like "FTPS". Remember that the exclamation marks and spaces will need to be quoted somehow when entered at many command shells.

OPTIONS

-a Continue running the rest of the test cases even if one test fails. By default, the test script stops as soon as an error is detected.

-bN Use N as the base TCP/UDP port number on which to start the test servers.

-c <curl>
Provide a path to a custom curl binary to run the tests with. Default is the curl executable in the build tree.

-d Enable protocol debug: have the servers display protocol output.

-e Run the test event-based (if possible). This will make runtests invoke curl with --test-event option. This option only works if both curl and libcurl were built debug-enabled.

-g Run the given test(s) with gdb. This is best used on a single test case and curl built --disable-shared. This then fires up gdb with command line set to run the specified test case. Simply (set a break-point and) type 'run' to start.

-h Displays a help text about this program's command line options.

-k Keep output and log files in log/ after a test run, even if no error was detected. Useful for debugging.

-l Lists all test case names.

-n Disable the check for and use of valgrind.

-p Prints out all files in "log/" to stdout when a test case fails. Very practical when used in the automated and distributed tests since then the people checking the failures and the reasons for them might not have physical access to the machine and logs.

-r Display run time statistics. (Requires Perl Time::HiRes module)

-rf Display full run time statistics. (Requires Perl Time::HiRes module)

-s Shorter output. Speaks less than default.

-t[num] Selects a **torture** test for the given tests. This makes runtests.pl first run the tests once and count the number of memory allocations made. It then reruns the test that number of times, each time forcing one of the allocations to fail until all allocs have been tested. By setting *num* you can force the allocation with that number to be set to fail at once instead of looping through everyone, which is very handy when debugging and then often in combination with *-g*.

-v Enable verbose output. Speaks more than default.

-vc <curl>

Provide a path to a custom curl binary to run when verifying that the servers running are indeed our test servers. Default is the curl executable in the build tree.

RUNNING TESTS

Many tests have conditions that must be met before the test case can run fine. They could depend on built-in features in libcurl or features present in the operating system or even in third-party libraries that curl may or may not use.

The test script checks most of these by itself to determine when it is safe to attempt to run each test. Those which cannot be run due to failed requirements will simply be skipped and listed at the completion of all test cases. In some unusual configurations, the test script cannot make the correct determination for all tests. In these cases, the problematic tests can be skipped using the "!keyword" skip feature documented earlier.

WRITING TESTS

The simplest way to write test cases is to start with a similar existing test, save it with a new number and then adjust it to fit. There’s an attempt to document the test case file format in the tests/FILEFORMAT.