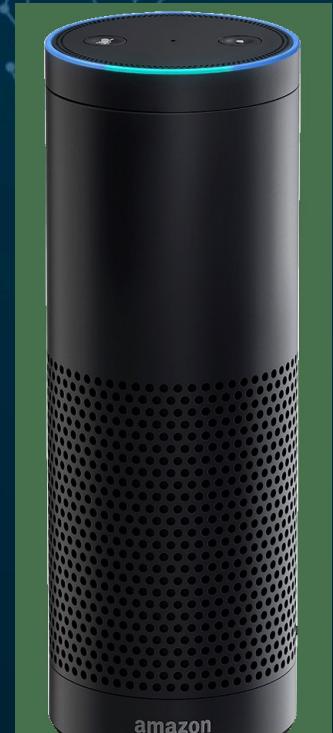




Alexa Skills- Towards Building a Customized Personal Assistant

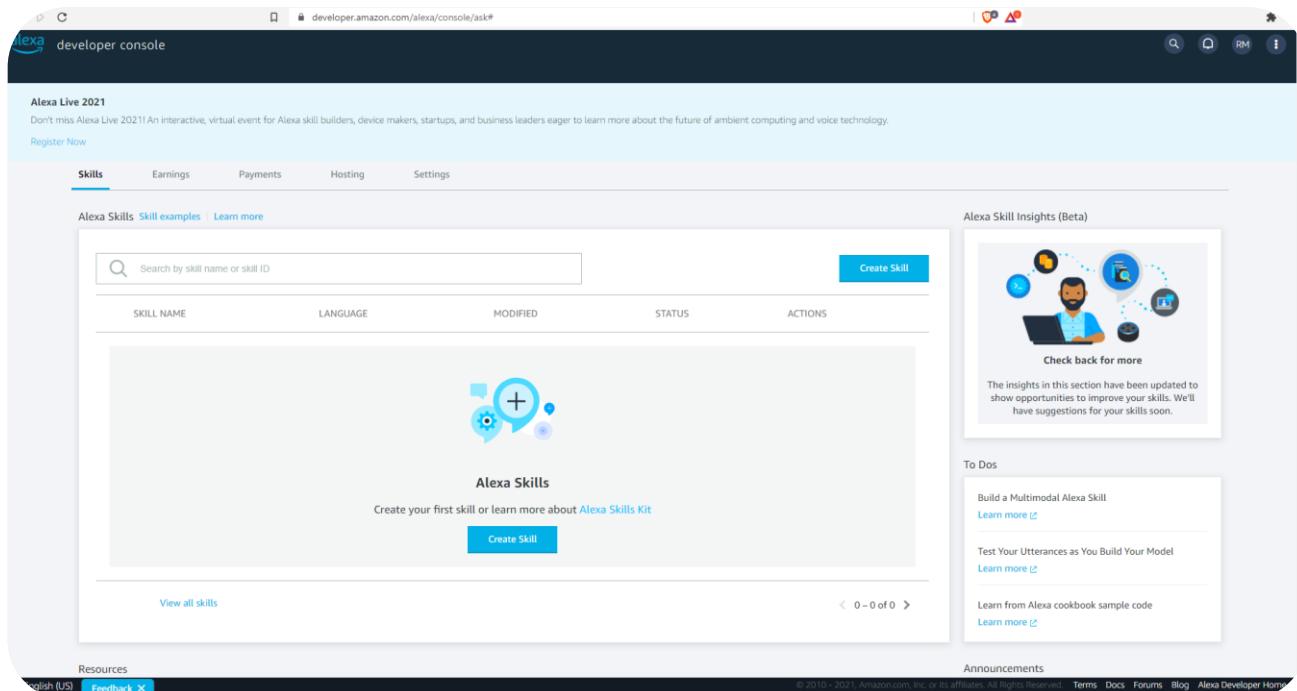
BY:

- REDA MASTOURI



Getting Started

- ▶ Creating a developer account on Alexa Developer Console Portal:
 - ▶ <https://developer.amazon.com/alexasdk/ask#>
 - ▶ - Skill Configuration
 - ▶ - Intents
 - ▶ - Slots
 - ▶ - Utterances
- ▶ Skill Testing:
 - ▶ - Lambda Test Events
 - ▶ - Developer Portal Skill Simulator
 - ▶ - Alexa App
 - ▶ - Amazon Echo -Hardware



What is Alexa Skills?

- ▶ Alexa is a voice service from Amazon.
- ▶ Alexa powers devices like Echo Show, Echo Spot, Echo Dot etc.
- ▶ The most important feature of the Alexa is off course it's hands-free voice control, which lets users ask questions without touching the device.
- ▶ Alexa listen to the voice commands and respond with appropriate responses to fulfill.
- ▶ Alexa comes with many capabilities like playing music, reading the news, creating your shopping list, setting alarms, know about the weather etc.
- ▶ Alexa comes with many capabilities like playing music, reading the news, creating your shopping list, setting alarms, know about the weather etc.



Alexa Features



Smart home

Simplify your life by automating and securing your home with smart devices.



Productivity

Alarms, timers, calendars, email Alexa helps with the details so you can stay focused.



Shopping

Use your voice to manage your shopping list, reorder household items, and track orders.



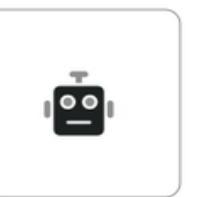
Entertainment

Listen to music, radio, podcasts, and audiobooks or watch movies, TV, and videos.



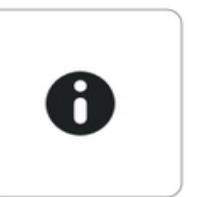
Care Hub

This free feature in the Alexa app gives you new ways to check in on loved ones.



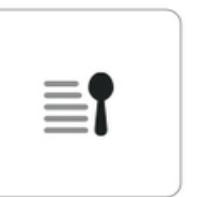
Kids and Family

Enable parental controls and discover features that help kids learn and grow.



Information

Ask Alexa questions, get language translations, general knowledge and more.



Kitchen

Cook along with recipes, get unit conversions, set timers, and multitask with ease.



Communications

Communicate with friends and family with voice or video calling, Drop In and Announcements.



News

Alexa can give you news when you need it from your favorite providers.



Routines

Set up custom shortcuts and make multiple things happen at the same time.



Fun and games

Download new Alexa game skills to play by yourself or with family and friends.



Multi-room audio

See how Alexa can bring music and entertainment to your entire home.



Bedroom

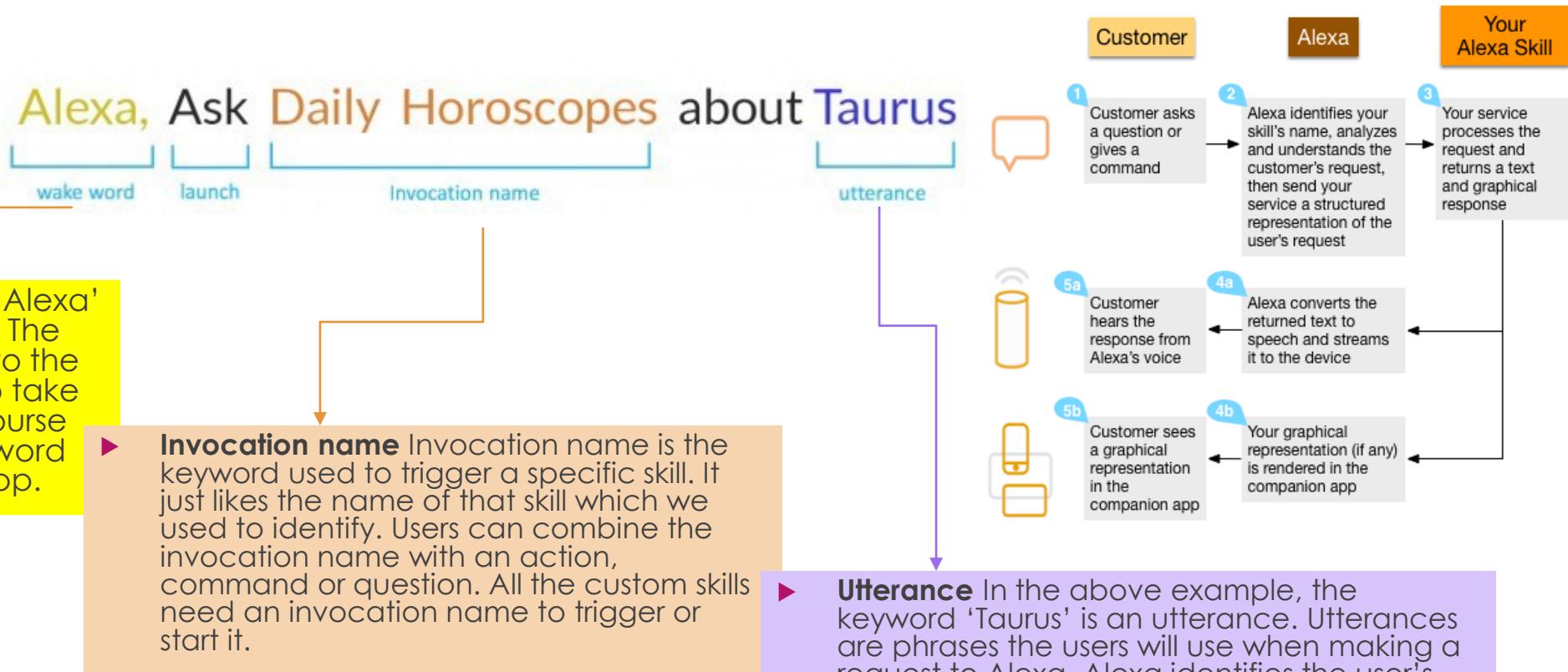
Alexa helps you rest easy and always wake up on the right side of the bed.



