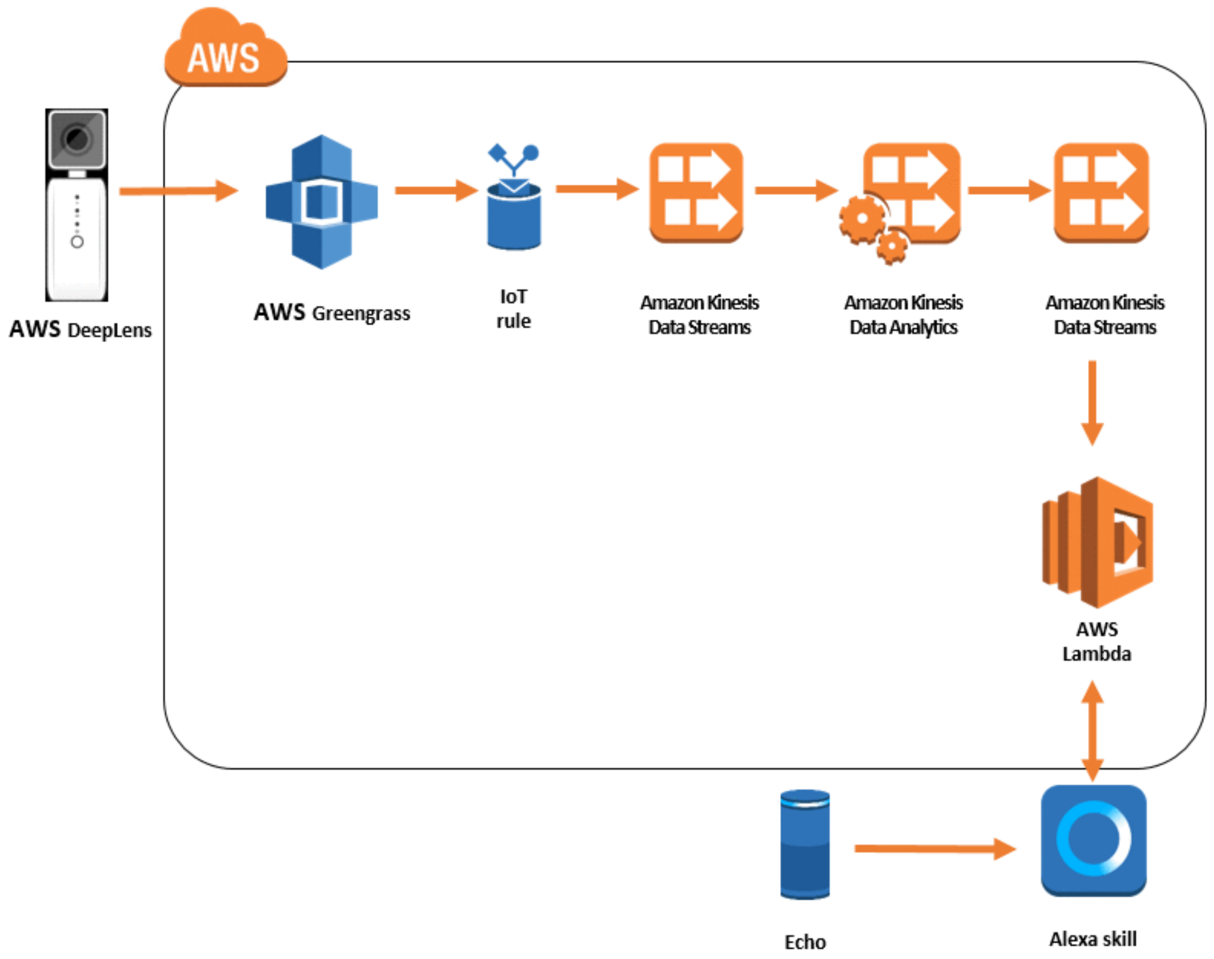
# Playing “QuickDraw” game on AWS DeepLens and Alexa Echo Device via Alexa Skills

**A brief intro of Quick Draw game**: QuickDraw allows people to interact with Alexa by drawing objects on a whiteboard with a black background or a non-noisy background that Alexa asks the players to draw within 20 seconds. If what the players draw matches what Alexa asks, they get 10 points. After 6 rounds, whoever has the highest score wins.

A high-level overview of the system for this game:


AWS
AWS DeepLens
AWS Greengrass
IoT rule
Amazon Kinesis Data Streams
Amazon Kinesis Data Analytics
Amazon Kinesis Data Streams
AWS Lambda
Echo
Alexa skill

## Setting up DeepLens and deploying the Quick Draw model

1. Open the AWS DeepLens console.
2. Register your AWS DeepLens device if it’s not registered. You can follow this link for a step by step guide to register the device.
3. Import model:

   On the S3 buckets page from S3 console, choose **Create bucket** and enter a name for a bucket that starts with deeplens-. Then, upload the **quickdraw_model.tar.gz** from the **QuickDrawGame.zip** file.

   After successfully upload it, go back to DeepLens console. Choose **Import model to DeepLens** on the Models page and choose **Externally trained**

