Composer CMS: Content Make System

Gary B. Genett

v3.0 (2022-05-11)

Contents

Chapter 1

Composer CMS

	"Creating Made Simple."
Composer CMS v3.0 Gary B. Genett	License: GPL composer@garybgenett.net

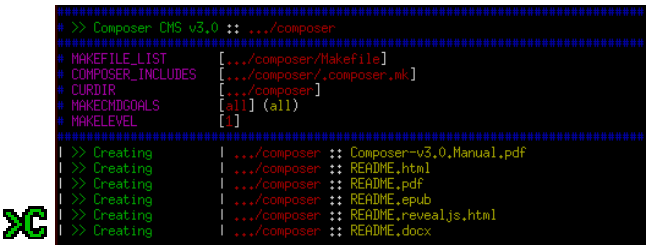
1.1 Overview

Composer is a simple but powerful CMS based on Pandoc, Bootstrap and GNU Make. It is a document and website build system that processes directories or individual files in Markdown format.

Traditionally, CMS stands for Content Management System. Composer is designed to be a Content **Make** System. Written content is vastly easier to manage as plain text, which can be crafted with simple editors and tracked with revision control. However, professional documentation, publications, and websites require formatting that is dynamic and feature-rich.

Pandoc is an extremely powerful document conversion tool, and is a widely used standard for processing Markdown into other formats. While it has reasonable defaults, there are a large number of options, and additional tools are required for some formats and features.

Composer consolidates all the necessary components, simplifies the options, and prettifies the output formats, all in one place. It also serves as a build system, so that large repositories can be managed as documentation archives or published as Bootstrap Websites.



1.2 Quick Start

Use make help to get started:

```
make [-f .../Makefile] [variables] <filename>.<extension>
make [-f .../Makefile] [variables] <target>
```

Fetch the necessary binary components (see Repository Versions):

```
make _update-all
```

Create documents from source Markdown files (see Formatting Variables):

```
make README.html
make Composer-v3.0.Manual.html c_list="README.md LICENSE.md"
```

Save a persistent configuration (see Recommended Workflow, Configuration Settings and Special Targets):

```
make template >.composer.mk
$EDITOR .composer.mk
    book-Composer-v3.0.Manual.html: README.md LICENSE.md
make clean
make all
```

Recursively install and build an entire directory tree (see Recommended Workflow):

```
cd .../documents
mv .../composer .Composer
make -f .Composer/Makefile install-all
make all-all
```

See help-all for full details and additional targets.

1.3 Principles

The guiding principles of Composer:

- All source files in readable plain text
- Professional output, suitable for publication
- Minimal dependencies, and entirely command-line driven
- Separate content and formatting; writing and publishing are independent
- Inheritance and dependencies; global, tree, directory and file overrides
- Fast; both to initiate commands and for processing to complete

Direct support for key document types (see Document Formatting):

- HTML & Bootstrap Websites
- PDF
- EPUB
- Reveal.js Presentations
- Microsoft Word & PowerPoint

1.4 Requirements

Composer has almost no external dependencies. All needed components are integrated directly into the repository, including Pandoc. It does require a minimal command-line environment based on GNU tools, which is standard for all GNU/Linux systems. The Windows Subsystem for Linux for Windows and MacPorts for macOS both provide suitable environments.

The one large external requirement is TeX Live, and it can be installed using the package managers of each of the above systems. It is only necessary for creating PDF files.

Below are the versions of the components in the repository, and the tested versions of external tools for this iteration of Composer. Use check to validate your system.

Repository	Commit	License
Pandoc	2.18	GPL
YQ	v4.24.2	MIT

Repository	Commit	License
Bootstrap	v5.1.3	MIT
Markdown Viewer	059f3192d4ebf5fa9776	MIT
Reveal.js	4.3.1	MIT

Project	Composer Version
GNU Bash	5.0.18
- GNU Coreutils	8.31
- GNU Findutils	4.8.0
- GNU Sed	4.8
GNU Make	4.2.1
- Pandoc	2.18
- YQ	4.24.2
- TeX Live (pdf)	2021 3.14159 2.6-1.40.22

Markdown Viewer is included both for its CSS stylesheets, and for real-time rendering of Markdown files as they are being written. To install, follow the instructions in the README.md, and select the appropriate manifest.*.json file for your browser.

The versions of the integrated repositories can be changed, if desired (see Repository Versions).

Chapter 2

Composer Operation

2.1 Recommended Workflow

The ideal workflow is to put Composer in a top-level .Composer for each directory tree you want to manage, creating a structure similar to this:

```
.../.Composer
.../
.../tld/
.../tld/sub/
```

Then, it can be converted to a Composer documentation archive (Quick Start example):

```
make -f .Composer/Makefile install-all
make all-all
```

If specific settings need to be used, either globally or per-directory, .composer.mk files can be created (see Configuration Settings, Quick Start example):

```
make template >.composer.mk
$EDITOR .composer.mk
```

Custom targets can also be defined, using standard GNU Make syntax (see Custom Targets).

GNU Make does not support file and directory names with spaces in them, and neither does Composer. Documentation archives which have such files or directories will produce unexpected results.

It is fully supported for input files to be symbolic links to files that reside outside the documentation archive:

```
cd .../tld
ln -rs .../README.md ./
make README.html
```

Finally, it is best practice to install-force after every Composer upgrade, in case there are any changes to the Makefile template (see Primary Targets).

