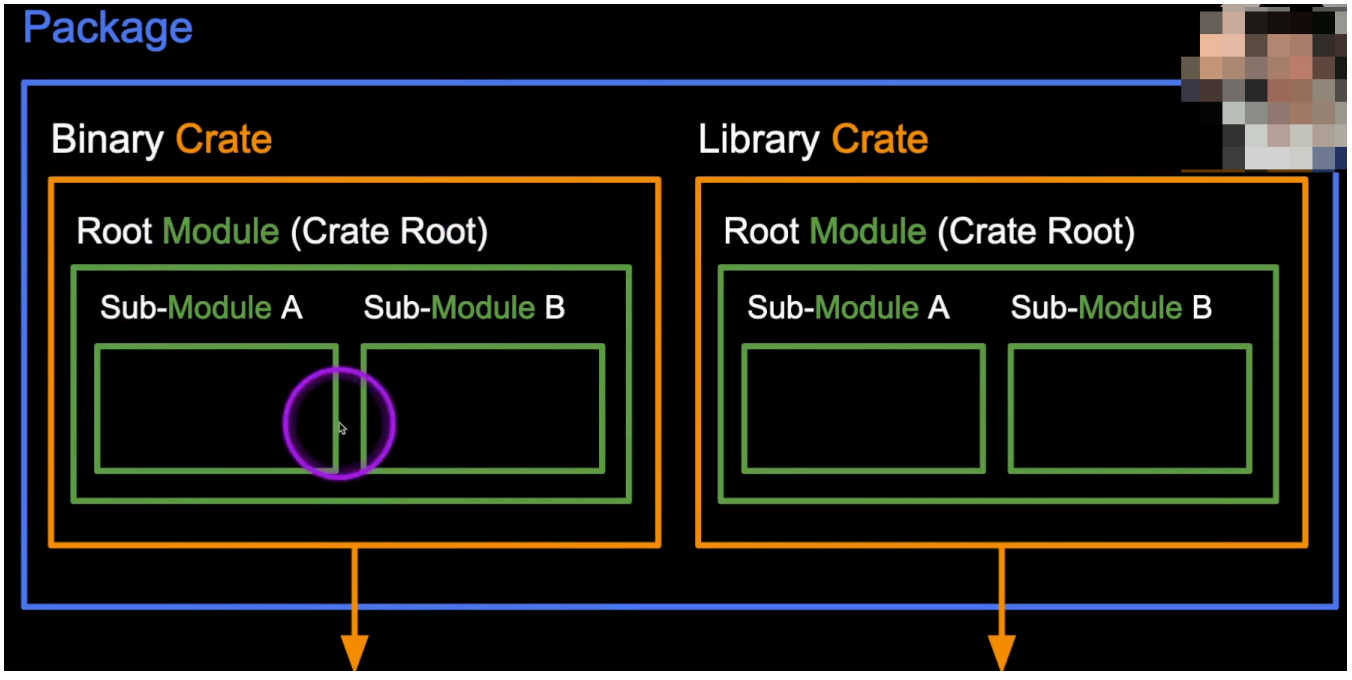
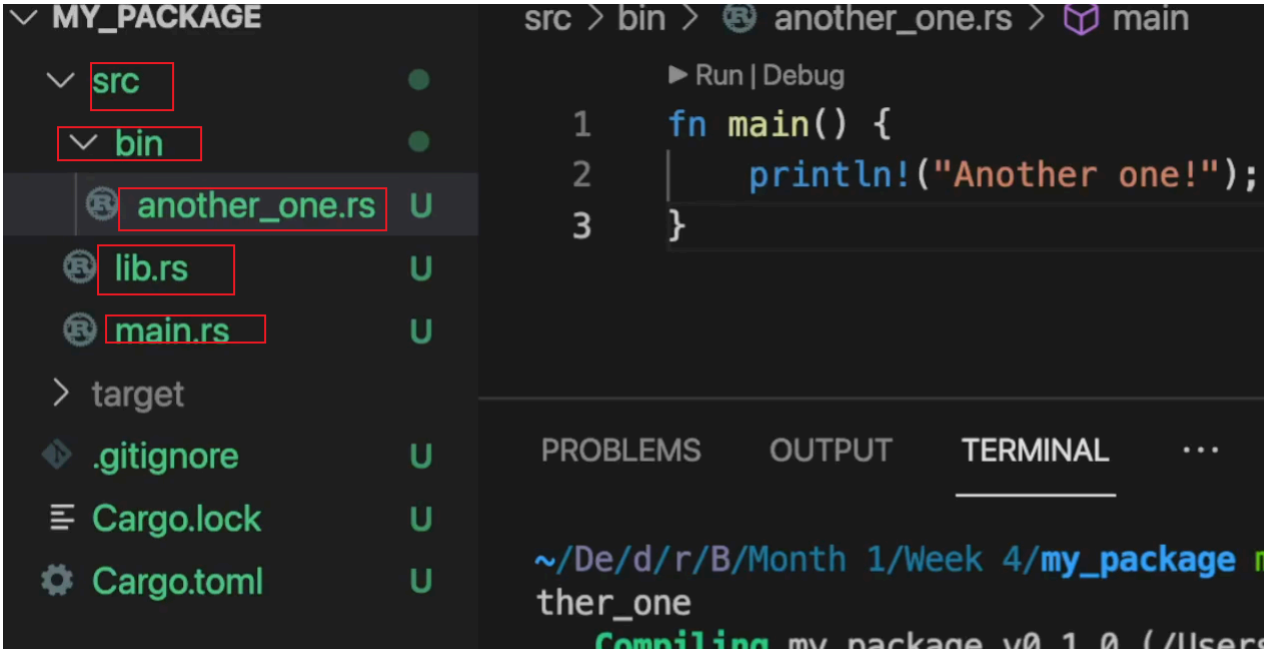
Rust Project Stucture

