# 技术文档

## WebService

### WebService客户端

WebService客户端调用服务器的方式有多种，这里介绍一下几种方式，是在项目中用到的（后续会补充其它方式）。

#### 方式一：

这个是在做电信项目的时候用到的。

try {

Service service = new Service();

Call call = (Call) service.createCall();

// 这个是服务器发布的WSDL的地址

String url = “http://localhost:8080/ ftms/services/MssCashBudgetSync?wsdl”;

// 设置WSDL地址

call.setTargetEndpointAddress(new URL(url));

// 下面的这几行代码需要重点记录一下。

// 作用：当WebService服务端发布的方法中，对象类型存在自定义类型

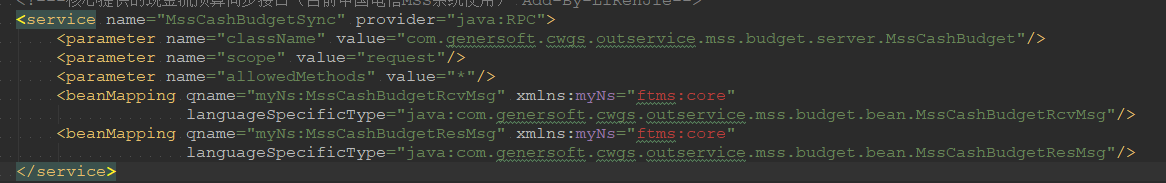
// 的时候，需要对这些类型进行序列化。这些代码的作用就是序列化的。

// 说明一下具体的参数意义。

// 多说一句服务器端发布WSDL的配置文件（这些会在服务器那边重点写）

// 配置文件名字是：server-config.wsdd

// 发布的方法的配置方法：

// 

// QName的参数1：对应上面配置中的myNs

// 参数2：对应要序列化的类名

// registerTypeMapping参数1：序列化的类名

// 参数2：QName对象

// 参数3和参数4：就是这样写

// 做完上面这些之后，就可以正常使用自定义的对象了。

QName budgetRMQname = new QName("ftms:core","MssCashBudgetRcvMsg");

call.registerTypeMapping(

MssCashBudgetRcvMsg.class,

budgetRMQname,

new org.apache.axis.encoding.ser.BeanSerializerFactory(

MssCashBudgetRcvMsg.class,

budgetRMQname),

new org.apache.axis.encoding.ser.BeanDeserializerFactory(

MssCashBudgetRcvMsg.class,

budgetRMQname));

} catch(Exception e) {

// 打印异常

}

## JS

### Dialog

js里面的dialog均不是阻塞的，所以如果想要根据dialog的返回值执行相应的操作的话，实现方式如下：

var callback = function(data) {

// data是对话框返回的值

}

var d = dialog({

// 主要参数

id: “ID”,

title:”标题”,

type:”dialog的类型,例如iframe”,

width:900,

height:600,

[url:”设置打开的页面](url:)”,

onclose:function(){

// 下面的代码是设置对话框的返回值

var retValue = this.returnValue;

if (callback) {

callback(retValue);

}

}

});

对话框的返回值是在url那个页面里返回的，设置方法：

// 获取打开这个页面的dialog

var dlg = parent.dialog.get(window);

dlg.close(/\*这里写具体的返回值\*/);

// 或者这样设置

window.returnValue=””;

### JS代码添加控件

#### 添加表格

// tbObj是表格对象<table>

// 插入一行

var dataRow = tbObj.insertRow(rowIndex);

// 新建单元格

var xh = dataRow.insertCell(0);

// 设置属性

xh.setAttribute("width", "5%");

var mc = dataRow.insertCell(1);

mc.setAttribute("width", "20%");

xh.style.color = "#4f6b72";

xh.style.border = "1px solid #C1DAD7";

xh.style.background = "#CAE8EA";

xh.style.padding = "6px 6px 6px 12px";

#### JS控件事件

##### 表格

<td ondbclick=”dbclick()”></td>

function dbclick() {

// 具体的方法实现

}

### JS的公共方法

#### 判断是否全是数字

isNaN

#### 字符串和整型的转化

parseFloat()

parseInt()

或

Number()

