# 简介

本SDK提供了访问态神数据库内核的相关接口，仅支持含网络功能的态神产品（含Windows或者嵌入式版本的网络版本）。目前支持Python、NodeJs、COM组件（微软component，可在VB、C#、微软JScript中调用）。

# 安装

## Python

需要Python 3.x以上版本。

1. 将SDK包中的对应平台的TsDBAccess.cp38-win\_amd64.pyd（或者TsDBAccess.cpython-38-x86\_64-linux-gnu.so、TsDBAccess.cpython-38-arm-linux-gnueabihf.so）和TsDBAccess-1.0-py3.8.egg-info复制到python下的site-packages子目录。
2. 如果python不是3.8X版本的话，则需要将上述的文件改名。比如安装的是python3.9.1，则将上述所有含“38”、“3.8”的文件名分别改为“39”、“3.9”。

## NodeJs

将SDK包中的对应平台的TsDBAccess.node复制到相关目录即可。

## COM组件

复制：先把SDK包中的TsDBAccess.dll复制到相关目录。

注册：用管理员打开命令行，cd到TsDBAccess.dll所在目录，然后在命令行中输入Regsvr32 TsDBAccess.dll。

注：如果本机安装了软件，则安装目录已经包含了TsDBAccess.dll。

# 例子

## Python

压缩包中包含Test.py文件，命令行进入Test.py目录后，运行python test.py即可。

## NodeJs

压缩包中包含Test.js文件，命令行进入Test.js目录后，运行node test.js即可。

## COM

Jscript可以参考test.js，除了创建对象不同，其它大致类似。

VB：提供了AccessDB\_VB演示工程。

C#：暂无例子。

# 开发

## 创建DBCore对象

### Python

先引入TsDBAccess扩展模块，然后调用TsDBAccess.DBCore()创建一个DBCore对象。如：

import TsDBAccess

mydb = TsDBAccess.DBCore()

### NodeJs

先引入TsDBAccess扩展模块，然后调用DBCore()创建一个DBCore对象。如：

const addon = require('安装目录/TsDBAccess');

const mydb = new addon.DBCore();

### COM组件

VB、C#中需要先添加引用该组件。Jscript直接创建。

VB：Dim WithEvents mydb As TsDBAccessLib.DBCore

mydb = New TsDBAccessLib.DBCore

C#：TsDBAccessLib.DBCore mydb;

mydb = new TsDBAccessLib.DBCore();

Jscript：mydb = new ActiveXObject(“TsDBAccess.DBCore”);

## 连接数据库（DBCore函数）

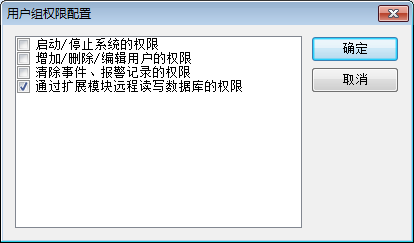
Int ret = Connect(Host, UserName, Password)

Host：服务器/设备主机IP或名称+端口号，如：192.168.1.16:8080。字符串类型。

UserName：用户名，字符串类型。

Password：密码，字符串类型。

返回值：0-成功；-1-用户名/密码错误；-2-网络错误；-3-未知错误。

说明：用户名、密码在工程的用户管理里配置，默认都具备读值权限，如果需要写值权限，则需要在所属的组

## 获取驱动对象（DBCore函数）

drvObj = GetDriver(drvName)

根据驱动名称获取驱动对象（DBDriver对象），如：mydrv = mydb.GetDriver(‘ABC’)

## 设置变量通知事件函数（DBDriver函数）

SetTagNotify(回调通知函数);

设置后，当变量值改变时，会调用回调通知函数。回调通知函数的原型为NotifyFunc(drvName, tagName, value)。

drvName：驱动名称。

tagName：变量名称。

value：值。

### Python

如：

def tagNotify(drv, tag, value) :

print('变量：' + drv + '.' + tag + '：' + str(value))

drv.SetTagNotify(tagNotify)

### NodeJs

如：

mydrv.SetTagNotify(function(drv, tag, value){

console.log(" 变量：" + drv + "." + tag + ":" + value);

