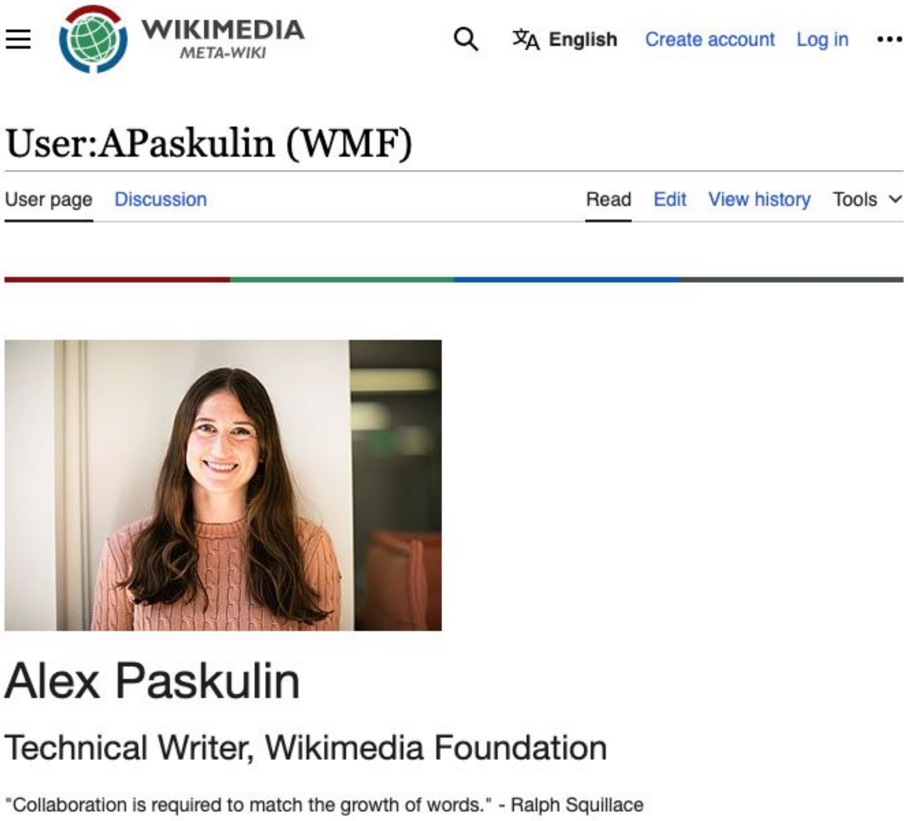
WIKIMEDIA
FOUNDATION

Technical Documentation 101

Alexandra Paskulin
Wiki Mentor Africa Hackathon
January 20, 2024

About me

- Based in California, USA
- Joined the Wikimedia Foundation in 2019
- Part of the Technical Documentation Team
- IRC/Phabricator: apaskulin



User:APaskulin (WMF)

Alex Paskulin

Technical Writer, Wikimedia Foundation

"Collaboration is required to match the growth of words." - Ralph Squillace

https://meta.wikimedia.org/wiki/User:APaskulin_(WMF)

Today’s session

1. Defining our terms
2. Why write documentation
3. Guiding principles
4. Types of documentation
   a. Code comments
   b. Developer documentation
   c. User documentation
   d. Project documentation
5. Organizing documentation pages
6. Structuring the pages
7. Writing style
8. Working on documentation
9. Resources and getting help

WIKIMEDIA FOUNDATION

1. Defining our terms

Documentation is writing that explains or instructs.

Documentation works when users can:

- **find** the information they need,
- **understand** what they find, and
- **act** on it confidently.

"Five Steps to Plain Language", centerforplainlanguage.org

WIKIMEDIA
FOUNDATION

2. Why write documentation

Knowledge is power.

Documentation helps us:

- multiply our efforts and
- show empathy for each other.

3. Guiding principles

For every document, consider:

- who will use it
- how will they find it
- what are they trying to do

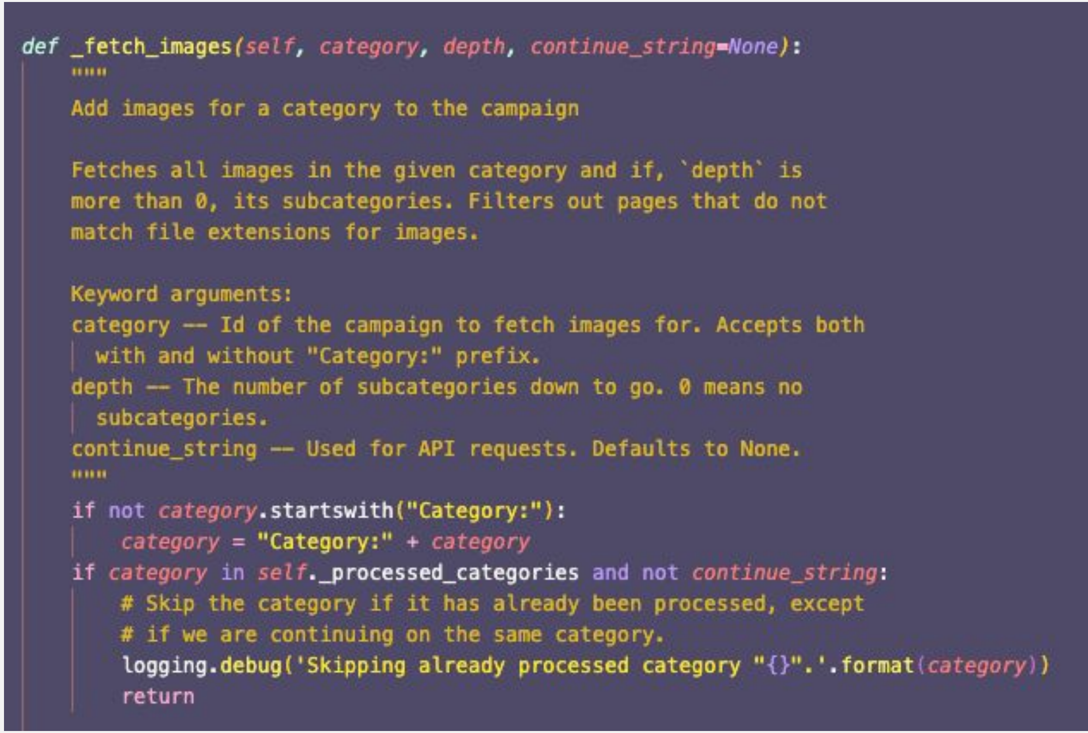
WIKIMEDIA
FOUNDATION

4. Types of documentation

Code comments

- Audience: developers
- Stored in code files
- Uses language-specific syntax

labs-tools-Isa/isa/campaigns/image_updater.py



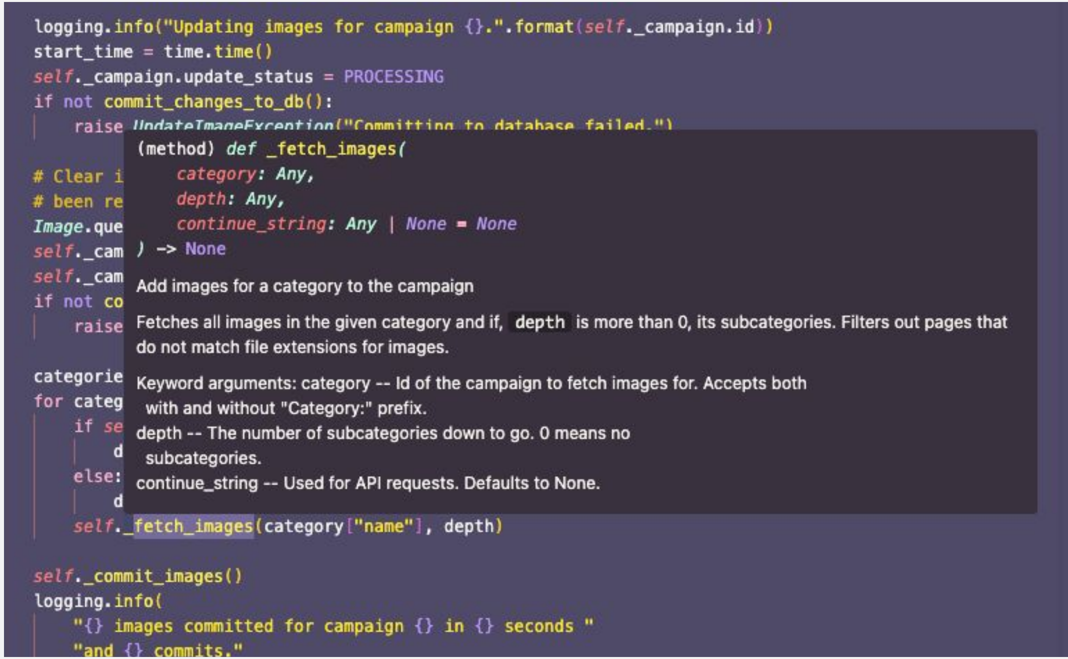
```
def _fetch_images(self, category, depth, continue_string=None):
    """
    Add images for a category to the campaign

    Fetches all images in the given category and if, `depth` is
    more than 0, its subcategories. Filters out pages that do not
    match file extensions for images.

    Keyword arguments:
    category -- Id of the campaign to fetch images for. Accepts both
      with and without "Category:" prefix.
    depth -- The number of subcategories down to go. 0 means no
      subcategories.
    continue_string -- Used for API requests. Defaults to None.
    """
    if not category.startswith("Category:"):
        category = "Category:" + category
    if category in self._processed_categories and not continue_string:
        # Skip the category if it has already been processed, except
        # if we are continuing on the same category.
        logging.debug('Skipping already processed category "{}".'.format(category))
        return
```

Reading code comments in a code editor

labs-tools-Isa/isa/campaigns/image_updater.py



```
logging.info("Updating images for campaign {}.".format(self._campaign.id))
start_time = time.time()
self._campaign.update_status = PROCESSING
if not commit_changes_to_db():
    raise UpdateImageException("Committing to database failed.")

# Clear i
# been re
Image.que
self._cam
self._cam
if not co
    raise

categorie
for categ
    if se
        d
    else:
        d
    self._fetch_images(category["name"], depth)

self._commit_images()
logging.info(
    "{} images committed for campaign {} in {} seconds "
    "and {} commits."
```

```
(method) def _fetch_images(
    category: Any,
    depth: Any,
    continue_string: Any | None = None
) -> None
```

Add images for a category to the campaign

Fetches all images in the given category and if, `depth` is more than 0, its subcategories. Filters out pages that do not match file extensions for images.

Keyword arguments: category -- Id of the campaign to fetch images for. Accepts both
 with and without "Category:" prefix.
depth -- The number of subcategories down to go. 0 means no
 subcategories.
continue_string -- Used for API requests. Defaults to None.

