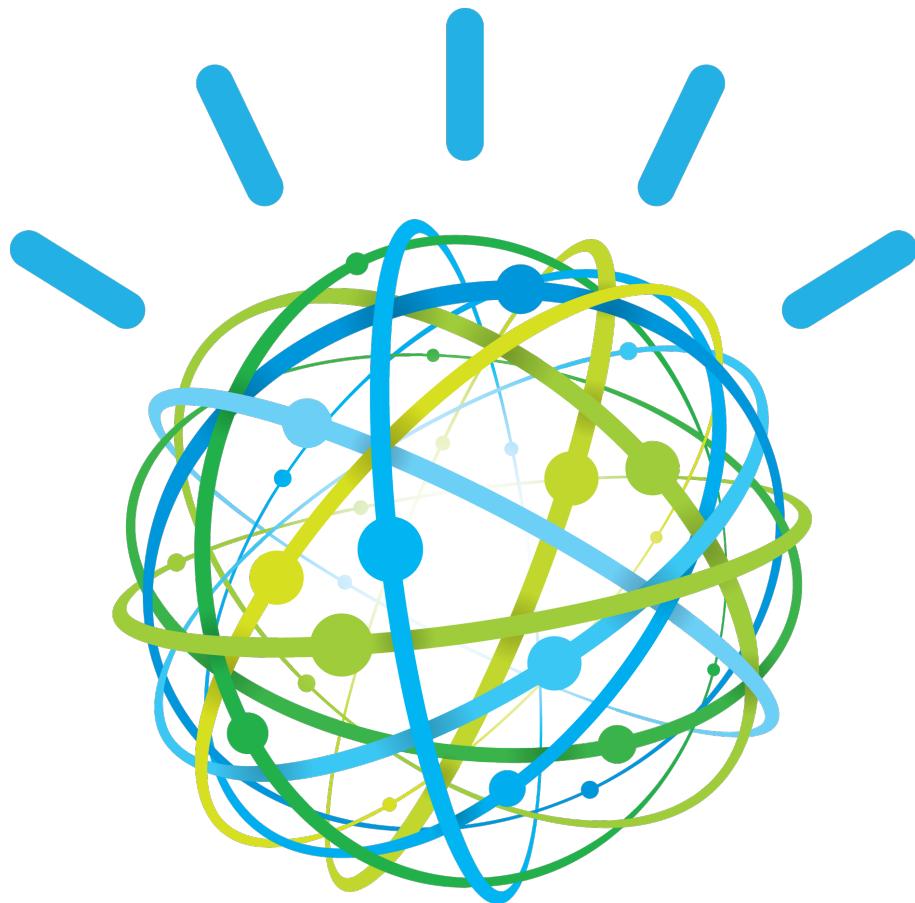


Lab 1: Building a conversation

# IBM Watson Assistant



## Lab Instructions

Laurent Vincent

# Content

<b>Let's get started .....</b>	<b>3</b>
1. Overview .....	3
2. Objectives .....	3
3. Prerequisites .....	3
4. Scenario.....	3
5. What to expect when you are done .....	4
<b>Create an Assistant in IBM Cloud .....</b>	<b>5</b>
6. Create your Watson Assistant Service .....	5
7. Manage your first Assistant .....	7
8. Manage your first Skill .....	11
<b>Intents .....</b>	<b>13</b>
9. Group your user's inputs by intents .....	13
10. Create your first intent .....	18
11. Import intents .....	20
12. Test and improve your Intents .....	22
13. Content Catalog.....	28
<b>Entities.....</b>	<b>30</b>
14. Create your first entity.....	30
15. Import entities .....	33
16. Test your entities .....	34
17. Contextual entities .....	35
18. Entity pattern.....	40
19. Test your pattern.....	41
20. Enable system entities.....	41
21. Test System Entities .....	42
<b>Conversation Dialog.....</b>	<b>44</b>
22. Create your first Dialog .....	44
23. Create your first nodes .....	49
24. Test your conversation .....	51
<b>Enrich your data by using context variables .....</b>	<b>54</b>
<b>Save your skill to your computer (optional).....</b>	<b>56</b>

# Let's get started

---

## 1. Overview

The [IBM Watson Developer Cloud](#) (WDC) offers a variety of services for developing cognitive applications. Each Watson service provides a Representational State Transfer (REST) Application Programming Interface (API) for interacting with the service. Some services, such as the Speech to Text service, provide additional interfaces.

The [Watson Assistant](#) (WA) service combines several cognitive techniques to help you build and train a bot - defining intents and entities and crafting dialog to simulate conversation. The system can then be further refined with supplementary technologies to make the system more human-like or to give it a higher chance of returning the right answer. Watson Assistant allows you to deploy a range of bots via many channels, from simple, narrowly focused bots to much more sophisticated, full-blown virtual agents across mobile devices, messaging platforms like Slack, or even through a physical robot.

The **illustrating screenshots** provided in this lab guide could be slightly different from what you see in the Watson Assistant service interface that you are using. If there are colour or wording differences, it is because there have been updates to the service since the lab guide was created.

## 2. Objectives

In this lab, you will:

- Learn how to use the IBM Cloud web user interface to create and manage Watson services
- Learn how to train your chat bot to answer some asked questions

## 3. Prerequisites

Before you start the exercises in this guide, you will need to complete the following prerequisite tasks:

- Download all files needed for this lab using the link provided by the instructor provided you the link to get labs content. You may download each file individually.

## 4. Scenario

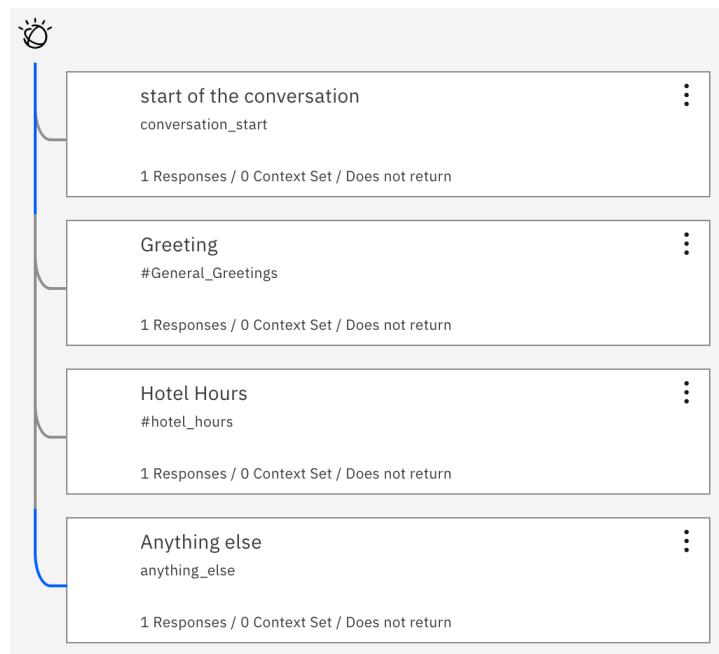
**Use case:** A Hotel Concierge Virtual assistant that is accessed from the guest room and the hotel lobby.

**End-users:** Hotel customers

## 5. What to expect when you are done

At the end of this lab, you should have a simple dialog using

- around 23 intents,
- around 11 entities.



# Create an Assistant in IBM Cloud

## 6. Create your Watson Assistant Service

IBM Cloud offers services, that provide additional functionality that is ready to use for your application's running code.

You have two options for working with applications and services in IBM Cloud: via web user interface or the command-line interface. In this lab, you will use the web user interfaces.

1. In a web browser, navigate to the following URL:

Location	URL
worldwide	<a href="https://cloud.ibm.com/login">https://cloud.ibm.com/login</a>

2. Log in with your IBM Cloud credentials. This should be your IBM ID.

You should start on your dashboard that contains metrics related to your IBM Cloud activities.

The screenshot shows the IBM Cloud dashboard. On the left, there's a sidebar with icons for various services like VPC, Clusters, Cloud Foundry apps, and Services. The main area has three main sections: 'Récapitulatif des ressources' showing 486 resources (Devices: 116, Storage: 243, VPC infrastructure: 5, Clusters: 1, Apps: 13, Cloud Foundry apps: 3, Developer tools: 16, Cloud Foundry services: 3, Schemas workspaces: 4, Services: 47), 'Infrastructure classique' with links to Device list, Support cases, User list, Subnets, Network monitoring, Block Storage, and Compliance reports; and a 'Pour vous' section for Watson Assistant and Watson Studio.

3. Click on **Create resource** (top right of the page)

Create resource

4. In the search box, enter **Watson Assistant** as filter

The screenshot shows the IBM Cloud Catalog search results for 'watson assistant'. A red box highlights the first result, 'Watson Assistant', which is described as letting you build conversational interfaces into any application, device, or channel. It is listed under the 'AI' category and is marked as 'Lite • Free • IAM-enabled'.

5. Click on the **Watson Assistant** service tile.
6. Review the details for this service. To see a description of the service, open the tab **About**. After you viewed the description, open the tab **Create**, and review the pricing plans. Notice that the Plus trial provides an access to all features for 30 days.
7. At the top, you enter the geographical location for your new service. Select location [Frankfurt](#). If Frankfurt is not available, choose another location close to you, such as London.

**Note:** remember which location you chose! You will use this same location in next labs.

8. In the pricing plan, select [Plus Trial](#).

9. At the bottom, fill out the fields as follows.

Field	Value
Service name	<a href="#">Conversation-XXXXXX-LV</a>
Resource group	default

Replace **XXXXXX** with the date of the creation with the format yymmdd.

Replace **LV** with your initials.

10. Then click **Create** (bottom right).

**Note:** You can provision only one Lite or Plus Trial instance of a Watson Assistant service at a time. So, if you already have such instance, delete it now and create a new one as shown in previous steps.

IBM Cloud will create a new service instance. In order to use this specific instance in your application, you will need to obtain its credentials. The credentials are displayed in the **Manage** tab or in **Service credentials** tab. The credentials consist of **API Key** and **URL** for your service instance.

The screenshot shows the IBM Cloud Watson Assistant interface. On the left, there's a sidebar with 'Manage' selected, followed by 'Service credentials', 'Plan', and 'Connections'. The main area has a heading 'Start by launching the tool' and three buttons: 'Launch Watson Assistant' (blue), 'Getting started tutorial' (white with blue border), and 'API reference'. Below this is a 'Credentials' section with fields for 'API key:' and 'URL:', both of which are highlighted with a red box. At the top of the credentials section are 'Download' and 'Show credentials' buttons.

If you want to view the API key, click on **Show Credentials**. In later exercises, you will enter these credentials into a JSON configuration file for your application. Copy now the credentials to a temporary file, or just return to this IBM Cloud page later when you will need the credentials.

#### 11. Click the **Launch Watson Assistant** button.

Your first step in the Watson Assistant tool will be to create an Assistant and a Skill.

## 7. Manage your first Assistant

An Assistant is a virtual container to which you add your skills. Assistant enables a skill for interaction with your customers.

The Watson Assistant tool has created for you your first Assistant, your first Skill and named them **My first assistant** and **My first skill**.

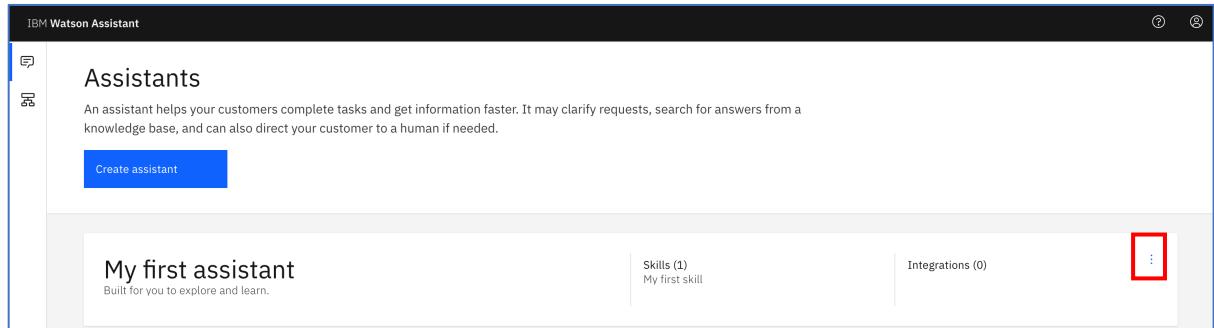
1. You should get a start tour invitation window. Just click outside the window to close it.

2. Rename your Assistant to **Concierge** by following these steps:

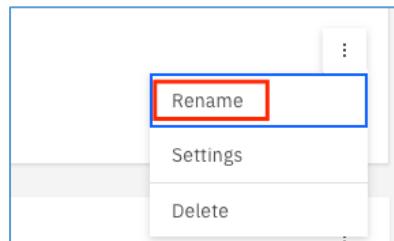
- Go to **Assistants** page, by clicking **Assistants** icon top left.

The screenshot shows the 'Assistants' page. At the top, there's a banner for 'IBM Watson Assistant Plus trial | 30 days left | Upgrade'. Below this is a list of assistants: 'My first skill' (selected, indicated by a red box around its icon), 'Intents', 'Entities', 'Dialog', 'Options', and 'Analytics'. The 'My first skill' item has a 'Version: Development' badge.

- b. In the **My first assistant** tile, click on 3 dots on the right-hand side.



- c. Select **Rename**.



