



Workshop

# Build Your First Skill for Amazon Alexa

---

**MLH** localhost

 amazon alexa

1

*Using your Web Browser,  
Open this URL:*

**<http://mlhlocal.host/lhd-resources>**

---

2

*Click on the workshop you're attending, and find:*

- Setup Instructions
- The Code Samples
- A demo project
- A Workshop FAQ
- These Workshop Slides
- More Learning Resources



***Our Mission** is to Empower Hackers.*

**65,000+**  
HACKERS

**12,000+**  
PROJECTS CREATED

**3,000+**  
SCHOOLS

***We hope you learn something awesome today!***

*Find more resources: <http://mlh.io/>*

# What will you **learn** today?

- 1 Understand Voice User Interfaces & what you can build using them.
- 2 Meet Alexa, an intelligent personal assistant developed by Amazon.
- 3 Create your first voice powered app with Amazon Alexa.

# Table of Contents



- 1.** Introduction to Alexa & Voice UIs
- 2.** Developing For Alexa
- 3.** Develop Your Alexa Skill
- 4.** Customize The Skill
- 5.** Test The Skill
- 6.** Publish Your Skill
- 7.** Review & Quiz
- 8.** Next Steps

# What is Alexa?

Alexa is a Voice User Interface (VUI), that lets you **speak** commands, instead of clicking buttons or typing on your keyboard.



Alexa listens to spoken input, uses it to execute tasks or skills in the cloud, and then returns output -- just like a JavaScript function.

# Why do Voice UIs Matter?



Instead of typing, clicking, or tapping - we can physically separate ourselves from our devices and speak commands naturally.

Voice UIs can run code in the cloud and communicate with IoT devices, making them ideal for homes, cars, & more.



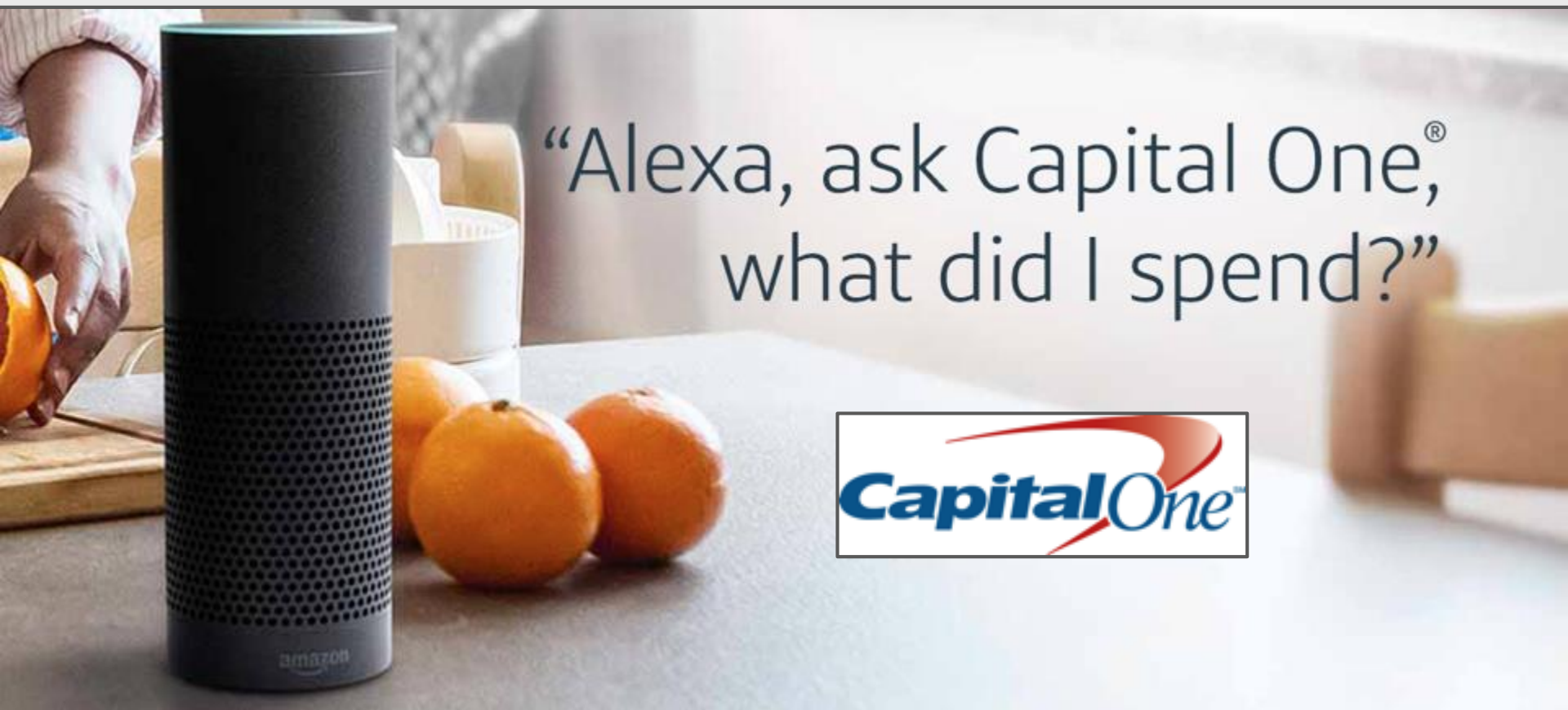
# What can you build with Alexa?



"Alexa, ask Lyft for a Lyft Line to work."



# What can you build with Alexa?



“Alexa, ask Capital One<sup>®</sup>,  
what did I spend?”



“Alexa, ask Capital One, what did I spend?”

# What can you build with Alexa?

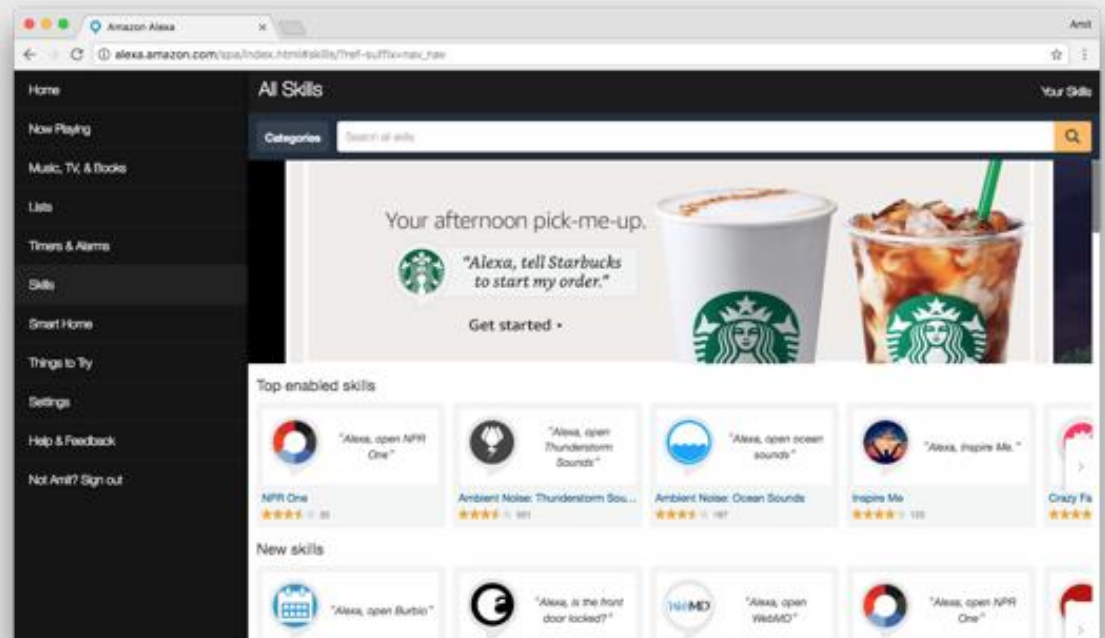


"Alexa, tell Starbucks start my order."

# Alexa has Skills. Lots of them.

Companion app for device setup, skills, remote control, and more.