Visual Studio Code

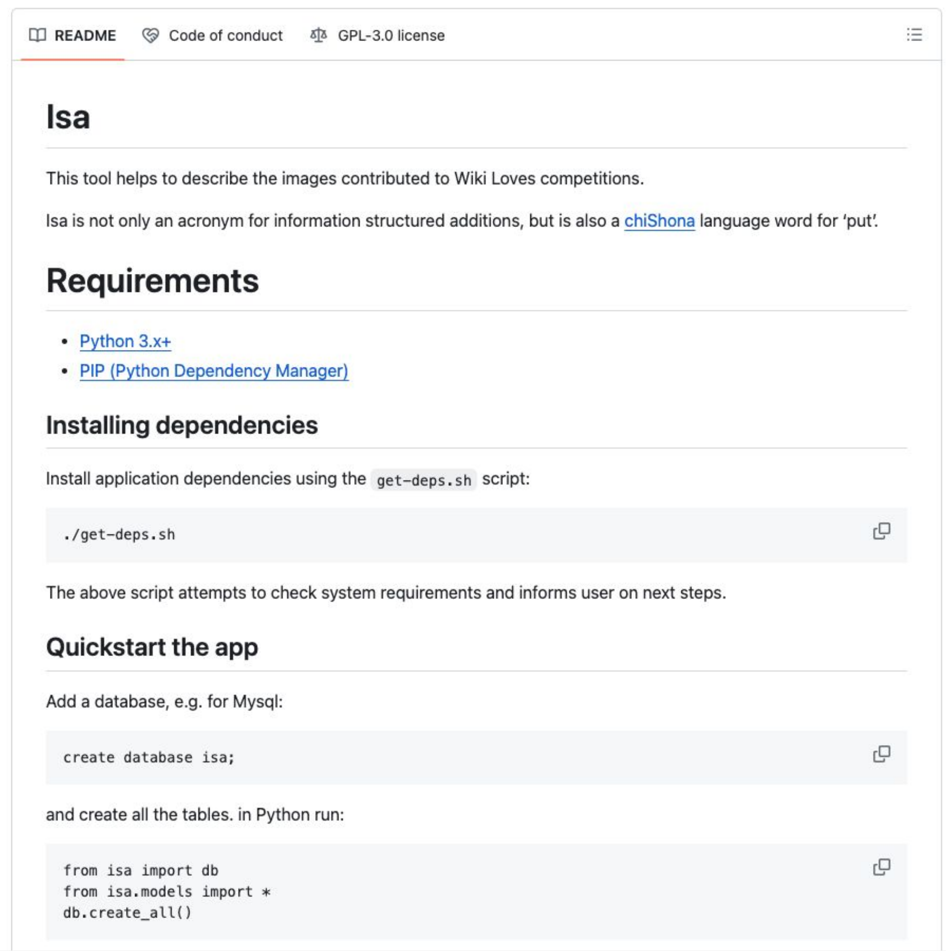
Code comments

Best practices:

- Update code comments alongside code changes
- Follow existing conventions
- Explain the why

Developer documentation

- READMEs
- How to contribute
- Often uses Markdown syntax



Isa

This tool helps to describe the images contributed to Wiki Loves competitions.

Isa is not only an acronym for information structured additions, but is also a chiShona language word for 'put'.

Requirements

- Python 3.x+
- PIP (Python Dependency Manager)

Installing dependencies

Install application dependencies using the `get-deps.sh` script:

```
./get-deps.sh
```

The above script attempts to check system requirements and informs user on next steps.

Quickstart the app

Add a database, e.g. for Mysql:

```
create database isa;
```

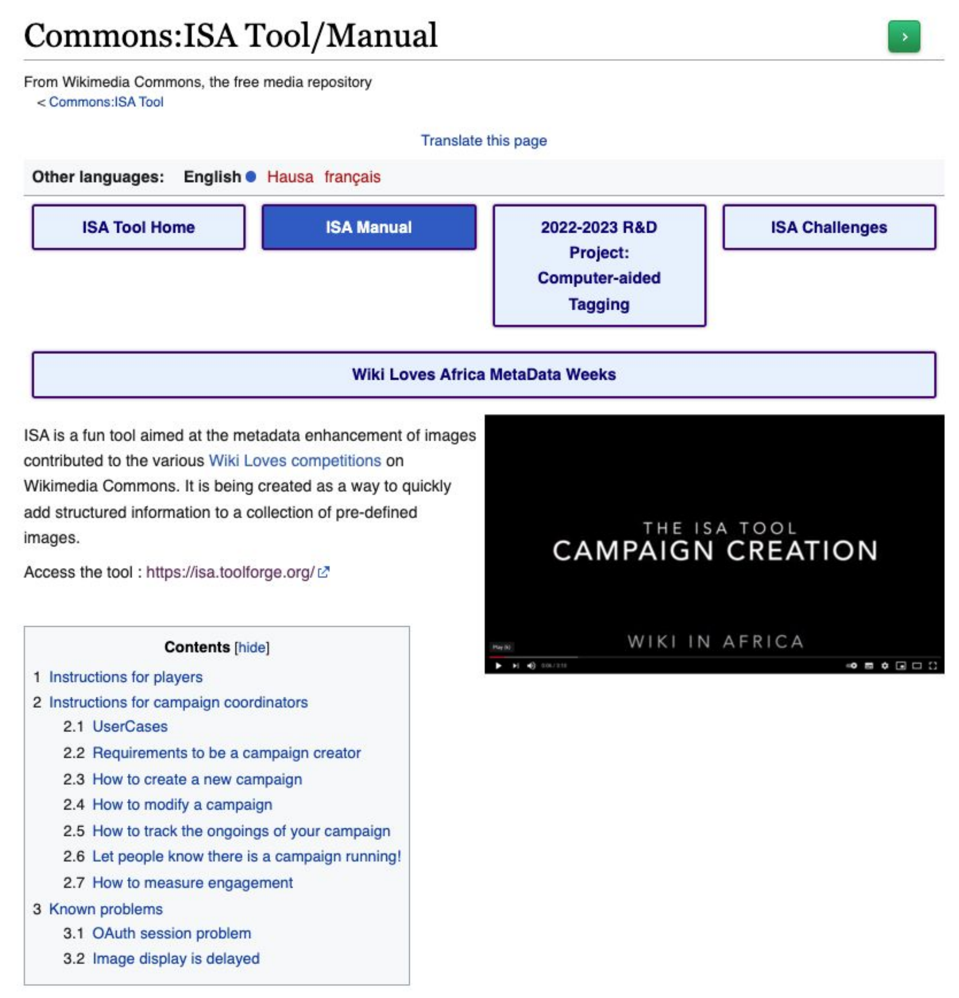
and create all the tables. in Python run:

```
from isa import db
from isa.models import *
db.create_all()
```

labs-tools-Isa/isa/README.md

User documentation

- Audience: end users
- Often serves multiple types of users
- Manuals and how-to guides
- Wiki pages



Commons:ISA Tool/Manual

From Wikimedia Commons, the free media repository
< Commons:ISA Tool

Translate this page

Other languages: English ● Hausa français

ISA Tool Home
ISA Manual
2022-2023 R&D Project: Computer-aided Tagging
ISA Challenges
Wiki Loves Africa MetaData Weeks

ISA is a fun tool aimed at the metadata enhancement of images contributed to the various Wiki Loves competitions on Wikimedia Commons. It is being created as a way to quickly add structured information to a collection of pre-defined images.

