第 3 章

CHAPTER 3

Rocket 生命周期

在本章中读者将认识到 Rocket 框架的生命周期，需要注意，这里的生命周期和 Rust 中的生命周期并不是同一个概念，Rocket 框架中的生命周期指的其实是 Web 请求进行处理的周期，其目的是可以更好地管理 Web 应用程序的不同部分。通过将 Web 请求分解为不同的阶段，可以更容易地识别和解决潜在的问题。这种方法还可以帮助开发人员更好地理解 Web 应用程序的工作原理，并使他们能够更轻松地维护和扩展应用程序。

3.1 Rocket 生命周期解析

本节介绍 Rocket 框架中每个请求的生命周期，Rocket 框架的主要任务是侦听传入的 Web 请求，将请求分发给应用程序代码，并向客户端返回响应。我们将从请求到响应的过程称为“生命周期”。可以将生命周期总结为以下 4 个主要步骤。

（1）路由：Rocket 框架将传入的 HTTP 请求解析为代码间接操作的本机结构，也就是将 HTTP 请求字符串转换为 Rust 的结构体。Rocket 框架通过匹配应用程序中声明的路由属性来确定要调用哪个请求处理程序。

（2）验证：Rocket 框架根据匹配路由中存在的类型和守卫来验证传入请求。如果验证失败，则 Rocket 的请求守卫会将请求发送到下一个匹配路由或调用错误处理程序。这个概念取自拦截器。

（3）加工：与路由关联的请求处理程序使用经过验证的参数调用。这是应用程序的主要业务逻辑。这里的真实逻辑是由开发者所编写的，用于针对请求进行一系列的特殊处理。

（4）响应：经过响应业务处理后，Rocket 会将结果包装为 Response 进行返回。Rocket 框架会生成适当的 HTTP 响应并将其发送到客户端。这就完成了生命周期。Rocket 继续侦听请求，为每个传入的请求重新启动生命周期。

请求的生命周期如图 3-1 所示。

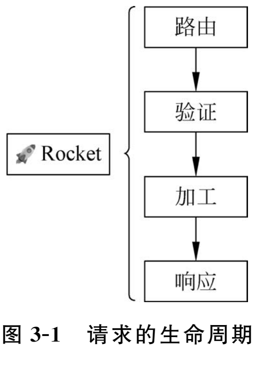

图 3-1 请求的生命周期

3.2 从请求到响应的详细流程

本节将介绍从客户端请求到服务器端处理，最后响应回到客户端的详细流程。以 http://127.0.0.1:8080/api/hello 为例，首先前端进行请求，请求会被 API 框架接受，例如

将 Axios 框架封装为 HTTP Request 传输到服务器端，服务器端由 Rocket 框架接受请求并开启生命周期，Rocket 框架根据转换后得到的请求结构体提取路由信息进行验证，若出现错误或并没有匹配到相关路由，则进行相关的路由处理，若成功匹配，则根据程序的业务逻辑进行相关的加工处理，得到返回值，再由 Rocket 框架对返回值进行包装，然后转换为 HTTP Response 并返给前端，如图 3-2 所示。

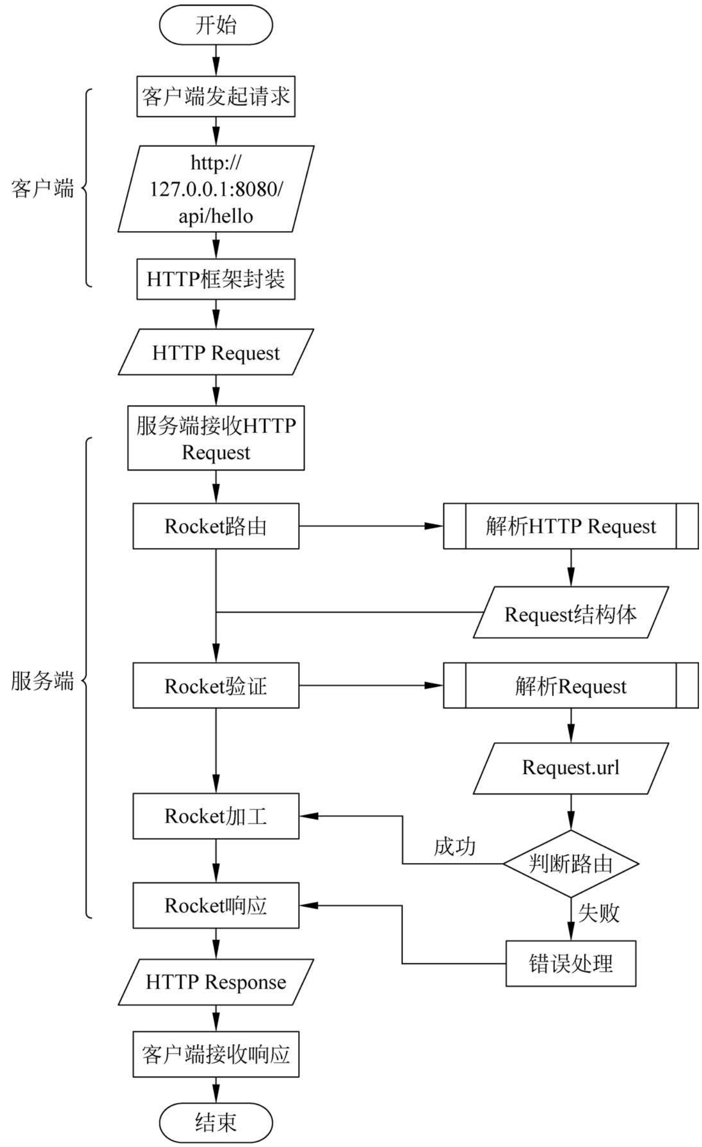

图 3-2 请求到响应的流程