Magento

an ebay inc company

Magento Performance Toolkit (Beta) User Guide

Magento Enterprise Edition 1.12
Magento Enterprise Edition 1.13
Magento Enterprise Edition 1.14

October 9th, 2014

Contents

About This Document....1

What Is the Magento Performance Toolkit....1

A Quick Overview....1

Database Generation Script....2

Script Location....2

Parameters....2

Database Profiles....2

System Requirements....3

Generate.php parameters....3

Example Output....3

Expected Execution Time....5

Benchmarking with jMeter....6

User Behavior Distribution....6

Installing jMeter....6

Environment....6

jMeter....6

jMeter Plugins....6

Usage....6

Location....6

Options....7

Run Using Console....8

Run Using jMeter GUI....8

Reports....8

Appendix A: Magento Performance Toolkit Files....i

About This Document

This guide describes the usage of the Magento Performance Toolkit for Enterprise Edition.

What Is the Magento Performance Toolkit

The Magento Performance Toolkit is a set of automated tools that enables you to quickly and consistently measure the application performance of Magento using Apache jMeter. The toolkit includes a PHP script for generating a test database and a JMX script for the performance measurement.

The most recent version of the Magento Performance Toolkit for 1.x is published at https://github.com/magento/magento-performance-toolkit.

The most recent version of the Magento Performance Toolkit for 2.x is published at https://github.com/magento/magento2 and included in the /dev/tools/performance_toolkit folder.

A Quick Overview

To use the Magento Performance Toolkit effectively you need some expertise with installing Magento, running PHP scripts in console, and using jMeter.

Test results do not guarantee system performance in a production environment. The toolkit does help identify performance bottlenecks in code. Also, software or hardware environment changes can be quickly evaluated by analyzing the test results.

The steps described below summarize the tasks that you need to complete to test the performance of a Magento instance using this toolkit. More details can be found in the detail sections of this guide.

To measure the performance of a Magento instance:

1. Install a fresh Magento instance on your test server.
2. Install jMeter on your load generator system.
3. Install the jMeter Plugins on your load generator system.
4. Copy Performance Toolkit files to `<magento_dir>/dev/tools/performance_toolkit`
5. Run the database generation script (generate.php) to configure the Magento instance and populate the database using one of the pre-defined profile sizes (e.g., small, medium, large, or extra large). For more details see System Requirements

The following table contains system requirements for generating each of the pre-defined database profiles.

Profile	PHP Memory limit	MySQL max_allowed_packet
Small	512m	8m
Medium	1G	8m
Large	8G	16m
extra large	16G	16m

Please consider these numbers and memory consumption dependency trend when creating a custom profile configuration.

6. Generate.php.
7. Run the jMeter scenario. For more details see Benchmarking with jMeter.

8. View the generated reports. For more details see Reports.

Database Generation Script

The database generation script, `generate.php`, allows you to build a database that can be used for testing. You use script parameters to specify the size of the database to be generated and a temporary file path.

Script Location

The database generation script is located at `<magento_dir>/dev/tools/performance_toolkit/generate.php`.

To see details about the script's file structure please refer to Appendix A: Magento Performance Toolkit Files.

Parameters

When running the `generate.php`, you can configure the following parameters:

Parameter	Required	Description
`profile`	yes	Path to a database profile.
`tmp_dir`	no	Path to the directory for temporary files, with current user write access. If not specified, the current working directory is used.

Database Profiles

You can use one of the four pre-defined database profiles, or create your own:

- `small.xml`
- `medium.xml`
- `large.xml`
- `extra_large.xml`

Each profile contains the information about the entities that will be generated.

For example:

```
/dev/tools/performance_toolkit/profiles/small.xml
```

```
<config>
    <profile>
        <websites>1</websites>
        <!-- Number of websites to generate -->
        <store_groups>1</store_groups>
        <!--Number of stores-->
        <store_views>1</store_views>
        <!-- Number of store views -->
        <simple_products>800</simple_products>
        <!-- Simple products count -->
        <configurable_products>50</configurable_products>
        <!--Configurable products count (each configurable has 3 simple products
as options, that are not displayed individually in catalog) -->
        <categories>30</categories>
        <!-- Number of categories to generate -->
        <categories_nesting_level>3</categories_nesting_level>
        <!-- Nesting level for categories -->
        <catalog_price_rules>10</catalog_price_rules>
        <!-- Number os catalog price rules -->
        <catalog_target_rules>2</catalog_target_rules>
        <!-- NUmber of catalog target rules -->
        <cart_price_rules>10</cart_price_rules>
        <!-- Number of shopping cart price rules -->
        <cart_price_rules_floor>2</cart_price_rules_floor>
        <!-- The price rule condition: minimum products amount in shopping cart
for price rule to be aaplied -->
        <customers>20</customers>
        <!-- Number of customers to generate -->
    </profile>
</config>
```

System Requirements

The following table contains system requirements for generating each of the pre-defined database profiles.

