Transtats Documentation

Release 0.8.8

Sundeep Anand

Mar 24, 2023

Table of Contents

transtats.org

In a software release cycle, some of the necessary localization steps lack attention which affect translation quality and delivery. These steps (*for a package*) are extracting or updating language resource, pushing that to translation platform, pulling and packaging translations, quality checks etc. Transtats helps to tie up loose ends and make packages ready to ship with translation completeness. Furthermore, its an attempt to bring automation in `i18n l10n` space through jobs.

Transtats Server is a simple django application with PostgreSQL backend, which analyses and processes the translation data for meaningful representations. It has a CLI and some ansible playbooks for deployments. And, can be deployed on shared, dedicated or container based environments.

The `Jobs framework` is a centralized processor through which we can solve multiple sets of problems in a flexible way. Because, it has information about Product Release (*and its Schedule*), Package Source Repo, Package Translation Platform and Package Build System. Multiple sets of problems can be captured and executed in the form of Job Templates. Moreover, as the jobs are YAML based, they're flexible.

Transtats can..

- **Tell us how we're doing on delivering translations for product releases.**
  - From a language and a package perspective.
- Transport translations from one place to another.
- Help spot differences: source repository vs translation platform vs build system.

CHAPTER 1

Help

Join **#transtats** channel at Libera.Chat, write to transtats@redhat.com

1.1 Overview

1.1.1 Configuration

1. **Inventory** Languages & their sets, translation platforms and product releases are grouped as inventory. One product can have multiple releases. Like, Fedora and Fedora 34, Fedora 35 are releases. Sample data contains some of them. Inventory are basis to releases, packages, jobs and graph rules. And hence probably the first thing to look at.
2. **Releases** A particular release which has a schedule and information regarding *in how many languages it will be available*. Releases are primary to branch mapping. A package associated with any of the product(s), automatically being tracked for all releases underneath. One release can be associated with one language set.
3. **Packages** Translation progress would be tracked for added packages. They should have upstream repository URL and translation platform project. A package can be linked with multiple products and should have a branch mapping. Once all versions / tags are sync'd with respective sources, translation differences could be generated. Packages should be sync'd at intervals.

1.1.2 Summary and Details

1. **Releases - Translation Progress** Translation update volume estimation for a product release at an early stage of a release cycle. It has three views: combined, detailed and language-wise. Combined will sum up translated and untranslated for all languages for a package. Detailed contains language wise %age representation. Language-wise shows overall picture as well as translated, untranslated statistics of every package for each language.
2. **Packages - Translation Completeness** Translation progress, gaps, errors of a package by syncing with source repositories, translation platforms and build systems. Package needs to get sync'd with all three to fill

statistics. Stats are represented in tabular and graph forms, per language also. This shows translation trends of last few releases. Summary highlights out-of-sync packages.

3. **Insights - Translation Coverage** Coverage of a group of packages for a specific release in associated or selected languages. These are based on graph rules. Packages must have branch mapping and respective sync done before they can participate in graph rule.

1.1.3 Workflow Automation

1. **Job templates** Currently, seven templates are to choose from. They can be used as-is.

   `syncupstream`, `syncdownstream`, `stringchange`, `pushtrans`, `dpushtrans`, `pulltrans` and `pulltransmerge`

   - ***syncupstream*** clone package source repository, filter translation files and calculate statistics
   - ***syncdownstream*** locate latest built SRPM, unpacks, filter translation files and calculate statistics
   - ***stringchange*** clone package source repo, generates template (POT) as per given command, download respective template from platform and compare/find diff
   - ***pushtrans*** clone package source repository, filter translations and upload to the CI platform.
   - ***dpushtrans*** download translations from package translation platform and upload to the CI platform.
   - ***pulltrans*** download translations from CI platform and submit back.
   - ***pulltransmerge*** download translations from CI platform and creates merge request.

