はじめての xR ~ Amazon Sumerian ~ハンズオン

2019/08/23

アマゾン ウェブ サービス ジャパン
エバンジェリスト
亀田（hkameda@amzon.co.jp）

はじめてに：

本ハンズオンでは、VR/AR アプリケーションの構築を実現するサービスの Amazon Sumerian、チャットボットサービスである Amazon Lex を用いて、３D 化された Google Map への音声による検索アシスタントを作ります。2019 年 7 月時点で Amazon Lex は日本語化がされていないため、音声 I/F は英語になります。また、本ハンズオンではすべての作業を【バージニア北部】リージョンで行いますので、画面右上の表記に注意ください。

また、Sumerian はクラウドサービスですが、ローカル環境で js 等が多く実行されるため、可能な限り使わないタブやプログラムを停止して作業を行うことをお勧めします。

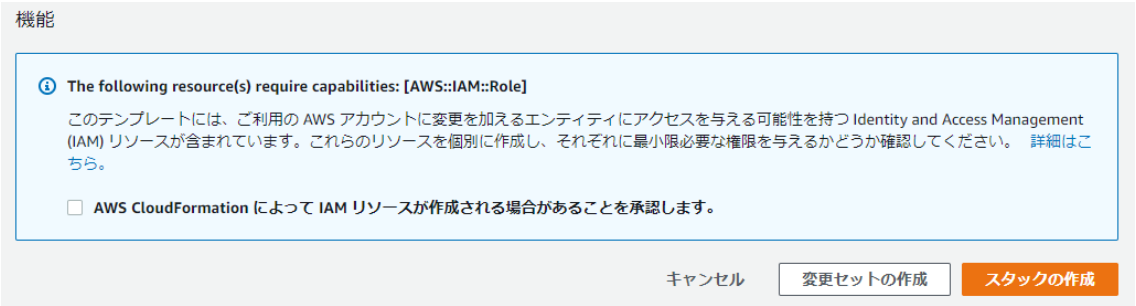
ブラウザは Chrome か Firefox を推奨しますが、作成した環境を起動させるのは、Firefox を推奨します。（Chrome だとデフォルトではマイクが認識されない環境があるため）

１．Amazon Cognito と AWS CloudFormation セットアップ

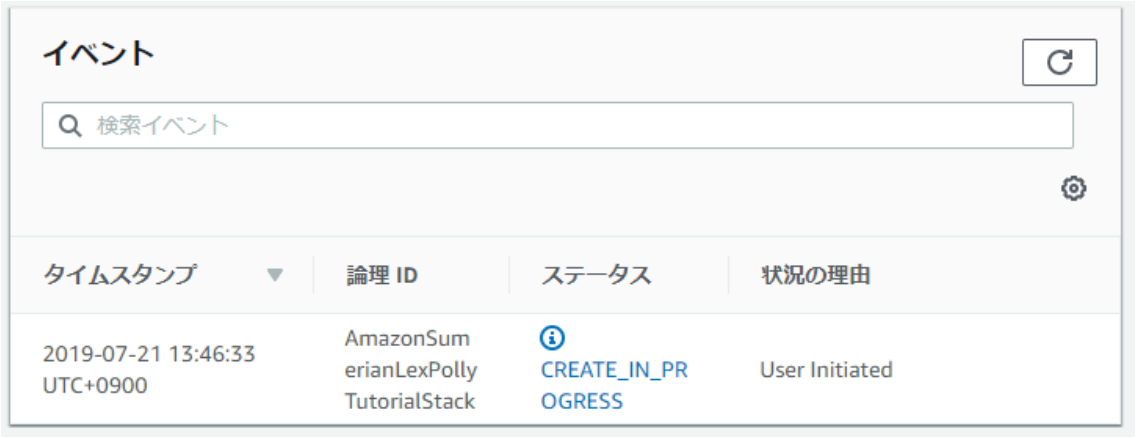
Sumerian は作成されるオブジェクトが全て管理され、オブジェクトから AWS リソースを呼び出す場合 AWS IAM で制御されるため、その管理を Cognito で行います。初期設定で Cognito の設定を CloudFormation で行います。

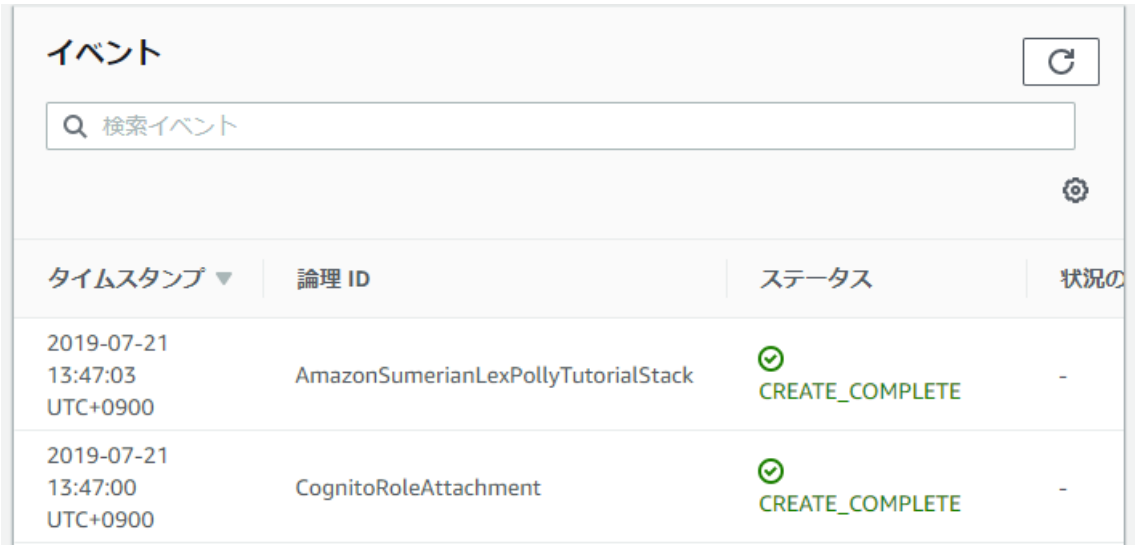
AWS アカウントへログインしたのち https://amzn.to/2JTx6Pc へアクセスをします。

２．【スタックの名前】に適当な名称（ご自身の UserID 等アルファベットを入れる）を付け、以下のチェックを付けて【スタックの作成】を押します。(ハイフンは使えませんのでご注意ください)

