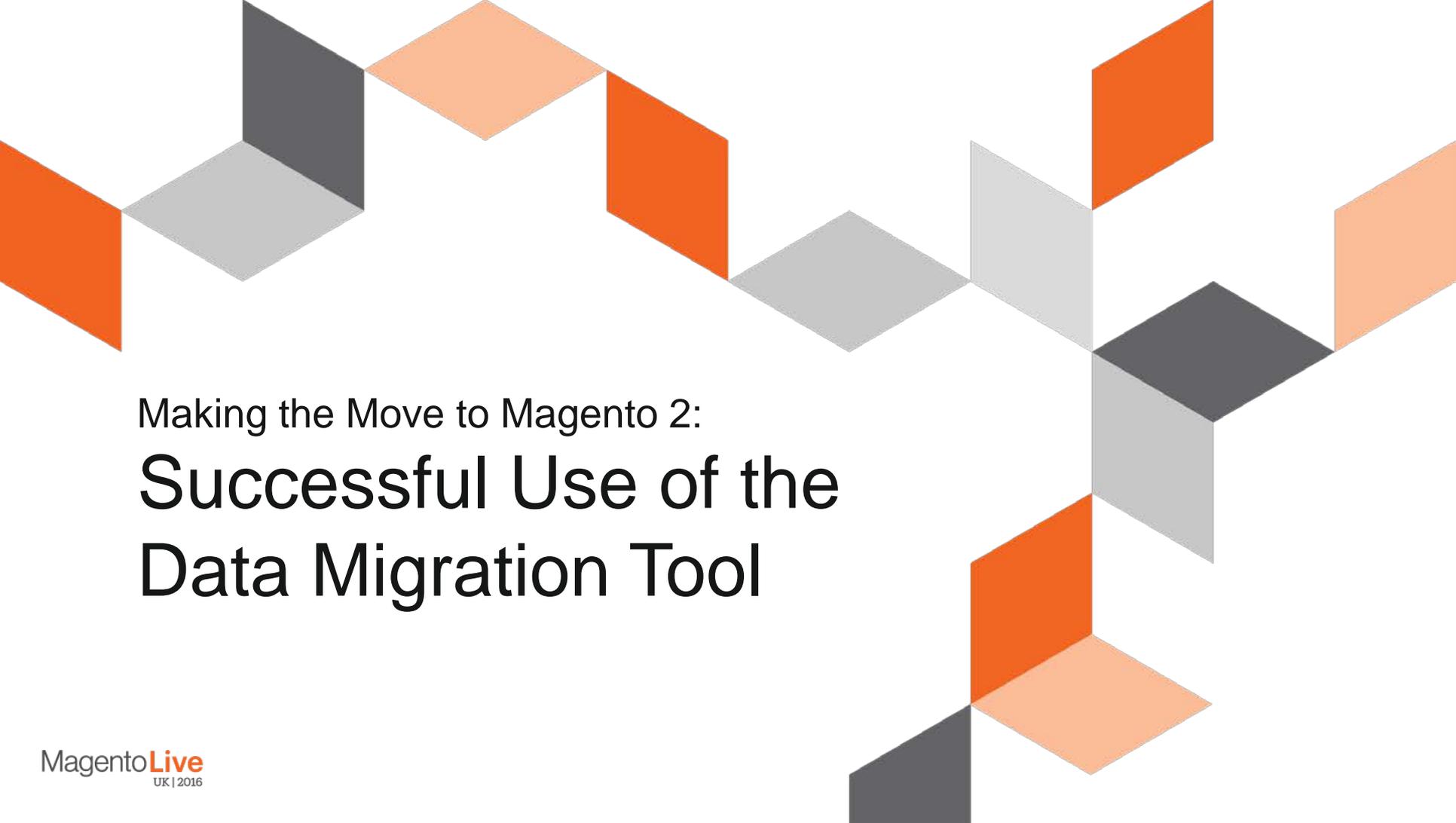




Magento**Live**
UK | 2016



Making the Move to Magento 2:
**Successful Use of the
Data Migration Tool**

Cost-Benefit Analysis

Cost of an Upgrade

- Magento: Treat upgrading to Magento 2 as if it were a re-platform
- Moving to Magento 2 is a big step, and migrating your data is difficult
- We estimate data migration takes between 4 and 12 weeks
- Cost depends on age and complexity of your database
- Expensive, but worth it

Risks of a Re-Platform

- Loss of search engine rankings
- Loss of customers
- Temporary drop in revenue
- A well planned migration of your data can mitigate all of the above

What Should be Migrated?

- Ideally: everything
- Realistically: attributes, customers, catalog, orders
- The more you migrate, the more expensive the migration
- Your data has value that you can leverage in Magento 2
- Loss of data incurs costs, both hidden and not

