**TwinCAT 3 ADS.NET 使用说明详解**

|  |  |
| --- | --- |
|  | 作者：余洋  职务：技术支持工程师  日期：2023-03-08 |
| **摘 要：**  TwinCAT可以通过ADS和多种高级语言进行通讯。但是例如在高级语言C#于TwinCAT进行通讯之前，需要安装 ['Beckhoff.TwinCAT.Ads Nuget'](https://www.nuget.org/packages/Beckhoff.TwinCAT.Ads/) 、在Visual Studio中选择合适的框架以及其他的相关步骤，本文档介绍了相关概念以及如何自行使用C#与TwinCAT3进行通讯，并给了四个通讯例程进行实例演示。 | |
| **附 件：**   |  |  |  | | --- | --- | --- | | 序 号 | 文件名 | 备注 | | 1 | BeckhoffAdsTester2 | VS代码例程1 | | 2 | BeckhoffAdsTester3 | VS代码例程2 | | 3 | BeckhoffAdsTester4 | VS代码例程3 | | 4 | BeckhoffAdsTester5 | VS代码例程4 | | 5 | TwinCAT Ads Tester1 | PLC例程 | | |
| **历史版本：**   |  |  |  | | --- | --- | --- | | 2020-07-02 | 冯国城 | TwinCAT3与C#语言ADS通讯.docx | |  |  |  | |  |  |  | |  |  |  | |  |  |  | | |
| **免责声明：**  我们已对本文档描述的内容做测试。但是差错在所难免，无法保证绝对正确并完全满足您的使用需求。本文档的内容可能随时更新，如有改动，恕不事先通知，也欢迎您提出改进建议。 | |
| **参考信息：** | |

目 录

[1. 准备工作 4](#_Toc17885)

[1.1. TwinCAT3版本须知 4](#_Toc3162)

[1.2. 开发框架须知 4](#_Toc15525)

[1.3. 库文件（依赖项）须知 4](#_Toc6323)

[2. 库文件的选择以及安装步骤 4](#_Toc5418)

[2.1. ‘Beckhoff.TwinCAT.Ads Nuget' package 4](#_Toc10125)

[2.2. ‘Beckhoff.TwinCAT.Ads.Reactive' package 5](#_Toc27644)

[2.3. 安装步骤 5](#_Toc13079)

[4. 安装添加的NuGet包 5](#_Toc19198)

[3. 例程 6](#_Toc31825)

[3.1. 关闭/启动PLC 6](#_Toc31541)

[3.2. 读写基本的数据 8](#_Toc3109)

[3.3. 读取字符串类型数据 9](#_Toc7663)

[3.4. 读写PLC内部的系统 数据（时间以及日期） 11](#_Toc4723)

[4. 4.X与6.X版本之间的指令区别 12](#_Toc4145)

[5. 结束语 13](#_Toc145)

# 准备工作

## TwinCAT3版本须知

在系统之间使用ADS进行通讯时，需要TwinCAT3担任ADS路由器的角色。因此，TwinCAT3的版本需要和ADS.NET的版本适配。具体的版本对应如下所示：

[Version 4.X](https://infosys.beckhoff.com/content/1033/tc3_ads.net/12209070859.html) ：所有版本皆可适用

[Version 5.X](https://infosys.beckhoff.com/content/1033/tc3_ads.net/12209070859.html) ：TwinCAT >= 3.1.4024.10

[Version 6.X](https://infosys.beckhoff.com/content/1033/tc3_ads.net/12209070859.html) ：TwinCAT >= 3.1.4024.10

## 开发框架须知

在使用C#语言开发上通讯程序之前，需要选择合适的软件开发工具包或者框架。至少需要满足下方列出的SDKs或者Frameworks其中之一：

.NET 5.0 or later

.NET Core 3.1 or later

.NET Framework 4.61 or later

.NET Standard 2.0 compatible SDK or later

注：微软对 .NET5 的支持于 2022 年 5 月 8 日结束。因此，建议将 Beckhoff.TwinCAT 软件包从版本 5 更新到版本 6。（微软.NET技术支持周期查询：[.NET and .NET Core official support policy (microsoft.com)](https://dotnet.microsoft.com/en-us/platform/support/policy/dotnet-core)）