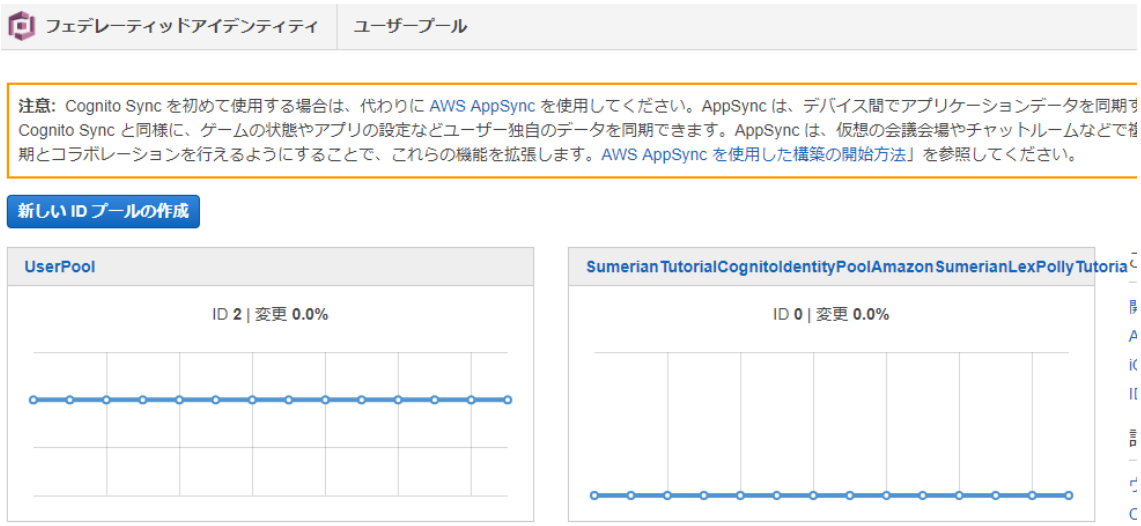


3．しばらくすると[CREATE_IN_PROGRESS]が[CREATE_COMPLETE]になります。





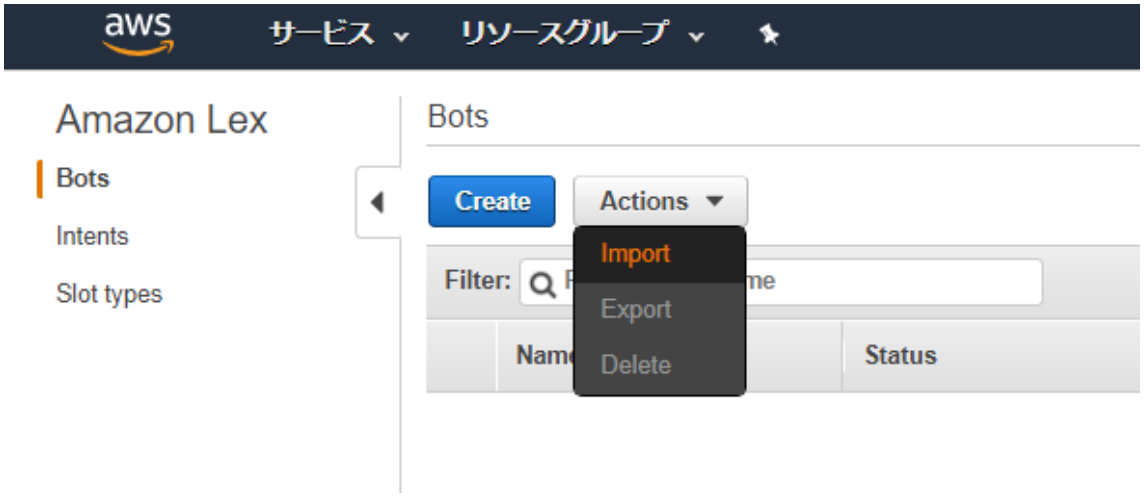
4．念のため Cognito の ID プールの管理画面で、正しく作成されているかを確認します。

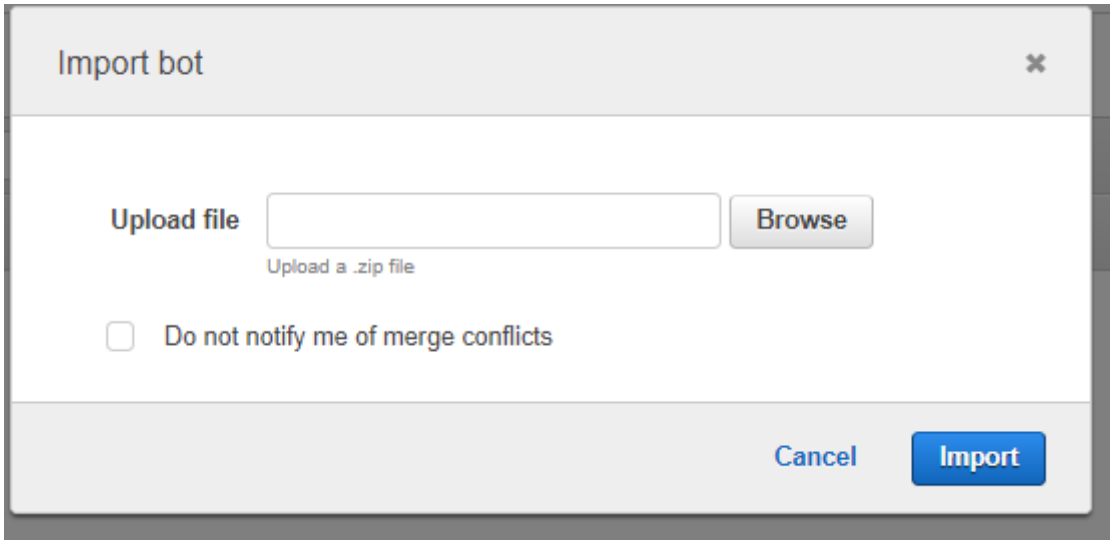


5．作成された ID プールをクリックし、画面左ペインの【サンプルコード】をクリックします。赤字の部分を Sumerian に連携用に設定する必要があるため、どこかにコピペしておきます。（“ は含みません）

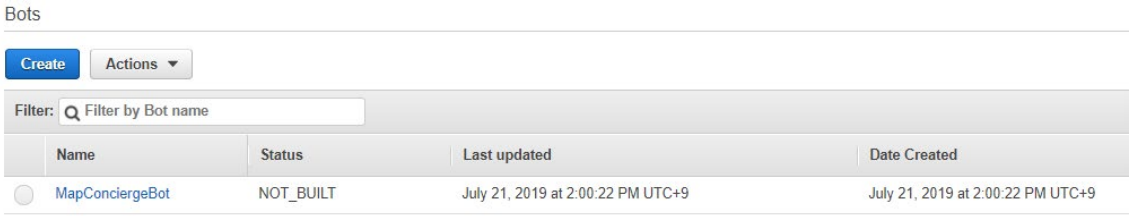
6．https://bit.ly/2M3Ssw2　から Lex 用の設定ファイルをダウンロードします。

7．Lex のマネージメントコンソールへ移動します。画面以下のように【Action】→【Import】から先ほどの zip ファイルを import します。（チュートリアル画面が出た場合、画面下部までスクロールして【cancel】を押します。

