02-echor

目标：实现简单版的echo命令

实现

使用 `clap` 库：

在 `Cargo.toml` 文件中添加依赖：

C#

```
[dependencies]
clap = "2.33"
```

- `ArgMatches::values_of`
  - Returns Option<Values>
- `ArgMatches::values_of_lossy`
  - Returns Option<Vec<String>>

- Option

Rust

```
pub enum Option<T> {
    None,
    Some(T),
}
```

- Values
  - An iterator for getting multiple values out of an argument.
- The `Option::unwrap` function will take the value out of `Some<T>` to get at the payload `T`.
  - 如果是 None 就会 panic

- `main.rs`

Rust

```rust
use clap::{App, Arg};

fn main() {
    let matches = App::new("echor")
        .version("0.1.0")
        .author("liujianhao")
        .about("Rust echo")
        .arg(
            Arg::with_name("text")
                .value_name("TEXT")
                .help("Input text")
                .required(true)
                .min_values(1),
        )
        .arg(
            Arg::with_name("omit_newline")
                .help("Do not print newline")
                .takes_value(false)
                .short("n"),
        )
        .get_matches();

    let text = matches.values_of_lossy("text").unwrap();
    let omit_newline = matches.is_present("omit_newline");
    let ending = if omit_newline { "" } else { "\n" };
    print!("{}{}", text.join(" "), ending);

    // println!("{:#?}", matches);
}
```

测试

更新测试的依赖：

- `Cargo.toml`

Makefile

```
[dev-dependencies]
assert_cmd = "1"
predicates = "1"
```

- `fs::read_to_string`
  - 返回 `Result`
- Use `?` instead of `Result::unwrap` to unpack an `Ok` value or propagate an `Err`.
  - 一个语法糖，要么正常返回Ok里的内容，要么直接返回报错