## 库文件（依赖项）须知

在使用C#语言开发上通讯程序的时候，需要选择倍福开发的ADS专用库文件（VS中称之为依赖项），可以选择下方两者中的其中之一：

The[‘Beckhoff.TwinCAT.Ads Nuget'](https://www.nuget.org/packages/Beckhoff.TwinCAT.Ads/) package.（推荐项）

the[‘Beckhoff.TwinCAT.Ads.Reactive'](https://www.nuget.org/packages/Beckhoff.TwinCAT.Ads.Reactive/) package.（另选项）

# 库文件的选择以及安装步骤

现在安装 TwinCAT ads .NET Communication API 的首选方法是使用 NuGet.org 包管理器。

## [‘Beckhoff.TwinCAT.Ads Nuget'](https://www.nuget.org/packages/Beckhoff.TwinCAT.Ads/" \t "https://infosys.beckhoff.com/content/1033/tc3_ads.net/_blank) package

这是实现 ADS 客户端功能的主要库文件。在建立与本地和远程设备的 ADS 连接时是必需的。它包含了.NET Core和.NET Full Framework使用的ADS通信协议的客户端实现。包括开发自己的.NET应用程序（例如HMI，数据记录器）以与TwinCAT设备（例如PLC，NC或IO设备）通信的所有内容。

安装地址：[NuGet Gallery | Beckhoff.TwinCAT.Ads 6.0.216](https://www.nuget.org/packages/Beckhoff.TwinCAT.Ads/)

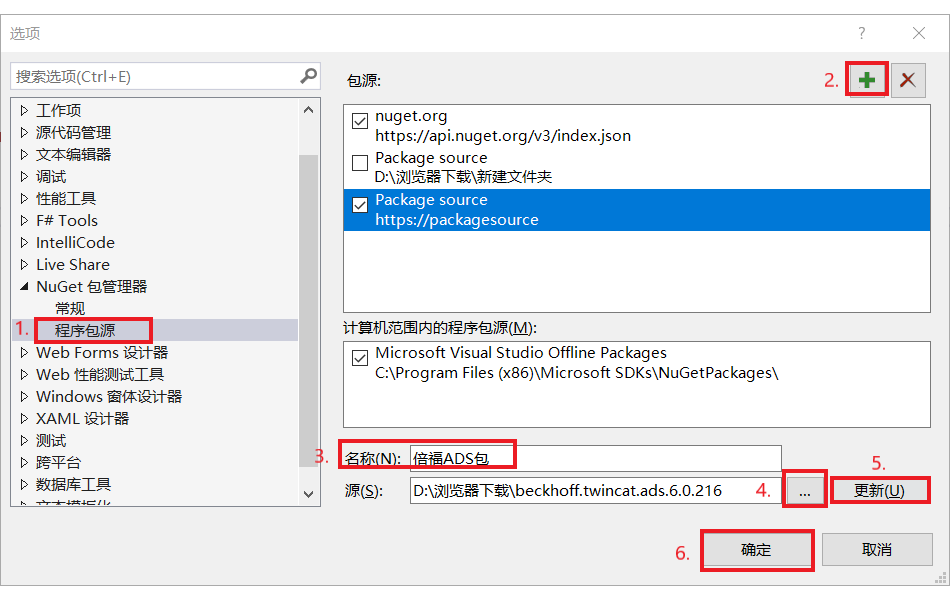
## [‘Beckhoff.TwinCAT.Ads.Reactive'](https://www.nuget.org/packages/Beckhoff.TwinCAT.Ads.Reactive/" \t "https://infosys.beckhoff.com/content/1033/tc3_ads.net/_blank) package

