

Programación de Skills con Alexa

Introducción a la programación de Skills

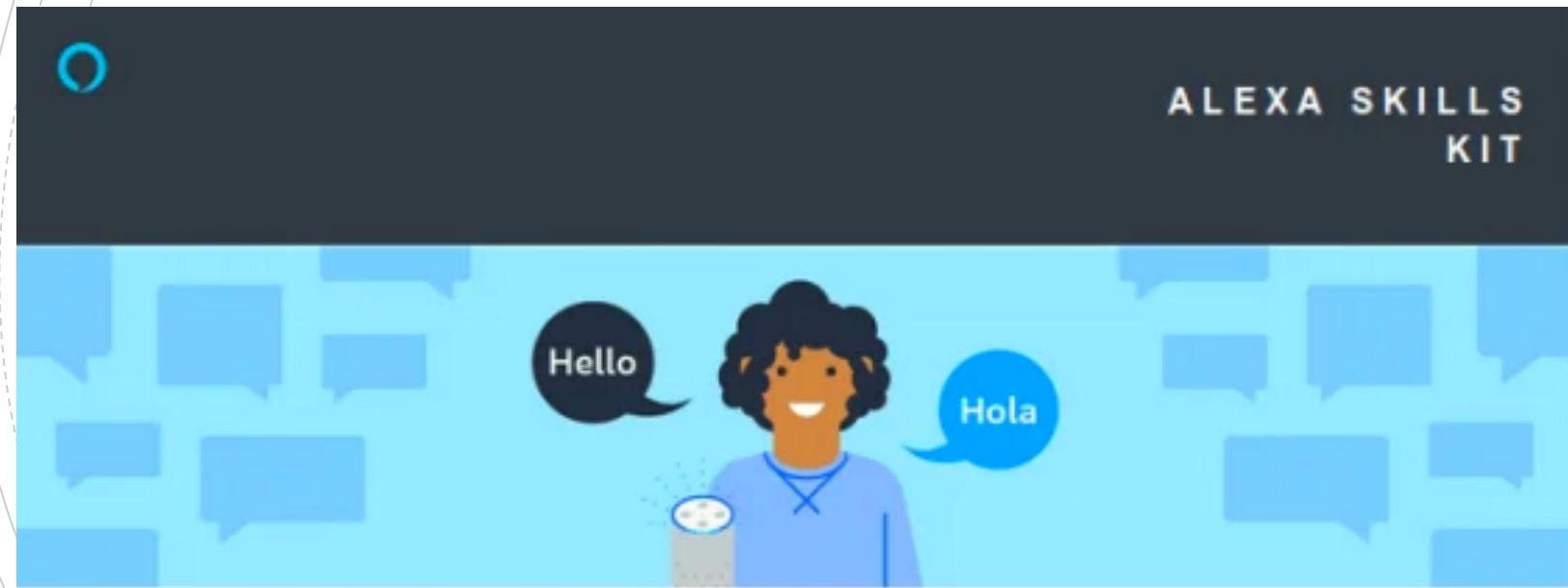
José Luis González Sánchez

@joseluisgonsan



¿Qué vamos a aprender?

- Conceptos fundamentales
- Arquitectura
- Cómo crear una skill
- Mi primera Skill



Dónde aprender

- El site principal de Amazon para los developers de Alexa en español: <https://developer.amazon.com/es/alexa>. Desde ahí se puede acceder a toda la doc e info básica para empezar
- La consola del Alexa Skills Kit como punto de partida para crear un skill nuevo y gestionar los existentes: <https://developer.amazon.com/alexa/console/ask>
- El site con doc: <https://developer.amazon.com/es/documentation> Aquí de primeras destacar:
 - La doc para construir el skill en sí mismo <https://developer.amazon.com/es/docs/ask-overviews/build-skills-with-the-alexa-skills-kit.html>
 - La doc base para entender un custom skill: <https://developer.amazon.com/es/docs/custom-skills/understanding-custom-skills.html>. Hay distintos tipos que puedes ver aquí: <https://developer.amazon.com/es/docs/ask-overviews/understanding-the-different-types-of-skills.html#skill-models>
 - Y la doc para el SDK en Java, lo hay para Nodejs y Python también: <https://developer.amazon.com/es/docs/sdk/alexa-skills-kit-sdk-for-java.html>

Dónde aprender

- Seguiremos estos tutoriales, donde tendrás toda la información disponible que usamos aquí
 - <https://developer.amazon.com/es-ES/alexa/alexa-skills-kit/get-deeper/tutorials-code-samples/build-an-engaging-alexa-skill/module-1>
 - <https://developer.amazon.com/es-ES/alexa/alexa-skills-kit/get-deeper/tutorials-code-samples/build-an-engaging-alexa-skill/module-2>
 - <https://developer.amazon.com/es-ES/alexa/alexa-skills-kit/get-deeper/tutorials-code-samples/build-an-engaging-alexa-skill/module-3>



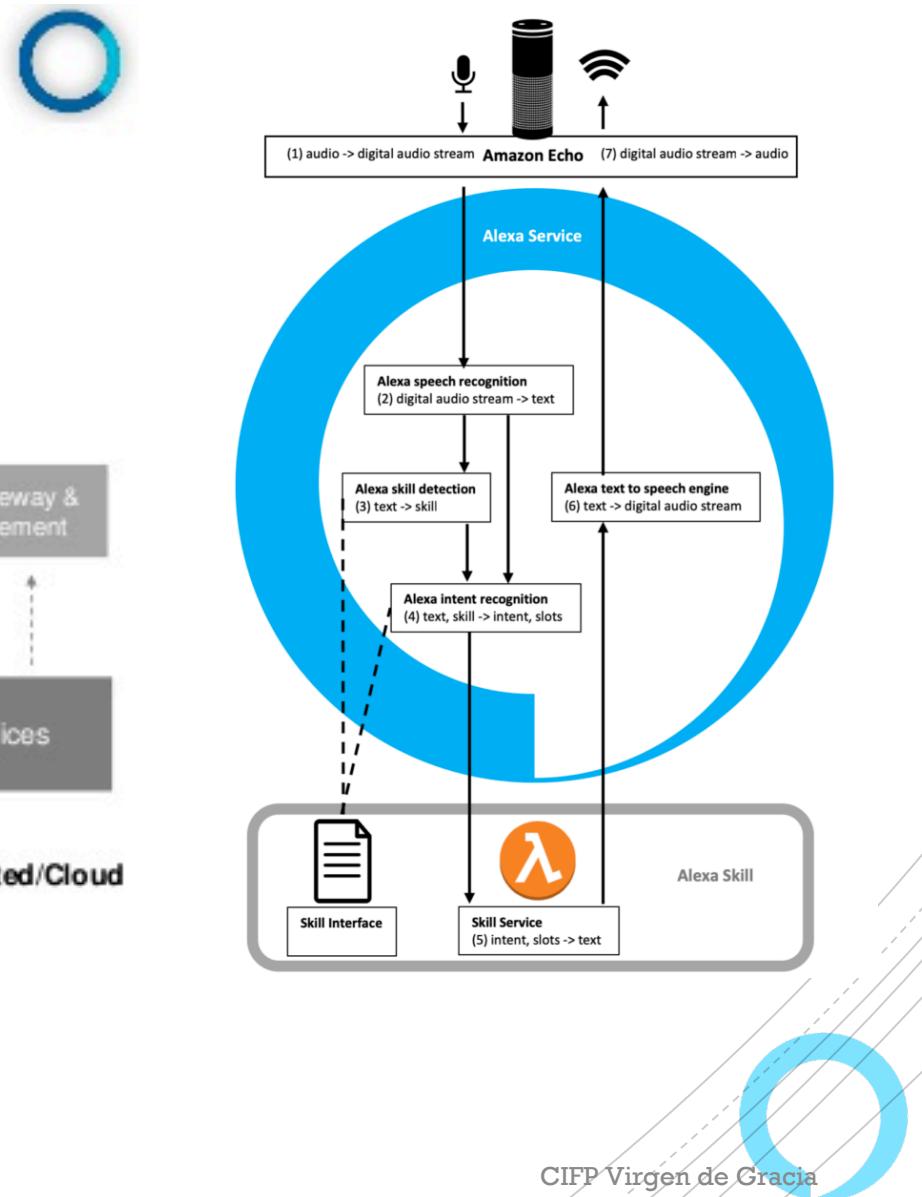
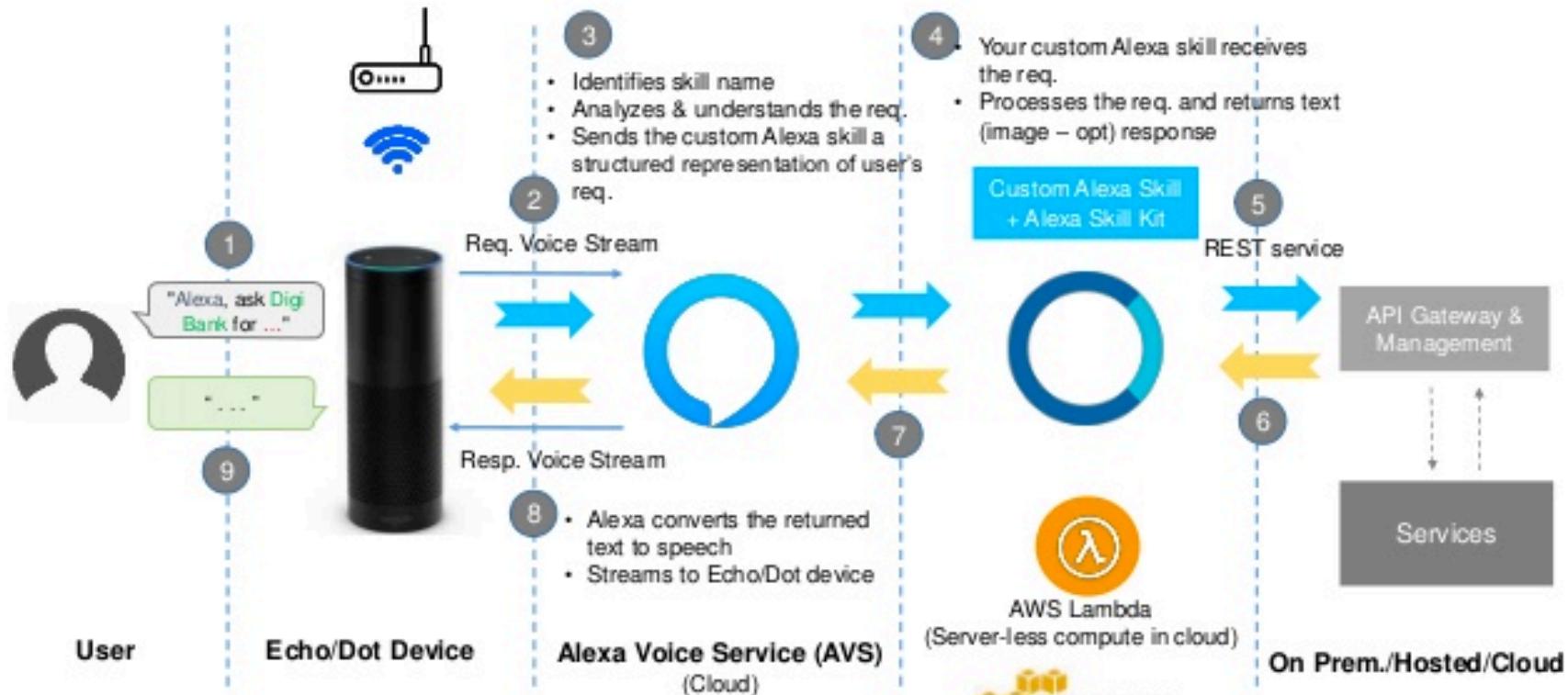
Ecosistema de Alexa