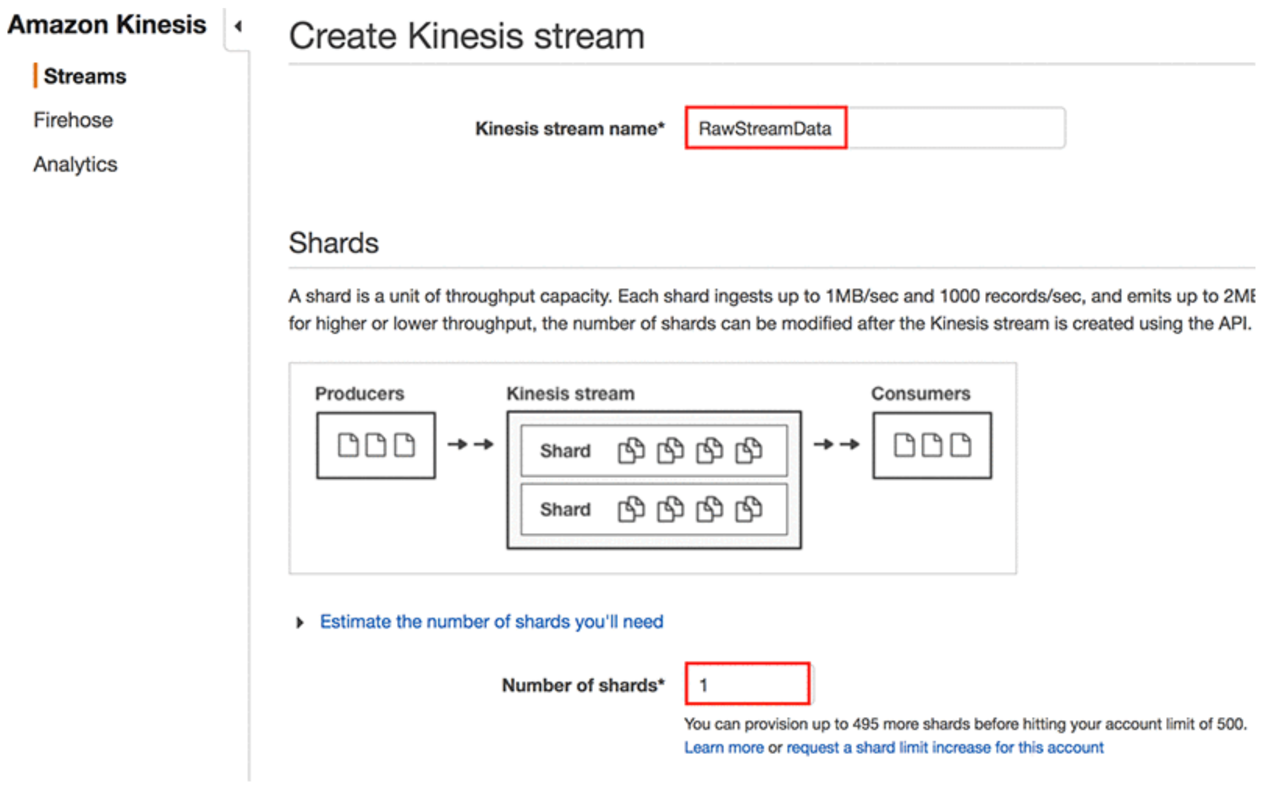
**model.** Enter the model path from S3, the model name that starts with deeplens- and choose **MXNet** for the model framework. Click **Import model.**

4. Create the inference Lambda function: refer to this link to create a Lambda function for this game. However, instead of using the .zip file they provide in this link, use the **deeplens-quickdraw-function.zip**.
5. Choose Create new project on the Projects page. On the Choose project type page, choose the Create a new blank project option and click Next. On the Specify project details page, fill out the project name (e.g deeplens-quick-draw-game). Below that, click **Add model** -> choose the model you imported earlier. Click **Add function** -> choose the function you imported earlier. Then **Create** and proceed.
6. Choose the project and choose **Deploy to device.**

Now you have successfully deployed to the project to DeepLens.

## Setting up Kinesis Streams and Kinesis Analytics

1. Open the Amazon Kinesis Data Streams console.
2. Create a new Kinesis Data Stream. Give it a name that indicates it's for raw incoming stream data—for example, RawStreamData. For Number of shards, type 1.



Kinesis stream name* RawStreamData

Shards

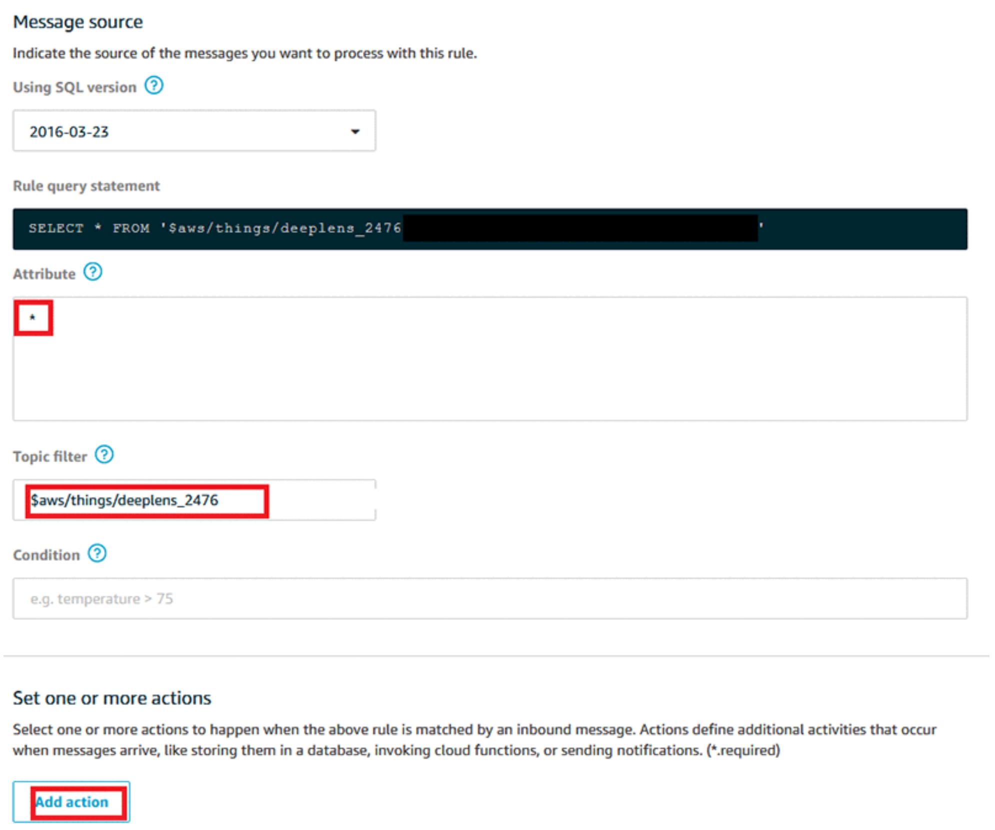
A shard is a unit of throughput capacity. Each shard ingests up to 1MB/sec and 1000 records/sec, and emits up to 2MI
for higher or lower throughput, the number of shards can be modified after the Kinesis stream is created using the API.

