

MASTERs Brasil

# Como controlar seu dispositivo IoT utilizando Amazon Alexa

---

## Lab Manual



Ricardo Seiti  
Microchip Technology Inc.

  
**MICROCHIP**  
MASTERs Conference

# Como controlar seu dispositivo IoT utilizando Amazon Alexa

## Conteúdo

<b>Lab 1: Como criar um skill utilizando Alexa .....</b>	<b>1-1</b>
<b>Lab 2: Como conectar um dispositivo ao AWS IoT.....</b>	<b>2-1</b>
<b>Lab 3: Como controlar um dispositivo utilizando Alexa.....</b>	<b>3-1</b>
<b>Anexo A: Informações de Hardware.....</b>	<b>A-1</b>
<b>Anexo B: Função Lambda do Exercicio um .....</b>	<b>A-2</b>
<b>Anexo C: Como configurar JITR .....</b>	<b>A-3</b>
<b>Anexo D: Projeto embarcado da placa Sensor Board .....</b>	<b>A-4</b>
<b>Anexo E: SAM-BA Bootloader para a placa Sensor Board .....</b>	<b>A-5</b>

## Introdução

Este material foi preparado para ser utilizado tanto durante o treinamento como também como material de consulta após o treinamento.

Para o treinamento, as configurações do computador e da conta AWS já estão preparadas para o fluxo normal das práticas preparadas. Para configurar corretamente o seu ambiente após o treinamento, siga as instruções que estão nos apêndices.

## Ferramentas necessárias

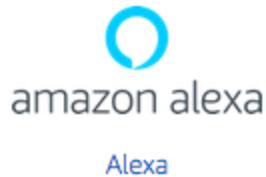
Para o treinamento, precisaremos das seguintes ferramentas:

### Hardware

- Computador com Windows 7 ou superior

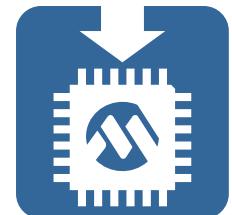
### Software

- Navegador Google Chrome ou equivalente



# Lab 1

## Como criar uma skill Alexa



### ?

## Proposta

Criar uma Skill simples para Alexa:

- Como criar uma nova Skill
- Como testar uma Skill utilizando Alexa Simulator

### ✓ Requisitos

**Ambiente de desenvolvimento:** Computador com navegador web

**Hardware :** Nenhum

**Arquivos no pendrive:** D:/masters/alexa/lab1

### TargetException

Para a criação da primeira skill, o instrutor irá mostrar passo a passo de como criar uma conta de desenvolvedor e como criar a sua primeira Skill no Alexa Developer.

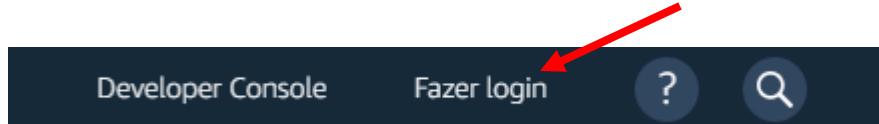


## Passo a passo

Lab 1 Como criar uma conta no Amazon Developer e como Criar a primeira Skill

### Passo 1: Abrir ou criar uma conta no Amazon Developer

- 1.1 Abra a pagina <https://developer.amazon.com/> .
- 1.2 Clique em “Fazer login” no topo direito do navegador.



- 1.3 Criar uma nova conta (ou fazer o login in se você já tiver uma conta)

#### Conta Existente

##### Fazer login

E-mail ou número de telefone celular

Senha

 [Esqueci a senha](#)

**Fazer login**

Ao continuar, você concorda com os [Condições de Uso](#) e com a [Política de Privacidade](#) da Amazon.

Novo na ?

**Criar sua conta da**

#### Nova conta



##### Criar conta

Seu nome

E-mail

Senha

Pelo menos 6 caracteres

i As senhas devem ter pelo menos 6 caracteres.

Insira a senha nova mais uma vez

**Criar sua conta da**

Ao criar uma conta, você concorda com as [Condições de uso da Amazon](#) e com a [Política de privacidade](#).

Você já tem uma conta? [Fazer login](#) ›

- 1.4 Vá diretamente para o passo 2

- 1.5 Vá para o passo 1.6



## Passo a passo

### 1.6 Código de verificação será enviado para o email de cadastro

**Amazon Sign-in:**

Verifique sua nova conta da Amazon

Insira o seguinte código:  
**535976**

Não compartilhe este código com ninguém visto que concederia acesso a sua conta da Amazon.

Obrigado por comprar conosco! Esperamos vê-lo novamente em breve.

**Amazon Developer Portal - Verify email address:**

Verificar o endereço de e-mail

Para verificar seu email, enviamos um código para ricardo.seiti@microchip.com (Alterar)

Inserir o código

Verificar

Reenviar código

### 1.7 Entre com todas as informações mandatórias (\*) e clique “Save and Continue”

#### Registration

1. Profile Information    2. Amazon Developer Services Agreement

\* indicates a required field.

Country/Region *	Brazil
First name *	Ricardo
Last name *	Seiti
Email address *	ricardo.seiti@microchip.com
Phone number *	e.g. 212-555-1212, +44 0161 715 3369
Fax number	
Developer name or company name *	If you decide to publish App or Skill on amazon.com then this name will be displayed
Developer description or Company description	Maximum characters 4000, Remaining: 4000
Address 1 *	
Address 2	
City *	
State/Province/Region *	
Zip code/Postal code *	

### 1.8 Accept license

**\*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.

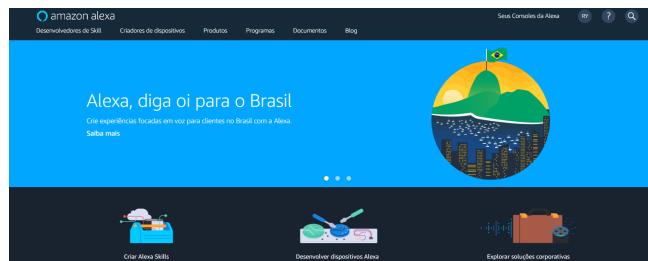
Cancel  Accept and Continue



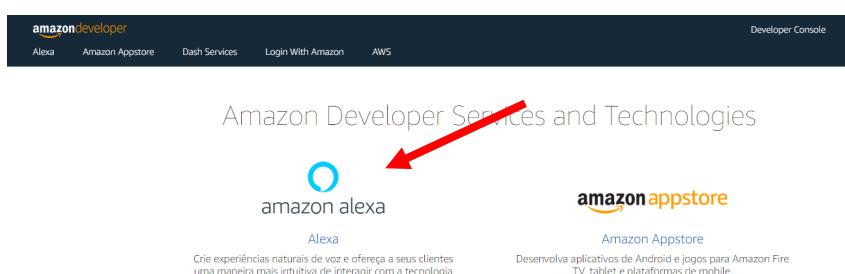
## Passo a passo

### Passo 2 Como criar uma Skill

2.1 É possível acessar o console Alexa diretamente pelo link:  
<https://developer.amazon.com/alexa>



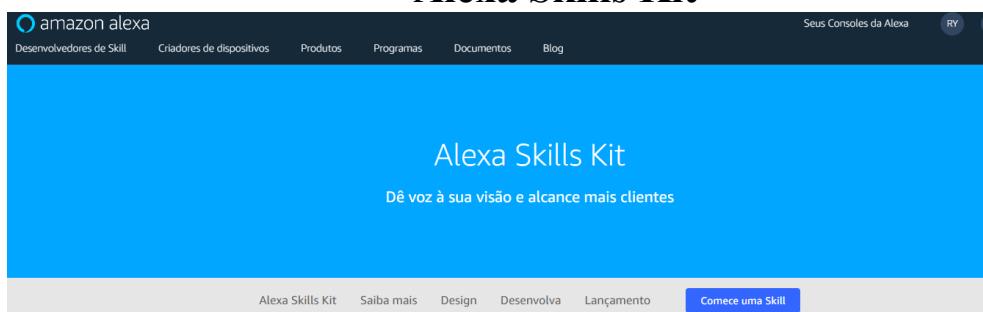
Ou acessar pela pagina inicial do developer, clicar em Amazon Alexa



2.2 Clique na aba Produtos -> Alexa Skill kit



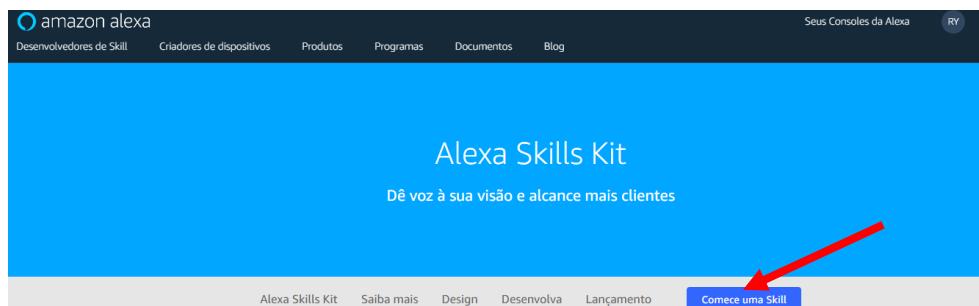
### Alexa Skills Kit



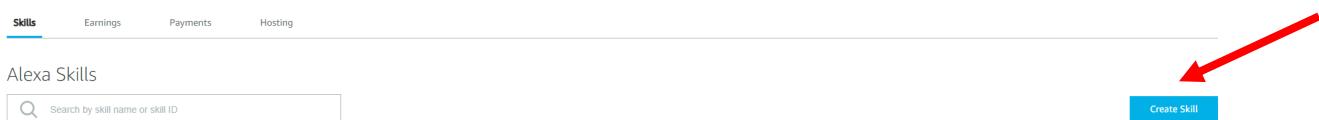


## Passo a passo

### 2.3 Como criar uma Skill. Clique em “Comece uma Skill”



### 2.4 Clique no botão “Create Skill” do lado direito do navegador



### 2.5 Defina o nome do skill e o idioma, no caso “Portuguese (BR)”

#### Create a new skill

Skill name

4/50 characters

Default language

More languages can be added to your skill after creation

### 2.6 Abaixo em “Choose a model to add to your skill”, selecione o “Custom” e clique “Create skill” (no topo direito do navegador)

#### 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 package of intents and utterances that you can add to your skill.

<b>Custom</b> SELECTED	<b>Flash Briefing</b> 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?"	<b>Smart Home</b> Give users control of home devices. This lets users turn off other devices without  "Alexa, turn on the lights"
---------------------------	--	--

Feedback forum

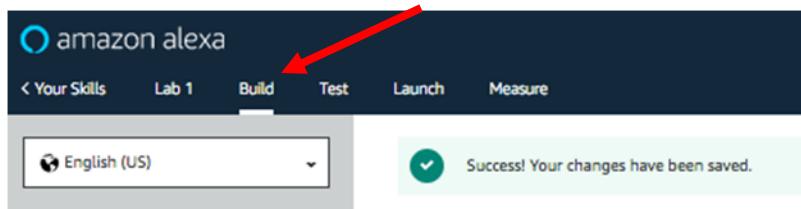
Cancel Create skill



## Passo a passo

### Passo 3 Criar a palavra de invocação de seu skill

#### 3.1 clique na aba build “Build”



#### 3.2 Clique em “Invocation” do lado esquerdo do navegador e escolha um nome para ser utilizado como palavra de invocação de sua skill. Exemplo “exercicio um”.

**Invocation**

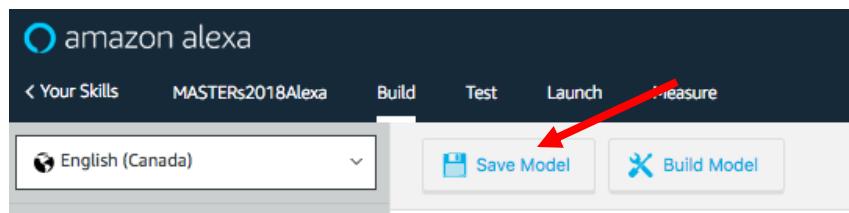
Users say a skill's invocation name to begin an interaction with a particular custom skill. For example, if the invocation name is "daily horoscopes", users can say:

User: Alexa, ask daily horoscopes for the horoscope for Gemini

Skill Invocation Name [?](#)

exercicio um

Dica, clique em “SAVE”  
após pequenas alterações





## Passo a passo

### Step 4 Criar as “intents” da sua skill



- Por padrão, 4 intents já estão presentes.

#### 3.3 Clique em “+ Add” Intent:

Intents (4)

Built-In Intents (4)

- AMAZON.CancelIntent
- AMAZON.HelpIntent
- AMAZON.StopIntent
- AMAZON.NavigateHomeIntent

#### 3.4 Criar a intent chamada “secretKey”. Precisa ser criada exatamente desta forma, cuidando com as letras maiúsculas e minúsculas! Clique em “Create custom intent”

##### Add Intent

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

- Create custom intent ?



#### 3.5 Adicione as seguintes “Utterances”, não esqueça de clicar em “+” depois de digitar cada Utterance. Clique “Save Model”

Intents / secretKey

Sample Utterances (3) ?

What might a user say to invoke this intent?

Qual é a palavra secreta	<input type="button" value="Delete"/>
Que segredo	<input type="button" value="Delete"/>
Qual é o segredo	<input type="button" value="Delete"/>



## Passo a passo

### Passo 5 Criar um Slots

Antes de criar um Segundo intent, vamos criar um Slot

**5.1** Clique em “+” ao lado do Slot type para criar um slot.

The screenshot shows the AWS Lambda function configuration interface. In the top left, there's a tree view with 'Intents (5)' expanded, showing items like 'secretKey' and 'Built-In Intents (4)'. Below it is a section for 'Slot Types (0)'. A red arrow points to the '+ Add' button next to the 'Slot Types' heading. At the bottom, there's a 'JSON Editor' tab.

**5.2** Crie o slot “KEY\_TYPE” (letras maiúsculas) e clique em “Create custom slot type”:

#### Add Slot Type

Slot types define how data in an intent slot is recognized and handled. All intents using slot types.

The screenshot shows the 'Add Slot Type' dialog. It has a radio button for 'Create custom slot type' which is selected. Below it is an input field containing 'KEY\_TYPE' with a red arrow pointing to it. To the right of the input field is a blue 'Create custom slot type' button with a red arrow pointing to it.

**5.3** Adicione o valor “masters” e clique em “+”

#### Slot Types / KEY\_TYPE

Slot Values (0)

Bulk Edit

Export

Search

masters

+



## Passo a passo

### Step 6 Criando o segundo Intent

Agora que temos o slot, vamos criar o Segundo Intent.

**6.1** Clique em “+ Add” Intents, para criar o Intent “secretMessage”, conforme passo anterior.

The screenshot shows the 'Add Intent' interface. On the left, there's a sidebar with 'CUSTOM' selected, showing sections for 'Interaction Model', 'Invocation', 'Intents (4)', 'Built-In Intents (3)', and 'Slot Types (0)'. Under 'Intents (4)', there's a slot named 'secretKey'. A blue 'Add' button is located next to the 'Intents (4)' section. The main area is titled 'Add Intent' with the sub-instruction: 'An intent represents an action that fulfills a user's spoken request. Learn more about intents.' It contains two radio buttons: 'Create custom intent' (selected) with an input field containing 'secretMessage', and 'Use an existing intent from Alexa's built-in library' with a search bar. A blue 'Create custom intent' button is at the bottom right.

**6.2** Adicione “{key}”, e clique “+” para adicionar a Utterance.

### Intents / secretMessage

Sample Utterances (0)

Bulk Edit Export

{key}

+

Select an Existing Slot

No existing slots

**6.3** Adicione o Segundo Utterance “o segredo é {key}” e clique “+”:

### Intents / secretMessage

Sample Utterances (1)

Bulk Edit Export

o segredo é {key}

+

{key}



## Passo a passo

**6.4** Para o “Intent Slots” na parte inferior do navegador, defina o SLOT TYPE como “**KEY\_TYPE**” para a Intent Slot “key”.

### Intents / secretMessage

Sample Utterances (3) [?](#)

Bulk Edit Export

What might a user say to invoke this intent? [+](#)

a senha é {key}



o segredo é {key}



{key}



[◀ 1 – 3 of 3 ▶](#)

Dialog Delegation Strategy [?](#)

Dialog management is not enabled f...[▼](#)

Why is this disabled?

Intent Slots (1) [?](#)

ORDER <a href="#">?</a>	NAME <a href="#">?</a>	SLOT TYPE <a href="#">?</a>	ACTIONS
<a href="#">^</a> 1	key	KEY_TYPE <a href="#">▼</a>	<a href="#">Edit</a> <a href="#">Dialog</a> <a href="#">Delete</a>

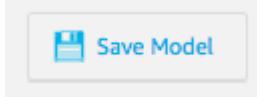
**6.5 Desabilite o “Intent Confirmation”**

### Intent Confirmation

Does this intent require confirmation? [?](#)



**6.6 Clique em “Save Model”**





## Passo a passo

### Step 7 Definindo um EndPoint

Agora precisamos conectar a nova skill criada com uma função Lambda que criamos previamente. O link ou ARN é :

**arn:aws:lambda:us-east-1:280639739381:function:MastersBR\_Alexa**

**7.1** Clique em “Endpoint” (1), selecione o “AWS Lambda ARN” (2) e entre com a ARN acima no “default region” (3)

**7.2** Clique “Save Endpoint” (4)

**1** **2** **3** **4**

**Salve a configuração do Endpoint:  
Na parte superior do navegador**



## Passo a passo

### Passo 8 Building Skill



- Neste momento iremos criar o nosso skill. Estamos pronto para compilar o primeiro skill

**8.2 Clique em “Invocation” e depois em “Build Model”**



Vai levar alguns minutos para compilar

The screenshot shows the AWS Lambda Skills Console interface. At the top, there's a navigation bar with tabs: 'Your Skills', 'MASTERs2018Alexa', 'Build', 'Test', 'Distribution', 'Certification', and 'Analytics'. Below the navigation bar, there's a dropdown menu set to 'English (CA)'. To the right of the dropdown are two buttons: 'Save Model' and 'Build Model', with a red arrow pointing to 'Build Model'. On the left, there are sections for 'CUSTOM' (Interaction Model) and 'Invocation'. A red arrow points to the 'Invocation' section, which is highlighted in blue. In the center, under 'Invocation', there's a box with the heading 'Invocation' and a sub-section titled 'Users say a skill's invocation name to begin an interaction with a p...'. Below this, a modal window titled 'Build Started' with a warning icon says: 'You will be notified when the model build is complete.' At the bottom of the screen, there's a progress bar labeled 'Building' with a 'Save Model' button to its left.

Build Started

You will be notified when the model build is complete.

Save Model

Building

Build Successful

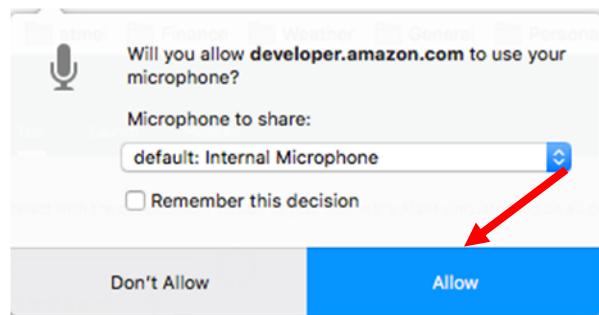
If you make any new changes, you will need to rebuild your model for them to take effect. - Saturday, Apr 7, 2018, 8:52 AM



## Passo a passo

Como testar. Clique na aba “Test” na parte superior do Browser:

**9.1** Caso necessário, permita que a pagina da Amazon utilize o microfone:



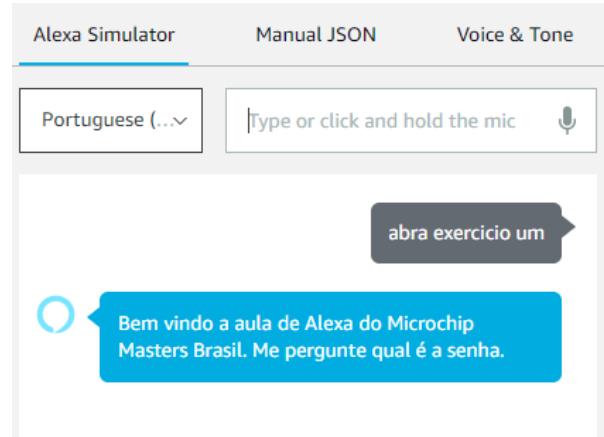
**9.2** Habilite o teste em modo de desenvolvimento “development”

**9.3** Digite a mensagem “abra exercicio um”. Ou pressione e segure o microfone ao lado da Caixa de texto e diga “abra exercicio um”.



**9.4** É possível ver o objeto JSON enviado e recebido entre a plataforma Alexa e a função Lambda.

**9.5** Em seguida entre com a pergunta configurada nos intents.



**9.6** É possível tanto digitar o comando como também utilizar o botão de microfone ao lado para enviar um áudio para ser interpretado. Na parte de baixo será possível visualizar o dialogo. Tente falar a senha errada para ver o que a Alexa responde. (A implementação da função Lambda está no anexo B.