此库文件在[‘Beckhoff.TwinCAT.Ads Nuget'](https://www.nuget.org/packages/Beckhoff.TwinCAT.Ads/" \t "https://infosys.beckhoff.com/content/1033/tc3_ads.net/_blank) package之上安装了额外的扩展包。

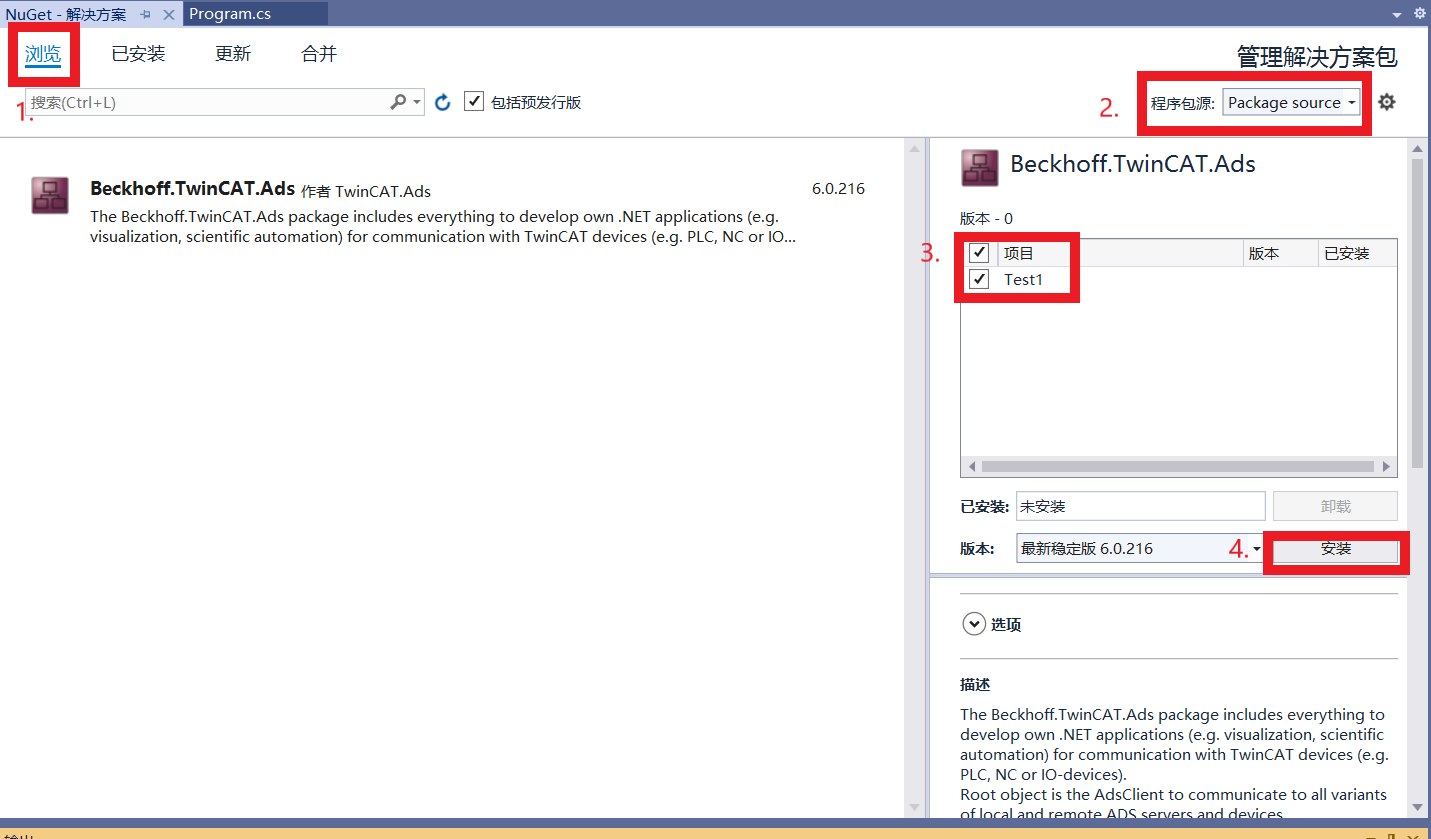
安装地址：[NuGet Gallery | Beckhoff.TwinCAT.Ads.Reactive 6.0.216](https://www.nuget.org/packages/Beckhoff.TwinCAT.Ads.Reactive/)

