Getting Started with Skills

Introduction

Welcome to the Store portal. This Getting Started guide is intended for anyone that is new to the world of creating skills and wants to know how to get started. It is also intended for experienced developers who write code in languages like C# or Python and need to know how they can bring their code into the Workspace portal and start building skills.

Whether you are a new developer or an experienced developer, you will learn how to create a basic skill. You will also learn how to bring in your code created in other languages or a Bot Framework bot and be able to chain them together in a skill.

Purpose

The purpose of this tutorial is to teach you the fundamentals of how to create a simple skill, test your skill's invocation out using the Test Chat Tool, and help you become familiar with the concepts that make up the composition of flows.

Use Case Scenario

Today, you hear all the rage about all the diverse types of digital devices that enable users to interact with them in numerous ways that give back many kinds of responses and data types.

Examples of such devices can be things like fitness watches or home entertainment systems that take in a user's voice commands, and then output the responses by finding their favorite TV program or telling them what the five day weather forecast will be in their region.

You might be wondering how does this all work. You also might be asking yourself, "How can I develop a skill that will make my digital device respond to things I want it to find, say, or do?"

The simple answer is described below in the "Hello World!" code sample written in the Semantic Composition Language (SCL). The major goal of SCL is composition. What this means is that you can compose APIs, C# code, Bot Framework bots into a skill using SCL. For simplicity purposes, we are using SCL in creating the skill for this tutorial.

There are two major steps in skills creation. These are:

1. Creating a botlet with metadata in the Workspace.

Note: This step is mandatory because you cannot write code for a skill without having a basic botlet shell under a parent organization directory.

2. Write the code for the skill.

Example: **Hello World!**

1. Create a botlet by using the **+ Create New** wizard in the Workspace.

For more details about creating botlets, see: Botlets.

2. Enter the following SCL syntax in the Code Viewer, and then save the code.

```
SAY "Hello World!"
```

3. Use the Test Chat Tool to test your skill's functionality by typing a message (e.g., "Hi") in the text field.

Note: The Test Chat Tool is an internal runtime channel that is used for testing a botlet's functionality. For more details about using the Test Chat Tool and creating test cases on a botlet, Test Chat Tool.

4. Connect your skill to a channel like Cortana.

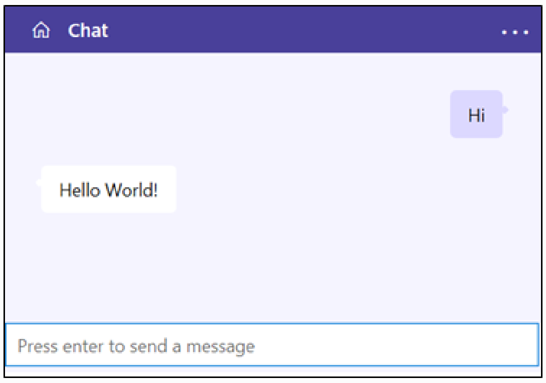
For more details about publishing botlets, see: Publishing.

Response Outcome

Pressing Start new session should give you the response "Hello World!" in the Chat window, or if you type a message in the text field (e.g., "Hi"), the response given back should say "Hello World!".

Additional Information

The SCL syntax used in this example sends text to the botlet identified in the line starting with `SAY` "Hello World!". The text that follows `SAY` gets displayed into a single bubble in the Test Chat Tool.

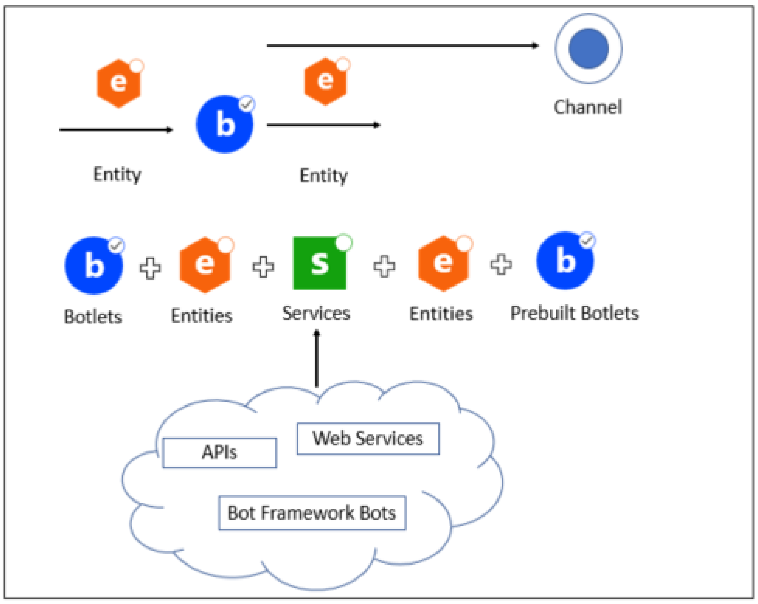


Hi

Hello World!

Now that you've had a chance to create a simple skill, the information that follows will describe the concepts of a compositional flow. Furthermore, the compositional flow concepts will be reinforced by breaking down a skill's anatomy using a flower ordering botlet as an example.

The diagram shown below shows the compositional flow of a skill. As you will notice, a flow is composed of botlets. A botlet can consist of other botlets, service calls, and prebuilt botlets connected together with SCL. Examples of such botlets can be anything like a multiplication game or a taxi ordering app.



Services are how botlets communicate to the external world. Services allow you to register a web-hosted service, Bot Framework bot or API in the Store so that it can be used in a flow.

We know that many code components are repetitive and reused in multiple flows. That's why Store contains lots of prebuilt botlets that accomplish these tasks for you. A benefit of using these prebuilt botlets is that you don't need to create code from scratch and they behave consistently across all platforms and canvases. Entities are used to chain together all these flow elements and pass data between them.

Finally, when the botlet logic is ready, it can be published to a channel like Cortana.

If you want your botlet to be semantic (i.e., easily chainable with other botlets) and interoperable with other skills, define actions as its interface.

For more information about services and actions, see: Services.

Take as an example a flower ordering botlet. This type of botlet can consume the name of the flower and its color. It can also send out the flower order number in such a way that it can be reused in other flows and connected to, for example, delivery botlet. This refers to the concept of botlet "chaining."

Example: Flower Ordering Skill

Let's assume you want to create a flower ordering skill. It can contain several elements.

It can get some information about the person who is ordering the flowers. It can allow the user to order a courier service. The skill can also allow the user to add a gift item to the order like a box of chocolates.

These additions to the flower order are handled by sub botlets. For instance, you might already have the logic that finds a courier written in C#. You might have found a public API that calls a courier that is hosted in the cloud. Additionally, a prebuilt botlet in the Store can find the location of the user. You can combine all these elements into a skill by registering the first two components as services in the Store and connecting them to the third one using entity types.

Let's assume that you found the location of a user who this order is supposed to be sent to. The botlet sends out the actual location, and then you take in the courier service that consumes this location. These are the elements of the flow. You "stitch" these botlets together using SCL.

When you know that your skill flow is complete and functioning as expected, you may publish it to the Cortana channel.

Previous

Next