

金蝶云 Web API 接口详细说明文档

整理人：钟镔峰

创建时间：2016-8-22

最后修改：2016-9-22

版本：Ver1.0

联系方式：QQ 476637670

Email: 476637670@qq.com

目的

三方集成，提供第三方系统与 Cloud 集成调用接口。

技术实现

HTTP + Json

提供标准接口

编号	名称	说明
1	Kingdee.BOS.WebApi.ServicesStub.AuthService.ValidateUser	用户验证
2	Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Save	保存
3	Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Audit	审核
4	Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Delete	删除
5	Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.UnAudit	反审核
6	Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Submit	提交
7	Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.View	查看
8	Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.StatusConvert	状态转换
9	Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.ExecuteBillQuery	查询
10	Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.BatchSave	批量保存

示例

三方客户端 引用 Kingdee.BOS.WebApi.Client.dll (注:此客户端引用 Api 封装的是 Http 请求)。

币别->保存 示例

```
public void TestMethodSave_BD_Currency()
{
    //Cloud 业务站点Url
    ApiClient client = new ApiClient("http://localhost:1200/");
    //调用登陆接口 参数 数据中心Id, 用户名, 密码, 语言id
    bool ret = client.Login("005056a30125ad4311e40291d44c593a",
"Administrator", "888888", 2052);
    // 登陆成功
    if (ret)
    {
        //业务对象Id
        string sFormId = "BD_Currency";
        //Json字符串
        string sJson = "{\"Creator\":\"三方调用者标识\", \"NeedUpdateFields\": [\"FNumber\", \"FName\", \"FCODE\"], \"Model\": {\"FCURRENCYID\": 0, \"FNumber\": \"编码\", \"FName\": \"名称\", \"FCODE\": \"货币代码\", \"FPRICEDIGITS\": 4, \"FAMOUNTDIGITS\": 2, \"FPRIORITY\": 0, \"FIsTrans\": false, \"FIsShowCSymbol\": false, \"FIsSysPreset\": false, \"FDescription\": \"info\"}}";
        object[] saveInfo = new object[]
        {
            sFormId,
            sJson
        };
        //调用保存接口

        client.Execute<string>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Save", saveInfo);
    }
}
```

示例说明:

其中主要关注构造的 Json 字符串

通过 Json 编译工具, 可查看及构造相应 Json 数据

powered by ace


```
1 {
2   "Creator": "三方调用者标识",
3   "NeedUpdateFields": [
4     "FNumber",
5     "FName",
6     "FCODE"
7   ],
8   "Model": {
9     "FCURRENCYID": 0,
10    "FNumber": "编码",
11    "FName": "名称",
12    "FCODE": "货币代码",
13    "FPRICEDIGITS": 4,
14    "FAMOUNTDIGITS": 2,
15    "FPRIORITY": 0,
16    "FIsTrans": false,
17    "FIsShowCSymbol": false,
18    "FIsSysPreset": false
19  }
20 }
```

Json 数据是币别界面字段对应的。

币别

新增 ▾ 保存 ▾ 提交 审核 ▾ 业务操作 ▾ 前一 ▾ 后一 ▾ 选项 ▾ 退出

编码 *

名称  *

货币代码 *

☒ 单据显示货币符号

货币符号 ▾

显示格式 ▾

单价精度 4

金额精度 2

优先级

☐ 中间币

包含单据体的 Json 数据构造：

凭证界面构造的 Json 数据，如下图：

```
1 {
2   "Creator": "三方调用者标",
3   "NeedUpdateFields": [
4     "String"
5   ],
6   "Model": {
7     "FVOUCHERID": 0,
8     "FAccountBookID": {
9       "FNumber": "账簿编码"
10    },
11    "FDate": "2014-7-22",
12    "FSystemID": {
13      "FNumber": "来源系统编码"
14    },
15    "FVOUCHERGROUPID": {
16      "FNumber": "凭证字"
17    },
18    "GL_VOUCHER_FEntity": [
19      {
20        "FEXPLANATION": "摘要",
21        "FACCOUNTID": {
22          "FNumber": "科目编码"
23        },
24        "FDEBIT": "100"
25      },
26      {
27        "FEXPLANATION": "摘要",
28        "FACCOUNTID": {
29          "FNumber": "科目编码"
30        },
31        "FCREDIT": "100"
32      }
33    ]
34  }
35 }
```

财务总账凭证保存

在对接之前先熟悉总账流程，在新建账簿、结束初始化等等动作之后，便可以开始对接。注意，该文档凭证只对接到科目，不涉及核算维度。

DoNet 环境下 demo

```
// 使用 webapi 引用组件 Kingdee.BOS.WebApi.Client.dll
//Cloud 3.0
public void TestMethodSave_GL_VOUCHER()
{
    //Cloud 业务站点 Url
    ApiClient client = new ApiClient("http://localhost:1200/");
    //调用登陆接口 参数 数据中心 Id, 用户名, 密码, 语言 id
    bool ret = client.Login("005056a30125ad4311e40291d44c593a", "Administrator",
"888888", 2052);
    // 登陆成功
    if (ret)
    {
        //业务对象 Id 凭证，这里无需修改
        string sFormId = "GL_VOUCHER";
        //Json 字符串
        string sJson =
"{'Creator': 'String', 'NeedUpdateFields': ['String'], 'Model': {'FVOUCHERID': 0, 'FAccountBookID': {'FNumber': '004'}, 'FDate': '2012-1-31', 'FSystemID': {'FNumber': 'gl'}, 'FVOUCHERGROUPID': {'FNumber': 'PRE001'}, 'GL_VOUCHER__FEntity': [{'FEXPLANATION': '11', 'FACCOUNTID': {'FNumber': '1001'}, 'FDEBIT': '100'}, {'FEXPLANATION': '22', 'FACCOUNTID': {'FNumber': '1101'}, 'FCREDIT': '100'}]}}";
        object[] saveInfo = new object[]
        {
            sFormId,
            sJson
        };
        //调用保存接口

client.Execute<string>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Save",
saveInfo);
    }
}
```

Json 字串参数说明

Json 对应树结构参考上面的图

参数名	字段类型	说明
FVOUCHERID	Int	凭证 ID, 如果是新增, 默认给 0, 如果是修改, 对应凭证的 ID, 大于 0
FAccountBookID.FNumber	nvarchar(50)	FAccountBookID 对应账簿, FNumber 对应账簿的内码。账簿请参考:【总账】-【账簿】
FDate	date	凭证日期, 一般给当前日期
FSystemID.FNumber	nvarchar(50)	FSystemID 对应来源系统, FNumber 对应来源系统编码, 这里可以给默认 "GL"
FVOUCHERGROUPIP. FNumber	nvarchar(50)	FVOUCHERGROUPIP 对应凭证字, FNumber 凭证字的内码。通常为: 记、收、付、转, 对应的值分别为: PRE001、PRE002、PRE003、PRE004。如果需要自定义请参考:【总账】-【币别】
GL_VOUCHER__FEntity	-----	凭证分录信息节点, 无需修改
FEXPLANATION	nvarchar(500)	摘要信息
FACCOUNTID.FNumber	nvarchar(50)	FACCOUNTID 对应科目, FNumber 对应科目内码。对接到 K3/CLOUD 总账系统具体那个科目, 请参考【总账】-【科目】
FDEBIT	decimal(23,10)	借方金额
FCREDIT	decimal(23,10)	贷方金额, 注意: 同一张凭证多分录借贷平衡

确定是否对接成功, 可以在【总账】-【凭证查询】界面查看到, 对接过程中遇到任何问题可以随时联系我们技术、业务相关人员, 谢谢。

WebAPI 【财务凭证】核算维度赋值【分享】

WebAPI 维度关联字段功能增强说明

维度关联字段 已改成统一直接按编码构造, 即可实现“维度关联字段”值的保存。
用户无需保证内码值存在, 或者修改为固定列的方式进行保存。

服务端会自动处理,
如有之前的维度关联字段的值组合, 就会使用之前的内码,
如果没有, 会按新传的组合值, 新生成一个内码。

6.0 大版本, 安装 7 月 22 号 6.1 的补丁即有此功能,
如果是 5.0 的版本, 后续补丁也会进行相关功能的同步。

----以下是原帖，会对理解维度关联字段数据库表关系有一定帮助----

写本帖子的目的，

- 1、帮助小伙伴知道下 维度关联字段（弹性域）类型字段 ——K/3 Cloud 的一处强大的特色功能。
- 2、解决 Web API 系统集成调用，此类型字段赋值问题。
- 3、可对核算维度、辅助属性、仓位等维度关联字段的帐表、报表相关取值有一定帮助。

本示例以财务凭证为原型，使用“蓝海机械演示帐套”，可以直接用本帖子提供的 Json 字串进行测试成功。

【财务凭证】（凭证 GL_VOUCHER）的“核算维度”字段，控件的元素实际类型即为“维度关联字段”类型。也常会称之为“弹性域”类型字段。

弹性从字面上理解就是有弹性，能缩能伸。

此类型控件元素，可以看做是 K/3 Cloud 对几种有代表性的业务需求做的业务上的封装。

举个实际我们经常穿的衣服，

如果需要用数据表存储衣服的一条数据信息，我们该如何设计数据表呢，

可能比较容易想到 尺码、品牌、颜色、类型（男装、女装、童装.....）、产地.....

具体到某一件服装类型，可能还会到具体的细节描述，领口 是圆领还是尖领、袖口 纽扣数量..... n（n 可能是

衣服可以看做一种物料，物料的辅助属性。

还有本文帖子中用到的 财务凭证的核算维度。

仓库的仓位划分，一个仓库划分 N 多仓位，这个 N 多仓位下，还会有 N 多小仓位、之后又 N 多小小仓位、又..

