Can You Roll Your Own Virtual Assistant?

a.k.a. Talking to the DIYVA
(Do It Yourself Virtual Assistant)
How Did This All Start?

a.k.a. Every DIYVA Has an Origin Story
From WAY Back
How Did This All Start?

a.k.a. Every DIYVA Has an Origin Story
Enter Lex & Polly
presentation format
Demo the DIYVA
Setting Up DIYVA

- Login with Amazon
- AWS Cognito
- AWS Lex & AWS Polly
- Apache Cordova
Setting Up

LOGIN WITH amazon

GREG BULMASH / @YIDDISHNINJA / TECHNICAL EVANGELIST / LOGIN WITH AMAZON
Go to developer.amazon.com

and click on Login with Amazon
Create a Security Profile

Name your new Security Profile

Choose a name for this security profile. You can create multiple security profiles. You will associate a security profile with one or more apps. Apps that use the same security profile can share some types of data (for example, a "My App - Free" and a "My App - HD" could share data). For a shared security profile, choose a name that applies to all the apps that will use it (for example, "My App profile"). Learn More

* Indicates a required field

Security Profile Name *
DIYVA

Security Profile Description *
Do It Yourself Virtual Assistant

Consent Privacy Notice URL *
https://diyva.example.com

Consent Logo Image

Save
Manage your security profile

Login with Amazon

Login with Amazon allows users to login to registered third party websites or apps ('clients') using their Amazon user name and password. Clients may ask the user to share some personal information from their Amazon profile, including name, email address, and zip code. To get started, select an existing Security Profile or create a new Security Profile. Learn More

Create a New Security Profile

Login with Amazon Configurations

<table>
<thead>
<tr>
<th>Security Profile Name</th>
<th>OAuth2 Credentials</th>
<th>Manage</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIYVA</td>
<td>Show Client ID and Client Secret</td>
<td>Security Profile</td>
</tr>
<tr>
<td>LWA Hackathon Demos</td>
<td>Show Client ID and Client Secret</td>
<td></td>
</tr>
</tbody>
</table>
Copy Your Profile & Client IDs

<table>
<thead>
<tr>
<th>Security Profile Name</th>
<th>DIYVA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security Profile Description</td>
<td>Do It Yourself Virtual Assistant</td>
</tr>
<tr>
<td>Security Profile ID</td>
<td>No ID listed</td>
</tr>
<tr>
<td>Client ID</td>
<td>No ID listed</td>
</tr>
<tr>
<td>Client Secret</td>
<td>No ID listed</td>
</tr>
<tr>
<td>Consent Privacy Notice URL</td>
<td><a href="http://diyva.example.com">http://diyva.example.com</a></td>
</tr>
<tr>
<td>Consent Logo Image</td>
<td>![Logo Image]</td>
</tr>
</tbody>
</table>
Set Your Origin

DIYVA - Security Profile

To use Login with Amazon with a website, you must specify either an allowed JavaScript origin (for the Implicit grant) or an allowed return URL (for the Authorization Code grant). If you are using Pay with Amazon, you must specify an allowed JavaScript origin. [Learn More]

- **Allowed Origins**: https://yiddishninja.github.io
- **Allowed Return URLs**: https://yiddishninja.github.io/LWA-proxy/signin.html

[Edit]
Setting Up Cognito
Go to Cognito from the Console

AWS services

Find a service by name or feature (for example, EC2, S3 or VM, storage).

Recently visited services
- Lambda
- Cognito
- Polly
- IAM
- Lex

All services
- Compute
  - EC2
  - EC2 Container Service
  - Lightsail
  - Elastic Beanstalk
  - Lambda
  - Batch
- Storage
  - S3
  - EFS
  - Glacier
  - Storage Gateway
- Developer Tools
  - CodeStar
  - CodeCommit
  - CodeBuild
  - CodeDeploy
  - CodePipeline
  - X-Ray
- Management Tools
  - CloudWatch
  - CloudFormation
  - CloudTrail
  - Config
  - CostExplorer
- Internet of Things
  - AWS IoT
- Contact Center
  - Amazon Connect
- Game Development
  - Amazon GameLift
- Mobile Services
  - Mobile Hub
  - Cognito
Amazon Cognito

Amazon Cognito makes it easy for you to have users sign up and sign in to your apps, federate identities from social identity providers, secure access to AWS resources and synchronize data across multiple devices, platforms, and applications.