Structure

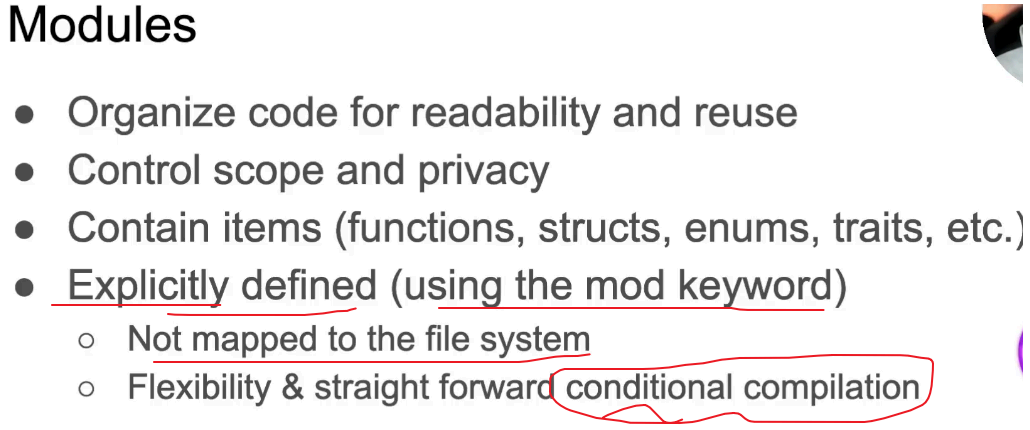
Basic Components

- **Packages**: Contain one or more crates that provide a set of functionality. Packages allow you to build, test, and share crates.
  - **Cargo.toml** - Describes the package and defines how to build crates.
  - **Rules**
    - Must have a least 1 crate
    - At most 1 library crate
    - Any number of binary crates
- **Crates**: A tree of modules that produces a library or executable.
- **Modules**: Let you control the organization, scope, and privacy.