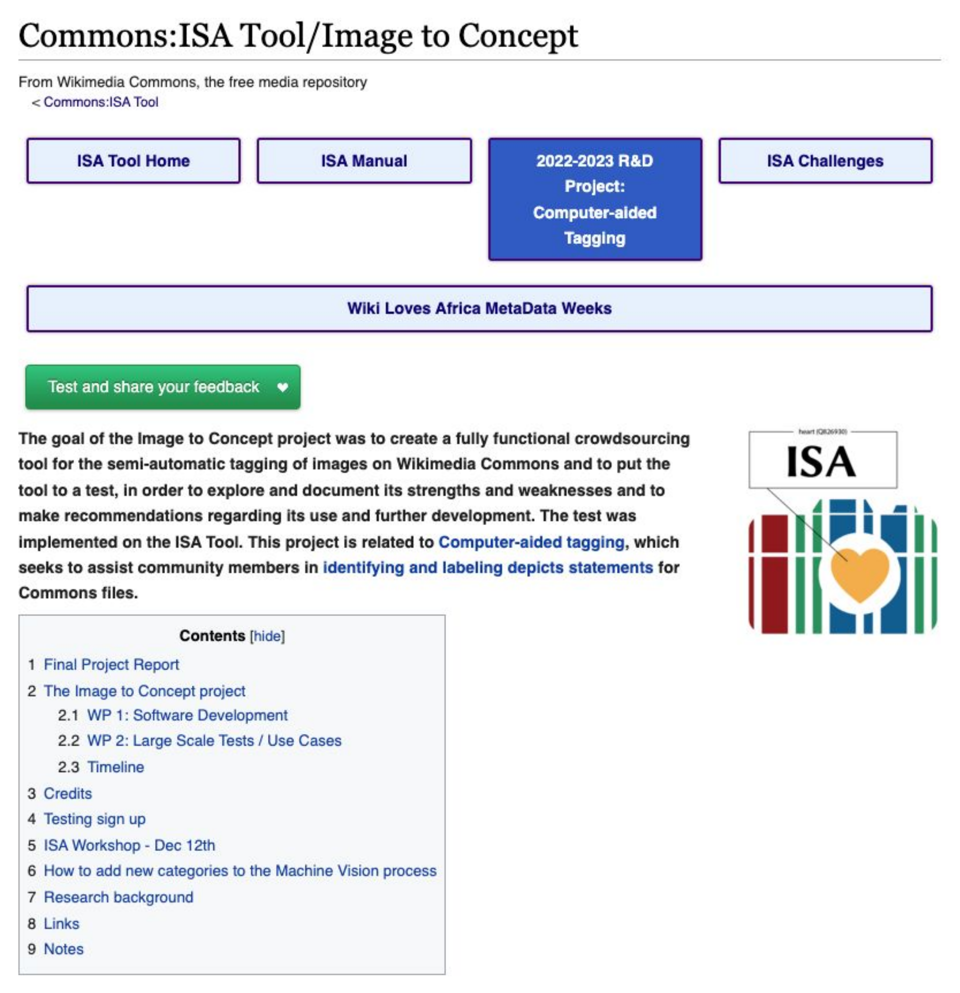
Access the tool : https://isa.toolforge.org/

Contents [hide]

https://commons.wikimedia.org/wiki/Commons:ISA_Tool/Manual

Project documentation

- Audience: contributors
- Historical
- Wiki pages



Commons:ISA Tool/Image to Concept

From Wikimedia Commons, the free media repository

< Commons:ISA Tool

ISA Tool Home
ISA Manual
2022-2023 R&D Project: Computer-aided Tagging
ISA Challenges
Wiki Loves Africa MetaData Weeks
Test and share your feedback ♥
ISA

The goal of the Image to Concept project was to create a fully functional crowdsourcing tool for the semi-automatic tagging of images on Wikimedia Commons and to put the tool to a test, in order to explore and document its strengths and weaknesses and to make recommendations regarding its use and further development. The test was implemented on the ISA Tool. This project is related to Computer-aided tagging, which seeks to assist community members in identifying and labeling depicts statements for Commons files.

Contents [hide]

https://commons.wikimedia.org/wiki/Commons:ISA_Tool/Image_to_Concept

WIKIMEDIA
FOUNDATION

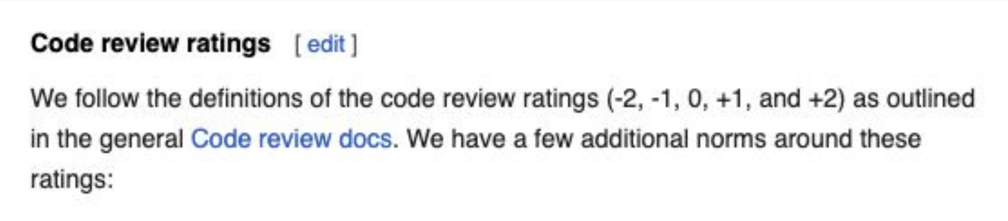
5. Organizing documentation pages

Documentation is a process.

Organizing for maintainability

- Know the audience
- Define the scope
- Avoid duplication

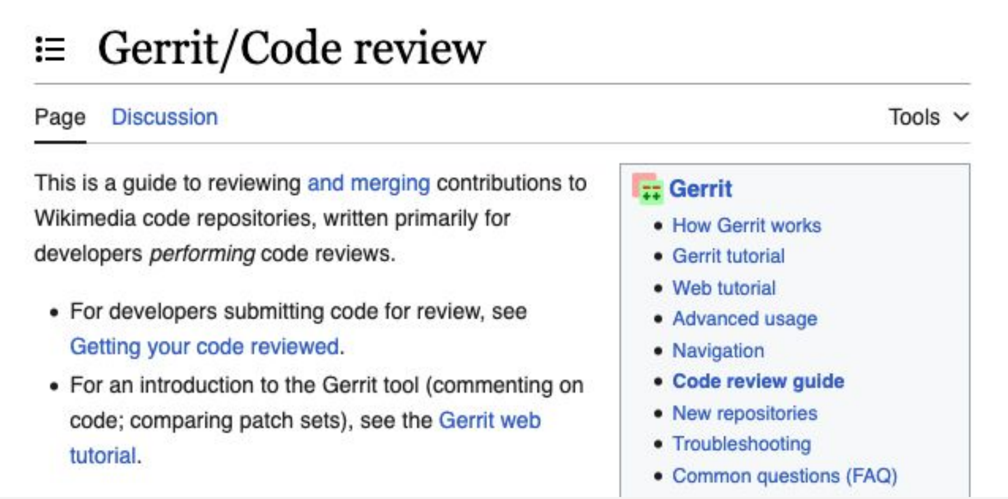
https://www.mediawiki.org/wiki/Design_Systems_Team/Team_Processes



Code review ratings [edit]

We follow the definitions of the code review ratings (-2, -1, 0, +1, and +2) as outlined in the general Code review docs. We have a few additional norms around these ratings:

https://www.mediawiki.org/wiki/Gerrit/Code_review



Gerrit/Code review

This is a guide to reviewing and merging contributions to Wikimedia code repositories, written primarily for developers *performing* code reviews.