Manage your User Pools  Manage Federated Identities
We’re off to use the Wizard

Getting started wizard

Step 1: Create identity pool
Step 2: Set permissions

Create new identity pool

Identity pools are used to store end user identities. To declare a new identity pool, enter a unique name.

Identity pool name*: DIYVA
Example: My App Name

Unauthenticated identities

Amazon Cognito can support unauthenticated identities by providing a unique identifier and AWS credentials for users who do not authenticate with an identity provider. If your application allows customers to use the application without logging in, you can enable access for unauthenticated identities. Learn more about unauthenticated identities.

☐ Enable access to unauthenticated identities

Authentication providers

Amazon Cognito supports the following authentication methods with Amazon Cognito Sign-In or any public provider. If you allow your users to authenticate using any of these public providers, you can specify your application identifiers here. Warning: Changing the application ID that your identity pool is linked to will prevent existing users from authenticating using Amazon Cognito. Learn more about public identity providers.

<table>
<thead>
<tr>
<th>Cognito</th>
<th>Amazon</th>
<th>Facebook</th>
<th>Google+</th>
<th>Twitter / Digits</th>
<th>OpenID</th>
<th>SAML</th>
<th>Custom</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Amazon App ID: amzn1.application....
Example: amzn1.application.188a56d827a7d6555a8b67a5d
Create a new IAM role

Your Cognito identities require access to your resources

Assigning a role to your application end users helps you restrict access to your AWS resources. Amazon Cognito integrates with Identity and Access Management (IAM) and lets you select specific roles for both your authenticated and unauthenticated identities. Learn more about IAM.

By default, Amazon Cognito creates a new role with limited permissions - end users only have access to Cognito Sync and Mobile Analytics. You can modify the roles if your application needs access to other AWS resources, such as S3 or DynamoDB.

Hide Details

Role Summary

<table>
<thead>
<tr>
<th>Role</th>
<th>Your authenticated identities would like access to Cognito.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>IAM Role</td>
<td>Create a new IAM Role</td>
</tr>
<tr>
<td>Role Name</td>
<td>Cognito_DiyvaAuth_Role</td>
</tr>
</tbody>
</table>

Don't Allow  Allow
Get Your Pool ID

Edit identity pool

From this page you can modify the details of your identity pool. An identity pool must have a unique name and a set of authenticated and unauthenticated roles. The roles are saved with your identity pool and whenever we receive a request to authorize a user we will automatically utilize the roles you specify here. You will be required to specify the identity pool id from this page when initializing the Amazon Cognito client SDK. Learn more about using IAM roles with Amazon Cognito.

Identity pool name*  
Diyva

Identity pool ID

Unauthenticated role  
Cognito_DiyvaUnauth_Role

Authenticated role  
Cognito_DiyvaAuth_Role
Welcome to Identity and Access Management

IAM users sign-in link:

IAM Resources

<table>
<thead>
<tr>
<th>Users: 0</th>
<th>Roles: 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Groups: 0</td>
<td>Identity Providers: 0</td>
</tr>
<tr>
<td>Customer Managed Policies: 0</td>
<td></td>
</tr>
</tbody>
</table>

Security Status

- Activate MFA on your root account
- Create individual IAM users
- Use groups to assign permissions
- Apply an IAM password policy

1 out of 4 complete.
Select the Auth_Role

<table>
<thead>
<tr>
<th>Role name</th>
<th>Description</th>
<th>Creation Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cognito_DiysaAuth_Role</td>
<td></td>
<td>2017-05-06 22:31 CDT</td>
</tr>
<tr>
<td>Cognito_DiysaUnauth_Role</td>
<td></td>
<td>2017-05-08 11:14 CDT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2017-05-06 22:31 CDT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2017-05-08 11:14 CDT</td>
</tr>
<tr>
<td>IncidenceResponseClientRole-DO-NOT-DELETE</td>
<td></td>
<td>2017-05-06 22:31 CDT</td>
</tr>
</tbody>
</table>
We need to attach policies
## Adding Lex

### Attach Policy

Select one or more policies to attach. Each role can have up to 10 policies attached.