## Passo a Passo

**9.7** O Simulador tem as mesmas funcionalidades do Amazon “Echo Plus” ou “DOT”. Assim é possível perguntar no simulador diversas perguntas como temperatura e horário. No exemplo abaixo foi perguntado a Alexa qual a temperatura e horário em Joinville Brasil.

que horas são em joinville

Em Joinville são 14 horas e 58 minutos.

qual é a data de hoje

É segunda-feira, 2 de setembro.

qual a temperatura em joinville

Agora, em Joinville Brasil, está 16 graus Celsius. À noite, a mínima será de 13 graus.

**9.8** Teste outras perguntas no simulador !

quando nasceu airton senna

qual a cotação do dolar



A data de nascimento de Ayrton Senna é 21/3/1960.



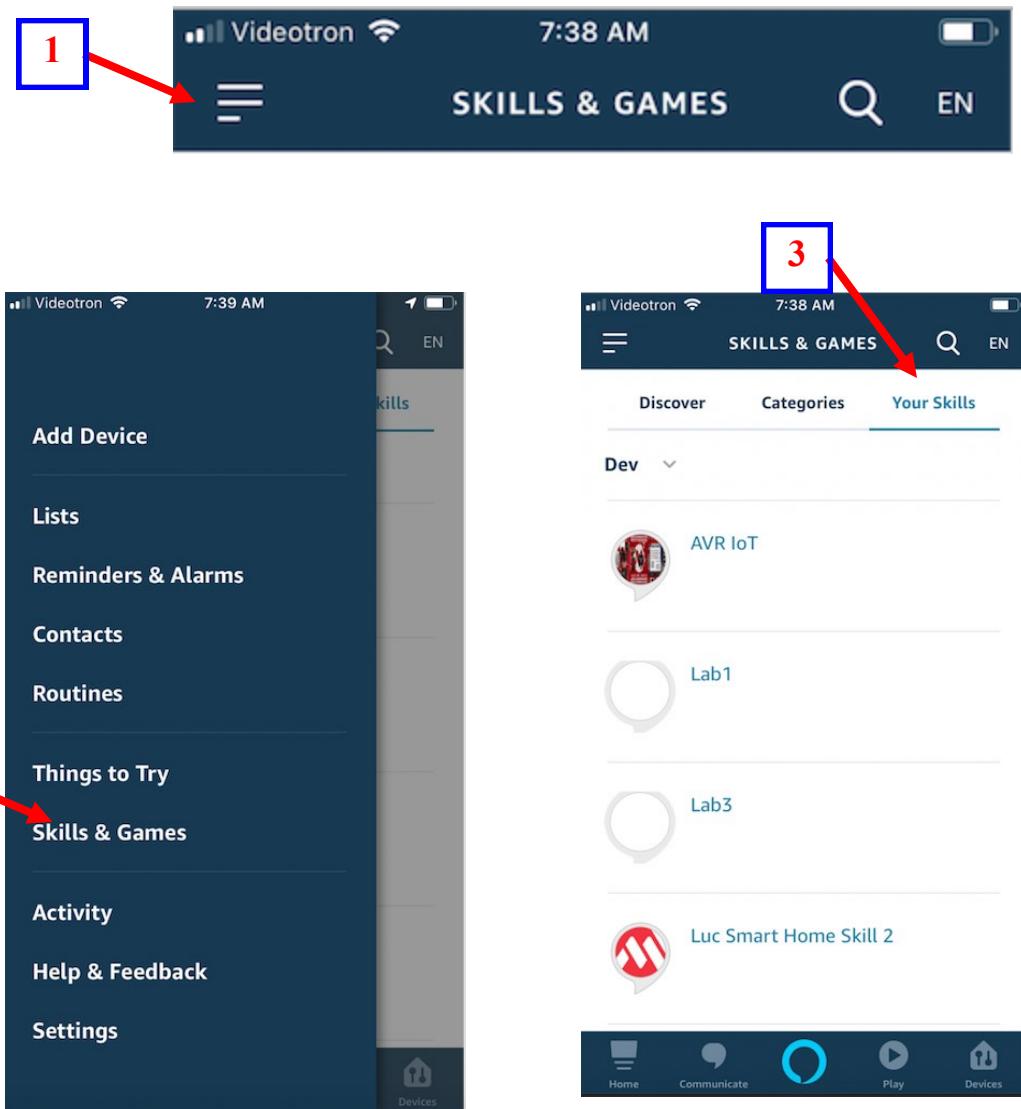
1 dólar equivale a R\$4,14.



## Passo a Passo

9.9 Se você tem o “**Alexa App**” no seu celular, você também pode usá-lo como assistente de voz. Verifique se você está usando a mesma conta de email que usou para criar a skill anterior..

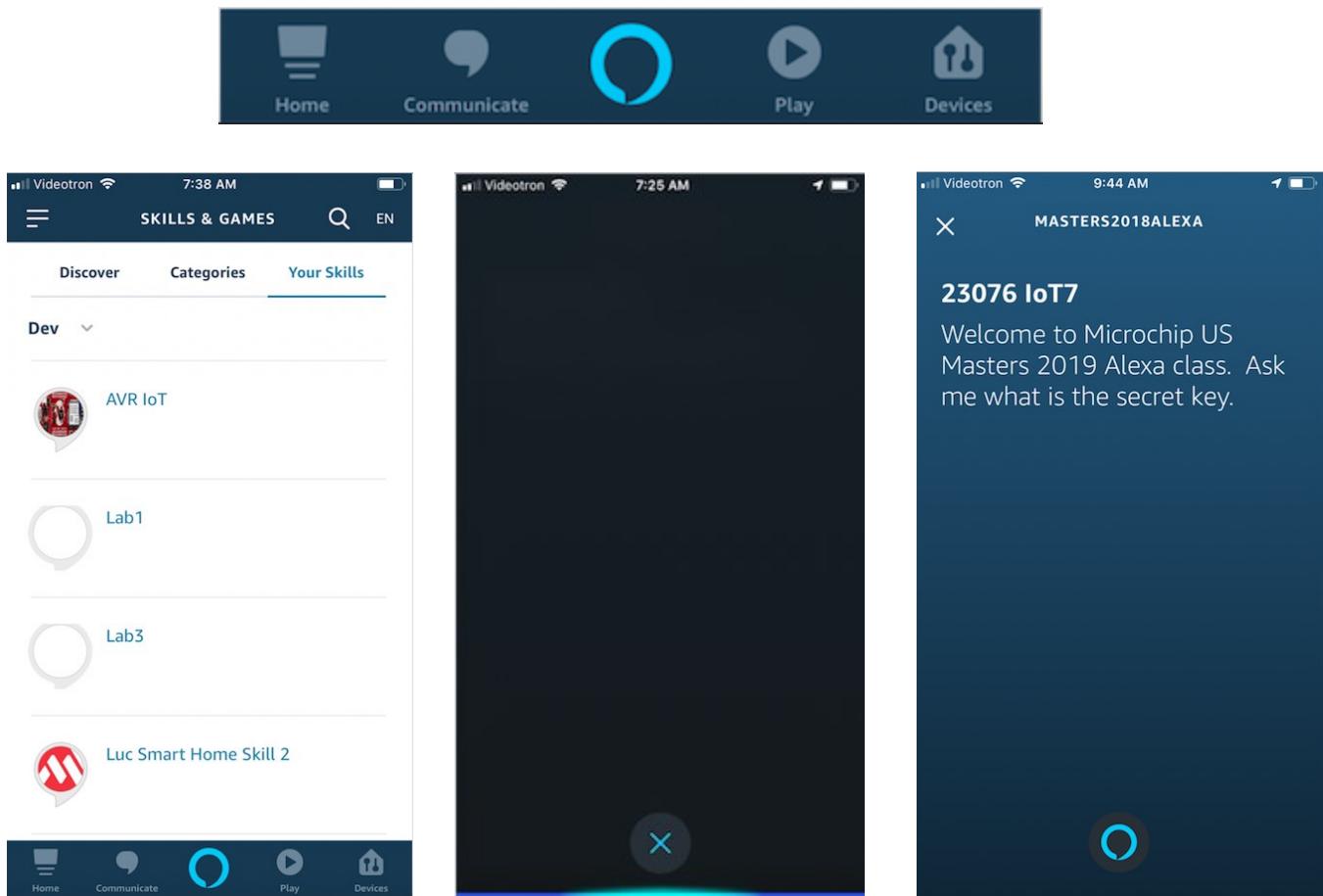
Abra o App, no Menu (1) na parte superior esquerda, selecione “Skills & Games” (2) e selecione “Your Skills” (3). Neste menu deve aparecer todos os Skill criados.

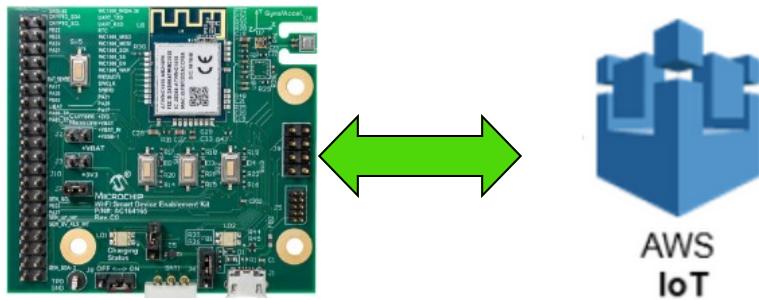




## Passo a passo

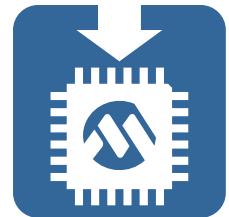
**9.10** Clique no logo Alexa central e diga “**abra exercicio um**”. Até o final da escrita deste documento, o entendimento em português não estava disponível.





# Lab 2

**Como conectar um dispositivo ao AWS IoT**



## ?

### Proposta

Create simple Alexa Custom Skill:

- 

## ✓ Requisitos

**Ambiente de desenvolvimento:** Computador com acesso a Internet

**Hardware :** Sensor Board, Cabo USB

**Arquivos estão disponíveis na pasta do projeto**

## TargetException

Neste lab, iremos conectar a placa Sensor Board ao AWS IoT utilizando o chip de criptografia ATEC-C608A já comissionado conforme o JITR (Just in Time Registration) na conta AWS preparada para este treinamento. Para configurar a sua conta, verifique o passo a passo no Anexo C. A comunicação entre o AWS IoT e a placa usará protocolo MQTT.



## Passo a passo

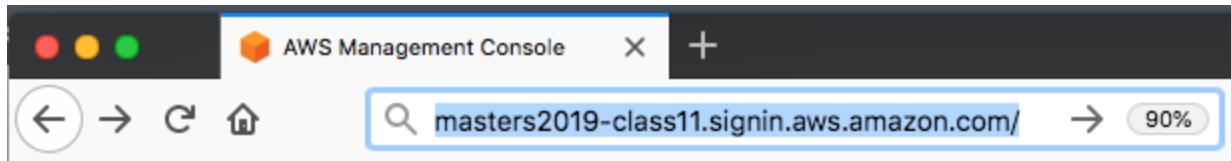
**NÃO conecte a placa até o ponto certo  
da documentação.**

**Se você conectou, teremos que  
RESETAR a sua conta.**

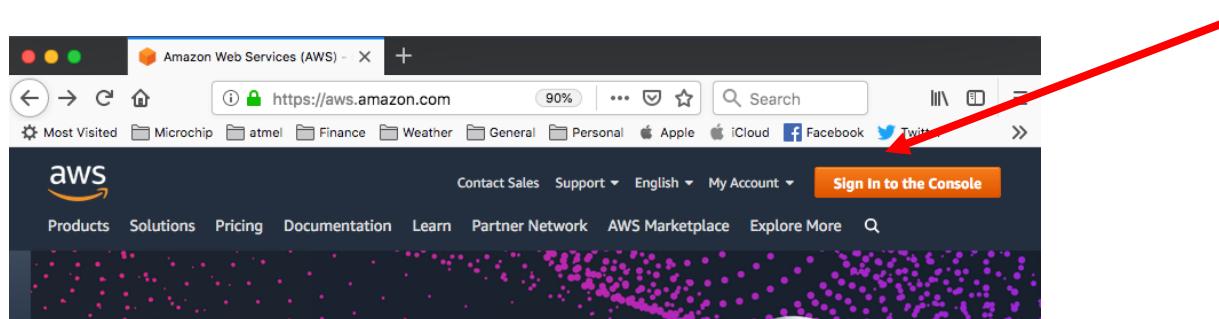


### Passo 1 Entrar na sua pagina da AWS.

1.0 Abra um navegador e entre na URL disponibilizada para vocês,  
<https://masters2019-classXY.signin.aws.amazon.com/console>



1.1 Você também pode acessar pela pagina principal usando o botão “Sign in”



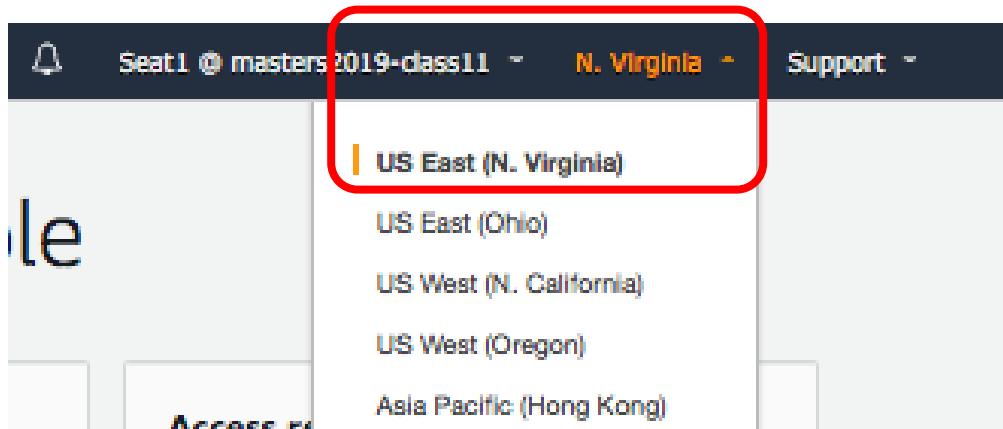
1.2 O Account ID já deveria estar preenchido. Entre com o “IAM user name” e “Password” disponibilizado para vocês e clique em ‘Sign in’

The screenshot shows the AWS sign-in page. It has three input fields: "Account ID or alias" containing "masters2019-class11", "IAM user name" containing "Student1", and "Password" containing "\*\*\*\*\*". Below the fields is a large blue "Sign In" button. At the bottom of the page is a link "Sign-in using root account credentials". Red annotations are present: an arrow points to the "masters2019-class11" field with the text "Já preenchido"; another arrow points to the "Student1" field with the text "Entre com User & `Password`"; and a third arrow points to the "\*\*\*\*\*" field with the same text.



Conectar ao seu console AWS.

**1.3 Tenha certeza que você está usando a região US East (N. Virginia)**  
 Clique em “**Serviços**”, no menu digite “**IoT**” para filtrar o serviço “**IoT Core**”.



**Clique em Serviços**

**Enter IoT**

**Selezione IoT Core**

Serviço	Descrição
IoT 1-Click	Acione funções do AWS Lambda em dispositivos simples
IoT Analytics	Colete, pré-processa, armazene, analise e visualize dados de dispositivos de IoT
<b>IoT Core</b>	Conecte dispositivos à nuvem
IoT Device Defender	Conecte dispositivos à nuvem
IoT Device Management	Gerencie com segurança frotas de apenas um dispositivo ou milhões de dispositivos
IoT Events	Monitore frotas de dispositivos para alterações e acione alertas para responder



### Passo 2 Explorando serviço AWS IoT

2.1 Se a página abaixo apareceu, quer dizer que nenhuma configuração do AWS IoT ainda foi feita. Clique em “**Get Started**”.

Caso necessite mudar de idioma para português, é possível fazer na parte inferior esquerda.

A screenshot of the AWS IoT console homepage. The page features a central circular icon with three stylized human figures. Below it is the text "AWS IoT". A main paragraph describes the service: "AWS IoT is a managed cloud platform that lets connected devices - cars, light bulbs, sensor grids, and more - easily and securely interact with cloud applications and other devices." A prominent blue button labeled "Get started" is centered below the text. At the bottom of the page, there is a footer bar with icons for Feedback, English (US), Privacy Policy, and Terms of Use. Two red arrows point to the "English (US)" language selection button and the "Get started" button.



## Passo 2 Explorando interface AWS IoT

**2.2** Na parte esquerda do navegador, é possível ver diversos menus, segue um pequeno resumo dos menus que utilizaremos no lab:

- **Gerencial** : Onde ficam as “**Coisas**” conectadas
- **Secure** : Armazenas os certificados dos dispositivos, políticas e CAs (Autoridades Certificadoras)
- **Test** : MQTT Client test. Iremos utilizar posteriormente para ver a comunicação com a placa.
- **Settings** : É possível confirmar o EndPoint (URL) que será usado no demo e no Lambda para se conectar ao AWS IoT

The screenshot shows the AWS IoT console interface. On the left, there is a vertical navigation menu with the following items:

- Monitoramento
- Incluir
- Gerenciar
- Greengrass
- Proteger
- Defender
- Agir
- Teste**
- Software
- Configurações
- Aprendizado

Red arrows point from the menu items to specific parts of the interface:

- An arrow points from "Gerenciar" to the "Gerenciar" section of the sidebar.
- An arrow points from "Proteger" to the "Proteger" section of the sidebar.
- An arrow points from "Teste" to the "Assinaturas" tab of the "Cliente MQTT" panel.
- An arrow points from "Configurações" to the "Endpoint personalizado" section of the "Configurações" panel.

The central area of the interface includes:

- Cliente MQTT** panel: Shows tabs for "Assinaturas" (selected), "Assinar um tópico", and "Publicar em um tópico". It also has sections for "Assinar" and "Tópico de assinatura".
- Configurações** panel: Shows the "Endpoint personalizado" section with the text: "Esse é seu endpoint personalizado que permite a conexão com o AWS nesse endpoint. Isso também é uma propriedade importante a ser inserida no IoT." and "Seu endpoint foi provisionado e está pronto para uso. Agora, você pode usá-lo." Below it is a "Endpoint" field containing the URL: "a3bfi0sj2cogdp-ats.iot.us-east-1.amazonaws.com".



## Passo 2 Explorando interface AWS IoT

### 2.3 Gerenciar em detalhes:

Clique em **Gerenciar -> Coisas**

Este é o local onde você encontrará todos as “**Coisas**” conectadas no seu AWS IoT. Nenhuma “coisa” deveria aparecer uma vez que você ainda não ligou a placa.

A screenshot of the AWS IoT Things management page. On the left, there's a sidebar with navigation links: Monitoramento, Incluir, Gerenciar (with Coisas selected), Tipos, Grupos de coisas, Grupos de faturamento, Trabalhos, Proteger, Agir, and Teste. The main content area features a central graphic of a yellow car inside a globe with a green windmill and a red thermometer. Below the graphic, the text "Você ainda não tem coisas" is displayed, followed by the subtitle "Coisa é a representação de um dispositivo na nuvem." At the bottom are two buttons: "Saiba mais" and "Registrar uma coisa".

### Apresentação de lista de coisas

A screenshot of the AWS IoT Things list page. The sidebar shows the same navigation as the previous page. The main area has a header "Coisas" with a search bar and a "Criar" button. Below is a table with columns "Nome" and "Tipo". One item is listed: "54c9f4cd15aa46f47f86f634c44a515c6e82970f" with the type "MICROCHIP-ZERO-TOUCH-KIT". There are also "Pesquisar itens", "Configurar indexação de frota", and "Lista" dropdown buttons.



## Passo 2 Explorando interface AWS IoT

### 2.4 Proteger em detalhes

Clique **Proteger -> Certificados**

Aqui você pode encontrar o certificado digital de cada um dos dispositivos conectados. O certificado digital válido é obrigatório para que um dispositivo se conectar ao AWS IoT. Nenhum certificado deve aparecer.



Monitoramento

Incluir

Gerenciar

**Proteger**

**Certificados**

Políticas

CAs

Aliases da função

Autorizadores

Agir

Teste



Você ainda não tem nenhum certificado

Os certificados ajudam as coisas a estabelecer uma conexão segura.

[Saiba mais](#)

[Criar um certificado](#)



### Passo 2 Explorando interface AWS IoT

#### 2.5 Proteger em detalhes:

Clique em **Proteger -> Políticas**

Aqui é possível ver todas as políticas individuais anexadas a cada certificado de dispositivo. A política prove certas autorizações para que o seu dispositivo possa ou não executar. Nenhuma política deve aparecer.

A screenshot of the AWS IoT Policies page. On the left, there is a sidebar with the AWS IoT logo and the following menu items: Monitoramento, Incluir, Gerenciar, Proteger (which is selected and highlighted in blue), Certificados, Políticas (which is also selected and highlighted in blue), CAs, Aliases da função, Autorizadores, Agir, and Teste. The main content area has a large circular icon with a red shield and a globe. Below the icon, the text "Você ainda não tem políticas" is displayed. Underneath this, a smaller text states: "As políticas do AWS IoT concedem às coisas permissão para acessar os recursos do AWS IoT (como outras coisas, tópicos MQTT ou sombras de coisas)." At the bottom of the content area are two buttons: "Saiba mais" (in a white box) and "Criar uma política" (in a blue box).

AWS IoT

Monitoramento

Incluir

Gerenciar

**Proteger**

- Certificados
- Políticas**
- CAs
- Aliases da função
- Autorizadores

Agir

Teste

Você ainda não tem políticas

As políticas do AWS IoT concedem às coisas permissão para acessar os recursos do AWS IoT (como outras coisas, tópicos MQTT ou sombras de coisas).