- d. Rename your Assistant to **Concierge** and give it a description as shown below.

Rename Assistant

Choose a name that describes your assistant's purpose, such as "Banking help" or "Sales bot."

Name 64 characters max

Concierge

Description (optional)

For concierge demo

Cancel Save

- a. Close **Save**.

You should see your Assistant with the new name: *Concierge*. You can also see that your Assistant has one Skill which is named *My first skill*. You are going to update the name of the skill later on.

The screenshot shows the 'Assistants' section of the IBM Watson Assistant interface. A card for the 'Concierge' assistant is displayed. The card includes the assistant's name ('Concierge'), a description ('For concierge demo'), a 'Skills (1)' section (highlighted with a red box) containing 'My first skill', an 'Integrations (0)' section, and a three-dot menu icon. A 'Create assistant' button is located at the top left of the main area.

3. Click on the **Concierge** tile
4. (Top right) Click **Add integration**

The screenshot shows the detailed view of the 'Concierge' assistant. The 'Actions' section (Beta) contains a list of benefits for building conversations, with a 'Learn more' link and a 'Add an actions skill' button. The 'Dialog' section shows the 'My first skill' configuration, including language (English (US)), trained data (0 intents, 0 entities, 2 dialog nodes), version (draft), description (\*\*\*), and creation date (Jan 15, 2021 10:11 AM CET). The 'Integrations' section (Beta) shows options to 'Integrate web chat' and 'Choose a channel to deploy your assistant' (with a red box around the 'Add integration' button).

5. Here you can pick several different integration types for your Assistant, such as Web Chat, Facebook Messenger etc. You will choose **Preview link**, so click on it.

The screenshot shows the 'Stand-alone integrations' section. It lists two options: 'Web chat' (represented by a blue icon with a speech bubble) and 'Preview link' (represented by a black icon with a double arrow). Below each option is a brief description and supported service desks (Zscaler, Zendesk, and ServiceNow). At the bottom, there are 'Phone', 'Beta', and 'Plus' buttons.

The Preview link is a URL at which your Assistant is available to the outside world.

6. Keep the default setting and click **Create**

7. An URL will be presented to you. This is the external URL (the preview link) of your Assistant.

### Preview link integration

Integration name

Preview Link

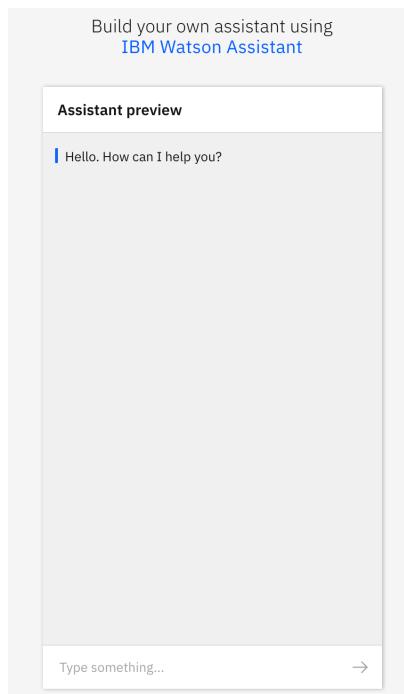
Try it out and share the link

Use of the assistant embedded in this web page incurs billing charges. ⓘ

<https://integrations.us-south.assistant.watson.cloud.ibm.com/web/public/2b4e85ca-bf32-40a7-80a2-550ff384e478>

**Save changes**

8. Click on the **URL** to see how your Assistant looks to external world. A new tab will open in your browser, looking like this:



9. Your new assistant greets you and waits for your input. **Bookmark** this URL, you will use it later to work with your Assistant as an external user.

10. Go back to the window with the Preview link integration and click **Save changes**.

The screenshot shows a web-based configuration interface for a preview link integration. At the top, it says "Preview link integration". Below that, "Integration name" is set to "Preview Link". Underneath, there's a section titled "Try it out and share the link" which includes a note about billing charges and a generated URL: <https://integrations.us-south.assistant.watson.cloud.ibm.com/web/public/2b4e85ca-bf32-40a7-80a2-550ff384e478>. A blue "Save changes" button at the bottom is highlighted with a red rectangular border.

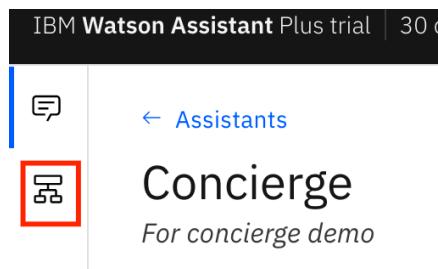
You have successfully managed your first Assistant and created an external URL (preview link).

## 8. Manage your first Skill

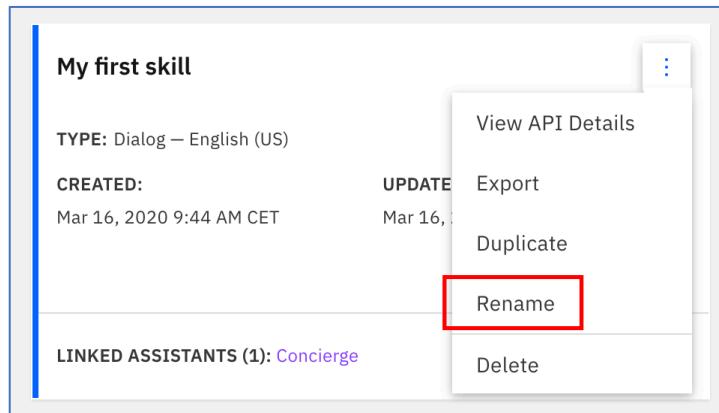
To use the new Assistant, you will need to train it with the intents, entities, and/or dialog nodes relevant to your application's use case. You will create these items in a Skill. A Skill is a container for the artefacts that define a piece of the behaviour of your service instance.

You are going to update the pre-created Skill for the lab. For the exercise, you will use the English (US) language which is the default language of a skill. As we cannot update the language of a skill, If you want to use another language (not for the lab as all content is in English), you must delete the default skill and create a new one with the right language.

1. In the menu on the left, Click to **skills** icon



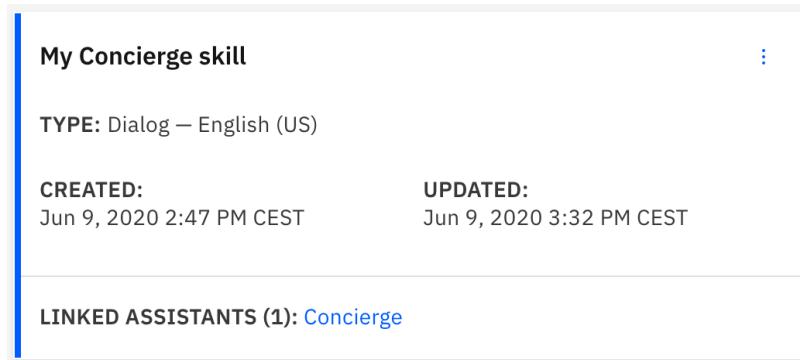
2. In **My first skill** tile, click on three dots on the right, then click **Rename**.



3. Rename your skill to *My Concierge Skill*, give it a description as shown below and click the **Rename** button.

The screenshot shows a modal dialog titled "Rename My first skill". It contains two input fields: "Name" with the value "My Concierge skill" and "Description (optional)" with the value "For concierge demonstration". At the bottom of the dialog is a blue "Rename" button.

4. You should now see your skill with the new name.



Note: the language of the skill is English (US)

# Intents

In order to use the new skill, you will need to train it with the intents, entities, and/or dialog nodes relevant to your application's use case. First, you will need to define some intents to help control the flow of your dialog. An *intent* is the purpose or goal of a user's input. In other words, Watson will use natural language processing to recognize the intent in the user's question / statement. This will help Watson select the corresponding dialog branch in your automated conversation.

## 9. Group your user's inputs by intents

The initial activity should be to organise the user's examples you have collected, and group them by intents. To speed up this step we are going to leverage Intent recommendations feature of WA.

1. To open the skill page, click on the tile **My Concierge Skill**
2. If not already there, click the **Intents** tab at the top of your skill.
3. On the **Intents** tab, click **intent recommendations** in the dialog window.

What is an intent?

An intent is a collection of user statements that have the same meaning. By creating intents, you train your assistant to understand the variety of ways users express a goal. [Learn more](#)

You will find some pre-made intents in the content catalog. [Browse content catalog](#)

[Create intent](#) + [Intent recommendations](#) [Import intents](#)

4. The data source is a CSV file, click **upload CSV** in the dialog window.

 Missing data source

There is no data source available to analyze for recommendations. Add a data source.

Note: Logs from a live assistant are stored for 30 days. If no customers interact with it during that time, then no log data will be available.

[Connect live assistant](#) [Upload CSV files](#)

5. Click **Add file** in the dialog window,  
then select the file [WA\\_Lab\\_examples Before Session 4.csv](#).

The file is in folder **Labs Sources**. You should have downloaded this folder from the workshop materials before starting this lab. If you haven't already, download it now.

Watson is grouping your utterances.

## Intent Recommendations Plus

Watson helps you find the intents your users need most by grouping the most common and similar user examples. Recommended intents are sourced from connected live assistants and CSV files. You can manage your recommendation sources [here](#).

Watson is grouping your utterances

Note: periodically, refresh your page and go back to the intent recommendation to see if you get the result of the job.

After the execution you should have several proposed intents:

Recommended intents  
Source: CSV files

going_rain_today <i>is it going to rain It is going to rain Is it going to rain?</i>	10 ↗	whats_weather <i>what's the weather how is the weather? tell me the weather</i>	10 ↗	whats_temperature_outside <i>What's the temperature outside? how hot is it outside hi, what is the temperature</i>	7 ↗
will_rain_end <i>when will this rain end? how long till it rains When will the rain stop</i>	7 ↗	recommend_good_restaurant <i>Can you recomend a restaurant? I am looking for a restaurant Can you recommend a Japanese restaurant?</i>	6 ↗	cold <i>It's cold here it's cold today it's getting cold</i>	5 ↗
pool <i>Where is the pool Is the pool heated? Is the pool outside?</i>	5 ↗	rain <i>what is the rain? when rain ends how long the rain?</i>	5 ↗	will_sun <i>when will the sun begin? When will it be sunny? when will the sun set?</i>	5 ↗
gym <i>Whereabouts is the gym? Where is the gym? When is the gym open?</i>	4 ↗	hear <i>I didn't hear that I didn't hear you I couldn't hear you</i>	4 ↗	order_pizza <i>Can I order a pizza without pineapple Can I order a pizza with more cheese Can I get a pizza margarita; small please</i>	4 ↗

6. Click on the tile **order\_pizza**

## 7. Select all relevant examples to order a pizza

### Review and create

Select any utterances that you want to add. Choose whether to create the intent or add the utterances to an existing intent as new user examples.

[Create new intent](#)    [Add to existing intent](#)

Intent name  
order\_pizza

4 items selected

User utterance

- Can I order a pizza without pineapple
- Can I order a pizza with more cheese
- Can I get a pizza margarita; small please
- Can I get a hawaiian pizza with no pineapple

## 8. Click **Create new intent**

In this case, the Intent [#order\\_pizza](#) will be created with 4 examples.

## 9. We are going to enrich the intent, click **Show recommendations**

[←](#) | #order\_pizza

Intent name  
Name your intent to match a customer's question or goal  
#order\_pizza

Description (optional)  
Add a description to this intent

User example  
Add unique examples of what the user might say. (Pro tip: Add at least 5 unique examples)

Type a user example here, e.g. I want to pay my credit card bill

Add example    [Show recommendations](#)

10. On the right frame, select all examples related to “order a pizza”, and click **Add**

The screenshot shows a 'Recommendations' interface with a red border. At the top, there are tabs for 'Recommendations' and 'Search Logs'. Below the tabs, a message says 'Select recommended user examples to add'. A blue header bar indicates '7 items selected' with 'Add' and 'Cancel' buttons. A grey section titled 'Recommended user examples' contains a list of 10 items, each with a checkbox and a description. The first 9 items have checked boxes, while the last one is unchecked. The descriptions include various pizza orders like 'large hawaiian with extra pineapple', 'pizza with onions, beef, chicken and extra cheese', and 'order a pizza'.

Description
I'd like a large hawaiian with extra pineapple
can I have a pizza with onions, beef, chicken and extra cheese
I want to order a pizza
I want to order a large pizza
Can you deliver a vegetarian pizza
can I have a carnivore pizza but no chicken
Can I get a pizza with extra red peppers but no green peppers
Can I order food at the pool?

11. You could continue to import example form the CSV File but go back to the intent page by clicking on the arrow (top left)

The screenshot shows the IBM Watson Assistant interface. At the top, it says 'IBM Watson Assistant'. Below that is a search bar containing '#order\_pizza'. To the left of the search bar is a blue back arrow icon. Below the search bar is a light grey bar labeled 'Intent name'.

12. Go back to the intent recommendation page by clicking on **intent recommendations**

The screenshot shows the 'Intent recommendations' page. At the top, there are icons for sorting and a 'Recommendation sources' dropdown. Below that is a blue header bar with 'Intent recommendations' (which has a red box around it), 'Create intent', and a '+' button. The main area is a table with columns: 'Intents (0) ↑', 'Description', 'Modified ↑', 'Conflicts ↑', and 'Examples ↑'. One row is visible, showing '#order\_pizza' in the 'Description' column, '5 minutes ago' in the 'Modified' column, and '10' in the 'Examples' column.

