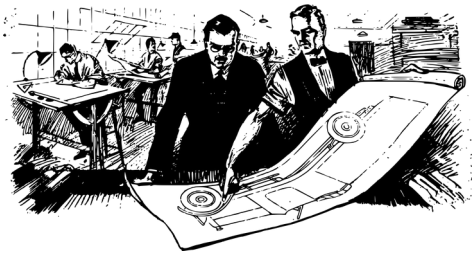
Sphinx-Needs

A Sphinx Extension



Overview

Sphinx-Needs allows the definition, representation and filtering of class-like objects in documentations based on Sphinx. Its main use case is the traceability handling of requirements, specifications and test cases in software development projects.

Maintainer: Daniel Woste (useblocks GmbH)
Documentation: https://sphinxcontrib-needs.readthedocs.io
Source code: https://github.com/useblocks/sphinxcontrib-needs (License: MIT)

User base & numbers (date 17.01.2020)

Sphinx-Needs target user base are large development teams in process-driven companies. By nature their projects are closed-source and not reachable for further analysis. So the following numbers may not cover the complete situation.

Git & Github numbers

Created: **30.11.2016**	Commits: **278**
Contributors: **4** (3 not from useblocks)	Pull Requests: **0/13** (open/closed)
Issues: **17 / 83** (open /closed)	github Stars: **25**

Project usage

Open source projects: **16**	Closed source projects: **~10** (pers. known)
Open source customers: **6**	Closed source customers: **~3**
Biggest open source project: **ONAP** Source: https://github.com/onap	Biggest closed source project: **Automotive OEM** (Germany)
Use case: Cross-company project documentations (>1.000 needs)	Use case: ISO 26262 compliant documentations for common tool-development process and CI tools.

pypi – Python Package Index

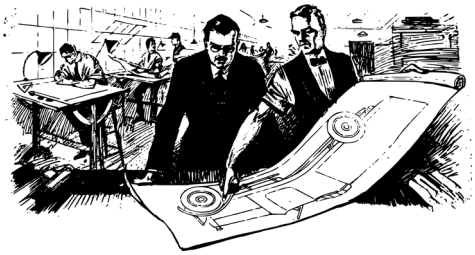
Releases overall: **~60**	Downloads per day: **~39**
Last release: **0.4.3** (5.12.2019)	Downloads per week: **~261**
Last release: URL	Downloads per month: **~821**
Download stats: https://pypistats.org/packages/sphinxcontrib-needs	

Competitors

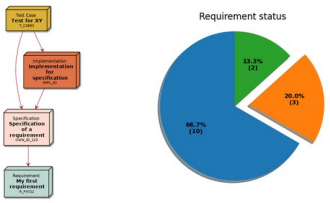
Name	**Last release**	**Downloads** (d,m,y)	**Github dependents**
mlx.traceability	28.12.1019	7, 32, 498	0
sphinxcontrib-requirements	2011	5, 11, 78	n/a (on Bitbucket)

Sphinx-Needs

A Sphinx Extension



Features (version >= 0.5.0)

Create needs Define needs everywhere in your documentation project. No special page structure or order is needed.	Requirement: **What is a need** REQ_1 tags: introduction links incoming: SPEC_1 A need is a generic object, which can become everything you want for your sphinx documentation: A requirement, a test case, a user story, a bug, an employee, a product or anything else. But whatever you chose it shall be and how many of them you need, each need is handled the same way.
Filter needs Use tables and powerful filter-strings to get the insight you need. Tables can also be searched, exported and sorted on the fly.	Show 10 entries Columns Copy Excel PDF Search: ID | Title | Status | Outgoing EX_ROW_1 | Implemented spec | implemented | EX_ROW_2 | Not implemented spec | open | EX_ROW_3 | Spec under progress | in progress | Showing 1 to 3 of 3 entries Previous 1 Next
Analyze needs Take a look on the traceability of needs or compare their field values. Free definable flow charts and pie charts provide expert insights into your data.	
Customize needs Define own need types, links or fields to fit Sphinx-Needs to your data. Also field-value dependent layouts and colors are supported (e.g. show failed tests with red border)	FEATURE_4 **Customizing everything** Feature tags: introduction links outgoing: SPEC_1 `Sphinx-needs` allows to customize needs-types, needs-options, colors, layouts, ids, checks, ... The pages Configuration and Layouts & Styles are full of possibilities to adopt `Sphinx-needs` for your own processes and workflows. layout: complete style: yellow

Ex/Import needs
Want to reuse the data from your sphinx documentation? Get all your data as a single json-file, including all needs and selected table results.
Or provide custom data as json, to import it into your documentation.
	{
	"created": "2017-07-03T11:54:42.433876",
	"current_version": "1.5",
	"project": "needs test docs",
	"versions": {
	"1.0": {
	"created": "2017-07-03T11:54:42.433868",
	"filters": {
	"FILTER_1": {
	"amount": 1,
	"export_id": "FILTER_1",
	"filter": "",
	"result": [
	"IMPL_01",
	],
	"status": [],
	"tags": "",
	"types": []
	},
	"needs": {
	"IMPL_01": {
	"description": "Incoming links of this spec: :need
	"id": "IMPL_01",
	"links": [

Automate needs
Sphinx-needs provides mechanisms to calculate field data based on values from other needs (e.g. set status on done, if all linked test cases have passed).
It also can warn and stop build, if customized constraints are not fulfilled.
	`Checking sphinx-needs warnings`
	`asset_life_cycle_exist: failed`
	`failed needs: 1 (ASSET_HW__tux01)`
	`used filter: type == 'asset' and not life_cycle`
	`asset_life_cycle_enum: passed`
	`failed`

Extend needs Sphinx-Needs provides an open API to let other sphinx extensions easily create needs for their custom use case. For instance Sphinx-Test-Reports, which collects test cases and their results and presents them as need.