www.phoronix-test-suite.com

Phoronix Test Suite v8.0.0 (Aremark)

User Manual

Getting Started

Overview

The Phoronix Test Suite is the most comprehensive testing and benchmarking platform available for Linux, Solaris, Mac OS X, and BSD operating systems. The Phoronix Test Suite allows for carrying out tests in a fully automated manner from test installation to execution and reporting. All tests are meant to be easily reproducible, easy-to-use, and support fully automated execution. The Phoronix Test Suite is open-source under the GNU GPLv3 license and is developed by Phoronix Media in cooperation with partners. Version 1.0 of the Phoronix Test Suite was publicly released in 2008.

The Phoronix Test Suite client itself is a test framework for providing seamless execution of test profiles and test suites. There are more than 200 tests available by default, which are transparently available via **OpenBenchmarking.org** integration. Of these default test profiles there is a range of sub-systems that can be tested and a range of hardware from mobile devices to desktops and worksrtations/servers. New tests can be easily introduced via the Phoronix Test Suite's extensible test architecture, with test profiles consisting of XML files and shell scripts. Test profiles can produce a quantitative result or other qualitative/abstract results like image quality comparisons and pass/fail. Using Phoronix Test Suite modules, other data can also be automatically collected at run-time such as the system power consumption, disk usage, and other software/hardware sensors. Test suites contain references to test profiles to execute as part of a set or can also reference other test suites. Test suites are defined via an XML schema.

Running the Phoronix Test Suite for the first time can be as simple as issuing a command such as *phoronix-test-suite benchmark c-ray*, which would proceed to install a simple CPU test, execute the test, and report the results. Along with the results, the system's hardware/software information is collected in a detailed manner, relevant system logs, and other important system attributes such as compiler flags and system state. Users can optionally upload their results to OpenBenchmarking.org for sharing results with others, comparing results against other systems, and to carry out further analysis.

OpenBenchmarking.org

OpenBenchmarking.org is an open, collaborative testing platform that makes the Phoronix Test Suite an even more extensible platform for conducting automated tests with complete integration into Phoronix Test Suite test client. OpenBenchmarking.org serves as a repository for storing test profiles, test suites, and result data. Test profiles and suites are stored in the OpenBenchmarking.org cloud to allow for new/updated tests to be seamlessly obtained via the Phoronix Test Suite without needing to manually update the Phoronix Test Suite client. OpenBenchmarking.org also makes it easy to facilitate side-by-side comparisons with any other results stored in the OpenBenchmarking.org cloud. Any Phoronix Test Suite user is permitted to upload their test results, test profiles, and suites to OpenBenchmarking.org.

When finding a set of results on OpenBenchmarking.org (e.g. **an example result file**), it's as easy as running the Phoronix Test Suite with that OpenBenchmarking.org ID to perform an automated side-by-side comparison (e.g. *phoronix-test-suite benchmark 1203160-BY-NVTEGRA3785*).

Thanks to the wealth of test data (results, system logs, etc) from crowd-sourced benchmarking via the

Phoronix Test Suite, a plethora of analytical features are also available from OpenBenchmarking.org.

Phoromatic

Phoromatic is a remote management system for the Phoronix Test Suite that allows the automatic scheduling of tests, remote installation of new tests, and the management of multiple test systems all through an intuitive, easy-to-use web interface. Tests can be scheduled to automatically run on a routine basis across multiple test systems. Phoromatic can also interface with revision control systems to offer support for issuing new tests on a context-basis, such as whenever a Git commit has been pushed or new daily image available. The test results are then available from this central, secure location.

Phoromatic is an add-on to the Phoronix Test Suite that's primarily intended for enterprise users when facilitating tests across a wide-spectrum of hardware within a test lab or when needing to carry out tests on a routine basis.

User Options

The following options are currently supported by the Phoronix Test Suite client. A list of available options can also be found by running *phoronix-test-suite help.*

Test Installation

force-install *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will force the installation (or re-installation) of a test or suite. The arguments and process is similar to the install option but even if the test is installed, the entire installation process will automatically be executed. This option is generally used when debugging a test installation problem.

install *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will install the selected test(s) inside the testing environment directory. The install process from downloading of the test files to the installation is fully automated. The install option needs to be supplied with the test name or suite as an argument. Optionally, a OpenBenchmarking.org ID or the name of a saved results file can be supplied as well and the test(s) to install will automatically be extracted from that information. If the test is already installed and was run by the latest version of the installation process, no action will be taken. Multiple arguments can be supplied to install additional tests at the same time.

install-dependencies *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will install the external dependencies needed by the selected test(s) using the distribution's package management system. For example, some tests depend upon GCC for compiling code. If GCC is not detected on the system, the Phoronix Test Suite will attempt to install GCC using the distribution's package management system. If you are running this command as a local user, you may be prompted for the root password while the process is running. For unsupported distributions, the dependency names will be displayed along with common names for the package. The install-dependencies option needs to be supplied with the test name or suite as an argument. When using the install option, the external dependencies are automatically checked.

make-download-cache

This option will create a download cache for use by the Phoronix Test Suite. The download cache is created of test files already downloaded to the local system. If passing any test/suite names to make-download-cache, the needed files for those test profiles will first be automatically downloaded before creating the cache.

remove-installed-test *[Test]*

This option will permanently remove a installed test by the Phoronix Test Suite.

Testing

auto-compare

This option will autonomously determine the most relevant test(s) to run for any selected sub-system(s). The tests to run are determined via OpenBenchmarking.org integration with the global results pool. Related test results from OpenBenchmarking.org are also merged to provide a straight-forward and effective means of carrying out a system comparison. If wishing to find comparable results for any particular test profile(s), simply pass the test profile names as additional arguments to this command.

benchmark *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will install the selected test(s) (if needed) and will proceed to run the test(s). This option is equivalent to running phoronix-test-suite with the install option followed by the run option. Multiple arguments can be supplied to run additional tests at the same time and save the results into one file.

estimate-run-time *[Test | Suite | OpenBenchmarking ID | Test Result]*

This option will provide estimates for test run-time / length.

finish-run *[Test Result]*

This option can be used if a test run had not properly finished running all tests within a saved results file. Using this option when specifying a saved results file where all tests had not completed will attempt to finish testing on the remaining tests where there are missing results.

run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will run the selected test(s).

run-random-tests

This option will query OpenBenchmarking.org to run random benchmarks and result comparisons on the system. This test can be used for simply supplying interesting results from your system onto OpenBenchmarking.org, stressing your system with random workloads, seeding new OpenBenchmarking.org results, etc. Basic options are provided at start-up for tuning the randomness of the testing when running this command.

run-tests-in-suite

This option can be used if you wish to run all of the tests found in a supplied suite, but you wish to re-configure each of the test options rather than using the defaults supplied by the suite.

stress-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will run the passed tests/suites in the multi-process stress-testing mode. The stress-run mode will not produce a result file but is rather intended for running multiple test profiles concurrently to stress / burn-in the system. The number of tests to run concurrently can be toggled via the PTS_CONCURRENT_TEST_RUNS environment variable and by default is set to a value of 2.

Batch Testing

batch-benchmark *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option and its arguments are equivalent to the benchmark option, but the process will be run in the Phoronix Test Suite batch mode.

batch-install *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

If you wish to run the install process in the Phoronix Test Suite batch mode but do not wish to run any tests at this time. Running the install process in the batch mode will use the default values and not prompt the user of any possible options, to ensure the process is fully automated.

batch-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option and its arguments are equivalent to the run option, but the process will be run in the Phoronix Test Suite batch mode.

batch-setup

This option is used to configure the batch mode options for the Phoronix Test Suite, which is subsequently written to the user configuration file. Among the options are whether to automatically upload the test results to OpenBenchmarking.org and prompting for the saved file name.

default-benchmark *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will install the selected test(s) (if needed) and will proceed to run the test(s) in the defaults mode. This option is equivalent to running phoronix-test-suite with the install option followed by the default-run option.

default-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option will run the selected test(s). The name of the test or suite must be supplied or the OpenBenchmarking.org ID or saved local file name. Multiple arguments can be supplied to run additional tests at the same time and save the results in a suite-like fashion. Unlike the normal run option, the default-run will not prompt the user to select from the available test options but will instead use the default options as automatically set by pts-core or the test profile. Use batch-run to automatically test all of the available options.

internal-run *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option and its arguments pre-set the Phoronix Test Suite batch run mode with sane values for carrying out benchmarks in a semi-automated manner and without uploading any of the result data to the public OpenBenchmarking.org.

