

Situational Design

How to Shift from Screen-First to Voice-First Design



[Paul Cutsinger](#)

Head of Alexa Voice Design Education

Amazon Alexa

© 2018 Amazon.com, Inc., or its affiliates

Introduction

With the rise of voice services like Amazon Alexa, voice is fast-becoming a common user interface. Customers of all ages are using their voice to play games, get the latest news, and control their smart home devices. People are embracing voice-first user interfaces because they are natural, conversational, and user-centric. A great voice experience allows for the many ways people might express meaning and intent. It is rich and flexible. Because of this, building for the voice isn't the same as building graphical user interfaces (GUIs) for the web or mobile.

With every big shift in human-computer interaction, we first apply the design concepts of the previous generation to the new technology. We eventually adapt to create new design patterns that take advantage of the control surface. When we added a mouse, we first came up with complicated shortcut-key schemas. When we added touch, we first treated taps like simple clicks instead of fuller gestures. When we added voice, we first saw it as a replacement for a keyboard and tapping controls instead of embracing its potential to enable conversational interactions.

In order to evolve from building voice keyboards to creating truly conversational, voice-first experiences, you first need to embrace the principles of voice design. You can't simply add voice to your app and call it a voice experience; you have to reimagine the entire experience from a voice-first perspective. Designing for voice is different from designing for screens. There are subtle but potent differences to consider. Here are four design patterns that are unique to voice-first interactions:

1. **Be adaptable:** Let users speak in their own words
2. **Be personal:** Individualize your entire interaction
3. **Be available:** Collapse your menus; make all options top-level
4. **Be relatable:** Talk with them, not at them

Learn and adopt these design patterns in order to create a rich voice-first experience that users find engaging.



1

Be Adaptable: Let Users Speak in Their Own Words

User experience designers work hard to make things easy to learn. But at the end of the day, it is the users who are required to learn and often change their ways to use the technology. Think about how smartphone users rarely switch from one phone type to another. Or how most people use a qwerty keyboard versus a dvorak keyboard. They've had to learn the UI and don't want to have to learn another from scratch.

When you're building experiences for mobile or the web, you present a singular UI to all users. You, the designer, pick things like text, icons, controls, and layout. Then the users leverage their internal pattern matching and natural-language-understanding (NLU) engine, their brain, to make sense of the UI you've laid out and act accordingly. A simple but common example is the "OK" button on your app or site. Before clicking on the button, users will scan the whole page, looking for what they want to do. In this moment, they apply their own language-understanding criteria and determine that "OK" is the highest confidence match to what they want to do. Once they click, the OK function is called.

In voice-first interactions, the OK function still exists, but it is called an intent—what user wants to accomplish. The difference is that in voice, there is no screen of options for the user to scan, no button labeled "OK" to guide them. This means that users lack this navigational guidance. However, it also means they don't have to bend to the technology.

When we speak, we likely won't say "OK" every time or perhaps at all; we might instead say, "next," "now what," "yep," "let's do it," "sounds good," "got it," "super," "roger that," and so on. In voice, these phrases are called utterances, and a conversational UI will accommodate all of these responses. Through automatic speech recognition (ASR) and natural language understanding (NLU), voice services like Alexa will resolve them to an "OK" intent.

The key difference with voice is that Alexa is providing the NLU, not the user. That means the designer's job shifts from picking the one label that works best for everyone to providing a range of utterances to train the NLU engine. The technology must bend to the user. That's because we all have our unique style of speech, and without visual guidelines to constrain us into uniformity, our inputs will vary greatly.

This isn't to say that voice-first UIs can't provide clues and guidelines for the user. For example, it may share a list of things users can say in order to help us get started. However, the experience should not feel limiting or predestined; one of the strengths of voice experiences is its conversational nature and its ability to accommodate human tendencies such as emotions, personality, and variety. People should be able to simply talk as they do in everyday life; they should be able to say, "Please turn off the lights," "Turn off the lights," and "Good night." This means unlike a singular graphical user interface, a voice-first UI needs to account for the many paths that a user may reach the same destination. While a GUI's ambition is to enable customers to learn the interface once and use many times over, a voice-first UI's ambition is to make it so customers never have to learn the UI since it allows the customers to set the course.



