

Article  
[王喆](#) · Oct 18, 2022 4m read

# How IRIS performs CRUD operations



## preface

Since the contact with IRIS, it has been used most in hospitals as an ESB integration engine, as a link in the hospital's integration platform+data center. However, you can also develop some restful pis to be used as the backend of front end and back end separation projects

## List of articles

Introduction

## List of articles

- I. Development of technologies and tools
- II. Development Path and Relevant Codes
  1. database
  2. Preparatory work
  3. Steps associated with developing an interface
  4. Presentation of achievements

Summary

This article will show the CRUD operations related to IRIS in a simple page :



Yes, I plan to reconstruct the Production page of IRIS to show the results I have developed :



呃的 It's like a dime, ha ha

## I. Development of technologies and tools

If you want to do a good job, you must first use your tools:

Back End: HealthConnect+Global

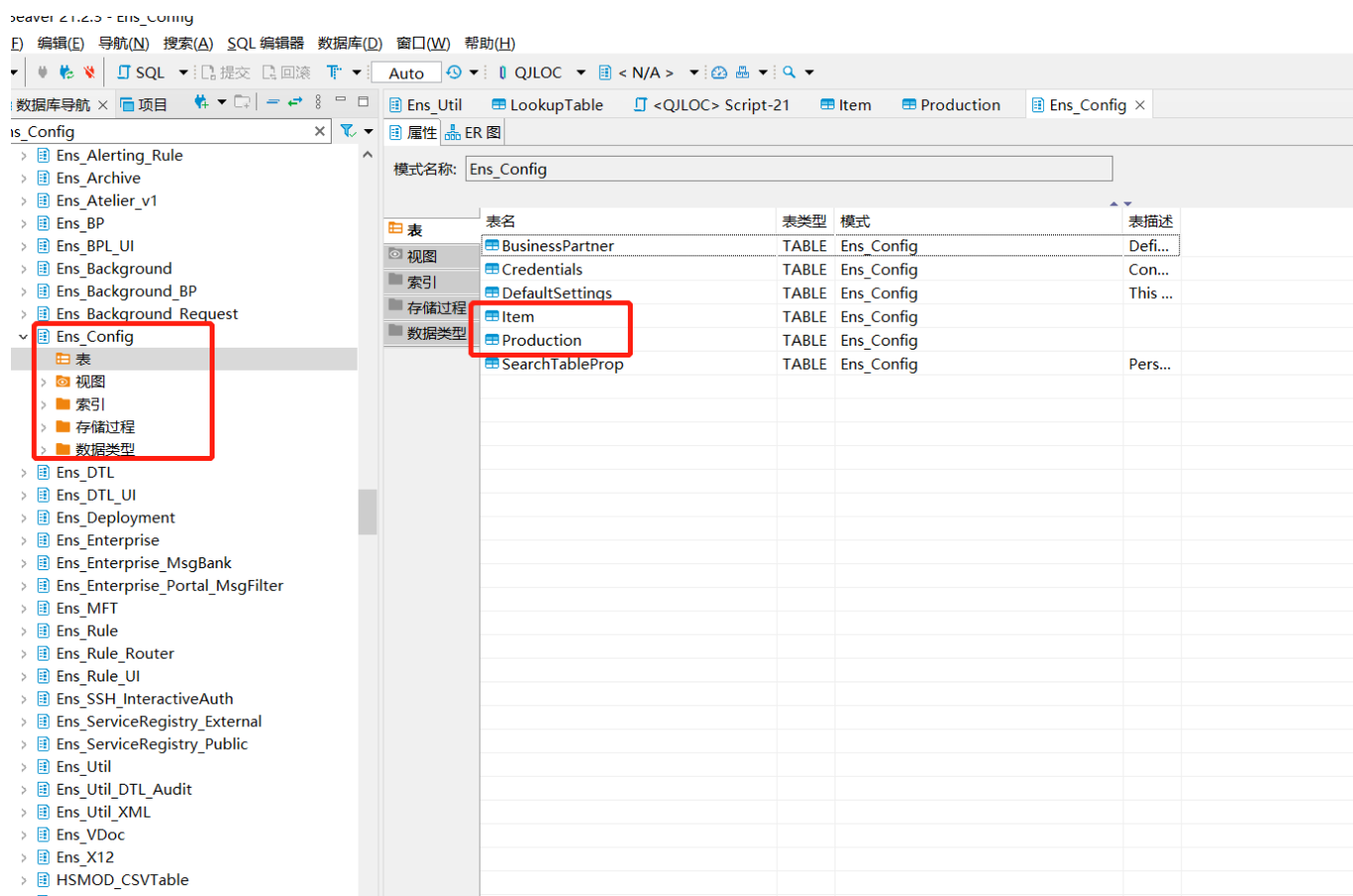
Front End: vue

Use the Rest API interface developed by %CSP. REST of IRIS. The front-end uses vue to simply display the interface related content. You can edit and add components to control the start, stop and update detection status of production

## II. Development Path and Relevant Codes

### 1. database

First, we find the related tables of Production, which are: Ens\_Config.Item and Ens\_Config.Production, as shown in the figure



So the idea of the code is very clear. Add, delete, modify and query [Ens\_Config. Item].