推荐使用Number。parseInt在转化00100的时候，会当作二进制进行转化。

转string，使用toString。

还有几个操作，参考：

<https://www.cnblogs.com/qiantuwuliang/archive/2009/05/15/1457657.html>

## JSP

内嵌Java代码：

<%

// 添加Java代码

%>

在script里面使用java表达式：

var val = <%=getInt()%>; // 这个是返回一个非字符串的值，不需要加双引号

var val=”<%=getString()%>”;// 这个是字符串值

## 数据库

SELECT \* FROM 数据库表名 别名 WHERE 别名.列名 = ？

UPDATE 数据库表名 别名 SET 别名.列名 = ？, SET 别名.列名 = ？ WHERE 别名.列名 = ？

ALTER TABLE 表名 RENAME COLUMN 现在的列名 TO 新的列名

多个字段作为主键：

CREATE TABLE 表名(

COLUMN1,

COLUMN2 PRIMARY KEY(COLUMN1, COLUMN2),

COLUMN3

);

修改主键：

ALTER TABLE 表名 WITH NOCHECK ADD

CONSTRAINT [PK\_表名] PRIMARY KEY

(

[字段名1],

[字段名2]

)

普通字段建立索引：

CREATE INDEX 索引名字 ON 表名 (字段名字);

### ORACLE

#### ORACLE基本概念

反正还是不明白。

##### 数据库名

这个很容易理解。就是存放数据的仓库，用来区分数据的内部标识。不多做解释。

##### 实例名

是我们登录时用的名字。例如我们平常在PLSQL上登录时，就是使用实例名登录。

##### ORACLE\_SID



ORACLE\_SID是操作系统的环境变量。ORACLD\_SID用于与操作系统交互，也就是说，从操作系统的角度访问实例名，必须通过ORACLE\_SID。

#### RAC集群，JDBC双机连接

jdbc:oracle:thin:@(DESCRIPTION =

(ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.11.39)(PORT = 1521))

(ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.11.40)(PORT = 1521))

(LOAD\_BALANCE = yes)(FAILOVER = ON)(CONNECT\_DATA =(SERVER = DEDICATED)(SERVICE\_NAME = ctfcdb)(FAILOVER\_MODE=(TYPE = SELECT)(METHOD = BASIC)(RETIRES = 20)(DELAY = 15))))

#### ORACLE基本操作

##### 环境变量

echo $NLS\_LANG

echo $ORACLE\_SID

查看数据库字符集编码

select userenv(‘language’) from dual;

##### 监听管理

lsnrctl status 查看监听状态

lsnrctl start 启动监听

lsnrctl stop 停止监听

##### 启动停止数据库

启动：

sqlplus / as sysdba

SQL> startup

停止：

sqlplus / as sysdba

SQL> shutdown immediate

##### 数据库参数检查

##### 创建表空间

1. 查询表空间路径

select \* from dba\_data\_files;

1. 创建表空间

create tablespace 表间名 datafile '数据文件名' size 表空间大小

例如：

create temporary tablespace FTMS\_TEMP

tempfile '路径名/ftms\_temp.dbf'

size 2G

extent management local;

create tablespace FTMS\_DATA

logging

datafile '路径名/ftms\_data01.dbf'

size 30G

autoextend on

next 500M maxsize unlimited

extent management local;

create tablespace FTMS\_IDX

datafile '路径名/ftms\_index01.dbf'

size 20G

autoextend on

next 500M maxsize unlimited

extent management local;

