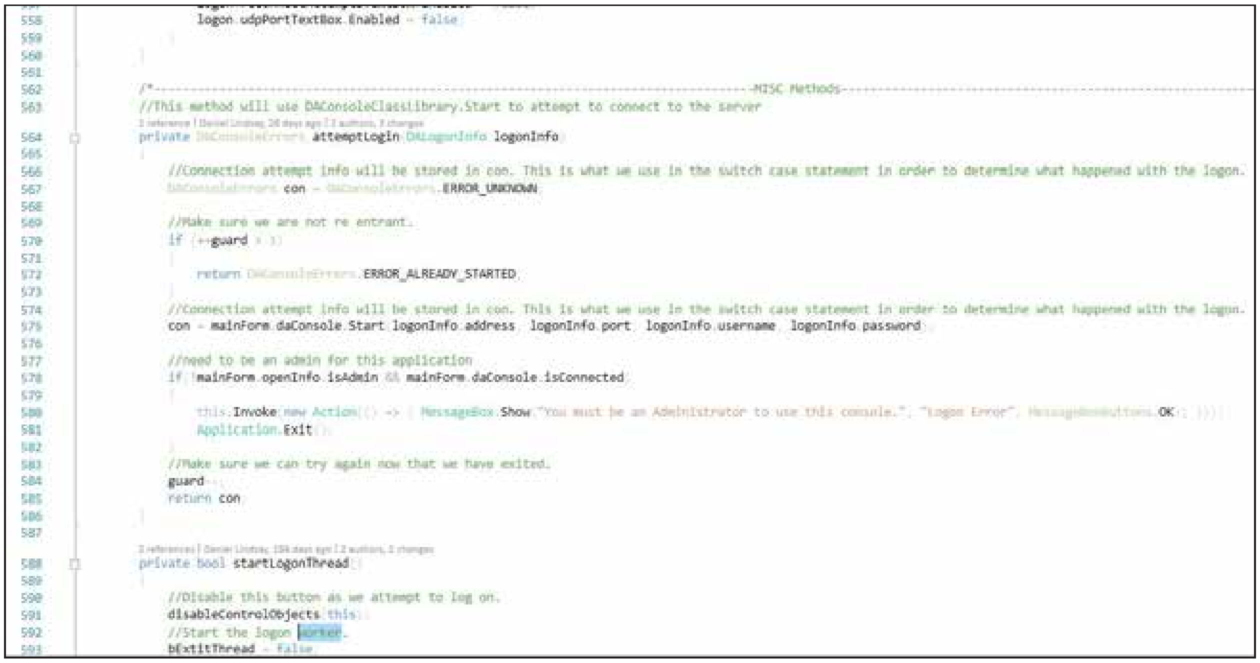
Digital Acoustics

TalkMaster FOCUS
.NET Console SDK
Overview



```
                logon.udpPortTextBox.Enabled = false;
            }
        }

        /*----------------------------------------------------------------------------------MISC Methods----------------------------
        //This method will use DAConsoleClassLibrary.Start to attempt to connect to the server
        1 reference | Daniel Lindsay, 26 days ago | 2 authors, 2 changes
        private DAConsoleErrors attemptLogin(DALogonInfo logonInfo)
        {
            //Connection attempt info will be stored in con. This is what we use in the switch case statement in order to determine what happened with the logon.
            DAConsoleErrors con = DAConsoleErrors.ERROR_UNKNOWN;

            //Make sure we are not re entrant.
            if (++guard > 1)
            {
                return DAConsoleErrors.ERROR_ALREADY_STARTED;
            }
            //Connection attempt info will be stored in con. This is what we use in the switch case statement in order to determine what happened with the logon.
            con = mainForm.daConsole.Start(logonInfo.address, logonInfo.port, logonInfo.username, logonInfo.password);

            //need to be an admin for this application
            if (!mainForm.openInfo.isAdmin && mainForm.daConsole.isConnected)
            {
                this.Invoke(new Action(() => { MessageBox.Show("You must be an Administrator to use this console.", "Logon Error", MessageBoxButtons.OK); }));
                Application.Exit();
            }
            //Make sure we can try again now that we have exited.
            guard--;
            return con;
        }

        2 references | Daniel Lindsay, 156 days ago | 2 authors, 2 changes
        private bool startLogonThread()
        {
            //Disable this button as we attempt to log on.
            disableControlObjects(this);
            //Start the logon worker.
            bExitThread = false;
```

Release Notes:

January 2018

- Added the ability to send server-based Messages to individual IP Endpoints. If using TalkMaster V5.2, the messages will always be sent via TCP. If using TalkMaster V6 or greater, the messages will be sent via RTP if the IP Endpoint has the **Use UDP/RTP** option set

May 2017 - Initial Release

- In order to use the TalkMaster FOCUS .NET SDK, it must connect to TalkMaster FOCUS Server V5.2.4 or greater. TalkMaster FOCUS Server will report that it is using an API Version of 1.8.1 or greater

Glossary

Call Queue - A Call Queue entry is generated by the TalkMaster FOCUS Server when it detects an incoming call from an IP Endpoint (i.e. the IP Endpoint's TALK button has been pressed). The Call Queue Entry can include audio and is sent to all Console Applications involved in the associated queue of the device to be answered. Recording audio in the call queue is a configuration option made in the TalkMaster FOCUS Administrator console.