## 安装步骤

1. 新建一个Visual Studio项目（本文当以Visual Studio 2019为例）
2. 选择工具—Nuget包管理器—程序包管理设置（下载后要将nuget包放进一个文件夹内）



1. 选择工具—Nuget包管理器—管理解决方案的NuGet程序包
2. 安装添加的NuGet包



安装完之后，就可以在Viual Studio中使用C#进行编程了。

# 例程

需要注意的是：TwinCAT与其他高级语言通讯的时候用到的通讯方式为同步和异步，对这个概念有不理解的读者可以去查阅虚拟学院中的Ads课程专区。而对变量数据进行读取的方法也分为句柄（变量名）以及Ads通讯地址读写（首地址Index-Group以及偏移地址Index-Offset）。本文档演示的通讯方法都为异步，而读写方式都为句柄的方式。

（注：以下代码例程仅仅适用于配套的TwinCAT例程，如需和自己的项目进行通讯，需要自行修改代码来实现）

## 关闭/启动PLC

控制PLC运行状态的代码如下：

using System;

//调用倍福Ads库中的内容

using TwinCAT.Ads;

namespace BeckhoffAdsTester2

{

class Program

{

static void Main(string[] args)

{

//创建实例

AdsClient tcClient = new AdsClient();

try

{

//连接851端口，AmsNetId这里如果是本地的话可不写，默认为本地，下方有另一种写法的演示

tcClient.Connect(851);

//人机交互信息

Console.WriteLine(" R为PLC运行 S为PLC停止");

Console.WriteLine("\r\n请选择R或者S来控制PLC的状态");

string sInput = Console.ReadLine().ToLower();

//根据用户的选择来执行PLC状态（如果选择R，PLC切换至运行模式；如果选择S，PLC停止；如果都不是，重复人机交互信息）

do

{

switch (sInput)

{

case "r": tcClient.WriteControl(new StateInfo(AdsState.Run, tcClient.ReadState().DeviceState)); break;

case "s": tcClient.WriteControl(new StateInfo(AdsState.Stop, tcClient.ReadState().DeviceState)); break;

default: Console.WriteLine("请选择R或者S来控制PLC的状态"); sInput = Console.ReadLine().ToLower(); break;

}

} while (sInput != "r" && sInput != "s");

}

//释放缓存

finally

{

tcClient.Dispose();

}

}

}

}

首先，在进行编程之前，需要用到倍福Ads库中的相关函数或者方法，因此在常规的using system下方需要添加using TwinCAT.Ads;。(注：C#是大小写敏感的编程语言)。

接下来，在主程序中将AdsClient类实例化，之后才可以进行调用。

然后通过AmsNetId连接851端口，这里默认就是本地，如果有实际控制器的话需要填写目标控制器的AmsNetId。用到的方法为：AdsClient.Connect（详情见：<https://infosys.beckhoff.com/content/1033/tc3_ads.net/9407735563.html?id=880100549233664563>）。

之后，是常规的人机交互内容，这里就不加赘述了。然后是一个常规的判断执行逻辑：根据用户的选择来执行PLC状态。这里在执行中，用到的是一个函数AdsClient.WriteControl Method (StateInfo)。（详情见：<https://infosys.beckhoff.com/content/1033/tc3_ads.net/9407886091.html?id=6073269950881686912>）。

最后，所有程序执行完成之后，用一个函数AdsClient.dispose()释放缓存作为结束。（详情见：<https://infosys.beckhoff.com/content/1033/tc3_ads.net/9407750923.html?id=2767147550989773101>）。

