**Alexa - Build an engaging Alexa app**

**Introduction - How a skill works**

The following is a simple workflow that demonstrates how Alexa works. In this example, the user invokes a simple Alexa skill called Hello World.

* To launch the skill, the user says, "Alexa, open Hello World."
* The Alexa-enabled device sends the utterance to the Alexa service in cloud. There, the utterance is processed via automatic speech recognition, for conversion to text, and natural language understanding to recognize the intent of the text.
* Alexa sends a JavaScript Object Notation (JSON) request to handle the intent to an AWS Lambda function in the cloud. The Lambda function acts as the backend and executes code to handle the intent. In this case, the Lambda function returns, "Welcome to the Hello World skill."

**Designing an Engaging Voice UI**

* Wake word: The wake word tells Alexa to start listening to your commands.
* Launch word: A launch word is a transitional action word that signals Alexa that a skill invocation will likely follow. Sample launch words include tell, ask, open, launch, and use.
* Invocation name: To begin interacting with a skill, a user says the skill's invocation name. For example, to use the Daily Horoscope skill, the user could say, "Alexa, read my daily horoscope."
* Utterance: Simply put, an utterance is a user's spoken request. These spoken requests can invoke a skill, provide inputs for a skill, confirm an action for Alexa, and so on. Consider the many ways a user could form their request.
* Prompt: A string of text that should be spoken to the customer to ask for information. You include the prompt text in your response to a customer's request.
* Intent: An intent represents an action that fulfills a user's spoken request. Intents can optionally have arguments called slots.
* Slot value: Slots are input values provided in a user's spoken request. These values help Alexa figure out the user's intent..
* Slots can be defined with different types. The travel date slot in the above example uses Amazon's built-in AMAZON.DATE type to convert words that indicate dates (such as "today" and "next Friday") into a date format, while both from City and to City use the built-in AMAZON.US\_CITY slot.

**Interaction Model**

To create an interaction model, define the requests (intents) and the words (sample utterances). Your Lambda skill code then determines how your skill handles each intent. You can start defining the intents and utterances on paper and iterate on those to try to cover as many possible ways the user can interact with the skill.

Then, go to the [Alexa developer console](https://developer.amazon.com/alexa/console/ask) and start creating the intents, utterances, and slots. The console creates JSON code of your interaction model. You can also create the interaction model in JSON yourself using any JSON tool and then copy and paste it in the developer console.

Before you write one line of code, you should work really hard to think through how your customers will interact with your skill. Skipping this step will result in a poorly written skill that will not work well with your users.

**Situational Design**

Situational Design is a voice-first method to design a voice user interface. You start with a simple dialog which helps keep the focus on the conversation. Each interaction between your customer and the skill represents a turn. Each turn has a situation that represents the context. If it's the customer's first time interacting with the skill, there is a set of data that is yet unknown. Once the skill has stored the information, it will be able to use it the next time the user interacts with the skill.

With situational Design, you start with the conversation and work backwards to your solution.

**Characteristics of a Well-Designed Voice User Interface**

* Uses natural forms of communication
  + When talking with a machine, a user should not be required to learn a new language or remember the rules. A machine should conform to the user's paradigm, not the other way around.
* Navigates through information easily
  + Your skill’s VUI should offer an easy way to cut through layers of information hierarchy by using voice commands to find important information.
* Creates an eyes- and hands-free experience
  + Voice interfaces should allow a user to perform tasks while their eyes and hands are occupied.
* Creates a shared experience
  + Voice experiences let users collaborate, contribute, or play together through natural conversation

There are inherent challenges with voice interfaces, including context switching or ambiguity in the conversation, discovering intent, and being unaware of the user's current state or mood. For a good user experience, you should plan for these challenges when developing your skill.

**Five Best Practices for Voice Design**

#### Stay close to Alexa's persona

#### Alexa's personality is friendly, upbeat, and helpful. She's honest about anything blocking her way but also fun, personable, and able to make small talk without being obtrusive or inappropriate.

#### Try to keep the tone of your skill’s VUI as close to Alexa’s persona as possible. One way to do this is by keeping the VUI natural and conversational.

#### Slightly vary the responses given by Alexa for responses like "thank you" and "sorry". Engaging the user with questions is also a good technique for a well-designed VUI.

#### Write for the ear, not the eye

#### The way we speak is far less formal than the way we write. Therefore, it's important to write Alexa’s prompts to the user in a conversational tone.

#### No matter how good a prompt sounds when you say it, it may sound odd in text-to-speech (TTS).

#### It is important to listen to the prompts on your test device and then iterate on the prompts based on how they sound.

#### Be contextually relevant

#### List options in order from most to least contextually relevant to make it easier for the user to understand. Avoid giving the user options in an order that changes the subject of the conversation, then returns to it again. This helps the user understand and verbalize their choices better without spending mental time and energy figuring out what's most relevant to them. The following is an example.

#### Be brief

#### Reduce the number of steps to complete a task wherever possible to keep the conversation brief. Simplify messages to their essence wherever possible. The following is an example.

#### Write for engagement to increase retention

#### Alexa skills should be built to last and grow with the user over time. Your skill should provide a delightful user experience, whether it's the first time a user invokes the skill or the 100th.

#### Design the skill to phase out information that experienced users will learn over time. Give fresh dialog to repeat users so the skill doesn't become tiresome or repetitive.

**Collecting Slots Turn-by-Turn With Auto-Delegation**

The user may not have responded for a variety of reasons. The skill should pose the initial question again but do so in a natural way. The reprompt should provide more context to help the user provide an answer. Specify the reprompt text by creating a new variable named reprompt\_text.