- **Alexa** es el servicio de voz ubicado en la nube de Amazon disponible en los dispositivos de Amazon y dispositivos tercios con Alexa integrada. Con Alexa, puedes crear experiencias de voz naturales para ofrecer a los clientes una forma más intuitiva de interactuar con la tecnología que usan a diario.
- **Alexa Voice Service (AVS)**, El servicio basado en la nube que permite a los fabricantes de dispositivos integrar un conjunto cada vez mayor de características y funciones de Alexa en un producto conectado.
- **Alexa Skill Kit (ASK)**, te permite crear y diseñar nuevas skills para Alexa.



Arquitectura de una Skill

Alexa Custom Skill - Reference Architecture



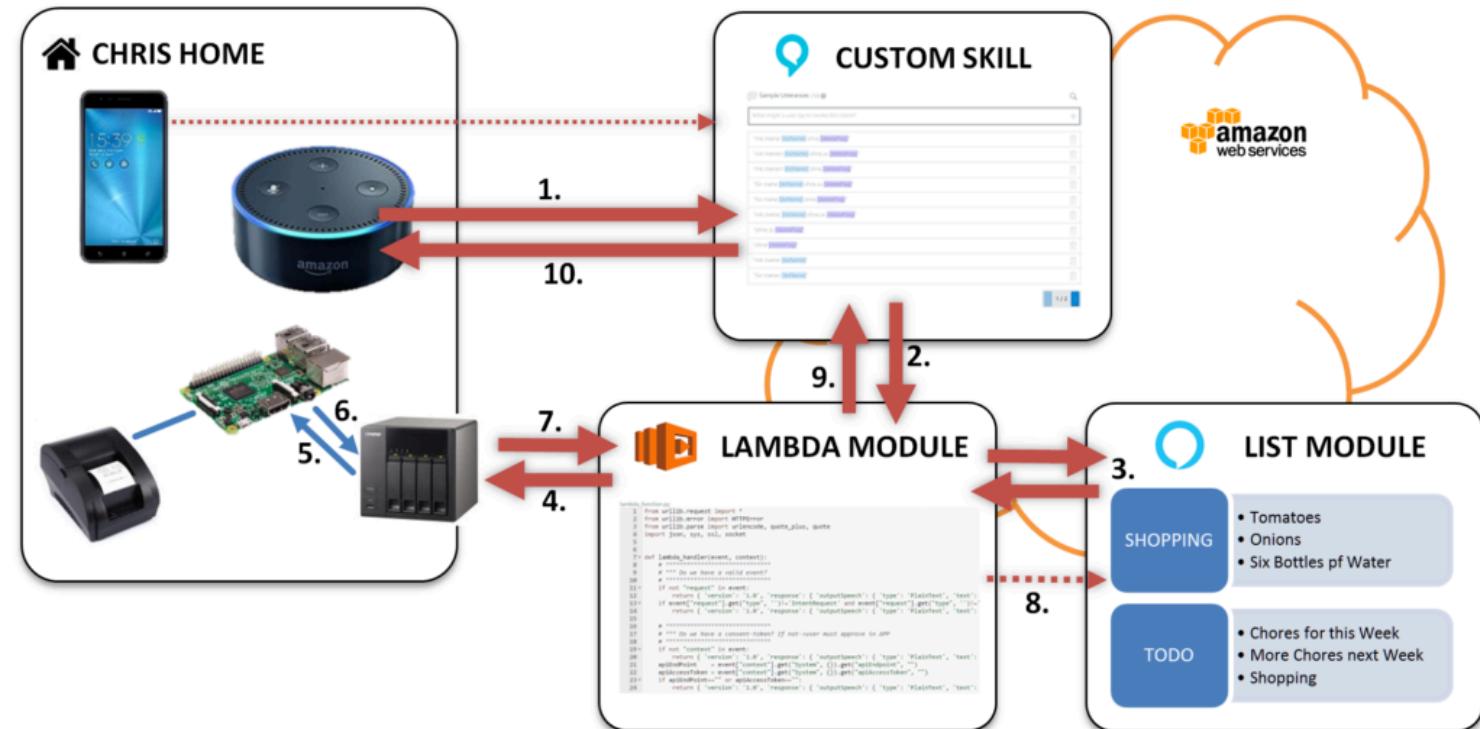
Arquitectura de una Skill

- Una **Skill** se basa en el siguiente funcionamiento:
 - Esta **skill** se lanzará a través de la palabra de inicio (Alexa) seguida de **invocation name** (nombre de invocación, skill) que a su vez tendrá una serie de frases de declaración **de intenciones (intent)**.
 - Un intent está compuesto por dos partes: los **utterances (sentencias)** que van a servir para invocarlo y los **slots (argumentos)**, opcionales, que puede tener.
 - Por lo tanto tendremos un ejemplo: **Alexa, abre informática y dime los módulos de 2 DAM**.
 - Palabra de Activación: Alexa
 - Invocation name: informática
 - Intent: módulo
 - Slot: 2 DAM
- Esta es la idea básica para saber cómo crear un sistema de diálogo basado en estímulo respuesta, es decir, recibe una frase, se analiza por el AVS, se procesa en el Lambda Service y se ofrece una respuesta. Lo iremos viendo poco a poco, así como todos estos conceptos. No te preocupes.