OpenBenchmarking.org

clone-result *[OpenBenchmarking ID] ...*

This option will download a local copy of a file that was saved to OpenBenchmarking.org, as long as a valid public ID is supplied.

list-recommended-tests

This option will list recommended test profiles for benchmarking sorted by hardware sub-system. The recommended tests are determined via querying OpenBenchmarking.org and determining the most popular tests for a given environment based upon the number of times a test profile has been downloaded, the number of test results available on OpenBenchmarking.org for a given test profile, the age of the test profile, and other weighted factors.

make-openbenchmarking-cache

This option will attempt to cache the test profile/suite meta-data from OpenBenchmarking.org for all linked repositories. This is useful if you're going to be running the Phoronix Test Suite / Phoromatic behind a firewall or without any Internet connection. Those with unrestricted Internet access or not utilizing a large local deployment of the Phoronix Test Suite / Phoromatic shouldn't need to run this command.

openbenchmarking-changes

This option will list recent changes to test profiles of enabled OpenBenchmarking.org repositories.

openbenchmarking-login

This option is used for controlling your Phoronix Test Suite client options for OpenBechmarking.org and

syncing the client to your account.

openbenchmarking-refresh

This option is used for refreshing the stored OpenBenchmarking.org repostory information and other data. The Phoronix Test Suite will automatically refresh this data every three days or when other thresholds are exceeded, but this command can be used to manually refresh/updates the data.

openbenchmarking-repositories

This option will list the OpenBenchmarking.org repositories currently linked to this Phoronix Test Suite client instance.

openbenchmarking-uploads

This option will list any recent test result uploads from the system's IP address to OpenBenchmarking.org.

upload-result *[Test Result]*

This option is used for uploading a test result to OpenBenchmarking.org.

upload-test-profile

This option can be used for uploading a test profile to your account on OpenBenchmarking.org. By uploading your test profile to OpenBenchmarking.org, others are then able to browse and access this test suite for easy distribution in a seamless manner by other Phoronix Test Suite clients.

upload-test-suite

This option can be used for uploading a test suite to your account on OpenBenchmarking.org. By uploading your test suite to OpenBenchmarking.org, others are then able to browse and access this test suite for easy distribution.

System

diagnostics

This option will print information that is useful to developers when debugging problems with the Phoronix Test Suite and/or test profiles and test suites.

interactive

A simple text-driven interactive interface to the Phoronix Test Suite.

php-conf

This option will print information that is useful to developers when debugging problems with the Phoronix Test Suite and/or test profiles and test suites.

shell

A simple text-driven shell interface / helper to the Phoronix Test Suite. Ideal for those that may be new to the Phoronix Test Suite

system-info

Display the installed system hardware and software information as detected by the Phoronix Test Suite Phodevi Library.

system-sensors

Display the installed system hardware and software sensors in real-time as detected by the Phoronix Test Suite Phodevi Library.

Information

info *[Test | Suite | OpenBenchmarking ID | Test Result]*

This option will show details about the supplied test, suite, virtual suite, or result file.

list-available-suites

This option will list all test suites that are available from the enabled OpenBenchmarking.org repositories.

list-available-tests

This option will list all test profiles that are available from the enabled OpenBenchmarking.org repositories where supported on the system and are of a verified state.

list-available-virtual-suites

This option will list all available virtual test suites that can be dynamically created based upon the

available tests from enabled OpenBenchmarking.org repositories.

list-installed-dependencies

This option will list all of the packages / external test dependencies that are already installed on the system that the Phoronix Test Suite may potentially depend upon by test profiles.

list-installed-suites

This option will list all suites that are currently installed on the system.

list-installed-tests

This option will list all test profiles that are currently installed on the system.

list-missing-dependencies

This option will list all of the packages / external test dependencies that are missing from the system that the Phoronix Test Suite may potentially need by select test profiles.

list-not-installed-tests

This option will list all test profiles that are supported and available but presently NOT installed on the system.

list-possible-dependencies

This option will list all of the packages / external test dependencies that are are potentially used by the Phoronix Test Suite.

list-saved-results

This option will list all of the saved test results found on the system.

list-test-usage

This option will list various details about installed tests and their usage.

list-unsupported-tests

This option will list all available test profiles that are available from the enabled OpenBenchmarking.org repositories but are NOT SUPPORTED on the given hardware/software platform. This is mainly a debugging option for those looking for test profiles to potentially port to new platforms, etc.

search

This option provides command-line searching abilities for test profiles / test suites / test results. The search query can be passed as a parameter otherwise the user is prompted to input their search query..

Asset Creation

build-suite

This option will guide the user through the process of generating their own test suite, which they can then run. Optionally, passed as arguments can be the test(s) or suite(s) to add to the suite to be created, instead of being prompted through the process.

create-test-profile

This option can be used for creating a Phoronix Test Suite test profile by answering questions about the test for constructing the test profile XML meta-data and handling other boiler-plate basics for getting started in developing new tests.

debug-benchmark *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option is intended for use by test profile writers and is identical to the run option but will yield more information during the run process that can be used to debug issues with a test profile or to verify the test profile is functioning correctly.

debug-install *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option is intended for use by test profile writers and is identical to the install option but will yield more information during the run process that can be used to debug issues with a test profile installer or to verify the test profile is functioning correctly.

debug-result-parser *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This option is intended for use by test profile writers and is used for debugging a result parser. No test execution is done, but there must already be PTS-generated .log files present within the test's installation directory.

debug-test-download-links *[Test | Suite | OpenBenchmarking ID | Test Result]*

This option will check all download links within the specified test profile(s) to ensure there are no broken URLs.

download-test-files *[Test | Suite | OpenBenchmarking ID | Test Result] ...*

This will download the selected test file(s) to the Phoronix Test Suite download cache but will not install the tests.

inspect-test-profile *[Test]*

This option can be used for inspecting a Phoronix Test Suite test profile with providing inside details on test profiles for debugging / evaluation / learning purposes.

result-file-to-suite *[Test Result]*

This option will guide the user through the process of generating their own test suite, which they can then run, that is based upon an existing test results file.

validate-result-file

This option can be used for validating a Phoronix Test Suite result file as being compliant against the OpenBenchmarking.org specification.

validate-test-profile *[Test]*

This option can be used for validating a Phoronix Test Suite test profile as being compliant against the OpenBenchmarking.org specification.

validate-test-suite

This option can be used for validating a Phoronix Test Suite test suite as being compliant against the OpenBenchmarking.org specification.

Result Management

analyze-all-runs *[Test Result]*

This option will generate a candlestick graph showing the distribution of results from all trial runs. The candlestick graph is similar to the Japanese candlestick charts used by the financial industry, except instead of representing stock data it is numerical result data from all trial runs.\n\nThe tip of the upper-wick represents the highest value of the test runs with the tip of the lower-wick representing the lowest value of all test runs. The upper-edge of the candle body represents the first or last run value

and the lower-edge represents the first or last run value. Lastly, if the last run value is less than the first run value, the candle body is the same color as the graph background, otherwise the last run value is greater.

auto-sort-result-file *[Test Result]*

This option is used if you wish to automatically attempt to sort the results by their result identifier string.

edit-result-file *[Test Result]*

This option is used if you wish to edit the title and description of an existing result file.

extract-from-result-file *[Test Result]*

This option will extract a single set of test results from a saved results file that contains multiple test results that have been merged. The user is the prompted to specify a new result file name and select which result identifier to extract.

merge-results *[Test Result] ...*

This option will manually merge multiple sets of test results generated by the Phoronix Test Suite.

refresh-graphs *[Test Result]*

This option will re-render and save all result graphs within a saved file. This option can be used when making modifications to the graphing code or its color/option configuration file and testing the changes.

remove-from-result-file *[Test Result]*

This option is used if there is a set of test results you wish to remove/delete from a saved results file. The user must specify a saved results file and then they will be prompted to select the results identifier associated with the results they wish to remove.

remove-result *[Test Result]*

This option will permanently remove the saved file set that is set as the first argument.

rename-identifier-in-result-file *[Test Result]*

This option is used if you wish to change the name of the identifier in a test results file that is shown in the Phoronix Test Suite Results Viewer and the contained graphs.

rename-result-file *[Test Result]*

This option is used if you wish to change the name of the saved name of a result file.

reorder-result-file *[Test Result]*

This option is used if you wish to manually change the order in which test results are shown in the Phoronix Test Suite Results Viewer and the contained graphs. The user must specify a saved results file and then they will be prompted to select the results identifiers one at a time in the order they would like them to be displayed from left to right.

result-file-to-csv *[Test Result]*

This option will read a saved test results file and output the system hardware and software information along with the results to a CSV output. The CSV (Comma Separated Values) output can then be loaded into a spreadsheet for easy viewing.

result-file-to-json *[Test Result]*

This option will read a saved test results file and output the basic result information to JSON (JavaScript Object Notation).

result-file-to-pdf *[Test Result]*

This option will read a saved test results file and output the system hardware and software information along with the results to a PDF file.

result-file-to-text *[Test Result]*

This option will read a saved test results file and output the system hardware and software information to the terminal. The test results are also outputted.

show-result *[Test Result]*

Open up the test results in the Phoronix Test Suite Result Viewer or on OpenBenchmarking.org.

winners-and-losers *[Test Result]*

This option is used if you wish to analyze a result file to see which runs produced the most wins/losses of those result identifiers in the saved file.

Other

commands

This option will display a short list of possible Phoronix Test Suite commands.

debug-dependency-handler

This option is used for testing the distribution-specific dependency handler for external dependencies.

debug-render-test

This option is used during the development of the Phoronix Test Suite software for testing of the result and graph rendering code-paths This option will download a large number of reference test results from LinuxBenchmarking.com.

debug-self-test

This option is used during the development of the Phoronix Test Suite software for testing of internal interfaces, commands, and other common code-paths. The produced numbers should only be comparable for the same version of the Phoronix Test Suite, on the same hardware/software system, conducted on the same day of testing. This isn't intended as any scientific benchmark but simply to stress common PHP code-paths and looking for hot areas to optimize, etc.

help

This option will display a list of available Phoronix Test Suite commands and possible parameter types.

version

This option will display the Phoronix Test Suite client version.