Number of shards* 1

You can provision up to 495 more shards before hitting your account limit of 500.
Learn more or request a shard limit increase for this account

3. Go back to the AWS IoT console and in the left navigation pane choose Act. Then choose Create to set up a rule to push MQTT messages from the AWS DeepLens device to the newly created Kinesis data stream.

4. On the create rule page, give it a name. In the Message source section, for Attribute enter *. For Topic filter, enter the DeepLens device MQTT topic. Choose Add action.



Message source

Indicate the source of the messages you want to process with this rule.

Using SQL version: 2016-03-23

Rule query statement

```
SELECT * FROM '$aws/things/deeplens_2476[redacted]'
```

Attribute: *

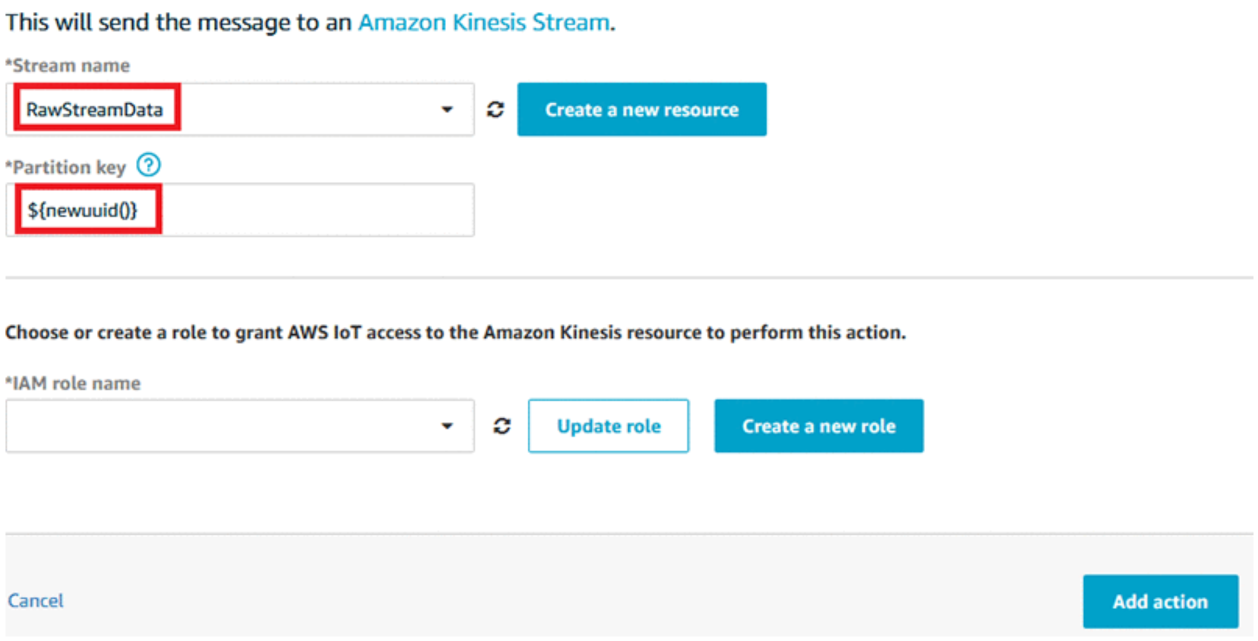
Topic filter: $aws/things/deeplens_2476

Condition: e.g. temperature > 75

Set one or more actions

Select one or more actions to happen when the above rule is matched by an inbound message. Actions define additional activities that occur when messages arrive, like storing them in a database, invoking cloud functions, or sending notifications. (*.required)

5. Choose Sends message to an Amazon Kinesis Stream, then click Configuration action. Select the Kinesis data stream created earlier, and for Partition key type ${newuuid()} in the text box. Then choose Create a new role or Update role and choose Add action. Choose Create rule to finish the setup.



This will send the message to an Amazon Kinesis Stream.

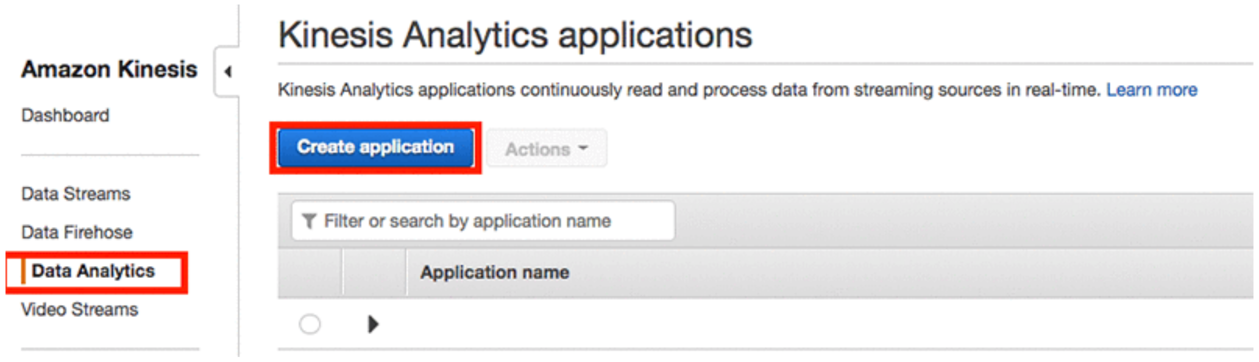
*Stream name: RawStreamData

*Partition key: ${newuuid()}

Choose or create a role to grant AWS IoT access to the Amazon Kinesis resource to perform this action.

