

Managing Code | External Modules

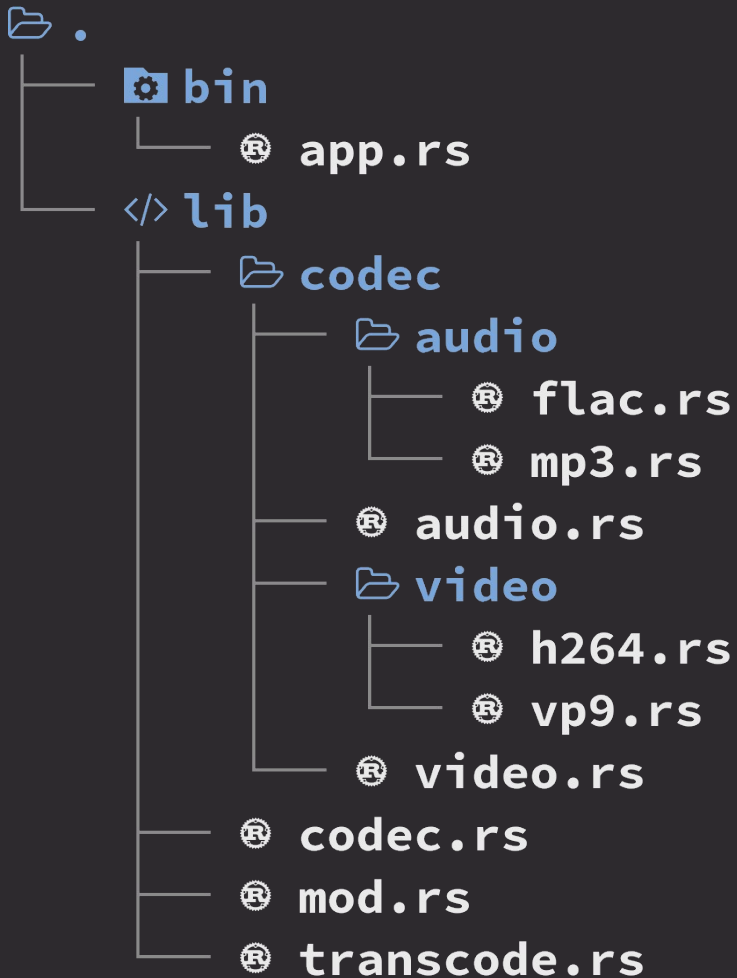
■ External Modules

- ◆ Allows code to be compartmentalized
 - Organized source code management
 - Better collaboration
- ◆ More intuitive coding
 - Quickly identify where imported code is used

■ Module Details

- ◆ Can have any name
- ◆ Hierarchical organization
- ◆ Private by default
 - Use ***pub*** keyword to make a module public
- ◆ *External* modules can be a:
 - Directory
 - ▶ Can contain additional modules
 - File

Directory Structure



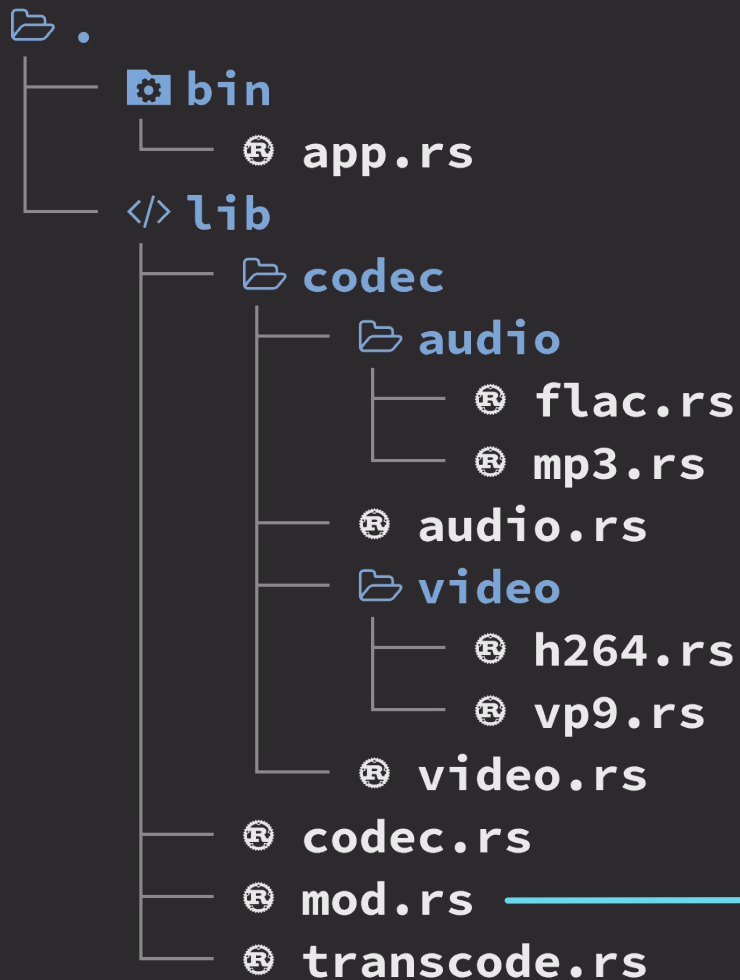
```
# Cargo.toml
```

```
[lib]
```

```
name = "demo"
```

