Introduction to Magento 2
Vijay Golani
Magento U Authorized Trainer
Vice President of Development, Wagento
Platform & Architecture
Platform & Architecture

Magento 2 platform:

- A flexible, open source commerce platform and content management system
- Written in PHP; leverages elements of Zend Framework and MVC architecture
- Extremely configurable
Platform & Architecture | Goals

- Streamline Customization Process
- Easier Upgrades
- Update Technology
- Simplify Integrations
- Improved Performance & Scalability
- High Quality, Tested Code
## Platform & Architecture | Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Location</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration</td>
<td>app/etc/</td>
<td>env.php, config.php</td>
</tr>
<tr>
<td>Framework</td>
<td>lib/internal/Magento/Framework</td>
<td>Framework classes</td>
</tr>
<tr>
<td>Modules</td>
<td>app/code/Magento</td>
<td>Business logic</td>
</tr>
<tr>
<td>Command-line Tool</td>
<td>bin/magento</td>
<td>Important utility; Run &quot;php bin/magento list&quot; to see all available commands.</td>
</tr>
<tr>
<td>Themes</td>
<td>app/design/</td>
<td>Contains static files that belong to a theme.</td>
</tr>
<tr>
<td>Dev Tools</td>
<td>dev</td>
<td>Various dev tools, like testing framework, sample data installer, ...</td>
</tr>
</tbody>
</table>
File Types

- Configuration files (XML files, and a few PHP files)
- PHP classes
- Layout instructions (*.xml files)
- Templates (*.phtml files)
- JavaScript modules (*.js files)
- JavaScript templates (*.html files)
- Static assets (CSS, images, …)
File Types

• Global config files:
  – In the app/etc folder
  – di.xml, env.php, config.php, vendor_path.php
• Core and custom modules’ config files located in the module’s etc folder
• Theme configuration files, …
Magento 2 PHP Classes

- Model / resource model / collection classes
- API interfaces
- Controllers
- Blocks
- Observers (continued on next slide)
Magento 2 PHP Classes

- Plugins
- Helpers
- Setup / upgrade scripts
- Ui components
- Other…
Request Processing

Your Class
Different ways to enable custom code in Magento 2:

- Create and register a module
- Run `bin/magento setup:upgrade` to execute setup / upgrade scripts
- Modify core classes by creating a plugin
- Create observers
- Add your class to the core’s class array in constructor
- Controllers
- System configuration
File System
File System | Root Folders

- app
- bin
- dev
- lib
- vendor
- pub
- setup
- var
File System | app Folder Contents

- app/etc
- app/code
- app/design
- app/i18n
- app/bootstrap.php
- app/autoload.php
- app/functions.php
File System | Core Source Code

```
app/code/Magento
```

- Backend
- Bundle
- Catalog
- Checkout
- Cms
- Core
- Customer
- Eav
- Index
- Indexer
- Install
- Module
- Payment
- Sales
- Tax
- WishList

**and more...**
File System | Module Structure

** and more…
Modes
Modes

There are three primary modes in Magento 2:

Developer

Production

Default
Modes | Developer Mode

Development phase

- Static file materialization is not enabled
- Uncaught exceptions displayed in the browser
- Exceptions thrown in error handler, not logged
- System logging in `var/report`, highly detailed
Modes | Production Mode

Production

- Deployment phase on the production system; highest performance
- Exceptions are not displayed to the user – written to logs only
- This mode disables static file materialization
- The Magento docroot can have read-only permissions
Modes | Default Mode

Default

- Used when no other mode is specified
- Hides exceptions from the user and writes them to log files
- Static file materialization is enabled
- Not recommended / not optimized for production; caching impacts performance negatively
## Modes | Summary of Mode Features

<table>
<thead>
<tr>
<th>Modes</th>
<th>Static File Caching</th>
<th>Exceptions Displayed</th>
<th>Exceptions Logged</th>
<th>Performance (-) Impacted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developer</td>
<td>✅</td>
<td>✅</td>
<td></td>
<td>✅</td>
</tr>
<tr>
<td>Production</td>
<td></td>
<td></td>
<td></td>
<td>✅</td>
</tr>
<tr>
<td>Default</td>
<td>✅</td>
<td></td>
<td>✅</td>
<td>✅</td>
</tr>
</tbody>
</table>
Modes | Maintenance Mode

Maintenance

- Used to make site unavailable to public during updates or other changes
- `Bootstrap::assertMaintenance()` controls this mode; must create the flag `var/.maintenance.flag` to enable mode
- Specify group of people that can have access: `var/.maintenance.ip`
- Maintenance mode is an out-of-the-box feature
Specify mode in one of two ways...

- Use an environment variable
  - or -
- Use the web server or php-fpm environment
Use the MAGE_MODE system environment variable to specify a mode:

- MAGE_MODE=[developer|default|production]

After setting the mode, restart the web server:

- Ubuntu: service apache2 restart
- CentOS: service httpd restart
Modes | Specify Mode: Web Server Environment

• Apache web servers with mod_php support this method, using mod_env directives.

• Set in the .htaccess file:

  SetEnv MAGE_MODE=[developer|default|production]

• Apache directive is slightly different in versions 2.2 and 2.4; consult Apache documentation for more information/guidance.
**Modes | Specify Mode: php-fpm Environment**

- Specify mode in php-fpm config, or in system environment in which php-fpm is started.

- In php-fpm config, value set as follows:

  ```
  env[MAGE_MODE]=[developer|default|production]
  ```

- Location of php-fpm config file depends upon system.
The CLI performs both installation and configuration tasks:

• Install Magento

• Clear cache

• Manage indexes

• Generate non-existent classes (factories, interceptors for plugins, DI configuration for the object manager)

• Enable/disable available modules

• Deploy (or clear) static view files
Magento has a Magento\Framework\Cache library component for implementing Magento-specific caching.
Magento 2 sets a default caching configuration with installation.

It is unlikely that you will want to change these settings, but if you do want to create a custom mechanism, the basic default list of cache types are located in: app/etc/env.php
Cache types group the cached data based on functional role.

Operations like clearing, disabling, or enabling the cache can be limited to specific cache types.

You can manage the cache via the Cache Management page in the Admin panel, or via the command-line tool `bin/magento`.

Modes | Cache Type
There are three ways to clean the cache:

- From the Admin (backend)
- Using the command-line tool bin/magento:
  ```sh
  ./bin/magento cache:clean
  ```
- Manually removing cache files
  
  To remove cache files in Linux, run this command in the Magento root folder:
  ```sh
  rm -rf var/cache/*
  ```
Q & A
Technical Documentation & Resources
https://magento.com/resources/technical

Magento Forums
https://community.magento.com/?_ga=1.233502680.771106929.1468874877

Magento Blog
https://magento.com/blog

Magento U Authorized Trainer Program
https://u.magento.com/magento-u-authorized-trainer-program

Email us at training@magento.com
Thank you