Modules

auto-load-module

This option can be used for easily adding a module to the LoadModules list in the Phoronix Test Suite user configuration file. That list controls what PTS modules are automatically loaded on start-up of the Phoronix Test Suite.

list-modules

This option will list all of the available Phoronix Test Suite modules on this system.

module-info *[Phoronix Test Suite Module]*

This option will show detailed information on a Phoronix Test Suite module such as the version, developer, and a description of its purpose.

module-setup *[Phoronix Test Suite Module]*

This option will allow you to configure all available end-user options for a Phoronix Test Suite module. These options are then stored within the user's configuration file. Not all modules may have options that can be configured by the end-user.

test-module *[Phoronix Test Suite Module]*

This option can be used for debugging a Phoronix Test Suite module.

unload-module

This option can be used for easily removing a module from the LoadModules list in the Phoronix Test Suite user configuration file. That list controls what modules are automatically loaded on start-up of the Phoronix Test Suite.

User Configuration

enterprise-setup

This option can be run by enterprise users immediately after package installation or as part of an in-house setup script. Running this command will ensure the phoronix-test-suite program is never interrupted on new runs to accept user agreement changes and defaults the anonymous usage reporting to being disabled and other conservative defaults.

network-info

This option will print information detected by the Phoronix Test Suite around the system's network configuration.

network-setup

This option allows the user to configure how the Phoronix Test Suite connects to OpenBenchmarking.org and other web-services. Connecting through an HTTP proxy can be configured through this option.

user-config-reset

This option can be used for resetting the Phoronix Test Suite user configuration file to its default state.

user-config-set

This option can be used for setting an XML value in the Phoronix Test Suite user configuration file.

Web / GUI Support

gui

Launch the Phoronix Test Suite HTML5 web user-interface in the local GUI mode (no remote web support) and attempt to auto-launch the web-browser. THIS FEATURE IS CURRENTLY EXPERIMENTAL AND NO LONGER ACTIVE DEVELOPMENT. See Phoronix Test Suite Phoromatic as an alternative web UI approach.

Phoromatic

start-phoromatic-server

Start the Phoromatic web server for controlling local Phoronix Test Suite client systems to facilitate automated and repeated test orchestration and other automated features targeted at the enterprise.

Module Options

The following list is the modules included with the Phoronix Test Suite that are intended to extend the functionality of pts-core. Some of these options have commands that can be run directly in a similiar manner to the other Phoronix Test Suite user commands. Some modules are just meant to be loaded directly by adding the module name to the LoadModules tag in ~/.phoronix-test-suite/user-config.xml or via the PTS_MODULES environment variable. A list of available modules is also available by running *phoronix-test-suite list-modules.*

Backup Creation + Restore

This is a module for creating backups of the Phoronix Test Suite / Phoromatic and allows for restoring of created backups. The backup will be in ZIP or TAR format. If only a path is specified, the file-name will be auto-generated with a current time-stamp.

phoronix-test-suite backup.create

phoronix-test-suite backup.restore

Dummy Module

This is a simple module intended for developers to just demonstrate some of the module functions.

phoronix-test-suite dummy_module.dummy-command

Generate Perf FlameGraphs For Tests

Setting FLAME_GRAPH_PATH=<path to flamegraph path> will auto-load and enable this Phoronix Test Suite module. The module will generate a Linux perf FlameGraph for each test run during the benchmarking process. Details on FlameGraph @ https://github.com/brendangregg/FlameGraph

This module utilizes the following environmental variables: FLAME_GRAPH_PATH.

Graphics Override

This module allows you to override some graphics rendering settings for the ATI and NVIDIA drivers while running the Phoronix Test Suite.

This module utilizes the following environmental variables: FORCE_AA, FORCE_AF.

Result Exporter To HTML

This module allows basic exporting of results to HTML for saving either to a file locally (specified using the EXPORT_RESULTS_HTML_FILE_TO environment variable) or to a mail account (specified using the EXPORT_RESULTS_HTML_EMAIL_TO environment variable). EXPORT_RESULTS_HTML_EMAIL_TO supports multiple email addresses delimited by a comma.

This module utilizes the following environmental variables: EXPORT_RESULTS_HTML_EMAIL_TO, EXPORT_RESULTS_HTML_FILE_TO.

Linux Perf Framework Reporter

Setting LINUX_PERF=1 will auto-load and enable this Phoronix Test Suite module. The module also depends upon running a modern Linux kernel (supporting perf) and that the perf binary is available via standard system paths.

This module utilizes the following environmental variables: LINUX_PERF.

Log Exporter

This module allows for easily exporting test run logs and system logs to external locations via specifying the directory paths via the COPY_TEST_RUN_LOGS_TO and COPY_SYSTEM_LOGS_TO environment variables.

This module utilizes the following environmental variables: COPY_TEST_RUN_LOGS_TO, COPY_SYSTEM_LOGS_TO.

MATISK

My Automated Test Infrastructure Setup Kit

phoronix-test-suite matisk.run

phoronix-test-suite matisk.template

OpenBenchmarking.org Auto Comparison

This module prints comparable OpenBenchmarking.org results in the command-line for reference purposes as tests are being run. OpenBenchmarking.org is automatically queried for results to show based on the test comparison hash and the system type (mobile, desktop, server, cloud, workstation, etc). No other system information or result data is transmitted..

phoronix-test-suite ob_auto_compare.debug

Performance Per Dollar/Cost Calculator

Setting the COST_PERF_PER_DOLLAR= environment variable to whatever value of the system cost/component you are running a comparison on will yield extra graphs that calculate the performance-per-dollar based on the test being run. The COST_PERF_PER_DOLLAR environment variable is applied just to the current test run identifier. Set the COST_PERF_PER_UNIT= environment variable if wishing to use a metric besides dollar/cost. The COST_PERF_PER_HOUR value can be used rather than COST_PERF_PER_DOLLAR if wishing to calculate the e.g. cloud time or other

compute time based on an hourly basis.

phoronix-test-suite perf_per_dollar.add

This module utilizes the following environmental variables: COST_PERF_PER_DOLLAR, COST_PERF_PER_UNIT, COST_PERF_PER_HOUR.

Performance Tip Prompts

This module alerts the user if the system configuration may not be the right one for achieving the best performance with the target benchmark(s). This initial version of the module actually cares only about the BFQ I/O scheduler and powersave governor checks.

phoronix-test-suite perf_tips.show

This module utilizes the following environmental variables: SUPPRESS_PERF_TIPS.

Phoromatic Client

The Phoromatic client is used for connecting to a Phoromatic server (Phoromatic.com or a locally run server) to facilitate the automatic running of tests, generally across multiple test nodes in a routine manner. For more details visit http://www.phoromatic.com/. This module is intended to be used with Phoronix Test Suite 5.2+ clients and servers.

phoronix-test-suite phoromatic.connect

phoronix-test-suite phoromatic.explore

phoronix-test-suite phoromatic.upload-result

phoronix-test-suite phoromatic.set-root-admin-password

phoronix-test-suite phoromatic.list-results

phoronix-test-suite phoromatic.clone

phoronix-test-suite phoromatic.export-results-for-account-schedules

Pushover.net

Submit notifications to your iOS/Android mobile devices of test results in real-time as push notifications, etc. Using the Pushover.net API.

This module utilizes the following environmental variables: PUSHOVER_NET_USER.

Phoronix Test Suite v8.0.0

Test Client Documentation

Report Test Time Graphs

Setting the RUN_TIMES_ARE_A_BENCHMARK=1 environment variable will automatically create additional graphs for each test run plotting the run-time needed for each test being executed. Setting the INSTALL_TIMES_ARE_A_BENCHMARK=1 environment variable will automatically create additional graphs for each test run plotting the time required for the test installation. Setting the INSTALL_SIZES_ARE_A_BENCHMARK=1 environment variable will automatically create additional graphs for each test run plotting the size of the installed test directory.

This module utilizes the following environmental variables: RUN_TIMES_ARE_A_BENCHMARK, INSTALL_TIMES_ARE_A_BENCHMARK, INSTALL_SIZES_ARE_A_BENCHMARK.

Result Notifier

A notification module.

Custom Result Export Methods

A simple example module about interfacing with Phoronix Test Suite core for dumping result files in a custom format.

phoronix-test-suite results_custom_export.nf

System Monitor

This module contains sensor monitoring support.

This module utilizes the following environmental variables: MONITOR, PERFORMANCE_PER_WATT, MONITOR_INTERVAL, MONITOR_PER_RUN.

Timed Screenshot

This is a module that will take a screenshot of the system at a pre-defined interval. ImageMagick must be installed onto the system prior to using this module.

This module utilizes the following environmental variables: SCREENSHOT_INTERVAL.

Toggle Screensaver

This module toggles the system's screensaver while the Phoronix Test Suite is running. At this time, the GNOME and KDE screensavers are supported.

This module utilizes the following environmental variables: HALT_SCREENSAVER.