说到这里你可能会问，难道这些在 K/3 Cloud 通通能够进行实现？

这里可以肯定的告诉你，答案是肯定的。

在 K/3 Cloud 系统中，此类业务需求被抽象成一个叫“维度关联字段”的元素类型。

论坛中也曾有人问到过

弹性域和普通的基础资料有什么差别？

<http://club.kisdee.com/forum.php?mod=viewthread&tid=653948>

具体拖一个维度关联类型的字段，

“维度数据表单”可关联的下拉中，

我们可以看到

针对【财务凭证】的核算维度，把 Web API 弹性域赋值方式，特写了此分享帖

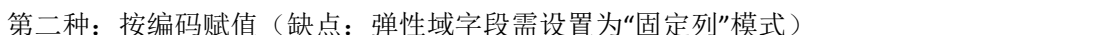
如下:

本帖针是对凭证的核算维度字段赋值，对物料的辅助属性，及仓库的仓位同样参考价值

还是先上截图

第一种：按内码赋值（缺点：内码需要是事先存在的）

ConsoleApplication.WebAPI.Program Invoke()




```

1  --凭证 根据导入Id 数据库中直接查询出的数据
2  select * from T_GL_VOUCHER
3  where FVOUCHERID in ('110405','110407')
4  select FDETAILID, * from T_GL_VOUCHERENTRY
5  where FVOUCHERID in ('110405','110407')
6
7  select * from T_BD_FLEXITEMDETAILV--核算维度数据表 --FDetailID 100880 CUST0001/同益科技
8
9
10
11

```

结果 消息

	FVOUCHERID	FACCOUNTBOOKID	FDATE	FYEAR	PPERIOD	FBILLNO	FVOUCHERGROUPID	FVOUCHERGROUPNO	FATTACH
1	110405	100192	2015-12-09 00:00:00.000	2014	8	24	1	22	0
2	110407	100192	2015-12-09 00:00:00.000	2014	8	25	1	23	0

	FDETAILID	FVOUCHERID	FENTRYID	FEXPLANATION	FACCOUNTID	FDETAILID	FAMOUNTFOR	FAMOUNT	FCURRENCYID	FEXCHANGE
1	100880	110405	102423	1按内码	4001	100880	100.0000000000	100.0000000000	1	1
2	0	110405	102424	2	4004	0	100.0000000000	100.0000000000	1	1
3	100880	110407	102426	11按编码	4001	100880	100.0000000000	100.0000000000	1	1
4	0	110407	102427	22	4004	0	100.0000000000	100.0000000000	1	1

	FID	FOPCODE	FFLEX4	FFLEX5	FFLEX6	FFLEX7	FFLEX8	FFLEX9	FFLEX10	FFLEX11	FF100002	FF100003	FCALCOL
42	100264		0	100810	0	0	0	20142	0	0	0		010081000020...
43	100331		0	0	0	0	0	0	0	100004	0		00000001000040
44	100347		0	100202	0	0	0	20142	0	0	0		010020200020...
45	100348		0	100203	0	0	0	20142	0	0	0		010020300020...
46	100361		0	0	0	0	0	0	3	0	0		000000300
47	100835		0	100812	0	0	0	20142	0	0	0		010081200020...
48	100880		0	0	100044	0	0	0	0	0	0		00100044000000
49	100883		100122	0	0	0	0	0	0	0	0		10012200000000
50	100928		0	100205	0	0	0	20045	0	0	0		010020500020...

```
1. sFormId = "GL VOUCHER";
```

```
2. //按内码赋值 FDetailID 100880 CUST0001/同益科技
3. //sContent =
   "{\"Creator\":\"String\",\"NeedUpdateFields\":[],\"Model\":{\"FVOUCHERID\":\"0\",\"FAccountBookID\":{\"FNumber\":"
   "\"1 按内码\",\"FACCOUNTID\":{\"FNumber\":\"1001\"},\"FDetailID\":\"100880\",\"FDEBIT\":\"100\"},{\"FEXPLANAT
4. //按编码赋值 FDetailID 100880 CUST0001/同益科技
5. sContent =
   "{\"Creator\":\"String\",\"NeedUpdateFields\":[],\"Model\":{\"FVOUCHERID\":\"0\",\"FAccountBookID\":{\"FNumber\":"
   "\"11 按编码\",\"FACCOUNTID\":{\"FNumber\":\"1001\"},\"FDetailID\":{\"FDetailID__FFLEX6\":{\"FNumber\":\"C
6.
复制代码
```

其中：

第一种按内码赋值的关键代码段：

```
\"FDetailID\":\"100880
```

值 100880 通过第三张截图可以看出是核算维度的一个内码。此种方式前面也有说内码是事先存在的

第二种按编码赋值的关键代码段：

```
\"FDetailID\":{\"FDetailID__FFLEX6\":{\"FNumber\":\"CUST0001\"}}
```

值 CUST0001 是蓝海机械测试帐套一个客户的编码值（同益科技）。

金蝶K/3Cloud

demo0727 demo 101.2 总装事业部 帮助

凭证查询 凭证 - 修改

新增 保存 提交 审核 还原 出纳复核 业务操作 前一 后一 现金流量 关联查询 选项 自动计算 退出

记账凭证

日期: 2015/12/9

账簿: 变电器公司主账簿

核算组织: 变电器公司

凭证字: 记

2014年第08期

凭证号: 23

附件数:

新增行	删除行	插入行	供应商	客户	物料
			同益科技		

上图固定列的方式是在这里设置的：

属性

核算维度-FDetailID

快速查询位置	0
过滤比较符号	0
停靠	无
维度关联字段	单据体.科目编码
维度关联字段	核算维度
维度数据表单	核算维度
维度显示样式	固定列
列表默认显示	100
帮助上下文标	

维度显示样式

元素的维度显示样式配置，支持弹出窗与固定列

其实凭证核算维度字段弹出窗默认是这个样子的：

192.168.18.133/k3cloud

金蝶K/3 Cloud demo0727 demo 101.2 总装事业部 帮助 注销

凭证查询 凭证 - 修改

新增 保存 提交 审核 还原 出纳复核 业务操作 前一 后一 现金流量 关联查询 选项 自动计算 退出

记账凭证

日期 2015/12/9 1

凭证字 记

账簿 变电器公司主账簿 2014年第0期 凭证号 23

核算组织 变电器公司 附件数

序号	摘要	科目编码	科目全名	核算维度	借方金额	贷方金额	结算方式	结算号
1	11按编码	1001	库存现金	CUST0001/同益科技	¥100.00			
2	22	1101	交易性金融资产			¥100.00		
3								
4								
5								
6								
7								
8								

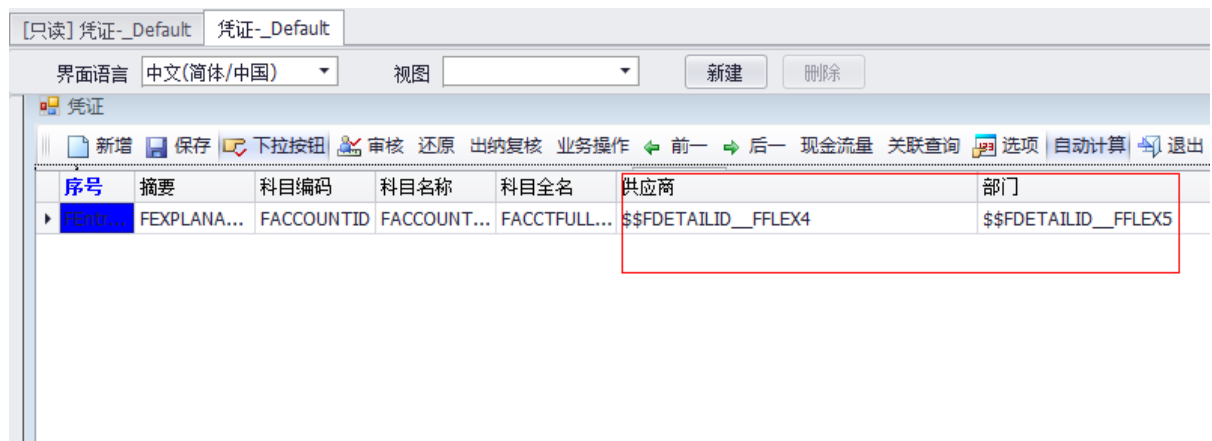
维度数据录入

客户 同益科技

确定 取消

FDetailID_FFLEX6 这个 Key 值

要看 BOS IDE 的字段命名，及结合 T_BD_FLEXITEMDETAILV 数据库维度表的查询 找出 Key 值



SQL 语句:

--凭证 根据导入 Id 数据库中直接查询出的数据

```
select * from T_GL_VOUCHER
```

```
where FVOUCHERID in ('110405','110407')
```

```
select FDETAILID, * from T_GL_VOUCHERENTRY
```

```
where FVOUCHERID in ('110405','110407')
```

```
select * from T_BD_FLEXITEMDETAILV--核算维度数据表 --FDetailID 100880 CUST0001/同益科技
```

另附:

```
select * from T_BD_FLEXITEMDETAILV--辅助属性数据表
```

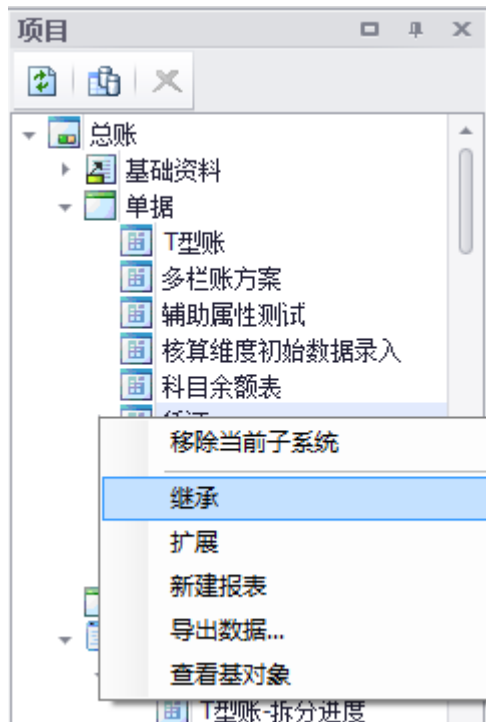
```
select * from T_BAS_FLEXVALUESDETAIL --值集资料维度数据 仓位数据表
```

至此：相信遇到所有维度关联字段（弹性域）类型的单据字段的 WebAPI 方式赋值，相信你都可以做到了。

1、什么你想说不想改变原配置单据的维度关联字段的“弹出窗”模式，因为这会影响到用户的界面操作体验。

提醒下 别忘了 K/3 Cloud 单据在有强大的扩展功能的同时，也是有继承方式的

BOS IDE 中右键



什么 新来的小伙伴继承及扩展还是第一次听说？

这里有个经典帖

K/3 Cloud BOS 表单的继承和扩展的区别:

<http://club.kisdee.com/forum.php?mod=viewthread&tid=534543>

2、什么还有其他的 该没有其他的了

来个终结吧：

比较**推荐第一种按内码的方式进行维度关联字段赋值**，

即在调用接口前，把需要的数据在系统里都事先准备好，而不是临时去新增。

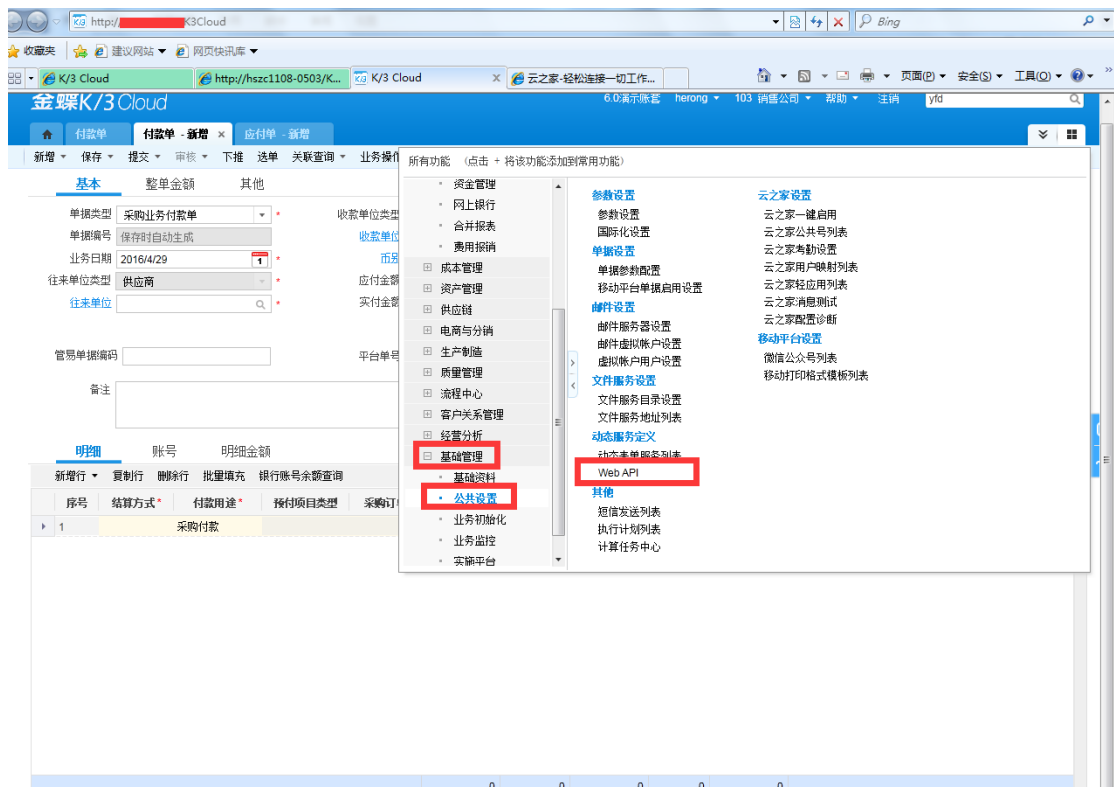
原因：

- 1、**从数据正确性考虑** 主要是维度关联字段想按编码赋值，在接口里把数据控制好不是一件易事，很容易就出错
- 2、**从性能角度考虑** 按内码方式赋值，一定是有性能上的提高的，而且对性能提高还不容小觑，尤其是对此种类型

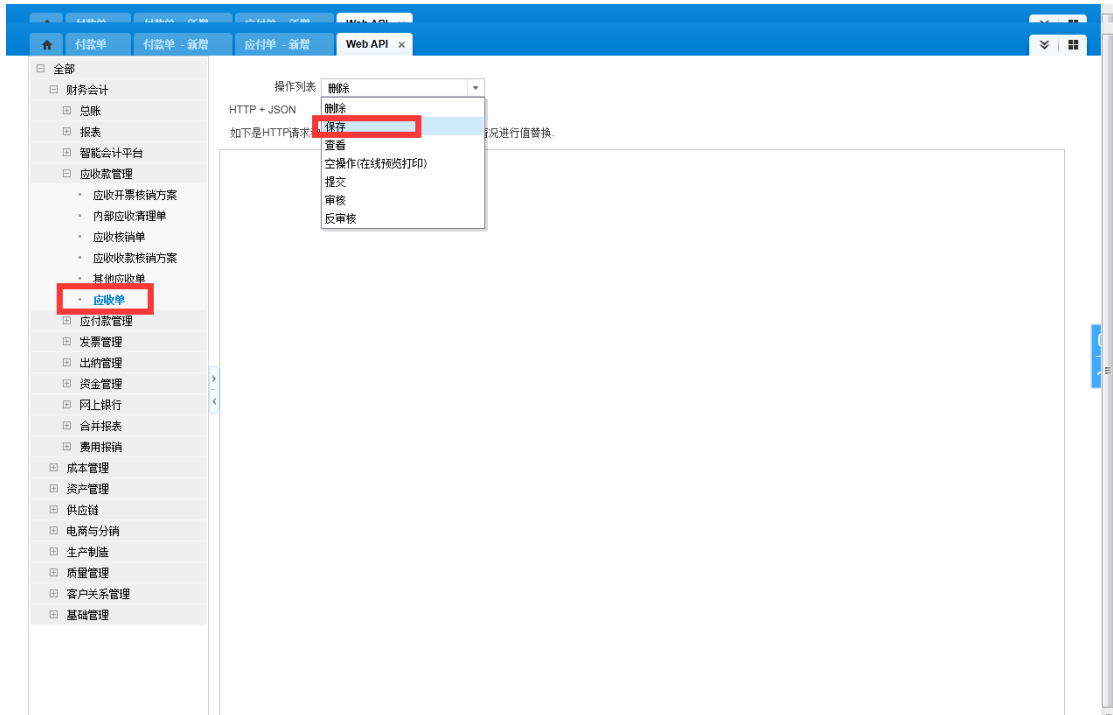
金蝶云 API 接口列表

1. 进入金蝶云

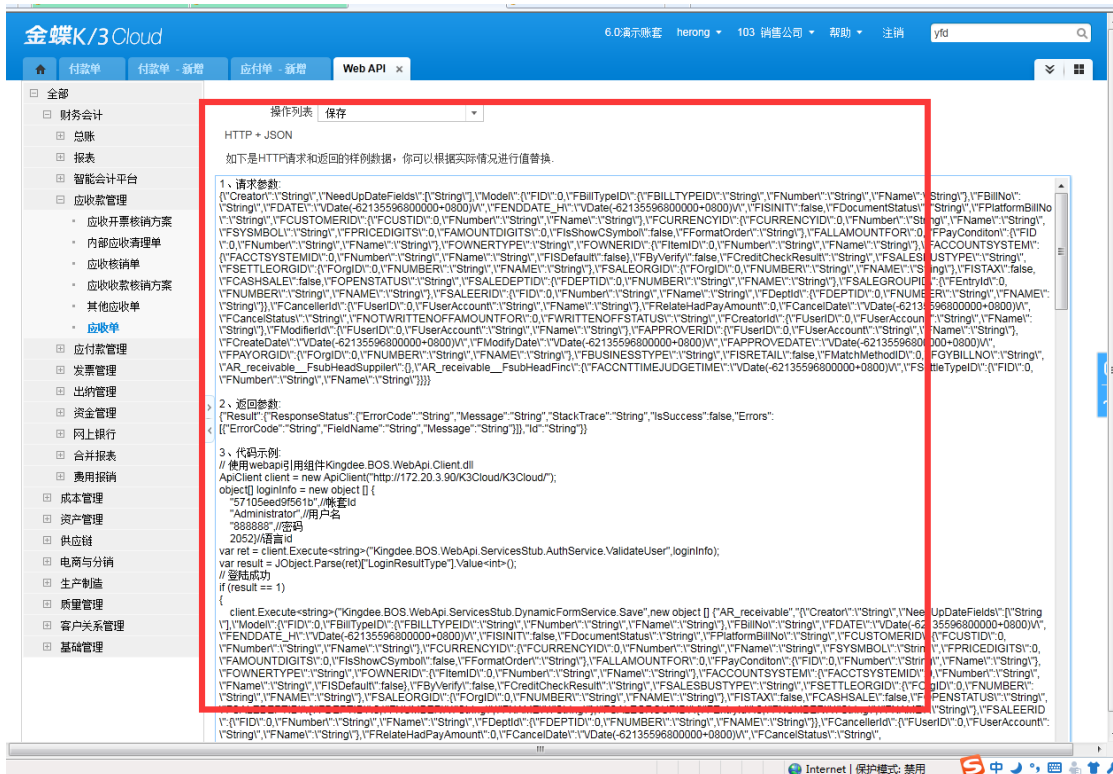
2. 找到菜单 web API



3. 进入打如下



选择保存后，会显示



4. 根据实例选择进行编写程序，即可进行数据保存

5. 论坛参考案例代码

<http://club.kisdee.com/forum.php?mod=viewthread&tid=741861>

6. 标准接口后缀地址说明前缀统一

http://XXXX/K3Cloud/

接口后缀串	说明	备注
Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Save.common.kdsvc	保存	
Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.BatchSave.common.kdsvc	批量保存	
Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.View.common.kdsvc	查看	
Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Submit.common.kdsvc	提交	
Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Audit.common.kdsvc	审核	
Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.UnAudit.common.kdsvc	反审核	
Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.StatusConvert.common.kdsvc	状态转换	

调用详细接口

http://XXXX/K3Cloud/

Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.StatusConvert.common.kdsvc

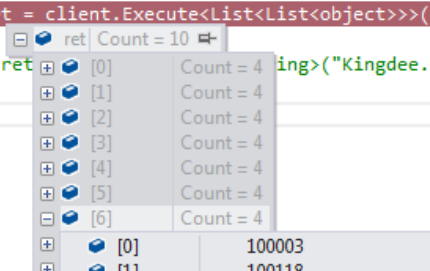
如果在浏览器中能数据返回即可表示服务正常

7. 查询数据

```
public static void Query()
{
    //Cloud 业务站点Url
    ApiClient client = new ApiClient("http://localhost/K3Cloud/");
    //调用登陆接口 参数 数据中心Id, 用户名, 密码, 语言id
    string dbId = "57b182d696bae8";
    bool bLogin = client.Login(dbId, "administrator", "888888", 2052);
    // 登陆成功
    if (bLogin)
    {
        object[] paramInfo = new object[]
        {
            "{\"FormId\": \"PUR_PurchaseOrder\", \"+// 采购订单formid
            \"TopRowCount\": 0, \"+// 最多允许查询的数量, 0或者不要此属性表示不
限制
            \"Limit\": 10, \"+// 分页取数每页允许获取的数据, 最大不能超过2000
            \"StartRow\": 0, \"+// 分页取数开始行索引, 从0开始, 例如每页10行数据,
第2页开始是10, 第3页开始是20, 以此类推, 当不提供此属性, 表示仅查询Limit中填写的数据量
            \"FilterString\": \"\", \"+// 过滤条件
            \"OrderString\": \"FID ASC\", \"+// 排序条件
            \"FieldKeys\": \"FID, FSupplierId, FMaterialId, FNumber, FMaterialName\"} // 获取采
购订单内码, 供应商id, 物料id, 物料编码, 物料名称
        };
        //调用保存接口
        var ret =
client.Execute<List<List<object>>>>("Kingdee.BOS.WebApi.ServicesStub.DynamicForm
Service.ExecuteBillQuery.common.kdsvc", paramInfo);
    }
}
```

```
public static void Query()
{
    //Cloud 业务站点Url
    ApiClient client = new ApiClient("http://localhost:1200/");
    //调用登陆接口 参数 数据中心Id, 用户名, 密码, 语言id
    string dbId = "5643eebf415b3d";
    bool bLogin = client.Login(dbId, "demo", "888888", 2052);

    // 登陆成功
    if (bLogin)
    {
        object[] paramInfo = new object[]
        {
            "{ \"FormId\": \"PUR_PurchaseOrder\", \"+// 采购订单formid
            \"TopRowCount\": 0, \"+// 最多允许查询的数量, 0或者不要此属性表示不限制
            \"Limit\": 10, \"+// 分页取数每页允许获取的数据, 最大不能超过2000
            \"StartRow\": 0, \"+// 分页取数开始行索引, 从0开始, 例如每页10行数据, 第2页开始是10, 第3页开始
            \"FilterString\": \"\", \"+// 过滤条件
            \"OrderString\": \"FID ASC\", \"+// 排序条件
            \"FieldKeys\": \"FID,FSupplierId,FMaterialId.FNumber,FMaterialName\" }\"// 获取采购订单数据
        };
        //调用保存接口
        var ret = client.Execute<List<List<object>>>>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormSe
        //var ret = client.Execute<List<List<object>>>>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Exe
    }
}
```