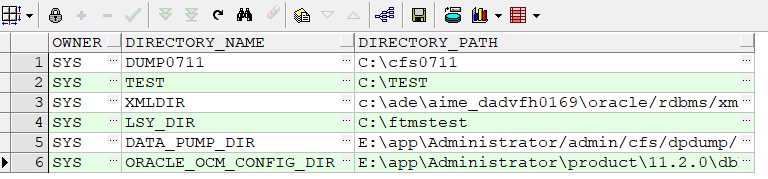
##### 备份数据库

1. 查看目录

ORACLE创建目录的目的就是指定一些文件的存放位置。比如，在利用数据泵导入导出数据的时候，定义数据导出和导入的目录。还有一些其它的用处，用到了再补充。

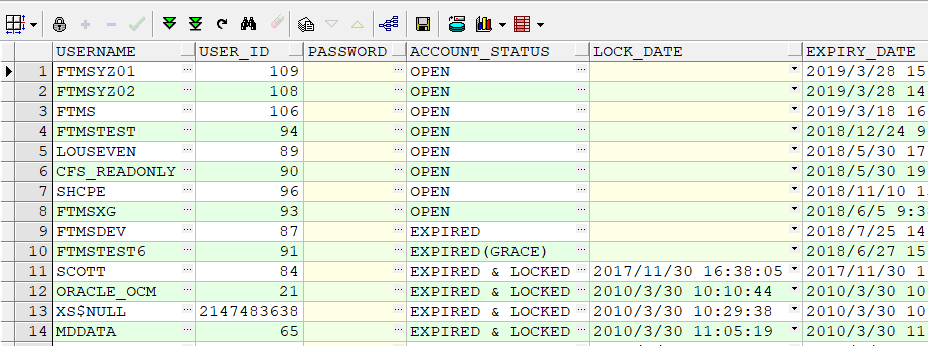
例如：这是项目中用到的一个数据库

SELECT \* FROM DBA\_DIRECTORIES;



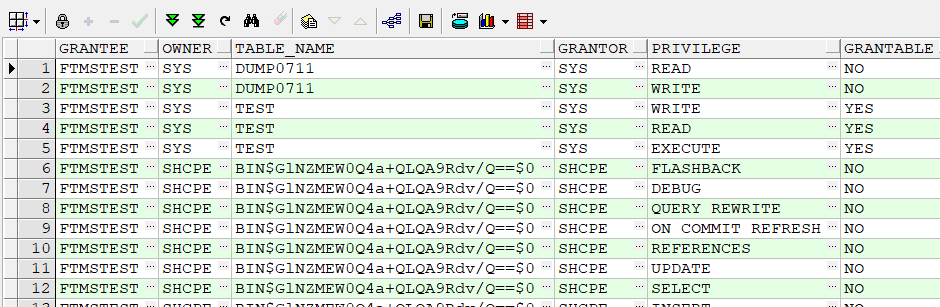
1. 查找所有数据库所有用户

SELECT \* FROM DBA\_USERS;



1. 查看用户的权限

SELECT \* FROM DBA\_TAB\_PRIVS A WHERE A.grantee = 'FTMSTEST';



1. 创建目录

CREATE OR REPLACE DIRECTORY DATA\_DUMP\_DIR AS ‘具体的路径’;

1. 执行授权

GRANT READ,WRITE ON DIRECTORY DATA\_DUMP\_DIR TO 用户名;

1. 执行备份

expdp 用户名/密码 DIRECTORY=DATA\_DUMP\_DIR DUMPFILE=备份数据库名字 LOGFILE=日志文件

1)按用户导

expdp scott/tiger@orcl schemas=scott dumpfile=expdp.dmp DIRECTORY=dpdata1;

2)并行进程parallel

expdp scott/tiger@orcl directory=dpdata1 dumpfile=scott3.dmp parallel=40 job\_name=scott3

3)按表名导

expdp scott/tiger@orcl TABLES=emp,dept dumpfile=expdp.dmp DIRECTORY=dpdata1;

4)按查询条件导

expdp scott/tiger@orcl directory=dpdata1 dumpfile=expdp.dmp Tables=emp query='WHERE deptno=20';

5)按表空间导

expdp system/manager DIRECTORY=dpdata1 DUMPFILE=tablespace.dmp TABLESPACES=temp,example;

6)导整个数据库

expdp system/manager DIRECTORY=dpdata1 DUMPFILE=full.dmp FULL=y;

##### 恢复数据库

1. 建立新数据库用户

CREATE USER 用户名

IDENTIFIED BY 密码

DEFAULT TABLESPACE 表空间目录名字(上面建立的应该是)

TEMPORARY TABLESPACE 临时表空间目录名字(应该也是上面建立的)

PROFILE DEFAULT;

1. 设置权限

// 基本权限：创建表、删除表、查询、更新

* GRANT CONNECT TO 用户名;