2

Be Personal: Individualize Your Entire Interaction

It's important to create a personal experience for users, no matter what your form factor. However, the way we accomplish this in voice-first UIs is different from the way we achieve this for screen interactions. For example, on the web or mobile, you might have a single UI that dynamically surfaces personal content such as music preferences. In voice interactions, in addition to personal content, the interaction itself must be personalized. This is because a singular interaction for all users, even with dynamic content, will feel robotic and sterile rather than authentic and approachable. Voice-first interactions should offer richer design. They should be predictable yet varied and delightful.

The simplest implementation of this is to greet your user with a random welcome. Instead of saying, "Hello," every time, you can say, "Howdy," "Welcome back," "Happy Friday," "How goes it?" and so on, depending on the tone you want to set. More sophisticated voice-first experiences will have a memory. At first, users need help understanding the scope of the experience that you offer and what they can do. So, the first experience might be more verbose. Over time, users will want to get right into the action; they won't want to hear the long introduction every time they visit, and your welcome message should evolve to match expectations.

An experience with memory would enable users to pick things up right where they left off. It would also get to know the user better over time. It might greet users with "Welcome back, Paul," "Wow, you've been here to work out eight days in a row. Crushing it! Would you like your normal one-minute workout, or are you up for trying 90-second yoga?" or "Welcome back. Last time you were here, you were on question 12 of your SAT prep. Would you like to continue or start over?" Each of these is personable and respectful of what the user has done in the past.

Think about how our relationships with each other evolve over time, and how our conversations also advance to match. If you've talked to someone many times over and he or she returns to asking rudimentary questions you've already covered in previous interactions, you'd likely feel unheard and, as a result, become unengaged. Likewise, if your conversations with an acquaintance consists of the same questions or flow every single time, your enjoyment will become more automatic over time. While we prefer consistent GUIs, we have low tolerance for repeat voice interactions. Therefore, it is appropriate to remain predictable and concise yet also surprising and delightful when designing voice-first UIs.

3

Be Available: Collapse Your Menus; Make All Options Top-Level

When we design for the screen, it's important to determine what's surfaced at the top level and what gets organized into menus or a similar information architecture. This is because there are a limited number of pixels and visual concepts that we surface at once. A GUI's information architecture effectively sets the customer's path. The menus set the hierarchy, surfacing the most important buckets as top-level navigation and nesting the less important items within those menus. On the screen, pixels are the scarcity, and this constraint makes hierarchical menus mandatory; not every item can live at the top level. More importantly, presenting too many options at once adds a cognitive load on the user.

With voice UIs, the opposite is true. Hierarchical menus don't help the user; they constrain what they can users can do in the moment and they add cognitive load to figure out how the experience is organized. In order to enable conversational, freeform interactions, voice-first UIs require a different arrangement of their features, with all available options presented at the top level.

Imagine a banking app that includes the ability for users to look up the routing number. This is an important piece of information but something users will only rarely need. In an app, the best way to present this information would be via a menu system. In voice, the opposite is true. You'd never want the user to have to say, "Ask Bank of Redmond for: menu, menu, menu, routing number." Instead, you'd want to enable the user to simply say, "Ask Bank of Redmond for my routing number." In other words, the routing number intent is now presented at the top level. This doesn't add cognitive load because users aren't seeing all the options presented at once. In fact, it allows for more serendipitous discovery because they can simply try things that are reasonable to expect from an experience like this.

While menus add depth to GUIs, they introduce friction to voice-first UIs. Voice interactions should instead offer their experience at the top level—without the need to learn its information architecture.



4

Be Relatable: Talk with Them, Not at Them

As we've discussed, GUIs are inherently inflexible; users have to first learn how to navigate them before they can use them to find what they need. A good GUI is one that clearly presents itself in an understandable and usable way. It presents a consistent path each time so users can habitually perform the task and achieve the same goal every time in minimal time. It is declarative and inherently non-cooperative.

