**Integrando Dialogflow com Cortana**

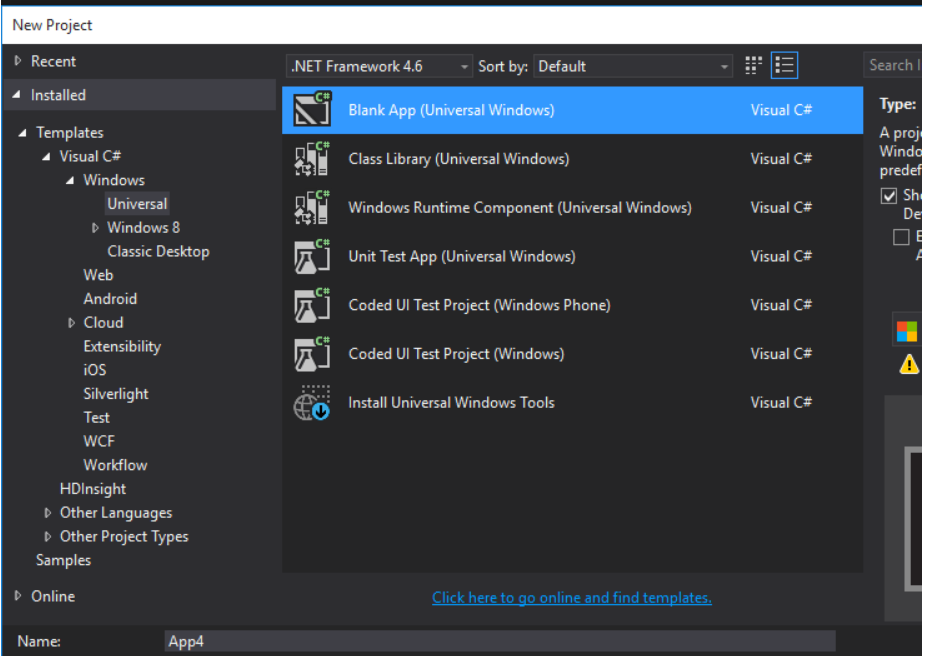
Visão geral

O **Integração Cortana** permite exportar os seus agentes num [Formato compatível de VCD de Cortana](https://msdn.microsoft.com/en-us/library/windows/apps/xaml/dn706593.aspx) e usá-lo para construir aplicativos.

Abaixo estão as instruções.

## Preparação

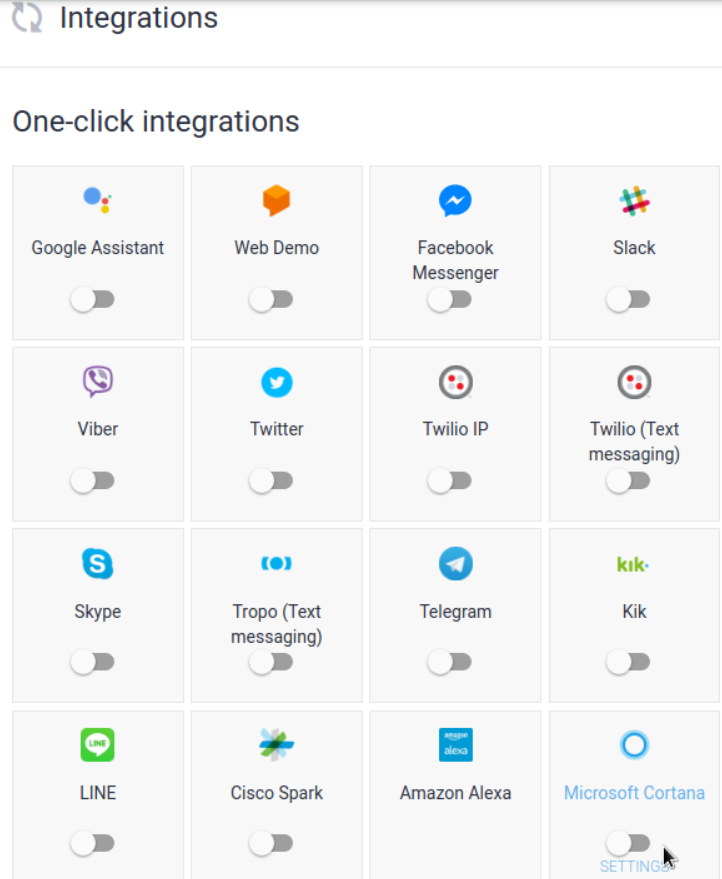
Criar um projeto universal do Windows no Visual Studio.



