Alexa Skill Builder's Guide

10 Things Every Developer Should Do to Build for Voice

Tips to Build High-Quality Alexa Skills

We get a lot of great questions from developers and skill builders about how to build high-quality voice experiences.

Building great voice experiences is both an art and a science. There are technical aspects to building for voice, as well as creative concepts that go into designing natural voice experiences. Both are important things to consider as you build engaging Alexa skills.

With more than 40,000 skills in the Alexa Skills Store, we've learned a lot about what makes a skill great, and what you can do to create incredible voice experience for your customers.

Here are 10 things to keep in mind when you're building for voice:

1. Do One Thing Really Well
2. Use a Memorable Invocation Name and Utterances
3. Focus on Intents Rather Than Commands
4. Simplify Choices
5. Pass the One-Breath Test
6. Include a Variety of Responses
7. Handle the Unexpected Gracefully
8. Use Analytics to Make Enhancements
9. Provide Contextual Help
10. Do Beta Testing

In this guide, we'll explore each tip and share how you can incorporate these suggestions into your Alexa skill. We'll also point you to additional resources and code samples to help you get started.

.

1

Do One Thing Really Well

As you begin to design your skill, you'll probably think of all the great things your customer could do with it. Skills **can** be very adept at handling a variety of requests. But with a voice interface, it can be difficult cult for users to remember all the different options they have to engage with your skill.

Since Alexa skills are enabled and built voice-first, it's important to make sure that a user can easily understand what your skill is capable of doing, and that the skill does what the user expects.

This same practice can be seen across the mobile application landscape. There are many successful apps that focus on doing one thing really well, and their developers build separate apps to handle those other edge cases.

An example of a skill that does one thing really well is a skill called Games Back. The sole purpose of this skill is to tell you how many games back from first place your favorite baseball team currently is. It doesn't have live game scoring, and it doesn't even tell you if your team won their last game. It only tells you the current standings for your selected team.

Carving out a specific purpose for your skill will also help you reach passionate users that care about the problem you're solving for them. When you start designing your skill and are nailing down its purpose, remember that your skill doesn't need to be everything to everyone. Your skill should serve a specific purpose, and serve that purpose better than any other skill.

Learn more: Voice Design Guide: How to Think Through the Design of a Voice Experience

2

Use a Memorable Invocation Name and Utterances

It can be easy to come up with an invocation name for your skill, but it's more important to come up with an invocation name that is memorable. A great example of this is "Alexa, open the Magic Door."

The invocation name, paired with the starting phrase "open" paints a rich image in your head. When you open this skill, you're even presented with the sound of a creaking door. This kind of invocation name makes it much easier for customers to remember the name of the skill and return to it in the future.

You're not just limited to the word "open," however. The list of starting phrases a user can say is pretty extensive, including words like "ask," "begin," launch," "load," "play," and "tell."

A user can say any of the starting words supported by Alexa to start using your skill, but by taking the time to consider and present a memorable example phrase, you can design one that sticks in their head.

Learn more: Documentation: Understanding How Users Invoke Custom Skills

3

Focus on Intents Rather Than Commands

When you build intents and sample utterances, think about the ways a customer would ask for that intent. For example, with the Dev Tips skill, the core functionality is to answer common questions that developers might ask. But it is **very** command focused, meaning the user has a specific request in mind. Sample requests include:

> *"Alexa, ask Dev Tips about {topic}."*
> *"Tell me about {topic}."*
> *"How do I use {topic}?"*

In each of those cases, the user is given an answer to a specific request. This is perfect for developers who know exactly what they want to learn, but what about someone who is new to skill building? What if they want something they haven't heard before?

When building this skill, we added a second intent, GetRandomTipIntent, for this specific purpose. If you ask Alexa for a "random tip," the skill will pick something from a list of answers and teach the user something new.

Applying this context to a user's request will make your skill more engaging and helpful. It will also keep your users coming back for more.

Learn more: Voice Design Guide: Making Sure Alexa Understands What People Are Saying

4

Simplify Choices

Sometimes a skill needs to present the user with choices. When you're building this response into your skill, it's important to think about what you are asking your user for, and how they might respond.

