

# Sviluppare per Alexa in Python

Alberto Anceschi - Nicolò Gasparini



Firenze, 4 Maggio 2019 - Pycon X



# Webranking

Webranking è una web agency che offre consulenza strategica di **digital marketing** per i più importanti brand italiani e internazionali attraverso attività di *media planning* e *data intelligence*



SEO



ADVERTISING



WEB DEV



CREATIVE & UX



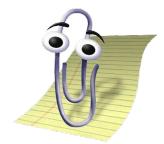
WEB ANALYTICS

# Webranking

Webranking è una web agency che offre consulenza strategica di **digital marketing** per i più importanti brand italiani e internazionali attraverso attività di *media planning* e *data intelligence*



# Smart Assistants



Clippy!



Google Now



Alexa



Bixby



1996

2011

2012

2014

2014

2016

2017

2017

Siri

Cortana

Google Assistant

AliGenie



# Cos'è Amazon Alexa?

*Alexa* è l'assistente vocale virtuale di *Amazon* basato su cloud

È il cuore dei dispositivi *Echo* ed è possibile implementarlo in qualsiasi dispositivo, anche custom-made

Utilizza tecniche di **Natural Language Understanding**,  
**Processing**, e **Speech Recognition** per individuare i  
desideri degli utenti ed è costantemente migliorato dall'**AI**

# Cos'è Amazon Alexa?

Alexa è l'assistente vocale virtuale di *Amazon* basato su cloud

È il cuore dei dispositivi *Echo* ed è possibile implementarlo in qualsiasi dispositivo, anche custom-made

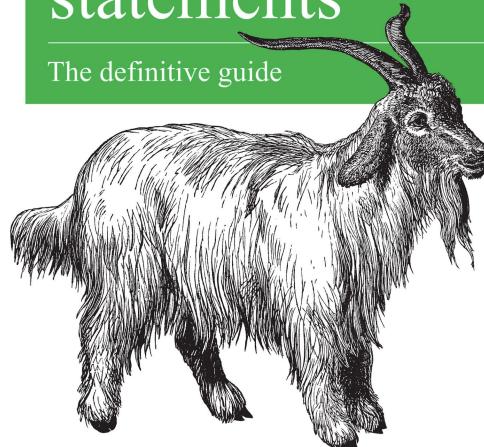
Utilizza tecniche di **Natural Language Understanding**,  
**Processing**, e **Speech Recognition** per individuare i  
desideri degli utenti ed è costantemente migliorato dall'**AI**

Nonostante abbia diverse funzionalità intrinseche per  
*intrattenimento*, *smart home* e come *organizer*, le sue  
potenzialità vengono sbloccate appieno attraverso  
l'installazione di funzionalità aggiuntive custom

O'REALLY?

AI based on  
if / else  
statements

The definitive guide



Harry Perci

# Voice Market

Gli assistenti vocali rendono bene

52  
milioni

dispositivi *Google Home* venduti  
(12/2018)

100  
milioni

dispositivi *Alexa* venduti  
(01/2019)

1,650  
m\$

premio per  
*Alexa Prize* 2019

# Voice Market

Gli assistenti vocali rendono bene (ma non benissimo)

 **Amazon Alexa Developers**  
Sponsored · 

We are offering a free Echo Dot device and Alexa Dev Socks to the first 5,000 developers in the US who publish an Alexa skill in August.



**Publish a Skill, Get an Echo Dot and Developer Socks**  
You can build a skill quickly using one of our code templates.

[DEVELOPER.AMAZON.COM](http://DEVELOPER.AMAZON.COM) [Learn More](#)

# Cosa sono le skill?

## Casa Intelligente

Per l'utilizzo di dispositivi per la *casa intelligente*, lampadine, riscaldamento...



## Sommario Quotidiano

Personalizza il news feed degli utenti, fornendo *aggiornamenti* da ascoltare

## Custom

Permette di progettare un'*esperienza unica e libera* per gli utenti.

# Cosa sono le skill?

## Casa Intelligente

Per l'utilizzo di dispositivi per la *casa intelligente*, lampadine, riscaldamento...



## Sommario Quotidiano

Personalizza il news feed degli utenti, fornendo *aggiornamenti* da ascoltare

## Custom

Permette di progettare un'*esperienza unica e libera* per gli utenti.

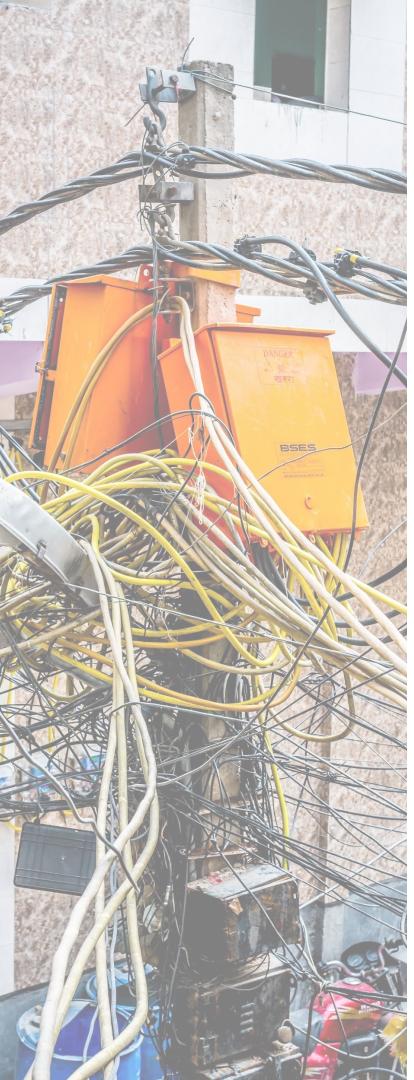
Per lo sviluppo delle skill vengono consigliati gli **strumenti AWS** (Lambda, DynamoDB...) che semplificano il processo, ma non sono un obbligo.