<table>
<thead>
<tr>
<th>Policy Name</th>
<th>Attached Entities</th>
<th>Creation Time</th>
<th>Edited Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>AmazonLexFullAccess</td>
<td>0</td>
<td>2017-04-11 18:20 CDT</td>
<td>2017-04-14 14:45 CDT</td>
</tr>
<tr>
<td>AmazonLexReadOnly</td>
<td>0</td>
<td>2017-04-11 18:13 CDT</td>
<td>2017-04-11 18:13 CDT</td>
</tr>
<tr>
<td>AmazonLexRunBotsOnly</td>
<td>0</td>
<td>2017-04-11 18:06 CDT</td>
<td>2017-04-11 18:06 CDT</td>
</tr>
</tbody>
</table>
The result

---

### Managed Policies

The following managed policies are attached to this role. You can attach up to 10 managed policies.

<table>
<thead>
<tr>
<th>Policy Name</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWSLambdaExecute</td>
<td>Show Policy</td>
</tr>
<tr>
<td>AmazonLexRunBotsOnly</td>
<td>Show Policy</td>
</tr>
</tbody>
</table>

### Inline Policies

This view shows all inline policies that are embedded in this role.

| Create Role Policy |
Setting Up Our Lex Bot
Let’s Get Started

Amazon Lex

Amazon Lex is a service for building conversational interfaces using voice and text. With Lex, the same deep learning engine that powers Alexa is now available to any developer, enabling you to bring sophisticated, natural language chatbots to your new and existing applications.

Get Started

Getting Started Guide
Creating Our First Bot

Create your Lex bot

Amazon Lex enables any developer to build conversational chatbots quickly and easily. With Amazon Lex, no deep learning expertise is necessary—you just specify the basic conversational flow directly from the console, and then Amazon Lex manages the dialogue and dynamically adjusts the response. To get started, you can choose one of the sample bots provided below or build a new custom bot from scratch.

CREATE YOUR OWN

- Custom bot

TRY A SAMPLE

- BookTrip
- OrderFlowers
- ScheduleAppointment

Bot name

DIYVABot
As we scroll down...

- **OrderFlowers**
  - **Intents**: A particular goal that the user wants to achieve
  - **Utterances**: Spoken or typed phrases that invoke your intent
  - **Slots**: Data the user must provide to fulfill the intent
  - **Prompts**: Questions that ask the user to input data
  - **Fulfillment**: The business logic required to fulfill the user’s intent

**Conversation**
- I’d like to order flowers.
- What kind?
- 1 dozen roses please.
- Where should we deliver?
- Home.
- Can I go ahead with your order of a dozen roses to your name?
- Yes.
- Thank you. Your order went through successfully.
Last, but not least

IAM role: AWSServiceRoleForLexBots
Automatically created on your behalf

Child-Directed? (Mandatory)
Is your use of Amazon Lex related to a website, program, or other application that is directed or targeted, in whole or in part, to children under age 13 and subject to the Children's Online Privacy Protection Act (COPPA)? Please check "Yes" or "No" as appropriate. If your answer is "Yes", you must obtain any required verifiable parental consent under COPPA.

- Yes
- No

[Submit button]
One reason I did screen captures
An invocation and a test
The slots

1. **FlowerType**
   - **Slot type**: FlowerTypes
   - **Prompt**: What type of flowers would you like?

2. **PickupDate**
   - **Slot type**: AMAZON.DATE
   - **Prompt**: What day do you want the flowers by?

