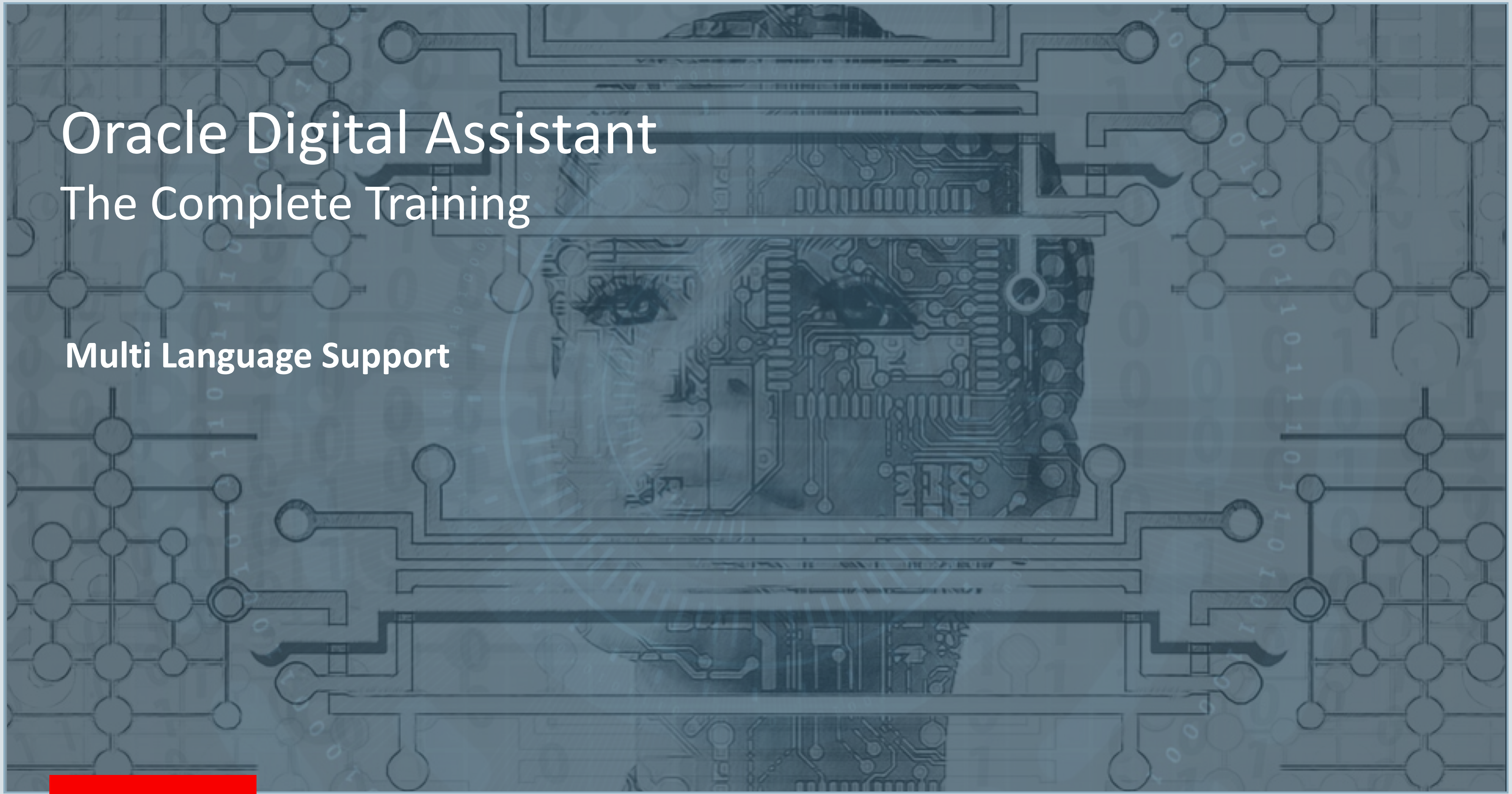


# Oracle Digital Assistant

## The Complete Training

**Multi Language Support**

ORACLE

Image courtesy of pixabay.com

## Safe Harbor Statement

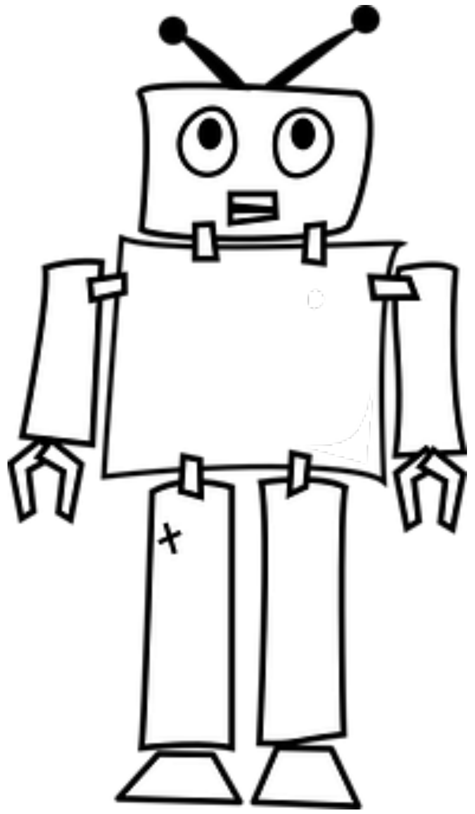
## Topic agenda

ORACLE

## Topic agenda

ORACLE

## What do you expect from a **multi language** chatbot?



## Ok, so what might our options be for building **multi-language** bots?

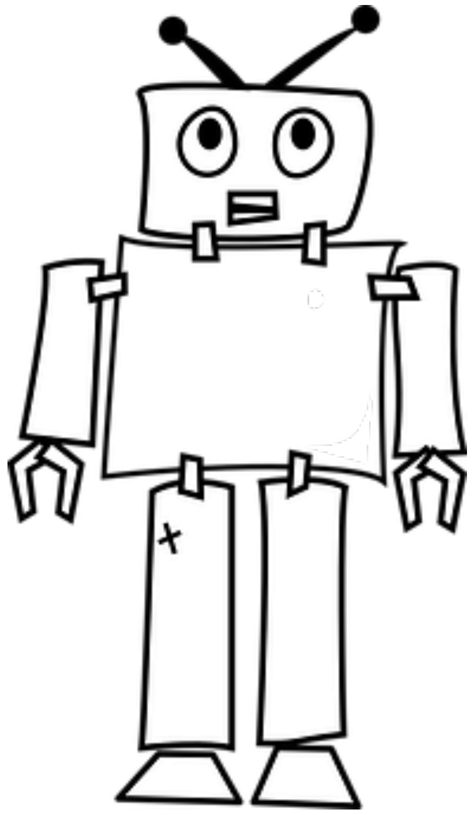


Image courtesy of Farbentek at FreeDigitalPhotos.net

# Two approaches to building multi-language bots

## Native language bots

- Build the bot in the language it will be used in
- Platform has to support NLP for that (and every) language
  - Utterances
  - Entities
  - Prompts, titles, descriptions
- What happen for other languages?

