ROCm Docs Core

Release 1.33.1

Advanced Micro Devices, Inc.

Mar 21, 2026

CONTENTS

ROCm Docs Core is a collection of utilities, styling, scripts and additional HTML content common to all ROCm projects' documentation.

ROCm Docs Core is distributed as a pip package available from PyPi as rocm-docs-core

CHAPTER

ONE

OVERVIEW

User Guide

- *Spell Check*
- *Linting*
- *Linking*
- *Article Info*
- *Doxygen Integration*

Developer Guide

- *Just*
- *Python Linting*
- *Commitizen*
- *External Intersphinx Project Mapping*
- *Dependabot*

Demo Doxygen Documentation Integration

- *ROCm Docs Core Demo Doxygen Docs*

CHAPTER

TWO

USER GUIDE

The User Guide describes how users can make use of functionality in `rocm-docs-core`

2.1 Spell Check

`rocm-docs-core` has spell checks run on every pull request (PR) via GitHub Actions.

If a PR fails spell check, the errors must be addressed before it can be merged.

The results of a spell check is viewable in the "Checks" tab for a PR.

2.1.1 Types of Errors

2.1.1.1 Spelling Mistake

View the "Details" for a spell check and fix the misspelled words.

2.1.1.2 New words or Acronyms

Spell check may flag errors if it does not recognize a word.

The reason could be that the word is not familiar to the dictionary used or an unfamiliar acronym.

To get spell check to recognize this word, add it to the `.wordlist.txt` file, located at the root of the project (for this project, `rocm-docs-core`, that would be the `rocm-docs-core` folder).

2.1.1.3 Special keywords

If the word is not meant to be a dictionary word, but is still valid technical terminology, wrap the word with the backtick (`) key.

2.1.1.3.1 Keyword examples

- File or folder names
- Commands or arguments
- Code blocks
- Names of executables or binaries

2.1.2 More Information

For more information, see the GitHub Action spellcheck-github-actions

2.2 Linting

`rocm-docs-core` has linting to ensure correct Markdown and ReStructuredText formatting on every pull request (PR) via GitHub Actions.

If a PR fails linting, the errors must be addressed before it can be merged.

The results of linting is viewable in the "Checks" tab for a PR.

2.2.1 Markdown Linting

The current linter used for Markdown is `markdownlint`, which uses the following rules.

2.2.1.1 Optional

`markdownlint` can also be installed and run locally using the `markdownlint` extension on Visual Studio Code.

2.3 Linking

Markdown

reStructuredText (RST)

2.3.1 Linking in Markdown

2.3.1.1 Markdown

2.3.1.1.1 Cross References to Other Projects

The `projects.yaml` configuration file contains the names of projects that should be used when making links that cross-reference documentation sites.

When making links that cross-reference documentation sites, the following format should be used:

```
{doc}`Text here<project_name:path/to/page_name>`
```

Cross-references are achieved via Intersphinx. For more information, refer to the Sphinx documentation or Read the Docs documentation on Intersphinx.

2.3.1.1.1.1 Example

The following Markdown:

```
{doc}`ROCm Documentation<rocm:about/license>`
```

will be rendered as the following link:

ROCm Documentation

2.3.1.1.2 Relative Links to Current Project

2.3.1.1.2.1 Example

The following Markdown:

```
[Link Text](../index)
```

will be rendered as the following link:

Link Text

2.3.1.1.3 Absolute Links to External Sites

For other links, usual Markdown conventions should be used.

2.3.1.1.3.1 Example

The following Markdown:

```
[Link Text](https://github.com/ROCm/ROCm)
```

will be rendered as the following link:

Link Text

2.3.2 Linking in RST

2.3.2.1 reStructuredText (RST)

2.3.2.1.1 Cross References to Arbitrary Locations in Other Projects

The projects.yaml configuration file contains the names of projects that should be used when making links that cross-reference documentation sites.

Cross references to anchors or arbitrary locations in documentation can be done using labels.

See the Sphinx documentation on cross-referencing arbitrary locations for information on labels.

The format using a label would appear as follows:

```
:ref:`Text here<project_name:label_name>`
```

Cross-references are achieved via Intersphinx. For more information, refer to the Sphinx documentation or Read the Docs documentation on Intersphinx.