// 基本权限(同上)，其它的不知道

* GRANT RESOURCE TO 用户名;

// 拥有所以权限

* GRANT DBA TO 用户名;

1. 数据库恢复

是否需要进入指定目录

IMPDP 用户名/密码 DIRECTORY=DATA\_PUMP\_DIR LOGFILE=日志文件 REMAP\_SCHEMA=’导出时的用户名’:’此时的用户名’

还原数据

1)导到指定用户下

impdp scott/tiger DIRECTORY=dpdata1 DUMPFILE=expdp.dmp SCHEMAS=scott;

2)改变表的owner

impdp system/manager DIRECTORY=dpdata1 DUMPFILE=expdp.dmp TABLES=scott.dept REMAP\_SCHEMA=scott:system;

3)导入表空间

impdp system/manager DIRECTORY=dpdata1 DUMPFILE=tablespace.dmp TABLESPACES=example;

4)导入数据库

impdb system/manager DIRECTORY=dump\_dir DUMPFILE=full.dmp FULL=y;

5)追加数据

impdp system/manager DIRECTORY=dpdata1 DUMPFILE=expdp.dmp SCHEMAS=system TABLE\_EXISTS\_ACTION

##### 执行DELETE等需要提交的操作，忘了提交

select t2.username,

t2.sid,

t2.serial#,

t3.object\_name,

t2.OSUSER,

t2.MACHINE,

t2.PROGRAM,

t2.LOGON\_TIME,

t2.COMMAND,

t2.LOCKWAIT,

t2.SADDR,

t2.PADDR,

t2.TADDR,

t2.SQL\_ADDRESS,

t1.LOCKED\_MODE

from v$locked\_object t1, v$session t2, dba\_objects t3

where t1.session\_id = t2.sid

and t1.object\_id = t3.object\_id

order by t2.logon\_time;

上面查询出锁表的SID和serail#,下面kill掉

alter system kill session '139,3787';

##### 删除用户、新建用户

删除：

drop user 要删除的用户名 CASCADE;

直接删除的时候，可能会有提示“无法删除正在连接的用户”，下面的就是查找到哪些用户在连接，然后使用alter system kill(这个命令并不是真正杀死) 把他们都杀掉。

ALTER SYSTEM DISCONNECT SESSION '194,95' IMMEDIATE;（真正杀死 ）

--断开用户联接

select 'alter system kill session '''||sid||','||serial#||''';' from v$session where username='CFS\_YGZ06';

新建：

create user ftms

identified by password

default tablespace FTMS\_DATA

temporary tablespace FTMS\_TEMP

profile DEFAULT;

grant connect,resource,dba to ftms;

## WebLogic

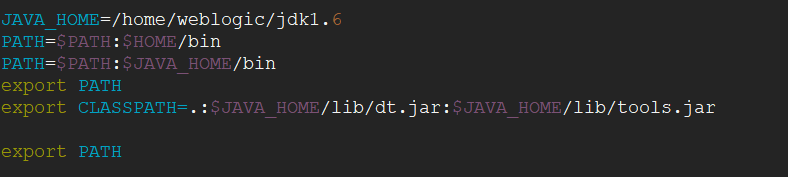
安装步骤：

## Java

### 环境变量(LINUX)

修改用户根目录下的 .bash\_profile文件

修改如下：



JAVA\_HOME=/home/weblogic/jdk1.6

PATH=$PATH:$HOME/bin

PATH=$PATH:$JAVA\_HOME/bin

export PATH

export CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

export PATH

修改完成之后，source一下这个文件。

然后java –version 看一下是否是已经修改的路径。

如果不是，那么是被系统路径的覆盖了，使用which java，看一下系统自带的路径，一般是/usr/bin/java。这个时候，java会有一个连接：java 🡪 /etc/alternatives/java,先把/etc/alternatives/java删除掉，然后把java删除掉。就可以了。

## nginx

### 配置

nginx位置：/usr/local/nginx

配置文件：nginx.conf

http{

……

upstream enetbankCluster {

# ip\_hash;

# 下面这两个是实际提供服务的地址。是我们希望nginx帮我们负载均衡的服务地# 址

server 192.168.11.11:8000 max\_fails=3 weight=1 fail\_timeout=300s;

server 192.168.11.12:8000 max\_fails=3 weight=2 fail\_timeout=300s;

