Syndesis Developer Handbook

Version , 2017-12-19

Syndesis Developer Handbook

Chapter 1. Introduction

This handbook contain all information required for installing, running and developing Syndesis from a developer’s point of view.

Chapter 2. Building Syndesis

2.1. One tool to rule them all

Chapter 3. Syndesis Development Process

3.1. Issue and Pull-Request Labels

We use GitHub labels in order to categorize epics, issues and tasks. They are the foundation of our process, so please use labels for issues.

> Label are living entities. This document describes the current status and might be slightly outdated. Please send a PR to adapt this document if the label structure changes. Also feel free to discuss the label structure anytime. It’s essential that labels describe our process, not that we have to adapt our process for these labels.

Labels are grouped. Each label consists of two part: A **Group** and a **Name** which are separated by a slash (`/`). For example the label `module/ui` is used to mark issue which are relevant to the Syndesis UI module.

The following label groups are available. There must be only at most one label from the "Exclusive" groups.

Group	Description	Excl.
cat/	Misc categories which are can be added freely	
prio/	Priority of the issue. Only one `prio/` label must be added per issue. `prio/p0` is of highest priority, `prio/p4` the lowest one.	
ext/	Reference to external projects	
d/	Day labels for tagging an Epic to talk about at specific day in the daily meeting	
module/	Internal Syndesis modules	
notif/	Notification label which can be added and removed to ping certain subteams	
size/	Tee shirt size for issues. This is a subjective assesment and should be done relative to other issues.	
status/	Status of an issue or PR.	
target/	Target milestone for an Epic. These are the Technicla Preview (TP) targets and GA.	

Each label group serves a certain purpose and for each issue and PR it should be considered whether a label from a group applies.

3.1.1. Modules

Labels from this group reference our application components like "rest", "ui", "uxd" or "connectors". Each sub team is responsible for one or more modules, and every module has a 'owning' team. That does not mean that members of other teams are allowed to work on such modules. Contrary, this is even encouraged. But its just there so that teams can filter on issues and PRs which are relevant to them.

An issue can carry many module labels. Especially Epics will carry more than such label as they touches more than one module (otherwise it wouldn’t be an epic).

3.1.2. Categories

Labels from the `cat/` group are labels which can be always applied and which does not fit in other category. Currently we have these categories:

Category	Description
cat/bug	A bug which needs fixing.
cat/blocker	A blocker, which is a bug which needs to be fixed as soon as possible.
cat/enhancement	PR label for an enhancement of an existing feature.
cat/feature	PR label for a new feature
cat/discussion	This issues requires a discussion.
cat/question	For issues holding a question.
cat/build	For issues which has relevance for the build system.
cat/design	A concrete UX design. Use this for PRs containing UX designs.
cat/process	Development process related issues carry this label.
cat/research	Label used for issues which describe some research work
cat/retro	Label for action items which are the result of a retrospective.
cat/starter	An issue which is easy to solve and can be used for ramping up new developers.
cat/techdebt	Label for issues identifying technical debt.
cat/techdoc	Technical deceloper information (likes this hanbook ;-) related issues.
cat/tooling	Issues used for tooling around the development process (note: could be merged with "build")

3.1.3. Notification

Notification labels from the `notif/` group serve a special purpose. They are used when one team wants to notify another team that a certain issue might have them relevance to them.

Currently we have two notifcation labels:

Notification	Description
notif/doc	This issue needs some attention from the docs team. This might because a new feature has been introduced or, more important, an existing feature has changed for which a documentation already exists.
notif/uxd	This label should be used for issues which needs some attention from the UX team. This might because a new feature has been introduced or, more important, an existing feature has changed for which a a UX design already exists.

It is important to note that these labels also be removed when the notification has been received.

For example, when an UI feature like a input form changes. Then the UI team attaches a `notif/uxd` label to the PR which introduces this change. The UX team, detects with a filter search on this label, that there is a new notification. It then decides, whether UX design needs to be updated or not. In any case they are removing the `notif/uxd` label and add a `module/uxd` label if this PR indeed requires an UX design update. If no update is required, then the label is removed without replacement.

3.1.4. External references

This label group should be used if an external system is referenced, which is not part of the Syndesis mono repo.

External Project	Description
ext/atlasmap	atlasmap data mapper
ext/camel	Camel
ext/qe	syndesis-qe suite
ext/docs	syndesis-documentation End user documentation

For the future we plan to add more of these external repos into the Syndesis mono repo (like documentation or QE). If this happens, then labels should be converted to `module/` kind of labels.

Daily Meeting Labels

This category holds five labels: `d/mon`, `d/tue`, `d/wed`, `d/thu`, `d/fri`, one for each working day. They are used to mark an Epic so that it is talked about the daily meeting at that day. The reason for this selection is, that we want to keep the daily meetings still at 15 mins but dont yet want to split up. More than one of such label can and actually should be added to one Epic. At least two day labels must be added to an epic.

Status

Status labels are special since they may trigger some automatic actions.

The current status labels are:

Status	Description
status/blocked	The current issue is blocked by another issuer. Refer to the issue itself to see what is blocking this issued. This label is purely informal.
Approved	This label will be automatically applied to an an PR as soon as the PR has been approved in a review. It is an indicator for our PR bot to automatically merge the pull request if it passes all required tests. (Note: Should probably be renamed to `status/approved`)
status/wip	This is a PR request label which should be used for "Work-in-Progress" kind of PRs which has been submitted for early review. If this label is present on a PR, the PR is not merged, even when it is "Approved"

Status	Description
status/2s2f	Use this label to mark issues which should be self-merged without requiring a PR review, because its "too small too fail". Be very careful with this label, and remember a review is a service to you to help in your code quality. It alone your responsibility when you chose this label. It’s useful for minor doc updates or one line where you are 100% sure that it doesn’t break the system. Please use it sparingly and responsibly. (*Need still to be implemented*)