Update Checker

Phoronix Test Suite v8.0.0

Test Client Documentation

This module checks to see if the Phoronix Test Suite -- and its tests and suites -- are up to date.

System Event Watchdog

This module has support for stopping/interrupting tests if various system issues occur, like a temperature sensor exceeds a defined threshold.

This module utilizes the following environmental variables: WATCHDOG_SENSOR, WATCHDOG_SENSOR_THRESHOLD.

Phoronix Test Suite

Phoronix Test Suite v8.0.0

Test Client Documentation

Installation Instructions

Setup Overview

The Phoronix Test Suite supports Linux, Apple Mac OS X, Microsoft Windows, Solaris, Hurd, BSD, and other operating system environments. The only Linux distribution-specific code deals with the external dependencies support feature that are set by individual test profiles. If you are not running one of the supported Linux distributions, Solaris, BSD, or Mac OS X, you may need to install a package manually (as instructed by the Phoronix Test Suite) in order for a test to run. An example of an external dependency would be GCC and the OpenGL Utility Toolkit being needed for test profiles that build an OpenGL benchmark from source-code.

Among the distributions where the Phoronix Test Suite has been officially tested include Ubuntu, Fedora, Mandriva / Mageia, Gentoo, PCLinuxOS, Arch Linux, Pardus, OpenSuSE, Optware, webOS, Zenwalk, CentOS, Red Hat Enterprise Linux, Oracle Linux, Scientific Linux, Debian, Mint, MEPIS, Alpine Linux, Void Linux, Intel Clear Linux, and Amazon Linux EC2.

Among the tested BSD distributions are FreeBSD, PC-BSD, NetBSD, OpenBSD, Debian GNU/kFreeBSD, and DragonflyBSD. Tested Solaris distributions include OpenSolaris, Solaris Express 11, Oracle Solaris 11, OpenIndiana, Illumos, and Nexenta.

Dependencies

The only required dependency for the Phoronix Test Suite is PHP 5.3 or newer. On Linux distributions, the needed package is commonly called *php5-cli* or *php-cli* or *php7* or *php*. It is important to note that only PHP for the command-line is needed and not a web server (Apache) or other packages commonly associated with PHP and its usage by web-sites. The PHP5 version required is PHP 5.3+ and can also be found at **www.php.net**. PHP 7 is fully supported by the Phoronix Test Suite as well as HHVM.

For installing PHP on Windows, the **Microsoft Web Platform Installer** provides an automated alternative that will install PHP into the directory expected by Phoronix Test Suite. Users opting to install PHP manually must extract the files to *C:\Program Files (x86)\PHP* or a subdirectory.

As part of the PHP requirement, the following PHP extensions are required and/or highly recommended in order to take advantage of the Phoronix Test Suite capabilities:

PHP DOM is needed for XML operations and must be installed for the Phoronix Test Suite to function.

PHP ZIP is needed for file compression and decompression and specifically dealing with test profiles and suites obtained via OpenBenchmarking.org or when uploading such tests and suites.

Phoronix Test Suite v8.0.0

Test Client Documentation

PHP OpenSSL is used for enabling HTTPS communication with Phoronix Test Suite / OpenBenchmarking.org servers.

PHP GD is highly recommended for rendering of test data to JPEG and PNG image formats and is selectively used in other image operations.

PHP Zlib is highly recommended for greater data compression when dealing with remote OpenBenchmarking.org assets.

PHP PCNTL is used for multi-threaded system sensor monitoring support during the testing process and other threaded tasks by the Phoronix Test Suite module framework.

PHP POSIX is used for reliably obtaining more system information in an efficient manner.

PHP CURL is supported as an alternative networking library for improved network performance in downloading test files and other operations.

PHP FPDF is used to generate PDF reports of test data.

Without all of these extensions, some capabilities of the Phoronix Test Suite will not be available. Many of these packages are enabled by default and do not require any additional installation steps on most Linux distributions, otherwise they are often found in the package vendor's repository.

Notes

General

You may need to modify the *php.ini* file on your system in order to support uploading results to OpenBenchmarking.org or logging into your OpenBenchmarking.org account. The *allow_url_fopen*, *file_uploads*, and *allow_url_include* options must be set to true in the PHP configuration. See additional information in this **Phoronix Forums support thread**.

Major updates to the Phoronix Test Suite are released on a quarterly basis. The latest stable and development versions of the Phoronix Test Suite are available at **Phoronix-Test-Suite.com**. The Git repository where the latest Phoronix Test Suite code is provided is hosted at **GitHub.com/phoronix-test-suite** and can be cloned/pulled from the *https://github.com/phoronix-test-suite/phoronix-test-suite.git* repository location. The latest upstream development code is housed in the master tree while older Phoronix Test Suite releases are available in their respective Git branches based upon the release's code-name.

If building the PHP package from upstream sources, it should just be a matter of running *./configure* with the *--enable-zip* flag (all other requirements should be apart of the stock PHP configuration) to satisfy the PHP needs of the Phoronix Test Suite.

Phoronix Test Suite v8.0.0

Test Client Documentation

File Structure

If manually changing the location of the *phoronix-test-suite* launcher file, the *PTS_USER_PATH* environment variable must be adjusted inside the file to reflect the absolute location that leads to the root directory of the *pts* and *pts-core* folders. The *pts-core* directory contains the "engine" of the Phoronix Test Suite.

Running Locally

The Phoronix Test Suite can be simply extracted from the downloaded *.tar.gz* or *.zip* file or it can also be installed system-wide. If you just wish to run the Phoronix Test Suite without installing it, open a terminal and run *./phoronix-test-suite <options>* from the same directory.

Generic Installation

Running *install-sh* from the root directory of the Phoronix Test Suite will install the software for system-wide access. By default the *phoronix-test-suite* executable is in */usr/bin/*, the Phoronix Test Suite files in */usr/share/phoronix-test-suite/*, and the documentation in */usr/share/doc/phoronix-test-suite/*. Root access is required. The default installation prefix is /usr/ but can be adjusted as the first argument (example: *install-sh /home/user/* to install the Phoronix Test Suite in your home directory).

Debian/Ubuntu Installation

Debian/Ubuntu users are able to follow the Generic Installation instructions or can obtain a Debian Package from the Phoronix Test Suite web-site. The package contains the *phoronix-test-suite* executable in */usr/bin/*, the Phoronix Test Suite files in */usr/share/phoronix-test-suite/*, and the documentation in */usr/share/doc/phoronix-test-suite/*.

A phoronix-test-suite package can also be found in the official Ubuntu universe repository beginning with the Ubuntu 9.04 (Jaunty) release, but it may be out of date compared to the latest upstream version.

On Ubuntu 13.04 and newer if you experience an issue of system sensor monitoring not working, other multi-process tasks, or related issues reported as PHP PCNTL missing, the pcntl functions need to be removed from the *disable_functions* line within the */etc/php5/cli/php.ini* file. This may also affect other Linux distribution support too if the vendor disables access to the PCNTL functions.

Fedora / Red Hat Installation

The Phoronix Test Suite can be installed on Fedora, Red Hat Enterprise Linux, and CentOS systems using the generic installation method. Alternatively, a *phoronix-test-suite* package is available in recent versions of the Fedora repository and in the EPEL (Extra Packages for Enterprise Linux) repository for Red Hat Enterprise Linux. At this time, however, the Fedora-supplied package is vastly out of date.

Phoronix Test Suite v8.0.0

Test Client Documentation

Gentoo Installation

Gentoo ebuild's for the Phoronix Test Suite have been created by users and can be currently found in the **Gentoo BugZilla**.

Mandriva Installation

Phoronix Test Suite packages are available through Mandriva Cooker starting with the Mandriva Linux 2009 release.

OpenSolaris Installation

The Phoronix Test Suite core (pts-core) is supported on Solaris and OpenSolaris platforms, but some features may not fully function. Additionally, the PTS External Dependencies feature is supported on OpenSolaris using IPS (the Image Packaging System). However, not all test profiles are supported on Solaris / OpenSolaris, and it may be in violation of your end-user agreement to perform such benchmarks on the Sun Solaris operating system. Using the OpenSolaris Image Packaging System, installing *SUNWphp524* will provide PHP, PHP GD, and related dependencies. The Phoronix Test Suite has also been tested on Oracle Solaris 11 Express, OpenIndiana, StormOS, and Nexenta Core Platform.

BSD Installation

The Phoronix Test Suite also supports *BSD operating systems. However, like the Solaris support, not all test profiles are compatible with BSD operating systems, but pts-core has been tested against FreeBSD 7.x/8.x/9.x, NetBSD, PC-BSD 7/8/9, DragonFlyBSD 3.0/3.2/3.4/3.6/4.2, and OpenBSD 4.4/4.6. The Phoronix Test Suite will also work with Debian GNU/kFreeBSD.

For DragonFlyBSD, it's a matter of just using *pkg_radd* with php5, php5-json, php5-dom, and php5-zip or *pkgsrc*. With DragonFlyBSD using dports, the command is *pkg install php5 php5-dom php5-zip php5-json php5-simplexml*. Under FreeBSD/PC-BSD, PHP can be obtained from the *php5* and *php5-extensions* Ports collection.