## Single base-language bots

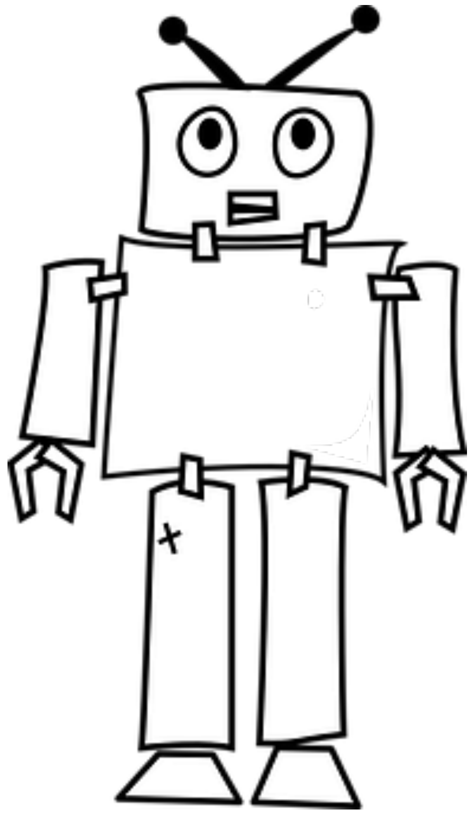
- Serve multiple languages from a single base language
- Uses translation service
  - Prompts are translated at runtime to detected user language
  - User input translated to base language
- Only one NLP engine required
- Only "appears" as native language

ORACLE

## Why single base-language bots work

If no mistake have you made, yet losing you are … **a different game you should play**"

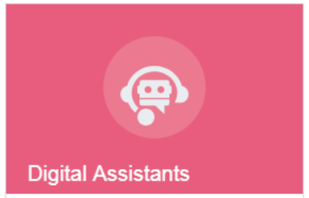
- Yoda, Star Wars



# Multi language support in Oracle Digital Assistant

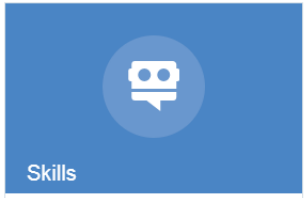
## Digital assistant

- Oracle Digital Assistant supports English (19.1.5)
  - Intents, routing
- "Predominant" language allows developing single language non – English Digital Assistant

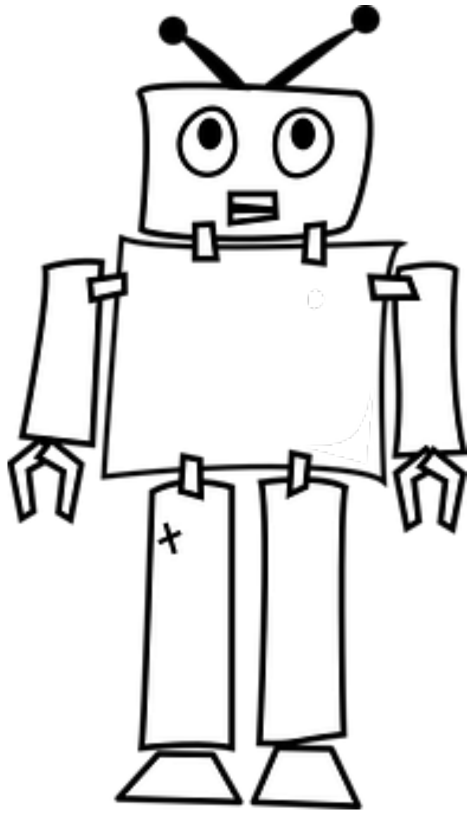


## Skills

- True multi-language support
  - You need to use skill as stand alone bots
  - Expose skills directly on a messaging channel
  - Uses translation service (Microsoft, Google) to translate messages