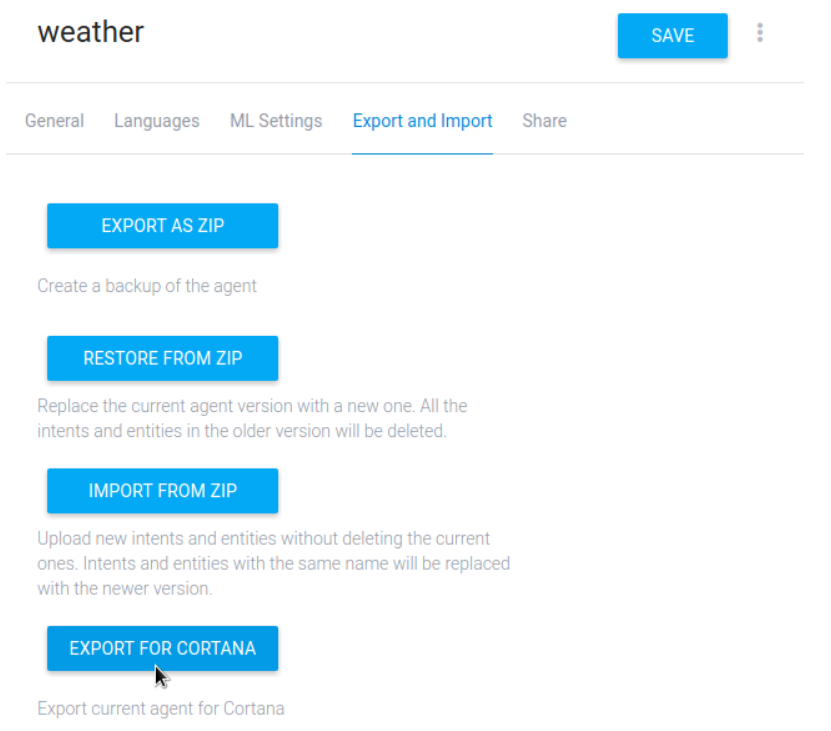
Adicione uma referência à biblioteca Dialogflow .NET usando [Nuget](https://www.nuget.org/packages/ApiAiSDK/).

Crie uma nova página de aplicação e adicione um botão de microfone/escuta. Em seguida, adicione TextBlock controle para exibir resultados.

No console do desenvolvedor do Dialogflow, vá para **Integrações** no menu do lado esquerdo e habilitar a integração do Microsoft Cortana.



Baixe um arquivo de comandos de voz Cortana do console do desenvolvedor do Dialogflow > configurações do agente > exportar e importar.



Descompacte e copie o \_W10. xml arquivo para o seu projeto. Renomeá-lo VoiceCommands. xml. Definir o **Ação de compilação** do arquivo para Conteúdo (Content).

Adicionar um novo campo de AIService Digite a classe App.

C#

public AIService AIService { get; private set; }

Inicializar o aiService campo no construtor de aplicativo.

C#

var config = new AIConfiguration("CLIENT\_ACCESS\_TOKEN", SupportedLanguage.English);  
AIService = AIService.CreateService(config);

Há duas opções de integração em Cortana:

* Registre os padrões de comando de voz e Cortana irá abrir o seu aplicativo e passar alguns parâmetros para ele. [Official Cortana documentation](https://msdn.microsoft.com/en-us/library/windows/apps/xaml/dn630430.aspx)
* Registre o serviço de comando de voz para fornecer lógica personalizada para processamento de solicitações. O serviço de comando de voz pode fornecer respostas sem abrir seu aplicativo. [Official Cortana documentation](https://msdn.microsoft.com/en-us/library/windows/apps/xaml/dn974228.aspx)

## Integração com comandos de voz

Adicionar código para registrar VoiceCommands. xml no final de Em lançamento Método.

C#

try  
{  
    var storageFile = await StorageFile.GetFileFromApplicationUriAsync(new Uri("ms-appx:///VoiceCommands.xml"));  
    await AIService.InstallVoiceCommands(storageFile);  
}  
catch (Exception ex)  
{  
Debug.WriteLine(ex);  
}

Implementar Onactivated método para receber parâmetros de Cortana. O Dialogflow SDK tem um método pre-built para processamento de parâmetros de inicialização de aplicativo (passado por Cortana) – AIService. ProcessOnActivatedAsyncpara o processamento. Este método chama Dialogflow para obter uma ação de Dialogflow com base na entrada de texto/fala.

