

## SDD - Command Line Interface (CLI)

In dieser Dokumentation wird ausschließlich die Struktur und der Aufbau des Moduls beschrieben. Eine nähere Beschreibung der Komponenten befindet sich in Doc-Kommentare im Code selber.

- Rust Crate: episko\_cli
- Dokumentation: docs.rs

### Verwendete Sprachen

Die CLI-Komponente ist ausschließlich in der Programmiersprache Rust geschrieben.

### Verwendete Librarys

| Library    | Zweck  | Version |
|------------|--|---------|
| camino     | UTF8-Pfade zur Verwendung bei CLI-Argumenten   | 1.1.9   |
| clap       | Übergabe und Verwaltung von CLI-Argumenten.<br>Automatisch generierte Hilfe-Beschreibung | 4.5.26  |
| color_eyre | Optische Verbesserungen der interaktiven Nutzereingabe                                   | 0.6.3   |
| dialoguer  | Interaktive Nutzereingaben   | 0.11.0  |
| tokio      | Ermöglicht asynchrone Funktionsaufrufe für die<br>Datenbankanbindung                     | 1.43.0  |

### Modulbeschreibung

| ModulBeschreibung |   |
|-------------------|---|
| cli               | Beinhaltet die clap-Einstellungen für die CLI (Argumente und Flags)   |
| creation          | Beinhaltet die Sammlung von Eingabedaten des Nutzers (interaktiv und nicht-interaktiv), Verarbeitung und anschließender Weitergabe dieser an die entsprechende interne Library-Funktion zum Erstellen und Aufnehmen einer neuen Manifestdatei |
| removal           | Beinhaltet den Aufruf zur Löschung einer Manifestdatei und ihrer internen Speicherung   |
| validation        | Beinhaltet den Aufruf zur Validierung und zum Caching einer Manifestdatei   |

### Modul creation

Da das creation-Modul umfangreich ist, wurden die Nutzereingaben, die Prompts, in das Modul cli/prompts ausgelagert.

## CLI-Argumente

Das Programm wird wie folgt aufgerufen: **episko** <COMMAND>

Die Angabe eines Commands ist verpflichtend.

Auf die Angabe von clap standardmäßig bereitgestellten Commands wird folgend verzichtet! ### Liste aller Commands | Command | Nächstes Argument | Beschreibung |

|  | create   | [OPTIONS] | Erzeugen einer neuen Manifestdatei                                   |
|--|----------|-----------|--|
|  | remove   | <FILE>    | Entfernen einer gegebenen Datei aus dem Dateisystem und dem Programm |
|  | cache    | <FILE>    | Hinzufügen einer gegebenen Datei in das Programm                     |
|  | validate | <FILE>    | Validierung einer gegebenen Datei                                    |

Die Angabe einer Datei ist verpflichtend. Die Angabe von Optionen allerdings ist optional und wird im folgenden genauer beschrieben. ### Optionen des create-Commands | Short-Flag | Long-Flag | Argument | Bedeutung | Syntax |

|  | n | non-interactive | - | Anschalten des nicht-interaktiven Modus                               |
|--|---|-----------------|---|---|
|  | d | directory       |   | Verzeichnis des Projekts   Datenangabe   1 x Pfad                     |
|  | t | title           |   | Titel des Projekts   Datenangabe   1 x Text                           |
|  | c | categories      |   | Kategorien   Datenangabe   n x Text (Leerzeichen-separiert)           |
|  | l | languages       |   | Sprachen   Datenangabe   n x Text:Version (Leerzeichen-separiert)     |
|  | p | preferred_id    |   | Entwicklungsumgebung   Datenangabe   1 x Text:Version                 |
|  | b | build-systems   |   | Build-System   Datenangabe   n x Text:Version (Leerzeichen-separiert) |
|  | D | description     |   | Beschreibung   Datenangabe   1 x Text                                 |
|  | r | repository-url  |   | Repository-Link   Datenangabe   1 x Text                              |

Wenn der Nutzer Daten per Flag eingibt und er sich im interaktiven Modus befindet (**n**-Flag nicht gesetzt), werden alle Daten abgefragt, bis auf die gegeben. Diese werden ohne weitere Eingabe verwendet. Im nicht-interaktiven Modus (**n**-Flag gesetzt) wird der Nutzer nie nach Daten gefragt. Diese muss er alle per Flag mitgeben. Falls hierbei notwendige Daten fehlen (Verzeichnis und Titel) schlägt das Programm fehl.