Here's another example using the Dev Tips skill: A user asked the skill for more information about Alexa Developer Rewards. Once the skill has provided the answer, we want to keep the conversation going by giving the user another opportunity to ask a question. It is very common to provide prompts like:

> *"Is there something else I can help you with?"*
> *"Do you have another question?"*
> *"Would you like to know something else?"*

The hope with those prompts is that the user will ask another question. However, you can quickly realize that these are all yes/no questions. If the user responds with "yes," Alexa now has to ask again, in a different way, about what their question is. If the user responds "no," what is the expected action the skill should take? Should the skill quit? Should the skill prompt the user with things they could ask the skill? As you can see, it leads to confusing interactions. Instead, the skill should ask more specific questions like:

> *"What else would you like to know about?"*
> *"What can I help you with?"*
> *"What topic can I assist you with?"*

Each of these prompts directs the user to ask their next question, and doesn't leave any ambiguity about what the skill is expecting from them.

You can also use this type of prompting when providing actual choices, like in a skill for ordering food.

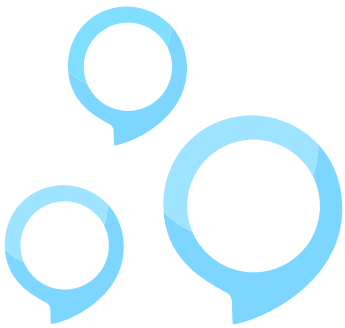
> *"Which would you like? Pizza, pasta or a salad?"*
> *"Do you want a pizza, pasta, or salad?"*

In these two examples, the user might actually want more than one of the options. You might have the ability to handle this with some well-crafted slots and utterances, but what about when they respond with "yes?" By making your phrasing more specific, you give users more confidence in their answers, and also get the kinds of responses you want.

> *"We have pizza, pasta, and salads. For your first item, what would you like?"*

By indicating that we're just talking about the first item, the user will know that they'll have an opportunity later to order the pasta, and can just order the pizza on this turn. In this response, we have also front-loaded the possible answers for the user before asking the question. This reduces the cognitive load on a user by giving them a list of options first and then asking them which one they want.

Learn more: Voice Design Guide: Prompt with Guidance for the User



5

Pass the One-Breath Test

For most interactions with Alexa, the user doesn't want a skill to drone on and on with options or descriptions in a single response. There are obvious exceptions to this rule, like with storytelling skills and adventure games. In general, a good rule of thumb for all of your responses is to give it the "one breath test."

If you can say the response out loud without taking a breath, the response is probably the right length. If you need to take a breath, think about how you could shorten your response, or break it into segments as the user progresses through the ow of your skill.

Learn more: Voice Design Guide: Use Brevity, Arrangement, and Pacing When Listing Options

6

Include a Variety of Responses

You've heard the saying "a picture is worth 1,000 words?" This is absolutely true when dealing with web and mobile design. There are consistent design elements on web and mobile apps because they are using design patterns that communicate large amounts of information with very little explanation.

However, unlike web and mobile apps, voice doesn't have a visual language. When designing for voice, you have to fall back to the consistent design elements of a conversation between two humans.

Imagine walking into your office every morning, and consistently saying "good morning" to one of your colleagues. Aside from a generic "good morning" in response, you can't predict exactly how that colleague would respond.

Now imagine your colleague, every morning, responds with "Hi. Good morning. It's going to be a great day today." After receiving the same response every day, you might eventually stop your interactions with them over time. It's unnatural for someone say the same thing every time you talk to them. The same holds true when speaking to Alexa.

Hearing something unexpected in a response makes us pay closer attention. This means that every time a user interacts with your skill, you are determining exactly how engaged they are going to be.

Here are some common anti-patterns that we see in many skills:

- Starting the skill with the same response every time:
    - *"Welcome to {myskill}. What can I do for you?"*

- Reprompting a user with the same words after each interaction:
    - *"What else can I help you with?"*

- Communicating the same information the same way each time:
    - *"The weather in {city} will be {condition} on {day}. The high will be {temp}."*

It's easy to add variety to your responses. It keeps the conversation going between Alexa and your users, and it adds the attention-grabbing variety that every conversation needs to thrive.