A good voice-first UI is cooperative because conversations are cooperative. As linguist Paul Grice stated in his cooperative principle, participants of a conversation cooperate in order to achieve mutual conversational ends. In other words, both participants look beyond the explicit statements to explore the implicit meaning in order to help advance the conversation.

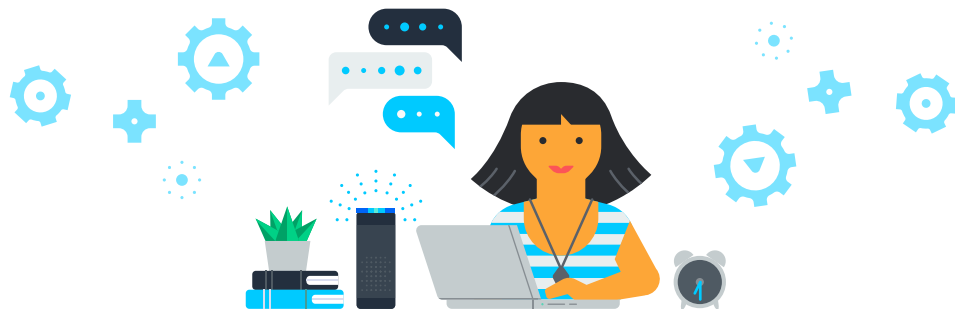
This cooperative spirit carries over to voice user interfaces. If the user says, "I'd like to get a gigantic dog," the VUI might respond with, "I have several large dog breeds in mind. Would you prefer more of a family dog or guard dog?" That response indicates implicit confirmation that "gigantic" was understood. The response also asks a forward-moving question in an attempt to help the user narrow down the choices.

Behind the scenes here, we are looking to gather values like {size} and {temperament}. The size our database or API requires is large, med, and small. But rather than saying, "You must now say large, med, or small. What size do you want?" the VUI understands that large has synonym words or phrases like "huge," "gigantic," "waist-high," "as big as a pony," "that my daughter can ride," and so on. In this way, you can match what a person will say ("medium" or a synonym thereof) to what your API expects ("med").



Additionally, voice-first experience should account for situations where the user overanswers or answers a different question. For example, if the prompt is “Where would you like to go?” and the answer is “I’d like to go to Portland for some kayaking,” or even “kayaking,” the next prompt should not be, “What would you like to do there?” The user already provided “kayaking” as the activity.

In screen UIs, a wizard is a fairly common metaphor for asking users for information in a constrained and linear path. In voice, users will provide information in the way they think is most appropriate. Where you might use a wizard or a form in GUI, in voice, you’ll want to use dialog management to manage the state and flow of the conversation. In other words, instead of talking at users the same way each time whether they understand or not, voice-first UIs talk with them and cooperatively carry the interaction forward.



Build for Voice, the UI of the Future

These four design principles are fundamental to rich and compelling voice-first interactions; however, they are just the start. Explore the [Amazon Alexa Voice Design Guide](#) to dive deeper into the principles and to start designing your own interaction.

We are in the early days of voice technology. Join the pioneers who are redefining the way people interact with technology via voice. Help the community and the industry discover the design patterns that will shape conversational interactions. [Start building with the Alexa Skills Kit \(ASK\)](#), a collection of self-service APIs, tools, documentation, and code samples that makes it fast and easy for anyone to build skills. Make Alexa smarter and reach customers where they are through tens of millions of Alexa-enabled devices. And earn money for engaging skills that customers love through the [Alexa Developer Rewards program](#).

Start creating now and help shape the UI of the future.

Additional Resources

[Alexa Skills Kit: An Introduction](#)
[Alexa Skills Kit: Quick Start Guide](#)
[Skills Templates and Tutorials](#)
[Amazon Alexa Voice Design Guide](#)
[Alexa Developer Rewards](#)

Find Us on Social

Twitter: [@AlexaDevs](#), [@PaulCutsinger](#)
Facebook: [Alexa Developers](#)
LinkedIn: [Amazon Alexa Developers](#)
YouTube: [Alexa Developers](#)

