**Magento 2 Certification Guide**

1. Fundamentals Introduction
2. Magento 2 Fundamentals: Unit One

This course contains three different units -- Preparation & Configuration, Request Routing, and Rendering. You access each of the units as a separate course.

Each unit contains a number of modules. Here, in Unit One, there are seven modules.

1. Requirements To Install Magento 2

* Magento 2 requires the familiar LAMP (Linux, Apache, MySQL, and PHP) structure along with PHP Composer,a tool for managing external dependencies. Install dependent module like symphony
* System requirement

PHP Extensions

* PDO / Mysql
* Mbstring
* Mhash
* Simplexml
* Curl
* GD2, IMagMagick 6.3.7+, Or both
* SOAP

Application Parameters

Website

Environmental Parameters

1. Modes In Magento 2

Magento 2 has three primary modes – (1) Developer, (2) Production, and (3) Default.

There is also a maintenance mode, but that operates in a different way, only to prevent access to the system.

* Developer Mode
* You should use the Developer mode while you are developing the code for the application.
* The main benefit to this mode is that error messages are visible to you.It should not be used once in production because of its impact on performance.
* Developer mode, static view files are not cached; instead, they are written to the Magento docroot every time they are called.
* System logging in var/report is highly detailed in this mode.
* Production Mode
* Production mode provides the highest performance in Magento 2.
* The most important aspect of this mode is that errors are logged to the file system and are never displayed to the user.
* View files are not materialized; instead, URLs for the files are composed on-the-fly, without going through the fallback mechanism.
* Magento docroot has read-only permissions.

Maintenance Mode

* Used when you want to make the site unavailable to the public during updates or other changes.
* The method Bootstrap::assertMaintenance() controls this mode, and you have to create a flag file (var/.maintenance.flag) to enable the mode.
* You can specify a group of people to have access to the site while this mode is employed by placing the associated IPs in the file (var/.maintenance.ip).
* Maintenance mode is an out-of-the-box feature in Magento 2.
* Default Mode
* Default mode is how the Magento software operates if no other mode is specified.
* Errors are logged to file reports at the server, and are never shown to a user. Static file materialization is enabled.
* Default mode is not optimized for a production environment, primarily because of the adverse performance impact of static files being cached rather than materialized.
* Creating static files and caching them has a greater performance impact than generating them using the static file creation tool.

1. Specifying a Mode

basically two ways in which you can set the mode for your system – using an (1) environment variable, or using the (2) web server environment.

* Environment Variable

environment variables allows you to set the mode for your Linux, Windows, Apple, or any other system.

All the environment variables differ depending upon how they are set.

* Web Server Environment

The other approach to setting mode is to set environment variables in the web server.

The most popular server type is probably Apache.

.htaccess file is a good example.

1. Overview and Architecture
2. Magento 2 Platform

Magento 2 continues to provide the same flexibility and extensibility of Magento 1, while introducing structural changes that address challenges around describing dependencies across the system (for easier upgrades), and around containing business logic within one system layer. All modules are developed in PHP so they are related in a consistent way.

Magento 2's instantiation mechanism and code generation provides numerous ways to customize, such as through the use of plugins.

1. Magento 2 Goals

Built on a new and modern technology stack, Magento 2 integrates better with third-party solutions, and is more accessible and open to frontend developers. With an improved implementation process, partners and merchants will realize faster deployments, simplified upgrades, and faster time to value.

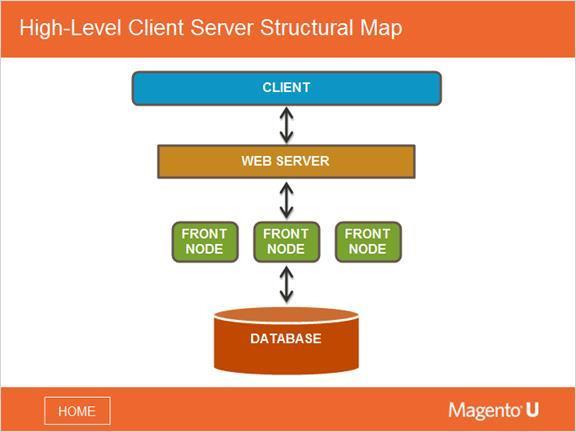
There were six key goals in designing and producing the Magento 2 platform:

* Streamline the customization process.
* Update the technology stack.
* Improve performance and scalability.
* Reduce upgrade efforts and costs.
* Simplify integrations.
* Provide high quality tested code, testing resources, and documentation (and increase engagement with the Magento community).

1. Magento 2 Architecture

Magento 2, like Magento 1, is a modular system but to a much higher degree. Magento 2 assigns greater independence to the modules – they function more as standalone units.

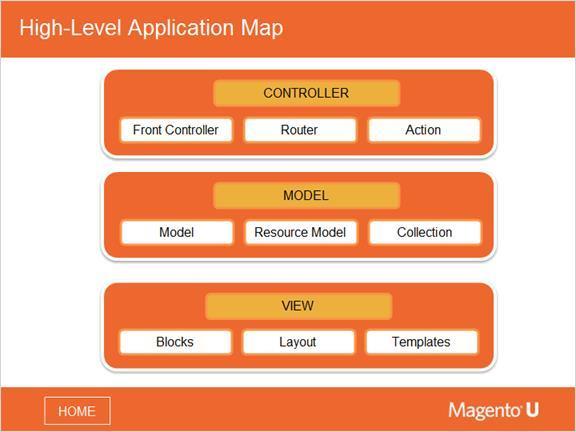
1. High-Level Client Server Structural Map



This structural map shows a standard client-server setup. A client sends a request to the web server or cloud, which then distributes the request among one or more front nodes. This depends on the website setup of the customer.