2.3.2.1.1.1 Example

The following RST:

```
:ref:`ROCm for AI Install<rocm:rocm-for-ai-install>`
```

will be rendered as the following link:

ROCm for AI Install

2.3.3 Linking in the Table of Contents

2.3.3.1 Syntax

Variables of the form `${<variable>}` are substituted, currently the following list is supported:

- `${branch}` or `{branch}`: the name of the current branch
- `${url}` or `{url}`: GitHub URL of the current project
- `${project:<project_name>}`: base URL of the documentation of `<project_name>` based on Intersphinx mapping

2.3.3.1.1 Example

```
- url: "{url}/tree/{branch}"
- url: ${project:python}
- url: ${project:rocm-docs-core}
- url: ${project:hipify}
```

2.4 Doxygen Integration

2.4.1 What

`doxysphinx` is a package that handles integration of Doxygen and Sphinx documentation.

`doxysphinx` allows displaying Doxygen documentation in the Sphinx documentation.

`rocm-docs-core` applies some additional formatting and styling on top of this to stay in line with our themes.

Since `doxysphinx` automatically integrates the Doxygen documentation, developers only have to update the documentation strings in the source code if they are formatted for Doxygen.

For more information on `doxysphinx`, see the GitHub repository or the doxysphinx documentation.

2.4.2 How

Some examples of how to use `doxysphinx` with `rocm-docs-core` are included below.

Assuming Doxygen documentation is already configured correctly, several changes must be made to the configuration file (`conf.py`) located in the `docs` folder and the requirements files (`requirements.in` and `requirements.txt`) located in the `sphinx` folder.

For the configuration file:

- Include the `rocm_docs.doxygen` extension in the `extensions` list.
- Include the path to the Doxygen configuration in `doxygen_root`. For ROCm projects, this value is usually `doxygen`.
- Set `doxysphinx_enabled` to True.
- Define a `doxygen_project` dictionary and set a `name` and `path`. For ROCm projects, the value of path is usually `doxygen/xml`.

For the requirements files:

- Specify the `api_reference` in the `requirements.in` (example: `rocm-docs-core[api_reference]==0.36.0`)
- Use `pip-tools` to compile the `requirements.in`
  - `pip install pip-tools`
  - `pip-compile requirements.in --resolver=backtracking`

Then add the Doxygen output to the table of contents (`_toc.yml.in`).

Optionally, specify custom style sheets to use in the Doxygen configuration (`Doxyfile`). These style sheets are a part of `rocm-docs-core`.

- `HTML_HEADER`
- `HTML_FOOTER`
- `HTML_STYLESHEET`

- `HTML_EXTRA_STYLESHEET`

When building the documentation with the API reference enabled, the console output will also make configuration recommendations to make documentation builds succeed. If documentation builds are still failing, please follow the recommendations.

This project has *Demo Doxygen Docs here*. See the source code for details.

The `tests` folder in the `rocm-docs-core` project on GitHub also has example configuration files.

See this PR for a simple example of adding a Doxygen code snippet.

2.5 Article Info

Article info is disabled by default and must be enabled in `conf.py`.

2.5.1 Settings

Legend: `setting name (setting data type):` *explanation*

- `setting_all_article_info (bool)`: Setting this value to true will enable article info for all pages and use default values. See possible settings for default values.
- `all_article_info_os (list[str])`: Determines which supported OS appear in the article info. Possible strings are `"linux"` and `"windows"`. Default value is `["linux"]`.
- `all_article_info_author (str)`: Determines the author. Default is empty string.
- `all_article_info_date (str)`: Determines date of publication. Default is the date the file was last modified in git.
- `all_article_info_read_time (str)`: Determines the read time. Default is calculated based on the number of words in the file.
- `article_pages (list[dict])`: Used for specific settings for a page. These override any of the general settings above (eg: `all_article_info_<field>`).

Example:

```
article_pages = [
    {
        "file":"index",
        "os":["linux", "windows"],
        "author":"Author: AMD",
        "date":"2023-05-01",
        "read-time":"2 min read"
    },
    {"file":"developer_guide/commitizen"}
]
```

CHAPTER

THREE

DEVELOPER GUIDE

The Developer Guide provides additional information on the processes and toolchains in `rocm-docs-core`

3.1 Just

`just` is a command runner used to quickly setup an environment for documentation development.