È quindi possibile interracciarsi anche con **altri servizi API esterni**

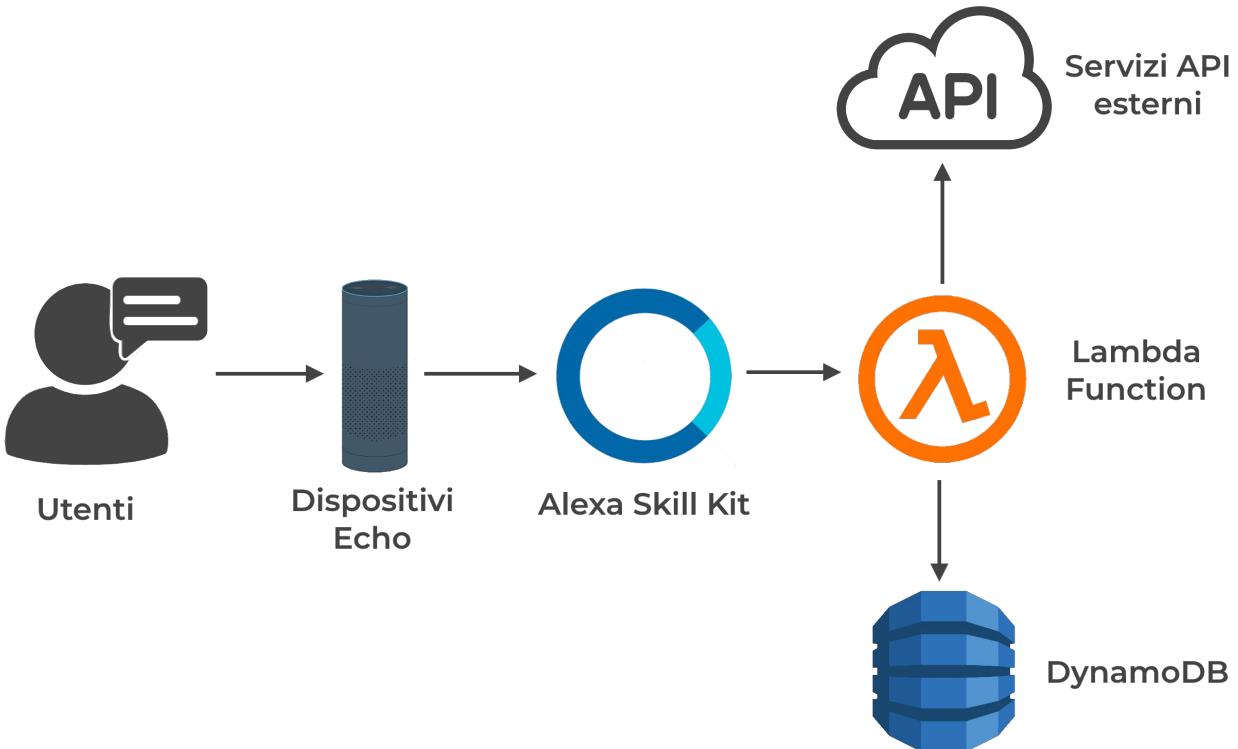
È solamente necessario rispettare alcuni requisiti.

02

## Sviluppo



# Architettura



# Alexa Developer Console

È il portale che Amazon mette a disposizione per sviluppare il modello di interazione delle Skill attraverso un'interfaccia grafica.

The screenshot shows the Alexa Developer Console interface. At the top, there's a navigation bar with links for Your Skills, Web Analytics, Build, Code, Test, Distribution, Certification, and Analytics. A search bar and a feedback forum link are also at the top right. The main area has a dark background with the "amazon alexa" logo and the text "Developer Console: Build". On the left, there's a sidebar titled "CUSTOM" which includes sections for Interaction Model, Invocation (with Intents and Built-In Intents), Slot Types, JSON Editor, Interfaces, Endpoint, Intent History, Display (Beta), and Alexa Design Guide (Beta). The central area has sections for "How to get started" (with a video thumbnail) and "Resources" (Documentation, Sample Alexa Projects, Alexa Presentation Language (APL) Documentation, and Alexa Design Guide). To the right, there's a "Skill builder checklist" with four required steps: 1. Invocation Name, 2. Intents, Samples, and Slots, 3. Build Model, and 4. Endpoint, all of which are marked as completed with green checkmarks. There's also an optional section for In-Skill Products.

# ASK CLI

Inizializzazione, clone e deploy della skill e di tutte le risorse correlate

`ask new`

Permette di creare una skill partendo da un template, con tutti i file necessari al deploy.

`ask clone`

Clona una skill presente nell'ambiente di sviluppo in una cartella locale.

