beanpay

提供钱包帐户的创建、查询，余额加款、扣款、退款、查询功能

提供服务包创建、查询，数量增加、减少、撤销、查询功能

√ 支持钱包余额的加、扣、退

√ 支持服务包数量的增、减、撤

√ 独立服务运行，调用 api,rpc 服务集成

√ 集成服务运行，使用源代码直接集成

√ 高并发支持

√ 基于 hydra 构建

接口开发规范

一、独立服务

独立启动apiserver对外提供api,rpc服务

1、启动apiserver

- 编译apiserver服务

```
 ~/work/bin$ go install -tags "prod" github.com/micro-
plat/beanpay/apiserver #mysql

 或

 ~/work/bin$ go install -tags "prod oracle" github.com/micro-
plat/beanpay/apiserver #oracle
```

- 安装服务, 根据向导设置数据库连接串, 生成数据库表结构

```
apiserver install -r zk://192.168.106.18 -c t
```

- 启动服务

```
apiserver run -r zk://192.168.106.18 -c t
```

2、测试服务

- 传入用户编号、名称创建钱包帐户。返回创建好的帐户信息

```
~/work/bin$ curl "http://192.168.4.121:9090/account/create?
ident=beanpay&group=up&eid=colin&name=colin"

{
    "account_id": 86000,
    "account_name": "colin",
    "balance": 0,
    "credit": 0
}
```

- 传入用户编号或商户编号,外部交易编号(幂等判断)，加款金额进行钱包加款。返回加款记录信息

```
~/work/bin$ curl "http://192.168.4.121:9090/account/balance/add?
ident=beanpay&group=up&eid=colin&trade_no=86009981&amount=10000"
{
    "account_id": 86000,
    "record_id": 8010,
    "trade_no": "86009981",
    "change_type": 1,
    "amount": 10000,
    "balance": 10000,
    "create_time": "20191114140842"
}
```

- 传入用户编号或商户编号,外部交易编号(幂等判断)，加款金额进行钱包扣款。返回扣款记录信息

```
~/work/bin$ curl "http://192.168.4.121:9090/account/balance/deduct?
ident=beanpay&group=up&eid=colin&trade_no=8970876&trade_type=1&amount=200"
{
    "account_id": 86000,
    "record_id": 8018,
    "trade_no": "8970876",
    "change_type": 3,
    "amount": -200,
    "balance": 9500,
    "create_time": "20191114143240"
}
```

- 传入用户编号、外部扣款交易编号(幂等判断)，退款金额进行钱包退款，退款金额不能大于扣款金额，同一笔扣款只允许一次退款操作。返回退款记录信息

```
~/work/bin$ curl "http://192.168.4.121:9090/account/balance/refund?
ident=beanpay&group=up&eid=colin&trade_no=8970876&deduct_no=123123&trade_ty
pe=1&amount=200"
```

```
{
    "account_id": 86000,
    "record_id": 8019,
    "trade_no": "89708761",
    "change_type": 4,
    "amount": 200,
    "balance": 9700,
    "create_time": "20191114144059"
}
```

其它接口，参考开发规范

二、代码集成

将数据表创建到业务系统，或独立库，业务系统使用本地代码，直接进行帐户、服务包操作

1、创建数据库

使用 beanpay 创建数据库，支持 mysql, oracle

- 编译 beanpay

```
 ~/work/bin$ go install github.com/micro-plat/beanpay #mysql

 或

 ~/work/bin$ go install -tags "oracle" github.com/micro-plat/beanpay #
oracle
```

- 创建数据表

`beanpay [注册中心地址] [平台名称]` 即可将 `beanpay` 需要的表创建到`/平台/var/db/db` 配置的数据库中。

```
~/work/bin$ beanpay  mysql oms:123456@oms/192.168.0.36 #数据库类型 [用户名]:
[密码]@[数据库名]/数据库ip

或

~/work/bin$ beanpay oracle oms:123456@orcl136 #数据库类型 [用户名]:[密码]@[tns
名称]
```

2、编码

- 创建帐户

```
//根据ident系统标识与group用户
bp :=
beanpay.GetAccount(ctx.Request.GetString("ident"),ctx.Request.GetString("gr
oup"))
        account, err := bp.CreateAccount(ctx,
                ctx.Request.GetString("eid"),
                ctx.Request.GetString("name"))
return account
```

- 帐户加款

```
bp:=beanpay.GetAccount(ctx.Request.GetString("ident"),ctx.Request.GetString
("group"))
record, err := bp.AddAmount(ctx,
ctx.Request.GetString("eid"),
ctx.Request.GetString("trade_no"),
ctx.Request.GetInt("amount"))
if err != nil {
    return err
}
return record
```

- 帐户扣款

```
bp:=beanpay.GetAccount(ctx.Request.GetString("ident"),ctx.Request.GetString
("group"))
record, err := bp.DeductAmount(ctx,
ctx.Request.GetString("eid"),
ctx.Request.GetString("trade_no"),
ctx.Request.GetInt("trade_type"),
ctx.Request.GetInt("amount"))
if err != nil {
    return err
}
return record
```

> 其它操作请查看http://github.com/micro-plat/beanpay/beanpay/service.go