The archive is ready, and each directory is both a part of the collective and its own individual instance. Targets can be run per-file, per-directory, or recursively through an entire directory tree. The most commonly used targets are in Primary Targets.

Welcome to Composer. Happy Making!

2.2 Document Formatting

As outlined in Overview and Principles, a primary goal of Composer is to produce beautiful and professional output. Pandoc does reasonably well at this, and yet its primary focus is document conversion, not document formatting. Composer fills this gap by specifically tuning a select list of the most commonly used document formats.

Further options for each document type are in Formatting Variables. All improvements not exposed as variables will apply to all documents created with a given instance of Composer.

Note that all the files referenced below are embedded in the 'Embedded Files' and 'Heredoc' sections of the Makefile. They are exported by the _release target, and will be overwritten whenever it is run.

2.2.1 HTML

In addition to being a helpful real-time rendering tool, Markdown Viewer includes several CSS stylesheets that are much more visually appealing than the Pandoc default, and which behave like normal webpages, so Composer uses them for all HTML-based document types, including EPUB.

Information on installing Markdown Viewer for use as a Markdown rendering tool is in Requirements.

2.2.2 Bootstrap Websites

Bootstrap is a leading web development framework, capable of building static webpages that behave dynamically. Static sites are very easy and inexpensive to host, and are extremely responsive compared to truly dynamic webpages.

Composer uses this framework to transform an archive of simple text files into a modern website, with the appearance and behavior of dynamically indexed pages.

(This feature is reserved for a future release as the site target, along with page and post in Special Targets.)

2.2.3 PDF

The default formatting for PDF is geared towards academic papers and the typesetting of printed books, instead of documents that are intended to be purely digital.

Internally, Pandoc first converts to LaTeX, and then uses TeX Live to convert into the final PDF. Composer inserts customized LaTeX to modify the final output:

```
.../artifacts/pdf.latex
```

2.2.4 EPUB

The EPUB format is essentially packaged HTML, so Composer uses the same Markdown Viewer CSS stylesheets for it.

2.2.5 Reveal.js Presentations

The CSS for Reveal.js presentations has been modified to create a more traditional and readable end result. The customized version is at:

```
.../artifacts/revealjs.css
```

It links in a default theme from the .../revealjs/dist/theme directory. Edit the location in the file, or use c_css to select a different theme.

It is set up so that a logo can be placed in the upper right hand corner on each slide, for presentations that need to be branded. Simply copy an image file to the logo location:

```
.../artifacts/logo.img
```

To have different logos for different directories (using Recommended Workflow, Configuration Settings and Precedence Rules):

```
cd .../presentations
cp .../logo.img ./
ln -rs .../.Composer/artifacts/revealjs.css ./.composer.css
echo 'override c_type := revealjs' >>./.composer.mk
make all
```

2.2.6 Microsoft Word & PowerPoint

The internal Pandoc templates for these are exported by Composer, so they are available for customization:

```
.../artifacts/reference.docx
.../artifacts/reference.pptx
```

They are not currently modified by Composer.

2.3 Configuration Settings

Composer uses .composer.mk files for persistent settings and definition of Custom Targets. By default, they only apply to the directory they are in (see COMPOSER_INCLUDE in Control Variables). The values in the most local file override all others (see Precedence Rules).

The easiest way to create a new .composer.mk is with the template target (Quick Start example):

```
make template >.composer.mk
$EDITOR .composer.mk
```

All variable definitions must be in the override [variable] := [value] format from the template target. Doing otherwise will result in unexpected behavior, and is not supported. The regular expression that is used to detect them:

```
override[[:space:]]+([^[:space:]]+)[[:space:]]+[:][=]
```

Variables can also be specified per-target, using GNU Make syntax (these are the settings used to process the Composer README.* files):

```
README.%: override c_css := css_alt
README.%: override c_toc := 0
README.epub: override c_css :=
README.revealjs.html: override c_css :=
README.revealjs.html: override c_toc :=
```

In this case, there are multiple definitions that could apply to README.revealjs.html, due to the % wildcard. Since the most specific target match is used, the final values for both c_css and c_toc would be empty.

2.4 Precedence Rules

The order of precedence for .composer.mk files is global-to-local (see COMPOSER_INCLUDE in Control Variables). This means that the values in the most local file override all others.

Variable aliases, such as COMPOSER_DEBUGIT/c_debug/V are prioritized in the order shown, with COMPOSER_* taking precedence over c_*, over the short alias.

Selection of the CSS file can be done using .composer.css or the c_css variable, with .composer.css taking precedence (unless c_css comes from .composer.mk). The process for .composer.css files is identical to .composer.mk (see COMPOSER_INCLUDE in Control Variables).

All values in .composer.mk take precedence over everything else, including .composer.css and environment variables.

2.5 Specifying Dependencies

If there are files or directories that have dependencies on other files or directories being processed first, this can be done simply using GNU Make syntax in .composer.mk:

```
LICENSE.html: README.html
all-subdirs-artifacts: all-subdirs-bootstrap
```

This would require README.html to be completed before LICENSE.html, and for bootstrap to be processed before artifacts. Directories need to be specified with the all-subdirs-* syntax in order to avoid conflicts with target names (see Custom Targets). Good examples of this are the internal docs and test targets, which are common directory names.

Chaining of dependencies can be as complex and layered as GNU Make will support. Note that if a file or directory is set to depend on a target, that target will be run whenever the file or directory is called.