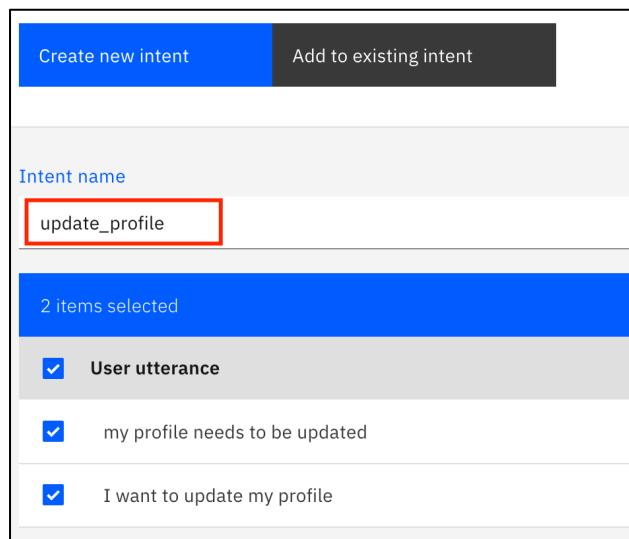
Intents (0) ↑	Description	Modified ↑	Conflicts ↑	Examples ↑
	#order_pizza	5 minutes ago		10

Note: The order\_pizza proposition should have been removed.

13. Scroll down, you should find a tile **profile**, click on it.



14. Select all utterances, update the name with [update\\_profile](#).



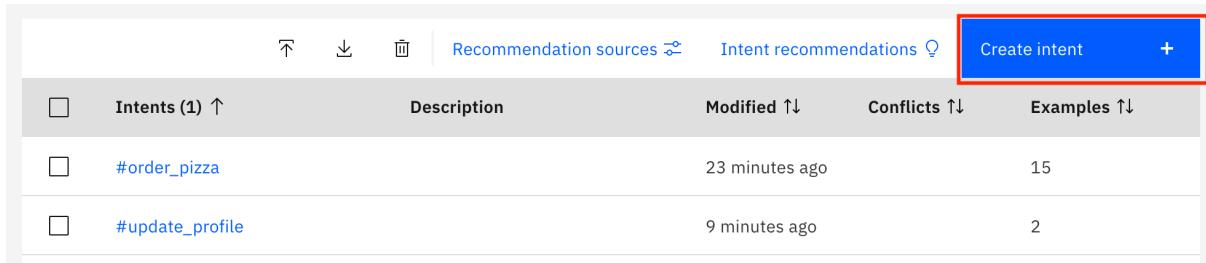
15. Click **create new intent**.

We could continue to analyse the classification proposed by WA. Right now, we are going to use other WA capabilities.

## 10. Create your first intent

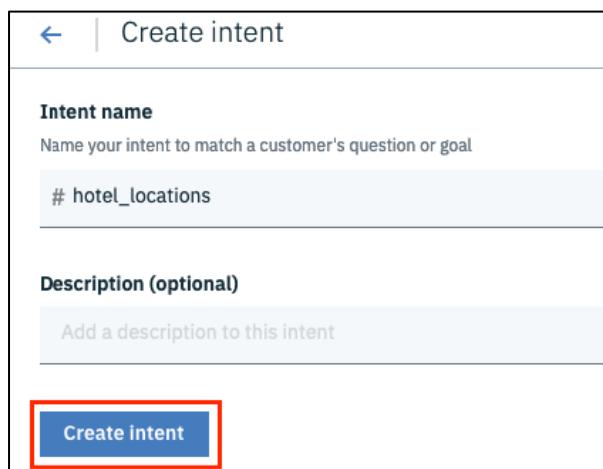
16. If not already there, Go back to the **Intents** tab of your skill.

17. Click **Create intent**.



Intents (1) ↑	Description	Modified ↑↓	Conflicts ↑↓	Examples ↑↓
<input type="checkbox"/> #order_pizza		23 minutes ago		15
<input type="checkbox"/> #update_profile		9 minutes ago		2

18. Enter *hotel\_locations* as Intent name then click **Create intent**



← | Create intent

**Intent name**  
Name your intent to match a customer's question or goal

**Description (optional)**

**Create intent**

19. Enter the examples as defined below in the **User example** field. Click **Add example** button after entering each example. The *user examples* are phrases that will help Watson recognize the new *intent*.

Field	Value
Intent name	<i>hotel_locations</i>
User example	<i>Is the pool outside?</i> <i>Where is the gym?</i> <i>Do you have a sauna?</i>

The screenshot shows the Watson Assistant interface for creating a new intent. The intent name is '#hotel\_locations'. The 'User example' field contains three examples: 'Is the pool outside?', 'Where is the gym?', and 'Do you have a sauna?'. These examples are listed in a table with columns for checkbox, example text, added time, and conflicts.

	User examples (3) ↑	Added ↑↓	Conflicts (0) ↑↓
<input type="checkbox"/>	Do you have a sauna?	a few seconds ago	
<input type="checkbox"/>	Is the pool outside?	a few seconds ago	
<input type="checkbox"/>	Where is the gym?	a few seconds ago	

20. When finished, click on the **arrow** icon (top left side of the window) to go back to list of existing intents

At this point, you have defined 3 *intents* for the application along with the example utterances that will help train Watson to recognize and control the conversation flow.

## 11. Import intents

Apart of creating intents manually (as you did in previous step), intents can also be imported into the Watson Assistant tool using a Comma Separated Values (CSV) file saved with UTF-8 encoding.

In order to import the CSV file successfully, the file must include only two columns:

- one column for user examples
- one column for intents.

A CSV file that we'll use in this exercise has already been created for you. It was created this way:

- Copy the end-user examples into a single column.
- Label each end-user example with its assigned intent in the column beside it.
- Do not insert any column headings for the intent import file.
- Use appropriate text editor (such as Numbers on Mac and Notepad++ on Windows) to make sure your file is UTF-8 coded and saved as a .csv file. Using the right text editor is important because many text processing software (such as MS Word) add spaces and other hidden characters into a text when it is copied and pasted into a new source. When such file is uploaded into Watson Assistant tooling, spaces might be converted into symbols that will negatively affect learning by adding text that the user did not intend.

### 1. Click **Import Intents** icon

<input type="checkbox"/>	Intents (3) ↑	Description	Conflicts ↑↓	Examples ↑↓
<input type="checkbox"/>	#hotel_locations			3

### 2. Click on **Choose a file** then [WA\\_Lab\\_Intents Before Session 4.csv](#).

Import intents

Select a CSV file with the appropriate formatting to import intents and examples. [Learn more](#)

10MB max file size

Choose a file

WCS\_Lab\_intents Before Session 4.csv

Cancel Import

### 3. Click **Import**



**Note:** Notice that the Watson Assistant tooling did not import examples that were exact duplicates of questions that you manually entered. Duplicates are not case sensitive. For example, “Do you have a sauna?” is the same as “do you have a sauna”. It did import questions that had semantic differences, such as the absence of punctuation, extra spaces between words, or misspelled words.

#### 4. Click **Done**, you will see a number of new intents.

My Concierge skill Version: Development

Intents

Entities  
Dialog  
Options  
Analytics  
Versions  
Content Catalog

Intents (13) ↑	Description	Modified ↑↓	Conflicts ↑↓	Examples ↑↓
#around_hotel		4 minutes ago	9	
#eat		4 minutes ago	8	
#exit		4 minutes ago	9	
#feedback		4 minutes ago	8	
#hotel_hours		4 minutes ago	16	
#hotel_info		4 minutes ago	27	
#hotel_locations		20 minutes ago	26	
#hotel_procedure		4 minutes ago	10	

Save new version Try it

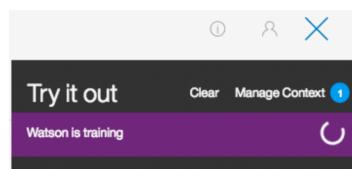
## 12. Test and improve your Intents

You can use the **Try it** panel for testing your chatbot during initial development.

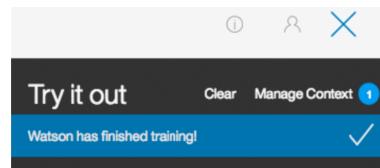
1. Open the **Try it** panel by clicking on its icon (upper right):

The screenshot shows the Watson Assistant interface. At the top, there's a navigation bar with 'Version: Development' on the left, a search icon, 'Save new version', and a red box highlighting the 'Try it' button on the right. Below the navigation is a table header with columns: 'Intents (14) ↑', 'Description', 'Modified ↑↓', 'Conflicts ↑↓', and 'Examples ↑↓'. Underneath the header, there's a single row showing an intent named '#around\_hotel' with a modified timestamp of '2 minutes ago' and 9 examples.

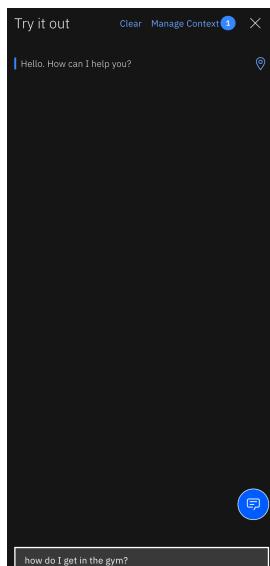
When WA is training on the recently added data, you will see a message highlighted with purple: "Watson is training".



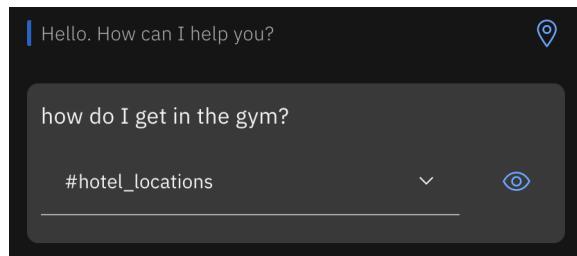
Wait until this message is cleared before you test newly added intents, utterances, entities. Otherwise, Watson will respond, but you will get unpredictable results because the training is not complete yet. When training is finished, the purple box with the text, "Watson is training" will disappear and for a short time you will see a message in blue box saying: "Watson has finished training!"



2. In the input field at the bottom of the Try it panel, type in *how do I get in the gym?*



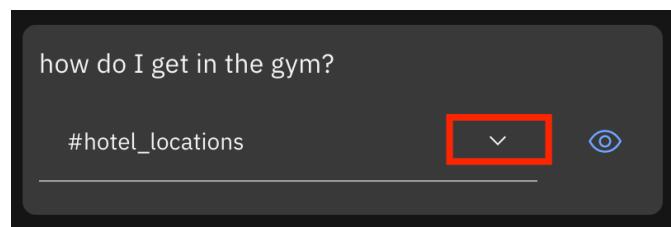
3. Press enter



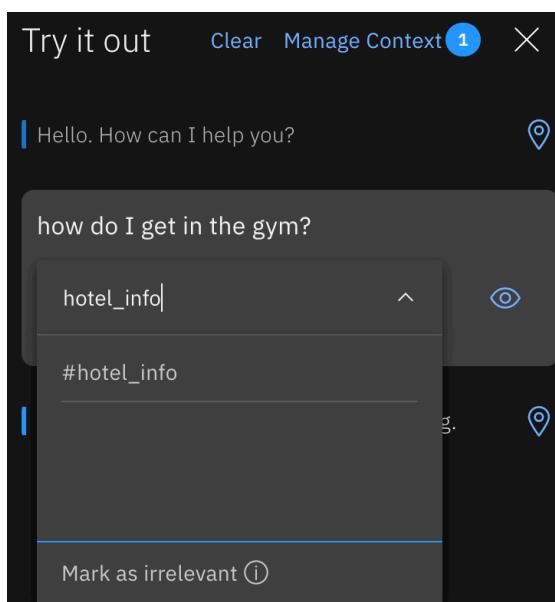
Notice the grey box with the text that you entered. Watson assigned your question to the intent `#hotel_locations`. The `#hotel_locations` intent is meant for questions that refer to finding directions to places within the hotel. However in this case, the user wants to know how to get in the gym, not how to get to the gym. A small difference in wording can change the correct intent for a particular utterance.

Let's change the intent for this user example to avoid misinterpretation.

4. Click the arrow next to `#hotel_locations`

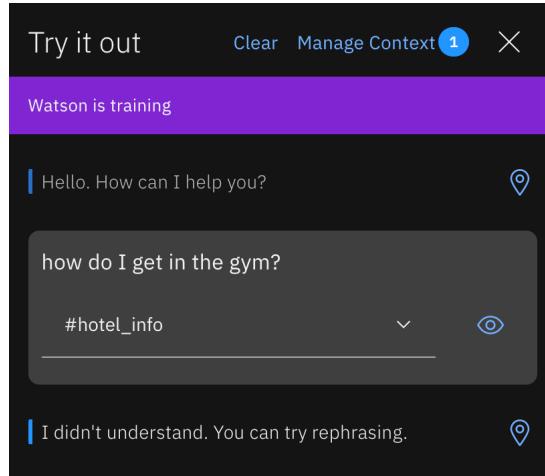


5. Type in `hotel_info`



6. Select the intent `#hotel_info` that appears in the drop-down box

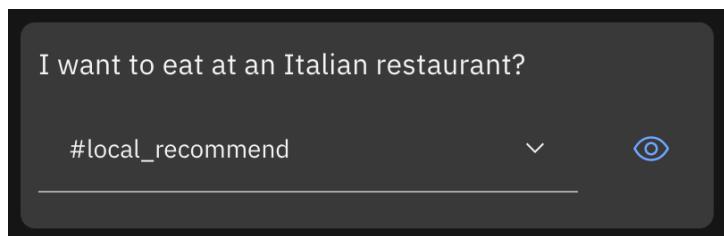
Your **try it out** panel will show a message “*Watson is training*”



**Note:** If you go to your Intents panel, and open the `#hotel_info` intent, you will now see that your input has been added to this section. As this user's example has been used in the intent `#hotel_locations` this update should cause a conflict between the 2 intents. In another lab you are going to use ‘Watson Conflict Resolution’ feature to fix it.

