自定义包

概述

在rust中，自定义包有4个概念

- package:包，既一个项目成为一个包
- crate 包的树形结构
- modules和use 模块和作用域私有性
- path 模块的路径

package 和 crate

cargo new project crate **src/main.rs** (二进制包: binary) **src/lib.rs**(库包: library) **package**相同

申明module

```
/Cargo.toml
/src/
/src/main.rs
/src/ffmpeg_encoder/mod.rs
/src/ffmpeg_encoder/book.rs
/src/ffmpeg_encoder/people.rs
```

crate根文件中申明 **src/main.rs** **src/lib.rs**

```
# src/main.rs


// 引入模块
use crate::ffmpeg_encoder::book::Book;
use crate::ffmpeg_encoder::people::Employ;
use crate::ffmpeg_encoder::people::Sex;
use crate::ffmpeg_encoder::Shop;

// 定义oo module
pub mod ffmpeg_encoder;
```

```
fn main() {

    println!("{}", Shop::to_string());

    println!("{}", Book::new().to_string());

    let em = Employ::new(String::from("董列涛'), 33, Sex::Woman);
    println!("{}", em.to_string());
}
```

```
# 申明子模块，rust根据约定采用固定的文件命令方式，建议采用第一种，可读性更强
# src/ffmpeg_encoder/mod.rs
# src/ffmpeg_encoder.rs


pub mod book;
pub mod people;

pub struct Shop {}

impl Shop {
    pub fn to_string() -> String {
        return String::from("Shop to_string");
    }
}
```

```
# src/ffmpeg_encoder/book.rs

pub struct Book;

impl Book {
    pub(crate) fn new() -> Book {
        Book {}
    }

    pub(crate) fn to_string(&self) -> String {
```

```
        return String::from("Book to_string");
    }
}
```

```
# 编写代码文件
# src/ffmpeg_encoder/people.rs


#[derive(Debug)]
pub struct Employ {
    name: String,
    age: u32,
    sex: Sex,
}

impl Employ {
    pub fn new(name: String, age: u32, sex: Sex) -> Employ {
        return Employ {
            name: name,
            age: age,
            sex: sex,
        };
    }
    // 第一个参数self为实例本身
    pub fn to_string(&self) -> String {
        return self.name.to_string();
    }
}
#[derive(Debug)]
pub enum Sex {
    Main = 0,
    Woman = 1,
}
```

版本号 #5
由 董列涛 创建于 26 十二月 2023 10:33:16
由 董列涛 更新于 27 十二月 2023 02:33:45