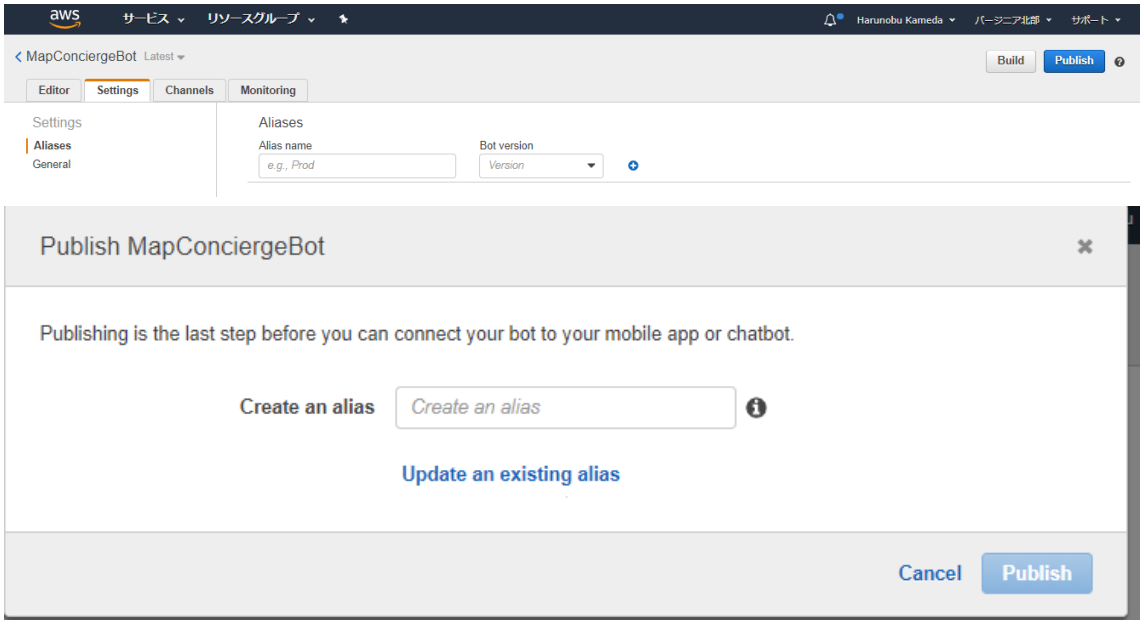




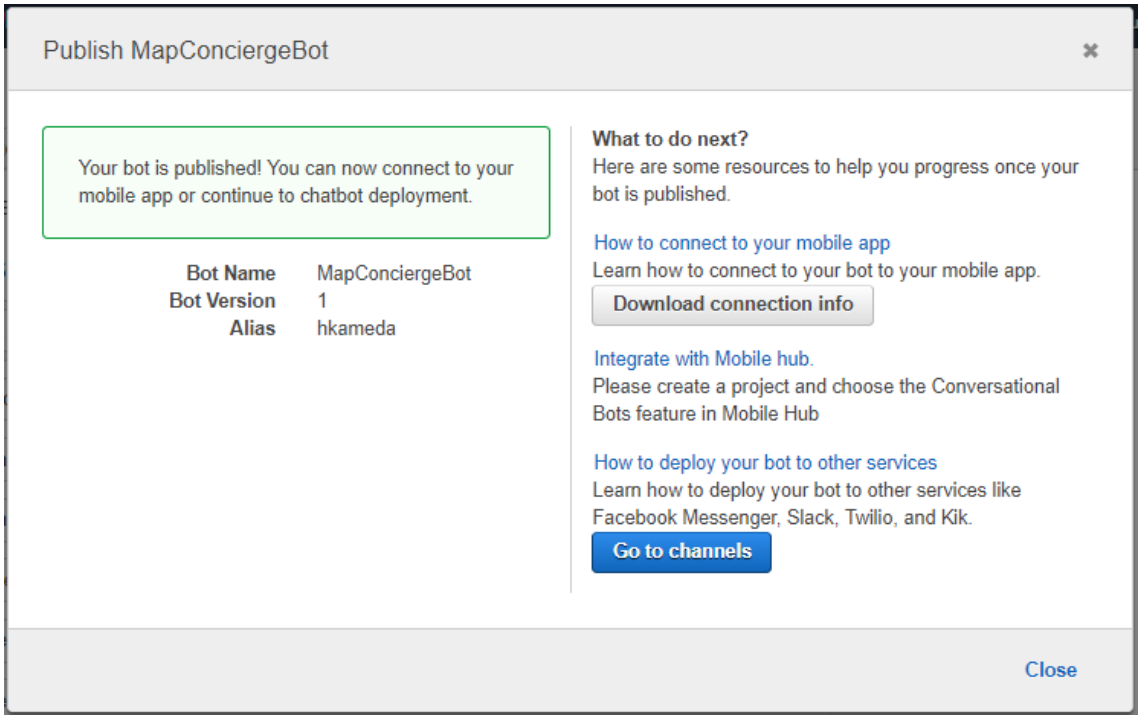
8．Import が完了したら以下の表示になります。



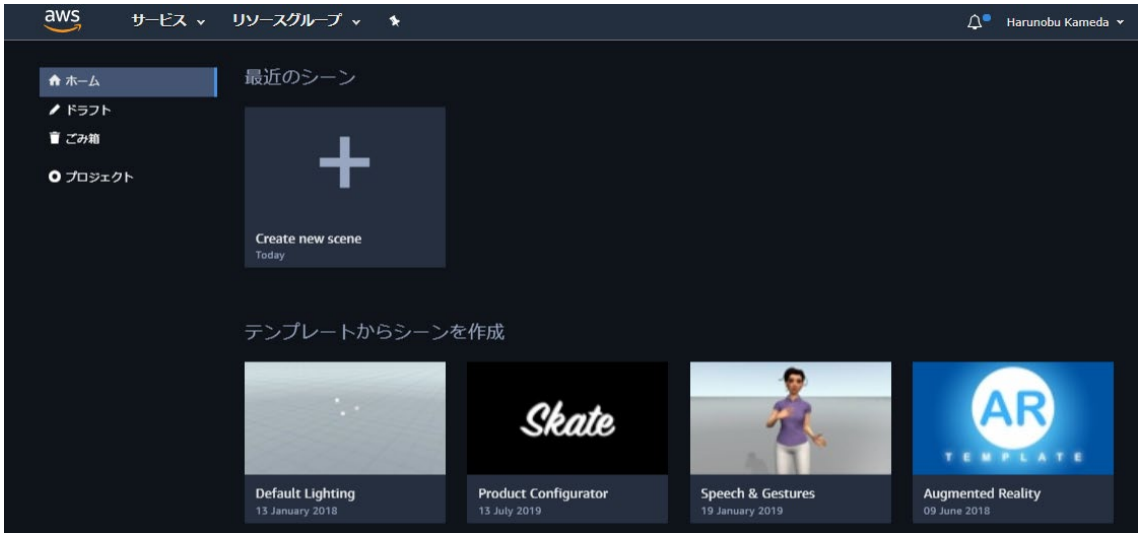
9．【MapConciergeBot】のリンクを開き、画面右上の【Build】を押し、「MapConciergeBot build was successful」が表示されるまで待ってから【Publish】を押します。【Publish】の際に【Alias】の入力が求められますので適当な名前を付けます。（数字が入らないため、アルファベットのみです）



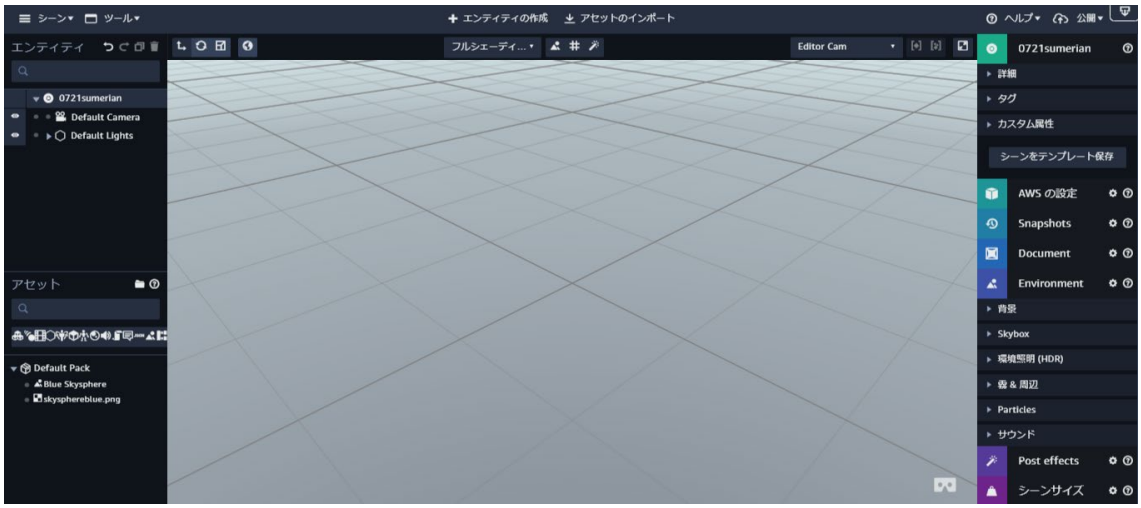
１０．以下の画面が表示されたら成功です。画面右下の【Close】を押して画面を閉じます。



１１．Sumerian のトップページにアクセスをします。



１２．【Create New Scene】を押して適当な名前を付け【作成】ボタンを押します。しばらくすると Sumerian のシーン編集の画面が起動します。



１３．この画面では専門用語が多く出てきますが、まず以下３つを覚えます。
【エンティティ】：画面左上です。ここで配置されたオブジェクトが真ん中のシーンの画面に出てきます。
【アセット】画面左下です。外部からオブジェクトやテクスチャーなどのインポートを行うとここに表示されます。ここで管理されているアセットを上のエンティティに登録することでシーンに利用ができるようになります。
【インスペクタパネル】画面右です。エンティティで登録されているオブジェクトの挙動を設定します。

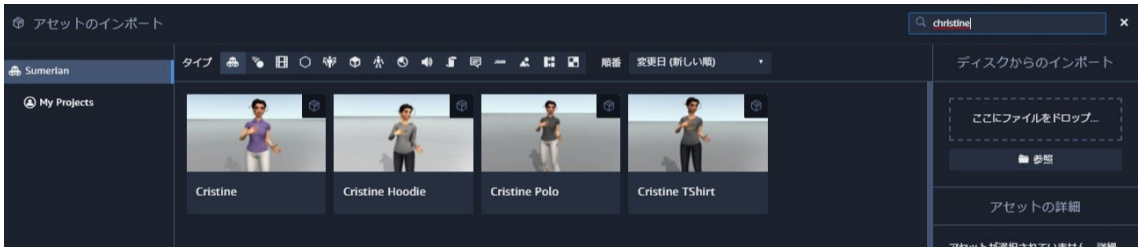
例えば、人間を配置して Lex と連携したボットを作る場合、【アセット】で人間を読み込んだのち、【エンティティ】に配置します。その後【エンティティ】でそのボットを選択し【インスペクタパネル】で Lex の設定などを行います。