From here, **this session will focus on** building muli language **skills.**

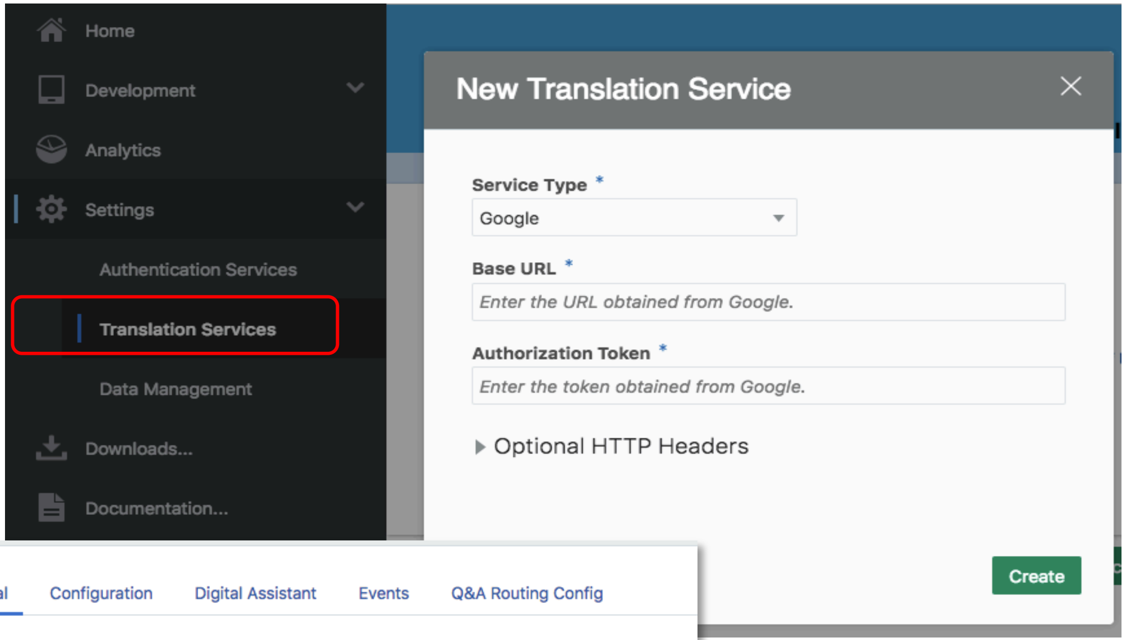


## Topic agenda

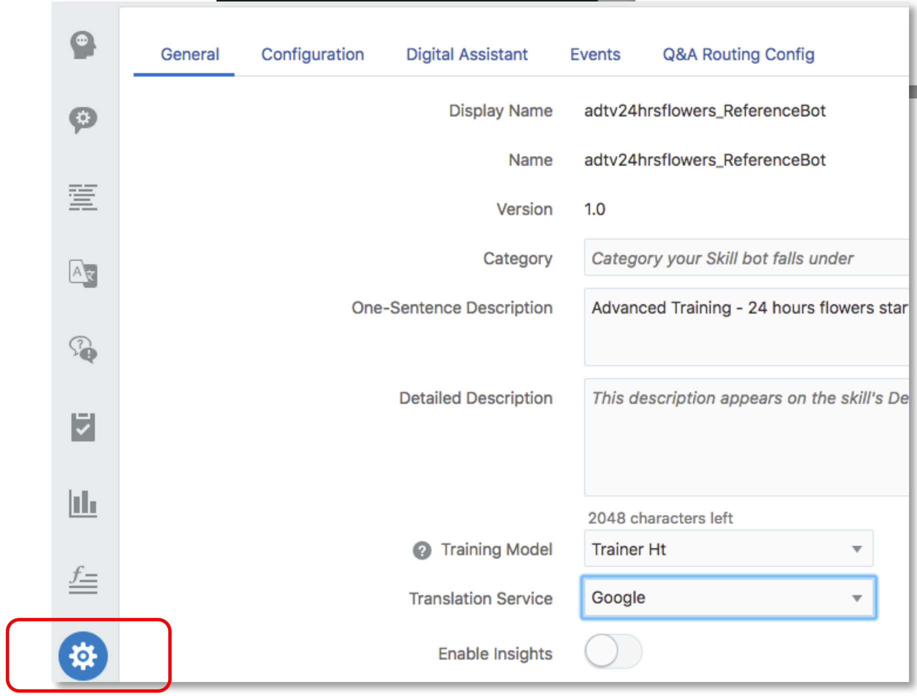
ORACLE

## Setting up a translation service

- Use Google Translation API or Microsoft Translator Services
- Bring your own license
  - Authorization Token / Key
- Translation service is used for input and output messages at runtime
- Open kill Settings
- Select *Translation Service* in "General" tab



Service Type * Google
Base URL * Enter the URL obtained from Google.
Authorization Token * Enter the token obtained from Google.

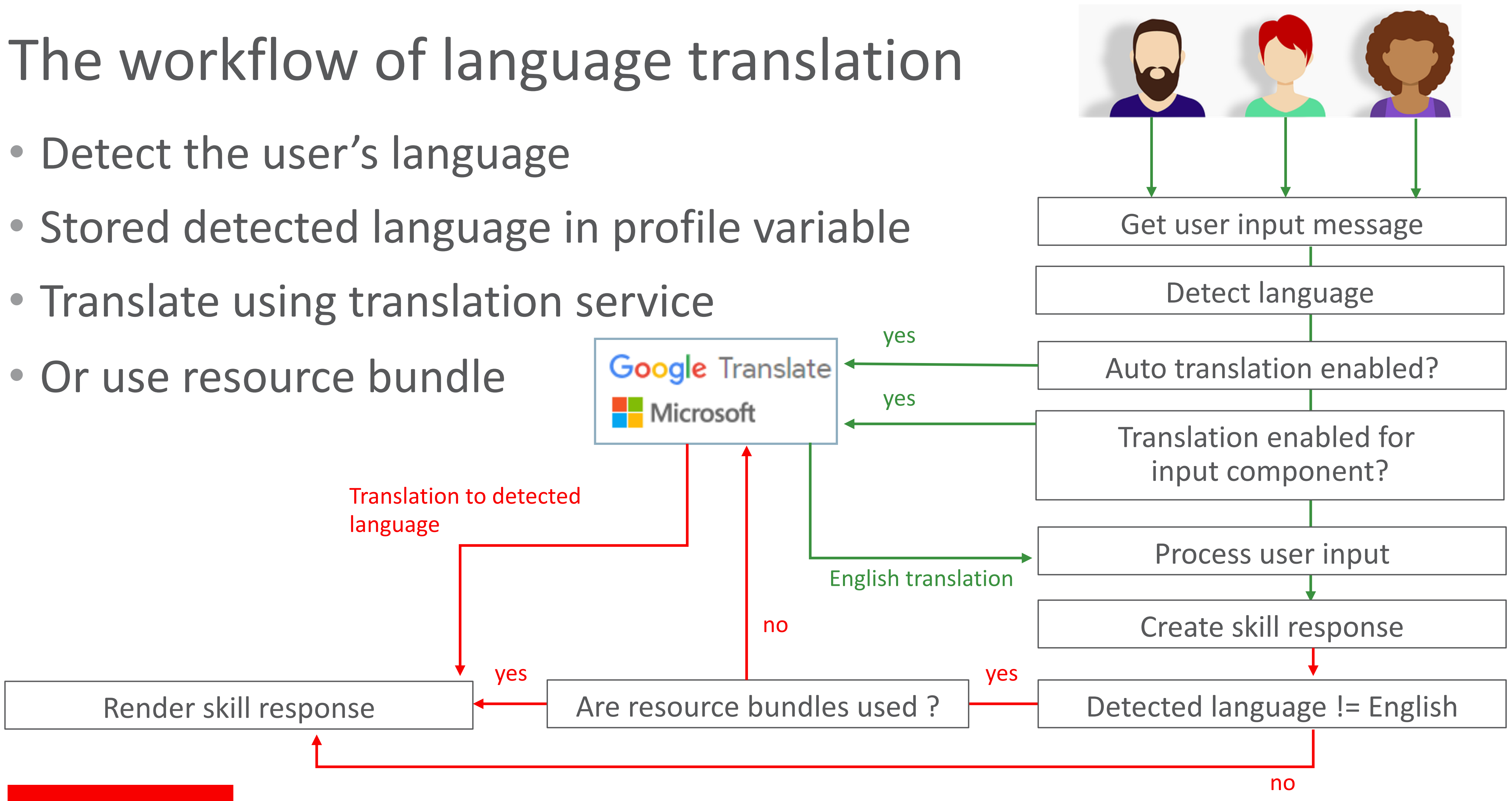


| | |
|---|---|
| Display Name | adtv24hrsflowers_ReferenceBot |
| Name | adtv24hrsflowers_ReferenceBot |
| Version | 1.0 |
| Category | Category your Skill bot falls under |
| One-Sentence Description | Advanced Training - 24 hours flowers star |
| Detailed Description | This description appears on the skill's De |
| | 2048 characters left |
| Training Model | Trainer Ht |
| Translation Service | Google |
| Enable Insights | |