Try: Code Sample: Break Up the Monotony; Vary Your Responses

7

Handle the Unexpected Gracefully

When building a skill, you might find yourself making assumptions about what a user might say. It is incredibly important to make sure that you're anticipating something completely outside your expected set of responses, and handling it in a way that allows the user to get back on the rails.

In the Dev Tips skill, for example, we encourage developers to ask the skill about the issue or topic that they want to know more about. This also means that there will be times that a user says something we didn't expect.

When testing how a skill handles unexpected utterances, we like to use the phrase "pizza pie." We use this to see if the skill handles words the developer didn't plan for, and see how the skill responds. In the case of Dev Tips, "pizza pie" will deliver a response similar to this one:

> *"I heard you say pizza pie. I'm sorry, I don't know how to help you with that."*

The skill acknowledges that it heard the user, and it even repeats the words captured in the slot value so that the user understands why it missed. This gives the user an opportunity to try their question again, or ask a different one.

By handling these errors gracefully, the user understands that what they requested wasn't available, but they can continue to interact with the skill.

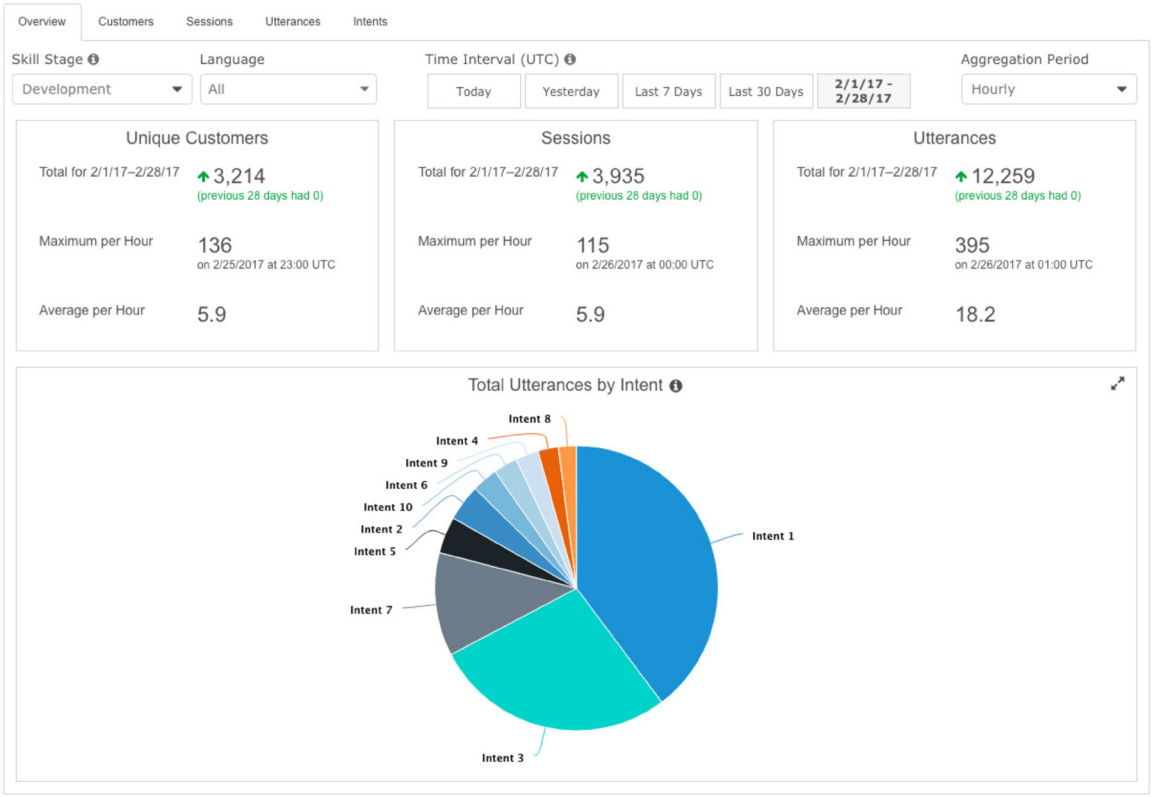
Learn more: Voice Design Guide: How to Handle Problems

8

Use Analytics to Make Enhancements

It's important to monitor your skills' analytics and interactions and use that data to make improvements to your skill.

