BookStack CLI

Documentation for the bookstack-cli-rs tool

- Generic Packages in CI/CD
- Getting Started
- CLI Reference

Generic Packages in CI/CD

Forgejo (and Gitea) provide a **generic package registry** that allows you to publish arbitrary files from CI/CD pipelines — ideal for distributing compiled binaries, configuration files, or any static assets.

Authentication

Forgejo requires **Basic authentication** for the generic packages API. This differs from the standard REST API which accepts `Authorization: token` headers.

```
curl --user "<username>:<PAT>" ...
```

Where `<PAT>` is a Personal Access Token with the `write:package` scope.

Publishing Files

Upload a single file

```
curl -X PUT \
  --user "<username>:<PAT>" \
  --upload-file ./mybinary \
  https://git.example.com/api/packages/<owner>/generic/<repo>/<version>/<filename>
```

Path components:

Component	Description
`<owner>`	Your username or organization name
`<repo>`	Repository name (used as package namespace)
`<version>`	Version string (e.g. `v1.0.0`, `0.2.1-rc1`)
`<filename>`	The actual file to upload

Upload multiple files for the same version

```
curl -X PUT \
  --user "<username>:<PAT>" \
  --upload-file ./linux-binary \
  https://git.example.com/api/packages/<owner>/generic/<repo>/<version>/linux-binary

curl -X PUT \
  --user "<username>:<PAT>" \
  --upload-file ./macos-binary \
  https://git.example.com/api/packages/<owner>/generic/<repo>/<version>/macos-binary
```

Each file is stored independently under the same version path.

Downloading Files

Direct download with authentication

```
curl -fsSL --user "<username>:<PAT>" \
  https://git.example.com/api/packages/<owner>/generic/<repo>/<version>/<filename> \
  --output ./downloaded-binary
```

The `-fsSL` flags:

- `-f`: Fail silently on HTTP errors
- `-s`: Silent mode (no progress meter)
- `-S`: Show errors if silent mode is used
- `-L`: Follow redirects

Without authentication (public packages)

If the package repository is public, you can download without credentials:

```
curl -fsSL \
  https://git.example.com/api/packages/<owner>/generic/<repo>/<version>/<filename> \
  --output ./downloaded-binary
```

GitHub Actions Workflow Example

```
name: Release
on:
  push:
    tags:
      - "v*"

permissions:
  contents: write
  packages: write

jobs:
  release:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4

      - name: Build release binary
        run: cargo build --release

      - name: Upload to generic registry
        env:
          GITEA_TOKEN: ${{ secrets.PACKAGE_TOKEN }}
        run: |
          curl -X PUT \
            --user "${{ github.actor }}:${GITEA_TOKEN}" \
            --upload-file target/release/myapp \
            https://git.example.com/api/packages/${{ github.repository_owner
}}/generic/myapp/${{ github.ref_name }}/myapp
```

Gitea Actions Workflow Example

```
name: Release
on:
  push:
    tags:
      - "v*"

jobs:
```

```
  release:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4

      - name: Build release binary
        run: cargo build --release

      - name: Upload to generic registry
        env:
          GITEA_TOKEN: ${{ secrets.PACKAGE_TOKEN }}
        run: |
          curl -X PUT \
            --user "username:$GITEA_TOKEN" \
            --upload-file target/release/myapp \
            https://git.example.com/api/packages/${{ github.repository_owner
}}/generic/myapp/${{ github.ref_name }}/myapp
```

Troubleshooting

Error	Cause	Solution
`401 Unauthorized`	Missing or invalid PAT	Verify token has `write:package` scope
`403 Forbidden`	Insufficient permissions	Ensure user has write access to the repository
`404 Not Found`	Wrong path or private package	Check owner/repo/version/filename; use auth for private packages
`405 Method Not Allowed`	Wrong HTTP method	Use `PUT`, not `POST`
`500 Internal Server Error`	Content-Type mismatch	Remove `Content-Type: application/json`; let curl auto-detect

Key Differences: Forgejo vs Gitea

Both Forgejo and Gitea support generic packages, but authentication differs:

Endpoint	Gitea Auth	Forgejo Auth
`/api/v1/...` (REST)	`Authorization: token <PAT>`	`Authorization: token <PAT>`
`/api/packages/...` (packages)	`Authorization: token <PAT>`	**Basic auth** (`--user user:token`)

Always use Basic auth for the packages API to ensure compatibility across both platforms.

Getting Started

Getting Started

Overview

`bookstack-cli-rs` is a command-line interface for the BookStack API, written in Rust. It supports profile-based authentication, all CRUD operations for books/chapters/pages/shelves, and full-text search. Container filtering lets you narrow list results by shelf (books), book (chapters), or book/chapter (pages).

Installation

From Source

```
cargo install --path .
```

Pre-built Binaries

Download from Gitea generic packages:

```
curl -fsSL \
  --user "<username>:$GITEA_TOKEN" \
  https://git.example.com/api/packages/<owner>/generic/bookstack-cli-rs/v0.2.0/bookstack-cli-rs-x86_64-unknown-linux-gnu \
  -o bookstack-cli-rs && chmod +x bookstack-cli-rs
```

Version

Current version: **v0.2.0**

Configuration

Profiles store your BookStack instance URL and API tokens. Config is saved to `~/.config/bookstack-cli-rs/profiles.toml`.