## 读写基本的数据

读写uint，double，bool类型变量的代码如下：

using System;

using TwinCAT.Ads;

namespace BeckhoffAdsTester3

{

class Program

{

static void Main(string[] args)

{

//创建客户端实例类，通过对应的AmsNetId连接到851端口，新建变量

AdsClient client = new AdsClient();

uint varHandle1 = 0;

uint varHandle2 = 0;

uint varHandle3 = 0;

uint varHandle4 = 0 ;

client.Connect(AmsNetId.Local, 851);

try

{

//给读回写入的变量赋初值

uint valueToRead1 = 0;

double valueToRead2 = 0;

bool valueToRead3 ;

uint valueToWrite = 42;

//将想读的变量给varHandle1

varHandle1 = client.CreateVariableHandle("MAIN.i\_value");

varHandle2 = client.CreateVariableHandle("MAIN.f\_value");

varHandle3 = client.CreateVariableHandle("MAIN.b\_value");

varHandle4 = client.CreateVariableHandle("MAIN.i\_valuewritten");

//写入值

//client.WriteAny(varHandle4, valueToWrite);

//读取值

valueToRead1 = (uint)client.ReadAny(varHandle1, typeof(uint));

valueToRead2 = (double)client.ReadAny(varHandle2, typeof(double));

valueToRead3 = (bool)client.ReadAny(varHandle3, typeof(bool));

//输出结果

Console.WriteLine("读取到的变量值i\_value为：{0}", valueToRead1);

Console.WriteLine("读取到的变量值f\_value为：{0}", valueToRead2);

Console.WriteLine("读取到的变量值b\_value为：{0}", valueToRead3);

}

finally

{

//释放缓存

client.DeleteVariableHandle(varHandle1);

client.DeleteVariableHandle(varHandle2);

client.DeleteVariableHandle(varHandle3);

}

}

}

}

本例程的作用是读写unit类型变量，库文件依然用的是system和TiwnCAT.Ads。逻辑和上一例程相同，这里就不多加赘述了，如又不懂之处可以查看注释。这里用到的方法为AdsClient.ReadAny，可以实现多种变量的读取（详情请见：

<https://infosys.beckhoff.com/content/1033/tc3_ads.net/9407764235.html?id=6680771466346705379)>以及 AdsClient.DeleteVariableHandle（详情见：<https://infosys.beckhoff.com/content/1033/tc3_ads.net/9407747851.html?id=6310906797627741841>）。

## 读取字符串类型数据

读写字符串的代码如下：

using System;

using TwinCAT.Ads;

using TwinCAT.TypeSystem;

namespace BeckhoffAdsTester4

{

class Program

{

static void Main(string[] args)

{

//创建客户端实例类，连接851端口和AmsNetId，再声明需要读的变量

AdsClient client = new AdsClient();

client.Connect(AmsNetId.Local, 851); // 通过对应的AmsNetId连接到851端口

// 将变量handle指向想要读取的变量

uint handle1 = client.CreateVariableHandle("MAIN.s\_value");

uint handle2 = client.CreateVariableHandle("MAIN.s\_valuewritten");

try

{

// 大小设置为80字节(可大不可小)，实例化字节大小类，实例化强制转换类并读取数据

int byteSize = 80;

byte[] buffer = new byte[byteSize];

int readBytes1 = client.Read(handle1, buffer.AsMemory());

PrimitiveTypeMarshaler converter = new PrimitiveTypeMarshaler(StringMarshaler.DefaultEncoding);

//声明一个新的变量value，强制转换数据，并输出结果

string value = null;

converter.Unmarshal<string>(buffer.AsSpan(), out value);

Console.WriteLine("读取到的字符串为：{0}", value);

//实例化写入字节大小类，给予变量新的值，转换并写入

byte[] writeBuffer = new byte[byteSize];

value = "Hello TwinCAT";

converter.Marshal(value, writeBuffer);

client.Write(handle2, writeBuffer);

}

//捕捉错误代码

catch(Exception err)

