# Context:

Contextcontext.Context**IO**Context

GoHTTP/RPC""ContextContextContext

Context



model

```
const (
        //
        ContextKey = "ContextKey"
)

//
type Context struct {
        Session *ghttp.Session // Session
        User    *ContextUser   //
        Data    g.Map          // KV
}

//
type ContextUser struct {
        Id       uint   // ID
        Passport string //
        Nickname string //
        Avatar   string //
}
```

1. model.ContextKeycontext.Contextcontext.Context/
2. model.ContextSessionSessionGoFrameHTTPSession
3. model.ContextUsernil
4. model.ContextDataKVcontext.Contextmodel.ContextData

service

```
//
var Context = new(contextService)

type contextService struct{}

//
func (s *contextService) Init(r *ghttp.Request, customCtx *model.Context) {
        r.SetCtxVar(model.ContextKey, customCtx)
}

// nil
func (s *contextService) Get(ctx context.Context) *model.Context {
        value := ctx.Value(model.ContextKey)
        if value == nil {
                return nil
        }
        if localCtx, ok := value.(*model.Context); ok {
                return localCtx
        }
        return nil
}

//
func (s *contextService) SetUser(ctx context.Context, ctxUser *model.
ContextUser) {
        s.Get(ctx).User = ctxUser
}
```

HTTPGoFrameGRPCservicemiddleware

```
//
func (s *middlewareService) Ctx(r *ghttp.Request) {
        //
        customCtx := &model.Context{
                Session: r.Session,
                Data:    make(g.Map),
        }
        service.Context.Init(r, customCtx)
        if userEntity := Session.GetUser(r.Context()); userEntity != nil {
                customCtx.User = &model.ContextUser{
                        Id:       userEntity.Id,
                        Passport: userEntity.Passport,
                        Nickname: userEntity.Nickname,
                        Avatar:   userEntity.Avatar,
                }
        }
        //
        r.Assigns(g.Map{
                "Context": customCtx,
        })
        //
        r.Middleware.Next()
}
```

context.Context*model.Context

Session*model.Context

context.Contextservice

```
//
func (s *userService) Login(ctx context.Context, loginReq *define.
UserServiceLoginReq) error {
    ...
}

//
func (s *contentService) GetList(ctx context.Context, r *define.
ContentServiceGetListReq) (*define.ContentServiceGetListRes, error) {
    ...
}

//
func (s *replyService) Create(ctx context.Context, r *define.
ReplyServiceCreateReq) error {
    ...
}
```

errorerror

## Context

servicecontext.Contextcontext.ContextGoFrameHTTPr.Context()GRPCpbcontext.
Context

```
service.Context.Get(ctx)
```

## Key-Value

/key-value

```
//
service.Context.Get(ctx).Data[key] = value

...

//
service.Context.Get(ctx).Data[key]
```

1.
2. ctx