Photos

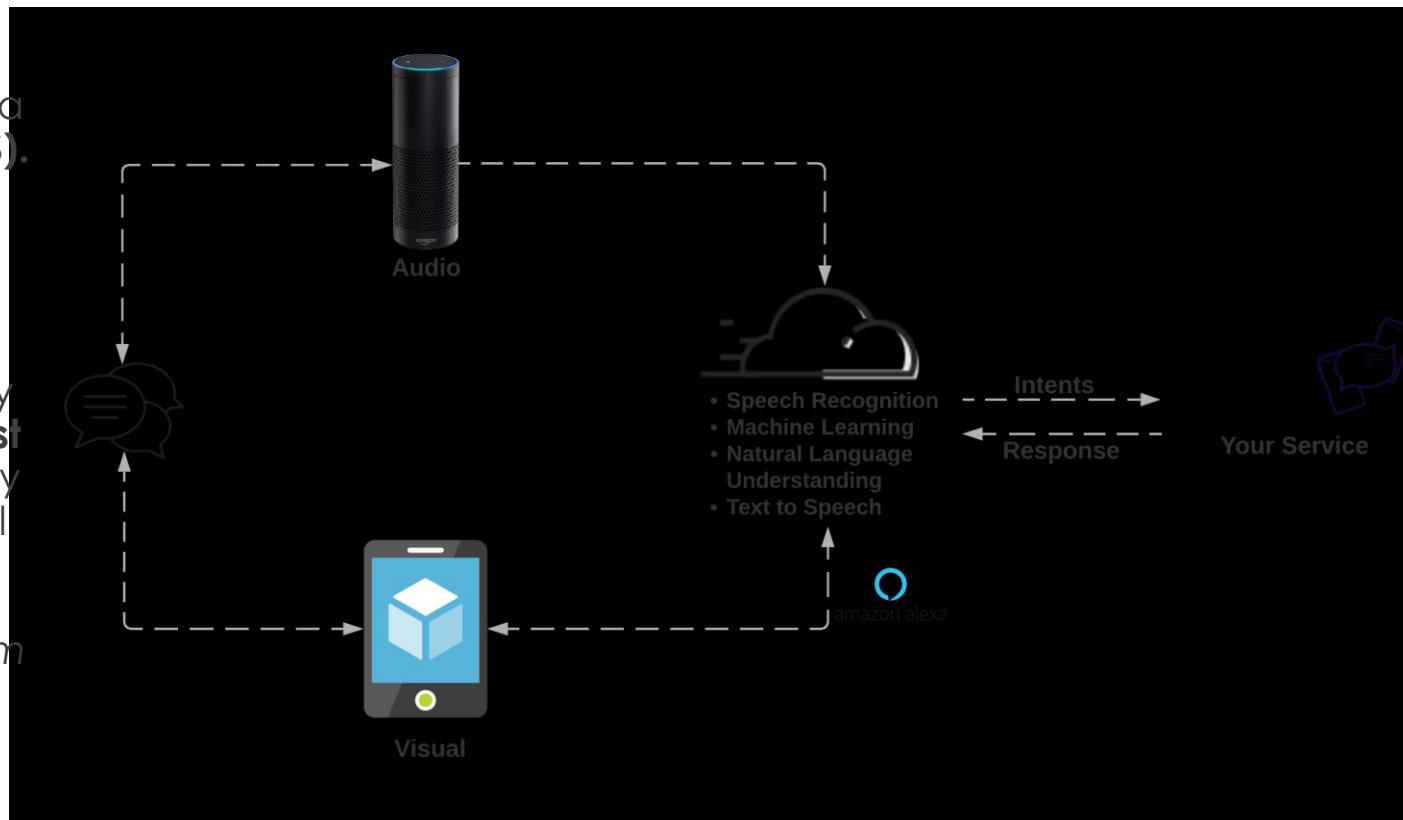
Showcase your best photos, share favorites with your Alexa contacts, and even take pictures.

How Alexa Skills Work? (Part I)



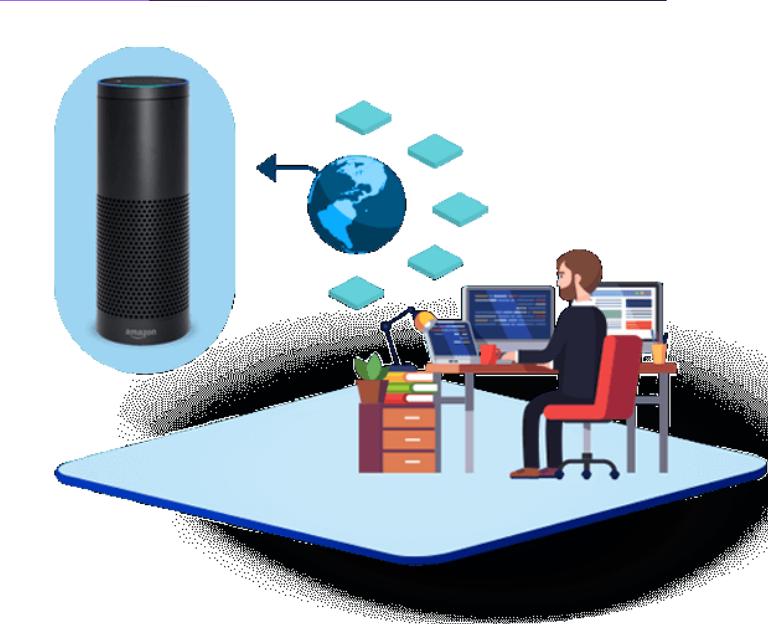
How Alexa Skills Work? (Part II)

- ▶ Alexa enabled devices sends the user's instruction to a cloud-based service called **Alexa Voice Service (AVS)**. Think the Alexa Voice Service as the brain of Alexa enabled devices and perform all the complex operations such as **Automatic Speech Recognition (ASR)** and **Natural Language Understanding (NLU)**.
- ▶ Alexa Voice Service process the response and identify the user's **intent**, then it makes the web service **request** to third party server if needed. This request is necessary if the Skills needs to fetch information from an external service.
For example, Flash briefing skills need to interact with news website/server to fetch the latest headlines. From there, the Alexa service sends the response back to the Alexa and read it to the users.

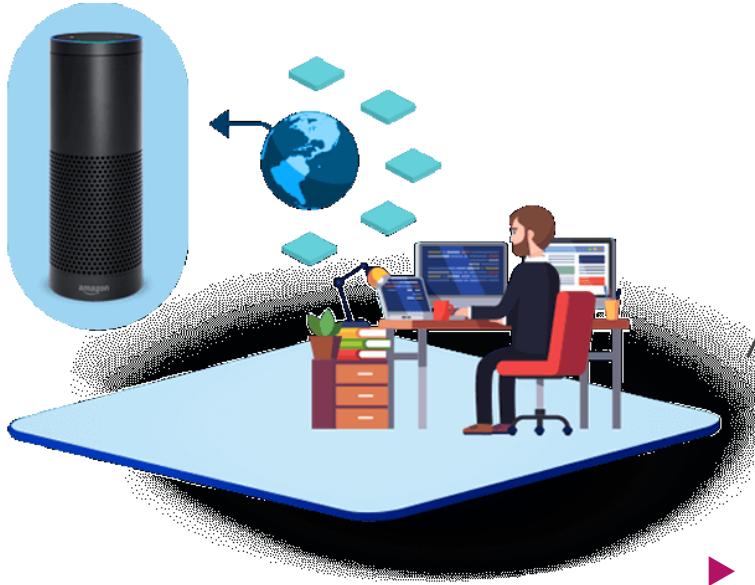


How does FD benefit from developing an Alexa Skill? (Part I)

- ▶ Digital assistants are quickly becoming a part of our lives. Amazon Alexa, Apple's HomePod, Google Assistant, Microsoft Cortana, Samsung's Bixby, and China's Chumenwenwen and Xiaoice are just a few to mention.
- ▶ Virtual assistants are fast becoming commonplace in the home through products like these, with more than 100 million Alexa-enabled devices having been sold till date.
- ▶ Amazon has extended Alexa's API so developers can build Alexa Skills, which are essentially third-party apps that take advantage of Alexa's amazing capabilities.



How does FD benefit from developing an Alexa Skill? (Part II)



- ▶ The Alexa Skills marketplace is rapidly growing, from 1,000 to 15,000 in just one year in the Alexa Skill Store, part of the reason that analysts estimate Alexa to become a \$10 billion industry by 2020. For businesses, it's time to get on board now to build with Alexa as it grows.
- ▶ And there will be fight over talent skilled in Alexa Skills and other voice-assistant technologies.
- ▶ With the expected meteoric growth, voice could become the primary way people interact with the internet in as little as 10–15 years.
- ▶ The unprecedented opportunities are appealing not only to the owners of smart devices.
- ▶ Having taken residence in millions of homes, Amazon and Google are paving the way for other brands that offer entertainment, information, goods, and services.

How does FD benefit from developing an Alexa Skill? (Part III)

- ▶ The first advantage is the vast market. Amazon Echo family commands almost 72% market share of current U.S. smart speakers. Loyal users will likely fill their houses with compatible devices. With a legion of connected fridges, door locks, thermostats, lamps, and other appliances that may use Alexa Voice Service API, the potential of Alexa brand market penetration is infinite.
- ▶ Your investment in the technology is safe. With Alexa controlling such a huge percentage of the marketplace already, it's likely to maintain that lead for some time. Choosing to develop an Alexa Skills (rather than for another voice platform) simplifies development from both the product strategy and technical development standpoints. Keeping things simple at first means you don't need to develop for 3-4 platforms at one time. You can develop for Siri, Cortana and the Google Assistant later as necessitated by the market.
- ▶ Promoting more efficient sorting and accurate matching in the marketplace, AI platforms will be simultaneously winning people's trust and loyalty.
- ▶ At the moment, Amazon has tremendous incentive to help companies succeed with Alexa Skills and offers strong customer support for Skills development. This level of support will be likely to dissipate as the market grows.
- ▶ The Alexa Skills platform will grow more complex. As the technology matures, so will expectations for features and functionality. Getting in early and developing a basic Skill gives you the opportunity to learn without much consequence and iterate as you mature. Right now, you're not expected to have a completely mature Skill. There's time to evolve as Alexa's technology evolves. This will change, just as it did for mobile apps.
- ▶ With voice being relatively new, it's an incredible opportunity to beat your competitors to market with a compelling consumer experience that can differentiate you in your industry. Getting ahead of the curve means you won't be scrambling to figure things out after your competitors have already done so.
- ▶ As an added bonus, the companies that are first to market will be acknowledged as innovators and may receive a substantial amount of positive media attention.



Demo: Proof-of-Concept



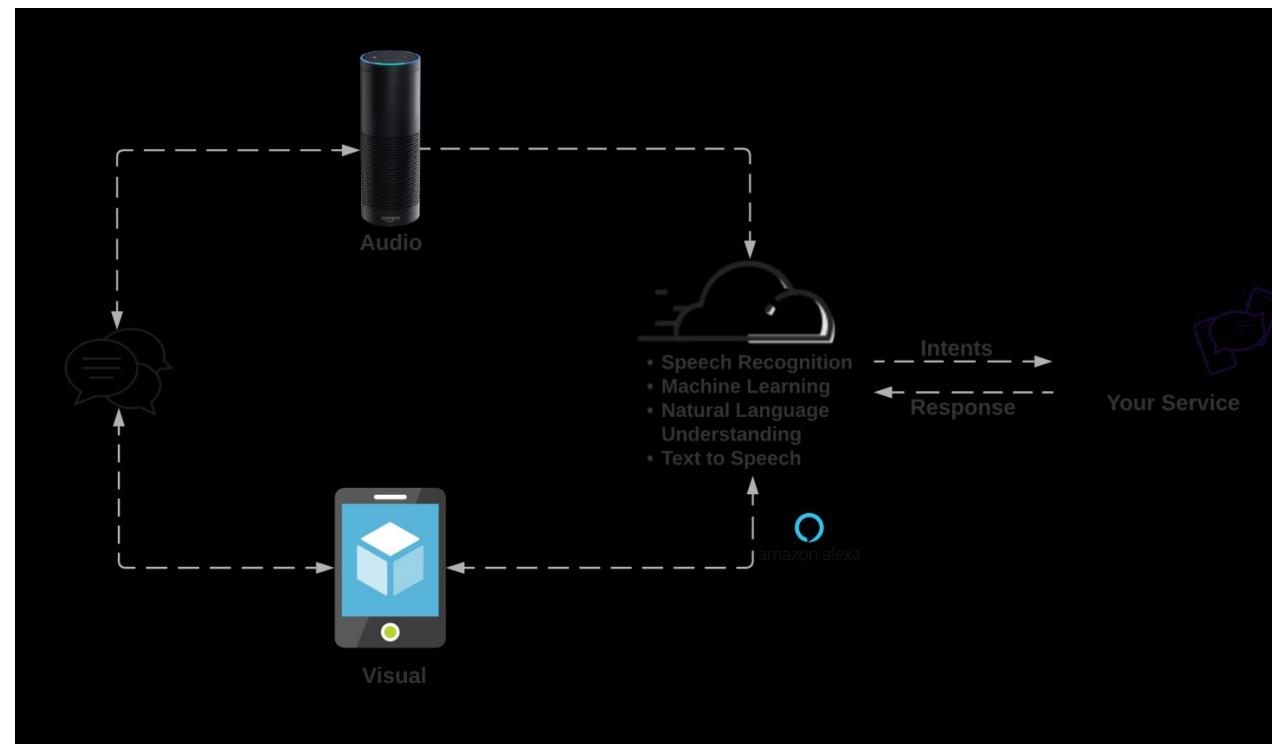
10
Fulcrum Digital



Starting to build a Schedulify Skill: Checklist



- ▶ Personal Assistant Bot:
 - ▶ Check financial portfolio
 - ▶ Manage servers
 - ▶ Send voice messages
 - ▶ Read out emails
 - ▶ Tell monthly goals
- ▶ Build a publicly accessible Alexa Custom Skill & Monetize:
 - ▶ Schedulify