C#

protected async override void OnActivated(IActivatedEventArgs e)  
{  
    AIResponse aiResponse = null;  
    try  
    {  
        aiResponse = await AIService.ProcessOnActivatedAsync(e);  
    }  
    catch (Exception)  
    {  
    // ignored  
    }  
  
    NavigateToMain(aiResponse);  
}  
  
private void NavigateToMain(AIResponse aiResponse)  
{  
    Frame rootFrame = Window.Current.Content as Frame;  
  
    // Do not repeat app initialization when the Window already has content,  
    // just ensure that the window is active  
    if (rootFrame == null)  
    {  
        // Create a Frame to act as the navigation context and navigate to the first page  
        rootFrame = new Frame();  
  
        rootFrame.NavigationFailed += OnNavigationFailed;  
  
        // Place the frame in the current Window  
        Window.Current.Content = rootFrame;  
    }  
  
    rootFrame.Navigate(typeof(MainPage), aiResponse);  
  
    // Ensure the current window is active  
    Window.Current.Activate();  
}

Na página principal do aplicativo, adicione o processamento para o AIResponse Resultado. Por exemplo, a saída para um Textblock.

C#

protected override void OnNavigatedTo(NavigationEventArgs e)  
{  
    base.OnNavigatedTo(e);  
  
    var response = e.Parameter as AIResponse;  
    if (response != null)  
    {  
        var aiResponse = response;  
        resultTextBlock.Text = JsonConvert.SerializeObject(aiResponse, Formatting.Indented);  
    }  
}

## Integração com o serviço de comando de voz

Adicionar o projeto de componente do Windows Runtime à solução.

Na parte Extensões nó do seu aplicativo Package.appxmanifest Adicione as seguintes tags XML.

XML

<Extensions>  
    <uap:Extension Category="windows.appService"   
        EntryPoint="DialogflowDemo.VoiceCommands.DialogflowVoiceCommandService">  
      <uap:AppService Name="DialogflowVoiceCommandService" />  
    </uap:Extension>  
    <uap:Extension Category="windows.personalAssistantLaunch"/>  
</Extensions>

Adicionar classe DialogflowVoiceCommandService ao seu componente de tempo de execução do Windows e adicione o seguinte código à sua classe VoiceCommandService.

C#

public sealed class DialogflowVoiceCommandService : IBackgroundTask  
{  
    public async void Run(IBackgroundTaskInstance taskInstance)  
    {  
    }  
}

Além disso, você vai precisar BackgroundTaskDeferral para esperar até que o comando é processado, e até VoiceCommandServiceConnection interage com Cortana e Dialogflow interage com o serviço Dialogflow.

C#

private BackgroundTaskDeferral serviceDeferral;  
private VoiceCommandServiceConnection voiceServiceConnection;  
private Dialogflow Dialogflow;

No método Run iremos processor Cortana request usando os seguintes passos.

Loja BackgroundTaskDeferral instância para esperar até que o trabalho seja concluído.

C#

serviceDeferral = taskInstance.GetDeferral();  
taskInstance.Canceled += OnTaskCanceled;

Inicializar Dialogflow Instância.

C#

var config = new AIConfiguration("YOUR\_CLIENT\_ACCESS\_TOKEN",  
                SupportedLanguage.English);  
  
Dialogflow = new Dialogflow(config);

Obter AppServiceTriggerDetails para se VoiceCommandServiceConnection.

C#

var triggerDetails = taskInstance.TriggerDetails as AppServiceTriggerDetails;  
  
if (triggerDetails != null)  
{  
    voiceServiceConnection = VoiceCommandServiceConnection.FromAppServiceTriggerDetails(triggerDetails);  
    voiceServiceConnection.VoiceCommandCompleted += VoiceCommandCompleted;  
    var voiceCommand = await voiceServiceConnection.GetVoiceCommandAsync();  
...

Usar VoiceCommandServiceConnection para receber o texto da solicitação e o nome do comando.

C#

var recognizedText = voiceCommand.SpeechRecognitionResult?.Text;  
var voiceCommandName = voiceCommand.CommandName;

Verifique o nome do comando para casos diferentes. Por exemplo, você pode:

* Faça solicitações para Dialogflow e inicie seu aplicativo com AIResponse do Dialogflow.
* Faça o pedido ao Dialogflow e emita a resposta a Cortana com SendResponseToCortanaAsync Método. *(por exemplo, veja o processamento diferente para "tipo" e "desconhecidos" comandos de voz abaixo)*

C#

switch (voiceCommandName)  
{  
    case "type":  
        var aiResponse = await Dialogflow.TextRequestAsync(recognizedText);  
        await Dialogflow.LaunchAppInForegroundAsync(voiceServiceConnection, aiResponse);  
        break;  
    case "unknown":  
        var aiResponse = await Dialogflow.TextRequestAsync(recognizedText);  
        if (aiResponse != null)  
        {  
        await Dialogflow.SendResponseToCortanaAsync(voiceServiceConnection, aiResponse);  
        }  
        break;  
}

Naturalmente, você precisará de envolver o código inteiro com um bloco try...catch e tonar a tarefa concluída no Finalmente Bloco.