2.6 Custom Targets

If needed, custom targets can be defined inside a .composer.mk file (see Configuration Settings), using standard GNU Make syntax. Naming them as *-clean or *-all will include them in runs of the respective targets. Targets with any other names will need to be run manually, or included in COMPOSER_TARGETS (see Control Variables).

There are a few limitations when naming custom targets. Targets starting with the regular expression [_.] are hidden, and are skipped by auto-detection. Additionally, there is a list of reserved targets in Reserved, along with a list of reserved variables.

Any included .composer.mk files are sourced early in the main Composer Makefile, so matching targets and most variables will be overridden. In the case of conflicting targets, GNU Make will produce warning messages. Variables will have their values changed silently. Changing the values of internal Composer variables is not recommended or supported.

A final note is that *-clean and *-all targets are stripped from COMPOSER_TARGETS. In cases where this results in an empty COMPOSER_TARGETS, there will be a message and no actions will be taken.

2.7 Repository Versions

There are a few internal variables used by _update to select the repository and binary versions of integrated components (see Requirements). These are exposed for configuration, but only within .composer.mk:

- PANDOC_VER (must be a binary version number)
- PANDOC_CMT (defaults to PANDOC_VER)
- YQ_VER (must be a binary version number)
- YQ_CMT (defaults to YQ_VER)
- BOOTSTRAP_CMT
- MDVIEWER_CMT
- REVEALJS_CMT

Binaries for Pandoc and YQ are installed in their respective directories. By moving or removing them, or changing the version number and foregoing _update-all (see Additional Targets), the system versions will be used instead. This will work as long as the commit versions match, so that supporting files are in alignment.

It is possible that changing the versions will introduce incompatibilities with Composer, which are usually impacts to the prettification of output files (see Document Formatting).

Chapter 3

Composer Variables

3.1 Formatting Variables

Variable	Purpose	Value
c_type ~ T	Desired output format	html
c_base ~ B	Base of output file	README
c_list ~ L	List of input files(s)	README.md
c_lang ~ g	Language for document headers	en-US
c_css ~ s	Location of CSS file	(.composer.css)
c_toc ~ c	Table of contents depth	
c_level ~ l	Chapter/slide header level	2
c_margin ~ m	Size of margins (PDF)	0.8in
c_options ~ o	Custom Pandoc options	

Values: c_type	Format	Extension
html	HyperText Markup Language	*.html
pdf	Portable Document Format	*.pdf
epub	Electronic Publication	*.epub
revealjs	Reveal.js Presentation	*.revealjs.html
docx	Microsoft Word	*.docx
pptx	Microsoft PowerPoint	*.pptx
text	Plain Text (well-formatted)	*.txt
markdown	Pandoc Markdown (for testing)	*.md.txt

- *Other c_type values will be passed directly to Pandoc*
- *Special values for c_css:*
  - *css_alt ~ Use the alternate default stylesheet*
  - *0 ~ Revert to the Pandoc default*
- *Special value 0 for c_toc ~ List all headers, and number sections*
- *Special value 0 for c_level ~ Varies by c_type (see help-all)*
- *An empty c_margin value enables individual margins:*
  - *c_margin_top ~ mt*
  - *c_margin_bottom ~ mb*
  - *c_margin_left ~ ml*
  - *c_margin_right ~ mr*

3.1.1 c_type / c_base / c_list

The compose target uses these variables to decide what to build and how. The output file is [c_base].<extension>, and is constructed from the c_list input files, in order. The <extension> is selected based on the c_type table above. Generally, it is not required to use the compose target directly for supported c_type files, since it is run automatically based on what output file <extension> is specified.

The automatic input file detection works by matching one of the following (Quick Start example):

```
make README.html                    ~ README (empty [COMPOSER_EXT])
make README.html                    ~ README.md
make README.md.html                 ~ README.md
make Composer-v3.0.Manual.html      c_list="README.md LICENSE.md"
```

Other values for c_type, such as json or man, for example, can be passed through to Pandoc manually:

```
make compose c_type="json" c_base="README" c_list="README.md"
make compose c_type="man" c_base="Composer-v3.0.Manual" c_list="README.md"
```

Any of the file types supported by Pandoc can be created this way. The only limitation is that the input files must be in Markdown format.

3.1.2 c_lang

- Primarily for PDF, this specifies the language that the table of contents (c_toc) and chapter headings (c_level) will use.

3.1.3 c_css

- By default, a CSS stylesheet from Markdown Viewer is used for HTML and EPUB, and one of the Reveal.js themes is used for Reveal.js Presentations. This variable allows for selection of a different file in all cases.
- The special value css_alt selects the alternate default stylesheet. Using 0 reverts to the Pandoc default.
- This value can be overridden by the presence of .composer.css files. See Precedence Rules for details.

3.1.4 c_toc

- Setting this to a value of [1-6] creates a table of contents at the beginning of the document. The numerical value is how many header levels deep the table should go. A value of 6 lists all header levels.
- Using a value of 0 lists all header levels, and additionally numbers all the sections, for reference.

3.1.5 c_level

- This value has different effects, depending on the c_type of the output document.
- For HTML, any value enables section-divs, which wraps headings and their section content in <section> tags and attaches identifiers to them instead of the headings themselves. This is for CSS styling, and is generally desired.
- For PDF, there are 3 top-level division types: part, chapter, and section. This sets the top-level header to the specified type, which changes the way the document is presented. Using part divides the document into "Parts", each starting with a stand-alone title page. With this division type, each second-level heading starts a new "Chapter". A chapter simply starts a new section on a new page, and lower-level headings continue as running portions within it. Finally, section creates one long running document with no blank pages or section breaks (like a HTML page). To set the desired value:
  - part ~ 0
  - chapter ~ 2
  - section ~ Any other value
- For EPUB, this creates chapter breaks at the specified level, starting the section on a new page. The special 0 simply sets it to the default value of 2.

