Config接口规范

```
// Config 结构体
type Config struct {}

// Config 操作接口
type IConfig interface{
  LoadConfig() (*Config, error)
  SaveConfig(config *Config) error
}
```

Server接口

- 控制内部模块依赖关系

```
type Server interface {
  GetDB() DB
  GetLogger() Logger
  GetAuth() Auth
  GetConfig() Config
}
```

DB接口规范

- 对数据库的所有操作都定义为接口
- 默认提供一种特定实现

- 模型对应的接口命名方式

```
规则：<模型> -> I<模型>
例如：
  # 模型名称 -> 接口名称
  User -> IUser
  Dog -> IDog
```

- 统一的数据库接口名称：Repo

```
type IUser interface {
}

type IDog interface {
}

type Repo {
  IUser
  IDog
}
```

Auth接口规范

- 用户登录
- 用户操作权限判断
- Token发放

- Token检验

```
type Auth interface {
  Login(ctx context.Context) (user *User, token string, error)
  GetUser(ctx context.Context) (user *User, error)
  CanDo(ctx context.Context) (interface{}, error) // 返回操作的实例
}
```

任务调度接口规范

- 任务执行

```
type Scheduler interface {
  ExecTask(ctx context.Context, task *Task) error
}
```

Plugin接口规范

```
type Plugin interface {
  Exec(ctx context.Context, task *Task) error
  Result(ctx context.Context, task *Task) (interface{}, error)
  Error(ctx context.Context, task *Task) error
}
```

Log接口规范

```
type Logger interface {
    SetField(name string, value interface{})
    Debug(v ...interface{})
    Debugf(format string, v ...interface{})
    Info(v ...interface{})
    Infof(format string, v ...interface{})
    Warn(v ...interface{})
    Warnf(format string, v ...interface{})
    Error(v ...interface{})
    Errorf(format string, v ...interface{})
}
```