Note that often the database and frontend are located in separate servers.

1. High-Level Application Map



This diagram presents a very high level application map. You are probably already familiar with the Model-View-Controller (MVC) concept. The model (or data layer) manipulates the data, the view layer renders the data, and the controller layer makes it all work together.

CONTROLLER

Magento has its own version of the MVC structure with respect to controllers. Unlike Magento 1, where one controller could process many requests, Magento 2 assigns a single controller to a single request. It is not possible to process multiple URLs in one file. The controller layer is composed of 3 systems - the front controller, the routing system, and other controllers.

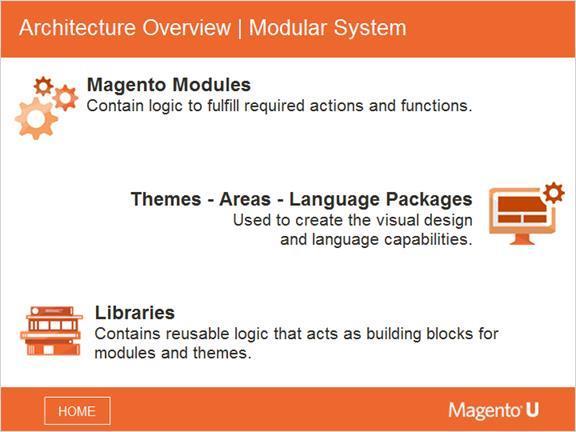
MODEL

The model is the part of the application that works with stored data. Magento 2 retains the resources model and collections that were in Magento 1, while adding an additional system of repositories and services.

VIEW

Within Magento 2, the view system and layout remains conceptually the same as with Magento 1, but the organization of the layout, templates, and blocks is different. Consequently, the approach for frontend development has changed dramatically.

1. Modular System

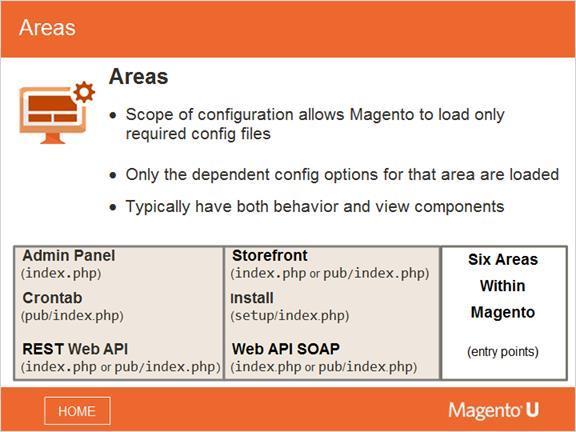


The Magento framework provides core business logic and functionality, base classes, resource models, and data access capabilities. The fundamental concepts and rules for how the components of the website behave are defined within the Magento framework. The framework provides core components with base functionality that can then be inherited by custom components for a specific website or application.

The final behavior, look-and-feel, and capabilities of the website are determined by how the components are extended and customized. Modules and themes are the units of customization in Magento -- modules for business features, and themes for the user experience and look-and-feel. Both have a lifecycle allowing them to be installed, deleted, disabled, and so on.

The libraries contain reusable logic that acts as building blocks for modules and themes.

1. Areas



Area is a scope in configuration that allows you to load configuration files and options. Within Magento 2, there are areas for frontend, admin and install, just as in Magento 1.

The purpose of areas within Magento is to increase efficiency by not requiring the loading of the entire configuration for every request.

For example, if you are invoking a REST web service, a corresponding area (such as /rest) loads code that answers only the REST call and not the code that generates HTML pages using layouts.

Each area can have completely different code on how to process URLs and requests. Magento 2 processes a URL request by first stripping off the base URL.

Typically, an area contains behavior and view components that operate separately. However, an area can have only one component --for instance, the cron area, which has no view component. If your extension works in several areas, you should make sure it has separate behavior and view components for each area.

1. Libraries

Libraries play a big role in configuration and file management. Magento 2 places a special focus on libraries - the library component is much larger and better than in Magento 1. You can have the native implementation or substitute with your own implementation, which is a different approach from Magento 1. Magento 2 also uses interfaces much more than Magento 1.

Magento is a unique application where owners can sell products, process orders, and receive payments. To provide this functionality, Magento combines framework features like those in Symphony and Zend Framework with application features. This mix can sometimes make the system difficult to work with.

Magento 2 separates the framework from the application, which helps to alleviate this issue. The framework code and remaining code are stored in the code folder.

The most important library is Magento/Framework, located in lib/internal/Magento/Framework.

We will provide more detailed information about libraries later in the course, in Unit Three: Rendering.

1. Language Packs

Magento 2 allows you to present a user interface in multiple languages without having to modify the application source code.

By default, all labels and system messages are expressed in English in the source code, but they can be replaced with another language when the code is by providing dictionary files that contain phrases from en\_US translated into a different language. The dictionary packages in other languages either ship with Magento out-of-the-box, or are provided by the community.

1. Themes

Magento 2 introduces significant changes to how Magento handles themes.

Theming now primarily focuses on creating CSS files and incorporating responsive design. Also, backend tasks and frontend tasks have been separated. In this course, we will discuss the backend part of theming, which involves blocks, templates, and more.