- For Reveal.js Presentations, the top-level headings can persist on the screen when moving through slides in their sections, or they can rotate out as their own individual slides. Setting to 0 enables persistent headings, and all other values use the default.
- An empty value defers to the Pandoc defaults in all cases.

3.1.6 c_margin

- The default margins for PDF are formatted for typesetting of printed books, where there is a large amount of open space around the edges and the text on each page is shifted away from where the binding would be. This is generally not what is desired in a purely digital PDF document.
- This is one value for all the margins. Setting it to an empty value exposes variables for each of the individual margins: c_margin_top, c_margin_bottom, c_margin_left and c_margin_right.

3.1.7 c_options

- In some cases, it may be desirable to add additional Pandoc options. Anything put in this variable will be passed directly to Pandoc as additional command-line arguments.

3.2 Control Variables

Variable	Purpose	Value
MAKEJOBS	Parallel processing threads	1 (makejobs)
COMPOSER_DOCOLOR	Enable title/color sequences	(boolean)
COMPOSER_DEBUGIT	Use verbose output	(debugit)
COMPOSER_INCLUDE	Include all: .composer.mk	(boolean)
COMPOSER_DEPENDS	Sub-directories first: all	(boolean)
COMPOSER_LOG	Timestamped command log	.composed
COMPOSER_EXT	Markdown file extension	.md
COMPOSER_TARGETS	See: all/clean	config/targets
COMPOSER_SUBDIRS	See: all/clean/install	config/targets
COMPOSER_IGNORES	See: all/clean/install	config

- *MAKEJOBS ~ c_jobs ~ J*
- *COMPOSER_DOCOLOR ~ c_color ~ C*
- *COMPOSER_DEBUGIT ~ c_debug ~ V*
- *(makejobs) = empty is disabled / number of threads / 0 is no limit*
- *(debugit) = empty is disabled / any value enables / 0 is full tracing*
- *(boolean) = empty is disabled / any value enables*

3.2.1 MAKEJOBS

- By default, Composer progresses linearly, doing one task at a time. If there are dependencies between items, this can be beneficial, since it ensures things will happen in a particular order. The downside, however, is that it is very slow.
- Composer supports GNU Make parallel execution, where multiple threads can be working through tasks independently. Experiment with lower values first. When recursing through large directories, each make that instantiates into a sub-directory has it's own jobs server, so the total number of threads running can proliferate rapidly.
- This can drastically speed up execution, processing thousands of files and directories in minutes. However, values that are too high can exhaust system resources. With great power comes great responsibility.
- A value of 0 does parallel execution with no thread limit.

3.2.2 COMPOSER_DOCOLOR

- Composer uses colors to make all output and help text easier to read. The escape sequences used to accomplish

this can create mixed results when reading in an output file or a $PAGER, or just make it harder to read for some.
- This is also used internally for targets like debug-file and template, where plain text is required.

3.2.3 COMPOSER_DEBUGIT

- Provides more explicit details about what is happening at each step. Produces a lot more output, and can be slower. It will also be hard to read unless MAKEJOBS is set to 1.
- Full tracing using 0 also displays GNU Make debugging output.
- *When doing debug, this is used to pass a list of targets to test (see Additional Targets).*

3.2.4 COMPOSER_INCLUDE

- On every run, Composer walks through the MAKEFILE_LIST, all the way back to the main Makefile, looking for .composer.mk files in each directory. By default, it only reads the one in its main directory and the current directory, in that order. Enabling this causes all of them to be read.
- In the example directory tree below, normally the .composer.mk in .Composer is read first, and then tld/sub/.composer.mk. With this enabled, it will read all of them in order from top to bottom: .Composer/.composer.mk, .composer.mk, tld/.composer.mk, and finally tld/sub/.composer.mk.
- This is why it is best practice to have a .Composer directory at the top level for each documentation archive (see Recommended Workflow). Not only does it allow for strict version control of Composer per-archive, it also provides a mechanism for setting Composer Variables globally.
- Care should be taken setting "Local" variables from template (see Templates) when using this option. In that case, they will be propagated down the tree. This may be desired in some cases, but it will require that each directory set these manually, which could require a lot of maintenance.
- This setting also causes .composer.css files to be processed in an identical manner (see Precedence Rules).

Example directory tree (see Recommended Workflow):

```
.../.Composer/Makefile
.../.Composer/.composer.mk
.../Makefile
.../.composer.mk
.../tld/Makefile
.../tld/.composer.mk
.../tld/sub/Makefile
.../tld/sub/.composer.mk
```

3.2.5 COMPOSER_DEPENDS

- When doing all-all, Composer will process the current directory before recursing into sub-directories. This reverses that, and sub-directories will be processed first.
- In the example directory tree in COMPOSER_INCLUDE above, the default would be: .../, .../tld, and then .../tld/sub. If the higher-level directories have dependencies on the sub-directories being run first, this will support that by doing them in reverse order, processing them from bottom to top.
- It should be noted that enabling this disables MAKEJOBS, to ensure linear processing, and that it has no effect on install or clean.