`ask deploy`

Carica una skill e tutte le sue risorse nell'ambiente di sviluppo.

```
skill project folder
|
| -- .ask/
|   | -- config
| -- hooks/
|   | -- post_new_hook.ps1
|   | -- pre_deploy_hook.ps1
| -- lambda/
|   | -- lambda_function.py
| -- models/
|   | -- it-IT.json
|   | -- en-US.json
| -- skill.json
```

# ASK CLI

post\_new\_hook

1. controlla nel file manifest quali sono le sourceDir
2. all'interno di ognuna viene creato un *virtualenv*
3. per ogni *virtualenv* viene eseguito *pip* per l'installazione delle dipendenze

# ASK CLI

## post\_new\_hook

1. controlla nel file manifest quali sono le sourceDir
2. all'interno di ognuna viene creato un *virtualenv*
3. per ogni *virtualenv* viene eseguito *pip* per l'installazione delle dipendenze

## pre\_deploy\_hook

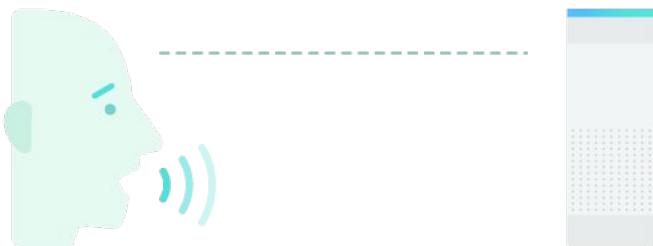
1. controlla nel file manifest quali sono le sourceDir
2. all'interno di ognuna viene creata la cartella `lambda_upload`
3. copia il codice sorgente all'interno di `lambda_upload`
4. copia le dipendenze dal *virtualenv* a `lambda_upload`
5. aggiorna il file manifest `skill.json`

# Interaction model

Invocation name, utterance, e slot

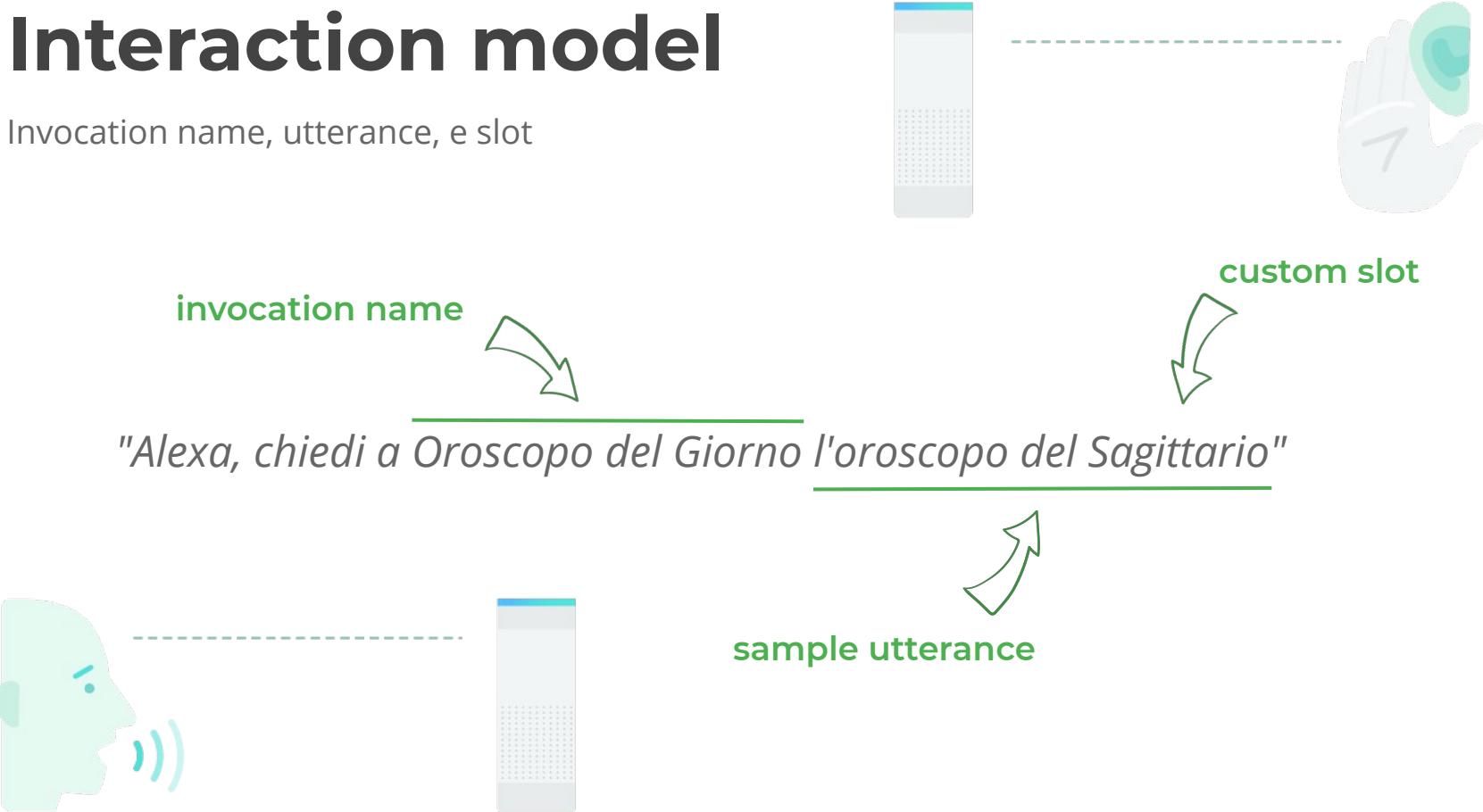


*"Alexa, chiedi a Oroscopo del Giorno l'oroscopo del Sagittario"*



# Interaction model

Invocation name, utterance, e slot



# Interaction model

Invocation name, utterance, e slot



# Interaction model

Utilizza Alexa Skill Kit per sviluppare una voice user interface (VUI) con cui mappare l'**input vocale** dell'utente a determinati **intenti**.