Notice the reprompt gives an example of what Alexa expects the user to say by having the skill provide Alexa's birthday in the format it is looking for. Providing examples like this is a best practice.

There is a potential complication with asking the user for their birthday. They might respond in many different ways. For example, the user might give only the month and day, or they might say something like, "Next Tuesday."

**Intents & Slots to Capture Information**

Now make some adjustments to the skill's front end. Specifically, you need to create an intent that will interpret how the user responds to Alexa's question. When you name an intent, think about what the intent is going to do. In this case, the intent is going to capture the user's birthday, so name it CaptureBirthdayIntent. Notice the words are not separated by spaces, and each new word begins with an uppercase letter.

Remember, an intent is an action to fulfill a user's request. An utterance is what invokes the intent. In response to the birthday question, a user might say - "I was born on November seventh, nineteen eighty three." You will add this utterance to the CaptureBirthdayIntent by typing it in exactly the way the user is expected to say it

From this utterance, there are three key pieces of information to collect: month, day, and year. These are called slots. You need to let Alexa know which words are slots and what kind of slots they are. Start with the month slot. In the utterance, you will replace the word representing the month (November) with the word month in curly brackets ({ }). This creates a slot called month.

There are two ways to create a slot. The first way is to select the word in the sample utterance where the slot should go and type the name of the slot in curly brackets (for example, {month}). The second way is to select the word in the sample utterance and use the Select an Existing Slot dialog box when it appears. In the dialog box, click the field under Create a new slot, type the name of the slot without curly brackets (for example, month), and click Add.

**Slot Types**

There are two types of slot types: custom and built-in. Wherever possible, use built-in slots. Alexa manages the definitions of built-in slots. These slots begin with AMAZON followed by what they define (for example, AMAZON.Month).

If an applicable built-in slot does not exist, create a custom slot and define the values it represents.

**Dialog Management**

Slots can be required or optional. That is, if you need a given value from the user, you can designate a slot as required using dialog management. Marking a slot as required triggers Alexa to actively work to fill it. Start by making each of the slots required.

One of the great things about dialog management is that the skill doesn't break or get confused if the user leaves out a piece of information or provides it out of the expected order. Instead, Alexa takes on the responsibility of collecting information designated as required to ensure a useful experience.

**Define a New Header**

Remember modifying the LaunchRequestHandler? This time, you are going to build a new handler. This handler will acknowledge that the user provided their birthday and repeat the birthday back to the user.

If you look at the code, you will notice the HelloWorldIntentHandler. But you deleted the HelloWorldIntent, right? Not entirely. The intent is gone from the front end, but the backend handler is still there. You need a new handler, so make things easier and reuse this handler for a new one called CaptureBirthdayIntentHandler.

Now you need to update the logic within the handler so the skill will confirm to the user that their birthday was heard correctly. In this case, you will have Alexa read the birthday back to the user, like this: “Thanks, I'll remember that you were born on {month} {day} {year}.”

Start by creating three variables in the handler to save the slots the skill is collecting.

Next, update the speak\_output. To do this, we use string interpolation to substitute values of our variables into placeholders in our string. While Python supports multiple ways to do this, we will use .format() function to replace placeholder values inside braces {} with the values of our variables.

Remember changing the HelloWorldIntentHandler to CaptureBirthdayIntentHandler? In every skill that uses the SDK, there is a place to notify the SDK of the available handlers. This is called registering. Update the code to register the new handler.

**Adding Memory to Your Skill-Module**

First, you adjusted the Cake Time skill to use the AttributesManager to save and read user information to Amazon S3. Then, you added code to the CaptureBirthdayIntentHandler to save the user’s birthday. Lastly, you created a new handler (HasBirthdayLaunchRequestHandler) so Alexa doesn’t repeatedly ask the same user for their birthday.

**Publish Your Skills**

View skill preview on distribution tab

After saving your information on this page, you can preview the skill card in the Alexa app. The app shows both published skills and your own skills currently in development, so you can see how your card will appear to users.

**Privacy & Compliance**

In this area, you provide information about whether your skill allows users to make purchases or spend money, collects personal information, is directed to or targets children under the age of 13, or contains advertising. You must also certify that your skill meets export compliance requirements.

Finally, provide instructions for the Alexa team to test your skill. Be sure to include any account or hardware requirements. Note that this information is not shown to users.

**Availability**

In this area, you choose who should have access to the skill and where it should be available.

* Public
  + Publish the skill to the general public. Anyone with an Alexa device can enable and use your skill, provided it is available in their region and language.
* Alexa for business organizations
  + Publish the skill as a private skill available to select businesses.
* Beta test
  + Make the skill available for up to 500 beta users (per skill) that you personally select. Invite friends or family, your social network contacts, or any person for whom you have an email address to test your skill and provide feedback.

At any time, you can add testers, remove testers, or end a test. You can see tester feedback in the Amazon Developer Portal. You can get feedback on your skill throughout the testing cycle and reduce costs by making fixes before you release the skill to production.

**Validate & Test Your Skill**

Validation does the following:

* Verifies that all required skill store fields are complete
* Performs a set of pre-certification tests on the skill. These tests send requests to your skill's endpoint.

The tests are run for all locales that your skill supports. Issues are listed by locale.

These tests provide immediate feedback for common certification failures. You can start the tests and then navigate away from the page. Return to the page to check the status and test results.

To run these tests, open your skill in the Alexa developer console, and click the Certification tab. The Validation page opens and displays any issues found.

After fixing one or more issues, return to the Certification tab. On the Validation page, click Run to re-run the tests.

For a smoother certification process, fix any issues reported by the tests. When there are no remaining issues, the page displays "Zero errors found."