Add a Profile (CLI)

```
bookstack profiles add --name mywiki \
  --url https://wiki.example.com \
  --token-id ABCDEF1234567890 \
  --token-secret 0123456789ABCDEF
```

Add a Profile (Environment Variables)

```
export BOOKSTACK_URL=https://wiki.example.com
export BOOKSTACK_TOKEN_ID=ABCDEF1234567890
export BOOKSTACK_TOKEN_SECRET=0123456789ABCDEF

bookstack profiles add --name mywiki
```

List Profiles

```
bookstack profiles list
```

Output:

```
Configured profiles:
  [default] My Wiki - https://wiki.example.com (default)
  [mywiki] Another Wiki - https://wiki2.example.com
```

Delete a Profile

```
bookstack profiles delete --name mywiki
```

Using Profiles

Specify which profile to use with `-p` or `--profile`:

```
bookstack -p mywiki list-books
bookstack list-books                    # uses default profile if no -p given
```

Basic Usage Examples

List All Books

```
bookstack list-books
```

Filter Books by Shelf

```
bookstack list-books --shelf-id 1
```

List Chapters in a Book

```
bookstack list-chapters --book-id 9
```

List Pages in a Chapter

```
bookstack list-pages --chapter-id 42
```

Search Content

```
bookstack search "rust"
bookstack search "rust" --type page --count 10
```

Get Page with Full Content

```
bookstack get-page 308 --content      # show HTML content
bookstack get-page 308 --markdown     # show markdown content
```

Next Steps

- See the CLI Reference for all commands and options.
- Check out Generic Package Publishing Debugging for CI/CD integration notes.

CLI Reference

CLI Reference

Global Options

Option	Description
`-p, --profile <NAME>`	Profile name to use (uses default if not specified)

Profiles

Manage authentication profiles.

`profiles add`

Add a new profile with URL and API tokens.

```
bookstack profiles add --name mywiki --url https://wiki.example.com \
  --token-id <TOKEN_ID> --token-secret <TOKEN_SECRET>
```

Option	Description	Environment Variable
`--name <NAME>`	Profile name	—
`--url <URL>`	BookStack instance URL	`BOOKSTACK_URL`
`--token-id <ID>`	API Token ID	`BOOKSTACK_TOKEN_ID`
`--token-secret <SECRET>`	API Token Secret	`BOOKSTACK_TOKEN_SECRET`

`profiles list`

List all configured profiles.

```
bookstack profiles list
```

`profiles delete`

Delete a profile by name.

```
bookstack profiles delete --name mywiki
```

Books

`list-books`

List all books, optionally filtered by shelf.

```
bookstack list-books [--shelf-id <ID>] [--count <NUM>]
```

Option	Description
`--shelf-id <ID>`	Filter by shelf ID
`--count <NUM>`	Max results (default: 20)

`get-book`

Get details of a single book.

```
bookstack get-book <ID>
```

`create-book`

Create a new book.

```
bookstack create-book <NAME> [DESCRIPTION]
```

`update-book`

Update an existing book.

```
bookstack update-book <ID> [--name <NAME>] [--description <DESC>]
```

`delete-book`

Delete a book.

```
bookstack delete-book <ID>
```

Chapters

`list-chapters`

List all chapters, optionally filtered by book.

```
bookstack list-chapters [--book-id <ID>] [--count <NUM>]
```

Option	Description
`--book-id <ID>`	Filter by book ID
`--count <NUM>`	Max results (default: 20)

`get-chapter`

Get details of a single chapter.

```
bookstack get-chapter <ID>
```

`create-chapter`

Create a new chapter in a book.

```
bookstack create-chapter --book-id <ID> --name <NAME> [--description <DESC>]
```

`update-chapter`

Update an existing chapter.

```
bookstack update-chapter <ID> [--name <NAME>] [--description <DESC>] [--book-id <ID>]
```

`delete-chapter`

Delete a chapter.

```
bookstack delete-chapter <ID>
```

Pages

`list-pages`

List all pages, optionally filtered by book or chapter.

```
bookstack list-pages [--book-id <ID>] [--chapter-id <ID>] [--count <NUM>]
```

Option	Description
`--book-id <ID>`	Filter by book ID
`--chapter-id <ID>`	Filter by chapter ID (includes sub-chapters)
`--count <NUM>`	Max results (default: 20)

`get-page`

Get a page with optional content display.

```
bookstack get-page <ID> [--content|--markdown]
```

Option	Description
`--content`	Show full HTML content
`--markdown`	Show markdown content

`create-page`

Create a new page in a book or chapter.

```
bookstack create-page --name <NAME> \
  [--book-id <ID>] [--chapter-id <ID>] \
```

```
  [--html <HTML>] [--markdown <MD>] [--content <TEXT>]
```

At least one of `--book-id` or `--chapter-id` is required. At least one of `--html`, `--markdown`, or `--content` is required.

update-page

Update an existing page.

```
bookstack update-page <ID> \
  [--name <NAME>] [--html <HTML>] [--markdown <MD>] [--content <TEXT>] \
  [--book-id <ID>] [--chapter-id <ID>]
```

delete-page

Delete a page.

```
bookstack delete-page <ID>
```

Shelves

list-shelves

List all shelves.

```
bookstack list-shelves [--count <NUM>]
```

get-shelf

Get details of a single shelf.

```
bookstack get-shelf <ID>
```

create-shelf

Create a new shelf.

```
bookstack create-shelf <NAME> [DESCRIPTION]
```

Search

Search across all content types.

```
bookstack search <QUERY> [--type <TYPE>] [--count <NUM>]
```

Option	Description
`--type <TYPE>`	Filter by type: `page`, `chapter`, `book`, `bookshelf`
`--count <NUM>`	Max results (default: 20)

Examples

```
# Search all content for "rust"
bookstack search "rust"

# Search only pages
bookstack search "rust" --type page

# Search with limit
bookstack search "configuration" --count 5
```