この３つの用語はこれから繰り返し出てきますので、覚えておいてください。

１４．【エンティティ】から一番上を選択し、【インスペクタパネル】から【AWS の設定】を選び、先ほどメモした Cognito ID プールの値をコピペします。

１５．画面一番上の【アセットのインポート】を押します。

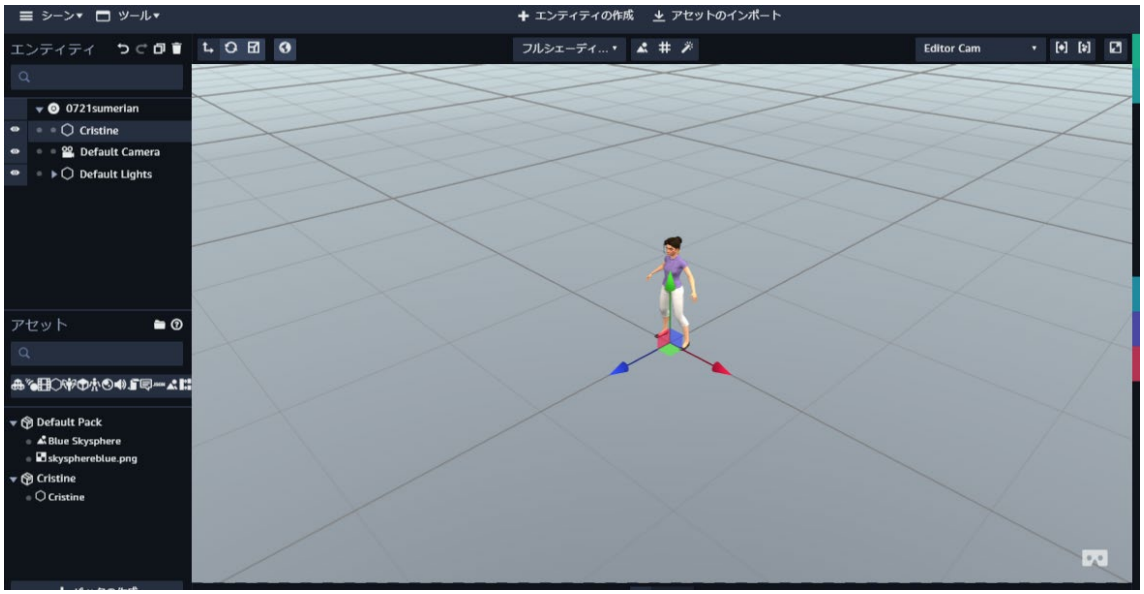
１６．画面右上に【Christine】と入力し、女性を選択し、画面右下の【追加】を押します。（もちろん人間でも家具でもなんでもこのハンズオンは動作しますが、とりあえず Christine で作業をしましょう！）



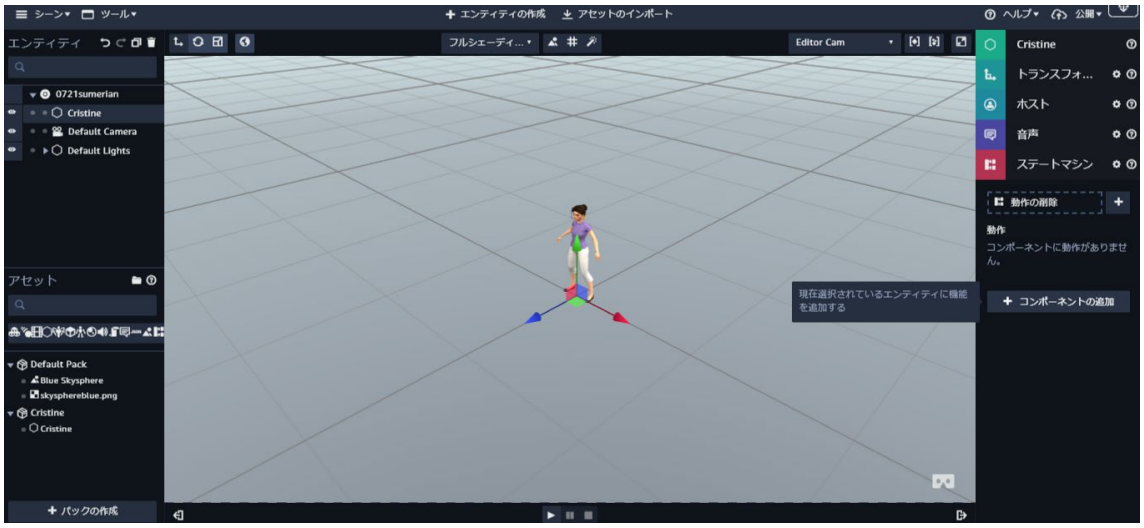
１７．【アセット】に Christine が登録されています。

１８。【アセット】から六角形の Christine を選び、真ん中の画面にドラッグ＆ドロップし

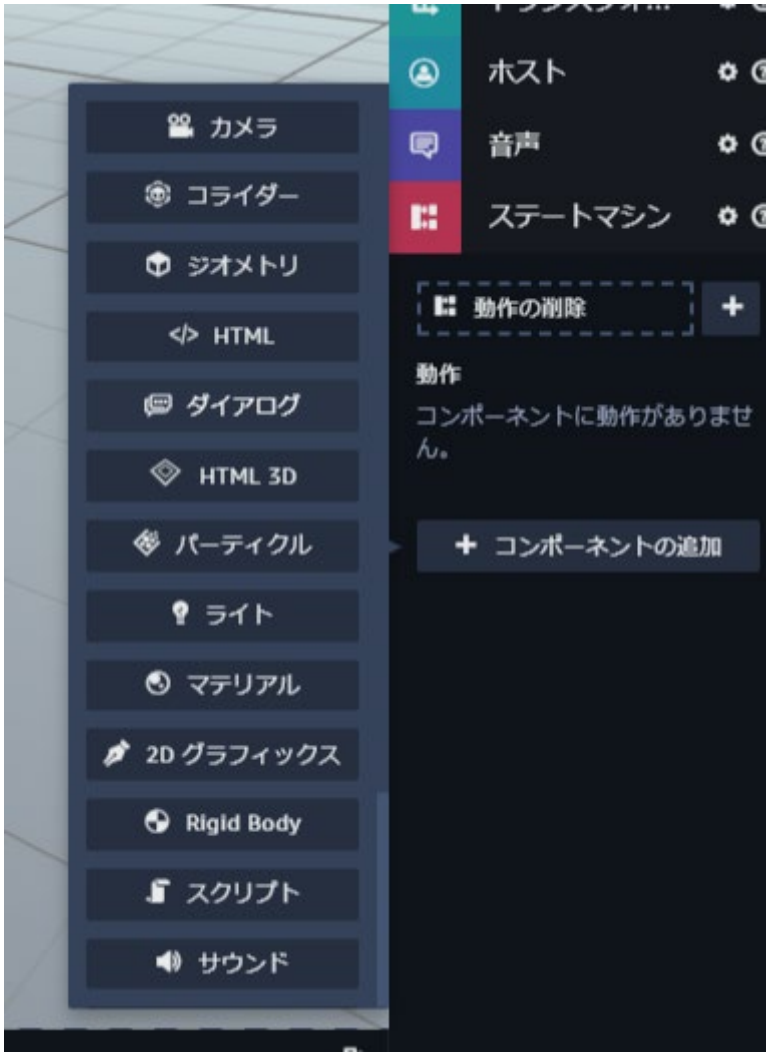
ます。同時に【エンティティ】にも Christine が登録されます。



１９.【エンティティ】で Christine を選び【インスペクタパネル】で【コンポーネントの追加】を押します。



２０.【ダイアログ】を選択します。



２１.【インスペクタパネル】にダイアログが出てきます。

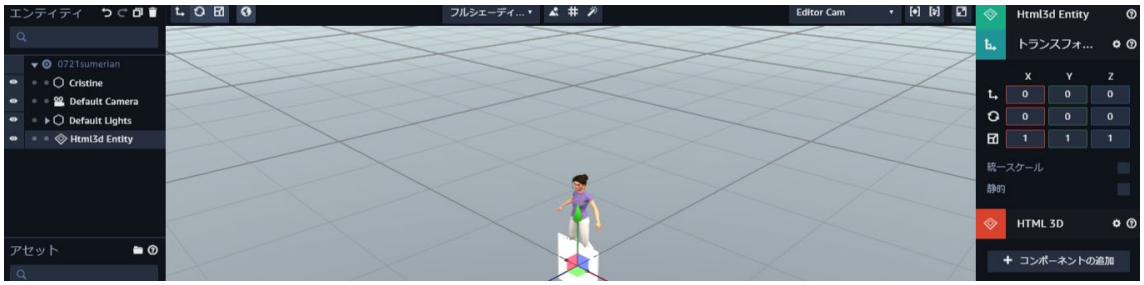
２２．【名前】に【MapConciergeBot】を入力します。【エイリアス】には【$LATEST】を入れます。これは Lex の関数名、そして関数の最新バージョンと連携させることを意味します。

