

IPLink

Overview

IPLink is a lightweight point-to-point protocol built on top of TCP/IP sockets that allows a set of sequenced messages to flow between a server and a client. IPLink does not demand a single type of carrier (e.g., it will work with leased lines, frame relay, Internet, etc.), nor a single security protocol. It leaves many of these decisions to the individual firms that are using it.

IP Link Demo

Download the IPLink Demo 180 KB - (.zip)

What it is

The IpLinkDemo application was developed as a sample of the features of IPLink. As such, it is not a complete trading application, and exists solely to demonstrate one way of implementing the IPLink specification.

How it was written

The application was written in C++ using the MFC framework. Programmers who are not familiar with the MFC application framework may be confused by all the code in the application, but most of that code is generated by the MFC Application Wizard, and does not have much significance in the actual application's data flow.

Getting Started

The first step is to unzip the sample using WinZip or a compatible product. Take note that the Use Folder Names option must be checked so that the RES directory is created correctly. Once that is done, open the file IpLinkDemo.dsp in Microsoft Visual Studio, using the Open Project option in the File Menu. Once that is done, you build the project as per normal. When you run the application, a login screen is displayed, followed by a Trade dialog that allows you to place trades to the API server. All output is then displayed in the document window.

All the different windows in the sample application interact with the IPLink module, which contains all the code to create IPLink messages, as well as the code to convert the incoming messages into a manageable form.

Incoming messages consist of a string of tags. The IPLink module contains code, which extracts all the tags and builds a CString Array which has the values at the appropriate positions in the array. So, when a message is received, it is processed and tag 1's text will be in the String Array's index 1, Tag 2 is at CString[2], etc