附录

K3Cloud WebAPI 接口说明书_V1.0

修改记录

Ver. No	发版日期	编制人	批准人	修改的章节号
V1.0	2016/8/9	刘兵	赖碧云	初始版本

目 录

1. 概述	20
1.1. 目的	20
1.2. 范围	20
1.3. 适用对象	20
1.4. 参考资料	20
2. 问题与解决策略	20
3. 目标和约束	21
4. WebAPI 架构	21
4.1. 采用的技术	21
4.1.1. Kingdee.BOS.WebApi.FormService.dll	21
4.1.2. Kingdee.BOS.WebApi.ServicesStub.dll	21
4.1.3. Kingdee.BOS.WebApi.Client.dll	21
4.1.4. 开发工具	22
5. WebAPI 接口详细描述	22
5.1.1. 登陆验证接口	22
5.1.2. 查看表单数据接口	25
5.1.3. 保存表单数据接口	26
5.1.4. 批量保存表单数据接口	29
5.1.5. 提交表单数据接口	32
5.1.6. 审核表单数据接口	34
5.1.7. 反审核表单数据接口	35
5.1.8. 删除表单数据接口	36
5.1.9. 表单数据查询接口	37
5.1.10. 自定义 WebAPI 接口	39
6. 附录（集成相关知识分享）	42

1. 概述

1.1. 目的

为异构系统访问 K/3Cloud 系统数据提供通用的接口。

当企业规模逐渐增大时，作为支撑业务运营的 IT 建设也变得越来越重要，不过往往企业的 IT 建设过程中会发现某一家软件供应商基本不能完全覆盖企业所有的业务运营流程，这样的结果就是，企业上的 IT 系统很多很全，从 ERP 到 HR、CRM、PDM、OA 等，貌似所有的业务都覆盖到了，但实际上因为这些系统的不集成，而形成了企业很多新的信息孤岛，非常不利于企业的后续的管理和战略发展。K/3Cloud 从现今和往后的发展趋势来看，也不可避免会遇到上述问题，毕竟企业经营的多样化，并不是所有的业务都能在 K/3Cloud 中完成，所以我们在产品架构上支持更好的与外部系统进行协同。

1.2. 范围

➤ 适用版本：V5.0、V6.0、V6.1

1.3. 适用对象

本文档适用于：

➤ 开发工程师：**参考**，对系统集成的实现获取**全局性**的设计**指导**。

1.4. 参考资料

2. 问题与解决策略

愿景	关注点	描述与示例

3. 目标和约束

目标:

- 提供对 K/3Cloud 单据和基础信息的查看、保存、提交、审核、反审核和删除等功能;
- 提供对 K/3Cloud 单据和基础信息的查询功能;

约束:

- 数据操作接口仅支持以基础资料编码、单据编号或直接以表单主键去操作数据;
- 支持对某一具体单据的数据查询, 但多单关联查询需要二开接口实现。

4. WebAPI 架构

4.1. 采用的技术

K/3 Cloud WebAPI 是一种轻量级的、可维护的、可伸缩的 Web 服务。采用 HTTP+JSON, 也就是用 RESTful 的方式来开发。使用 .NET Framework 4.0 为开发平台, 源代码使用 C# 编写。整个框架由三个组装件组成。

4.1.1. Kingdee.BOS.WebApi.FormService.dll

此组装件包含 WebAPI 主要接口的功能实现。部署在应用层服务器。

4.1.2. Kingdee.BOS.WebApi.ServicesStub.dll

此组装件主要包含 WebAPI 接口定义, 扩展接口定义以及登陆验证接口。部署在应用层服务器。

4.1.3. Kingdee.BOS.WebApi.Client.dll

此组装件为 WebAPI 的客户端组件, 封装了一些在异构系统客户端访问 WebAPI 的方法, 适用于 C# 程序调用。由于它应用于异构系统客户端, 所以此组装件需要拷贝到异构系统客户

端环境中。非 C# 程序调用可以不用拷贝。

4.1.4. 开发工具

.Net Framework 4.0

Microsoft Visual Studio 2012

5. WebAPI 接口详细描述

5.1.1. 登陆验证接口

服务地址：

<http://ServerIp/K3Cloud/Kingdee.BOS.WebApi.ServicesStub.AuthService.ValidateUser.common.kdsvc>

接口参数：

参数列表	参数含义	备注
acctID	账套 Id, 从管理中心数据库查询获得 参考查询: select FDATACENTERID from T_BAS_DATACENTER	必须
username	用户登陆名	必须
password	密码	必须
lcid	语言 id, 选择哪种语言访问, 参考: 中文 2052, 英文 1033, 繁体 3076	非必须, 引用 SDK 组件辅助类调用则必须

返回参数：

参数列表	参数含义	备注
------	------	----

LoginResultType	<pre>//激活 Activation = -7, //云通行证未绑定Cloud账号 EntryCloudUnBind = -6, //需要表单处理 DealWithForm = -5, //登录警告 Wanning = -4, //密码验证不通过（强制的） PWInvalid_Required = -3, //密码验证不通过（可选的） PWInvalid_Optional = -2, //登录失败 Failure = -1, //用户或密码错误 PWEError = 0, //登录成功 Success = 1</pre>	<p>管理员登陆可能出现返回-5 的情况，这种情况在api 验证时也可认为是允许的，具体可以根据实际情况来定。</p> <pre>var result = JsonObject.Parse(ret)["LoginResult Type"].Value<int>(); if (result == 1 result==5) { return true; }</pre>
-----------------	---	--

调用参考：

1) SDK 辅助类示例（引用 Kingdee.BOS.WebAPI.Client.dll）：

说明：下文中出现的 client 都为此处进行过登陆验证的 ApiClient 实例。

```
ApiClient client = new ApiClient("http://192.168.66.60/k3cloud/");
string dbId = "5756960b27blaa"; //AotuTest117
bool bLogin = client.Login(dbId, "demo", "888888", 2052);
if (bLogin)
{
    //todo:登陆成功处理业务
}
```

2) 无引用组件示例（不引用金蝶的组件）：

说明：以下 HttpClient 为自定义客户端调用辅助类，下文中所指 HttpClient 都表示这个类。

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Net;
using System.Text;
using Newtonsoft.Json.Linq;
using Newtonsoft.Json;

public class HttpClient
{
    /// <summary>
```

```
/// Seivice URL
/// </summary>
public string Url { get; set; }
/// <summary>
/// 内容
/// </summary>
public string Content { get; set; }
/// <summary>
/// Cookie, 保证登录后, 所有访问持有一个Cookie;
/// </summary>
static CookieContainer Cookie = new CookieContainer();

/// <summary>
/// HTTP访问
/// </summary>
public string AsyncRequest()
{
    HttpWebRequest httpRequest = HttpWebRequest.Create(Url) as
HttpWebRequest;
    httpRequest.Method = "POST";
    httpRequest.ContentType = "application/json";
    httpRequest.CookieContainer = Cookie;
    httpRequest.Timeout = 1000 * 60 * 10;//10min

    using (Stream reqStream = httpRequest.GetRequestStream())
    {
        JObject jsonObj = new JObject();
        jsonObj.Add("format", 1);
        jsonObj.Add("useragent", "ApiClient");
        jsonObj.Add("rid",
Guid.NewGuid().ToString().GetHashCode().ToString());
        jsonObj.Add("parameters", Content);
        jsonObj.Add("timestamp", DateTime.Now);
        jsonObj.Add("v", "1.0");
        string sContent = jsonObj.ToString();
        var bytes = UnicodeEncoding.UTF8.GetBytes(sContent);
        reqStream.Write(bytes, 0, bytes.Length);
        reqStream.Flush();
    }
    using (var repStream = httpRequest.GetResponse().GetResponseStream())
    {
        using (var reader = new StreamReader(repStream))
        {
            return ValidateResult(reader.ReadToEnd());
        }
    }
}
```



```
    }  
    }  
}  
  
private static string ValidateResult(string responseText)  
{  
    if (responseText.StartsWith("response_error:"))  
    {  
        return responseText.TrimStart("response_error:".ToCharArray());  
    }  
    return responseText;  
}  
}
```

登陆验证参考:

```
HttpClient httpClient = new HttpClient();  
httpClient.Url =  
"http://192.168.66.60/k3cloud/Kingdee.BOS.WebApi.ServicesStub.AuthService.ValidateUser.common.kdsvc";  
List<object> Parameters = new List<object>();  
Parameters.Add("558cbb01bfc79b");//帐套Id  
Parameters.Add("Administrator");//用户名  
Parameters.Add("888888");//密码  
Parameters.Add(2052);  
httpClient.Content = JsonConvert.SerializeObject(Parameters);  
var iResult =  
JsonObject.Parse(httpClient.AsyncRequest())["LoginResultType"].Value<int>();  
if (iResult == 1){  
    //todo:验证成功, 处理业务  
}
```

5.1.2. 查看表单数据接口

服务地址:

<http://ServerIp/K3Cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.View.common.kdsvc>

接口参数:

参数列表	参数含义	备注
formid	表单 Id, 格式参考: "IV_SALESIC"//表示发票	必须

data	数据包，Number 表示单据编号或者基础资料编码，Id 为主键，CreateOrgId 为创建组织 Id。格式参考： "{\"CreateOrgId\":0,\"Id\":0,\"Number\": \"SVINV00000003\"}"	CreateOrgId: 非必须 Id 和 Number: 任选一个
------	---	---

返回参数:

参数列表	参数含义	备注
ResponseStatus	操作状态	操作是否成功，如果失败，具体失败原因
Result	单据数据包	单据完整数据内容

调用参考:

1) SDK 辅助类示例（引用 Kingdee.BOS.WebAPI.Client.dll）:

```
string sJson = "{\"Number\":\"SVINV00000003\"}";  
var result =  
client.Execute<string>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.View",  
new object[] { "IV_SALESIC", sJson });
```

2) 无引用组件示例（不引用金蝶的组件）:

```
HttpClient httpClient = new HttpClient();  
httpClient.Url =  
"http://192.168.66.60/k3cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.View.common.kdsvc";  
List<object> Parameters = new List<object>();  
//业务对象Id  
String formid = "IV_SALESIC";//发票  
Parameters.Add(formid);  
//Json字串  
string data = "{\"Number\":\"SVINV00000003\"}";  
Parameters.Add(data);  
httpClient.Content = JsonConvert.SerializeObject(Parameters);  
var result = httpClient.AsyncRequest();
```

5.1.3. 保存表单数据接口

服务地址:

<http://ServerIp/K3Cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Save.common.kdsvc>

接口参数:

参数列表	参数含义	备注
formid	表单 Id, 格式参考: "BD_Currency"//表示币别	必须
data	数据包, 数据格式参考以币别为例: {"Creator\":"\", "NeedUpdateFields\":["\", "Model\":{"FNumber\":"PRENB001\", "FName\":"牛币\", "FCODE\":"CNY\", "FPRICEDIGITS\":6, "FAMOUNTDIGITS\":2, "FPRIORITY\":0, "FIsShowCSymbol\":true }}"	Creator: 非必须 NeedUpdateFields: 非必须, 如果是更新单据而不是新增一个单据, 以设置这个数据集合, 内容为字段标识, 使用逗号分隔, 如果需要更新表体数据, 则需要填写表体的标识。一般情况下, 如果 Model 中只填写了需要更新的字段信息, 没有其他冗余的字段, 那么这个参数不需要设置, 否则其他冗余字段会覆盖 ERP 中的源单数据。 Model: 必须, Model 数据格式说明: 所有字段和实体都是用元素的标识来识别。单据头字段直接填写字段标识和数据, 子单据头字段需要先声明是子单据头信息, 然后再填写其字段信息, 例如 "SubHeadEntity\":{"FBarcode\":"20305\"}如果是单据体字段则需要申明为集合, 例如: "EntryDetail\":[{第1条明细},{第N条明细}], 需要用中括号括起来。 另外要注意的是更新单据时, 表体信息必须填写明细表体的主键, 如果不填写, 将会删除源单的表体, 而没有主键的数据会视为新增, 有主键的视为更新。

返回参数:

参数列表	参数含义	备注
ResponseStatus	操作状态 {"ResponseStatus":{"ErrorCode":""," "IsSuccess":false, "Errors":[{"FieldName":""," "Message":"","DIndex":0}], "SuccessEntitys":[{"Id":"","Number":"","DIndex":0}], "SuccessMessages":[{"FieldName":"","Message":"","DIndex":0}]}	IsSuccess:操作是否成功, Errors:如果失败, 具体失败原因 DIndex:原始数据编号

调用参考:

1) SDK 辅助类示例 (引用 Kingdee.BOS.WebAPI.Client.dll):

```
//业务对象Id
string sFormId = "SAL_OUTSTOCK"; //销售出库单
//Model字符串
string sContent = "{\"Creator\":\"\", \"NeedUpdateFields\":[], \"Model\": \" +
\"{\\\"FID\\\":\\\"0\\\", \\\"FStockOrgId\\\":{\\\"FNumber\\\":\\\"210\\\"}, \\\"FBillTypeID\\\":{\\\"FNU
```

```
mber\":"XSKD01_SYS\"},"FBillNo\":"CSDGBC21002\","FCustomerID\":{"FNumber\":"CUST0073\"},"SubHeadEntity\":{"FExchangeRate\":6.51},"FEntity\":[{"FEntryID\":"0","FMATERIALID\":{"FNumber\":"03.001\"},"FStockID\":{"FNumber\":"CK002\"},"FRealQty\":324,"FBaseUnitQty\":324},{"FEntryID\":"0","FMATERIALID\":{"FNumber\":"03.001\"},"FStockID\":{"FNumber\":"CK004\"},"FRealQty\":220,"FBaseUnitQty\":220}]}}";  
object[] saveInfo = new object[]  
{  
    sFormId,  
    sContent  
};  
//调用保存接口  
var ret =  
client.Execute<string>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Save", saveInfo);
```

以下示例说明如何更新单据的表体信息：

更新内码为 100017 的单据，新增两行表体数据，原有 100024 和 100025 行数据保留，如果源单中还有其他分录则删除。

```
string sFormId = "STK_MISCELLANEOUS";//其他入库单  
string sContent =  
"{\"Creator\":"\"\", \"NeedUpdateFields\":[], \"Model\":{"FID\":"100017\", \"FEntity\":[{\"FEntryID\":"100024\"},{\"FEntryID\":"100025\"},\" +  
\"{\\"FMATERIALID\":"\"A.0060480933\", \"FUnitID\":"\"FNumber\":"\"03\", \"FSTOCKID\":"\"FNumber\":"\"PRTSH\", \"FQty\":"\"200\", \"FBASEQTY\":"\"200\", \"FPlanAmount\":"\"132856\", \"FBASEUNITID\":"\"FNumber\":"\"03\", \"FEntryNOTE\":"\"2015-7-29\", \"FAmount\":"\"132856\", \"FPRICE\":"\"664.28\", \"FEntrySelfA9725\":"\"2015072802\", \"FEntrySelfA9733\":"\"19881\", \"FEntrySelfA9734\":"\"2015-6-3\", \"FEntrySelfA9737\":"\"200\", \"FEntrySelfA9740\":"\"0\", \"FEntrySelfA9738\":"\"1\", \"FEntrySelfA9739\":"\"2\", \"FEntrySelfA9735\":"\"664.28\"},\" +  
\"{\\"FMATERIALID\":"\"A.0060480933\", \"FUnitID\":"\"FNumber\":"\"03\", \"FSTOCKID\":"\"FNumber\":"\"PRTSH\", \"FQty\":"\"200\", \"FBASEQTY\":"\"200\", \"FPlanAmount\":"\"132856\", \"FBASEUNITID\":"\"FNumber\":"\"03\", \"FEntryNOTE\":"\"2015-7-29\", \"FAmount\":"\"132856\", \"FPRICE\":"\"664.28\", \"FEntrySelfA9725\":"\"2015072802\", \"FEntrySelfA9733\":"\"19881\", \"FEntrySelfA9734\":"\"2015-6-3\", \"FEntrySelfA9737\":"\"200\", \"FEntrySelfA9740\":"\"0\", \"FEntrySelfA9738\":"\"1\", \"FEntrySelfA9739\":"\"2\", \"FEntrySelfA9735\":"\"664.28\"}]}}\"";  
  
object[] saveInfo = new object[]  
{  
    sFormId,  
    sContent  
};
```

```
//调用保存接口
var ret=
client.Execute<string>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.S
ave", saveInfo);
```

2) 无引用组件示例（不引用金蝶的组件）:

```
HttpClient httpClient = new HttpClient();
httpClient.Url =
"http://192.168.66.60/k3cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormSer
vice.Save.common.kdsvc";
List<object> Parameters = new List<object>();
//业务对象Id
String formid = "SAL_OUTSTOCK";//销售出库为例
Parameters.Add(formid);
//Json字符串
string data = "{\"Creator\":\"\",\"NeedUpdateFields\":[],\"Model\":\" +
{\"FID\":\"0\",\"FStockOrgId\":{\"FNumber\":\"210\"},\"FBillTypeID\":{\"FNu
mber\":\"XSKD01_SYS\"},\"FBillNo\":\"CSDGBC21002\",\"FCustomerID\":{\"FNu
mber\":\"CUST0073\"},\"SubHeadEntity\":{\"FExchangeRate\":6.51},\"FEntity\":[{
\"FEntryID\":\"0\",\"FMATERIALID\":{\"FNumber\":\"03.001\"},\"FStockID\":{\"
FNumber\":\"CK002\"},\"FRealQty\":324,\"FBaseUnitQty\":324},{\"FEntryID\":\"
0\",\"FMATERIALID\":{\"FNumber\":\"03.001\"},\"FStockID\":{\"FNumber\":\"CK0
04\"},\"FRealQty\":220,\"FBaseUnitQty\":220}]}}";
Parameters.Add(data);
httpClient.Content = JsonConvert.SerializeObject(Parameters);
var result = httpClient.AsyncRequest();
```

5.1.4. 批量保存表单数据接口

服务地址:

[http://ServerIp/K3Cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.BatchSave.com
mon.kdsvc](http://ServerIp/K3Cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.BatchSave.common.kdsvc)

接口参数:

参数列表	参数含义	备注
formid	表单 Id, 格式参考: "SAL_SaleOrder"//表示销售订单	必须

data	<p>数据包，格式参考以销售订单为例：</p> <pre> "{"NeedUpdateFields\":[],\BatchCount\":"2", \Model\":[{"FID\":"0",\FBILLTYPEID\":{"FNumber\":"XSDD01_SYS"}, \FSaleOrgId\":{"FNumber\":"NCIC"},\FSaleDeptId\":{"FNumber\":"01.A13"}, \FCustId\":{"FNumber\":"500"},\FDATE\":"2016/2/23", \FSettleCurrId\":{"FNumber\":"RMB"},\FSalerId\":{"FNumber\":"B027"}, \FBusinessType\":"NORMAL",\FSaleOrderFinance\":{"FEXCHANGETYPE\":"FNumber\":"NCIC"}, \FExchangeRate\":"1"},\FSaleOrderEntry\":[{"FMATERIALID\":{"FNumber\":"A.0060480931"}, \FUNITID\":{"FNumber\":"03"},\FQty\":"1",\FTaxPrice\":"100",\FEntryTaxRate\":"17.00"}]}, {"FID\":"0",\FBILLTYPEID\":{"FNumber\":"XSDD01_SYS"}, \FSaleOrgId\":{"FNumber\":"NCIC"},\FSaleDeptId\":{"FNumber\":"01.A13"}, \FCustId\":{"FNumber\":"500"},\FDATE\":"2016/2/23", \FSettleCurrId\":{"FNumber\":"RMB"},\FSalerId\":{"FNumber\":"B027"}, \FBusinessType\":"NORMAL",\FSaleOrderFinance\":{"FEXCHANGETYPE\":"FNumber\":"NCIC"}, \FExchangeRate\":"1"},\FSaleOrderEntry\":[{"FMATERIALID\":{"FNumber\":"A.0060480931"}, \FUNITID\":{"FNumber\":"03"},\FQty\":"1",\FTaxPrice\":"100",\FEntryTaxRate\":"17.00"}]}, {"FID\":"0",\FBILLTYPEID\":{"FNumber\":"XSDD01_SYS"}, \FSaleOrgId\":{"FNumber\":"NCIC"},\FSaleDeptId\":{"FNumber\":"01.A13"}, \FCustId\":{"FNumber\":"500"},\FDATE\":"2016/2/23", \FSettleCurrId\":{"FNumber\":"RMB"},\FSalerId\":{"FNumber\":"B027"}, \FBusinessType\":"NORMAL",\FSaleOrderFinance\":{"FEXCHANGETYPE\":"FNumber\":"NCIC"}, \FExchangeRate\":"1"},\FSaleOrderEntry\":[{"FMATERIALID\":{"FNumber\":"A.0060480931"}, \FUNITID\":{"FNumber\":"03"},\FQty\":"1",\FTaxPrice\":"100",\FEntryTaxRate\":"17.00"}]}]" </pre>	<p>BatchCount：非必须。此参数主要用于优化性能，当传入的单据数据量较大时，可以设定此参数的并行分批执行次数。例如传入100张单据数据，此参数设定为10，则表示在k3cloud系统中，以10个单据为一批，分10批，同时并发保存，提升效率。</p> <p>数据包参数格式和Save接口的类似，主要差别在于批量保存是传入多张单据的数据，Modle数据用[]括起来，而Save仅传入一张单据数据。再一个区别是批量保存多了一个参数BatchCount。</p>
------	---	---

返回参数：

参数列表	参数含义	备注
ResponseStatus	<p>操作状态</p> <pre> {"ResponseStatus":{"ErrorCode":""," "IsSuccess":false, "Errors":[{"FieldName":""," "Message":"","DIndex":0}], "SuccessEntitys":[{"Id":"","Number":"","DIndex":0}], "SuccessMessages":[{"FieldName":"","Message":"","DIndex":0}]} </pre>	<p>IsSuccess:操作是否成功，</p> <p>Errors:如果失败，具体失败原因</p> <p>DIndex:原始数据号，如果接口参数用了BatchCount，则DIndex返回的顺序不是所有数据的顺序，而是每批次的顺序。</p>

调用参考:

1) SDK 辅助类示例 (引用 Kingdee.BOS.WebAPI.Client.dll):