}

);

### Com

Jscript，如：

mydrv.SetTagNotify(function(drv, tag, value){

document.write (" 变量：" + drv + "." + tag + ":" + value);

}

);

VB、C#中不能使用此函数，需要响应事件函数OnTagNotify，具体请参考VB、C#相关资料。

## 连接变量（DBDriver函数）

订阅变量，订阅后，当该变量值改变时会调用上述值变通知函数。

ConnectTag(变量集)

变量集：字符串类型，每个变量用逗号隔开。如：

mydrv.ConnectTag('Logo水平位置, Logo垂直位置, Logo旋转角度');

## 取变量值（DBDriver函数）

ret = GetTagValue(变量集)

取指定变量的值。

变量集：字符串类型，每个变量用逗号隔开。

如果只有一个变量，则返回单个值，否则返回数组。返回空表示失败。如：

uu = drv.GetTagValue('Logo水平位置, Logo垂直位置, Logo旋转角度'); //返回uu是数组

uu = drv.GetTagValue('Logo水平位置'); //返回uu是单个值

## 设变量值（DBDriver函数）

ret = SetTagValue(变量集, 值集)

设置指定变量的值。

变量集：字符串类型，每个变量用逗号隔开。

如果只有一个变量，则设置单个值，否则设置数组。如：

uu = new Array(3);

uu[0] = 13.5;

uu[1] = 23.7;

uu[2] = 33.9;

ret = drv.SetTagValue('Logo水平位置, Logo垂直位置, Logo旋转角度', uu); //uu是数组

ret = drv.SetTagValue('Logo水平位置', 36.6); //单个值情况

返回值：0-成功；-1-用户无写值权限；-2-网络错误；-3-参数错误。

## SetRtAlarmNotify（DBCore函数）

SetRtAlarmNotify (回调通知函数, notifyExisting);

设置实时报警通知函数。

notifyExisting：是否通知现有的实时报警

### Python

如：

def rtAlarmNotify(alarmDate, eventDate, tagName, eventType, alarmType, alarmGroup, level, value, limit, status) :

evtType = '报警'

if eventType != 0:

evtType = '确认'

print('报警时间：' + alarmDate.strftime("%Y-%m-%d, %H:%M:%S") + '，事件时间：' + eventDate.strftime("%Y-%m-%d, %H:%M:%S") + '，变量名：' + tagName + '，事件类型：' + str(evtType) + '，报警组：' + alarmGroup + '，级别：' + str(level) + '，报警值：' + str(value) + '，限值：' + str(limit) + '，状态：' + str(status))

mydb.SetRtAlarmNotify(rtAlarmNotify, 1)

### NodeJs

如：

mydb.SetRtAlarmNotify(function(alarmDate, eventDate, tagName, eventType, alarmType, alarmGroup, level, value, limit, status){

const evtType = eventType ? "确认" : "报警";

console.log("报警时间：" + alarmDate + "，事件时间：" + eventDate + "，变量名：" + tagName + "，事件类型：" + evtType +

"，报警组：" + alarmGroup + "，级别：" + level + "，报警值：" + value + "，限值：" + limit + "，状态：" + status);

}, true);

### Com

如：

mydb.SetRtAlarmNotify(function(alarmDate, eventDate, tagName, eventType, alarmType, alarmGroup, level, value, limit, status){

const evtType = eventType ? "确认" : "报警";

document.write ("报警时间：" + alarmDate + "，事件时间：" + eventDate + "，变量名：" + tagName + "，事件类型：" + evtType +

"，报警组：" + alarmGroup + "，级别：" + level + "，报警值：" + value + "，限值：" + limit + "，状态：" + status);

}, true);

VB、C#中不能使用此函数，需要响应事件函数OnSetRtAlarmNotify，具体请参考VB、C#相关资料。

## KeepALive（DBCore函数）

Nodejs专有函数。因为Nodejs垃圾回收会自动清理无用对象，某对象如果长时间没有被调用，则会被清除。DBCore中保持着与数据库端的网络连接，而驱动等读写值函数是基于DBCore网络连接的。

此函数是空函数。