bioconda2rpm Cheat-Sheet (2026-02-28)

1) What It Does

- Converts Bioconda recipes into Phoreus-style RPM artifacts.
- Canonical build order per package is always:
  - `SPEC -> SRPM -> RPM`
- Default behavior resolves Bioconda dependencies and builds dependencies first.

2) Core Command

```bash
bioconda2rpm build <tool...>
```

3) Fast Start Examples

Single package:

```bash
cargo run -- build samtools
```

Batch queue build:

```bash
cargo run -- build bbmap samtools blast fastqc \
  --sync-recipes \
  --queue-workers 4 \
  --parallel-policy adaptive \
  --build-jobs 4
```

Packages file:

```bash
cargo run -- build \
  --packages-file ./docs/verification_software.txt
```

4) Controlled Container Profiles

Allowed values:

- `almalinux-9.7` (default)
- `almalinux-10.1`
- `fedora-43`

Usage:

```bash
cargo run -- build fastqc \
  --container-profile almalinux-10.1
```

Behavior:

- If selected image is already local: build starts immediately.
- If selected image is missing: bioconda2rpm auto-builds it from:
  - `containers/rpm-build-images/Dockerfile.almalinux-9.7`
  - `containers/rpm-build-images/Dockerfile.almalinux-10.1`
  - `containers/rpm-build-images/Dockerfile.fedora-43`

Managed recipes behavior:

- first run auto-clones `https://github.com/bioconda/bioconda-recipes`
- default managed root: `~/bioconda2rpm/bioconda-recipes/recipes`
- `--sync-recipes` refreshes managed refs from origin
- `--recipe-ref <branch|tag|commit>` checks out explicit ref

5) Important Build Flags

- `--stage spec|srpm|rpm` (default: `rpm`)
- `--recipe-root <path>` (optional override)
- `--sync-recipes`
- `--recipe-ref <branch|tag|commit>`
- `--dependency-policy run-only|build-host-run|runtime-transitive-root-build-host`
- `--no-deps` (disable dependency closure)
- `--missing-dependency fail|skip|quarantine` (default: `quarantine`)
- `--parallel-policy serial|adaptive` (default: `adaptive`)
- `--build-jobs <N|auto>` (default: `4`)
- `--queue-workers <N>` (batch queue concurrency)
- `--arch host|x86-64|aarch64` (default: `host`)
- `--ui plain|ratatui|auto` (default: `auto`)
- `--container-engine docker|podman|...` (default: `docker`)

6) UI and Runtime Visibility

- `--ui auto`: `ratatui` dashboard on interactive terminals, plain logs otherwise.
- `--ui ratatui`: always use terminal dashboard.
- `--ui plain`: always plain progress logs.

The dashboard/logs surface:

- package queue status
- dependency-follow progress
- current package state
- elapsed runtime
- failure details and reason

7) Output Layout (Default)

Default topdir:

- `~/bioconda2rpm`

Artifact/report layout:

- `~/bioconda2rpm/SPECS`
- `~/bioconda2rpm/SOURCES`
- `~/bioconda2rpm/targets/<target-id>/SRPMS`
- `~/bioconda2rpm/targets/<target-id>/RPMS`
- `~/bioconda2rpm/targets/<target-id>/reports`
- `~/bioconda2rpm/targets/<target-id>/BAD_SPEC`

`<target-id>` is derived from resolved container image + target arch.

8) Regression and Priority Helpers

Regression (PR top-N):

```bash

```
cargo run -- regression \
  --tools-csv ../software_query/tools.csv \
  --mode pr --top-n 25
```

Regression (curated software list):

```bash
cargo run -- regression \
  --tools-csv ../software_query/tools.csv \
  --software-list ./docs/verification_software.txt
```

Priority generation helper:

```bash
cargo run -- generate-priority-specs \
  --tools-csv ../software_query/tools.csv \
  --top-n 10
```

Managed recipes command:

```bash
cargo run -- recipes --sync
cargo run -- recipes --recipe-ref 2025.07.1
```

9) Operational Rules

- Prefer `build <tool>` as production path.
- Keep one SPEC per software; use `%ifarch` for arch-specific sections.
- Treat generated SPEC/SRPM/RPM artifacts as ephemeral outputs.
- Keep crate workspace clean; build artifacts belong under `topdir`.

10) Failure Triage (Quick)

- Check package note:
  - `~/bioconda2rpm/targets/<target-id>/BAD_SPEC/<tool>.txt`
- Check build log:
  - `~/bioconda2rpm/targets/<target-id>/reports/build_logs/<tool>.log`
- Check dependency graph:
  - `~/bioconda2rpm/targets/<target-id>/reports/dependency_graphs/<tool>.md`