## Intents

Azione che risponde ad una particolare *intenzione* dell'utente.  
Possono contenere degli *slot*.

## Sample utterances

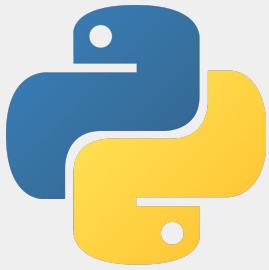
Insieme di possibili *frasi di esempio* associate ad un particolare intento.  
Più sono, meglio è.

## Custom slot types

Lista contenente i possibili valori di uno *slot*. Da utilizzare quando gli slot *built-in* non bastano.

## Dialog model (opzionale)

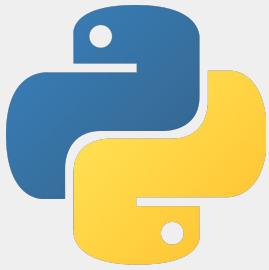
Struttura che identifica i passi per una *conversazione a più turni* tra la tua skill e l'utente.



# Alexa Skill Kit SDK

Gli elementi principali sono:

- *SkillBuilder*, oggetto principale della funzione
- *Interceptors*, utilizzati per frapporsi fra la richiesta e la sua gestione
- *Handlers*, definiscono a quali intenti possono rispondere e la logica di processo



# Alexa Skill Kit SDK

Gli elementi principali sono:

- *SkillBuilder*, oggetto principale della funzione
- *Interceptors*, utilizzati per frapporsi fra la richiesta e la sua gestione
- *Handlers*, definiscono a quali intenti possono rispondere e la logica di processo

Supporta la riproduzione **video**,  
**musicale**, la gestione delle  
**carte**, delle **liste**, il **login utente**,  
gli **slot** non correttamente  
compresi ed altre funzionalità

Package	Version	pypi	downloads
ask-sdk-runtime	v1.9.0	37k	
ask-sdk-core	v1.9.0	50k	
ask-sdk-dynamodb-persistence-adapter	v1.9.0	37k	
ask-sdk	v1.9.0	36k	
ask-sdk-webservice-support (Beta)	v0.1.1	352	
flask-ask-sdk (Beta)	v0.1.0	366	
django-ask-sdk (Beta)	v0.1.0	326	

# DynamoDB persistence adaptor

```
from ask_sdk_dynamodb.adapter import DynamoDbAdapter, user_id_partition_keygen
def __init__(self):
    dynamodb = boto3.resource('dynamodb',
                              region_name="eu-west-1",
                              aws_access_key_id="xxx",
                              aws_secret_access_key="xxx")
    dynamo_client = DynamoDbAdapter(table_name="skill-wall-street-users",
                                    partition_key_name="user_id",
                                    partition_keygen=user_id_partition_keygen,
                                    create_table=False,
                                    dynamodb_resource=self.dynamodb)
```

## Connessione



## Salvataggio e recupero dati



```
attrs = dynamo_client.get_attributes(request_envelope=handler_input.request_envelope)
dynamo_client.save_attributes(request_envelope=handler_input.request_envelope,
                               attributes={'key': 'value'})
```

# Localizzazione e Traduzione

È possibile gestire la traduzione di una skill per i diversi linguaggi supportati da *Alexa*.

L'aggiunta delle lingue supportate viene fatta dalla *Developer Console*.

Questo creerà automaticamente i modelli (*file JSON*) di linguaggio per ogni lingua



# Localizzazione e Traduzione

È possibile gestire la traduzione di una skill per i diversi linguaggi supportati da *Alexa*.

L'aggiunta delle lingue supportate viene fatta dalla *Developer Console*.

Questo creerà automaticamente i modelli (*file JSON*) di linguaggio per ogni lingua

All'interno dei testi è possibile inserire  
anche meta tag **SSML**, per sbloccare  
funzionalità aggiuntive del linguaggio  
di Alexa



```
#. NOTE: skill name
msgid "SKILL_NAME"
msgstr "<lang xml:lang='en-US'>Web Analytics</lang>"
#. NOTE: whisper response for main intent
msgid "WHISPER_RESPONSE"
msgstr "<amazon:effect name="whispered">ciao a
tutti</amazon:effect>"
```

File .po

# Localizzazione e Traduzione

```
class LocalizationInterceptor(AbstractRequestInterceptor):
    """ Add function to request attributes, used to load localized phrases"""

    def process(self, handler_input):
        # type: (HandlerInput) -> None
        locale = handler_input.request_envelope.request.locale
        if DEBUG:
            logger.info("LOCALE = {}".format(locale))
        i18n = gettext.translation('data', localedir='locales',
                                  languages=[locale], fallback=True)
        handler_input.attributes_manager.request_attributes['_'] = i18n.gettext
```



Risposta



```
_ = handler_input.attributes_manager.request_attributes['_']
speech_text = _("WELCOME").format(text="test")
handler_input.response_builder.speak(speech_text)
return handler_input.response_builder.response
```