3.2.6 COMPOSER_LOG

- Composer appends to a .composed log file in the current directory whenever it executes Pandoc. This provides some accounting, and is used by list to determine which *.md files have been updated since the last time Composer was run.
- This setting can change the name of the log file, or disable it completely (empty value).
- It is removed each time clean is run.

3.2.7 COMPOSER_EXT

- The Markdown file extension Composer uses: *.md. This is for auto-detection of files to add to COMPOSER_TARGETS, files to output for list, and other tasks. This is a widely used standard, including GitHub. Another commonly used extension is: *.markdown.
- In some cases, they do not have any extension, such as README and LICENSE in source code directories. Setting this to an empty value causes them to be detected and output. It also causes all other files to be processed, because it becomes the wildcard *, so use with care. It is likely best to use COMPOSER_TARGETS to explicitly set the targets list in these situations.

3.2.8 COMPOSER_TARGETS

- The list of output files to create or delete with clean and all. Composer does auto-detection using c_type and COMPOSER_EXT, so this does not usually need to be set. Hidden files that start with . are skipped.
- Setting this manually disables auto-detection. It can also include non-file targets added into a .composer.mk file (see Custom Targets).
- The .null target is special, and when used as a value for COMPOSER_TARGETS or COMPOSER_SUBDIRS it will display a message and do nothing. A side-effect of this target is that an actual file or directory named .null will never be created or removed by Composer.
- An empty value triggers auto-detection
- Use config or targets to check the current value.

3.2.9 COMPOSER_SUBDIRS

- The list of sub-directories to recurse into with install, clean, and all. The behavior and configuration is identical to COMPOSER_TARGETS above, including auto-detection and the .null target. Hidden directories that start with . are skipped.
- An empty value triggers auto-detection
- Use config or targets to check the current value.

3.2.10 COMPOSER_IGNORES

- The list of COMPOSER_TARGETS and COMPOSER_SUBDIRS to skip with install, clean, and all. This allows for selective auto-detection, when the list of items to process is larger than those to leave alone.
- Use config to check the current value.

Chapter 4

Composer Targets

4.1 Primary Targets

Target	Purpose
help	Basic help overview (default)
help-all	Console version of README.md (mostly identical)
template	Print settings template: .composer.mk
compose	Document creation engine (see Formatting Variables)
site	Recursively create Bootstrap Websites
install	Current directory initialization: Makefile
install-all	Do install recursively (no overwrite)
install-force	Recursively force overwrite of Makefile files
clean	Remove output files: COMPOSER_TARGETS :: *-clean
clean-all	Do clean recursively: COMPOSER_SUBDIRS
*-clean	Any targets named this way will also be run by clean
all	Create output files: COMPOSER_TARGETS :: *-all
all-all	Do all recursively: COMPOSER_SUBDIRS
*-all	Any targets named this way will also be run by all
list	Print updated files: ∗.md » .composed

4.1.1 help / help-all

- Outputs all of the documentation for Composer. The README.md has a few extra sections covering internal targets, along with reserved target and variable names, but is otherwise identical to the help-all output.

4.1.2 template

- Prints a useful template for creating new .composer.mk files (see Configuration Settings and Templates).

4.1.3 compose

- This is the very core of Composer, and does the actual work of the Pandoc conversion. Details are in the c_type / c_base / c_list section of Formatting Variables.

4.1.4 site

- *(This feature is reserved for a future release to create Bootstrap Websites. It will also include page and post from Special Targets.)*

4.1.5 install / install-all / install-force

- Creates the necessary Makefile files to set up a directory or a directory tree as a Composer archive. By default, it will not overwrite any existing files.
- Doing a simple install will only create a file in the current directory, whereas install-all will recurse through the entire directory tree. A full install-force is the same as install-all, with the exception that it will overwrite all Makefile files.
- The topmost directory will have the Makefile created for it modified to point to Composer.

4.1.6 clean / clean-all / *-clean

- Deletes all COMPOSER_TARGETS output files in the current directory, after first running all *-clean targets, including those for Specials.
- Doing clean-all does the same thing recursively, through all the COMPOSER_SUBDIRS.

4.1.7 all / all-all / *-all

- Creates all COMPOSER_TARGETS output files in the current directory, after first running all *-all targets, including those for Specials.
- Doing all-all does the same thing recursively, through all the COMPOSER_SUBDIRS.

4.1.8 list

- Outputs all the COMPOSER_EXT files that have been modified since COMPOSER_LOG was last updated (see both in Control Variables). Acts as a quick reference to see if anything has changed.
- Since the COMPOSER_LOG file is updated whenever Pandoc is executed, this target will primarily be useful when all is the only target used to create files in the directory.

4.2 Special Targets

There are a few targets considered Specials, that have unique properties:

Base Name	Purpose
book	Concatenate a source list into a single output file
page	*(Reserved for the future site feature)*
post	*(Reserved for the future site feature)*

For each of these base names, there are a standard set of actual targets:

Target	Purpose
%s-clean	Called by clean, removes all %−∗ files
%s-all	Called by all, creates all %−∗ files
%s	Main target, which is a wrapper to %s−all
%-*	Target files will be processed according to the base

4.2.1 book

An example book definition in a .composer.mk file (Quick Start example):

```
book−Composer−v3.0.Manual.html: README.md LICENSE.md
```

This configures it so that books will create Composer−v3.0.Manual.html from README.md and LICENSE.md, concatenated together in order. The primary purpose of this Special is to gather multiple source files in this manner, so that larger works can be comprised of multiple files, such as a book with each chapter in a separate file.

