Building Alexa Skills

with the JAVA AlexaSkillsKit SDK

Simple to Set Up & Use

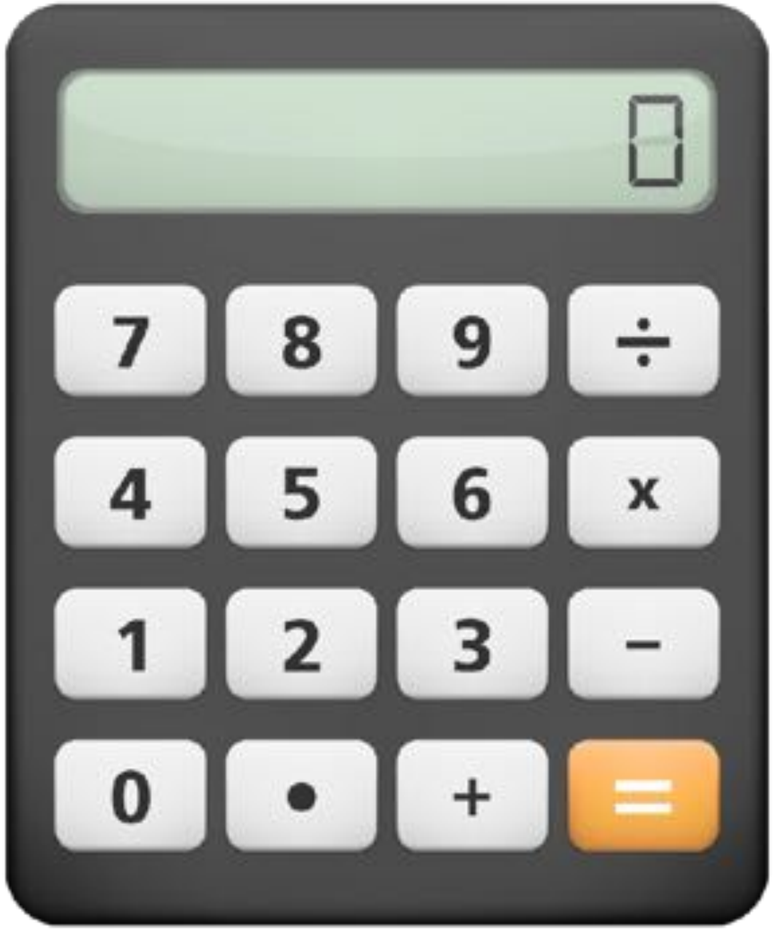
1. Plug in Echo Dot

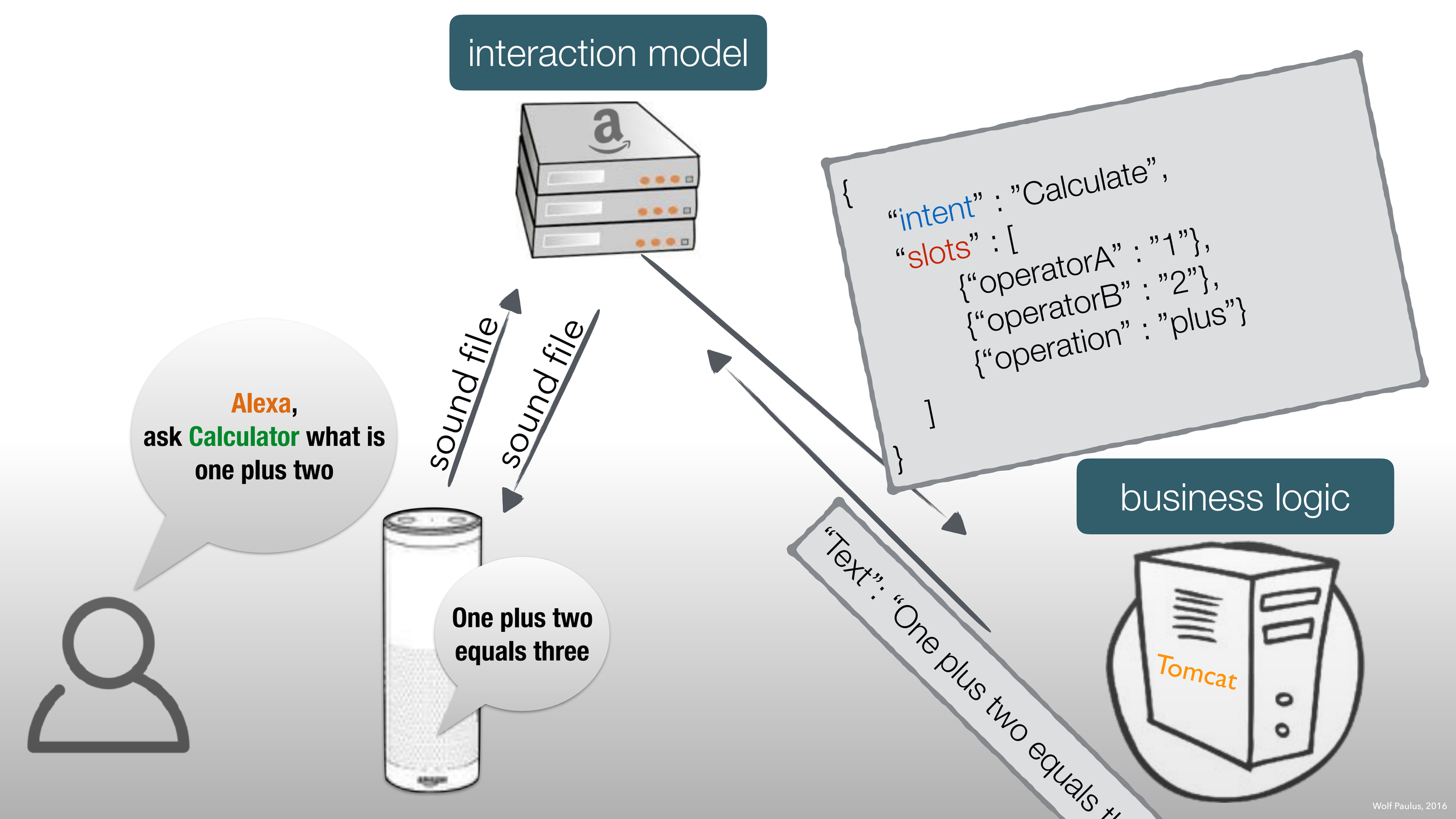
2. Connect to the internet with the Alexa App

3. Just ask for music, weather, news, and more

Alexa App is available for Android, iOS, and Fire devices.

A Very Simple Calculator Skill

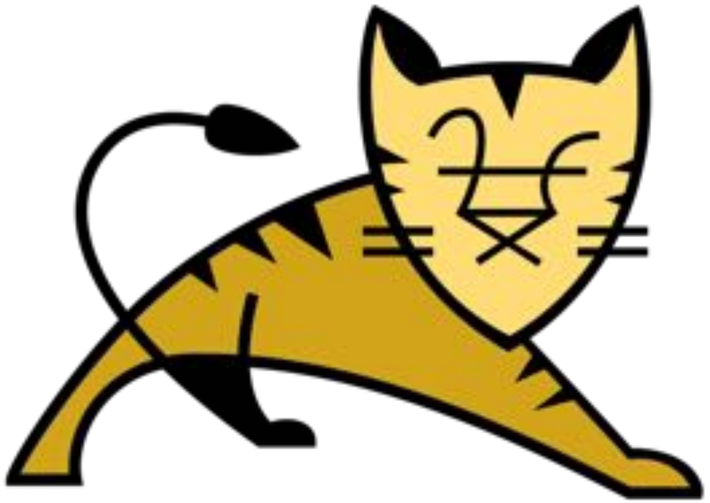




Wolf Paulus, 2016

Prerequisite

Java Web Container (e.g. Tomcat)
SSL Certificate



Putting the Certificate into a Java Keystore

- Assuming you already have a private key and public certificate in PEM format

- Export the private key and certificate to PKCS12

```
openssl pkcs12 -keypbe PBE-SHA1-3DES -certpbe PBE-SHA1-3DES
-inkey private-key.pem -in certificate.pem -export -out
marcie_cert.p12 -name tomcat
```