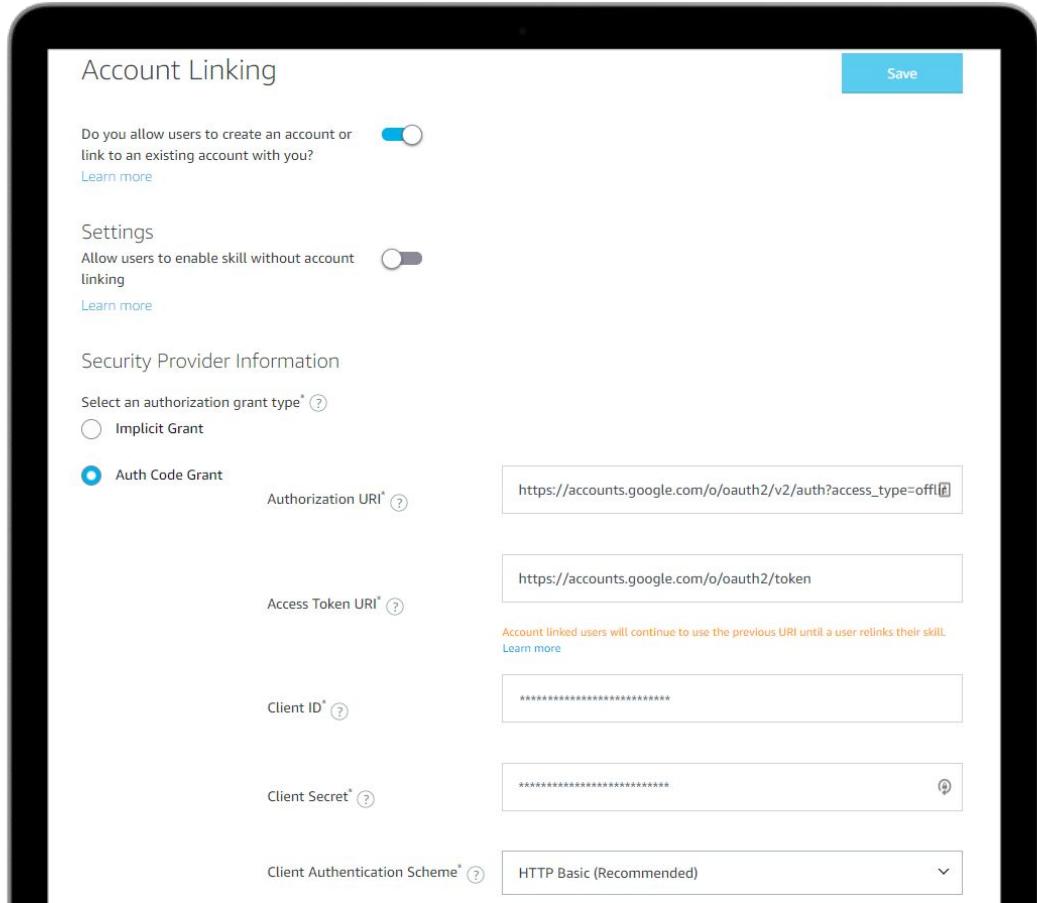


# Account Linking

Per applicazioni che necessitano  
dell'**autenticazione** di un utente, è

prevista questa procedura  
*gestita completamente da Alexa*

Ad ogni richiesta dell'utente,  
attraverso il **protocollo Oauth**,  
viene passato un *access token*,  
utilizzabile poi per accedere ai dati  
dell'utente



03

## Pubblicazione della skill

## Il Rifiutologo



Alberto Anceschi

14

DISATTIVA SKILL

## Inizia dicendo

*"Alexa, apri raccolta differenziata"*

Grazie a questa Skill puoi chiedere ad Alexa in che contenitore gettare i rifiuti. Ti basterà dire: "Alexa, chiedi a raccolta differenziata dove posso buttare uno scontrino", e Alexa ti risponderà in che contenitore dovrà essere gettato. Se il rifiuto necessita uno smaltimento particolare, il Rifiutologo...

[Visualizza altro](#)

## Informazioni

Valutato Questa Skill contiene contenuto dinamico

Frasi *Alexa, apri raccolta differenziata*

Lingue italiano (IT)

## Valutazioni

4.1 stelle su 5

14 valutazioni dei clienti

6 recensioni dei clienti

# Pubblicazione

Le Skill devono seguire una procedura di validazione prima di essere rilasciate sullo store.

## Testing Automatici + Controllo Manuale

Prima di mandarla in approvazione controlla la *Submission Checklist* sulla documentazione ufficiale.

Sono necessari un' icona, la descrizione della skill, keyword e tre frasi di esempio con cui l'utente potrebbe invocare la Skill.



# Errori da evitare

- Errori grammaticali nel testo di presentazione della Skill  
(attenzione alle maiuscole)



# Errori da evitare

- Errori grammaticali nel testo di presentazione della Skill (attenzione alle maiuscole)
- Le risposte che lasciano il parametro `should_end_session` a `False`, devono lasciare l'utente con una domanda



# Errori da evitare

- Errori grammaticali nel testo di presentazione della Skill (attenzione alle maiuscole)
- Le risposte che lasciano il parametro `should_end_session` a `False`, devono lasciare l'utente con una domanda
- La risposta fornita dall'intento `HelpIntent` deve spiegare chiaramente cosa è possibile fare all'interno della Skill



# Errori da evitare

- Errori grammaticali nel testo di presentazione della Skill (attenzione alle maiuscole)
- Le risposte che lasciano il parametro `should_end_session` a `False`, devono lasciare l'utente con una domanda
- La risposta fornita dall'intento `HelpIntent` deve spiegare chiaramente cosa è possibile fare all'interno della Skill
- In caso esistano già diverse Skill simili pubblicate, è importante che la Skill si "presenti" in risposta ad una invocazione generica dell'utente