[alexa.amazon.com](https://alexa.amazon.com)



# Table of Contents

**1.** Introduction to Alexa & Voice UIs



**2.** Developing For Alexa

**3.** Develop Your Alexa Skill

**4.** Customize The Skill

**5.** Test The Skill

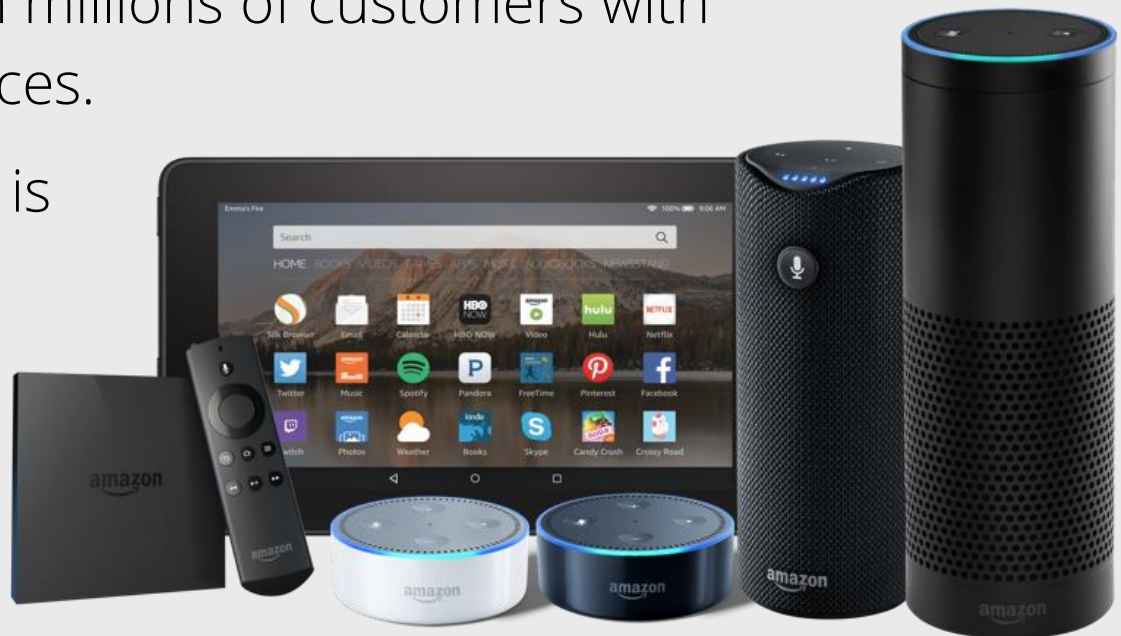
**6.** Publish Your Skill

**7.** Review & Quiz

**8.** Next Steps

# Why build an Alexa Skill?

1. Alexa, Amazon's voice service, is already integrated into Echo devices and can perform hundreds of skills.
2. Your skill can reach millions of customers with Alexa enabled devices.
3. Building with Alexa is free, easy, & fun!




# Alexa Skills are made of 2 parts:

1. **Front End** - The Alexa Voice UI handles text to speech, converting the audio into something our app can use, etc.
1. **Back End** - The logic code that actually powers our app. Usually this is written using the Alexa Code Editor.



# Table of Contents

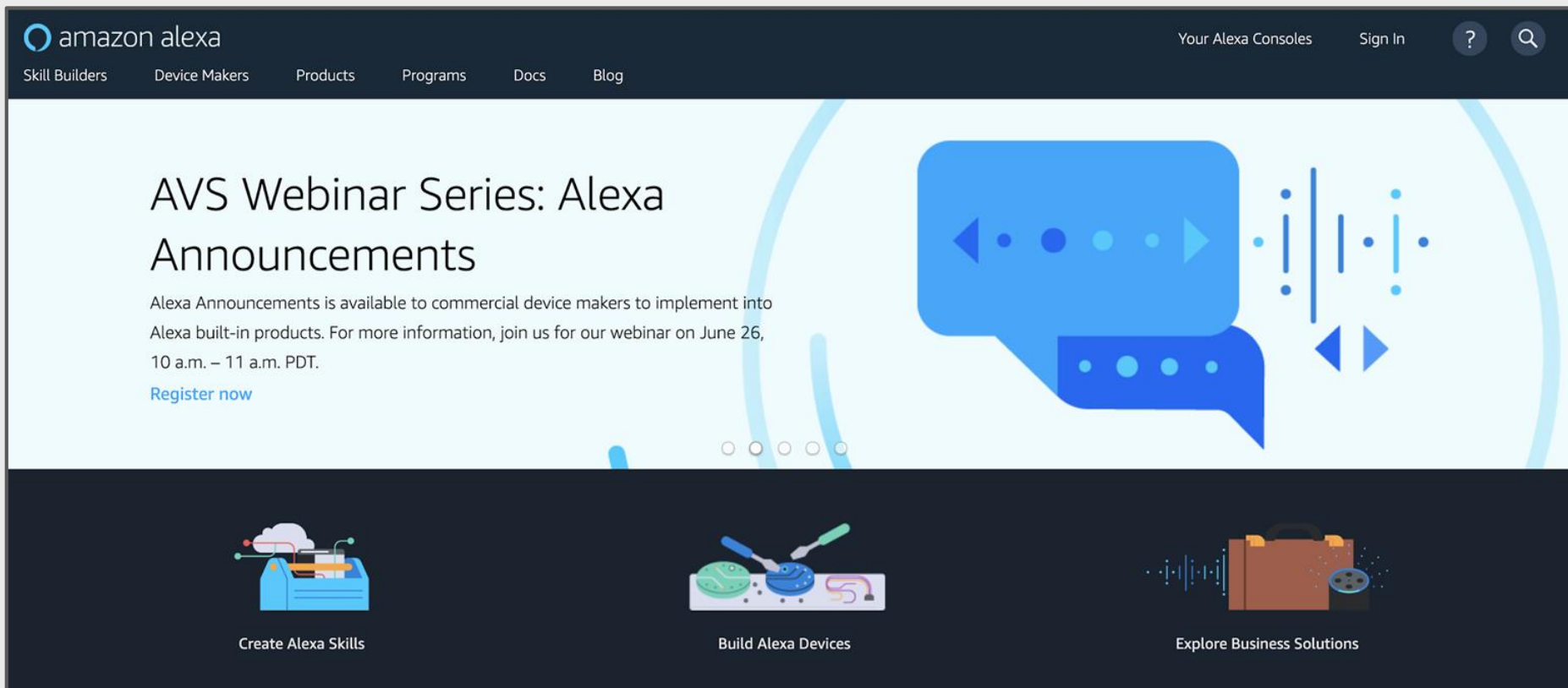
1. Introduction to Alexa & Voice UIs
2. Developing For Alexa
-  3. Develop Your Alexa Skill
4. Customize The Skill
5. Test The Skill
6. Publish Your Skill
7. Review & Quiz
8. Next Steps

**Now that you know more about how  
Alexa works, let's make an Alexa  
Developer account.**



# Sign into the Amazon Developer Portal

1. Navigate to [mlhlocal.host/alexa-portal](http://mlhlocal.host/alexa-portal).
2. Click **Sign In**.



# Create your account

3. If you have an Amazon Developer Account, sign in. If you do not, select **Create your Amazon Developer account**.

## Sign in

Email (phone for mobile accounts)


Password [Forgot your password?](#)

Sign in

By continuing, you agree to Amazon's [Conditions of Use](#) and [Privacy Notice](#).

New to Amazon Developer?

Create your Amazon Developer account



## Create account

Your name

Email

Password

Re-enter password

Create your Amazon Developer account

# Create your Developer Profile

4. Fill out the registration form.

## Registration

1. Profile Information

2. App Distribution Agreement

3. Payments

\* Indicates a required field.

Country/Region *	United States
First name *	Local
Last name *	Host
Email address *	localhost@mlh.io
Phone number * <small>e.g. 212-555-1212, +44 0161 715 3369</small>	212-555-1212
Fax number	
Developer name or company name * <small>Displayed on your apps at Amazon.com</small>	MLH Localhost
Developer description <small>Maximum characters: 4000, Remaining: 3611</small>	Major League Hacking (MLH) is the official student hackathon league. Each year, we power over 200 weekend-long Invention competitions that inspire innovation, cultivate communities, and teach compute science skills to more than 65,000 students around the world. MLH is an engaged and passionate maker community, consisting of the next generation of technology leaders and entrepreneurs.
Address 1 *	149 East 23rd St
Address 2	#438
City *	New York
State *	New York
Zip code/Postal code *	10159
Customer support email address	
Customer support phone	
Customer support website	

Cancel

Save and Continue

# Create your Developer Profile

## 5. Agree to the Terms of Use.

1. Profile Information

2. App Distribution Agreement

3. Payments

English | 中文 (Chinese)\* | 日本語 (Japanese)\*

We've revised the royalty structure for PC Games, PC Game In-App Products, PC Software, and PC Software In-App Products. Please review the full text of the updated Agreement carefully.

**Changes to App Distribution and Services Agreement posted January 1, 2018**

We have increased the Royalties we pay you for movies and TV Subscription In-App Products. We've updated the App Distribution and Services Agreement to add terms related to Royalties and payments. Please review the full text of the updated Agreement carefully.

**Changes to App Distribution and Services Agreement posted October 4, 2017**

We are expanding the Alexa voice service to India. We've updated the App Distribution and Services Agreement to designate the Amazon entity party to the Agreement with developers who reside in India and develop Alexa Skills. Please review the full text of the updated Agreement carefully.

**Changes to App Distribution and Services Agreement posted August 31, 2017**

Alexa now supports kid skills. We've updated the App Distribution and Services Agreement to add terms related to the distribution on Alexa of skills directed to children under 13. Please review the full text of the updated Agreement carefully.

**Changes to App Distribution and Services Agreement posted April 18, 2017**

We have made a new app testing service available to developers. We've updated the App Distribution and Services Agreement so that the terms applicable to Live App Testing apply to any app testing services we make available. Please review the full text of the updated Agreement carefully.