Saiba mais

Criar uma política



## Passo 2 Explorando interface AWS IoT

### 2.6 Proteger em detalhes.

Clique em **Proteger -> CAs**

Aqui é possível ver todos os CAs (Autoridade certificadora). O ATECC608 que está na placa foi assinado utilizando o CA que aparece na lista de CA. Caso não tenha nenhum CA na sua conta, peça ajuda. Sem o CA, não é possível autenticar que o dispositivo pode entrar na sua rede.



### Autoridades certificadoras

Monitoramento

Incluir

Gerenciar

Greengrass

**Proteger**

Certificados

Políticas

**CAs**

Aliases da função

Autorizadores

Pesquisar CAs por ID de certificado da CA



Nome

Status

8acb7be6bacf93b9aa10ab7b07a9e2fb010d33be64e27ecffdd2bd920a49906

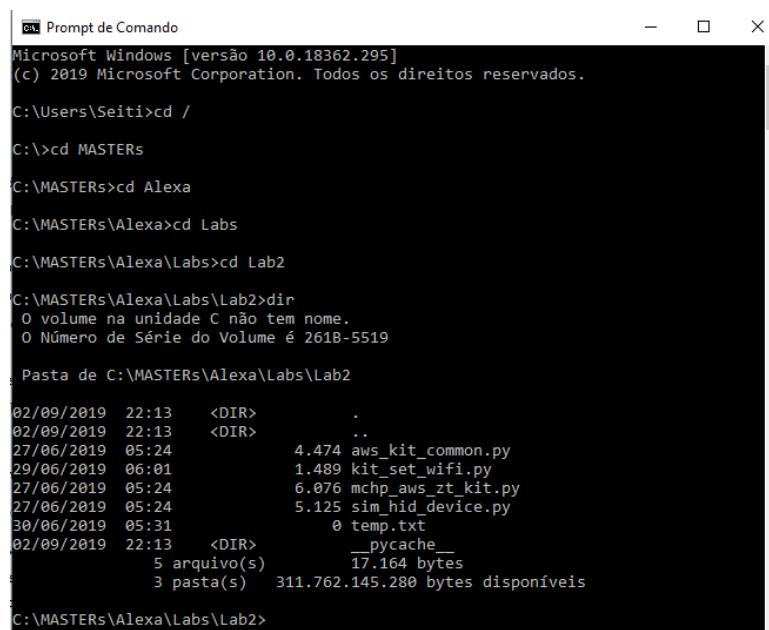
Ativo

## Passo 3 Comissionar o Wi-Fi da sua placa

**3.1 Agora**, vamos conectar o cabo USB à placa e comissionar com as informações corretas do WiFi.

**3.2** Abra o prompt de comandos e navegue até a pasta **c:\masters\Alexa\Labs\Lab2**.

**3.3** Digite o comando “**dir**” para ter certeza que está na pasta certa. Veja se encontra o arquivo **kit\_set\_wifi.py**



```
Prompt de Comando
Microsoft Windows [versão 10.0.18362.295]
(c) 2019 Microsoft Corporation. Todos os direitos reservados.

C:\Users\Seiti>cd /d C:\MASTERs\Alexa\Labs\Lab2
C:\MASTERs\Alexa>dir
O volume na unidade C não tem nome.
O Número de Série do Volume é 261B-5519

Pasta de C:\MASTERs\Alexa\Labs\Lab2

02/09/2019 22:13    <DIR>      .
02/09/2019 22:13    <DIR>      ..
27/06/2019  05:24           4.474 aws_kit_common.py
29/06/2019  06:01           1.489 kit_set_wifi.py
27/06/2019  05:24           6.076 mchp_aws_zt_kit.py
27/06/2019  05:24           5.125 sim_hid_device.py
30/06/2019  05:31           0 temp.txt
02/09/2019  22:13    <DIR>      __pycache__
                           5 arquivo(s)   17.164 bytes
                           3 pasta(s)  311.762.145.280 bytes disponíveis

C:\MASTERs\Alexa\Labs\Lab2>
```

**3.4** digite o comando

“**python kit\_set\_wifi.py --ssid ARTIMAR --password ARTIMAR2019**”

**python kit\_set\_wifi.py --ssid <SSID> --password <PWD>**

**NOTE1:** Confirmar a rede e a senha da sala de treinamento.

**NOTE2:** Antes do ssid e password, são 2 x o sinal de menos (-)

A mensagem abaixo deve aparecer no terminal quando as configurações forem feitas corretamente. O Led Vermelho da placa deverá estar piscando. **RESET** a placa pressionando o botão de reset conforme a figura abaixo.

```
C:\MASTERs\Alexa\Labs\Lab2>python kit_set_wifi.py --ssid ARTIMAR --password ARTIMAR2019

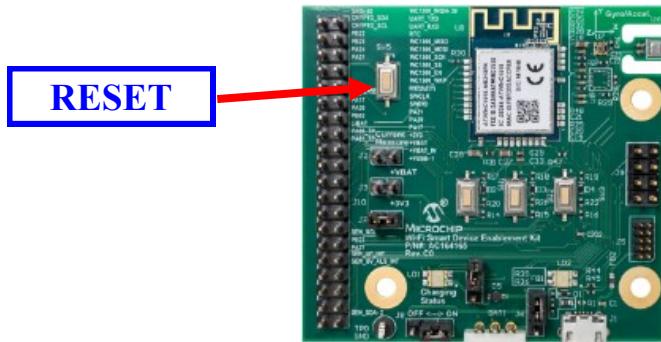
Opening AWS Zero-touch Kit Device

Initializing Kit
    ATECC508A SN: 0123B072E58A73A4EE

Setting WiFi Information

Done

C:\MASTERs\Alexa\Labs\Lab2>
```



**3.4** Após o RESET, O LED Azul irá piscar lentamente e depois de modo rápido. Neste momento, a placa estará se registrando ao AWS IoT utilizando o método JITR - Just in Time Registration.

Como dito anteriormente, só é permitido o acesso ao AWS IoT se o dispositivo possuir um certificado digital válido. A princípio, não existe nenhum certificado digital válido, como visto anteriormente, mas o certificado que está no dispositivo, foi assinado pelo CA (Autoridade Certificadora) válida. Desta forma, A função Lambda JITR executa e registra o novo certificado como sendo válido.

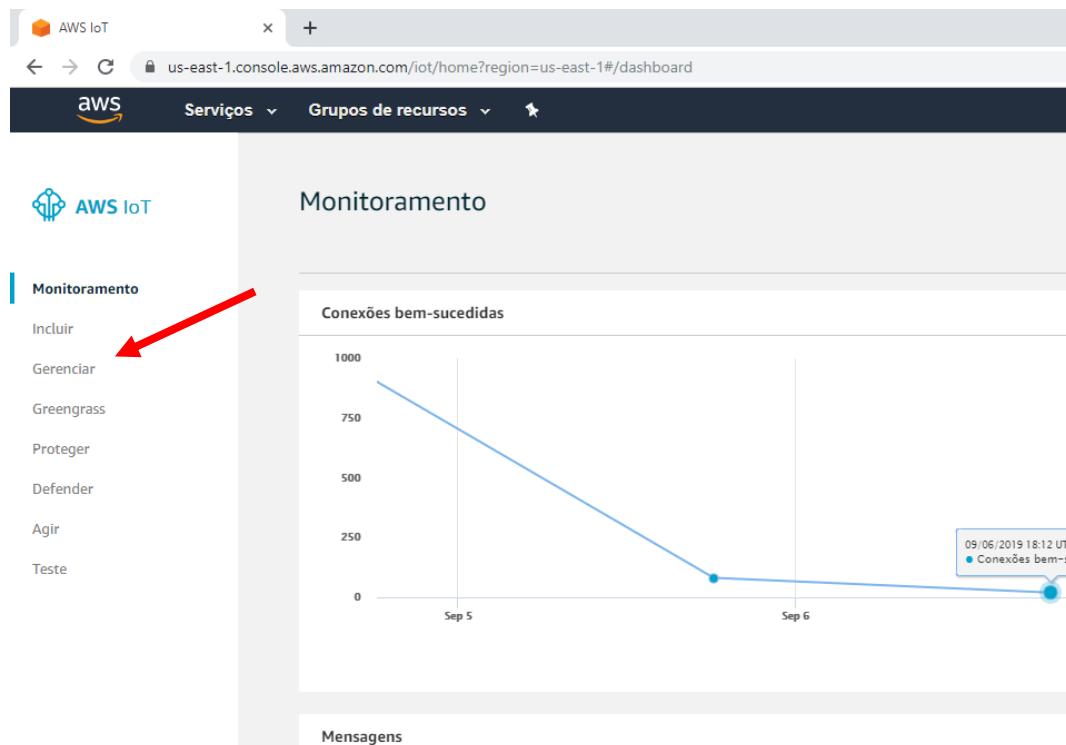
Uma vez que esta operação é feita, na próxima tentativa, o AWS IoT autoriza a conexão e os LED ficam estáticos. Provavelmente AZUL, mas podem estar de outra cor se foi definido assim em outro treinamento.

**SE DEPOIS DE 10 SEGUNDOS O LED CONTINUAR PISCANDO, RESET A PLACA**



**Passo 4 Verifique que a sua placa está conectada e enviadndo dados.**

### 4.1 Clique em **Gerenciar** no menu esquerdo



**Talvez seja necessário atualizar a tela para que o gráfico com numero de conexões apareça. É possível fazer esta atualização usando o botão ao lado**





## Passo 5 Confirmar que o seu dispositivo está listado na lista de *Coisas*

5.1 Clique na aba “**Coisas**” e selecione o seu dispositivo - Na tela deve aparecer apenas um dispositivo.

The screenshot shows the AWS IoT Things Hub interface. The left sidebar has a teal vertical bar and lists categories: Monitoramento, Incluir (with Gerenciar selected), Coisas (selected and highlighted in teal), Tipos, Grupos de coisas, Grupos de faturamento, Trabalhos, Greengrass, Proteger, Defender, Agir, and Teste. The main content area is titled 'Coisas' and contains a search bar with 'Pesquisar itens'. Below it is a card for a device: '54c9f4cd15aa46f47f8...' and 'MICROCHIP-ZERO-TOUCH-KIT'. A red arrow points from the 'Coisas' link in the sidebar to the device card in the main area.



## Passo 6 Verificar o numero ARN de seu dispositivo.

**6.1** ARN (*Amazon Resource Name*) é a identificação única dada a sua placa.

**6.2** Copie o ID do seu dispositivo. O ID é um número hexadecimal de 40 dígitos. Nós usaremos este ID no passo **10.2**.

The screenshot shows the AWS IoT Things console. At the top, there's a navigation bar with the AWS logo, 'Serviços' (Services) dropdown, 'Grupos de recursos' (Resource Groups) dropdown, and a star icon. Below the navigation, a breadcrumb trail shows 'Coisas > 54c9f4cd15aa46f47f86f634c44a515c6e82970f'. A red arrow points from the text '6.2' in the previous step to the device ID '54c9f4cd15aa46f47f86f634c44a515c6e82970f' which is highlighted with a blue box. The device details table below includes columns for 'Detalhes' (Details), 'Nome de recurso da Amazon (ARN) da coisa' (Amazon resource name of the thing), 'Segurança' (Security), 'Um Nome de recurso da Amazon (ARN) de uma coisa identifica de forma única essa coisa' (A thing's Amazon resource name identifies it uniquely), 'Grupos de coisas' (Thing groups), 'Arn:arn:aws:iot:us-west-2:280639739381:thing/54c9f4cd15aa46f47f86f634c44a515c6e82970f', 'Grupos de faturamento' (Billing groups), 'Sombra' (Shadow), 'Tipo' (Type), 'Interagir' (Interact), 'Atividades' (Activities), and 'Trabalhos' (Jobs). The 'Arn' value in the 'Groups of things' row is also highlighted with a blue box.

Detalhes	Nome de recurso da Amazon (ARN) da coisa
Segurança	Um Nome de recurso da Amazon (ARN) de uma coisa identifica de forma única essa coisa
Grupos de coisas	Arn:arn:aws:iot:us-west-2:280639739381:thing/54c9f4cd15aa46f47f86f634c44a515c6e82970f
Grupos de faturamento	
Sombra	Tipo
Interagir	<input type="text"/> microchip-zero-touch-kit
Atividades	
Trabalhos	



## Passo 7 Visualizar os dados de sombra de seu dispositivo

7.1 Clique em “Sombra” no menu esquerdo para ver as informações da sombra de seu dispositivo.

The screenshot shows the AWS IoT Device Shadow interface for a device with ARN: arn:aws:iot:us-west-2:280639739381:thing/54c9f4cd15aa46f47f86f634c44a515c6e82970f. The left sidebar has 'Sombra' selected. The main area shows the shadow document with the following state:

```
{  
  "reported": {  
    "button1": "up",  
    "button2": "up",  
    "button3": "up"  
  }  
}
```

A red arrow points from the 'Sombra' link in the sidebar to the shadow document area. Another red arrow points from the 'reported' object in the shadow document to the 'button1' field.



## Passo 8 Visualização dos dados sombra de seu dispositivo

**8.1** No campo “Estado da sombra” é possível ver o último estado reportado pelo dispositivo. Incluindo ver os estados dos botões, LEDs assim como ver a contagem incremental do número de PUBLICAÇÕES que o dispositivo enviou após a ultima autenticação na AWS. Estes dados em formato JSON é publicado no Broker MQTT no tópico específico que descreveremos abaixo.

### Shadow state:

```
1  {
2    "reported": {
3      "macAddr": "f8f005ecb6a3",
4      "uv": 630000,
5      "COUNT": 19, 
6      "BUTTON_1": 1,
7      "BUTTON_2": 1,
8      "BUTTON_3": 1,
9      "LED_R": 0,
10     "LED_G": 1,
11     "LED_B": 0,
12     "hum": 50
13   }
14 }
```



## Passo 9 Explore as abas “Proteger”

**9.1** Entre na aba “Certificados e certifique que agora existe um certificado válido. Este certificado foi inserido aos certificados válidos automaticamente pelo processo de Just In Time Registration. Descrito em detalhes no Anexo C.

É possível verificar na aba “Políticas” que o processo de Just In Time Registration também criou um política específica para o dispositivo que acabou de se conectar. Esta política permite que o nosso dispositivo possa se conectar, publicar e se subscrever ao broker MQTT.

The image contains two screenshots of the AWS IoT console interface. Both screenshots show the left navigation bar with the 'Proteger' tab selected, which includes options like Certificados, Políticas, CAs, Aliases da função, and Autorizadores.

**Screenshot 1: Certificados**

- The title is "Certificados".
- A search bar is labeled "Pesquisar certificados".
- A single certificate entry is shown: "065b107539544a4e7..." followed by "... ATIVO".

**Screenshot 2: Políticas**

- The title is "Políticas".
- A search bar is labeled "Pesquisar políticas".
- A single policy entry is shown: "ZTPolicy\_d4b2f82f8b..." followed by "...".

```
{"Version": "2012-10-17",
"Statement": [
  {
    "Effect": "Allow",
    "Action": [
      "iot:Connect"
    ],
    "Resource": [
      "arn:aws:iot:us-west-2:280639739381:client/094a043c83bd164b7f065a94f38b336dc192"
    ]
  },
  {
    "Effect": "Allow",
    "Action": [
      "iot:Publish",
      "iot:Receive"
    ],
    "Resource": [
      "arn:aws:iot:us-west-2:280639739381:topic/${iot:ClientId}/*",
      "arn:aws:iot:us-west-2:280639739381:topic/$aws/things/${iot:ClientId}/shadow/*"
    ]
  },
  {
    "Effect": "Allow",
    "Action": [
      "iot:Subscribe"
    ],
    "Resource": [
      "arn:aws:iot:us-west-2:280639739381:topicfilter/${iot:ClientId}/#",
      "arn:aws:iot:us-west-2:280639739381:topicfilter/$aws/things/${iot:ClientId}/sha"
    ]
  },
  {
    "Effect": "Allow",
    "Action": [
      "iot:UpdateThingShadow",
      "iot:GetThingShadow"
    ],
    "Resource": [
      "arn:aws:iot:us-west-2:280639739381:topic/$aws/things/${iot:ClientId}/shadow/*"
    ]
  }
]
```



### Passo 10 Verifique as mensagens publicadas no Broker MQTT pelo dispositivo

**10.1** Clique em **Teste** (1) e entre na opção **Assinar um tópico** (2) para se subscrever em um tópico específico de seu Broker MQTT. Neste teste podemos subscrever em 2 tópicos:

1 - “\$aws/things/#” para visualizar mensagens publicadas por todos os dispositivos.

AWS IoT

Cliente MQTT ⓘ

Conectado como iotconsole-1568227239454-2

Monitoramento

Incluir

Gerenciar

Greengrass

Proteger

Defender

Agir

Teste

Assinaturas

Assinar um tópico

Publicar em um tópico

Assinar

Os dispositivos publicam mensagens MQTT em tópicos. Você pode usar esse cliente para assinar um tópico e receber essas mensagens.

Tópico de assinatura

\$aws/things/#

Assinar ...

Captura máxima de mensagens ⓘ

100

Qualidade do serviço ⓘ

2 - “\$aws/things/<DEVICE\_ID>/shadow/update” para verificar os dados publicados apenas por um dispositivo específico. **Neste caso o <DEVICE\_ID> precisa ser substituído pelo ID copiado no passo 6.**

Assinaturas	
Assinar um tópico	Assinar
Publicar em um tópico	Os dispositivos publicam mensagens MQTT em tópicos. Você pode usar esse cliente para assinar um tópico e receber essas mensagens.
	Tópico de assinatura
	\$aws/things/54c9f4cd15aa46f47f86f634c44a515c6
	Assinar ...



**10.2** A cada mensagem enviada pelo dispositivo 3 tópicos receberão dados.

- **“\$aws/things/<DEVICE\_ID>/shadow/update”**  
Tópico que o dispositivo envia os dados.
- **“\$aws/things/<DEVICE\_ID>/shadow/update/accepted”**  
Tópico que apresenta o dado que foi aceito para o mecanismo “sombra”
- **“\$aws/things/<DEVICE\_ID>/shadow/update/documents”**  
Tópico que apresenta a documentação completa dos dados, incluindo dados anteriores e dados desejados.

```

1 {
2   "message": "Hello from AWS IoT console"
3 }

```

\$aws/things/2f5554c82bdc77dc086cafaf1464c... Jul 5, 2018 9:31:33 AM -0400 Export View

\$aws/things/2f5554c82bdc77dc086cafaf1464c... Jul 5, 2018 9:31:33 AM -0400 Export View

\$aws/things/2f5554c82bdc77dc086cafaf1464c... Jul 5, 2018 9:31:33 AM -0400 Export Hide

We cannot display the message as JSON, and are instead displaying it as UTF-8 string.

```

{
  "state": {
    "reported": {
      "macAddr": "F8F005ecb805",
      "uv": 5873000,
      "COUNT": 842,
      "BUTTON_1": 1,
      "BUTTON_2": 1,
      "BUTTON_3": 1,
      "LED_R": 1,
      "LED_G": 1,
      "LED_B": 0
    }
  }
}

```

We cannot display the message as JSON, and are instead displaying it as UTF-8 string.

**10.3** No mesmo menu “TESTE” é possível enviar dados aos tópicos sobescritos. Assim podemos enviar dados para o dispositivo para ver se o mesmo está recebendo dados do tópico subscrito.

Assine o tópico “\$aws/things/<DEVICE\_ID>/shadow/update” caso ainda não esteja assinado.

Digite a mensagem JSON abaixo e clique em publicar.

```
{"state": {"desired": {"LED_R":1,"LED_G":1,"LED_B":1}}}
```

Note que agora o LED RGB de sua placa está com a cor BRANCA uma vez que os LEDs R,G e B estão ligados. Publique outras combinações para fixar o conceito.

