P02: Create documentation for an ONAP release (ONAP project teams)

TITLE	**P02: Create documentation for an ONAP release (ONAP project teams)**
SUBJECT	Procedure #02 describes the required tasks which must be performed by ONAP project teams to create their release-specific documentation for an ONAP main and maintenance release.
TARGET AUDIENCE	ONAP project teams: Maintainers of project repository
RELEASE RELEVANCE	R12 'London' - R11 'Kohn'
TABLE OF CONTENTS	
AUTHOR	Thomas Kulik
REVIEWER	Toine Siebelink Luke Gleeson
NOTES	This procedure is focussing on documentation even if it largely describes the release process of an ONAP project. But there is no guarantee on completeness for the full ONAP project release process. Not every step within a single task is listed. For example, common tasks like 'git' operations (e.g. add, remove, commit, review) are not included. Jira or Gerrit activities (e.g. review, submit, cherry-pick) related to file changes are also not listed. A development environment to write and preview reStructuredText is highly recommended. The following tasks may have a different relevance depending on the type of release (main, maintenance) you are working on. Optional tasks for a maintenance release marked as 'maintenance (opt)'. They are expected to be performed only if you need to fix issues or you have to incorporate smaller changes. The placeholder '{release}' needs to be substituted by the name of the new release, e.g. 'istanbul' or 'jakarta'. ⚠️ **With introduction of the ONAP 'jakarta' release a project/repository containing formal ONAP documentation (rst format) needs to create a release branch in any case. Otherwise the project/repository can not be included to the release-specific version of documentation in ReadTheDocs.**

EXAMPLE CONFIGS	Together with the 'doc' repository the documentation team provides basic 'tox', 'sphinx' and ReadTheDocs example configuration files to be used by other ONAP projects (except 'doc'). Every project containing documentation in ReStructuredText format needs them. The files are:
	- conf.py - requirements-docs.txt - tox.ini .readthedocs.yaml - ribbon.css
	⚠️ Always check that the configuration files used in your repository are in sync with the provided examples. The example configs of course can be extended to fulfill specific needs of other ONAP projects. Please note README.md for the details.
	EXAMPLES ONLY: 'examples' directory in the 'doc' repository (master):
	📄 Unknown macro: 'list-git-folder'
	FULL: Full file and folder structure of the 'doc' repository (master) to show where active config files are generally stored in a repository.
	⚠️ **The *content* of active configuration files in the 'doc' repository may differ from the *content* of config files for other ONAP projects!**
	📄 Unknown macro: 'list-git-folder'
1.0	# MASTER BRANCH PREPARATION
1.1	## Create a new Jira task and use the ID for all upcoming changes
	RELEASE: main + maintenance
	BRANCH: not applicable
	FILE: n/a
	DESCR: Of course it is up to your project team *how* to manage the assignment of Issue-IDs. But generally you need to add one to any gerrit change. Please follow the instructions below or the agreed process in your project team to create the Issue-ID.
	1. Open https://https://jira.onap.org/ 2. Create a new task for your project and name it e.g. "Create docs for '{release}' {main|maintenance} release". Assign the release '{release}' to it. The ID is required for submitting all related changes in gerrit. The format to be used in gerrit is 'Issue-ID: {YOUR-PROJ}-nnn'.
1.2	## Clone 'master' branch of your repository to your local development environment
	RELEASE: main + maintenance
	BRANCH: 'master'
	FILE: all
	DESCR: If not already done earlier (depending on common practice in your project team), please use the command below to clone the 'master' branch.
	```git clone -b master ssh://<username>@gerrit.onap.org:29418/{repository}```
	Now you can start to modify files in the 'master' branch.
2.0	# MASTER BRANCH UPDATES

2.1	## Ensure correct 'defaultbranch' in '.gitreview'

RELEASE: main

BRANCH: 'master'

FILE: {repository}/.gitreview

DESCR: Check that the 'defaultbranch' entry is pointing to 'master' ('defaultbranch=master'). This can help to avoid that changes are accidentally committed or even published to a wrong branch. Add/update the 'project' entry to show the name of your project.

