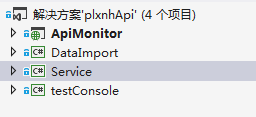
**新农合接口调用说明**

## 一、项目工程说明

开发工具：Visual Studio Utilmate 2013

,Net Framework 4.5 (核心代码实现没有使用高级特性，可以在低版本的环境使用)



（其中DataImport这个项目是 .net 4.0框架X86的设置，是为了兼容使用读取Excel的第三方驱动）

解决方案里有4个项目：

1. ApiMonitor是接口模拟，模拟厂商的接口，供Service项目生成对应的服务引用；
2. DataImport是导入配置数据进去数据库表的工具，使用excel模板；
3. Service项目是核心，是后续集成到HIS系统的类库项目，完成接口的调用和报文封装和解析，以提供dll的方式完成集成；
4. TestConsole项目可以不用管，只做测试接口使用；

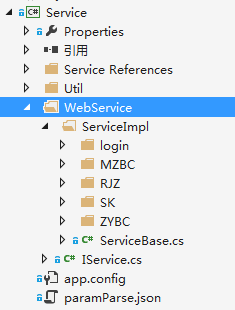
### 代码框架

根据接口文档，主要接口的调用框架如下：

1. 接口的IService主要定义执行的接口调用的关键操作；
2. 子类ServiceBase实现IService的操作，并新增了部分属性；
3. ConfigUtil类主要提供静态方法，完成初始化配置等工作；
4. ServiceInvoke是完成WebService服务调用的代理实现，因为目前无法连通厂商的测试环境，因此代码里模拟了一个WSDL接口，后续如果接口调用有差异，主要修改ServiceInvoke即可；



以上的框架，主要为了减少重复代码编写，轻量，而且方便后续维护，大部分的实现都可以在ServiceBase实现，后续任何具体的业务，可以通过继承ServiceBase，并覆盖对应的方法即可，再使用段逻辑，通过面向接口的方式使用，能够通过此方式最大化接口业务逻辑代码，方便后续扩展。



Service项目里，代码文件夹业务含义：

|  |  |
| --- | --- |
| 文件夹 | 含义 |
| login | 登录的验证结果子类 |
| MZBC | 门诊补偿 |
| RJZ | 日记账 |
| SK | 刷卡 |
| ZYBC | 住院补偿 |

注意：目前具体业务子类并没有特殊实现，如果特殊实现，可以覆盖ServiceBase的方法

### 配置文件

paramParse.json配置文件是完成调用接口、请求参数和接口执行结果的每个服务过程的配置，通过此文件：

1. 配置每个服务的输入参数，以及每个参数的顺序业务含义；
2. 配置配个服务的执行返回结果，以及结果中每个分割字符的顺序业务含义；
3. 可以进一步减少使用厂商原有接口的复杂性，并将参数的组装和解析更加易用；
4. 明确每个服务的调用，服务调用无需硬编码实现；

**网站配置文件：**

Web.config是运行网站的配置文件，有几个关键的配置节点：

<!--此节点为Service项目引用厂商接口的自动生产的配置，需要从Service项目的app.config拷贝过来-->

<system.serviceModel>

<bindings>

<basicHttpBinding>

<binding name="Data\_Update\_MSoap"/>

</basicHttpBinding>

</bindings>

<client>

<endpoint address="http://localhost:8044/Data\_Update\_M.asmx" binding="basicHttpBinding" bindingConfiguration="Data\_Update\_MSoap" contract="ServiceReference1.Data\_Update\_MSoap" name="Data\_Update\_MSoap"/>

</client>

</system.serviceModel>

<appSettings>

<!--paramParse.json配置文件的绝对路径-->

<add key="configPath" value="D:\\GitHub\\CSharpHub\\VSWork\\plxnhApi\\ApiMonitor\\paramParse.json"/>

<!--日志文件路径-->

<add key="logPath" value="D:\\GitHub\\mylogs\\"/>

<!--Oracle连接字符串，请确保运行环境安装了OracleClient-->

<add key ="oracleConnection" value="User ID=acs;Password=acs;Data Source=(DESCRIPTION =(ADDRESS = (PROTOCOL = TCP)(HOST = SIT-Oracle-02.quark.com)(PORT = 1521))(CONNECT\_DATA =(SERVER = DEDICATED)(SERVICE\_NAME = CORESIT.quark.com)))"/>

</appSettings>

以上关键点的配置信息，请根据实际情况进行相应的修改配置

## 二、代码使用示例

使用此服务，直接引用Service编译出来的dll即可，然后根据相应的服务和接口文档说明传递参数，例如登录验证的服务调用示例：

IService service = new LoginAuth();

//直接塞参数进入Dictionary，由框架自动组装顺序

Dictionary<string, string> requestParam = new Dictionary<string, string>();

requestParam.Add("USER\_CODE", "123456");

requestParam.Add("USER\_PASS", "xxxxx");

//使用executeSql重载的Dictionary参数方法

string response = service.executeSql("", requestParam, "&");

Dictionary<string, string> responseDict = service.getResponseResultWrapperMap();

//通过Dictionary直接获取值

string USER\_ID = responseDict["USER\_ID"];

string USER\_CODE = responseDict["USER\_CODE"];

//...