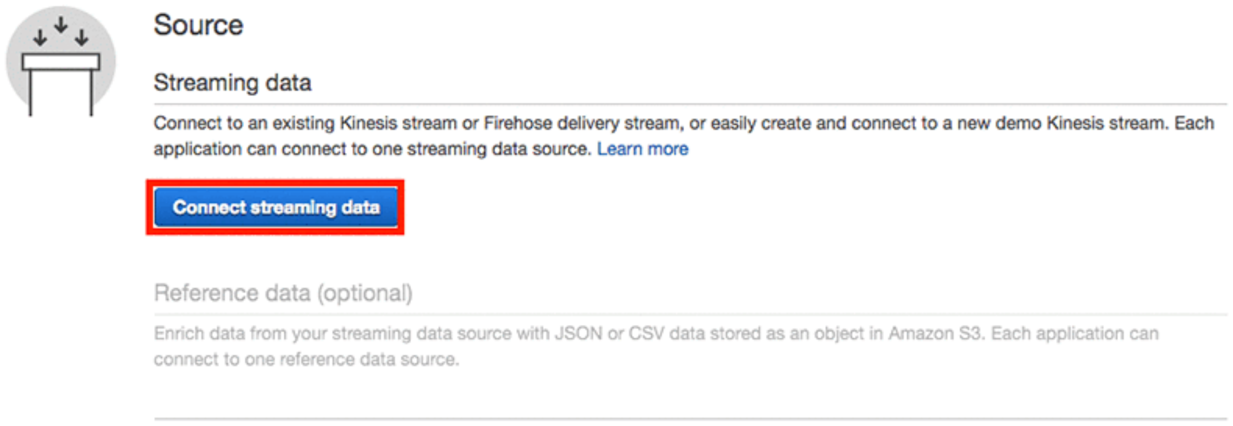
*IAM role name:

6. Now that the rule is set up, messages will be loaded into the Kinesis data stream. Now we can use Kinesis Data Analytics to aggregate the data and load the result to the final Kinesis data stream.
7. Open the Amazon Kinesis Data Streams console.
8. Create another new Kinesis Data Stream (follow instruction steps 1 and 2). Give it a name that indicates that it's for aggregated incoming stream data—for example, AggregatedDataOutput. For Number of shards, type 1.
9. In the Amazon Kinesis Data Analytics console, create a new application.


Kinesis Analytics applications

Kinesis Analytics applications continuously read and process data from streaming sources in real-time. Learn more

10. Give it a name and choose Create application. In the source section, choose Connect stream data.


Source

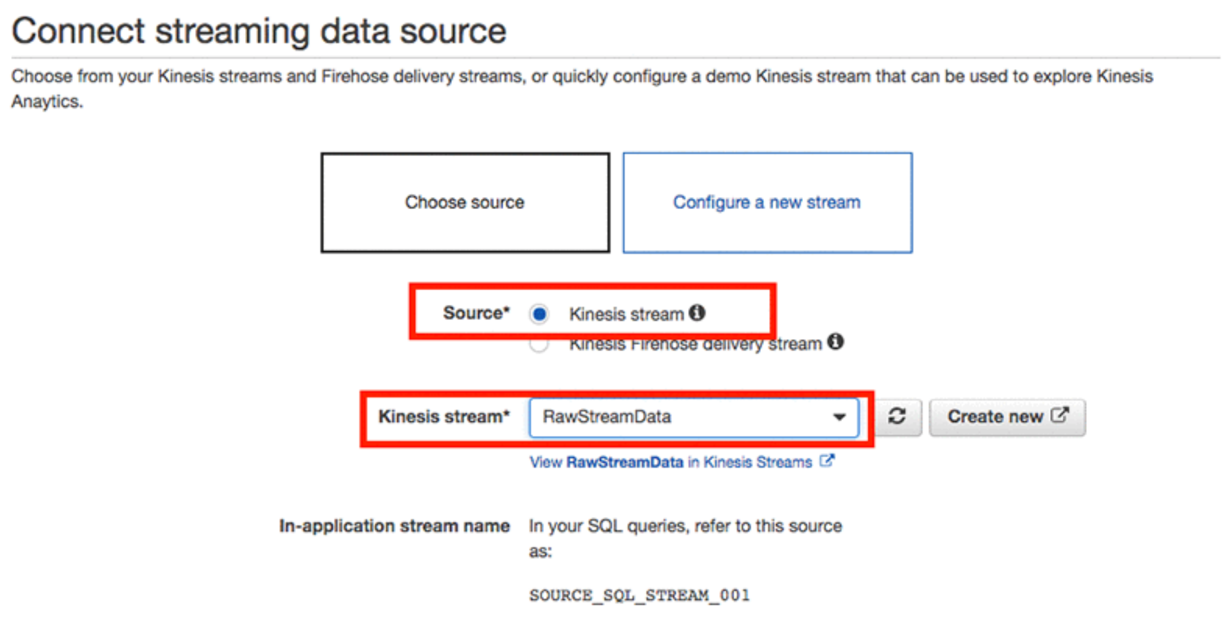
Streaming data

Connect to an existing Kinesis stream or Firehose delivery stream, or easily create and connect to a new demo Kinesis stream. Each application can connect to one streaming data source. Learn more

Reference data (optional)

Enrich data from your streaming data source with JSON or CSV data stored as an object in Amazon S3. Each application can connect to one reference data source.

11. Select Kinesis stream as Source and select the source stream created in step 2. Choose Discover schema to let Kinesis Data Streams to auto-discover the data schema.



Connect streaming data source

Choose from your Kinesis streams and Firehose delivery streams, or quickly configure a demo Kinesis stream that can be used to explore Kinesis Anaytics.