**Changes to App Distribution and Services Agreement posted February 10, 2016**

In order to enable developers to use the Advertising ID available on certain Fire devices, we are making available the Advertising ID API in our portal. We've updated the App Distribution and Services Agreement to add terms related to the Advertising ID API and to reflect that Amazon Digital Services, Inc. has converted into a limited liability company, Amazon Digital Services LLC. Please review the full text of the updated Agreement carefully.

Print Agreement

\*Please note: All non-English translations are provided for informational purposes only and are non-binding. By clicking "Accept and Continue" below, you agree to the English language version of the Mobile App Distribution Agreement.

CancelAccept and Continue

# Create your Developer Profile

6. Select "No" for both options on this screen.
7. Save and Continue.

## Registration

✓ 1. Profile Information

✓ 2. App Distribution Agreement

3. Payments

\* indicates a required field.

Do you plan to monetize your digital content, such as charging for apps or games or selling in-app items or in-game items, or by receiving cash rewards for your skills? \*

☒ No  
☐ Yes

Do you plan to monetize apps by displaying ads from the Amazon Mobile Ad Network or Mobile Associates? \*

☒ No  
☐ Yes

Note: You may still monetize later if you select "No" by entering payment and tax information from the Settings menu.

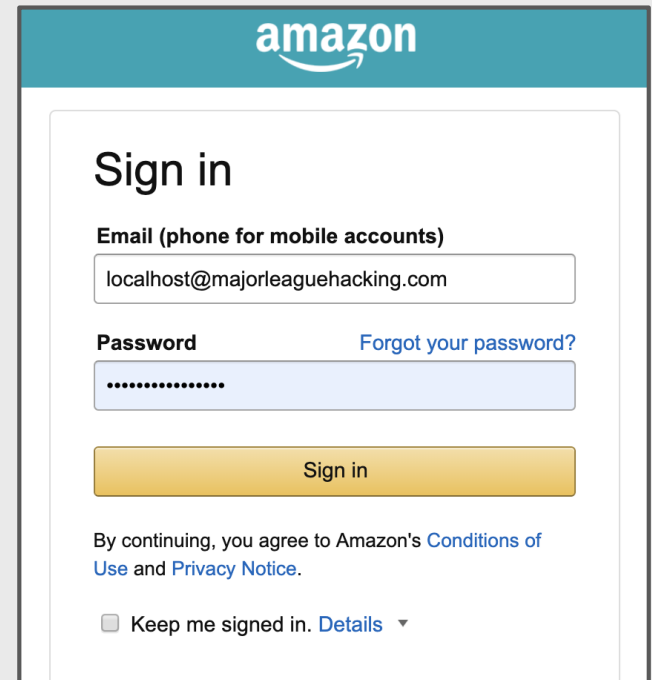
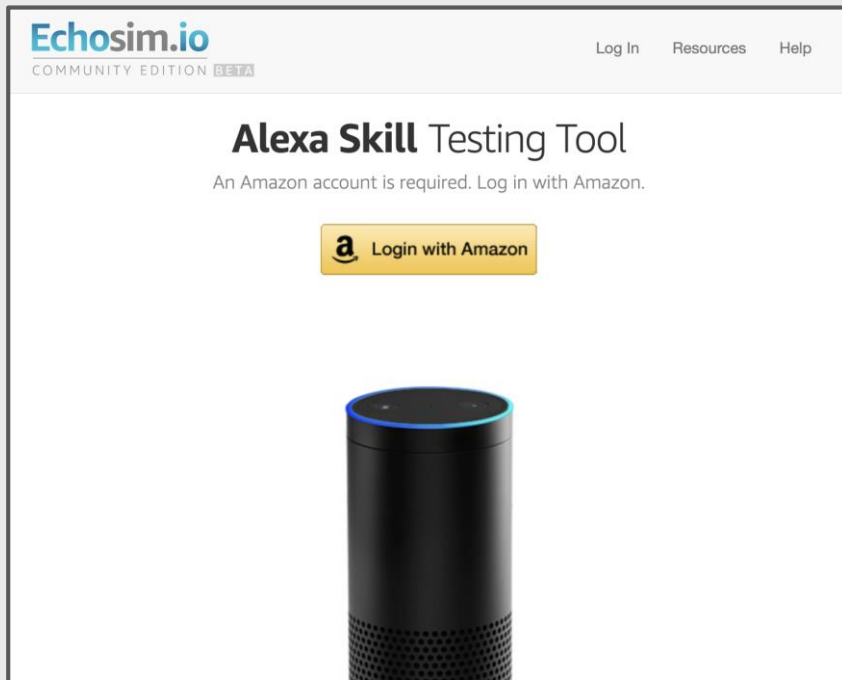
Cancel

Save and Continue

**Now that you have an Amazon  
Developer account, we can test the skill  
we're going to build!**

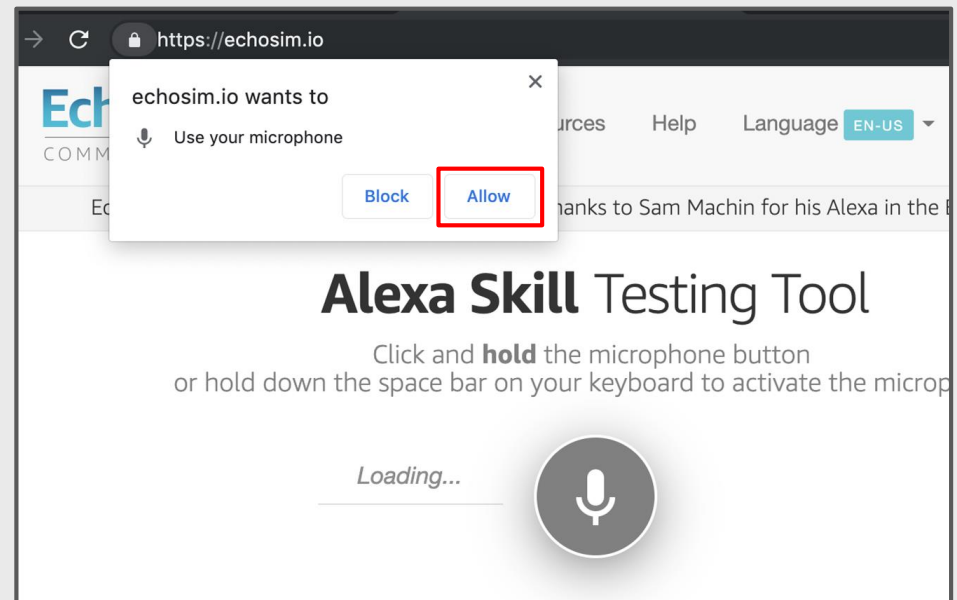
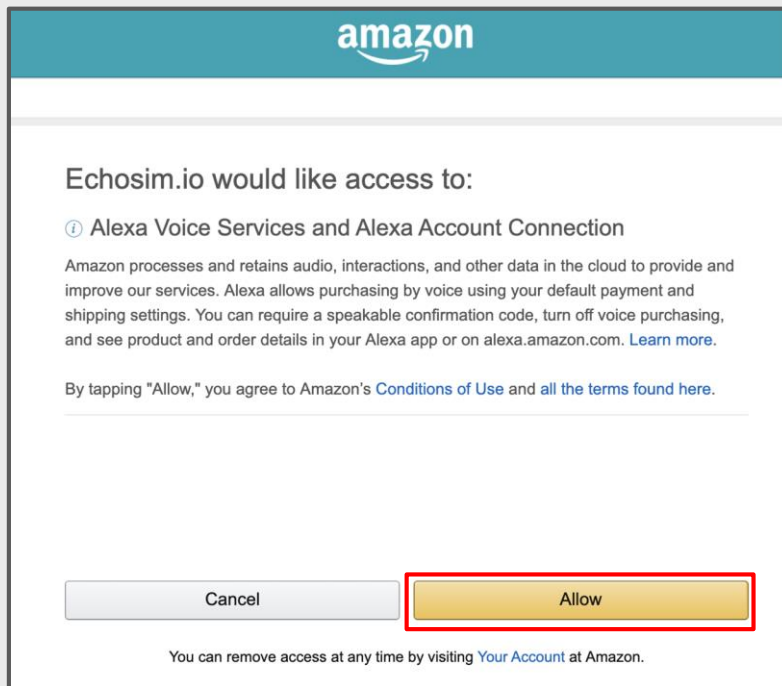
# Demo: Cake Walk Skill

1. In another tab in your browser, navigate to **Echosim.io**.
2. Select **Login with Amazon**.
3. Enter your information and sign in.



