30天拿下Rust之深入Cargo

> 如果想阅读最新的文章，或者有技术问题需要交流和沟通，可搜索并公众号“希望睿智”。

概述

在Rust生态系统中，Cargo扮演着至关重要的角色，它是官方的构建系统和包管理器。Cargo简化了项目的构建过程，提供了依赖项管理，以及一系列方便的工作流程工具，极大提升了开发效率和协作体验。

新建项目

新建Rust项目时，首先使用cargo new命令：cargo new my_project。这将在当前目录下生成一个名为my_project的目录，其中包括源码文件夹src、项目配置文件Cargo.toml等。Cargo.toml是Rust项目的核心配置文件，由TOML格式编写，用于描述项目的基本信息、依赖项以及其他项目特定的配置。

```
[package]
name = ”hello_rust”
version = ”0.1.0”
edition = ”2021”

# See more keys and their definitions at https://doc.rust-lang.org/cargo/reference/manifest.html
```

添加依赖

要在项目中添加依赖，只需在Cargo.toml中指定库名和版本即可。执行cargo build或cargo run命令后，Cargo会自动解析依赖关系，下载所需的依赖包，并编译项目的源代码。构建完成后，可以在target/debug或target/release目录下找到生成的可执行文件或库文件。

```
[dependencies]
rand = ”0.8” # 添加随机数生成库rand，版本号为0.8
clap = ”3.0”

```

添加测试

Cargo内置了测试框架，我们可以编写测试代码来验证项目的正确性。在Rust项目中，测试文件通常放在tests目录下，并以_test.rs为后缀。比如：可以创建一个名为project_test.rs的测试文件，并编写以下测试代码。

```
fn add(a: i32, b: i32) -> i32 {
    return a + b;
}

#[test]
fn test_add() {
    assert_eq!(add(2, 3), 5);
    assert_eq!(add(-1, 1), 0);
}

```

在Rust中，#[test]是一个属性宏，用于标识一个函数为测试函数。当运行cargo test命令时，Rust会自动执行所有标记了#[test]属性的函数，用于验证代码的正确性。

工作空间

在Rust中，工作空间是一种组织多个相关Cargo项目的结构。工作空间允许我们在一个顶级的Cargo.toml文件下管理多个库和可执行程序项目，便于共享依赖项、统一构建配置以及更好地组织项目结构。

要创建一个Rust工作空间，首先在根目录下创建一个顶级的Cargo.toml文件，并在其中加入[workspace]配置节。

```
# Cargo.toml (顶级工作空间配置文件)
[workspace]
members = [
    ”crate1”,
    ”crate2”,
    ”bin1”,
]
```

在上述配置中：[workspace]表明这是一个工作空间，members列表包含了工作空间中的成员项目，它们是相对于此顶级Cargo.toml文件的路径。每个成员项目（比如：crate1、crate2、bin1）都有自己的Cargo.toml文件，可以独立地定义自己的依赖项和构建目标。然而，共享的依赖项和配置将从顶级工作空间的Cargo.toml继承。

以crate1项目为例，其结构可能如下。

```
root_workspace/
├── Cargo.toml (顶级工作空间配置)
├── crate1/
│   ├── Cargo.toml (crate1的配置)
│   └── src/
│       └── lib.rs
├── crate2/
│   ├── Cargo.toml (crate2的配置)
│   └── src/
│       └── lib.rs
└── bin1/
    ├── Cargo.toml (bin1的配置，一个可执行项目)
    └── src/
        └── main.rs
```

在这样的工作空间中，我们可以通过cargo test、cargo build或cargo run等命令在顶层目录执行操作。Cargo将会遍历所有成员项目，并执行相应的构建或测试任务。

文档生成

在Rust中，用于生成文档的标准工具是rustdoc，它是Rust编译器的一部分。rustdoc不仅可以提取代码中的注释来生成API文档，还支持编写Markdown和一些特殊的Rustdoc标记来丰富文档内容。

要为Rust项目生成文档，可按照以下步骤操作。

1、在Rust源代码中，使用///开始的三斜杠注释来编写函数、模块、结构体、枚举等的文档注释。

```
/// This is an example of a library function.
///
/// # Example
///
/// ```
/// use my_crate::print_text;
///
/// print_text(”World”);
/// ```
pub fn print_text(text: &str) {
    println!(”{}”, format!(”{} World”, text));
}

fn main() {
    print_text(”Hello”);
}
```

2、打开终端，导航到项目的根目录（包含Cargo.toml的地方），然后运行：cargo doc。这个命令会编译项目并为所有公共接口生成文档，输出结果默认位于target/doc目录下。打开target/doc/<crate_name>/index.html文件，即可查看生成的文档。

3、如果要在本地启动一个Web服务器实时预览文档，可在生成文档后执行：cargo doc --open，这样rustdoc将自动在浏览器中打开文档索引页。

总结

通过以上的介绍，我们可以看到Cargo在Rust项目中的重要作用。它不仅简化了构建和分发过程，还提供了丰富的功能和灵活的扩展性。无论是初学者还是经验丰富的开发者，都可以通过Cargo来更加高效地创建、测试、文档化和分享自己的Rust项目。

原文链接: https://juejin.cn/post/7372094258792628278