C#

try  
{  
    ...  
}  
catch(Exception e)  
{  
    var message = e.ToString();  
    Debug.WriteLine(message);  
}  
finally  
{  
    serviceDeferral?.Complete();  
}

O código completo do serviço será parecido com este.

C#

public sealed class DialogflowVoiceCommandService : IBackgroundTask  
{  
    private BackgroundTaskDeferral serviceDeferral;  
    private VoiceCommandServiceConnection voiceServiceConnection;  
    private Dialogflow Dialogflow;  
  
    public async void Run(IBackgroundTaskInstance taskInstance)  
    {  
        serviceDeferral = taskInstance.GetDeferral();  
        taskInstance.Canceled += OnTaskCanceled;  
  
        var triggerDetails = taskInstance.TriggerDetails as AppServiceTriggerDetails;  
  
        if (triggerDetails != null)  
        {  
            var config = new AIConfiguration("YOUR\_CLIENT\_ACCESS\_TOKEN", SupportedLanguage.English);  
  
            Dialogflow = new Dialogflow(config);  
            Dialogflow.DataService.PersistSessionId();  
  
            try  
            {  
                voiceServiceConnection = VoiceCommandServiceConnection.FromAppServiceTriggerDetails(triggerDetails);  
                voiceServiceConnection.VoiceCommandCompleted += VoiceCommandCompleted;  
                var voiceCommand = await voiceServiceConnection.GetVoiceCommandAsync();  
                var recognizedText = voiceCommand.SpeechRecognitionResult?.Text;  
                var voiceCommandName = voiceCommand.CommandName;  
  
                switch (voiceCommandName)  
                {  
                    case "type":  
                        {  
                            var aiResponse = await Dialogflow.TextRequestAsync(recognizedText);  
                            await Dialogflow.LaunchAppInForegroundAsync(voiceServiceConnection, aiResponse);  
                        }  
                        break;  
                    case "unknown":  
                        {  
                            if (!string.IsNullOrEmpty(recognizedText))  
                            {  
                                var aiResponse = await Dialogflow.TextRequestAsync(recognizedText);  
                                if (aiResponse != null)  
                                {  
                                    await Dialogflow.SendResponseToCortanaAsync(voiceServiceConnection, aiResponse);  
                                }  
                            }  
                        }  
                        break;  
  
                    case "greetings":  
                        {  
                            var aiResponse = await Dialogflow.TextRequestAsync(recognizedText);  
  
                            var repeatMessage = new VoiceCommandUserMessage  
                            {  
                                DisplayMessage = "Repeat please",  
                                SpokenMessage = "Repeat please"  
                            };  
  
                            var processingMessage = new VoiceCommandUserMessage  
                            {  
                                DisplayMessage = aiResponse?.Result?.Fulfillment?.Speech ?? "Pizza",  
                                SpokenMessage = ""  
                            };  
  
                            var resp = VoiceCommandResponse.CreateResponseForPrompt(processingMessage, repeatMessage);  
                            await voiceServiceConnection.ReportSuccessAsync(resp);  
                            break;  
                        }  
  
                    default:  
                        if (!string.IsNullOrEmpty(recognizedText))  
                        {  
                            var aiResponse = await Dialogflow.TextRequestAsync(recognizedText);  
                            if (aiResponse != null)  
                            {  
                                await Dialogflow.SendResponseToCortanaAsync(voiceServiceConnection, aiResponse);  
                            }  
                        }  
                        else  
                        {  
                            await SendResponse("Cannot recognize");  
                        }  
  
                        break;  
                    }  
  
            }  
            catch(Exception e)  
            {  
                var message = e.ToString();  
                Debug.WriteLine(message);  
            }  
            finally  
            {  
                serviceDeferral?.Complete();  
            }  
        }  
    }  
      
    private void OnTaskCanceled(IBackgroundTaskInstance sender, BackgroundTaskCancellationReason reason)  
    {  
        serviceDeferral?.Complete();  
    }  
  
    private void VoiceCommandCompleted(VoiceCommandServiceConnection sender, VoiceCommandCompletedEventArgs args)  
    {  
        serviceDeferral?.Complete();  
    }  
  
    private async Task SendResponse(string textResponse)  
    {  
        var userMessage = new VoiceCommandUserMessage  
        {  
            DisplayMessage = textResponse,  
            SpokenMessage = textResponse  
        };  
  
        var response = VoiceCommandResponse.CreateResponse(userMessage);  
        await voiceServiceConnection.ReportSuccessAsync(response);  
    }  
}