Developing ChatBots using Amazon Lex

Written for AllState, Belfast, June 2018

Contents

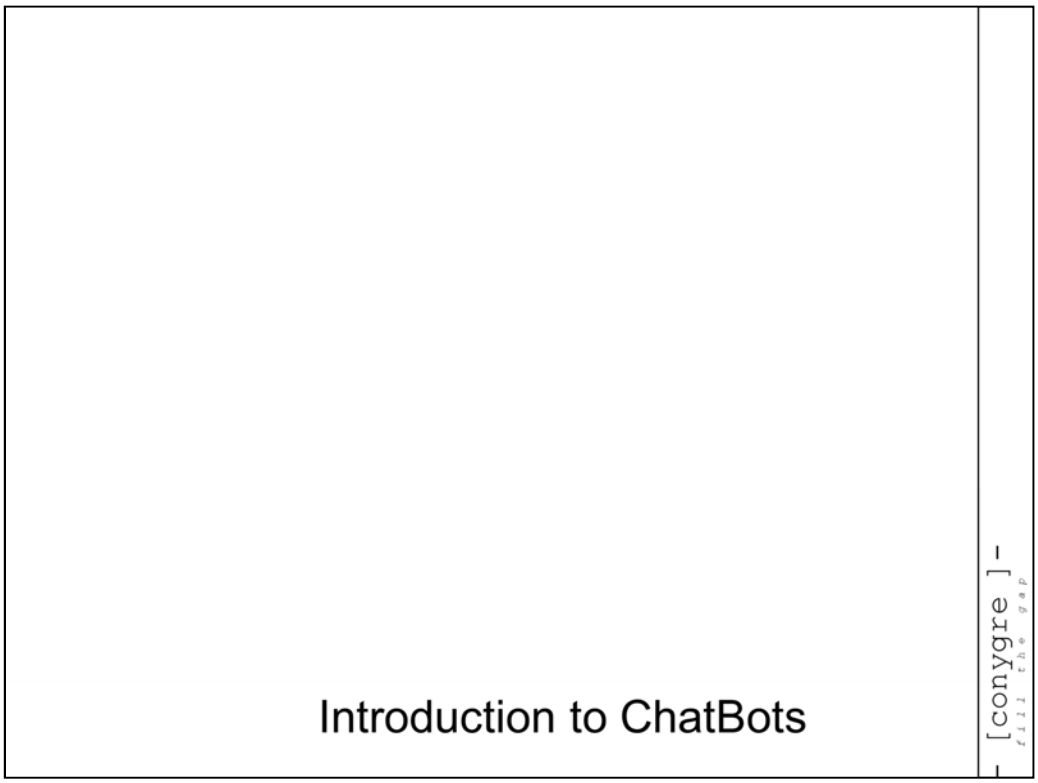
©Copyright 15/06/2018 Conygre IT Limited
All rights reserved

All trademarked product and company names are the property of their respective trademark holders.

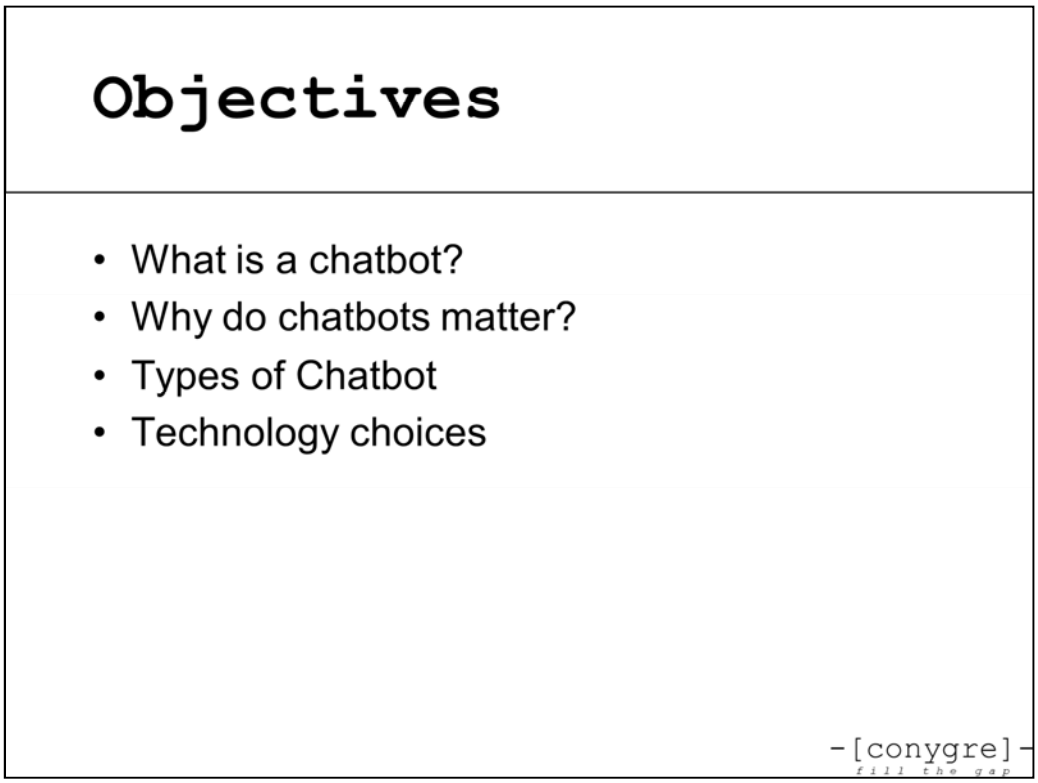
No part of this publication may be reproduced, stored in a retieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, or translated into any language, without the prior written consent of the publisher.

Conygre IT Limited shall not be liable in any way in respect of any loss, damages, costs, liabilities, or expenses suffered by the user, whether directly or indirectly as a result of the content, format, presentation, or any other aspect of the materials.

Introduction to ChatBots



© Conygre IT Ltd. Jun-18
Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

What is a ChatBot?

- According to the dictionary, a chat bot is
  - *"a computer program designed to simulate conversation with human users, especially over the Internet."*
- So a chatbot is a piece of software that can talk to people via chat over a messaging channel

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Why do ChatBots Matter?

- Messaging apps have become increasingly popular over recent years, supplanting social networks
  - *'over 2.5 billion people have at least one messaging app installed' and 'many teenagers now spend more time on smartphones sending instant messages than perusing social networks. WhatsApp users average nearly 200 minutes each week using the service.'* – The Economist

-[conygre]-
fill the gap

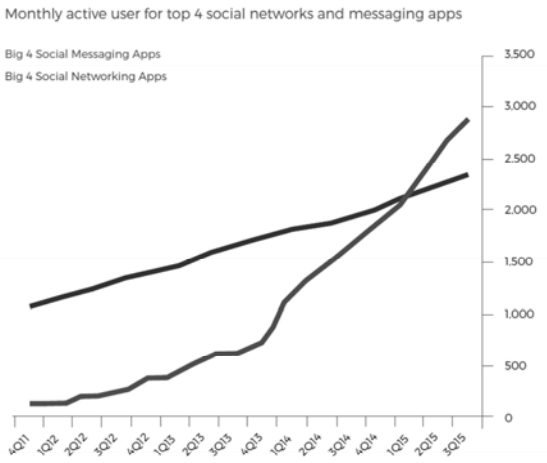
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Messenger App Usage

- Messenger apps have surpassed social media apps

- *Source: The Messaging App Report, Business Insider*

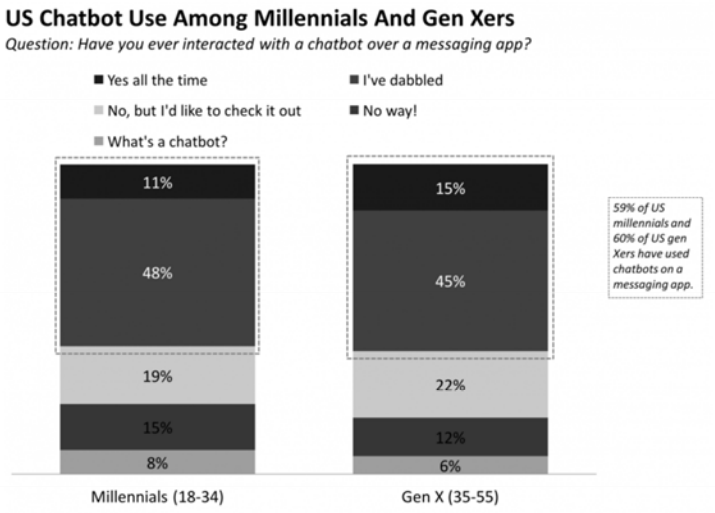


-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Chat Bot Usage

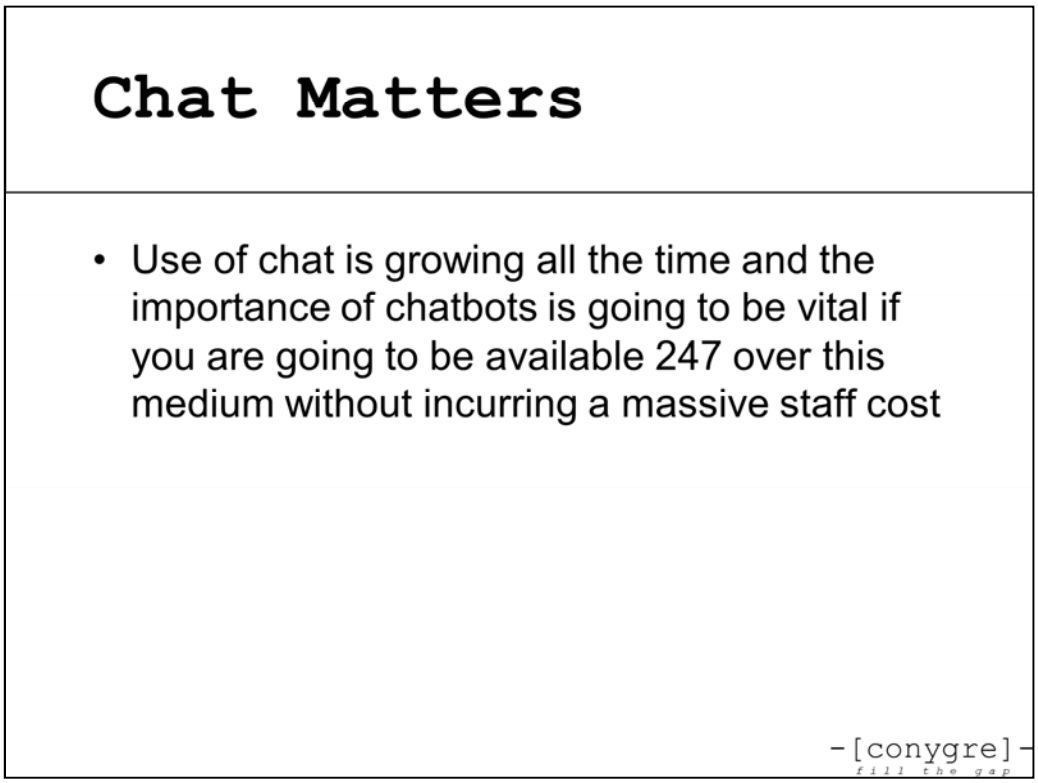


© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

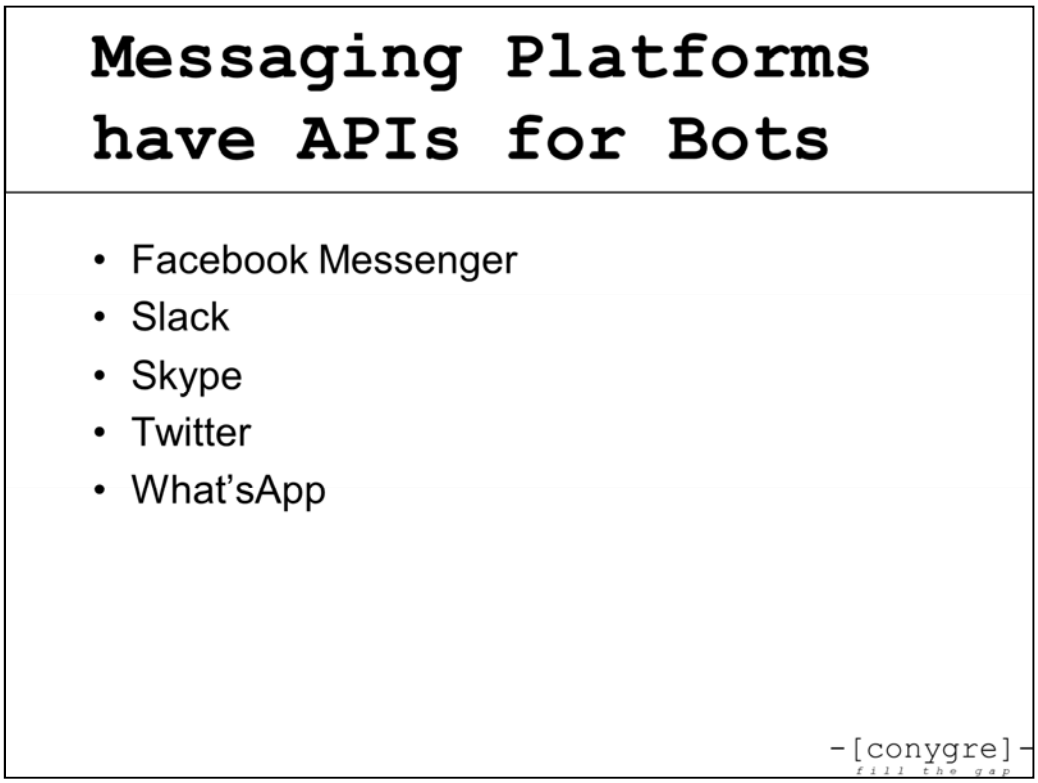
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



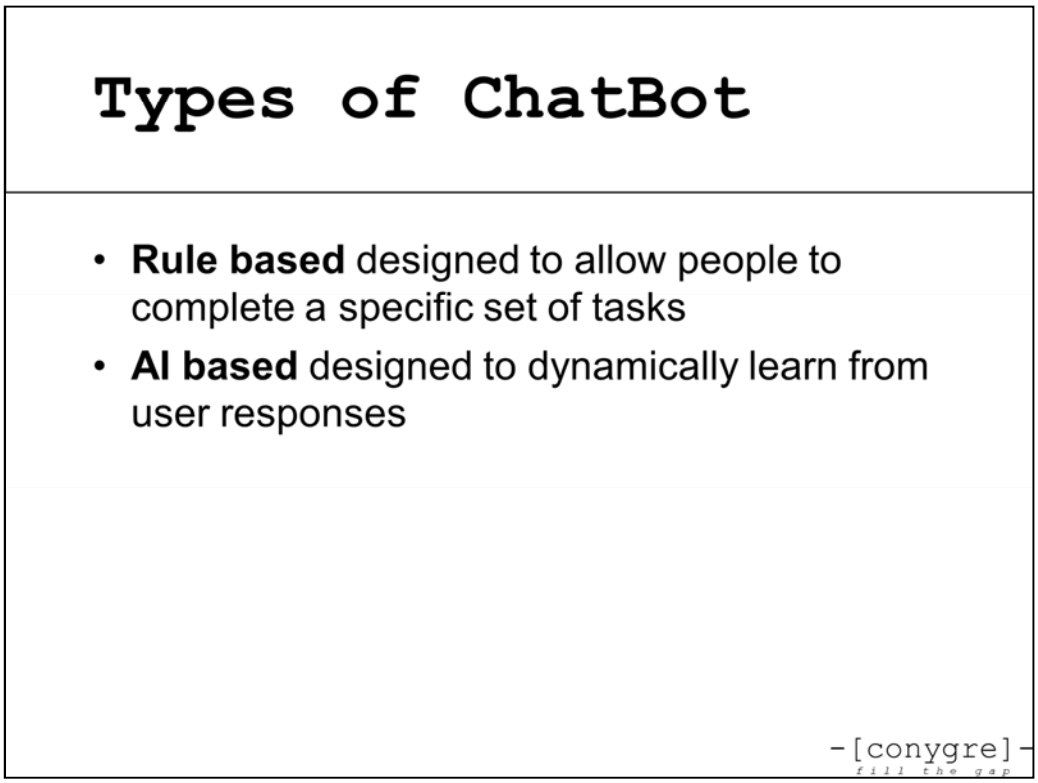
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



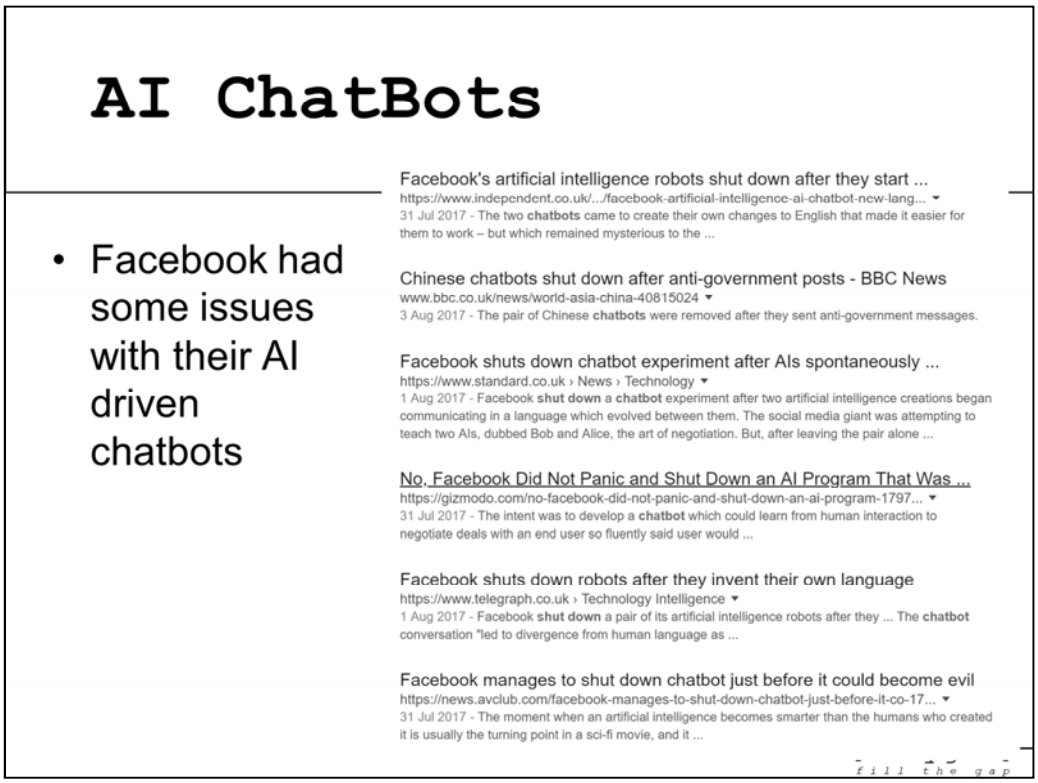
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Modern AI Chatbots

- Modern AI Chatbots are available such as
  - DialogFlow
  - GupShup
  - Amazon Lex

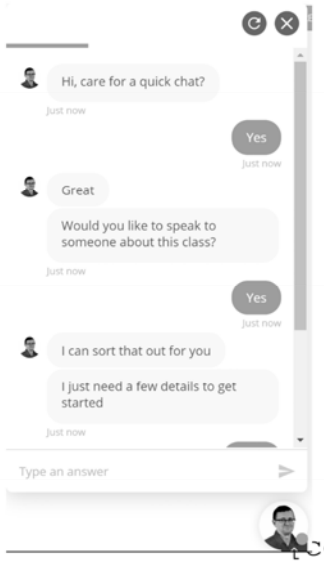
-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Rules Based ChatBots

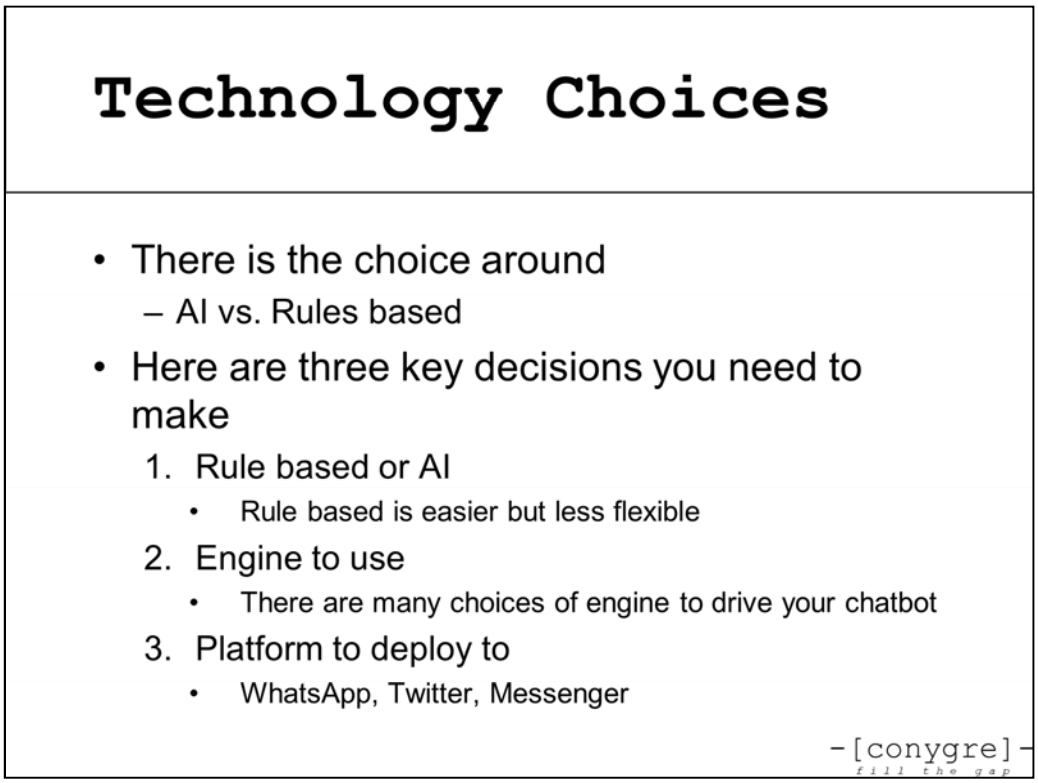
- Rules based chatbots are only as clever as the rules you create
  - Collect.chat
- These are good for simple task oriented bots



[conygre]
fill the gap

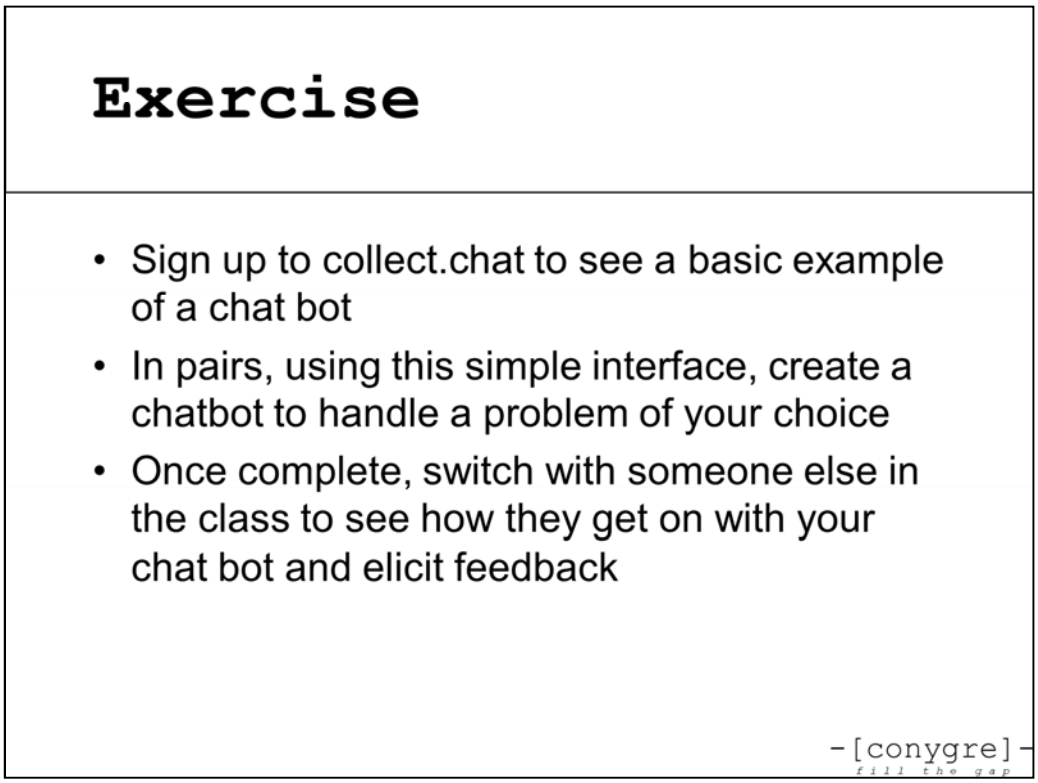
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



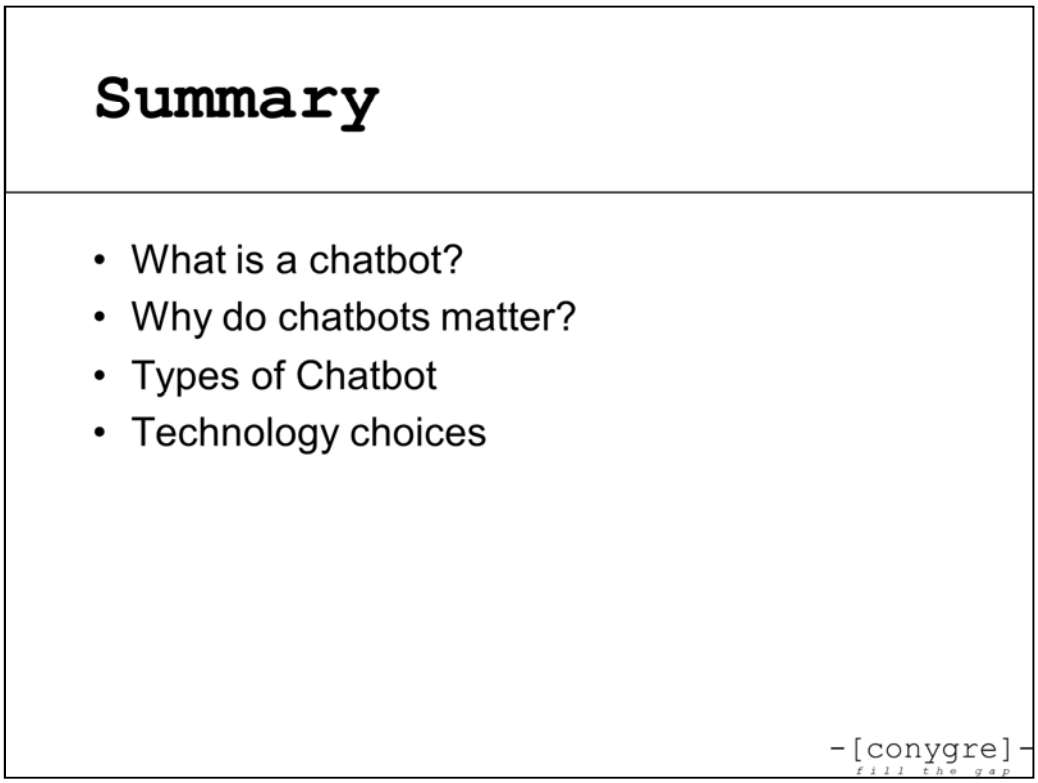
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