## Demo: Cake Walk Skill

4. Select **Allow** to connect your account.
5. If prompted, allow the browser to use your microphone.



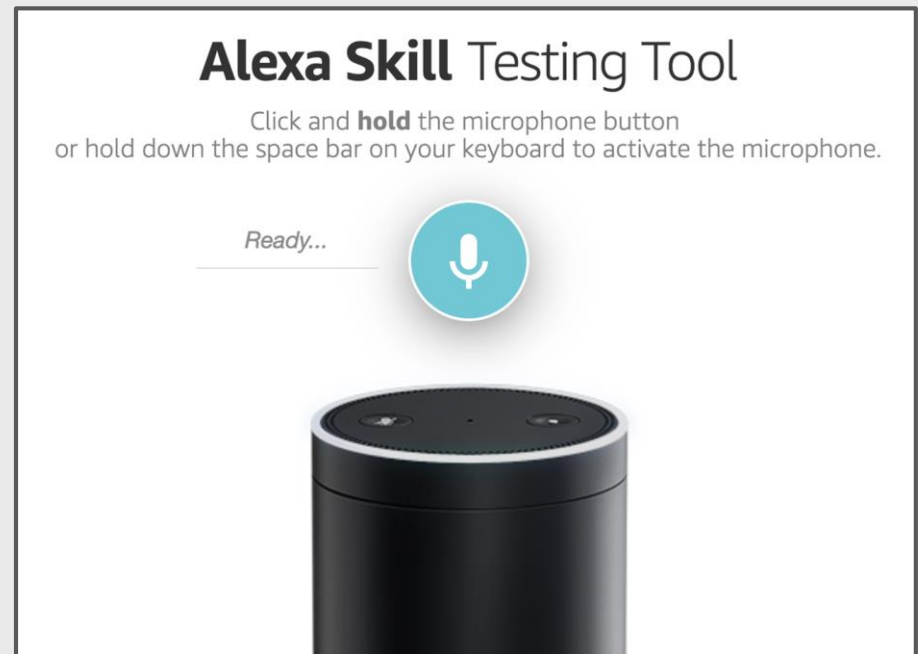


## Demo: Cake Walk Skill

6. Select the microphone button. Then, say the following:

“Alexa, open Cake Walk.”

“Alexa, ask Cake Walk  
for my birthday.”



**Awesome! Now that you know what  
you're going to build, let's return to the  
Alexa Developer Portal.**

# Steps to Build Your Skill:

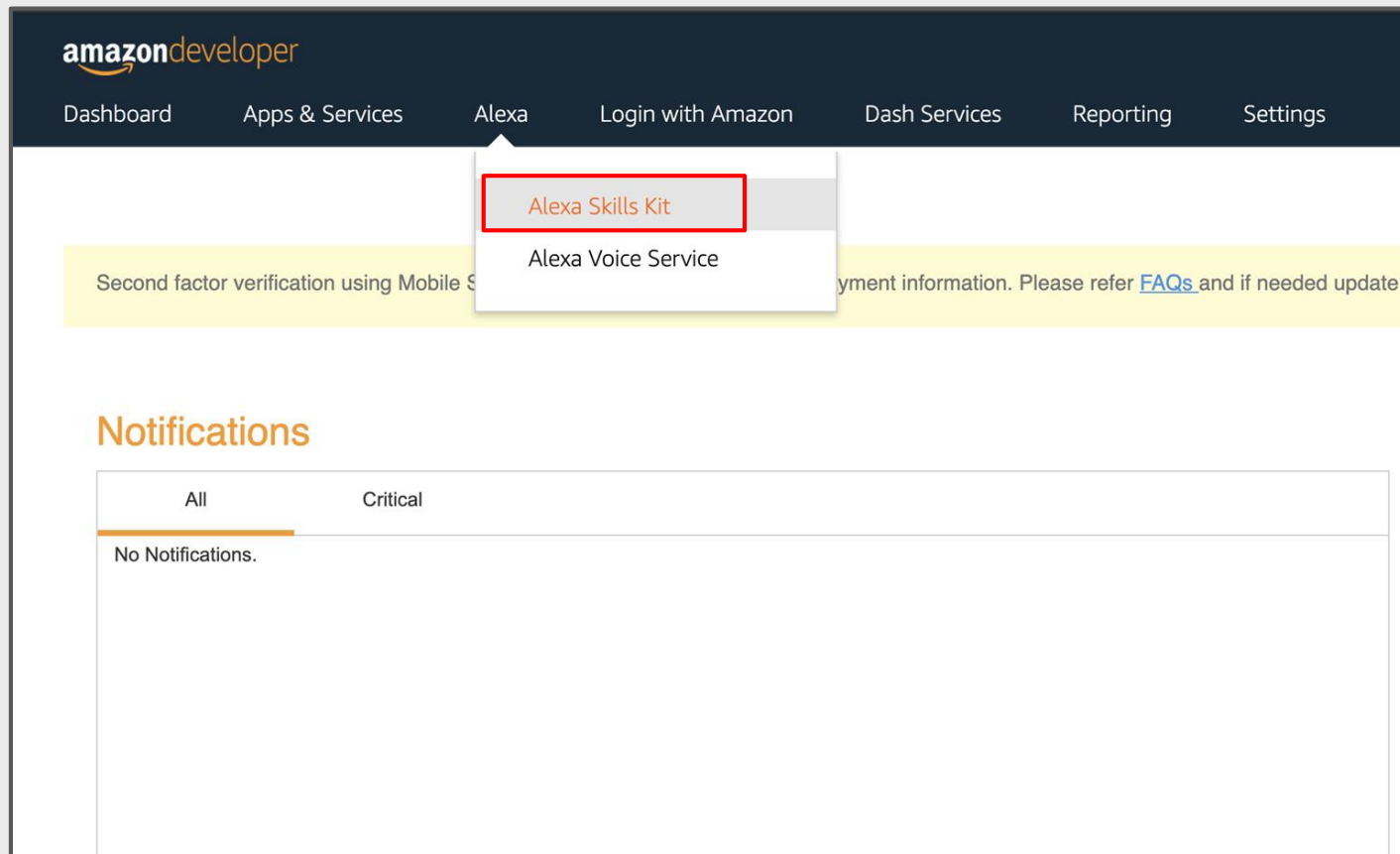
To build your first skill, we'll complete the following steps:

1. Create an Alexa Skill in the Developer Portal
2. Write the Backend Code for Your Skill
3. Test your Skill
4. Publish Your Skill



# Create a New Alexa Skill

1. Navigate to Alexa Skills Kit.



# Create a New Alexa Skill

## 2. Select **Create Skill**.

The screenshot shows the Amazon Alexa Skills Kit Developer Console. At the top, there's a navigation bar with the 'amazon alexa' logo, 'Your Alexa Consoles', and user icons. Below this is a banner for '\*\*\* Make Money with Your Alexa Skills \*\*\*'. The main content area has a 'Welcome to the new Alexa Skills Kit Developer Console' message. There are three tabs: 'Skills', 'Earnings', and 'Payments'. The 'Skills' tab is active, showing 'Alexa Skills' with a 'Create Skill' button in the top right. Below this is a table with columns: SKILL NAME, LANGUAGE, TYPE, MODIFIED, STATUS, and ACTIONS. The table is currently empty. At the bottom of the main content area, there's a large light blue box with a plus icon and the text 'Alexa Skills' and 'Create your first skill or learn more about [Alexa Skills Kit](#)'. A red box highlights the 'Create Skill' button in this section. The footer shows 'English' and copyright information.

amazon alexa

Your Alexa Consoles

Feedback forum

\*\*\* Make Money with Your Alexa Skills \*\*\*

On May 3, we announced general availability of in-skill purchasing (ISP) and Amazon Pay for Alexa Skills. With ISP, you can sell premium digital content that enriches your skill experience. With Amazon Pay, you can sell physical goods or services through your skill. You can also continue to make money for eligible skills that drive some of the highest customer engagement through Alexa Developer Rewards. [Learn more.](#)

Welcome to the new Alexa Skills Kit Developer Console

Curious about what's new? [Watch the video overview](#) or [read about what's changed.](#)

Skills Earnings Payments

Alexa Skills

Create Skill

SKILL NAME	LANGUAGE	TYPE	MODIFIED	STATUS	ACTIONS
------------	----------	------	----------	--------	---------

Alexa Skills

Create your first skill or learn more about [Alexa Skills Kit](#)

Create Skill

English

© 2010–2018, Amazon.com, Inc. or its affiliates. All Rights Reserved. Terms Alexa Developer Blog Alexa Skills Kit

# Create a New Alexa Skill

3. Name your skill Cake Walk (or whatever you wish).

## Create a new skill

**Skill name**

Cake Walk

9/50 characters

**Default language**

English (US) ▼

More languages can be added to your skill after creation

# Create a New Alexa Skill

4. Select **Custom** for Skill Model Type.

## Choose a model to add to your skill

There are many ways to start building a skill. You can design your own custom model or start with a pre-built model. Pre-built models are interaction models that contain a package of intents and utterances that you can add to your skill.

### Custom

Design a unique experience for your users. A custom model enables you to create all of your skill's interactions.

SELECTED

### Flash Briefing

Give users control of their news feed. This pre-built model lets users control what updates they listen to.

"Alexa, what's in the news?"

### Smart Home

Give users control of their smart home devices. This pre-built model lets users turn off the lights and other devices without getting up.