- For developers submitting code for review, see Getting your code reviewed.
- For an introduction to the Gerrit tool (commenting on code; comparing patch sets), see the Gerrit web tutorial.

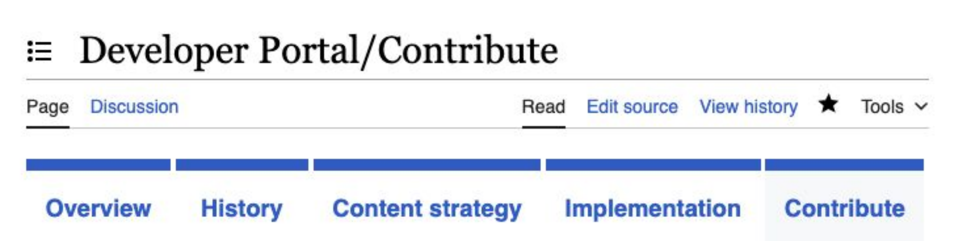
Gerrit

- How Gerrit works
- Gerrit tutorial
- Web tutorial
- Advanced usage
- Navigation
- **Code review guide**
- New repositories
- Troubleshooting
- Common questions (FAQ)

Discoverability

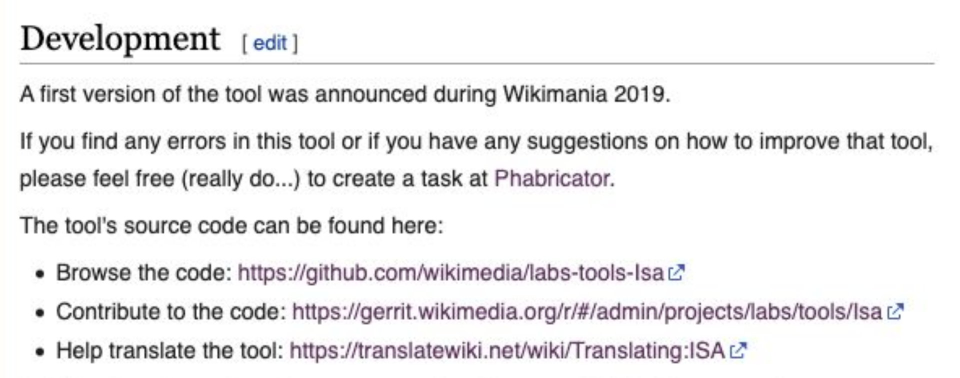
- Navigation
- Interlinking
- Next steps

https://www.mediawiki.org/wiki/Developer_Portal/Contribute



Developer Portal/Contribute

https://commons.wikimedia.org/wiki/Commons:ISA_Tool



Development [edit]

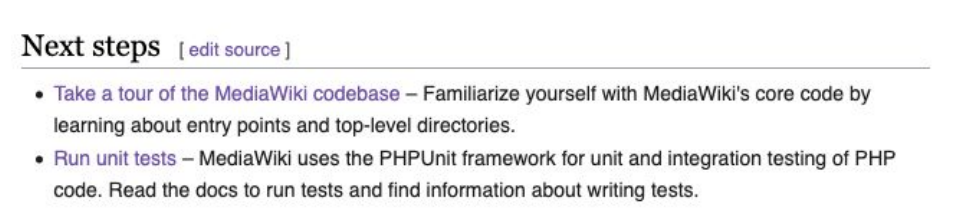
A first version of the tool was announced during Wikimania 2019.

If you find any errors in this tool or if you have any suggestions on how to improve that tool, please feel free (really do...) to create a task at Phabricator.

The tool's source code can be found here:

- Browse the code: https://github.com/wikimedia/labs-tools-Isa
- Contribute to the code: https://gerrit.wikimedia.org/r/#/admin/projects/labs/tools/Isa
- Help translate the tool: https://translatewiki.net/wiki/Translating:ISA

https://www.mediawiki.org/wiki/Local_development_quickstart



Next steps [edit source]

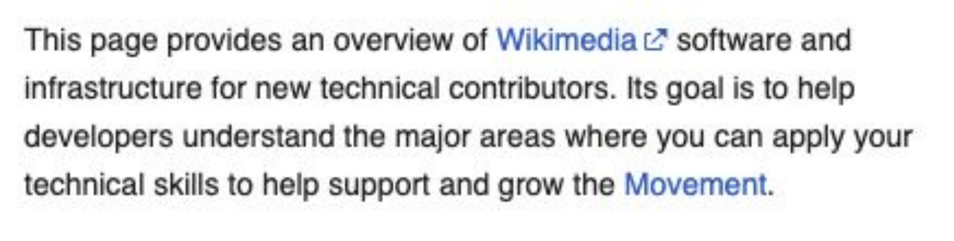
- Take a tour of the MediaWiki codebase – Familiarize yourself with MediaWiki's core code by learning about entry points and top-level directories.
- Run unit tests – MediaWiki uses the PHPUnit framework for unit and integration testing of PHP code. Read the docs to run tests and find information about writing tests.