Starting to build a Schedulify Skill

Example of available skills



Editor's Picks

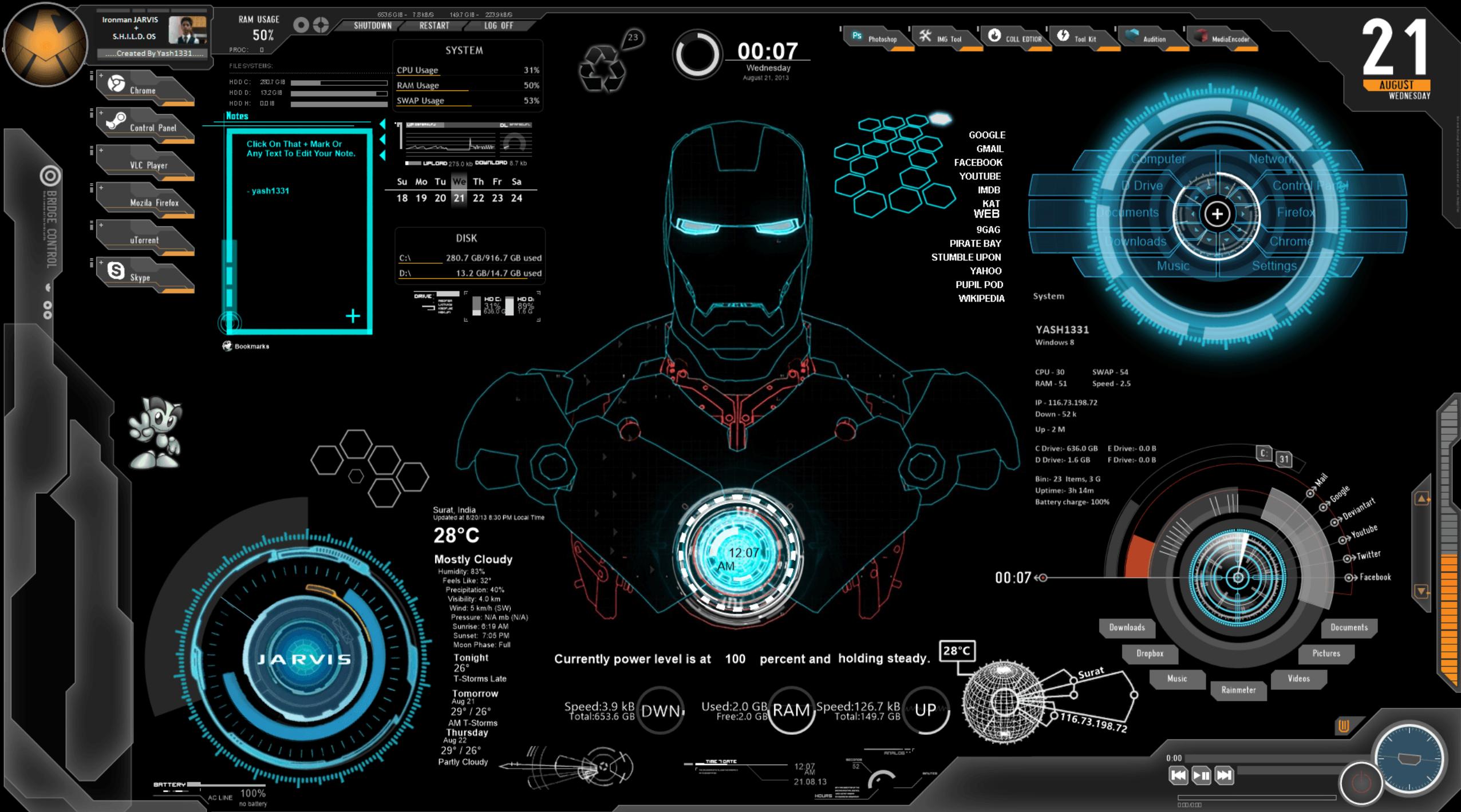
"Alexa open HBO max recommends"	"Alexa, start SpongeBob Challenge"	"Alexa, turn on Xbox"	"Alexa, introduce me to Samuel L. Jackson"	"Alexa, open Food Network Kitchen"	"Alexa, open Headspace"	"Alexa, Inspire Me"	
HBO Max Recommends	The SpongeBob Challenge	Xbox	Samuel L. Jackson--celebrity personality for Alexa	Food Network Kitchen	Headspace: Guided Meditation for Everybody	Inspire Me	Disney
★★★★★ 13	★★★★★ 8,555	★★★★★ 29,551	★★★★★ 80,840	★★★★★ 136	★★★★★ 1,161	★★★★★ 1,474	★★★★★

"Alexa, recommend some skills."

"Alexa open Buddha Says"	"Alexa, ask hi there alien"	"Alexa open quran radio"	"Alexa, start arabian fm"	"Alexa, open Beautiful Adhan."	"Alexa, open Prayer Times"	"Alexa, open azan prayer"	
Buddha Says	Hi there Alien!	Quran Radio 247	Arabian FM	Beautiful Azan	Prayer Times	Azan (APL Enabled)	holy q
★★★★★ 84	★★★★★ 9	★★★★★ 17	★★★★★ 7	★★★★★ 59	★★★★★ 70	★★★★★ 18	★★★★★

Premium Skills

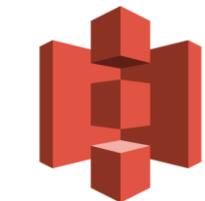
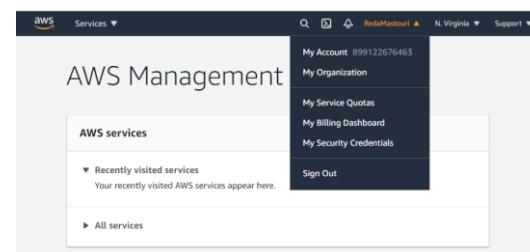
"Alexa, launch My Loft"	"Alexa, Open Trivia Battle"	"Alexa, open Sleep Sounds"	"Alexa, play true or false"	"Alexa, ask Question of the Day."	"Alexa, start 7 Minute Workout."	"Alexa, Open one two three"	
My Loft	Trivia Battle	Sleep Sounds	True or False	Question of the Day	7-Minute Workout	1-2-3 Math	Knight
★★★★★ 23	★★★★★ 3,208	★★★★★ 5,157	★★★★★ 1,082	★★★★★ 22,809	★★★★★ 4,425	★★★★★ 2,105	★★★★★



Working Environment: Cloud Computing



Amazon CloudWatch



Amazon S3





JavaScript Basics: Source-Code

- ▶ <https://github.com/technotablet/alexa-course-custom-skill.git>
- ▶ GitHub Repository <https://github.com/technotablet/alexa-course-custom-skill>
- ▶ Checklist – Interactive, Web Based <https://vivekk.com/teach-alexa-cs>
- ▶ Amazon S3 REST API Introduction <http://docs.aws.amazon.com/AmazonS3/latest/API/>
- ▶ Alexa Built-in Slot Type Reference <https://developer.amazon.com/docs/custom-skills/slot-type-reference.html>
- ▶ Alexa Built-in Intents <https://developer.amazon.com/docs/custom-skills/standard-built-in-intents.html>
- ▶ Alexa Skills Kit (ASK) for Node.js (:tell, :ask, :tellWithCard etc. Response Methods)
<https://github.com/alexa/alex-skills-kit->
- ▶ sdk-for-nodejs Google Maps Directions API Documentation
<https://developers.google.com/maps/documentation/directions/intro>
- ▶ Time zones Syntax https://en.wikipedia.org/wiki/List_of_tz_database_time_zones



JavaScript Basics: JS101- Promises and Await Expressions in Async Functions- 2

- ▶ Changing to asynchronous setup, where function a is executed within an anonymous function through setTimeout with 3000 ms delay

```
>Welcome  promises.html U x
JS_Fund > promises.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Promises-Reda</title>
8  </head>
9  <body>
10     <script>
11
12         var d = new Date(); // current date
13         var n = 3000 + d.getTime(); //Add 3 seconds to current date
14
15         function a() {
16             while (new Date() < n) { } // Do nothing if current time is < advanced time 'n'
17             console.log("Function A completes in " + (new Date() - d) + "ms / " + new Date());
18         }
19
20         function b() {
21             console.log("Function B completes. " + new Date());
22         }
23
24         console.log("Starts at " + d);
25         // Calling the two function a and then b
26         // Changing to asynchronous setup, where function a is executed within an anonymous function through setTimeout with 3000 ms delay
27         setTimeout(function(){
28             a();
29             }, 3000);
30         b();
31
32     </script>
33
34
35
36 </body>
37 </html>
```

The screenshot shows the browser's developer tools open to the 'Console' tab. The output pane displays the following log entries:

- Starts at Tue Aug 24 2021 05:54:35 GMT-0400 (Eastern Daylight Time)
- Function B completes. Tue Aug 24 2021 05:54:36 GMT-0400 (Eastern Daylight Time)
- Function A completes in 3011ms / Tue Aug 24 2021 05:54:39 GMT-0400 (Eastern Daylight Time)

JavaScript Basics:

JS101- Promises and Await Expressions in Async Functions- 2



- ▶ setTimeout() is one of the earliest asynchronous methods in JS.

- ▶ Pros:

- ▶ Execute multiple functions in parallel
- ▶ Non-blocking code

- ▶ Cons:

- ▶ Functions might be dependent on each other
- ▶ One function's output is another function's input

Screenshot of a browser developer tools console tab showing the execution of the code below.

```

    Starts at Tue Aug 24 2021 05:48:18 GMT-0400 (Eastern Daylight Time)
    Function A completes in 3000ms / Tue Aug 24 2021 05:48:21 GMT-0400 (Eastern Daylight Time)
    Function B completes. Tue Aug 24 2021 05:48:21 GMT-0400 (Eastern Daylight Time)
  
```

Screenshot of a code editor showing the source code for promises.html.