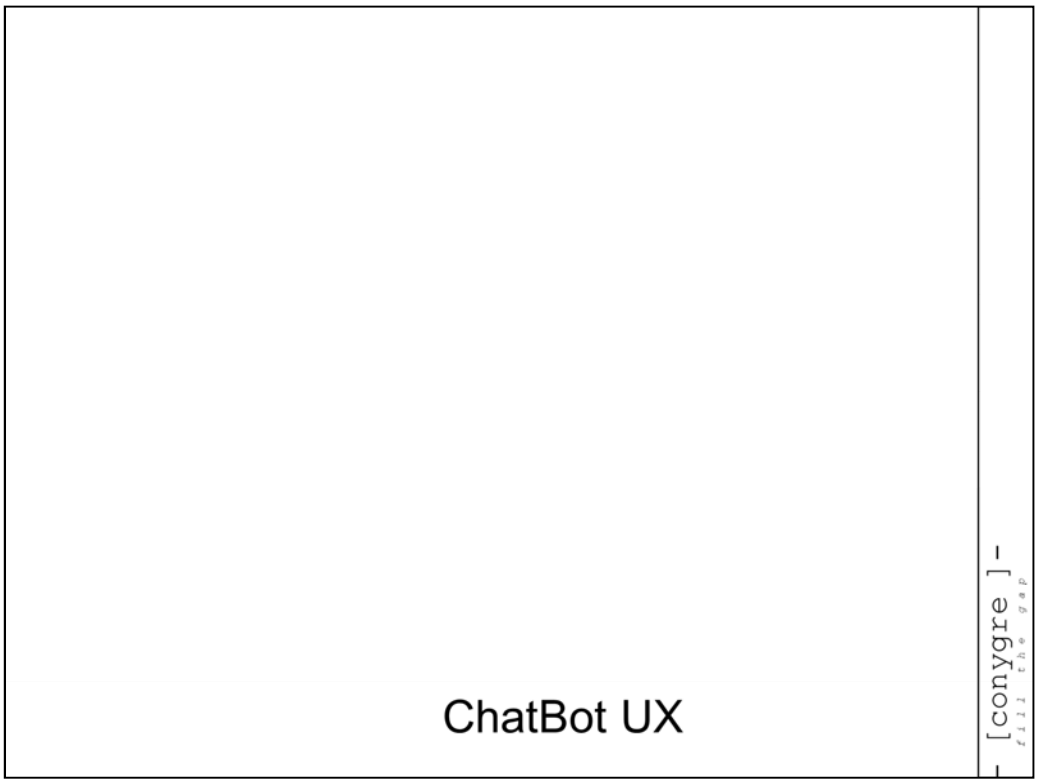
Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

ChatBot UX



© Conygre IT Ltd. Jun-18
Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Objectives

- Review a number of best practices identified for the creation of chatbots
  - *Source: https://chatbotsmagazine.com/19-best-practices-for-building-chatbots-3c46274501b2*

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

#1 Expectation Engineering

- Don’t give people the impression that this is a person
- Don’t give people the impression that it is a chat bot with sophisticated capabilities if it doesn’t have them

Hi, my name is Nick and I am sat by my computer in case you have any questions

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

#2 Provide a Way Back

- Create a mechanism for users to return to the initial options
  - *How do I get back to the beginning?*

Thanks for entering all that information. Please submit your order. I assume you don't want to change anything.

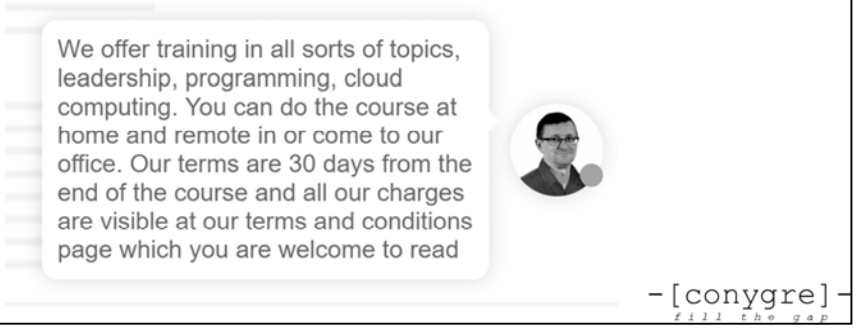
-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

#3 Avoid tl;dr

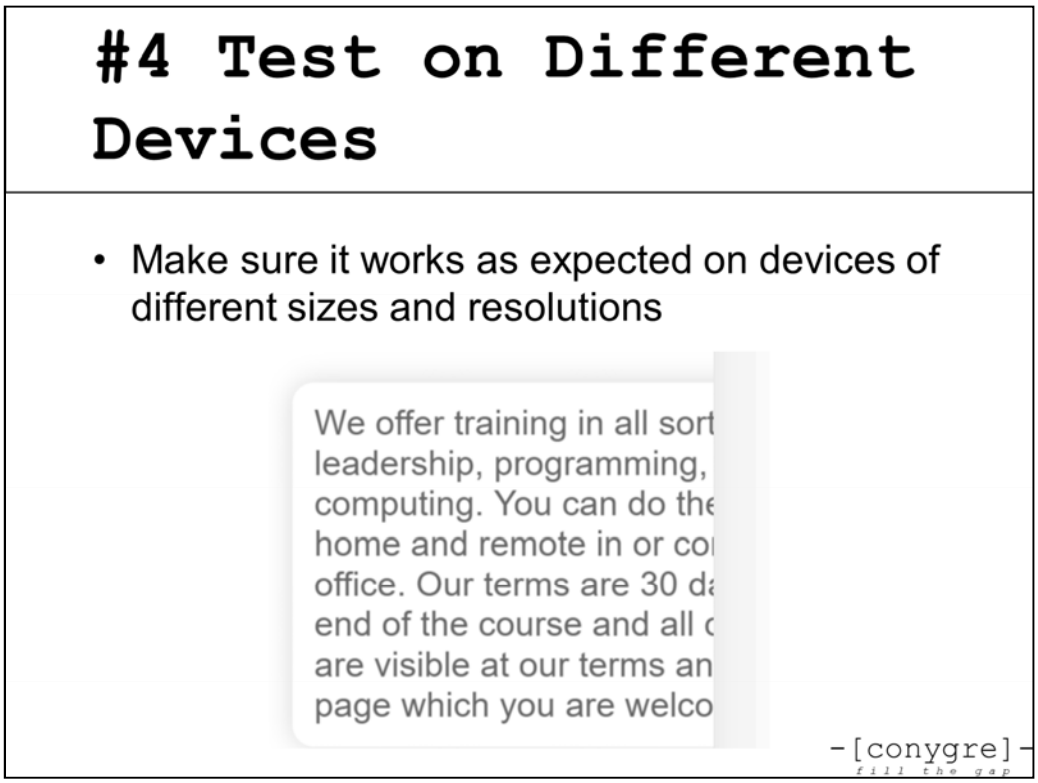
- AVOID Too long, didn't read
- Big scrolling messages are a turn off to most people
  - They just give up



-[conygre]-
fill the gap

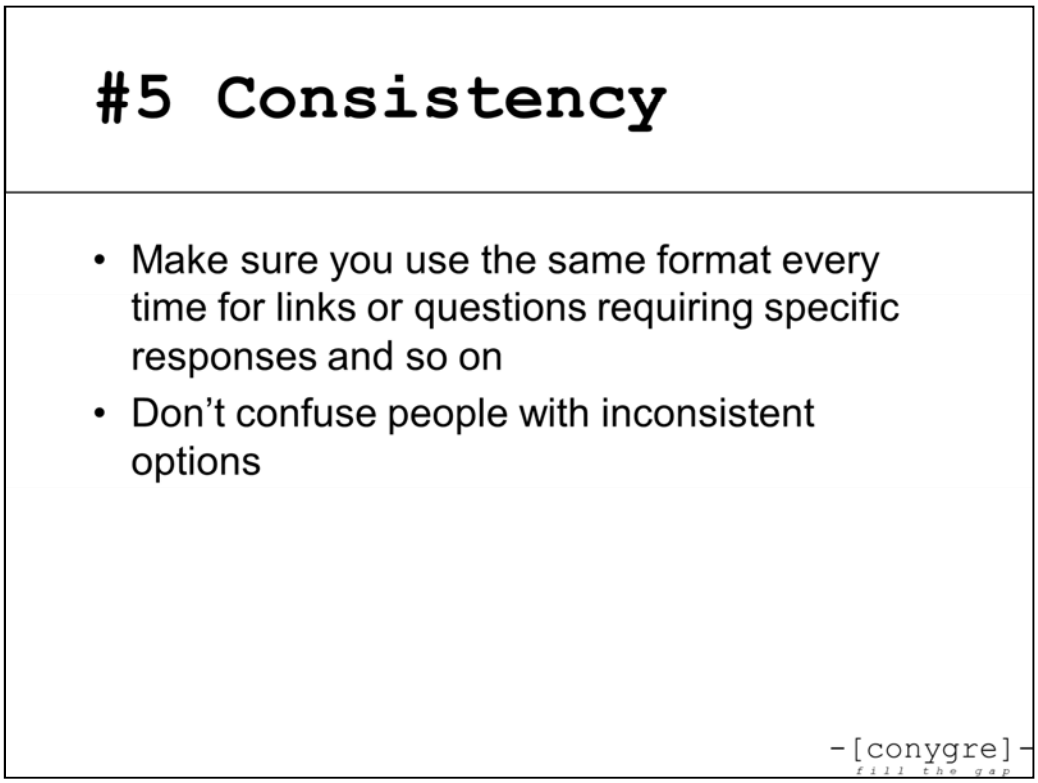
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

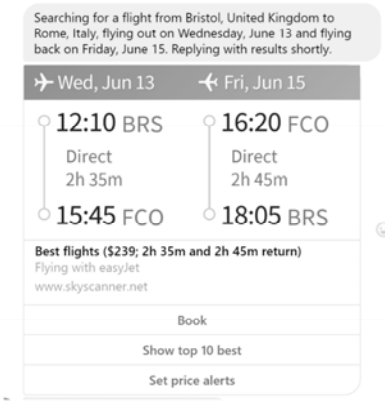


© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

#6 Redirect with Web Views

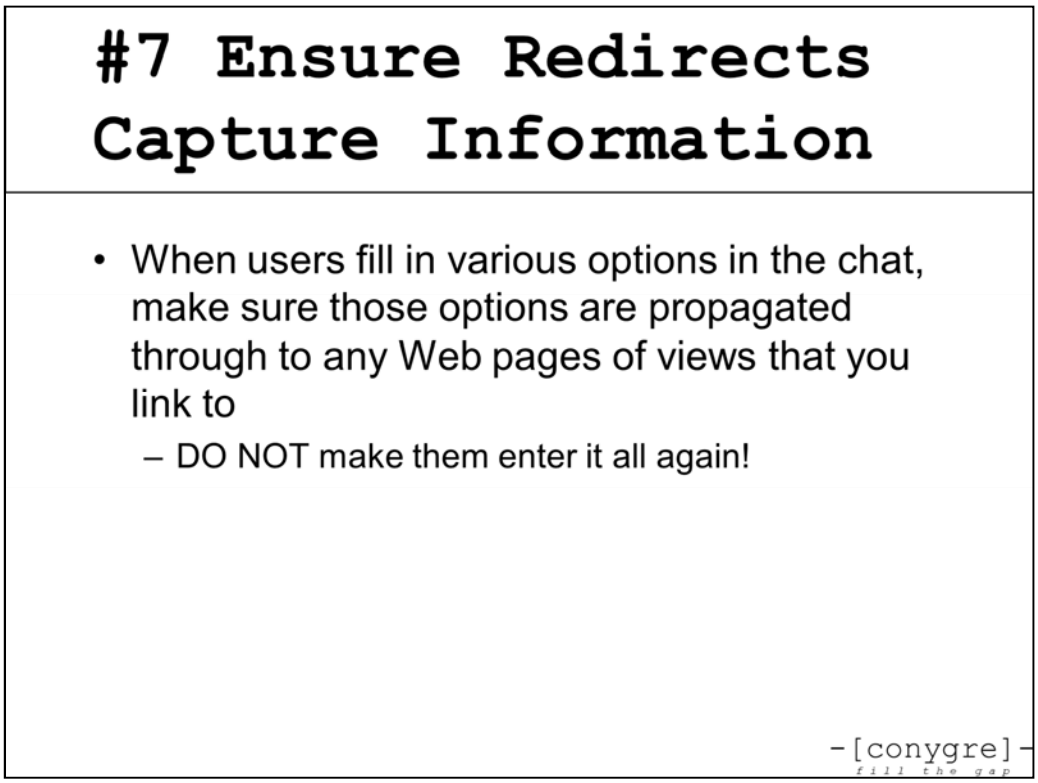
- Don’t just link out to other web sites, but rather display Web views within the chat itself



-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

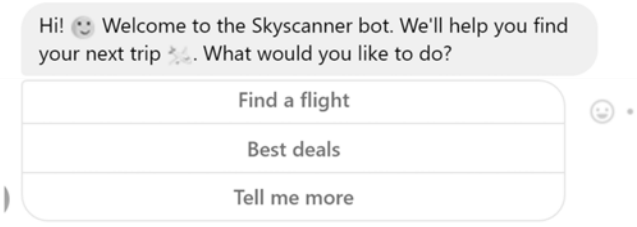


© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

#8 Provide Guard Rails

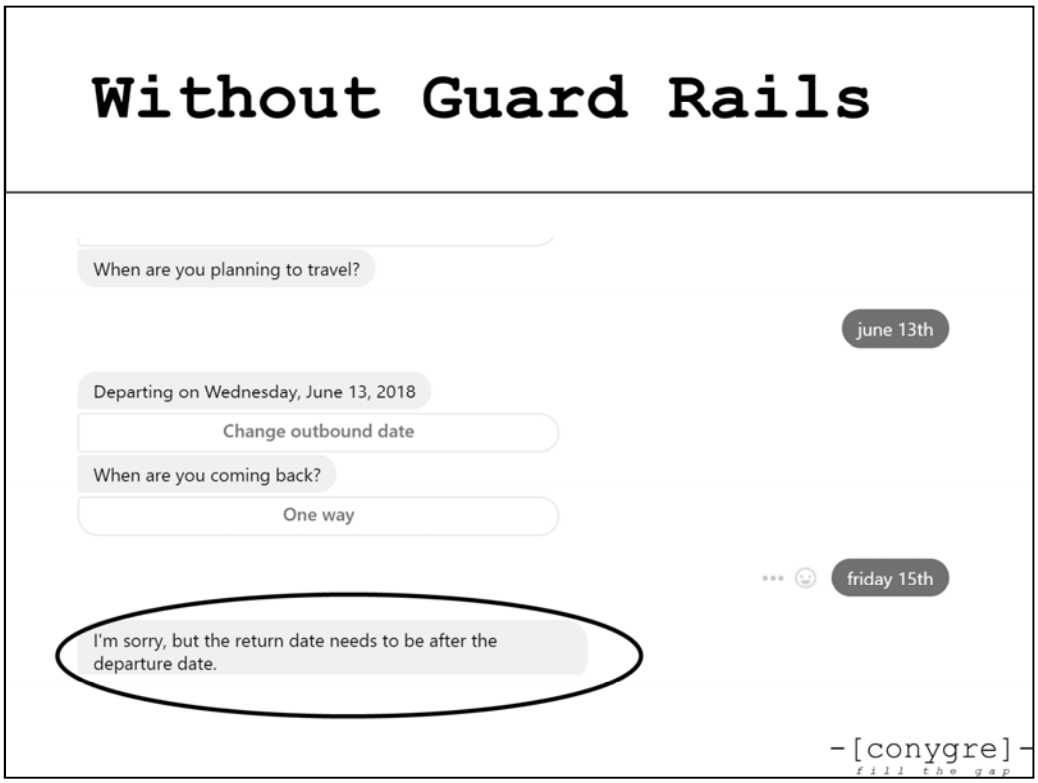
- Providing specific options can make it easier for people and keeps them on track



-[conygre]-
fill the gap

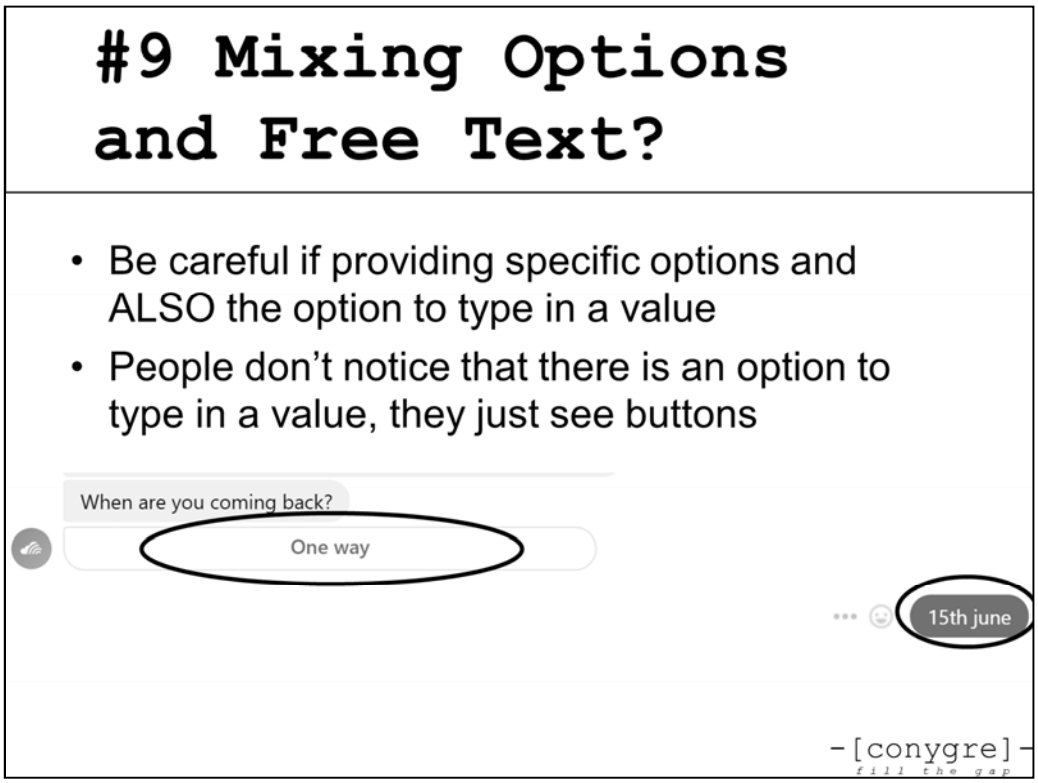
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



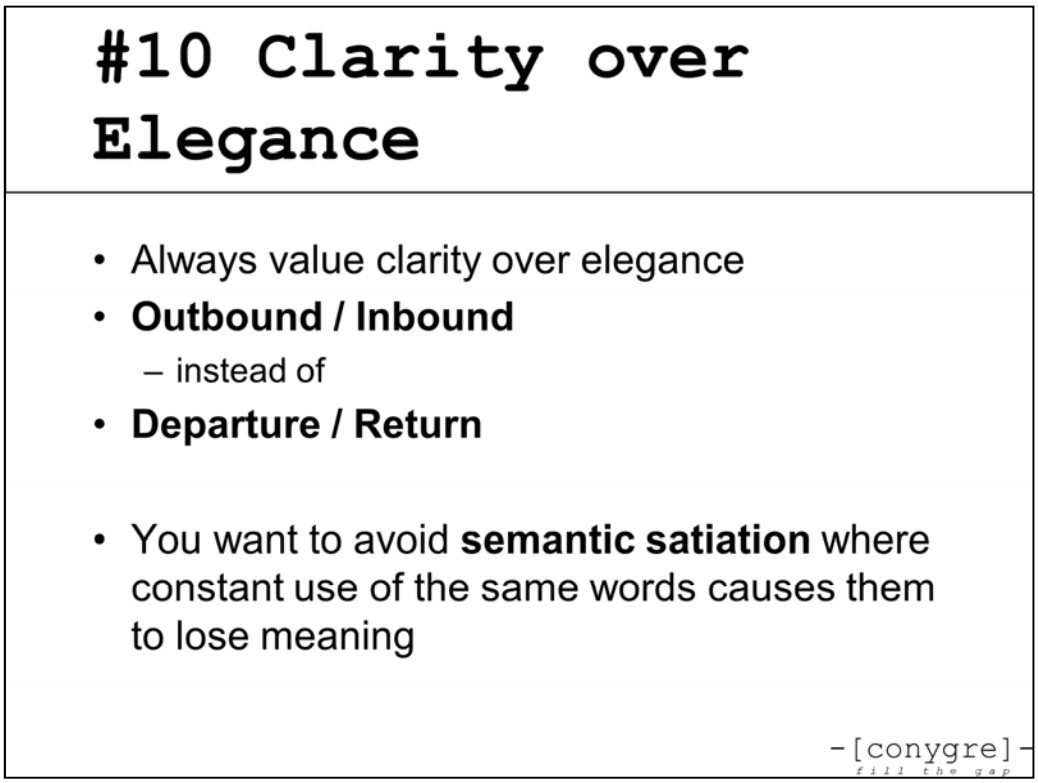
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

#11 Confirm by Asking not Stating

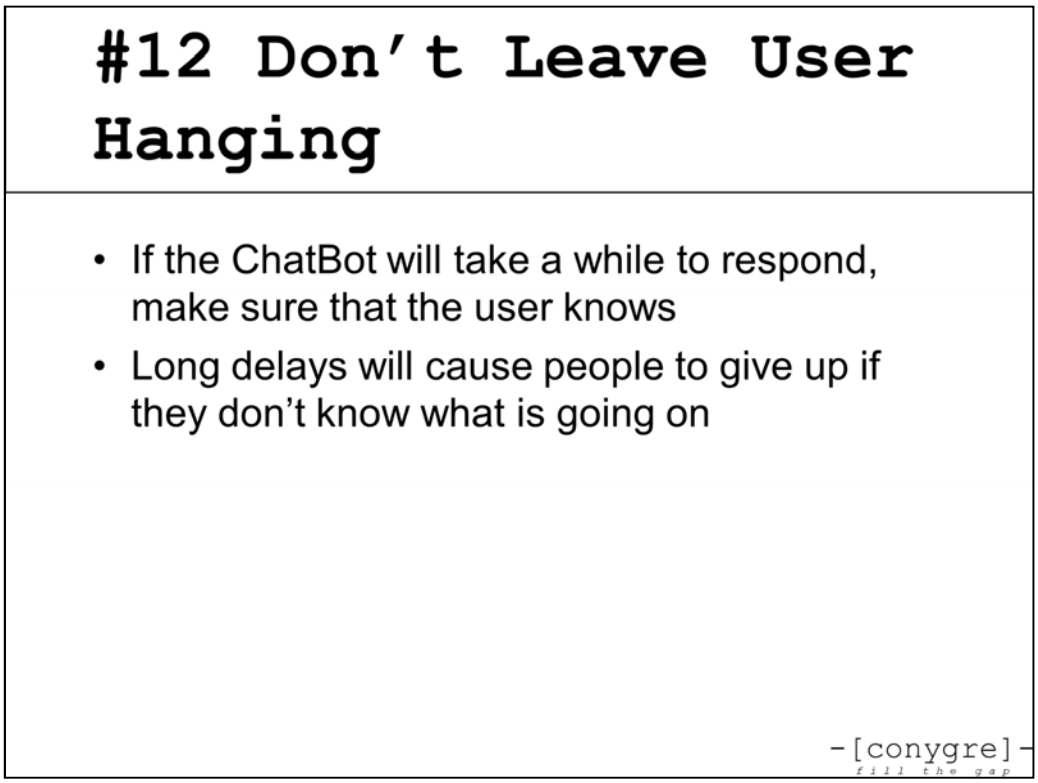
- Ensure that any confirmations are asked for not simply stated
  - "You ordered flowers for Wednesday"

  vs
  - "Is it correct you want flowers on Wednesday?"
- Stating can make the user feel like they are being corrected if it is wrong

fill the gap

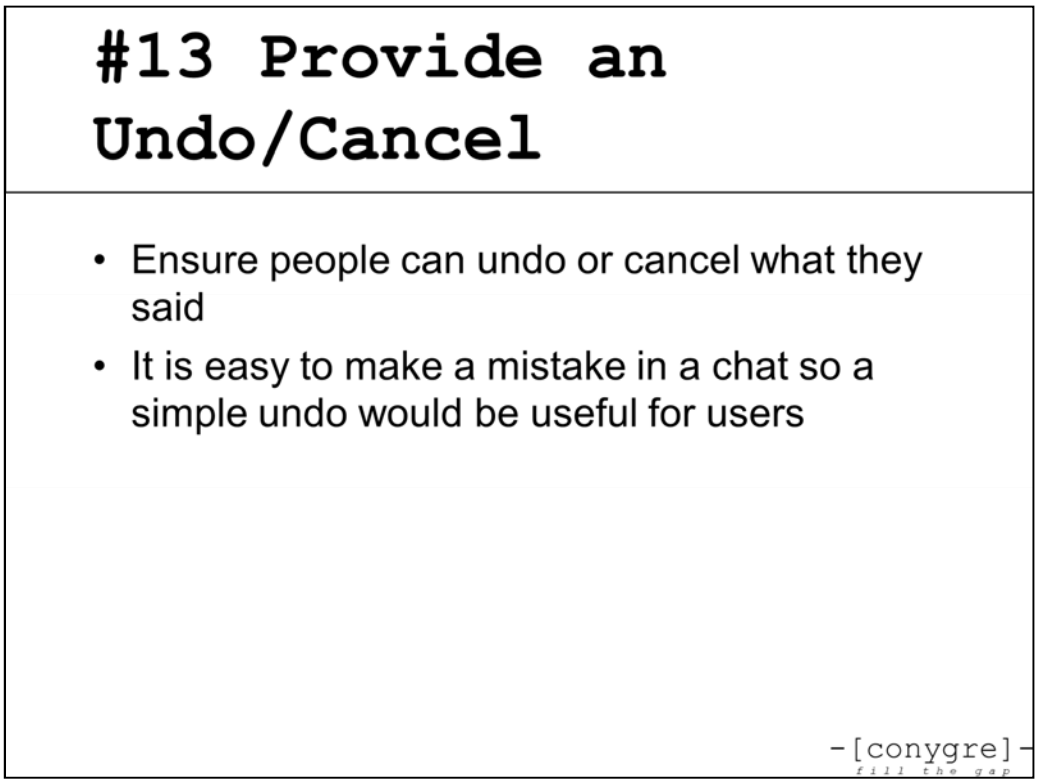
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

