

# hydra

hydra 是基于 go 语言和众多开源项目实现的分布式微服务框架

hydra[ˈhaɪdrə]致力于提供统一的，丰富功能的后端开发框架，降低后端开发的复杂性，提高开发效率。目前已支持的服务类型有：http api 服务，rpc 服务，websocket 服务, mqc 消息消费服务，cron 定时任务服务, web 服务，静态文件服务。

## 特性

- 后端一体化框架, 支持 6+服务器类型
- 微服务的基础设施, 服务注册发现，熔断降级，集成监控与统一配置管理
- 多集群模式支持，对等，主备，分片等
- 丰富的后端库支持 redis,memcache,activieMQ,mqtt,influxdb,mysql,oracle,elasticsearch,jwt 等等
- 跨平台支持(linux,macOS 10.9+,windows 7+)
- 20+线上项目实践经验
- 全 golang 原生实现

## 示例

### 1. 编写代码

新建文件夹 hello ,并添加 main.go 文件,输入以下代码:

```

package main

import (
    "github.com/micro-plat/hydra/context"
    "github.com/micro-plat/hydra/component"
    "github.com/micro-plat/hydra/hydra"
)

func main() {
    app := hydra.NewApp(
        hydra.WithPlatName("myplat"), //平台名称
        hydra.WithSystemName("demo"), //系统名称
        hydra.WithClusterName("test"), //集群名称
        hydra.WithServerTypes("api"), //服务器类型为http api
        hydra.WithRegistry("fs://../"), //使用本地文件系统作为注册中心
        hydra.WithDebug())

    app.API("/hello", hello)
    app.Start()
}

func hello(ctx *context.Context) (r interface{}) {
    return "hello world"
}

```

## 2. 编译生成服务

```
go install hello
```

## 3. 安装本地服务和生成注册中心配置

```
hello install
```

## 4. 运行服务

```
./hello run
```

## 5. 测试服务

```
curl http://localhost:8090/hello
```

```
{"data":"hello world"}
```

hydra 微服务开发规范

构建 API 服务一

构建 API 服务二

构建消息消费服务（MQC）

构建任务调度服务（CRON）

构建远程调用服务(RPC)

构建静态文件服务

构建 webscoket 服务(WS)