２３．続いて Google Map を表示させる HTML3D キャンバスを作成します。画面上の【エンティティの作成】を押します。

２４．【HTML 3D】を選択して、【閉じる】を押します。

２５．白いキャンバスが画面に表示されます。

２６.【エンティティ】から【Html3d Entity】を選び【インスペクタパネル】から【トランスフォーム】を開きます。



２７．以下の値を入力します。

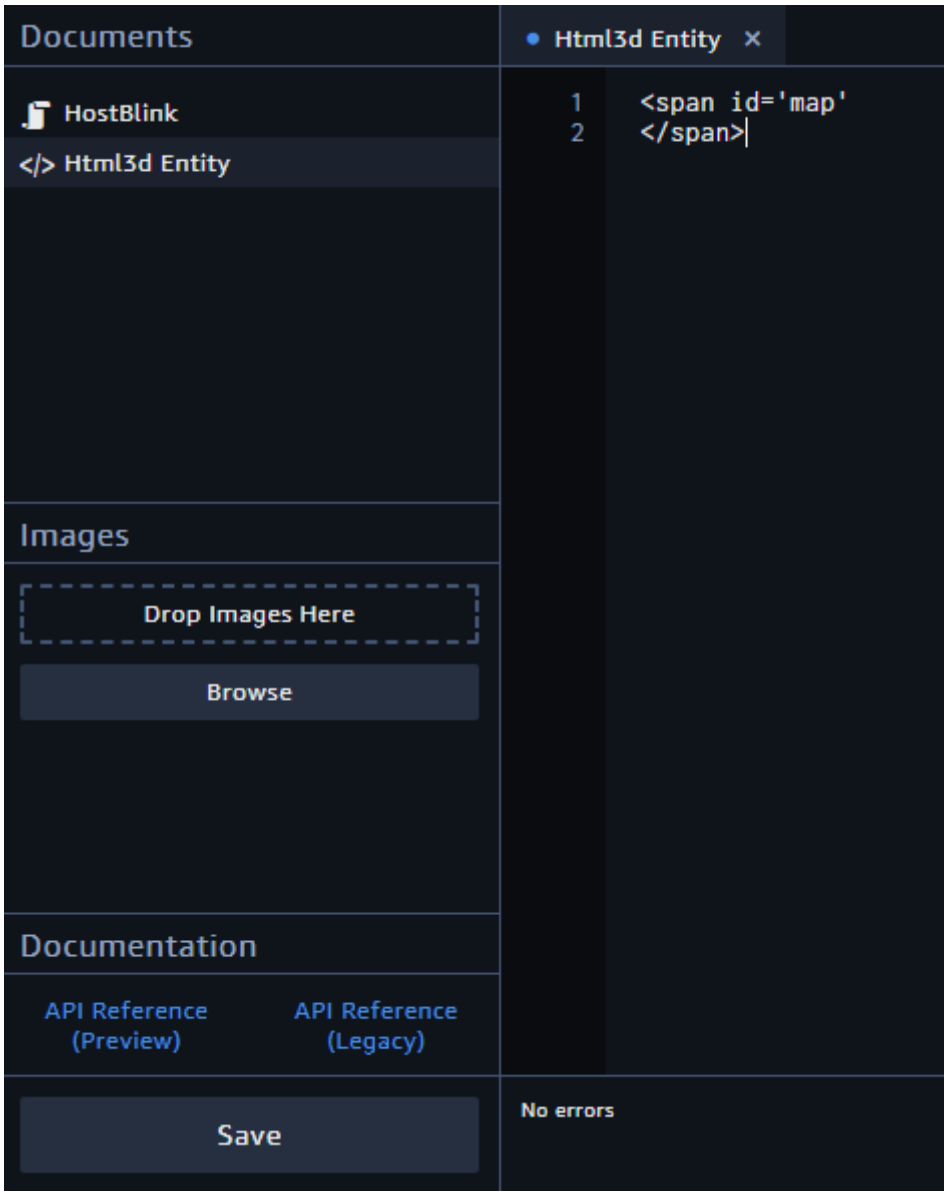
２８．画面左上【ツール】から【テキストエディタ】を選びます。

２９．【Html3d Entity】を選び。画面右のコードをすべて消します。
消したのち以下を入力します。

```
<span id='map'
 </span>
```

（注意：PDF からのコピペはおおよそ正しく動作しないため、手入力か、一度テキストエディタにペーストした後、再度コピペしてください）

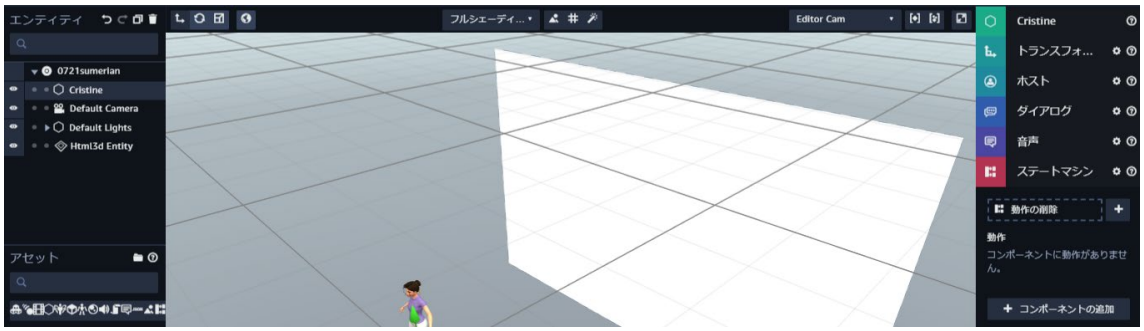
３０．【Save】を押して保存します。



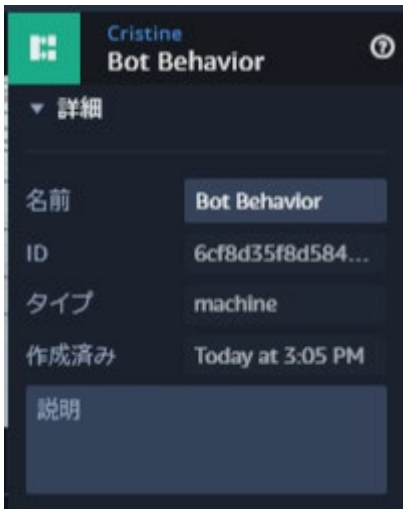
３１．表示が【Save】から【Saved】に変わったら保存完了ですので、ウインドウを閉じます。

３２．[a]キーで Christine が反応を開始するような一連の処理を作成します。

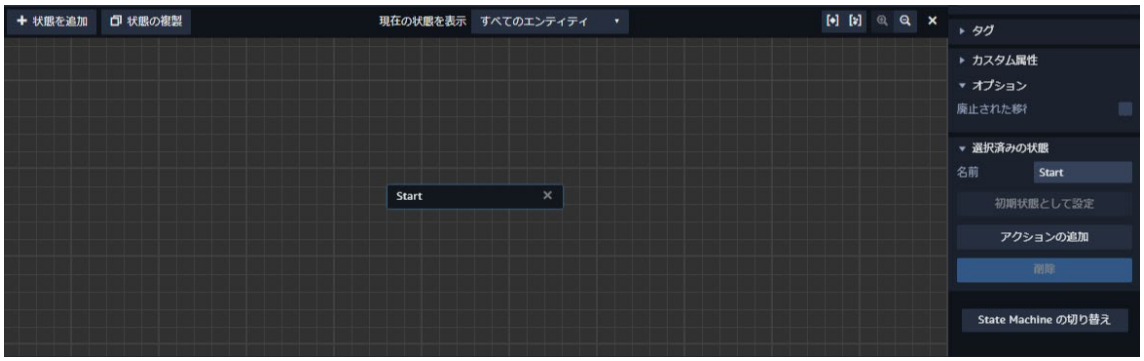
【エンティティ】から Christine を選んで、【インスペクタパネル】から【ステートマシン】を選び【+】を押します。