6. Structuring the page

Introduction

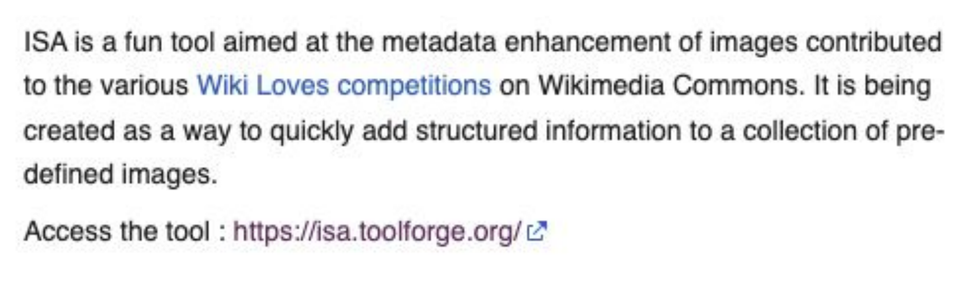
- Directly follows the title
- Explains the topic and purpose of the page
- Helps the reader decide if the page is right for them

https://www.mediawiki.org/wiki/New_Developers/Introduction_to_the_Wikimedia_Technical_Ecosystem



This page provides an overview of Wikimedia software and infrastructure for new technical contributors. Its goal is to help developers understand the major areas where you can apply your technical skills to help support and grow the Movement.

https://commons.wikimedia.org/wiki/Commons:ISA_Tool/Manual



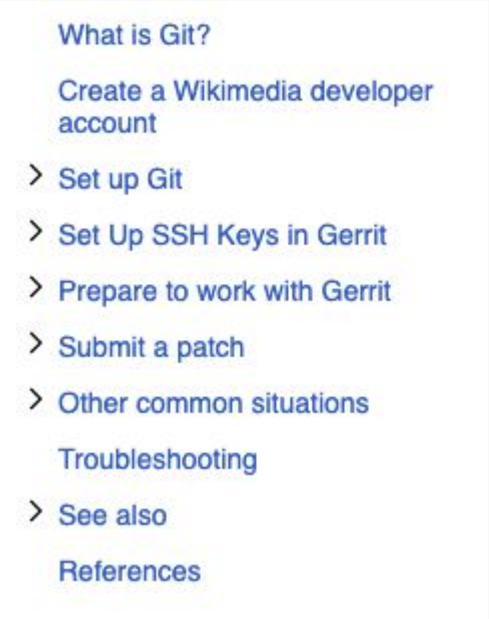
ISA is a fun tool aimed at the metadata enhancement of images contributed to the various Wiki Loves competitions on Wikimedia Commons. It is being created as a way to quickly add structured information to a collection of pre-defined images.

Access the tool : https://isa.toolforge.org/

Headings

https://www.mediawiki.org/wiki/Gerrit/Tutorial

- Focus on what users want to do
- Use short phrases that users will recognize
- Use sequential heading levels for accessibility
- Start with general or widely-applicable information and end with edge cases
- The table of contents should read like a summary of the page



What is Git?

Create a Wikimedia developer account

› Set up Git

› Set Up SSH Keys in Gerrit

› Prepare to work with Gerrit

› Submit a patch

› Other common situations

Troubleshooting

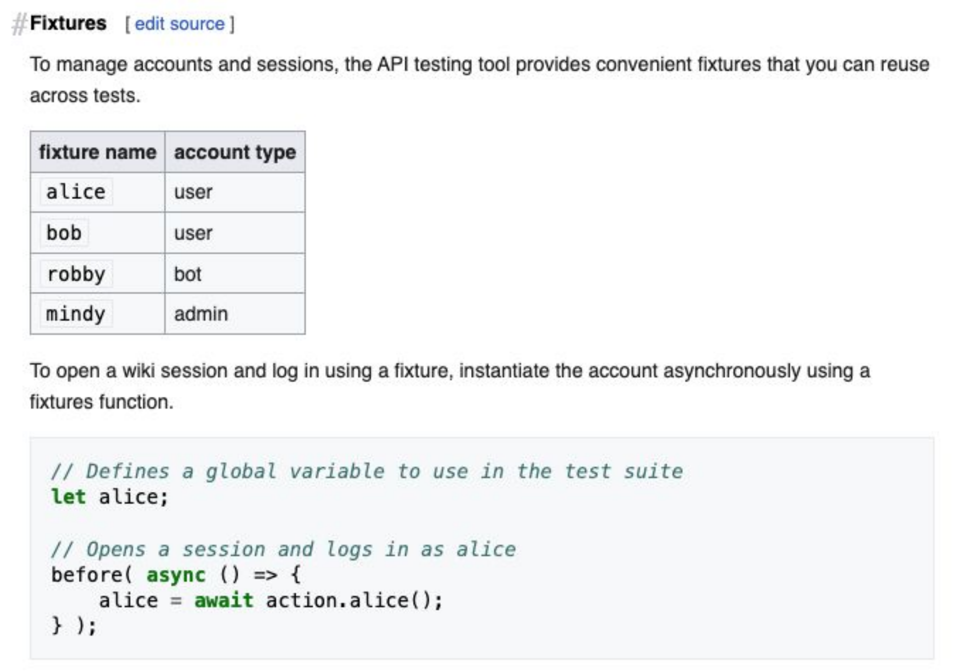
› See also

References

Make the page easy to scan

- Use meaningful headings
- Alternate paragraphs with lists, tables, and code samples
- Put key words at the beginnings of paragraphs and sentences
- Make it easy for readers to skip things that aren't relevant to them

https://www.mediawiki.org/wiki/MediaWiki_API_integration_tests



Fixtures [edit source]

To manage accounts and sessions, the API testing tool provides convenient fixtures that you can reuse across tests.

fixture name	account type
`alice`	user
`bob`	user
`robby`	bot
`mindy`	admin

To open a wiki session and log in using a fixture, instantiate the account asynchronously using a fixtures function.

```
// Defines a global variable to use in the test suite
let alice;

// Opens a session and logs in as alice
before( async () => {
    alice = await action.alice();
} );
```

Exercise

Imagine a page with this table of contents:

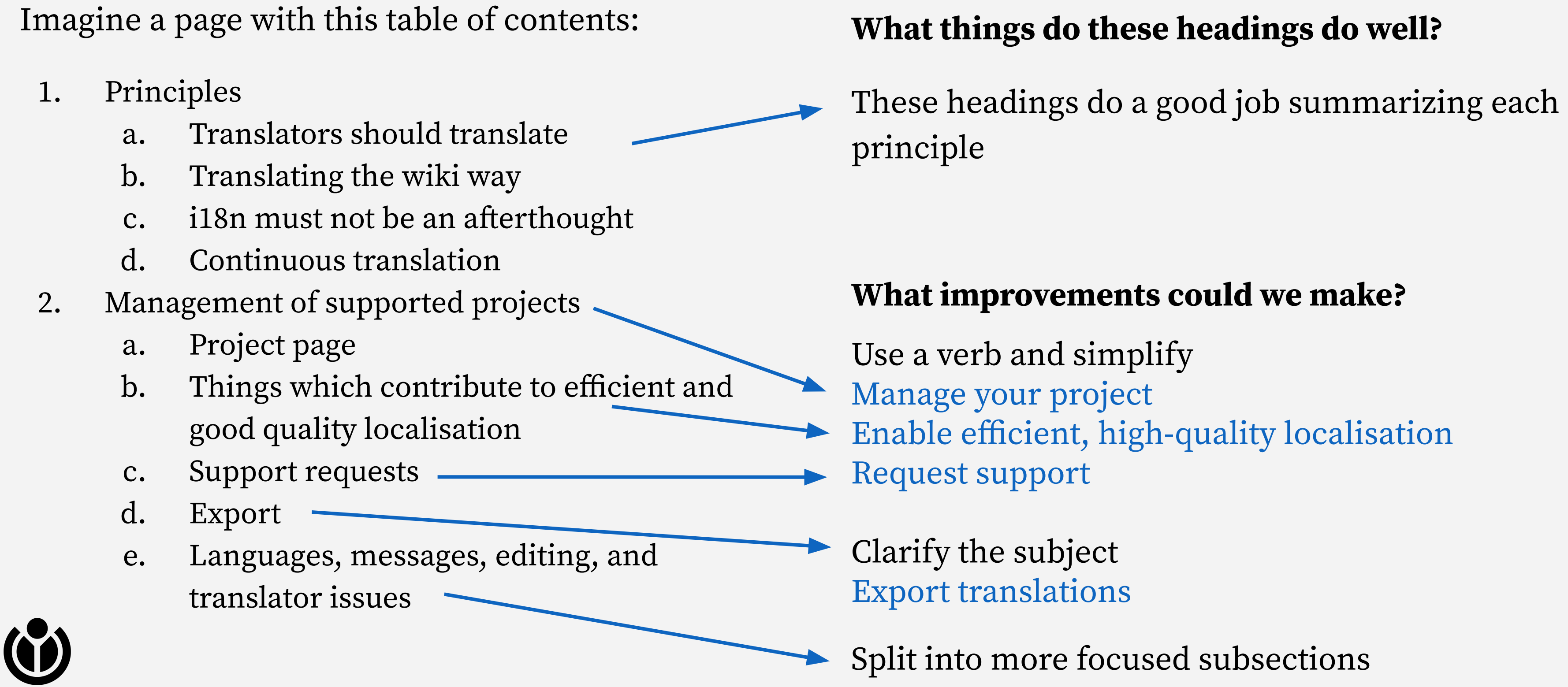
1. Principles
   a. Translators should translate
   b. Translating the wiki way
   c. i18n must not be an afterthought
   d. Continuous translation
2. Management of supported projects
   a. Project page
   b. Things which contribute to efficient and good quality localisation
   c. Support requests
   d. Export
   e. Languages, messages, editing, and translator issues

What things do these headings do well?

What improvements could we make?

From https://commons.wikimedia.org/wiki/File:Technical_documentation_best_practices_for_Data_Documentathon_2022.pdf

Exercise



Imagine a page with this table of contents:

1. Principles
   a. Translators should translate
   b. Translating the wiki way
   c. i18n must not be an afterthought
   d. Continuous translation
2. Management of supported projects
   a. Project page
   b. Things which contribute to efficient and good quality localisation
   c. Support requests
   d. Export
   e. Languages, messages, editing, and translator issues

What things do these headings do well?

These headings do a good job summarizing each principle

What improvements could we make?

Use a verb and simplify
Manage your project
Enable efficient, high-quality localisation
Request support

Clarify the subject
Export translations

Split into more focused subsections

WIKIMEDIA
FOUNDATION

7. Writing style

Understandable writing is inclusive writing.

Words

- Use simple words with obvious meanings
- Define your terms and use them consistently
- Avoid acronyms and idioms
- Avoid pronouns like “it”

Sentences

- Write short, clear sentences
- One concept per sentence
- Be clear about who is doing what

Editing

- Check for ambiguity
- Ask for review
- Use tools
    - https://www.expresso-app.org/
        - readability grade
        - rare words
        - detached subjects (who is doing what)
        - entity substitutions (pronouns)
        - extra long sentences

Exercise 1

This is a double-edged guide: to writing content with an eye to making it translatable, and to translating Wikimedia content effectively.

Exercise 1

This is a double-edged guide: to writing content with an eye to making it translatable, and to translating Wikimedia content effectively.

This guide has two purposes: to help write content that is easier to translate and to help translate Wikimedia content effectively.

Exercise 2

You do not harm Wikimedia technology infrastructure and follow the policies for that infrastructure.

Exercise 2

You do not harm Wikimedia technology infrastructure and follow the policies for that infrastructure.

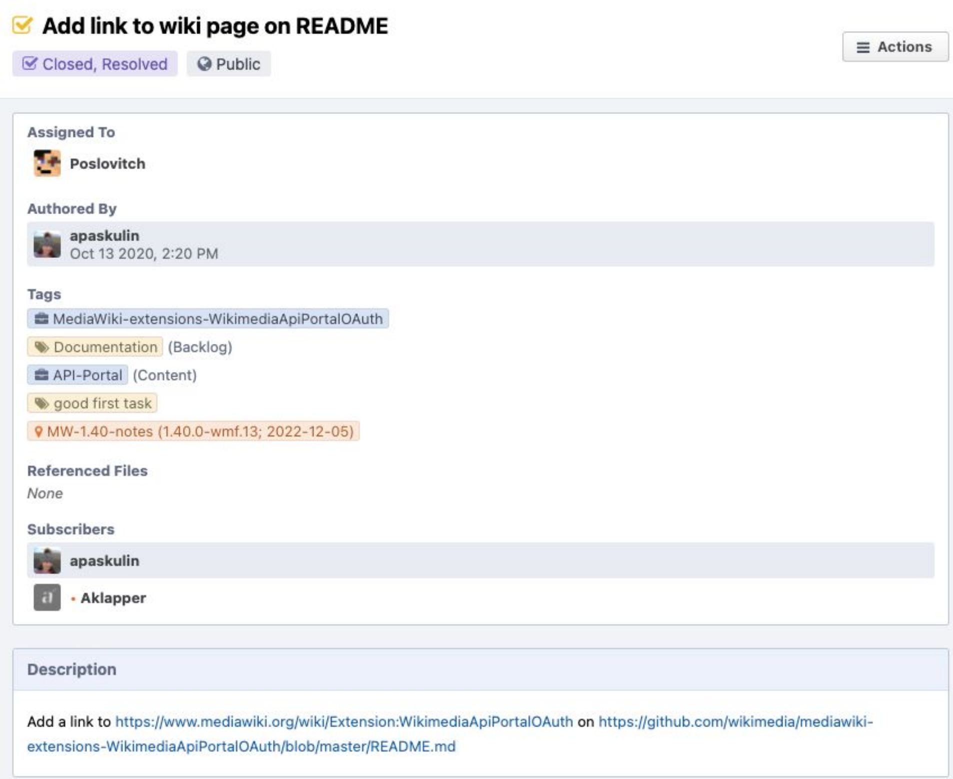
You do not harm Wikimedia technology infrastructure, and you follow the policies for that infrastructure.