- Import the PKCS12 certificate and key bundle into a Java keystore

```
keytool -importkeystore -deststorepass ****** -destkeypass
****** -destkeystore tomcat.jks -srckeystore marcie_cert.p12
-srcstoretype PKCS12 -srcstorepass ****** -alias tomcat
```

Configure Tomcat to use HTTPS / SSL

/usr/share/tomcat/conf/server.xml

<Connector port="**443**"

protocol="org.apache.coyote.http11.Http11NioProtocol"

maxThreads="150"

SSLEnabled="true"

scheme="https"

secure="true"

clientAuth="false"

sslProtocol="TLS"

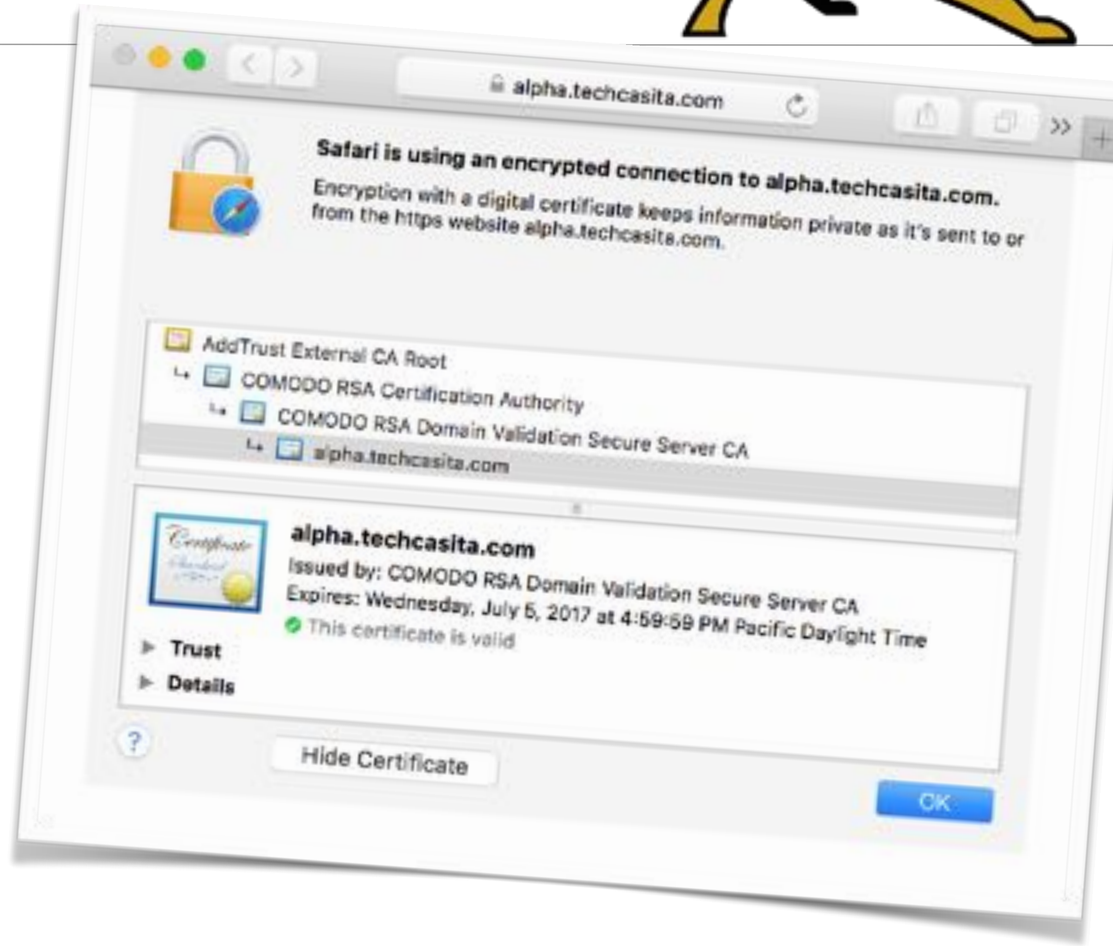
keyAlias="tomcat"