4.2.2 page / post

(Both page and post are reserved for the future site feature, which will build website pages using Bootstrap.)

4.3 Additional Targets

Target	Purpose
debug	Diagnostics, tests targets list in COMPOSER_DEBUGIT
debug-file	Export debug results to a plain text file
check	List system packages and versions (see Requirements)
check-all	Complete check package list, and system information
config	Show values of all Composer Variables
config-all	Complete config, including environment variables
targets	List all available targets for the current directory
_commit	Timestamped Git commit of the current directory tree
_commit-all	Automatic _commit, without $EDITOR step
_release	Full upgrade to current release, repository preparation
_update	Update all included components (see Requirements)
_update-all	Complete _update, including binaries: Pandoc, YQ

4.3.1 debug / debug-file

- This is the tool to use for any support issues. Submit the output file to: composer@garybgenett.net
- Internally, it also runs:
  - test
  - check-all
  - config-all
  - targets
- If issues are occurring when running a particular set of targets, list them in COMPOSER_DEBUGIT.
- For general issues, run in the top-level directory (see Recommended Workflow). For specific issues, run in the directory where the issue is occurring.

For example:

```
make COMPOSER_DEBUGIT="books README.html" debug-file
```

4.3.2 check / check-all / config / config-all / targets

- Useful targets for validating tooling and configurations.
- Use check to see the list of components and their versions, in relation to the system installed versions. Doing check-all will show the complete list of tools that are used by Composer.
- The current values of all Composer Variables is output by config, and config-all will additionally output all environment variables.
- A structured list of detected targets, available Specials, *-clean and *-all targets, COMPOSER_TARGETS, and COMPOSER_SUBDIRS is printed by targets.
- Together, config and targets reveal the entire internal state of Composer.

4.3.3 _commit / _commit-all

- Using the directory structure in Recommended Workflow, .../ is considered the top-level directory. Meaning, it is the last directory before linking to Composer.
- If the top-level directory is a Git repository (it has <directory>.git or <directory>/.git), this target creates a commit of the current directory tree with the title format below.
- For example, if it is run in the .../tld directory, that entire tree would be in the commit, including .../tld/sub. The purpose of this is to create quick and easy checkpoints when working on documentation that does not necessarily fit in a process where there are specific atomic steps being accomplished.

- When this target is run in a Composer directory, it uses itself as the top-level directory.

Commit title format:

```
[Composer CMS v3.0 :: 2022-05-11T12:26:31-07:00]
```

4.3.4 _release / _update / _update-all

- Using the repository configuration (see Repository Versions), these fetch and install all external components.
- The _update-all target also fetches the Pandoc and YQ binaries, whereas _update only fetches the repositories.
- In addition to doing _update-all, _release performs the steps necessary to turn the current directory into a complete clone of Composer.
- If rsync is installed, _release can be used to rapidly replicate Composer, like below.
- One of the unique features of Composer is that everything needed to compose itself is embedded in the Makefile.

Rapid cloning (requires rsync):

```
mkdir .../clone
cd .../clone
make -f .../.Composer/Makefile _release
```

4.4 Internal Targets

Target	Purpose
help-force	Complete README.md content (similar to help-all)
.template-install	The Makefile used by install (see Templates)
.template	The .composer.mk used by template (see Templates)
docs	Extracts embedded files from Makefile, and does all
headers	Series of targets that handle all informational output
headers-template	For testing default headers output
headers-template-all	For testing complete headers output
.make_database	Complete contents of GNU Make internal state
.all_targets	Extracted list of all targets from .make_database
.null	Placeholder to specify or detect empty values
test	Test suite, validates all supported features
test-file	Export test results to a plain text file
check-force	Minimized check output (used for Requirements)
subdirs	Expands COMPOSER_SUBDIRS into *-subdirs-* targets

(None of these are intended to be run directly during normal use, and are only documented for completeness.)

Chapter 5

Reference

5.1 Templates

The install target Makefile template (for reference only):

```
override COMPOSER_MY_PATH := $(abspath $(dir $(lastword $(MAKEFILE_LIST))))
override COMPOSER_TEACHER := $(abspath $(dir $(COMPOSER_MY_PATH)))/Makefile
include $(COMPOSER_TEACHER)
```

Use the template target to create .composer.mk files:

```
# >> Global
# override MAKEJOBS := 1
# override COMPOSER_DOCOLOR :=
# override COMPOSER_DEBUGIT :=
# override COMPOSER_INCLUDE :=
# override COMPOSER_DEPENDS :=
# override COMPOSER_LOG := .composed
# override COMPOSER_EXT := .md
# override c_type := html
# override c_lang := en-US
# override c_css :=
# override c_toc :=
# override c_level := 2
# override c_margin := 0.8in
# override c_margin_top :=
# override c_margin_bottom :=
# override c_margin_left :=
# override c_margin_right :=
# override c_options :=
# >> Local
# override COMPOSER_TARGETS := README.html README.pdf README.epub README.revealjs.html READ
# override COMPOSER_SUBDIRS :=
# override COMPOSER_IGNORES :=
# override c_base := README
# override c_list := README.md
# >> Special
# book-Composer.book.html: README.md LICENSE.md
# page-Composer.page.html: README.md LICENSE.md
# post-Composer.post.html: README.md LICENSE.md
```