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Promises-Reda</title>
</head>
<body>
<script>

var d = new Date();
var n = 3000 + d.getTime(); //Add 3 seconds to current date

function a() {
  while (new Date() < n) { } // Do nothing if current time is < advanced time 'n'
  console.log("Function A completes in " + (new Date() - d) + "ms / " + new Date());
}

function b() {
  console.log("Function B completes. " + new Date());
}

console.log("Starts at " + d);
a();
b();

</script>
</body>
</html>
  
```

JavaScript Basics:

JS101- Promises and Await Expressions in Async Functions- 3



- ▶ Removing this loop: `while (new Date() < n) {} // Do nothing if current time is < advanced time 'n'`
- ▶ and adding an asynchronous method `setTimeout` of 3s and and print a counter as 2

Inspector Console Debugger Network Style Editor Performance Memory

Filter Output Errors Warnings Logs Info Debug

```

Starts at Wed Aug 25 2021 03:07:03 GMT-0400 (Eastern Daylight Time)
Function B completes. Counter Value= 1 , and the date is: Wed Aug 25 2021 03:07:03 GMT-0400 (Eastern Daylight Time)
Function A completes in 3018ms / Wed Aug 25 2021 03:07:06 GMT-0400 (Eastern Daylight Time)

```

```

JS_Fund > promises.html > html > body > script
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Promises-Redux</title>
8  </head>
9  <body>
10 <script>
11
12     var d = new Date(); // current date
13     var n = 3000 + d.getTime(); //Add 3 seconds to current date
14
15     // Introducing a variable called counter
16     // and assign it a value of 1 by default
17     var counter = 1;
18
19     function a() {
20         // Removing this loop: while (new Date() < n) {} // Do nothing if current time is < advanced time 'n'
21         // and adding an asynchronous method setTimeout of 3s and and print a counter as 2
22         setTimeout(function() {
23             console.log("Function A completes in " + (new Date() - d) + "ms / " + new Date());
24             counter = 2;
25         }, 3000);
26     }
27
28     function b() {
29         //
30         console.log("Function B completes. Counter Value= " + counter + " , and the date is: " + new Date());
31     }
32
33     console.log("Starts at " + d);
34     a();
35     b();
36
37
38
39
40
41 </script>
42 </body>
43 </html>

```

JavaScript Basics:

JS101- Promises and Await Expressions in Async Functions- 4



- ▶ A Callback is a function which is passed to another function as an argument
- ▶ E.g. An anonymous function as an argument to setTimeout function
- ▶ It let the code executes in a linear way

```
<script>
    function outerFunction(param) {
        //1 - we can pass a function as an argument
        var x = "Some Info";
        param(x);
    }

    function callbackFunction(data) {
        console.log(data);
    }

    outerFunction(callbackFunction);
</script>
```

```
setTimeout(
    function () {
        console.log("It works!");
    },
    3000
);
```

```
2   <html lang="en">
3     <head>
4       <meta charset="UTF-8">
5       <meta http-equiv="X-UA-Compatible" content="IE=edge">
6       <meta name="viewport" content="width=device-width, initial-scale=1.0">
7       <title>Promises-Reda</title>
8   </head>
9   <body>
10
11
12   <script>
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
</script>
</body>
</html>
```

The code illustrates the use of callbacks and promises. It starts with a basic setTimeout example. Then, it moves on to more complex scenarios involving nested functions and callbacks. The code includes comments explaining each step and the flow of execution.

JavaScript Basics:

JS101- Promises and Await Expressions in Async Functions - 5



- ▶ Using asynchronous function in sequence

Screenshot of a browser developer tools console showing the output of the following JavaScript code:

```
Starts at Wed Aug 25 2021 03:32:33 GMT-0400 (Eastern Daylight Time)
Function A completes in 3022ms / Wed Aug 25 2021 03:32:36 GMT-0400 (Eastern Daylight Time)
Function B completes. Counter Value= 2 , and the date is: Wed Aug 25 2021 03:32:36 GMT-0400 (Eastern Daylight Time)
```

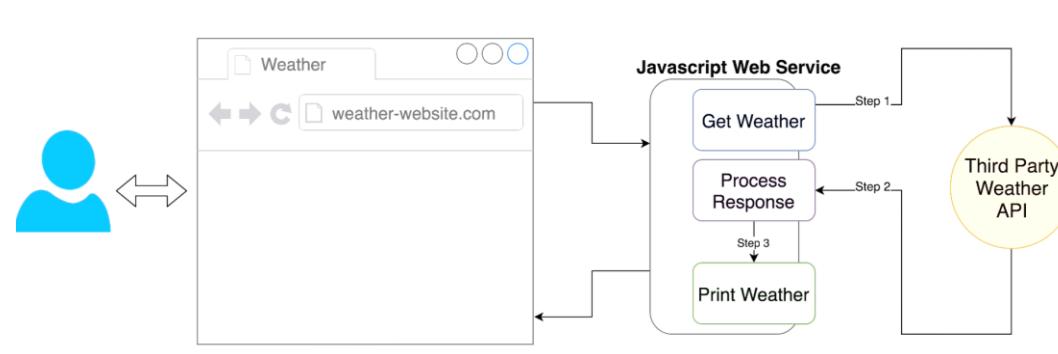
```
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>Promises-Reda</title>
8  </head>
9  <body>
10     <script>
11
12         var d = new Date(); // current date
13         var n = 3000 + d.getTime(); //Add 3 seconds to current date
14
15         // Introducing a variable called counter
16         // and assign it a value of 1 by default
17         var counter = 1;
18
19         function a(newfunc) {
20             setTimeout(function() {
21                 console.log("Function A completes in " + (new Date() - d) + "ms / " + new Date());
22                 counter = 2;
23                 newfunc();
24             }, 3000);
25
26         }
27
28         function b() {
29             //
30             console.log("Function B completes. Counter Value= " + counter + " , and the date is: " + new Date());
31         }
32
33         console.log("Starts at " + d);
34         // Calling b within a
35         a(b); //Special: Whenever I call function a, I can do some activities, and I can pass any function
36         // as an argument to function a();, and since b() doesn't accept any further argument, I can right like a(b)
37
38     </script>
39  </body>
40 </html>
```



JavaScript Basics: Callback

- ▶ Primarily used in Node.js 6.10 and below for Lambda
- ▶ Establish connections to independent interfaces such as Databases or APIs
- ▶ On response, process further (dependent functions)
 - ▶ If there are a large number of functions, dependent of each other, then we possibly get into something called Callback Hell (Because we need to nest the dependent functions, within a function). It can be very confusing and messy.

```
a(function (resultsFromA) {  
  b(resultsFromA, function (resultsFromB) {  
    c(resultsFromB, function (resultsFromC) {  
      d(resultsFromC, function (resultsFromD) {  
        console.log(resultsFromD);  
      })  
    })  
  })  
});
```



JavaScript Basics: Promises



- ▶ Promise is a most sophisticated solution to implementing asynchronous code in JavaScript
- ▶ It currently provides 3 methods
 - ▶ .then(): handles success and can manage both success and failure scenarios
 - ▶ .catch(): handles failure and meant for errors
 - ▶ .finally(): handles completion and meant for the tasks that should execute no matter what the result is (Executes irrespective of success or error)

```
function testBoolean() {  
  return new Promise(function (resolve, reject) {  
    if (Math.random() >= 0.5) {  
      resolve('success');  
    } else {  
      reject('failed');  
    }  
  });  
  
testBoolean()  
  .then()      // handle success  
  .catch()     // handle failure  
  .finally()   // handle completion;
```



JavaScript Basics: Promises 2

- ▶ Promise is a most sophisticated solution to implementing asynchronous code in JavaScript
- ▶ We can also use a chained promises where next function executes if the previous one was successful.

```

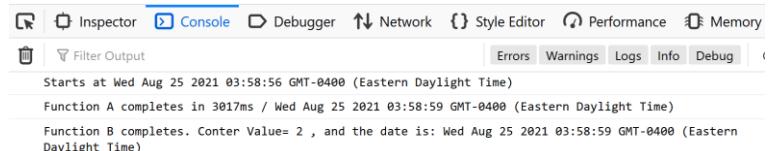
new Promise(function (resolve, reject) {
    setTimeout(() => resolve(1), 1000);
}).then(function (result) {
    return result * 2;
}).then(function (result) {
    return result * 3;
}).then(function (result) {
    return result * 2;
});

```

```

JS_Fund > promises.html > HTML > body > script > a
1  <html lang="en">
2   <head>
3     <meta charset="UTF-8">
4     <meta http-equiv="X-UA-Compatible" content="IE=edge">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Promises-Redux</title>
7   </head>
8   <body>
9     <script>
10
11    var d = new Date(); // current date
12    var n = 3000 + d.getTime(); //Add 3 seconds to current date
13    var counter = 1;
14
15    function a() {
16      // 1 - Let's modify the code while adding a promise
17      let promise = new Promise(function(resolve, reject) {
18
19        setTimeout(function() {
20          console.log("Function A completes in " + (new Date() - d) + "ms / " + new Date());
21          counter = 2;
22          resolve(counter); //2 - add resolve method
23        }, 3000);
24      });
25      // end of the promise
26
27      return promise; //3- the var to return
28    }
29
30    function b() {
31      //
32      // console.log("Function B completes. Counter Value= " + counter + " , and the date is: " + new Date());
33    }
34
35    console.log("Starts at " + d);
36    //here calling out all the promise methods .then(), .catch() and .finally()
37    a().then(b);
38    // in this example we didn't use any reject or error scenario
39
40  </script>
41 </body>
42 </html>

```



JavaScript Basics: Promises 3



- ▶ If we comment resolve, no success will be done by the promise
- ▶ Therefore, function b() won't be executed

Screenshot of a browser developer tools console tab showing the output of a JavaScript script. The output shows two timestamped logs:

```

Starts at Wed Aug 25 2021 04:04:54 GMT-0400 (Eastern Daylight Time)
Function A completes in 3018ms / Wed Aug 25 2021 04:04:57 GMT-0400 (Eastern Daylight Time)
  
```

```

JS_Fund > promises.html > html > body > script > a > promise > <function> > setTimeout() callback
  2   <html lang="en">
  3     <head>
  4       <meta charset="UTF-8">
  5       <meta http-equiv="X-UA-Compatible" content="IE=edge">
  6       <meta name="viewport" content="width=device-width, initial-scale=1.0">
  7       <title>Promises-Reda</title>
  8     </head>
  9     <body>
 10       <script>
 11
 12         var d = new Date(); // current date
 13         var n = 3000 + d.getTime(); //Add 3 seconds to current date
 14         var counter = 1;
 15
 16         function a() {
 17           // 1 - Let's modify the code while adding a promise
 18           let promise = new Promise(function(resolve, reject) {
 19
 20             setTimeout(function() {
 21               console.log("Function A completes in " + (new Date() - d) + "ms / " + new Date());
 22               counter = 2;
 23               // resolve(counter); //if we comment resolve, no success will be done by the promise
 24             }, 3000);
 25           } // end of the promise
 26
 27           return promise; //3- the var to return
 28
 29         }
 30
 31         function b() {
 32           //
 33           console.log("Function B completes. Counter Value= " + counter + " , and the date is: " + new Date());
 34         }
 35
 36         console.log("Starts at " + d);
 37         a().then(b);
 38
 39       </script>
 40     </body>
 41   </html>
  
```

JavaScript Basics: Promises Hell