"Alexa, turn on the kitchen lights"

# Create a New Alexa Skill

5. Select **Alexa-Hosted** for Skill Backend Resource.

## Choose a method to host your skill's backend resources

You can provision your own backend resources or you can have Alexa host them for you. If you decide to have Alexa host your skill, you'll get access to our code editor, which will allow you to deploy code directly to AWS Lambda from the developer console.

### Provision your own

Provision your own endpoint and backend resources for a skill. With this option, you will not gain access to the console's code editor.

### Alexa-Hosted

Alexa will host skills in your account up to the AWS Free Tier limits. You will gain access to an AWS Lambda endpoint, 5 GB of media storage with 15 GB of monthly data transfer, and a table for session persistence. [Learn more](#)

SELECTED



# Create a New Alexa Skill

6. Scroll back to the top. Select **Create skill** in the top right corner.

Create a new skill

Cancel

Create skill

Skill name

Cake Walk

9/50 characters


Default language

English (US) ▼

More languages can be added to your skill after creation

**It will take about a minute for your skill to provision. Then, you will be redirected to the Skill Builder.**

# Table of Contents

1. Introduction to Alexa & Voice UIs
2. Developing For Alexa
3. Develop Your Alexa Skill
-  4. Customize The Skill
5. Test The Skill
6. Publish Your Skill
7. Review & Quiz
8. Next Steps

# Code The Backend

1. Select the Code tab at the top of the Skill Builder. Let's understand what the code inside **index.js** does.

The screenshot shows the Amazon Alexa Skill Builder interface. At the top, there are tabs for < Your Skills, Cake Walk, Build, Code (selected), Test, Distribution, Certification, and Analytics. On the right, there are buttons for Save, Deploy, and Promote to live, along with a Feedback forum link. Below the tabs, there's a file explorer on the left showing the Skill Code directory with files like index.js, package.json, and util.js. The main area displays the code for index.js, which is a JavaScript file for handling Alexa intents. The code includes comments and logic for a LaunchRequest and a HelloWorldIntent. A green notification box in the bottom right corner indicates a successful build, stating: 'Build Successful. If you make any new changes, you will need to rebuild your model for them to take effect. Go to build'. The notification also shows the date and time: Thursday, May 23, 2019, 7:24 PM.

```
1 // This sample demonstrates handling intents from an Alexa skill using the Alexa Skills Kit SDK (v2).
2 // Please visit https://alexa.design/cookbook for additional examples on implementing slots, dialog management,
3 // session persistence, api calls, and more.
4 const Alexa = require('ask-sdk-core');
5
6 const LaunchRequestHandler = {
7   canHandle(handlerInput) {
8     return handlerInput.requestEnvelope.request.type === 'LaunchRequest';
9   },
10  handle(handlerInput) {
11    const speechText = 'Welcome, you can say Hello or Help. Which would you like to try?';
12    return handlerInput.responseBuilder
13      .speak(speechText)
14      .reprompt(speechText)
15      .getResponse();
16  }
17 };
18 const HelloWorldIntentHandler = {
19   canHandle(handlerInput) {
20     return handlerInput.requestEnvelope.request.type === 'IntentRequest'
21       && handlerInput.requestEnvelope.request.intent.name === 'HelloWorldIntent';
22   },
23   handle(handlerInput) {
24     const speechText = 'Hello World!';
25     return handlerInput.responseBuilder
26       .speak(speechText)
27       // .reprompt('add a reprompt if you want to keep the session open for the user to re
28       .getResponse();
29   }
30 };
```

Build Successful  
If you make any new changes, you will need to rebuild your model for them to take effect.  
[Go to build](#)  
Thursday, May 23, 2019, 7:24 PM

# Code Review: The Handlers Object

- Alexa skills are composed of **handlers**.
- A **handler** object tells Alexa how to handle various actions.
- **LaunchRequestHandler{}** contains the code that our Skill will run when someone invokes our skill.

## Index.js

```
6 const LaunchRequestHandler = {
7   canHandle(handlerInput) {
8     return handlerInput.requestEnvelope.request.type === 'LaunchRequest';
9   },
10  handle(handlerInput) {
11    const speechText = 'Welcome, you can say Hello or Help. Which would you like to try?';
12    return handlerInput.responseBuilder
13      .speak(speechText)
14      .reprompt(speechText)
15      .getResponse();
16  }
17 };
```

# Code Review: The Handler Function

- In the last lines of **index.js**, we export each handler so that they can be used by other parts of the app.
- These are the default handlers included with the Cake Walk Skill. You can also write your own!

## Index.js

```
108 exports.handler = Alexa.SkillBuilders.custom()  
109     .addRequestHandlers(  
110         LaunchRequestHandler,  
111         HelloWorldIntentHandler,  
112         HelpIntentHandler,  
113         CancelAndStopIntentHandler,  
114         SessionEndedRequestHandler,  
115         IntentReflectorHandler) // IntentReflectorHandler is last to not over  
116     .addErrorHandlers(  
117         ErrorHandler)  
118     .lambda( );  
119
```

# Edit your Code

1. When a user opens an Alexa Skill, Alexa replies with a greeting to let the user know it has opened. Customize what Alexa will say when users invoke your skill on line 11.

## Index.js

```
6 const LaunchRequestHandler = {
7   canHandle(handlerInput) {
8     return handlerInput.requestEnvelope.request.type === 'LaunchRequest';
9   },
10  handle(handlerInput) {
11    const speechText = 'Welcome, you can say Hello or Help. Which would you like to try?';
12    return handlerInput.responseBuilder
13      .speak(speechText)
14      .reprompt(speechText)
15      .getResponse();
16  }
17 };
```

## Edit your Code

2. The Alexa Skills SDK has some built in functions to help us. One of those is **.reprompt()**, which tells the app to repeat. We don't want to repeat, so add 2 slashes (//) in front of **.reprompt()** to comment it out.

### Index.js

```
6 const LaunchRequestHandler = {
7   canHandle(handlerInput) {
8     return handlerInput.requestEnvelope.request.type === 'LaunchRequest';
9   },
10  handle(handlerInput) {
11    const speechText = 'Welcome, you can say Hello or Help. Which would you like to try?';
12    return handlerInput.responseBuilder
13      .speak(speechText)
14      // .reprompt(speechText)
15      .getResponse();
16  }
17 };
```



# Save And Deploy

3. Save your Skill.
4. Then Deploy it so that we can test it!

alex developer console

< Your Skills Cake Walk Build Code Test Distribution Certification Analytics Feedback forum

Save Deploy Promote to live

*Deploy before testing your skill*

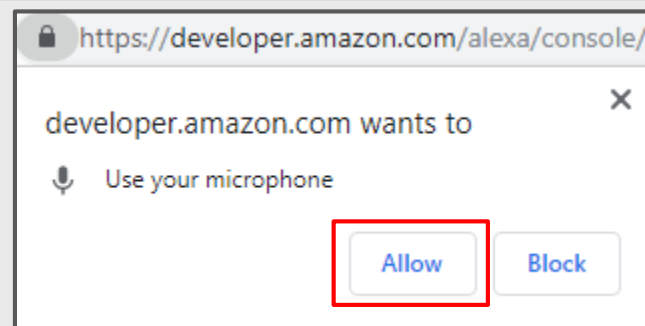
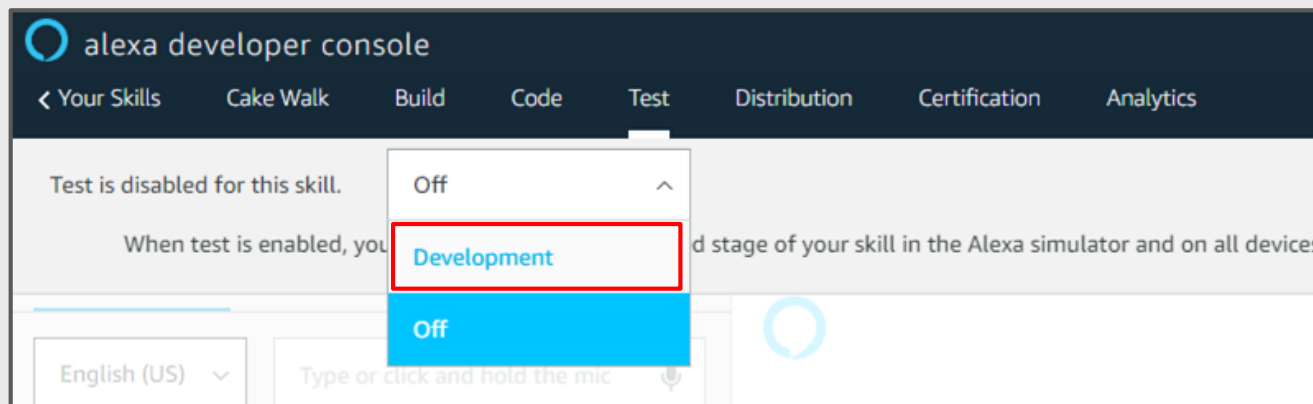
Skill Code