Arquitectura de una Skill

- Cuando hacemos una Skill necesitamos
 - Frontend, interfaz de usuario por voz (estímulos), basada en AVS, procesando Intenciones, sentencias y argumentos.
 - Backend, programación y respuesta a los estímulos en un AWS Lambda Server-less. Lógica de la programación.
- Comenzaremos usando la herramienta **Alexa-Hosted Skills:** alexa.design/hosted
- Nuestra skill tendrá esta estructura
 - /models, donde procesaremos la interfaz, tendremos un fichero por cada idioma soportado (frontend)
 - /lambda, donde se aloja el código de procesamiento (backend)
 - skill.json, tiene todos los archivos de los metadatos de la skill



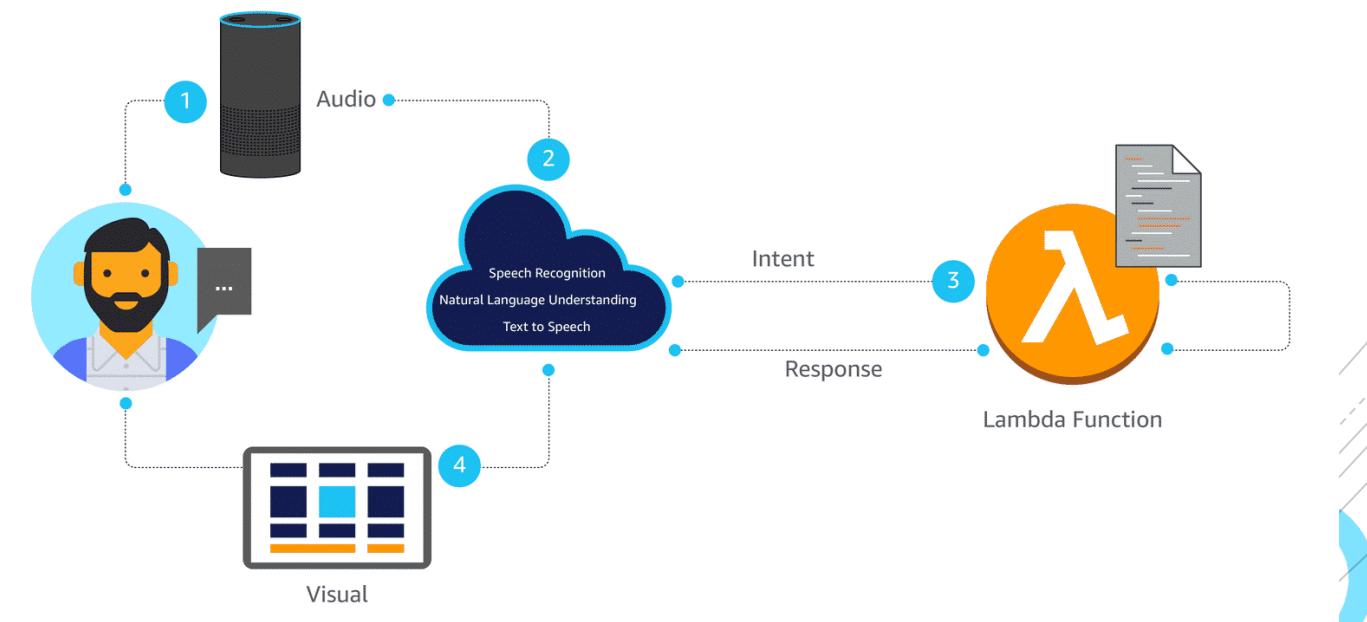
Handler

- Antes de seguir, aunque luego lo veremos tenemos un concepto importante para la parte de Backend y son los Handler, métodos que tendremos que tener en nuestro Lambda (Fichero ejecutable del servidor por así decirlo). Así se denominan los controladores.
- Ellos detectan qué tipo invocación llegan y si pueden o no inicializarla (método canHandle) y posteriormente invocan al método handle del mismo que procesa esa intención y genera la salida.
- En el ejemplo solo se activa si la intención es de LaunchRequest (método canHandle) y devuelve un mensaje de bienvenida en el método handle

```
//LANZAMIENTO - INTENCION
const LaunchRequestHandler = {
  canHandle(handlerInput) {
    return Alexa.getRequestType(handlerInput.requestEnvelope)
    === 'LaunchRequest';
  },
  // Proceso
  handle(handlerInput) {
    const mensaje = 'Hola, Hola, soy tu mensaje de bienvenida';

    return handlerInput.responseBuilder
      .speak(mensaje)
      .getResponse();
  }
};
```

José Luis González Sánchez.



Handler

- Todos los controladores o handlers a utilizar deben estar importados justo en el final de código en una variable que se llama exports.handler = Alexa.SkillBuilders.custom()
- Allí quedarán registrados y empezarán a resolverse por orden de prioridad de arriba a bajo.

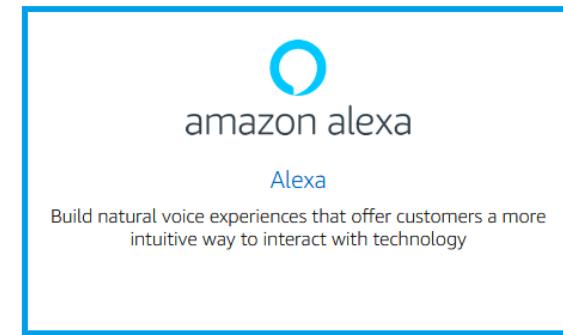
```
exports.handler = Alexa.SkillBuilders.custom()
  .addRequestHandlers(
    LaunchRequestHandler,
    MiIntencionHandler,
    HelpIntentHandler,
    CancelAndStopIntentHandler,
    FallbackIntentHandler,
    SessionEndedRequestHandler,
    IntentReflectorHandler)
  .addErrorHandlers(
    ErrorHandler)
  .lambda();
```

Mi primera Skill

- Lo primero es hacerse una cuenta y entrar en Amazon Developer: <https://developer.amazon.com/es/>
- Despues de iniciar sesión, se te guiará a la vista general del servicio. Selecciona aquí “Amazon Alexa”.



Amazon Developer Services and Technologies



Amazon Appstore

Develop Android apps and games for Amazon Fire TV, Fire tablet, and mobile platforms



Dash Services

Build Amazon reordering experiences into your devices



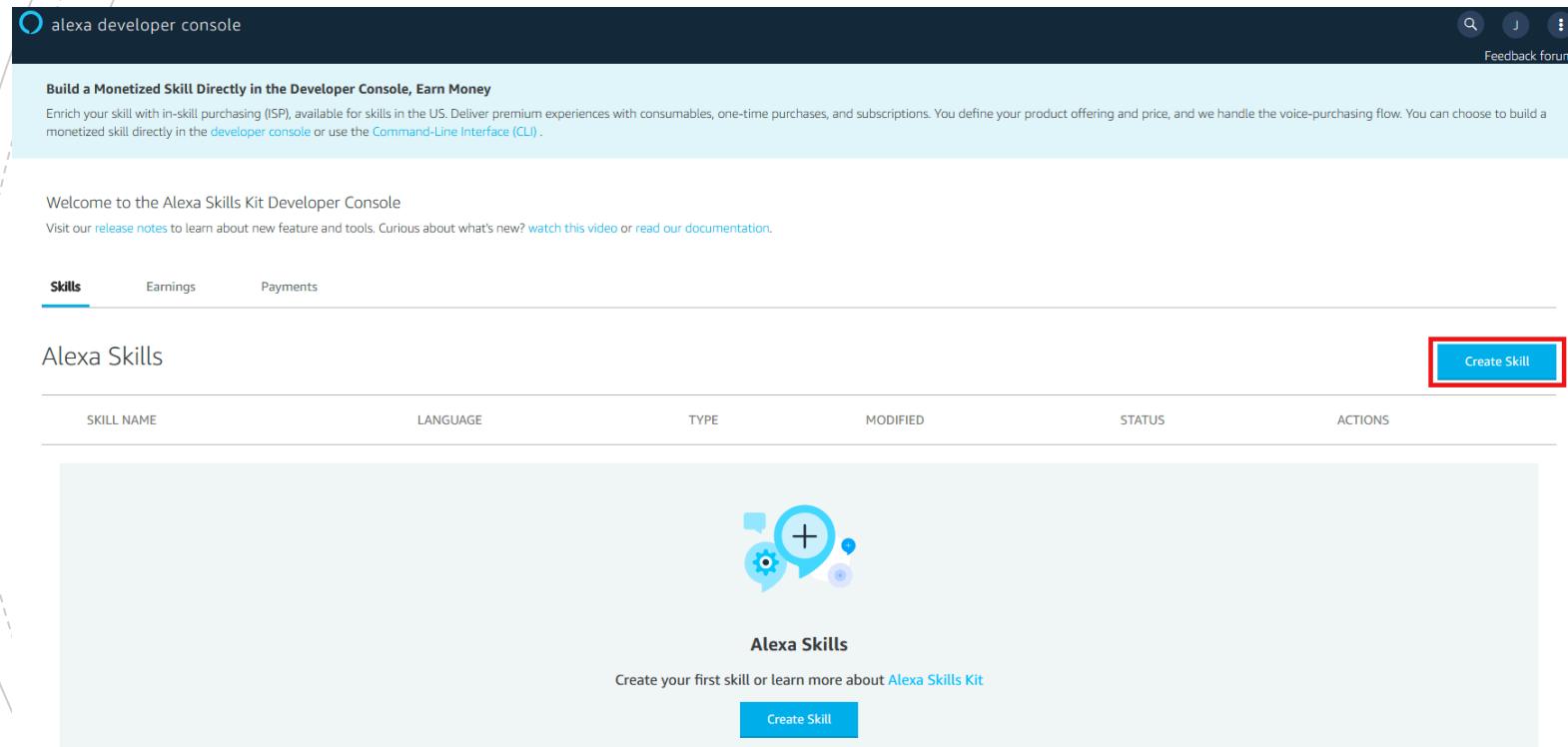
AWS Developer Center

Find tools, documentation, and sample code to build applications in your favorite language



Mi primera Skill

- Haz clic en “Alexa Skills Kit” en el menú desplegable bajo “Productos” y luego en “Develop Skill” para abrir la consola de desarrollo de Alexa.
- Haz clic en “Crear Skill” para crear una nueva Skill de Alexa. O directamente en:
<https://developer.amazon.com/alexas/console/ask>



Mi primera Skill

- Desde aquí podrás crear tus skills y verás listados todos los que tengas con info y acciones directas sobre algunas secciones como las analíticas o la gestión de la beta. Tienes un par de links sobre documentación de la consola, tanto en vídeo como en texto. Son bastante útiles para hacerte una idea del site. Una vez publicas una skill te aparecerá por partida doble pero con distinto estado, la de desarrollo y la que está en live en la "tienda de Alexa skills" de Amazon.
- Lo primero que nos pide para crear nuestro skill es: nombre, idioma del skill y de qué tipo es. Puedes consultar los distintos tipos [aquí](#). Los tipos disponibles dependen del idioma por defecto que elijas. El tipo que casaba con mi idea de skill era un "[custom skill](#)".