## Topic agenda

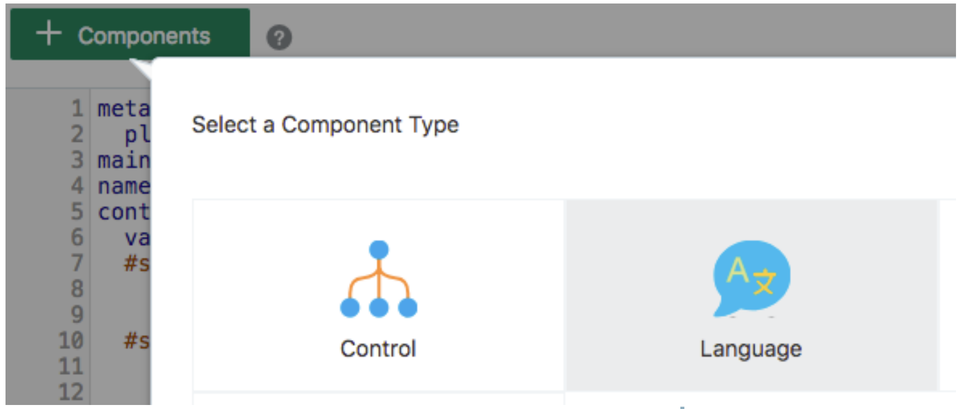
# The workflow of language translation

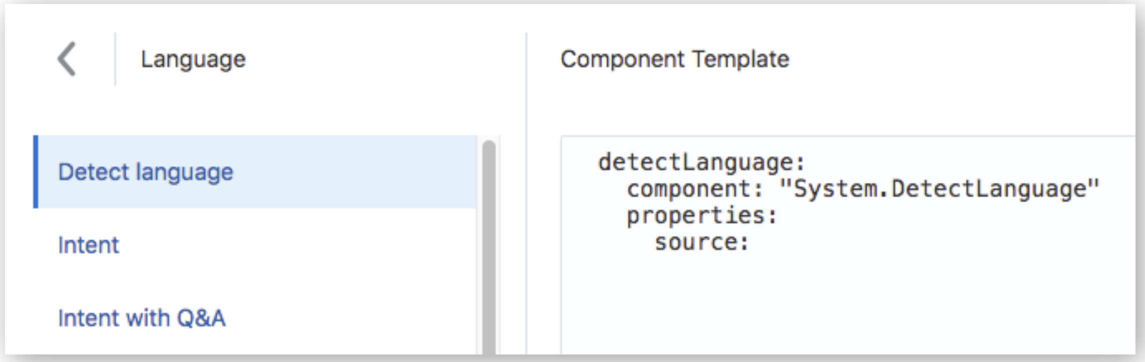
- Detect the user's language
- Stored detected language in profile variable
- Translate using translation service
- Or use resource bundle



## System.DetectLanguage

- Detects language from user input
  - Uses translation service
  - Detects language from user message
    - Optional 'source' property can be used to read user message from variable
- Sets profile.languageTag variable
  - User language saved as 2 character code
    - E.g. "fr", but not "fr-ca"





```
detectLanguage:
  component: "System.DetectLanguage"
  properties:
    source:
```

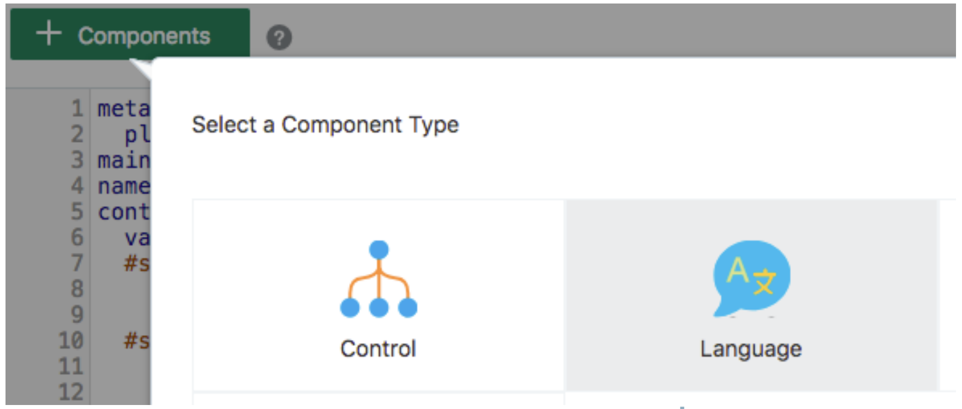
## Enabling / disabling auto-translation

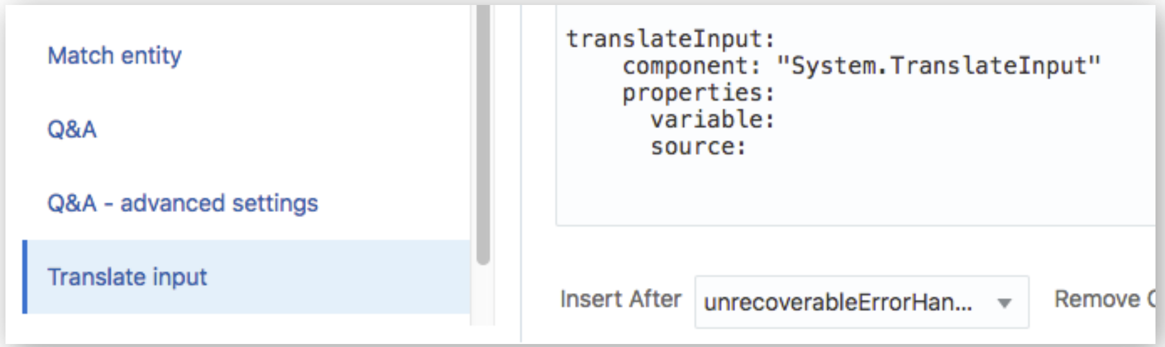
- Enabled / disabled auto-translation
  - Define "autoTranslate" context variable of type boolean
  - Set variable value to true to enable auto-translation
- When enabled
  - User messages are translated to English
  - skill messages are translated to user language

```
variables:
  autoTranslate: "boolean"
  …
states:
  …
  enableAutotranslation:
    component: "System.SetVariable"
    properties:
      variable: "autoTranslate"
      value: true
    transitions: {}
```

## System.TranslateInput

- Translates user message from detected language to English
  - Uses translation service
- Translates user entered messages
  - Optional '*source*' property used to reference variable holding string to translate
- *'variable'* property references dialog flow variable to store the translated string

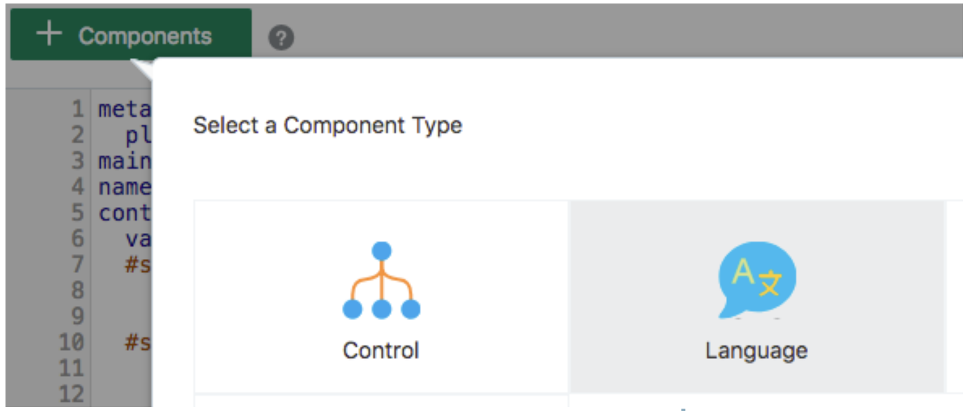


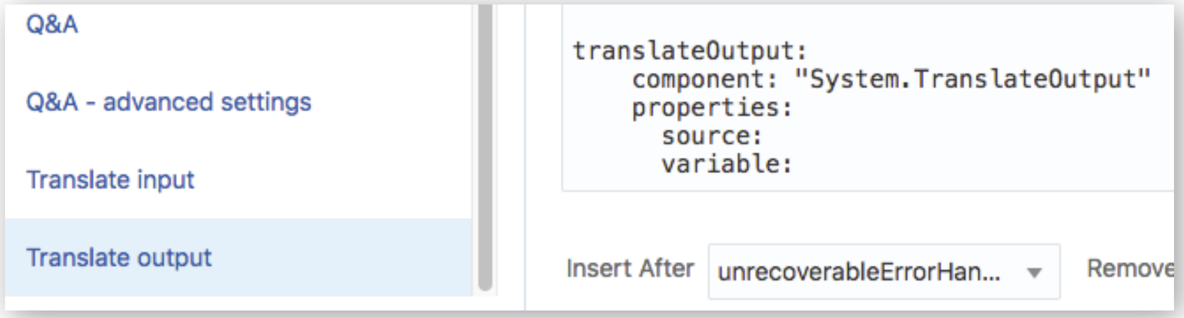


```
translateInput:
    component: "System.TranslateInput"
    properties:
      variable:
      source:
```