```
function a(param) {
    return new Promise(function (resolve, reject) {
        resolve(param * 2);
    });
}

function b(param) {
    return new Promise(function (resolve, reject) {
        resolve(param * 3);
    });
}

a(1)
    .then((result) => {
        b(result)
            .then((result) => {
                console.log('Pyramid of Doom!', result);
            });
    });
}
```

JavaScript Basics: Await Expressions in Async Functions

- ▶ Await expression works in an Asynch function
- ▶ Supported in Lambda for Node.js 8.10
- ▶ Cleaner code
 - ▶ Used in Lambda Functions, including Alexa Skills
 - ▶ Makes code easier to debug
 - ▶ Execution in sequence
 - ▶ Reliability



```
JS_Fund > promises.html < html > body > script
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Promises-Redux</title>
8   </head>
9   <body>
10  <script>
11
12  var d = new Date(); // current date
13  var n = 3000 + d.getTime(); //Add 3 seconds to current date
14  var counter = 1;
15
16
17  function a() {
18    // 1 - Let's modify the code while adding a promise
19    let promise = new Promise(function(resolve, reject) {
20
21      setTimeout(function() {
22        console.log("Function A completes in " + (new Date() - d) + "ms / " + new Date());
23        counter = 2;
24        // we will prefix sync to function b() because only asynchronous function can contain await Expr.
25        resolve(counter);
26      }, 3000);
27
28    }
29
30    return promise;
31
32  // async prefix added
33  async function b() {
34    // adding this block
35    var x;
36    try{
37      x = await a();
38      console.log("Function B completes. Counter Value= " + counter + ", and the date is: " + new Date());
39    } catch (error){
40      console.log("Error a sst!");
41    }
42
43    console.log("Starts at " + d);
44    // then call just function b
45    (b);
46
47
48  </script>
49  </body>
50  </html>
```





AWS Lambda Basics

Create Lambda Function

- ▶ S3-get-object from Lambda blueprints

Screenshot of the AWS Lambda 'Create function' wizard. The 'Use a blueprint' option is selected. The 'Blueprints' section shows three results for 'S3': 's3-get-object-python', 'rekognition-python', and 's3-get-object'. The 's3-get-object' blueprint is highlighted with a blue border.

Choose one of the following options to create your function.

- Author from scratch
- Use a blueprint
- Container image
- Browse serverless app repository

Blueprints

- s3-get-object-python
- rekognition-python
- s3-get-object

AWS Lambda Basics

Create Lambda Function



Lambda > Functions > Create function > Configure blueprint s3-get-object

Basic information Info

Function name: takeS3backup

Execution role: Use an existing role (S3LambdaRole)

Existing role: S3LambdaRole

e.g. jpg

Lambda will add the necessary permissions for Amazon S3 to invoke your Lambda function from this trigger. [Learn more](#) about the Lambda permissions model.

Lambda function code

Code is preconfigured by the chosen blueprint. You can configure it after you create the function. [Learn more](#) about deploying Lambda functions.

```

1 // Import the AWS SDK
2 const aws = require('aws-sdk');
3
4 const s3 = new aws.S3({ apiVersion: '2006-03-01' });
5
6
7 exports.handler = async (event, context) => {
8   //console.log('Received event:', JSON.stringify(event, null, 2));
9
10  // Get the object from the event and show its content type
11  const bucket = event.Records[0].s3.bucket.name;
12  const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, ' '));
13  const params = {
14    Bucket: bucket,
15    Key: key,
16  };
17  try {
18    const { ContentType } = await s3.getObject(params).promise();
19    console.log('CONTENT TYPE:', ContentType);
20    return ContentType;
21  } catch (err) {
22    console.log(err);
23    const message = `Error getting object ${key} from bucket ${bucket}. Make sure they`;
24    console.log(message);
25    throw new Error(message);
26  }
27}
28
29

```

Cancel
Create function



AWS Lambda Basics

Create Lambda Function

The screenshot shows the AWS Lambda function editor interface. The top navigation bar includes the AWS logo, 'Services' dropdown, search bar ('Search for services, features, marketplace products, and docs [Alt+S]'), and tabs for 'Test' and 'Deploy'. A green button indicates 'Changes deployed'. The left sidebar shows the 'Environment' section with a 'takeS3backup -' folder containing an 'index.js' file. The main editor area displays the following code:

```
1 console.log('Loading function');
2
3 const aws = require('aws-sdk');
4
5 const s3 = new aws.S3({ apiVersion: '2006-03-01' });
6
7
8 exports.handler = async (event, context) => {
9     //console.log('Received event:', JSON.stringify(event, null, 2));
10
11     // Get the object from the event and show its content type
12     const bucket = event.Records[0].s3.bucket.name;
13     const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, ' '));
14     const params = {
15         Bucket: bucket,
16         Key: key,
17     };
18     try {
19         const { ContentType } = await s3.getObject(params).promise();
20         console.log('CONTENT TYPE:', ContentType);
21         return ContentType;
22     } catch (err) {
23         console.log(err);
24         const message = `Error getting object ${key} from bucket ${bucket}. Make sure they exist and your bucket is in the same region as this function.`;
25         console.log(message);
26         throw new Error(message);
27     }
28 };
29
```



AWS Lambda Basics

Prepare Lambda test Event to Check connectivity

Configure test event

A function can have up to 10 test events. The events are persisted so you can switch to another computer or web browser and test your function with the same events.

- Create new test event
- Edit saved test events

Event template

s3-put

▼

Event name

CheckS3Put

```

10      "principalId": "EXAMPLE"
11    },
12    "requestParameters": {
13      "sourceIPAddress": "127.0.0.1"
14    },
15    "responseElements": {
16      "x-amz-request-id": "EXAMPLE123456789",
17      "x-amz-id-2": "EXAMPLE123/mnoprstuvwxyzABCDEFH"
18    },
19    "s3": {
20      "s3SchemaVersion": "1.0",
21      "configurationId": "testConfigRule",
22      "bucket": {
23        "name": "redaOne",
24        "ownerIdentity": {
25          "principalId": "EXAMPLE"
26        },
27        "arn": "arn:aws:s3:::redaOne"
28      },
29      "object": {
30        "key": "Doesnotexist.html",
31        "size": 1024,
32        "eTag": "0123456789abcdef0123456789abcdef",
33        "sequencer": "0A1B2C3D4E5F678901"
34      }
35    }
36  }
37 ]
38 }
    
```

Cancel
Format JSON
Create



AWS Lambda Basics

Upgrade Lambda Function

Tools Window Test Deploy Changes not deployed

index.js Execution results +

```
3 const aws = require('aws-sdk');
4
5 const s3 = new aws.S3({ apiVersion: '2006-03-01' });
6
7
8 exports.handler = async (event, context) => {
9     //console.log('Received event:', JSON.stringify(event, null, 2));
10
11    // Get the object from the event and show its content type
12    const bucket = event.Records[0].s3.bucket.name;
13    const key = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, ' '));
14    const params = {
15        Bucket: bucket,
16        Key: key,
17    };
18    try {
19        const { ContentType } = await s3.getObject(params).promise();
20        console.log('CONTENT TYPE:', ContentType);
21        // return ContentType; //commenting this no longer needed!
22
23        //target variable where the backup will be stored
24        var targetBucket = bucket+"-backup"; //redone-backup
25        //printing something to the screen saying that file being copied:
26        console.log("Copying files: "+key+" from: "+bucket+" to the following bucket: "+ targetBucket );
27        //Creating a contant which we have to specify under CopySource parameters as dictionary with the URL
28        // of file location
29        // target bucket name
30        // and file name
31        const paramsNew = {
32            CopySource : bucket + "/" + encodeURIComponent(key),
33            Bucket : targetBucket,
34            Key : key
35        };
36
37        // here we create a constant finalOutput where we use await to call S3's copyObject method and pass the parameters for the promise
38        const finalOutput = await s3.copyObject(paramsNew).promise();
39        // printing out the params
40        console.log("File: "+key+" has been successfully copied from "+ bucket +" to "+ targetBucket+ " as expected!");
41        return "File copied successfully!";
42
43    }
44    catch (err) {
45        console.log(err);
46        const message = `Error getting object ${key} from bucket ${bucket}. Make sure they exist and your bucket is in the same region as this function.`;
47        console.log(message);
48        throw new Error(message);
49    }
50};
```

AWS Lambda Basics

Testing & Verification



Tools Window Test Deploy Changes deployed

index.js

```

18 try {
19     const { ContentType } = await s3.getObject(params).promise();
20     console.log('CONTENT TYPE:', ContentType);
21     // return ContentType; //commenting this no longer needed!
22
23     //target variable where the backup will be stored
24     var targetBucket = bucket+"-backup"; //redone-backup
25     //printing something to the screen saying that file being copied:
26     console.log("Copying files: "+key+" from: "+bucket+" to the following bucket: "+ targetBucket );
27     //Creating a contanc which we have to specify under CopySource parameters as dictionary with the URL
28     // of file location
29     // target bucket name |
30     // and file name
31     const paramsNew = {
32         CopySource : bucket + "/" + encodeURIComponent(key),
33         Bucket : targetBucket,
34         Key : key
35     };
36
37     // here we create a constant finalOutput where we use await to call S3's copyObject method and pass the parameters for the promise
38     const finalOutput = await s3.copyObject(paramsNew).promise();
39     // printing out the params
40     console.log("File: "+key+" has been successfully copied from "+ bucket +" to " + targetBucket+ " as expected!");
41     return "File copied successfully!";

```

Execution results

Execution results

Test Event Name: CheckS3Put

Response: "File copied successfully!"

Status: Succeeded

Function Logs