Console Application - Application developed using the Digital Acoustics SDK that logs into the TalkMaster FOCUS Server in order to communicate with one or more IP Endpoints.

Console Operator - A user, configured through the TalkMaster FOCUS Administrator console, that can login to the TalkMaster FOCUS Server through a Console Application or Console Service.

ii3 - A single zone IP based Intercom, Speaker or Amplifier. ii3 IP Endpoints have not been manufactured since 2008, but are still supported by TalkMaster FOCUS Server as a TalkMaster Endpoint.

IP Endpoint - A generic term for an IP7 or ii3 IP Audio device.

IP7 - A single zone IP based Intercom, Speaker or Amplifier. IP7s can be configured as a TalkMaster Endpoint, a SIP Endpoint or as hybrid TalkMaster/SIP Endpoint.

IP7-MZC - An IP7 endpoint that can have up to forty-eight programmatically controlled Microphone and Speaker zones. It can also be configured with an added telephone as a handset to carry on a private conversation through the IP connection.

Message - A pre-recorded audio file assigned to a Group. A group can have multiple Messages. Messages can be .WAV or .MP3 file formats, and will be converted into an 8K sample rate format before being sent to the end points(s).

Queues - Server based collection of (typically) IP Intercoms. Queues can be used to determine which IP Endpoints should be managed and/or displayed by a Console Operator User. An IP Endpoint can only exist in one Queue.

Paging Group - Server based Groups (aka Paging Zones) are a collection of IP Endpoints used to simultaneously send live or pre-recorded audio pages. An IP Endpoint can exist in multiple Groups. Server stored Messages, or pre-recorded audio files, are associated with a Group to help Console Operators easily select messages to send to these devices.

System Variables - The TalkMaster Server can store Variables and their associated values. These can be created, read, updated and deleted by an application.

TalkMaster FOCUS Server - A Windows® service that handles the connections from Console Applications and IP Audio devices. The Server passes commands and audio between the Console Applications and IP Audio devices.

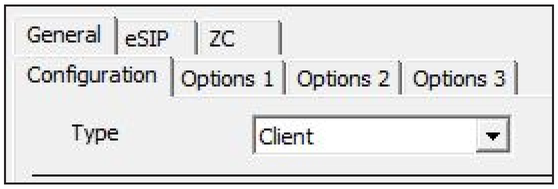
Z4XL - A four zone expander board for the IP7-MZC. Up to twelve ZX4L boards can be daisy-chained to an IP7-MZC providing up to 48 different microphone and speaker zones..

SDK Introduction

This document describes the Digital Acoustics TalkMaster FOCUS .NET SDK (DAConsoleClassLibrary.dll, a .NET Assembly) that interacts with the Digital Acoustics TalkMaster FOCUS Server and IP Audio Endpoints. The SDK will enable a developer to write a custom application using C# or a managed memory application to communicate and control the Digital Acoustics' IP7 Audio Endpoints connected to TalkMaster FOCUS Server.

Each instance of the custom application will connect to the TalkMaster™ FOCUS Server and will consume a Console license. TalkMaster™ FOCUS Server keeps track of Console License usage and availability through its licensing scheme. If the TalkMaster FOCUS Server runs out of console licenses, the connection will fail with a local error of DAConsoleClassLibrary.DAConsoleErrors.ERROR_OUT_OF_LICENSES. -32.

IP Endpoints must be configured with a **TYPE** of ***Client*** in the TalkMaster FOCUS Administrator Console in order to connect to the TalkMaster FOCUS Server and be accessible by the .NET SDK. The IP Endpoints can send/receive audio via TalkMaster FOCUS and/or via the SIP Protocol.



Type: Client

Here are some examples of functionality provided by the Assembly:

- Create a console connection to TalkMaster FOCUS Server
- Send audio from the local computer's microphone and play it on the IP7's speaker
- Receive audio from an IP7's microphone and play it on the local computer's speaker
- Send audio from the local computer's microphone to a group of IP7's and play it on their speakers
- Send a local or server based audio file to a single device or group of IP7's and play it on their speakers
- Control a single or group of IP7's dry contact relay. Turn on/off, and see relay status
- Send Zone changes to an IP7-MZC to switch audio zones before or during a call.
- Receive and send a raw audio stream from/to an IP7
- Send SIP commands to an IP7 to start or end a SIP call.