３３.【詳細】を選び名前を【Bot Behavior】に変更します。



３４.【State１】を選び名前を【Start】に変更します。その後【アクションの追加】を押します。

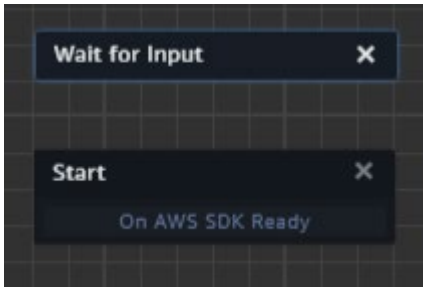


３５.【アクションの追加】ウインドウで【AWS SDK Ready】を選んで【追加】を押します。

３６．以下の表な表示に代わります。これにより Christine は初期状態で AWS SDK の呼び出しが出来るようになります。

３７．【状態を追加】を押し、【Wait for Input】に名前を変えます。

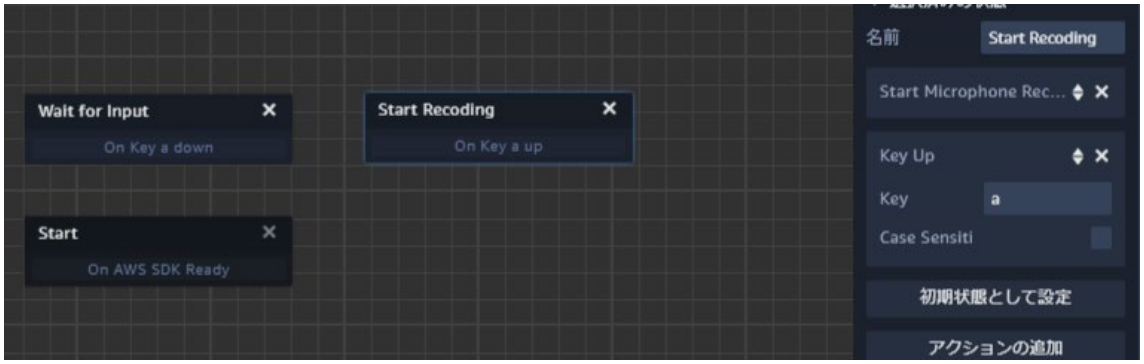
【Start】の上に重なった場合は少し離して見やすくします。



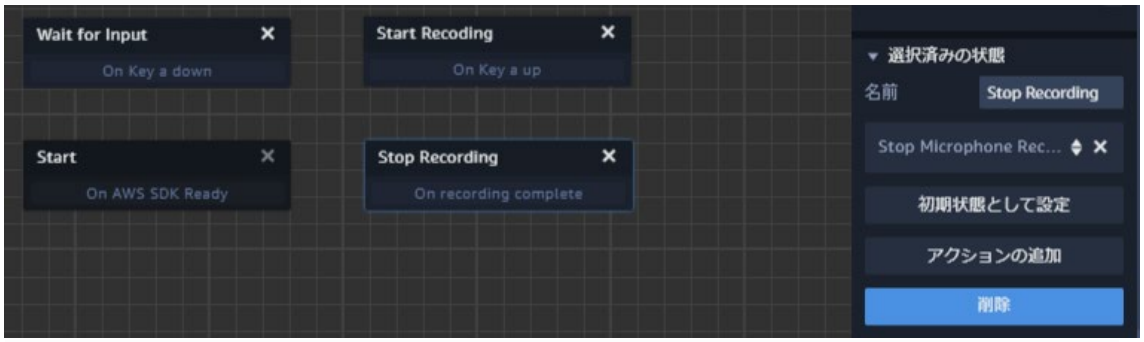
３８．アクションの追加から、Key Down を選びます。

３９．【Wait for Input】を選び右側の【選択済の状態】で【Key】のところに[a]と入力します。（大文字の A を小文字の a に変更します）これで[a]を押すと、Christine が一連の処理を開始します。

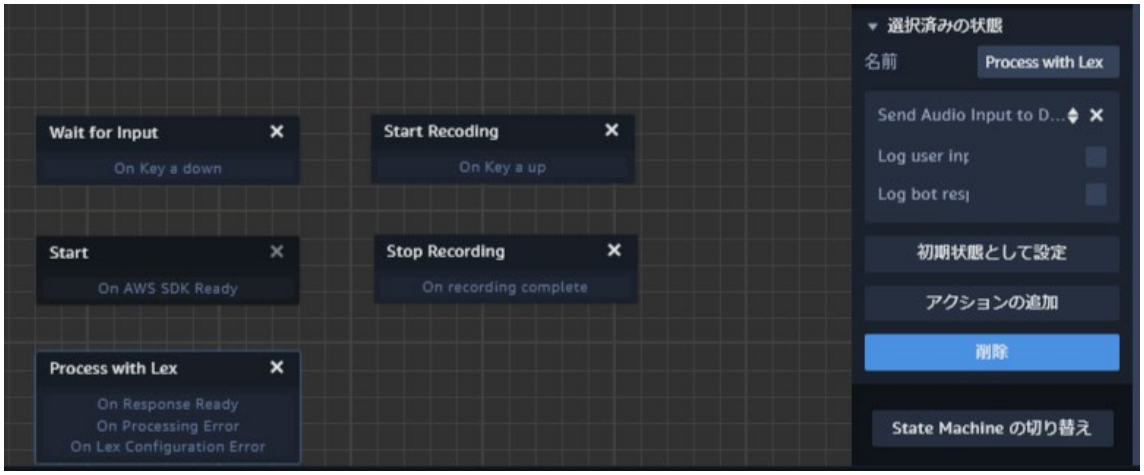
４０．同じ要領で今度は【Start Recording】というステートを作成し、【Start MicrophoneRecording】と【Key Up】を２つ追加し作成します。



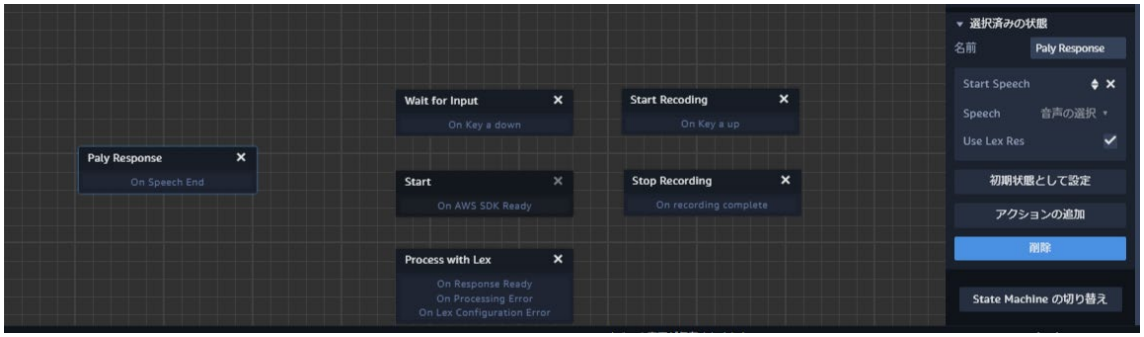
４１．同じ要領で【Stop Recording】を作成します。今度は【Stop Microphone Recording】のみです。



４２．次に【Process with Lex】を作ります。【Send Audio Input to Dialogue Bot】アクションを選びます。



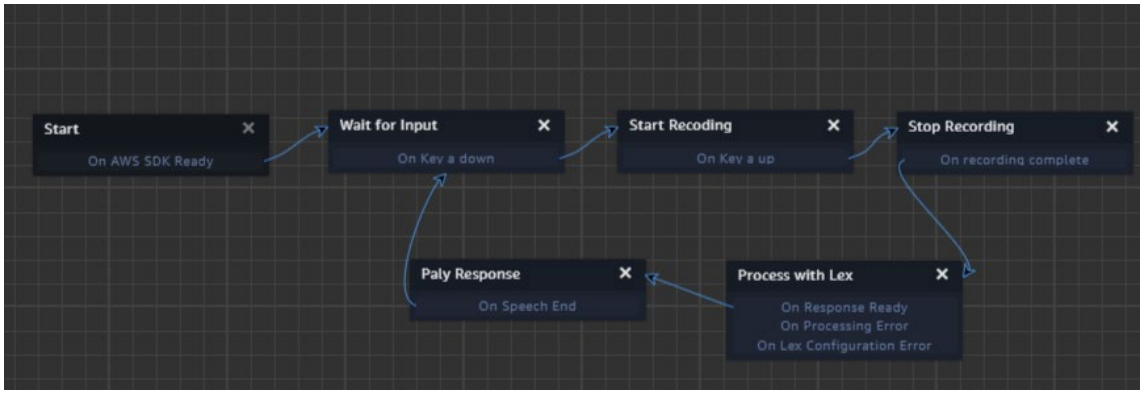
４３．次に【Play Response】を作ります。【Start Speech】アクションを追加します。【Use Lex Res】に必ずチェックを入れます。