}

server {

# 下面list和server\_name是配置nginx的监听地址。一般是linux服务器的地

# 址

listen 80;

server\_name 192.168.11.11;

location / {

proxy\_pass http://enetbankCluster;

proxy\_next\_upstream http\_500 http\_502 http\_503 error timeout invalid\_header;

include proxy.conf;

}

location /login {

proxy\_pass http://enetbankCluster;

proxy\_next\_upstream http\_500 http\_502 http\_503 error timeout invalid\_header;

include proxy.conf;

}

location /main {

proxy\_pass http://enetbankCluster;

proxy\_next\_upstream http\_500 http\_502 http\_503 error timeout invalid\_header;

include proxy.conf;

}

} // end server

}// end http

注意：proxy.conf这个文件，默认安装的nginx没有。

## LINUX

### 时间服务器同步设置

来自：https://www.cnblogs.com/jczhu/p/5851268.html

安装ntp服务器

1. rpm -ivh ntp-4.2.2p1-8.el5.centos.1.rpm
2. yum install -y ntp ntpdate
3. 配置

首先设置局域网内的一台机器作为时间服务，修改的配置文件：/etc/ntp.conf

修改的地方如下：

# restrict、default定义默认访问规则，nomodify禁止远程主机修改本地服务器

restrict default kod nomodify notrap nopeer noquery

#配置，notrap拒绝特殊的ntpdq捕获消息，noquery拒绝btodq/ntpdc查询

restrict -6 default kod nomodify notrap nopeer noquery

# 这里的查询是服务器本身状态的查询

restrict 127.0.0.1

restrict -6 ::1

# 这句是手动增加的，意思是指定的192.168.1.0--192.168.1.254的服务器都

#可以使用ntp服务器来同步时间。

restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap

#这句也是手动添加的，可以将局域网中的指定ip作为局域网内的ntp服务器。

server 192.168.1.117

# 这3个域名都是互联网上的ntp服务器，也还有许多其他可用的ntp服务器，能连上

# 外网时，本地会跟这几个ntp服务器上的时间保持同步。

server 0.centos.pool.ntp.org

server 1.centos.pool.ntp.org

server 2.centos.pool.ntp.org

# 当服务器与公用的时间服务器失去联系时，就是连不上互联网时，以局域网内的时

# 间服务器为客户端提供时间同步服务。

server 127.127.1.0

fudge 127.127.1.0 stratum 10

1. 启动

# 当前启动ntpd服务

/etc/init.d/ntpd start

# 下次开机自启ntpd服务

chkconfig ntpd on

1. 客户端使用

ntpdate -u 192.168.11.37 (IP就是刚才设置的时间服务器的地址)

### 用户组、用户增加删除

增加：

groupadd 组名

useradd 用户名 –g 组名 –G 次要组名 –d HOME目录 –p 密码

把用户加入组：

usermod –a –G 组名 用户名

usermod –G 组名 用户名(该用户只属于这个组)

删除：

userdel 用户名

groupdel 组名

如果删除的时候，提示该用户正在登录，需要使用：killall –u 用户名 即可

### 记录账户的文件

/etc/passwd

### 记录账户组的文件

/etc/group

### 修改用户名密码

首先使用要修改密码的用户登录linux，然后使用命令：

passwd即可。

或者使用root用户登录，然后使用命令：

passwd 用户名

### 修改文件或者文件夹的用户和用户组

chown –R 用户组:用户名 文件或者文件夹

### 系统提示Resource temporarily unavailable

1. 编辑以下文件/etc/security/limits.conf,调整以下参数即可.