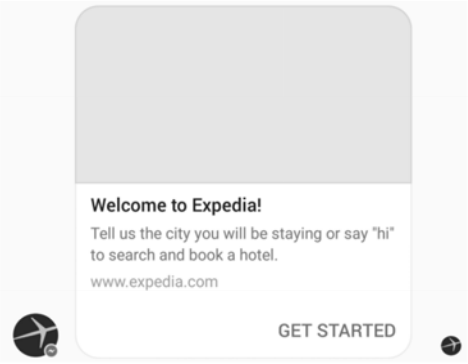


© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

#14 Beware of Captions

- The caption here contains the actual instructions!
  - You could be forgiven for missing it

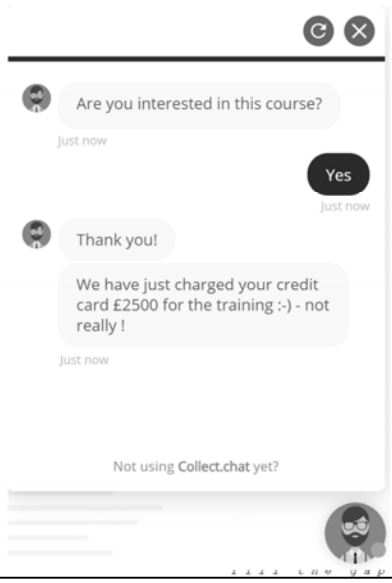


-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18
Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

#15 Be Careful with Humour

- Make sure your humour is not too off putting

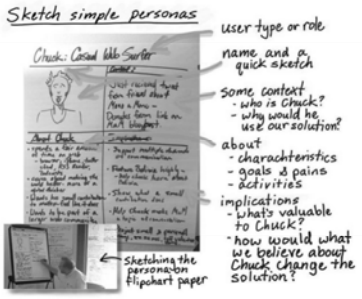


© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

#16 Consider a Voice

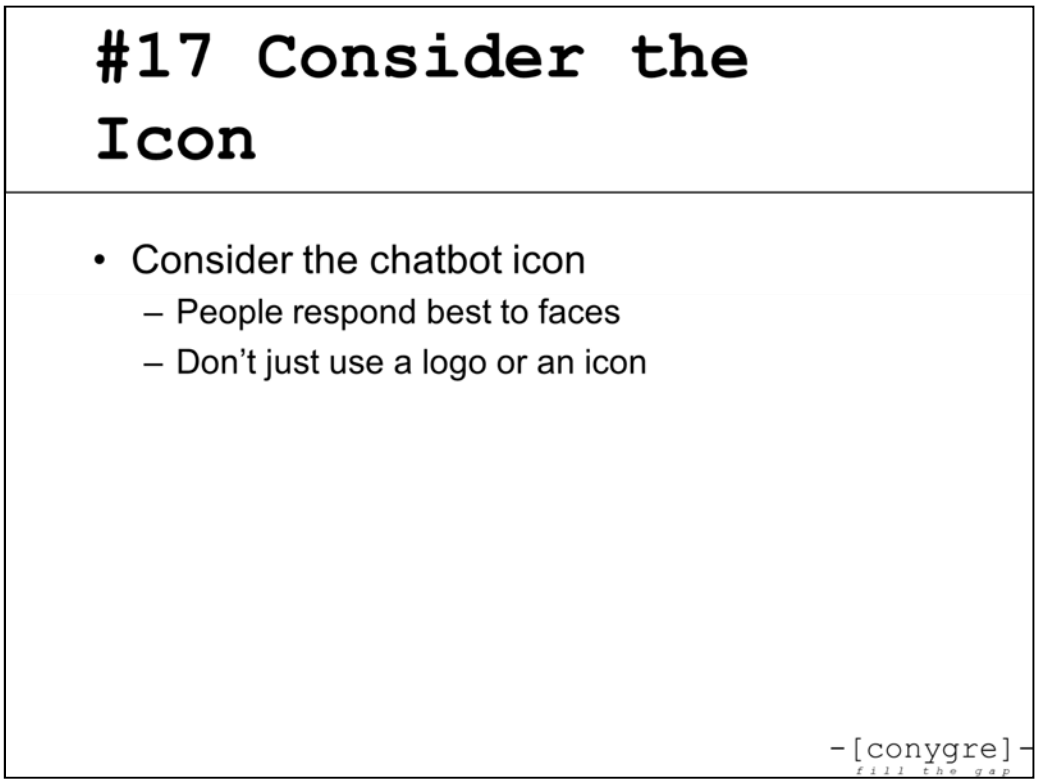
- It can be helpful to create a persona for your chatbot
- It allows you to think of your chatbot as a person and therefore how they might respond to the user
- It can make it more believable and engaging for the audience



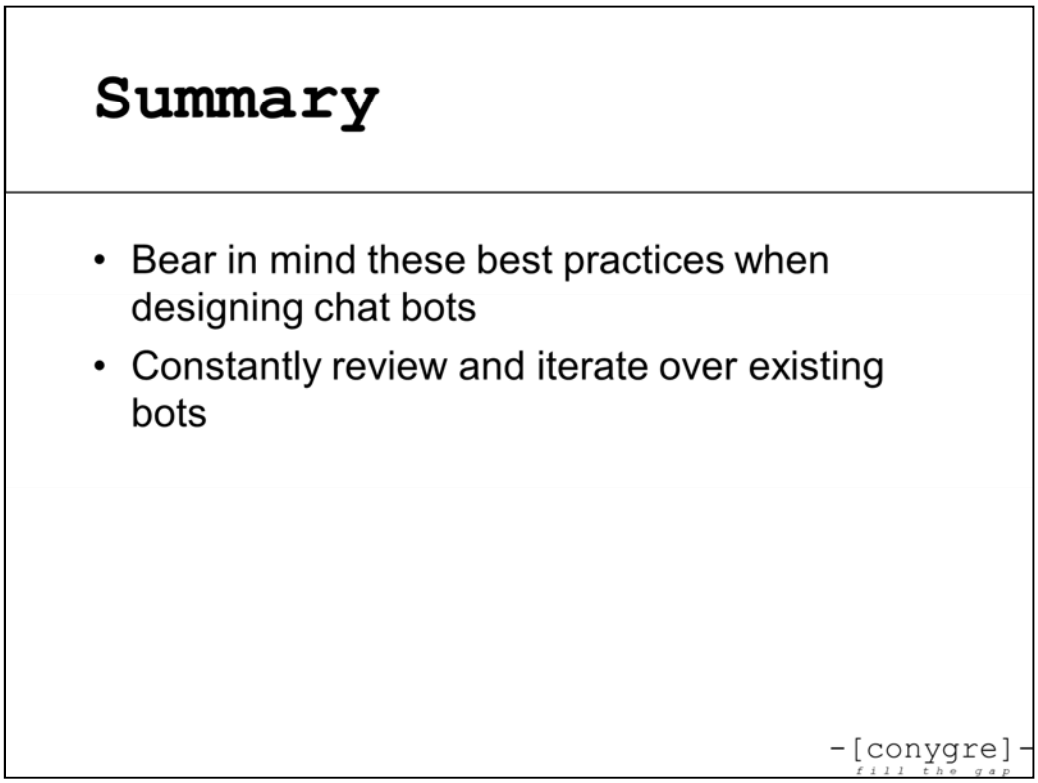
-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

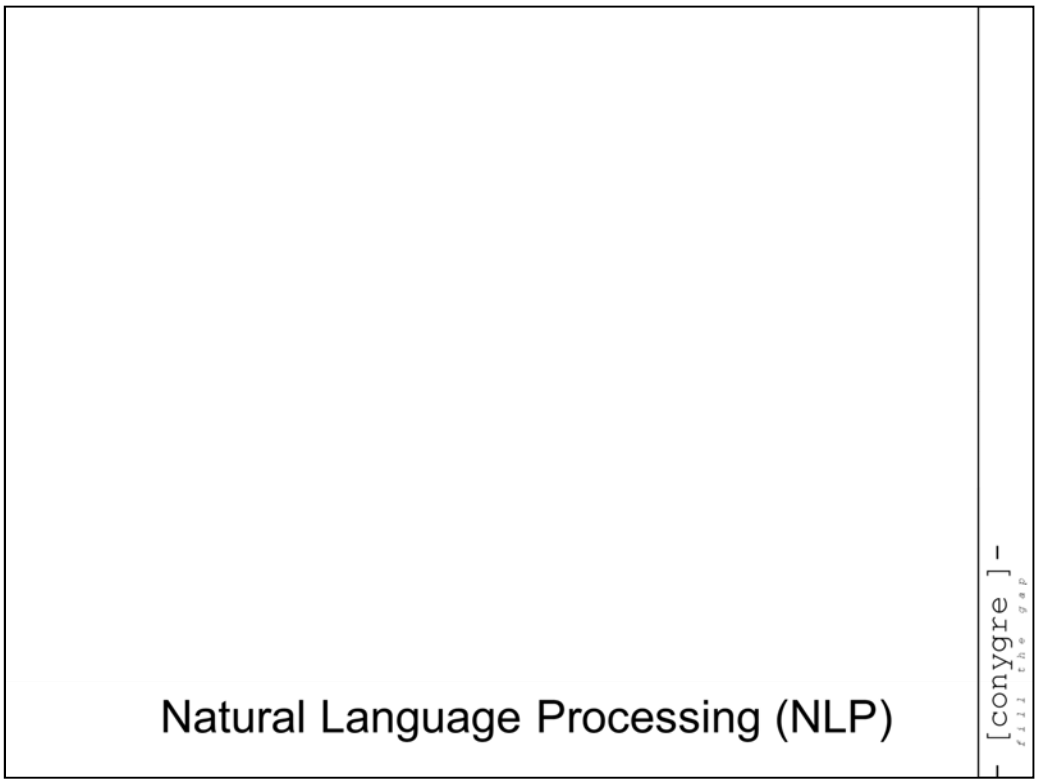


© Conygre IT Ltd. Jun-18
Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

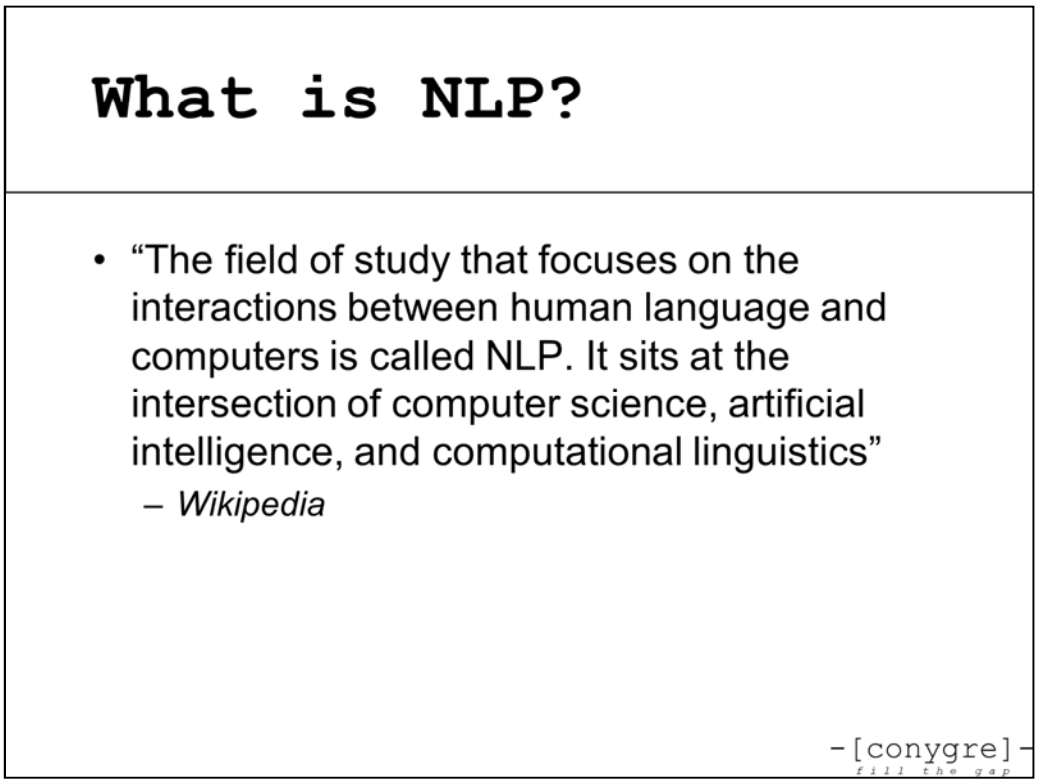
Objectives

- What is NLP?
- What can you do with NLP?
- Overview of Natural Language Processing
- Worked Python example of Sentiment analysis
- NLP and Chatbots

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

What Can You Do With NLP?

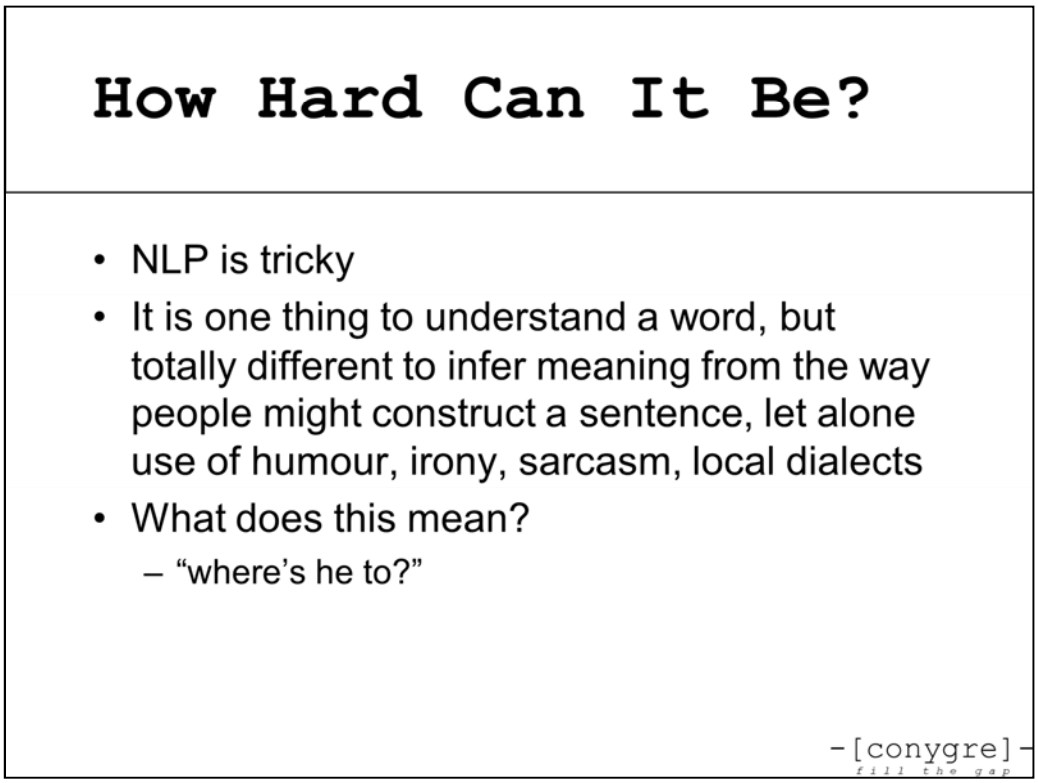
- Automate summarization
- Translation
- Entity recognition
- Sentiment analysis
- Topic segmentation

- All of these could be useful in a chatbot!

-[conygre]-
fill the gap

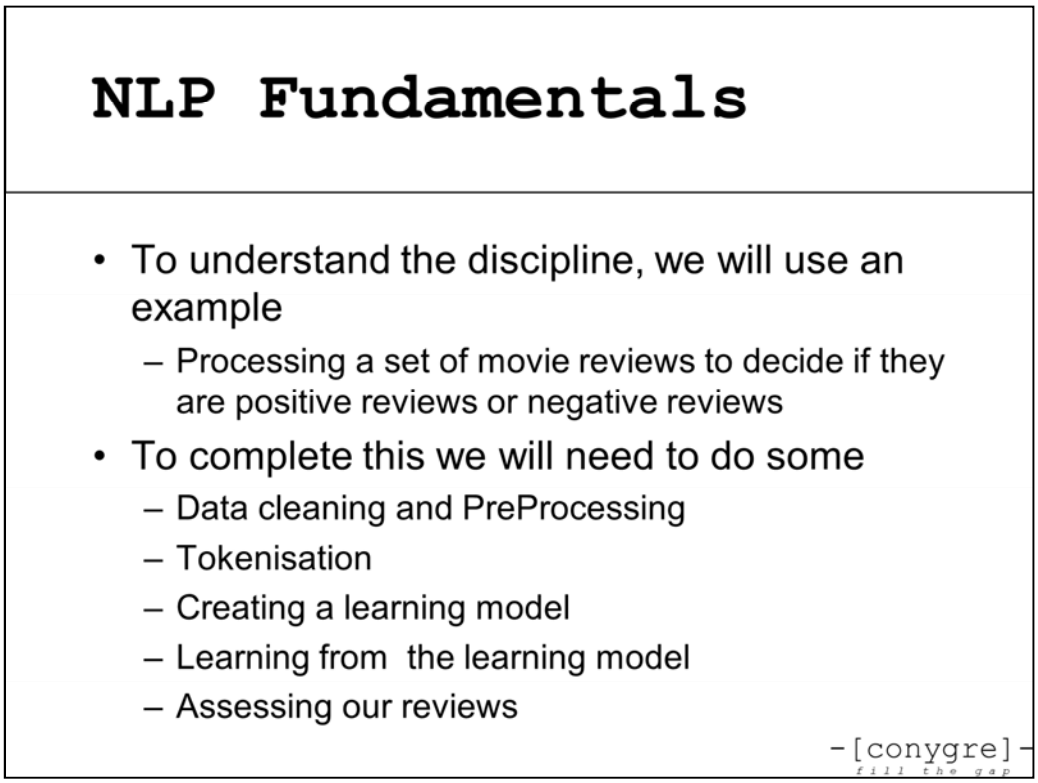
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Our DataSet

- Our dataset is a set of reviews which appear as follows in a TSV file consisting of a review id and a review

id	review
12311_10	Naturally, a film who's main …
8348_2	This movie is a disaster within a disaster film …

- Are these reviews positive or negative?

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Our Learning Model

- In order to complete sentiment analysis, we will need some data for our **machine** to **learn** from

id	sentiment	review
5814_8	1	With all this stuff going down at the …
2381_9	0	The Classic War of the Worlds by …

- These reviews have already been marked as either
  - positive 1
  - negative 0

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Loading the Data

- We can load the data sets using the Python library **pandas**

```
train = pd.read_csv(os.path.join(os.path.dirname(__file__), 'data', 'labeledTrainData.tsv'),
header=0, \
            delimiter="\t", quoting=3)
test = pd.read_csv(os.path.join(os.path.dirname(__file__), 'data', 'testData.tsv'), header=0,
delimiter="\t", \
            quoting=3 )
```

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Data Cleansing

- Before you can process text you might have to tidy it up
  - Fix any spelling errors
  - Remove HTML markup (if processing web content)
- When using Python
  - **BeautifulSoup** for HTML removal
  - SpellChecker
    - **https://github.com/pirate/spellchecker** based on work by Peter Norvig

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Splitting into Words

- Once we have cleaned up the text, we can then split the text up into units
- A simple splitting is to
  - Separate by the space characters into words
  - Remove any non word characters
  - Change all text to lower case
- The act of splitting is called **tokenisation**

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Tokenization in Python

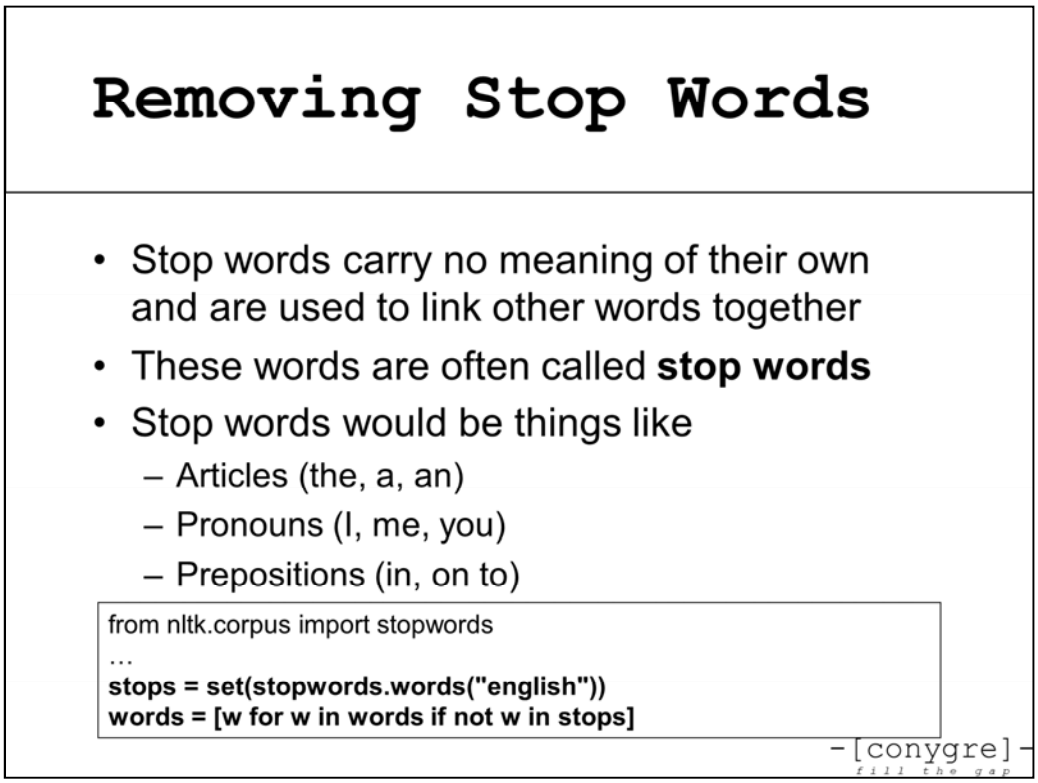
- To complete the tasks in Python

```
# remove the HTML
review_text = BeautifulSoup(review, "html.parser").get_text()
# remove the non-letters
review_text = re.sub("[^a-zA-Z]"," ", review_text)
#convert words to lower case and split
words = review_text.lower().split()
```

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Complete Cleaning Example

- Below is a Python function to return fully cleansed list of words

```
def review_to_wordlist( review, remove_stopwords=False ):
    review_text = BeautifulSoup(review, "html.parser").get_text()
    words = review_text.lower().split()
    if remove_stopwords:
        stops = set(stopwords.words("english"))
        words = [w for w in words if not w in stops]
    return(words)
```

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Create a Bag of Words

- So far we have one long (cleansed) list of all the words in the review
- Now we need to find out how many times these words are appearing in the reviews
- This list of words along with the frequencies is called a **bag of words**

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Simple Bag of Words Example

- Consider the following short reviews
  - the movie was really bad
  - the movie was quite bad in places
- The bag of words for the first one would be
  - {the, movie, was, really, bad, quite, in, places}
  - {1,1,1,1,1,0,0,0}
- The bag of words for the second one would be
  - {the, movie, was, really, bad, quite, in, places}
  - {1,1,1,0,1,1,1,1}

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Using scikit-learn

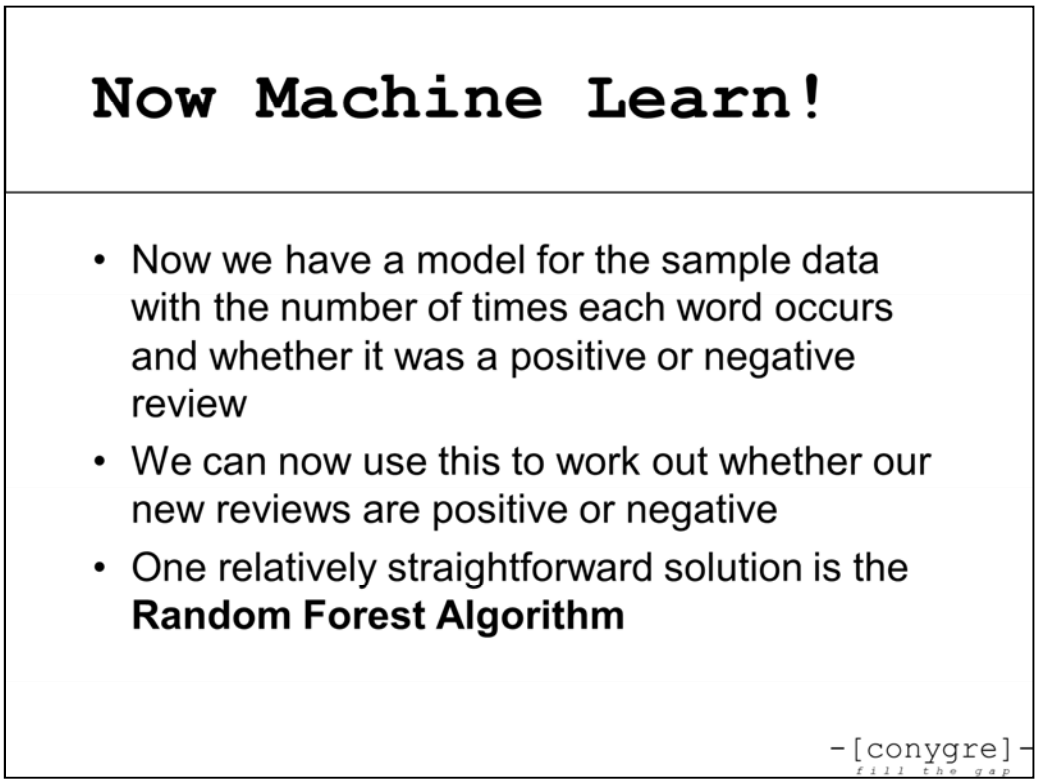
- The scikit-learn library can create a bag of words for each review
- There are loads of words in the reviews so we will limit it to the top 5000
  - Note the **CountVectorizer** could have been used to tokenize for us

```
vectorizer = CountVectorizer(analyzer = "word",   \
                             tokenizer = None,    \
                             preprocessor = None, \
                             stop_words = None,   \
                             max_features = 5000)
 # create the bag of words here
 train_data_features = vectorizer.fit_transform(clean_train_reviews)
 # convert it into a numpy array for convenience
 np.asarray(train_data_features)
```