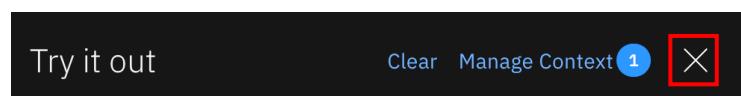
7. Now we will practice changing intent using another technique. Type in Type *I want to eat at an Italian restaurant?*

**Note:** Notice that Watson assigns this question the intent `#Local_recommend`. (*the classification could differ*). This intent is meant for questions that refer to finding recommendation for sightseeing, restaurant in the hotel. In this case, the user wants to eat something.



Let's say we do not like such classification and we want to move our example to another intent. This time, we will do it in a different way - by using the search capability of the engine.

8. Close the **Try it out** Panel by clicking the X on top right.



## 9. Top right, click on the search icon

The screenshot shows the 'My Concierge Skill' interface with 'Version: Development'. On the left, there's a sidebar with 'Intents', 'Entities', 'Dialog', 'Options', and 'Analytics'. The main area is titled 'Intents' with a sub-section 'Intents (13) ↑'. It includes columns for 'Description', 'Modified ↑', 'Conflicts ↑↓', and 'Examples ↑↓'. A single intent entry is visible: '#around\_hotel' modified 5 minutes ago with 9 examples.

## 10. Type in **eat** in the search field and press enter.

If you see the message “Your content is being indexed...”, press the refresh button on your browser occasionally until the message disappears. Then resubmit your query.

The screenshot shows a search interface with a search bar containing 'eat'. Below it, a message says 'Your content is being indexed. This is a one-time process. Resubmit your query shortly.' A magnifying glass icon is shown above the message. The text 'No results yet.' is displayed. Below that, instructions say 'Intent search will return results grouped by the intent in which they appear. You can try searching for user examples, intent names, and descriptions.' A 'Learn more' link is at the bottom.

The search returns all intents including the '**eat**' word.

The screenshot shows a search interface with a search bar containing 'eat'. Below it, a message says 'Showing 7 intents containing: 'eat''. A list of intents is shown, each with a collapse arrow and a count of matches: #weather (21 matches), #repeat\_request (4 matches), #eat (4 matches), #hotel\_locations (1 match), #local\_recommend (5 matches), #hotel\_procedure (1 match), and #hotel\_info (1 match).

11. Let refine the research, enter for the sentence *I want to eat at an* in the search field, then expand the returned intent *#local\_recommend*.

I want to eat at an

Showing 1 intent containing:

'I want to eat at an'

^ #local\_recommend  
1 match

User example  
*I want to eat at an Italian ...*

12. We see a list of examples that belongs to this intent (only one in our case). The above example looks like the one we used during the test, so click on it.
13. On the left, Watson opens the *#local\_recommend* intent with all its user examples and it highlights our example.

← | #local\_recommend

User example  
Add unique examples of what the user might say. (Pro tip: Add at least 5 unique examples to help Watson understand)

Type a user example here, e.g. I want to pay my credit card bill

Add example Show recommendations

User examples (33) ↑

Hi! Where's a hole in the wall restaurant that's not on your list and you would take your famil

I want to eat at an Italian restaurant. What are the best ones around?

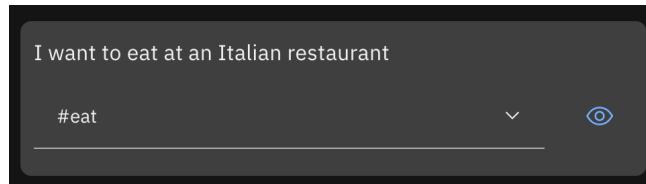
14. Select the example by checking the box next to it, then click on **Move** in order to move this example to another intent.

The screenshot shows the Watson Assistant interface with a modal dialog titled "Move examples". At the top left of the dialog is a "User examples (33) ↑" button. On the right side, there are buttons for "Move ↪", "Delete", and "Cancel". Below these buttons, there are two rows of user examples. The first row contains an unchecked checkbox followed by the text "Hi! Where's a hole in the wall restaurant that's not on your list and you would take your family to." and the timestamp "an hour ago". The second row contains a checked checkbox followed by the text "I want to eat at an Italian restaurant. What are the best ones around?" and the timestamp "an hour ago".

15. Enter **eat** as intent select it then click on **Move examples**.

The screenshot shows the "Move examples" dialog with a search bar containing the text "eat". Below the search bar, a list of intents is displayed: "#eat" (highlighted in blue), "#weather", and "#repeat\_request". At the bottom right of the dialog is a blue "Move examples" button.

**Note:** Now, your user example is moved to the intent **#eat**. If you will do a new test, the system will return the right intent. (Make sure that Watson finished training).



You have seen two options to improve the Watson classification:

- add a new user example in an existing intent,
- move an existing user example to the right intent.

You are going to review several other options during this workshop.

## 13. Content Catalog

**Content Catalogs** provide an easy way to add common intents to your Watson Assistant service skill.

You are going to add to your Skill general conversation topics intents from the Content Catalog.

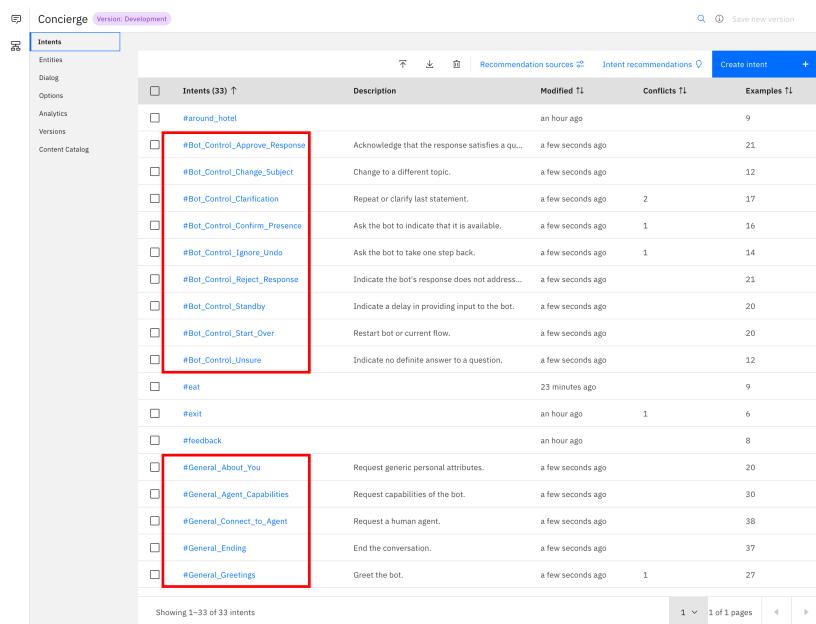
1. If you are not on your Skill page, press the left arrow on top left, to get there.
2. Click on the **Content Catalog** tab, then click **Add to skill** to add **General** and **Bot Control** categories into your training set as shown in the picture.

Intents	Entities	Dialog	Options	Analytics	Versions	Content Catalog
Get started faster by adding existing intents from the content catalog. These intents are trained on questions that customers commonly ask.						

Category	Description	Intents	Add to skill +
Banking	Basic transactions for a banking use case.	13	Add to skill +
Bot Control	Functions that allow navigation within a conversation.	9	Add to skill +
Customer Care	Understand and assist customers with information about...	18	Add to skill +
eCommerce	Payment, billing, and basic management tasks for order...	14	Add to skill +
General	General conversation topics most users ask.	10	Add to skill +
Insurance	Issues related to insurance policies and claims.	12	Add to skill +
Mortgage	Common questions related to the mortgage industry	20	Add to skill +
Telco	Questions and issues related to a user's telephony service...	21	Add to skill +
Utilities	Help a user with utility emergencies and their utility services...	10	Add to skill +

3. Go back to **Intents** tab, you will find new intents with prefixes **#General\_** and **#Bot\_Control\_**. If you do not see the **#General\_** intents, scroll down.



The screenshot shows the Watson Assistant interface with the 'Intents' tab selected. A sidebar on the left lists various service tabs: Concierge (selected), Entities, Dialog, Options, Analytics, Versions, and Content Catalog. The main area displays a table of intents. A red box highlights a group of intents under the 'Bot Control' category: #Bot\_Control\_Approve\_Response, #Bot\_Control\_Change\_Subject, #Bot\_Control\_Clarity, #Bot\_Control\_Confirm\_Presence, #Bot\_Control\_Ignore\_Undo, #Bot\_Control\_Reject\_Response, #Bot\_Control\_Standby, #Bot\_Control\_Start\_Over, and #Bot\_Control\_Unsure. Another red box highlights a group of intents under the 'General' category: #General\_About\_You, #General\_Agent\_Capabilities, #General\_Connect\_to\_Agent, #General\_Ending, and #General\_Greetings. The table columns include Intent name, Description, Modified time, Conflicts count, and Examples count.

Intent	Description	Modified	Conflicts	Examples
#around_hotel	an hour ago	9		
#Bot_Control_Approve_Response	Acknowledge that the response satisfies a question.	a few seconds ago	21	
#Bot_Control_Change_Subject	Change to a different topic.	a few seconds ago	12	
#Bot_Control_Clarity	Repeat or clarify last statement.	a few seconds ago	2	17
#Bot_Control_Confirm_Presence	Ask the bot to indicate that it is available.	a few seconds ago	1	16
#Bot_Control_Ignore_Undo	Ask the bot to take one step back.	a few seconds ago	1	14
#Bot_Control_Reject_Response	Indicate the bot's response does not address the question.	a few seconds ago	21	
#Bot_Control_Standby	Indicate a delay in providing input to the bot.	a few seconds ago	20	
#Bot_Control_Start_Over	Restart bot or current flow.	a few seconds ago	20	
#Bot_Control_Unsure	Indicate no definite answer to a question.	a few seconds ago	12	
#eat	23 minutes ago	9		
#exit	an hour ago	1	6	
#feedback	an hour ago	8		
#General_About_You	Request generic personal attributes.	a few seconds ago	20	
#General_Agent_Capabilities	Request capabilities of the bot.	a few seconds ago	30	
#General_Connect_to_Agent	Request a human agent.	a few seconds ago	38	
#General_Ending	End the conversation.	a few seconds ago	37	
#General_Greetings	Greet the bot.	a few seconds ago	1	27

Let's remove all useless pre-created intents.

4. Select the following intents:

*Bot\_control\_Change\_Subject,  
Bot\_control\_Confirm\_Presence,  
Bot\_control\_Ignore\_Undo,  
Bot\_control\_Standby,  
Bot\_control\_Unsure,  
General\_Jokes,  
General\_About\_you,  
General\_Human\_or\_Bot,  
General\_Security\_Assurance.*

5. Click Delete

9 items selected				
	Intents (32) ↑	Description	Modified ↑↓	Conflicts ↑↓
	Examples ↑↓			
<input type="checkbox"/>	#around_hotel		9 minutes ago	9
<input type="checkbox"/>	#Bot_Control_Approve_Response	Acknowledge that the response sat...	3 minutes ago	21
<input checked="" type="checkbox"/>	#Bot_Control_Change_Subject	Change to a different topic.	3 minutes ago	12

You should have 23 intents.

# Entities

## 14. Create your first entity

To use the new conversation, you will need to train it with not only with intents, but also with entities and dialog nodes relevant to your application's use case. An [entity](#) is the portion of the user's input that you can use to provide a different response or action to an intent.

**Note:** for every skill, you will need to create some entities. Typically, an entity is an object or subject of our conversation. Entities can be used to help clarify users' questions/phrases. You should only create entities for things that matter and might alter the way a bot responds to an intent.

1. Click on **Entities / My entities** menu on the left-hand side.

The screenshot shows the 'Concierge Virtual Assistant' interface. On the left, a sidebar menu includes 'Intents', 'Entities' (which is currently selected and highlighted in blue), 'My entities' (selected), 'System entities', 'Dialog', 'Options', 'Analytics', 'Versions', and 'Content Catalog'. To the right of the sidebar is a circular icon with two nodes connected by a line. Below the icon, the text 'What is an entity?' is displayed. A detailed description follows: 'Entities are like nouns or keywords. By building out your business terms in entities your assistant can provide targeted responses to queries.' A 'Learn more' link is provided. Further down, it says, 'You can also enable pre-built system entities to capture common phrases such as dates, times and numbers.' At the bottom of the screen, there are three buttons: 'Create entity' (highlighted with a red box), '+', and 'Import entities'.