Source* [✓ Kinesis stream] Kinesis Firehose delivery stream

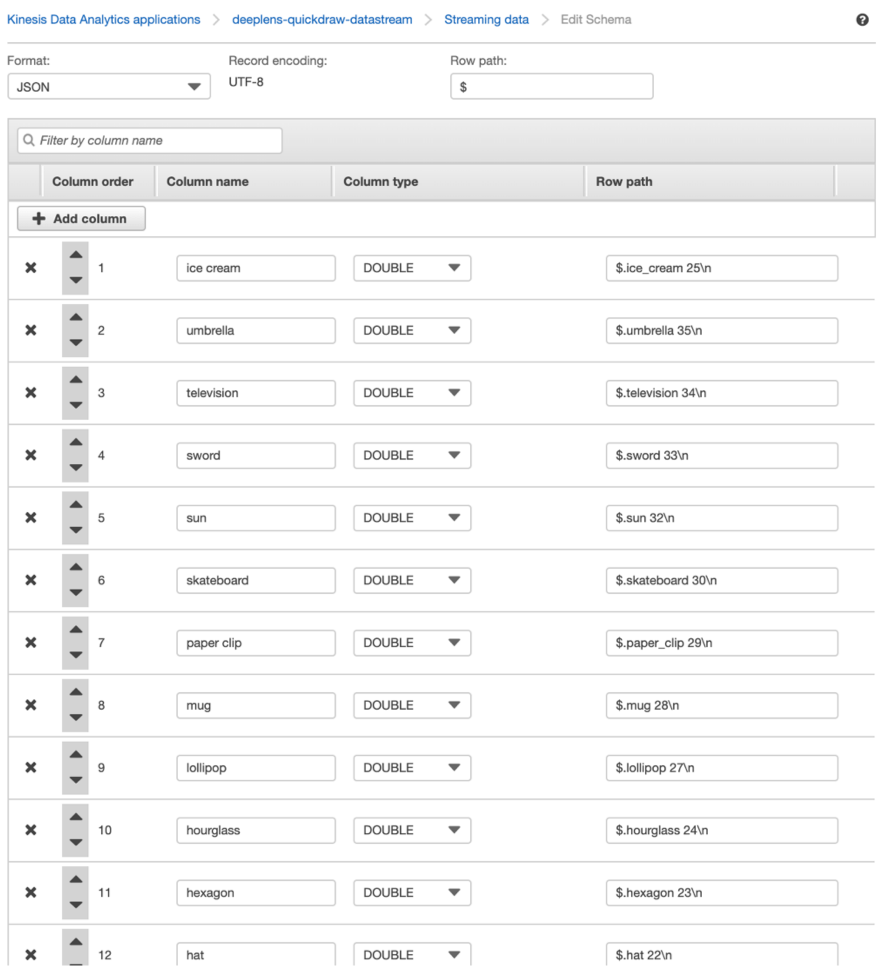
Kinesis stream*: RawStreamData

In-application stream name: In your SQL queries, refer to this source as:

SOURCE_SQL_STREAM_001

12. The QuickDraw model deployment can detect up to 36 objects. You can add 36 objects manually to the schema by choosing the Edit schema button.

13. On the schema page, add the rest of the objects then choose Save schema and update stream. Wait for the update to complete then click Exit (done).



Kinesis Data Analytics applications > deeplens-quickdraw-datastream > Streaming data > Edit Schema