Mi primera Skill

- Nombra tu Skill, selecciona el idioma deseado y elige uno de los **cuatro tipos de modelo** para el modelo de interacción de tu habilidad:
 - Modelo personalizado (modelo de interacción definido por el usuario)
 - Modelo Flash Briefing (modelo de interacción predefinido para feeds de noticias)
 - Modelo Smart Home (modelo de interacción predefinido para aplicaciones Smart Home)
 - Modelo de vídeo (modelo de interacción predefinido para aplicaciones de vídeo)
 - En este tutorial de Alexa Skill te mostraremos cómo crear **modelos de interacción personalizados** partiendo del Custom Skill Model.
 - Introduce como “Nombre de la Skill” (“Skill name”) el nombre con el que se mostrará tu Skill de Alexa más adelante en la vista general y en la tienda de Alexa Skills.
- En nuestro ejemplo, elegimos Hello World y seleccionamos inglés, porque luego aprenderemos a traducir. Simplemente te lo digo porque el número de plantillas es mayor
- Indicamos el modo servidor, en mi caso Node.js o Python. El objetivo es que con ellos nos automatiza la subida y configuración del AWS Lambda Serve-less, por lo que es interesante. Se podría hacer con JAVA, pero nos implica dar más pasos. Esta es nuestra primera APP
- A continuación, haz clic en “Crear Skill” (“Create Skill”) para iniciar el proceso de desarrollo. Se te redirigirá automáticamente a la vista de edición de la consola de desarrollo de Alexa.

Mi primera Skill

- Seleccionamos la plantilla por defecto, hay muchas, por eso lo de inglés 😊

Create a new skill

Skill name

Informatica Virgen de Gracia

28/50 characters

Default language

Spanish (ES)

More languages can be added to your skill after creation

1. Choose a model to add to your skill

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

Custom

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, pon el resumen de noticias."

Smart Home

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

"Alexa, enciende las luces de la cocina"

Video

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

"Alexa, pon Interstellar"

2. Choose a method to host your skill's backend resources

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

Provision your own

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

Alexa-Hosted (Node.js)

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

Alexa-Hosted (Python)

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

Mi primera Skill

- Seleccionamos la plantilla por defecto, hay muchas, por eso lo de inglés o español si quieres 😊

Choose a template to add to your skill

Select a quick start template to get started with a skill that contains an interaction model and backend code that work together from the start.

Hello World Skill

This skill gets you started with skill building by providing basic "Hello World" functionality and rapidly generating a voice response from Alexa. [Learn more](#)

SELECTED

Cake Walk (Tutorial)

This skill celebrates your birthday! Users can tell Alexa their birthday, and she will count down the days until the big day.

[Learn more](#)

➔ Includes: slots, dialog management, session persistence

Fact Skill

Build an engaging fact skill about any topic. Alexa will select a fact at random and share it with the user when the skill is invoked. [Learn more](#)

➔ Includes: custom intents

Sauce Boss Skill

Build multi-modal experiences on screen enabled Echo devices. This skill allows a user to ask for a recipe and receive a description and images. [Learn more](#)

➔ Includes: custom intents, Alexa Presentation Language (APL)

Mi primera Skill

- Por ello tendremos algo así

- Siguiendo el "Skill builder checklist" completarás lo mínimo necesario para tener un skill listo. Hay dos partes básicas: el interaction model del skill y el endpoint al server que gestionará las peticiones al skill. En este post sólo voy a hablar sobre lo primero.

The screenshot shows the Alexa Skills Kit Developer Console interface. On the left, there's a sidebar with navigation links: Spanish (ES), CUSTOM, Interaction Model, Utterance Conflicts (0), Invocation, Intents (5) (with HelloWorldIntent and Built-In Intents (4) like AMAZON.CancelIntent, AMAZON.HelpIntent, AMAZON.StopIntent, AMAZON.NavigateHomeIntent), Slot Types (0), JSON Editor, Interfaces, Endpoint, Intent History, Annotation Sets, Display, IN-SKILL PRODUCTS, and ACCOUNT LINKING. The main area has sections for How to get started (with a video thumbnail for the Alexa Skills Kit Developer Tutorial for Programmers), Resources (including Update your live skill instantly, Catalog Management, Feature Updates & Releases, Getting Started: A Comprehensive Course (Cake Walk), and Documentation), and a Skill builder checklist. The checklist consists of four required steps: 1. Invocation Name (checked), 2. Intents, Samples, and Slots (checked), 3. Build Model (checked), and 4. Endpoint (checked).

Interaction Model: Invocación

- El interaction model del skill consta de, al menos, dos partes: el nombre de invocación de tu skill y el conjunto de intent (acciones) que deben corresponder con las peticiones de los clientes. Es el **Frontend** de nuestra aplicación
- **Nombre de invocación:** El nombre de invocación será lo que el usuario use para comenzar a interactuar con tu skill y debe cumplir una serie de requisitos.
- Creo que es bastante importante la elección que se hace porque el usuario podrá invocar el skill y ejecutar un intent de una sola vez, one-shot invocation. Por ejemplo, "Alexa, abre informática virgen de gracia y dime los módulos de segundo dam" y tiene que tener sentido. El usuario también podría conseguir el mismo resultado con: "Alexa, abre informática virgen de gracia dam".
- A mi me gusta bastante la forma de usarlo de one-shot invocation y me he querido centrar en eso al diseñar el modelo.

Interaction Model: Invocación

- Cambiamos la Invocación, nombre de la Skill. En cada cambio que hagamos en todo, pulsamos Salvar y Build.

CUSTOM

- Interaction Model
- Utterance Conflicts (0)
- Invocation**
- Intents (5) + Add
- HelloWorldIntent trash
- Built-In Intents (4)
 - AMAZON.CancelIntent
 - AMAZON.HelpIntent

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 ?

informatica virgen de gracia

Interaction Model: Intents

- Un intent está compuesto por dos partes: los utterances (sentencias) que van a servir para invocarlo y los slots (argumentos), opcionales, que puede tener. Un par de cosas sobre los utterances:
 - Dentro de un mismo intent no puedes tener dos utterances repetidos, pero sí el mismo con distintos slots.
 - Los utterance los usa Alexa para inferir qué intent corresponde a lo que quiere hacer el usuario.
 - Es importante que fijes el orden porque irá resolviendo por cómo los diseña.
- **Intents obligatorios**
 - Al crear un skill nuevo hay algunos intents obligatorios, predefinidos ya por Alexa, que tienes que tener cubiertos y te los indica la consola debajo de "Built-in intents". Verás que son básicos, sin slots. Se pueden extender
 - *NavigateHome*, que se usa para skill multimodal, los que tienen pantalla, y equivale a un exit
 - *Cancel*, se usaría para cancelar alguna interacción en proceso
 - *Stop*, para dejar de usar el skill
 - *Help*, se invocaría por el usuario para pedir ayuda a nuestro skill
 - Para cada una de estos tendrás que elegir utterances que vayan a invocarlo