keystoreFile="/home/tomcat/ssl/tomcat.jks"

keystorePass="******"

sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"

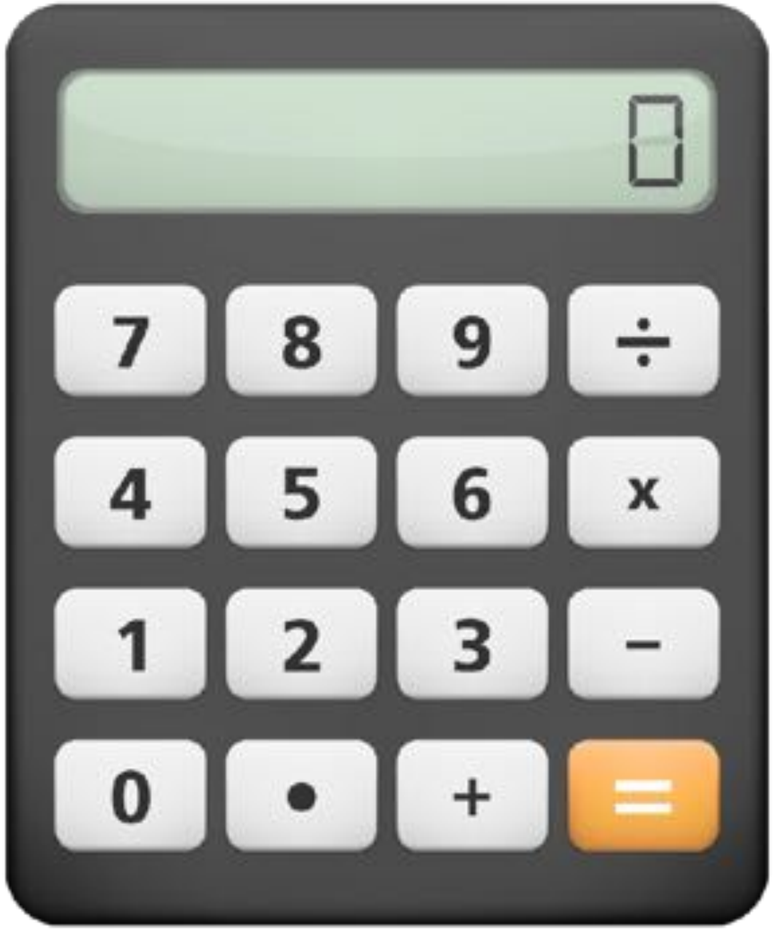
ciphers= "TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,TLS_ECDHE_RSA_WITH_RC4_128_SHA,
TLS_RSA_WITH_AES_128_CBC_SHA256,TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256_CBC_SHA256,
TLS_RSA_WITH_AES_256_CBC_SHA,SSL_RSA_WITH_RC4_128_SHA"/>



Hosting an Alexa Skill yourself

https://wolfpaulus.com/journal/java-journal/alexa-ssl/

Simple Calculator Skill



Sample Utterances

- What is two times three
- Tell me what is five divided by two
- Please calculate eleven minus five
- Compute three plus five