```

START RequestId: df7ee8c6-2e90-461c-a55f-443066f2181d Version: $LATEST
2021-08-26T06:55:05.738Z  df7ee8c6-2e90-461c-a55f-443066f2181d  INFO  CONTENT TYPE: text/html
2021-08-26T06:55:05.738Z  df7ee8c6-2e90-461c-a55f-443066f2181d  INFO  Copying files: first.html from: redaone to the following bucket: redaone-backup
2021-08-26T06:55:06.378Z  df7ee8c6-2e90-461c-a55f-443066f2181d  INFO  File: first.html has been successfully copied from redaone to redaone-backup as expected!
END RequestId: df7ee8c6-2e90-461c-a55f-443066f2181d
REPORT RequestId: df7ee8c6-2e90-461c-a55f-443066f2181d Duration: 1336.81 ms    Billed Duration: 1337 ms    Memory Size: 128 MB Max Memory Used: 90 MB Init Duration: 372.47 ms

```

Request ID: df7ee8c6-2e90-461c-a55f-443066f2181d

Search for services, features, marketplace products, and docs [Alt+S]

We're continuing to improve the S3 console to make it faster and easier to use. If you have feedback on the updated experience, choose Provide feedback.

Provide feedback

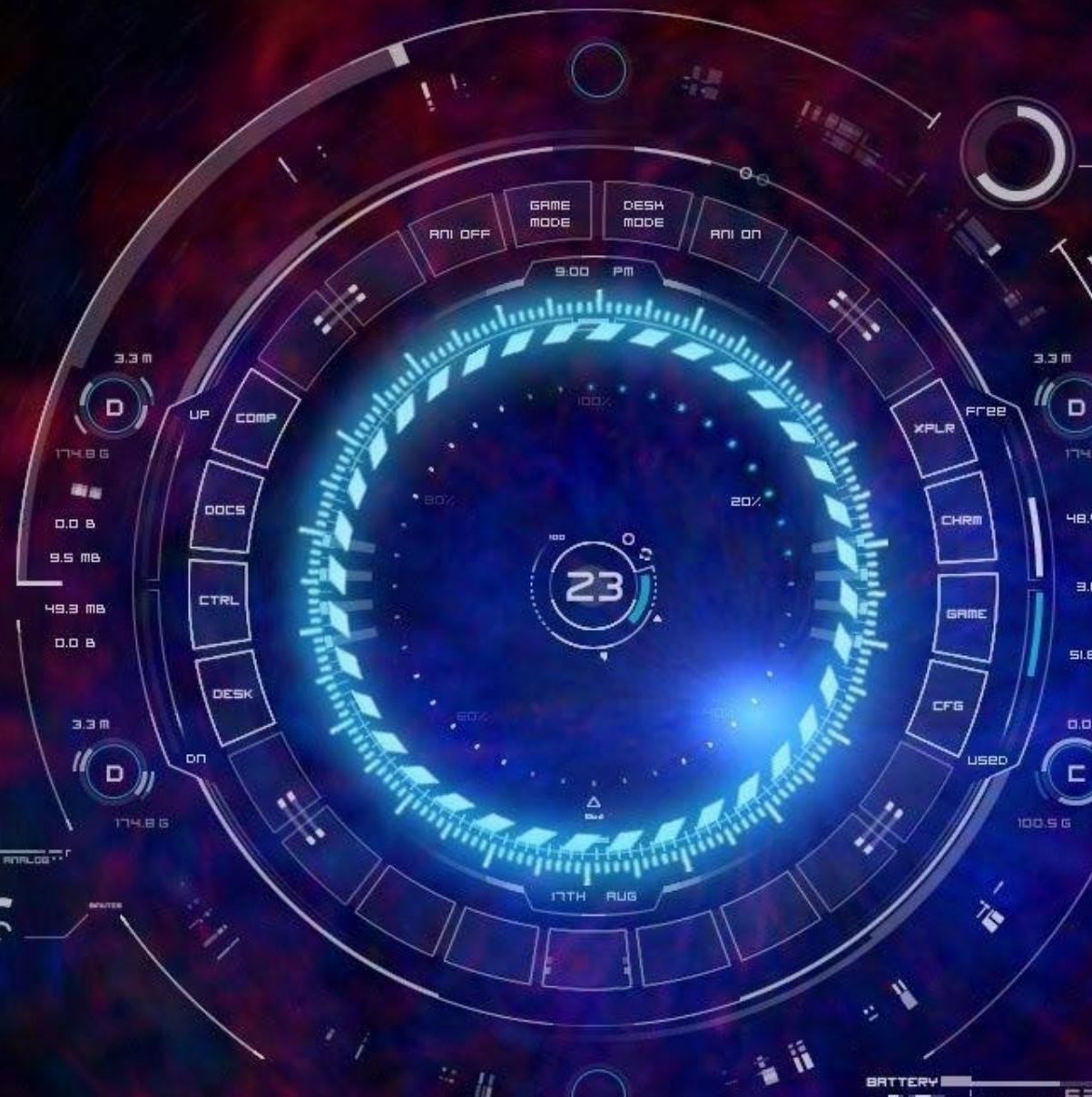
Amazon S3 > redaone-backup

redaone-backup [Info](#)

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Actions	Copy S3 URI	Copy URL	Download	Open	Delete	Actions	Create folder	Upload
Find objects by prefix								
	Name	Type	Last modified					
	first.html	html	August 26, 2021, 02:55:07 (UTC-04:00)					
				Size				Storage class
				739.0 B				Standard



21:00

SATURDAY

17 AUGUST 2013

Ahmedabad, India
Updated at 8/17/13 8:10 PM Local Time

29°C

Haze

Humidity: 84%
Feels Like: 35°
Precipitation: 40%
Visibility: 4.0 km
Wind: 11 km/h (SW)
Pressure: 1002.0 mb (steady)
Sunrise: 6:17 AM
Sunset: 7:11 PM
Moon Phase: Waxing Gibbous

Tonight
25°
Scattered T-Storms

Tomorrow
Aug 18
32° / 25°
AM Clouds / PM Sun

Monday
Aug 19
31° / 25°
Mostly Cloudy

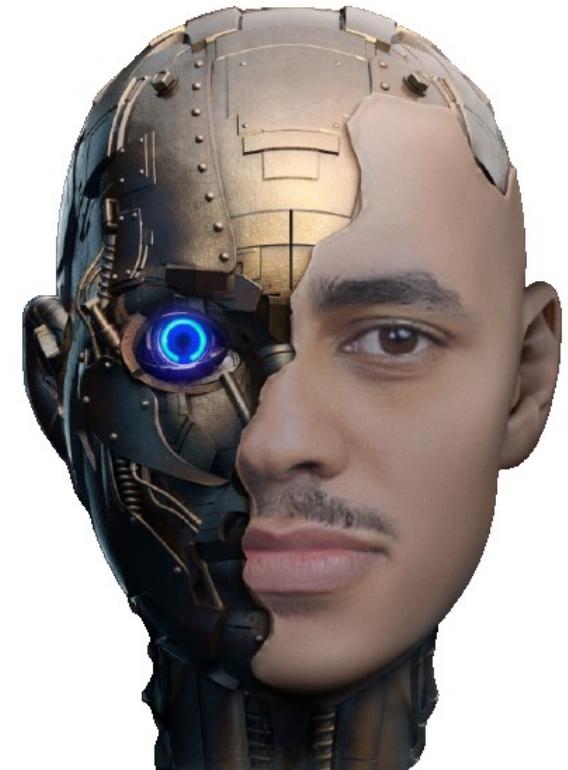
BATTERY
62%
NO BATTERY

Starting to build a Schedulify Skill

Designing a Voice Interaction Model



- ▶ Invocation Name:
 - ▶ Reda
- ▶ Job:
 - ▶ Read out Quotes
 - ▶ Example: Alexa, Ask Reda for a quote from Einstein
 - ▶ Example: Alexa, Ask Reda what did Abraham Lincoln say

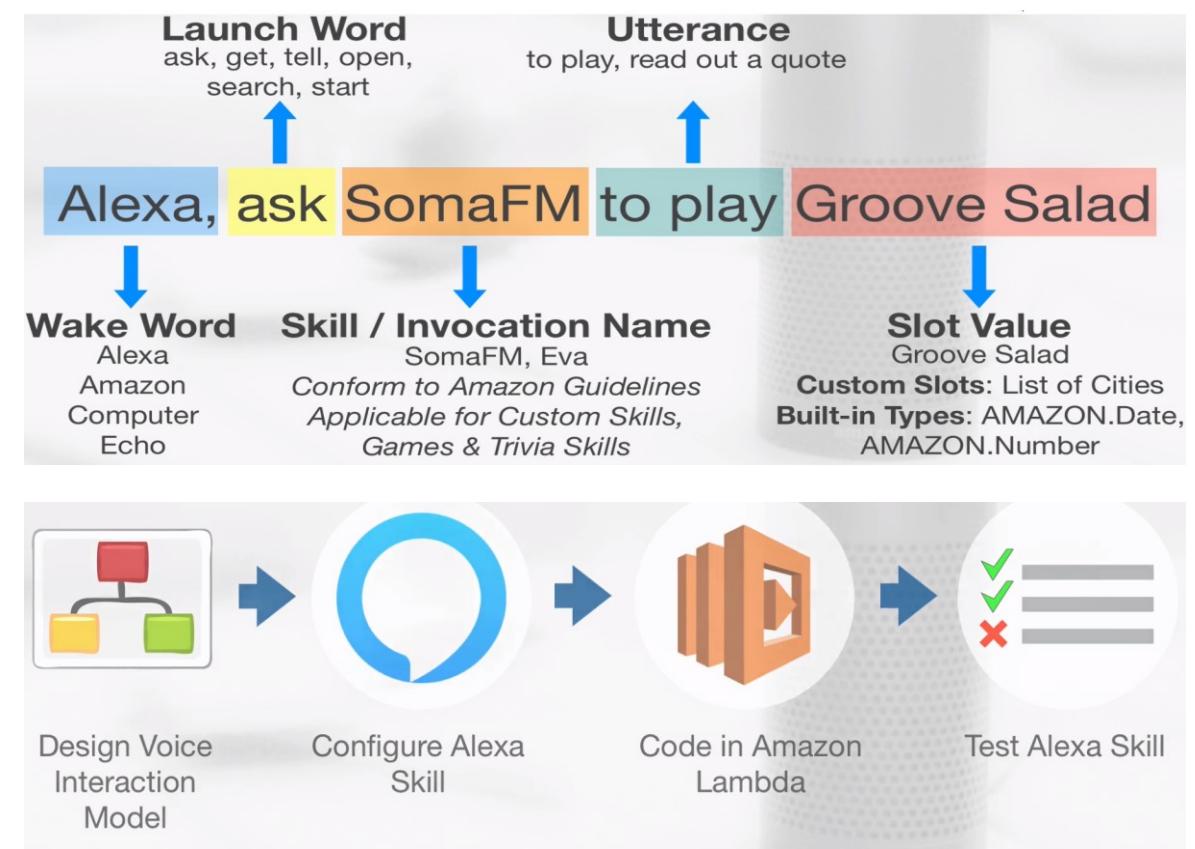


Starting to build a Schedulify Skill

Designing a Voice Interaction Model

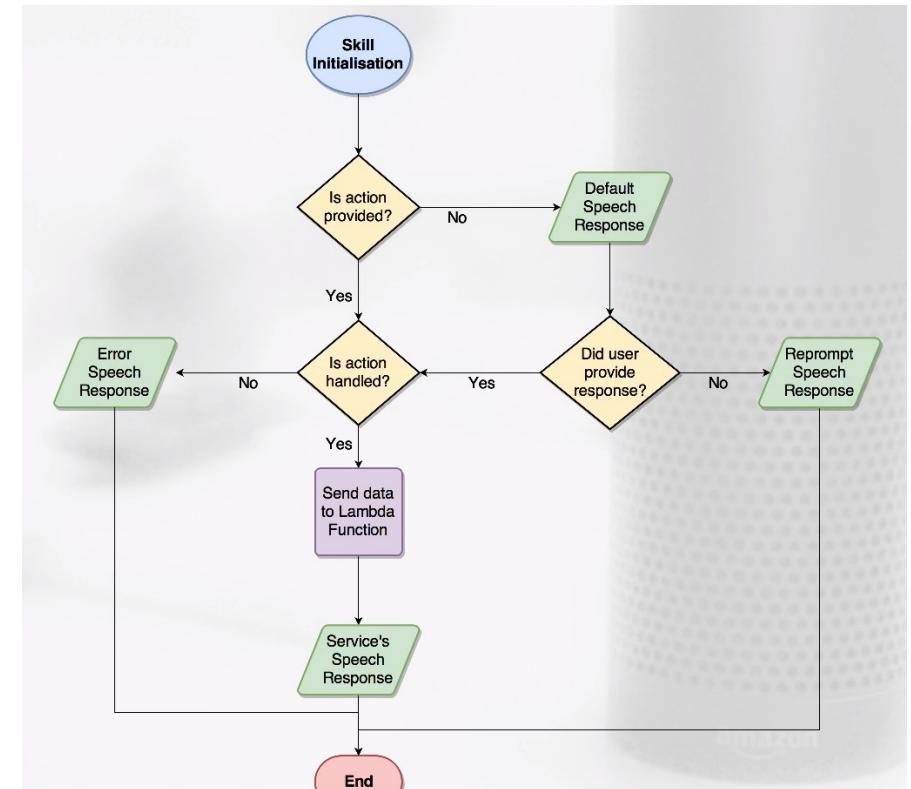
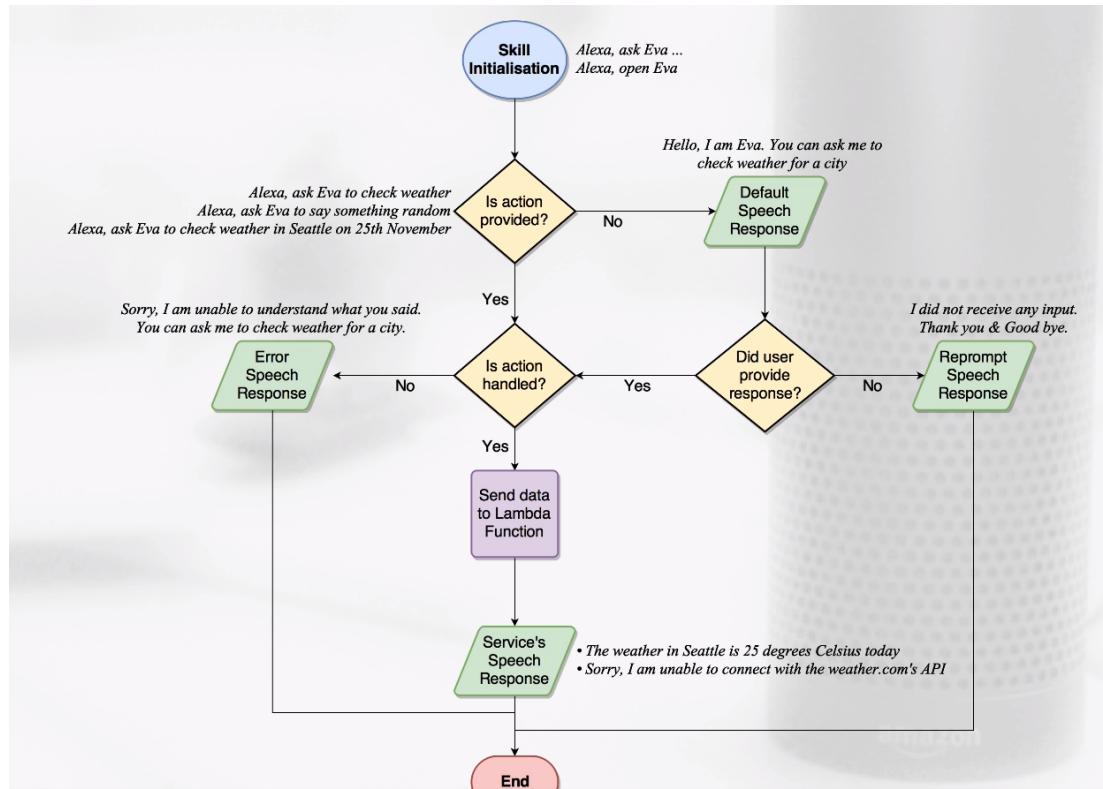


- ▶ Step 1: Preliminary work
 - ▶ Create a Voice Interaction Model
 - ▶ Prepare Intents, Slots, and Utterances
 - ▶ Data - Authors and their Quotations
- ▶ Step 2: Skill Configuration
 - ▶ On Amazon Developer Portal
- ▶ Step 3: Skill Development
 - ▶ Create a Lambda function
 - ▶ Link Lambda function with Alexa Skill through ARN
- ▶ Step 4: Skill Testing
 - ▶ Lambda test events
 - ▶ Skill simulator on Developer Portal
 - ▶ Physical device / Amazon Echo





Starting to build a Schedulify Skill



Starting to build a Schedulify Skill



Create a new skill

Skill name
Reda - The Awesome Personal Assistant

37/50 characters

Brand names are only allowed if you provide proof of rights in the testing instructions or if you use the brand name in a referential manner that doesn't imply ownership (examples of terms that can be added to a brand name for referential usage: unofficial, unauthorized, fan, fandom, for, about).

Primary locale
A locale refers to a language and the location (country) in which it's spoken. Your primary locale is what you will start building your skill in. You can add locales after your skill is created.

English (US)
Sync locales

Sync all locales with the same language as the Primary locale. Changes you make to your skill in the Primary locale automatically propagate to all other locales of the same language. You can manage these settings or turn off this feature anytime in Language settings. [Learn more](#)

1. 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 RELATED

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

"Alexa, what's in the news?"

Flash Briefing

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

"Alexa, turn on the kitchen lights"

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, play music by Lady Gaga"

Music

Give users complete control of their music. This pre-built model lets users search, pause, skip, or shuffle in your skill.

"Alexa, play interstellar"

Video

Let users find and consume video content. This pre-built model supports content searches and content suggestions.

"Alexa, how do I get on the Wi-Fi?"

Knowledge

Give users Q&A on bus upload via spreadsheet personal devices. Alexa for Hospitality, or Alexa for Residential.

"Alexa, what's the weather like in New York?"

meeting rooms in their office.

Skill builder checklist

Complete these steps to be able to test your skill using the simulator in the test tab, or with your echo device.

2. 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.

Alexa-hosted (Node.js) RELATED

Alexa will host skills in your account and get you started with a Node.js template. You will gain access to AWS Lambda endpoints in all Alexa service regions, a DynamoDB table for data persistence, and S3 for media storage. [Learn more](#)

Alexa-hosted (Python)

Alexa will host skills in your account and get you started with a Python template. You will gain access to AWS Lambda endpoints in all Alexa service regions, a DynamoDB table for data persistence, and S3 for media storage. [Learn more](#)

Provision your own

Provision your own endpoint and backend resources for your skill. This is recommended for skills that have significant data transfer requirements. You will not gain access to the console's code editor.

Create skill

Model: Custom
Host: Alexa Hosted (Node.js)
Hosting Region: US East (N. Virginia)

Choose a template to add to your skill

Select a skill template from the list below or import a skill shared by the Alexa community as a public skill.

Start from Scratch SELECTED

This skill gets you started with the required intents and with code demonstrating "Hello World" functionality if you are building an Alexa-hosted skill.

[Learn more](#)

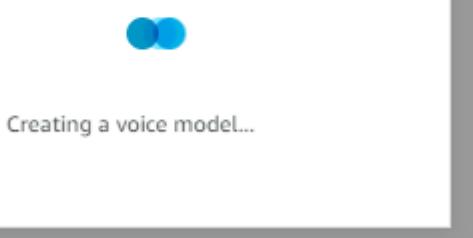
By Alexa

Fact Skill

Build a skill that answers questions on topics of your choice. It can be used to answer questions on anything from sports to science to history.

Creating your Alexa hosted skill.

It will take about a minute.



Starting to build a Schedulify Skill

Configuration of Reda 1.0 Custom Skill on Dev Portal



The image shows two screenshots of the Alexa developer console interface, illustrating the configuration of a custom skill named "Reda - The Awesome Personal Assistant".

Left Screenshot (Invocation Configuration):

- Header:** Alexa developer console, Your Skills, Reda - The Awesome Personal Assistant, Build, Code, Test, Distribution, Certification, Analytics.
- Left Sidebar:** English (US), CUSTOM, Invocations, Skill Invocation Name (selected), Skill Launch Phrases, Intent Launch Phrases, Interaction Model, Intents (6).
- Content Area:**
 - Invocation:** A section explaining what users say to invoke the skill. It shows a sample utterance: "User: Alexa, ask daily horoscopes for the horoscope for Gemini".
 - Skill Invocation Name:** A field containing "reda skill". A note states: "Brand names are only allowed if you provide proof of rights in the referential manner that doesn't imply ownership (examples of term usage: unofficial, unauthorized, fan, fandom, for, about)."

Right Screenshot (Intent and Slot Configuration):

- Header:** Your Skills, Reda - The Awesome Personal Assistant, Build, Code, Test, Distribution, Certification, Analytics.
- Left Sidebar:** English (US), CUSTOM, Invocations, Skill Invocation Name, Skill Launch Phrases, Intent Launch Phrases, Interaction Model, Intents (7) (selected), HelloWorldIntent, RandomQuote (selected), Built-In Intents (5) (sub-section), AMAZON.CancelIntent, AMAZON.HelpIntent, AMAZON.StopIntent, AMAZON.NavigateHomeIntent, AMAZON.FallbackIntent.
- Content Area:**
 - Intents / RandomQuote:** A section showing sample utterances for the "RandomQuote" intent. Examples include "tell me a random quote", "tell me a quote", "give me a quote", "random quote", and "random quotation".
 - Slot Types / LIST_OF_AUTHORS:** A section defining slot types. It lists "Slot Values (7)" with entries: "harry s truman", "richard feynman". Below is a table for managing slot values:

Value	ID (OPTIONAL)	SYNONYMS (OPTIONAL)
Lincoln	Enter ID	Add synonym