Mac OS X Installation

The Phoronix Test Suite is fully supported on Apple's Mac OS X operating system as of Mac OS X 10.5 with improved support under Mac OS X 10.6. PHP ships with Mac OS X by default so it's simply a matter of downloading the Phoronix Test Suite package, extracting it, and running the executable. For tests that rely upon a compiler, Apple's XCode with GCC and LLVM can be utilized.

Windows Installation

The Phoronix Test Suite support on Microsoft Windows is in development and will function just fine, but is not yet feature complete compared to the other operating systems support. Only a small subset of test profiles, mostly related to graphics, are supported. In terms of the version support, right now the

Phoronix Test Suite v8.0.0

Test Client Documentation

Phoronix Test Suite is developed and tested solely against Microsoft Windows 7 x64 and Microsoft Windows 8 x64.

At this time the Phoronix Test Suite client is dependent upon the *php.exe* executable being installed to *C:\Program Files (x86)\PHP* or one of its subdirectories (see PHP installation notes above). Once this requirement is met, simply download the Phoronix Test Suite package, extract it to a directory of your choice, and run *phoronix-test-suite.bat*

NOTE: because of lack of external dependencies support, it is necessary to manually download and install the files for some test profiles, such as the Unigine graphics benchmarks. See Troubleshooting for more information.

Facebook HipHop

The Phoronix Test Suite can use Facebook's **HipHop HHVM** virtual machine as an alternative to the de facto PHP implementation. The Phoronix Test Suite has been tested against HHVM 2.0. If HHVM is present on the system but not PHP, it will automatically be used by the Phoronix Test Suite. Alternatively, the *PHP_BIN* environment variable can be set to point to an alternative *hhvm* binary.

The Phoronix Test Suite also supports the older version of Facebook HipHop that serves as a PHP to C++ translator and compiler with GCC. This support though is primitive and not officially supported. To compile the Phoronix Test Suite using HipHop/GCC, run *find . -name "*.php" > files.list && hphp --input-list=files.list -k 1 --log=3 --include-path="." --cluster-count=50 -v "AllDynamic=true" -v "AllVolatile=true"* from the root *phoronix-test-suite/* directory. It can then be executed in the form of */tmp/hphp_XXX/program -f pts-core/phoronix-test-suite.php system-info*.

External Dependencies

The Phoronix Test Suite has a feature known as "External Dependencies" where the Phoronix Test Suite can attempt to automatically install some of the test-specific dependencies on supported distributions. If running on a distribution where there is currently no External Dependencies profile, the needed package name(s) are listed for manual installation.

Below are a list of the operating systems that currently have external dependencies support within the Phoronix Test Suite for the automatic installation of needed test files.

Alpine Linux
Amazon
Angstrom
Arch Linux
CentOS
ClearOS
ClearOS Core Server
Debian
DragonFlyBSD
Fedora
Fluxbuntu
GNU KFreeBSD
Gentoo
Goobuntu
HP
Joli Cloud
Linaro
Linux Embedded Development Environment
Linux Mint
MEPIS
Mac OS X
MacPorts
Mageia
Mandriva
MeeGo
Microsoft Windows
MidnightBSD
Moblin
Mythbuntu
NetBSD
Nexenta Core

Phoronix Test Suite v8.0.0

Test Client Documentation

OLPC
OpenIndiana
OpenMandriva
OpenSolaris
OpenSuSE
Optware
Oracle Server
PCLinuxOS
PTS Desktop Live
PTS Linux Live
Palm
Pardus Linux
Red Hat Enterprise
Red Hat Enterprise Server
SUSE
SUSE Linux
Scientific
ScientificSL
Solus
Solus Linux
Termux
Ubuntu
Void Linux
Zenwalk
gNewSense
macOS Brew

Configuration

User Files & Folders

~/.phoronix-test-suite/user-config.xml

This is a per-user configuration file. Among the information stored here is the test options, locations for storing files, and batch mode options. This file is formatted in XML.

~/.phoronix-test-suite/graph-config.json

This is a per-user configuration file for storing graph attributes. The adjustable options include HTML hex color codes for different areas of the graph, dimensions of the graph, and font sizes. This file is formatted in JSON.

~/.phoronix-test-suite/download-cache/

This directory contains test packages that have been downloaded for test profiles. For more information on the download cache.

~/.phoronix-test-suite/installed-tests/

This directory is where tests are installed by default. Each test has its own directory within a sub-directory of *installed-tests/* based upon its OpenBenchmarking.org repository. In the test's folder is a *pts-install.xml* file used for managing the installation.

~/.phoronix-test-suite/test-results/

This directory is where tests results are saved by default. Each saved file has its own directory. In the saved directory is then a *composite.xml* file containing the useful results while in the *test-X.xml* files are back-ups of the results.

~/.phoronix-test-suite/modules-data/

This is the directory where any Phoronix Test Suite modules should save any files to, within a sub-directory of the module's name. The module configuration settings are also stored within this directory.

~/.phoronix-test-suite/test-profiles/

This is the directory where test profiles are stored.

~/.phoronix-test-suite/test-suites/

Phoronix Test Suite v8.0.0

Test Client Documentation

This is the directory where test suites are stored.

Environment Variables

TEST_RESULTS_NAME

When this variable is set, the value will be used as the name for automatically saving the test results.

TEST_RESULTS_IDENTIFIER

When this variable is set, the value will be used as the test identifier when automatically saving the test results.

TEST_RESULTS_DESCRIPTION

When this variable is set, the value will be used as the test results description when saving the test results.

PRESET_OPTIONS

For setting any test option(s) from an environment variable rather than being prompted for the options when running a test. Example: *PRESET_OPTIONS="stream.run-type=Add" ./phoronix-test-suite benchmark stream*. Multiple options can be passed to this environment variable when delimited by a semicolon.

SKIP_TESTS

If there are any test(s) to exempt from the testing process, specify them in this variable. Multiple tests can be waived by delimiting each test identifier by a comma. A test hardware type (i.e. Graphics) can also be supplied for skipping a range of tests.

SKIP_TESTS_HAVING_ARGS

If any of the test(s) have an argument matching any strings contained in this environment variable, the test execution will be skipped. Multiple strings can be set when delimiting by a comma.

RUN_TESTS_IN_RANDOM_ORDER

Setting this environment variable will cause the tests to be run in a random order.

SKIP_TESTING_SUBSYSTEMS

If you are running a set of benchmarks (namely a result file) but wish to skip some of the tests that don't belong to a certain test type group, you can set the hardware types to test via this environment variable. E.g. setting *SKIP_TESTING_SUBSYSTEMS=Graphics* will skip all test profiles to run that are not of the

graphics test group. Multiple types should be delimited by a comma.

PTS_MODULE_SETUP

This variable can be used to load Phoronix Test Suite module settings automatically when using the *module-setup* option. An example would be: *PTS_MODULE_SETUP="phoromatic.remote_host=http://www.phoromatic.com/; phoromatic.remote_account=123456; phoromatic.remote_verifier=ABCD" phoronix-test-suite module-setup phoromatic*.

PTS_MODULES

If there are any Phoronix Test Suite modules to additionally load, they can be specified here. Multiple modules can be supplied by delimiting them with a comma. The more appropriate way of loading Phoronix Test Suite modules for longer periods of time is by using the *~/.phoronix-test-suite/user-config.xml* configuration.

NO_PHODEVI_CACHE

This is a debugging option to disable the Phodevi cache from being loaded of cached software/hardware information. Instead, all software/hardware will be polled from the Phodevi library without caching.

EXTERNAL_PHODEVI_CACHE

This option can be used for loading an external Phodevi cache. Such as loading the native hardware/software information from within a Windows Wine client from a native system host.

PTS_DISPLAY_MODE

If you wish to load a non-default display mode for a single instance, specify the mode in this variable.

TOTAL_LOOP_TIME

When running any test(s), if you would like the test(s) to continue running as a loop until a certain time has been reached, this variable can be used. The value should be the number of minutes to run the testing process before the loop is ended. The testing will finish whenever the currently active test has finished once the time has elapsed. The minimum value allowed is 10 minutes.

LIMIT_ELAPSED_TEST_TIME

If you want to ensure that the time for a given Phoronix Test Suite process doesn't elapse past a certain number of minutes, specify the number of minutes for this environment variable. When the amount of time spent testing exceeds that amount, the testing will end prematurely while still saving the tests that were completed in time.

Phoronix Test Suite v8.0.0

Test Client Documentation

TOTAL_LOOP_COUNT

When running any test(s), if you would like the test(s) to continue running for a number of times, this variable can be used. The value should be the number of times to loop the testing process before ending.

FORCE_TIMES_TO_RUN

If you wish to override the number of times to run each test -- rather than the Phoronix Test Suite using the number of times specified in each test profile -- this variable can be used.

FORCE_MIN_TIMES_TO_RUN

This is similar to the FORCE_TIMES_TO_RUN option but will only be used if the test profile's run count is less than this defined value.

NO_FILE_HASH_CHECKS

To disable MD5/SHA256 check-sums from being checked when downloading test files, set this variable to 1. This variable used to be known as *NO_MD5_CHECKS*, which is still honored but was changed to *NO_FILE_HASH_CHECKS* to reflect other kind of file hash sum checks.

NO_HTTPS

Set this environment variable to 1 if you don't wish to use HTTPS download links for test profiles (or the system/network lacks HTTPS support). When enabled, HTTPS links will then be done over HTTP.