Modules



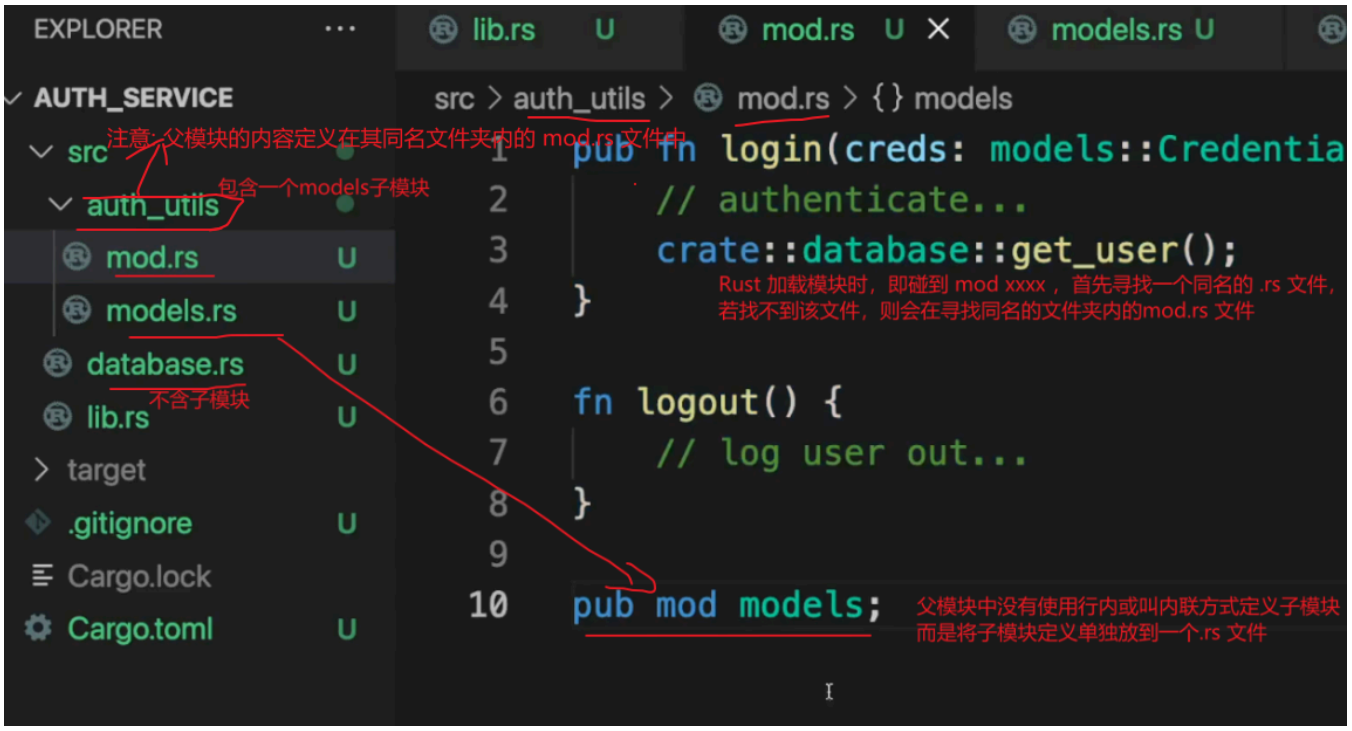
- 模块可以嵌套
- cargo-modules 插件: 可视化Module Tree ，可视化命令: `cargo modules generate tree` 或 `cargo modules generate tree --with-types`
- 模块相对路径、模块绝对路径
- `use`: 创建模块的local 绑定名称
- 子模块必须在父模块中声明
- 模块定义风格: 行内定义 或 单独文件定义，若是单独文件定义，一般在lib.rs中需要使用 `mod 模块名称` 来声明
- mod.rs 与模块同名的文件夹：一般用于有父子模块的情况（子模块在父模块文件夹内呈现为一个独立的文件）

模块定义

> 最终表现：所有模块（无子模块的模块 、含子模块的模块、子模块本身）都在单独的 .rs 文件中

mod.rs 风格

> 有点类似javascript中的 index.js 方式
>
> 此种方式downside: 当在编辑器中打开很多mod.rs 文件时候，很难确定当编辑的是哪个module，根据文件名称看不出来
>
> 2018版rust引入了另一方式



非mod.rs 风格

> 父模块内容直接在同名的 .rs 文件内书写，子模块在与父模块同名的文件夹内定义