2. As you did for intent, create your first entity manually by clicking **Create entity**. In this lab, several intents will indicate that the user is looking for a restaurant. So, you will need to create a new entity representing different restaurant categories. You will name your first entity [restaurant](#), give it a value [pizza\\_restaurant](#) and enter synonyms as shown below ([pizza restaurant](#), [pizza](#), [pizza pie](#) and [pizzeria](#)).

3. Type in *restaurant* as entity name, then click **Create entity**

Entity name  
Name your entity to match the category of values that it will detect.

@restaurant

Create entity

4. Fill it like below, value *pizza\_restaurant* and enter synonyms *pizza restaurant*, *pizza*, *pizza pie* and *pizzeria* by clicking on plus sign next to a synonym.

Entity name  
Name your entity to match the category of values that it will detect.

@restaurant

Fuzzy matching  On

Value

pizza\_restaurant

Synonyms

pizza restaurant

pizza pie

pizza

pizzeria

Type a synonym, e.g Deposit +

Add value Recommend synonyms

5. Click **Add Value** (bottom left)

6. Fill the second value like below, value *french\_restaurant* and enter synonyms *french restaurant*, *brasserie* by clicking on plus sign next to a synonym.

Entity name  
Name your entity to match the category of values that it will detect.

@restaurant

Value

french\_restaurant

Synonyms

french restaurant

brasserie

Type a synonym, e.g Deposit +

Add value Recommend synonyms

Dictionary (1) Annotation (0) Beta

<input type="checkbox"/> Values (1) ↑	Type
<input type="checkbox"/> pizza_restaurant	Synonyms pizza restaurant, pizza, pizza pie, pizzeria

7. Click **Add Value** (bottom left)

The screenshot shows the Entity creation interface. At the top, there's a red-bordered header bar with a back arrow, the entity name '@restaurant', and various status indicators like 'Last updated: a few seconds ago' and a 'Try it' button. Below this is a form for defining the entity. It has sections for 'Value' (with a text input field containing 'Type a value, e.g. Checking'), 'Synonyms' (with a dropdown menu set to 'Synonyms' and a text input field for 'Type a synonym, e.g Deposit'), and buttons for 'Add value' and 'Recommend synonyms'. A red rectangle highlights the 'Fuzzy matching' toggle switch, which is turned 'On'. Below these are tabs for 'Dictionary (2)' (which is selected) and 'Annotation (0) Beta'. The main table lists two entries under 'Values': 'french\_restaurant' and 'pizza\_restaurant', each with its type listed as 'Synonyms'.

Values (2) ↑	Type
french_restaurant	Synonyms french restaurant, brasserie
pizza_restaurant	Synonyms pizza restaurant, pizza, pizza pie, pizzeria

8. Enable the **Fuzzy Matching** option if it is not already enabled. You will find it on top right (see red rectangle above).
9. When finished, go back to the Entities page by clicking on the left arrow on top.

## 15. Import entities

Entities, like intents can be imported into the Assistant tool using a Comma Separated Values (CSV) file saved with UTF-8 encoding. This way you can quickly get many entities into your skill, rather than manually creating them.

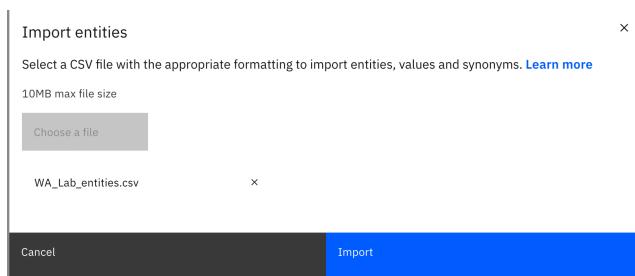
In order to import the CSV file successfully, the file must include a minimum of two columns:

- first column for entity name
- second column for entity value
- optional columns for the synonyms. Each synonym must be in separate column

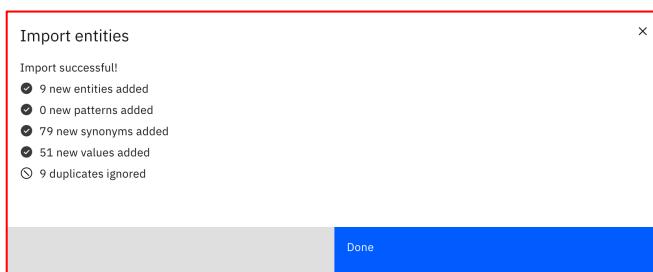
1. Return to your skill, on the Entities tab, click **Import entities** icon. (top right)

The screenshot shows the 'My Concierge skill' Entities tab. A red box highlights the 'Import entities' button at the top right. Below it, there's a table with one row. The first column has a checkbox and the text 'Entity (1) ↑'. The second column contains a checkbox and the text '@restaurant'. The third column is labeled 'Values' and contains 'french\_restaurant, pizza\_restaurant'. At the bottom right of the table, it says 'Modified ↑' and '2 minutes ago'.

2. Click **Choose a file** then select file *WA\_Lab\_Entities.csv*. You should have downloaded this file from the workshop materials, it is in the folder Lab Sources.



3. Click **Upload**.

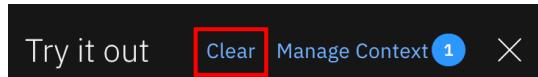


4. Click **Done**.

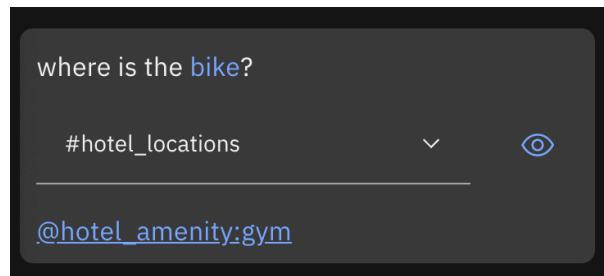
**Note:** the Assistant tooling did not import any examples that were exact duplicates of entities that you manually entered. So, the existing restaurant entity name, values, synonyms were ignored.

## 16. Test your entities

1. Open the **Try it out** panel, once Watson is done training.
2. Click on **Clear** to clear your panel from previous messages.

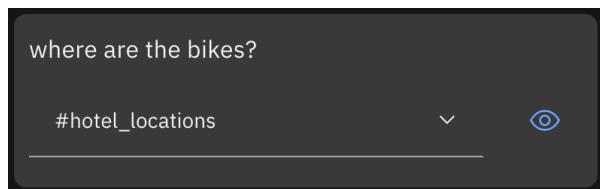


3. Type in *where is the bike?*



**Note:** Notice that Watson has identified the term *bike* with the correct intent, entity, and entity value (@*entity\_name*:*entity\_value*).

4. Type *where are the bikes?*



**Note:** Notice that Watson does not identify *bikes* the same as it identified *bike*. This is because entity matching is brute-force rules-based string matching without fuzzy matching.

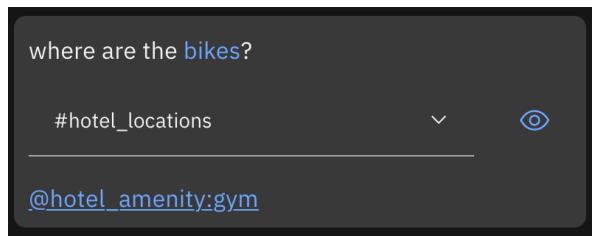
Let's turn on fuzzy matching to see how it improves entity recognition.

5. Open the *Hotel\_amenity* entity and turn on **Fuzzy Matching**.

A screenshot of the Entity view for "Hotel\_amenity". The URL bar shows "[@hotel\\_amenity](#)". The page includes fields for "Entity name" (set to "@hotel\_amenity") and "Description" (Name your entity to match the category of values that it will detect.). On the right, there is a "Last updated:" timestamp and a "Fuzzy matching" section with a green toggle switch labeled "On", which is highlighted with a red box.

6. To complete the training, enable the **Fuzzy Matching** for the entities [@pizza-toppings](#) and [@pizza\\_notoppings](#).

7. When Watson is done training, type *where are the bikes?*

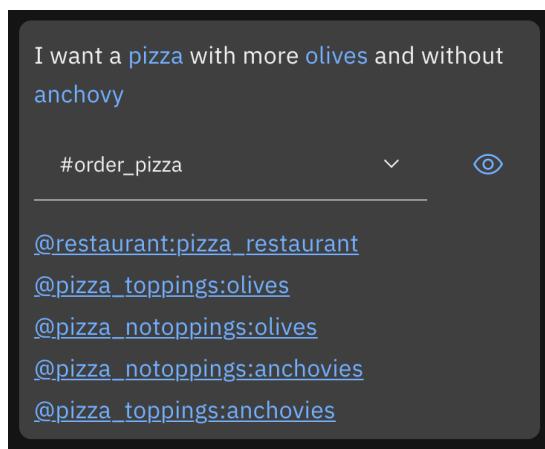


**Note:** notice that the correct `@entity_name:entity_value` (`@hotel_amenity:gym`) value pair is now recognized for *bikes*.

## 17. Contextual entities

When you define specific values for an entity, the service finds entity mentions only when a term in the user input exactly matches (or closely matches if fuzzy matching is enabled) a value or synonym defined. When you define a contextual entity, a model is trained on **both the entity value and the context** in which the entity is used in sentences that you annotate. This contextual entity model enables WA to calculate a confidence score that identifies how likely a word or phrase is to be an instance of an entity, based on how it is used in the user input

1. Open **Try it out** panel.
2. Enter *I want a pizza with more olives and without anchovy* at the bottom of the chat window.



The bot understands the `#order_pizza` intent and the entities `@pizza_toppings` and `@pizza_notoppings` but it can't determine what the user want and don't want. This is an example where Watson needs to know the context. We are going to **annotate** the user's examples to identify the context, i.e. define how the entities are used in this particular case.

3. Go to **Intents** tab and select `#Order_pizza`

4. Enable the option **Annotate entities**

← | #order\_pizza

---

Intent name  
Name your intent to match a customer's question or goal  
`#order_pizza`

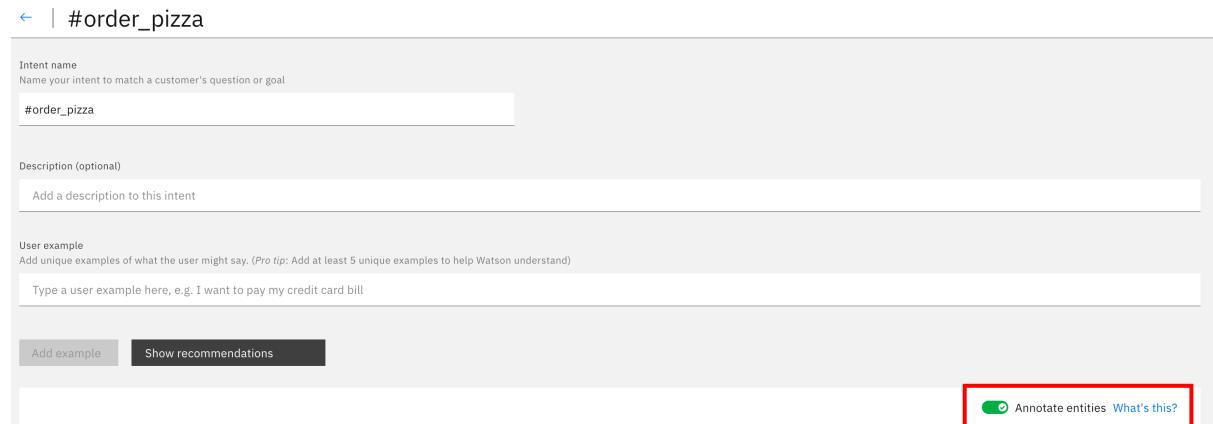
Description (optional)  
Add a description to this intent

User example  
Add unique examples of what the user might say. (Pro tip: Add at least 5 unique examples to help Watson understand)

Type a user example here, e.g. I want to pay my credit card bill

Add example Show recommendations

Annotate entities [What's this?](#)



5. In the 4<sup>th</sup> example, highlight the group *red peppers* then enter `@pizza_toppings`

User examples (17) ↑

`I'd like a large vegetarian without onions`

`Can I get a hawaiian pizza with no pineapple`

`Can I get a pizza margarita small please`

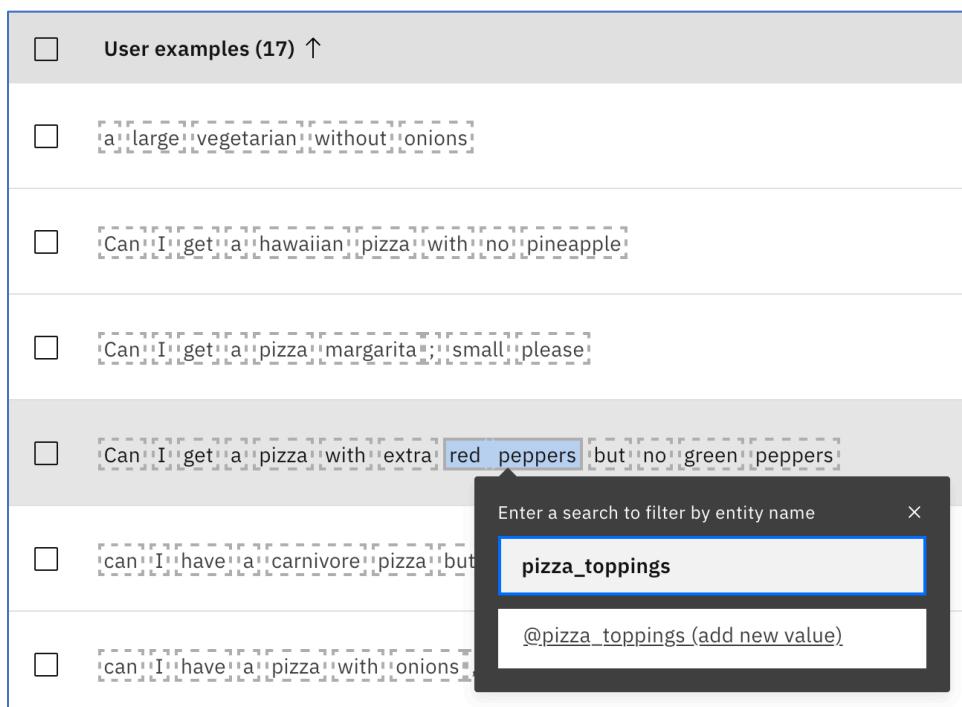
`Can I get a pizza with extra red peppers but no green peppers`

