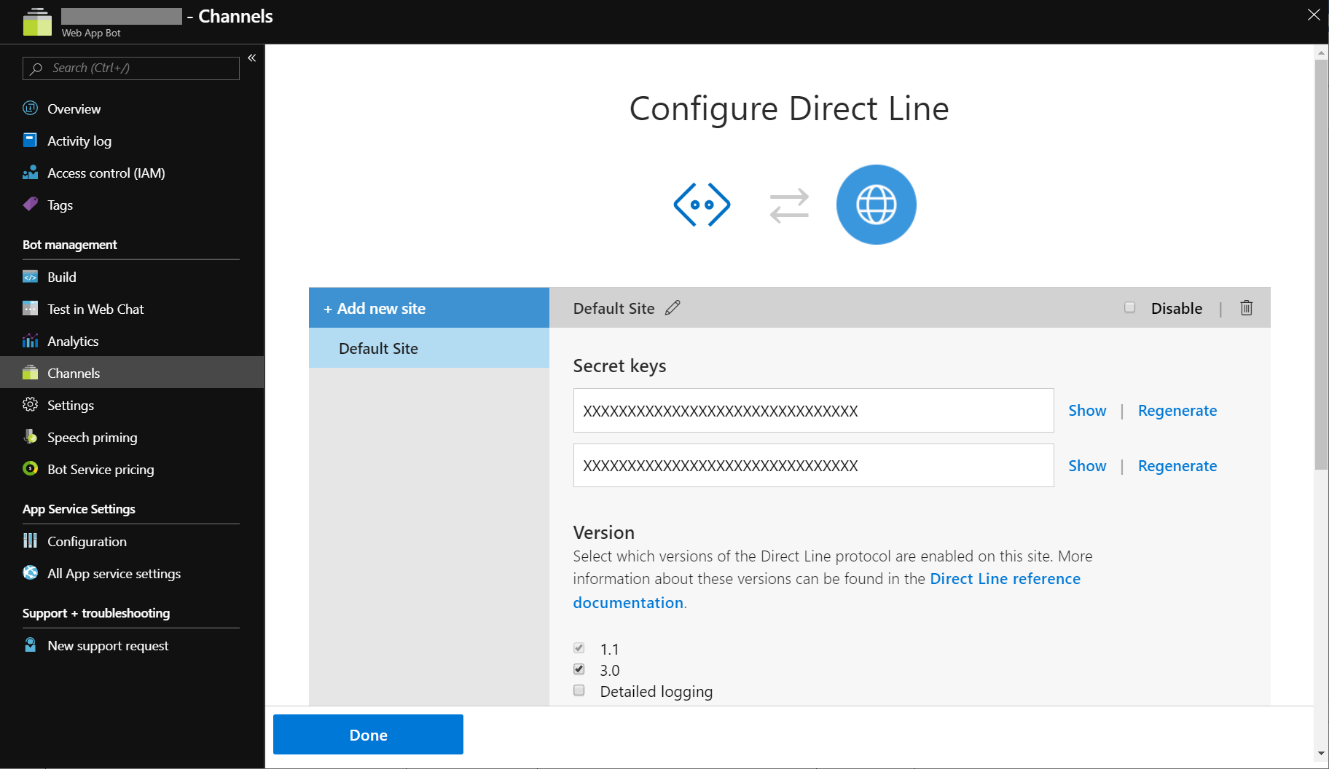
**How To: Create C# Clients to Connect to Direct Line App Service Extension**

This guide describes how to create a client to the Direct Line App Service Extension using C#.

***Gather your Direct Line Client Secret***

1. Open a web browser and go to <https://portal.azure.com/>
2. In the Azure Portal, locate your Azure Bot Service resource.
3. Click on “Channels” to configure the bot’s channels
4. If it is not already enabled, click on the Direct Line channel to enable it. If it is already enabled, in the Connect to channels table click on the “Edit” link on the Direct Line row.
5. Scroll to the Sites section (there is typically a Default Site unless you have deleted or renamed it). Click on the Show link to reveal one of the keys, then copy its value.



This value is your Direct Line Client Secret and can be used to connect to Direct Line App Service Extension. You can create additional sites if you’d like and use those secret values as well.

***Add the Preview Nuget Package Source***

All of the preview NuGet packages needed to create a C# Direct Line client can be found in a NuGet feed.

1. Open Visual Studio and navigate to the Tools 🡪 Options menu item.
2. Select the *NuGet Package Manager* 🡪 *Package Sources* item.
3. Click on the + button to add a new package source with these values:
   1. Name: DL ASE Preview
   2. Source: <https://botbuilder.myget.org/F/experimental/api/v3/index.json>
4. Click on the Update button to save the values, then click OK to exit the Package Sources configuration.

**Create a C# Direct Line Client**

Interactions with the Direct Line App Service Extension happen differently than traditional Direct Line in that most communication with Direct Line App Service Extension is done by opening a WebSocket and having the Direct Line commands flow through that WebSocket. The updated Direct Line Client includes helper classes for opening and closing this WebSocket, sending commands through the WebSocket, and receiving Activities back from the bot. This section describes how to create a simple C# client to interact with a bot:

1. Create a new .NET Core 2.2 console application project in Visual Studio.
2. Add the DirectLine client NuGet to your project
   1. Click on Dependencies in the Solution tree
   2. Select Mange Nuget Packages…
   3. Change the Package source to the one you defined from above (DL ASE Preview)
   4. Find the package “Microsoft.Bot.Connector.Directline” that is version v3.0.3-Preview1 or later.
   5. Click on Install Package.
3. Create a client and generate a token using a secret. This step is the same building any other C# Direct Line client except the endpoint you need use it your bot’s, appended with the ‘.bot’ path:

string endpoint = "https://<YOUR\_BOT\_HOST>.azurewebsites.net/.bot/";

string secret = "<YOUR\_BOT\_SECRET>";

var tokenClient = new DirectLineClient(

new Uri(endpoint),

new DirectLineClientCredentials(secret));

var conversation = await tokenClient.Tokens.GenerateTokenForNewConversationAsync();

1. Once you have a conversation reference from generating a token, you can use this conversation Id to open a WebSocket using the new StreamingConversations property on the DirectLineClient. To do this you need to create a callback that will be invoked when the Bot wans to send ActivitySets to the client:

public static void ReceiveActivities(ActivitySet activitySet)

{

if (activitySet != null)

{

foreach (var a in activitySet.Activities)

{

if (a.Type == ActivityTypes.Message && a.From.Id.Contains("bot"))

{

Console.WriteLine($"<Bot>: {a.Text}");

}

}

}

}

1. Now you are ready to open the WebSocket on the StreamingConversations property using the conversation’s token, conversationId, and your ReceiveActivities callback:

var client = new DirectLineClient(

new Uri(endpoint),

new DirectLineClientCredentials(conversation.Token));

await client.StreamingConversations.ConnectAsync(

conversation.ConversationId,

ReceiveActivities);

1. The client can now be used to start a conversation and send Activities to the bot:

var startConversation = await client.StreamingConversations.StartConversationAsync();

var from = new ChannelAccount() { Id = "123", Name = "Fred" };

var message = Console.ReadLine();

while (message != "end")

{

try

{

var response = await client.StreamingConversations.PostActivityAsync(

startConversation.ConversationId,

new Activity()

{

Type = "message",

Text = message,

From = from

});

}

catch (OperationException ex)

{

Console.WriteLine(

$"OperationException when calling PostActivityAsync: ({ex.StatusCode})");

}

message = Console.ReadLine();

}