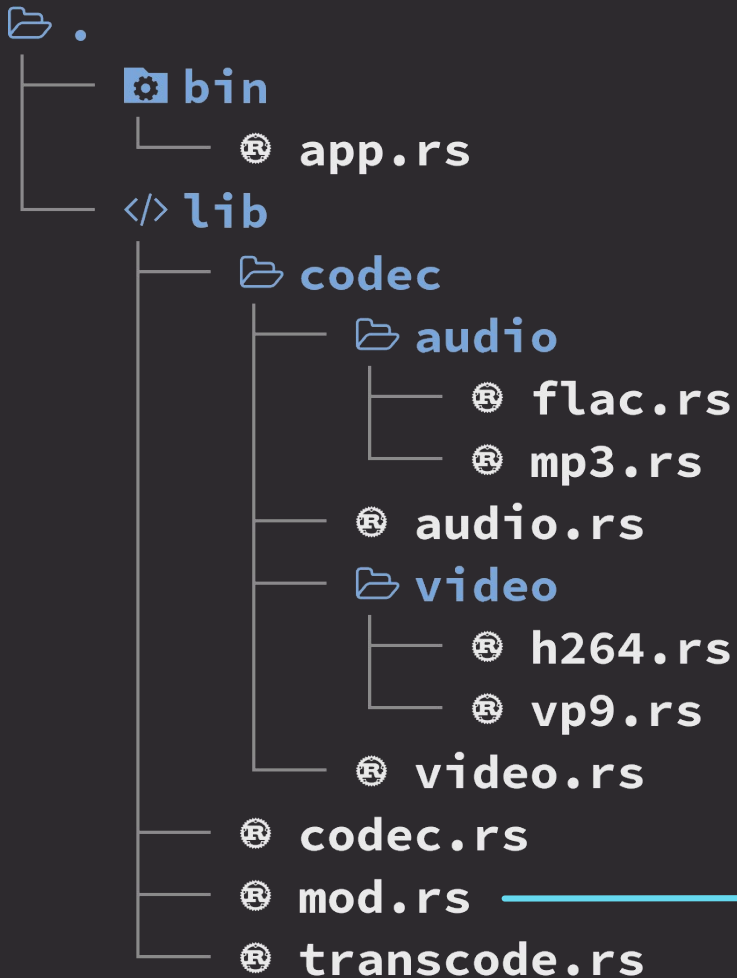
```
path = "src/lib/mod.rs"
```

Module Declaration



```
pub mod codec;  
pub mod transcode;
```

Module Declaration



```
mod inner {
    // ...
}
```

```
pub mod codec;
pub mod transcode;
```

Module Declaration



.

 **bin**

 **app.rs**

 **lib**

 **codec**

 **audio**

 **flac.rs**

 **mp3.rs**

 **audio.rs**

 **video**

 **h264.rs**

 **vp9.rs**

 **video.rs**

 **codec.rs**

 **mod.rs**

 **transcode.rs**

```
pub mod audio;
```

```
pub mod video;
```

```
pub mod codec;
```

```
pub mod transcode;
```

Module Declaration



.

 bin

 app.rs

 lib

 codec

 audio

 flac.rs

 mp3.rs

 audio.rs

 video

 h264.rs

 vp9.rs

 video.rs

 codec.rs

 mod.rs

 transcode.rs

```
pub mod h264;  
pub mod vp9;
```

```
pub mod audio;  
pub mod video;
```

```
pub mod codec;  
pub mod transcode;
```


Module Declaration



.