# Errori da evitare

- Errori grammaticali nel testo di presentazione della Skill (attenzione alle maiuscole)
- Le risposte che lasciano il parametro `should_end_session` a `False`, devono lasciare l'utente con una domanda
- La risposta fornita dall'intento `HelpIntent` deve spiegare chiaramente cosa è possibile fare all'interno della Skill
- In caso esistano già diverse Skill simili pubblicate, è importante che la Skill si "presenti" in risposta ad una invocazione generica dell'utente
- Evitare bug che possano mandare in errore l'esecuzione del codice della funzione lambda



# Errori da evitare

- Errori grammaticali nel testo di presentazione della Skill (attenzione alle maiuscole)
- Le risposte che lasciano il parametro `should_end_session` a `False`, devono lasciare l'utente con una domanda
- La risposta fornita dall'intento `HelpIntent` deve spiegare chiaramente cosa è possibile fare all'interno della Skill
- In caso esistano già diverse Skill simili pubblicate, è importante che la Skill si "presenti" in risposta ad una invocazione generica dell'utente
- Evitare bug che possano mandare in errore l'esecuzione del codice della funzione lambda

<https://gist.github.com/WRinnovation/4dfc935a627f0be6d8ba82a52683dbd1>

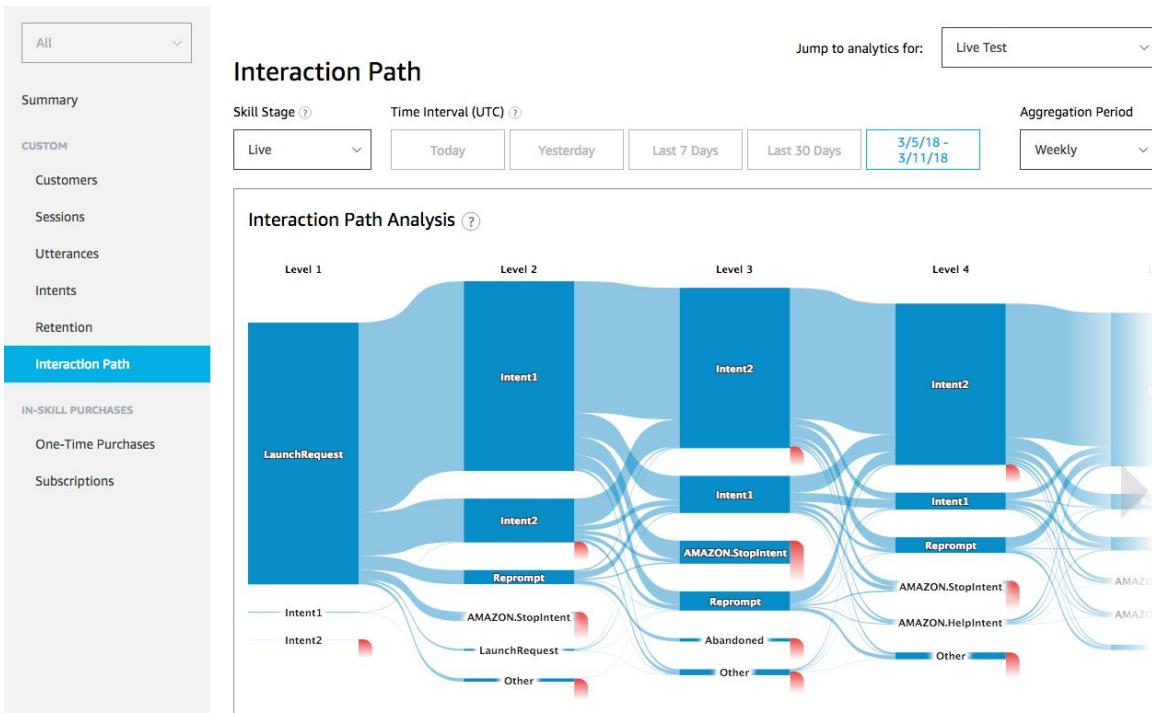
04

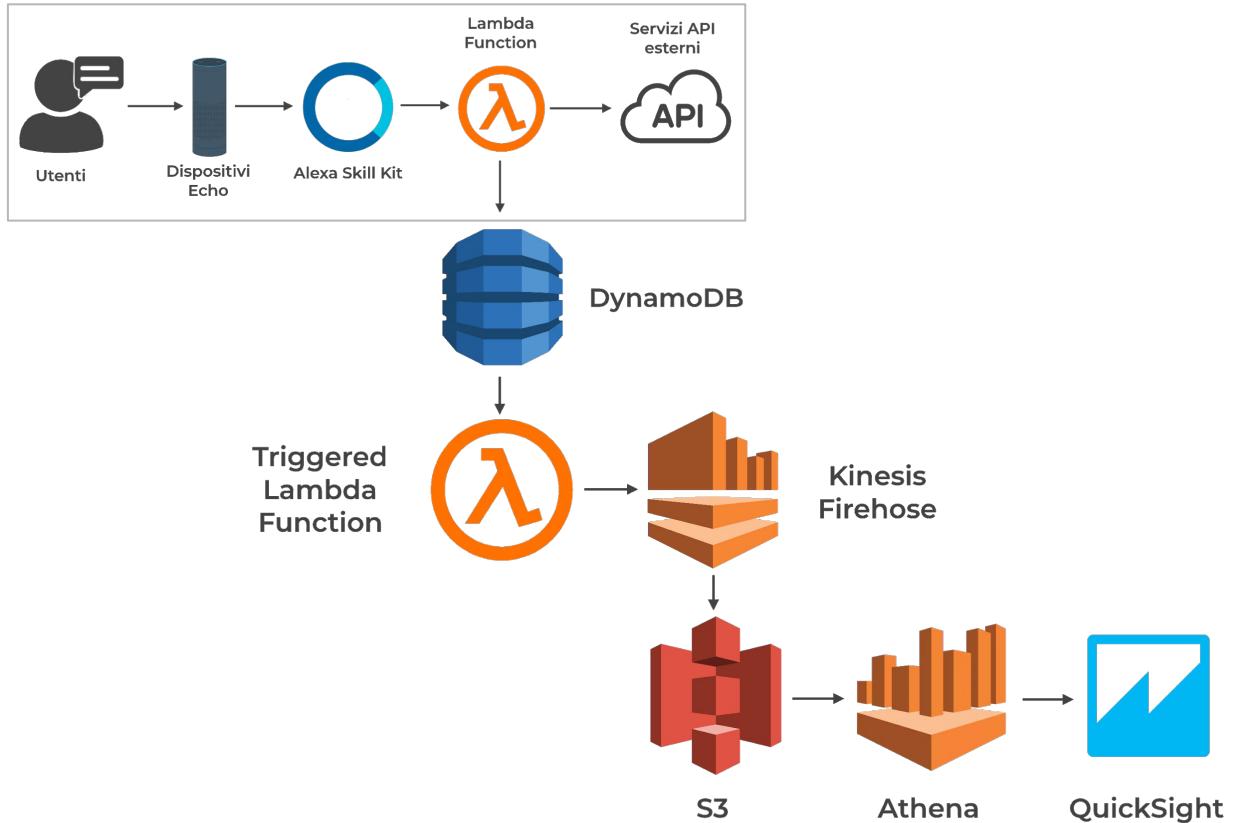
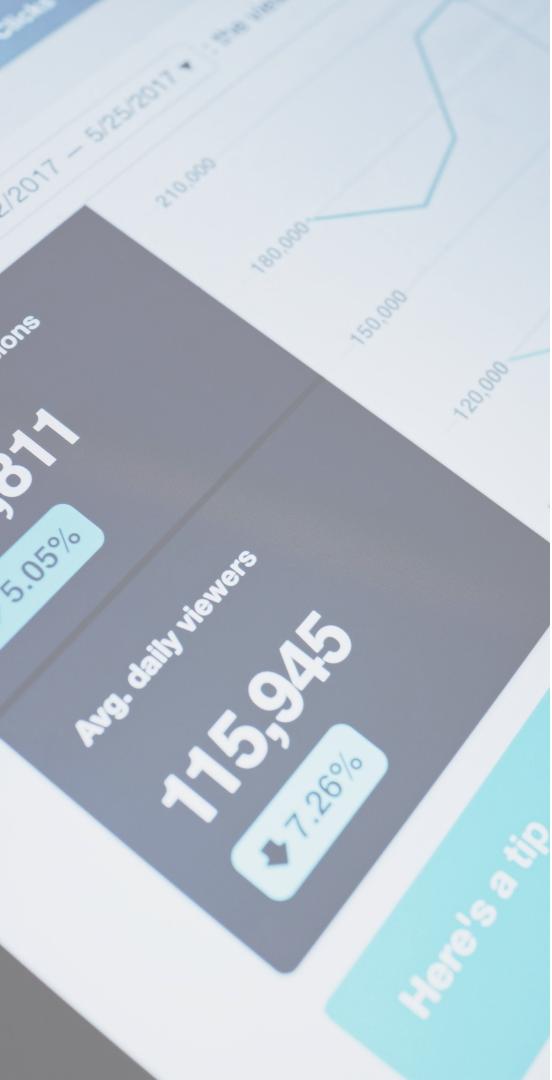