3. **PickupTime**
   - **Slot type**: AMAZON.TIME
   - **Prompt**: At what time do you want the flowers by?

**Confirmation prompt**
- Confirm
  - Okay, your {FlowerType} will be ready for pickup by {PickupTime} on {PickupDate}.

- Cancel (if the user says "no")
  - Okay, I will not place your order.

**Fulfillment**
- AWS Lambda function
- Return parameters to client
A custom slot

[Diagram of a custom slot interface with options for intents, slot types, and error handling.]

- **Intents**: OrderFlowers
- **Slot types**: FlowerTypes
- **Value**:
  - e.g. Small
  - roses
  - lilies
  - tulips
Bot settings

DIYVABot

General
- Name: DIYVABot
- Description: Handling Various DIYVA Intents
- Language: English (US)
- Output voice: Kendra
- Session timeout: 10 min
Bot aliases

DIYVABot

Settings

General

Aliases

Alias name

Bot version

e.g., Prod

Version

Dev

Latest

Prod

1
Let’s add an intent
Create intent

Give a unique name for the new intent

KnockKnock
Building our intent

KnockKnock

Sample utterances

- e.g. I would like to book a flight.

Lambda initialization and validation

Slots

<table>
<thead>
<tr>
<th>Priority</th>
<th>Required</th>
<th>Name</th>
<th>Slot type</th>
<th>Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>e.g. Location</td>
<td>e.g. AMA...</td>
<td>e.g. What city?</td>
</tr>
</tbody>
</table>
Making a custom slot

Intents
- OrderFlowers
- KnockKnock

Slot types
- thereIsWho
- whoIsThere
- FlowerTypes

Error Handling

Value
- e.g. Small
- itch
- arch
- orange
- banana

Responses to "Who's there?"
Putting the slot type to work
The knock-knock intent

**Intents**
- OrderFlowers
- KnockKnock

**Slot types**
- thereIsWho
- whoIsThere
- FlowerTypes

**Error Handling**

**Slots**

<table>
<thead>
<tr>
<th>Priority</th>
<th>Required</th>
<th>Name</th>
<th>Slot type</th>
<th>Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>e.g. Location</td>
<td>e.g. AMA...</td>
<td>e.g. What city?</td>
</tr>
<tr>
<td>1.</td>
<td>✔️</td>
<td>whoIsThere</td>
<td>whoIsTh...</td>
<td>Who's there?</td>
</tr>
<tr>
<td>2.</td>
<td></td>
<td>thereIsWho</td>
<td>thereIsW...</td>
<td>{whoIsThere} who?</td>
</tr>
</tbody>
</table>

**Confirmation prompt**

- Confirmation prompt

Confirm:

Ha ha ha. That's very funny. Are you a professional comedian?

Cancel (if the user says "no")

You should be.
Three important functions

• Get credentials
• Get audio from user
• Send audio to Lex
Code to download: AWS SDK

https://sdk.amazonaws.com/builder/js/
Code to download: SmallRecorder

https://github.com/YiddishNinja/LittleTools/tree/master/smallrecorder
Make The Cordova Client
From WAY Back
Make The Cordova Client

```html
<h1 id="welcome">You are Not Logged In</h1>
<p>Please click the button below to open a browser window to take you to Login with Amazon. Once you have done so on their site, they’ll pass you back to us with a token we can use to provide you with services.</p>

<a href="#" id="lwa_button"><img src="images/lwa_button_312.png" style="height:64px; width:312px;margin:auto 0 auto 0"/></a>

<h1>Login with Amazon Complete</h1>
<h2>Getting Cognito Credentials from AWS</h2>

<script type="text/javascript" src="cordova.js"></script>
<script type="text/javascript" src="scripts/platformOverides.js"></script>
<script type="text/javascript" src="scripts/environment.js"></script>
```
Bringing it all together

• Getting an LwA access token
• Trading it to Cognito
• Sending user audio to Lex
Getting an LwA access token