PTS_DOWNLOAD_CACHE

While non-standard Phoronix Test Suite download caches can be specified within the *user-config.xml* file, an additional directory to look for potential Phoronix Test Suite download files can be specified by this variable.

GRAPH_HIGHLIGHT

If this variable is set with a valid test identifer from a result file whether you are using the *refresh-graphs* command or any other related to the rendering of test results on a bar graph, the specified test identifier's result will be rendered in a different color than the other test results. Multiple identifiers can be specified when delimited by a comma. Additionally, for each key it is possible to provide the actual color value, or an index in the color palette. Example:
"will_be_different,group1a=1,group1b=1,blue=#0000ff"

TEST_EXEC_PREPEND

Set this variable to any command/environment variable that you may be passed prepended to the test

execution string at runtime.

VIDEO_MEMORY

If Phodevi fails to detect the system's video memory capacity or is incorrectly detected, the video memory capacity (in MB) can be specified by this variable.

OVERRIDE_VIDEO_MODES

If Phodevi fails to detect all of the system's monitor video modes or a separate set of modes would be preferred, the modes can be specified in this variable. Example: *OVERRIDE_VIDEO_MODES=800x600,1024x768,1280x1024 phoronix-test-suite benchmark nexuiz*.

SKIP_TEST_SUPPORT_CHECKS

If this environment variable is set, it will not honor the support checks made by individual test profiles. I.e. test profiles that would normally be considered un-supported on a given platform are attempted to install and run regardless.

SKIP_ALL_TEST_SUPPORT_CHECKS

If this environment variable is set, all tests will be permitted on the client for execution. SKIP_ALL_TEST_SUPPORT_CHECKS is more liberal than SKIP_TEST_SUPPORT_CHECKS in letting disk tests run on RAM-based file-systems, attempt to run 2D/3D tests on VESA display drivers, and other special cases.

DEFAULT_VIDEO_MODE

If Phodevi fails to detect the system's monitor standard / default resolution, the mode can be specified in this variable. Example: *DEFAULT_VIDEO_MODE=1680x1050 phoronix-test-suite benchmark nexuiz* .

SKIP_EXTERNAL_DEPENDENCIES

To skip the Phoronix Test Suite external dependency checking/installation when installing a test, set this environment variable to *1*. If wishing to skip only certain external dependencies, set this variable's value to the name of the external dependencies (the generic dependency names used by the Phoronix Test Suite) to not install. Multiple dependencies to skip can be delimited by a comma.

Phoronix Test Suite

Phoronix Test Suite v8.0.0

Test Client Documentation

General Information

Frequently Asked Questions

Q: May I use the Phoronix Test Suite when running benchmarks for my own publication or blog? Are there any publishing restrictions?

A: Anyone is more than welcome to use the Phoronix Test Suite for their own publication or purpose. While the Phoronix Test Suite came out of our internal test tools for carrying out Linux hardware reviews at **Phoronix.com**, we invite other hardware review web-sites, technology journals, and independent publications to use our software too. While not required, we would just kindly ask that you mention in your review/article that the *Phoronix Test Suite* was used for carrying out your testing, and ideally to link to **www.phoronix-test-suite.com** so that your readers will know where to obtain the software if they are interested in running the tests. You are also more than welcome to upload your results to **OpenBenchmarking.org** so that others may compare their results against yours in an easy manner.

We also try to make the Phoronix Test Suite easy-to-use by independent publications. For example, if you would like to watermark your web-site's URL into the graphs containing your test results, that can be easily modified in *~/.phoronix-test-suite/graph-config.json*. The colors and other graph settings are also stored in this XML file. If you are a publication and run into any issues with the Phoronix Test Suite or have a feature request, please let us know.

A partial list of other publications using the Phoronix Test Suite can be found in **this forum thread**.

Q: Why does the Phoronix Test Suite not use my distribution's package management system for acquiring all needed packages?

A: The tests themselves are generally downloaded from source and built locally on the machine, rather than fetching any distribution-specific packages. This is done to ensure more comparable results across operating systems / releases, etc. The distribution packager could be applying a number of unknown patches to the software, building the software with unique build options, or making other changes to the software that could skew the results.

Q: Besides being a developer, documentation writer, or having any other unique technical abilities, how else can I contribute to the Phoronix Test Suite?

A: Independent code contributions are very welcome as well as creating your own test profiles and suites. We also appreciate any feedback, comments, or other ideas either by emailing us, posting on the forums, or sending a message to the mailing list. You can also see this (outdated) **forum thread**.

Q: Do you offer technical support for the Phoronix Test Suite

A: Paid, professional support is available and is done via **our commercial services**. We also offer Phoromatic licenses for use within a corporate intranet and other custom services. Free, community support is offered via our **mailing list**, IRC channel (*#phoronix* on *FreeNode.net*, and the **Phoronix Forums**.

Q: May I put the Phoronix Test Suite logo on my company's web-site or on my product packaging?

A: **Contact us** for licensing information and details regarding the Phoronix Certification & Qualification Suite.

Q: How often is the Phoronix Test Suite updated?

A: We provide major updates on a quarterly basis with an occasional point release to fix outstanding bugs or address other issues. The latest work going into the Phoronix Test Suite is accessible via our Git repository at **GitHub.com/phoronix-test-suite**.

Tips & Tricks

General

- The desktop's screensaver will automatically be shutdown when a test is running and will be restored to its previous state upon the test's completion. This is supported for GNOME, KDE, and other XDG-supportive desktop environments.

- If you have many computers you wish to benchmark, once all of your tests have been downloaded, run *phoronix-test-suite make-download-cache* to generate a copy of these files at *~/.phoronix-test-suite/download-cache/*. A download cache is used for conserving time and bandwidth by eliminating the need for the Phoronix Test Suite to download files that have already been downloaded once. Copy this folder to the other systems or copy it to a DVD or USB hard drive, connect it to the next test system, and the Phoronix Test Suite will automatically use this local download cache. Or store these files on a local HTTP/FTP server and then update your *~/.phoronix-test-suite/user-config.xml* file to reflect the location of this download cache directory.

Running

- When running a test in batch mode (through the use of the *batch-run* or *batch-benchmark* options) that normally has end-user options (such as the sub-test to run or resolution), the Phoronix Test Suite will run the test with each unique combination of options possible, if configured appropriately.

- When running a test where you are prompted to enter any test options, multiple selections can be performed -- which will result in multiple test runs for each combination of the selected option(s) -- by separating each intended test option / number with a comma.

- When being prompted for the test identifier or the file name for saving the results, several user

variables are supported. These include *$VIDEO_RESOLUTION*, *$VIDEO_CARD*, *$OPERATING_SYSTEM*, *$PROCESSOR*, *$MOTHERBOARD*, *$CHIPSET*, and *$KERNEL_VERSION*. If any of these variables are entered, the Phoronix Test Suite will replace them with their respective values before saving the results.

- If *RemoveDownloadFiles* is set to *TRUE* within the *user-config.xml* file, once a test has been installed the originally downloaded files for that test will be automatically removed. This conserves disk space but will cause these files to be re-downloaded the next time the test needs to be re-installed. This will also not back up the downloaded files to the Phoronix Test Suite download cache. Enabling this option is just recommended for users with very limited disk space.

- If the amount of video memory for your graphics card is incorrectly reported by the Phoronix Test Suite (you can check by running *phoronix-test-suite diagnostics*), you can use the *VIDEO_MEMORY=* environment variable for overriding the video memory capacity (in Megabytes) used by the Phoronix Test Suite.

- If the *DISPLAY* environment variable is not set or *NO_GRAPHICS_TESTS* environment variable is set, tests of type *Graphics* will not be run. Likewise, if *NO_SYSTEM_TESTS* environment variable is set, tests of type *System* will not run. This applies to all test types where *NO_<TEST TYPE>_TESTS* is set.

- If while running multiple tests you want to quit the testing prematurely, in a new terminal window type *touch ~/.phoronix-test-suite/halt-testing*. All results for tests that had already run will be saved (permitting you opted to save the results), except for the test currently being run.

- If you wish to stop the current test run prematurely but continue the testing process, in a new terminal window type *touch ~/.phoronix-test-suite/skip-test*.

- If you want the specified test(s) to run in a loop for a set period of time, use the *TOTAL_LOOP_TIME* environment variable. For instance, running *TOTAL_LOOP_TIME=120 phoronix-test-suite benchmark ffmpeg* would keep running the ffmpeg test profile for 120 minutes.

- If you want the specified test(s) to run in a loop for a set number of times, use the *TOTAL_LOOP_COUNT* environment variable. For instance, running *TOTAL_LOOP_COUNT=3 phoronix-test-suite benchmark ffmpeg* would keep running the ffmpeg test profile three times.

- When any tests are being installed and when tests are being run, a lock is created in the system's temporary directory with the name *phoronix-test-suite.active* (i.e. */tmp/phoronix-test-suite.active*) and is removed upon completion. Thus if you have any system scripts that you wish to run when tests are not running or being installed as to not impact the results, one simple way to handle this is by having the script check for the existence of this lock.