# System.TranslateOutput

- Translates English strings to detected user language
- Uses translation service
- *'source'* property references variable holding the English string to translate
- variable referenced in component '*variable*' property gets updated with translated string





```
translateOutput:
    component: "System.TranslateOutput"
    properties:
      source:
      variable:
```

## Topic agenda

## Language profile variables

- profile.locale
  - Set by the messenger client based on user setting
  - ${profile.locale}
- profile.languageTag
  - Holds language detected at runtime
  - Set by System.DetectLanguage component
  - Can be set manually using System.SetVariable
  - Precedes profile.locale setting
  - ${profile.languageTag}
- Determine the language used by QnA

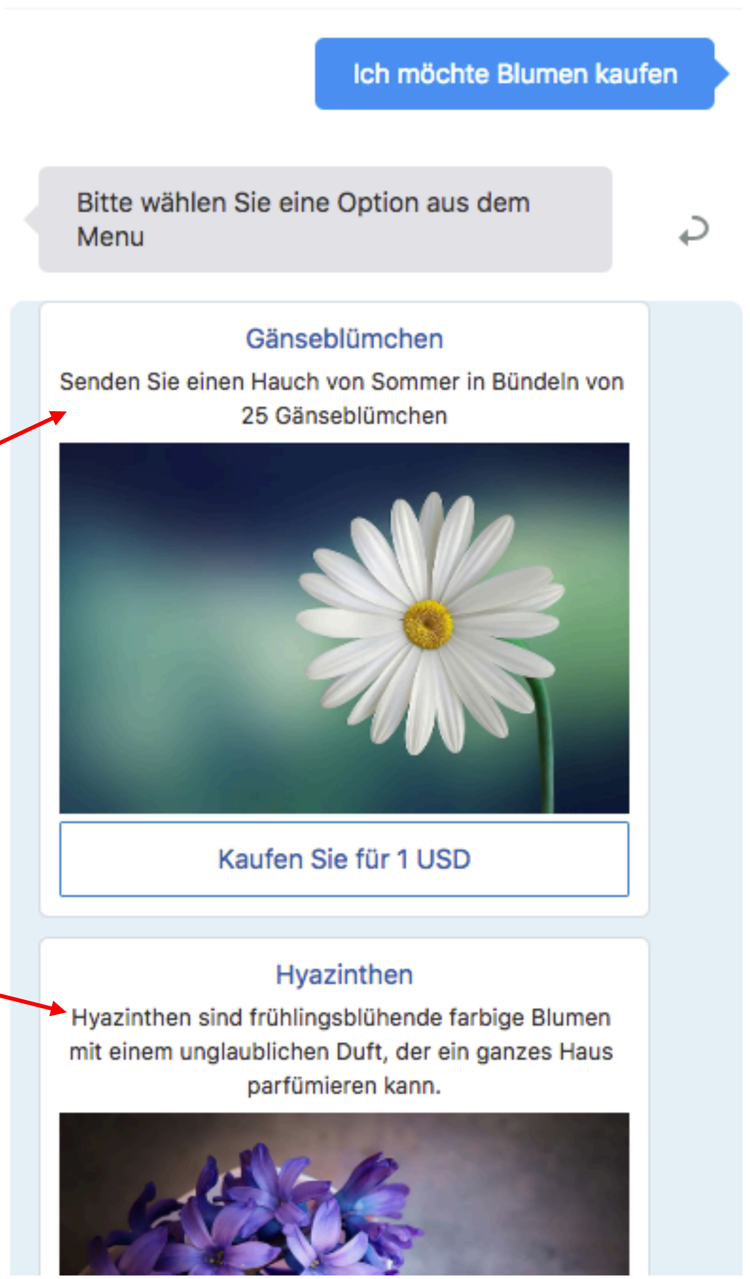
## Topic agenda

## Component 'translate' property

- Boolean property
  - Enables / disables auto-translation for components
  - Set to true
    - If auto-translation is not enabled but component should use translation service
    - If component input is not expected to be English and auto-translation is disabled
    - Always on System.Intent components to be able to resolve intents from non-English language
  - Set to false for components that use resource bundles
- Requires System.DetectLanguage to be used early in dialog flow

# 'translate' property in action

```
showFlowersMenu:
  component: "System.CommonResponse"
  properties:
    variable: "flowersName"
    nlpResultVariable: "iResult"
    processUserMessage: true
    translate: true
    metadata:
      responseItems:
      - type: "text"
        text: "${rb.orderFlowersMenuPrompt}"
      - type: "cards"
        cardLayout: "vertical"
        cards:
        - title: "${menu.title}"
          description: "${menu.description}"
          imageUrl: "${advtImagesHost.value}${menu.image}"
          iteratorVariable: "menu"
          rangeStart: "${orderMenuRangeIndex.value}"
          rangeSize: "${orderMenuRangeSize.value}"
          actions:
          - label: "${rb.orderBuyFor} ${menu.price} USD"
            type: "postback"
            payload:
              action: "copyValueAction"
              variables:
                flowersName: "${menu.title}"
                flowerCost: "${menu.price}"
```