EXAMPLE: active '.gitreview' for 'master' branch, 'doc' project

Unknown macro: 'view-git-file' |
| **2.2** | ## Update Sphinx configuration file 'conf.py'

RELEASE: main + maintenance (opt)

BRANCH: 'master'

FILE: {repository}/docs/conf.py

DESCR: The Sphinx build configuration file 'conf.py' contains configuration needed to customize Sphinx input and output behavior. Please follow the instructions below.

1. Set 'release' and 'version' to 'master'.
2. Set 'branch' to 'latest'.
3. Set 'intersphinx_mapping' for projects/repositories you are referencing in your documentation files (see 'intershinx_mapping' entries in the examples below)
4. Remove 'intersphinx_mapping' for projects/repositories you are no longer referencing in your documentation files.

Also compare the version used in your repository with the example provided in the 'doc' repo and update if required. Be careful if your active configuration file was already extended.

EXAMPLE #1: example 'conf.py' for 'master' branch

Unknown macro: 'view-git-file'

EXAMPLE #2: active 'conf.py' for 'master' branch, 'doc' project

Unknown macro: 'view-git-file' |
| **2.3** | ## Update Sphinx configuration file 'requirements-docs.txt'

RELEASE: main

BRANCH: 'master'

FILE: {repository}/docs/etc/requirements-docs.txt   or   {repository}/docs/requirements-docs.txt

DESCR: The file contains the required libraries to be used by Sphinx. Compare the version used in your repository with the example provided in the 'doc' repo and update if required. Be careful if your active configuration file was already extended.

EXAMPLE: example 'requirements-docs.txt' for 'master' branch

Unknown macro: 'view-git-file'

⚠ **Depending on your setup the "etc" directory exists or not.** |

2.4	**Update Tox configuration file 'tox.ini'**
	RELEASE: main
	BRANCH: 'master'
	FILE: {repository}/tox.ini
	DESCR: The file is required to customize different tox environments. It is stored in the root folder ⚠️ of your project. Compare the version used in your repository with the example provided in the 'doc' repo and update if required. Be careful if your active configuration file was already extended.
	1. Set correct branch (**master**) for all ⚠️ lines containing:
	-chttps://git.onap.org/doc/plain/etc/upper-constraints.onap.txt?h=**master**
	EXAMPLE: example 'tox.ini' for 'master' branch
	📃 Unknown macro: 'view-git-file'
	OPEN: Update 'upper-constraints' entries here after the final version of the file is available in the 'doc' repo
2.5	**Update ReadTheDocs build configuration file '.readthedocs.yaml'**
	RELEASE: main + maintenance (opt)
	BRANCH: 'master'
	FILE: {repository}/.readthedocs.yaml
	DESCR: The ReadTheDocs build configuration file '.readthedocs.yaml' contains configuration needed to customize ReadTheDocs input and output behavior. It is stored in the root folder ⚠️ of your project. Please follow the instructions below.
	1. Compare the version of the file currently used in your repository with the (basic) version provided in the examples
	2. If required, change the file in your repository
	Also compare the version used in your repository with the example provided in the 'doc' repo and update if required. Be careful if your active configuration file was already extended.
	EXAMPLE: example '.readthedocs.yaml', same for 'master' branch and '{release}'
	📃 Unknown macro: 'view-git-file'
2.6	**Update ReadTheDocs configuration file 'ribbon.css'**
	RELEASE: main + maintenance (opt)
	BRANCH: 'master'
	FILE: {repository}/docs/_static/css/ribbon.css
	DESCR: The ReadTheDocs configuration file 'ribbon.css' has - among others - an effect on the width of a RTD documentation page. The width has been limited to ensure good readability. Please follow the instructions below.
	1. Compare the version of the file currently used in your repository with the (basic) version provided in the examples
	2. If required, change the file in your repository
	Also compare the version used in your repository with the example provided in the 'doc' repo and update if required. Be careful if your active configuration file was already extended.
	EXAMPLE #1: example 'ribbon.css', same for 'master' branch and '{release}'
	📃 Unknown macro: 'view-git-file'
	EXAMPLE #2: active 'ribbon.css', same for 'master' branch and '{release}'
	📃 Unknown macro: 'view-git-file'

2.7	**Update your project release notes**
	RELEASE: main + maintenance (opt)
	BRANCH: 'master'
	FILE: e.g. {repository}/docs/release-notes.rst
	DESCR: The file contains a description of the latest changes in your project/repo. Update it accordingly for the new release. If there was no change in your project, please add a line about this fact as well.