Query [Ens\_Config. Production]

## 2. Preparatory work

We have to do some preparatory work before the formal development: such as an open gateway

- Create an API class, as shown in the figure :

```
k > src > BKIP > Rest > api.cls > BKIP.Rest.api > UrlMap
/// Date: 2022年7月20日 10:00 <br/>
/// Author: 王喆 <br/>
Class BKIP.Rest.api Extends %CSP.REST
{
Parameter SpecificationClass = "petstore.spec";
/// 此应用程序的默认内容类型。
Parameter CONTENTTYPE = "application/json";
/// 默认情况下, 将输入流转换为Unicode
Parameter CONVERTINPUTSTREAM = 1;
/// 默认响应字符集是utf-8
Parameter CHARSET = "utf-8";
XData UrlMap [ XMLNamespace = "http://www.intersystems.com/urlmap" ]
{
<Routes>
<Route Url="/" Method="GET" Call="test" Cors="true" />
</Routes>
}

/// 测试使用
Debug this method
ClassMethod test() As %Status
{
Set sc = $$$OK
w "测试成功"
Return sc
}
}
```

· Enter the HealthConnect management page: System Management - Security - Application - Web Application, as shown in the figure



To create a new web application, I define the class controlled by the api in the [/api] code as BKIP.Rest.api



Give a [ALL] 's role

系统 > 安全管理 > Web 应用程序 > 编辑 Web 应用程序 - (安全设置)

## 编辑 Web 应用程序

保存 取消

为 Web 应用程序 /api 编辑角色:

常规 应用程序角色 匹配角色

用户进入此应用程序时,以下角色将自动添加到当前角色集:

应用程序角色	
%All	移除

通过选择一个或多个可用角色并按 [分配],添加额外的应用程序角色.

可用

----- 选择一个或多个 -----

- %DB\_%DEFAULT
- %DB\_ENSLIB
- %DB\_HSCUSTOM
- %DB\_HSLIB
- %DB\_HSSYS
- %DB\_IRISAUDIT
- %DB\_IRISLIB
- %DB\_IRISLOCALDATA
- %DB\_IRISYS
- %DB\_IRISTEMP
- %DB\_USER
- %Developer
- %EnsRole\_Administrator
- %EnsRole\_AlertAdministrator
- %EnsRole\_AlertOperator