Ich möchte Blumen kaufen

Bitte wählen Sie eine Option aus dem Menu

**Gänseblümchen**

Senden Sie einen Hauch von Sommer in Bündeln von 25 Gänseblümchen

Kaufen Sie für 1 USD

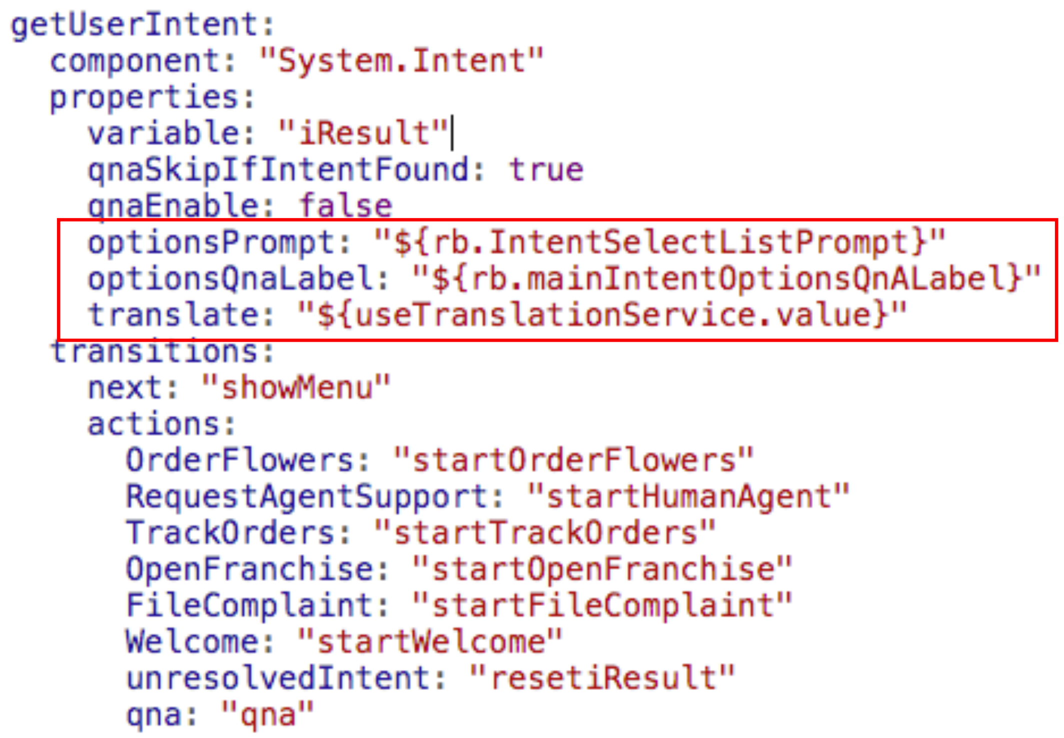
**Hyazinthen**

Hyazinthen sind frühlingsblühende farbige Blumen mit einem unglaublichen Duft, der ein ganzes Haus parfümieren kann.

# Topic agenda

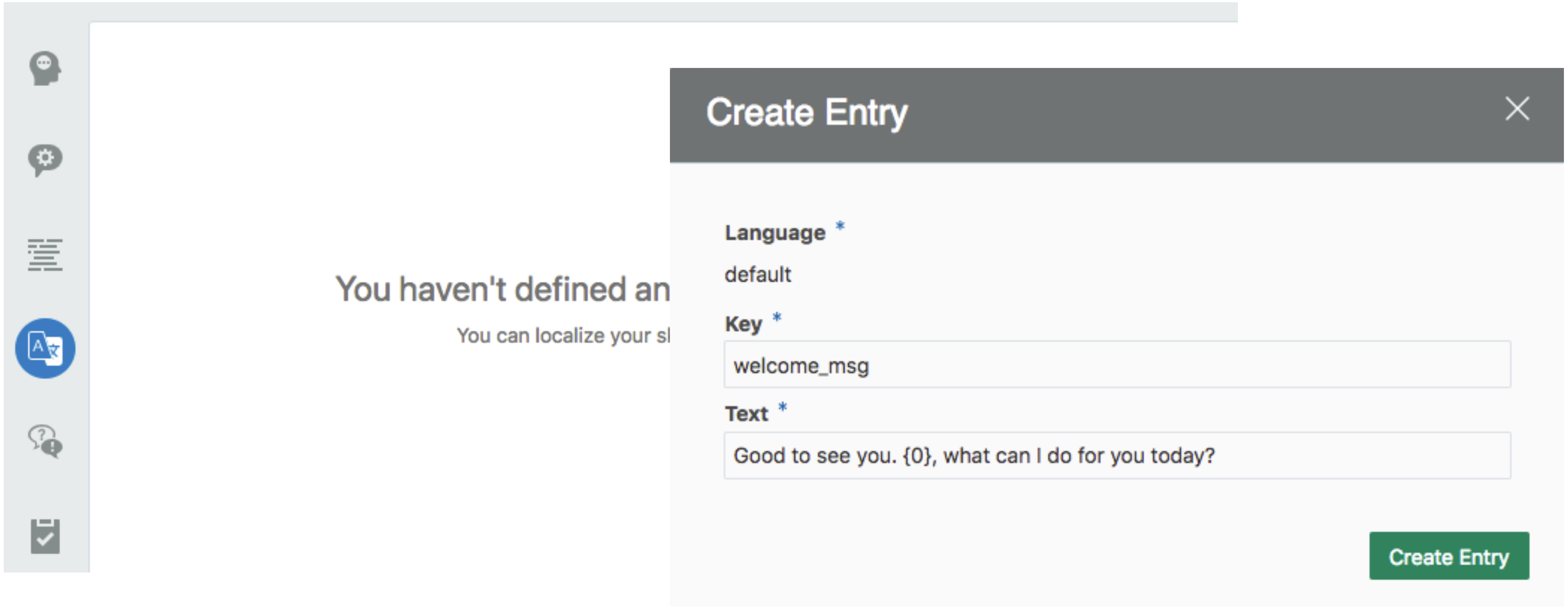
## Use resource bundle for prompts and skill messages

- Ensures appropriate language and tone presented to user
- Doesn't require a translation service
- Set component '*translate*' property to false if auto translation is enabled for a skill (opt-out)
  - Property can be set dynamically at runtime



```
getUserIntent:
  component: "System.Intent"
  properties:
    variable: "iResult"
    qnaSkipIfIntentFound: true
    qnaEnable: false
    optionsPrompt: "${rb.IntentSelectListPrompt}"
    optionsQnaLabel: "${rb.mainIntentOptionsQnALabel}"
    translate: "${useTranslationService.value}"
  transitions:
    next: "showMenu"
    actions:
      OrderFlowers: "startOrderFlowers"
      RequestAgentSupport: "startHumanAgent"
      TrackOrders: "startTrackOrders"
      OpenFranchise: "startOpenFranchise"
      FileComplaint: "startFileComplaint"
      Welcome: "startWelcome"
      unresolvedIntent: "resetiResult"
      qna: "qna"
```