Profile	PHP Memory limit	MySQL max_allowed_packet
Small	512m	8m
Medium	1G	8m
Large	8G	16m
extra large	16G	16m

Please consider these numbers and memory consumption dependency trend when creating a custom profile configuration.

Generate.php parameters

To run the database generation script, execute the following:

```
php -f generate.php -- --profile=<path_to_profile>
```

or

```
php -f generate.php -- --profile=<path_to_profile> --
tmp_dir=<path_to_tmp_dir>
```

Example Output

```
$ cd /home/www/magento/dev/tools/performance_toolkit
$ php -f generate.php -- --profile=profiles/small.xml --
tmp_dir=/home/www/magento/var
```

Output during generation:

```
 |- Websites: 1
 |- Store Groups: 1
 |- Store Views: 1
 |- Simple Products: 800
 |- Configurable Products: 50
 |- Categories: 30
 |- Catalog Price Rules: 10
 |- Catalog Target Rules: 2
 |- Cart Price Rules: 10
 |- Regitered Customers: 20
Generating stores, store views and websites...  done in 00:00:00
Generating categories...  done in 00:00:02
Generating simple products...  done in 00:00:06
Generating EAV variations for configurable products...  done in 00:00:00
Generating configurable products...  done in 00:00:04
Generating customer accounts...  done in 00:00:00
Generating shopping cart price rules...  done in 00:00:00
Generating catalog price rules...  done in 00:00:00
Generating tax rates...  done in 00:06:06
Generating catalog target rules...  done in 00:00:04
Disabling form key usage for admin...  done in 00:00:00
Enabling FlatRate shipping method...  done in 00:00:00
php -f /home/www/magento/dev/tools/performance_toolkit/indexer.php --
reindexall 2>&1
Product Attributes index was rebuilt successfully in 00:00:00
Product Prices index was rebuilt successfully in 00:00:00
Catalog URL Rewrites index was rebuilt successfully in 00:00:09
```

```
Product Flat Data index was rebuilt successfully in 00:00:04
Category Flat Data index was rebuilt successfully in 00:00:00
Category Products index was rebuilt successfully in 00:00:00
Catalog Search Index index was rebuilt successfully in 00:00:01
Stock Status index was rebuilt successfully in 00:00:00
Tag Aggregation Data index was rebuilt successfully in 00:00:00
Total execution time: 00:06:45
```

Expected Execution Time

Small profile generation takes 5-6 minutes and medium profile generation takes 16-17 minutes on the following described test system:

Test Environment (Hardware)

CPU, Intel(R) Xeon(R) CPU E5620, 2.40GHz x 16

RAM, 72G

Test Environment (Software)

MySQL 5.6.20

PHP 5.4.30

memory_limit in php.ini = 4G

xDebug = enabled

Benchmarking with jMeter

The jMeter script (`benchmark.jmx`) provided in the Magento Performance Toolkit emulates real customer usage scenarios on a Magento store.

User Behavior Distribution

User sessions are distributed among scenario threads according to the values you specify in the script parameters.

By default the distribution is as follows:

- Browsing, adding items to a cart and abandoning the cart: 62%
- Just browsing: 30%
- Browsing, adding items to a cart and checking out as a guest: 4%
- Browsing, adding items to a cart and checking out as a registered customer: 4%.

The number of simultaneously running threads depends on two parameters: the test duration time, and the number of users or orders.

The generated report contains statistics on average response time on each step, like opening category page, adding product to cart, performing checkout, and so on, and average errors percentage and count on each step.

Installing jMeter

Environment

We strongly recommended you to execute the jMeter scenarios from a load generator that is a separate system than the Magento server. jMeter consumes system resources freely, so it may affect your server performance resulting in irrelevant statistics if it is run from the same system. The scenarios can be executed in GUI or console mode from any UNIX, Mac or Windows machine. Note, that jMeter requires Java Runtime Environment, please install Java 6 or later.

jMeter

You can download jMeter from the Source section at http://jmeter.apache.org/download_jmeter.cgi.

jMeter Plugins

You can download jMeter Plugins from the Download section at http://jmeter-plugins.org.

Usage

You can run the jMeter script in console or use the jMeter GUI.

The number of simultaneously running threads during script execution depends on two main parameters: the test duration time (`ramp_period`), and the number of users (`users`) or orders (`orders`). For example, to test a Magento store with 1000 users per hour, you can specify either `users=1000`, and `ramp_period=3600`, or `users=100`, and `ramp period=360`. To test a Magento store performance with 500 orders per hour with average distribution, specify `orders=500` and `ramp_period=360`.