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

The Random Forest Algorithm

- The random forest will look at all the sets of words in the data we need to look at and then apply this algorithm to see if they are positive or negative
- For information on the algorithm, see
  - https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Applying the Algorithm

- To apply the algorithm using Python, you can use the **sklearn** package

```
forest = RandomForestClassifier(n_estimators = 100) # 100 trees
# Fit the forest to the training set, using the bag of words as
# features and the sentiment labels as the response variable
forest = forest.fit( train_data_features, train["sentiment"] )
# Create an empty list and append the clean reviews one by one
clean_test_reviews = []
print ("Cleaning and parsing the test set movie reviews...\n")
for i in range(0,len(test["review"])):
    clean_test_reviews.append(" ".join(Word2VecUtility.review_to_wordlist(test["review"][i], True)))
# Get a bag of words for the test set, and convert to a numpy array
test_data_features = vectorizer.transform(clean_test_reviews)
np.asarray(test_data_features)
# Use the random forest to make sentiment label predictions
result = forest.predict(test_data_features)
```

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Create the Output

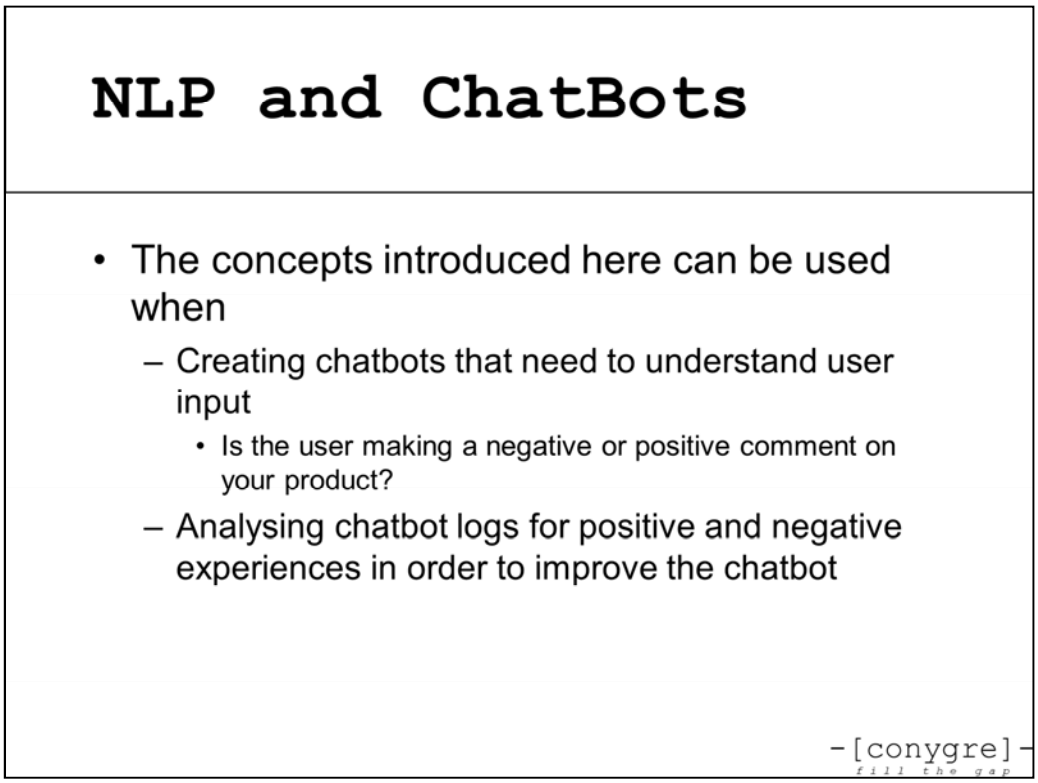
- Finally, using the panda library, an output file can be created with the predicted sentiment

```
output = pd.DataFrame( data={"id":test["id"], "sentiment":result} )
output.to_csv(os.path.join(os.path.dirname(__file__), 'data', 'Bag_of_Words_model.csv'),
index=False, quoting=3)
```

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Summary

- What is NLP?
- What can you do with NLP?
- Overview of Natural Language Processing
- Worked Python example of Sentiment analysis
- NLP and Chatbots

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Exercise

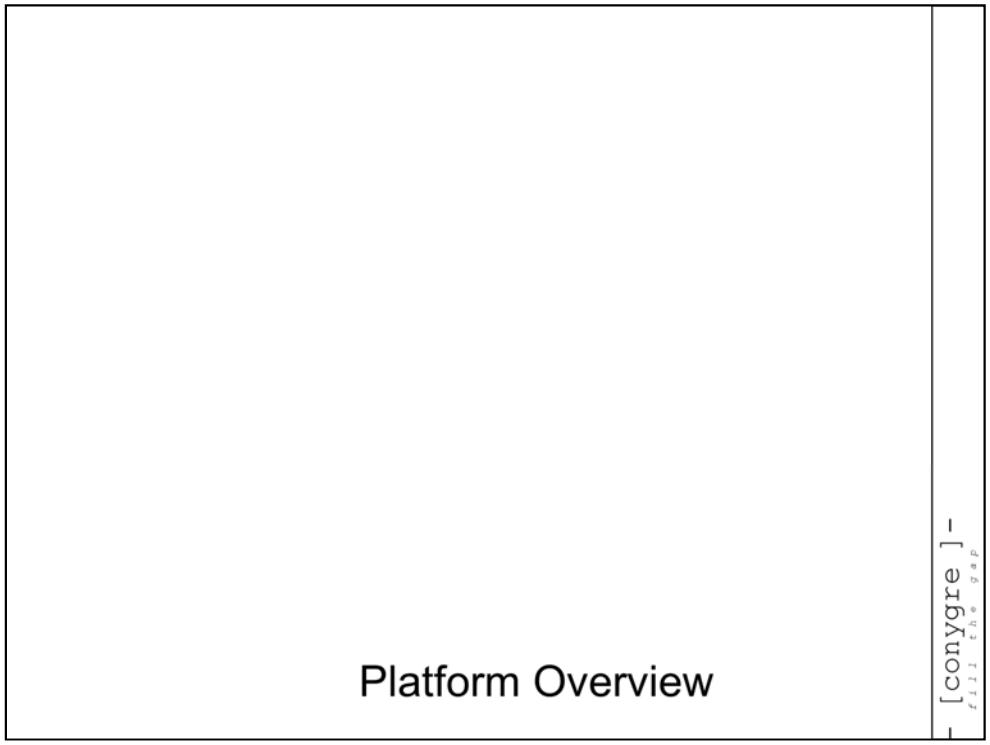
- Modify the sample application to work on some chatbot logs from chat responses that need to be followed up
  - flagged
  - not_flagged
- Flagging is based on seriousness of response to question:
  - "Describe a time when you have acted as a resource for someone else"
- You can split the sample data into two and use one for training and one for analysis
  - Since you have the actual correct response for each, you can see how good your algorithm is!

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

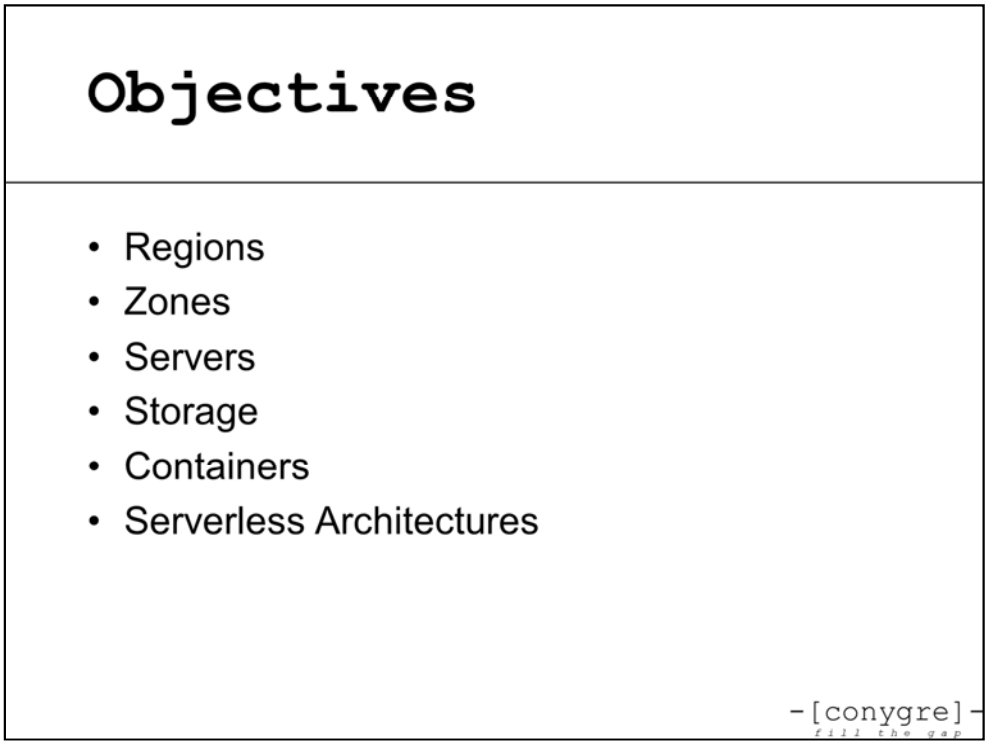
Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Platform Overview



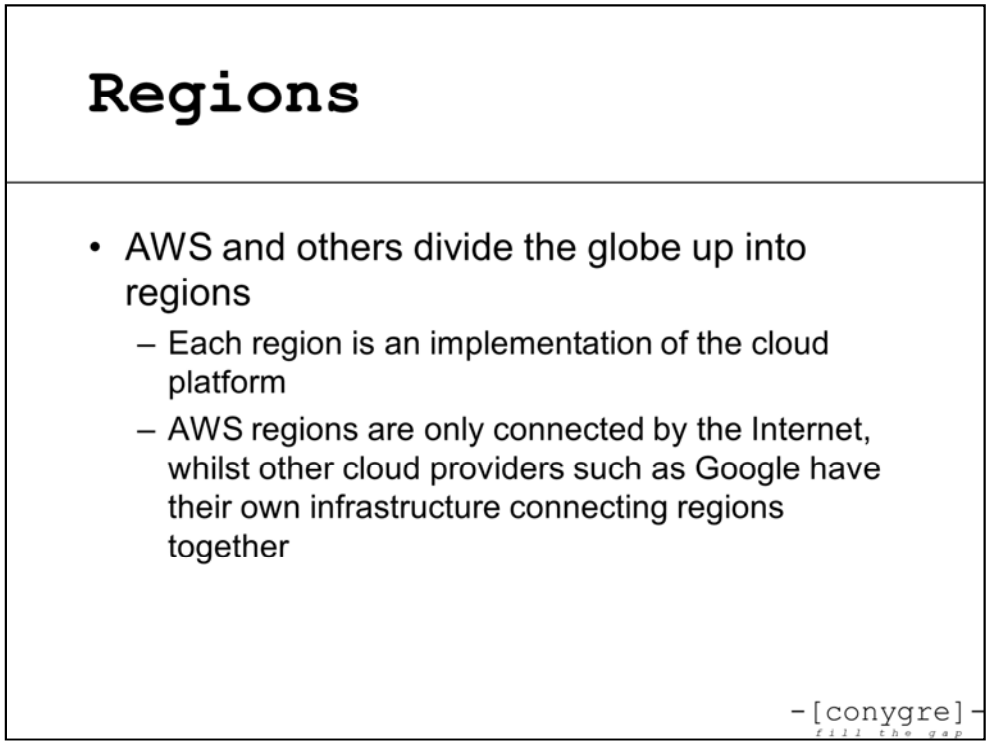
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



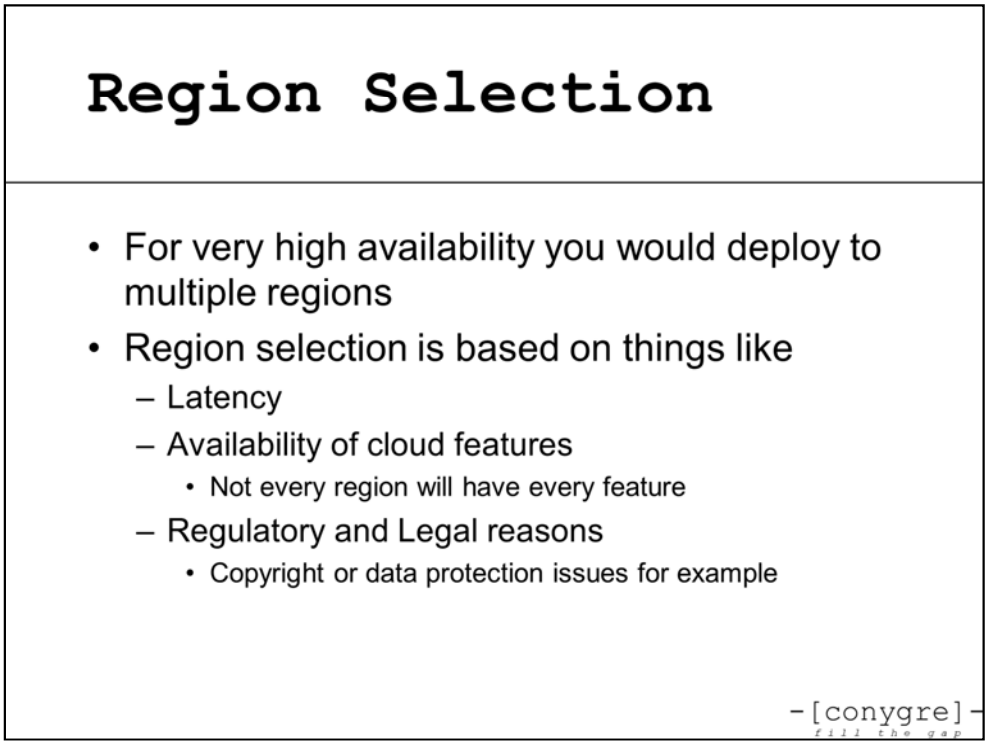
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



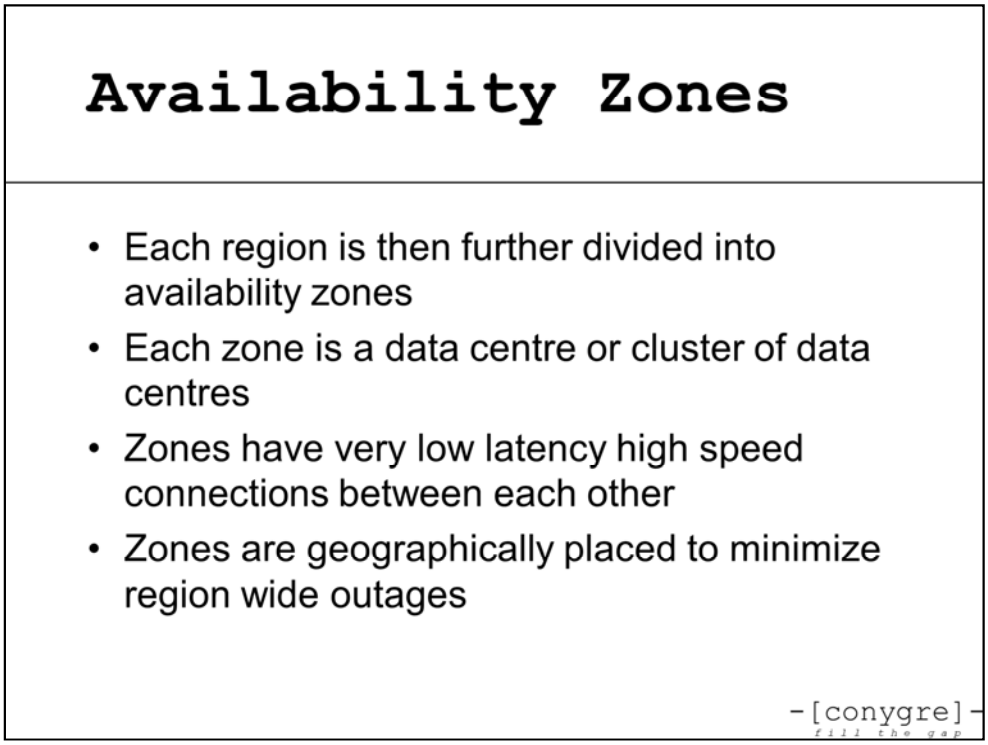
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



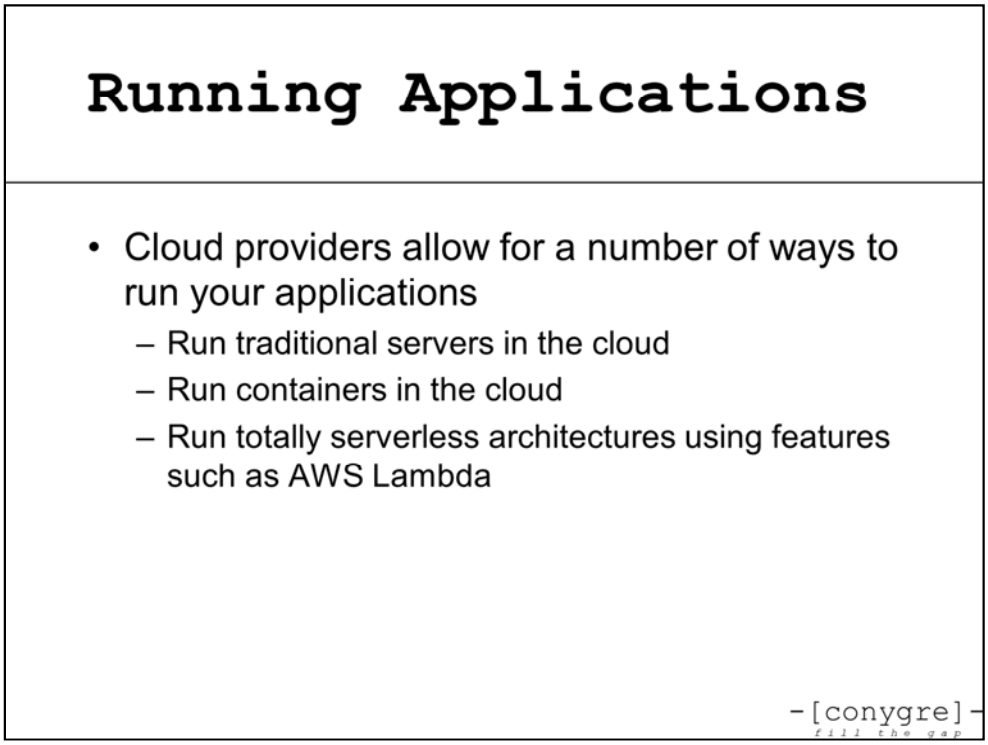
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



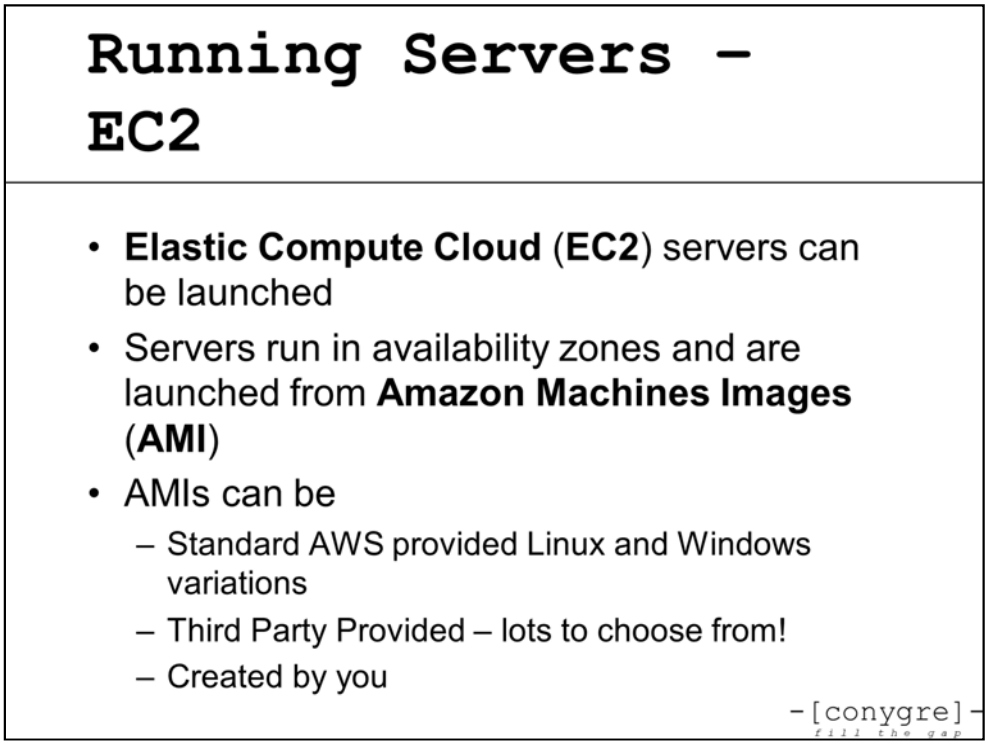
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



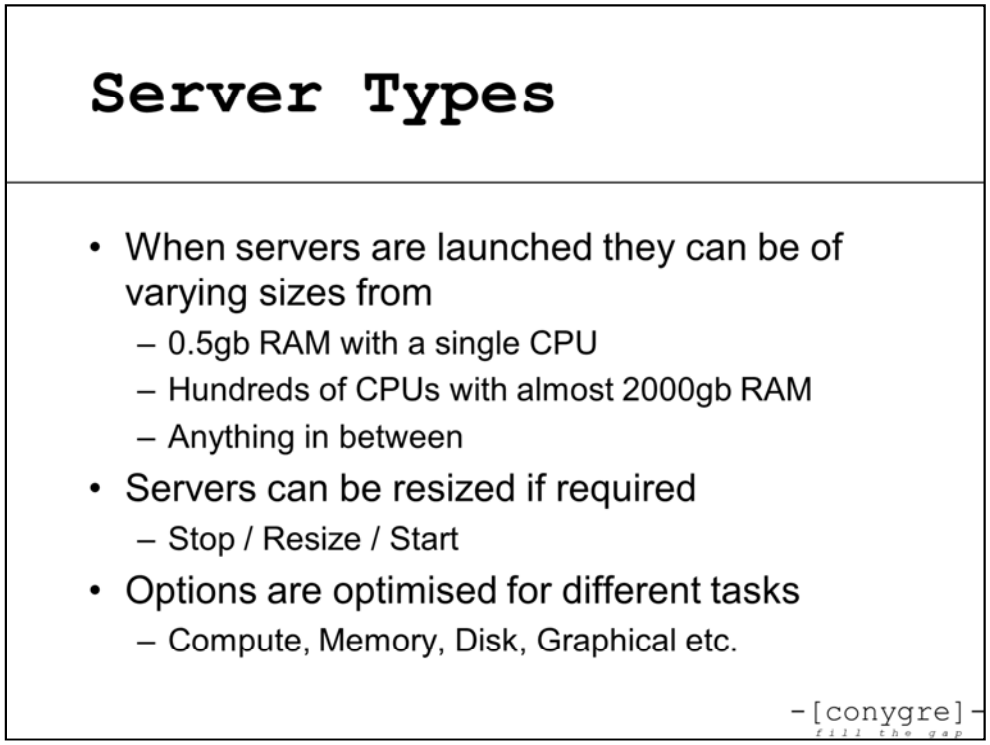
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



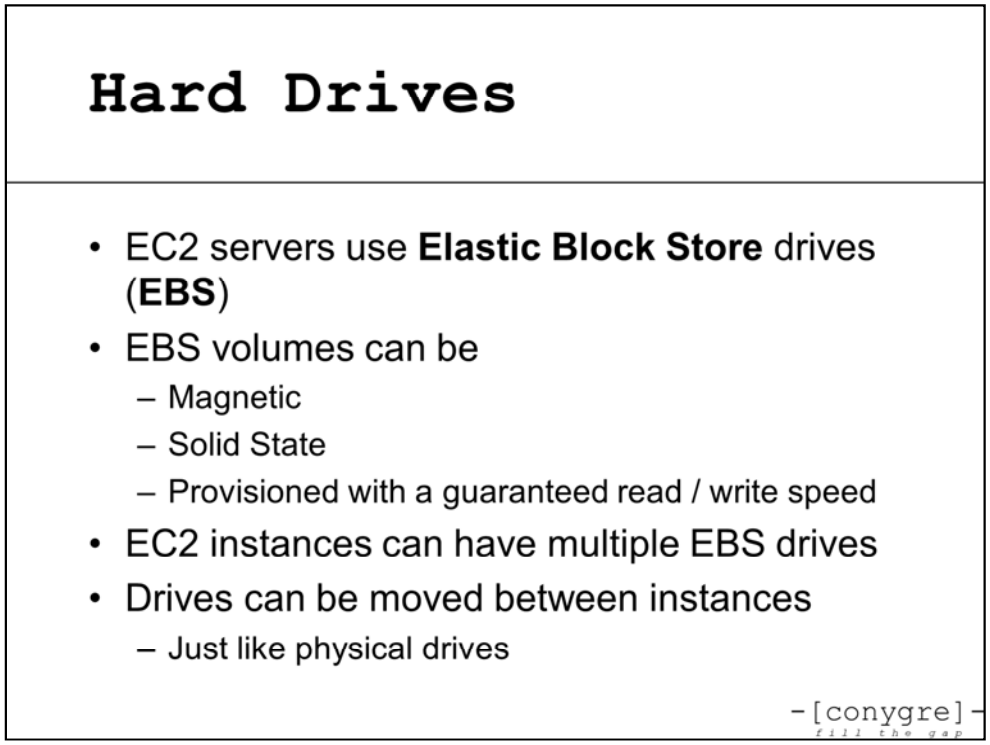
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