The aim is to make contributing to `rocm-docs-core` more approachable by providing ready-to-use environments for development.

3.1.1 Usage

3.1.1.1 Setting up a development environment

```
just devenv
```

This creates a Python virtual environment, installs the dependencies, and sets up the pre-commit hooks.

3.1.1.2 Running linting commands

These commands work on both Linux and Windows

```
just check-codestyle
```

Check files for formatting errors and report them.

```
just fix-codestyle
```

Automatically fix errors that have suggested fixes.

3.1.2 GitHub

GitHub Actions CI is extended to run these tools on PRs (using the `just`-based entry-points).

Development container setup and settings are added for Visual Studio Development containers (and GitHub Codespaces) and Gitpod.

- VS Code Dev Containers Guide
  - Dev Container Dockerfile and Configuration
- Ona (Gitpod) Guide
  - Gitpod Configuration

3.2 Python Linting

The following list of tools is used for checking correctness in Python code. These tools are set up to run as git pre-commit hooks via pre-commit.

- `Ruff` for linting
  - Usage
- `Mypy` for static type checking
  - Usage
- `Black` for code formatting
  - Usage
- `isort` for import sorting
  - Usage

Some non-Python-specific hooks are also enabled:

- Check `yaml`, `toml`, and `json` validity
- Check for no trailing whitespaces and additional newline at the end of files

3.3 Commitizen

The ROCm Docs Core repository uses Commitizen to enforce Conventional Commits and implement automatic release tagging based on commit messages.

3.3.1 Commit message rules

A commit message should like like this:

```
<type>[optional scope]: <subject>

[optional body]

[optional footer(s)]
```

Commit messages start with a header line that notes the type of change, followed by an optional list of scopes, then the description of the change.

Type must be one of:

- **`build`**: Changes that affect the build system or external dependencies (example scopes: `gulp`, `broccoli`, `npm`)
- **`ci`**: Changes to our CI configuration files and scripts
- **`docs`**: Documentation only changes
- **`feat`**: A new feature, *corresponds to a minor version bump*
- **`fix`**: A bug fix, *correspond to a patch version bump*
- **`perf`**: A code change that improves performance
- **`refactor`**: A code change that neither fixes a bug nor adds a feature
- **`style`**: Changes that do not affect the meaning of the code (white-space, formatting, missing semi-colons, etc)
- **`test`**: Adding missing tests or correcting existing tests

- **chore**: Other changes that don't modify source code, test or build files (e.g changing `.gitignore`)

The **scope** if included should be the area the change affects in parentheses i.e. `(theme)` for theming changes or `(deps)` for changes to the list of dependencies. Multiple scopes can be given and `(all)` may be used if change affects the entire project.

The **subject** should be a succinct description of the change in present imperative tense (i.e. *update wording* not *updated wording*)

A longer description can be included in the **body**, including the motivation of the change and comparison with the previous version. Comparisons with the prior state if shouldn't detail literal changes, those are recorded by the git history, but the semantic or higher level effect of the modifications.

The footer contains information about **Breaking Changes** and references to GitHub issues that the change closes or relates to. Breaking changes should be noted with the words `BREAKING CHANGE:` followed by a space and the description of the change. Including a breaking change in a commit will result in a major version bump of the next release including such commit.

For a more detailed description of conventional commits please refer to the specification linked above.

3.3.2 Committing with Commitizen

Commitizen contains a utility to walk through making a commit following the rules above.

3.3.2.1 Installation

Commitizen is marked as a "development" dependency of `rocm-docs-core`, installable via `pip install .[development]` from the root of the repository.

3.3.2.2 Usage

Run `cz commit` or `cz c` to start making a commit after staging the files to be added. Commitizen will ask for each part of the commit message in turn.

Refer to the documentation of Commitizen (linked above) on other options, and other features of Commitizen.

3.4 External Intersphinx Project Mapping

Projects should be defined in `projects.yaml` and should set which key they correspond to by setting external_projects_current_project to this key.