## 三、接口字典值配置表

接口存在大量的字典值使用场景，因此新建表xnh\_sys\_cfg来存储和查询：

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 说明 |
| id | number | Id |
| cfg\_key | varchar2(256) | 配置键 |
| cfg\_val | varchar2(512) | 配置值 |
| cfg\_type | varchar2(128) | 配置类型 |
| create\_time | datetime | 创建时间 |

例如：

关于S201-03(诊疗科室).xls的数据，

|  |  |
| --- | --- |
| 01 | 预防保健科 |
| 02 | 全科医疗科 |

可以初始化为：

insert into xnh\_sys\_cfg(id,cfg\_key,cfg\_val,cfg\_type,create\_time)

values(seq\_xnh\_sys\_cfg.nextval,'01','预防保健科','S201-03',sysdate)

insert into xnh\_sys\_cfg(id,cfg\_key,cfg\_val,cfg\_type,create\_time)

values(seq\_xnh\_sys\_cfg.nextval,'02','全科医疗科','S201-03',sysdate)

commit;

比如需要S201-03类别查询01的含义字典值时，可以通过以下查询获得：

select t.\* from xnh\_sys\_cfg t where t.cfg\_type='S201-03' and t.cfg\_key='01'

其他情况的字典值，可以通过配置不同的cfg\_type类别来完成。

### 1、利用工具导入配置数据至表xnh\_sys\_cfg

源码工程中，DataImport项目是导入Excel数据进入表xnh\_sys\_cfg的工具，提供了template.xls和template.xlsx不同版本的Excel模板编辑数据，按固定的模板即可导入数据到表xnh\_sys\_cfg，注意编辑的Excel表里，不要有特殊字符，比如‘’这样的字符。

另外，实际导入数据的时候，修改配置文件DataImport.exe.config

<!--配置的Excel文件路径-->

<add key="filePath" value="D:\GitHub\CSharpHub\VSWork\plxnhApi\DataImport\template.xlsx"/>

<!--导入数据的Excel的Sheet名称-->

<add key="sheetName" value="Sheet1"/>

<!--Oracle数据库连接字符串，运行环境需要要OracleClient驱动-->

<add key ="oracleConnection" value="User ID=acs;Password=acs;Data Source=(DESCRIPTION =(ADDRESS = (PROTOCOL = TCP)(HOST = SIT-Oracle-02.quark.com)(PORT = 1521))(CONNECT\_DATA =(SERVER = DEDICATED)(SERVICE\_NAME = CORESIT.quark.com)))"/>

请根据实际情况修改以上配置信息。

导入数据模板template.xls或template.xlsx，请不要修改第一行的列名，否则读取数据不成功。

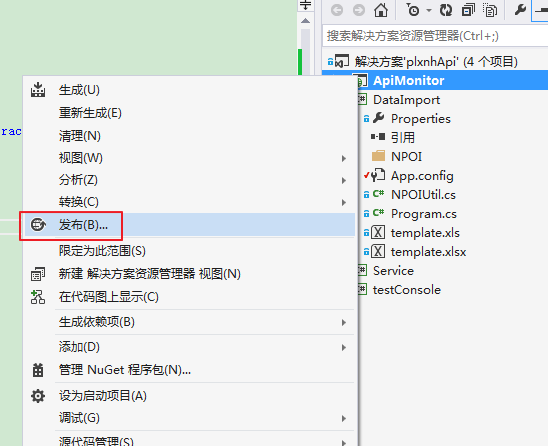
|  |  |  |
| --- | --- | --- |
| key | value | type |

以上Excel里的第一行的列名不能修改。

配置完成之后，运行DataImport.exe即可。

## 四、部署

编译plxnhApi解决方案，如果编译报错，请注意编译环境时候安装了OracleClient驱动，还有是否按装了.Net Framework 4.5，编译完成之后。



配置好发布的路径，即可完成发布，发布出来的程序，即是部署在IIS上的应用。

另外，请关注IIS的版本配置信息，如下图所示，需为v4.0。