# Creating resource bundles


You haven't defined an

You can localize your sl

**Language** *

default

**Key** *

welcome_msg

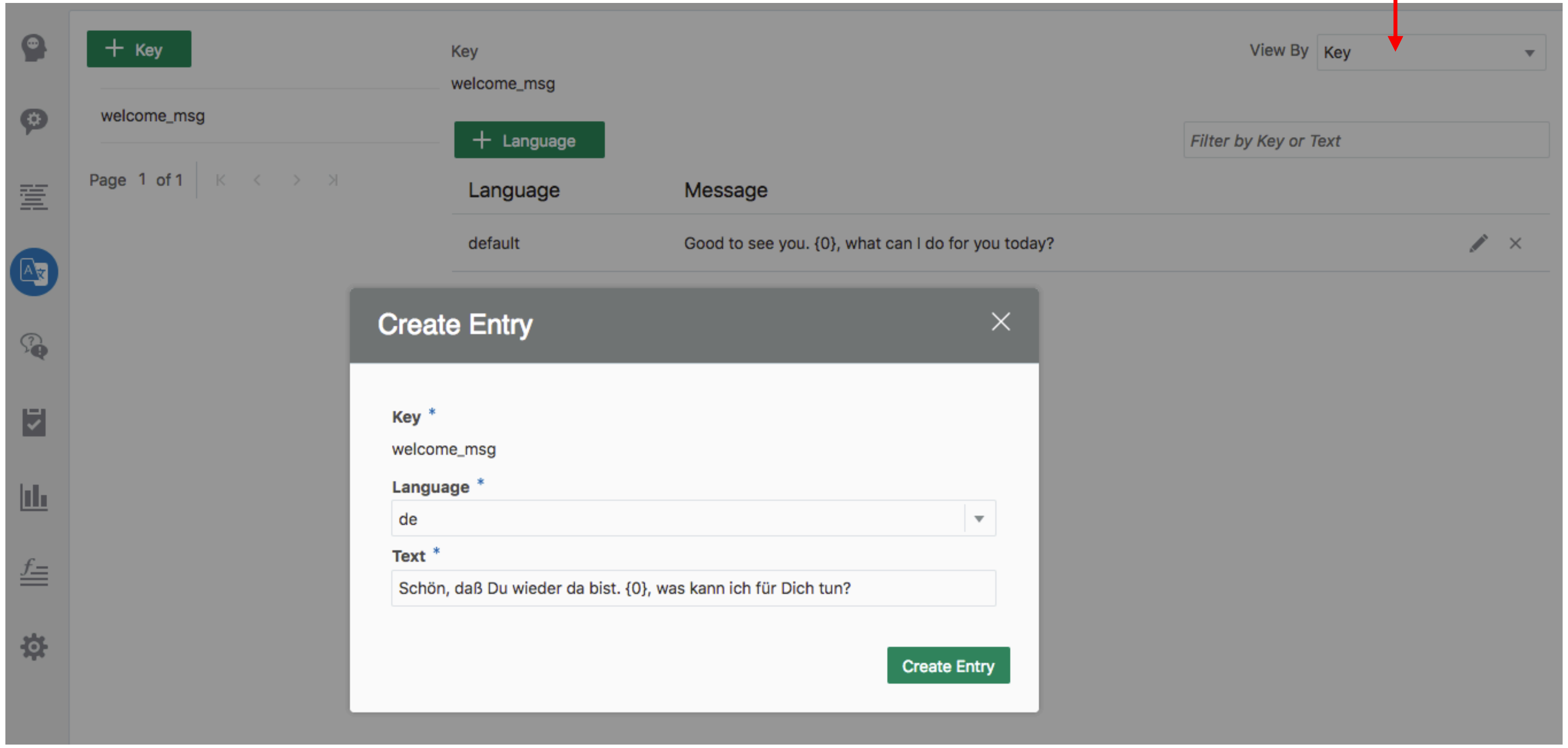
**Text** *

Good to see you. {0}, what can I do for you today?

# Creating a translation

- Select "+ Language"
- Add or select a two letter language code
  – "de", "fr", "es" etc.
- Select a key
- Provide a translation string for the English message string

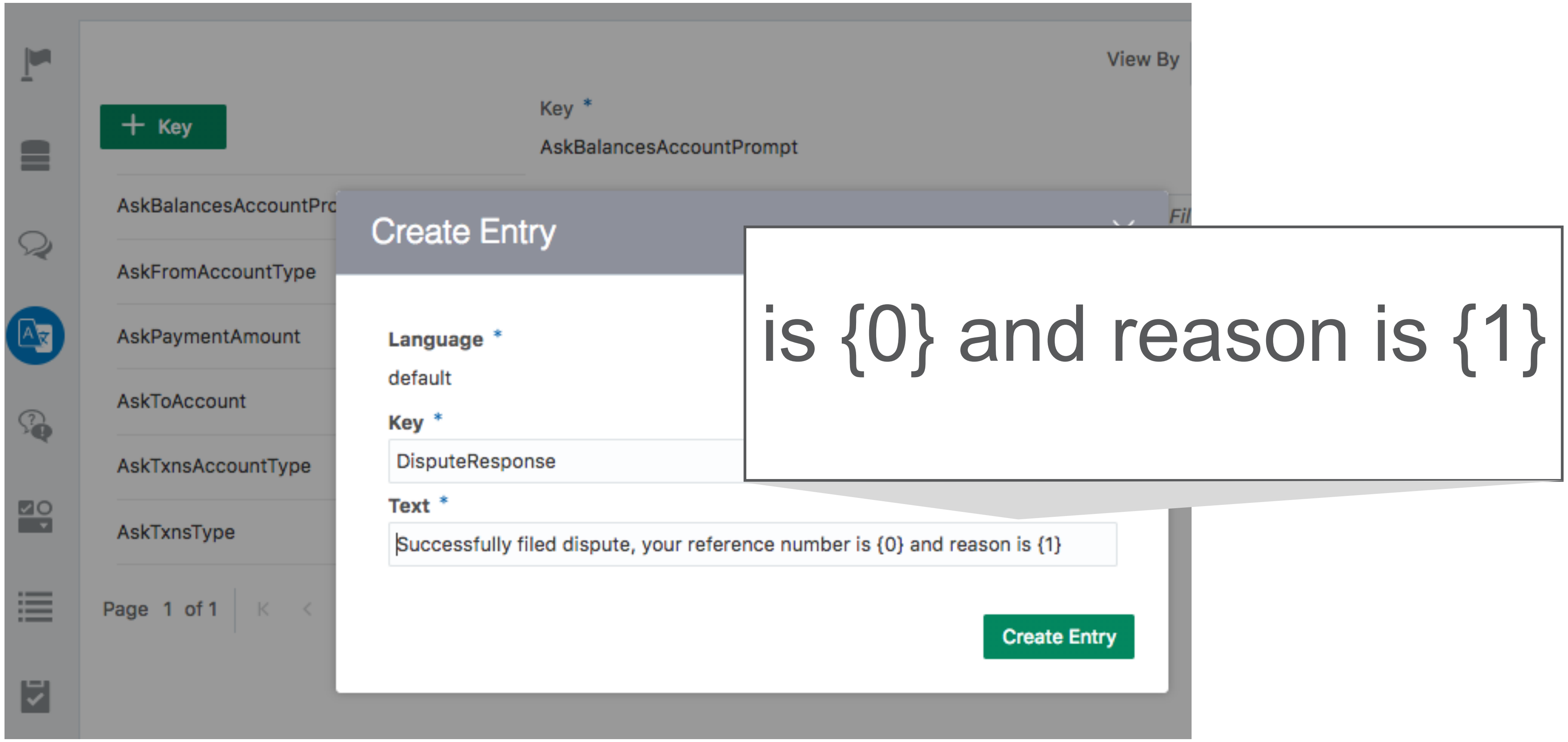
View strings by key or by language



| Language | Message |
|---|---|
| default | Good to see you. {0}, what can I do for you today? |

Key *
welcome_msg
Language *
de
Text *
Schön, daß Du wieder da bist. {0}, was kann ich für Dich tun?