## Analisi delle richieste

# Skill Analytics

La Alexa Developer Console fornisce alcuni insight interessanti sull'andamento della propria skill.

Ad esempio vengono mostrate le **attivazioni, invocazioni, nuovi utenti e il loro percorso di interazione** con la skill.





05

## Custom skill code

# Custom Skill - Lambda

```
sb = SkillBuilder()

class RequestLogger(AbstractRequestInterceptor):
    """ Logs the alexa request """
    def process(self, handler_input):
        logger.debug("ALEXA REQUEST: {}".format(handler_input.request_envelope.request))

    sb.add_request_handler(LaunchRequestHandler())
    sb.add_request_handler(HelpIntentHandler())
    sb.add_request_handler(ExitIntentHandler())

    sb.add_request_handler(KpiHandler())
    sb.add_global_request_interceptor(RequestLogger())

lambda_handler = sb.lambda_handler()
```

# Custom Skill - Handler

```
class CustomTimeHandler(AbstractRequestHandler):

    def can_handle(self, handler_input):
        # type: (HandlerInput) -> bool
        return (is_request_type("IntentRequest") (handler_input)
                and is_intent_name("CustomTimeIntent") (handler_input))

    def handle(self, handler_input):
        # type: (HandlerInput) -> Response

        speech_text = "Sono le {}".format(datetime.now().strftime("%H e %M"))

        handler_input.response_builder.speak(speech_text)

        return handler_input.response_builder.response
```