```
string sFormId = "SAL_SaleOrder";//销售订单为例
string sContent =
"{\"NeedUpdateFields\": [], \"BatchCount\": \"2\", \"Model\": [{\"FID\": \"0\", \"FBILLTYPEID\": {\"FNumber\": \"XSDD01_SYS\"}, \"FSaleOrgId\": {\"FNumber\": \"NCIC\"}, \"FSaleDeptId\": {\"FNumber\": \"01.A13\"}, \"FCustId\": {\"FNumber\": \"500\"}, \"FDATE\": \"2016/2/23\", \"FSettleCurrId\": {\"FNumber\": \"RMB\"}, \"FSalerId\": {\"FNumber\": \"B027\"}, \"FBusinessType\": \"NORMAL\", \"FSaleOrderFinance\": {\"FEXCHANGETYPE\": {\"FNumber\": \"NCIC\"}, \"FExchangeRate\": \"1\"}, \"FSaleOrderEntry\": [{\"FMATERIALID\": {\"FNumber\": \"A.0060480931\"}, \"FUNITID\": {\"FNumber\": \"03\"}, \"FQty\": \"1\", \"FTaxPrice\": \"100\", \"FEntryTaxRate\": \"17.00\"}], {\"FID\": \"0\", \"FBILLTYPEID\": {\"FNumber\": \"XSDD01_SYS\"}, \"FSaleOrgId\": {\"FNumber\": \"NCIC\"}, \"FSaleDeptId\": {\"FNumber\": \"01.A13\"}, \"FCustId\": {\"FNumber\": \"500\"}, \"FDATE\": \"2016/2/23\", \"FSettleCurrId\": {\"FNumber\": \"RMB\"}, \"FSalerId\": {\"FNumber\": \"B027\"}, \"FBusinessType\": \"NORMAL\", \"FSaleOrderFinance\": {\"FEXCHANGETYPE\": {\"FNumber\": \"NCIC\"}, \"FExchangeRate\": \"1\"}, \"FSaleOrderEntry\": [{\"FMATERIALID\": {\"FNumber\": \"A.0060480931\"}, \"FUNITID\": {\"FNumber\": \"03\"}, \"FQty\": \"1\", \"FTaxPrice\": \"100\", \"FEntryTaxRate\": \"17.00\"}]}]}, {\"FID\": \"0\", \"FBILLTYPEID\": {\"FNumber\": \"XSDD01_SYS\"}, \"FSaleOrgId\": {\"FNumber\": \"NCIC\"}, \"FSaleDeptId\": {\"FNumber\": \"01.A13\"}, \"FCustId\": {\"FNumber\": \"500\"}, \"FDATE\": \"2016/2/23\", \"FSettleCurrId\": {\"FNumber\": \"RMB\"}, \"FSalerId\": {\"FNumber\": \"B027\"}, \"FBusinessType\": \"NORMAL\", \"FSaleOrderFinance\": {\"FEXCHANGETYPE\": {\"FNumber\": \"NCIC\"}, \"FExchangeRate\": \"1\"}, \"FSaleOrderEntry\": [{\"FMATERIALID\": {\"FNumber\": \"A.0060480931\"}, \"FUNITID\": {\"FNumber\": \"03\"}, \"FQty\": \"1\", \"FTaxPrice\": \"100\", \"FEntryTaxRate\": \"17.00\"}]}]}]\";
object[] saveInfo = new object[]
{
    {
        sFormId,
        sContent
    }
};
var ret =
client.Execute<string>(\"Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.BatchSave\", saveInfo);
```

2) 无引用组件示例 (不引用金蝶的组件):

```
HttpClient httpClient = new HttpClient();
httpClient.Url =
\"http://192.168.66.60/k3cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.BatchSave.common.kdsvc\";
```

```
List<object> Parameters = new List<object>();  
//业务对象Id  
String formid = "SAL_SaleOrder";//销售订单为例  
Parameters.Add(formid);  
//Json字符串  
string data =  
"{\"NeedUpdateFields\": [], \"BatchCount\": \"2\", \"Model\": [{\"FID\": \"0\", \"FBILLTYPEID\": {\"FNumber\": \"XSDD01_SYS\"}, \"FSaleOrgId\": {\"FNumber\": \"NCIC\"}, \"FSaleDeptId\": {\"FNumber\": \"01.A13\"}, \"FCustId\": {\"FNumber\": \"500\"}, \"FDATE\": \"2016/2/23\", \"FSettleCurrId\": {\"FNumber\": \"RMB\"}, \"FSalerId\": {\"FNumber\": \"B027\"}, \"FBusinessType\": \"NORMAL\", \"FSaleOrderFinance\": {\"FEXCHANGETYPE\": {\"FNumber\": \"NCIC\"}, \"FExchangeRate\": \"1\"}, \"FSaleOrderEntry\": [{\"FMATERIALID\": {\"FNumber\": \"A.0060480931\"}, \"FUNITID\": {\"FNumber\": \"03\"}, \"FQty\": \"1\", \"FTaxPrice\": \"100\", \"FEntryTaxRate\": \"17.00\"}], {\"FID\": \"0\", \"FBILLTYPEID\": {\"FNumber\": \"XSDD01_SYS\"}, \"FSaleOrgId\": {\"FNumber\": \"NCIC\"}, \"FSaleDeptId\": {\"FNumber\": \"01.A13\"}, \"FCustId\": {\"FNumber\": \"500\"}, \"FDATE\": \"2016/2/23\", \"FSettleCurrId\": {\"FNumber\": \"RMB\"}, \"FSalerId\": {\"FNumber\": \"B027\"}, \"FBusinessType\": \"NORMAL\", \"FSaleOrderFinance\": {\"FEXCHANGETYPE\": {\"FNumber\": \"NCIC\"}, \"FExchangeRate\": \"1\"}, \"FSaleOrderEntry\": [{\"FMATERIALID\": {\"FNumber\": \"A.0060480931\"}, \"FUNITID\": {\"FNumber\": \"03\"}, \"FQty\": \"1\", \"FTaxPrice\": \"100\", \"FEntryTaxRate\": \"17.00\"}}]}]\"};  
Parameters.Add(data);  
httpClient.Content = JsonConvert.SerializeObject(Parameters);  
var result = httpClient.AsyncRequest();
```

5.1.5. 提交表单数据接口

服务地址:

<http://ServerIp/K3Cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Submit.common.kdsvc>

接口参数:

参数列表	参数含义	备注
formid	表单 Id, 格式参考: "BD_Currency"//表示币别	必须
data	数据包, Number 表示单据编号或者基础资料编码, Id 为主键, CreateOrgId 为创建组织 Id。数据格式参考以币别为例: {"CreateOrgId":0,"UseOrgId":0,"Numbers":["PRE002","PRE006"]}	CreateOrgId: 创建组织 Id, 非必须 UseOrgId: 使用组织 Id, 非必须 Numbers: 必须

返回参数:

参数列表	参数含义	备注
ResponseStatus	操作状态 {"ResponseStatus":{"ErrorCode":"","IsSuccess":false,"Errors":[{"FieldName":"","Message":"","DIndex":0}], "SuccessEntitys":[{"Id":"","Number":"","DIndex":0}], "SuccessMessages":[{"FieldName":"","Message":"","DIndex":0}]}	IsSuccess: 操作是否成功, Errors: 如果失败, 具体失败原因 DIndex: 原始数据行号

调用参考:

1) SDK 辅助类示例 (引用 Kingdee.BOS.WebAPI.Client.dll):

```
string sJson = "{\"CreateOrgId\":0,\"Numbers\":[\"PRE002\",\"PRE006\"]}";
var result =
client.Execute<string>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Submit",
new object[] { "BD_Currency", sJson });
```

2) 无引用组件示例 (不引用金蝶的组件):

```
HttpClient httpClient = new HttpClient();
httpClient.Url =
"http://192.168.66.60/k3cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Submit.common.kdsvc";
List<object> Parameters = new List<object>();
//业务对象Id
String formid = "BD_Currency";//币别为例
Parameters.Add(formid);
//Json字符串
string data = "{\"CreateOrgId\":0,\"Numbers\":[\"PRE002\",\"PRE006\"]}";
Parameters.Add(data);
httpClient.Content = JsonConvert.SerializeObject(Parameters);
```

```
var result = httpClient.AsyncRequest();
```

5.1.6. 审核表单数据接口

服务地址:

<http://ServerIp/K3Cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Audit.common.kdsvc>

接口参数:

参数列表	参数含义	备注
formid	表单 Id, 格式参考: "BD_Currency"//表示币别	必须
data	数据包, Number 表示单据编号或者基础资料编码, Id 为主键, CreateOrgId 为创建组织 Id。数据格式参考以币别为例: "{\"CreateOrgId\":0,\"UseOrgId\":0,\"Numbers\":[\"PRE002\",\"PRE006\"]}"	CreateOrgId: 创建组织 Id, 非必须 UseOrgId: 使用组织 Id, 非必须 Numbers: 必须

返回参数:

参数列表	参数含义	备注
ResponseStatus	操作状态 { "ResponseStatus": { "ErrorCode": "", "IsSuccess": false, "Errors": [{ "FieldName": "", "Message": "", "DIndex": 0 }], "SuccessEntitys": [{ "Id": "", "Number": "", "DIndex": 0 }], "SuccessMessages": [{ "FieldName": "", "Message": "", "DIndex": 0 }] }	IsSuccess: 操作是否成功, Errors: 如果失败, 具体失败原因 DIndex: 原始数据行号

1) SDK 辅助类示例 (引用 Kingdee.BOS.WebAPI.Client.dll):

```
string sJson = "{\"CreateOrgId\":0,\"Numbers\":[\"PRE002\",\"PRE006\"]}";  
var result =  
client.Execute<string>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Audit",  
new object[] { "BD_Currency", sJson });
```

3) 无引用组件示例 (不引用金蝶的组件):