oracle soft nproc 65536

oracle hard nproc 65536

oracle soft nofile 65536

oracle hard nofile 65536

oracle是对应的用户。

1. 编辑文件/etc/security/limits.d/90-nproc.conf

\* soft nproc 65536

### 统计文件和目录个数

ls -l | grep “^-” | wc -l

ls -l | grep “^d” | wc -l

## 加密签名验签

### 加密和签名的区别

私钥用来签名的，公钥用来验签的。

公钥加密私钥解密是秘送，私钥加密公钥解密是签名。

### 私钥公钥

# 会计学、银行业务相关

## 概念介绍

### 会签和抢占

会签和抢占是审批流中的两种审批模式。

会签：拥有统一岗位级别的多个用户，需要所有人审批通过才能进入下一个审批环节。

抢占：一个人审批通过即可。

### 关联方

疑问

### 存款账户

#### 对公账户

公司对公帐户分为四类：基本账户、一般账户、临时账户及专用账户。 其中： 基本账户一个公司只能开一个。 其他的 例如：一般账户，一个公司可以根据业务需要开立多个，没有数量限制。 人民银行的账户管理规定：一个企业只能开立一个基本账户，但是可以开立多个一般账户，如果另外还要开一个可以取款的账户，而且资金方有一定的专门用处，那么你可以在该行申请办理一个专用账户也是可以取款的。

#### 对私账户

对私结算账户是指存款人凭个人身份证件以自然人名称开立的银行结算帐户。

对私结算账户即个人银行结算帐户。

## 业务知识

### 银行账户管理模式

#### 收支一条线

收支一条线也称为门户账户，基本的管理原则是资金的收入和支出不分开。集团资金管理机构只关注成员单位账户的日末余额，不关注成员单位日间的收、支情况。属于较为宽松的管理模式。

收支一条线的管理模式表现在对成员单位银行账户管理上可以有两种：

一种是通过前期较为彻底的账户清理工作，只给每个成员单位保留一个账户（通常为基本户）用于日常结算和归集，这是较理想的状态。

#### 收支两条线

基本的管理原则是资金的收入和支出要分开。集团资金管理机构不光关注成员单位账户的日末余额，还关注成员单位日间的收支情况。属于较为严格的管理模式。

#### 联动账户模式

联动账户是银行提供的一种资金归集模式。这种模式的原理是成员单位的账户和集团管理机构的归集账户变动同时联动，即成员单位账户支出一笔资金，管理机构的归集账户同时支出一笔相等金额资金；成员单位账户收入一笔资金，管理机构的归集账户同时收入一笔相等金额的资金。由于一笔资金同时在两个账户上显示，故存在集团整体资金虚增、虚减一倍的情况，成员单位银行账户的余额称为“可用余额”。

#### 代理行账户模式

代理行账户模式的管理原则是成员单位在银行不开立银行账户，实际的资金收、支均通过集团资金管理机构的账户完成。在这种模式下银行账户集中管理的程度较高。

**存在的问题：**

1、由于集团内部各单位的收入都直接汇入集团资金管理机构的主账户，虽然减少了资金归集的过程，但同时也带来了一个问题，即如何将汇入集团主账户的资金准确地划分给各成员单位。

2、成员单位的支出被退回后，集团资金管理机构很难知道资金的确切归属。

3、为保证成员单位付款实名制，需要银行提供特定的服务。

4、只要集团资金管理机构主账户所在行直联接口出问题，会影响全集团的结算业务；

5、在集团推广前需要跟银行洽谈柜面的应急处理机制，银行要能提供大力支持；

6、由于有些银行要求提供代理行账号，且各集团编制账号的规则可能趋同，会出现代理行账号重复导致业务处理失败的情况。

### 记账方法

复式记账的记账规则：

借方：

资产和支出的增加，余额在借方。

负债、权益、收益的减少，余额在贷方

贷方：

资产和支出的减少，余额在借方

负债、权益、收益的增加，余额在贷方



# 财务公司

## 概念介绍

### 四个账户

#### 内部账户

财务公司因本身业务经营或内部资金核算需要而开立的账户，如为利息收入、手续费及佣金收入等科目开立的核算账户。

#### 银行账户

财务公司在外部银行开立的结算账户，即财务公司的头寸户。

#### 客户账户

成员单位在财务公司开立的结算账户。

#### 客户银行账户

成员单位在外部银行开立的结算账户，用于资金上划下拨，关联客户账户，实现客户账户的自动入账，客户银行账户在系统中主要是登记管理不做具体账务处理。