`I can have a carnivore pizza but`

`I can have a pizza with onions`

Enter a search to filter by entity name  X

`@pizza_toppings (add new value)`



(Save your time by starting with typing, then select the right entity from the list. The list will change as you proceed typing)

This way, we identified (i.e. annotated) the topping requested by the user – red peppers in this case

7. In a similar way, annotate these entities with `@pizza_toppings`.

<input type="checkbox"/> User examples (17) ↑
<input type="checkbox"/> [a]large[vegetarian]without[onions]
<input type="checkbox"/> [Can][I]get[a]hawaiian[pizza]with[no]pineapple]
<input type="checkbox"/> [Can][I]get[a]pizza[margarita][small]please]
<input type="checkbox"/> [Can][I]get[a]pizza[with]extra red[peppers]but[no]green[peppers]
<input type="checkbox"/> [can][I]have[a]carnivore[pizza]but[no]chicken]
<input type="checkbox"/> [can][I]have[a]pizza[with]onions,beef, chicken[and]extra cheese]
<input type="checkbox"/> [Can][I]order[a]pizza[with]more cheese]
<input type="checkbox"/> [Can][I]order[a]pizza[without]pineapple]
<input type="checkbox"/> [Can][you]deliver[a]vegetarian[pizza]
<input type="checkbox"/> [I][d]like[pepperoni]pizza]
<input type="checkbox"/> [I]don't[want]red[peppers]or[onions]on[my]vegetarian[pizza]
<input type="checkbox"/> [I]love anchovies[so]please[send]me[a]pizza[full]of[them]
<input type="checkbox"/> [I]want[to]order[a]large[pizza]
<input type="checkbox"/> [I]want[to]order[a]pizza]
<input type="checkbox"/> [I'd]like[a]large[hawaiian]with[extra]pineapple
<input type="checkbox"/> [I'd]like[a]small[margherita]but[please]no[anchovies]
<input type="checkbox"/> [medium]pizza[please]and[I]don't[want]any[red]peppers]

8. Now, we are going to identify the toppings that the user does **not** want. Annotate the intent using [@pizza\\_notoppings](#) entity like below:

<input type="checkbox"/> User examples (17) ↑
<input type="checkbox"/> [a][large][vegetarian] without onions
<input type="checkbox"/> [Can][I][get][a][hawaiian][pizza][with][no][pineapple]
<input type="checkbox"/> [Can][I][get][a][pizza][margarita];[small][please]
<input type="checkbox"/> [Can][I][get][a][pizza][with][extra][red][peppers][but][no][green][peppers]
<input type="checkbox"/> [can][I][have][a][carnivore][pizza][but][no][chicken]
<input type="checkbox"/> [can][I][have][a][pizza][with][onions],[beef],[chicken][and][extra][cheese]
<input type="checkbox"/> [Can][I][order][a][pizza][with][more][cheese]
<input type="checkbox"/> [Can][I][order][a][pizza][without][pineapple]
<input type="checkbox"/> [Can][you][deliver][a][vegetarian][pizza]
<input type="checkbox"/> [I][d][like][pepperoni][pizza]
<input type="checkbox"/> [I][don't][want][red][peppers][or][onions][on][my][vegetarian][pizza]
<input type="checkbox"/> [I][love][anchovies][so][please][send][me][a][pizza][full][of][them]
<input type="checkbox"/> [I][want][to][order][a][large][pizza]
<input type="checkbox"/> [I][want][to][order][a][pizza]
<input type="checkbox"/> [I'd][like][a][large][hawaiian][with][extra][pineapple]
<input type="checkbox"/> [I'd][like][a][small][margherita][but][please][no][anchovies]
<input type="checkbox"/> [medium][pizza][please][and][I][don't][want][any][cheese]

If we open the entity `@pizza_notoppings`, and the **Annotation** tab, we can see the annotated examples (9 examples)

The screenshot shows a user interface for managing annotations. At the top, there are two tabs: "Dictionary (11)" and "Annotation (9) Beta". The "Annotation (9) Beta" tab is highlighted with a red border. Below the tabs, there is a section titled "User Examples (9)". Each example is preceded by a small checkbox. The examples listed are:

- a large vegetarian without onions
- Can I get a hawaiian pizza with no pineapple
- Can I get a pizza with extra red peppers but no green peppers
- can I have a carnivore pizza but no chicken
- Can I order a pizza without pineapple
- I don't want red peppers or onions on my vegetarian pizza
- I don't want red peppers or onions on my vegetarian pizza
- I'd like a small margherita but please no anchovies
- medium pizza please and I don't want any cheese.

If we open the entity `@pizza_toppings`, and the **Annotation** tab, we can see the annotated examples (8 examples)

The screenshot shows a user interface for managing annotations. At the top, there are two tabs: "Dictionary (11)" and "Annotation (8) Beta". The "Annotation (8) Beta" tab is highlighted with a red border. Below the tabs, there is a section titled "User Examples (8)". Each example is preceded by a small checkbox. To the right of each example, there is a column labeled "Intent" which contains a single link: "#order\_pizza". The examples listed are:

User Examples (8)	Intent
<input type="checkbox"/> can I have a pizza with onions, beef, chicken and extra cheese	#order_pizza
<input type="checkbox"/> can I have a pizza with onions, beef, chicken and extra cheese	#order_pizza
<input type="checkbox"/> can I have a pizza with onions, beef, chicken and extra cheese	#order_pizza
<input type="checkbox"/> can I have a pizza with onions, beef, chicken and extra cheese	#order_pizza
<input type="checkbox"/> Can I order a pizza with more cheese	#order_pizza
<input type="checkbox"/> I'd like a large hawaiian with extra pineapple	#order_pizza
<input type="checkbox"/> Can I get a pizza with extra red peppers but no green peppers	#order_pizza
<input type="checkbox"/> I love anchovies so please send me a pizza full of them	#order_pizza

9. Open **Try it out** panel.
10. Enter *I want a pizza with more salamy and without anchovy.*

I want a pizza with more salamy and without anchovy.

#order\_pizza

@restaurant:pizza\_restaurant  
@pizza\_toppings:salamy  
@pizza\_notoppings:anchovies

Now, Watson recognizes what user does and does not want. So, pretty cool

## 18. Entity pattern

In this section we will explore entity patterns.

1. Go back to the **Entities** tab, Click **Create entity**.
2. Enter *@pattern* as entity name
3. Click **Create entity**
4. Enter
  - a. as value: *email address*
  - b. as type select **Patterns**
  - c. as pattern: *\b[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}\b*

The above string is a regular expression for email address.

5. Disable the Fuzzy matching as we want WA to return the exact text defined by the regular expression.

← | @pattern

Entity name  
Name your entity to match the category of values that it will detect.

@pattern

Fuzzy matching ⓘ  
Off

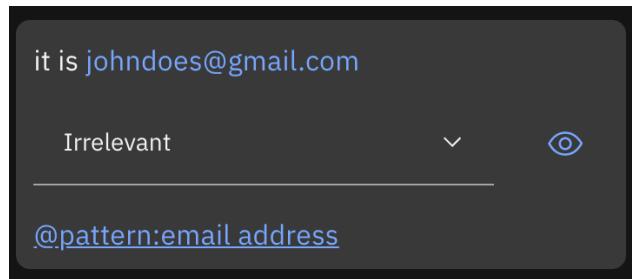
Value	Patterns	-	Type a pattern, e.g. (\d{3})
email address	Patterns	-	Type a pattern, e.g. (\d{3})

Add value

6. Click **Add value**, then click on top left arrow icon
- That's the way to extract an e-mail address from the user's input

## 19. Test your pattern

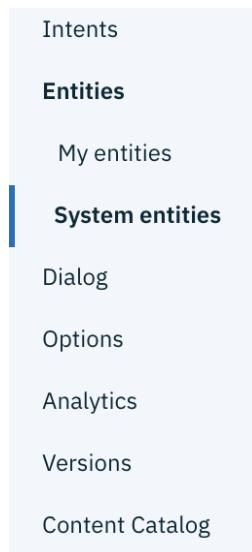
1. Open the **Try it out** panel, wait until Watson is done training.
2. Type in: *it is johndoes@gmail.com*



Watson recognizes *johndoes@gmail.com* as an entity email address.

## 20. Enable system entities

1. You will need to enable system entities. System entities can be used to help clarify a user's questions/phrases. If not already there, click the **Entities** and **System entities** tabs at the top of your skill.



You will get these system entities

- **@sys-number:** detects numbers that are written using either numerals or words. In either case, a numeric value is returned.
- **@sys-currency:** detects monetary currency values that are expressed in an utterance with a currency symbol or currency-specific terms. A numeric value is returned.
- **@sys-percentage:** detects percentages that are expressed in an utterance with the percent symbol or written out using the word percent. In either case, a numeric value is returned.

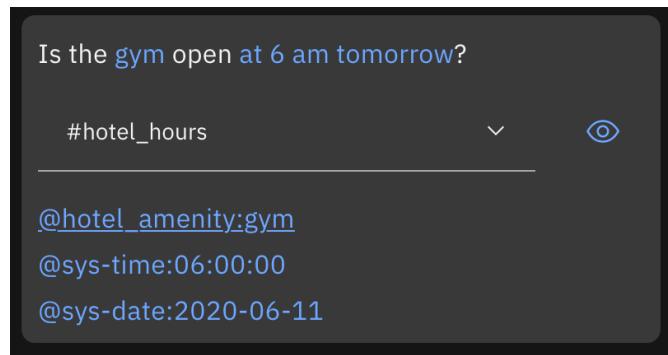
- **@sys-date, @sys-time and \$timezone:** Mentions of a date range such as the weekend, next week, or from Monday to Friday are extracted as a pair of @sys-date entity mentions that show the start and end of the range

## 2. Switch on all system entities.

Entities	The following entities are prebuilt by IBM to recognize references to things like numbers and dates in user input. Turn on a system entity to start using it. You cannot edit system entities. <a href="#">Learn more</a>		
	Name (5)	Description	Status
My entities	▼ <a href="#">@sys-time</a>	Extracts time mentions (at 10)	On
System entities	▼ <a href="#">@sys-date</a>	Extracts date mentions (Friday)	On
Dialog	▼ <a href="#">@sys-currency</a>	Extracts currency values from user examples including the amount and the unit. (20 cents)	On
Options	▼ <a href="#">@sys-percentage</a>	Extracts amounts from user examples including the number and the % sign. (15%)	On
Analytics	▼ <a href="#">@sys-number</a>	Extracts numbers mentioned from user examples as digits or written as numbers. (21)	On
Versions			
Content Catalog			

## 21. Test System Entities

1. Open the **try it out** panel, wait until Watson is done training.
2. Type in *Is the gym open at 6 am tomorrow?*



**Note:** notice that the time and date are recognized

**6 am** has been captured as **06:00:00**, the standard time format

**tomorrow** has been captured as **2020-06-11** (yyyy-mm-dd format)

the current date is June the 10<sup>th</sup>, 2020.

3. Type in *Did the restaurant charge a 15% tip for a 20\$ bill today?*

The screenshot shows a dialog node configuration. The input text is "Did the restaurant charge a 15% tip for a 20\$ bill today?". Below the input, there is a dropdown menu set to "Irrelevant" and an "Eye" icon. The system entities listed are:  
@places:restaurant  
@hotel\_amenity:hotel restaurant  
@sys-percentage:15  
@sys-currency:20  
@sys-date:2020-06-10

**Note:** The percentage, currency, number and date are recognized. You can use these system entities in your dialog node calculations the same way you would use your custom defined entities.

4. Type in *are you open the two next weeks?*

The screenshot shows a dialog node configuration. The input text is "are you open in the two next weeks?". Below the input, there is a dropdown menu set to "#hotel\_hours" and an "Eye" icon. The system entities listed are:  
@sys-number:2  
@sys-date:2021-01-17  
@sys-date:2021-01-23

**Note:** Assuming today is Friday the 15<sup>th</sup> Jan, WA returns a date range ie, the first date of the period (the next Sunday), the 17<sup>th</sup> Jan and the last day of the period (the Saturday 2 weeks after), the 23<sup>rd</sup> Jan.