The screenshot shows the AWS IoT MQTT Client interface. On the left sidebar, under the 'Teste' (Test) section, there is a red box labeled '1'. In the main area, there is a red box labeled '2' pointing to the 'Assinar um tópico' (Subscribe to a topic) button. Another red box labeled '3' points to the message content area where the JSON message is being typed. A red arrow points from the 'Publicar' (Publish) button at the top right towards the message content area. The message content area contains the following JSON:

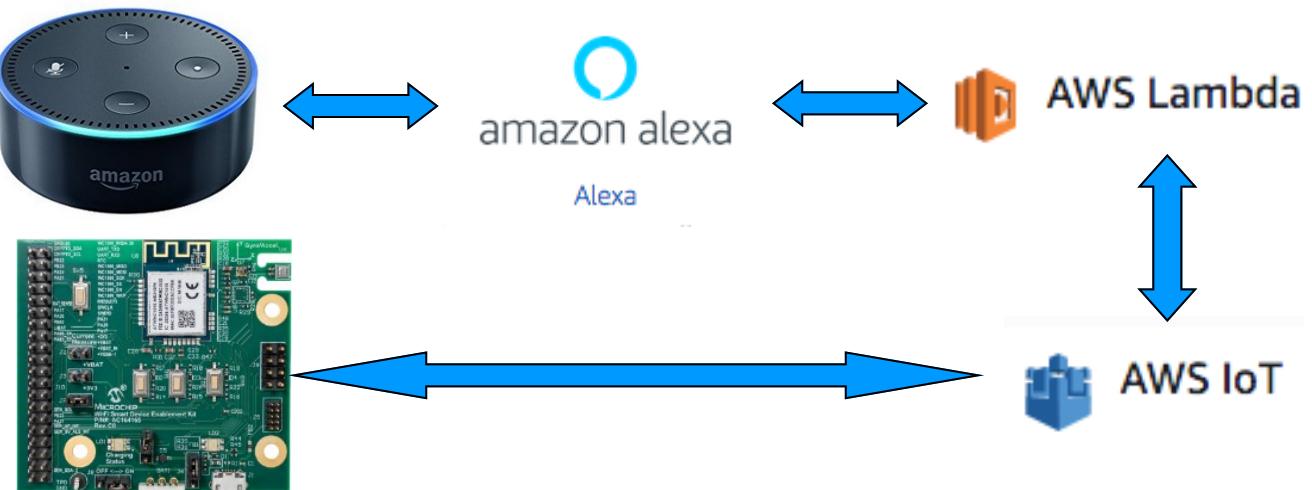
```
1 {"state": {"desired": {"LED_R":1,"LED_G":1,"LED_B":1}}}
```



**10.4** Veja nos tópicos Documents o estado completo dos dados principalmente o conteúdo de “desired”.

**Shadow state:**

```
{  
    "desired": {  
        "LED_R": 0,  
        "LED_G": 1,  
        "LED_B": 0  
    },  
    "reported": {  
        "macAddr": "f8f005acca8e",  
        "uv": 40656000,  
        "COUNT": 158,  
        "temp": 2854,  
        "hum": 26,  
        "pressure": 1001,  
        "BUTTON_1": 0,  
        "BUTTON_2": 0,  
        "BUTTON_3": 0,  
        "LED_R": 0,  
        "LED_G": 1,  
        "LED_B": 0,  
        "LED_INTENSITY": 100,  
        "Light": 1  
    }  
}
```



# Lab 3

## Controlando o seu dispositivo utilizando Alexa



### Proposta

Criar uma nova Skill para enviar comandos para o dispositivo pelo AWS IoT:

- Criar uma Skill personalizada.
- Testar a nova Skill utilizando o simulador Alexa em português e enviar comandos o dispositivo



### Requisitos

**Ambiente de desenvolvimento:**

Computador com acesso a internet e microfone

**Hardware:**

WiFi Sensor Board

**Arquivos do treinamento:**

C:\MASTERSAlexa\Labs\Lab3\...



### Objetivo

O objetivo deste LAB é enviar e receber dados da dispositivo utilizando comandos de voz da Alexa.

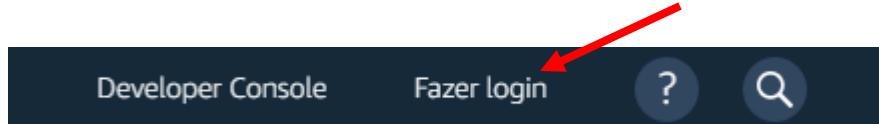


## Criar a Skill Alexa

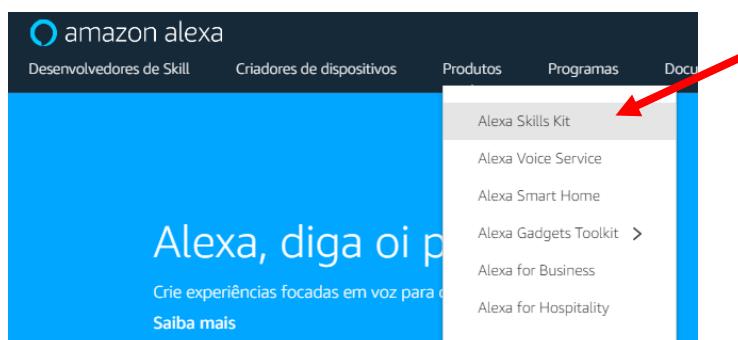
### Passo 1: Abra o sua conta Alexa Developer utilizada no LAB 1

1.1 Abra a pagina <https://developer.amazon.com/> .

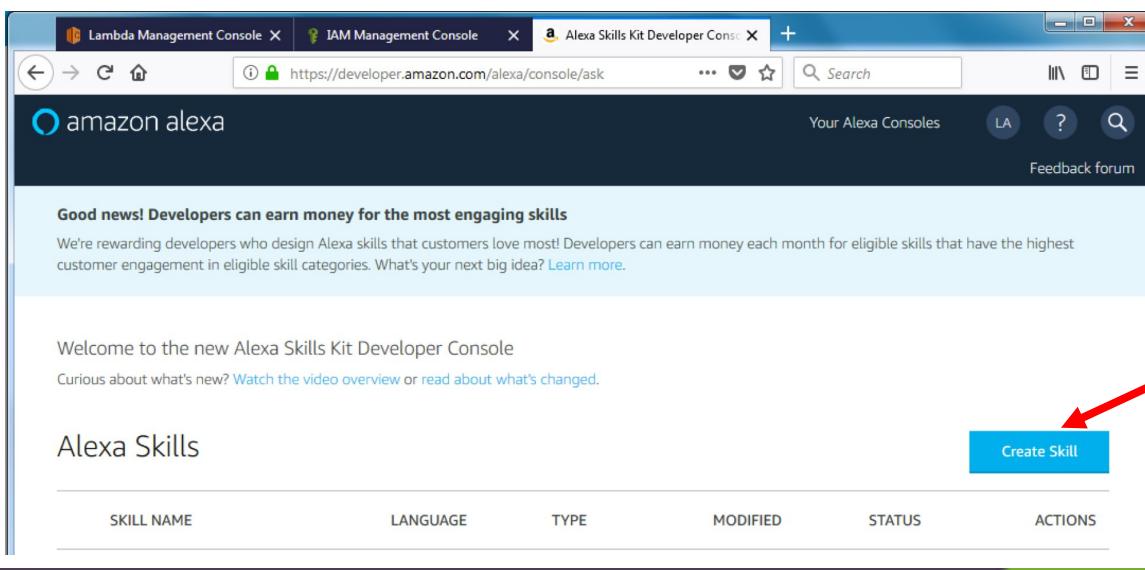
1.2 Clique em “Fazer login” no topo direito do navegador.



1.3 Entre no console Alexa diretamente pelo link <https://developer.amazon.com/alexa> ou navegando pelo menu Produtos->Alexa Skill Kit



1.4 Clique em “Create Skill”



**1.5** Defina o nome do skill como “**controle de dispositivos**” e o idioma, no caso “**Portuguese (BR)**”, selecione “**Custom**” de clique em “**Create skill**”

Create a new skill

Skill name

controle de dispositivos

24/50 characters

Default language

Portuguese (BR)

More languages can be added to your skill after creation

### 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.

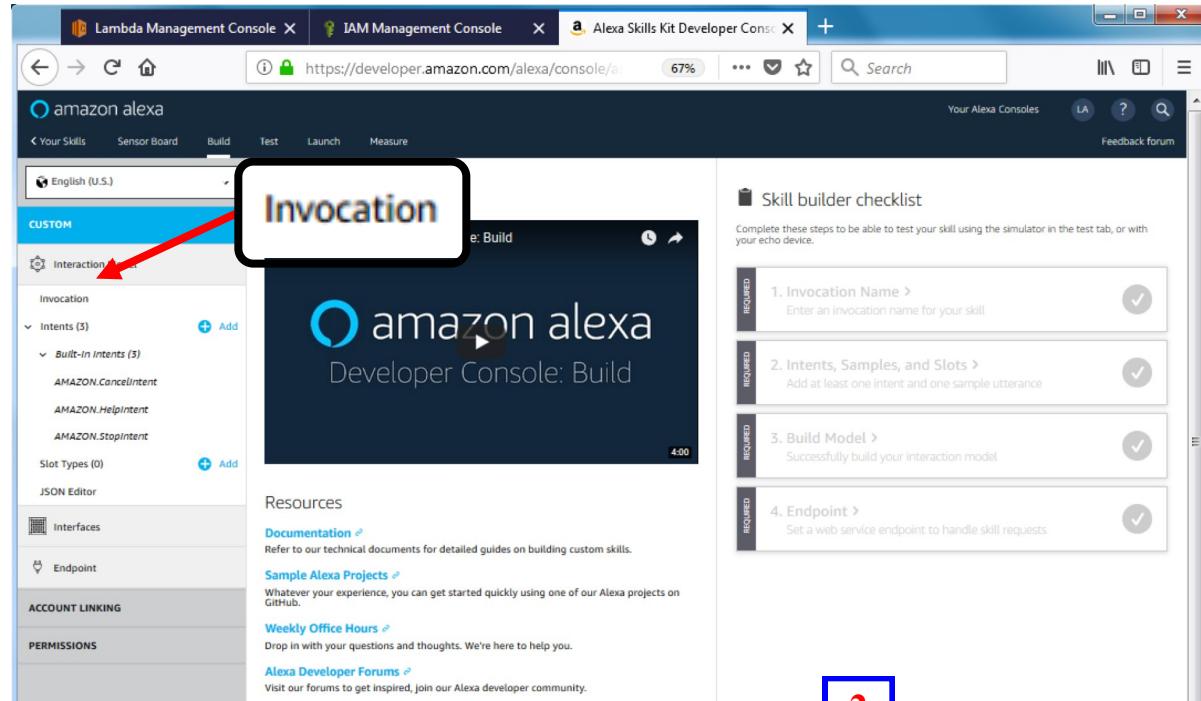
Flash Briefing

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

"Alexa, quais são as notícias?"

## Passo 2: Nome de Invocação

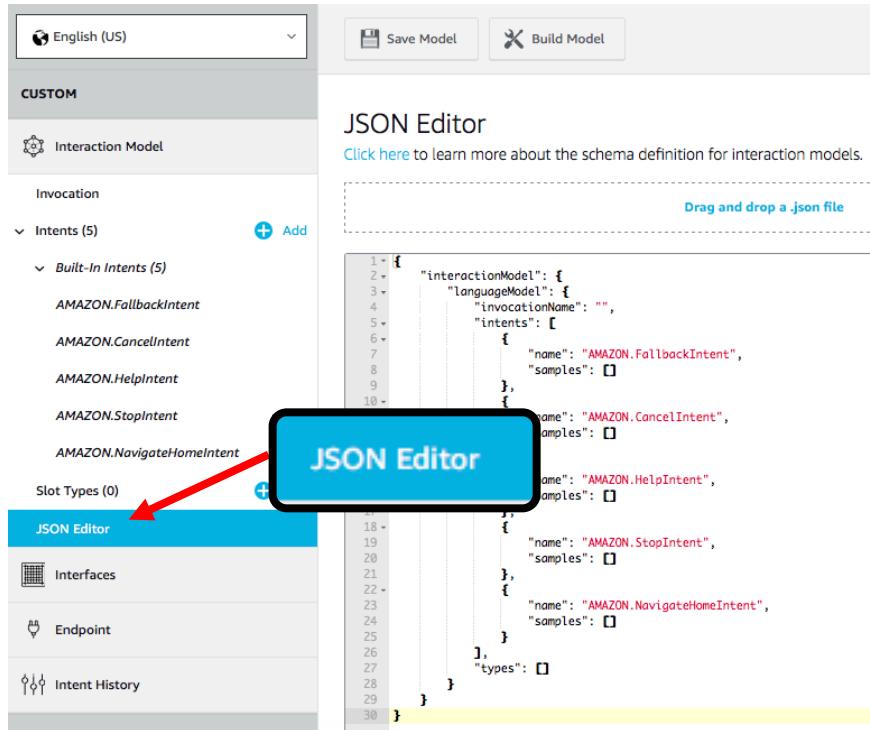
**2.1** Clique em “Invocation” do lado esquerdo do navegador e escolha o nome “controle de dispositivos” para ser utilizado como palavra de invocação de sua skill.



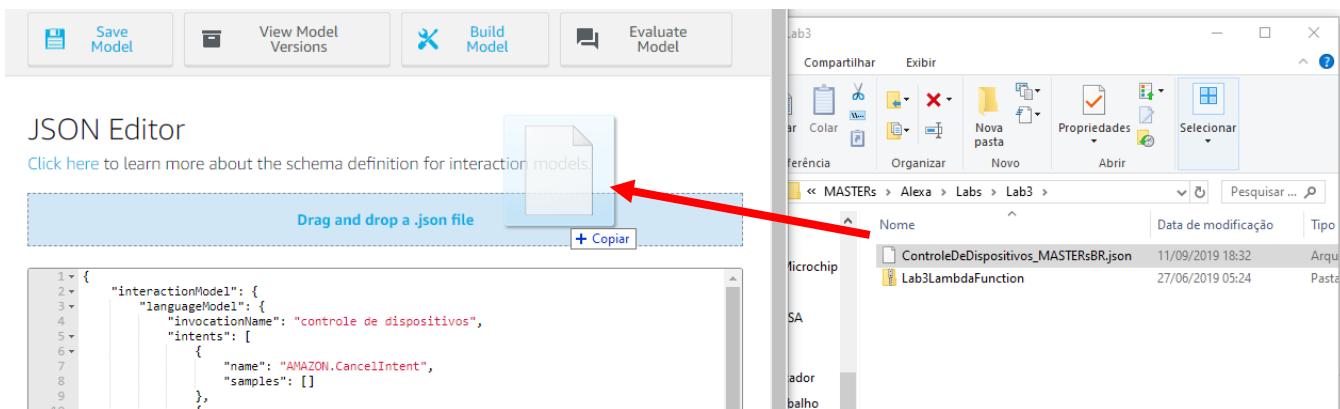
The screenshot shows the 'Invocation' section of the Alexa Skills Kit Developer Console. The left sidebar shows 'Portuguese (Brazil)' selected. Under 'CUSTOM', there's an 'Interaction Model' and the 'Invocation' section is highlighted with a blue bar. In the 'Invocation' section, there's a 'Save Model' button with a blue icon and the text 'Save Model'. Below it, there's a user interaction example: 'User: Alexa, ask daily horoscopes for the horoscope for Gemini' (with a red box labeled '1' around 'horoscope'). At the bottom, there's a 'Skill Invocation Name' input field containing the text 'controle de dispositivos' (with a red arrow pointing to it).

### Passo 3 Criação da Skill

**3.1** Clique em “**JSON Editor**” para acessar o modo de programação de skills por linha do comandos. Por padrão 4 comandos são criados durante a criação da Skill.



**2.4** Abra a pasta “C:\Masters/Alexa/Labs/Lab3”. Nesta pasta se encontra o arquivo **ControleDeDispositivos\_MASTERsBR.json**. Segue e arraste esta arquivo para o JSON Editor para copiar o conteúdo para dentro do Alexa Developer.



**3.2** Clique em “Save”. Note que após salvar, 5 novas Intents e 3 novos Slots foram adicionados ao menu.

The screenshot shows the AWS Lambda Interaction Model editor interface. On the left, there's a sidebar with a 'CUSTOM' tab at the top, followed by 'Interaction Model', 'Invocation', and 'Intents (9)'. Under 'Intents (9)', there are five custom intents: 'Information', 'LEDChange', 'lightState', 'getSensorStatus', and 'sensorType', each with a delete icon. Below them are four built-in intents: 'AMAZON.CancelIntent', 'AMAZON.HelpIntent', 'AMAZON.StopIntent', and 'AMAZON.NavigateHomeIntent'. To the right of the intent list is an 'Add' button. At the bottom of the sidebar is a 'Slot Types (3)' section with three entries: 'LIST\_OF\_STATE', 'LIGHT\_TYPE', and 'SENSOR\_TYPE', each with a delete icon. At the very bottom of the sidebar is a 'JSON Editor' button.

**Information:** Intent para informar ao usuário um pequeno tutorial das funções desta skill.

**LEDChange:** Intent para enviar o comando para mudar o estado do LED, podendo alterar a cor ou desligar ou ligar.

**getSensorStatus:** Intent para obter os dados de temperatura e umidade do dispositivo.

**LEDStatus:** Intent para obter o estado do LED RGB do dispositivo.

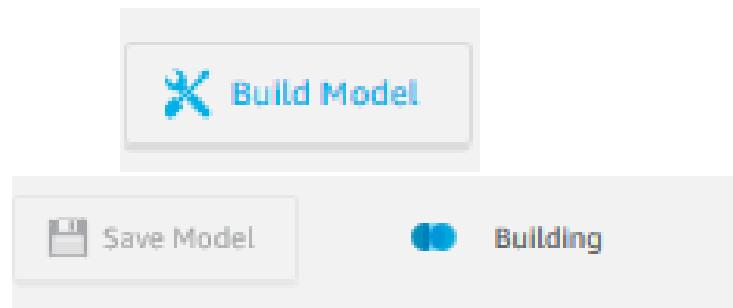
**HelloResponse:** Intent para responder a um agradecimento do usuário.

**LIST\_OF\_STATE:** Lista de possíveis estados do LED

**LIGHT\_TYPE:** Lista para definir diferentes modos de alterar o LED, no caso pode ser “led” ou “cor”

**SENSOR\_TYPE:** Lista de possíveis sensores, neste caso temperatura e umidade.

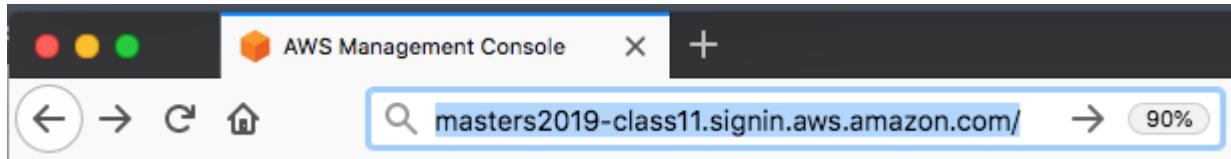
**3.3** Clique em “Build model”



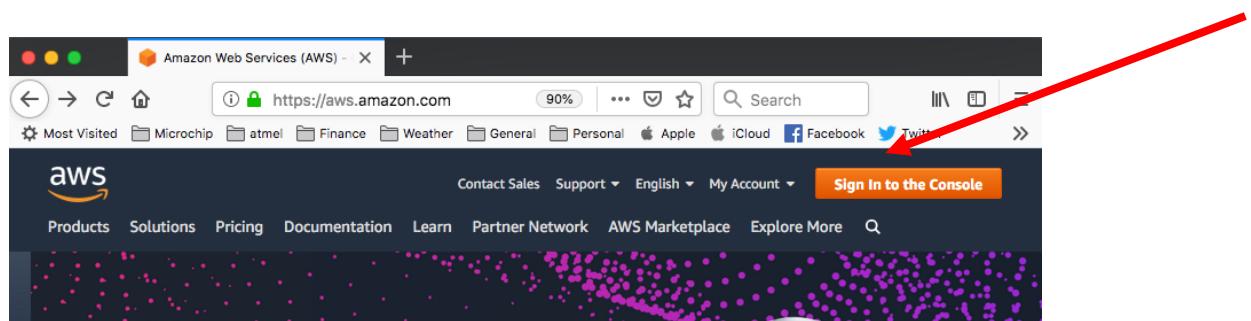
## Criando as respostas

**Passo 4 Como criar a função Lambda com as respostas.**

**4.1** Abra um navegador e entre na URL disponibilizada para vocês,  
<https://masters2019-classXY.signin.aws.amazon.com/console>



**4.2** Você também pode acessar pela pagina principal usando o botão “Sign in”



**4.3** O Account ID já deveria estar preenchido. Entre com o “IAM user name” e “Password” disponibilizado para vocês e clique em ‘Sign in’

Já preenchido

Entre com User & `Password`

Account ID or alias	masters2019-class11
IAM user name	Student1
Password	.....
<b>Sign In</b>	

[Sign-in using root account credentials](#)

**4.4 Tenha certeza que você está usando a região US East (N. Virginia)**  
Clique em “**Serviços**”, no menu digite “**Lambda**” para filtrar o serviço “**Lambda**”.

The screenshot shows the Alexa Developer Console interface. In the top navigation bar, there are three tabs: "AWS IoT", "Alexa Developer Console", and another partially visible tab. Below the tabs, the URL is "us-east-1.console.aws.amazon.com/iot/home?region=us-east-1#/test". The main content area has a sidebar on the left with links like "Histórico", "IoT Core", "Lambda", "Console Home", and "CloudWatch". On the right, there is a search bar with the word "lambda" typed in. A dropdown menu appears below the search bar, listing "Lambda" and "Amazon Lex". The "Lambda" item is highlighted and described as "Execute códigos sem se preocupar com servidores".

Nas contas utilizadas no treinamento deve haver apenas uma função Lambda criada, a função de Just In Time Registration, que utilizamos para registrar o nosso dispositivo no LAB2.

Clique em “Criar Função” para criar a função de resposta ao Alexa.

The screenshot shows the AWS Lambda Management Console. The browser address bar shows "https://console.aws.amazon.com/lambda/home". The main navigation bar includes "Services", "Resource Groups", and "N. Virginia". The left sidebar has "AWS Lambda" selected and "Functions" highlighted. A success message box says "Your Lambda function "sensor-board" was successfully deleted." Below it, the "Functions" table is shown with one item:

Function name	Description	Runtime	Code size	Last Modified
myStack-ZTLambdaJITR-1L56NB0WSNBV6		Python 3.6	1.1 kB	22 hours ago

#### 4.5 Na pagina de “Criar Função”:

1. Selecione “Criar do Zero”,
2. Entre com o nome da função “controleDeDispositivos”.
3. Expanda a aba “Escolher ou criar uma função de execução”
4. Selecione “Usar uma função Existente”
5. Selecione a função “AlexaBasicExecution Role”
6. Clique em “Criar Função”

The screenshot shows the 'Create Function' wizard in the AWS Lambda console. The top navigation bar includes 'Serviços', 'Grupos de recursos', and account information for Ricardo Seiti Yoshizaki in the Norte da Virgínia region. The main heading is 'Escolha uma das opções a seguir para criar a função.'