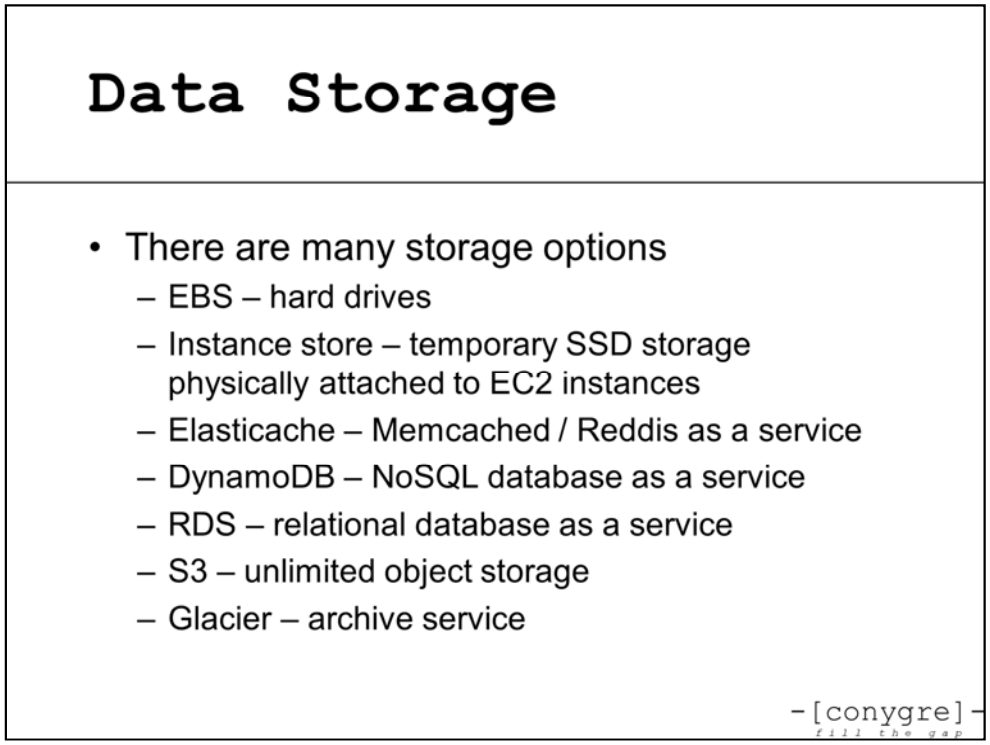
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



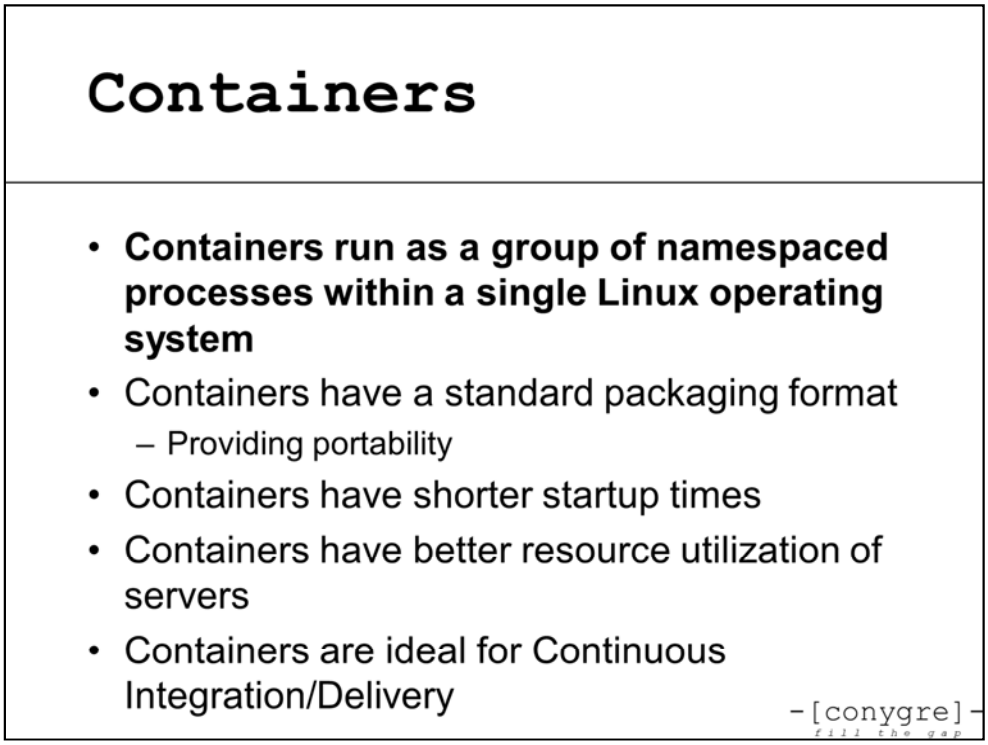
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



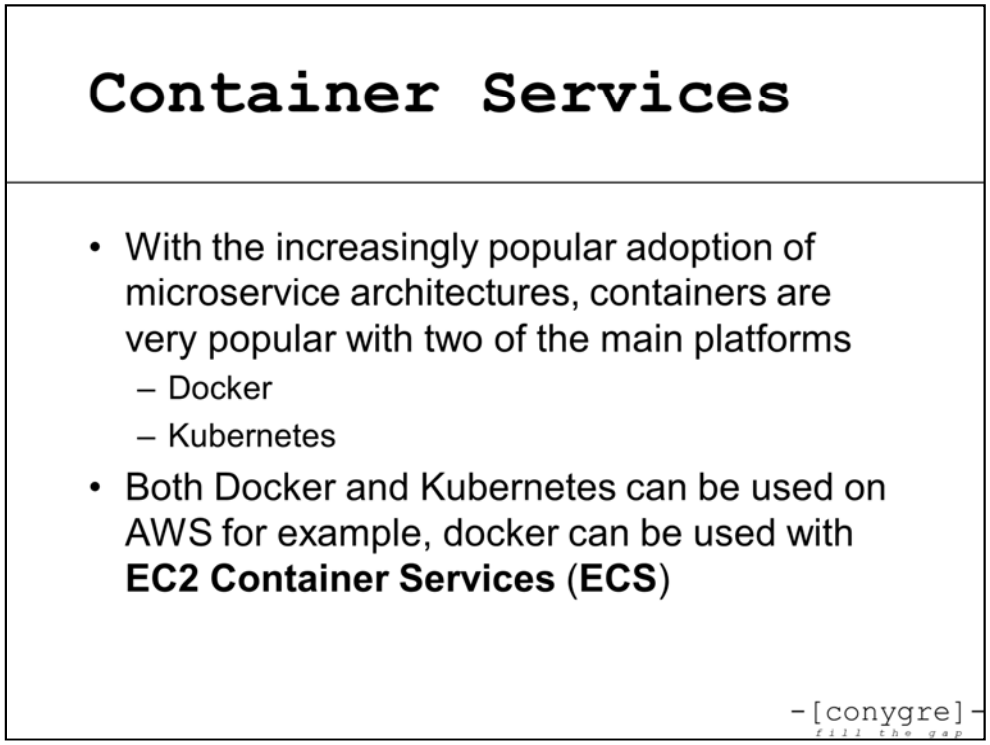
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



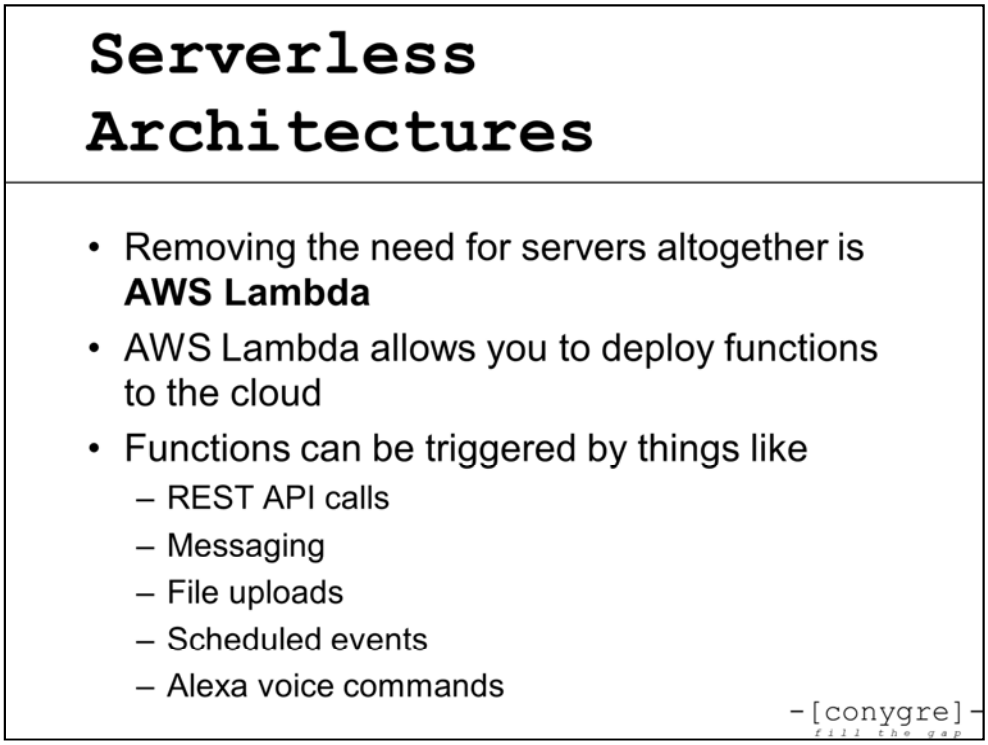
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



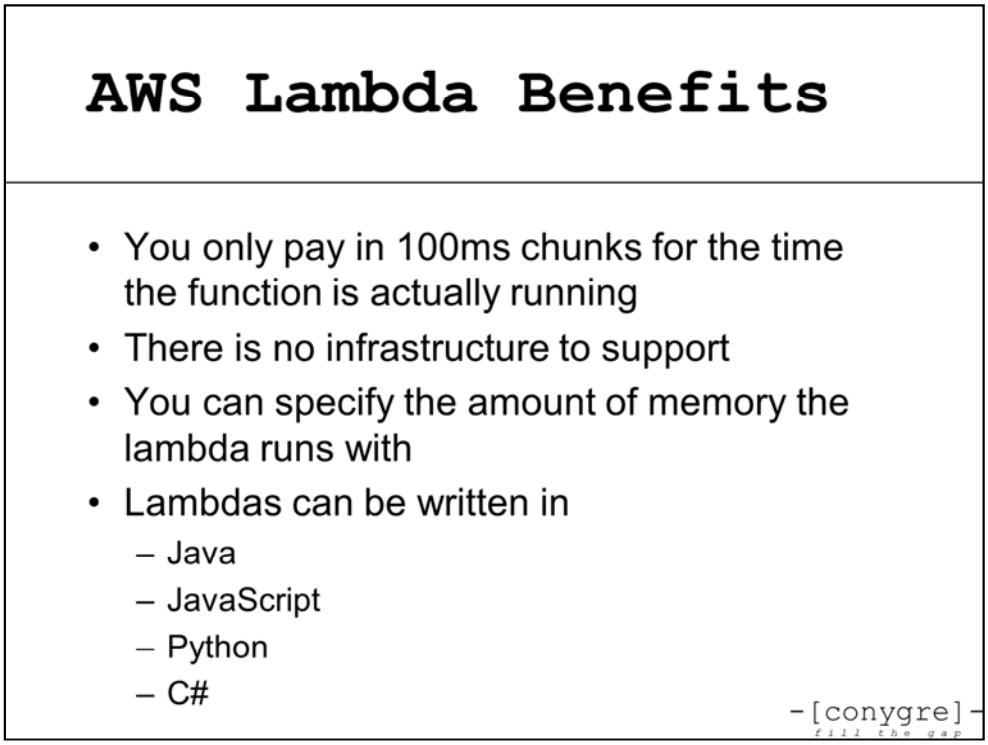
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



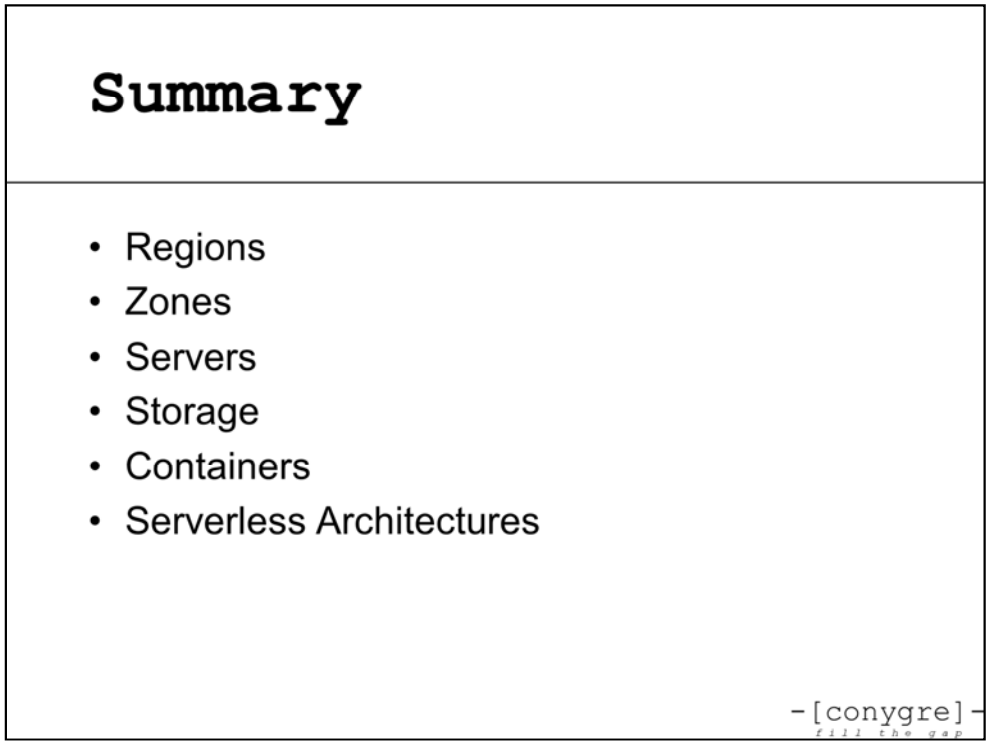
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



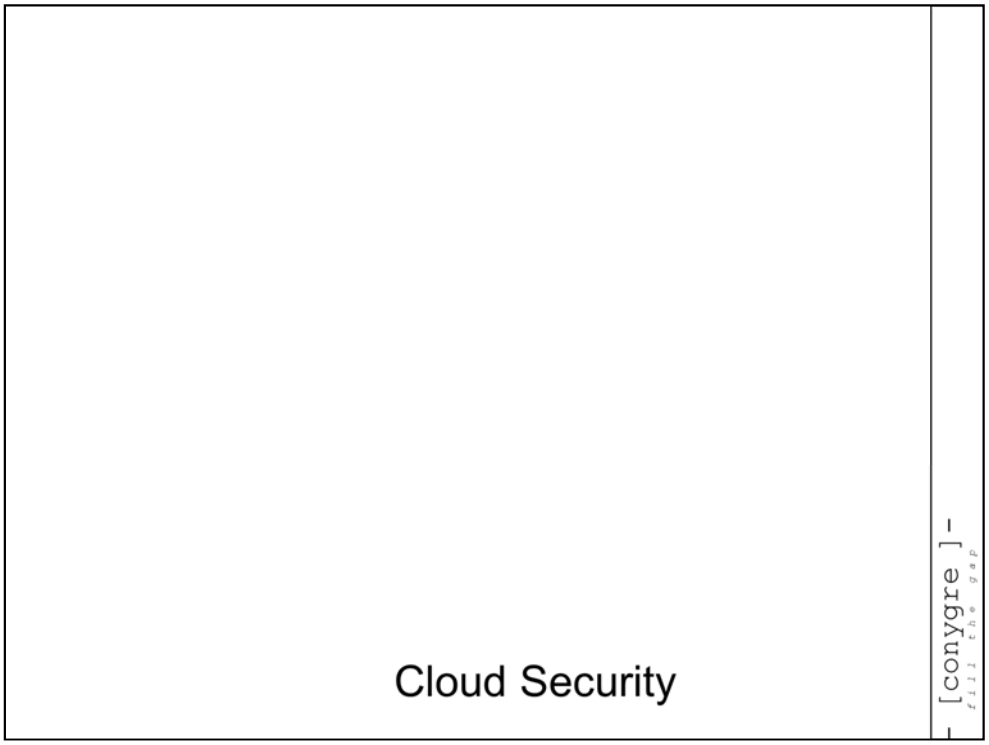
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Platform Overview



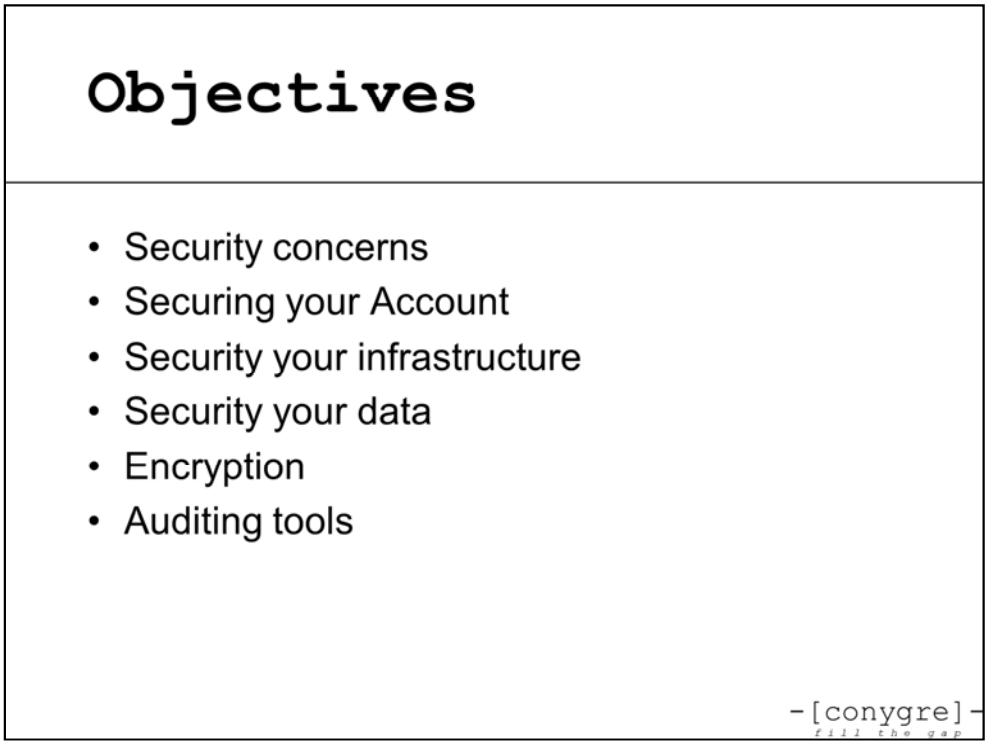
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



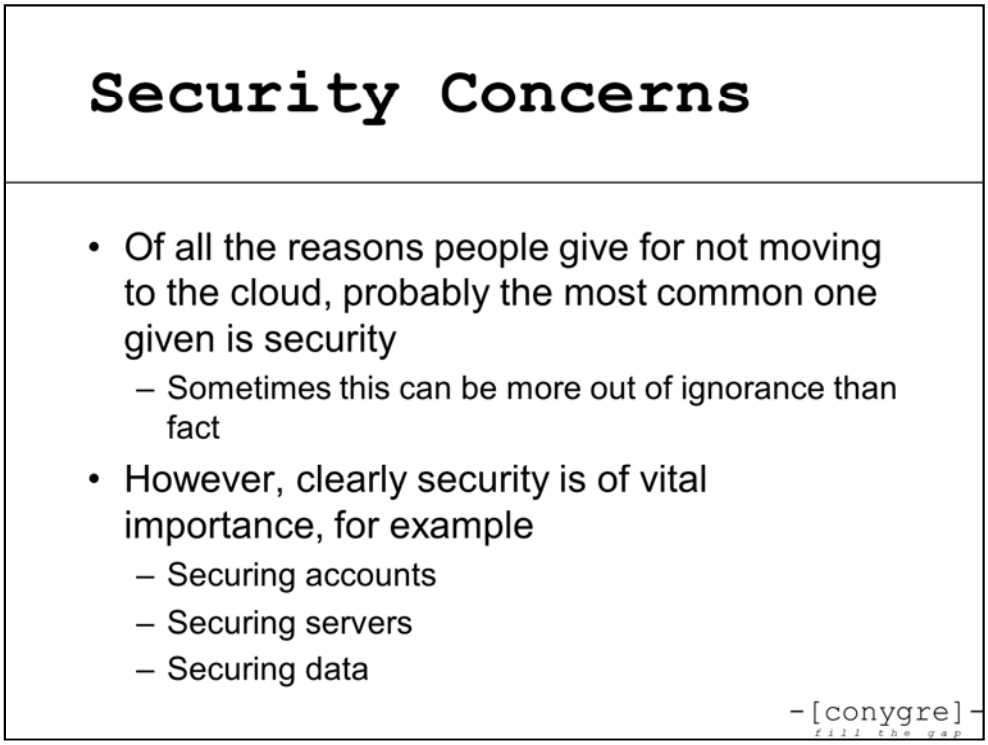
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



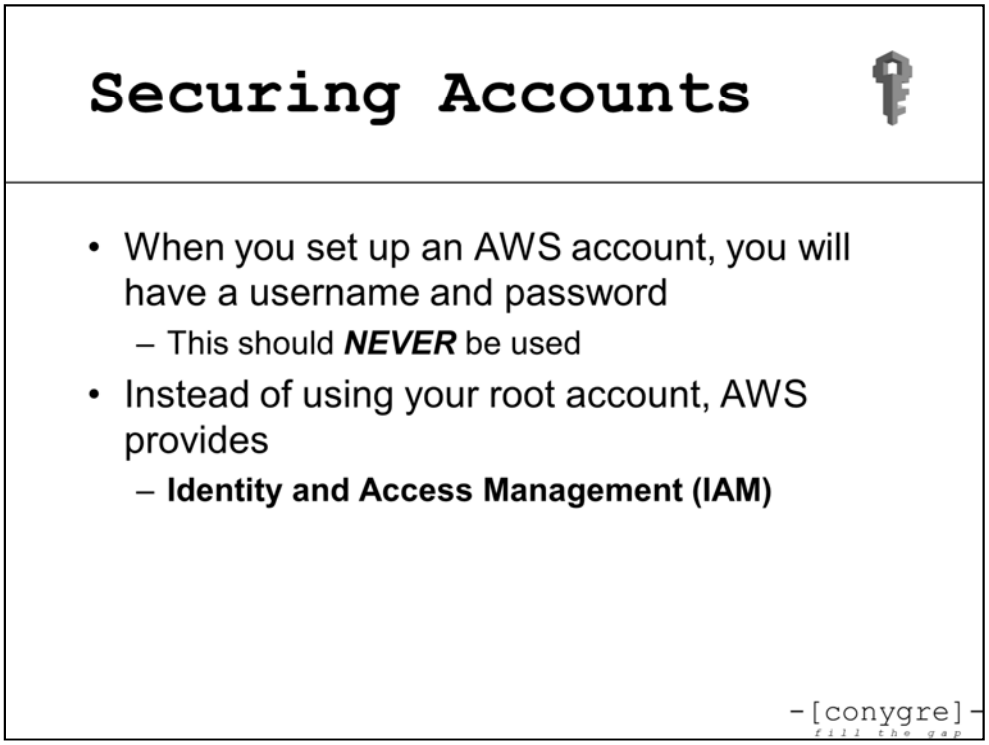
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



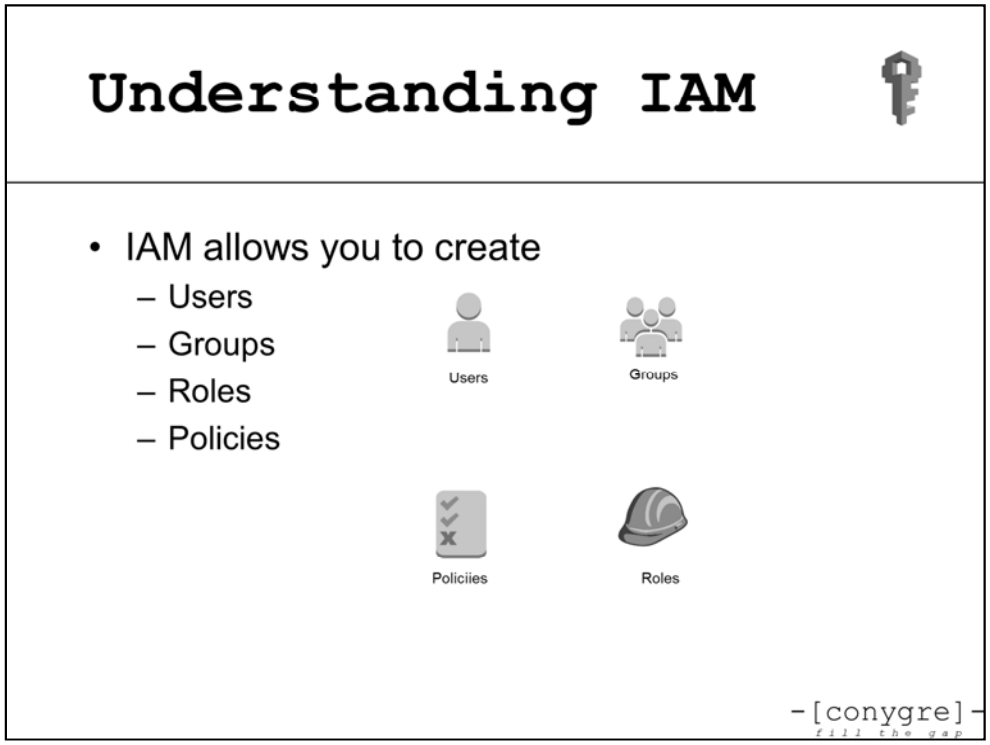
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



