CMS Certified Developer 9 LTS

Building plugins/modules with Extbase

Controller

What is a controller in MVC context.

Customising Extbase Repositories

How to extend standard functionality for persistence.

Error Actions

How to inform about validation errors.

Extbase Persistence

How Extbase interacts with the permanent storage.

Extbase PersistenceManager

The built-in PersistenceManager.

Extbase Request/Response

How request/response handling is done in Extbase.

Extbase View

Explaining the view part of MVC.

Extending Base Models

Modelling in Extbase.

Implementing Custom Validators

How to provide custom validation.

Implementing MVC in TYPO3

Introducing the concept of MVC in Extbase: base classes for models, controllers, repositories; the concept of services and utilities and when to use them.

Implementing ViewHelpers

Provides the foundations for extending a view with custom PHP code.

Standard Extbase Repository

Knowing the default repository class.

Switchable Controller Actions

Plugin-related action configuration.

Validation in Extbase

What concepts exist to validate controller action input parameters.

Building plugins/modules without Extbase

AbstractPlugin

I can decide when to use AbstractPlugin for new extensions, and I can find my way around existing extensions that use AbstractPlugin.

Extbase Annotations

PHPDoc annotations particular to Extbase.

Core Architecture & APIs

Accessing Flexform Data

Using data provided by Flexforms, in plugin CEs and custom records.

Contexts & Aspects

I know the difference between the ApplicationContext and the Context API. I can use the Context API, know which Aspects exists, and can access informations from Aspects.

DataHandler

In my extensions, I can read and modifiy the data entered by a user forms in the BE. I also can create and use new commands.

Debugging in PHP

Basic knowledge about error tracing possibilities present in PHP.

Environment

I can use the Environment class and know which informations it provides.

Error handling/Exception handling

I can configure the error and exception handling for development and production. I also know when errors are handled by the error handler, and where to find any logged errors.

Extending site configurations

I can place custom fields in the site configuration module offered by the core, and can read values stored in these fields.

Extension scanner

I can use the extension scanner from within the TYPO3 BE and from the command line, and I know what kind of problems the scanner finds, and the limitations of the scanner.

FAL

Know about transparent inclusion of external file sources into TYPO3.

Flash Messages

I know where the Core flash messages and notifications can be used (and where not), know which type of flash message to use for what purpose, and can create and output flash messages using PHP and JavaScript in my code. Also, I can output flash messages created in existing code.

Flexform Concepts

Basics about the concept of Flexforms and how it relates to the TCA.

Flexform Config Data Structure

How to create and configure a data structure for my plugin’s configuration.

Knowledge about Environment Error Logs

Ability to locate logs provided by system components.

Localising Extensions

How to implement translations for localisation.

Logging API

Knowledge about the logging API provided by TYPO3.

Mail API/SwiftMailer

I can configure the mail API for different contexts. I can also create new emails in my extensions and send them with SwiftMailer. I also know and take into account the limits of the different transports.

MetaTag API

I can use the MetaTag API to add, modify or remove meta tags. I can implement and register own MetaTagManagers.

Monolog Logging

Knowledge about the monolog logging de-facto standard in PHP.

PHP Debugging Utilities

Knowledge about debugging utilities provided by TYPO3.

PSR-15: Middlewares

I know what middlewares and request processors are in PSR-15 and how they relate to each other

PSR-7: Request/Response

Knowledge about the PSR-7 standard.

PageRenderer

I can use the PageRenderer to modify or extend CSS, JavaScript and other header information in frontend and backend.

PageTitle API

I can implement and register own PageTitleProviders using the PageTitleAPI to modify the document's title.

Request processing with middlewares

I understand how the core leverages middlewares to modularize request processing (middleware dispatcher, chained middlewares, …). I can write my own middlewares and hook them into frontend and backend request processing. I can overwrite core-provided middlewares.

Route Enhancers

I know what route enhancers are and about the different types that exist. I also can use route enhancers in my own extensions.

Symfony CLI (CommandController deprecated)

I know how to execute existing Symfony CLI commands. I can create and register new Symfony CLI commands and find my way around existing commands and can modify them.

Symfony Expression Language (should be renamed to reflect extending the functionality)

I know the Symfony component "Expression Language" and its syntax. I can write and register own

ExpressionFunctionProviders and can provide custom TypoScript conditions.

The Caching Framework

How to use existings caches, and register own caches. The types of cache frontends and backends. Expiration, tagging and flushing.

The USER TypoScript object

I know what the USER TypoScript object is, the differences to USER_INT, and how it relates to plugins.