5.2 Reserved

Reserved target names, including use as prefixes:

```
.all_targets
.make_database
.null
all
book
books
check
clean
compose
config
debug
docs
headers
help
install
list
page
pages
post
posts
site
subdirs
targets
template
test
_commit
_release
_update
```

Reserved variable names:

```
~
7Z
7Z_VER
BASE64
BASH
BASH_VER
BOOTSTRAP_CMT
BOOTSTRAP_DIR
BOOTSTRAP_LIC
BOOTSTRAP_SRC
CAT
CHECKIT
CHMOD
CLEANER
CODEBLOCK
COLUMNS
COLUMN_2
COMMENTED
COMPOSER
COMPOSER_ART
COMPOSER_BASENAME
COMPOSER_COMPOSER
COMPOSER_CONTENTS
```

COMPOSER_CONTENTS_DIRS
COMPOSER_CONTENTS_FILES
COMPOSER_CSS
COMPOSER_DEBUGIT
COMPOSER_DEBUGIT_ALL
COMPOSER_DEPENDS
COMPOSER_DIR
COMPOSER_DOCOLOR
COMPOSER_DOITALL_check
COMPOSER_DOITALL_clean
COMPOSER_DOITALL_config
COMPOSER_DOITALL_commit
COMPOSER_DOITALL_debug
COMPOSER_DOITALL_all
COMPOSER_DOITALL_install
COMPOSER_DOITALL_test
COMPOSER_DOITALL_update
COMPOSER_EXPORTED
COMPOSER_EXPORTED_NOT
COMPOSER_EXT
COMPOSER_EXT_DEFAULT
COMPOSER_FILENAME
COMPOSER_FIND
COMPOSER_FULLNAME
COMPOSER_HEADLINE
COMPOSER_IGNORES
COMPOSER_INCLUDE
COMPOSER_INCLUDES
COMPOSER_INCLUDES_LIST
COMPOSER_LICENSE
COMPOSER_LOG
COMPOSER_LOG_DEFAULT
COMPOSER_MY_PATH
COMPOSER_NOTHING
COMPOSER_OPTIONS
COMPOSER_PANDOC
COMPOSER_PKG
COMPOSER_REGEX
COMPOSER_REGEX_DEFINE
COMPOSER_REGEX_EVAL
COMPOSER_REGEX_OVERRIDE
COMPOSER_REGEX_PREFIX
COMPOSER_RELEASE
COMPOSER_RESERVED
COMPOSER_RESERVED_SPECIAL
COMPOSER_RESERVED_SPECIAL_TARGETS
COMPOSER_ROOT
COMPOSER_SETTINGS
COMPOSER_SRC
COMPOSER_SUBDIRS
COMPOSER_TAGLINE
COMPOSER_TARGETS
COMPOSER_TEACHER
COMPOSER_TECHNAME
COMPOSER_TIMESTAMP
COMPOSER_TMP

COMPOSER_VERSION
CONFIGS
CONVICT
COREUTILS_VER
CP
CREATOR
CSS_ALT
DATE
DATEMARK
DATENAME
DATESTAMP
DEBUGIT
DEPTH_DEFAULT
DEPTH_MAX
DESC_DOCX
DESC_EPUB
DESC_HTML
DESC_LINT
DESC_LPDF
DESC_PPTX
DESC_PRES
DESC_TEXT
DIFF
DIFFUTILS_VER
DISTRIB
DIST_ICON_v1.0
DIST_SCREENSHOT_v1.0
DIST_SCREENSHOT_v3.0
DIVIDE
DOFORCE
DOITALL
DOMAKE
DO_BOOK
DO_HEREDOC
DO_PAGE
DO_POST
ECHO
ENDOLINE
ENV
EXAMPLE
EXPR
EXTENSION
EXTN_DEFAULT
EXTN_DOCX
EXTN_EPUB
EXTN_HTML
EXTN_LINT
EXTN_LPDF
EXTN_PPTX
EXTN_PRES
EXTN_TEXT
FIND
FINDUTILS_VER
GIT
GIT_OPTS_CONVICT
GIT_REPO

GIT_REPO_DO
GIT_RUN
GIT_RUN_COMPOSER
GIT_VER
GZIP_BIN
GZIP_VER
HEAD
HEADERS
HEADER_L
HEAD_MAIN
HELPOUT
HEREDOC_GITATTRIBUTES
HEREDOC_GITIGNORE
HEREDOC_LICENSE
HEREDOC_REVEALJS_CSS
HEREDOC_TEX_PDF_TEMPLATE
HEREDOC_docs_.composer.mk
INPUT
INSTALL
LESS_BIN
LESS_VER
LINERULE
LISTING
LN
LS
MAKEFILE
MAKEFILE_LIST
MAKEFLAGS
MAKEJOBS
MAKEJOBS_DEFAULT
MAKEJOBS_OPTS
MAKE_DB
MAKE_OPTIONS
MAKE_VER
MARKER
MDVIEWER_CMT
MDVIEWER_CSS
MDVIEWER_CSS_ALT
MDVIEWER_DIR
MDVIEWER_LIC
MDVIEWER_SRC
MKDIR
MV
NEWLINE
NOTHING
NOTHING_IGNORES
NULL
OS_TYPE
OS_UNAME
OUTPUT
OUTPUT_FILENAME
OUT_LICENSE
OUT_MANUAL
OUT_README
PANDOC
PANDOC_BIN