1. //set up handling of the LWA login button
2.   var lwaButton = document.getElementById("lwa_button");
3.   lwaButton.addEventListener('click', function () {
4.       //open InAppBrowser to handle LWA web login
5.       var ref = cordova.InAppBrowser.open(diyvaENV.LWA_PROXY + "?state=" + 
        device.platform, '_blank');

6.       //add handling of return from LWA
7.       ref.addEventListener('loadstop', function (e) {
8.           //check loaded URL for completion, process login and pass hash
9.           if (e.url.includes(diyvaENV.LWA_PROXY_RETURN)) {
10.              logincomplete = true;
11.              ref.close();
12.              if (e.url.includes(diyvaENV.LWA_PROXY_RETURN)) {
13.                 handleReturnedHash(e.url);
14.               }
15.           }
16.       });
Getting an LwA access token – 2

```javascript
function handleReturnedHash(hashVal) {
    document.getElementById("welcome").style.display = "none";
    var fragment = hashVal.split("#"),
    if (fragment.length < 2) {
        return false;
    }
    var valarray = fragment[1].split('&'),
    var vals = [];
    valarray.forEach(function (valpair, index, array) {
        var subvals = valpair.split('=');
        vals[subvals[0]] = subvals[1];
    });
    if (!("access_token" in vals)) {
        return false;
    }
    window.localStorage.setItem("loggedIn", "true");
    window.localStorage.setItem("LWALoginTime", Math.floor(Date.now() / 1000).toString());
    window.localStorage.setItem("LWAToken", vals["access_token"]);
    window.localStorage.setItem("LWATTTL", vals["expires_in"]);
    window.location.href = "index.html";
}
```
function getCognitoCreds() {
    creds = new AWS.CognitoIdentityCredentials(
        IdentityPoolId: '[the identity pool ID you copied during Cognito setup]'
    );
    creds.params.Logins = {};
    creds.params.Logins['www.amazon.com'] = decodeURI(divind.vals["LWAToken"]);
    // Explicitly expire credentials so they are refreshed on the next request.
    creds.expired = true;
    AWS.config.update({
        region: 'us-east-1',
        credentials: creds
    });
    creds.get(function (err, data) {
        if (err) {
            console.log("we have an error", err); return false;
        }
        divind.CogCreds = creds;
        divind.lex = new AWS.LexRuntime({ apiVersion: '2016-11-28' });
    });
}
function endRec() {
  recorder && recorder.stop();

  recorder && recorder.exportWAV(function (blob) {
    var params = {
      botAlias: 'demoBot', //required
      botName: 'OrderFlowers', //required
      userId: 'nuttinSpecial', //required - could use LWA unique
      contentType: "audio/116; rate=16000; channels=1",
      inputStream: blob,
      accept: "audio/mpeg"
    };
    divind.lex.postContent(params, function (err, data) {
      if (err) console.log(err, err.stack); // an error occurred
      else processResponse(data); // successful response
    });
  });
}
function processResponse(data) {
    __log1('Processing Response');
    console.dir(data);

    var buf = btoa(String.fromCharCode.apply(null, data.audioStream));
    var asrc = "data:audio/mpeg;base64," + buf;
    var walter = document.getElementById('hmv');
    walter.src = asrc;
    walter.play();

    //object returned...
    //.audioStream - Uint8Array
    //.dialogState - "ElicitSlot"
    //.inputTranscript = "what I said"
    //.intentName
    //.message = "what she said"
    //.slotToElicit = 'whodis'
}
wrap it up

with a knock-knock joke
Login with Amazon
Puppet Theater

PRESENTS:
the trolling effectiveness of a broken knock-knock joke
AWS Chatbot Challenge

• Create a chatbot with Amazon Lex and AWS Lambda
• Prizes: $10,000 cash, $5,000 AWS credits, re:Invent tickets, more!
• Submissions due July 18

awschatbot2017.devpost.com
Tweets - @yiddishNinja

If there’s time you can ask questions and I can pretend I’m Sean Spicer.