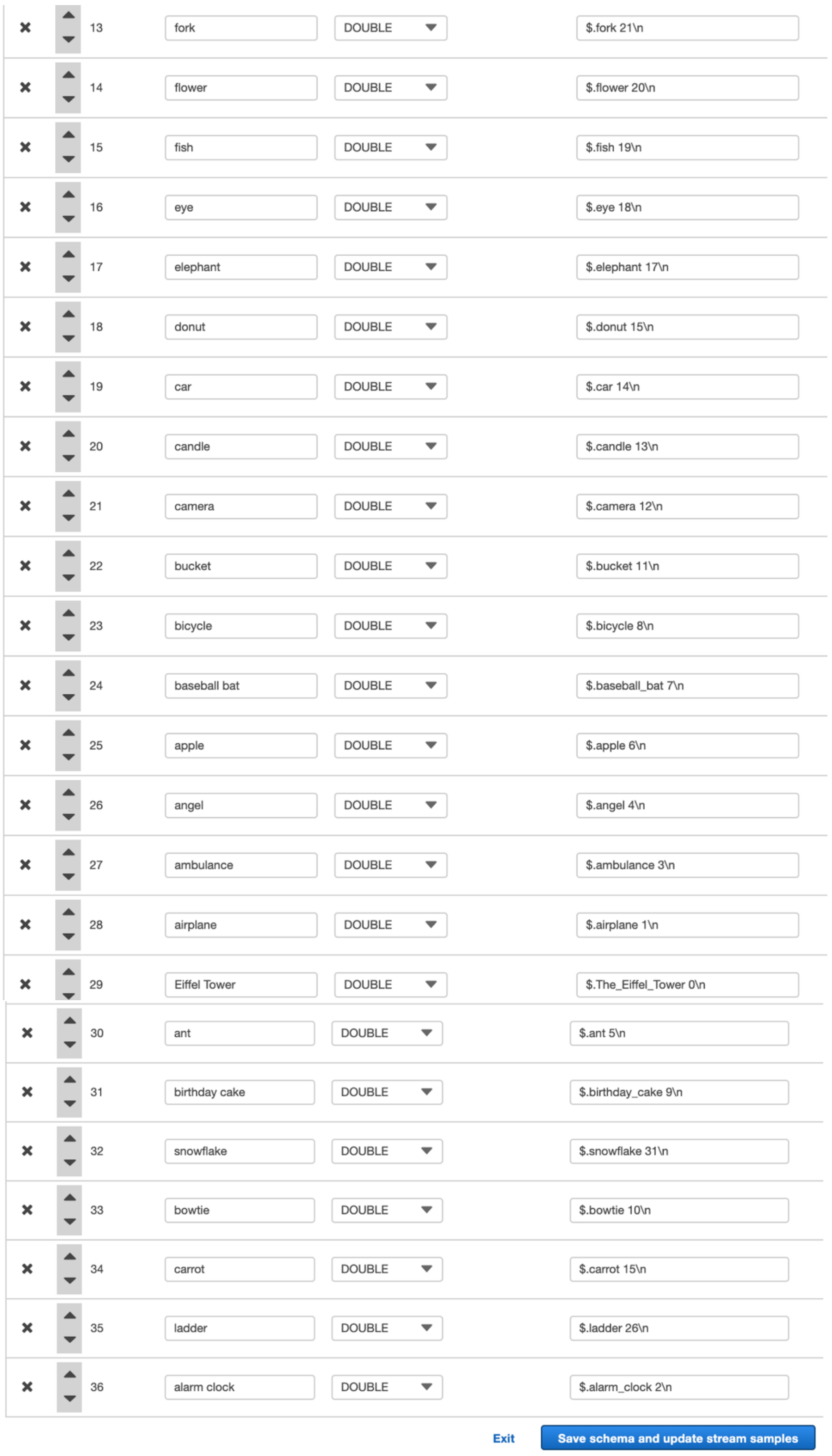
Format: JSON

Record encoding: UTF-8

Row path: $

Filter by column name
Add column

| Column order | Column name | Column type | Row path |
|---|---|---|---|
| 1 | ice cream | DOUBLE | $.ice_cream 25\n |
| 2 | umbrella | DOUBLE | $.umbrella 35\n |
| 3 | television | DOUBLE | $.television 34\n |
| 4 | sword | DOUBLE | $.sword 33\n |
| 5 | sun | DOUBLE | $.sun 32\n |
| 6 | skateboard | DOUBLE | $.skateboard 30\n |
| 7 | paper clip | DOUBLE | $.paper_clip 29\n |
| 8 | mug | DOUBLE | $.mug 28\n |
| 9 | lollipop | DOUBLE | $.lollipop 27\n |
| 10 | hourglass | DOUBLE | $.hourglass 24\n |
| 11 | hexagon | DOUBLE | $.hexagon 23\n |
| 12 | hat | DOUBLE | $.hat 22\n |



| | | | | |
|---|---|---|---|---|
| 13 | fork | DOUBLE | $.fork 21\n |
| 14 | flower | DOUBLE | $.flower 20\n |
| 15 | fish | DOUBLE | $.fish 19\n |
| 16 | eye | DOUBLE | $.eye 18\n |
| 17 | elephant | DOUBLE | $.elephant 17\n |
| 18 | donut | DOUBLE | $.donut 15\n |
| 19 | car | DOUBLE | $.car 14\n |
| 20 | candle | DOUBLE | $.candle 13\n |
| 21 | camera | DOUBLE | $.camera 12\n |
| 22 | bucket | DOUBLE | $.bucket 11\n |
| 23 | bicycle | DOUBLE | $.bicycle 8\n |
| 24 | baseball bat | DOUBLE | $.baseball_bat 7\n |
| 25 | apple | DOUBLE | $.apple 6\n |
| 26 | angel | DOUBLE | $.angel 4\n |
| 27 | ambulance | DOUBLE | $.ambulance 3\n |
| 28 | airplane | DOUBLE | $.airplane 1\n |
| 29 | Eiffel Tower | DOUBLE | $.The_Eiffel_Tower 0\n |
| 30 | ant | DOUBLE | $.ant 5\n |
| 31 | birthday cake | DOUBLE | $.birthday_cake 9\n |
| 32 | snowflake | DOUBLE | $.snowflake 31\n |
| 33 | bowtie | DOUBLE | $.bowtie 10\n |
| 34 | carrot | DOUBLE | $.carrot 15\n |
| 35 | ladder | DOUBLE | $.ladder 26\n |
| 36 | alarm clock | DOUBLE | $.alarm_clock 2\n |

14. Scroll down to the bottom of the page then choose Save and continue.
15. Back on the application configuration page, choose Go to SQL editor.
16. Copy and paste the following SQL statement to the Analytics SQL window, and then choose Save and run SQL. After SQL finishes saving and running, choose Close at the bottom of the page.

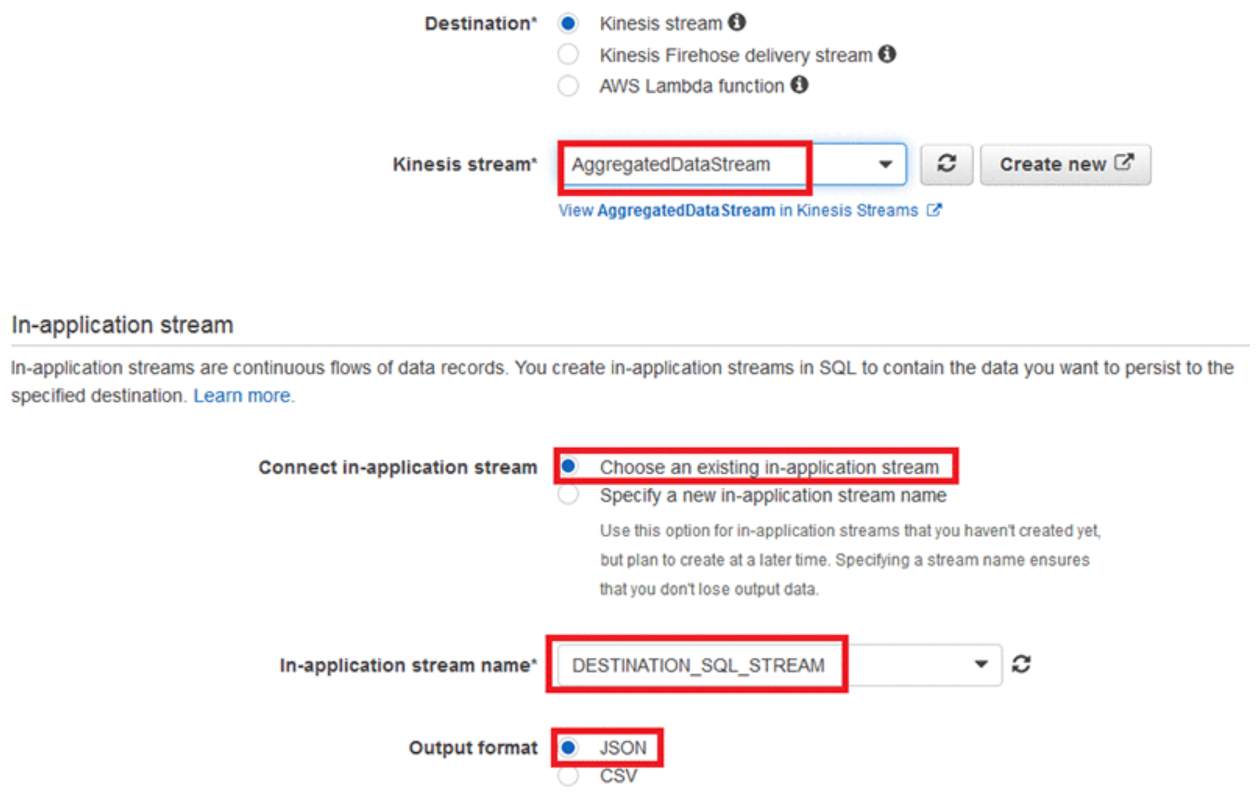
```
/**
 * Welcome to the SQL editor
 * =========================
 *
 * The SQL code you write here will continuously transform your streaming data
 * when your application is running.
 *
 * Get started by clicking "Add SQL from templates" or pull up the
 * documentation and start writing your own custom queries.
 */


-- ** Continuous Filter **
-- Performs a continuous filter based on a WHERE condition.
--          .----------.   .----------.   .----------.
--          |  SOURCE  |   |  INSERT  |   |  DESTIN. |
-- Source-->|  STREAM  |-->| & SELECT |-->|  STREAM  |-->Destination
--          |          |   |  (PUMP)  |   |          |
--          '----------'   '----------'   '----------'
-- STREAM (in-application): a continuously updated entity that you can SELECT from and INSERT into like
a TABLE
-- PUMP: an entity used to continuously 'SELECT ... FROM' a source STREAM, and INSERT SQL results into
an output STREAM
-- Create output stream, which can be used to send to a destination
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" ("Eiffel Tower" DOUBLE, "airplane" DOUBLE, "alarm
clock" DOUBLE,
"ambulance" DOUBLE, "angel" DOUBLE, "ant" DOUBLE, "apple" DOUBLE, "baseball bat" DOUBLE, "bicycle"
DOUBLE, "birthday cake" DOUBLE,
"bowtie" DOUBLE, "bucket" DOUBLE, "camera" DOUBLE, "candle" DOUBLE, "car" DOUBLE, "carrot" DOUBLE,
"donut" DOUBLE, "elephant" DOUBLE,
"eye" DOUBLE, "fish" DOUBLE, "flower" DOUBLE, "fork" DOUBLE, "hat" DOUBLE, "hexagon" DOUBLE,
"hourglass" DOUBLE, "ice cream" DOUBLE, "ladder" DOUBLE,
"lollipop" DOUBLE, "mug" DOUBLE, "paper clip" DOUBLE, "skateboard" DOUBLE, "snowflake" DOUBLE, "sun"
DOUBLE, "sword" DOUBLE, "television" DOUBLE,
"umbrella" DOUBLE);
-- Create pump to insert into output
CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
-- Select all columns from source stream
SELECT STREAM "Eiffel Tower", "airplane",
"alarm clock", "ambulance", "angel", "ant", "apple", "baseball bat", "bicycle",
"birthday cake", "bowtie", "bucket", "camera", "candle", "car", "carrot", "donut", "elephant",
"eye", "fish", "flower", "fork", "hat", "hexagon", "hourglass", "ice cream", "ladder", "lollipop",
"mug", "paper clip", "skateboard", "snowflake", "sun", "sword", "television", "umbrella"

FROM "SOURCE_SQL_STREAM_001";
```