已选择

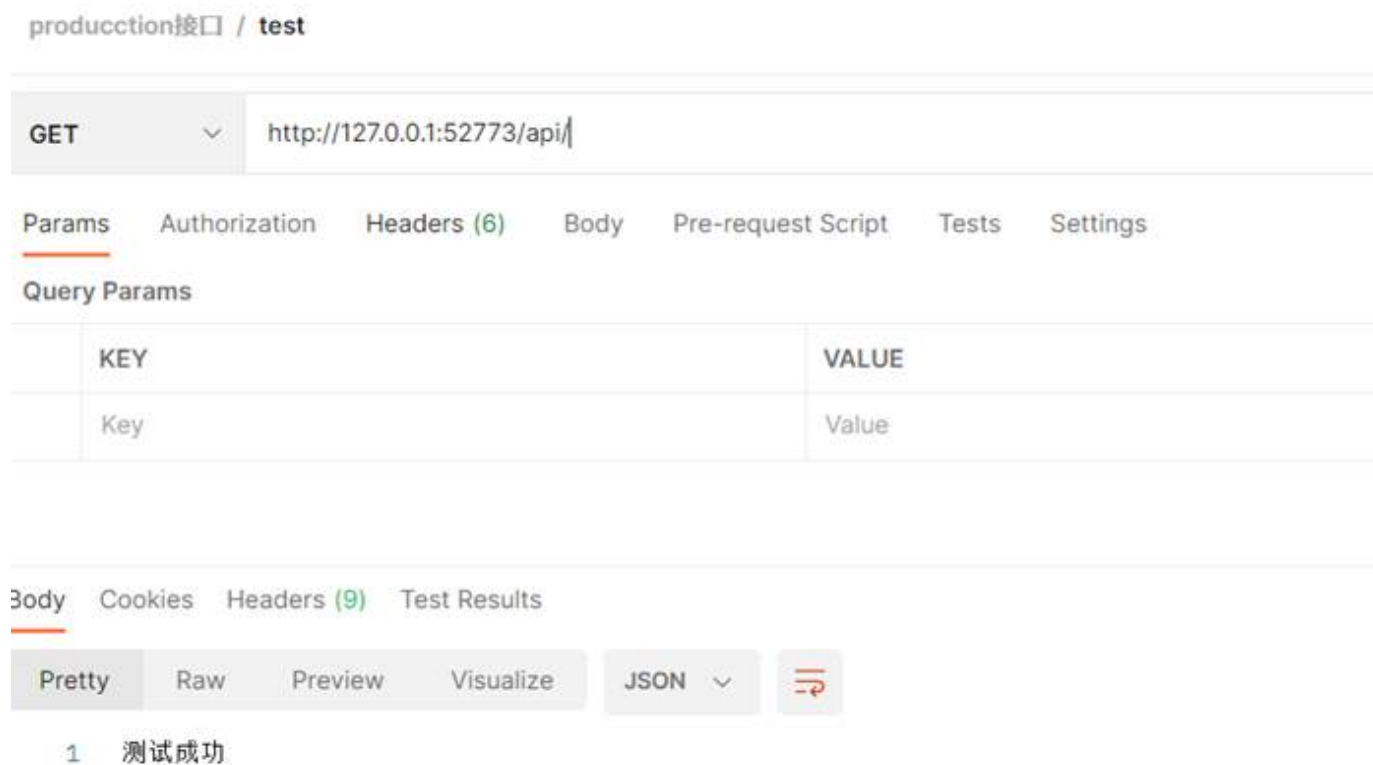
----- 选择一个或多个 -----

分配

按住 [Shift] 或 [Ctrl] 键的同时单击以选择多个角色.

Test it:





So far, we've opened the first Api interface, but also open the gateway for our programs.

### 3.Steps related to interface developmen

Interface development is divided into four steps: defining entity classes, organizing SQL, writing methods to receive result sets, organizing returns, and providing restfu

#### (1) Define entity class

- Production Entity:Production

```

> src > BKIP > Production > Message > Production.cls > BKIP.Production.Message.Production
Class BKIP.Production.Message.Production Extends (%SerialObject, %XML.Adaptor)
{

    /// XMLNAME
    Parameter XMLNAME = "Production";

    Parameter XMLIGNORENULL = 1;

    /// ID
    Property ID As %String;

    /// 名称
    Property Name As %String(MAXLEN = 128, XMLPROJECTION = "ATTRIBUTE");

    /// 描述
    Property Description As %String(MAXLEN = 500, TRUNCATE = 1);

    /// 参与者池大小
    Property ActorPoolSize As %Integer(MAXVAL = 128, MINVAL = 0) [ InitialExpression = 1 ];

    /// 测试是否开启 1-开启 0-关闭
    Property TestingEnabled As %Boolean(XMLPROJECTION = "ATTRIBUTE");

    /// 记录普通跟踪事件 是否开启 1-开启 0-关闭
    Property LogGeneralTraceEvents As %Boolean(XMLPROJECTION = "ATTRIBUTE") [ InitialExpression = 0 ];
}

```

```

> src > BKIP > Production > Message > Items.cls > BKIP.Production.Message.Items > XMLNAME
Class BKIP.Production.Message.Items Extends (%SerialObject, %XML.Adaptor)
{

    /// XMLNAME
    Parameter XMLNAME = "Item";

    /// ID
    Property ID As %String;

    /// 组件类型
    Property BusinessType As %Integer;

    /// 对生产对象的引用。
    Property Production As %String;

    /// 此配置项的名称。默认为类名。
    Property Name As %String;

    /// 此项目所属类别的可选列表，逗号分隔。这只用于
    /// 显示目的，不影响此项目的行为。
    Property Category As %String;

    /// 此配置项的类名。
    Property ClassName As %String;

    /// 此配置项要启动的作业数 <br>
    /// 默认值: <br>
    /// 0表示业务流程（即使用共享参与者池） <br>
    /// 1用于FIFO消息路由器业务流程（即使用专用作业） <br>
    /// 1用于业务运营 <br>
    /// 0表示无适配器业务服务 <br>
    /// 1对于其他 <br>
    /// 对于JobPerConnection=1的基于TCP的服务，如果其值大于1，则该值用于限制连接作业的数量。值0或1对连接作业的数量没有限制。
    Property PoolSize As %Integer;

    /// 是否启用此配置项。
    Property Enabled As %Boolean;
}

```

(2) Organization SQL(Only the query component is shown)



```

SELECT ID, AlertGroups, Category, ClassName, Comment, DisableErrorTraps, Enabled, Foreground, LogTraceEvents, Name, PoolSize, Production, Schedule, Settings
FROM Ens_Config.Item where 1=1
and ID = '1097'
and ClassName='BKIP.Util.JavaGateway'

```

行 #1	
ID	1,097
AlertGroups	[NULL]
Category	[NULL]
ClassName	BKIP.Util.JavaGateway
Comment	[NULL]
DisableErrorTraps	[NULL]
Enabled	1
Foreground	0
LogTraceEvents	0
Name	JavaGateway
PoolSize	0
Production	BKLINKPKG.FoundationProduction
Schedule	[NULL]
Settings	□□□□Port□□Host □□□□Address□□Host □0.0.0.0

