Getting Started

This topic helps you get started using the Automotive Ethernet Socket API for C. It includes information about using NI-XNET within Microsoft Visual C++.

Visual C++

Refer to the *Microsoft Visual Studio Support* section in the NI-XNET readme file for the versions of Microsoft Visual C/C++ that your NI-XNET software supports.

The `NIEXTCCOMPILERSUPP` environment variable is provided as an alias to the C language header file and library location. You can use this variable when compiling and linking an application.

For compiling applications that use the Automotive Ethernet Socket API for C, you must include the `nxsocket.h` header file in the code. To include the header file, add `#include` to the beginning of the code. For example:

```
#include "nxsocket.h"
```

In your project options for compiling, you must include the following statement to add a search directory to find the header file:

```
/I "$(NIEXTCCOMPILERSUPP)include"
```

For linking applications, you must add the nixntipstack.lib file and the following statement to your linker project options to search for the library (for a 32-bit OS, replace *lib64* with *lib32*):

```
/libpath:"$(NIEXTCCOMPILERSUPP)\lib64\msvc"
```

The reference for each NI-XNET IP Stack API function is in IP Stack Functions. The reference for each NI-XNET Socket API function is in Socket Functions, and each socket option is referenced in Socket Options.

nxIpStackCreate

Purpose

Creates an IP stack to use for TCP and/or UDP communication.

Format

```
nxStatus_t nxIpStackCreate (
   const char * stackName,
   const char * config,
   nxIpStackRef_t * stackRef);
```

Inputs

`const char * stackName`

The name that uniquely identifies the stack. The syntax for this name allows some special characters, such as space ( ). Invalid characters include forward slash (/), backslash (\), period (.), and tab (\t). The name is not case sensitive. If you do not enter a value for `StackName`, NI-XNET generates a name to ensure that each stack is unique.

`const char * config`

The configuration of the IP Stack as a JSON string. For a list of features supported in the configuration, refer to Supported Features.

Outputs

`nxIpStackRef_t stackRef`

The reference to the created IP Stack. This session reference is used with subsequent functions for the IP Stack (for example, to obtain runtime info), TCP Socket, and UDP Socket.

Return Value

`nxStatus_t`

The error code the function returns in the event of an error or warning. A value of 0 indicates success. A positive value indicates a warning. A negative value indicates an error. If an error is returned, nxgetlasterrorstr can be used to obtain detailed information about potential problems with the IP Stack configuration.

Description

The IP Stack enables you to create an implementation of the TCP/IP protocol suite for TCP and UDP communication, independent from the limitations of the IP stack native to the operating system.

National Instruments installs documentation for the XNET IP Stack configuration string. The configuration string uses JSON format, and the formal documentation is provided as a JSON schema file. The JSON schema file is supported by a variety of online tools, and in addition to formally describing each field, it can be used to validate your customized JSON configuration string for correctness. To find the XNET IP Stack JSON schema, select **Start»National Instruments»NI-XNET Documentation**.