One option to dissociate the 2 dates is :

From Date: <? entities['sys-date'].filter("d", "d.role.type == 'date\_from'")[0]?:value ?>  
To Date: <? entities['sys-date'].filter("d", "d.role.type == 'date\_to'")[0]?:value ?>

# Conversation Dialog

## 22. Create your first Dialog

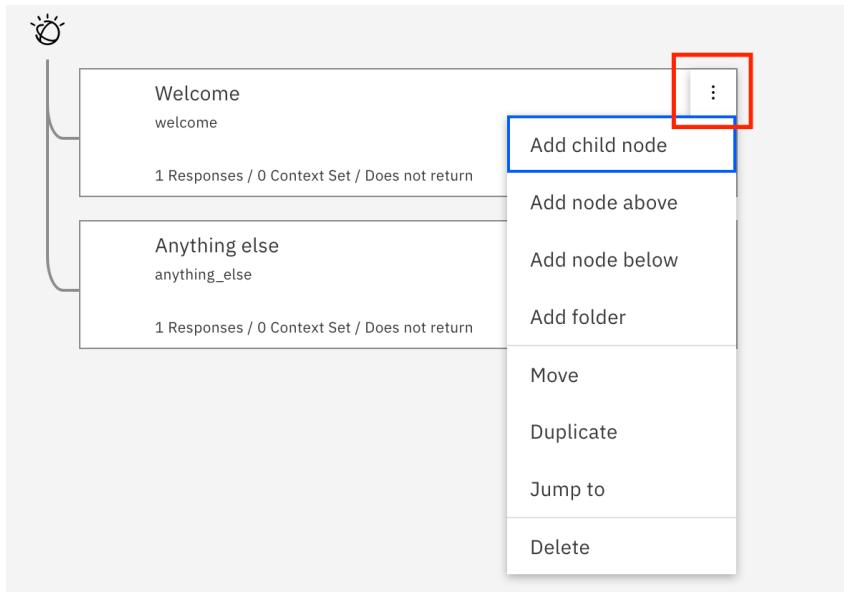
To use the new conversation, you will need to train it with the intents, entities, and/or dialog nodes relevant to your application's use case. A dialog uses the intent and entity that have been identified, plus context from the application, to interact with the user and provide a response.

A dialog chat flow is made up of nodes. **Nodes** define the steps in the conversation or chat flow. Dialog nodes are chained together in a tree structure, and each node can be defined by two parts: a condition and a response

A **condition** is the portion of the dialog node that determines whether the node is used in the conversation. Conditions can be defined using intents, entities, context variables, and special conditions. The Spring Expression (SpEL) language is used to write valid expressions for conditions.

A **response** is activated if the node's condition matches the user input. A response is returned to the end-user and can be either simple text or a more advanced process (such as walking the user through the process of changing their password).

Each node has a node menu which looks like this:



This guide will go through most of the actions that can be completed from the node menu but be sure to note that **delete** will remove the node and all of its child nodes.

## 1. Click on **Dialog** tab

The screenshot shows the Watson Assistant interface with the 'My Concierge skill' version set to 'Development'. The 'Dialog' tab is selected. On the left, there's a sidebar with options: Intents, Entities, Dialog (selected), Options, Analytics, Versions, and Content Catalog. At the top right are three buttons: 'Add node' (blue), 'Add child node' (grey), and 'Add folder' (dark grey). Below these buttons is a sun icon. The main area shows two dialog nodes in a tree structure. The first node is 'Welcome' (welcome), which is triggered by the intent 'welcome' and has 1 response. The second node is 'Anything else' (anything\_else), which is triggered by the intent 'anything\_else' and also has 1 response.

You can notice that Watson has already created two dialog nodes for you: the **“Welcome”** and the **“Anything else”** nodes.

### Note:

- The **Welcome** node is automatically added to your dialog chat flow. Welcome is a special condition in conversation that only evaluated as true during the first dialog turn when the application does not contain any user input. If the initial request from the application contains user input, Welcome node will not be triggered. Typically, welcome is used when you always want to display a message or a greeting at the beginning of the conversation.
- The **Anything else** node is also automatically added to your dialog chat flow. The Anything else node is set to always evaluate to true. Its purpose is to always provide a response to the user if the user's input does not evaluate to true for any other parent node in the tree.

- The panel that appears on the **right side of the screen** is the editor panel for the node. When you create a new node or when you click on a node to make edits, this panel will appear for the node that you click on. This is where all editing is done for the node. The image below explains the elements of the node editor.

The screenshot shows the Watson Node Editor interface with several configuration sections:

- Node Name:** Name your node in this field. (Input field: Enter node name)
- Condition:** Enter the main condition in this field. (Input field: Enter condition)
- Advanced editor menu:** Delete a response or open the Json editor for the response field. (Text area: Response variations are set to sequential. Set to random | multiline)
- Response:** Enter a response for the condition in this field. (Input field: Enter response text)
- Next action menu:** Drop down menu to choose which action Watson will take next after it returns a response to the user. (Options: Wait for user input, etc.)
- Disambiguation :** text display to the user. (Input field: Disambiguation : text display to the user.)

Optional note: Status: this node will not be used for features that require displaying an external node name unless an external node name is provided.

2. Click on the **Welcome** node to edit it and fill it as below

Field	Value
Node name	<i>Welcome</i>
If assistant recognizes	<i>welcome</i>
Assistant responds	<i>Hi! I am Watson, nice to meet you</i>

The screenshot shows the Watson Assistant interface with the 'Welcome' node selected. The top bar has a 'Customize' button with a gear icon and an 'X' button, both of which are highlighted with red boxes. The main area shows the node configuration:

- Node name:** Welcome
- If assistant recognizes:** welcome
- Assistant responds:** Hi! I am Watson, nice to meet you (highlighted with a red box)

Below the responses, there is a note: "Response variations are set to **sequential**. Set to [random](#) | [multiline](#)". A "Learn more" link is also present.

3. Close the **Welcome** node by clicking on **x** on top right.

4. Click on the **Anything else** node, you will see 3 pre-created responses.

The screenshot shows the configuration interface for the 'Anything else' node. At the top, there's a title bar with the node name 'Anything else' and a 'Customize' button with a gear icon. Below the title bar, a note says 'Node name will not be shown to customers for disambiguation.' followed by a 'Settings' link. The main area is divided into two sections: 'If assistant recognizes' and 'Assistant responds'. Under 'If assistant recognizes', there's a list with one item: 'anything\_else' followed by a minus sign and a plus sign. Under 'Assistant responds', there's a section titled 'Text' with a dropdown arrow. Below it are four response variations: 'I didn't understand. You can try rephrasing.', 'Can you reword your statement? I'm not understanding.', 'I didn't get your meaning.', and 'Enter response variation'. Each response has a minus sign to its right. At the bottom of the interface, a note states 'Response variations are set to **sequential**. Set to [random](#) | [multiline](#)' and a 'Learn more' link.

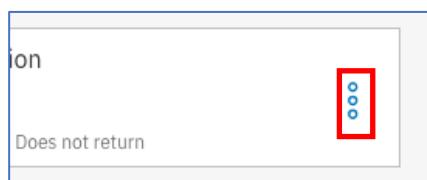
You may edit these if you wish.

5. Close the Anything else node by clicking on x on top right.

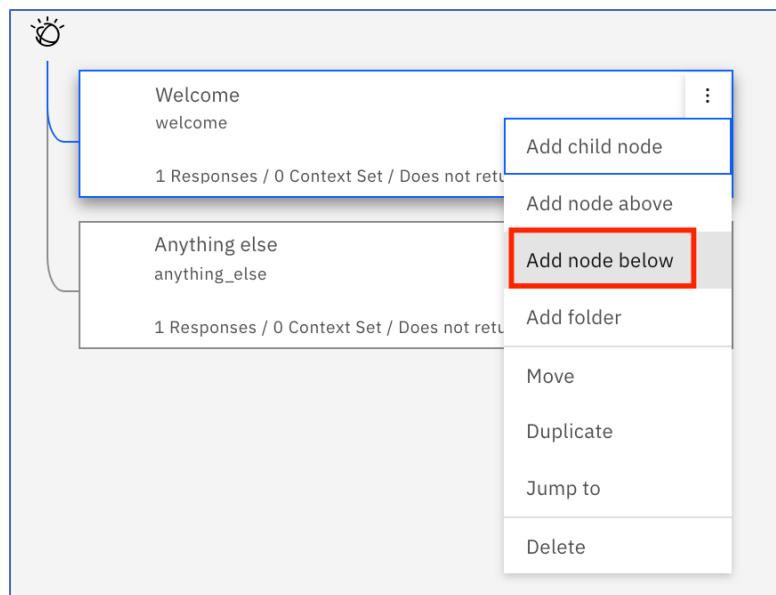
## 23. Create your first nodes

You can create a dialog branch for each of the *intents* you identified as well as the start & end of the conversation. You should also greet the end user as appropriate. For the greeting, you can create a new dialog node to respond to a *greeting* intent. To do this you are going to leverage the intent `#General_Greetings` that you have imported in earlier steps.

**Note:** When you click on **Add node** button at the top of the tab, a node will be added below the selected node. Which means you must know which node is the selected node. Thus, the best way to add a node into correct position in the dialog flow is to use a **node menu** – the 3 vertical dots on the right hand side of a node.



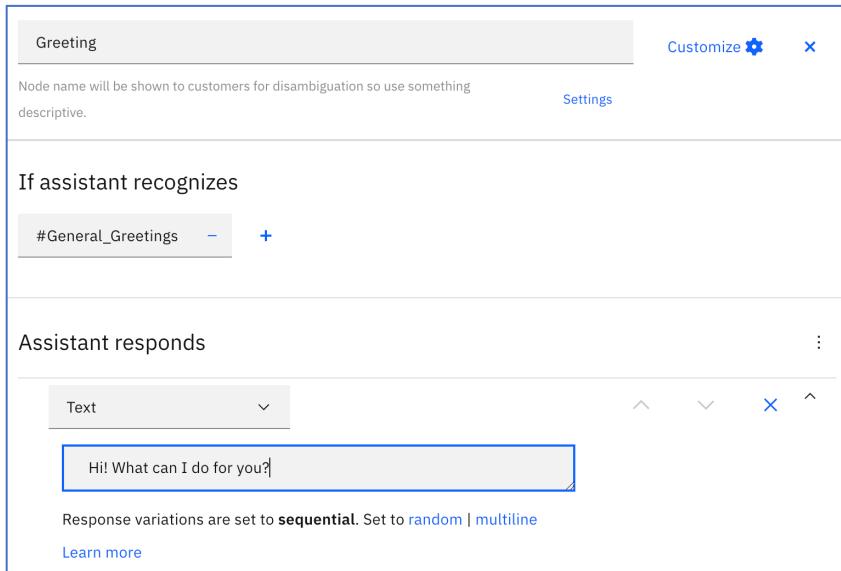
1. On the **start of the conversation** node, the **node menu** (3 dots) then click **Add node below**.



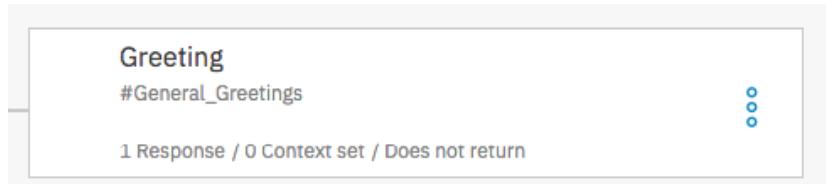
2. In this new node, enter the following values.

Field	Value
Node name	<i>Greeting</i>
If assistant recognizes	#General_Greetings
Assistant responds	<i>Hi! What can I do for you?</i>

By setting the condition (if assistant recognizes) to an *intent*, you are indicating that this node will be triggered by any input that matches the specified *intent*.



3. Close the node, by clicking **x** in the top right corner. Your new node should look like this:



**Note:** The figure above indicates that there is only one condition that is evaluated (the #General\_Greeting intent), one response and no context variable used.

You will create a dialog branch to respond to the #hotel\_hours intent. Because there are multiple possibilities to manage this intent, this branch will be more complex. For the beginning, we will keep it simple

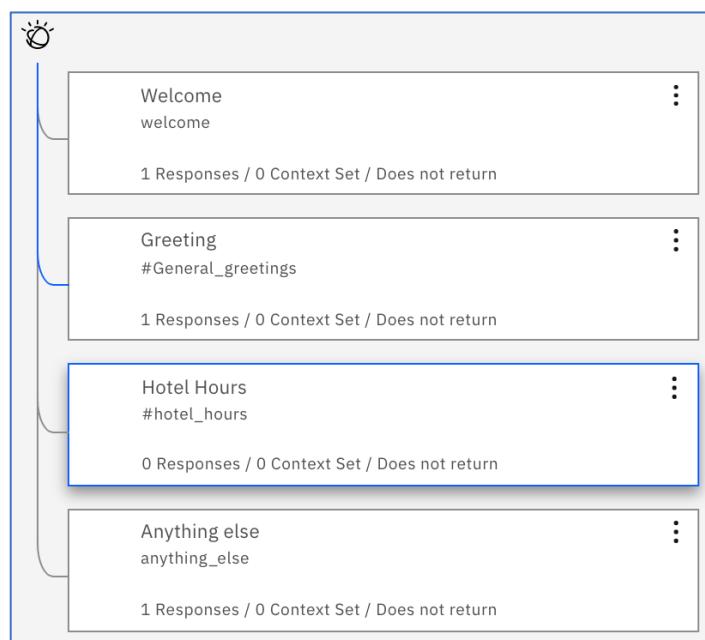
4. Add a node below the **Greeting** node as below

Field	Value
Node name	<i>Hotel Hours</i>
If assistant recognizes	<i>#hotel_hours</i>
Assistant responds	<i>The @hotel_amenity is open from 6 am to 9 pm</i>

*@hotel\_amenity* is the entity captured by Watson. The bot will return this information and demonstrate that it understood the request.