Here we need dynamic SQL Here 's what the code says:

```

Debug this method
ClassMethod selectItemListQuery(obj As BKIP.Production.Message.dto.selectItemList) As %SQL.StatementResult

/// 1、集中处理字符串为后面判断是否为空使用
set ID = ##class(%String).LogicalToDisplay(obj.ID)
set BusinessType = ##class(%String).LogicalToDisplay(obj.BusinessType)
set Production = ##class(%String).LogicalToDisplay(obj.Production)
set Name = ##class(%String).LogicalToDisplay(obj.Name)
set Category = ##class(%String).LogicalToDisplay(obj.Category)
set ClassName = ##class(%String).LogicalToDisplay(obj.ClassName)
set PoolSize = ##class(%String).LogicalToDisplay(obj.PoolSize)
set Enabled = ##class(%String).LogicalToDisplay(obj.Enabled)
set Foreground = ##class(%String).LogicalToDisplay(obj.Foreground)
set Comment = ##class(%String).LogicalToDisplay(obj.Comment)
set LogTraceEvents = ##class(%String).LogicalToDisplay(obj.LogTraceEvents)
set AlertGroups = ##class(%String).LogicalToDisplay(obj.AlertGroups)

/// 2、SQL语句组装
set sqlStr = "SELECT ID, AlertGroups, Category, ClassName, Comment, DisableErrorTraps, Enabled, Foreground, LogTraceEvents,
Name, PoolSize, Production, Schedule, Settings FROM Ens_Config.Item where 1=1 "

If $LENGTH(ID) > 0 { s sqlStr = sqlStr_"and ID = "_ID }
If $LENGTH(ClassName) > 0 { s sqlStr = sqlStr_"and ClassName = '"_ClassName_"' " }
If $LENGTH(Production) > 0 { s sqlStr = sqlStr_"and Production = '"_Production_"' " }
If $LENGTH(AlertGroups) > 0 { s sqlStr = sqlStr_"and AlertGroups = '"_AlertGroups_"' " }
If $LENGTH(Category) > 0 { s sqlStr = sqlStr_"and Category = '"_Category_"' " }
If $LENGTH(Comment) > 0 { s sqlStr = sqlStr_"and Comment = '"_Comment_"' " }
If $LENGTH(Enabled) > 0 { s sqlStr = sqlStr_"and Enabled = '"_Enabled_"' " }
If $LENGTH(Foreground) > 0 { s sqlStr = sqlStr_"and Foreground = '"_Foreground_"' " }
If $LENGTH(LogTraceEvents) > 0 { s sqlStr = sqlStr_"and LogTraceEvents = '"_LogTraceEvents_"' " }
If $LENGTH(Name) > 0 { s sqlStr = sqlStr_"and Name = '"_Name_"' " }
If $LENGTH(PoolSize) > 0 { s sqlStr = sqlStr_"and PoolSize = '"_PoolSize_"' " }
$$$TRACE(sqlStr)
Set rs = ##class(%SQL.Statement).%ExecDirect(,sqlStr)
Return rs

```

(3) Write a method to return a result set

- The method of processing and returning results is written in BO

```
ems.cis ... (REST)  receiveMessage.cis  Items.cis ... (message)  Production.cis  SendMessage.cis
nik > src > BKIP > Production > Operation > receiveMessage.cis > BKIP.Production.Operation.receiveMessage > select
Parameter ADAPTER = ENSLIB.SQL.OUTBOUNDADAPTER ;

Property Adapter As EnsLib.SQL.OutboundAdapter;

Parameter INVOCATION = "Queue";

XData MessageMap
{
  <MapItems>
    <MapItem MessageType="BKIP.Production.Message.dto.selectItemList">
      <Method>selectItemList</Method>
    </MapItem>
    <MapItem MessageType="BKIP.Production.Message.dto.selectItemById">
      <Method>selectItemById</Method>
    </MapItem>
    <MapItem MessageType="BKIP.Production.Message.dto.updateItemById">
      <Method>updateItemById</Method>
    </MapItem>
    <MapItem MessageType="BKIP.Production.Message.dto.insertItem">
      <Method>insertItem</Method>
    </MapItem>
    <MapItem MessageType="BKIP.Production.Message.dto.deleteItemById">
      <Method>deleteItemById</Method>
    </MapItem>
    <MapItem MessageType="BKIP.Production.Message.dto.productionList">
      <Method>selectProductionList</Method>
    </MapItem>
  </MapItems>
}
```

查询组件列表

Test the BO after writing it

PRODUCTIONBO

Production BKLINIKPKG.FoundationProduction

请求类型: Bkip.Production.Message.dto.selectItemList

请求详情

ID:

BusinessType:

Production: BKLINIKPKG.FoundationProduction

Name:

Category:

ClassName:

PoolSize:

Enabled: ☒

Foreground: ☐

Comment:

LogTraceEvents: ☒

AlertGroups:

调用测试服务

测试结果

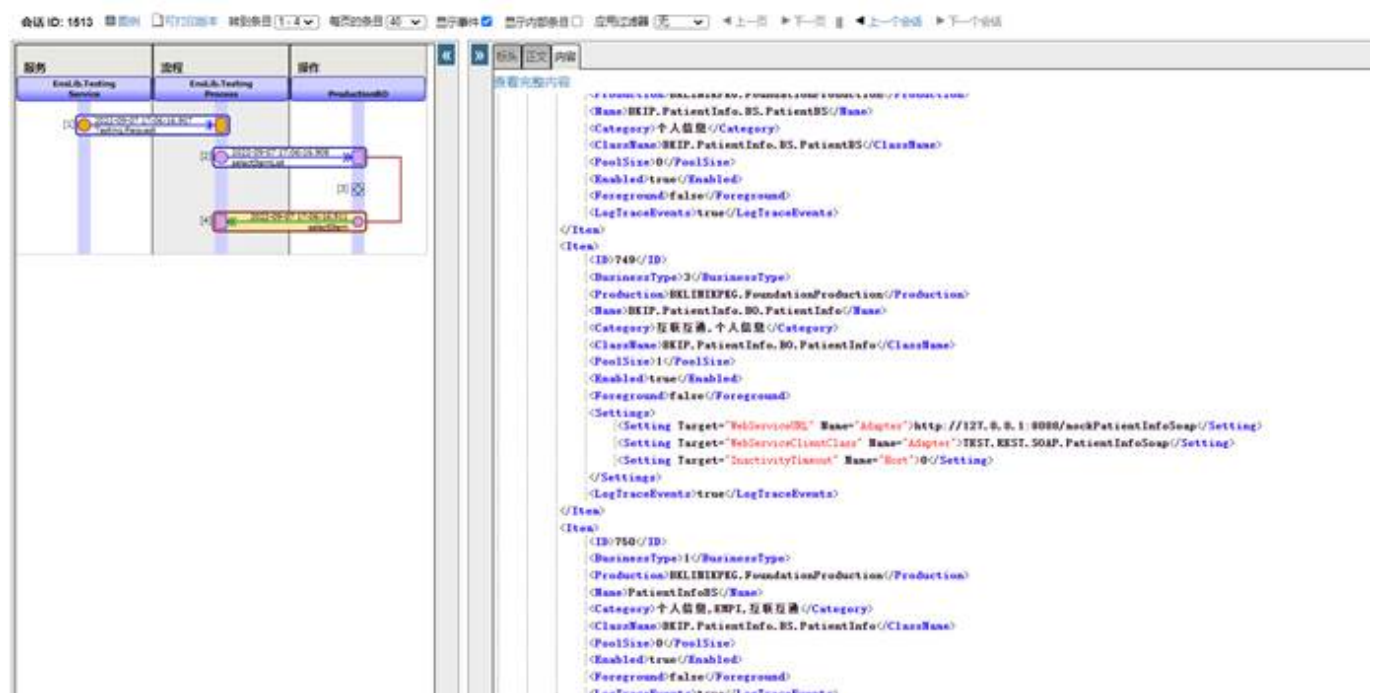
会话 ID: 1513 [可视化追踪](#)

请求已发送: 2022-09-07 17:06:16.906

已收到响应: 2022-09-07 17:06:16.912

Bkip.Production.Message.vo.selectItem

<ObjectId>	1451
code	200
msg	成功
total	0



No problem!!!

(In fact, this step of executing SQL includes getting the result set editing method, which can be directly written to a REST class. Why do you choose this? The [Visual Tracing] function of HealthConnect is very useful. If I forward the interface from BS – BO, is it convenient for debugging? Of course, this is also because IRIS has not yet found a very suitable DeBUG method.)

- Write a forwardedBS :

```

class ForwardedBS extends Ens.BusinessService
{
    Property TargetConfigName As %String(MAXLEN = 2000);

    Parameter SETTINGS = "TargetConfigName:Basic:selector?context={Ens.ContextSearch/ProductionItems?targets=1&productionName=@productionId}";

    Method OnProcessInput(pInput As %RegisteredObject, Output pOutput As %RegisteredObject) As %Status
    {
        Set tsc=..SendRequestSync(..TargetConfigName,pInput,..pOutput)
        Quit tsc
    }
}

```

Just forward the message and do nothing, deploy to Production, as shown in the figure:





#### (4) Organization return, provisionrestful

The method of how to open the url has been provided in the previous article, and the direct screenshot here is omitted :

```

/// 应用程序的默认内容类型。
Class %CSP.REST
{
    Parameter SpecificationClass = "petstore.spec";

    /// 此应用程序的默认内容类型。
    Parameter CONTENTTYPE = "application/json";

    /// 默认情况下，将输入流转换为Unicode
    Parameter CONVERTINPUTSTREAM = 1;

    /// 默认响应字符集是utf-8
    Parameter CHARSET = "utf-8";

    XData UrlMap [ XMLNamespace = "http://www.intersystems.com/urlmap" ]
    {
        <Routes>
            <!-- 组件接口 -->
            <Route Url="/list" Method="POST" Call="selectItemList" Cors="true"/>
            <Route Url="/:id" Method="GET" Call="selectItemById" Cors="true"/>
            <Route Url="/" Method="PUT" Call="updateItemById" Cors="true"/>
            <Route Url="/" Method="POST" Call="insertItem" Cors="true"/>
            <Route Url="/" Method="DELETE" Call="deleteItemById" Cors="true"/>
        </Routes>
    }
}