Intent: **Calculate**

- What is {operatorA} times {operatorB}
- Tell me what is {operatorA} divided by {operatorB}
- Please calculate {operatorA} minus {operatorB}
- Compute {operatorA} plus {operatorB}

operatorA and operatorB are Numbers

Intent: **Calculate**

- What is {operatorA} {operation} {operatorB}
- Tell me what is {operatorA} {operation} {operatorB}
- Please calculate {operatorA} {operation} {operatorB}
- Compute {operatorA} {operation} {operatorB}

operatorA and operatorB are Numbers

operation is {
- plus
- minus
- times
- multiplied by
- divided by

}

Interaction Model

```
{
  "intents": [
    {
      "intent": "Calculate",
      "slots": [
        {
          "name": "operatorA",
          "type": "AMAZON.NUMBER"
        },
        {
          "name": "operatorB",
          "type": "AMAZON.NUMBER"
        },
        {
          "name": "operation",
          "type": "OPERATION"
        }
      ]
    }
  ]
}
```

Custom Slot Type

OPERATION

- plus
- minus
- times
- multiplied by
- divided by

Sample Utterances

Built-in Slot Types	Sample Utterances	Value
AMAZON.LITERAL (deprecates on 11.30.2016)	“Hello World”	“Hello World”
AMAZON.NUMBER	“five”	“5”
AMAZON.TIME	“four in the morning”, “four a m”	“04:00”
AMAZON.DATE	“today”, “tomorrow”, or “july”	“2016-07-00T9”
AMAZON.DURATION	“five minutes”	“PT5M”
AMAZON.FOUR_DIGIT_NUMBER Provides recognition for four-digit numbers, such as years	“one two three four”	“1234”
AMAZON.US_FIRST_NAME (Popular first names, based on census and social security data)	“Joe”	“Joe”
AMAZON.US_STATE	“California”	“California”
AMAZON.US_CITY (Cities w/ a population over 100K are incl.)	“San Diego”	“San Diego”

Built-in Intents	Common Utterances
AMAZON.CancelIntent	cancel / never mind / forget it
AMAZON.HelpIntent	help / help me / can you help me
AMAZON.NextIntent	next / skip / skip forward
AMAZON.NoIntent	no / no thanks
AMAZON.PauseIntent	pause / pause that
AMAZON.PreviousIntent	go back / skip back / back up
AMAZON.RepeatIntent	repeat / say that again / repeat that
AMAZON.ResumeIntent	resume / continue / keep going
AMAZON.StartOverIntent	start over / restart / start again
AMAZON.StopIntent	stop / off / shut up
AMAZON.YesIntent	yes / yes please / sure

Interaction model

business logic

“Alexa, talk to Calculator”

Session Started

idea: pre-fetch data

OnLaunch (optional)

“Hello, how can ..”

“What is one plus two”

onIntent (..)

“One plus two equals …“

“Thanks”

“Talk to you soon”