3.4.1 Example

Given the following `projects.yaml` file:

```
projects:
  python: https://docs.python.org/3/
  rtd: https://docs.readthedocs.io/en/stable/
  sphinx: https://www.sphinx-doc.org/en/master/
  rocm-docs-core: https://rocm.docs.amd.com/projects/rocm-docs-core/en/${version}
```

The `conf.py` from `rocm-docs-core` should contain:

```
external_projects_current_project = "rocm-docs-core"
```

Adds support for specifying the "development" branch for each project defined in `projects.yaml`. This is achieved by setting the `development_branch` key for the project:

```
projects:
  ...
  project:
    target: <target-url>
    development_branch: master
```

With this the mapping between projects is changed to the following:

Development branches map between each other. Essentially, links on project A's `develop` branch point to the project B's `master` branch. Vice-versa if A has set its `development_branch` to `develop` and B sets it to `master`.

Symbolic versions "latest" and "stable" map to themselves in other projects.

Any other branch maps to "latest".

3.4.2 Explicitly list external projects

By default the inventories of all external projects defined in `projects.yaml` will be downloaded. This can take a long time as it requires a network request for each external project.

The `external_projects` configuration option can be set to a list with the names of remote projects to fetch inventories from & enable links to. The list must be a subset of the project names defined in `projects.yaml`. The default value of `"all"` means to fetch all projects.

Intersphinx references to projects that are not in `external_projects` will not be resolved. References in the the TOC like `${project:project_name}` will continue to be resolved to the URL of `project_name`, even if `project_name` is not set in `external_projects` (but it's defined in `projects.yaml`).

3.5 Dependabot

Dependabot is automated dependency management tool. It is used to keep ROCm documentation dependencies up to date.

Here is an example PR that dependabot made on the ROCm repository

Dependabot Configuration used by this project

GitHub Dependabot Configuration Guide

CHAPTER

FOUR

ROCM DOCS CORE DEMO DOXYGEN DOCS

4.1 Modules

4.2 Namespaces

4.2.1 Namespace List

4.2.2 Namespace Members

4.2.2.1 Namespace Members

4.2.2.2 Namespace Members

4.3 Classes

4.3.1 Class List

4.3.2 Class Index

4.3.3 Class Hierarchy

4.3.4 Class Members

4.3.4.1 Class Members

4.3.4.2 Class Members - Functions

4.3.4.3 Class Members - Variables

4.4 Files

4.4.1 File List

4.4.2 File Members

4.4.2.1 File Members

4.4.2.2 File Members

4.4.2.3 File Members

4.4.2.4 File Members

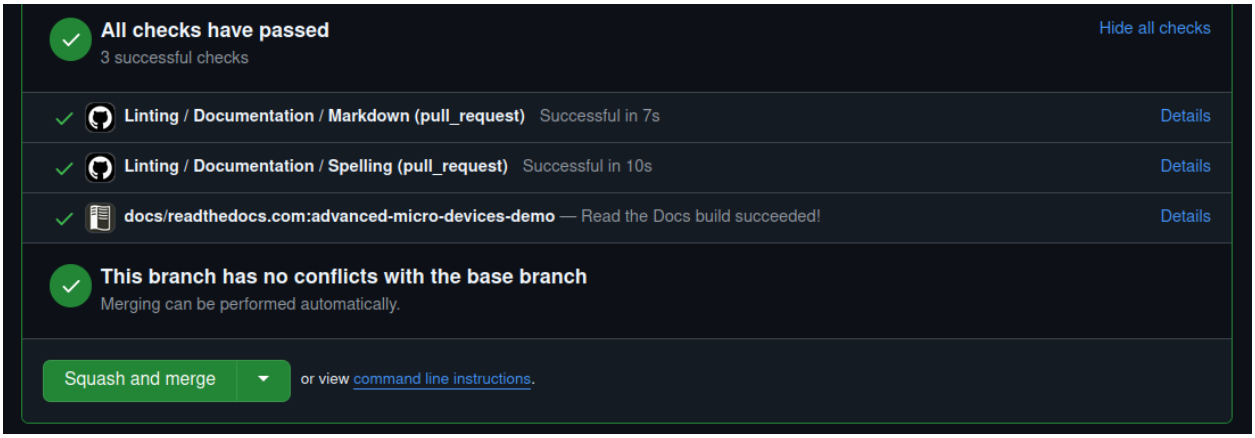
CHAPTER

FIVE

BUILDING DOCUMENTATION

5.1 GitHub

If you open a pull request and scroll down to the summary panel, there is a commit status section. Next to the line `docs/readthedocs.com:advanced-micro-devices-demo`, there is a `Details` link. If you click this, it takes you to the Read the Docs build for your pull request.