# Defining variables in a resource string



View By

Key *

AskBalancesAccountPrompt

AskBalancesAccountPro

AskFromAccountType

AskPaymentAmount

AskToAccount

AskTxnsAccountType

AskTxnsType

Page 1 of 1

Create Entry

Language *

default

Key *

DisputeResponse

Text *

Successfully filed dispute, your reference number is {0} and reason is {1}

is {0} and reason is {1}

## Using resource bundles for a skills

- Add variable of type "resourcebundle"
- Access resource string with no parameters
  - ${rb('message_key')} or ${rb.message_key}
- Access resource string with single parameter
  - ${rb('message_key','optional_parameter')}
- Access resource string with multiple parameter
  - ${rb('message_key','optional_parameter',' ...',' ...')}

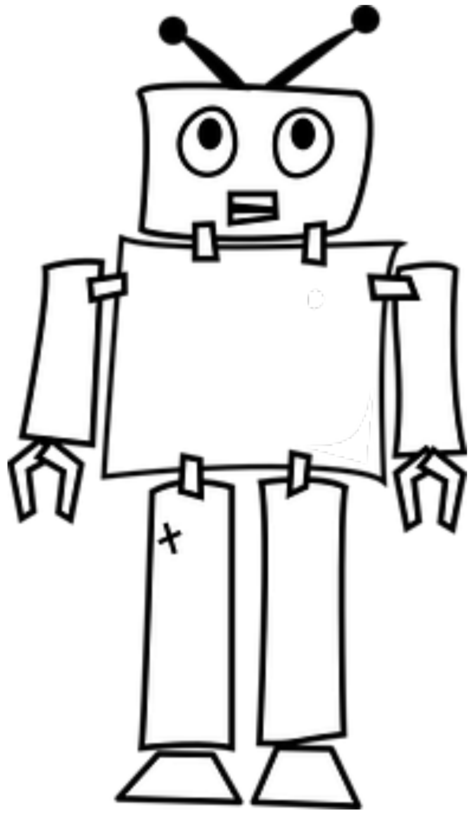
```
context:
  variables:
    rb: "resourcebundle"

getUserIntent:
  component: "System.Intent"
  properties:
    variable: "iResult"
    qnaSkipIfIntentFound: true
    qnaEnable: false
    optionsPrompt: "${rb.IntentSelectListPrompt}"
    optionsQnaLabel: "${rb.mainIntentOptionsQnALabel}"
    translate: "${useTranslationService.value}"
  transitions:
    next: "showMenu"
    actions:
      OrderFlowers: "startOrderFlowers"
      RequestAgentSupport: "startHumanAgent"
      TrackOrders: "startTrackOrders"
      OpenFranchise: "startOpenFranchise"
      FileComplaint: "startFileComplaint"
      Welcome: "startWelcome"
      unresolvedIntent: "resetiResult"
      qna: "qna"
```

```
confirmOrderAndQuantity:
  component: "System.Output"
  properties:
    text: "${rb('orderConfirmOrderQuantity','${orderQuantity.value.number}','\"${orderProductName.value}\"')}"
    keepTurn: true
  transitions:
    next: "askDeliveryOption"
```

## Topic agenda

ORACLE

**Custom components don't share** the translation service and resource bundles configured for a skill



## Options to return a custom component message response

- Custom component saves data in a dialog flow variable
  - Variable referenced from output component (e.g. System.CommonResponse)
  - Component's translation setting determines whether saved data gets auto-translated
- Custom component sends response directly to the messenger
  - Does not require system components
  - skill stays out of the loop and thus cannot help with translations
  - Translation must be part of the custom component design

# Example: translating data saved in a dialog flow variable

## Custom Component Code

```
…
let product = {product: "an apple", type: "fruit", origin: "Spain"} ;
conversation.variable('data_variable', product );
conversation.transition();
done();
```

## BotML

```
printProduct:
  component: "System.Output"
  properties:
    text: "The product in your cart is a ${data_variable.value.type}. It is
           ${data_variable.value.product} from ${data_variable.value.origin}"
    translate: true
```

## Topic agenda

## Translation strategies

### Opt-in

- Disable auto-translation
- Detect user languages
  - From user message
  - From profile
- Enable auto-translation on component
- Use message bundles

### Opt-out

- Enable auto-translation
- Detect user languages
  - From user message
  - From profile
- Test skill
- Disable translation on individual components and use message bundles instead

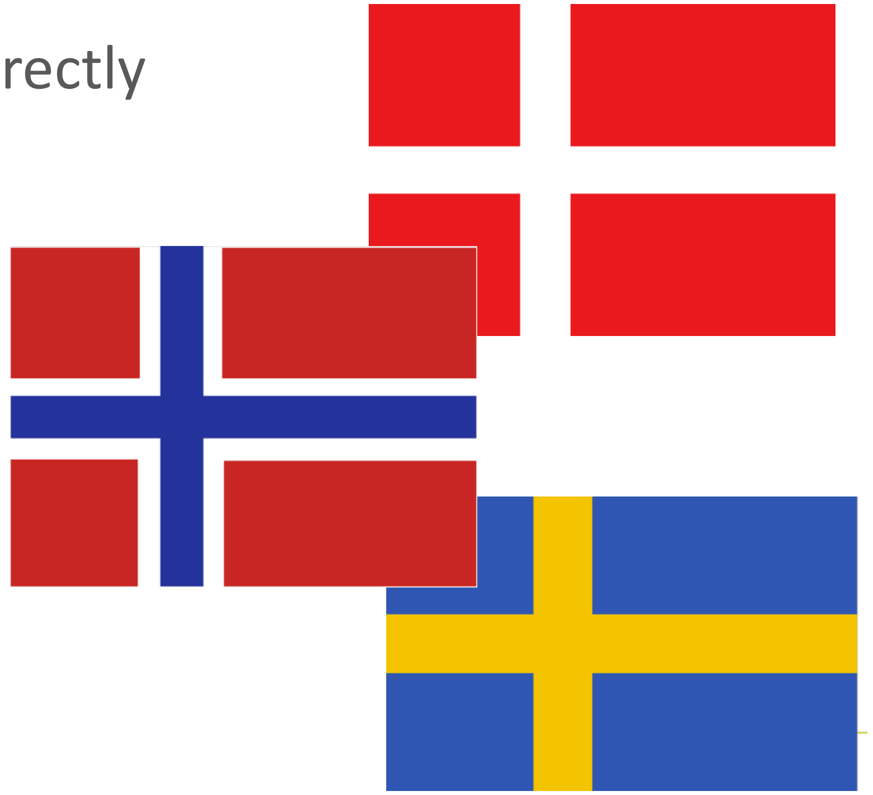
# Ensure good entity recognition

- Back-and-forth test the translation service
  - Translate an English string into a foreign language and the translate it back to English
  - Use synonyms in entities where the translation service deviates from the original
- Consider "blind testing" testing
  - skill developers knows about the utterances
  - Good testing aims for skills to fail, not to succeed
- Avoid use of abbreviations or slang even if understood in a region
  - E.g. use "checking account" instead of "checking"
- Guide users
  - Use value lists whenever possible

ORACLE

## Consider limitations of language detection

- Be aware of closely related languages
  - Translation service may fail to detect language correctly
- For example: "Good morning my friend"
  - Swedish: “God morgon min vän!”
  - Danish: “God morgen min ven!”
  - Norwegian: ”God morgen min venn!”
- Ask user if in doubt



## Control the languages to support

- Using a translation service your skill probably understands more languages than you need
- It does not make sense to support languages you don't speak or for which you have no expertise in house
- Limit the languages to support to those you regularly test and that you have resource bundles for
- To limit the set of languages
  - Detect a user language and compare it to a list of supported languages
  - Don't detect the user language but have the user selecting a preferred language

ORACLE

# Integrated Cloud

## Applications & Platform Services

## Oracle Digital Assistant  Hands-On

TBD