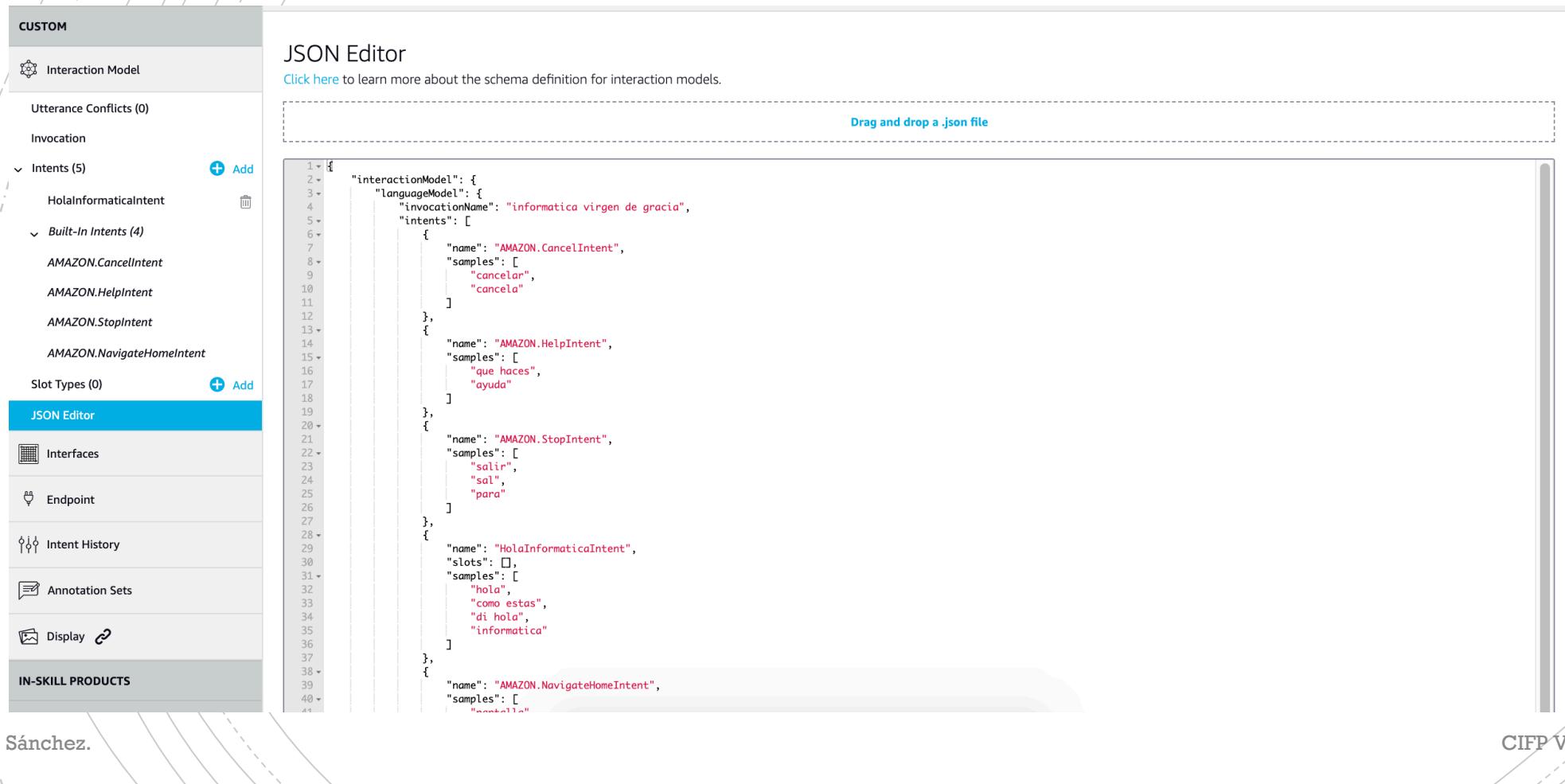
Interaction Model: Intents

- **Custom intents:** Estos intents serán los que definan realmente los casos de uso que queremos cubrir en nuestro skill.
- Al crear un intent tienes la opción de usar uno existente de la librería de Alexa, como los built-in intent que eran obligatorios, o uno custom.
- El nombre del intent debería ser explicativo de la acción correspondiente que se quiere cubrir. Se debe seguir la convección: "NombreAccionIntent"
- Una vez creado, lo único que tenemos que añadir, por el momento, son los utterance para ser invocado.

The screenshot shows the 'Interaction Model' section of the Alexa developer console. On the left, a sidebar lists 'CUSTOM' (selected), 'Utterance Conflicts (0)', 'Invocation', and 'Intents (5)'. Under 'Intents', 'HolainformaticalIntent' is selected, highlighted in blue. Below it, 'Built-In Intents (4)' is expanded, showing 'AMAZON.CancelIntent', 'AMAZON.HelpIntent', 'AMAZON.StopIntent', and 'AMAZON.NavigateHomeIntent'. At the bottom of the sidebar are 'Slot Types (0)' and a '+ Add' button. The main area is titled 'Intents / HolainformaticalIntent'. It contains a message about sample utterances and a table for 'Sample Utterances (4)'. The table has columns for the utterance text and a delete icon. The listed utterances are: 'hola', 'como estas', 'di hola', and 'information'. There is also a '+ Add' button at the top of the table and 'Bulk Edit' and 'Export' buttons at the bottom right.

Interaction Model: Build model

- **Build model.** Tenemos que pulsar el botón de "Build Model" y lo generará. El modelo que se genera no es más que un JSON con toda la información que hemos ido construyendo con la consola. Este JSON se puede ver una vez generado y modificar en vez de usar los input de utterance y todo eso que vimos antes.



The screenshot shows the AWS Lambda Interaction Model builder interface. On the left, there's a sidebar with tabs: CUSTOM, Utterance Conflicts (0), Invocation, Intents (5) (with an 'Add' button), Built-In Intents (4), Slot Types (0) (with an 'Add' button), and three collapsed sections: JSON Editor, Interfaces, Endpoint, Intent History, Annotation Sets, and Display. The 'JSON Editor' tab is currently selected and highlighted in blue. The main area is titled 'JSON Editor' and contains a 'Drag and drop a json file' placeholder above a code editor. The code in the editor is a JSON object representing the interaction model:

```
1 {  
2     "interactionModel": {  
3         "languageModel": {  
4             "invocationName": "informatica virgen de gracia",  
5             "intents": [  
6                 {  
7                     "name": "AMAZON.CancelIntent",  
8                     "samples": [  
9                         "cancelar",  
10                        "cancela"  
11                    ]  
12                },  
13                {  
14                    "name": "AMAZON.HelpIntent",  
15                    "samples": [  
16                        "que haces",  
17                        "ayuda"  
18                    ]  
19                },  
20                {  
21                    "name": "AMAZON.StopIntent",  
22                    "samples": [  
23                        "salir",  
24                        "sal",  
25                        "para"  
26                    ]  
27                },  
28                {  
29                    "name": "HolaInformaticaIntent",  
30                    "slots": {},  
31                    "samples": [  
32                        "hola",  
33                        "como estas",  
34                        "di hola",  
35                        "informatica"  
36                    ]  
37                },  
38                {  
39                    "name": "AMAZON.NavigateHomeIntent",  
40                    "samples": [  
41                        "navegar a la casa"  
42                    ]  
43                }  
44            ]  
45        }  
46    }  
47}
```

Code

- **Code.** Es la pestaña donde programamos la funcionalidad, **Backend**. Se implementan dentro de en un AWS Lambda. Con la ventaja que te lo interconecta todo automáticamente si usas Node.js o Python. Si quieres JAVA lo tendrás que hacer de manera manual.
- Importante
 - const Alexa = require('ask-sdk-core') -> Librería a usar
 - Tantos RequestHandler como necesitemos para manejar los intents que vengan del skill. mismo RequestHandler puede manejar más de un intent.
- El método canHandle se usa para checkear si el handler puede manejar la petición que llega. Aquí la lógica básica que se suele hacer es mirar el nombre del intent de la request.
- El método handle es el encargado de recibir el input y construir la respuesta para el usuario a partir de su request. Es el método donde irá el código más interesante del handler. En el caso del ejemplo me gustaría destacar tres cosas:
 - speak, le estamos dando a la respuesta el texto que Alexa dirá de voz al usuario.
 - simpleCard, aquí construimos una salida para Alexa que será util para dispositivos con pantalla, como la app del móvil.
 - shouldEndSession, con esto le indicamos a Alexa que, una vez manejada la request, no esperamos otra interacción con el usuario y cerramos la sesión, es decir, el skill.
- Si queremos elaborar una respuesta de dos tipos: texto que dirá Alexa con voz y, además, información para construir una "tarjeta" que mostrará Alexa en pantalla. Por último, con la opción de finalizar la sesión indicamos que no esperamos una interacción posterior. Sería un handler orientado a una petición concreta y sin diálogo.
- En el caso de que nuestro handler quiera indicar a Alexa que esperamos seguir interactuando con el usuario tenemos que construir la respuesta con reprompt.
- exports.handler = Alexa.SkillBuilders.custom()-> Ojo, el orden de los handlers importa a la hora de decidir cuál usará para manejar la petición del usuario. Irá validando los handlers en el orden que los hemos registrado y usará el primero que retorne true en la validación del método canHandle.

Code

- En la pestaña de código tenemos el código por defecto.