```

The idea of writing this Rest interface: provide a restful interface to receive JSON, objectify JSON to BS → BO, convert the returned object into JSON and return it to the upstream. The core source code is shown in the figure below

```
/// 查询列表中所有组件
/// ClassMethod selectItemList(production As %String) As %Status
Debug this method
ClassMethod selectItemList() As %Status
{
    Set sc = $$$OK
    s message = ##class(%String).LogicalToDisplay(%request.Content.Read())
    #Dim JsonUtils as BKIP.Rest.Utils.ClassAndJson = ##class(BKIP.Rest.Utils.ClassAndJson).%New()
    Set req = ##class(BKIP.Production.Message.dto.selectItemList).%New()
    Try {
        s tsc = JsonUtils.Json2Object(message,"BKIP.Production.Message.dto.selectItemList",.req) $$$ThrowOnError(tsc)
        s tsc = ..CallInterface("ProductionBS",req,.output)
        s output.total = output.data.Count()
        s tsc = JsonUtils.ObjectToJson(output,.resp)
    }
    Catch ex {
        #; w "错误出现"
        s resp = {}
        Do resp.%Set("code",-1)
        Do resp.%Set("msg",ex.DisplayString())
        s resp = resp.%ToJson()
    }
    Write resp
    Return sc
}
```

Test it:



production接口 / 查询production组件列表

---

POST ▼ http://127.0.0.1:52773/api/production/item/list

---

Params Authorization Headers (8) **Body** ● Pre-request Script Tests Settings


☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** ▼

---

```
1  {
2    "Production": "BKLINIKPKG.FoundationProduction"
3  }
4  }
```

---

Body Cookies Headers (11) Test Results

Pretty Raw Preview Visualize **JSON** ▼ 

```
1  {
2    "code": "200",
3    "msg": "成功",
4    "total": "40",
5    "data": [
6      {
7        "ID": "746",
8        "BusinessType": 3,
9        "Production": "BKLINIKPKG.FoundationProduction",
10       "Name": "Ens.Activity.Operation.Local",
11       "ClassName": "Ens.Activity.Operation.Local",
12       "PoolSize": 1,
13       "Enabled": true,
14       "Foreground": false,
15       "LogTraceEvents": false
16     },
17     {

```

Success!!!

Along these lines, I went on to write other related interfaces:

## JUMP TO

### Introduction

GET 查询production列表

GET 查询production状态

GET 启动production

GET 关闭production

GET 更新production

POST 查询production组件列表

GET 根据Id查production组件

PUT 根据ID修改production组件

POST 新增一个production组件

DEL 根据ID删除一个production组件

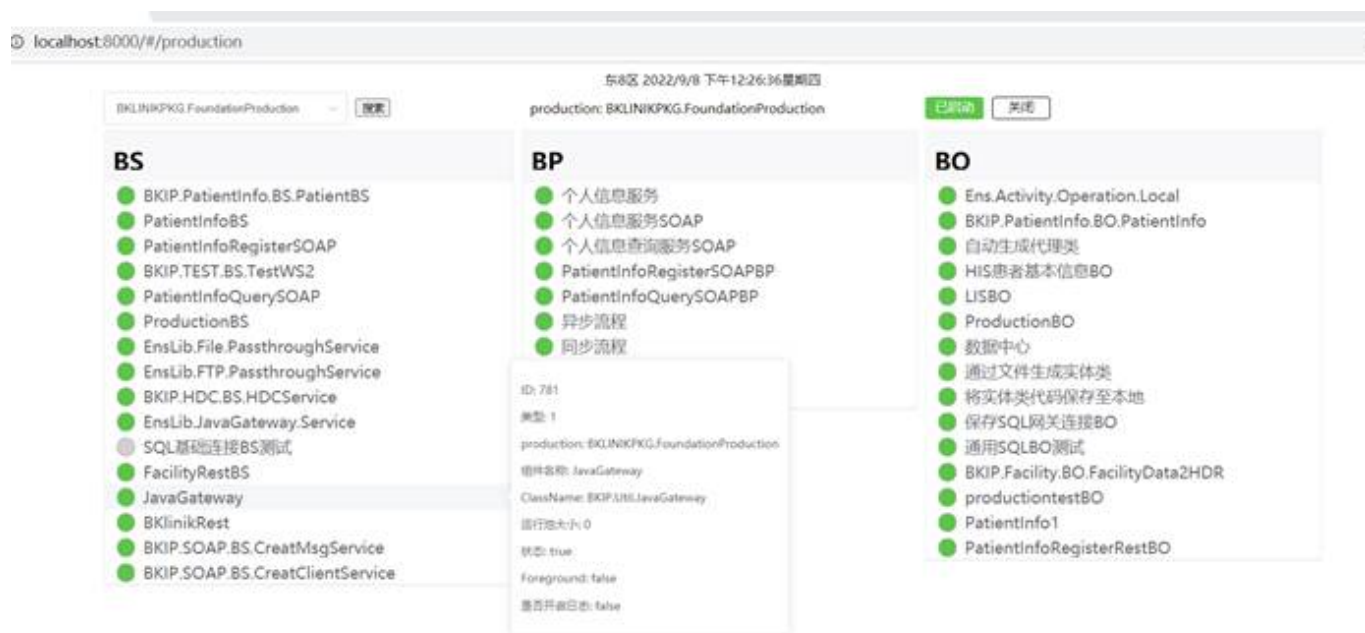
I wrote a simple front end, shown below.