４４．各 State の連携をドラッグアンドドロップ作成します。以下のように並び替え相関を作成してください。

（必ずしも以下の通りでなくてもいいですが、作業が詰まってチューターを呼ぶ際は以下の形にしてもらえるとわかりやすいです）

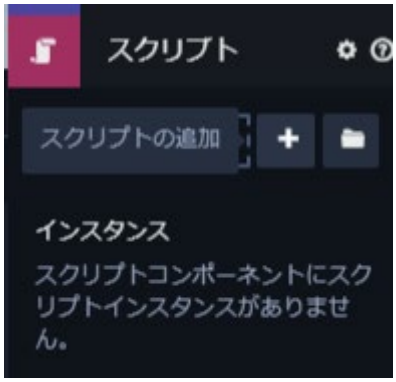
（Process with Lex の On Response Ready から正しく矢印が出ていることを確認してください。動作しない場合ほとんどのケースが、On Processing Error から矢印が出ています）



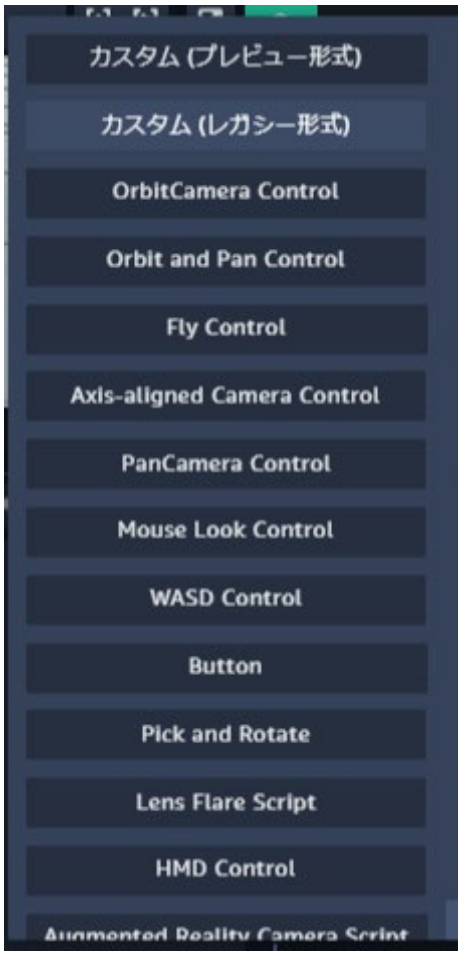
４５．【Start】を選択し、画面右の【初期状態として設定】を押します。すでに初期状態となっている場合、設定がなされていることのみの確認で OK です。

４６．これから Google Map を呼び出すスクリプトを作ります。【エンティティ】から Christine を選んで【インスペクタパネル】からコンポーネントの追加で【スクリプト】を選びます。

４７.【+】ボタンでスクリプトを開きます。

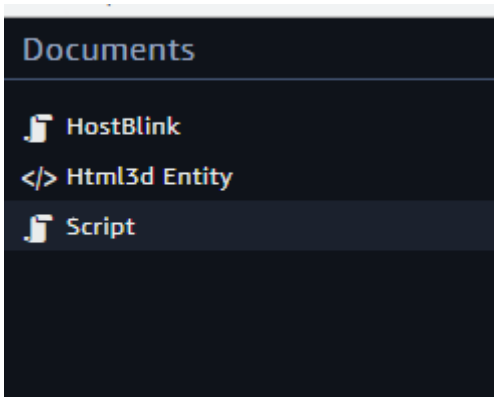


４８.【カスタム（レガシー形式)】を選びます。



４９．エンピツマークを選んで Scrip の編集画面に行きます。

５０．ここまでの作業が正しければ、先ほど作成した【Html3d Entity】も併せて表示されます。



５１．【Scrip】を選び、setup のところに以下をコピペします。

```
function setup(args, ctx) {
        ctx.onLexResponse = (data) => {
                if (data.dialogState === "Fulfilled") {
                        sumerian.SystemBus.emit(searchOnMap,
data.slots.localBusiness);
                        console.log(data.dialogState);
                }
                console.dir(data);
        }

sumerian.SystemBus.addListener(`${sumerian.SystemBusMessage.LEX_RESPONSE}.${c
tx.entity.id}`, ctx.onLexResponse);
 }
```

（注意：PDF からのコピペはおおよそ正しく動作しないため、手入力か、一度テキストエディタにペーストした後、再度コピペしてください）
同じように cleanup のところに以下をコピペします。

```
function cleanup(args, ctx) {
        sumerian.SystemBus.removeListener( `${sumerian.SystemBusMessage.LEX_RE
SPONSE}.${ctx.entity.id}`, ctx.onLexResponse);
}
```

５２．Script の名前を【LexResponse】に変更し、Save をします。Saved と表示されたら保存完了です。それぞれ以下のような画面になっていれば正しく入力されています。

```
'use strict';

// The sumerian object can be used to access Sumerian engine
// types.
//
/* global sumerian */

// Called when play mode starts.
//
 function setup(args, ctx) {
    ctx.onLexResponse = (data) => {
        if (data.dialogState === "Fulfilled") {
            sumerian.SystemBus.emit('searchOnMap', data.slots.localBusiness);
            console.log(data.dialogState);
        }
        console.dir(data);
    }
 sumerian.SystemBus.addListener(`${sumerian.SystemBusMessage.LEX_RESPONSE}.${ctx.entity.id}`
, ctx.onLexResponse);
 }

```

```
function cleanup(args, ctx) {
    sumerian.SystemBus.removeListener(
`${sumerian.SystemBusMessage.LEX_RESPONSE}.${ctx.entity.id}`, ctx.onLexResponse);
}

// Defines script parameters.
//
var parameters = [];

```

（左の行数などはブランク行などでずれますので、必ず同じである必要はありません）

５３．先ほど作成した Html3d Entity に Google Map との連携コードを入力します。

同じように Html3d Entity を【エンティティ】から選んで【インスペクタパネル】から【Scrip】を追加し、エンピツボタンで編集画面にいきます。

５４．【Script】の名前を【updateMap】に変更します。

Setup 以下をコピペします。

function setup(args, ctx) {

const mapIframeCode = '<iframe src="https://www.google.com/maps/embed?pb=!1m16!1m12!1m3!1d3022.617540796677!2d-73.98785308479034!3d40.74844047932796!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!2m1!1sSEARCH_TERM!5e0!3m2!1sen!2sus!4v1550399525495" width="600" height="450" frameborder="0" style="border:0" allowfullscreen></iframe>';

const mapElement = document.getElementById('map');

ctx.onSearchOnMap = (searchTerm) => {

const mapInnerHtml = mapIframeCode.replace('SEARCH_TERM', searchTerm);

```
            mapElement.innerHTML = mapInnerHtml;

            console.log(`Searching for ${searchTerm}`);
        };

        sumerian.SystemBus.addListener('searchOnMap', ctx.onSearchOnMap);

        // update map on opening to no search term
        const mapInnerHtml = mapIframeCode.replace('SEARCH_TERM', '');
        mapElement.innerHTML = mapInnerHtml;
 }
```

同じように cleanup に以下をコピペします。

```
sumerian.SystemBus.removeListener('searchOnMap', ctx.onSearchOnMap);
```