The screenshot shows the AWS Lambda code editor interface. At the top, there's a navigation bar with tabs: Your Skills, HelloWord, Build, Code (which is selected), Test, Distribution, Certification, Analytics, and Provide feedback. Below the navigation bar are several icons: a back arrow, a folder, a file, a trash can, a pencil, a magnifying glass, and a download icon. To the right of these are three buttons: Save, Deploy (which is highlighted in blue), and Promote to live. A message below the buttons says "Deploy before testing your skill". On the left, there's a sidebar with a tree view showing the project structure: Skill Code > lambda > index.js, package.json, and util.js. The package.json file is currently selected. On the right, the main area displays the content of the index.js file:

```
// This sample demonstrates handling intents from an Alexa skill using the Alexa Skills Kit SDK (v2).
// Please visit https://alexa.design/cookbook for additional examples on implementing slots, dialog management,
// session persistence, api calls, and more.
const Alexa = require('ask-sdk-core');

const LaunchRequestHandler = {
  canHandle(handlerInput) {
    return Alexa.getRequestType(handlerInput.requestEnvelope) === 'LaunchRequest';
  },
  handle(handlerInput) {
    const speakOutput = 'Welcome, you can say Hello or Help. Which would you like to try?';
    return handlerInput.responseBuilder
      .speak(speakOutput)
      .reprompt(speakOutput)
      .getResponse();
  }
};
const HelloWorldIntentHandler = {
  canHandle(handlerInput) {
    return Alexa.getRequestType(handlerInput.requestEnvelope) === 'IntentRequest'
      && Alexa.getIntentName(handlerInput.requestEnvelope) === 'HelloWorldIntent';
  },
  handle(handlerInput) {
    const speakOutput = 'Hello World!';
    return handlerInput.responseBuilder
      .speak(speakOutput)
      // Uncomment 'add a reprompt if you want to keep the session open for the user to respond'
  }
};

module.exports.handler = Alexa.SkillBuilders.custom().build();
```

At the bottom left of the editor, there are links for Logs: Amazon CloudWatch, Media storage: S3 [0/5GB], and Docs: Alexa Hosted Skills.

Code

- Código Final, cuidado que he cambiado algunos nombres para que coincida con los Intents, mira el código fuente final para no tener despistes. Recuerda revisar el registro de los controladores al final.

The screenshot shows the AWS Lambda code editor interface. At the top, there are buttons for Save, Deploy, and Promote to live, along with a timestamp: Last Deployed: Apr 07, 2020, 2:01 AM. On the left, a sidebar shows the project structure: Skill Code, lambda, index.js, package.json, and util.js. The main area displays the index.js code:

```
index.js x
  8  canHandle(handlerInput) {
  9    return Alexa.getRequestType(handlerInput.requestEnvelope) === 'LaunchRequest';
 10  },
 11  handle(handlerInput) {
 12    const speakOutput = 'Hola, esto es la skill del Departamento de Informática del CIFP Virgen de Gracia.';
 13    return handlerInput.responseBuilder
 14      .speak(speakOutput)
 15      .withSimpleCard(
 16        "Bienvenido/a",
 17        "Skill del Departamento de Informática del CIFP Virgen de Gracia")
 18      .reprompt(speakOutput)
 19      .getResponse();
 20  }
 21  };
 22
 23 // Como responder cuando digan Hola o información
 24 const HolaInformaticaIntentHandler = {
 25   canHandle(handlerInput) {
 26     return Alexa.getRequestType(handlerInput.requestEnvelope) === 'IntentRequest'
 27       && Alexa.getIntentName(handlerInput.requestEnvelope) === 'HolaInformaticaIntent';
 28   },
 29   handle(handlerInput) {
 30     const speakOutput = '¡Hola!, ¿Qué tal estás? Encantada de saludarte. En estos momentos no estamos disponibles, pero pronto volveremos. Un saludo para todos mis compañeros y c';
 31     return handlerInput.responseBuilder
 32       .speak(speakOutput)
 33       .withSimpleCard(
 34         "Hola",
 35         "En estos momentos no estamos disponibles, pero volveremos")
 36       //.reprompt('add a reprompt if you want to keep the session open for the user to respond')
 37       .getResponse();
 38   }
 39 };
 40
 41 // Evento de ayuda
 42 const HelpIntentHandler = {
 43
```

At the bottom right, a modal window displays a green checkmark icon and the text "Deployment Successful". Below it, a message says: "If you make any new changes, you will need to deploy for them to take effect." The footer of the page includes links to Logs: Amazon CloudWatch, Media storage: S3 [0/5GB], Docs: Alexa Hosted Skills, and the name José Luis González Sánchez.

Mi primera Skill

- Nos vamos a la pestaña de Test, y marcamos que estamos en Development y probamos a decir open Hello World o indicándolo en el cuadro de diálogo. En mi caso informática

The screenshot shows the Alexa Skills Kit (ASK) Testing interface. At the top, there are tabs for "Skill testing is enabled in:" (Development), "Skill I/O" (checked), "Device Display" (checked), and "Device Log" (unchecked). Below this, there are three tabs: "Alexa Simulator" (selected), "Manual JSON", and "Voice & Tone". The "Spanish (ES)" language is selected. In the "Type or click and hold the mic" field, the user has typed "abrir informatica virgen de gracia". The interface displays a series of interactions:

- A blue speech bubble says: "¡Hola!, ¿Qué tal estás? Encantada de saludarte. En estos momentos no estamos disponibles, pero pronto volveremos. Un saludo para todos mis compañeros y compañeras del departamento de informática. Se os echa mucho de menos."
- A grey button labeled "abrir informatica virgen de gracia" is shown.
- A blue speech bubble says: "Hola, esto es la skill del Departamento de Informática del CIFP Virgen de Gracia."
- A grey button labeled "ayuda" is shown.
- A blue speech bubble says: "Puedes decirme hola o solicitar información, preguntad por un modulo. ¿Lo intentas?"
- A grey button labeled "hola" is shown.
- A blue speech bubble says: "¡Hola!, ¿Qué tal estás? Encantada de saludarte. En estos momentos no estamos disponibles, pero pronto volveremos. Un saludo para todos mis compañeros y compañeras del departamento de informática. Se os echa mucho de menos."

On the right side, the "Skill Invocations" section shows "Viewing: 1 / 1". The "JSON Input 1" panel shows the following JSON:

```
1 {  
2   "version": "1.0",  
3   "session": {  
4     "new": false,  
5     "sessionId": "amzn1.echo-api.session.ad73c7c7-39ec-49d1-a7c5-fe8f6746dd3b",  
6     "application": {  
7       "applicationId": "amzn1.ask.skill.a4f66b33-d699-4b14-a562-b30eef62eda"  
8     },  
9     "user": {  
10       "userId": "amzn1.ask.account.AEPL7SJKNFP3NCS2ZWU6TURLS7RCD2VNPYRRTGQAXC"  
11     }  
12   },  
13   "context": {  
14     "System": {  
15       "application": {  
16         "applicationId": "amzn1.ask.skill.a4f66b33-d699-4b14-a562-b30eef62eda"  
17       },  
18       "user": {  
19         "userId": "amzn1.ask.account.AEPL7SJKNFP3NCS2ZWU6TURLS7RCD2VNPYRRTGQAXC"  
20       },  
21       "device": {  
22         "deviceId": "amzn1.ask.device.AGAC2H50VOXJNJHFW5I5CXHMNHHS0M6WUSOBPI",  
23         "supportedInterfaces": {}  
24       },  
25       "apiEndpoint": "https://api.eu.amazonalexa.com",  
26       "apiAccessToken": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6IjEifQ.e"  
27     },  
28     "Viewport": {}  
}
```