**Step 1:** The 'Criar do zero' option is selected (radio button highlighted with a blue box). A red arrow points from the text '1.' in the instructions above to this option.

**Step 2:** The 'Nome da função' input field contains 'controleDeDispositivos'. A red box highlights this field, and a red number '2' is placed over it. A red arrow points from the text '2.' in the instructions above to this field.

**Step 3:** The 'Tempo de execução' dropdown is set to 'Node.js 10.x'. A red box highlights this dropdown, and a red number '3' is placed over it. A red arrow points from the text '3.' in the instructions above to this dropdown.

**Step 4:** The 'Função' section is expanded, showing three options: 'Criar uma função com permissões básicas do Lambda' (radio button highlighted with a blue box), 'Usar uma função existente' (selected radio button highlighted with a blue box), and 'Criar uma função a partir da política da AWS templates'. A red box highlights the 'Usar uma função existente' option, and a red number '4' is placed over it. A red arrow points from the text '4.' in the instructions above to this option.

**Step 5:** The 'Função existente' dropdown is set to 'AlexaBasicExecutionRole'. A red box highlights this dropdown, and a red number '5' is placed over it. A red arrow points from the text '5.' in the instructions above to this dropdown.



Neste momento você criou a sua função Lambda em branco. Note que com a configuração que fizemos no passo anterior, esta função tem permissão de acessar o **AWS IoT e Amazon CloudWatch Logs**

Note também que ainda não tem um gatilho para iniciar a função Lambda

ⓘ A função **controleDeDispositivos** foi criada com êxito. Agora é possível alterar o código e a configuração dela. Para invocar sua função com um evento de teste, selecione "Testar". X

Lambda > Funções > controleDeDispositivos ARN - arn:aws:lambda:us-east-1:280639739381:function:controleDeDispositivos

**controleDeDispositivos** Controlar Qualificadores ▾ Ações ▾ Selecionar um evento d... ▾ Testar Salvar

Configuração Monitoramento

**Designer**

controleDeDispositivos

Layers (0)

+ Adicionar gatilho

AWS IoT

Amazon CloudWatch Logs

Os recursos aos quais a função tem acesso serão mostrados aqui

**Código da função** Informações

Tipo de entrada de código Editar código em linha ▾ Tempo de execução Node.js 10.x ▾ Manipulador index.handler

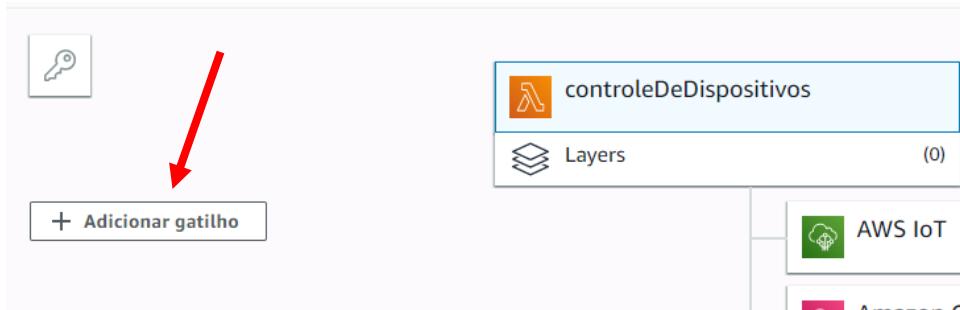
File Edit Find View Go Tools Window ⚙️

1-10



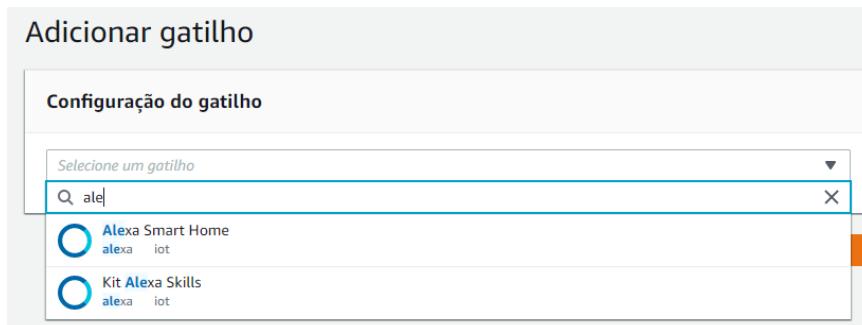
Agora vamos adicionar o gatilho a partir do **Alexa Skill Kit**

#### 4.6 Clique em “Adicionar gatilho”



#### 4.7 Selecione “Kit Alexa Skills”.

**CUIDADO NÃO SELECIONE O ALEXA SMART HOME**



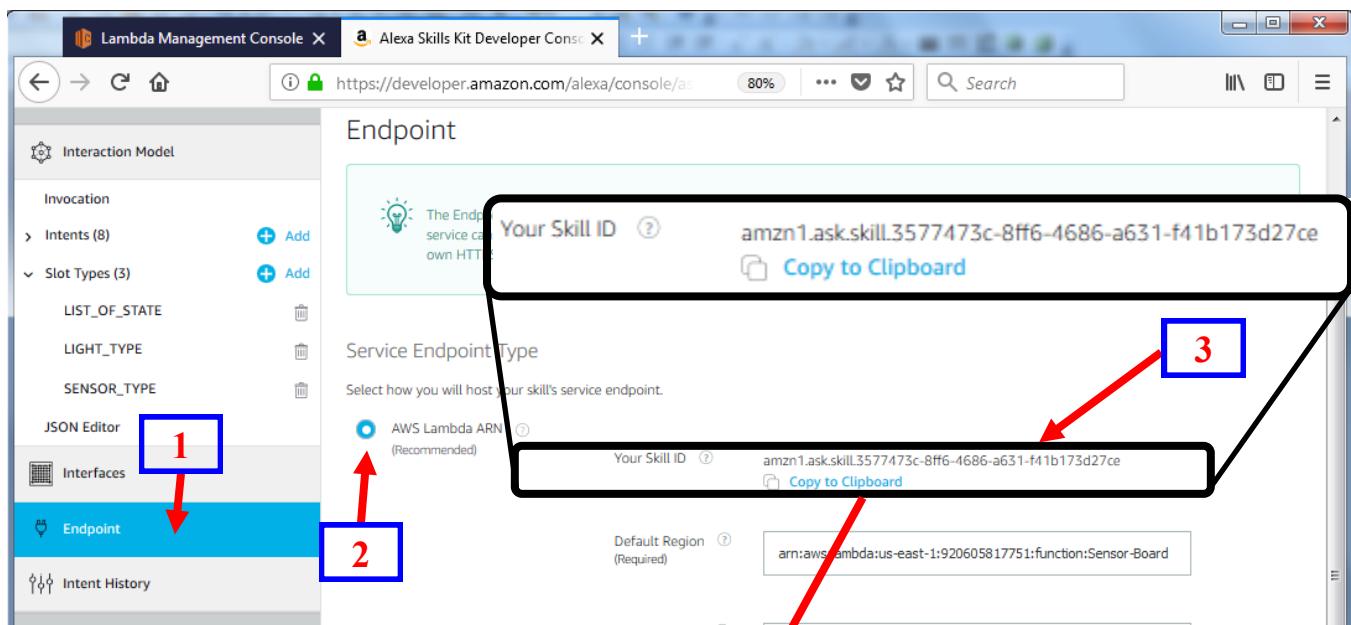
Deixe habilitado a verificação do ID da Habilidade e obtenha o ID da skill Alexa seguindo os próximos passos.

**4.8** Se você fechou a página do Alexa Developer, abra ela novamente em outra aba utilizando o endereço (<https://developer.amazon.com/alexam>).

Clique em “**ENDPOINT**”

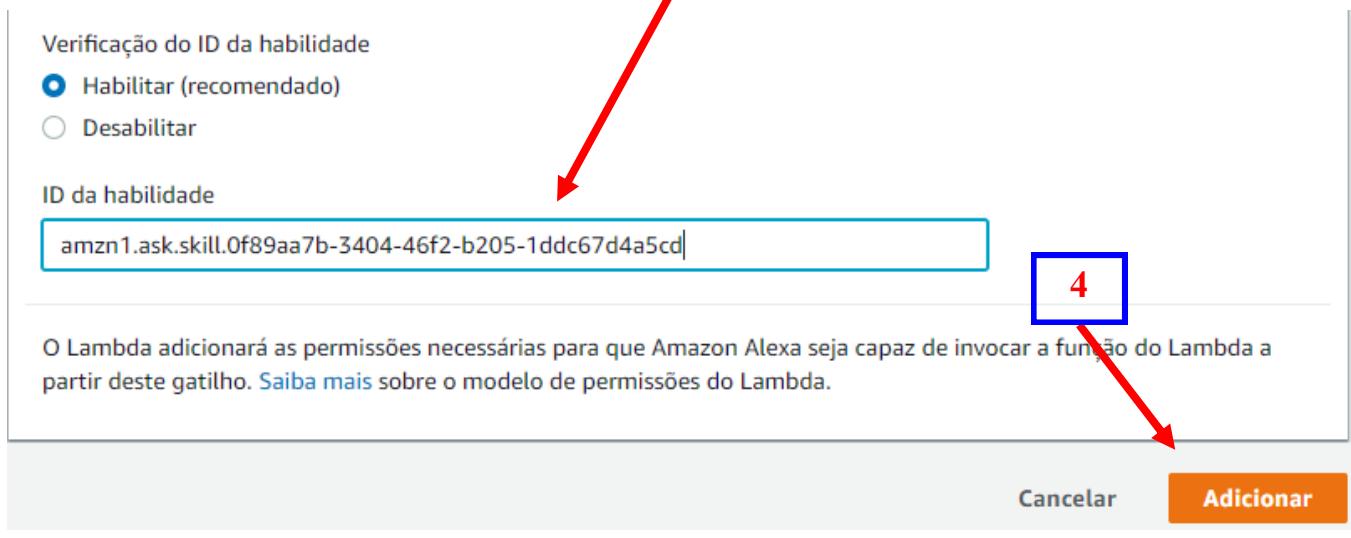
Selecione “**AWS Lambda ARN**”

Copie o “**Your Skill ID**”

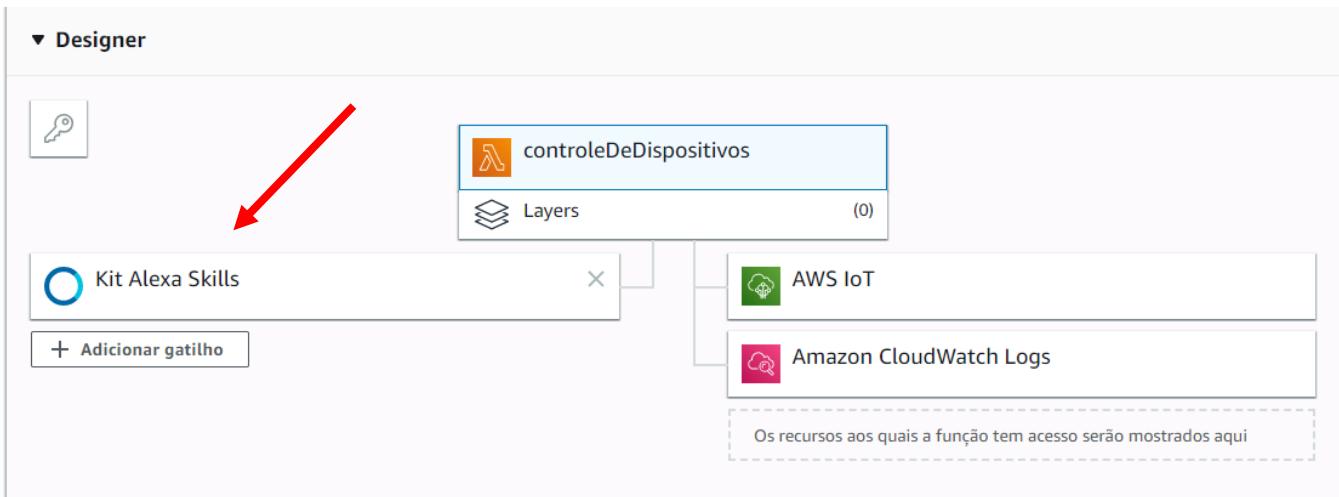


**4.9** Retorne ao console Lambda e cole o “**Your Skill ID**” que você copiou no passo anterior no campo “**ID da Habilidade**”

Clique em **adicionar**



#### 4.10 Verifique que agora existe um gatilho do “Kit Alexa Skills”



4.11 Agora vamos escrever a função Lambda propriamente dita. No topo da tela clique no ícone do nome da função. No nosso caso “**controleDeDispositivos**”. Logo abaixo do diagrama de blocos irá aparecer a janela “**Código de função**” nesta janela escreveremos o nosso código.

Tipo de entrada de código	Tempo de execução	Manipulador
Editar código em linha	Node.js 10.x	index.handler

```

File Edit Find View Go Tools Window
Environment controleDeDispositivos index.js
index.js
1 exports.handler = async (event) => {
2 // TODO implement
3 const response = {
4 statusCode: 200,
5 body: JSON.stringify('Hello from Lambda!'),
6 };
7 return response;
8 }
  
```

**4.12** Na opção “Tipo de entrada de Código”, selecione “Fazer upload de um arquivo .zip”.

Selecione o arquivo “Lab3LambdaFunction.zip” na pasta “C://MASTERS/Alexa/Labs/Lab3”.

- Salve as alterações no botão “Salvar”.

controleDeDispositivos

Controlar Qualificadores Ações Selecionar um evento d... Testar Salvar

▼ Designer

controleDeDispositivos (i) Alterações não salvas

Layers (0)

Kit Alexa Skills

+ Adicionar gatilho

AWS IoT

Amazon CloudWatch Logs

Os recursos aos quais a função tem acesso serão mostrados aqui

Código da função Informações

2

1

Tipos de entrada de código Fazer upload de um arquivo .zip

Tempo de execução Node.js 10.x

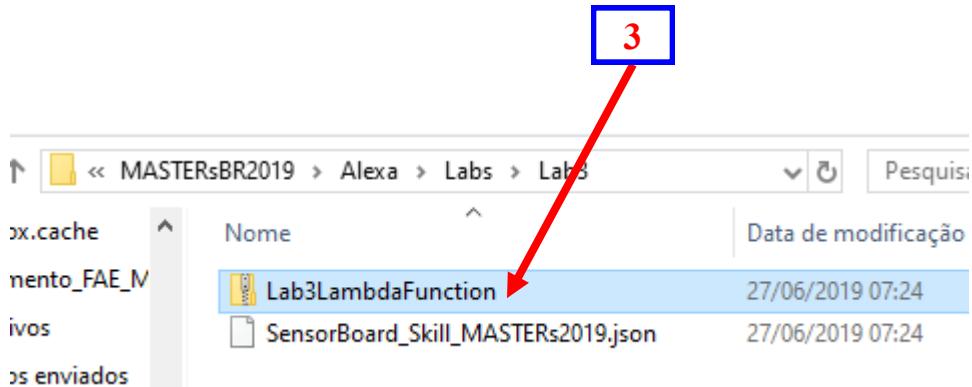
Manipulador Informações index.handler

Pacote e funções

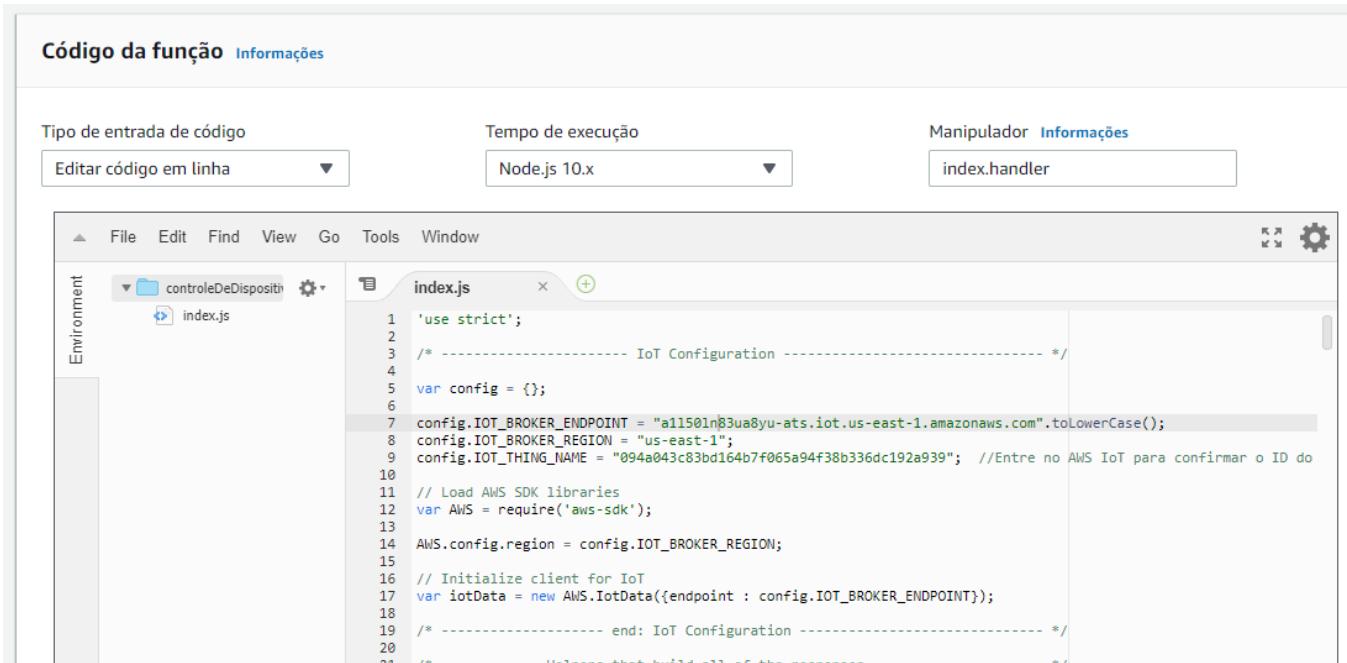
3

Fazer upload Lab3LambdaFunction.zip (3.9 kB)

Para arquivos maiores do que 10 MB, considere fazer upload usando o Amazon S3.



4.13 No que que agora o novo código fonte foi importado para o Lambda.



The screenshot shows the AWS Lambda function editor interface. At the top, there are three dropdown menus: 'Tipo de entrada de código' (Edit code online), 'Tempo de execução' (Node.js 10.x), and 'Manipulador' (index.handler). Below these, a toolbar includes File, Edit, Find, View, Go, Tools, and Window. On the left, there's an 'Environment' sidebar. The main workspace displays the 'index.js' file under a folder named 'controleDeDispositivo'. The code is as follows:

```

1 'use strict';
2
3 /* ----- IoT Configuration ----- */
4
5 var config = {};
6
7 config.IOT_BROKER_ENDPOINT = "a11501n83ua8yu-ats.iot.us-east-1.amazonaws.com".toLowerCase();
8 config.IOT_BROKER_REGION = "us-east-1";
9 config.IOT_THING_NAME = "094a043c83bd164b7f065a94f38b336dc192a939"; //Entre no AWS IoT para confirmar o ID do
10
11 // Load AWS SDK libraries
12 var AWS = require('aws-sdk');
13
14 AWS.config.region = config.IOT_BROKER_REGION;
15
16 // Initialize client for IoT
17 var iotData = new AWS.IotData({endpoint : config.IOT_BROKER_ENDPOINT});
18
19 /* ----- end: IoT Configuration ----- */
20
21 /**
 * Returns true if build all of the resources
 */

```

4.14 Para esta função Lambda enviar comandos para a sua placa, é necessário escrever no código 3 parâmetros que deve ser obtido no console do AWS IoT:

**IOT\_BROKER\_ENDPOINT** - endereço URL que as mensagens MQTT serão enviadas. Este endereço é único e específico para a sua conta AWS.