Difficult aspects of VoIP programming, such as access to the local machine's audio speaker and microphone, and properly receiving and delivering that audio so the playback is clear, is performed by this object. The sample application included with the SDK demonstrates many of the capabilities.

Architecture

The architecture of the .NET SDK conforms to standard OOD methods. DAConsoleClassLibrary.ClassAccess object is the main object of the .NET Assembly. All objects in the assembly fall under this object. This object is used to configure and use the rest of the objects in the namespace.

Event Handling

When the Start method is successfully issued, the TalkMaster FOCUS Server will send the following Events to the application:

- **AccessEventHandler:EVENT_CONSOLE_CONNECT** – All Console Operators and their connection status are reported to the Console Application
- **GroupEventHandler:EVENT_CONSOLE_GROUP_REPORT** – All Groups are reported to every Console. If Paging Groups are being used, it is the Console Application's responsibility to use only use the Groups that are assigned to the current OperatorID (specified in the DAStart API)
- **GroupEventHandler:EVENT_CONSOLE_GROUP_MESSAGE** – Information about server defined pre-recorded messages
- **GroupEventHandler:EVENT_CONSOLE_GROUP_PLAYING** - Information about a group that is currently playing a message
- **DeviceEventHandler:EVENT_NEW_CONNECTION** - Information about each IP Endpoint that is "Authorized" to the TalkMaster FOCUS Server. This includes devices that are in the operator's specified "Queue" and any others in the system.
- **CallQueueEventHandler:EVENT_CALL_QUEUE_MEMBERSHIP** - Information about the membership of this user and a given Call Queue object.
- **AccessEventHandler:EVENT_CONSOLE_CONSOLE_READY** - This event is sent by the TalkMaster FOCUS Server after all of the other events have been sent. The Console Application should wait for the EVENT_Console_Ready before issuing any other APIs.

These events will also be sent during the execution of the program.

IP Endpoints

ii3 and IP7 Differences:
The .NET SDK supports both the ii3 and the IP7 series IP Endpoints and Paging Endpoints. The ii3 series was replaced by the IP7 series in 2008.

The “series” of an IP Endpoint is determined from the Version property of the Class Access object:

- For an ii3 series device, *major* will be set to 4 or less
- For an IP7, *major* it will be set to 5 or greater

(See DADeviceInterface.DeviceVersion / DeviceVersionString for additional information about versions)

Here is a summary of the basic differences between the ii3 series and the IP7 series:

Capability	**Ii3 series**	**IP7 series**
Play 8-bit PCM Audio	Yes	Yes
Send 8-bit PCM Audio	Yes	Yes
Play g.711 uLaw Audio*	Yes	Yes
Send g.711 uLaw Audio*	No	Yes
Receive Multicast	No	Yes
Receive Unicast	Yes	Yes
Receive Broadcast	Yes	Yes
Supports RTP Protocol	No	Yes
Supports SIP Protocol	No	Yes
Supports Full Duplex	No	Yes

*Recommended format for best audio quality

IP7 MZC Overview:
The IP7-MZC - Multi Zone Controller is an IP Endpoint designed for the central station market with support for up to forty-eight zones of full or half duplex audio through a single IP Endpoint.

The IP7-MZC includes the following capabilities:

- Zones can be added in groups of four by adding a ZX4L Expansion board. At least one ZX4L is required.

- Each ZX4L includes individual zone connections for a 12VDC 2-wire or 3-wire microphone, a microphone sensitivity control, DVR Output pins for 24x7 microphone recording, speaker outputs to drive a 1-watt speaker, a powered speaker or analog amplifier, and a speaker volume control
- Zones can be programmatically turned on/off via software using the Digital Acoustics SDK or via SIP DTMF tones
- DVR screw terminals on the main MZC board are provided to record the “active” IP conversation.
- This product can be configured as a TalkMaster FOCUS client, a SIP 2.0 Phone endpoint, or a hybrid TalkMaster FOCUS client/SIP Endpoint. It must be configured as a TalkMaster FOCUS client or a hybrid TalkMaster FOCUS client/SIP Endpoint to be used by the SDK
- The product can also be configured with up to eight groups of zones (groups 1 -8). There are two reserved groups that are always available, group 0 turns off all zones and group 9 turns on all zones
- The speaker and/or microphones for each zone are controlled by using either the DADeviceInterface.MzcSetZoneFrame or the DADeviceInterface.MzcSetZoneGroup APIs
- Zones can be set or changed prior to or while sending audio to an endpoint from a console or during a SIP call
- An analog phone interface is provided to carry on a private conversation over the IP connection using a standard telephone connected to the main MZC board. The application should turn off all zones by selecting zone group 0 in order to have a private conversation.
- *Deviceinfo.MzcHasHandset* can be used to determine if the Analog phone (if configured) is Off-Hook or On-Hook
- All zones are automatically turned off at the completion of a SIP call

Console Audio

Using RTP Audio with TalkMaster FOCUS Server

The .NET SDK supports both TCP and RTP protocols for sending and receiving audio between the Console Applications and the TalkMaster FOCUS Server. Commands between the Console DLL and the Server are always sent/received using TCP, but audio can optionally be sent/received via UDP/RTP to prevent TCP delays from affecting console audio.

