使用说明

# 1.服务上已经部署的Demo



描述：demon里面包含了：appplatform和appserver的服务代码，以及对应的nodejs测试客户端代码。服务已经部署到dev（ip：192.168.145.30）机器，可以根据客户端代码进行调用测试。

# 2.Nodejs客户端调用Appplatform

## 1）编写appplatform-IDL文件：

IDL开发文档参考CF:http://cf.jd.com/pages/viewpage.action?pageId=75273751

IDL示例：ao\_applatform\_test.java

package basebiz.ao.node;

import com.paipai.lang.\*;

import com.paipai.lang.uint8\_t;

import com.paipai.lang.uint16\_t;

import com.paipai.lang.uint32\_t;

import com.paipai.lang.uint64\_t;

import com.paipai.lang.uint32\_t;

import com.paipai.util.annotation.ApiProtocol;

import com.paipai.util.annotation.Field;

import com.paipai.util.annotation.HeadApiProtocol;

import com.paipai.util.annotation.Member;

import java.util.\*;

@HeadApiProtocol(cPlusNamespace = "basebiz::ao::node", serviceType="ao", needInit=true)

public class compliteTest{

@Member(desc = "达人详情Po", cPlusNamespace="basebiz::ao::node")

public class Data{

@Field(desc = "version")

uint32\_t version;

@Field(desc = "达人ID")

uint64\_t darenId;

@Field(desc = "用户类型")

uint32\_t userType;

}

@ApiProtocol(cmdid = "0xfe471801L", desc = "通过用户信息获取达人信息")

public class GetNodeTest {

@ApiProtocol(cmdid = "0xfe471801L", desc = "通过用户信息获取达人信息请求")

class Req {

@Field(desc = "来源")

String source;

@Field(desc = "用户ID")

String userId;

@Field(desc = "用户类型")

uint32\_t userType;

Vector<Data> rlist;

}

@ApiProtocol(cmdid = "0xfe478801L", desc = "通过用户信息获取达人信息响应")

class Resp {

@Field(desc = "错误信息")

String errMsg;

@Field(desc = "达人信息")

String outResult;

@Field(desc = "达人信息")

Vector<Data> list;

Data data;

}

}

}

## 2）利用auto\_gen.sh 生成c++&nodejs的代码

格式为auto\_gen.sh IDL文件名 目标文件夹 参数（c utf-8 cpp,nodejs）

例：

comm/app\_platform/auto\_gen/auto\_gen.sh basebiz/ao/node/ao\_applatform\_test.java ~/targetDir/ao\_test/ c utf-8 cpp,nodejs

a)生成nodejs代码结构为

ao\_applatform\_test/

├── config.js

├── handler.js

├── stub4node.js

└── xxo.js

b)公共模块代码为:

platform/

├── byte\_stream.js

├── c2c\_pkg\_head.js

├── lang\_util.js

├── package.json

├── services.js

└── web\_stub\_cntl.js

## 3）编写c++业务逻辑代码，部署ao服务：

参照《App Platform 入门到精通》 CF链接为：

http://cf.jd.com/pages/viewpage.action?pageId=71807368&preview=/71807368/75273145/APP%20Platform%E5%85%A5%E9%97%A8%E5%88%B0%E7%B2%BE%E9%80%9A.doc

## 4）编写nodejs的客户端代码app.js

依赖步骤2生成的nodejs代码及platform代码，调用方式例如：

BossStubCtrl = require('platform/web\_stub\_cntl').WebStubCntl,

appsvrtest= require('ao\_applatform\_test/stub4node');

mydata = require('ao\_applatform\_test/xxo');

var bossStub = new BossStubCtrl();

var data = new mydata.Data();

var arry = [];

data.dwVersion = 2018; // uint32\_t

data.ddwDarenId = 10; // uint64\_t

data.dwUserType = 10; // uint32\_t

arry[0] = data;

req = new appsvrtest.GetNodeTestReq();

req.dwVersion=20170119;

req.ddwId=8017;

req.rlist.setType('ao\_applatform\_test/xxo.Data');

req.rlist.setValue(arry);

console.log(req.rlist);

resp = new appsvrtest.GetNodeTestResp();

bossStub.setCallerName('testapp.js');

bossStub.setPeerIPPort('192.168.145.31', 53101);//此处可通过配置中心直接获取，自测时可指定ip&端口

bossStub.invoke(req, resp, function(errmsg, result) {

console.log("appplatform invoke over: errmsg: %s, resp:iret=%d,resp:errMsg=%s,resp: outResult=%s!",

errmsg,resp.result, resp.errMsg,resp.outResult);

console.log("--------------appplatform test---------------");

}, 2000);