Troubleshooting

- If a test profile fails immediately after starting, check the test profile's directory in

~/.phoronix-test-suite/installed-tests/ to confirm that the needed files are present. On platforms without External Dependencies support (Windows), it may be necessary to download the files manually and place them in this directory. If this is the case, you will notice that the "Downloading" phase of test installation completes instantly.

- Inspect the scripts inside the above test profile's directory and confirm that directories or search paths for the test correspond to those on your system

- Try running the test profile with the *debug-benchmark* command, or reinstalling with the *debug-install* command and make note of any unusual output.

Configuration

- The user configuration options for the Phoronix Test Suite are stored in *~/.phoronix-test-suite/user-config.xml*. The batch mode options are also stored within this file and those can be adjusted by running *phoronix-test-suite batch-setup*.

- The colors, size, and other attributes for the graphs found within the Phoronix Test Suite Results Viewer can be modified via the file *~/.phoronix-test-suite/graph-config.json*.

Test / Suite Writing

- The Phoronix Test Suite recursively determines tests/suites and allows a suite to call another suite.

Module Writing

- By writing a Phoronix Test Suite module, you can easily extend the functionality of the Phoronix Test Suite. Some examples are being able to email the results upon completion, controlling the system's screensaver, updating a text LCD panel with the current test status, etc.

Virtual Test Suites

Virtual test suites are not like a traditional test suite defined by the XML suite specification. Virtual test suites are dynamically generated in real-time by the Phoronix Test Suite client based upon the specified test critera. Virtual test suites can automatically consist of all test profiles that are compatible with a particular operating system or test profiles that meet other critera. When running a virtual suite, the OpenBenchmarking.org repository of the test profiles to use for generating the dynamic suite must be prefixed.

Virtual test suites can be installed and run just like a normal XML test suite and shares nearly all of the same capabilities. However, when running a virtual suite, the user will be prompted to input any user-configuration options for needed test profiles just as they would need to do if running the test individually. When running a virtual suite, the user also has the ability to select individual tests within the suite to run or to run all of the contained test profiles. Virtual test suites are also only supported for an OpenBenchmarking.org repository if there is no test profile or test suite of the same name in the repository. Below is a list of common virtual test suites for the main Phoronix Test Suite repository, but the dynamic list of available virtual test suites based upon the enabled repositories is available by running *phoronix-test-suite list-available-virtual-suites.*

All PTS Tests *pts/all*

This is a collection of all test profiles found within the specified OpenBenchmarking.org repository.

Installed Tests *pts/installed*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository that are already installed on the system under test.

Linux Operating System Tests *pts/linux*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the Linux Operating System.

Solaris Operating System Tests *pts/solaris*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the Solaris Operating System.

BSD Operating System Tests *pts/bsd*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the BSD Operating System.

Phoronix Test Suite v8.0.0

Test Client Documentation

MacOSX Operating System Tests *pts/macosx*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the MacOSX Operating System.

Windows Operating System Tests *pts/windows*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the Windows Operating System.

Hurd Operating System Tests *pts/hurd*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the Hurd Operating System.

System Subsystem Tests *pts/system*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the System sub-system.

Processor Subsystem Tests *pts/processor*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the Processor sub-system.

Graphics Subsystem Tests *pts/graphics*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the Graphics sub-system.

Other Subsystem Tests *pts/other*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the Other sub-system.

Utility Tests *pts/utility*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Utility software test.

Simulator Tests *pts/simulator*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Simulator software test.

Phoronix Test Suite v8.0.0

Test Client Documentation

Scientific Tests *pts/scientific*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Scientific software test.

Benchmark Tests *pts/benchmark*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Benchmark software test.

Application Tests *pts/application*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Application software test.

Game Tests *pts/game*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Game software test.

Smp Tests *pts/smp*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing smp.

Cuda Tests *pts/cuda*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing cuda.

Python Tests *pts/python*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing python.

Go Tests *pts/go*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing go.

Mpi Tests *pts/mpi*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing mpi.

Phoronix Test Suite v8.0.0

Test Client Documentation

Openmp Tests *pts/openmp*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing openmp.

Vdpau Tests *pts/vdpau*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing vdpau.

Video Tests *pts/video*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing video.

Responsiveness Tests *pts/responsiveness*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing responsiveness.

Openmpi Tests *pts/openmpi*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing openmpi.

All SYSTEM Tests *system/all*

This is a collection of all test profiles found within the specified OpenBenchmarking.org repository.

Installed Tests *system/installed*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository that are already installed on the system under test.

Linux Operating System Tests *system/linux*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the Linux Operating System.

Solaris Operating System Tests *system/solaris*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the Solaris Operating System.

BSD Operating System Tests *system/bsd*

Phoronix Test Suite v8.0.0

Test Client Documentation

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the BSD Operating System.

MacOSX Operating System Tests *system/macosx*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the MacOSX Operating System.

Windows Operating System Tests *system/windows*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the Windows Operating System.

Hurd Operating System Tests *system/hurd*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being compatible with the Hurd Operating System.

System Subsystem Tests *system/system*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the System sub-system.

Processor Subsystem Tests *system/processor*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the Processor sub-system.

Disk Subsystem Tests *system/disk*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the Disk sub-system.

Graphics Subsystem Tests *system/graphics*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the Graphics sub-system.

Memory Subsystem Tests *system/memory*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the Memory sub-system.

Network Subsystem Tests *system/network*

Phoronix Test Suite v8.0.0

Test Client Documentation

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the Network sub-system.

Other Subsystem Tests *system/other*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a test of the Other sub-system.

Utility Tests *system/utility*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Utility software test.

Simulator Tests *system/simulator*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Simulator software test.

Scientific Tests *system/scientific*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Scientific software test.

Benchmark Tests *system/benchmark*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Benchmark software test.

Application Tests *system/application*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Application software test.

Game Tests *system/game*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified as being a Game software test.

Opencl Tests *system/opencl*

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing opencl.

Cuda Tests *system/cuda*

Phoronix Test Suite v8.0.0

Test Client Documentation

This is a collection of test profiles found within the specified OpenBenchmarking.org repository where the test profile is specified via an internal tag as testing cuda.

Component Testing

Compiler Testing & Masking

A majority of the test profiles provided by OpenBenchmarking.org to the Phoronix Test Suite are source-based tests. Relying upon the upstream source-code for each program under test allows for the tests to be easily brought to new platforms and architectures, avoids any out-of-tree / non-default packaging differences by different distributions and operating systems, and to allow the Phoronix Test Suite user to easily test new compilers and/or compiler options. For the source-based tests, the Phoronix Test Suite relies upon a compiler (e.g. GCC, LLVM/Clang, Sun Studio, Open64, et al) being present on the system under test. The Phoronix Test Suite does respect *CC/CXX* environment variables and test profiles are expected to honor *CFLAGS/CXXFLAGS* and other compiler settings.

As of Phoronix Test Suite 3.8, a compiler masking method is utilized for logging compiler options and other settings with each test profile installation. Prior to installing an open-source test, the Phoronix Test Suite determines the intended compiler to be used based upon the pre-set environment variables or the pre-set compiler(s) within the *PATH* environment variable. The Phoronix Test Suite then masks the compiler to ensure that any options/flags submitted to the compiler are first passed through pts-core so that they can be logged for later use, then is linked to the original, intended compiler. Additionally, other compiler binary names of the same type are blacklisted to prevent their un-intended use (i.e. if a test profile has hard-coded *gcc* in its build script, but *clang* is set as the compiler via *CC*, a sym-link will automatically be created from *gcc* to the masked *clang* for the duration of its test installation).

The logged compiler data is then used by the Phoronix Test Suite following the test execution process for automated result analysis. If there is a detected change in compiler settings, the differences are automatically reported to the test result graphs. Additionally, key compiler information (e.g. the compiler optimization level and key libraries that were linked at compile-time) is reported as a footnote on result graphs. The Phoronix Test Suite handles all of this in a fully automated manner; test profiles require no changes to take advantage of these compiler-reporting capabilities.

Separately, the Phoronix Test Suite attempts to automatically log the system compiler's build configuration (i.e. GCC's *gcc -v* "Configured with") output. If the Phoronix Test Suite detects there is a compiler build configuration change between test runs in a result file, it will report each compiler's build configuration to the system information table within the results viewer. Reported to the table is a reduced view of the build configuration options, with less relevant items being stripped away from this view to reduce verbosity. Upon clicking the text, the raw compiler information output can be viewed in full.

Disk / File-System Testing

By default tests are installed to *~/.phoronix-test-suite/installed-tests/*. However, the location can be updated from *~/.phoronix-test-suite/user-config.xml* or dynamically via the *PTS_TEST_INSTALL_ROOT_PATH* environment variable.

Phoronix Test Suite v8.0.0

Test Client Documentation

When any disk tests are executed, the Phoronix Test Suite will attempt to log the mount options and scheduler of the disk/partition being used for testing. This information is subsequently displayed within the system information table. If the scheduler / mount options are maintained the same throughout all test runs, only a single line is displayed otherwise the options for each test run will be automatically displayed. The file-system in use is always captured and shown in the system information table.

Phoronix Test Suite

Phoronix Test Suite v8.0.0

Test Client Documentation

Phoromatic

Overview

The Phoronix Test Suite now features its own built-in Phoromatic server and client implementations. See the *phoromatic.pdf* / *phoromatic.html* documentation for more information on this advanced capability.