TCP is used by default. To have the TalkMaster FOCUS .NET SDK send/receive audio with the TalkMaster FOCUS Server via RTP, open the TalkMaster FOCUS Admin Console and verify that the *Use Console UDP/RTP if Possible* option is checked on the *Setup* >> *Settings* tab and note the Port that is being used. The TalkMaster FOCUS Server will ensure that RTP can be exchanged with the Console Application and will automatically revert to TCP if necessary.

Use the daClient.ConfigUdp(server address, server port, local port) method to enable RTP audio between the .NET SDK and the TalkMaster FOCUS Server.   Note that a local port of 0 specifies that any available local UDP port can be used, and it is allocated by the Windows TCP Stack.

Best Practices

When writing an application using the TalkMaster FOCUS .NET SDK, the following "best practices" should be considered. A C# template that incorporates these standards is included with the SDK:

- The first step in any program using the managed API is to create a globally available new instance of the DAConsole.ClassAccess object .
- The second step is to add the event handlers to the newly created ClassAccess object (See Event Descriptions section for more details)
- The next step is to call the Open method from the ClassAccess object.
- The next step is to use the ClassAccess.ConfigUDP() method so the console can communicate via UDP to the server.
- Then call the Start method from the ClassAccess object to connect the the TalkMaster FOCUS Server using a UserID and Password of an existing Operator record, and the IP Address and Port of the TalkMaster FOCUS Server.
- Once the Console Application is running, if network communication with the TalkMaster FOCUS Server is interrupted or if the TalkMaster FOCUS Server is restarted, the AccessEventHandler:EVENT_STOP event will be received and the application must reissue the Start method in order to to log back onto the server.
- Consoles Applications written by Digital Acoustics retry unsuccessful logons and if available, attempt to logon to TalkMaster FOCUS "Fail Forward" Servers. After connecting to a "Fail Forward" server, the application can optionally use the DATestConnection method from the ClassAccess object to determine if the Primary TalkMaster FOCUS Server is available again. If it is available, it can logoff the Fail Forward Server and then re-logon to the Primary. The "retry interval", "number of attempts before Fail Forward", and "automatic return to primary" options should be synchronized with the corresponding values configured in the IP Endpoints in TalkMaster Administrator Console..
- When TalkMaster FOCUS Server sends the UpdateEventHandler : EVENT_CONSOLE_UPDATE_AVAILABLE, the Console Application will issue the SendUpdatedSettings method from the ClassAccess object to reload all of the settings from the TalkMaster FOCUS Server (as if the Start method was just issued). The application may wait for a time when it is "idle" in order to do this

A C# template included with the .NET SDK demonstrates these Digital Acoustics Best Practices.

Summary of Objects

ClassAccess Objects	Description
ClassAccess	The main object “ClassAccess” is the object that controls the instance of the entire interface.
AudioClass	Access to the wave in/out devices available on the machine is available through this object.
AudioConfiguration	If the object is created with the default wave in/out device and codec, separate API calls are not needed to set up the object.
DACallQueueDataInterface	Returned in an event for each Call Queue entry, and gives an interface to control, play, and delete the entry.
DAConsoleGroupMessages	Provides access to pre-recorded Messages that are related to a Paging Group. Part of GroupsInterfaces. Collect them on your group event handler.
DAConsoleGroupsInterface	Provides access to the Paging Groups and the individual IP Endpoints associated with them. Passed with GroupEventHandler. Collect them on your group event handler.
DAConsoleInterface	Displays information about the console’s connection to the TalkMaster server. Informational only
DADeviceInterface	Provides an Interface to each IP Audio device on the system.
DAEventArgs	This object is delivered to each event handler used by this namespace. Only certain objects are used for each individual event handler.
DAQueueInterface	Provides access to the Call Queue status information this user belongs to, and whether it is overflowing etc...
DASysVarInterface	Provides access to System Variables. This object is passed during the SystemVariableEventHandler event handler. Each object is linked together from the “Root” object, but the linkage is also available from the Parent and Children objects.
OpenInfo	Passed as a result of DAOpen. The Flags are bit mask entries for the capability for this API, and for the user. Things like whether this is a remote or direct API. The user is an admin or not. The FullName of the logged in user.