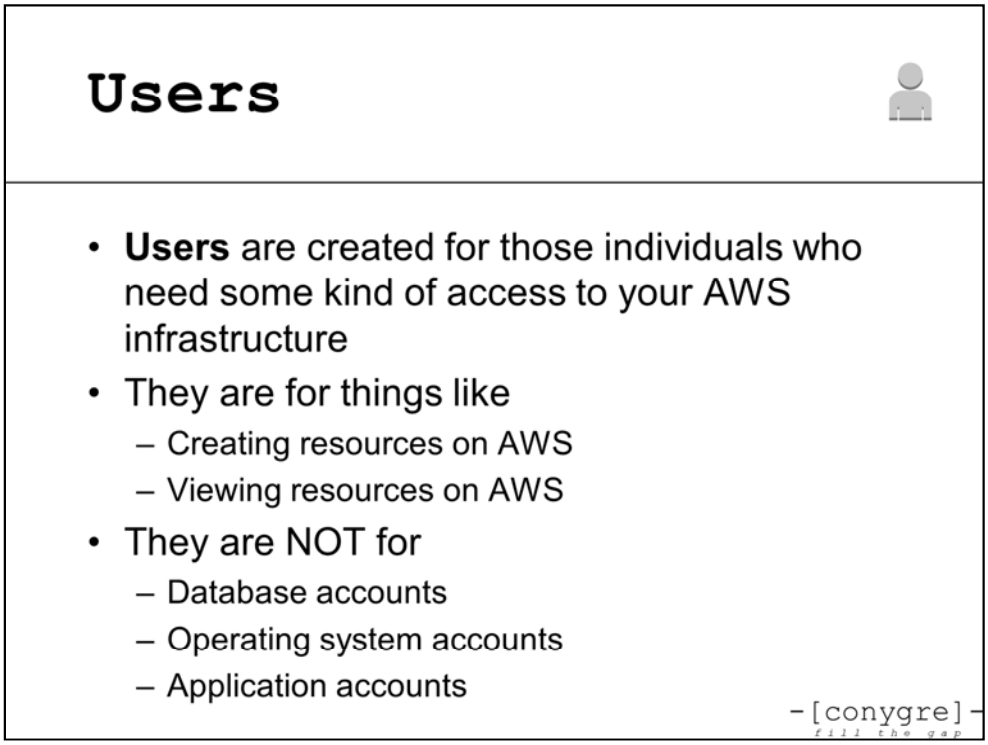
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



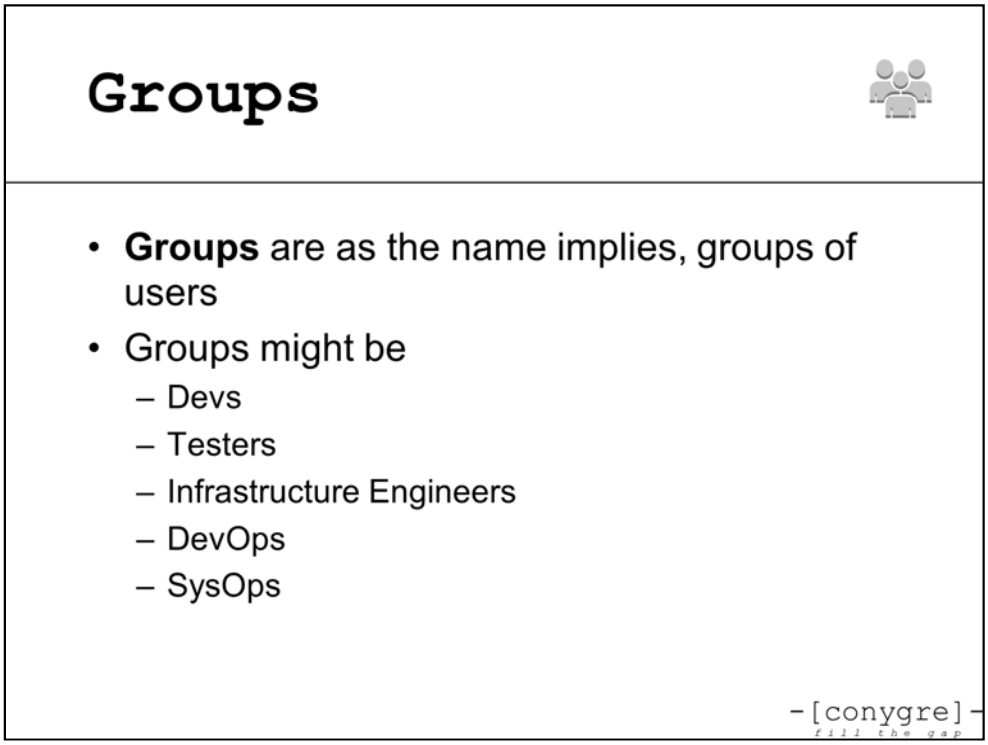
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security

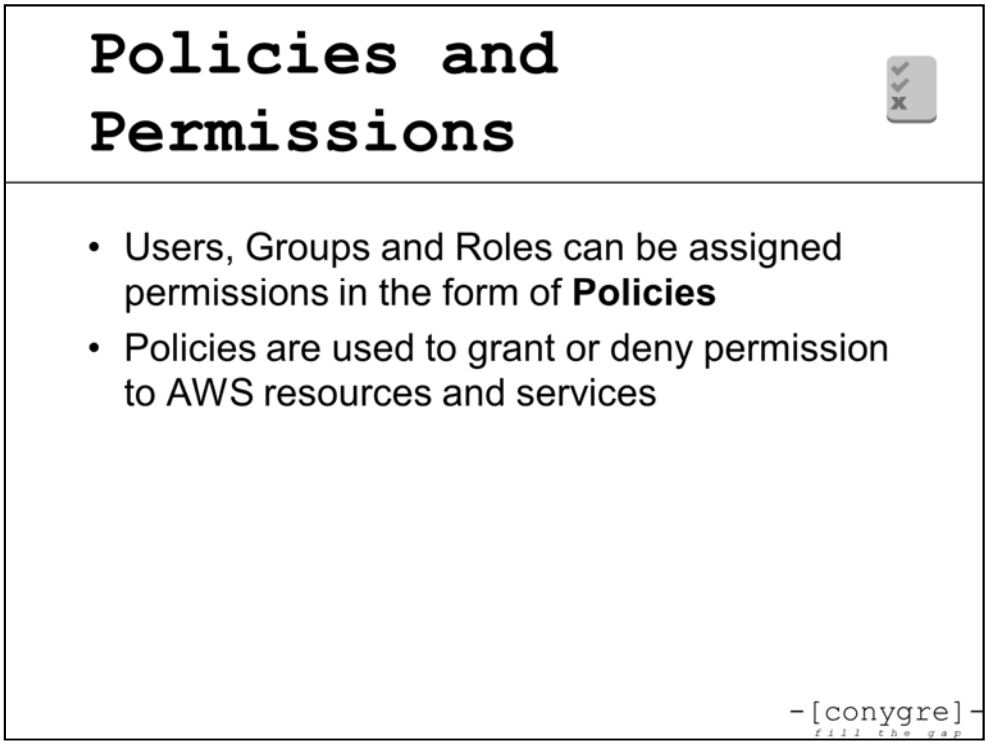
Roles

- **Roles** are used as
  - Temporary permissions for individuals
  - Temporary permissions for applications
    - Eg. A mobile app requiring permission to upload to S3
  - Services in AWS to have permission to do things
    - An EC2 instance to listen to a queue
    - A Lambda function to be allowed to stop and start servers

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security

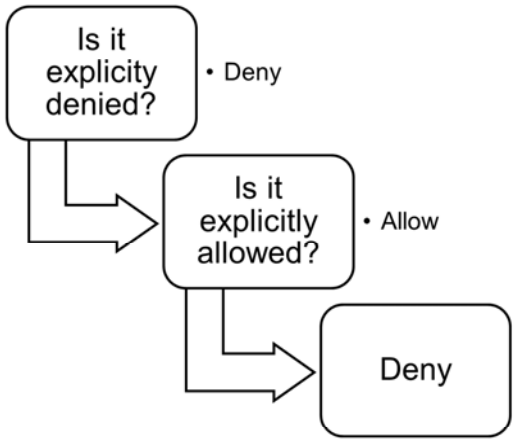


©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security

Default Permissions

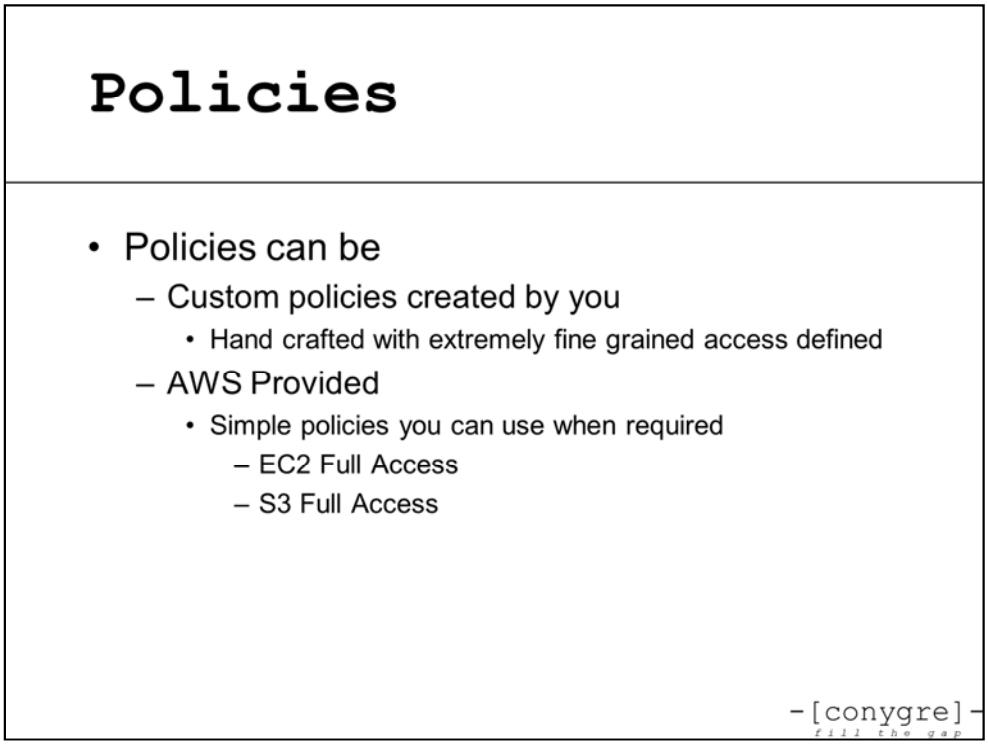
- Everything is denied by default unless explicitly allowed
- Anything explicitly denied can never be allowed



-[conygre]-
fill the gap

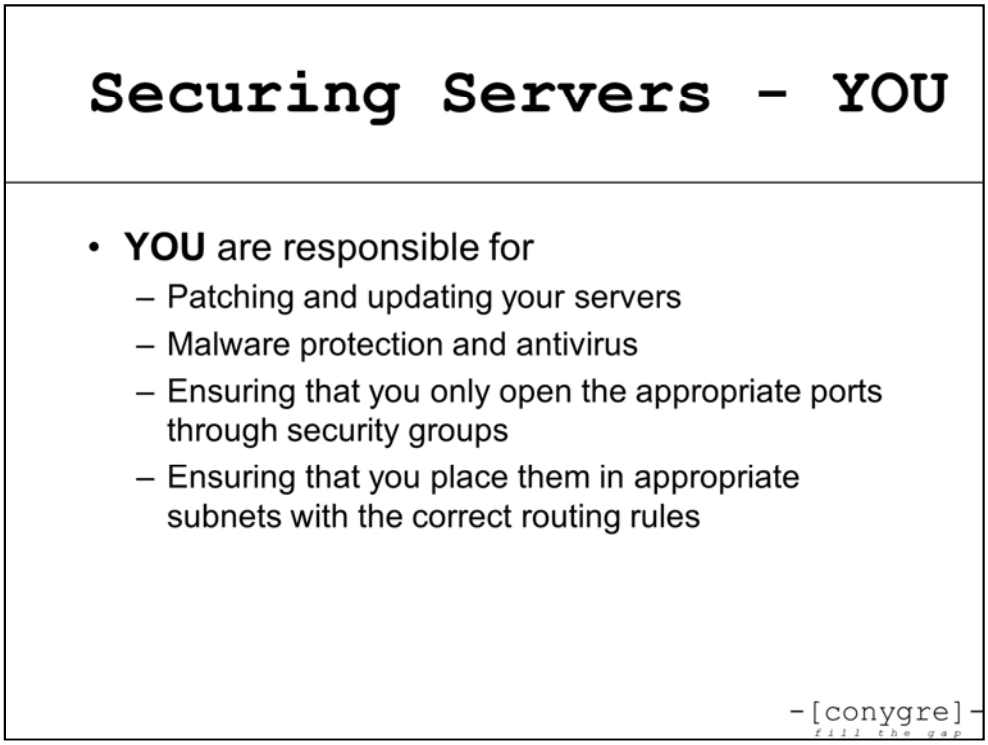
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



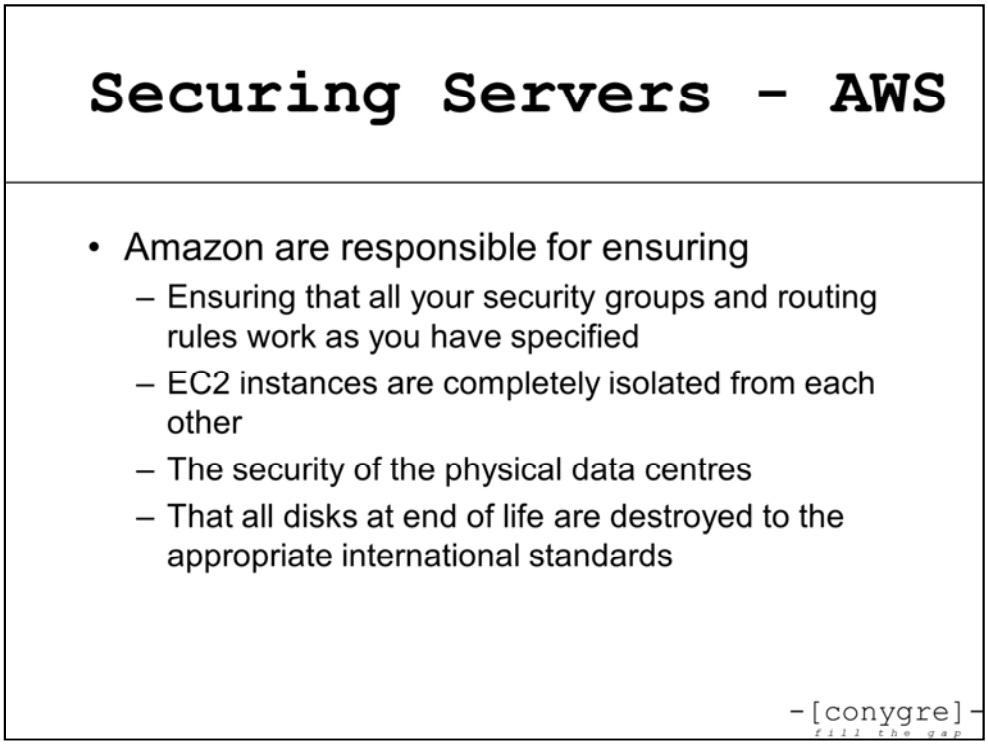
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



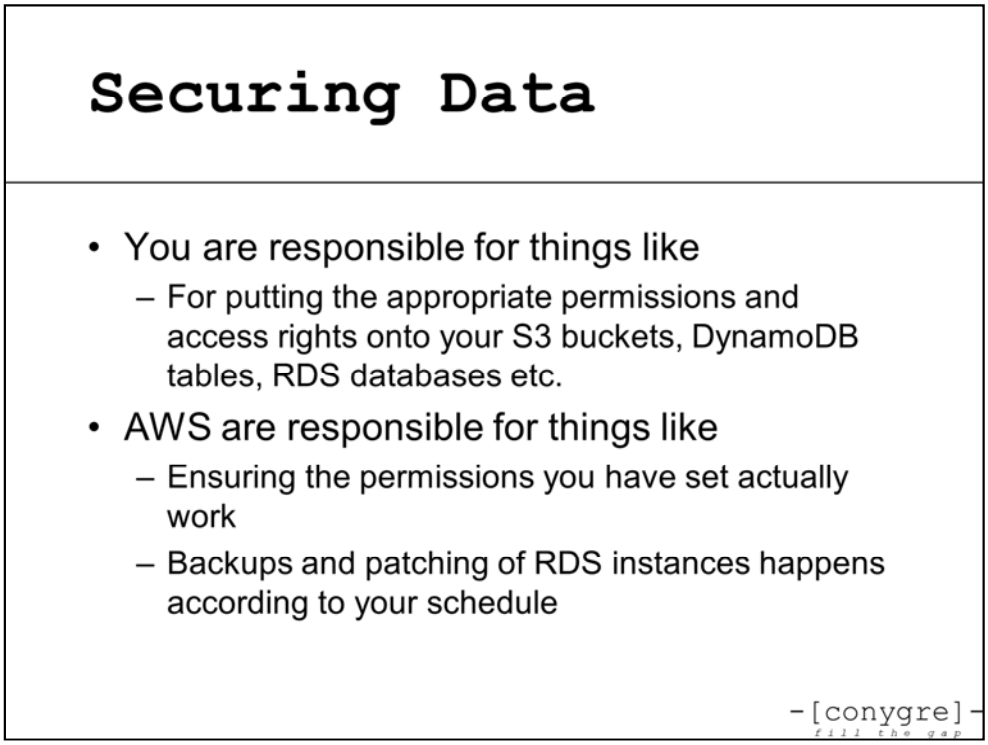
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



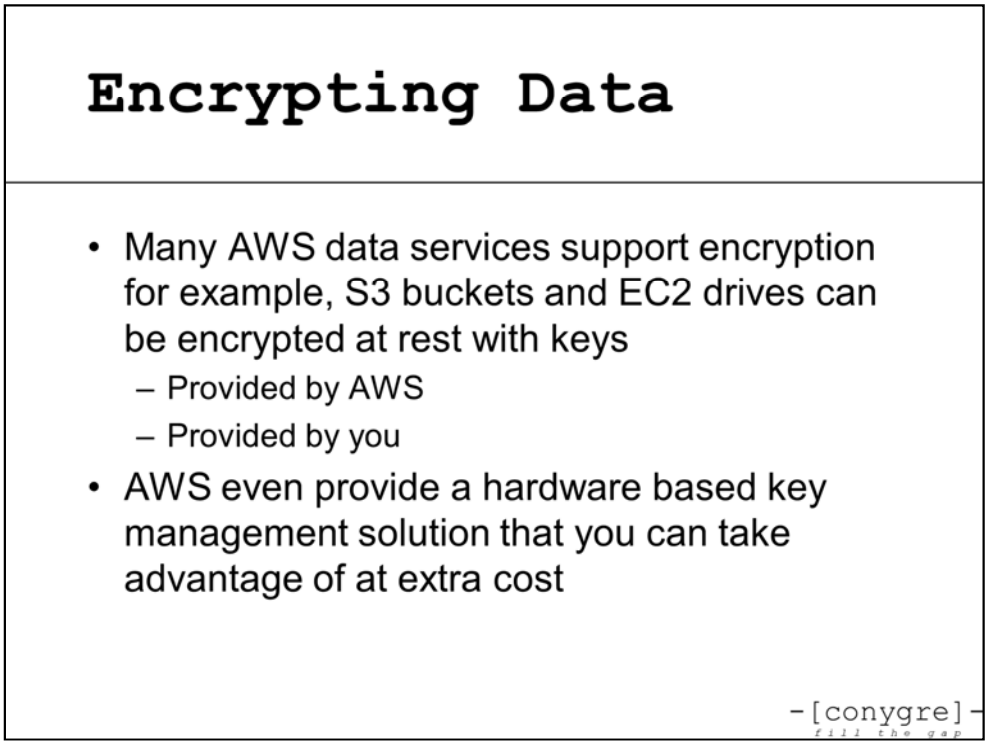
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



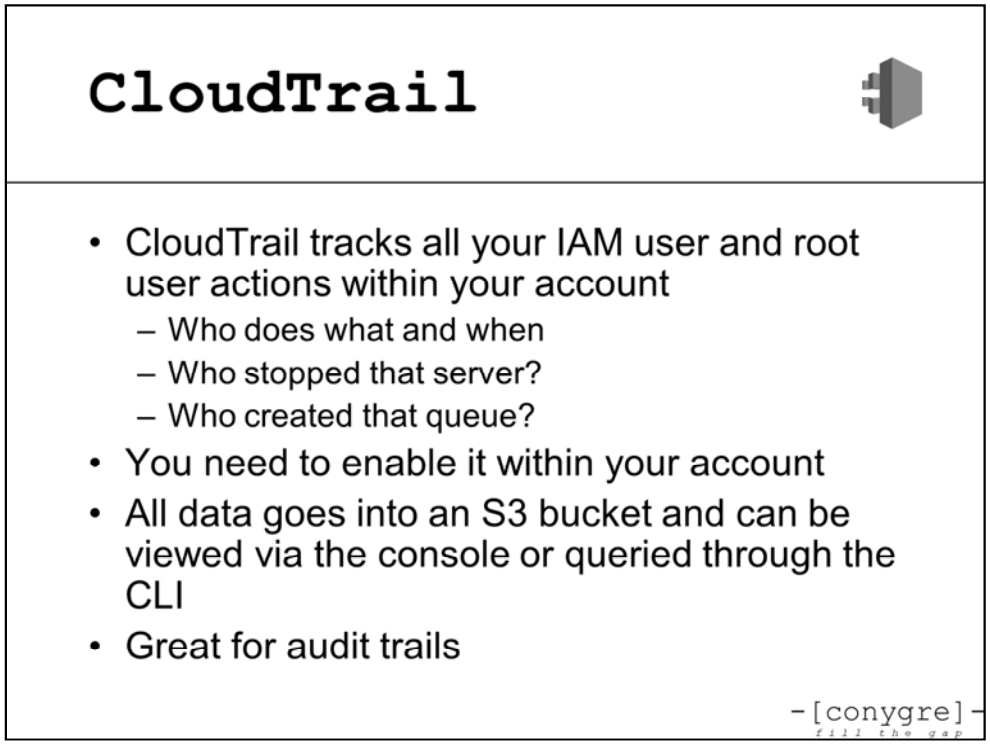
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



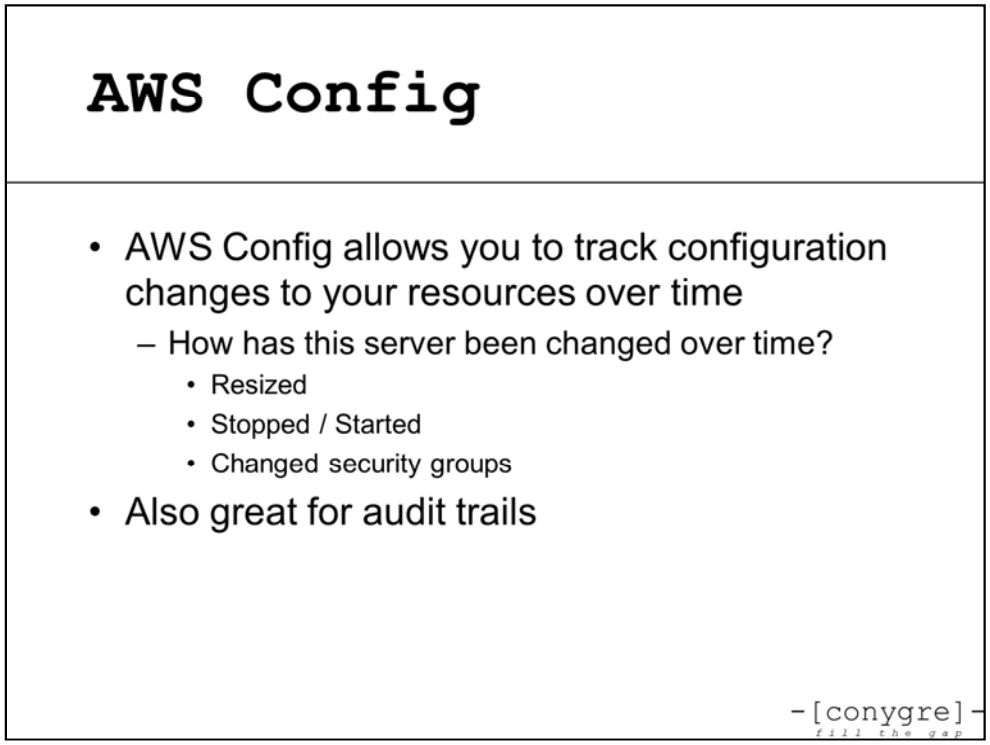
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



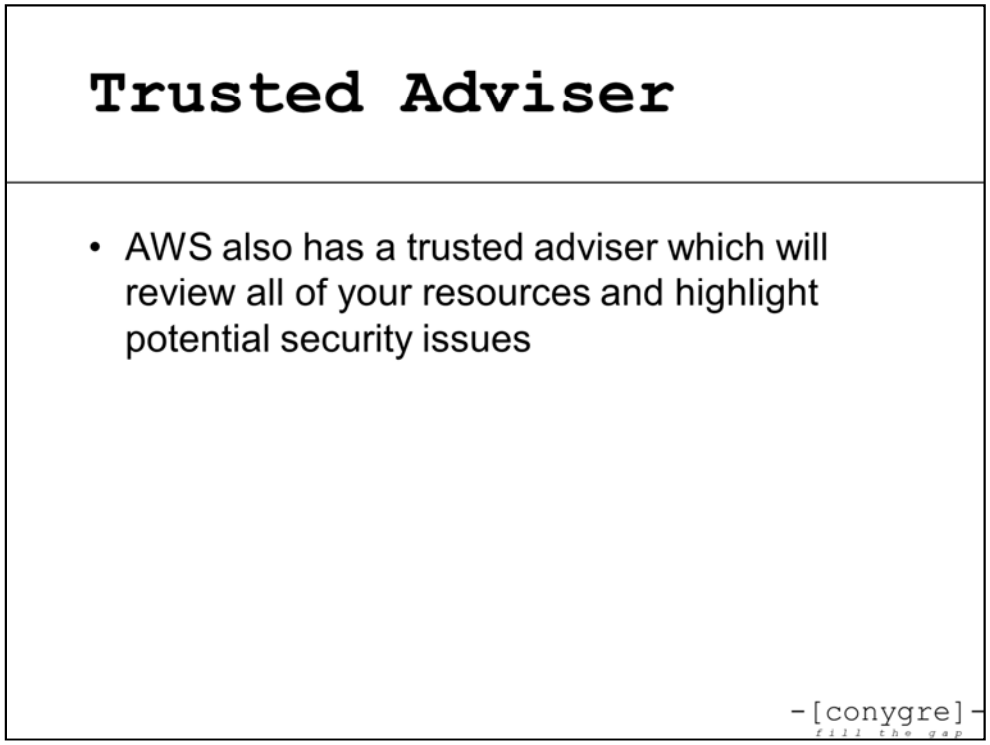
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