Abe Lincoln	Enter ID	Add synonym
Abraham Lincoln	Enter ID	Add synonym
Einstein	Enter ID	Add synonym
Albert Einstein	Enter ID	Add synonym
CP Jolis	Enter ID	Add synonym
CP	Enter ID	Add synonym

Bottom Right Overlay: A dark overlay with the text "Building Model" and two blue circular icons.

Starting to build a Schedulify Skill

Request handlers, Response Objects and Builder in Lambda function for Alexa



- ▶ Request Handlers
 - ▶ Custom handlers: Manage intents
 - ▶ Pre-defined handlers: Provided by Amazon
- ▶ Intent Request Handlers: Managing intents defined in intents schema
 - ▶ What is done: **RandomQuote** and **AuthorQuote**
 - ▶ It is called when Utterance matches Intent Slot Value is passed on if applicable
- ▶ Pre-defined Request Handlers
 - ▶ Like **LaunchRequest**: When skill is invoked without any action
 - ▶ Example: Alex, open Reda
 - ▶ Then there is **SessionEndedRequest**: When an Alexa session ends
- ▶ Built-in Intents
 - ▶ Such as **AMAZON.HelpIntent**: if user opens skill, and asks for help
 - ▶ Need to be provided in the Intent Schema. No sample Utterance required
- ▶ Response Objects
 - ▶ There are two: Audible Response and Visual Response
- ▶ Response Objects
 - ▶ There are two: Audible Response and Visual Response
- ▶ Response Objects
 - ▶ Audible Response Object:
 - ▶ Output Speech Response: A skill reads out a quote
 - ▶ Reprompt Speech Response: Ask user again if no input provided, while maintaining context
 - ▶ Visual Response Object:
 - ▶ For echo show/spot, Alexa app
 - ▶ **Cards**: they provide **Title & Content**
 - ▶ Using both Audio and Visual
 - ▶ In lambda function, provide:
 - ▶ Card values
 - ▶ Speech output
 - ▶ Response Builder – ASK v2
 - ▶ Helper methods to construct response
 - ▶ To end conversation: **.speak()** method
 - ▶ To ask for user's input: **.reprompt()** method
 - ▶ To use a card response: **.withSimpleCard()** and **.withStandardCard()**

Starting to build a Schedulify Skill

Reda bot introducing itself: On AWS Lambda



Browse serverless app repository

Deploy a sample Lambda application from the AWS Serverless Application Repository.

	Sort by	Best Match		
<	1	2	3	>

SpheroAlexaSkill

Control Sphero 2.0 using Alexa and a local Sphero Server. Change color and draw shapes just using your voice!

alex-skills-kit alexa sphero

Jenn J 6 deployments

alex-skills-kit-nodejs-howtoskill

Create a parameter-based skill using a template called 'Minecraft Helper'. Ask how to craft an item in the game Minecraft, and this skill will give you instructions.

howto skill alexa

Alexa Skills Kit AWS verified author 5K deployments

aws Services ▾ Search for services, features, marketplace products, and docs [Alt+S]

Lambda > Applications > serverlessrepo-RedaFunction

serverlessrepo-RedaFunction

Overview Deployments Monitoring

Getting started

Welcome to your new application view. From here, you can view the resources that make up your application, and monitor performance, errors, and traffic metrics. [Learn more](#)

Set up your development environment

You can use the following tools and services to build, test, and deploy your applications.

- AWS Cloud9** Write and test your function code in a managed environment with the AWS SDK, libraries, and plugins needed for serverless development.
- Visual Studio** Quickly create .NET Core functions from a blueprint. Compile, deploy, configure, and test your function from within Visual Studio.
- JetBrains** Configure the AWS Toolkit for JetBrains to edit your serverless applications in JetBrains IDEs.
- AWS SAM CLI** Define the infrastructure for your serverless application in an AWS SAM template. Deploy your function and other application resources from the command line. Build, test, and debug function code locally in a Docker container that emulates the Lambda execution environment.
- AWS Lambda Partners** provide services and tools that you can use with your Lambda functions. [View all the current AWS Lambda Partners](#)

Resources

Logical ID	Physical ID	Type	Last modified
codeRole	serverlessrepo-RedaFunction-codeRole-1RS1FQO4N77LP	IAM Role	14 seconds ago

Starting to build a Schedulify Skill

Reda bot introducing itself: On AWS Lambda



Lambda > Functions > serverlessrepo-RedaFunction-code-y9gtV6vRHM2D

serverlessrepo-RedaFunction-code-y9gtV6vRHM2D

This function belongs to an application. Click here to manage it.

Alexa Skills Kit

+ Add trigger

Description: Alexa Base Template with ask-sdk-core and Node.js 10.x Support

Last modified: 2 minutes ago

Function ARN: arn:aws:lambda:us-east-1:899122676463:function:serverlessrepo-RedaFunction-code-y9gtV6vRHM2D

Application: serverlessrepo-RedaFunction

Intent Launch Phrases

Interaction Model

Intents (7)

- RandomQuote
- author
- Built-In Intents (5)
- AMAZON.CancelIntent
- AMAZON.HelpIntent
- AMAZON.StopIntent
- AMAZON.NavigateHomeIntent
- AMAZON.FallbackIntent

Annotation Sets (New)

Intent History

Utterance Conflicts (0)

JSON Editor

Assets

Slot Types (1)

Multimodal Responses

Interfaces

Endpoint

Service Endpoint Type

Select how you will host your skill's service endpoint. Best practices in choosing lambda regions. Learn more.

AWS Lambda ARN (Recommended)

Your Skill ID: amznTaskskill.e72d70bf-c951-449e-b72b-6135da8966f6

Copy to Clipboard

Default Region (Required): amaws:lambda:us-east-1:803351116066:function:e72d70bf-c951-449e-b72b-6135da8966f6:Release_0

North America (Optional): amaws:lambda:us-east-1:803351116066:function:e72d70bf-c951-449e-b72b-6135da8966f6:Release_0

Europe and India (Optional): amaws:lambda:eu-west-2:803351116066:function:e72d70bf-c951-449e-b72b-6135da8966f6:Release_0

Far East (Optional): amaws:lambda:us-west-2:803351116066:function:e72d70bf-c951-449e-b72b-6135da8966f6:Release_0

HTTPS (Optional):

Lambda > Add trigger

Add trigger

Trigger configuration

Alexa Skills Kit alexa_iot

Skill ID verification is an easy way to verify the Skill ID in an incoming request from a Skill. To set this up, enter the Skill ID (also called Application ID) of your skill located in your Alexa Skills Kit dashboard. [Learn more](#).

Enable (recommended)

Disable

Skill ID:

Lambda will add the necessary permissions for Amazon Alexa to invoke your Lambda function from this trigger. [Learn more](#) about the Lambda permissions model.

Cancel Add

Starting to build a Schedulify Skill

Reda bot introducing itself: On AWS Lambda



paste

Lambda > Functions > serverlessrepo-RedaFunction-code-y9gtV6vRHM2D

serverlessrepo-RedaFunction-code-y9gtV6vRHM2D

The trigger e72d70bf-c951-449e-b72b-6135da8966f6 was successfully added to function serverlessrepo-RedaFunction-code-y9gtV6vRHM2D. The function is now receiving events from the trigger.

Function overview Info

Alexa Skills Kit (Required)

Copy

Description: Alexa Base Template with ask-sdk-core and Node.js 10.x Support
Last modified: 14 minutes ago
Function ARN: arn:aws:lambda:us-east-1:899122676463:function:serverlessrepo-RedaFunction-code-y9gtV6vRHM2D
Application: serverlessrepo-RedaFunction

Throttle | Copy ARN | Actions ▾

Service Endpoint Type

Select how you will host your skill's service endpoint. Best practices in choosing lambda regions. [Learn more](#).

AWS Lambda ARN (Recommended)

Your Skill ID ?

amzn1.ask.skill.e72d70bf-c951-449e-b72b-6135da8966f6 [Copy to Clipboard](#)

Default Region (Required)

North America (Optional)

arn:aws:lambda:us-east-1:899122676463:function:serverlessrepo-RedaFunction-code-y9gtV6vRHM2D

Europe and India (Optional)

arn:aws:lambda:eu-west-1:<aws_account_id>.function:<lambda_name>

i Build Started [Dismiss](#)

You will be notified when the model build is complete.

✓ Skill Manifest Saved Successfully [Dismiss](#)

The skill manifest has been saved without errors. The Model must be successfully built for changes to take effect.

Starting to build a Schedulify Skill

Testing Reda Skill Bot



alexa developer console

Your Skills Reda - The Awesome Personal Assistant Build Code Test Distribution Certification Analytics

Skill testing is enabled in: Development

Skill I/O Device Display Device Log

Alexa Simulator Manual JSON Voice & Tone

English (US) Type or click and hold the mic

alexa open reda skill

Hi Fulcrum Digital associates! Let me introduce myself.. I am Reda. I am glad to be proudly engineered, to serve you!.. Please, le me know how I may help you

Skill Invocations | Viewing: 1 / 1

JSON Input 1