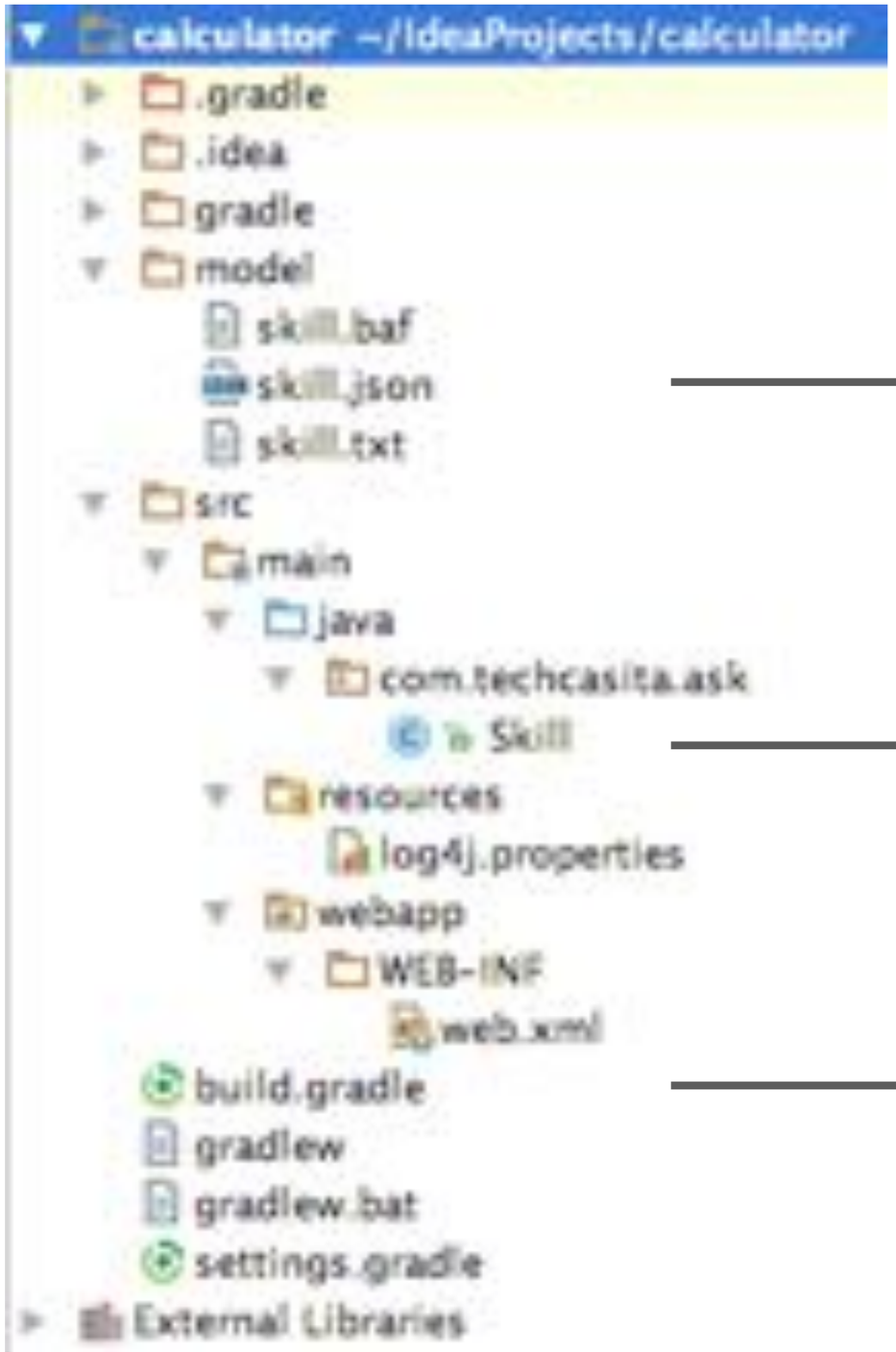
Session Ended

Life cycle

```
package com.techcasita.ask;
import com.amazon.speech.speechlet.*;
import com.amazon.speech.speechlet.servlet.SpeechletServlet;

public class Skill extends SpeechletServlet implements Speechlet {

    @Override
    public void onSessionStarted(SessionStartedRequest request, Session session) throws SpeechletException {

    }

    @Override
    public SpeechletResponse onLaunch(LaunchRequest request, Session session) throws SpeechletException {
        return null;
    }

    @Override
    public SpeechletResponse onIntent(IntentRequest request, Session session) throws SpeechletException {
        return null;
    }

    @Override
    public void onSessionEnded(SessionEndedRequest request, Session session) throws SpeechletException {

    }
}
```

SpeechletResponse

```
final SpeechletResponse response = new SpeechletResponse();
final PlainTextOutputSpeech speech1 = new PlainTextOutputSpeech();
final PlainTextOutputSpeech speech2 = new PlainTextOutputSpeech();
final Reprompt reprompt = new Reprompt();

speech1.setText("Hello, I am your Calculator");
speech2.setText("I can perform simple Addition and Subtraction …");
reprompt.setOutputSpeech(speech2);

response.setOutputSpeech(speech1);
response.setReprompt(reprompt);
response.setShouldEndSession(false);
```



Interaction model

business logic

build & deploy

Alexa Skills Kit Demo: Calculator

https://github.com/wolfpaulus/ask_calculator