**IOT\_BROKER\_REGION** - Região AWS que o dado será tratado. No nosso caso “us-east-1”

**IOT\_THINGS\_NAME** - Identificação do dispositivo que será enviado, no caso o complemento do tópico MQTT que a mensagem será enviada. “\$aws/things/<DEVICE\_ID>/shadow/update”

**4.14** Para obter o endpoint, acesse a plataforma AWS IoT->configurações pelo link “<https://console.aws.amazon.com/iot/home?region=us-east-1#/settings>” ou pela aba “configurações” no canto inferior esquerdo do console AWS IoT

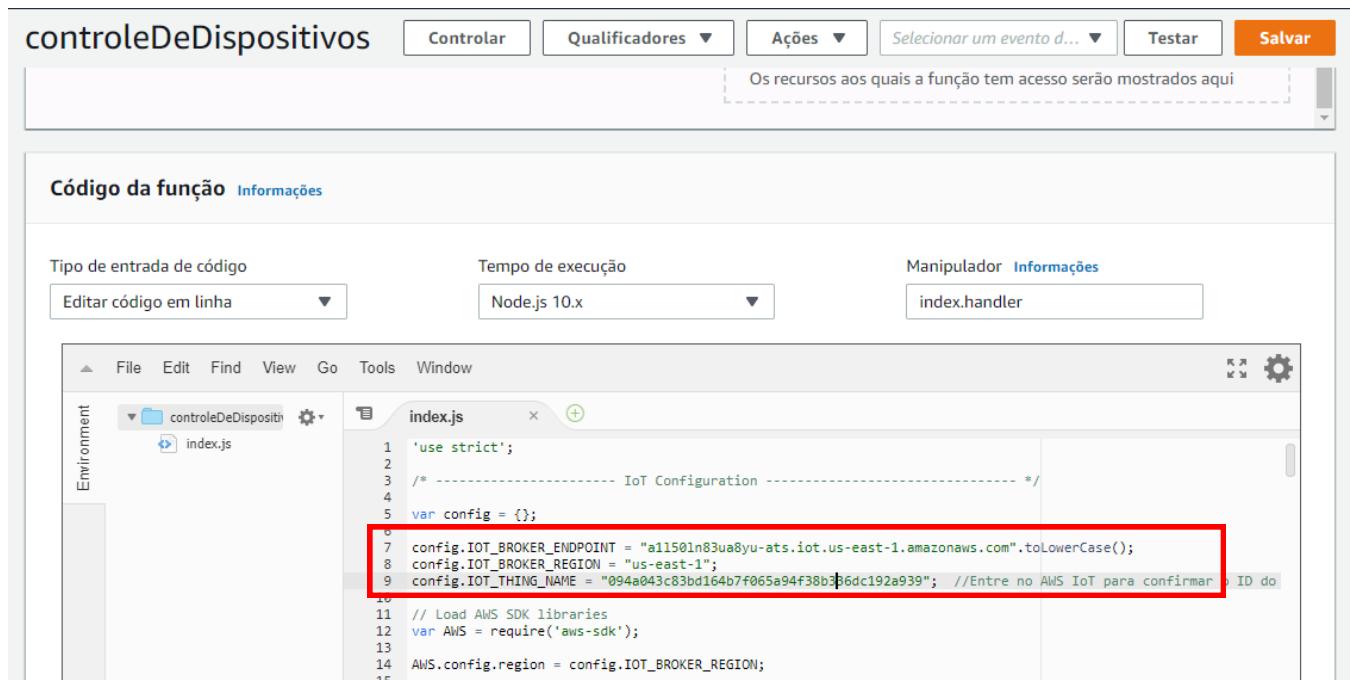
The screenshot shows the AWS IoT Settings page. On the left sidebar, under the 'Configurações' tab, there is a 'Endpoint personalizado' section. It contains a text input field with the value '54c9f4cd15aa46f47f86f634c44a515c6e82970f'. A red arrow labeled '1' points to the 'Configurações' tab in the sidebar. Another red arrow labeled '2' points to the endpoint value in the input field.

**4.15** O IOT\_THING\_NAME é o mesmo que utilizamos no LAB2 no passo 6.2

The screenshot shows the AWS IoT Things page. It displays a list of things, with one item highlighted: 'COISA' with the ARN '54c9f4cd15aa46f47f86f634c44a515c6e82970f'. A red arrow points to this ARN. Below the list, there is a table with columns 'Detalhes' and 'Nome de recurso da Amazon (ARN) da coisa'. The ARN 'arn:aws:iot:us-west-2:280639739381:thing/54c9f4cd15aa46f47f86f634c44a515c6e82970f' is shown in a dark box. The table also lists other details like 'Segurança' and 'Grupos de coisas'.

4.16 Altere o Código Lambda com as informações obtidas nos passos acima.

Após as alterações clique no botão “Salvar”



The screenshot shows the AWS Lambda function configuration interface. At the top, there's a navigation bar with tabs like 'Controlar', 'Qualificadores', 'Ações', 'Selecionar um evento d...', 'Testar', and a prominent orange 'Salvar' button. Below the navigation bar, there's a section titled 'Código da função' with a 'Informações' link. The main area contains three input fields: 'Tipo de entrada de código' set to 'Editar código em linha', 'Tempo de execução' set to 'Node.js 10.x', and 'Manipulador' set to 'index.handler'. Below these fields is a code editor window titled 'index.js'. The code is as follows:

```
1 'use strict';
2
3 /* ----- IoT Configuration ----- */
4
5 var config = {};
6
7 config.IOT_BROKER_ENDPOINT = "a11501n83ua8yu-ats.iot.us-east-1.amazonaws.com".toLowerCase();
8 config.IOT_BROKER_REGION = "us-east-1";
9 config.IOT_THING_NAME = "094a043c83bd164b7f065a94f38b36dc192a939"; //Entre no AWS IoT para confirmar o ID do dispositivo
10
11 // Load AWS SDK libraries
12 var AWS = require('aws-sdk');
13
14 AWS.config.region = config.IOT_BROKER_REGION;
```

A red rectangular box highlights the configuration parameters in lines 7, 8, and 9 of the code.



Agora a Função Lambda está pronta, com autorização para receber comandos pela Alexa e enviar dados para o AWS IoT. Agora precisamos informar a plataforma Alexa que envie dados para esta função Lambda específica.

4.17 No topo do função Lambda está o ARN da função que criamos. Aperte no botão copiar.

Lambda > Funções > controleDeDispositivos

**ARN - arn:aws:lambda:us-east-1:139739381:function:controleDeDispositivos**

**ARN - arn:aws:lambda:us-east-1:030505817751:function:sensor-board**

**controleDeDispositivos**

Configuração Monitoramento

Designer

controleDeDispositivos

Layers (0)

Kit Alexa Skills

AWS IoT

Amazon CloudWatch Logs

4.18 Entre novamente na página do Alexa skills

(<https://developer.amazon.com>):

Clique em “EndPoints

1. Cole o ARN no campo “default region”
2. Clique em “Save Endpoints”

amazon alexa

Your Alexa Consoles LA ? Feedback forum

English (Canada)

Save Endpoints **3**

**1**

Endpoint

arn:aws:lambda:us-east-1:477457934451:function:Sensor-Board

Service Endpoint Type

Select how you will host your skill's service endpoint.

AWS Lambda ARN

Your Skill ID **2**

amzn1.ask.skill.3577473c-8ff6-4686-a631-f41b173d27ce

Copy to Clipboard

Default Region (Required)

arn:aws:lambda:us-east-1:477457934451:function:Sensor-Board

4.19 Clique em “Invocation” e depois em “Build Model” para compilar e criar a skill.

Vai levar alguns minutos para compilar

English (CA)

Save Model

Build Model

CUSTOM

Interaction Model

Invocation

Invocation

Users say a skill's invocation name to begin an interaction with a p  
For example, if the invocation name is "daily horoscopes", users can

Build Started

You will be notified when the model build is complete.

Save Model

Building

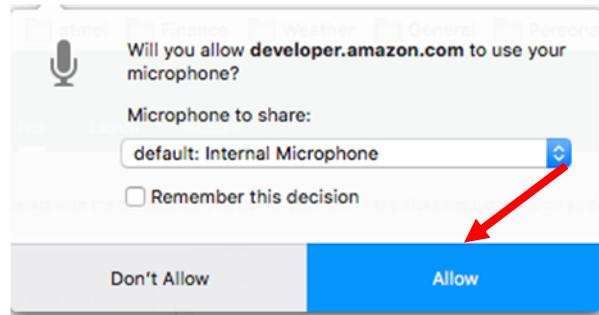
Build Successful

If you make any new changes, you will need to rebuild your model for them to take effect. - Saturday, Apr 7, 2018, 8:52 AM

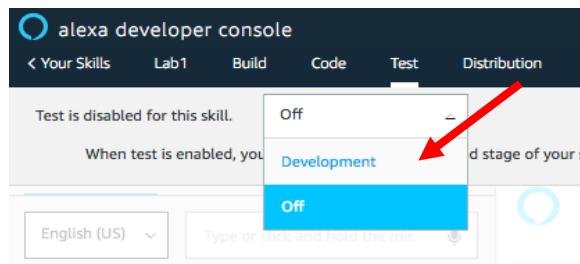
**4.20** Para testar a nova Skill, Clique na aba “Test” na parte superior do Browser:



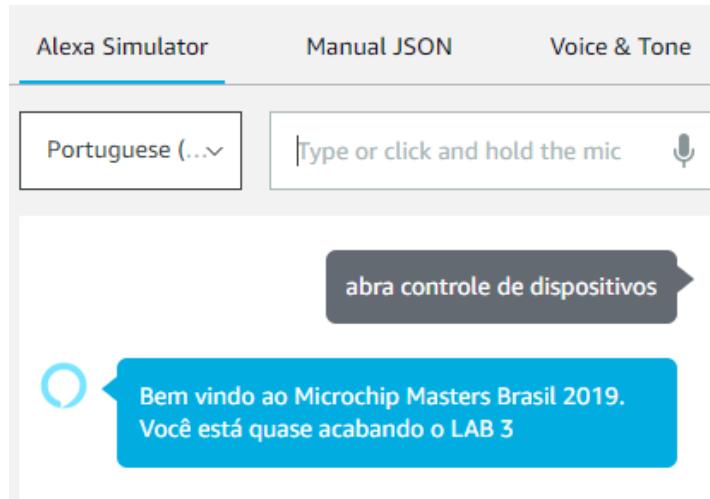
**4.21** Caso necessário, permita que a pagina da Amazon utilize o microfone:



**4.22** Habilite o teste em modo de desenvolvimento “development”



**4.23** Digite a mensagem “abra controle de dispositivos”. Ou pressione e segure o microfone ao lado da Caixa de texto e diga “abra controle de dispositivos”.



**4.24** Fale ou escreva “ajuda” para ter mais informações dos comandos que foram preparados para este treinamento

abra controle de dispositivos



Bem vindo ao Microchip Masters Brasil 2019.  
Você está quase acabando o LAB 3.

ajuda



É possível controlar a placa de sensores utilizando os comandos da Alexa. fale: mudar para vermelho, verde, azul ou amarelo para alterar a cor do LED. fale: ligar ou desligar LED para controlá-lo. fale: Qual a temperatura ou qual a umidade para receber os dados de temperatura ou umidade. fale: Qual a cor ou como está o LED para receber a informação da cor do LED.

**4.25** Digite ou escreva “mudar para amarelo”. De uma olhada nas mensagens JSON e escute a resposta.

Note nas flechas os caminhos da informação para a decisão da função Lambda.

The screenshot shows the Alexa Simulator interface with the following steps:

- User says "mudar para verde".
- JSON Input shows the intent request for "LEDChange".
- JSON Output shows the response card with "O LED está verde".
- User says "mudar para amarelo".
- JSON Input shows the intent request for "LEDChange".
- JSON Output shows the response card with "O LED está amarelo".
- User says "mudar para amarelo".
- JSON Input shows the intent request for "LEDChange".

The Skill I/O interface shows the JSON Input and JSON Output side-by-side, with red arrows pointing from the user input in the simulator to the corresponding JSON fields in the Skill I/O interface.

**Skills I/O**

**JSON Input**

```

52
53     }
54   }
55   }
56   }
57   "request": {
58     "type": "IntentRequest",
59     "requestId": "amzn1.echo-api.request.01f0c3d2-4e4a-401f-801f-0c3d24e4a41",
60     "timestamp": "2019-09-12T17:31:22Z",
61     "locale": "pt-BR",
62     "intent": {
63       "name": "LEDChange",
64       "confirmationStatus": "NONE",
65       "slots": {
66         "lightState": {
67           "name": "lightState",
68           "value": "verde",
69           "resolutions": {
70             "resolutionsPerAuthority": [
71               {
72                 "authority": "amzn1.echo-api.request.01f0c3d2-4e4a-401f-801f-0c3d24e4a41",
73                 "status": {
74                   "code": "ER_SUCCESS"
75                 },
76                 "values": [
77                   {
78                     "value": {
79                       "name": "verde",
80                       "id": "verde"
81                     }
82                   }
83                 ]
84               }
85             ]
86           }
87         }
88       }
89     }
90   }
91 }
```

**JSON Output**

```

1  {
2    "body": {
3      "version": "1.0",
4      "response": {
5        "outputSpeech": {
6          "type": "PlainText",
7          "text": "O LED está verde."
8        }
9      }
10     "card": {
11       "type": "Simple",
12       "title": "Sensor Board",
13       "content": "O LED está verde."
14     }
15     "reprompt": {
16       "outputSpeech": {
17         "type": "PlainText",
18         "text": "Diga outro comando"
19       }
20     }
21     "shouldEndSession": false,
22     "type": "_DEFAULT_RESPONSE"
23   }
24   "sessionAttributes": {
25     "desiredLEDStatus": "verde"
26   }
27 }
```

**index.js**

```

504 /**
505  * Function to handle the Intent Request
506 */
507 function onIntent(intentRequest, session, callback) {
508   console.log(`onIntent requestId = ${intentRequest.requestId}, session = ${session.sessionId}`);
509   const intent = intentRequest.intent;
510   const intentName = intentRequest.intent.name;
511   console.log(`inIntent => ${intentName}`);
512   // Dispatch to your skill's intent handlers
513   // Dispatch to your skill's intent handlers
514   if (intentName === 'LEDChange') {
515     setLEDState(intent, session, callback);
516   }
517   else if (intentName === 'WhatsLEDStatusIntent') {
518     getLEDStatusFromShadow(intent, session, callback);
519   }
520   else if (intentName === 'WhatsButtonStatusIntent') {
521     getButtonStatusFromShadow(intent, session, callback);
522   }
523   else if (intentName === 'WhatsSensorStatusIntent') {
524     getSensorStatusFromShadow(intent, session, callback);
525   }
526 }
```

```

119   function setLEDState(intent, session, callback) {
120   }
121   const cardTitle = intent.name;
122   const desiredLEDStateSlot = intent.slots.lightState;
123   let shadowLED_R = 0;
124   let shadowLED_G = 0;
125   let shadowLED_B = 0;
126   let repromptText = 'Diga outro comando';
127   let sessionAttributes = {};
128   const shouldEndSession = false;
129   let speechOutput = '';
130
131   if (desiredLEDStateSlot) {
132     const desiredLEDState = desiredLEDStateSlot.value;
133     sessionAttributes = createFavoriteLEDAttributes(desiredLEDState);
134
135     if ((desiredLEDState === 'amarelo')) {
136       shadowLED_R = 1;
137       shadowLED_G = 1;
138       shadowLED_B = 0;
139       speechOutput = "O LED está amarelo.";
140     }
141     else if ((desiredLEDState === 'vermelho')) {
142       shadowLED_R = 1;
143       shadowLED_G = 0;
144       shadowLED_B = 0;
145       speechOutput = "O LED está Vermelho.";
146     }
147     else if ((desiredLEDState === 'azul')) {
148       shadowLED_R = 0;
149       shadowLED_G = 1;
150       shadowLED_B = 1;
151       speechOutput = "O LED está Azul.";
152     }
153   }
154 }
```

**4.25** Abra também o AWS IoT -> Teste e subscreva ao tópico para ver os comandos que estão sendo enviados pelo dispositivo e também pela função Lambda:  
**“\$aws/things/<DEVICE\_ID>/#”**

No Document temos:

```
{
  "previous": {
    "state": {
      "desired": {
        "LED_R": 1,
        "LED_G": 1,
        "LED_B": 0,
        "Light": 1
      }
    },
    "reported": {
      "macAddr": "f8f00594d0bc",
      "uv": 43400000,
      "COUNT": 157,
      "hum": 35,
      "temp": 2582,
      "pressure": 940,
      "LED_R": 0,
      "LED_G": 1,
      "LED_B": 0,
      "Light": 1,
      "BUTTON_1": 0,
      "BUTTON_2": 0,
      "BUTTON_3": 0,
      "LED_INTENSITY": 100
    }
  },
  "version": 592
},
"current": {
  "state": {
    "desired": {
      "LED_R": 1,
      "LED_G": 1,
      "LED_B": 0,
      "Light": 1
    }
  },
  "reported": {
    "macAddr": "f8f00594d0bc",
    "uv": 43400000,
    "COUNT": 157,
    "hum": 35,
    "temp": 2582,
    "pressure": 940,
    "LED_R": 0,
    "LED_G": 1,
    "LED_B": 0,
    "Light": 1,
    "BUTTON_1": 0,
    "BUTTON_2": 0,
    "BUTTON_3": 0,
    "LED_INTENSITY": 100
  }
},
"version": 593
},
"timestamp": 1568310465
}
```

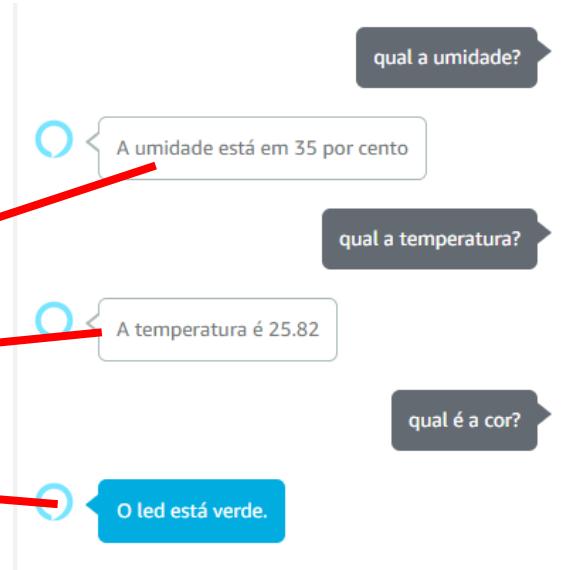
Previous: Dados que estavam gravados no estado anterior do shadow. Note que existe uma contagem que no caso apresentados está na versão 592.

Current: Dado enviado e computado agora. Note que a contagem de versão é 593.

Nest JSON temos:

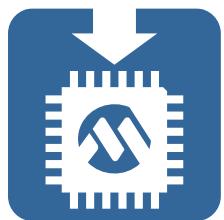
State Desired: Estados desejados por comandos, no nosso caso, enviamos para que o LED tenha a cor amarela. Acender os LEDs R e G.

State Reported: Estado que o dispositivo reportou. Neste caso vemos que o ultimo relatório de dados do dispositivo conta que o LED está Verde. No próximo relatório o deverá reportar que o LED alterou para amarelo. Desta relatório também é ligo os valores de temperatura e umidade.



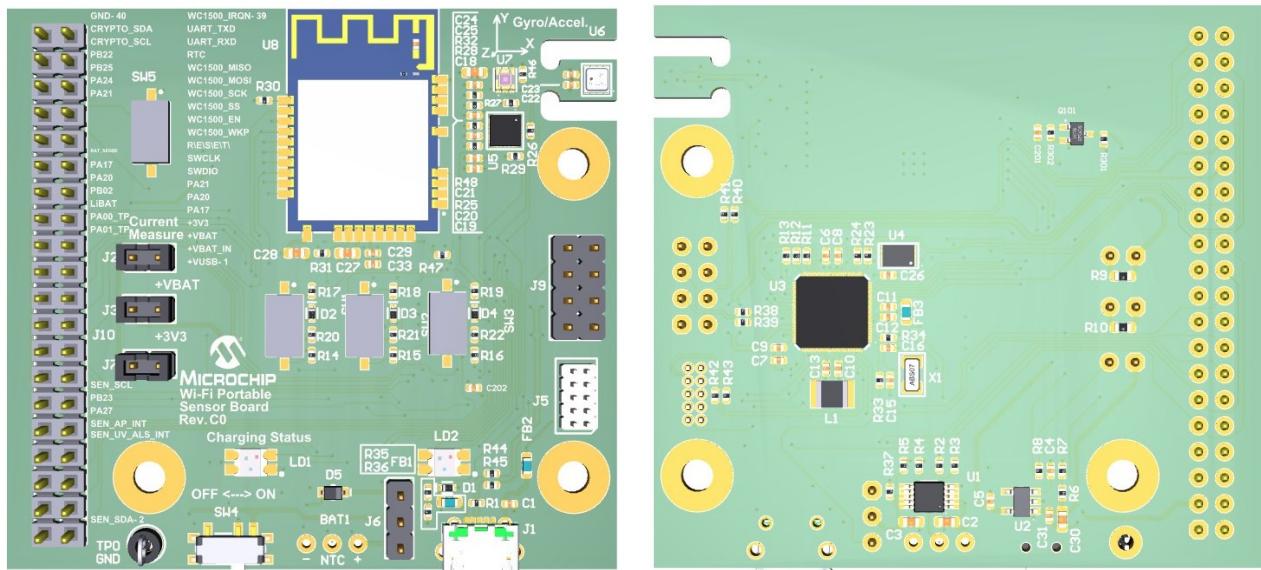
# **Anexo A**

*Informações de Hardware*





# Sensor Board



## Especificações

Microcontrolador	Microchip ATSAML21 32-bit ARM Cortex M0+
Ferramenta de Desenvolvimento	Atmel Studio e Atmel Software framework (ASF)
Módulo Wi-Fi	ATWINC150x0
Co-processador Criptográfico	Microchip ATECC608A Secure Element
Tensão de Operação	3.3V
Tensão de I/O	3.3V
Memória Flash interna	256 KBytes
SRAM	16 KBytes
Frequencia de Operação	48 MHz
Alimentação	5V USB e/ou bateria de Litio de 3.7V
Carregador de Bateria	500mA de corrente de carregamento
LED	RGB LED

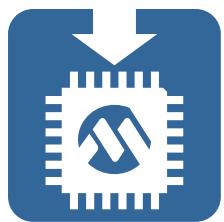
Sensor de temperatura

Umidade

Sensor de Luminosidade

# Anexo B

*Função Lambda do Exercício um*



Portuguese (...)

Type or click and hold the mic



abre exercicio um



Bem vindo a aula de Alexa do Microchip Masters Brasil. Me pergunte qual é a senha.

qual é a palavra secreta



A senha é masters. Me diga a senha e eu direi a mensagem secreta.

a senha é microchip



A senhamicrochip está incorreta. Tente novamente.

a senha é masters



Senha correta. A mensagem secreta é:  
Parabéns você concluiu o teste um criando a sua primeira skill. Bom treinamento!