If you don't see this line, click `Show all checks` to get an itemized view.

5.2 Command line

You can build our documentation via the command line using Python.

See the `build.tools.python` setting in the Read the Docs configuration file for the Python version used by Read the Docs to build documentation.

See the Python requirements file for Python packages needed to build the documentation.

Use the Python Virtual Environment (`venv`) and run the following commands from the project root:

```
python3 -mvenv .venv

.venv/bin/python     -m pip install -r docs/sphinx/requirements.txt
.venv/bin/python     -m sphinx -T -E -b html -d _build/doctrees -D language=en docs _
↪build/html
```

Navigate to `_build/html/index.html` and open this file in a web browser.

5.3 Visual Studio Code

With the help of a few extensions, you can create a productive environment to author and test documentation locally using Visual Studio (VS) Code. Follow these steps to configure VS Code:

1. Install the required extensions:
   - Python: `(ms-python.python)`
   - Live Server: `(ritwickdey.LiveServer)`
2. Add the following entries to `.vscode/settings.json`.

```
{
  "liveServer.settings.root": "/.vscode/build/html",
  "liveServer.settings.wait": 1000,
  "python.terminal.activateEnvInCurrentTerminal": true
}
```

   - `liveServer.settings.root`: Sets the root of the output website for live previews. Must be changed alongside the `tasks.json` command.
   - `liveServer.settings.wait`: Tells the live server to wait with the update in order to give Sphinx time to regenerate the site contents and not refresh before the build is complete.
   - `python.terminal.activateEnvInCurrentTerminal`: Activates the automatic virtual environment, so you can build the site from the integrated terminal.
3. Add the following tasks to `.vscode/tasks.json`.

```
{
  "version": "2.0.0",
  "tasks": [
    {
      "label": "Build Docs",
      "type": "process",
      "windows": {
        "command": "${workspaceFolder}/.venv/Scripts/python.exe"
      },
      "command": "${workspaceFolder}/.venv/bin/python3",
      "args": [
        "-m",
        "sphinx",
        "-j",
        "auto",
        "-T",
        "-b",
        "html",
        "-d",
        "${workspaceFolder}/.vscode/build/doctrees",
        "-D",
        "language=en",
        "${workspaceFolder}/docs",
        "${workspaceFolder}/.vscode/build/html"
      ],
      "problemMatcher": [
        {
```

(continues on next page)

(continued from previous page)

```
            "owner": "sphinx",
            "fileLocation": "absolute",
            "pattern": {
              "regexp": "^(?:.*\\.{3}\\s+)?(\\/[^:]*|[a-zA-Z]:\\\\[^:]*):(\\d+):\\
↪s+(WARNING|ERROR):\\s+(.*)$",
              "file": 1,
              "line": 2,
              "severity": 3,
              "message": 4
            }
          },
          {
          "owner": "sphinx",
            "fileLocation": "absolute",
            "pattern": {
              "regexp": "^(?:.*\\.{3}\\s+)?(\\/[^:]*|[a-zA-Z]:\\\\[^:]*):{1,2}\\
↪s+(WARNING|ERROR):\\s+(.*)$",
              "file": 1,
              "severity": 2,
              "message": 3
            }
          }
        ],
        "group": {
          "kind": "build",
          "isDefault": true
        }
      }
    ]
  }
```

Implementation detail: two problem matchers were needed to be defined, because VS Code doesn't tolerate some problem information being potentially absent. While a single regex could match all types of errors, if a capture group remains empty (the line number doesn't show up in all warning/error messages) but the `pattern` references said empty capture group, VS Code discards the message completely.

4. Configure the Python virtual environment (`venv`).

   From the Command Palette, run `Python: Create Environment`. Select `venv` environment and `docs/sphinx/requirements.txt`.

5. Build the docs.

   Launch the default build task using one of the following options:

   - A hotkey (the default is `Ctrl+Shift+B`)
   - Issuing the `Tasks: Run Build Task` from the Command Palette

6. Open the live preview.

   Navigate to the site output within VS Code: right-click on `.vscode/build/html/index.html` and select `Open with Live Server`. The contents should update on every rebuild without having to refresh the browser.