There are more details about script options in the following Options section.

Location

Magento jMeter script is located at

```
<magento_dir>/dev/tools/performance_toolkit/benchmark.jmx
```

Options

The scenario accepts the following options:

Option	Required	Description
Host	Yes	Fully qualified host name or IP address of the server on which Magento application is running.
base_path	Yes, if not empty	Path relative to the web server's docroot for the Magento application. It must start and end with forward slash sign ("/"). For example if you installed Magento application at `http://localhost/magento/`, the base path is `/magento`/.
Users	No	Number of concurrent users. Default (and recommended) value is 100. Minimum allowed is 10. Maximum depends on your environment.
Orders	No	Number of orders. If it is set, `users` value is recalculated as follows: `users` = `orders` * 100/(`guest_checkout_percent` + `customer_checkout_percent`) In the default distribution only 8% perform orders, so specifying `orders` = 100 will run scenario with 100 * 100 / 8 = 1250 users.
ramp_period	No	The duration of test execution, in seconds. Default value is 600 seconds.
report_save_path	No	Path to the directory where reports are saved. If not specified, reports are saved in the current working directory.
view_product_add_to_cart_percent	No	Percentage of users who are browsing, adding items to art and abandoning the cart. Default value is 62%.
view_catalog_percent	No	Percentage of users who are just browsing the site. Default value is 30%.
guest_checkout_percent	No	Percentage of users who are browsing, adding items to a cart and checking out as guests. Default value is 4%.
customer_checkout_percent	No	Percentage of users who are browsing, adding items to a cart and checking out as registered customers. Default value is 4%.
Loops	No	Number of times the test case for

		each thread is performed. Default value is 1.

Run Using Console

When running the jMeter scenario in console, all options must be prefixed with "J”. For example, for specifying the number of users (the `users` option), you should enter `–Jusers`.

Example:

```
> cd /directory_of_jMeter/bin/

> jmeter -n -t /path_to_benchmark_file/benchmark.jmx -
Jhost=magento.instance.com -Jbase_path=/ -Jusers=100 -Jramp_period=300 -
Jreport_save_path=./
```

When the `–n` option is specified the scenario runs in non-GUI mode.

Testing results are saved as `.log` files in the directory specified in `report_save_path`. You can open them using Jmeter GUI or like CSV files in any other appropriate application.

Run Using jMeter GUI

To run the scenario using the jMeter GUI:

1. Open the `jMeter/bin` directory, and run `jmeter.bat`.
2. Click File > Open (Ctrl+O).
3. Select `benchmark.jmx,` or drag and drop it to the opened GUI window. You can configure script options in the User Defined Variables section on the Test Toolkit tab.
4. Click Start (green arrow in the top menu).

After running the script in GUI you can choose the necessary report in the left panel.

Reports

Using the jMeter GUI you can view a number of different reports. Please use the following ones:

- Summary Report.
  This report contains aggregated information about threads. The report file name is `{report_save_path}/summary-report.log`. More details can be found at http://jmeter.apache.org/usermanual/component_reference.html#Summary_Report
- Detailed URLs report.
  This report contains information about each request that was processed. Please note that URLs are displayed only in the report file, and are not displayed in GUI. The report file name is `{report_save_path}/detailed-urls-report.log` (can be opened in CSV format).
  More details at
  http://jmeter.apache.org/usermanual/component_reference.html#View_Results_in_Table

Appendix A: Magento Performance Toolkit Files

The Magento Performance Toolkit file structure:

```
_<magento_dir>/dev/tools/performance_toolkit

      |_/fixtures

      |_/ framework

      |_/ profiles

      |_/ generate.php
```

Fixtures (files to generate Magento DB data):

```
_<magento_dir>/dev/tools/performance_toolkit/fixtures

      |_/cart_price_rules.php

      |_catalog_price_rules.php

      |_catalog_target_rules.php

      |_categories.php

      |_configurable_products.php

      |_customers.php

      |_disable_form_key_usage.php

      |_eav_variations.php

      |_shipping_flatrate_enabled.php

      |_simple_products.php

      |_stores.php

      |_tax_rates.php
```

Framework files:

```
_<magento_dir>/dev/tools/performance_toolkit/framework

      |_/Magento

      |_/tests

      |_/bootstrap.php
```

Fixture settings for different database size:

```
_<magento_dir>/dev/tools/performance_toolkit/profiles

      |_/extra_large.xml

      |_/large.xml

      |_/medium.xml

      |_/small.xml
```