The registry API

I can use the registry API to read and write data. I can decide when the registry makes sense to use and when not.

Tracing Errors in Logs

Knowledge about log sources and how to utilise them in order to resolve errors.

Using Restrictions

How to restrict queries.

Using the QueryBuilder

How to write queries in TYPO3.

What is Doctrine DBAL?

Basic knowledge about the Doctrine component for transparently accessing several DBMS in PHP.

What is a DBMS?

Knowledge about DBMS.

XLIFF

Understanding XLIFF.

General extension architecture

AJAX handling in backend

I can register routes for backend AJAX calls and can access registered Ajax URLs in JavaScript.

Composer for TYPO3 Extensions

I know which fields to use in the composer.json of my extension, what they mean and how they work, so that I can create working, readable and helpful composer.json files.

I can use 3rd-party extensions and my own extensions in my site project via Composer.

I also know the benefits and drawbacks of using Composer and can advise my coworkers and clients when to use which approach.

In addition, I can build composer.json and ext_emconf.php files in such a way that they their data is synchronized correctly.

Also, I know to to update the autoloader data for non-Composer installations.

Creating Scheduler Tasks

How to build Scheduler tasks.

Database Schema Definition

I can create an ext_tables.sql for new and existing tables and know which fields are generated automatically from the TCA.

Extension File and Folder Structure

Knowing the file structure of an extension.

JavaScript handling in backend

I can write and use a RequireJS module and can change RequireJS configuration to provide new shims for libraries that do not support RequireJS out of the box.

PSR autoloading standards

The different PSR autoloading standards (PSR-0 and PSR-4), how they map PHP classes to files and folders, and how the PSR autoloading is configured.

Registering plugins and modules

I can correctly register new cached/uncached plugins. I know the Extbase and the non-Extbase way of registering.

Table Configuration Array (TCA)

Configuring database tables for usage in the back end and with Extbase.

Update wizards

I can implement an update wizard and know when to use it. I know about the "class.ext_update.php" file and how to use it, and can create this file for my extensions.

What is SQL?

Knowledge about how to access and manipulate data in DBMS.

Security

Handling User Input

Knowledge about possible ways to retrieve and process user input.

User Input in DB Queries

Knowledge about securely processing user input to DB.

User Input in PHP Code

Knowledge about securely processing user input in PHP code.

User Input in Views

Knowledge about securely outputting values gained through user input.

Web Vulnerability Types

Knowledge about possible vulnerabilities in web development.

Software Quality

Coding Standards in TYPO3

Knowing how to use the coding standard recommendations in the context of TYPO3.

Docblocks in PHP

The syntax and general purpose of documentation blocks, and the most used annotations.

Software architecture for extensions

Domain-Driven Design (DDD)

Understanding the concept of DDD.

Model

What is a model in MVC context.

PHP

Ability to develop and execute PHP scripts.

PSR Standards

Knowing the PSR programming standards.

View

What is a view in MVC context.

What is MVC?

This is a placeholder skill.

We will inform you as soon as this Skill becomes available at a more detailed level or is replaced by a whole package.

Your progress and SkillUps will be saved!

What is a Templating Engine?

Introduction to the basic concept of a templating engine/processor.

Templating & other outputs

Extbase Fluid View

The Fluid-based HTML view of Extbase.

Extbase JSON View

The built-in JSON View in Extbase.

Importing ViewHelpers

Using third party ViewHelpers in your templates.

JSON

The JSON format and its usages in web development.

Outputting Variables

Basic usage of variables inside Fluid templates.

Templates, Layouts, Partials

Fluid Templates, Layouts, Partials - their interaction and properties.

Using the Fluid View

Knowing and understanding the different parts of a Fluid template view.

ViewHelper: Comment

Including comments in Fluid templates.

ViewHelper: Debug

Utilising output to access Fluid variables.

ViewHelper: For-Loop

Iterating through a list with the for ViewHelper.

ViewHelper: Form

The form ViewHelper of TYPO3 Fluid.

ViewHelper: Format

Using the Fluid ViewHelper f:format().

ViewHelper: If/Else

Creating if/else statements within Fluid templates.

ViewHelper: Image

Using the Fluid ViewHelper f:image().

ViewHelper: Link, Uri

Using the Fluid ViewHelpers f:link.*() and f:uri.*().

ViewHelper: Render

Using f:render to render sections and partials as well as passing arguments.

ViewHelper: Widgets

Learn the most important f:widget.*() ViewHelpers.

ViewHelpers

Getting to know what a ViewHelper is.

What is Fluid?

Definition of Fluid and its possible use with regards to the Model View Controller pattern.