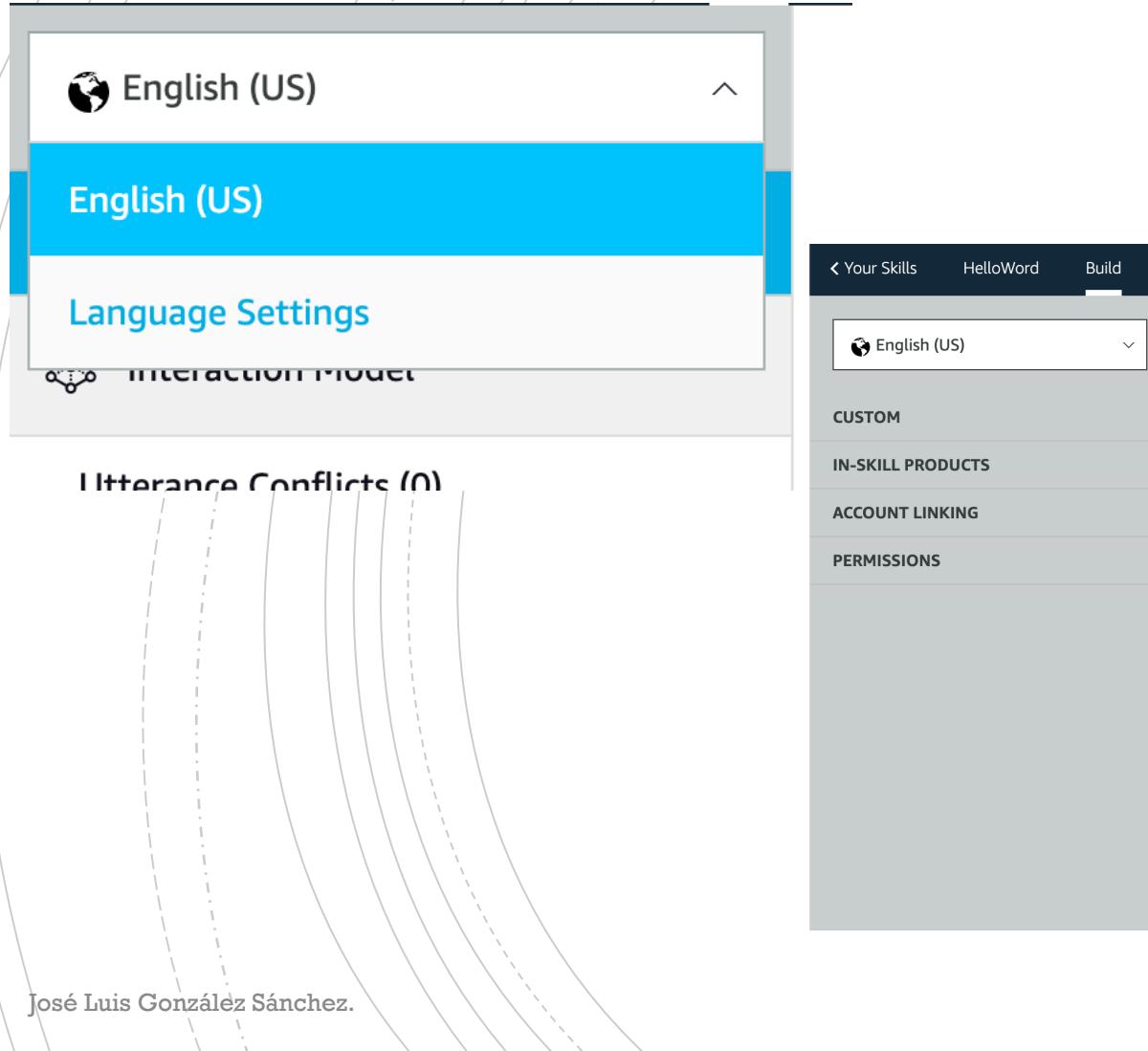
The "JSON Output 1" panel shows the following JSON:

```
1 {  
2   "body": {  
3     "version": "1.0",  
4     "response": {  
5       "outputSpeech": {  
6         "type": "SSML",  
7         "ssml": "<speak>¡Hola!, ¿Qué tal estás? Encantada de saludarte. En estos momentos no estamos disponibles, pero pronto volveremos.  
Un saludo para todos mis compañeros y compañeras del departamento de informática. Se os echa mucho de menos.</speak>"  
8       },  
9       "card": {  
10         "type": "Simple",  
11         "title": "Hola",  
12         "content": "En estos momentos no estamos disponibles, pero pronto volveremos"  
13       },  
14       "type": "DEFAULT_RESPONSE"  
15     },  
16     "sessionAttributes": {},  
17     "userAgent": "ask-node/2.7.0 Node/v10.19.0"  
18   }  
19 }
```

In the bottom left corner, the name "José" is visible.

Traduciendo la Skill a varios idiomas

- Damos a Lenguaje Setting y añadimos Español es.



◀ Your Skills HelloWord Build Code Test Distribution Certification Analytics Provide feedback

English (US)

CUSTOM

IN-SKILL PRODUCTS

ACCOUNT LINKING

PERMISSIONS

Language settings

Saved

LANGUAGE	ACTIONS
English (US)	Remove
Choose language to add:	X
English (AU)	
English (CA)	
English (IN)	
English (UK)	
French (CA)	
French (FR)	
German (DE)	
Hindi (IN)	
Italian (IT)	
Japanese (JP)	
Portuguese (BR)	
Spanish (ES)	
Spanish (MX)	
Spanish (US)	

Traduciendo la Skill a varios idiomas

- Copiamos nuestro JSON de interfaz en inglés (español), y nos cambiamos a la pestaña de Lenguaje en español y lo pegamos.
- Traducimos nuestras frases. Le damos a construir y probamos. ¡Las respuestas siguen en inglés! (y british 😊)
- Hemos localizado el frontend, pero no el backend. Ahora debes abrir el fichero index.js y traducir cada frase, si quieres. Esto es para que utilices plantillas, si no puedes usarlo todo en español desde el comienzo.