# Custom Skill - Handler

```
class CustomTimeHandler(AbstractRequestHandler):

    def can_handle(self, handler_input):
        # type: (HandlerInput) -> bool
        return (is_request_type("IntentRequest") (handler_input)
                and is_intent_name("CustomTimeIntent") (handler_input))

    def handle(self, handler_input):
        # type: (HandlerInput) -> Response

        speech_text = "Sono le {}".format(datetime.now().strftime("%H e %M"))
        card_text = "{}".format(datetime.now().strftime("%H:%M"))
        card = SimpleCard(title="Orario", content=card_text)

        handler_input.response_builder.speak(speech_text).set_card(card)

        return handler_input.response_builder.response
```

# Custom Skill - Handler

```
class CustomTimeHandler(AbstractRequestHandler):

    def can_handle(self, handler_input):
        # type: (HandlerInput) -> bool
        return (is_request_type("IntentRequest") (handler_input)
                and is_intent_name("CustomTimeIntent") (handler_input))

    def handle(self, handler_input):
        # type: (HandlerInput) -> Response
        slots = handler_input.request_envelope.request.intent.slots
        slot_value = slots['City'].value

        speech_text = "A {} sono le {}".format(slot_value, datetime.now().strftime("%H e %M"))
        card_text = "{}".format(datetime.now().strftime("%H:%M"))
        card = SimpleCard(title="Orario di {}".format(slot_value), content=card_text)

        handler_input.response_builder.speak(speech_text).set_card(card)

        return handler_input.response_builder.response
```



# Web Analytics Skill

*Google Analytics* è un servizio di Web analytics gratuito di *Google* che consente di analizzare le statistiche di un sito web  
Raccoglie dati su sessioni, utenti, registrazioni ed altri eventi avvenuti sul sito web



Come *Webranking* abbiamo voluto esplorare le potenzialità della **voice search** e come gli utenti vi interagiscono, creando uno strumento il più semplice e completo possibile

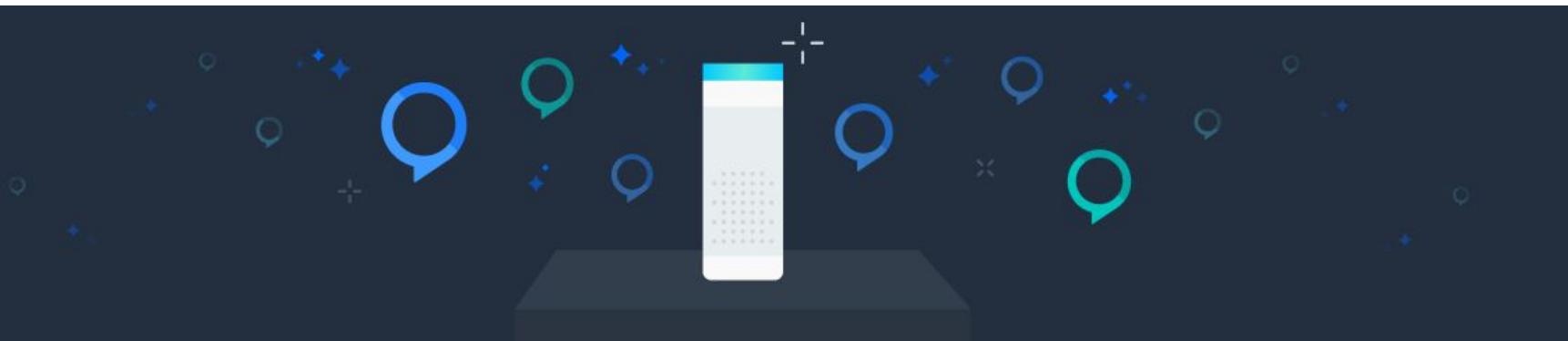
# Concludendo...

Lo sviluppo delle skill di *Alexa*, a prescindere dal linguaggio, è semplificato al massimo dalla possibilità di utilizzo degli **strumenti AWS** e dalle **SDK disponibili**. Lo sviluppatore può concentrarsi solamente sulle logiche di risposta, senza dover gestire il dialogo

Le potenzialità di Alexa sono praticamente illimitate grazie alle **custom skill**

Lo studio migliore è sugli esempi pratici trovati sulle pagine *GitHub* di *Alexa*

A questo link potete trovare un template di **progetto blank** per prendere ispirazione ed iniziare facilmente: <https://github.com/nicogaspa/alexa-skill-blank-template>



# Grazie

Alberto Anceschi  
[a.anceschi@webranking.it](mailto:a.anceschi@webranking.it)

Nicolò Gasparini  
[n.gasparini@webranking.it](mailto:n.gasparini@webranking.it)

# Sources

- <https://www.theverge.com/2019/1/4/18168565/amazon-alexa-devices-how-many-sold-number-100-million-dave-limp>
- <https://voicebot.ai/2018/12/24/rbc-analyst-says-52-million-google-home-devices-sold-to-date-and-generating-3-4-billion-in-2018-revenue/>
- <https://developer.amazon.com/alexaprize/challenges/current-challenge/faqs>
- <https://github.com/alexa/alexa-skills-kit-sdk-for-python>
- <https://medium.freecodecamp.org/how-to-create-an-alexa-skill-that-manages-to-do-lists-11c4bab29ea5>
- <https://github.com/alexa/skill-sample-nodejs-petmatch/tree/master/analytics>
- <https://developer.amazon.com/it/docs/account-linking/understand-account-linking.html>