//注意此处的超时时间可不设置，默认为2000ms。

## 5）测试客户端app.js，查看调用是否成功。

node app.js

示例如demo所示”test\_appplatform.js”

## 6）nginx配置客户端代码app.js

略

# 3.Nodejs客户端调用AppSvr

## 1）编写appsvr-c++版本服务及其协议头文件

命令格式：auto\_gen.sh 数据结构.h 协议.h /目标文件夹

通过auto\_gen.sh与xxstruch.h xxprotocol.h生成nodejs版本协议文件：

appsvrtest/

├── config.js

├── handler.js

├── stub4node.js

└── xxo.js

## 2）利用autogen生成nodejs业务代码

1.给数据结构和协议头文件打标签。

定义标签

#define CLASS\_BEGIN

#define CLASS\_END

#define DATA\_BEGIN

#define DATA\_END

#define FUNCTION\_NAME

1）在每一个类的上一行，用上面定义的空宏标识class的开始，在class结束后也标上类结束宏。

CLASS\_BEGIN

class test｛

｝；

CLASS\_END

2) 数据类需要注意public private protected 需要单独加标识.

DATA\_BEGIN

private：

int a；

bool b；

DATA\_EDN

DATA\_BEGIN

public：

int c；

std：：vector<int> d；

DATA\_END

3)注意！序列化，构造函数Dump 函数不需要加标识，其他自定义方法需要加FUNCTION\_NAME宏定义

FUNCTION\_BEGIN

test（）｛｝；//构造函数

template<class AR>

AR & serialize(AR & ar) //序列化

FUNCTION\_NAME int getNum（）｛return 1｝；

FUNCTION\_END

2.生成nodejs代码。

命令格式：

./autogen.sh 目标数据结构头文件 目标协议定义头文件 输出路径 输出的名称

例：

./autogen.sh ./testProtocal/app\_circle\_struct.h ./testProtocal/app\_circle\_protocol.h ./au togen\_nodejs/testoutput nodetest

## 3）编写nodejs的客户端代码app.js

在auto\_gen.sh 生成的node js代码的基础上，添加自己的业务数据，例：

BossStubCtrl = require('c2cboss/appsvr\_stub\_cntl').BossStubCtrl,

ao\_appsvr\_test= require('ao\_appsvr\_test/stub4node');

var iAppCircleUpdateDetailStub = new BossStubCtrl();

iAppUpdateDetailReq = new ao\_appsvr\_test.CAppUpdateDetailReq();

iAppUpdateDetailReq.dwVersion=20170618;

iAppUpdateDetailReq.ddwId=2019;

var iInfo = new ao\_appsvr\_test.CMsgInfo;

iInfo.ddwTestU64 = 6486;

iInfo.idwTestI64 = -6486;

iInfo.ddwTestU32 = 3286;

iInfo.idwTestI32 = -3286;

iInfo.ddwTestU16 = 16;

iInfo.idwTestI16 = -16;

iInfo.ddwTestU8 = 8;

iInfo.idwTestI8 = -8;

iInfo.uAppVer = 2;

iInfo.uType = 3;

iInfo.dwSendTime = Date.parse(new Date()) / 1000;

iInfo.sTitle = "618消息标题1-----";

iInfo.sContent = "618消息标题1-----";

var vecPlatform = [];

vecPlatform[0] = 321;

vecPlatform[2] = 322;

iInfo.vecPlatform.setValue(vecPlatform);

var mapResvInfo = {};

mapResvInfo[321] = "map<321, 字符>";

mapResvInfo[322] = "map<322, 字符>";

iInfo.mapResvInfo.setValue(mapResvInfo);

var iMapMsgList = {};

iMapMsgList[iInfo.dwSendTime] = iInfo;

iAppUpdateDetailReq.iMapMsgList.setValue(iMapMsgList);

console.log(iAppUpdateDetailReq.iMapMsgList.value);

oAppUpdateDetailResp = new ao\_appsvr\_test.CAppUpdateDetailResp();

iAppCircleUpdateDetailStub.setCallerName('testapp.js');

iAppCircleUpdateDetailStub.setPeerIPPort('192.168.145.31', 52521);

iAppCircleUpdateDetailStub.invoke(iAppUpdateDetailReq, oAppUpdateDetailResp, function(err, result) {

console.log("appsvr invoke over: %d, resp:ddwId=%d, sErrMsg=%s",

err, result.ddwId, result.sErrMsg);

console.log("oMapMsgList:");

console.log(result.oMapMsgList);

console.log("--------------appserver test---------------");

}, 3000);

## 3）测试客户端app.js，查看调用是否成功。

node test\_appsvr.js (示例如demo所示)

## 4）nginx配置客户端代码app.js

略