CHAPTER

SIX

ROCM DOCUMENTATION TOOLCHAIN

ROCm documentation relies on several open source toolchains and sites. Here is a list of a few important ones.

6.1 rocm-docs-core

rocm-docs-core is an AMD-maintained project that applies customizations for the ROCm documentation. This project is the tool most ROCm repositories use as part of their documentation build pipeline. It is available as a pip package on PyPI.

6.2 Sphinx

Sphinx is a documentation generator originally used for Python. It is now widely used in the open source community.

6.2.1 Sphinx External ToC

Sphinx External ToC is a Sphinx extension used for ROCm documentation navigation. This tool generates a navigation menu on the left based on a YAML file (`_toc.yml.in`) that contains the table of contents.

6.2.2 Sphinx-book-theme

Sphinx-book-theme is a Sphinx theme that defines the base appearance for ROCm documentation. ROCm documentation applies some customization, such as a custom header and footer, on top of the Sphinx Book Theme.

6.2.3 Sphinx Design

Sphinx design is a Sphinx extension that adds design functionality. ROCm documentation uses Sphinx Design for grids, cards, and synchronized tabs.

6.3 Doxygen

Doxygen is a documentation generator that extracts information from in-code comments. It is used for API documentation.

6.4 Breathe

Breathe is a Sphinx plugin for integrating Doxygen content.

6.5 MyST

Markedly Structured Text (MyST) is an extended flavor of Markdown (CommonMark) influenced by reStructuredText (RST) and Sphinx. It is integrated into the ROCm documentation with the `myst-parser` Sphinx extension.

See the MyST syntax cheat sheet at the Jupyter Book site.

6.6 Read the Docs

Read the Docs is the service that builds and hosts the HTML version of the ROCm documentation.

CHAPTER

SEVEN

LICENSE

rights are labeled or categorized. For purposes of this Public License, the rights specified in Section 2(b)(1)-(2) are not Copyright and Similar Rights.

d. Effective Technological Measures means those measures that, in the absence of proper authority, may not be circumvented under laws fulfilling obligations under Article 11 of the WIPO Copyright Treaty adopted on December 20, 1996, and/or similar international agreements.

e. Exceptions and Limitations means fair use, fair dealing, and/or any other exception or limitation to Copyright and Similar Rights that applies to Your use of the Licensed Material.

f. Licensed Material means the artistic or literary work, database, or other material to which the Licensor applied this Public License.

g. Licensed Rights means the rights granted to You subject to the terms and conditions of this Public License, which are limited to all Copyright and Similar Rights that apply to Your use of the Licensed Material and that the Licensor has authority to license.

h. Licensor means the individual(s) or entity(ies) granting rights under this Public License.

i. Share means to provide material to the public by any means or process that requires permission under the Licensed Rights, such as reproduction, public display, public performance, distribution, dissemination, communication, or importation, and to make material available to the public including in ways that members of the public may access the material from a place and at a time individually chosen by them.

j. Sui Generis Database Rights means rights other than copyright resulting from Directive 96/9/EC of the European Parliament and of the Council of 11 March 1996 on the legal protection of databases, as amended and/or succeeded, as well as other essentially equivalent rights anywhere in the world.

k. You means the individual or entity exercising the Licensed Rights under this Public License. Your has a corresponding meaning.

Section 2 – Scope.

a. License grant.

```
1. Subject to the terms and conditions of this Public License,
   the Licensor hereby grants You a worldwide, royalty-free,
   non-sublicensable, non-exclusive, irrevocable license to
   exercise the Licensed Rights in the Licensed Material to:

     a. reproduce and Share the Licensed Material, in whole or
        in part; and

     b. produce, reproduce, and Share Adapted Material.

2. Exceptions and Limitations. For the avoidance of doubt, where
   Exceptions and Limitations apply to Your use, this Public
   License does not apply, and You do not need to comply with
   its terms and conditions.

3. Term. The term of this Public License is specified in Section
   6(a).

4. Media and formats; technical modifications allowed. The
   Licensor authorizes You to exercise the Licensed Rights in
   all media and formats whether now known or hereafter created,
   and to make technical modifications necessary to do so. The
   Licensor waives and/or agrees not to assert any right or
```

(continues on next page)