   Requires four values:

   - Package Name (*derived stats can be saved only if this is added already*)
   - Build System (*build system associated with the release*)
   - Build Tag (*run sync task for build tags to keep the dropdown updated*)
   - Release Slug (*find this on the release page, at tooltip or in URL*)
   - Pipeline UUID (*gets generated as soon as a pipeline is created*)

   SyncDownstream could be run for any package (also for those not added in Transtats) and for any build tag available. This makes the job really wonderful tool to inspect SRPM.

   Dry runs are also supported. Each YAML job has a unique URL to see details and share!

2. **Tasks**

   Functions which are basis to fetch some essential data

   - sync with translation platform for project details.
   - sync with release schedule to update calendar.
   - **sync with build system for their latest build tags.** *required for branch mapping (this is one of the first steps)*

   Logs are kept.

1.2 Transtats Server

transtats is a django application with postgres backend.

1.2.1 Releases

Transtats can have one or multiple product(s) in one instance. And each Product can have multiple releases. Staff role can manage products and releases. Adding a release has following fields:

`Release Branch Name` Product release name with version. for example: Fedora 37

`Release Slug` Transtats will formulate slug. *And this slug will be validated with iCal retrieved from URL.* for example: fedora-37

`Current Phase` Phase in which this release is running. for example: Development

`Language Set` Language set which should be associated with this release. *This is useful when we need to track a release for custom set of languages.*

`iCal URL` Release schedule calendar URL. To fetch release milestone and dates.

`SCM Branch` (optional) Release Build System Default Tag. for example: f37

`Enable flags` Track Translation will enable translation tracking for this release.

1.2.2 Languages

Languages are the backbone of translations tracking and other operations. Adding a language has following fields:

`Language Name` Name of the language. for example: Japanese

`Locale ID` Locale in the form of LANG_TERRITORY. for example: ja_JP

`Locale Script` Script the language is based upon. for example: Hani

`Locale Alias` (optional) for example: zh-Hans can be an alias for zh_CN locale.

1.2.3 Platforms

In Transtats, Platform refers to a translation platform. for example Damned Lies, Phrase, Transifex, Weblate, Zanata, etc. Adding a Platform has following fields:

`Platform Engine` (dropdown) Platform engine helps system to determine specific tasks. for example: WEBLATE

`Platform Subject` for example: Fedora

`Server URL` for example: https://translate.fedoraproject.org

`Platform SLUG` (dropdown) for example: WLTEFED refers to Weblate Fedora instance

`Enable/Disable` To enable platform for sync operations.

`Server URL` for example: https://translate.fedoraproject.org

`Auth User` Username for API auth purposes.

`Auth Password/Token` Password or Token for API auth purposes.

`CI Enable/Disable` To enable platform for CI Pipeline operations.

1.2.4 Packages

An authenticated user can create and manage a package. (a project in Transtats) Adding a Package has following fields:

`Package Name` Package id as-in translation platform. *Existence is validated with the translation platform.*

`Upstream URL` Source repository location (Bitbucket, GitHub, Pagure etc).

`Upstream Localization URL` (optional) Source repository location with translation resources.

`Translation Format` (radio) File format translations are stored in the project.

`Translation Platform` (dropdown) Translation statistics will be fetched from this server.

`Products` (check) Translation progress for selected products will be tracked.

1.3 Transtats APIs

1. **Ping Server** : `<transtats_server>/api/ping`

   Returns server version.

   ```
   GET /api/ping HTTP/1.1
   ```

2. **Package Status** : `<transtats_server>/api/package/<package_name>`

   Returns translation stats of package for enabled languages, for example `abrt`.

   ```
   GET /api/package/abrt HTTP/1.1
   ```

3. **Package Health** : `<transtats_server>/api/package/<package_name>/health`

   Returns health of package w.r.t out-of-sync, for example `abrt`.