**3.1** A função lambda que roda quando há uma requisição na Alexa está descrita abaixo.

```
'use strict';

/* ----- Helpers that build all of the responses ----- */

function buildSpeechletResponse(title, output, repromptText, shouldEndSession, cardType)
{
var response = {
    outputSpeech : {
        //type : 'PlainText',
        type : 'SSML',
        text : output,
        //ssml: "<speak>" + output + "<break time=\"3s\"/>" + output + "</speak>"
        ssml: "<speak>" + output + "</speak>"
    },
    card :{
        type : cardType,
        title : `Sensor Board`,
        content : title,
    },
    reprompt :{
        outputSpeech :
        {
            type : 'PlainText',
            text : repromptText,
            ssml: "<speak> " + output + " </speak>"
        },
    },
    shouldEndSession,
};
;
return response;
}

function buildResponse(sessionAttributes, speechletResponse)
{
var response = {
    version:
    '1.0',
    sessionAttributes,
    response : speechletResponse,
};
return response;
}
}
```

```

/* ----- Functions that control the skill's behavior ----- */
function getWelcomeResponse(callback)
{
    // If we wanted to initialize the session to have some attributes we could add those here.
    const sessionAttributes = {Session:"New session"};
    const cardTitle = 'Bem vindo ao Masters Brasil.\n\n Me pergunte a senha';
    const speechOutput = 'Bem vindo a aula de Alexa do Microchip Masters Brasil. Me pergunte qual é a senha.';

    // If the user either does not reply to the welcome message or says something that is not understood, they will
    // be prompted again with this text.
    const repromptText = ' Me pergunte qual é a senha.';
    const shouldEndSession = false;
    const cardType = 'Simple';

    callback(sessionAttributes, buildSpeechletResponse(cardTitle, speechOutput, repromptText, shouldEndSession,
    cardType));
}

function getHelpResponse(callback)
{
    // If we wanted to initialize the session to have some attributes we could add those here.
    const sessionAttributes = {Session:"New session"};
    const cardTitle = "";
    const speechOutput = 'A opção é me perguntar qual é a senha.';

    // If the user either does not reply to the welcome message or says something that is not understood, they will
    // be prompted again with this text.
    const repromptText = 'Me pergunte qual é a senha';
    const shouldEndSession = false;

    callback(sessionAttributes, buildSpeechletResponse(cardTitle, speechOutput, repromptText, shouldEndSession,
    "Simple"));
}

function secretKey(callback) {

    const sessionAttributes = {Session:"New session"};
    const cardTitle = '';
    const shouldEndSession = false;
    const speechOutput = 'A senha é masters. Me diga a senha e eu direi a mensagem secreta./';
    const repromptText = 'A senha é MASTERs';

    callback(sessionAttributes, buildSpeechletResponse(cardTitle, speechOutput, repromptText, shouldEndSession,
    "Simple"));
}

```

```
function secretMessage(intentSlot,callback) {  
    var cardTitle = "Get SENSOR";  
    const repromptText = null;  
    const sessionAttributes = {};  
    var shouldEndSession = false;  
    var speechOutput="";  
    const teste = "masters";  
    if (intentSlot.key.value == "masters")  
    {  
        shouldEndSession = true;  
        speechOutput = 'Senha correta. A mensagem secreta é: Parabéns você concluiu o teste um criando a sua  
primeira skill. Bom treinamento!';  
    }else{  
        speechOutput = 'A senha' + intentSlot.key.value + 'está incorreta. Tente novamente.';  
    }  
    callback({}, buildSpeechletResponse(cardTitle, speechOutput, null, shouldEndSession,"Simple"));  
}  
  
/* ----- Events ----- */  
/**  
 * Called when the session starts.  
 */  
function onSessionStarted(sessionStartedRequest, session)  
{  
    console.log(`onSessionStarted requestId = ${sessionStartedRequest.requestId}, sessionId = ${ses-  
sion.sessionId}`);  
}  
  
/**  
 * Called when the user launches the skill without specifying what they want.  
 */  
function onLaunch(launchRequest, session, callback)  
{  
    console.log(`onLaunch requestId = ${launchRequest.requestId}, sessionId = ${session.sessionId}`);  
  
    // Dispatch to your skill's launch.  
    console.log("Calling getWelcomeResponse");  
  
    getWelcomeResponse(callback);  
}  
  
function handleSessionEndRequest(callback) {  
    const cardTitle = 'Session Ended';  
    const speechOutput = 'Obrigado por testar a demonstração da Microchip. Tenha um bom treinamento!';  
    const shouldEndSession = true;  
    callback({}, buildSpeechletResponse(cardTitle, speechOutput, null, shouldEndSession));  
}
```

```

/**
 * Called when the user specifies an intent for this skill.
 */

function onIntent(intentRequest, session, callback)
{
    console.log("");
    const intent = intentRequest.intent;
    const intentName = intentRequest.intent.name;
    const intentSlot = intentRequest.intent.slots;
    console.log("inIntent =>",intentName);
    // Dispatch to your skill's intent handlers
    if (intentName === 'secretKey') {
        secretKey(callback);}
    if (intentName === 'secretMessage') {
        secretMessage(intentSlot,callback);}
    else if (intentName === 'AMAZON.HelpIntent') {
        getHelpResponse(callback);}
    else if (intentName === 'AMAZON.StopIntent' || intentName === 'AMAZON.CancelIntent') {
        handleSessionEndRequest(callback);}
    } else if(intentName === 'closeSession'){
        handleSessionEndRequest(callback);}
    else {throw new Error('Invalid intent')}
}

/**
 * Called when the user ends the session.
 * Is not called when the skill returns shouldEndSession=true.
 */
function onSessionEnded(sessionEndedRequest, session)
{
    // Setting this to true ends the session and exits the skill.
    console.log(`onSessionEnded requestId = ${sessionEndedRequest.requestId}, sessionId = ${session.sessionId}`);
    // Add cleanup logic here
}

```

```
/* ----- Main handler ----- */

// Route the incoming request based on type (LaunchRequest, IntentRequest, etc.) The JSON body of the re-
quest is provided in the event parameter.
exports.handler = (event, context, callback) =>
{
  try{
    console.log("\rStarting handler =>\r");
    //return;
    console.log("Events", event);
    console.log("Context", context);
    console.log("callback", callback);
    /**
     * Uncomment this if statement and populate with your skill's application ID to
     * prevent someone else from configuring a skill that sends requests to this function.
     */
    /*
    if (event.session.application.applicationId !== 'amzn1.echo-sdk-ams.app.[unique-value-here]') {
      callback('Invalid Application ID');
    }
    */

    if (event.request.type == 'LaunchRequest')
    {
      onLaunch(event.request,
        event.session,
        (sessionAttributes, speechletResponse) =>{
          console.log("Returning from onLaunch");
          //callback(null, buildResponse(sessionAttributes, speechletResponse));
          context.succeed(buildResponse(sessionAttributes, speechletResponse));
        });
    } else if (event.request.type == 'IntentRequest')
    {
      onIntent(event.request,
        event.session,
        (sessionAttributes, speechletResponse) =>{
          console.log("Returning from onIntent");
          console.log("buildResponse returns =>", buildResponse(sessionAttributes, speechletResponse));

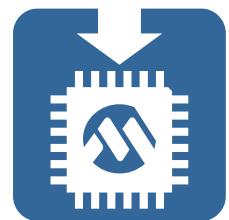
          //callback(null, buildResponse(sessionAttributes, speechletResponse));
          context.succeed(buildResponse(sessionAttributes, speechletResponse));

          console.log("Returning from callback");
        });
    } else if (event.request.type == 'SessionEndedRequest')
    {
      onSessionEnded(event.request, event.session);
      callback();
    }
  }

  catch(err)
  {
    callback(err);
  }
}
```

# Anexo C

*Como configurar conta AWS*



Nesta sessão iremos descrever a como fazer a configuração da conta AWS. Para o treinamento, todas as configurações foram executadas anteriormente.

No Link - <https://microchipdeveloper.com/iot:ztpk> - você encontra o passo a passo para instalar os softwares necessários para criar uma conta AWS e autorizar que seu computador acesse através do interface por linha de comandos:

## II. Software Installation

Sessão 2 descreve como instalar :

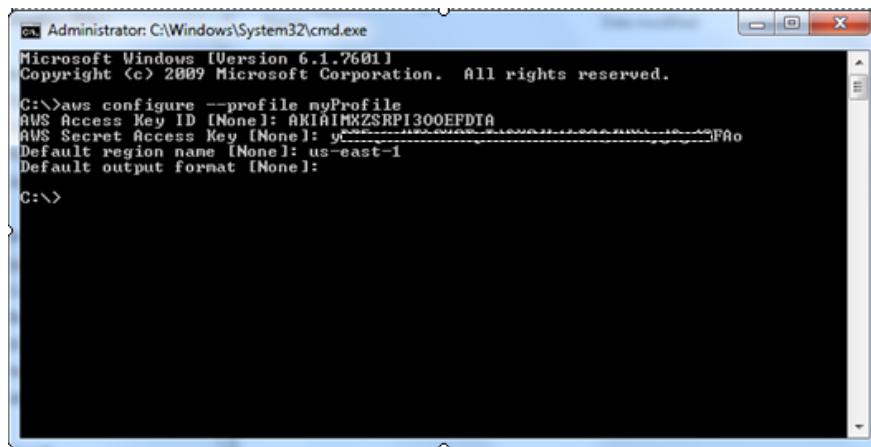
1. AWS CLI (Command Line Interface)
2. Terminal
3. Python 3
4. Pacotes Python necessários

## III. Create and Administer your own AWS Account

Sessão 3 descreve como criar uma conta AWS. Será necessário um cartão de crédito válido. Nesta sessão é descrito como fazer a programação do Just In Time Registration, criação das políticas e como autorizar o acesso pela interface CLI instalada anteriormente.

## IV. Configure AWS Credentials

Sessão 4 descreve como configurar o AWS CLI instalado no seu computador para que tenha acesso a sua conta AWS. Este acesso é feito por uma credencial criada na sessão 3 e depende de um *KEY ID* e uma *SECRET ACCESS KEY*. Além disso, é definido em qual região será trabalhado.



The screenshot shows a Windows Command Prompt window titled "Administrator: C:\Windows\System32\cmd.exe". The window displays the following command and its output:

```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

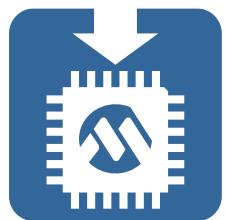
C:\>aws configure --profile myProfile
AWS Access Key ID [None]: AKIAIMXZSRPI300EFDTA
AWS Secret Access Key [None]: y
Default region name [None]: us-east-1
Default output format [None]:
```

## V. AWS IoT Just-In-Time Registration Setup

Sessão 5 descreve como criar um Autoridade Certificadora e como adicionar esta autoridade à lista de CAs válidos no AWS IoT. Esta sessão também descreve como adicionar a função Lambda para o JITR assim como as suas regras e políticas.

# Anexo D

*Projeto embarcado da placa Sensor Board*



Na pasta “SupportFiles/Projects/SensorBoard\_with\_Wifi\_Prov”, existem 2 projetos:

- saml21g18b\_sensor\_board\_demo\_ECC
- saml21g18b\_sensor\_board\_demo\_JITR

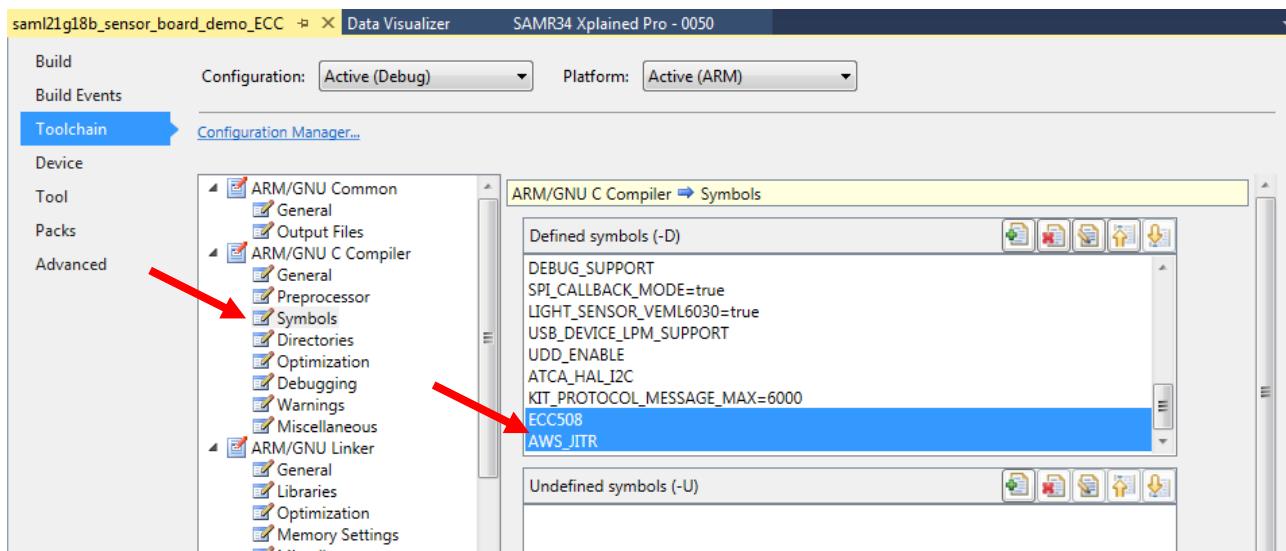
O primeiro projeto é utilizado para programar o ATECC608A com o certificado digital assinado por uma autoridade certificadora conhecida.

O segundo projeto possui a aplicação final em que utiliza o certificado digital previamente configurado para se conectar a AWS de forma segura. Neste projeto também está as funções de envio, recebimento e tratamento de dados da AWS IoT.

### 1 - Programando o ATECC608A.

1.1 - Abra o projeto “saml21g18b\_sensor\_board\_demo\_ECC.atsln”

1.2 - Em propriedades do projeto ->ARM/GNU C Compiler -> Symbols , certifique que os símbolos “ECC508” e “AWS\_JITR” estão presentes.



Estas diretivas são utilizadas para compilar partes do Código específicas para o comissionamento do ATECC608. Devido ao limitado espaço em flash da aplicação, é necessário dividir aplicação e comissionamento em 2 firmwares.

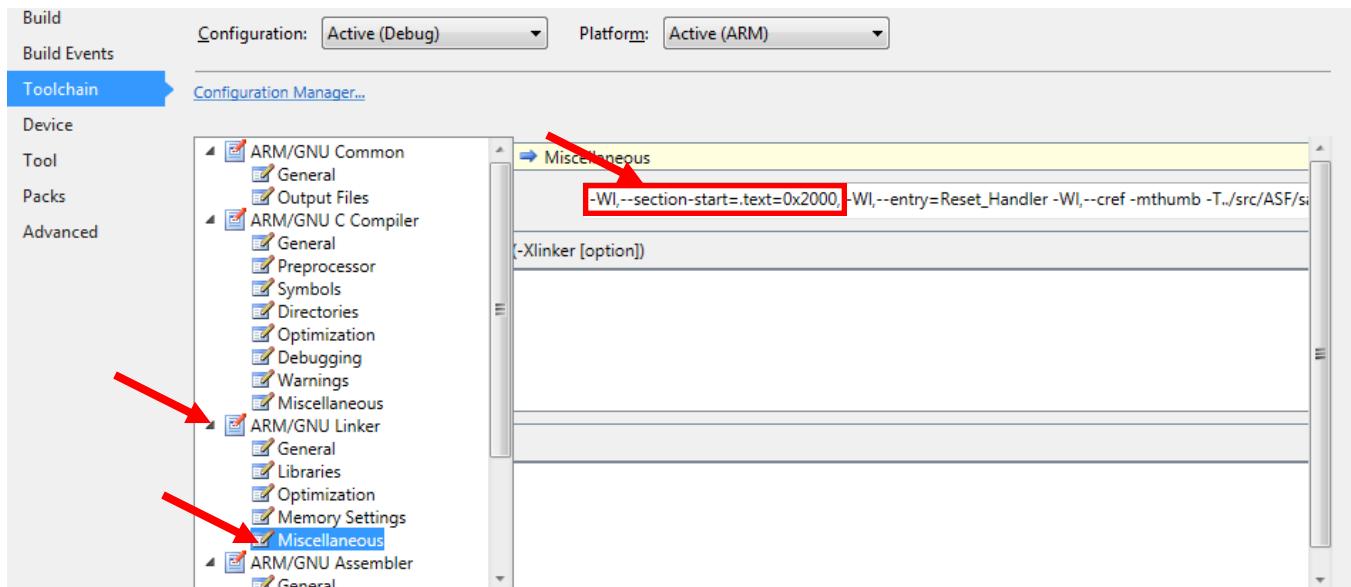
O símbolo “AWS\_JITR” habilita a interface USB\_HID que é utilizado como canal de comunicação com os scripts em PYTHON que fazer o comissionamento do ATECC608A.

O símbolo “ECC508” habilita a gravação do ATECC608A. Tanto os certificados válidos do AWS como as informações do WiFi são gravadas no ATECC608A. Enquanto estiver rodando esta parte do código o LED RGB da placa irá piscar a cor amarela.

Para facilitar a alteração do firmware, é gravado um bootloader de 8KBytes. Assim a aplicação deve iniciar a partir do endereço 0x2000. Esta configuração é feita nas propriedades de projeto -> ARM/GNU C Compiler -> Miscellaneous e adicionar a diretiva abaixo:

**-Wl,--section-start=.text=0x2000**

No Anexo E será discutido em detalhes o Bootloader gravado no endereço 0x0000.



Seguindo os passos do Anexo E, grave o firmware “saml21g18b\_sensor\_board\_demo\_ECC.bin” este firmware será utilizado apenas para fazer a programação do ATECC608A, não rodando nada da aplicação. Pode ser usado o arquivo HEX diretamente, sem a necessidade de compilar o código.

A Sequencia abaixo descreve os passos adicionais ao LAB2 para criar um CA, gerar um CA secundário, solicitar ou kit que envie o CSR (Certificate Signing Request) da placa, Assinar o CSR da placa com o CA secundário, gravar o certificado no ATECC608A, registrar o CA secundário como um CA válido na conta AWS e programar as credencias de WiFi.

1- Abra um prompt de comandos e navegue até c:\masters\Alexa\provisioning

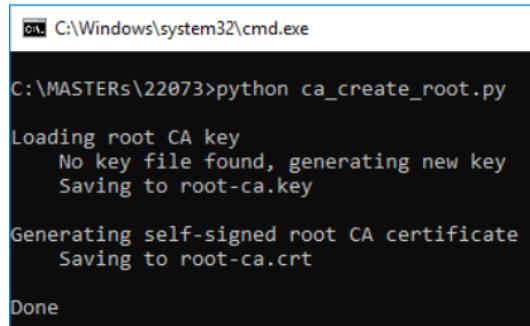
2 - Instalação de pacotes necessários:

Para fazer a instalação dos pacotes necessários, executar o comando abaixo:  
“**pip install requirements.txt**”

3 - Geração do Root CA:

Para rodar a função python para criar um par de chaves para se tornar a raiz de segurança do seu sistema, execute o comando abaixo:

“**python ca\_create\_root.py**”



```
C:\Windows\system32\cmd.exe
C:\MASTERS\22073>python ca_create_root.py
Loading root CA key
No key file found, generating new key
Saving to root-ca.key

Generating self-signed root CA certificate
Saving to root-ca.crt

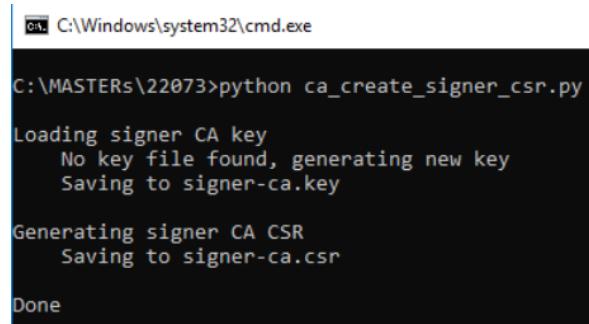
Done
```

**Esta forma de criação de Root CA deve ser usado apenas para demonstração. Já que a chave privada fica exposta. Para melhores práticas entre em contato com o seu suporte da Microchip.**

### 4 - Geração de certificados Assinantes:

Para gerar o par de chaves e o certificado destas chaves, execute os comandos abaixo. O primeiro comando gera o par de chaves e o CSR desta chave. O segundo comando utiliza a chave privada ROOT CA para criar o certificado da chave de assinante.