- lambda
- index.js
- package.json
- util.js

```
index.js x
1 // This sample demonstrates handling intents from an Alexa skill using the Alexa Skills Kit SDK (v2).
2 // Please visit https://alexa.design/cookbook for additional examples on implementing slots, dialog management,
3 // session persistence, api calls, and more.
4 const Alexa = require('ask-sdk-core');
5
6 const LaunchRequestHandler = {
7   canHandle(handlerInput) {
8     return handlerInput.requestEnvelope.request.type === 'LaunchRequest';
```

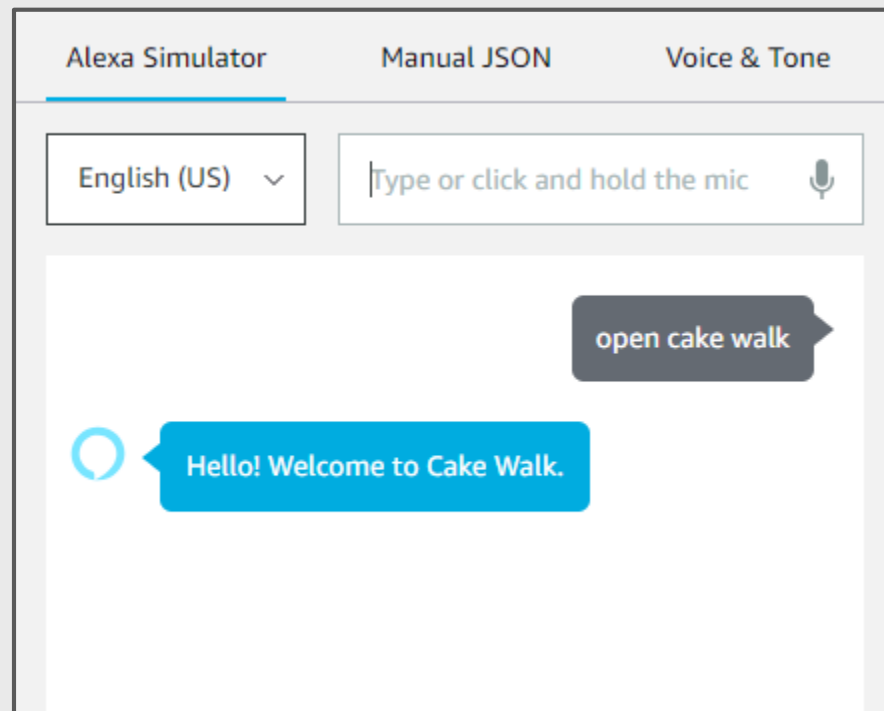
## Test Your Skill

5. Select the **Test** tab at the top of the page.
6. Enable testing by clicking **Development**.
7. Allow microphone access if prompted.



## Test Your Skill

8. You can speak to your Skill or use the type in the text text field. Say "Open Cake Walk."



**Great! Now you have created and tested a simple skill.**

**Let's improve this now.**

# Steps to Build Your Skill

Now that your skill is working, we're going to add the following functionality:

1. Ask the User a question.
2. Listen for the answer.
3. Respond.



# Customizing Your Skill

We will now ask the user for their birthday when the app launches.

1. Return to the Code tab.
2. In **index.js**, edit the `speechText` variable to ask for the user's birthday.

```
6 const LaunchRequestHandler = {
7   canHandle(handlerInput) {
8     return handlerInput.requestEnvelope.request.type === 'LaunchRequest';
9   },
10  handle(handlerInput) {
11    const speechText = 'Welcome to Cake Walk! When is your birthday?';
12    return handlerInput.responseBuilder
13      .speak(speechText)
14      // .reprompt(speechText)
15      .getResponse();
16  }
17 };
```

# Customizing Your Skill

3. Now we want to allow the Skill to re-prompt the user. Remove the slashes you added before.

```
6 const LaunchRequestHandler = {
7   canHandle(handlerInput) {
8     return handlerInput.requestEnvelope.request.type === 'LaunchRequest';
9   },
10  handle(handlerInput) {
11    const speechText = 'Welcome to Cake Walk! When is your birthday?';
12    return handlerInput.responseBuilder
13      .speak(speechText)
14      .reprompt(speechText)
15      .getResponse();
16  }
17 };
```

## Customizing Your Skill

4. The best practice is to keep the speech and reprompt text different, so now we're going to add a variable for the reprompt text. Add Line 12 to your code.
5. Then, change the variable name on Line 15 to **repromptText**.

```
6 const LaunchRequestHandler = {
7   canHandle(handlerInput) {
8     return handlerInput.requestEnvelope.request.type === 'LaunchRequest';
9   },
10  handle(handlerInput) {
11    const speechText = 'Welcome to Cake Walk! When is your birthday?';
12    const repromptText = "Sorry, I didn't catch that. When is your birthday?";
13    return handlerInput.responseBuilder
14      .speak(speechText)
15      .reprompt(repromptText)
16      .getResponse();
17  }
18 };
```



# Save And Deploy

6. Save your Skill.

7. Then Deploy it so that we can test it!

The screenshot shows the Alexa Developer Console interface. At the top, the 'Code' tab is selected. Below the navigation bar, there are three buttons: 'Save', 'Deploy', and 'Promote to live'. The 'Save' and 'Deploy' buttons are highlighted with red rectangles. Below these buttons, a warning message reads 'Deploy before testing your skill'. On the left, the 'Skill Code' section shows a file tree with 'index.js' selected. The code editor displays the following JavaScript code:

```
index.js x
1 // This sample demonstrates handling intents from an Alexa skill using the Alexa Skills Kit SDK (v2).
2 // Please visit https://alexa.design/cookbook for additional examples on implementing slots, dialog management,
3 // session persistence, api calls, and more.
4 const Alexa = require('ask-sdk-core');
5
6 const LaunchRequestHandler = {
7   canHandle(handlerInput) {
8     return handlerInput.requestEnvelope.request.type === 'LaunchRequest';
```

**Great! You told Alexa to ask for input.  
But how are you going to save it and  
respond to it?**

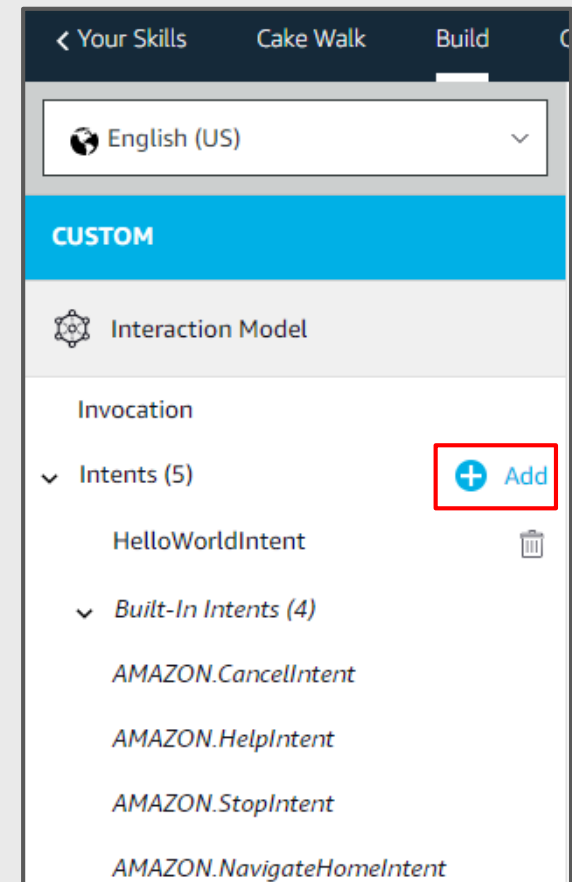
# Create An Intent

We need to create an intent!

1. Return to the **Build** tab.
2. You should be able to see a list of **Intents** on the left hand side.
3. Click on the **Add** button to create a new intent.

## Key Term

**Intent**: an action to fulfill a user's request



# Create An Intent

4. Name the intent **CaptureBirthdayIntent**, and click on **Create custom intent**.

## Add Intent

An intent represents an action that fulfills a user's spoken request. [Learn more](#) about intents.

☒ Create custom intent ?

Create custom intent

## Customize your Utterances

5. In the Sample Utterances field, type I was born on September Sixth Two Thousand Fourteen.
6. Add a few other ways someone might tell Alexa their birthday, such as, "My birthday is October Thirteenth Nineteen Eighty-Nine" or simply, "March Fourth."

[Intents](#) / [CaptureBirthdayIntent](#)

Sample Utterances (0) ?

[Bulk Edit](#) [Export](#)

I was born on September sixth two thousand thirteen

+

# Add Slots

We need Alexa to collect 3 pieces of information - the date, the month and the year of the user's birthday. These are called **slots**.

## Key Term

**Slot:** a piece of information Alexa can capture.

1. Scroll down to the Intent slots section.
2. Name the slot **month** and click the **+** button.




### Intent Slots (0) ?

ORDER ?	NAME ?	SLOT TYPE ?	ACTIONS
1	month	+ <input type="text" value="Select a slot type"/>	Edit Dialog   Delete

## Add Slots

3. Add slots for day and year as well.
4. SLOT TYPE for Month is amazon.month
5. Date is amazon.ordinal. Year is amazon.fourdigit
6. Save.