 bin

Ⓜ app.rs

 lib

 codec

 audio

Ⓜ flac.rs

Ⓜ mp3.rs

Ⓜ audio.rs

 video

Ⓜ h264.rs

Ⓜ vp9.rs

Ⓜ video.rs

Ⓜ codec.rs

Ⓜ mod.rs

Ⓜ transcode.rs

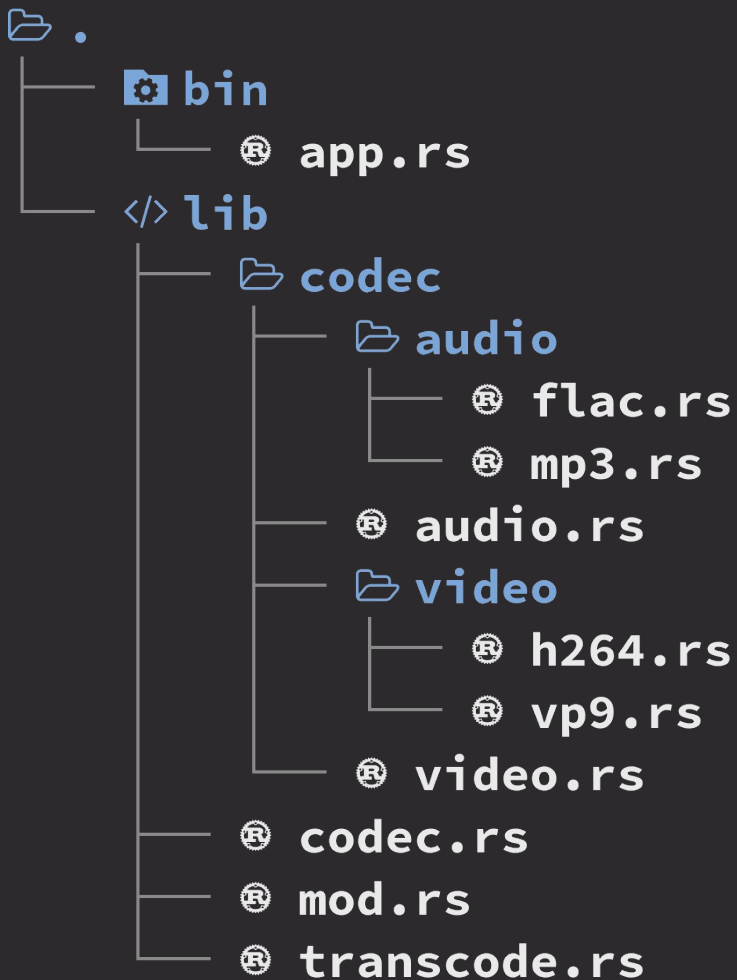
```
pub mod flac;  
pub mod mp3;
```

```
pub mod h264;  
pub mod vp9;
```

```
pub mod audio;  
pub mod video;
```

```
pub mod codec;  
pub mod transcode;
```

■ Accessing Functionality



```
// flac.rs
use super::mp3;
pub fn sample() {
    mp3::some_fn();
    super::mp3::some_fn();
    crate::codec::audio::mp3::some_fn();
    super::super::video::h264::some_fn();
}
```

Module Aliases



.



bin



app.rs



lib



codec



audio



flac.rs



mp3.rs



audio.rs



video



h264.rs



vp9.rs



video.rs



codec.rs



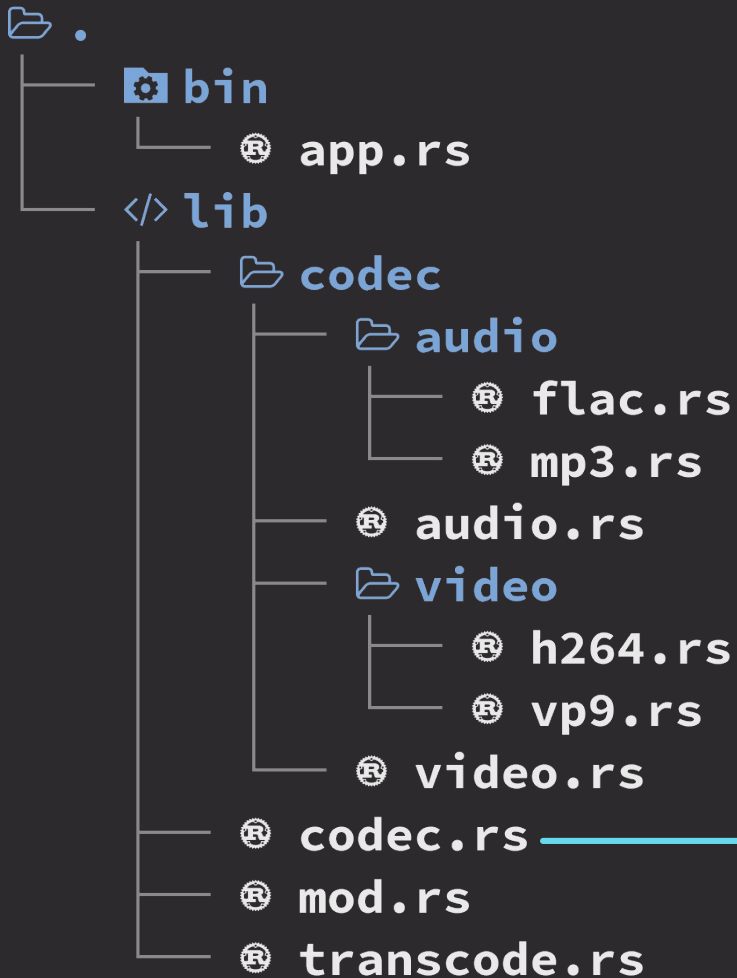
mod.rs



transcode.rs

```
pub fn sample() {  
    use crate::transcode as tc;  
    tc::some_fn();  
}
```

Re-exporting Modules



```
use crate::codec::mp3;
```

```
pub mod audio;
pub mod video;
pub use audio::mp3;
```

Recap

- ◆ Modules are organized hierarchically
 - Use ***super*** to go up one level
 - Use ***crate*** to start from the top
- ◆ The ***as*** keyword can be used to create an alias for a module
- ◆ The ***mod*** keyword is used to declare a module
 - No curly braces for external modules
- ◆ Modules can be re-exported with the ***use*** keyword
- ◆ ***pub*** indicates the module may be accessed from anywhere
 - Omitting ***pub*** restricts access to only the containing module and sub-modules