	EXAMPLE: n/a
2.8	**Update your documentation files**
	RELEASE: main + maintenance (opt)
	BRANCH: 'master'
	FILE: e.g. {repository}/docs/...
	DESCR: Update the files accordingly for the new release. ℹ️ Advice: Updating ReadTheDocs documents as needed while delivering user stories might be a good approach to prevent a big batch update at the end of each release.
	EXAMPLE: n/a
3.0	**RELEASE BRANCH PREPARATION**
3.1	**Create new branch of your repository using gerrit**
	RELEASE: main
	BRANCH: source/initial='master', target/new='{release}'
	FILE: n/a
	DESCR: Please follow the instructions below.
	1. Open https://gerrit.onap.org/r/admin/repos, select your repository and list available branches 2. Use [CREATE NEW] function on the top right corner 3. Set 'Branch name' to the name of the upcoming release (e.g. 'istanbul') 4. Set 'Initial revision' to 'master' 5. Use [CREATE] to create the new branch
	Note: You are not allowed to delete a branch. If you need to (maybe due to a typo in the process of creation) you have to raise a ticket at the LF IT Support.
	⚠️ **IMPORTANT: To trigger the documentation build in ReadTheDocs, you must change at least one of the ReStructuredText files (.rst) in the new branch. Only branching the repository does not trigger the RTD doc build process. Consequently the '{release}' version of documentation is missing in RTD.**
3.2	**Clone '{release}' branch of your repository to your local development environment**
	RELEASE: main + maintenance
	BRANCH: {release}
	FILE: all
	DESCR: Please use the command below.
	`git clone --branch <newbranch> ssh://<username>@gerrit.onap.org:29418/{repository}`
	Now you can start to modify files in the new branch.
4.0	**RELEASE BRANCH UPDATES**

4.1	**Set correct 'defaultbranch' in '.gitreview'**
	RELEASE: main
	BRANCH: {release}
	FILE: {repository}/.gitreview
	DESCR: Add/update the 'defaultbranch' entry ('defaultbranch={release}'). This can help to avoid that changes are accidentally committed or even published to a wrong branch. Add/update the 'project' entry to show the name of your project.
	EXAMPLE: active '.gitreview' for 'jakarta' branch, 'doc' project
	🗋 Unknown macro: 'view-git-file'
4.2	**Update Sphinx build configuration file 'conf.py'**
	RELEASE: main + maintenance (opt)
	BRANCH: {release}
	FILE: {repository}/docs/conf.py
	DESCR: The Sphinx build configuration file 'conf.py' contains configuration needed to customize Sphinx input and output behavior. Please follow the instructions below.
	1. Set 'release' and 'version' to '{release}'. 2. Set 'branch' to '{release}'.
	EXAMPLE: example 'conf.py' for '{release}' branch
	🗋 Unknown macro: 'view-git-file'
4.3	**Update Tox configuration file 'tox.ini'**
	RELEASE: main
	BRANCH: {release}
	FILE: {repository}/tox.ini
	DESCR: The file is required to customize different tox environments. Compare the version used in your repository with the example provided in the 'doc' repo and update if required. Be careful if your active configuration file was already extended.
	1. Set correct branch **{release}** for all ⚠ lines containing:
	-chttps://git.onap.org/doc/plain/etc/upper-constraints.onap.txt?h=**{release}**
	EXAMPLE #1: example 'tox.ini' for 'release' branch
	🗋 Unknown macro: 'view-git-file'
4.4	**Update at least one .rst file to trigger the RTD build**
	RELEASE: main
	BRANCH: {release}
	FILE: n/a
	DESCR: To trigger the documentation build in ReadTheDocs, you must change at least one of the ReStructuredText files (.rst) in the new branch. Only branching the repository does not trigger the RTD doc build process. Consequently the '{release}' version of documentation is missing in RTD.
	EXAMPLE: n/a
5.0	**IMPROVEMENTS FOR THIS PROCEDURE**

5.1	## To be discussed
	- **update release date in project/repo release note?**
	- **add corresponding milestone to a task (if applicable)?**
	## To be done
	- **Integrate changes due to the new repo doc/doc-best-practice**