### Intent Slots (3) ?

ORDER ?	NAME ?	SLOT TYPE ?	ACTIONS
1	 month	AMAZON.Month	<a href="#">Edit Dialog</a>   <a href="#">Delete</a>
2	 day	AMAZON.Ordinal	<a href="#">Edit Dialog</a>   <a href="#">Delete</a>
3	 year	AMAZON.FOUR_DIGIT_NUMBER	<a href="#">Edit Dialog</a>   <a href="#">Delete</a>

# Add More Sample Utterances

In the sample utterances, we can replace exact words with the slot variables. For example,

**I was born on September sixth two thousand thirteen**  
can be written as

**I was born on {month} sixth two thousand thirteen**

Add the following sample utterances to the list as well:

Sample Utterances (9) ⓘ

Bulk Edit

Export

What might a user say to invoke this intent?

+

I was born in {month} {year}

⌵

I was born on {month} {day} {year}

⌵

I was born on {month} {day}

⌵

{month} {year}

⌵

{month} {day} {year}

⌵

< 1 – 5 of 9 > Show All



**Now Alexa has a way to ask for input and somewhere to save it. But what if it goes wrong?**

# Add Dialog Management

- What if the user does not give out all the information in one go? What if he gives only the month or date?
- Dialog management can help us get the information to fill the missing slots.
- If you need a given value from the user, you can designate a slot as required and Alexa will ensure that the value of the slot is collected.

# Add Dialog Management

1. Go back to the Intent slots section. Click on the **Edit Dialog** option next to the month slot.
2. Turn on **Is this slot required to fulfill the intent?**

**Intent Slots (3)** ?

ORDER ?	NAME ?	SLOT TYPE ?	ACTIONS
1	month	AMAZON.Month	<a href="#">Edit Dialog</a> <a href="#">Delete</a>

**Slot Filling**

**Is this slot required to fulfill the intent?** ? ☒

**Alexa speech prompts** ?

What month were you born in? +

# Add Dialog Management

3. Add a question Alexa can use to prompt the user if they don't say the month.
4. Click the + sign to add the prompt.

**Intent Slots (3)** ?

ORDER ?	NAME ?	SLOT TYPE ?	ACTIONS
1	month	AMAZON.Month	<a href="#">Edit Dialog</a>   <a href="#">Delete</a>

**Slot Filling**

Is this slot required to fulfill the intent? ? ☒

Alexa speech prompts ?

What month were you born in?

+


# Add Dialog Management


5. Repeat for day and year.

**Dialogs** Validations

---

## Slot Filling

Is this slot required to fulfill the intent? 

Alexa speech prompts 

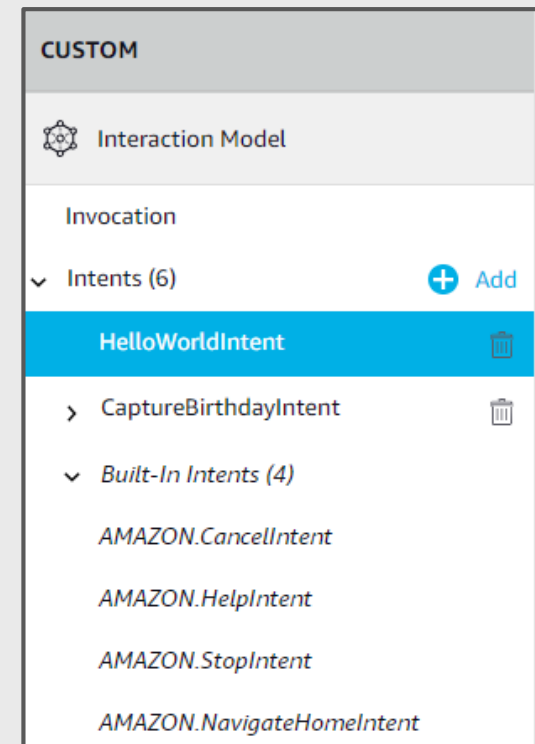
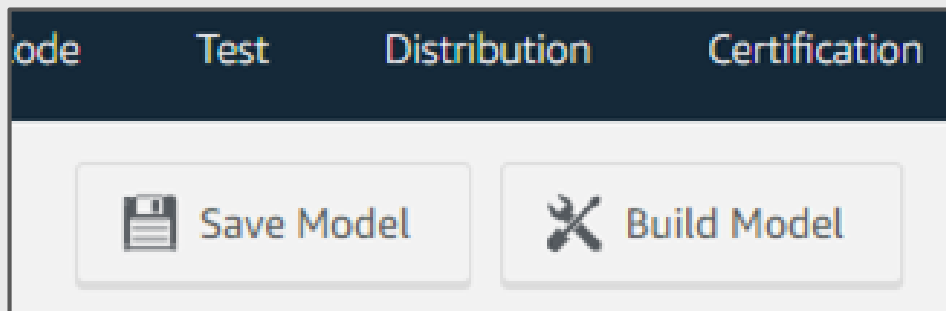
What day were you born in?

+

**Now Alexa can use the user for their birth month, day, and year if any of them is missing.**

# Build the Model

1. You will see a **HelloWorldIntent** above the **CaptureBirthdayIntent**. Delete the **HelloWorldIntent**.
2. Do NOT delete the built-in intents.
3. Save and build the model.



# Update the Backend

1. Go back to the code editor.
2. If you look at the code, you will see a **HelloWorldIntentHandler**, but we deleted the **HelloWorldIntent**. We will modify the **HelloWorldIntentHandler** code we require.

```
19 const HelloWorldIntentHandler = {
20   canHandle(handlerInput) {
21     return handlerInput.requestEnvelope.request.type === 'IntentRequest'
22       && handlerInput.requestEnvelope.request.intent.name === 'HelloWorldIntent';
23   },
24   handle(handlerInput) {
25     const speechText = 'Hello World!';
26     return handlerInput.responseBuilder
27       .speak(speechText)
28       .reprompt('add a reprompt if you want to keep the session open for the user to respond')
29       .getResponse();
30   }
31 };
```



## Update the Backend

3. In line 19, replace **HelloWorldIntentHandler** with **CaptureBirthdayIntentHandler**.
4. In line 22, replace the **HelloWorldIntent** with **CaptureBirthdayIntent**.

```
19 const CaptureBirthdayIntentHandler = {  
20     canHandle(handlerInput) {  
21         return handlerInput.requestEnvelope.request.type === 'IntentRequest'  
22         && handlerInput.requestEnvelope.request.intent.name === 'CaptureBirthdayIntent';  
23     },  
24 }
```

## Update the Backend

5. Add lines 25, 26, and 27 to your code. Note that if the names of your slots are not exactly month, day, and year, this code won't work properly.

```
19 const CaptureBirthdayIntentHandler = {
20   canHandle(handlerInput) {
21     return handlerInput.requestEnvelope.request.type === 'IntentRequest'
22       && handlerInput.requestEnvelope.request.intent.name === 'CaptureBirthdayIntent';
23   },
24   handle(handlerInput) {
25     const year = handlerInput.requestEnvelope.request.intent.slots.year.value;
26     const month = handlerInput.requestEnvelope.request.intent.slots.month.value;
27     const day = handlerInput.requestEnvelope.request.intent.slots.day.value;
28     const speechText = 'Hello World!';
29     return handlerInput.responseBuilder
30       .speak(speechText)
```

## Update the Backend

6. Update the value of `speechText` on or around Line 28 to match what you see below.

Note that we have changed the single quotation marks (") to be backticks (` `) instead.

```
const day = handlerInput.requestEnvelope.request.intent.slots.day.value;  
const speechText = `Thanks, I'll remember that your birthday is ${month} ${day} ${year}.`;   
return handlerInput.responseBuilder  
    .speak(speechText)
```

# Update the Backend


The final **handle()** method should look like this.

```
19 const CaptureBirthdayIntentHandler = {
20   canHandle(handlerInput) {
21     return handlerInput.requestEnvelope.request.type === 'IntentRequest'
22       && handlerInput.requestEnvelope.request.intent.name === 'CaptureBirthdayIntent';
23   },
24   handle(handlerInput) {
25     const year = handlerInput.requestEnvelope.request.intent.slots.year.value;
26     const month = handlerInput.requestEnvelope.request.intent.slots.month.value;
27     const day = handlerInput.requestEnvelope.request.intent.slots.day.value;
28     const speechText = `Thanks, I'll remember that your birthday is ${month} ${day} ${year}.`;
29     return handlerInput.responseBuilder
30       .speak(speechText)
31       // .reprompt('add a reprompt if you want to keep the session open for the user to respond')
32       .getResponse();
33   }
34 };
```

## Update the Backend

7. Finally, scroll to the bottom of your code and update the handler exports with the new handler you created by replacing **HelloWorldIntentHandler** with our new **CaptureBirthdayIntentHandler**.