   ```
   GET /api/package/abrt/health HTTP/1.1
   ```

4. **Add Package** : `<transtats_server>/api/package/create`

   Registers a new package with Transtats, for example `dnf`.

   ```
   POST /api/package/create HTTP/1.1

   example:
   $ curl -d '{"package_name": "dnf", "upstream_url": "https://github.com/
   ↪rpm-software-management/dnf", "transplatform_slug": "WLTEFED", "release_
   ↪streams": "fedora,RHEL"}' -H "Authorization: Token <your-transtats-api-
   ↪token>" -H "Content-Type: application/json" -X POST http://
   ↪localhost:8080/api/package/create

   output:
   {"dnf":"Package added Successfully."}
   ```

5. **Graph Rule Coverage** : `<transtats_server>/api/coverage/<graph_rule_name>`

   Returns translation coverage according to graph rule, for example `rhinstaller`.

   ```
   GET /api/coverage/rhinstaller HTTP/1.1
   ```

6. **Release Status** : `<transtats_server>/api/release/<release_branch_name>`

   Returns translation stats of packages which are being tracked for a given release, for example `fedora-29`.

   ```
   GET /api/release/fedora-29 HTTP/1.1
   ```

a. **Release Status Detail** : `<transtats_server>/api/release/<release_branch_name>/detail`

Returns per language translation stats of packages for a release.

```
GET /api/release/fedora-29/detail HTTP/1.1
```

b. **Release Status Locale** : `<transtats_server>/api/release/<release_branch_name>/locale/<locale>`

Returns translation stats of packages for a release of a single language.

```
GET /api/release/fedora-29/locale/ja_JP HTTP/1.1
```

7. **Job Details** : `<transtats_server>/api/job/<job-id>/log`

Returns job log against given job id, for example `2a6d4b23-6a6b-4d0e-b617-a0ece01d790f`.

```
GET /api/job/2a6d4b23-6a6b-4d0e-b617-a0ece01d790f/log  HTTP/1.1
```

8. **Run a Job** : `<transtats_server>/api/job/run`

Returns the job_id against given job_type: *syncupstream* or *syncdownstream* or *stringchange*

```
POST /api/job/run  HTTP/1.1

example:
$ curl -d '{"job_type": "syncdownstream", "package_name": "anaconda",
↪"build_system": "koji", "build_tag": "f33"}' -H 'Content-Type:␣
↪application/json' -H 'Authorization: Token <your-transtats-api-token>' -
↪X POST http://localhost:8080/api/job/run

output:
{"Success":"Job created and logged. URL: http://localhost:8080/jobs/log/
↪2a5966a9-3e5e-4ad1-b89e-1ee0e3b1651b/detail","job_id":"2a5966a9-3e5e-
↪4ad1-b89e-1ee0e3b1651b"}
```

1.4 Transtats CLI

transtats-cli is a command line interface to query transtats server.

- **Configuration**

`transtats.conf` should be placed inside `~/.config/` directory. Transtats server url and API token can be added as

```
[server]
server_url = https://transtats.fedoraproject.org
token = <API-token-from-server>
```

- **Usage**

```
$ transtats [OPTIONS] COMMAND [ARGS]...
```

- **Options**

--help Show help message and exit.

- **Commands**

1. **coverage** Translation coverage as per rule.

```
transtats coverage [OPTIONS] RULE_NAME
```

2. **job** Runs a job and/or show the job log.

```
transtats job [OPTIONS] COMMAND [ARGS]...
```

3. **package** Translation status of a package.

```
transtats package [OPTIONS] PACKAGE_NAME
```

4. **version** Display the current version.

```
transtats version [OPTIONS]
```

5. **release** Translation status of a release.

```
transtats release [OPTIONS] RELEASE_SLUG
```

1.5 Get Involved

Welcome! Read Transtats Contributing Guide for getting started.

1.5.1 Try and test

- **Docker**

  Get docker daemon running. Build or pull transtats image and get started.