```

1: {
2:   "version": "1.0",
3:   "session": {
4:     "new": true,
5:     "sessionId": "amzn1.echo-api.session.2abf942d-0d9d-4cdd-86da-6814a30fb91",
6:     "application": {
7:       "applicationId": "amzn1.ask.skill.e72d70bf-c951-449e-b72b-6135da8966f6"
8:     },
9:     "attributes": {},
10:    "user": {
11:      "userId": "amzn1.ask.account.AH256R55IB6WCJGJAPLCL3XTFGQOF22AQ7N6042XCCG366NR1WF4"
12:    }
13:  },
14:  "context": {
15:    "Viewports": [
16:      {
17:        "type": "API",
18:        "id": "main",
19:        "shape": "RECTANGLE",
20:        "dpi": 213,
21:        "presentationType": "STANDARD",
22:        "canRotate": false,
23:        "configuration": {
24:          "current": {
25:            "mode": "HUB",
26:            "video": {
27:              "codecs": [
28:                "H_264_42",
29:                "H_264_41"
30:              ]
31:            },
32:            "size": {
33:              "type": "DISCRETE",
34:              "width": 1280,
35:              "height": 720
36:            }
37:          }
38:        }
39:      }
40:    ]
41:  }
42: }

```

JSON Output 1

```

1: {
2:   "body": {
3:     "version": "1.0",
4:     "response": {
5:       "outputSpeech": {
6:         "type": "SSML",
7:         "ssml": "<speak>Hi Fulcrum Digital associates! Let me introduce myself.. I am Reda. I am glad to be proudly engineered, to serve you!.. Please, le me know how I may help you</speak>"
8:       },
9:       "type": "_DEFAULT_RESPONSE"
10:     },
11:     "sessionAttributes": {},
12:     "userAgent": "ask-node/2.7.0 Node/v10.24.1"
13:   }
14: }

```

Starting to build a Schedulify Skill

Extending Lambda function to achieve first goal



```
// Include the Alexa SDK v2const Alexa = require("ask-sdk-core");// The  
"LaunchRequest" intent handler - called when the skill is launchedconst  
LaunchRequestHandler = { canHandle(handlerInput) { return  
handlerInput.requestEnvelope.request.type === "LaunchRequest"; },  
handle(handlerInput) { const speechText = "Hi Fulcrum Digital associates! Let  
me introduce myself.. I am Reda. I am glad to be proudly engineered, to serve  
you!.. Please, le me know how I may help you"; // Speak out the speechText  
via Alexa return  
handlerInput.responseBuilder.speak(speechText).getResponse(); }};// Register  
the handlers and make them ready for use in Lambdaexports.handler =  
Alexa.SkillBuilders.custom() .addRequestHandlers(LaunchRequestHandler)  
.lambda();
```



Starting to build a Schedulify Skill



- ▶ Revenue
 - ▶ Expenditure
- ▶ Generating revenue
- ▶ Culinary suite

Next ..

Engage your customers with an Alexa skill

- ▶ Companies large and small are **building Alexa skills** to capitalize on this growing opportunity — there are now more than 30,000 skills published in Amazon's catalog.
- ▶ FulcrumOne team of **Alexa developers** can help your company extend your digital reach through a custom skill on millions of **Amazon Echo devices**.



Wrapping Up

- ▶ In a voice-first world, Alexa, Google Assistant, Siri, and other **AI-enabled assistants** will be a powerful marketing medium. It can combine sales and distribution channel, fulfillment, and service center. Following the gold rush of building brand mobile apps, the current trend is building branded skills for voice assistants.
- ▶ The increasingly capable voice assistants will be reshaping the brands' and retailers' connections with customers. The consumers' allegiance will be shifting from trusted brands **to a trusted platform that helps them navigate the ocean of choices**. Re-ordered essentials will be flowing to households like water or electricity. Growing creative audio content will advertise without annoying and alienating listeners. The AI will gather information about the consumers' preferences to improve the shopping experience and offer the portal to a boundless marketplace of entertainment, news, goods, and services. Amazon Alexa is sure to be one of those platforms.
- ▶ For publishers, music services, and smart-home tech companies, custom Alexa skills seem to be a must for increasing engagement with millions of users. Other company managers and marketers should consider creating a branded skill as well: it offers a natural way to introduce a brand to potential consumers and abundant opportunities for product improvement, marketing, and advertising, and helps cultivate customer loyalty.

References

- ▶ **How do Alexa Skills work?:** <https://chatbotsmagazine.com/how-does-alexa-skills-works-82a7e93dea04>
- ▶ **ASK Documentation:** <https://developer.amazon.com/en-US/docs/alexa/custom-skills/understanding-custom-skills.html>
- ▶ **How does your brand benefit from developing an Alexa Skill?:** <https://www.deligence.com/blog/benefit-from-developing-an-alexa-skill/>
- ▶ **Practical Hands-on Guide for Alexa Skill Development (2020):** <https://fulcrumdigital.udemy.com/course/build-your-rad-personal-assistant-with-amazon-alexa-custom-skills/>