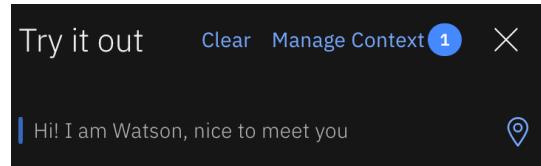
## 24. Test your conversation

Now, you should have a dialog with four nodes.



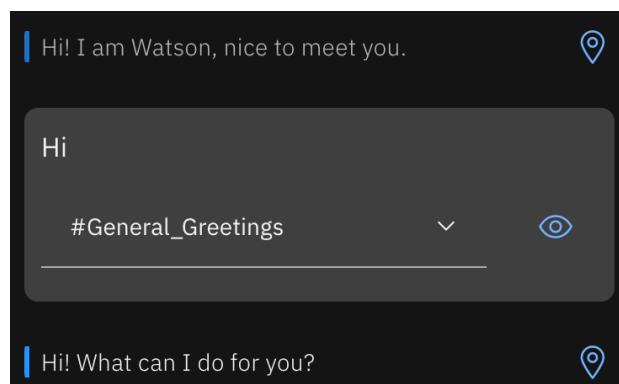
it's time to test it.

11. Open **Try it out** panel and click **Clear** on the top of the panel.  
A test user interface will immediately launch and, based on the root node you just created, provide a greeting to the end user. (You may see a message that Watson is being trained.)

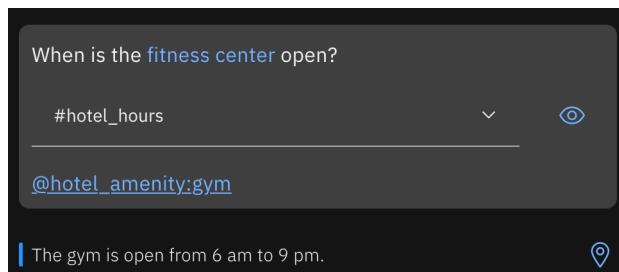


12. Enter *Hi* at the bottom of the chat window.

The bot understands the #General\_greetings intent and send you back the right response.

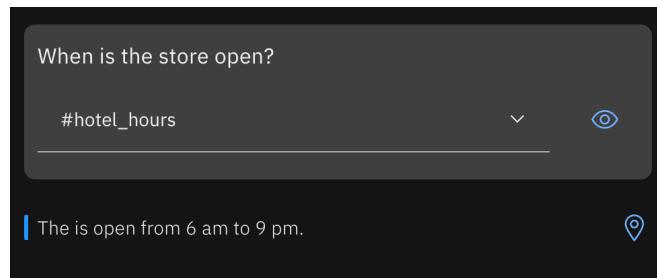


13. Enter *When is the fitness center open?*



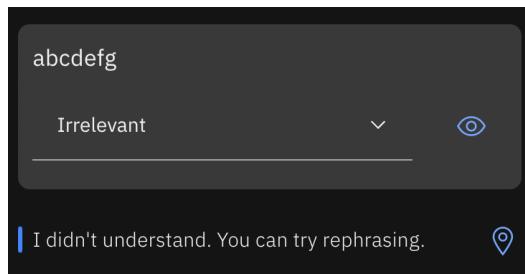
**Note:** Watson should return the response that is matched to `#hotel_hours` and `@hotel_amenity:gym` as *fitness center* is a synonyme of this entity.

14. Enter *When is the store open?*



**Note:** Watson understood the intent `#hotel_hours`. But as `store` is not defined as value of any entities, the system returns the incomplete answer without the `@hotel_amenity` value. During the next lab, we are going to understand how to improve the WA behaviour to manage such behaviour.

15. Enter *abcdefg*



Watson cannot identify anything so it executes the `anything_else` node.

# Enrich your data by using context variables

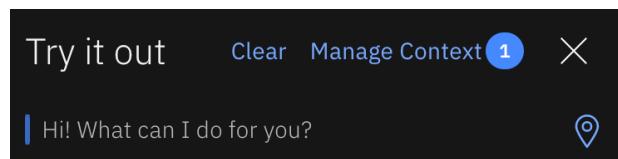
---

Watson Assistant allows you to store information. This is done using the **context**. For example, when asked where something is, you may wish to show a map. In such case, your dialog would store the user's location in the context, and your application can use it later to show that location on a map. You can set variables in the context using the **Json editor** or the **context editor**.

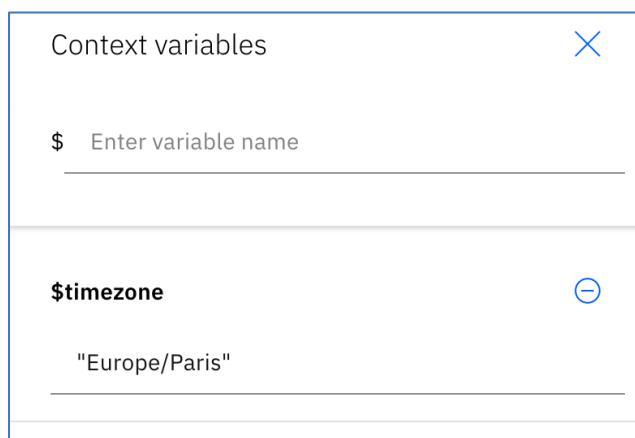
There are two methods to access a context variable:

- Shorthand: \$variable\_name
- Full Syntax: context.variable\_name

1. To view the context variables that are currently set in a conversation, open the **Try it out** panel, and click on **Manage Context**. The number besides Manage Context tells you how many context variables are currently set.



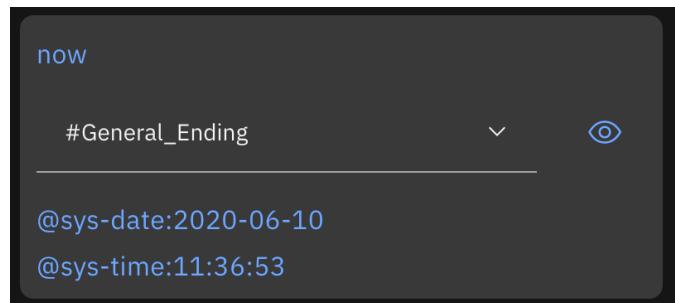
This will open the **Context variables** panel.



Since you enabled sys\_date and sys\_time system entities, Watson stores the user's time zone (which comes from the browser) into the \$timezone context variable.

2. Close **Context variables** by pressing the X sign on top right.

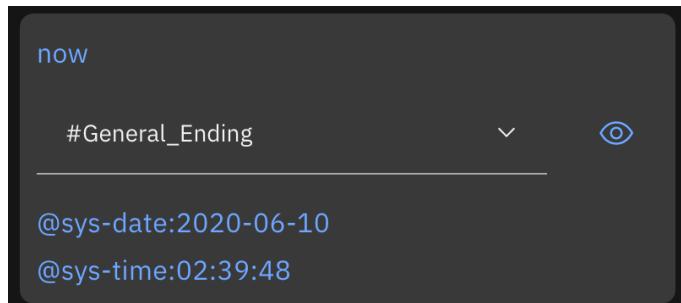
3. In the **try it out** panel, enter *now*



4. You will see the date and time in your time zone. Go back to **Manage Context** panel and change the *\$timezone* variable to something else.
5. Write down (or copy into clipboard) your current time zone, so you can use it later. Replace the time zone with *America/Los\_Angeles*

**\$timezone** ⊖  
"America/Los\_Angeles"

6. Close the Context variables panel. In the **Try it out** panel, enter again *now*



You will see that WA applied the new time zone.

7. To avoid any confusion in the time zone, set back the time zone as it was originally.

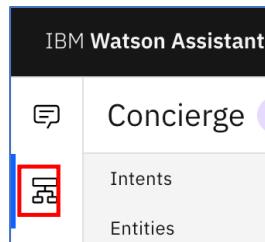
# Save your skill to your computer

(optional)

---

For backup purposes, you might want to save your Skill on your local machine.

1. On top left, click on the **Skills** icon.

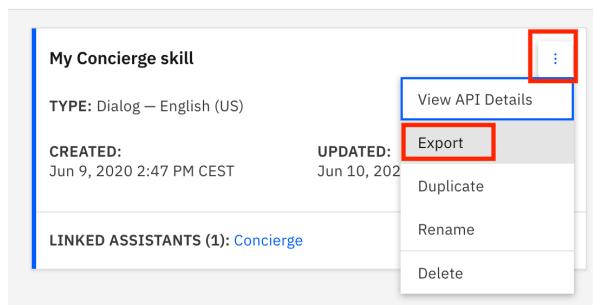


2. Click on the Skill's menu (3 dots), then select Export.

Skills

Skills contain the training to respond to your customer queries. Add skills to your skill stack.

Create skill

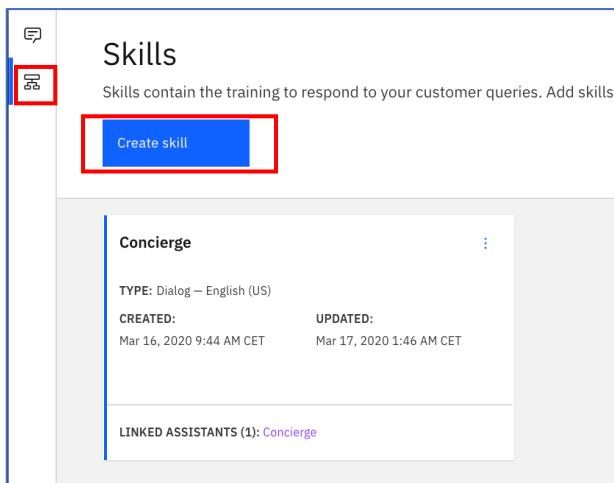


3. In the **Save As** field, keep the proposed name (skill-Concierge) and click **Save**. With this you saved your skill into a JSON file on your computer. At any time, you can retrieve this Skill back to WA by importing this JSON file.

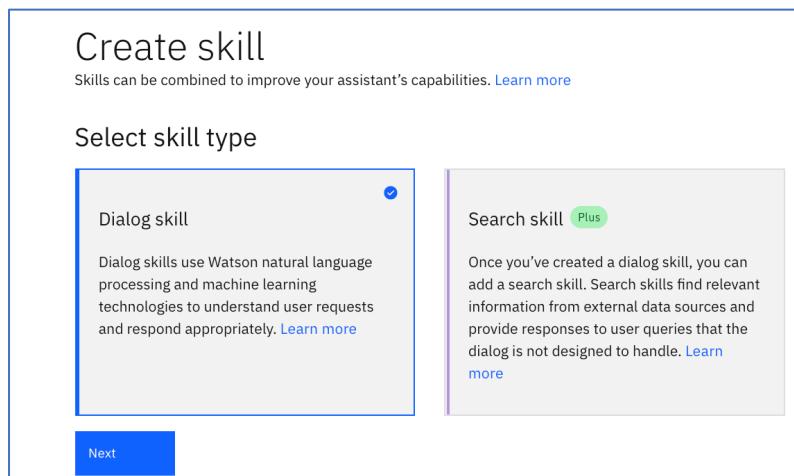
**Note:** if you want to compare your skill with the one we created, feel free to import into your Assistant our skill as it was after the end of this exercise. This skill is in file [skill-My-Concierge-Skill-Lab1.json](#).

You can import a skill into your Assistant this way:

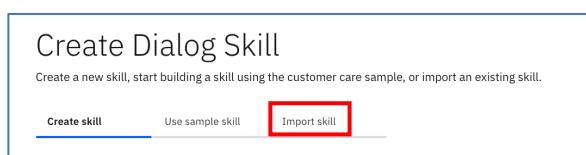
1. On Skills tab, click Create skill.



2. Select **Dialog skill** and click **Next**.



3. Click on **Import skill**.



4. Click **Choose JSON File**, then select the file with the exported skill.

## Create Dialog Skill

Create a new skill, start building a skill using the customer care sample, or import an existing skill.

Create skill

Use sample skill

**Import skill**

Select the JSON file for the dialog skill with the data you want to import.

**Choose JSON File**

skill-My-Concierge-Skill-Lab1.json ×

**Import**

5. Click **Import**.

6. Now, you can see the skill you just imported on your **Skills** page.

The screenshot shows the 'Skills' page. On the left, there's a sidebar with a message icon and a 'Skills' icon, which is highlighted with a red box. Below the sidebar, there's a 'Create skill' button. The main area displays two skill cards:

- Concierge**:
  - TYPE: Dialog – English (US)
  - CREATED: Mar 16, 2020 9:44 AM CET
  - UPDATED: Mar 17, 2020 1:46 AM CET
  - LINKED ASSISTANTS (1): Concierge
- My Concierge Skill**:
  - TYPE: Dialog – English (US)
  - CREATED: Mar 17, 2020 2:00 AM CET
  - UPDATED: Mar 17, 2020 2:00 AM CET

A red box highlights the 'My Concierge Skill' card.