LiquiDoc Documentarian Manual

Table of Contents

Introducing LiquiDoc and LiquiDoc CMF

Welcome to the LiquiDoc Documentarian Manual!

Here you will learn to write, develop, and manage great docs with LiquiDoc as a documentarian.

This Manual is intended to orient you to the role of **documentarian**, a LiquiDoc user mainly concerned with *creating and managing content*. This Manual expects that you have a pre-configured LiquiDoc CMF environment established by a qualified engineer, and you just need to get up and running so you can write/edit content. Hopefully, whoever did that configuring has edited this documentation in key areas to ensure it applies to your instance.

> Check with your LiquiDoc CMF administrator to ensure this guide is oriented toward your LDCMF instance; then maybe remind them to remove this note.

Here are the user operations covered in this guide:

- Get started with a liquidoc-ready repository
- Grasp the significance of liquidoc cmf
- Get familiar with this ldcmf environment
- Establish a great local toolset
- Write and edit content in asciidoc
- Locate and manage source files
- Commit changes to source control
- Write and edit small-data in yaml
- Submit and iterate data/content commits for publication
- Eview and approve data/content commits for publication
- Build a preconfigured docset
- Build a preconfigured docset with additional options
- Understand my rights as a contributor
- Understand how merge requests are evaluated

> This document contains lots of jargon. See the Jargon Guide if you get lost.

Setup

Getting Started with a LiquiDoc-ready Repository

User story	As a **documentarian**, I can **get started with a LiquiDoc-ready repository**.

Establishing a Local Toolset

User story	As a **documentarian**, I can **establish a great local toolset**.

Build

Building a Preconfigured Docset

User story	As a **documentarian**, I can **build a preconfigured docset**.

Build a Preconfigured Docset with Additional Options

User story	As a **documentarian**, I can **build a preconfigured docset with additional options**.

Contribute

Writing and Editing in AsciiDoc

User story	As a **documentarian**, I can **write and edit content in AsciiDoc**.

Locating & managing Source Files

User story	As a **documentarian**, I can **locate and manage source files**.

Understanding the architecture of any file- or content-management scheme takes time. Clarity and accessibility are aided by the strict use of conventions and standards shared *teamwide*.

So that you can **find content and data** when you need it, and so you can **determine where to save new content** as you create it, we offer LiquiDoc CMF's content and file management conventions. These are the rules governing how and where to store content and data of various types.

Discerning Content vs Data

Managing Content

Managing Data

Committing changes to Source Control

User story	As a **documentarian**, I can **commit changes to source control**.

Writing & Editing Small-Data in YAML

User story	As a **documentarian**, I can **write and edit small-data in YAML**.

Submitting Data & Content for Review

User story	As a **documentarian**, I can **submit and iterate data/content commits for publication**.

Reviewing Merge Requests

User story	As a **documentarian**, I can **eview and approve data/content commits for publication**.

Understand

Grasping the Significance of LiquiDoc CMF

User story	As a **documentarian**, I can **grasp the significance of LiquiDoc CMF**.

LiquiDoc CMF (LDCMF) is a “content management *framework*” — a set of tools, conventions, and standards for sensibly organizing and maintaining source content and data as well as producing deliverable artifacts. LDCMF is neither a content management *system* (CMS) nor a *component* content management system (CCMS), mainly because it does not revolve around a database or a user interface designed for a specific kind of document management. There are key differences you should be aware of:

Pros/Cons of LiquiDoc CMF vs Proprietary CMS/CCMS Solutions

There are several major differences between an open-source docs-as-code approach to creating, managing, and publishing technical documentation. Whether they are *pros* or *cons* in your book may depend very much on whether your background.

Some assembly required

Users are expected to heavily customize and extend their LDCMF environment rather than fall back on “turnkey” features and elements. While you can technically write and build a pretty straightforward docset based on existing, freely available LDCMF examples, your needs will almost certainly vary. The free-and-open model adhered to by the *framework* means you will never encounter a dead end imposed by the LDCMF platform. Hopefully you won’t need to actually *modify or extend* the base tooling to solve your needs, but you will need to *configure* a complex docs build if you have complex problems. Presumably, that’s how you ended up here anyway.

Small data is simple

LDCMF’s sources of data and content are far simpler than conventional CMS applications. Stored in flat files and directories rather than relational databases (RDBs), LDCMF’s relatively flexible and casual datasource options have their limitations. Most conventional CMS platforms take advantage of RDBs’ powerful indexing and querying capabilities, not only handling *content and data* but managing large amounts of site and content *metadata*. On the other hand, RDBs cannot be used with distributed source-control systems like Git.

GUI? What GUI?

LiquiDoc CMF’s user interface is command-line tools (CLIs) and free, open-source text/code editors, rather than proprietary desktop programs or web apps. For some, this is the epitome of power and freedom. For others, these blinking-cursor options are intimidating to the point of paralysis. While LDCMF can be deftly operated by *beginners* with both kinds of tools, there may be some initial discomfort. But then: *total freedom and power!*

In your role as a documentation contributor, you will not need to be able to explain the differences between LiquiDoc, LDCMF, and your team’s implementation, so this overview should suffice.

Getting to Know This LDCMF Environment

User story	As a **documentarian**, I can **get familiar with this LDCMF environment**.

Understanding My Rights as a Contributor

User story	As a **documentarian**, I can **understand my rights as a contributor**.

Understanding How Merge Requests are Evaluated

User story	As a **documentarian**, I can **understand how merge requests are evaluated**.

Appendix A: Jargon Guide

This is the full list of specialized terms used in this product documentation. They are also generated as JSON at `/data/terms.json` so we can highlight them in the text when we get to it. This is just to show the power of storing data in flat files reusable throughout product docs.

artifact

A digital package (file or archive) representing a discrete component of a product. Here we use *artifact* to describe a descrete instance of output, such as a single HTML or PDF file, or a Jekyll website or Deck.js slide presentation.

AsciiDoc

Dynamic, lightweight markup language for developing rich, complex documentation projects in flat files. (Resource)

Asciidoctor

Suite of open source tools used to process AsciiDoc markup into various rendered output formats. (Resource)

build

The (usually automated) series of actions necessary to compile and package software or documentation.

Liquid

Open source templating markup language maintained by Shopify (Resource)

YAML

a slightly dynamic, semi-structured data format for nested data (Resource)

Appendix B: NOTICE of Packaged Dependencies

The following open source packages are fully or partially included with LiquiDoc.

Package	**Jekyll Documentation Theme**
License	MIT
Author	Tom Johnson
Website	https://github.com/tomjoht/documentation-theme-jekyll

Package	**M+ OUTLINE FONTS (M+ TESTFLIGHT 058)**
License	unlimited
Author	M+ Fonts Project
Website	http://mplus-fonts.osdn.jp/about-en.html

Package	**Noto Fonts**
License	SIL OFL
Author	Google i18n
Website	https://www.google.com/get/noto/

Package	**Font Awesome**
License	SIL OFL 1.1
Author	Fonticons, Inc
Website	https://fontawesome.com/

Package	**"Coding Style Guide"**
License	MIT
Author	Dan Allen, Paul Rayner, and the Asciidoctor Project
Website	https://github.com/asciidoctor/jekyll-asciidoc/blob/master/coding-style-guide.adoc