# Migration from Magento 1 to Magento 2

Version 1.1|09-May-19

# Pre Requisites

* Make sure that the Data Migration Tool has a network access to connect to Magento 1 and Magento 2 databases. Open the corresponding ports in your firewall.
* Stop all activities in the Magento 1.x Admin Panel, except for order management, such as shipping, creating invoice, credit memos, etc (the list of allowed activities can be extended by adjusting settings of the Delta mode in the Data Migration Tool). Note: such activities must not be resumed until your Magento 2 store goes live.
* Stop all Magento 1.x cron jobs.
* Copy your Magento 1.x media files to Magento 2.x.

# Upgrade Activities

Platform Upgrade includes following activities and each of these activities are detailed out in the below sections.

* Data Migration
* Theme Migration
* Custom Extension Migration (Third Party / Community / Local)

# Data Migration

## Install the Data Migration Tool

Data migration tool is provided by Magento for making Data Migration activity simpler and automated.

1. To install the migration tool from GitHub, use the following steps:
   1. Log in to your Magento 2 server as a user with privileges to write to the Magento 2 file system or switch to the web server user.
   2. Go to Magento 2 root directory.
   3. Enter the following commands:

composer config repositories.data-migration-tool git <https://github.com/magento/data-> migration-tool

composer require magento/data-migration-tool:<version>

where <version> is release version (e.g. 2.3.0)

* 1. Wait while dependencies are updated.

1. Install the tool from repo.magento.com

To install the Data Migration Tool, you must update composer.json in the Magento root installation directory to provide the location of the migration tool package.

To install the migration tool, you must:

1. Decide the version of magento/data-migration-tool.
2. Run the composer config and composer require commands to update composer.json.
3. When prompted, enter your authentication keys. Your public key is your username; your private key is your password.
4. To update composer.json:
5. Log in to your Magento server as the web server user or as a user with root privileges.
6. Change to your Magento installation directory.
7. Enter the following command to reference Magento packages in composer.json:

composer config repositories.magento composer <https://repo.magento.com>

Enter the following command to get the current version of the package:

composer require magento/data-migration-tool:<version>

where <version> is either an exact version or next significant release syntax.

1. Wait while dependencies are installed.
2. You have installed tool into empty Magento2 installation with Composer. It’s not an extension, but standalone shell app that requires Magento2 framework to work. It’s located in vendor/magento/data-migration-tool/
3. check config.xml file available in the following directory or not?
4. vendor/magento/data-migration-tool/etc/config.xml
5. If not available, then kindly create new config.xml file and copy all contents from config.xml.disc and paste in your custom created config.xml
6. On supported default Magento installations, minimum required would be to configure source(M1) and destination(M2) database connections in config.xml:
7. <source>
8. <database host="localhost" name="magento1\_db" user="root"/>
9. </source>
10. <destination>
11. <database host="localhost" name="magento2\_db" user="root"/>
12. </destination>

## Data Migration

There are 3 main commands/steps for the migration: settings, data and delta. This and their sub-steps can be seen in config.xml.

Go to command line arguments and type following commands for upgrade settings, data and delta.

php bin/magento migrate:data vendor/magento/data-migration-tool/etc/<<Your Magento 1.X edition folder name>>/<<Magento 1.X version folder>>/config.xml

Possible modes:

settings Migrate system configuration

data Main migration of data

delta Migrate the data is added into Magento after the main migration

## known issues with data migration

This section provided details of know issues with data migration and the corrective action.

# Theme Migration

* To upgrade Magento 1 theme to Magneto 2, you have to validate if Magento 1 theme is compatible with Magento 2 version or not.
* If Yes, then you can install the same theme in Magento 2. This will make the process of migration a lot easier for you.
* If No, then you can either create a custom theme in Magento 2 or buy and install a Magento 2 theme from Magento Marketplace.
* If you want to create new theme in Magento 2, then copy all css, js and images from Magento 1 skin/frontend/<packagename>/<themename>/<css/js/images> directory and paste it in the Magento 2 app/design/frontend/<packagename>/<themename>/web/<css/js/images> directory
* Include all css/js files in the Magento 2 app/design/frontend/<packagename>/<themename>/Magento\_Theme/layout/default\_head\_blocks.xml file.
* You cannot include all js directly in your respective directory, you need to define js by using following method.
* Create requirejs-config.js in your themes root folder, below is the demo file for requirejs-config.js.

var config = {

deps: [

"js/main",

],

map: {

'\*': {

'owlcarousel': 'js/owl.carousel.min',

'sticky' : 'js/jquery.sticky',

'tether' : 'js/tether.min',

'bootstrap': 'js/bootstrap.min'

}

},

"shim": {

"owlcarousel": ["jquery"],

"tether": ["jquery"],

"bootstrap": ["jquery"]

}

};

In aboves example I register main.js by calling it in the Deps. The path is without the .js extention, requirejs is handeling this on its own.

Your main.js should be located at web/js/ in your theme.

The content of main.js should be like this:

define([

"jquery",

"whatever lib you want use..."

],

function($) {

"use strict";

$(document).ready(function($){

console.log("DID IT! jQuery log complete.");

});

return;

});

* You can override templates according to your needs and you can make changes in that.

|  |  |
| --- | --- |
| MAGENTO 1.X | MAGENTO 2.X |
| * Template Files      * Skin Files. | * Template & Skin Files are both in app/design/frontend folder |

# Community Extension Migration

* Validateif Magento 1 extensions are compatible with Magento 2 version or not.
* If yes, then you can integrate the same extensions in Magento 2 or download that extension for Magento 2 and install it.
* You will have to manually create your custom modules and upgrade your Magento 1 3rd party extensions to their Magento 2 upgraded versions (if there's one available).
* Explain the Code Migration Utility.
* Explain how the extension will be before and after the migration tool is executed.

# Effort Comparison

This section will provide details of effort between Manual Migration vs Automated Migration.

Provide the project/client name with complexity.

Low, Medium and High Complexity.

|  |  |  |  |
| --- | --- | --- | --- |
| Client / Project | Complexity | Effort w/o automation | Effort with automation |
| UK / <https://www.apricotonline.co.uk/>  (Magento 2.X) | Medium |  |  |
| UK/ <https://www.sidedoor.co.uk/>  (Magento 1.X) | High | 6 months | 4 months |
| USA / <https://www.conns.com/>  (Migrated from 1.X to 2.X) | High | 10 months | 6 months |
| USA / <https://www.lymanproducts.com/>  (Magento 1.X) | Medium | 8 months | 6 months |
| USA / <https://asecondchanceresale.com/>  (Magento 1.X) | Medium | 6 months | 4 months |
| USA / <https://www.icpamerica.com/>  (Magento 2.X) | Low |  |  |
| India / <https://www.luckylips.in/>  (Magento 2.X) | High |  |  |
| India / <http://chachamuseum.com/>  (Magento 1.X) | Low | 2 months | 1.5 months |
| Germany / <https://smileoptic.de/>  (Magento 1.X) | High | 10 months | 6 months |