## 4. Presentation of achievements

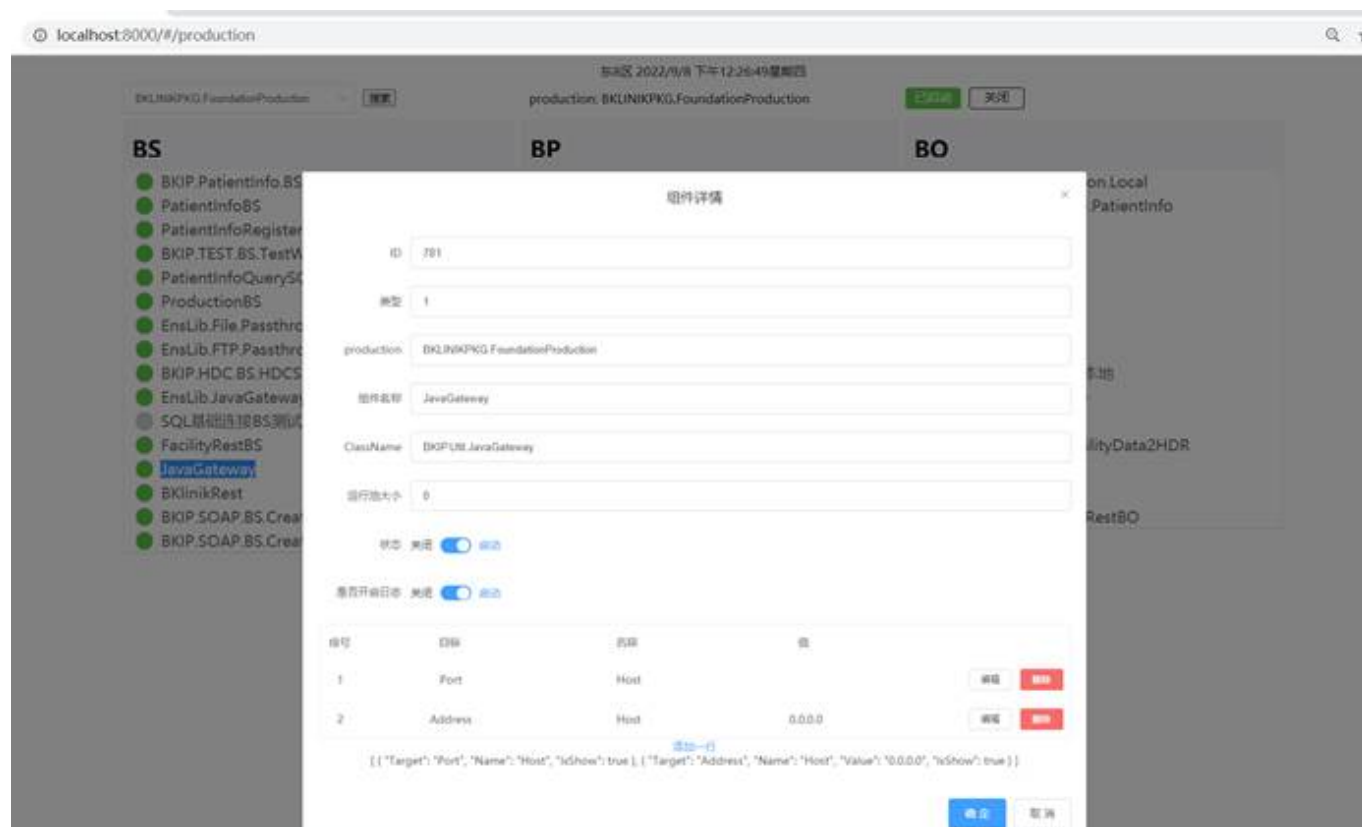
- First, it must be the display of the production interface :



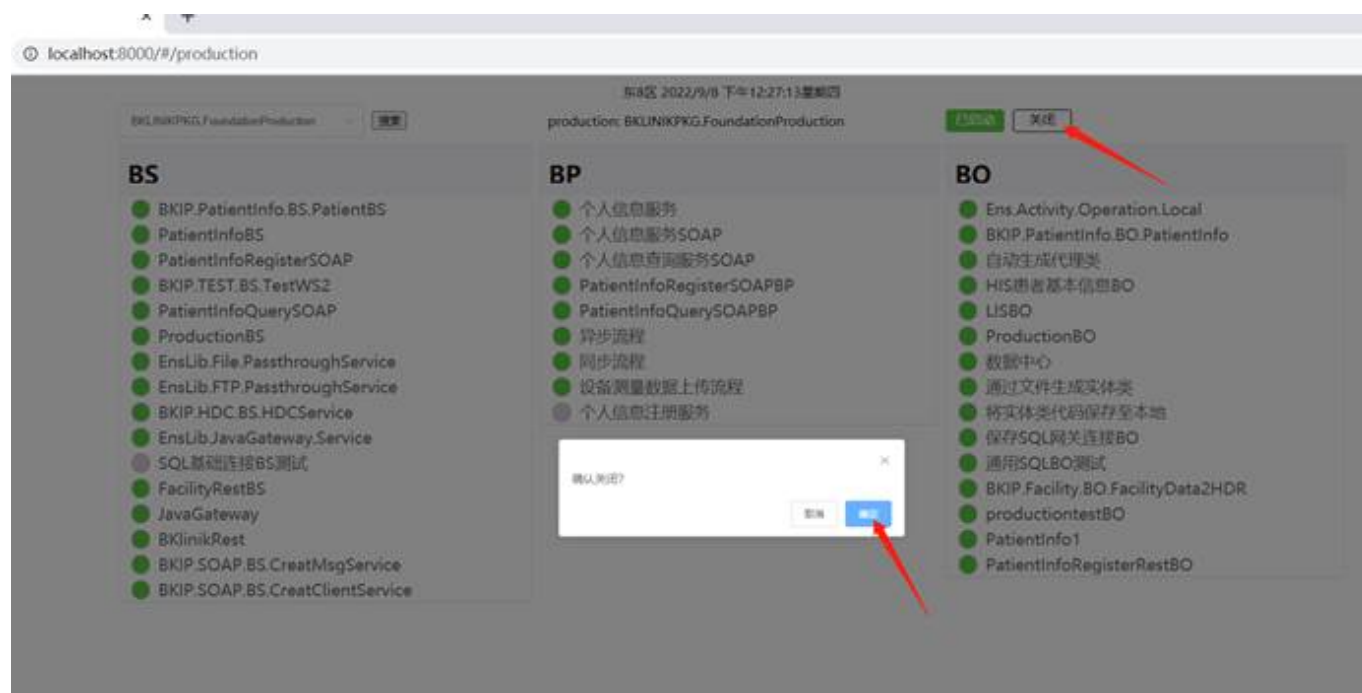
When the mouse is placed on the component, the relevant information will be displayed in a floating frame