**“python ca\_create\_signer\_csr.py”**

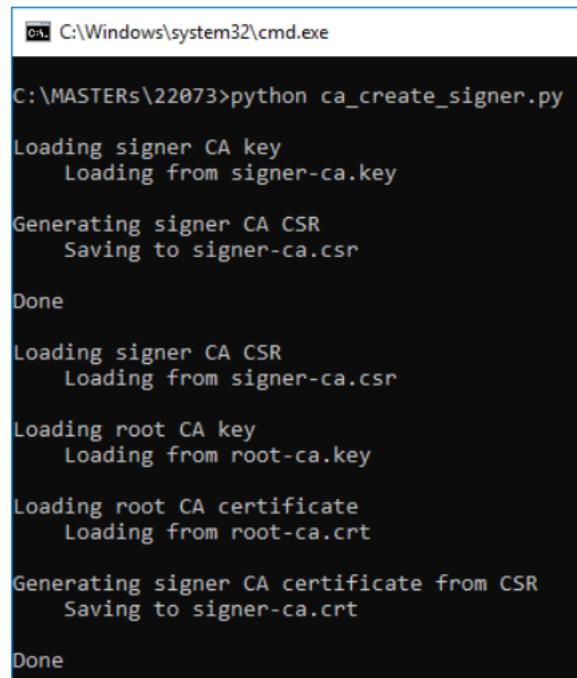


```
C:\Windows\system32\cmd.exe
C:\MASTERs\22073>python ca_create_signer_csr.py
Loading signer CA key
    No key file found, generating new key
        Saving to signer-ca.key

Generating signer CA CSR
    Saving to signer-ca.csr

Done
```

**“python ca\_create\_signer.py”**



```
C:\Windows\system32\cmd.exe
C:\MASTERs\22073>python ca_create_signer.py
Loading signer CA key
    Loading from signer-ca.key

Generating signer CA CSR
    Saving to signer-ca.csr

Done

Loading signer CA CSR
    Loading from signer-ca.csr

Loading root CA key
    Loading from root-ca.key

Loading root CA certificate
    Loading from root-ca.crt

Generating signer CA certificate from CSR
    Saving to signer-ca.crt

Done
```

## 5– Adicionar certificado ao AWS IoT

Para que o dispositivo com um ECC608A possa rodar o Just In Time Registration, é necessário registrar o CA que o assinou. Este próximo comando executa esta tarefa de adicionar o Signer CA à lista de CAs válidos no AWS IoT.

**“python aws\_register\_signer.py”**

```
C:\Windows\system32\cmd.exe
C:\MASTERs\22073>python aws_register_signer.py

Reading signer CA key file, signer-ca.key

Reading signer CA certificate file, signer-ca.crt

Initializing AWS IoT client
  Profile: default
  Region: us-west-2
  Endpoint: iot(https://iot.us-west-2.amazonaws.com)

Getting CA registration code from AWS IoT
  Code: 9d14f03cee6d4a0a3b5367d2608641129ea1f1b37f484a827e62275b1aeb018b

Generating signer CA AWS verification certificate
  Saved to signer-ca-verification.crt

Registering signer CA with AWS IoT
  ID: 669f9bdca333d6131d084e28ce4a3c819e465aef44fd555c4ec532e4bd3009e4

Getting AWS IoT device endpoint
  Hostname: a19y30c61anbyf.iot.us-west-2.amazonaws.com

Done
```

## 6 - Certificado digital do ATECC608A do kit

Para gerar o certificado digital do ATECC608A, digite o comando abaixo. Este comando solicita ao SAML21 que está na placa e se comunica via USB para enviar o CSR da chave privada que se encontra dentro do ATECC608A. Com o CSR, o programa utiliza a chave privada Signer para gerar o certificado digital do ATECC608A que tem como CA, o certificado que está na AWS IoT. O Programa envia o certificado digital do ATECC608A e do Signer CA que é gravado dentro do ATECC608A.

**“python kit\_provision.py”**

```
C:\Windows\system32\cmd.exe
(venv) C:\MASTERs\22073>python kit_provision.py

Opening AWS Zero-touch Kit Device

Initializing Kit
  ATECC508A SN: 01233B0145192965A5
  ATECC508A Public Key:
    X: E4755DF70F53E5E417088DB4F44E4FF77EB2188A7A7628B421E0AB197F499E6
    Y: B9B264501C3AC934D714AEE0FC62D0913093DD2F31896007D90B212768161936

Loading root CA certificate
  Loading from root-ca.crt

Loading signer CA key
  Loading from signer-ca.key

Loading signer CA certificate
  Loading from signer-ca.crt

Requesting device CSR
  Saving to device.csr

Generating device certificate from CSR
  Saving to device.crt

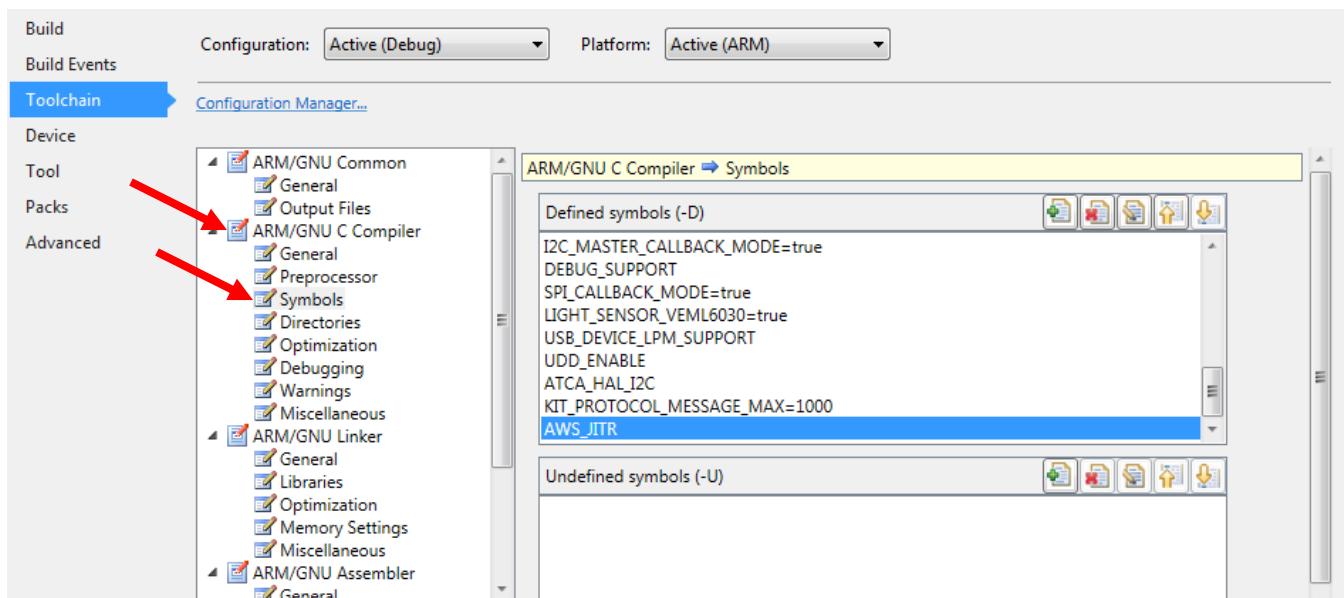
Provisioning device with AWS IoT credentials

Updating WiFi settings
  SSID: wifi-name
  Password: *****

Done
```

Abra o projeto “saml21g18b\_sensor\_board\_demo\_JTIR.atsln”.

Este projeto contém o código da aplicação. Este código faz toda a parte de comunicação com a AWS utilizando os certificados que foram gravados com o firmware anterior. Para compilar corretamente, certifique-se que o símbolo AWS\_JITR esteja presente e que o símbolo **ECC508 NÃO ESTEJA**



Sem o símbolo ECC508, a parte de provisionamento não fará parte do projeto.

```

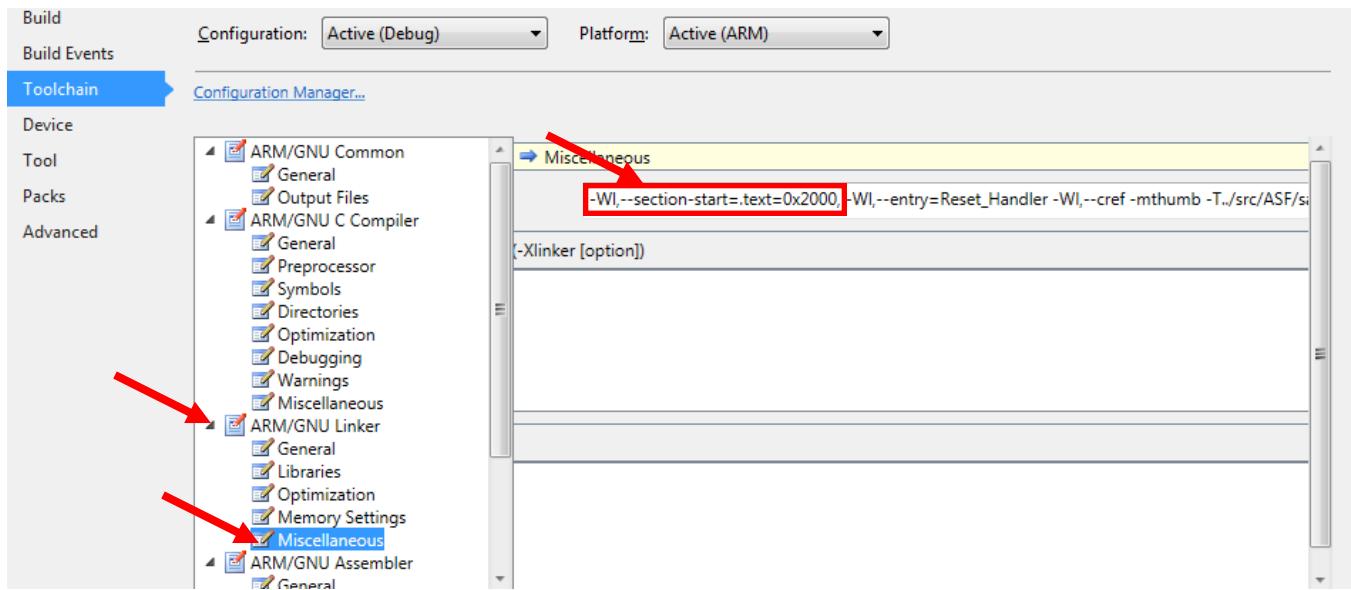
192
193 #ifdef AWS_JITR
194     // Initialize the USB HID interface
195     usb_hid_init();
196
197     ecc_provisioning_task();
198 #endif
199
200 #ifdef ECC508
201     while (1) {
202         zero_touch_provisioning_task();
203         led_ctrl_set_color(LED_COLOR_YELLOW, LED_MODE_BLINK_NORMAL);
204         //break;
205     }
206 #endif
207

```

Para facilitar a alteração do firmware, é gravado um bootloader de 8KBytes. Assim a aplicação deve iniciar a partir do endereço 0x2000. Esta configuração é feita nas propriedades de projeto -> ARM/GNU C Compiler -> Miscellaneous e adicionar a diretiva abaixo:

**-Wl,--section-start=.text=0x2000**

No Anexo E será discutido em detalhes o Bootloader gravado no endereço 0x0000.

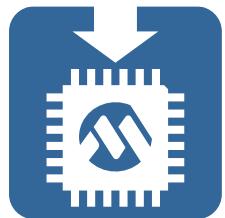


Seguindo os passos do Anexo E, grave o firmware “saml21g18b\_sensor\_board\_demo\_JITR.bin”.

Agora a sua placa está pronta para rodar os LAB2 e LAB3.

# Anexo E

*SAM-BA Bootloader para o Sensor Board*

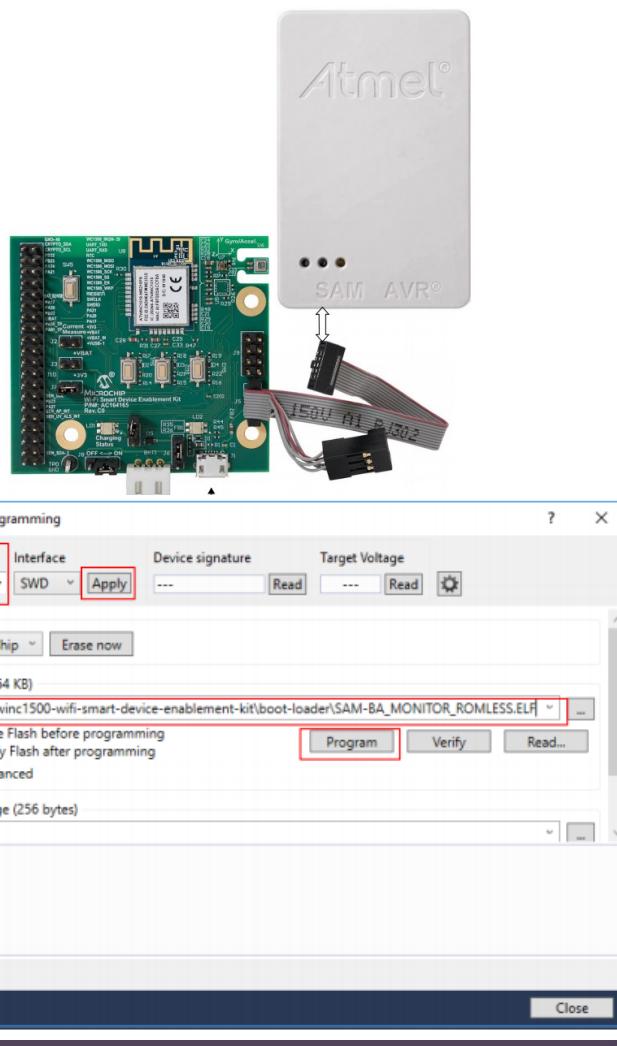


A Placa Wireless sensor board utilizada neste laboratório utiliza o microcontrolador SAML21G18B. Diferentemente de outras placa de desenvolvimento da Microchip, esta placa não possui gravador integrado. Desta forma, para poder gravá-la sem a necessidade de um programador externo, esta placa vem com um Bootloader pre-gravado que pode fazer a gravação pela interface USB ou por uma port UART. Este projeto de bootloader é baseado no SAM-BA bootloader, uma ferramenta de gravação da Microchip. É possível baixar esta ferramenta no link abaixo:

<https://www.microchip.com/DevelopmentTools/ProductDetails/PartNO/SAM-BA%20In-system%20Programmer>

Os Microcontroladores Cortex-M4 e Cortex-M7 da Microchip vem com o SAM-BA bootloader gravado em ROM. Como os Cortex-M0+ não possuem esta ROM, é preciso gravá-la no inicio da Flash.

1. Grave o bootloader “**SAM-BA\_MONITOR\_ROMLESS.ELF**” que se encontra na pasta C:\MASTERs\Alexa\firmware utilizando o Atmel Studio 7 e uma ferramenta de gravação externa.



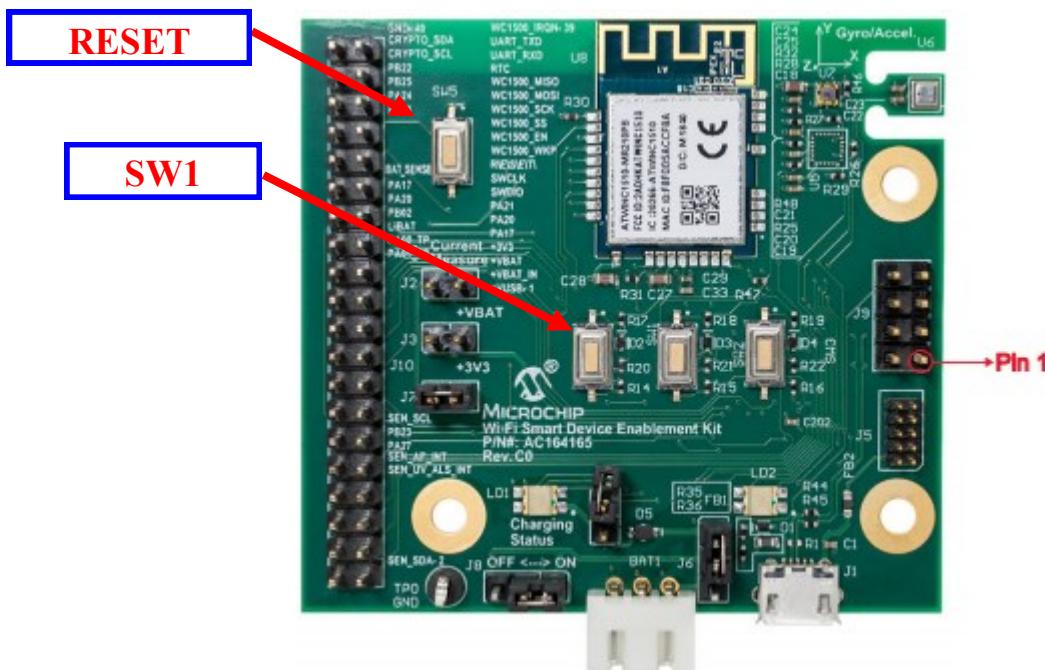
### 2.3 Gravação do Firmware de aplicação

Uma vez que o SAM-BA bootloader for gravado com sucesso, Ao ligar o kit W-FI Sensor board na USB do computador, o mesmo será enumerado como uma porta serial. O SAM-BA tem o conjunto de instruções para se comunicar via USB com o programa SAM-BA v2.18 e também apagar e gravar a área de aplicação.

Se uma aplicação válida estiver gravada e o botão SW1 não estiver pressionada no momento da boot, a aplicação final começará a rodar.

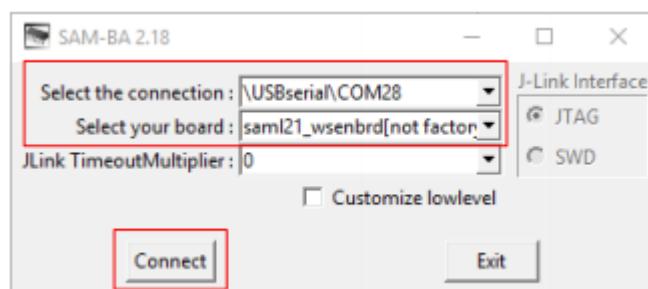
Caso não tenha nenhum firmware de aplicação válida ou o botão SW1 estiver pressionado no momento do boot, o microcontrolador irá rodar o programa de boot aguardando a comunicação USB.

1– Com o cabo USB ligado,复位 a placa segurando o botão SW1.



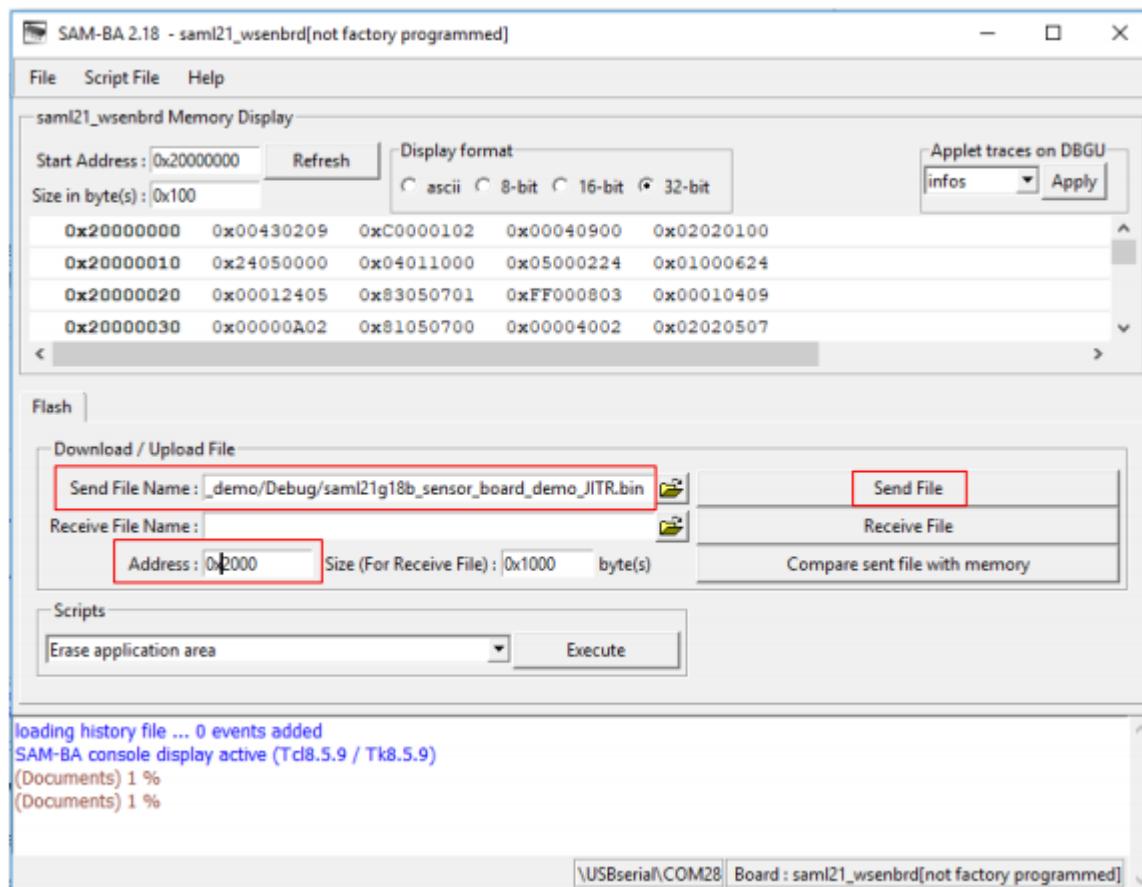
2– Abrir o programa SAM-BA v2.18.

3– Selecionar a serial correta e a placa SAML21\_WSENBRD.



4- Clicar em “connect” para abrir a interface gráfica do SAM-BA

5 - Na aba “Flash” selecionar o arquivo BIN que será gravado. E configurar corretamente o endereço inicial da aplicação. No caso do bootloader gravado anteriormente, **0x2000**



6 - Reiniciar a placa apertando o Botão de RESET sem apertar o botão SW1.