17. Back on application configuration page, choose Connect to a destination. In the Analytics destination section, make sure the Kinesis data stream (for example, AggregatedDataOutput) created in step 8 is selected, and enter

DESTINATION_SQL_STREAM as the In-application stream name and select JSON as the output format of. Note that you can also easily have Kinesis Data Analytics send data directly to a Lambda function. That Lambda function would write to other data stores, such as Amazon DynamoDB. Then have the Alexa read from DynamoDB upon user request.



Destination* [✓ Kinesis stream] Kinesis Firehose delivery stream AWS Lambda function

Kinesis stream* AggregatedDataStream

View AggregatedDataStream in Kinesis Streams

In-application stream

In-application streams are continuous flows of data records. You create in-application streams in SQL to contain the data you want to persist to the specified destination. Learn more.

Connect in-application stream [✓ Choose an existing in-application stream] Specify a new in-application stream name

Use this option for in-application streams that you haven't created yet, but plan to create at a later time. Specifying a stream name ensures that you don't lose output data.

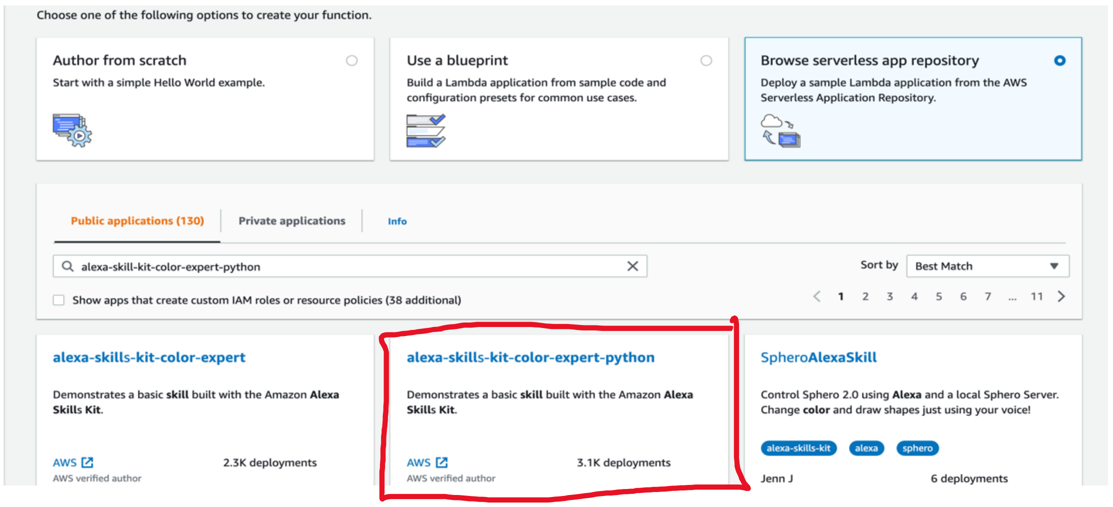
In-application stream name* DESTINATION_SQL_STREAM

Output format [✓ JSON] CSV

18. Messages are now aggregated and loaded into the final Kinesis data stream (for example, AggregatedDataOutput) that was created in step 8. Next, you will create an Alexa custom skill with AWS Lambda.