  - Build the image *(optional)*

    * **Clone the repo and build the image**

```
$ git clone https://github.com/transtats/transtats.git
$ cd transtats
$ sudo docker build -t transtats/transtats deploy/docker
```

  - **Pull the image *(No need to pull, if you have built the image)***

```
$ sudo docker pull docker.io/transtats/transtats
```

  - **Run the image**

```
$ sudo docker run -d --name container_name -p 8080:8015 transtats/
↪transtats
```

    or you can specify custom database credentials using environment variables

```
$ sudo docker run -d --name container_name -p 8080:8015 -e DATABASE_
↪NAME=db_name \
       -e DATABASE_USER=db_user -e DATABASE_PASSWD=db_passwd transtats/
↪transtats
```

  - Application should be available at `localhost:8080` with `transtats | transtats` as login credentials.

1.5.2 Hack and Develop

- Install and run Ansible, Docker and Vagrant.
- **This will setup devel environment and run container plus, *ssh* into it**

```
$ sudo vagrant plugin install vagrant-hostmanager
$ git clone https://github.com/transtats/transtats.git
$ cd transtats
$ sudo vagrant up
$ sudo vagrant ssh
```

- **Run application**

```
$ cd /workspace
$ make run
```

- Hit `localhost:8080` in browser
- Create migrations `make migrations`
- Collect static files `make static`
- Run tests `make lint test`
- Generate docs `make docs`

1.5.3 Contribute

- The *devel* branch is the release actively under development.
- The *master* branch corresponds to the latest stable release.
- If you find any bug/issue or got an idea, open a GitHub issue.
- Feel free to submit feature requests and/or bug fixes on *devel* branch.
- Transtats uses CircleCI for tests.

1.6 Roadmap

As the project evolves, a roadmap will be published for each major release. Comments, suggestions, and requests to the current roadmap are welcome. Our goal in publishing a roadmap is transparency and community inclusion.
A roadmap is the team's best guess based on experience, community requests, and feedback.

1.6.1 Releases

Please look github releases for details.
See changelog here.

Upcoming Releases

Versioning is mostly *MAJOR.MINOR.PATCH*
And releases are scheduled every quarter end.

Transtats 0.8.8

Target delivery: End of March 2023

1.6.2 ToDo List

- **New Feature, Bug fix, Enhancement**
  - Provision for package sets.
  - Search a job log.
  - Translation snapshots.
  - Tenants specific changes.
  - Translation files in multiple tarballs.
- **Technical Tasks**
  - Error handling, test cases.
  - Simultaneous jobs run (and multi-threading).
  - **Prepare deployment for OpenShift 4.**
    - Move from docker to buildah, single container to a pod.
  - **Find an alternative of vagrant in dev env.**
    - docker-compose may be.
- **Expanding Support**
  - **Translation File Format *in Jobs***
    - Java files (`properties`, `dtd`)
    - PHP, JS (`ini`, `json`, `js`)
  - More YAML Job Template Implementations
- **Automation**
  - Batch processing of pipeline actions.
  - Scheduling jobs based on release dates.
  - Scheduling of pipeline configurations.
- **User Personalization**
  - Multiple Authentication: FAS, SAML, Social.
  - API sync for package, language-group ownerships.
  - User panel to configure package and other settings.
  - Customization of interfaces as per logged in user.
  - Notifications to package and/or language maintainers.
- **Integration**
  - Fedora Apps, Internal tools
  - Bugzilla, JIRA, IRC bot

- **Documentation**
  - Use cases (`docs`, `blog`, `transtats.org`)
  - User guide (`pdf`, `screencast`)
  - Release notes, developers manual

For complete list please browse github issues.

1.7 License and Credits

Transtats is licensed under Apache License Version 2.0

Contributors

Build by awesome people contributed in one form or the other :)

- transtats
- transtats-cli
- test-automation
- transtats-ansible
- transtats.github.io