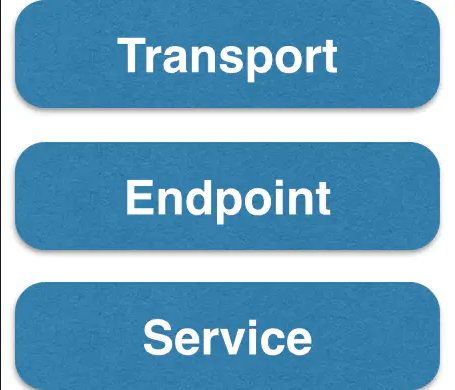
**Go kit**

1. **什么是Go kit**

Go kit是用于在Go中构建微服务（或优雅的整体式）的编程工具包。解决了分布式系统和应用程序体系结构中的常见问题，因此你可以专注于交付业务价值。它自身称为toolkit，并不是framework。也就是go kit是将一系列的服务集合在一起，提供接口，从而让开发者自由组合搭建自己的微服务项目

1. **Go kit 的架构**



Go kit的架构如上图所示，分为三层结构：Transport层，Endpoint层，Service层。

1. Transport层

Transport层主要负责与传输协议HTTP，GRPC，THRIFT等相关的逻辑。当你构建基于微服务的分布式系统时，服务通常使用HTTP或gRPC等具体传输或使用NATS等pub/sub系统相互通信。Go套件中的传输层绑定到具体运输。Go套件支持使用HTTP，gRPC，NATS，AMQP和Thrift提供服务的各种传输。由于Go kit服务仅专注于实现业务逻辑，并且不了解具体传输，因此你可以为同一服务提供多个传输。例如，可以使用HTTP和gRPC公开单个Go工具包服务。

2.Endpoint层

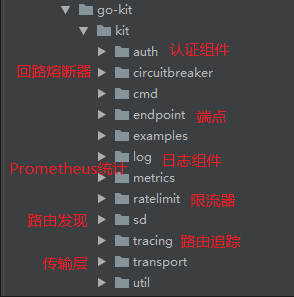
主要负责request／response格式的转换，以及公用拦截器相关的逻辑。在Go kit中，主要消息传递模式是RPC。端点表示单个RPC方法。Go工具包服务中的每个服务方法都转换为端点，以便在服务器和客户端之间进行RPC样式通信。每个端点使用传输层通过使用HTTP或gRPC等具体传输将服务方法公开给外部世界。可以使用多个传输来公开单个端点。

3.Service层

则专注于业务逻辑。Go kit服务被建模为接口。服务中的业务逻辑包含核心业务逻辑，它不应具有端点或HTTP或gRPC等具体传输的任何知识，或者请求和响应消息类型的编码和解码。这将鼓励你遵循基于Go套件的服务的干净架构。每种服务方法都通过使用适配器转换为端点，并使用具体传输进行公开。由于结构简洁，可以使用多个传输来公开单个Go工具包服务。

所有的具体方法写在这里，可以理解为单体web框架中的控制器部分

4.其他



Go kit除了经典的分层架构外，还在endpoint层提供了很多公用的拦截器，如log，metric，tracing，circuitbreaker，rate-limiter等，来保障业务系统的可用性。它们在设计上有一个共同特点：都是同传输协议无关的。在gin，beego等http框架中，这些拦截器同传输协议是紧紧耦合在一起的，如果，此时我需要将某些HTTP接口改造成GRPC协议的接口，那么这些拦截器我还得再基于grpc再实现一遍，设计上存在一定的冗余。因此，借助go kit这套工具集，我们就能很好的对transport协议，middleware进行扩展，且不会影响到业务本身的设计。