Creating Test Profiles

Creating A New Phoronix Test Suite / OpenBenchmarking.org Test Profile

A test profile itself consists of a main XML file containing the test's meta-data (*test-definition.xml*), an XML file if needed to specify required files, download URLs and their MD5/SHA256 hashes and file sizes (*downloads.xml*), and the *install.sh* script for installing the test and generating the run-script for execution by the Phoronix Test Suite at run-time. There are also other potential files like *pre.sh*, *interim.sh*, and *post.sh* for execution at pre-run, interim-run, and post-run stages by the Phoronix Test Suite. With the script files comprising test profiles, there is also the option of post-fixing them with e.g. *_linux* or *_windows* if wishing to supply different script files based upon the operating system being run during testing, if the setup steps may differ greatly based upon MacOS/Windows/Linux/BSD/Solaris platforms.

Traditionally the easiest way to learn/understand test profile development has been by looking at looking at a basic test like *c-ray*, *tesseract* or the hundreds of others that are publicly available. Simply run for example *phoronix-test-suite benchmark tesseract* and then look at the test profile's contents either via the default locations in *~/.phoronix-test-suite/test-profiles* or */var/lib/phoronix-test-suite/test-profiles/* if running as root on non-Windows platforms.

Versioning Mandate

The Phoronix Test Suite / OpenBenchmarking.org philosophy *mandates that with any change, a new version of the test profile be tagged*. This is done for reproducibility and being able to ensure the exact test profile state when a set of tests are conducted. Test profiles are versioned in a **test-X.Y.Z** format where *X* or *Y* are bumped whenever a change is made to the test profile that makes the results incomparable to a previous version of the tests (e.g. updating against a new upstream code-base, adjusting the parameters to what is benchmarked, etc). The Phoronix Test Suite then knows not to attempt any comparisons with an incompatible version difference or to go and fetch that specific version of the package. The *Z* is bumped when just making non-important changes such as just updating the test's meta-data, changing download URLs, etc. This is easy to enforce with the OpenBenchmarking.org infrastructure rather than having to worry about non-human-friendly Git hashes as test profile versions or having to create Git tags after every commit. The test profile version can optionally be specified when running a test, e.g. *phoronix-test-suite benchmark scimark2-1.2.1* instead of *phoronix-test-suite benchmark scimark2*, which would by default choose the latest available test profile version from OpenBenchmarking.org or the latest version on any local Phoromatic Server. The test profile versions are also always written out as part of the Phoronix Test Suite result XML data.

Writing A Sample Program

Writing a test profile for the Phoronix Test Suite is a relatively quick and easy process for anyone familiar with common Linux commands and the basics of XML. To help you understand the design of

Phoronix Test Suite v8.0.0

Test Client Documentation

the Phoronix Test Suite, this guide covers the steps needed to write a testing profile for a very simple application.

The first step in the profile writing process is to, well, have a piece of software you'd like to use with the Phoronix Test Suite. This software can be closed-source or open-source and be virtually anything as long as it is compatible with an operating system that is supported by the Phoronix Test Suite.

For this guide, the piece of software being used for demonstration is just a simple C++ program that calculates Pi to 8,765,4321 digits using the Leibniz formula. Below is this sample piece of software intended just for demonstration purposes.

```
#include <iostream>
#include <math.h>
int main()
{
double pi = 0;
 for(long int i = 1; i <= 87654321; i++)
pi += (double) pow(-1, i + 1) / (2 * i - 1);
 pi *= 4;
std::cout << "Done Calculating Pi..." << endl;
return 0;
}
```

The first step in the actual profile writing process is to name it. If you're looking to ultimately push this profile to be included in the Phoronix Test Suite, its name must be all lower case and consist of just alpha-numeric characters, but can contain dashes (-). A more advanced test profile capability is operating system prefixes, and if using those there is an underscore separating the prefix from the normal profile name. For this sample profile, we're calling it *sample-program* and the file-name would be *sample-program/test-definition.xml*. Our (very basic) profile is showcased below.

```
<PhoronixTestSuite>
Â Â Â Â Â <TestProfile>
Â Â Â Â Â Â Â Â Â Â <Version>1.1.0</Version>
Â Â Â Â Â Â Â Â Â Â <TestType>Processor</TestType>
Â Â Â Â Â Â Â Â Â Â <SoftwareType>Utility</SoftwareType>
Â Â Â Â Â Â Â Â Â Â <License>FREE</License>
Â Â Â Â Â Â Â Â Â Â <Status>PRIVATE</Status>
Â Â Â Â Â Â Â Â Â Â <Maintainer>Phoronix Media</Maintainer>
Â Â Â Â Â </TestProfile>
Â Â Â Â Â <TestInformation>
Â Â Â Â Â Â Â Â Â Â <Title>Sample Pi Program</Title>
Â Â Â Â Â Â Â Â Â Â <TimesToRun>3</TimesToRun>
Â Â Â Â Â Â Â Â Â Â <ResultScale>Seconds</ResultScale>
Â Â Â Â Â Â Â Â Â Â <Proportion>LIB</Proportion>
Â Â Â Â Â Â Â Â Â Â <Description>A simple C++ program that calculates Pi to 8,765,4321 digits using
```

the Leibniz formula. This test can be used for showcasing how to write a basic test profile.</Description>
Â Â Â Â Â Â Â Â Â Â <ExternalDependencies>build-utilities</ExternalDependencies>
Â Â Â Â Â </TestInformation>
</PhoronixTestSuite>

This XML profile is what interfaces with the Phoronix Test Suite and provides all the needed information about the test as well as other attributes. For a complete listing of all the supported profile options, look at the specification files in the documentation folder. In the case of *sample-program*, it lets the Phoronix Test Suite know that it's composed of free software, is designed to test the processor, is intended for private use only, and this profile is maintained by Phoronix Media. In addition, it tells the Phoronix Test Suite to execute this program three times and as no result quantifier is set, the average of the three runs will be taken. This profile also tells the Phoronix Test Suite that the generic *build-utilities* package is needed, which will attempt to ensure that default system C/C++ compiler and the standard development utilities/libraries are installed on your Linux distribution. This is needed as the C++ source-code will need to be built from source.

The next step is to write the *install.sh* file, which once called by the Phoronix Test Suite is intended to install the test locally for benchmarking purposes. The *install.sh* file is technically optional, but is generally used by all tests. Note: The first argument supplied to the install script is the directory that the test needs to be installed to. The *install.sh* file (in our instance) is to be placed inside *test-profiles/sample-program*. Below is the *install.sh* for the *sample-program*.

```
#!/bin/sh
tar -xjf sample-pi-program-1.tar.bz2
g++ sample-pi-program.cpp -o sample-pi-program
echo "#!/bin/sh
./sample-pi-program 2>&1
" > sample-program
chmod +x sample-program
```

This install file builds the code with GCC, and then creates a small script that is run by the Phoronix Test Suite. Where does the source-code come into play? Well, it needs to be downloaded now from a web server. The Phoronix Test Suite has built-in support for managing downloads from multiple servers in a random over, fall-back support if one mirror is done, and verification of MD5 check-sums. Below is the *downloads.xml* file for *sample-program* that covers all of this.

```
<PhoronixTestSuite>
Â Â Â Â Â <Downloads>
Â Â Â Â Â Â Â Â Â Â <Package>
Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â <URL>http://www.phoronix-test-suite.com/benchmark-files/sample-pi-program.cpp</URL>
Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â Â <MD5>e90fb790df8d1544696a1439c9b5bd8d</MD5>
Â Â Â Â Â Â Â Â Â Â </Package>
Â Â Â Â Â </Downloads>
```

```
</PhoronixTestSuite>
```

The final step in the profile writing process is to write a parser to strip all information but the reported result from the standard output or *$LOG_FILE*. In the case of a test profile just measuring how long it takes to run, it is as simple as a *results-definition.xml* looking like:

```
<?xml version="1.0"?>
<PhoronixTestSuite>
  <SystemMonitor>
    <Sensor>sys.time</Sensor>
  </SystemMonitor>
</PhoronixTestSuite>
```

After that, with all the files in their correct locations, just run: *phoronix-test-suite benchmark sample-program*. The Phoronix Test Suite should now handle the rest by installing the test, running the test, and recording the results (if you so choose). There is no additional work that needs to be done for the results to be recorded in the results viewer or even reporting the results to OpenBenchmarking.org. An up-to-date version of this test profile can be run via *phoronix-test-suite benchmark sample-program* and then by looking at the test profile source via *~/.phoronix-test-suite/test-profiles/pts/sample-program** or within */var/lib/phoronix-test-suite/test-profiles/pts/* if running as root.

Phoronix Test Suite v8.0.0

Test Client Documentation

Development Credits

The Phoronix Test Suite is based upon the extensive testing and internal tools developed by **Phoronix.com** since 2004 along with support from leading tier-one computer hardware and software vendors. The principal architects of the Phoronix Test Suite are **Michael Larabel** and Matthew Tippett. The phoronix-test-suite, pts_Graph, Phoromatic, Phodevi, tandem_Xml, and nye_Xml are some of the related open-source projects provided by **Phoronix Media**.