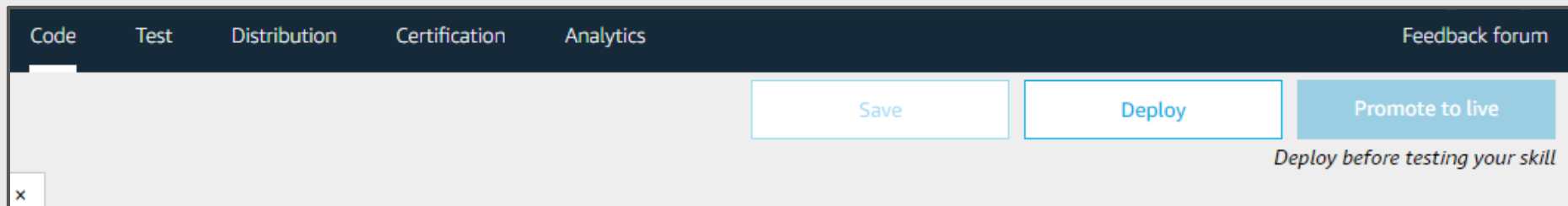
```
exports.handler = Alexa.SkillBuilders
    .addRequestHandlers(
        LaunchRequestHandler,
        HelloWorldIntentHandler,
        HelpIntentHandler,
        CancelAndStopIntentHandler,
        SessionEndedRequestHandler,
        IntentReflectorHandler) // make sure you include the
custom intent handlers
    .addErrorHandlers(
        ErrorHandler)
    .lambda();
```



```
exports.handler = Alexa.SkillBuilders
    .addRequestHandlers(
        LaunchRequestHandler,
        CaptureBirthdayIntentHandler,
        HelpIntentHandler,
        CancelAndStopIntentHandler,
        SessionEndedRequestHandler,
        IntentReflectorHandler) // make sure you include the
custom intent handlers
    .addErrorHandlers(
        ErrorHandler)
    .lambda();
```

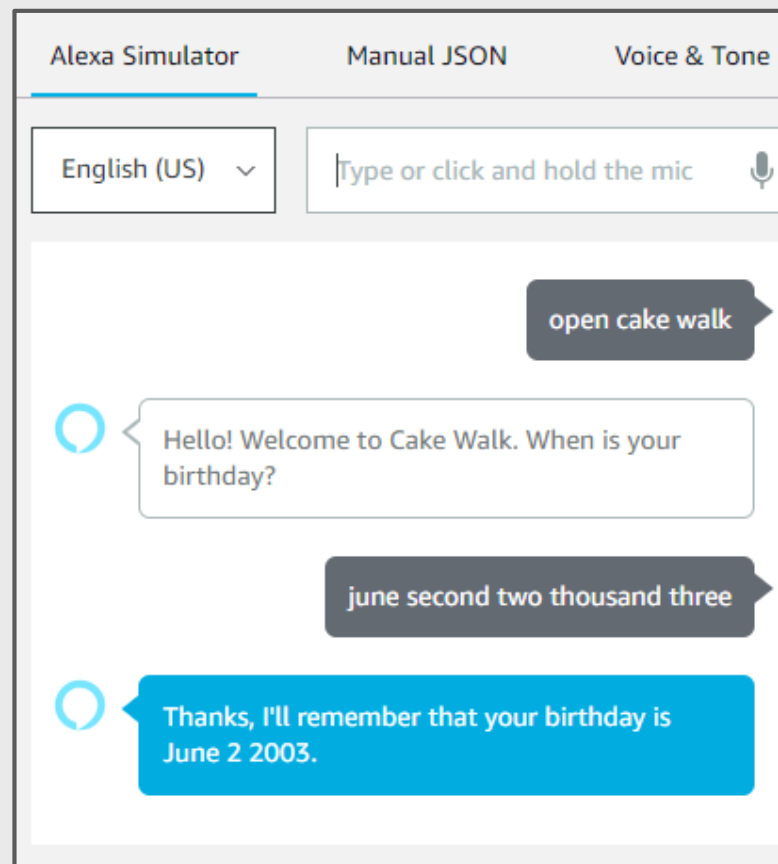
# Save and Deploy

Once again, save and deploy the code.



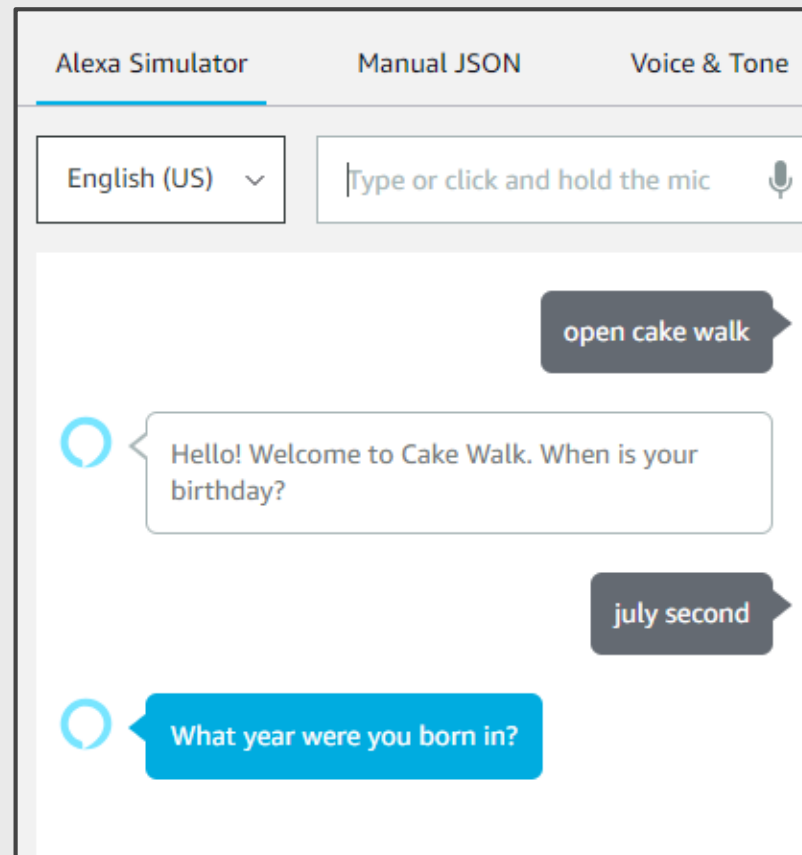
# Test Your Skill

Go back to the test tab to test your skill.



# Test Your Skill

What happens if you do not specify the month or date or year?





# Troubleshooting Your Skill

If you aren't getting a valid response, check the following:


1. Are you calling the right invocation name?
2. Are you saying launch, start or open?
3. Are you sure you have no other skills in your accounts with the same invocation name?
4. Did you Save and Deploy your changes?



## *Let's recap quickly...*

- 1 Voice User Interfaces allow us to physically separate ourselves from devices.
- 2 Amazon Alexa makes it easy for you to create apps (skills) that utilize Voice User Interfaces.
- 3 Alexa takes care of speech recognition and context so you can focus on making a great app.

# Table of Contents


1. Introduction to Alexa & Voice UIs
2. Developing For Alexa
3. Develop Your Alexa Skill
4. Customize The Skill
5. Test The Skill
6. Publish Your Skill
-  7. Review & Quiz
8. Next Steps

# What did you learn today?

We created a fun quiz to test your knowledge and see what you learned from this workshop.

**<http://mlhlocal.host/quiz>**

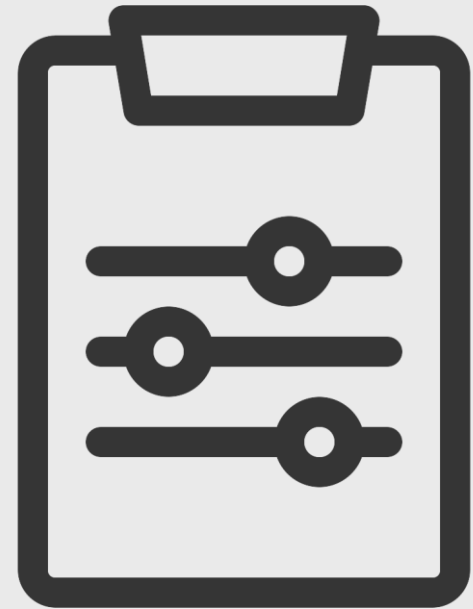
# Table of Contents

1. Introduction to Alexa & Voice UIs
2. Developing For Alexa
3. Develop Your Alexa Skill
4. Customize The Skill
5. Test The Skill
6. Publish Your Skill
7. Review & Quiz
-  8. Next Steps

## Next Steps: Store the user's birthday

Alexa also provides you with S3 storage that lets you persist data. See how you can leverage that to remember the user's birthday so that you do not ask them everytime.

1. Read up about S3 storage and how to use them in the Alexa Code Editor.
2. Retrieve the user's birthday, if it exists, on launch, so that you can decide whether to ask the question or not.



# Learning shouldn't stop when the workshop ends...

**Check your email for access to:**



- These workshop slides
- Practice problems to keep learning
- Deeper dives into key topics
- Instructions to join the community
- More opportunities from MLH!



Workshop

# Build Your First Skill for Amazon Alexa

---

**MLH** **localhost**

 **amazon alexa**