{

Console.WriteLine("{0}", err.Message);

}

//删除缓存

finally

{

client.DeleteVariableHandle(handle1);

client.DeleteVariableHandle(handle2);

}

}

}

}

本例程除了需要用到TwinCAT.Ads之外，还需要使用库文件TwinCAT.TypeSystem。逻辑与之前的例程有些许的不同之处，这是因为String类型的数据比较特殊，需要进行转换的操作，因此需要在中间实例化一个转换类。在读取时使用的方法为AdsClinet.Read（<https://infosys.beckhoff.com/content/1033/tc3_ads.net/9407761163.html?id=5916645842109940604>），根据引脚的格式因此在读取之前实例化字节大小类并给予一个数据大小的值。最后，在读取之后还需转换。写入String类型数据的方式同理，用到的方法是AdsClient.Write（详情见：<https://infosys.beckhoff.com/content/1033/tc3_ads.net/9407863563.html?id=3164779019983369961）>。最后释放缓存使用的方法和上方的例程一致，为AdsClient.DeleteVariableHandle。

## 读写PLC内部的系统 数据（时间以及日期）

读写PLC内部系统数据的代码如下：

using System;

using TwinCAT.Ads;

using TwinCAT.PlcOpen;

using TwinCAT.TypeSystem;

namespace BeckhoffAdsTester5

{

class Program

{

static void Main(string[] args)

{

//实例化客户端类，连接AmsNetId以及851端口，并创建变量

AdsClient client = new AdsClient();

client.Connect(AmsNetId.Local, 851);

uint handleTime = 0;

uint handleDate = 0;

try

{

//将创建的变量指向PLC变量

handleTime = client.CreateVariableHandle("MAIN.t\_value"); // TIME

handleDate = client.CreateVariableHandle("MAIN.d\_value"); // DATE

//给予读取的数据大小，最大的PLC Open类型的数据为8个字节大小

byte[] readBuffer = new byte[8];

byte[] writeBuffer = new byte[8];

// 读取时间

client.Read(handleTime, readBuffer.AsMemory(0, TIME.MarshalSize));

// 新建一个变量plcTime，处理读取的数据并赋值给plcTime

TIME plcTime = null;

PrimitiveTypeMarshaler.Default.Unmarshal(readBuffer.AsSpan(0, TIME.MarshalSize), out plcTime);

TimeSpan time = plcTime.Time;

Console.WriteLine("读取到的时间为：{0}", plcTime.Time);

// 写入时间

PrimitiveTypeMarshaler.Default.Marshal(time, writeBuffer.AsSpan());

client.Write(handleTime, writeBuffer.AsMemory(0, TIME.MarshalSize));

// 读取日期

DATE plcDate = null;

client.Read(handleDate, readBuffer.AsMemory(0, DATE.MarshalSize));

// 新建一个变量dateTime，处理读取的数据并赋值给dateTime

PrimitiveTypeMarshaler.Default.Unmarshal(readBuffer.AsSpan(0, DATE.MarshalSize), out plcDate);

DateTimeOffset dateTime = plcDate.Date;

Console.WriteLine("日期为：{0}", plcDate);

// 写入日期

PrimitiveTypeMarshaler.Default.Marshal(plcDate, writeBuffer.AsSpan());

client.Write(handleDate, writeBuffer.AsMemory(0, DATE.MarshalSize));

}

finally

{

//释放缓存

client.DeleteVariableHandle(handleTime);

client.DeleteVariableHandle(handleDate);

}

}

}

}

所有例程的程序逻辑都是类似的，都是实例化类，新建变量，读取数据，处理以及转换数据，最后输出数据。本例程将时间和日期类型的数据放在一起演示了，因此看上去会有些许复杂，但实际上都是类似的思想。值得一提的是，这里在库文件中，由于数据类型的特殊性，需要再添加一个库文件TwinCAT.PlcOpen。

