

# Systemnahe Programmierung in Rust

- "The Book" / Cargo and Crates.io / Kap. 14 -

Hubert Högl

Hochschule Augsburg / Informatik

2023-11-23 11:28:19

- Release profiles
  - Dev (`cargo build`)
  - Release (`cargo build --release`)
- Settings in Cargo.toml (optional)

```
[profile.dev]  
opt-level = 0
```

```
[profile.release]  
opt-level = 3
```

## Crate registry “crates.io”

### Documentation comments

- Markdown
- `cargo doc (rustdoc, Output in target/doc/)`
- `cargo doc --open`

```
/// Adds one to the number given.
///
/// # Examples
///
/// ```
/// let arg = 5;
/// let answer = my_crate::add_one(arg);
///
/// assert_eq!(6, answer);
/// ```
pub fn add_one(x: i32) -> i32 {
    x + 1
}
```

- Examples (run with `cargo test`)
- Panics
- Errors
- Safety

- `///` documents item that **follows** comment
- `//!` documents item that **contains** comment (crate)

```
//! # My Crate
```

```
//!
```

```
//! `my_crate` is a collection of utilities to make performing certain
```

```
//! calculations more convenient.
```

## Public API with `pub use`

- “re-export”
- Allows users of a library to see module paths different from the internal structure

Developer/internal view

```
pub use self::kinds::PrimaryColor;  
pub use self::kinds::SecondaryColor;  
pub use self::utils::mix;
```

Public API

```
use art::mix;  
use art::PrimaryColor;
```

## Publishing crates

Account on crates.io: cargo login ...

API Key

Metadata, crate name, description, license (SPDX)

Multiple licenses: MIT OR Apache-2.0

```
[package]
```

```
name = "guessing_game"
```

```
version = "0.1.0"
```

```
edition = "2021"
```

```
description = "A fun game where you guess what number the computer has chosen."
```

```
license = "MIT OR Apache-2.0"
```

```
[dependencies]
```

```
cargo publish
```

## Publishing crates (2)

Publishing is permanent (no overwrite, no delete)

New versions can be published

Deprecating versions with `cargo yank` ...



# Workspaces

Set of packages with shared Cargo.lock and same target directory

E.g. one binary crate, two library crates

- 1 Create directory
- 2 Add Cargo.toml

```
[workspace]
```

```
members = [  
    "adder",  
    "add_one",  
]
```

## Workspaces (2)

- 3 Run repeatedly `cargo new ...`

(Cargo.toml are individual in each package!)

Optional dependency

```
[dependencies]
add_one = { path = "../add_one" }
```

- 4 Build package: `cargo run -p adder`

Build workspace: `cargo build`

- 5 Tests

`cargo test` will run tests for all crates in the workspace

## Binary installation

```
cargo install ...
```

Installation in `~/.cargo/bin`

Only binary targets can be installed (`src/main.rs` or equivalent file)

## Extending cargo

### Subcommands

`cargo-something = cargo something`

Installation with `cargo install ...`

Example (on my computer):

<code>cargo-asm</code>	<code>cargo-fix</code>	<code>cargo-miri</code>	<code>cargo-profddata</code>
<code>cargo-clippy</code>	<code>cargo-fmt</code>	<code>cargo-nm</code>	<code>cargo-readobj</code>
<code>cargo-cov</code>	<code>cargo-generate</code>	<code>cargo-objcopy</code>	<code>cargo-size</code>
<code>cargo-espflash</code>	<code>cargo-llvm-ir</code>	<code>cargo-objdump</code>	<code>cargo-strip</code>