```
HttpClient httpClient = new HttpClient();  
httpClient.Url =
```

```
"http://192.168.66.60/k3cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Audit.common.kdsvc";

List<object> Parameters = new List<object>();
//业务对象Id
String formid = "BD_Currency";//币别为例
Parameters.Add(formid);

//Json字符串
string data = "{\"CreateOrgId\":0,\"Numbers\":[\"PRE002\\\", \"PRE006\\\"]}";
Parameters.Add(data);
httpClient.Content = JsonConvert.SerializeObject(Parameters);
var result = httpClient.AsyncRequest();
```

5.1.7. 反审核表单数据接口

服务地址:

<http://ServerIp/K3Cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.UnAudit.common.kdsvc>

接口参数:

参数列表	参数含义	备注
formid	表单 Id, 格式参考: "BD_Currency"//表示币别	必须
data	数据包, Number 表示单据编号或者基础资料编码, Id 为主键, CreateOrgId 为创建组织 Id。数据格式参考以币别为例: "{\"CreateOrgId\":0,\"UseOrgId\":0,\"Numbers\":[\"PRE002\\\", \"PRE006\\\"]}"	CreateOrgId: 创建组织 Id, 非必须 UseOrgId: 使用组织 Id, 非必须 Numbers: 必须

返回参数:

参数列表	参数含义	备注
ResponseStatus	操作状态 {"ResponseStatus":{"ErrorCode":"","IsSuccess":false,"Errors":[{"FieldName":"","Message":"","DIndex":0}], "SuccessEntitys":[{"Id":"","Number":"","DIndex":0}], "SuccessMessages":[{"FieldName":"","Message":"","DIndex":0}]}	IsSuccess: 操作是否成功, Errors: 如果失败, 具体失败原因 DIndex: 原始数据行号

1) SDK 辅助类示例（引用 Kingdee.BOS.WebAPI.Client.dll）：

```
string sJson = "{\"CreateOrgId\":0,\"Numbers\":[\"PRE002\",\"PRE006\"]}";  
var result =  
client.Execute<string>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.UnAudit", new object[] { "BD_Currency", sJson });
```

4) 无引用组件示例（不引用金蝶的组件）：

```
HttpClient httpClient = new HttpClient();  
httpClient.Url =  
"http://192.168.66.60/k3cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.UnAudit.common.kdsvc";  
List<object> Parameters = new List<object>();  
//业务对象Id  
String formid = "BD_Currency";//币别为例  
Parameters.Add(formid);  
//Json字符串  
string data = "{\"CreateOrgId\":0,\"Numbers\":[\"PRE002\",\"PRE006\"]}";  
Parameters.Add(data);  
httpClient.Content = JsonConvert.SerializeObject(Parameters);  
var result = httpClient.AsyncRequest();
```

5.1.8. 删除表单数据接口

服务地址：

<http://ServerIp/K3Cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Delete.common.kdsvc>

接口参数：

参数列表	参数含义	备注
formid	表单 Id，格式参考：“BD_Currency”//表示币别	必须
data	数据包，Number 表示单据编号或者基础资料编码，Id 为主键，CreateOrgId 为创建组织 Id。数据格式参考以币别为例： "{\"CreateOrgId\":0,\"UseOrgId\":0,\"Numbers\":[\"PRE002\",\"PRE006\"]}"	CreateOrgId: 创建组织 Id, 非必须 UseOrgId: 使用组织 Id, 非必须 Numbers: 必须

返回参数：

参数列表	参数含义	备注
ResponseStatus	操作状态 {"ResponseStatus":{"ErrorCode":""," "IsSuccess":false, "Errors":[{"FieldName":""," "Message":"","DIndex":0}], "SuccessEntitys":[{"Id":"","Number":"","DIndex":0}], "SuccessMessages":[{"FieldName":"","Message":"","DIndex":0}]}	IsSuccess:操作是否成功, Errors:如果失败,具体失败原因 DIndex:原始数据行号

1) SDK 辅助类示例 (引用 Kingdee.BOS.WebAPI.Client.dll):

```
string sJson = "{\"CreateOrgId\":0,\"Numbers\":[\"PRE002\",\"PRE006\"]}";
var result =
client.Execute<string>("Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Delete",
new object[] { "BD_Currency", sJson });
```

5) 无引用组件示例 (不引用金蝶的组件):

```
HttpClient httpClient = new HttpClient();
httpClient.Url =
"http://192.168.66.60/k3cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.Delete.common.kdsvc";
List<object> Parameters = new List<object>();
//业务对象Id
String formid = "BD_Currency";//币别为例
Parameters.Add(formid);
//Json字符串
string data = "{\"CreateOrgId\":0,\"Numbers\":[\"PRE002\",\"PRE006\"]}";
Parameters.Add(data);
httpClient.Content = JsonConvert.SerializeObject(Parameters);
var result = httpClient.AsyncRequest();
```

5.1.9. 表单数据查询接口

服务地址:

<http://ServerIp/K3Cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormService.ExecuteBillQuery.common.kdsvc>

接口参数:

参数列表	参数含义	备注
------	------	----

data	<p>数据包格式参考:</p> <pre>{\"FormId\": \"PUR_PurchaseOrder\", \"TopRowCount\": 0, \"Limit\": 10, \"StartRow\": 0, \"FilterString\": \"FMaterialId.FNumber='HG_TEST'\", \"OrderString\": \"FID ASC\", \"FieldKeys\": \"FID, FSupplierId, FMaterialId, FMaterialId.FNumber, FMaterialName\"}</pre>	<p>FormId: 查询表单元数据唯一标识</p> <p>TopRowCount: 最多允许查询的数量, 0 或者不要此属性表示不限制</p> <p>Limit: 分页取数每页允许获取的数据, 最大不能超过 2000</p> <p>StartRow: 分页取数开始行索引, 从 0 开始, 例如每页 10 行数据, 第 2 页开始是 10, 第 3 页开始是 20</p> <p>FilterString: 过滤条件</p> <p>OrderString: 排序条件</p> <p>FieldKeys: 待查询表单的字段列表</p>
------	---	--

返回参数:

调用参考:

1) SDK 辅助类示例 (引用 Kingdee.BOS.WebAPI.Client.dll):

```
object[] paramInfo = new object[]
{
    \"{\"FormId\": \"PUR_PurchaseOrder\", \"+// 采购订单formid
    \"{\"TopRowCount\": 0, \"+// 最多允许查询的数量, 0或者不要此属性表示不限
    \"{\"Limit\": 10, \"+// 分页取数每页允许获取的数据, 最大不能超过2000
    \"{\"StartRow\": 0, \"+// 分页取数开始行索引, 从0开始, 例如每页10行数据,
    第2页开始是10, 第3页开始是20
    \"{\"FilterString\": \"FMaterialId.FNumber='HG_TEST'\", \"+// 过滤
    条件
    \"{\"OrderString\": \"FID ASC\", \"+// 排序条件
    \"{\"FieldKeys\": \"FID, FSupplierId, FMaterialId, FMaterialId.FNumber, FMaterialNa
    me\"}\"\"// 获取采购订单数据参数, 内码, 供应商id, 物料id, 物料编码, 物料名称
};
//调用查询接口
List<List<object>> ret =
client.Execute<List<List<object>>>(\"Kingdee.BOS.WebApi.ServicesStub.DynamicF
ormService.ExecuteBillQuery.common.kdsvc\", paramInfo);
if (ret != null && ret.Count > 0)
{
    // TODO:
}
```

2) 无引用组件示例 (不引用金蝶的组件):

```
HttpClient httpClient = new HttpClient();
httpClient.Url =
"http://192.168.66.60/k3cloud/Kingdee.BOS.WebApi.ServicesStub.DynamicFormSer
vice.ExecuteBillQuery.common.kdsvc";
```

```
List<object> Parameters = new List<object>();  
  
//Json字符串  
string data =  
"{\"FormId\": \"PUR_PurchaseOrder\", \"TopRowCount\": 0, \"Limit\": 10, \"StartRow\" : 0, \"FilterString\": \"FMaterialId.FNumber='HG_TEST'\", \"OrderString\": \"FID  
D  
ASC\", \"FieldKeys\": \"FID, FSupplierId, FMaterialId, FMaterialId.FNumber, FMater  
ialName\"}";  
  
Parameters.Add(data);  
httpClient.Content = JsonConvert.SerializeObject(Parameters);  
  
var result = httpClient.AsyncRequest();
```

5.1.10. 自定义 WebAPI 接口

服务地址:

<http://ServerIp/K3Cloud/接口命名空间.接口实现类名.方法,组件名.common.kdsvc>

例如:

"http://192.168.66.60/k3cloud/[ApiServiceTest.AAA.CustomBusinessService.ExecuteService, ApiServiceTest.AAA.common.kdsvc](#)"

需要注意的是, 自定义 webapi 接口服务地址中比其他标准接口地址多了一个组件名, 如果需要和其他其他标准接口保持一致, 则二开自定义的 webapi 接口命名空间必须以.ServicesStub 结尾, 例如以下示例自定义接口则其服务地址可以参考如下格式:

"http://192.168.66.60/k3cloud/[Kingdee.K3Erp.WebAPI.ServiceExtend.ServicesStub.CustomBusinessService.ExecuteService.common.kdsvc](#)"

/*

* 服务需要引用组件如下

* Kingdee.BOS.dll、

Kingdee.BOS.ServiceFacade.KDServiceFx.dll、

Kingdee.BOS.WebApi.ServicesStub.dll

*/

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using Kingdee.BOS.ServiceFacade.KDServiceFx;

using Kingdee.BOS.WebApi.ServicesStub;

using Kingdee.BOS.JSON;

namespace Kingdee.K3Erp.WebAPI.ServiceExtend.ServicesStub

{

public class CustomBusinessService : AbstractWebApiBusinessService

{

```
public CustomBusinessService(KDServiceContext context)
    : base(context)
{
}
public JSONArray ExecuteService(string parameter)
{
    JSONArray jsonArray = new JSONArray();
    jsonArray.Add("Hello World");
    return jsonArray;
}
}
```

接口参数：

自定义。

返回参数：

自定义。

调用参考：

1) SDK 辅助类示例（引用 Kingdee.BOS.WebAPI.Client.dll）：

// 此接口为 K3Cloud 自定义接口，命名控件及类名仅供参考。

```
using Kingdee.BOS.Core.Metadata;
```

```
using Kingdee.BOS.Core.SqlBuilder;
```

```
using Kingdee.BOS.JSON;
```

```
using Kingdee.BOSOrm.DataEntity;
```

```
using Kingdee.BOSOrm.Metadata.DataEntity;
```

```
using Kingdee.BOS.ServiceHelper;
```

```
using System;
```

```
using System.Collections.Generic;
```

```
using System.Linq;
```

```
using System.Text;
```

```
namespace ApiServiceTest.AAA
```

```
{
```

```
    /// <summary>
```

```
    ///
```

```
    /// </summary>
```

```
    public class CustomBusinessService :
```

```
Kingdee.BOS.WebApi.ServicesStub.AbstractWebApiBusinessService
```

```
{
```

```
    public
```

```
CustomBusinessService(Kingdee.BOS.ServiceFacade.KDServiceFx.KDServiceContext
context)
```