Double-click to enter the editing page, which can be saved, and the component switch is also being edited



Click Close to close the home page Production

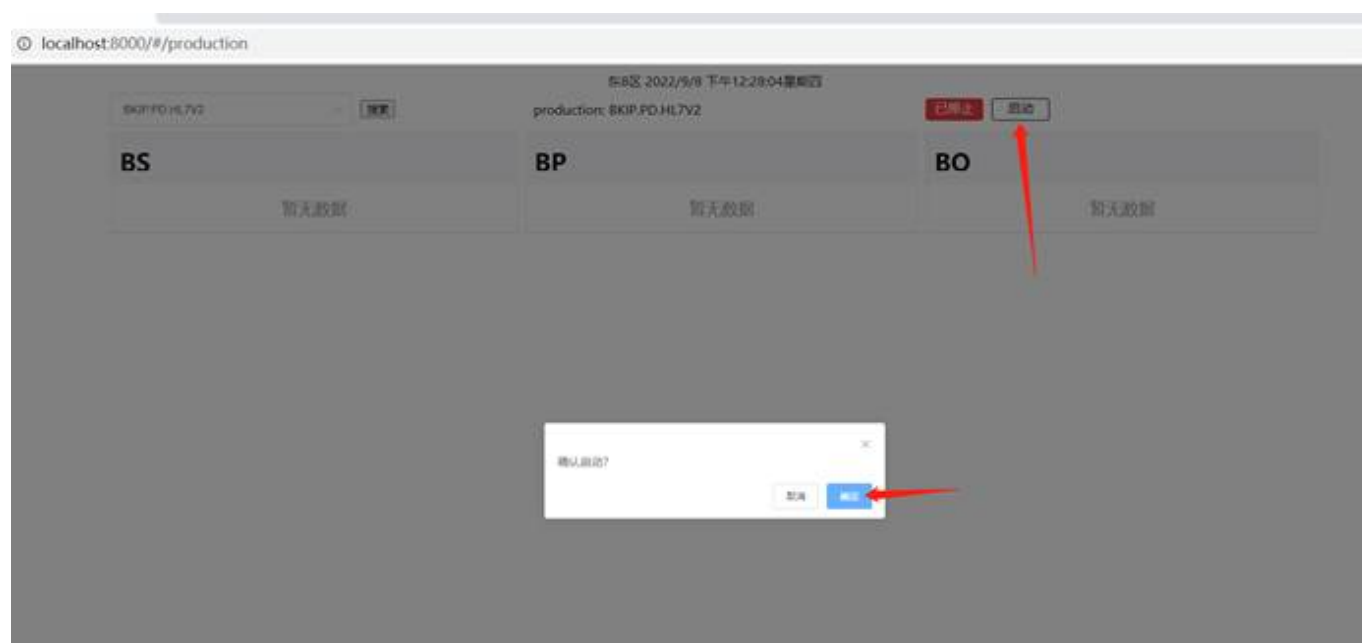




- The drop-down list can display the Production list under this namespace :



- Select the corresponding production name and click Start to start the corresponding Production :





## Summary

The above is the end of the new, modified, deleted and queried functions of IRIS to be demonstrated today. The main demonstration is an idea of background interface development, and of course, there are many other ways and methods, which are only shared here. I've seen it all. If you think it's good, please give me some praise! be deeply grateful

[#API](#) [#JSON](#) [#Management Portal](#) [#ObjectScript](#) [#Ensemble](#) [#HealthShare](#) [#InterSystems IRIS](#) [#VSCode](#)

Source URL: <https://community.intersystems.com/post/how-iris-performs-crud-operations>