The screenshot shows the AWS Lambda Skills Editor interface. On the left, the navigation bar includes 'Intents (5)', 'Slot Types (0)', and tabs for 'JSON Editor' (which is selected), 'Interfaces', 'Endpoint', 'Intent History', and 'Annotation Sets'. The main area displays the JSON configuration for intents, with some phrases translated into Spanish. The right side of the interface shows the 'Test' tab, which includes the Alexa Simulator, a transcript of a conversation, and two JSON panes for 'JSON Input 1' and 'JSON Output 1'.

```
1 {  
2   "interactionModel": {  
3     "languageModel": {  
4       "invocationName": "hola mundo",  
5       "intents": [  
6         {  
7           "name": "AMAZON.CancelIntent",  
8           "samples": []  
9         },  
10        {  
11          "name": "AMAZON.HelpIntent",  
12          "samples": []  
13        },  
14        {  
15          "name": "AMAZON.StopIntent",  
16          "samples": []  
17        },  
18        {  
19          "name": "HelloWorldIntent",  
20          "slots": [],  
21          "samples": [  
22            "hola",  
23            "come estas",  
24            "di hola",  
25            "hola",  
26            "di hola mundo",  
27            "decir hola"  
28          ]  
29        },  
30        {  
31          "name": "AMAZON.NavigateHomeIntent",  
32          "samples": []  
33        }  
34      ],  
35      "types": []  
},  
36    }  
37  }  
38 }  
39 }  
40 }  
41 }  
42 }  
43 }  
44 }  
45 }  
46 }  
47 }  
48 }  
49 }  
50 }  
51 }  
52 }  
53 }  
54 }  
55 }  
56 }  
57 }  
58 }  
59 }  
60 }  
61 }  
62 }  
63 }  
64 }  
65 }  
66 }  
67 }  
68 }  
69 }  
70 }  
71 }  
72 }  
73 }  
74 }  
75 }  
76 }  
77 }  
78 }  
79 }  
80 }  
81 }  
82 }  
83 }  
84 }  
85 }  
86 }  
87 }  
88 }  
89 }  
90 }  
91 }  
92 }  
93 }  
94 }  
95 }  
96 }  
97 }  
98 }  
99 }  
100 }  
101 }  
102 }  
103 }  
104 }  
105 }  
106 }  
107 }  
108 }  
109 }  
110 }  
111 }  
112 }  
113 }  
114 }  
115 }  
116 }  
117 }  
118 }  
119 }  
120 }  
121 }  
122 }  
123 }  
124 }  
125 }  
126 }  
127 }  
128 }  
129 }  
130 }  
131 }  
132 }  
133 }  
134 }  
135 }  
136 }  
137 }  
138 }  
139 }  
140 }  
141 }  
142 }  
143 }  
144 }  
145 }  
146 }  
147 }  
148 }  
149 }  
150 }  
151 }  
152 }  
153 }  
154 }  
155 }  
156 }  
157 }  
158 }  
159 }  
160 }  
161 }  
162 }  
163 }  
164 }  
165 }  
166 }  
167 }  
168 }  
169 }  
170 }  
171 }  
172 }  
173 }  
174 }  
175 }  
176 }  
177 }  
178 }  
179 }  
180 }  
181 }  
182 }  
183 }  
184 }  
185 }  
186 }  
187 }  
188 }  
189 }  
190 }  
191 }  
192 }  
193 }  
194 }  
195 }  
196 }  
197 }  
198 }  
199 }  
200 }  
201 }  
202 }  
203 }  
204 }  
205 }  
206 }  
207 }  
208 }  
209 }  
210 }  
211 }  
212 }  
213 }  
214 }  
215 }  
216 }  
217 }  
218 }  
219 }  
220 }  
221 }  
222 }  
223 }  
224 }  
225 }  
226 }  
227 }  
228 }  
229 }  
230 }  
231 }  
232 }  
233 }  
234 }  
235 }  
236 }  
237 }  
238 }  
239 }  
240 }  
241 }  
242 }  
243 }  
244 }  
245 }  
246 }  
247 }  
248 }  
249 }  
250 }  
251 }  
252 }  
253 }  
254 }  
255 }  
256 }  
257 }  
258 }  
259 }  
260 }  
261 }  
262 }  
263 }  
264 }  
265 }  
266 }  
267 }  
268 }  
269 }  
270 }  
271 }  
272 }  
273 }  
274 }  
275 }  
276 }  
277 }  
278 }  
279 }  
280 }  
281 }  
282 }  
283 }  
284 }  
285 }  
286 }  
287 }  
288 }  
289 }  
290 }  
291 }  
292 }  
293 }  
294 }  
295 }  
296 }  
297 }  
298 }  
299 }  
300 }  
301 }  
302 }  
303 }  
304 }  
305 }  
306 }  
307 }  
308 }  
309 }  
310 }  
311 }  
312 }  
313 }  
314 }  
315 }  
316 }  
317 }  
318 }  
319 }  
320 }  
321 }  
322 }  
323 }  
324 }  
325 }  
326 }  
327 }  
328 }  
329 }  
330 }  
331 }  
332 }  
333 }  
334 }  
335 }  
336 }  
337 }  
338 }  
339 }  
340 }  
341 }  
342 }  
343 }  
344 }  
345 }  
346 }  
347 }  
348 }  
349 }  
350 }  
351 }  
352 }  
353 }  
354 }  
355 }  
356 }  
357 }  
358 }  
359 }  
360 }  
361 }  
362 }  
363 }  
364 }  
365 }  
366 }  
367 }  
368 }  
369 }  
370 }  
371 }  
372 }  
373 }  
374 }  
375 }  
376 }  
377 }  
378 }  
379 }  
380 }  
381 }  
382 }  
383 }  
384 }  
385 }  
386 }  
387 }  
388 }  
389 }  
390 }  
391 }  
392 }  
393 }  
394 }  
395 }  
396 }  
397 }  
398 }  
399 }  
400 }  
401 }  
402 }  
403 }  
404 }  
405 }  
406 }  
407 }  
408 }  
409 }  
410 }  
411 }  
412 }  
413 }  
414 }  
415 }  
416 }  
417 }  
418 }  
419 }  
420 }  
421 }  
422 }  
423 }  
424 }  
425 }  
426 }  
427 }  
428 }  
429 }  
430 }  
431 }  
432 }  
433 }  
434 }  
435 }  
436 }  
437 }  
438 }  
439 }  
440 }  
441 }  
442 }  
443 }  
444 }  
445 }  
446 }  
447 }  
448 }  
449 }  
450 }  
451 }  
452 }  
453 }  
454 }  
455 }  
456 }  
457 }  
458 }  
459 }  
460 }  
461 }  
462 }  
463 }  
464 }  
465 }  
466 }  
467 }  
468 }  
469 }  
470 }  
471 }  
472 }  
473 }  
474 }  
475 }  
476 }  
477 }  
478 }  
479 }  
480 }  
481 }  
482 }  
483 }  
484 }  
485 }  
486 }  
487 }  
488 }  
489 }  
490 }  
491 }  
492 }  
493 }  
494 }  
495 }  
496 }  
497 }  
498 }  
499 }  
500 }  
501 }  
502 }  
503 }  
504 }  
505 }  
506 }  
507 }  
508 }  
509 }  
510 }  
511 }  
512 }  
513 }  
514 }  
515 }  
516 }  
517 }  
518 }  
519 }  
520 }  
521 }  
522 }  
523 }  
524 }  
525 }  
526 }  
527 }  
528 }  
529 }  
530 }  
531 }  
532 }  
533 }  
534 }  
535 }  
536 }  
537 }  
538 }  
539 }  
540 }  
541 }  
542 }  
543 }  
544 }  
545 }  
546 }  
547 }  
548 }  
549 }  
550 }  
551 }  
552 }  
553 }  
554 }  
555 }  
556 }  
557 }  
558 }  
559 }  
560 }  
561 }  
562 }  
563 }  
564 }  
565 }  
566 }  
567 }  
568 }  
569 }  
570 }  
571 }  
572 }  
573 }  
574 }  
575 }  
576 }  
577 }  
578 }  
579 }  
580 }  
581 }  
582 }  
583 }  
584 }  
585 }  
586 }  
587 }  
588 }  
589 }  
590 }  
591 }  
592 }  
593 }  
594 }  
595 }  
596 }  
597 }  
598 }  
599 }  
600 }  
601 }  
602 }  
603 }  
604 }  
605 }  
606 }  
607 }  
608 }  
609 }  
610 }  
611 }  
612 }  
613 }  
614 }  
615 }  
616 }  
617 }  
618 }  
619 }  
620 }  
621 }  
622 }  
623 }  
624 }  
625 }  
626 }  
627 }  
628 }  
629 }  
630 }  
631 }  
632 }  
633 }  
634 }  
635 }  
636 }  
637 }  
638 }  
639 }  
640 }  
641 }  
642 }  
643 }  
644 }  
645 }  
646 }  
647 }  
648 }  
649 }  
650 }  
651 }  
652 }  
653 }  
654 }  
655 }  
656 }  
657 }  
658 }  
659 }  
660 }  
661 }  
662 }  
663 }  
664 }  
665 }  
666 }  
667 }  
668 }  
669 }  
670 }  
671 }  
672 }  
673 }  
674 }  
675 }  
676 }  
677 }  
678 }  
679 }  
680 }  
681 }  
682 }  
683 }  
684 }  
685 }  
686 }  
687 }  
688 }  
689 }  
690 }  
691 }  
692 }  
693 }  
694 }  
695 }  
696 }  
697 }  
698 }  
699 }  
700 }  
701 }  
702 }  
703 }  
704 }  
705 }  
706 }  
707 }  
708 }  
709 }  
710 }  
711 }  
712 }  
713 }  
714 }  
715 }  
716 }  
717 }  
718 }  
719 }  
720 }  
721 }  
722 }  
723 }  
724 }  
725 }  
726 }  
727 }  
728 }  
729 }  
730 }  
731 }  
732 }  
733 }  
734 }  
735 }  
736 }  
737 }  
738 }  
739 }  
740 }  
741 }  
742 }  
743 }  
744 }  
745 }  
746 }  
747 }  
748 }  
749 }  
750 }  
751 }  
752 }  
753 }  
754 }  
755 }  
756 }  
757 }  
758 }  
759 }  
760 }  
761 }  
762 }  
763 }  
764 }  
765 }  
766 }  
767 }  
768 }  
769 }  
770 }  
771 }  
772 }  
773 }  
774 }  
775 }  
776 }  
777 }  
778 }  
779 }  
780 }  
781 }  
782 }  
783 }  
784 }  
785 }  
786 }  
787 }  
788 }  
789 }  
790 }  
791 }  
792 }  
793 }  
794 }  
795 }  
796 }  
797 }  
798 }  
799 }  
800 }  
801 }  
802 }  
803 }  
804 }  
805 }  
806 }  
807 }  
808 }  
809 }  
810 }  
811 }  
812 }  
813 }  
814 }  
815 }  
816 }  
817 }  
818 }  
819 }  
820 }  
821 }  
822 }  
823 }  
824 }  
825 }  
826 }  
827 }  
828 }  
829 }  
830 }  
831 }  
832 }  
833 }  
834 }  
835 }  
836 }  
837 }  
838 }  
839 }  
840 }  
841 }  
842 }  
843 }  
844 }  
845 }  
846 }  
847 }  
848 }  
849 }  
850 }  
851 }  
852 }  
853 }  
854 }  
855 }  
856 }  
857 }  
858 }  
859 }  
860 }  
861 }  
862 }  
863 }  
864 }  
865 }  
866 }  
867 }  
868 }  
869 }  
870 }  
871 }  
872 }  
873 }  
874 }  
875 }  
876 }  
877 }  
878 }  
879 }  
880 }  
881 }  
882 }  
883 }  
884 }  
885 }  
886 }  
887 }  
888 }  
889 }  
890 }  
891 }  
892 }  
893 }  
894 }  
895 }  
896 }  
897 }  
898 }  
899 }  
900 }  
901 }  
902 }  
903 }  
904 }  
905 }  
906 }  
907 }  
908 }  
909 }  
910 }  
911 }  
912 }  
913 }  
914 }  
915 }  
916 }  
917 }  
918 }  
919 }  
920 }  
921 }  
922 }  
923 }  
924 }  
925 }  
926 }  
927 }  
928 }  
929 }  
930 }  
931 }  
932 }  
933 }  
934 }  
935 }  
936 }  
937 }  
938 }  
939 }  
940 }  
941 }  
942 }  
943 }  
944 }  
945 }  
946 }  
947 }  
948 }  
949 }  
950 }  
951 }  
952 }  
953 }  
954 }  
955 }  
956 }  
957 }  
958 }  
959 }  
960 }  
961 }  
962 }  
963 }  
964 }  
965 }  
966 }  
967 }  
968 }  
969 }  
970 }  
971 }  
972 }  
973 }  
974 }  
975 }  
976 }  
977 }  
978 }  
979 }  
980 }  
981 }  
982 }  
983 }  
984 }  
985 }  
986 }  
987 }  
988 }  
989 }  
990 }  
991 }  
992 }  
993 }  
994 }  
995 }  
996 }  
997 }  
998 }  
999 }  
1000 }
```

Conclusiones y Ejercicio

- Todo este proceso de traducir, te lo puedes ahorrar haciéndola directamente en español.
- Ahora la puedes probar incluso en tu Echo o dispositivo Alexa con tu cuenta.
- Hacer una App que te devuelva la dirección del Virgen de Gracia y el Teléfono en la misma frase (aún no sabemos usar slots). Puedes basarte en la mía del departamento.
- Código de ejemplo tutorial Amazon: <https://github.com/germanviscuso/ASKVideoSeries/tree/master/02>
- Código de la Skill Informática: <https://github.com/joseluisgs/informatica-skill>
- Tutoriales seguidos:
 - <https://www.youtube.com/watch?v=mqlk4-Ry-cg>
 - <https://www.youtube.com/watch?v=x94qgqw02R4>