```
    : base(context)
```

```
{ }
```

```
    /// <summary>
```



```
/// 执行服务，根据参数来处理自己的业务并返回结果
/// </summary>
/// <param name="ctx"></param>
/// <param name="parameter"></param>
/// <returns></returns>
public JSONArray ExecuteService(string parameter)
{
    // TODO: 在此方法中处理业务，可以引用 Kingdee 现有接口，根据传入的参数来实
    现自己的业务逻辑，
    // 例如查询数据、单据转换、甚至数据同步等都可以
    // 以下为查询币别信息的示例代码
    // 构造查询参数
    QueryBuilderParameter para = new QueryBuilderParameter()
    {
        // 查询表单，测试用例中直接传入了“BD_Currency”
        FormId = parameter,
        // 需要查询币别主键、币别名称等信息，此处用字段的标识来构造
        SelectItems = SelectorItemInfo.CreateItems("FCURRENCYID,FName"),
    };
    // 调用查询接口
    DynamicObjectCollection list =
    QueryServiceHelper.GetDynamicObjectCollection(this.KDContext.Session.AppContext,
    para);

    // 返回结果类型可以自定义
    // 如果第三方系统不想引用 BOS 其他组件的，可以使用 json 字符串、List、
    Dictionary 等类型做为返回结果
    // 此处示例代码使用 BOS 定义的类型 JSONArray
    JSONArray jsonArray = new JSONArray();
    foreach (DynamicObject dynamicObject in list)
    {
        jsonArray.Add(ConvertDynamicObject2Json(dynamicObject));
    }
    return jsonArray;
}

/// <summary>
/// 自定义服务接口
/// </summary>
/// <param name="parameter"></param>
/// <param name="formId"></param>
/// <returns></returns>
public bool UpdateData(string parameter, string formId)
{
    return true;
}
```

```
}

private JSONObject ConvertDynamicObject2Json(DynamicObject dynamicObject)
{
    DynamicPropertyCollection dynamicPropertyCollection =
dynamicObject.DynamicObjectType.Properties;
    JSONObject obj = new JSONObject();
    foreach (DynamicProperty property in dynamicPropertyCollection)
    {
        obj.Put(property.Name, dynamicObject[property.Name]);
    }
    return obj;
}
}
```

```
var responseOut =
client.Execute<JSONArray>("ApiServiceTest.AAA.CustomBusinessService.ExecutesS
ervice,ApiServiceTest.AAA", new object[] { "BD_Currency" });
```

2) 无引用组件示例（不引用金蝶的组件）:

```
HttpClient httpClient = new HttpClient();
httpClient.Url =
string.Concat("http://192.168.66.60/k3cloud/ApiServiceTest.AAA.CustomBusines
sService.ExecuteService,ApiServiceTest.AAA.common.kdsvc");
List<object> Parameters = new List<object>();
Parameters.Add("BD_Currency");
httpClient.Content = JsonConvert.SerializeObject(Parameters);
var responseOut = httpClient.AsyncRequest();
```

6. 附录（集成相关知识分享）

Web API 方式相关链接置顶:

.net

K/3 Cloud Web API 集成开发客户端不引用组件示例【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=939582>

Java

K/3 Cloud Web API 销售订单 Java 完整示例【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=959863>

php

K/3 Cloud Web API 销售出库单 PHP 完整示例【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=986575>

WebApi 之新接口推广

<http://club.kisdee.com/forum.php?mod=viewthread&tid=752737>

K/3 Cloud WebAPI 接口说明文档

<http://club.kisdee.com/forum.php?mod=viewthread&tid=714662>

WebAPI【财务凭证】核算维度赋值【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=989427>

WebAPI 查询表单数据接口示例

<http://club.kisdee.com/home.php?mod=space&uid=421001&do=blog&id=119707>

WebAPI 批量保存接口示例

<http://club.kisdee.com/home.php?mod=space&uid=421001&do=blog&id=119714>

小技巧 - 如何逐步构建采购订单 Web API 保存接口参数

<http://club.kisdee.com/forum.php?mod=viewthread&tid=992078>

K3Cloud 系统集成实现下推单据转换调用【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=763172>

K/3Cloud WebAPI 调用任意操作实现方案【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=994209>

注意：WebAPI 方式调用是不需要发布站点的，需要发布站点的就不是 WebAPI 方式

—————置顶完毕 分割线—————

发布 SI 站点：

开单贴，K/3Cloud 系统集成配置及示例

<http://club.kisdee.com/forum.php?mod=viewthread&tid=566720>

发布 SI 站点出错解决：

【已解决】5.0 集成站点发布失败!!!

<http://club.kisdee.com/forum.php?mod=viewthread&tid=713356>

SI 的 Web Service 方式:

Web Service 方式 【销售普通发票】保存【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=708668>

Web Service 不引用 SDK 方式 新增【用户】 【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=716238>

SI 的 Json 格式:

K3Cloud 系统集成 Json 方式调用【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=700412>

采用 Json 方式集成调用批量查询数据【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=700648>

系统集成构造 Json 格式包含子单据体示例【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=751588>

Web Service 上传附件 Json 格式【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=937047>

Web API 方式:

K/3 Cloud WebAPI 接口说明文档

<http://club.kisdee.com/forum.php?mod=viewthread&tid=714662>

WebApi 之新接口推广

<http://club.kisdee.com/forum.php?mod=viewthread&tid=752737>

K/3 Cloud Web API 集成开发客户端不引用组件示例【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=939582>

WebAPI【财务凭证】核算维度赋值【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=989427>

PHP 方式调用:

Cloud 系统集成 PHP 完整调用示例【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=741699>

K/3 Cloud Web API 销售出库单 PHP 完整示例【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=986575>

Java 方式调用:

Cloud 系统集成 Java 完整调用示例【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=741827>

K/3 Cloud Web API 集成开发 Java 完整示例【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=959863>

另:

基于 Cloud 代理组件方式集成:

基于 .net Framework4.0 第三方通过 Cloud 代理组件集成开发【分享】

<http://club.kisdee.com/forum.php?mod=viewthread&tid=722365>

硬件集成:

【分享】客制化控件插件实现客户端设备集成

<http://club.kisdee.com/forum.php?mod=viewthread&tid=642724>

论坛资源入口

(2016-11-10 17:11)

<http://club.kisdee.com/forum.php?mod=viewthread&tid=741861>

K/3 Cloud 的系统集成从新发布站点角度看

需要新发布站点: 发布 K3CloudServiceInterface 站点方式集成调用(简称 SI 方式)其中可用 Json Soap1.1 Soap1.2 格式

无需新发布站点：直接使用业务站点，Web API 方式集成调用 Http+Json 格式（推荐的系统集成方式）

推荐使用 **Web API 方式**（在 5.0 版本全新支持的，其实 3.0 版本的后期几个补丁也有做了支持），

Web API 方式

轻量级的开发接口，
基础资料与单据类型直接调用，
公用业务站点组件，
无需特别实施，
开发维护成本小。

又有最近 5.0 的 2 个补丁（PT100226、PT100068），及 6.0 专门有对 Web API 方式做性能优化。

继续推荐使用 **Web API 方式**。

Web API 方式相关链接置顶：

.net

K/3 Cloud Web API 集成开发客户端不引用组件示例【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=939582>

Java

K/3 Cloud Web API 销售订单 Java 完整示例【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=959863>

java 版本的 K/3Cloud webapi 客户端 sdk (K/3 Cloud webapi client sdk for java)

<http://club.kingdee.com/forum.php?mod=viewthread&tid=1137264>

php

K/3 Cloud Web API 销售出库单 PHP 完整示例【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=986575>

K/3 Cloud WebAPI 最新接口说明文档

<http://club.kingdee.com/forum.php?mod=viewthread&tid=1122158>

K/3 Cloud WebAPI 接口说明文档

<http://club.kingdee.com/forum.php?mod=viewthread&tid=714662>

WebApi 之新接口推广

<http://club.kingdee.com/forum.php?mod=viewthread&tid=752737>

WebAPI【财务凭证】核算维度赋值【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=989427>

WebAPI 查询表单数据接口示例

<http://club.kingdee.com/home.php? ... 1&do=blog&id=119707>

WebAPI 批量保存接口示例

<http://club.kingdee.com/home.php? ... 1&do=blog&id=119714>

小技巧 - 如何逐步构建采购订单 Web API 保存接口参数

<http://club.kingdee.com/forum.php?mod=viewthread&tid=992078>

K3Cloud 系统集成实现下推单据转换调用【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=763172>

K/3Cloud WebAPI 调用任意操作实现方案【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=994209>

注意：WebAPI 方式调用是不需要发布站点的，需要发布站点的就不是 WebAPI 方式

———置顶完毕 分割线———

发布 SI 站点：

开单贴，K/3Cloud 系统集成配置及示例

<http://club.kingdee.com/forum.php?mod=viewthread&tid=566720>

发布 SI 站点出错解决：

【已解决】5.0 集成站点发布失败!!!

<http://club.kingdee.com/forum.php?mod=viewthread&tid=713356>

SI 的 Web Service 方式：

Web Service 方式 【销售普通发票】保存【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=708668>

Web Service 不引用 SDK 方式 新增【用户】 【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=716238>

SI 的 Json 格式：

K3Cloud 系统集成 Json 方式调用【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=700412>

采用 Json 方式集成调用批量查询数据【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=700648>

系统集成构造 Json 格式包含子单据体示例【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=751588>

Web Service 上传附件 Json 格式【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=937047>

Web API 方式：

K/3 Cloud WebAPI 最新接口说明文档

<http://club.kingdee.com/forum.php?mod=viewthread&tid=1122158>

K/3 Cloud WebAPI 接口说明文档

<http://club.kingdee.com/forum.php?mod=viewthread&tid=714662>

WebApi 之新接口推广

<http://club.kingdee.com/forum.php?mod=viewthread&tid=752737>

K/3 Cloud Web API 集成开发客户端不引用组件示例【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=939582>

WebAPI【财务凭证】核算维度赋值【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=989427>

PHP 方式调用:

Cloud 系统集成 PHP 完整调用示例【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=741699>

K/3 Cloud Web API 销售出库单 PHP 完整示例【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=986575>

Java 方式调用:

Cloud 系统集成 Java 完整调用示例【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=741827>

K/3 Cloud Web API 集成开发 Java 完整示例【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=959863>

另:

基于 Cloud 代理组件方式集成:

基于 .net Framework4.0 第三方通过 Cloud 代理组件集成开发【分享】

<http://club.kingdee.com/forum.php?mod=viewthread&tid=722365>

硬件集成:

【分享】客制化控件插件实现客户端设备集成

<http://club.kingdee.com/forum.php?mod=viewthread&tid=642724>