(continued from previous page)

```
   authority to forbid You from making technical modifications
   necessary to exercise the Licensed Rights, including
   technical modifications necessary to circumvent Effective
   Technological Measures. For purposes of this Public License,
   simply making modifications authorized by this Section 2(a)
   (4) never produces Adapted Material.

5. Downstream recipients.

     a. Offer from the Licensor -- Licensed Material. Every
        recipient of the Licensed Material automatically
        receives an offer from the Licensor to exercise the
        Licensed Rights under the terms and conditions of this
        Public License.

     b. No downstream restrictions. You may not offer or impose
        any additional or different terms or conditions on, or
        apply any Effective Technological Measures to, the
        Licensed Material if doing so restricts exercise of the
        Licensed Rights by any recipient of the Licensed
        Material.

6. No endorsement. Nothing in this Public License constitutes or
   may be construed as permission to assert or imply that You
   are, or that Your use of the Licensed Material is, connected
   with, or sponsored, endorsed, or granted official status by,
   the Licensor or others designated to receive attribution as
   provided in Section 3(a)(1)(A)(i).
```

b. Other rights.

```
1. Moral rights, such as the right of integrity, are not
   licensed under this Public License, nor are publicity,
   privacy, and/or other similar personality rights; however, to
   the extent possible, the Licensor waives and/or agrees not to
   assert any such rights held by the Licensor to the limited
   extent necessary to allow You to exercise the Licensed
   Rights, but not otherwise.

2. Patent and trademark rights are not licensed under this
   Public License.

3. To the extent possible, the Licensor waives any right to
   collect royalties from You for the exercise of the Licensed
   Rights, whether directly or through a collecting society
   under any voluntary or waivable statutory or compulsory
   licensing scheme. In all other cases the Licensor expressly
   reserves any right to collect such royalties.
```

Section 3 – License Conditions.

Your exercise of the Licensed Rights is expressly made subject to the following conditions.

a. Attribution.

```
1. If You Share the Licensed Material (including in modified
   form), You must:

     a. retain the following if it is supplied by the Licensor
        with the Licensed Material:

          i. identification of the creator(s) of the Licensed
             Material and any others designated to receive
             attribution, in any reasonable manner requested by
             the Licensor (including by pseudonym if
             designated);

         ii. a copyright notice;

        iii. a notice that refers to this Public License;

         iv. a notice that refers to the disclaimer of
             warranties;

          v. a URI or hyperlink to the Licensed Material to the
             extent reasonably practicable;

     b. indicate if You modified the Licensed Material and
        retain an indication of any previous modifications; and

     c. indicate the Licensed Material is licensed under this
        Public License, and include the text of, or the URI or
        hyperlink to, this Public License.

2. You may satisfy the conditions in Section 3(a)(1) in any
   reasonable manner based on the medium, means, and context in
   which You Share the Licensed Material. For example, it may be
   reasonable to satisfy the conditions by providing a URI or
   hyperlink to a resource that includes the required
   information.

3. If requested by the Licensor, You must remove any of the
   information required by Section 3(a)(1)(A) to the extent
   reasonably practicable.

4. If You Share Adapted Material You produce, the Adapter's
   License You apply must not prevent recipients of the Adapted
   Material from complying with this Public License.
```

Section 4 – Sui Generis Database Rights.

Where the Licensed Rights include Sui Generis Database Rights that apply to Your use of the Licensed Material:

a. for the avoidance of doubt, Section 2(a)(1) grants You the right to extract, reuse, reproduce, and Share all or a substantial portion of the contents of the database;

b. if You include all or a substantial portion of the database contents in a database in which You have Sui Generis Database Rights, then the database in which You have Sui Generis Database Rights (but not its individual contents) is Adapted Material; and

c. You must comply with the conditions in Section 3(a) if You Share all or a substantial portion of the contents of the database.

For the avoidance of doubt, this Section 4 supplements and does not replace Your obligations under this Public License where the Licensed Rights include other Copyright and Similar Rights.