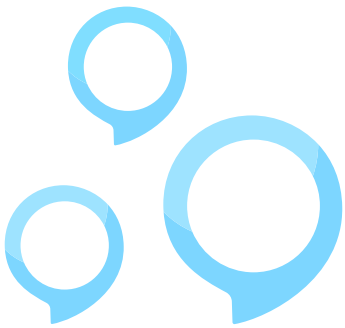
For common data points, like users, intents, sessions, and utterances, the Measure Dashboard inside the Alexa developer portal can help you collect and visualize engagement data for your skill. If you're trying to improve a published skill, you can look at these numbers at least once a week to see if any user behaviors are trending since your last release.



Using the Dev Tips skill as an example, we'll share how we monitored user behavior to further enhance the skill.

We made some additions to the sample utterances for the GetRandomTipIntent inside the Dev Tips skill, without anticipating how some of the changes would affect the overall behavior of the skill. After looking at the data, that intent had skyrocketed from 5% of interactions to nearly 40%. It was only because we were tracking these numbers that we even noticed the change. It allowed us to identify the issue, and then promptly fix it.

Learn more: What's Inside the Alexa JSON Request?



9

Provide Contextual Help

Help is often overlooked in a skill. But when done well, it is an invaluable part of the user's experience. As the developer, you often have full knowledge of what will and won't work in your skill. But users don't have the same deep knowledge. They are going to ask for help from time to time; the better your help experience is, the more likely your user will find what they are looking for.

Most help responses are static speech that gives the user a couple of ideas to try. Great help responses consider what the user is currently doing and what they've already tried, and then gives them contextual recommendations on how to continue.

Tracking your user's actions, and responding in a way that is specific to their current state will go a long way in helping them accomplish their tasks, and make your skill a reliable tool in their Alexa skill library.

Learn more: Voice Design Guide: Provide Contextual Help

10 Do Beta Testing

With any software, you can never have too much testing. Getting users in front of your software before it is available to the public gives you an opportunity to collect feedback about what works, what doesn't, and how you can make improvements.

With the Alexa developer portal, we give you the power to run a beta test of your Alexa skills. On the Launch tab, there is a section on Availability where you can set this up. All you need to provide is the email addresses of your testers (up to 500), and you can run a beta test for up to 90 days. Your beta users will get an email inviting them to participate. They simply click the link in the email and the skill will be enabled on their Alexa account.

Even just one week of beta testing with a few dozen users will go a long way in giving you the feedback you need to add the polish that your skill deserves. Beta testers will always find something that could be improved, or an edge case you hadn't anticipated. Beta users are your best friends because they expect bumps in the road. Those same bumps, when encountered by actual customers, will likely result in lower ratings in the Alexa Skills Store. Invest the time in beta testing and you can deliver a high-quality voice experience from the very beginning.

Learn more: Documentation: Skills Beta Testing

Make Money by Creating Engaging Skills Customers Love

When you create delightful skills with compelling content, customers win. You can make money through Alexa skills using in-skill purchasing or Amazon Pay for Alexa Skills. You can also make money for eligible skills that drive some of the highest customer engagement with Alexa Developer Rewards. Learn more about how you can make money with Alexa skills, and download our guide to learn which product best meets your needs.

Alexa Skills Kit

The Alexa Skills Kit (ASK) is a collection of self-service APIs, tools, documentation, and code samples that makes it fast and easy for anyone to add skills to Alexa. With ASK, you can leverage Amazon's knowledge and pioneering work in the field of voice design.

Additional Resources

Alexa Skills Kit: An Introduction
Alexa Skills Kit: Quick Start Guide
Skills Templates and Tutorials
Amazon Alexa Voice Design Guide
Make Money with Alexa Skills

Find Us on Social

Twitter: @AlexaDevs
Facebook: Alexa Developers
LinkedIn: Amazon Alexa Developers
YouTube: Alexa Developers