正しく入力されている場合、以下のようになります。

```
'use strict';

// The sumerian object can be used to access Sumerian engine
// types.
//
/* global sumerian */

// Called when play mode starts.
//
 function setup(args, ctx) {
    const mapIframeCode = '<iframe src="https://www.google.com/maps/embed?pb=!1m16!1m12!1m3!1d3022.617540796677!2d-73.98785308479034!3d40.74844047932796!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!2m1!1sSEARCH_TERM!5e0!3m2!1sen!2sus!4v1550399525495" width="600" height="450" frameborder="0" style="border:0" allowfullscreen></iframe>';

    const mapElement = document.getElementById('map');

    ctx.onSearchOnMap = (searchTerm) => {
        const mapInnerHtml = mapIframeCode.replace('SEARCH_TERM', searchTerm);
        mapElement.innerHTML = mapInnerHtml;

        console.log(`Searching for ${searchTerm}`);
    };

    sumerian.SystemBus.addListener('searchOnMap', ctx.onSearchOnMap);

    // update map on opening to no search term
    const mapInnerHtml = mapIframeCode.replace('SEARCH_TERM', '');
    mapElement.innerHTML = mapInnerHtml;
 }

// Called on every physics update, after setup(). When used in a
```

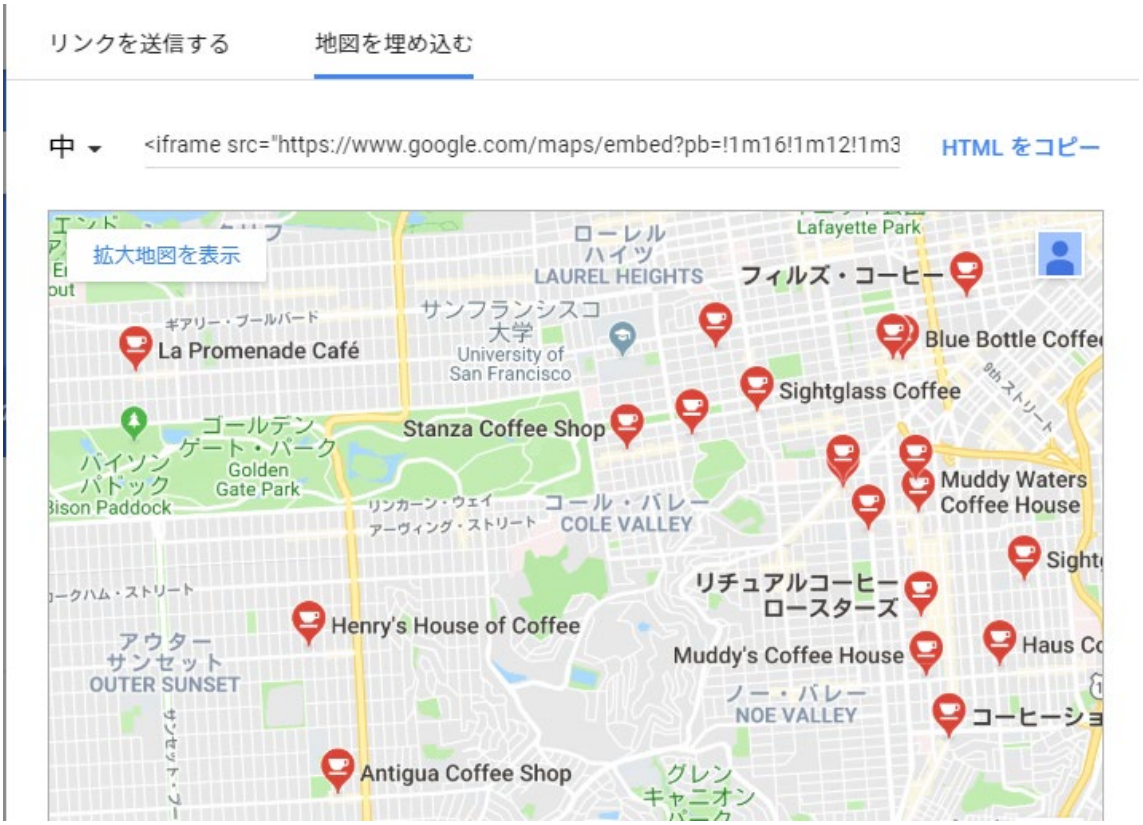
```
function cleanup(args, ctx) {
     sumerian.SystemBus.removeListener('searchOnMap', ctx.onSearchOnMap);
}

// Defines script parameters.
//
var parameters = [];
```

同様に行数はずれていても問題ありません。
５５．ブラウザの別のタブで Google Map を開き、San Francisco と検索します。（このデモは Google Map のどの検索結果でももちろん動作しますが、詰まる可能性のある方は手順をあわせましょう！）
５６．右下の【＋】ボタンで画面を２つぐらいズームします。その後[coffee shop]と入力します。
５７．左上のメニューを押します。

５８．【地図を共有または埋め込む】を選びます。

５９．【地図を埋め込む】を選び【HTML をコピー】を押します。



６０．コピーされた値をテキストエディタにペーストし、【coffee+shop】の値を【SEARCH_TERM】に変更します。これは Lex から渡ってきた値を表示させる、HTML タグに coffee shop の検索結果を置換するためです。

例：

<iframe src="https://www.google.com/maps/embed?pb=!1m16!1m12!1m3!1d25234.978541712953!2d-122.45511078663627!3d37.75786731313429!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!2m1!1sSEARCH_TERM!5e0!3m2!1sja!2sjp!4v1563693146066!5m2!1sja!2sjp" width="600" height="450" frameborder="0" style="border:0" allowfullscreen></iframe>

６１．置き換えが完了したら、すべての文字列を以下のように Script 編集画面の、【updateMap】の function setup の<iframe> ~~~~ </iframe>の部分を置き換えます。以下のようになります。

```
function setup(args, ctx) {
    const mapIframeCode = '<iframe src="https://www.google.com/maps/embed?
pb=!1m16!1m12!1m3!1d25234.978541712953!2d-
122.45511078663627!3d37.75786731313429!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!2m1!1sSEARCH_
TERM!5e0!3m2!1sja!2sjp!4v1563693146066!5m2!1sja!2sjp" width="600" height="450"
frameborder="0" style="border:0" allowfullscreen></iframe>';

    const mapElement = document.getElementById('map');

    ctx.onSearchOnMap = (searchTerm) => {
        const mapInnerHtml = mapIframeCode.replace('SEARCH_TERM', searchTerm);
        mapElement.innerHTML = mapInnerHtml;

        console.log(`Searching for ${searchTerm}`);
    };

    sumerian.SystemBus.addListener('searchOnMap', ctx.onSearchOnMap);
```

値は人によって変わりますが、各行の色がずれている場合は、作業ミスが発生していますので、再度手順を見直してください。

６２．【Save】をして【Saved】になったらウインドウを閉じてください。

６３．以上で手順は終了です。お疲れ様でした。作成したものを公開します。

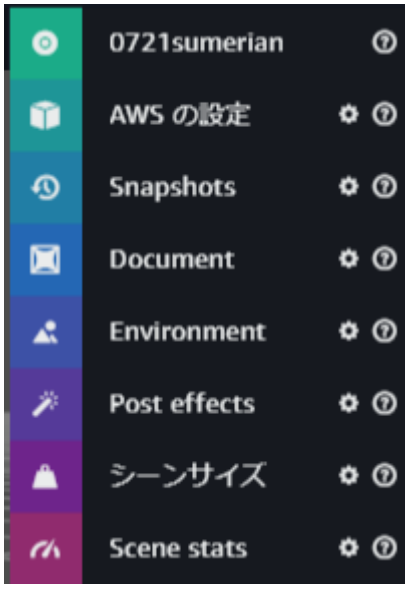
画面右上の【公開】ボタンを押しシーンを公開します。そこで生成された URL にアクセスします。

（Firefox を推奨します）

遊び方：キーボードの[a]を押して[Hello]を話しかけ、[a]を離します。

[What are you looking for ?]を聞いてくれれば成功です。検索したいものを英語で言ってください。話すときは[a]を押しっぱなしにすることを忘れないでください。英語が苦手な方は以下の単語の入力が通りやすいです。
「coffee shop」「Japanese food」「subway」
逆に「McDonald's」「Hotel」あたりは難易度が高めなので挑戦してみましょう！

時間に余裕のある方は、背景を色々カスタマイズしてみてください。
【背景のカスタマイズ方法】
１．インスペクターパネルの【Environment】を選択します



２．【雪】を押し、再実行すると雪が降ります。

３．【Skybox】を押し、シェープを【Sphere】にします。

５．適当な画像を入れて、再度【公開】してアクセスします。

Sphere は、空間全てを包むこむように登録した画像が、球体上に張り付けられます。

家のような空間を作りたい場合は、【Box】を選ぶとそれぞれの壁面毎のテクスチャを貼り

付けることができます。

おつかれさまでした。