程序中使用的读写方法为都与之前的例程一致，但需要在使用方法上做一些调整，具体可以通过查询官网学习，这里就不加赘述了。

# 4.X与6.X版本之间的指令区别

新的架构是基于.NET CORE之上的（如.NET6.0）。为了支持这些较新的版本的及其新功能，倍福提供了Beckhoff.TwinCAT.Ads 的新版本6.X。由于新的平台功能需要在 API 中进行一些重大更改才能获得支持，6.X删除或更改了一些功能。不过，大多数接口保持稳定，只有少数方法发生了变化。

4.X与6.X的方法指令区别如下：

**（1）实例化ADS客户端指令**

## Version 4.X：TcAdsClient client = new TcAdsClient()

## Version 6.X：AdsClient client = new AdsClient()

**（2）声明读写方法**

## Version 4.X：public int TcAdsClient.Read(uint indexGruop,uint indexOffset,AdsStream dataStream)

## Version 4.X：public int TcAdsClient.Write(uint indexGroup,uint indexOffset,AdsStream dataStream)

Version 4.X：public int TcAdsClient.Read(uint indexGroup,uint indexOffset,byte[] bytes, int offset, length)

Version 4.X：public int TcAdsClient.Write(uint indexGroup,uint indexOffset,byte[] bytes, int offset, length)

Version 6.X：public int AdsClient.Read(uint indexGroup,uint indexOffset,Memory<byte> buffer)

Version 6.X：public int AdsClient.Write(uint indexGroup,uint indexOffset,ReadOnlyMemory<byte> buffer)

**（3）声明句柄类型**

Version 4.X：public int TcAdsClient.CreateVariableHandle(string symbolPath)

Version 6.X：public uint AdsClient.CreateVariableHandle(string symbolPath)

**（4）声明通讯参数类**

Version 4.X：public int AddDeviceNotification(string symbolPath, AdsStream dataStream, int offset, int length, AdsTransMode transMode, int cycleTime, int maxDelay,Object userData)

Version 6.X：public uint AdsClient.AddDeviceNotification(string symbolPath,int dataSize, NotificationSettings settings, object userData)

# 结束语

此文档牵涉到许多知识点，如Ads通讯，Vsiual Studio中Nuget包的使用方法以及C#编程，适合有使用C#开发上位机经验和基础的工程师阅读，并且建议读者先去了解倍福虚拟学院上Ads专栏课程中的内容。为了让大多数读者能看懂，在文中已经对一些可能对读者造成困惑的概念进行了讲解，使用的例程代码也是尽可能简单，并没有使用OOP思想以及人机交互界面。总而言之，此文档仅仅是教会读者如何去使用Visual Stuido用C#进行编写，例程也仅是实现了最基本的功能。如果有个性化的上位机功能需求，可以查阅Beckhoff Information System对Twincat Ads库中的namespace进行详细的学习。

**上海（ 中国区总部）**

中国上海市静安区汶水路 299 弄 9号（市北智汇园）

电话: 021-66312666

**北京分公司**

北京市西城区新街口北大街 3 号新街高和大厦 407 室

电话: 010-82200036 邮箱: beijing@beckhoff.com.cn

**广州分公司**

广州市天河区珠江新城珠江东路32号利通广场1303室

电话: 020-38010300/1/2 邮箱: guangzhou@beckhoff.com.cn

**成都分公司**

成都市锦江区东御街18号 百扬大厦2305 室

电话: 028-86202581 邮箱: chengdu@beckhoff.com.cn

|  |  |
| --- | --- |
| 请用微信扫描二维码  通过公众号与技术支持交流 | 倍福官方网站：  https://www.beckhoff.com.cn  在线帮助系统：  https://infosys.beckhoff.com/index\_en.htm |
| 倍福虚拟学院：  https://tr.beckhoff.com.cn/ |
| 招贤纳士：job@beckhoff.com.cn  技术支持：support@beckhoff.com.cn  产品维修：service@beckhoff.com.cn  方案咨询：sales@beckhoff.com.cn |
|  |