WIKIMEDIA FOUNDATION

8. Working on documentation

Phabricator

- Documentation tag
- Asking for reviews
- Find documentation tasks for the Isa tool: https://phabricator.wikimedia.org/maniphest/query/CBz8solFje.e/#R



Add link to wiki page on README

Closed, Resolved · Public

Assigned To
Poslovitch

Authored By
apaskulin
Oct 13 2020, 2:20 PM

Tags
MediaWiki-extensions-WikimediaApiPortalOAuth
Documentation (Backlog)
API-Portal (Content)
good first task
MW-1.40-notes (1.40.0-wmf.13; 2022-12-05)

Referenced Files
None

Subscribers
apaskulin
• Aklapper

Description

Add a link to https://www.mediawiki.org/wiki/Extension:WikimediaApiPortalOAuth on https://github.com/wikimedia/mediawiki-extensions-WikimediaApiPortalOAuth/blob/master/README.md

Wiki pages

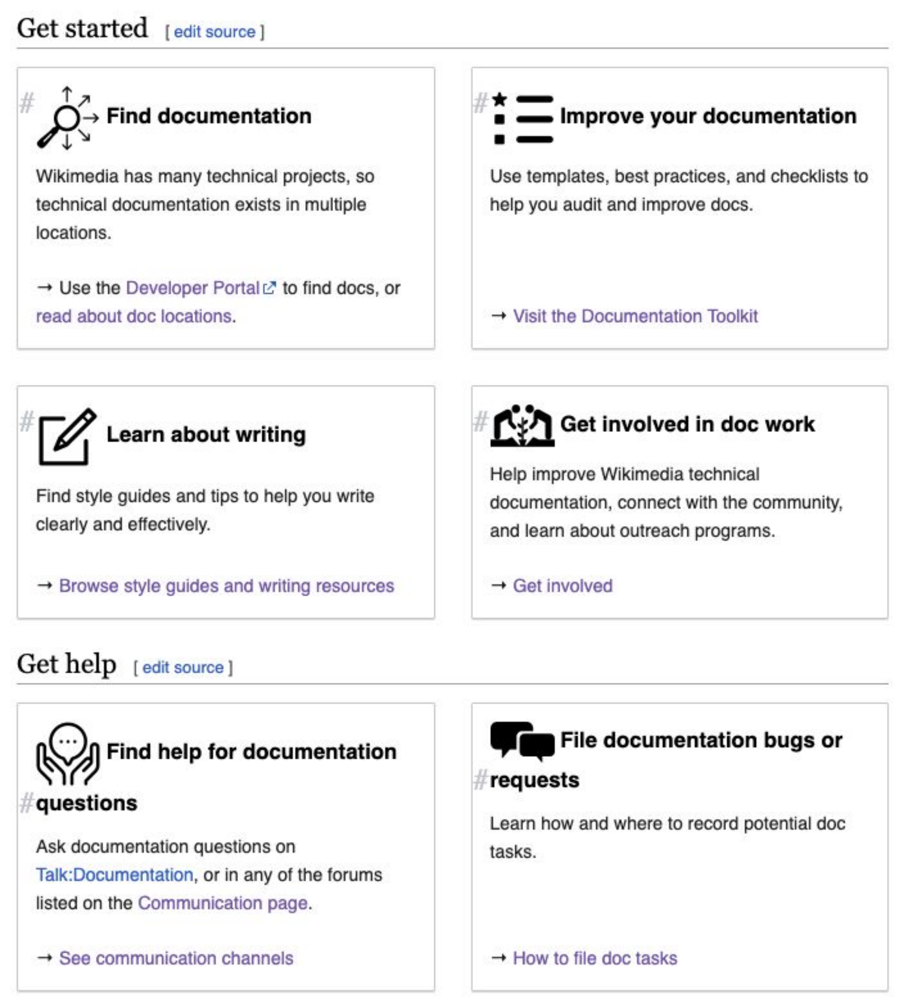
- Add a page to your watchlist to follow the latest edits
- Read comments on the talk page to find opportunities to improve the documentation

WIKIMEDIA FOUNDATION

9. Resources and getting help

Resources

- Templates and guides: https://www.mediawiki.org/wiki/Documentation
- Documentation tasks on Phabricator: https://phabricator.wikimedia.org/tag/documentation/
- Documentation style guide: https://www.mediawiki.org/wiki/Documentation/Style_guide



Get started [edit source]

Find documentation

Wikimedia has many technical projects, so technical documentation exists in multiple locations.

→ Use the Developer Portal to find docs, or read about doc locations.

Improve your documentation

Use templates, best practices, and checklists to help you audit and improve docs.

→ Visit the Documentation Toolkit

Learn about writing

Find style guides and tips to help you write clearly and effectively.

→ Browse style guides and writing resources

Get involved in doc work

Help improve Wikimedia technical documentation, connect with the community, and learn about outreach programs.

→ Get involved

Get help [edit source]

Find help for documentation questions

Ask documentation questions on Talk:Documentation, or in any of the forums listed on the Communication page.

→ See communication channels

File documentation bugs or requests

Learn how and where to record potential doc tasks.

→ How to file doc tasks

https://www.mediawiki.org/wiki/Documentation

Getting help

- Post your questions and review requests to https://www.mediawiki.org/wiki/Talk:Documentation
- Email the Wikimedia Foundation Technical Documentation team at techdocs@wikimedia.org

WIKIMEDIA
FOUNDATION

Thank you!

WMA
WIKI MENTOR AFRICA