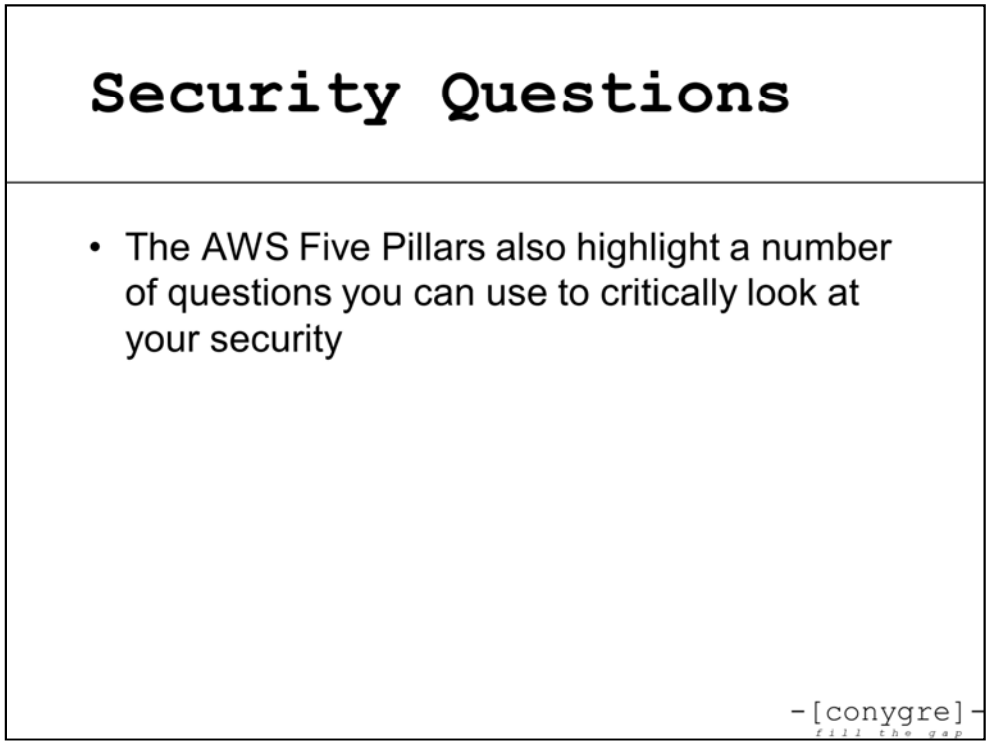
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



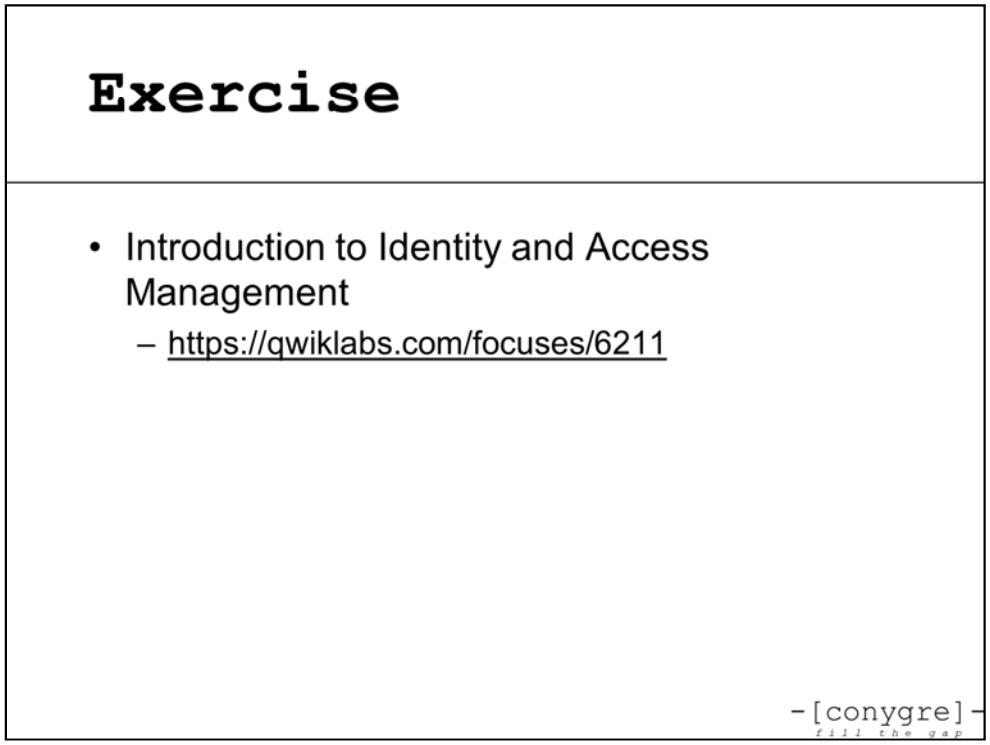
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security



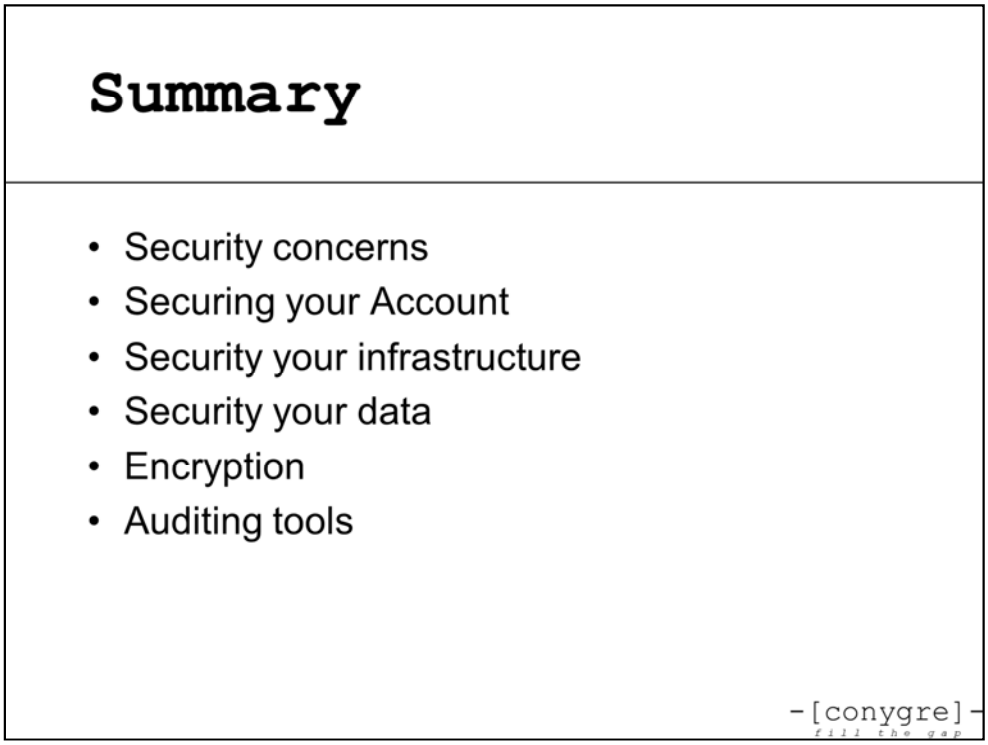
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security

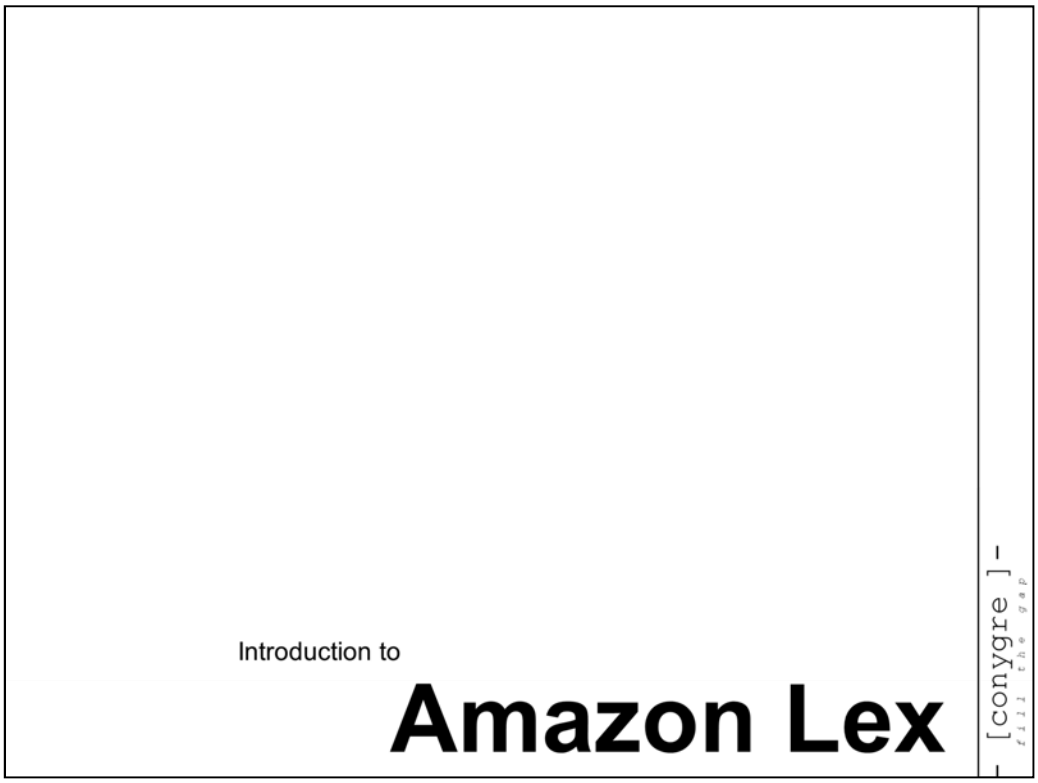


©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Security

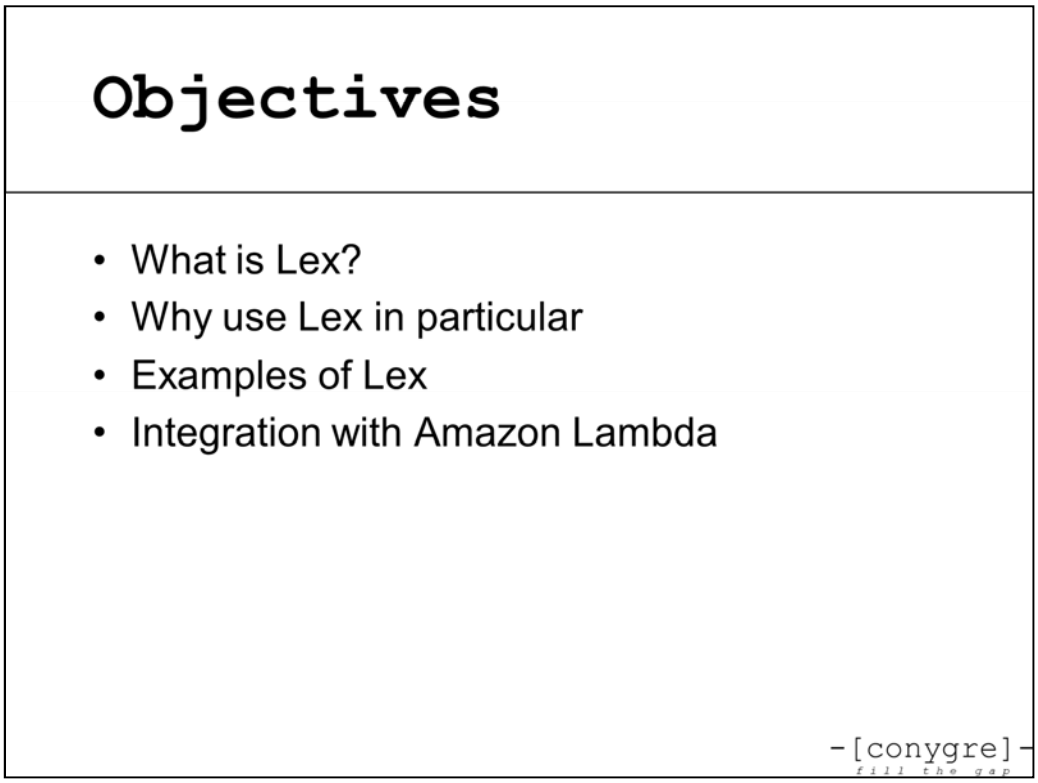


©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited



© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

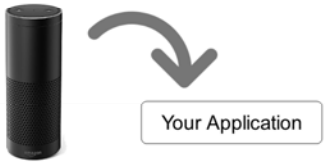


© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

What is Lex?

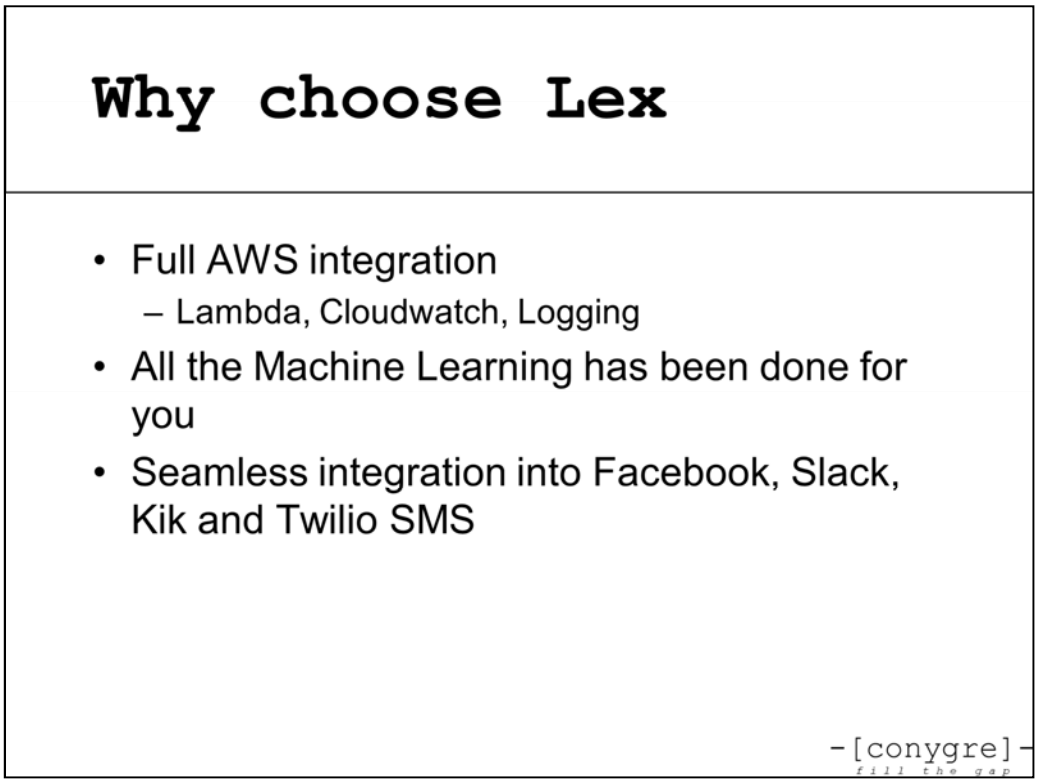
- Amazon describes Lex as a 'service for building conversational interfaces into applications using voice and text'.
- Essentially Lex takes all the **Natural Language Understanding** and **Automatic Speech Recognition** from the Amazon Echo and makes it available to developers in order to build a more engaging **UX.**



-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

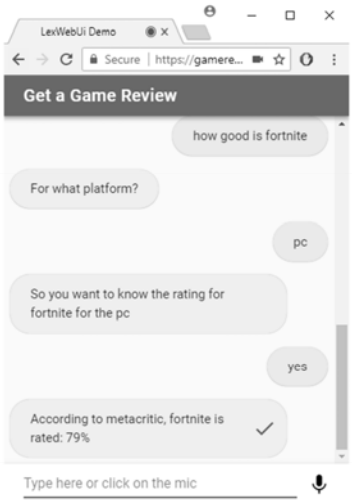


© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Example

- To demonstrate how Lex works you will see how to create a game review chatbot



-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Building a Chat Bot to Rate Games

- Need to establish how the conversation will flow.

Customer	Bot Response
Hello, I would like to get a rating for a game	Okay, what game would you like to rate?
Minecraft	Okay **Minecraft** for which **platform**?
PC	You want me to get a rating for **Minecraft** for the **PC**
Yes	
	According to Metacritic, Minecraft for PC is rated 93%

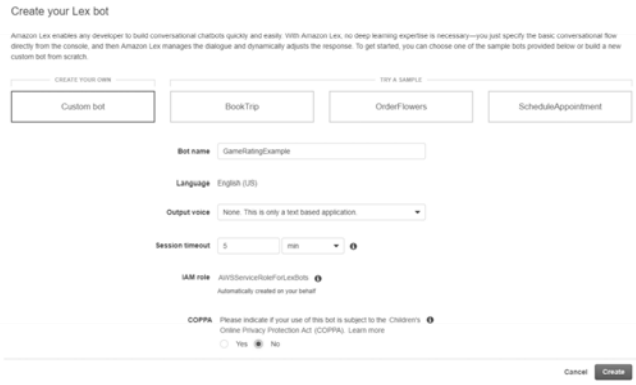
-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Creating the Bot

- Bots can be created through the AWS Web Console



-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Sample Bots

- There are some sample bots you can use to get started quickly
  - Order Flowers
  - Book a trip
  - Schedule an appointment
- Or you can create your own from scratch

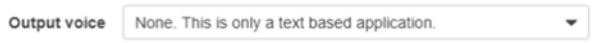
-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Voice Control

- Since Lex is the engine behind the Amazon Echo with Alexa you can optionally select that your chatbot can also work with voice as well as text



- If you do enable this, the red dot appears in the browser bar if the user enables it

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Roles

- ChatBots require permissions in order to invoke anything
- If voice is enabled, the Polly SynthesizeSpeech permission is required

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "polly:SynthesizeSpeech"
         ],
         "Resource": [
            "*"
         ]
      }
   ]
}
```

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Intents and Slots

- You now need to set up **Intents** and **Slots**
  - Intent – what the user wants to do
  - Slot – specific pieces of information provided by the user

Customer	Bot Response	Intents and Slots
Hello, I would like to get a rating for a game	Okay, what game would you like to rate?	Intent: **Rate Game** established
Minecraft	Okay **Minecraft** for which **platform**?	Slot: **Game** = Minecraft
PC	You want me to get a rating for **Minecraft** for the **PC**	Slot: **Platform** = PC
Yes		Initialise **Lambda** function to get rating from Metacritic
	According to Metacritic, Minecraft for PC is rated 93%	Intent is now **fulfilled**

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Creating an intent

- To create an Intent, click **Create Intent**
- You can have multiple intents
- In the example we require one intent and our **Intent** is to get a **Game Rating.**

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18
Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Invoking the Intent

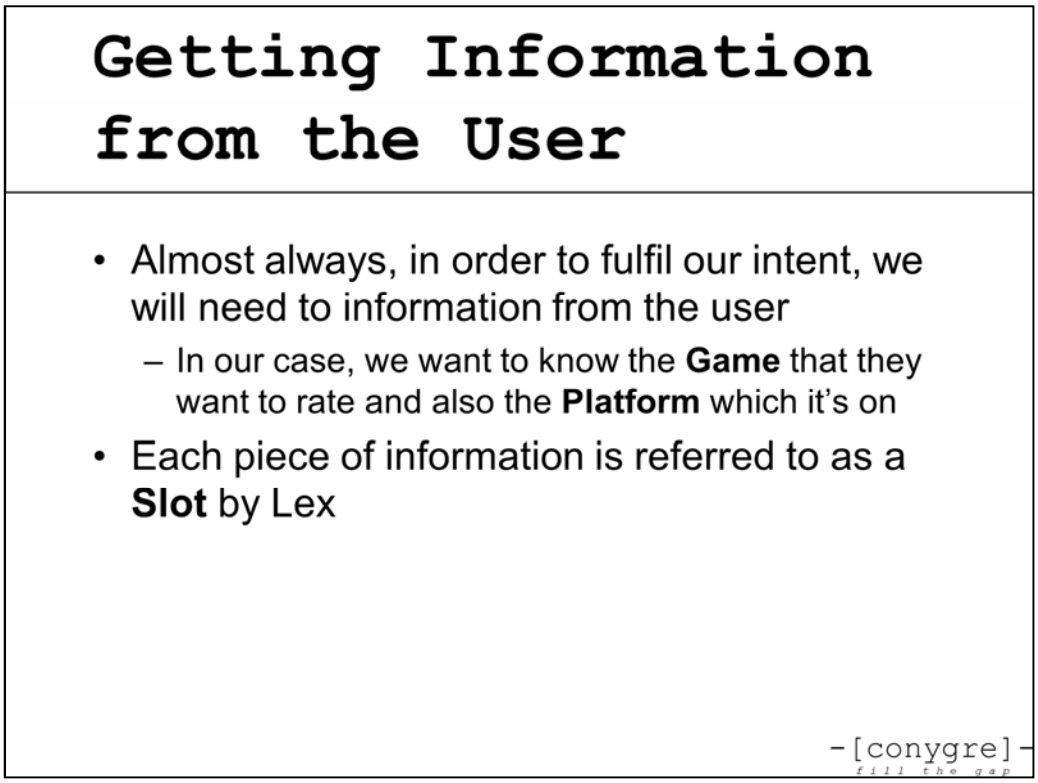
- We need to tell Lex what kinds of things the customer might enter in order to invoke our Intent
- Lex calls these phrases **utterances**

- The words in curly brackets are slots – discussed next

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

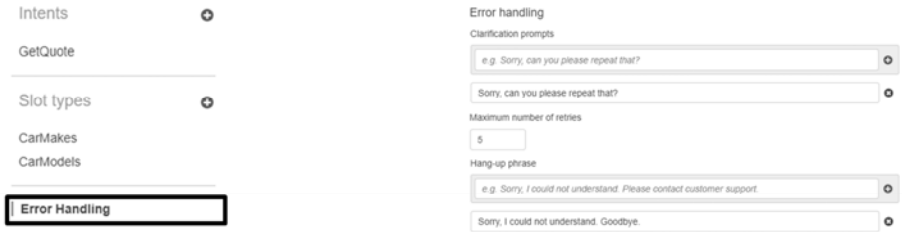


© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Error Handling

- Sometimes a user will enter something that Lex doesn't understand.
- In this scenario, Lex comes with features to prompt a user to re-enter some information (ideally in a way Lex **will** understand)



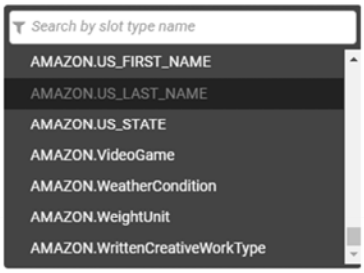
-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Slots and Slot Types

- User data is placed into **Slots**
- Slots always have a Slot Type
- Lex defines some commonly used types
  - There are lots of them!



-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Custom Slot Types

- For the **Platform** slot we need to define a custom slot type which lists all the possible platforms that our bot is able to get game ratings for
- In this case we want to make sure that the slot type is **not** set to expandable, as we only want Lex to populate the Platform slot with valid platforms
- Note that slot names are case sensitive

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Custom Slot Types

- Slots also have a Prompt which is the phrase the bot will use to elicit the information from the customer

- Always place your slot names in curly brackets when adding the prompts

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Creating the Slots

- Notice that now the slots have been defined the References to them in the **Utterances** section have been **{color-coded}**

How do people feel about {Game}

How good is {Game}

I would like to know the rating of a game

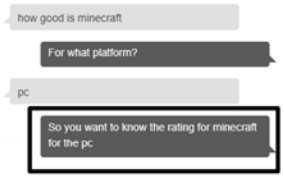
-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Confirmation

- An important ‘good practice’ of Bot Design is to make sure that the user knows what is going on
- Repetition, Confirmation and Repetition are some excellent ways to incorporate this.
- Lex provides a useful **Confirmation Prompt** section to easily ensure the user is happy with the information they have given



-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Fulfilling the Intent

- In Lex an intent can either be **Fulfilled** or **Failed**
- In order to determine this, we can either
  - Use the built in 'return parameters to client' function. In which case the intent will be **fulfilled** once Lex has populated all the required **Slots**

    Or
  - We can utilise Amazon **Lambda** is order to do some logic on the filled **Slots** and determine for ourselves whether or not the intent has *actually* been fulfilled

    [Similarly to above, the **lambda** will be called once all the required **slots** are filled.]

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Adding a lambda to Lex

- Incorporating the lambda into our lex bot is straightforward
  - In the fulfilment tab we just tell Lex to call our lambda function
  - The less straightforward part is how *exactly* Lex interacts with our lambda
- When the lambda is invoked it will be passed a Lex **Event**
  - This **event** is a JSON construct which describes the current intent and the slots that are filled

Event example


```
{
"messageVersion": "1.0",
"invocationSource": "FulfillmentCodeHook",
"userId": "pcp241vzjsv7m97i2og0qtvl2gbjjuob",
"sessionAttributes": {},
"requestAttributes": None,
"bot": {
   "name": "GameRating",
   "alias": "$LATEST",
   "version": "$LATEST"
   },
"outputDialogMode": "Text",
"currentIntent": {
   "name": "GetGameRating",
   "slots": {
      "Game": "minecraft",
      "Platform": "pc"
      },
   "slotDetails": {
      "Game": {
         "resolutions": [],
         "originalValue": "minecraft"
      },
      "Platform": {
         "resolutions": [{"value": "pc"}],
         "originalValue": "pc"
         }
      },
   "confirmationStatus": "Confirmed"
   },
"inputTranscript": "yes"
}
```


-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Adding a lambda to Lex

- A simple python script should do the trick for getting our game rating from Metacrititc
- First we import some libraries for dealing with HTML data and requests

```
from urllib.request import Request, urlopen
from urllib.error import HTTPError
from bs4 import BeautifulSoup
```

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Adding a lambda to Lex

- Then we add in the **lambda_handler** function which deals with the stuff from Lex