PANDOC_CMT
PANDOC_CMT_DISPLAY
PANDOC_DIR
PANDOC_EXTENSIONS
PANDOC_LIC
PANDOC_LNX_BIN
PANDOC_LNX_DST
PANDOC_LNX_SRC
PANDOC_MAC_BIN
PANDOC_MAC_DST
PANDOC_MAC_SRC
PANDOC_OPTIONS
PANDOC_OPTIONS_DATA
PANDOC_OPTIONS_ERROR
PANDOC_SRC
PANDOC_TEX_PDF
PANDOC_URL
PANDOC_VER
PANDOC_WIN_BIN
PANDOC_WIN_DST
PANDOC_WIN_SRC
PATH_LIST
PRINT
PRINTER
PRINTF
PUBLISH
READ_ALIASES
REALMAKE
REALPATH
REVEALJS_CMT
REVEALJS_CSS
REVEALJS_CSS_THEME
REVEALJS_DIR
REVEALJS_LIC
REVEALJS_LOGO
REVEALJS_SRC
RM
RSYNC
RSYNC_VER
RUNMAKE
SED
SED_VER
SHELL
SORT
SOURCE_INCLUDES
SPECIAL_VAL
SUBDIRS
TABLE_C2
TABLE_M2
TABLE_M3
TAIL
TAR
TARGETS
TAR_VER
TEE
TESTING

TESTING_COMPOSER_DIR
TESTING_COMPOSER_MAKEFILE
TESTING_DIR
TESTING_ENV
TESTING_LOGFILE
TEX_PDF
TEX_PDF_TEMPLATE
TEX_PDF_VER
TITLE_LN
TMPL_DOCX
TMPL_EPUB
TMPL_HTML
TMPL_LINT
TMPL_LPDF
TMPL_PPTX
TMPL_PRES
TMPL_TEXT
TOKEN
TR
TRUE
TYPE_DEFAULT
TYPE_DOCX
TYPE_DO_BOOK
TYPE_DO_PAGE
TYPE_DO_POST
TYPE_EPUB
TYPE_HTML
TYPE_LINT
TYPE_LPDF
TYPE_PPTX
TYPE_PRES
TYPE_TARGETS
TYPE_TEXT
UNAME
UPGRADE
VIM_FOLDING
VIM_OPTIONS
WC
WGET
WGET_PACKAGE
WGET_PACKAGE_DO
WGET_VER
XARGS
YQ
YQ_BIN
YQ_CMT
YQ_CMT_DISPLAY
YQ_DIR
YQ_LIC
YQ_LNX_BIN
YQ_LNX_DST
YQ_LNX_SRC
YQ_MAC_BIN
YQ_MAC_DST
YQ_MAC_SRC
YQ_SRC

YQ_URL
YQ_VER
YQ_WIN_BIN
YQ_WIN_DST
YQ_WIN_SRC
c_base
c_css
c_css_select
c_lang
c_level
c_list
c_margin
c_margin_bottom
c_margin_left
c_margin_right
c_margin_top
c_options
c_toc
c_type
template-print
template-var
template-var-static
headers
headers-dir
headers-file
headers-list
headers-note
headers-release
headers-rm
headers-run
headers-skip
headers-vars
headers-compose
headers-subdirs
help-force-targets-FORMAT
help-force-targets-SECTIONS
help-force-targets-TITLES
help-all-CUSTOM
help-all-DEPENDS
help-all-FORMAT
help-all-GOALS
help-all-LINKS
help-all-LINKS_EXT
help-all-ORDERS
help-all-OVERVIEW
help-all-REQUIRE
help-all-REQUIRE_POST
help-all-SECTION
help-all-SETTINGS
help-all-TARGETS_ADDITIONAL
help-all-TARGETS_INTERNAL
help-all-TARGETS_PRIMARY
help-all-TARGETS_SPECIALS
help-all-TITLE
help-all-VARIABLES_CONTROL
help-all-VARIABLES_FORMAT

```
help-all-VERSIONS
help-all-WORKFLOW
install-Makefile
subdirs-template
targets-list
test-COMPOSER_INCLUDE-done
test-COMPOSER_INCLUDE-init
test-count
test-done
test-fail
test-find
test-hold
test-init
test-load
test-log
test-make
test-mark
test-pwd
test-run
test-speed-init
test-speed-init-load
test-headers
_C
_D
_E
_F
_H
_M
_N
_S
```

Happy Making!

Chapter 6

Composer CMS License

6.1 Copyright

6.2 License

Source: https://www.gnu.org/licenses/gpl-3.0.html

GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007

6.2.1 Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

6.2.2 TERMS AND CONDITIONS

6.2.2.1 0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

6.2.2.2 1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a

major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

6.2.2.3 2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

6.2.2.4 3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

6.2.2.5 4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

6.2.2.6 5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section

7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6.2.2.7 6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has

substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

6.2.2.8 7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

6.2.2.9 8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

6.2.2.10 9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

6.2.2.11 10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

6.2.2.12 11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a

consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

6.2.2.13 12. No Surrender of Others’ Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

6.2.2.14 13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

6.2.2.15 14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address

new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

6.2.2.16 15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

6.2.2.17 16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (ITHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

6.2.2.18 17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

6.2.3 END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.> Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see https://www.gnu.org/licenses/.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

<program> Copyright (C) <year> <name of author> This program comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see https://www.gnu.org/licenses/.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read https://www.gnu.org/licenses/why-not-lgpl.html.

End Of File