## Creating Alexa custom skill with AWS Lambda

1. Open the AWS Lambda console and create a new function.
2. The easiest way to create an Alexa skill is to create the function from the existing serverless app repository provided by AWS Lambda and then overwrite the code with your own.



Choose one of the following options to create your function.

Author from scratch
Start with a simple Hello World example.

Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

Browse serverless app repository [✓]
Deploy a sample Lambda application from the AWS Serverless Application Repository.

Public applications (130) | Private applications | Info

alexa-skill-kit-color-expert-python

Sort by: Best Match

Show apps that create custom IAM roles or resource policies (38 additional)

< 1 2 3 4 5 6 7 ... 11 >

alexa-skills-kit-color-expert
Demonstrates a basic **skill** built with the Amazon **Alexa Skills Kit**.
AWS — AWS verified author — 2.3K deployments

alexa-skills-kit-color-expert-python
Demonstrates a basic **skill** built with the Amazon **Alexa Skills Kit**.
AWS — AWS verified author — 3.1K deployments

SpheroAlexaSkill
Control Sphero 2.0 using **Alexa** and a local Sphero Server. Change **color** and draw shapes just using your voice!
alexa-skills-kit, alexa, sphero
Jenn J — 6 deployments

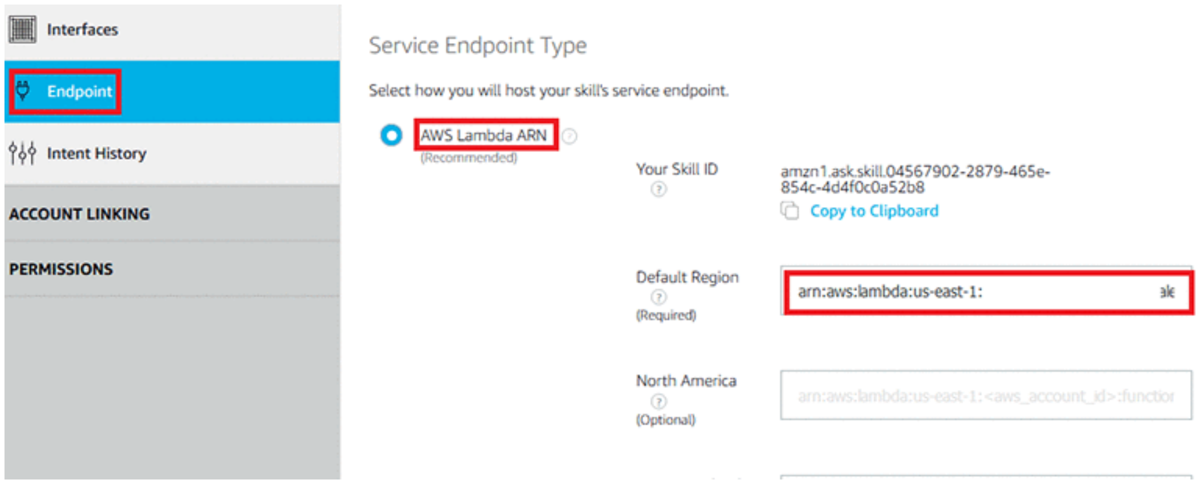
3. It is a security best practice to enable Alexa Skill ID when using a Lambda function. If you have not created a skill for this yet, you can disable it for now and then re-enable it later by adding another Alexa trigger to the Lambda function.
4. Copy the code from **alexa_lamda_function.py** and replace the sample code provided by the template. This code reads data from the Kinesis data stream and returns the result back to Alexa. Note: Change the default Northern Virginia AWS Region (us-east-1) in the code if you are following along in other Regions. Look for the following code to change region, kinesis = boto3.client('kinesis', region_name='us-east-1')

## Setting up an Alexa custom skill with AWS Lambda

1. Open the Amazon Alexa Developer Portal, and choose Create a new custom skill.
2. You can upload the JSON document in the Alexa Skill JSON Editor to automatically configure intents and sample utterances for the custom skill. Be sure to click **Save Model** to apply the changes. (Double-click to the code below to get into full code

```
{
    "interactionModel": {
        "languageModel": {
            "invocationName": "quick draw with deep lens",
            "modelConfiguration": {
                "fallbackIntentSensitivity": {
                    "level": "LOW"
                }
            },
            "intents": [
                {
                    "name": "StartGame",
                    "slots": [],
                    "samples": [
                        "play now",
                        "I'm ready",
                        "ready",
                        "play game"
                    ]
                },
                {
                    "name": "AMAZON.HelpIntent",
                    "samples": [
                        "repeat the rules",
                        "help",
                        "tell me the rules",
                        "help me with the rules",
                        "i am not familiar with the rules"
                    ]
                },
                {
                    "name": "AMAZON.StopIntent",
                    "samples": [
                        "quit",
                        "cancel",
                        "quit the game",
                        "cancel the game"
                    ]
                },
                {
                    "name": "AMAZON.NavigateHomeIntent",
                    "samples": []
                },
                {
                    "name": "AMAZON.YesIntent",
                    "samples": [
                        "please",
                        "sure",
                        "yes"
                    ]
```

3. Finally, in the endpoint section, enter the AWS Lambda function's Amazon Resource Name (ARN) that's created for Alexa Skill part. Your Alexa Skill is set and ready to tell you the objects detected by the AWS DeepLens device. You can wake Alexa up by saying "Alexa, play Quick Draw with DeepLens", and the game will start.



Service Endpoint Type

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

AWS Lambda ARN (Recommended)

Your Skill ID: amzn1.ask.skill.04567902-2879-465e-854c-4d4f0c0a52b8 Copy to Clipboard

Default Region (Required): arn:aws:lambda:us-east-1:

North America (Optional): arn:aws:lambda:us-east-1:<aws_account_id>:functior

## Setting up DeepLens and a whiteboard (white paper)

You will need to have a DeepLens located in front of a whiteboard or whitepaper so that it's about 2 feet apart. Make sure the background of a whiteboard or paper is clear and non-noisy.

The following picture is a setup with a whiteboard (using a black tape around the edges of the board to make sure the edges of the white rectangle are clear.