Section 5 – Disclaimer of Warranties and Limitation of Liability.

a. UNLESS OTHERWISE SEPARATELY UNDERTAKEN BY THE LICENSOR, TO THE EXTENT POSSIBLE, THE LICENSOR OFFERS THE LICENSED MATERIAL AS-IS AND AS-AVAILABLE, AND MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND CONCERNING THE LICENSED MATERIAL, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHER. THIS INCLUDES, WITHOUT LIMITATION, WARRANTIES OF TITLE, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, ABSENCE OF LATENT OR OTHER DEFECTS, ACCURACY, OR THE PRESENCE OR ABSENCE OF ERRORS, WHETHER OR NOT KNOWN OR DISCOVERABLE. WHERE DISCLAIMERS OF WARRANTIES ARE NOT ALLOWED IN FULL OR IN PART, THIS DISCLAIMER MAY NOT APPLY TO YOU.

b. TO THE EXTENT POSSIBLE, IN NO EVENT WILL THE LICENSOR BE LIABLE TO YOU ON ANY LEGAL THEORY (INCLUDING, WITHOUT LIMITATION, NEGLIGENCE) OR OTHERWISE FOR ANY DIRECT, SPECIAL, INDIRECT, INCIDENTAL, CONSEQUENTIAL, PUNITIVE, EXEMPLARY, OR OTHER LOSSES, COSTS, EXPENSES, OR DAMAGES ARISING OUT OF THIS PUBLIC LICENSE OR USE OF THE LICENSED MATERIAL, EVEN IF THE LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH LOSSES, COSTS, EXPENSES, OR DAMAGES. WHERE A LIMITATION OF LIABILITY IS NOT ALLOWED IN FULL OR IN PART, THIS LIMITATION MAY NOT APPLY TO YOU.

c. The disclaimer of warranties and limitation of liability provided above shall be interpreted in a manner that, to the extent possible, most closely approximates an absolute disclaimer and waiver of all liability.

Section 6 – Term and Termination.

a. This Public License applies for the term of the Copyright and Similar Rights licensed here. However, if You fail to comply with this Public License, then Your rights under this Public License terminate automatically.

b. Where Your right to use the Licensed Material has terminated under Section 6(a), it reinstates:

```
  1. automatically as of the date the violation is cured, provided
     it is cured within 30 days of Your discovery of the
     violation; or

  2. upon express reinstatement by the Licensor.

For the avoidance of doubt, this Section 6(b) does not affect any
right the Licensor may have to seek remedies for Your violations
of this Public License.
```

c. For the avoidance of doubt, the Licensor may also offer the Licensed Material under separate terms or conditions or stop distributing the Licensed Material at any time; however, doing so will not terminate this Public License.

d. Sections 1, 5, 6, 7, and 8 survive termination of this Public License.

Section 7 – Other Terms and Conditions.

a. The Licensor shall not be bound by any additional or different terms or conditions communicated by You unless expressly agreed.

b. Any arrangements, understandings, or agreements regarding the Licensed Material not stated herein are separate from and independent of the terms and conditions of this Public License.

Section 8 – Interpretation.

a. For the avoidance of doubt, this Public License does not, and shall not be interpreted to, reduce, limit, restrict, or impose conditions on any use of the Licensed Material that could lawfully be made without permission under this Public License.

b. To the extent possible, if any provision of this Public License is deemed unenforceable, it shall be automatically reformed to the minimum extent necessary to make it enforceable. If the provision cannot be reformed, it shall be severed from this Public License without affecting the enforceability of the remaining terms and conditions.

c. No term or condition of this Public License will be waived and no failure to comply consented to unless expressly agreed to by the Licensor.

d. Nothing in this Public License constitutes or may be interpreted as a limitation upon, or waiver of, any privileges and immunities that apply to the Licensor or You, including from the legal processes of any jurisdiction or authority.

===

Creative Commons is not a party to its public licenses. Notwithstanding, Creative Commons may elect to apply one of its public licenses to material it publishes and in those instances will be considered the "Licensor." The text of the Creative Commons public licenses is dedicated to the public domain under the CC0 Public Domain Dedication. Except for the limited purpose of indicating that material is shared under a Creative Commons public license or as otherwise permitted by the Creative Commons policies published at creativecommons.org/policies, Creative Commons does not authorize the use of the trademark "Creative Commons" or any other trademark or logo of Creative Commons without its prior written consent including, without limitation, in connection with any unauthorized modifications to any of its public licenses or any other arrangements, understandings, or agreements concerning use of licensed material. For the avoidance of doubt, this paragraph does not form part of the public licenses.

Creative Commons may be contacted at creativecommons.org.