The Possible Value of Future Data

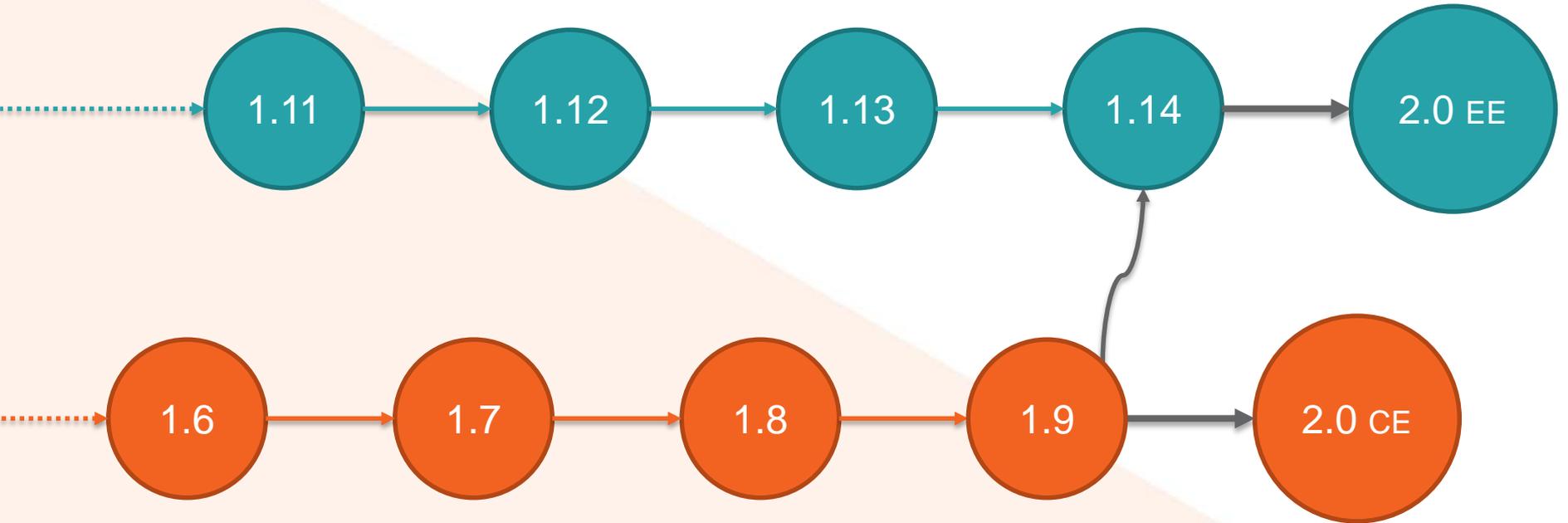
- Various tools can make use of historical data
- More data gives better insights and leveraging it can better target your marketing strategy
- Consider the opportunity cost incurred by not migrating your data

The Migration Process

Upgrading vs. Migrating

- Upgrading: 1.x to 1.y
- Migrating: 1 to 2
- Upgrading is easier than migrating
- Upgrade to the last M1 release, then migrate
- For CE to EE, upgrade to M1EE

Upgrading vs. Migrating



When to Migrate

- Work on the migration alongside the build
- Run the migration at the end of your project, just prior to launch
- Takes less than a day to run the full migration, probably much less
- Magento 1 sample data migrates in 6 minutes

Live Demonstration

Live Demonstration: Summary

- 1.7 CE, upgraded to 2.0 EE (via 1.9, 1.14)
- Start with a blank installation of Magento 2
- Start with sample data from 1.9
- Run bulk migration, then enter delta migration mode

Magento Data Migration Tool

Magento Data Migration Tool

- Support: 1.6 CE, 1.11 EE
- This is the initial installed version, not the current upgraded version
- Older versions are possible, but more work
- Comprehensively documented in the DevDocs, under “System Administrator”
- Tool installs as a Magento 2 module, through Composer
- Has three primary “modes”: Settings, Data, and Delta, which run in that order

Configuring the Migration Tool

- Uses XML for mapping configuration.
- Has mappings for all source versions in `vendor/magento/data-migration-tool/etc`
- Only default entities will be migrated out of the box
- It is relatively easy to add mappings for custom entities
- For anything you do not wish to migrate, you have to add an `<ignore>` tag, for a column, or for a table.

Stop Magento 1

1

- Magento: stop activity in the admin and disable cron, but keep your store open
- If you want to be safe, you can put your entire site into maintenance mode

2

3

4

5

Migrate Settings

1

- Migrates stores, store configuration, and sequence tables for generating order IDs and such

2

- Very few changes between Magento 1 and Magento 2
- Usually quite straightforward, and takes a few seconds

3

4

5

Migrate the Rest of Your Data

1

- Maps all of your old data to the new database
- Run this on a copy of your Magento 1 database

2

- Will find most of your data issues
- Takes the most time to run

3

- If you are not migrating your orders, you need to migrate the sequence tables by hand
- Sets up triggers for the delta mode

4

5

Turn on Delta Migration

1

- You can skip this step if you disabled your store
- Maps changes on Magento 1 store to the Magento 2 store in real time after the bulk migration

2

- For the time after deployment and before release (testing)

3

- When you release, all of your data is up to date
- Only maps data that customers can create: customers, orders; not catalog changes

4

- Designed to be run for only a brief time

5

Test and Deploy

1

- Test your Magento 2 site to see how it is
- Flip your DNS to complete the deployment
- Take a deep breath. You're on Magento 2

2

3

4

5



Tips for a Successful Migration

- You must understand Magento 1
- Script your progress
- Always be dumping
- Check your mapping files into source control
- Disable password rehash during development
- Run the migration on PHP 7
- Have source and destination databases on the same installation of MySQL

Sampling Your Data

- It is much faster to use a sample of your data during development
- 10% of your data will contain 90% of your data's issues
- Don't remove any attributes, just the volume of the data
- Remove 90% of your simple products, configurable products, etc.
- Remove all the empty categories
- Remove 90% of your customers, orders, etc.

Cleaning Your Data

- Fix anything that makes The Data Migration Tool throw an exception
- Remove attributes you no longer use, and attributes with third party attribute models
- DELETE FROM all logs, and all quotes
- Fill any optional attributes that are required in Magento 2 with dummy data



Marcin Szterling, Max Bucknell

Developers, Redbox Digital

@php4u

@maxbucknell