Atitit 提升代码可读性 fix 补充

目录

[1.1. 多使用static import导入机制。。注意函数名不要重复了。。 1](#_Toc25006)

[1.2. 代码通用化组件模式 文档众多 1](#_Toc30388)

[2. 业务脚本dsl模式 1](#_Toc5693)

[3. 业务代码不要分散在多处，尽可能简化调用链 1](#_Toc3009)

[3.1. 简化层侧，是否需要service dao层等 1](#_Toc7948)

[3.2. 内部类的也是不错的模式。。匿名类等lambda 1](#_Toc31003)

## 多使用static import导入机制。。注意函数名不要重复了。。

## 代码通用化组件模式 文档众多

# 业务脚本dsl模式

# 业务代码不要分散在多处，尽可能简化调用链

## 简化层侧，是否需要service dao层等

脚本比较重要，ide不能自动调整麻烦的。。

## 内部类的也是不错的模式。。匿名类等lambda

**public** **class** LoadBalanceTest {

**public** **class** KeyServerRule **extends** AbstractLoadBalancerRule{

@Override

**public** Server choose(Object key) {

List<Server> allServerList =**this**.getLoadBalancer().getAllServers();

**for** (Server server : allServerList) {

// System.out.println(server.getZone());

**if** (server.getZone().toString().toLowerCase().equals(key.toString().toLowerCase()))

**return** server;

}

**return** **null**;

}

@Override

**public** **void** initWithNiwsConfig(IClientConfig clientConfig) {

// **TODO** Auto-generated method stub

}

}

**public** **static** **void** main(String[] args) {

// extracted();

// 带key的实现机制

//LoadbalanceImp blb = new LoadBalanceTest().new LoadbalanceImp();

// DynamicServerListLoadBalancer blb=new DynamicServerListLoadBalancer();

BaseLoadBalancer blb=**new** BaseLoadBalancer();

Server svr1 = **new** Server("svr1");

svr1.setZone("key1");

Server svr2 = **new** Server("svr2");

svr2.setZone("key2");

List li = **new** ArrayList<>();

li.add(svr2);

li.add(svr1);

blb.addServer(svr1);

blb.addServer(svr2);

blb.setRule( **new** LoadBalanceTest(). **new** KeyServerRule());

// blb.addServers(li);

// System.out.println(blb.choose("key1"));

System.***out***.println(blb.getAllServers());

System.***out***.println(blb.chooseServer("key1"));

}