```
def lambda_handler(event, context):
```

- The **event** parameter contains our JSON from the chatbot
  - In the function we can extract our **Game** and **Platform** information

```
game = event['currentIntent']['slots']['Game']
platform = event['currentIntent']['slots']['Platform']
```

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Adding a lambda to Lex

- Then we need some logic for dealing with games with multiple words in their title
  - The Metacritic Web site uses hyphens to deal with this, so we need to hyphenate the game name

```
words = game.split()

    gameName = words[0]
    for word in range(1,len(words)):
        gameName = gameName+'-'+words[word]

    print("Normalized Game name: "+gameName)
    url = baseUrl+platform+"/"+gameName
```

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Adding a lambda to Lex

```
#make request to metacritic
    request = Request(url,headers={'User-Agent': 'Mozilla/5.0'}) #set a browser useragent
    fulfillmentType = 'Failed'  #assume failure unless we can get a rating
    try:
        htmlData = urlopen(request).read()

        soup = BeautifulSoup(htmlData, 'html.parser')

        #find the rating from the metacritic page if it exists
        try:
            rating = soup.findAll("span", {"itemprop": "ratingValue"})[0].string
            result = 'According to metacritic, '+game+' is rated: '+rating+'%'
            fulfillmentType = 'Fulfilled' #update fulfillment type to success
        except IndexError:
            result = 'Metacritic doesn\'t seem to have a rating for '+game+' yet'
    except HTTPError:
        result = 'I wasn\'t able to find a rating for: '+gameName
```

-[conygre]-
fill the gap

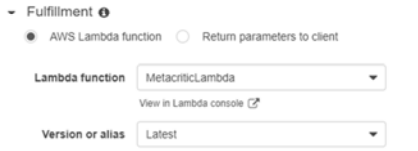
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Adding a lambda to Lex

- Finally we can create the JSON **response** to send back to Lex
- And tell Lex to use our Lambda as the fulfilment criteria

```
response = {
        "dialogAction":
          {
           "fulfillmentState":fulfillmentType,
           "type":"Close","message":
             {
              "contentType":"PlainText",
              "content": result
             }
          }
        }
return response
```



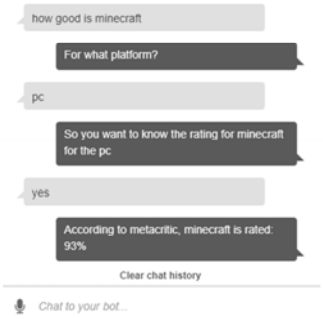
-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Testing the Bot

- In the Lex console make sure you remember to click the **Save Intent** button
- To test, expand Test Chatbot menu on the right



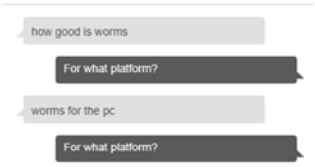
-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Some finishing touches

- When your Lex bot prompts users for input it currently will not necessarily be able to cope with all sensible inputs
  - Consider the following



- Lex didn't know how to respond to this longer phrase, however…

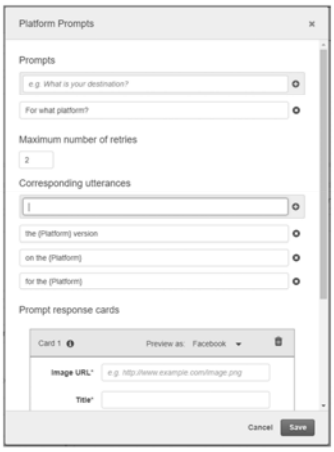
-[conygre]-
fill the gap

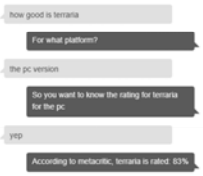
© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Some finishing touches

- Lex allows for the customisation of expected responses to prompts. In this case we can go into the prompt settings for **Platform** and put in some **Corresponding Utterances**.
- Now the conversation works.





-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Summary

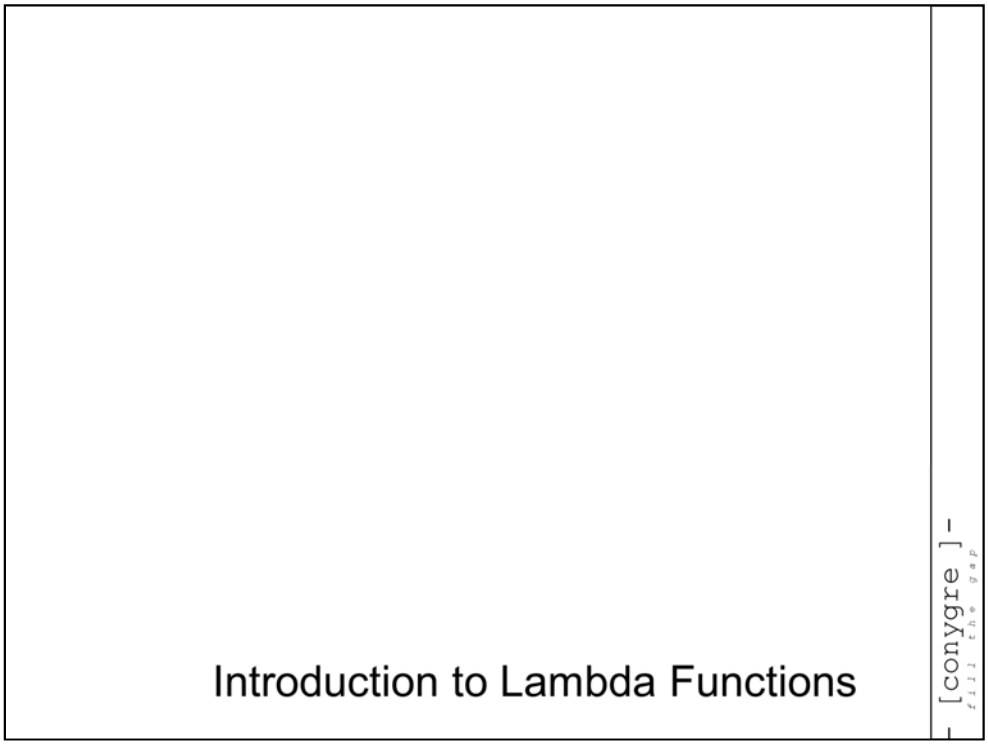
- Lex is an AWS Environment for **Natural Language Understanding**
- **Intents** describe what your bot will do
- **Slots** describe the different pieces of information your bot will need to perform the **intent**
- Slots can be explicitly **confirmed** to improve **UX**
- **Lambda** is implemented to create advanced intents that your bot can **Fulfil**

-[conygre]-
fill the gap

© Conygre IT Ltd. Jun-18

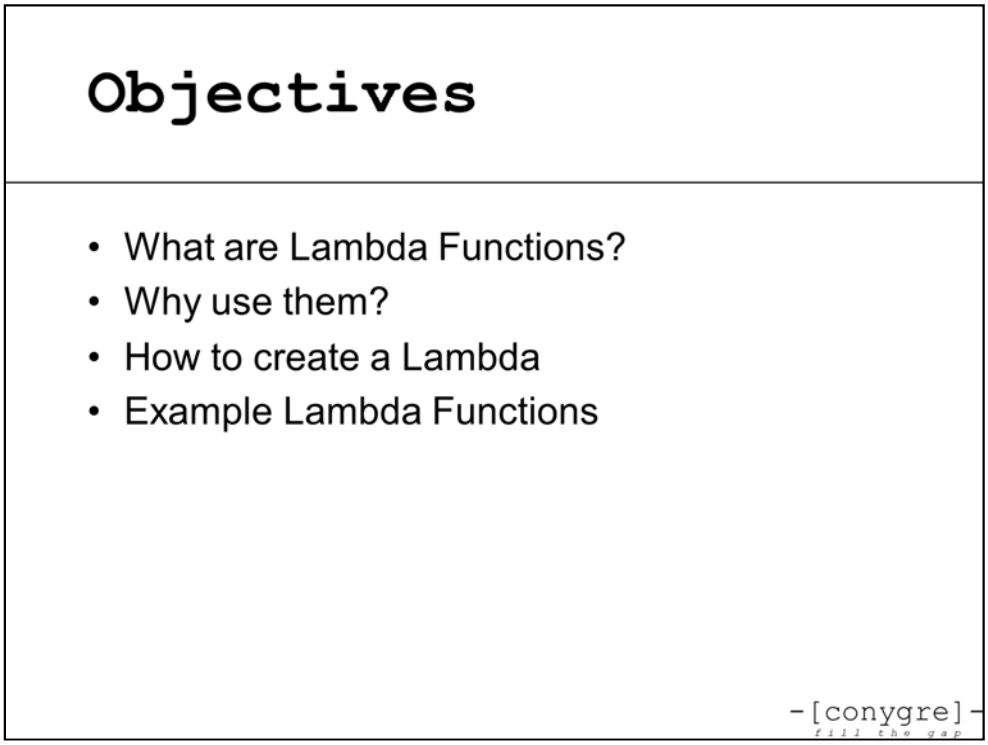
Copying or reproduction of this material is prohibited without authorisation from Conygre IT Ltd.

Introduction to AWS Lambda

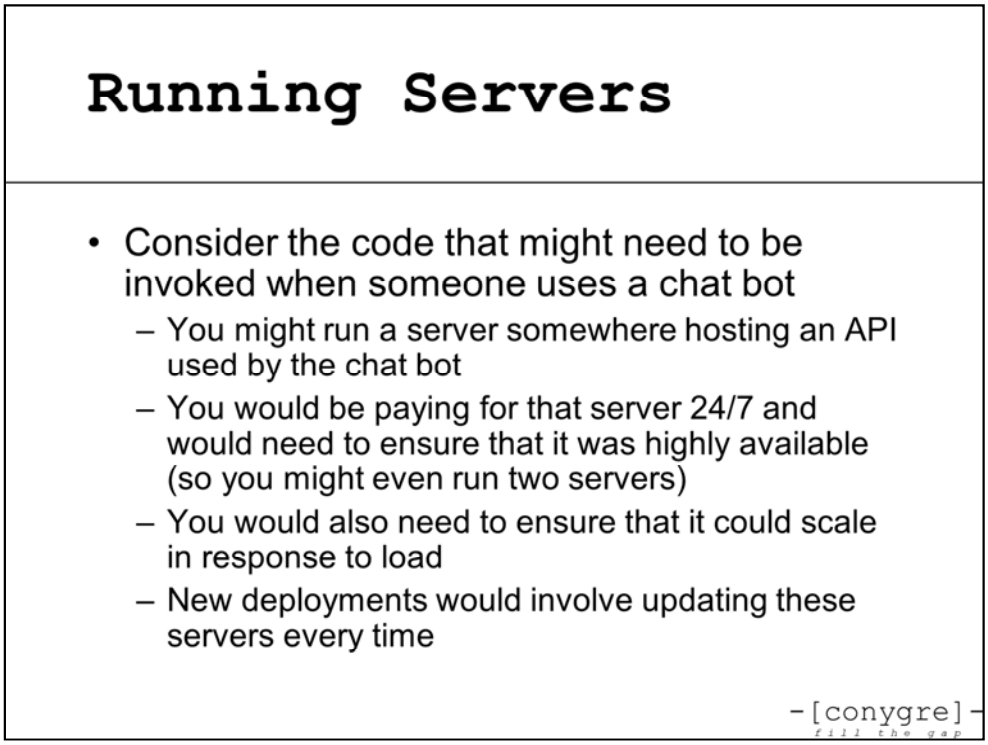


©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

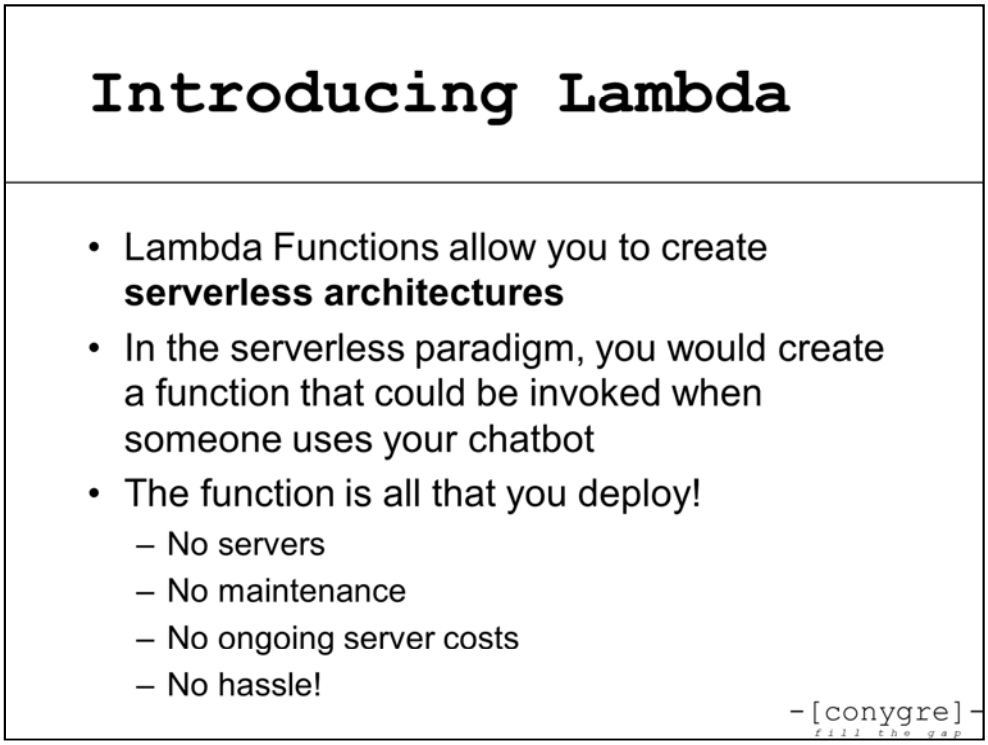


©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited



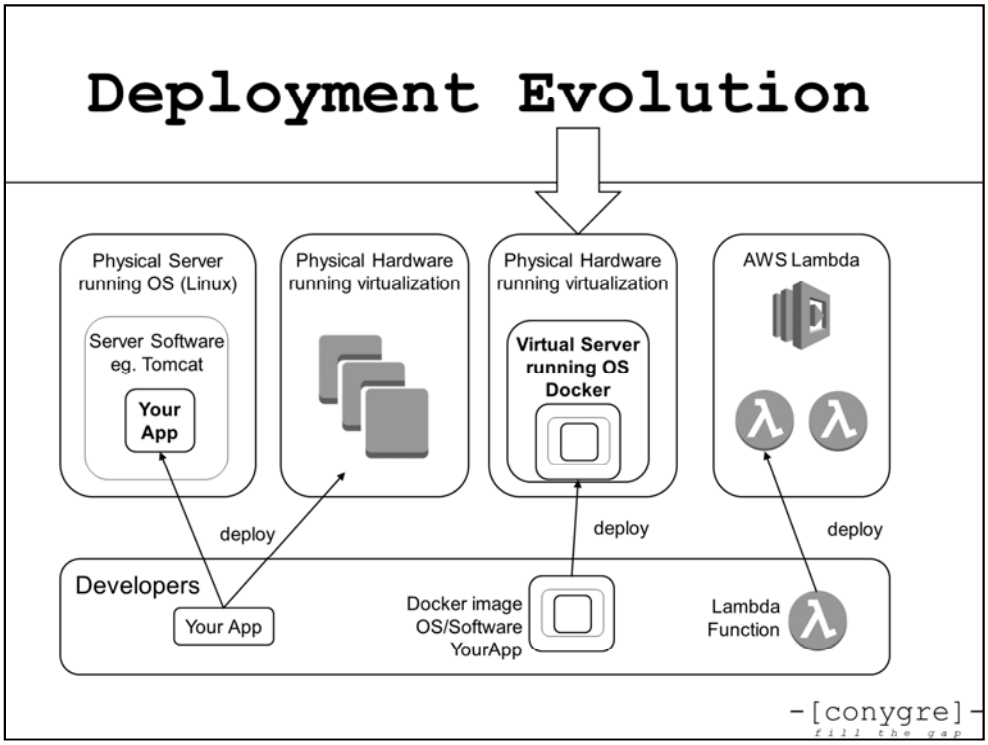
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

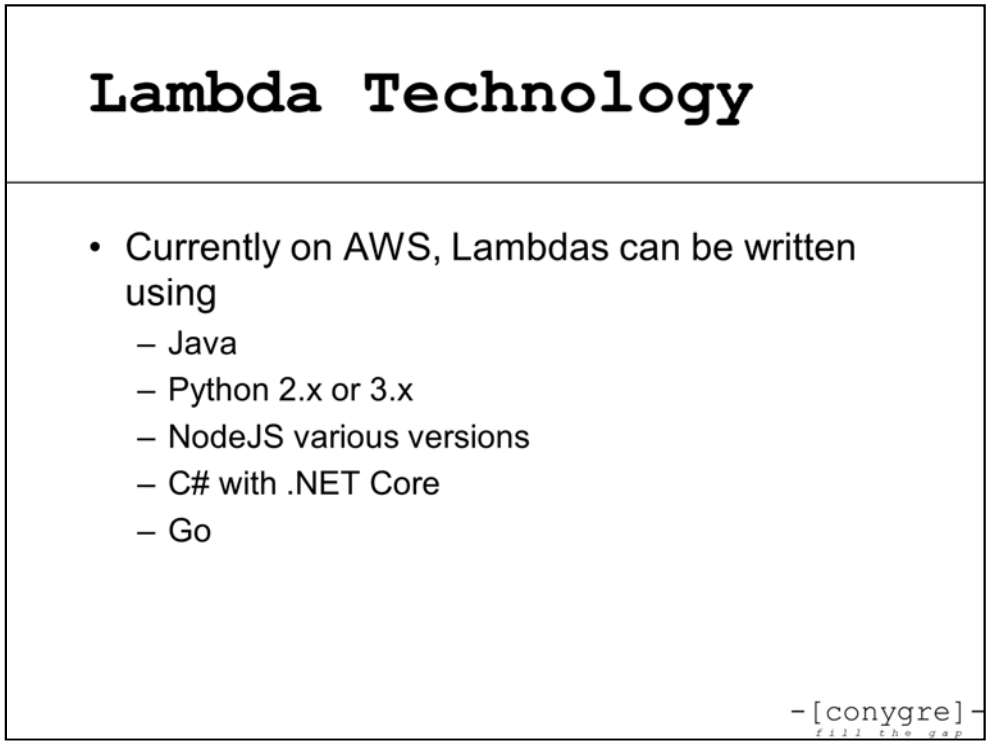


©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

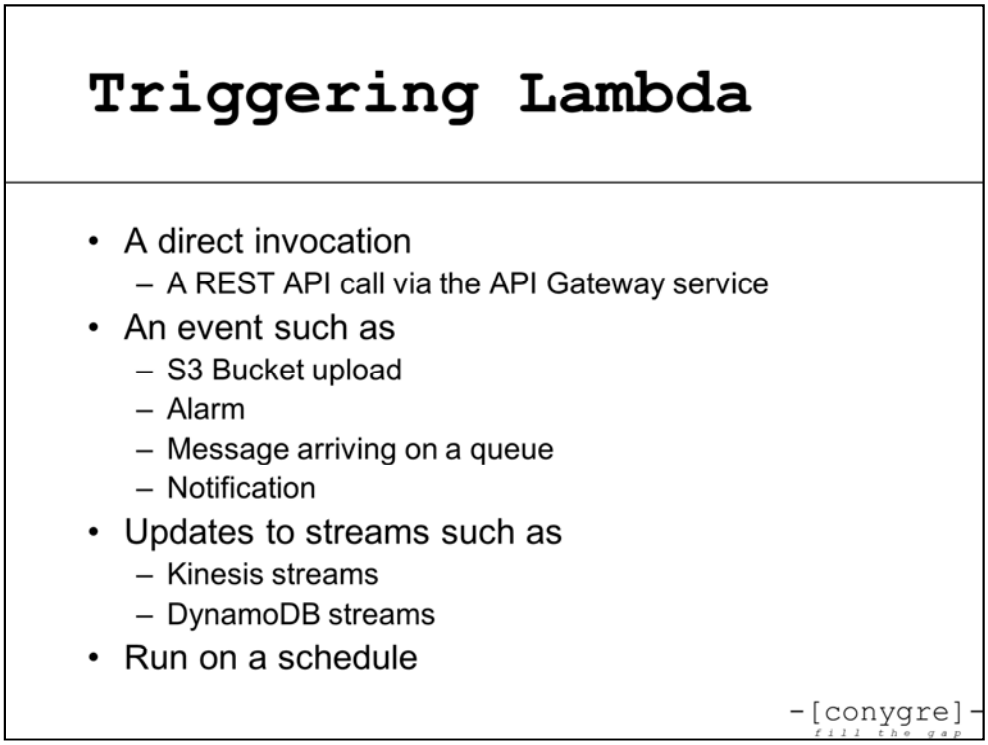
Introduction to AWS Lambda



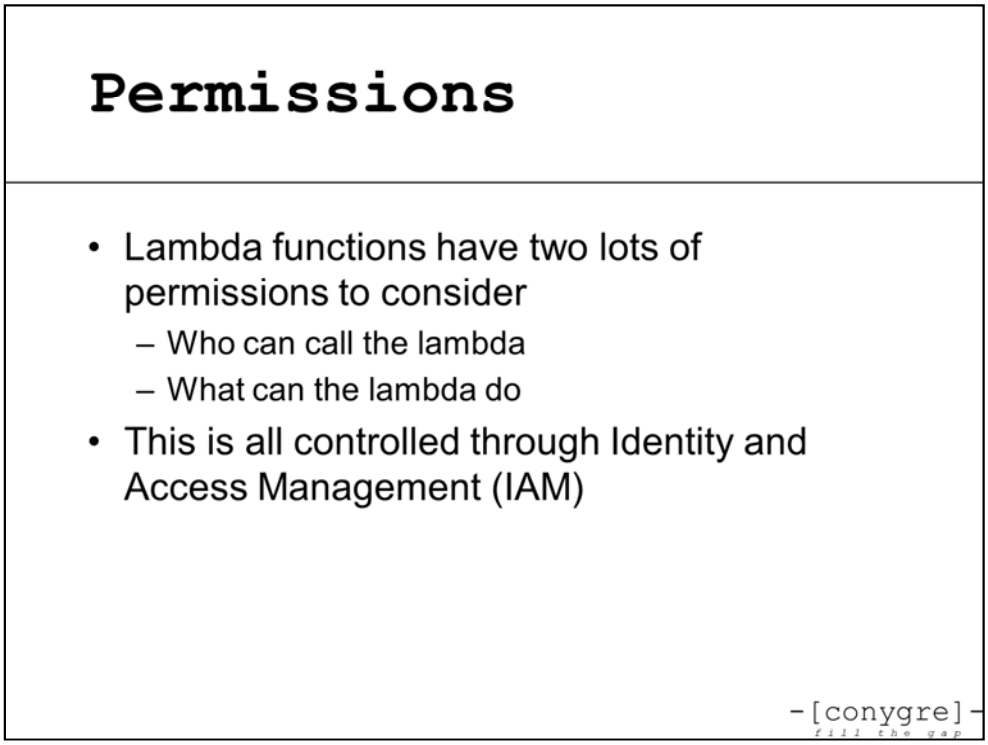
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited



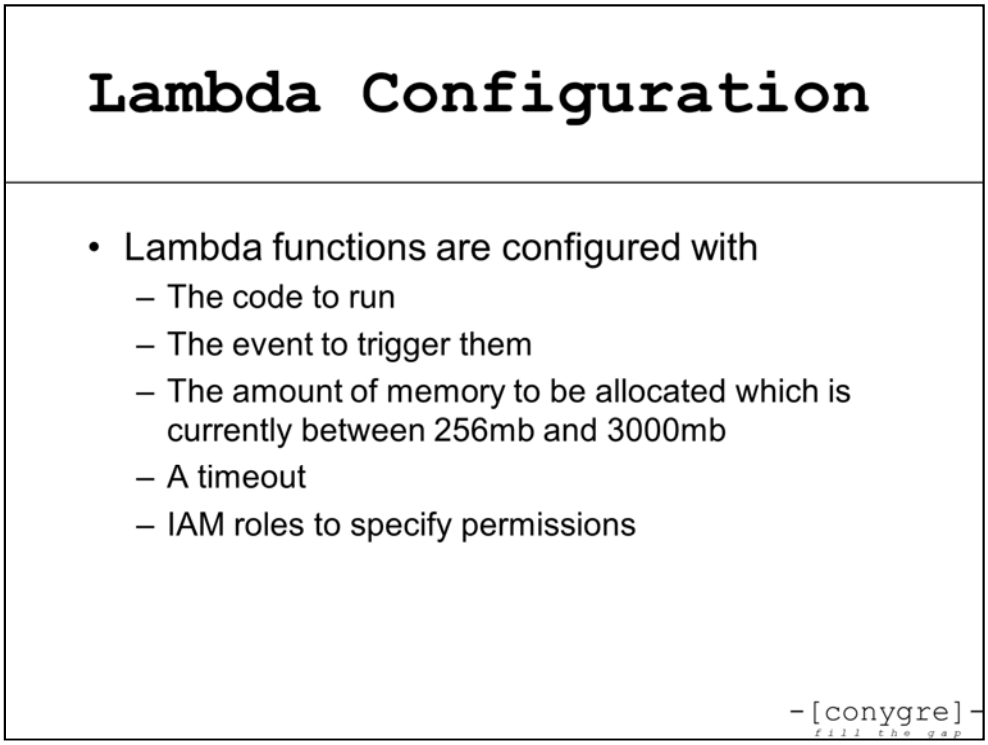
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

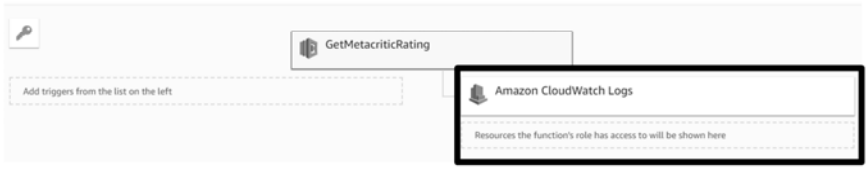


©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

Monitoring and Logging

- All Lambda metric data and logging goes to **CloudWatch**
- Cloudwatch is the AWS monitoring service
- To facilitate this you ensure that Lambdas run in a role with access to CloudWatch and CloudWatch Logs



-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

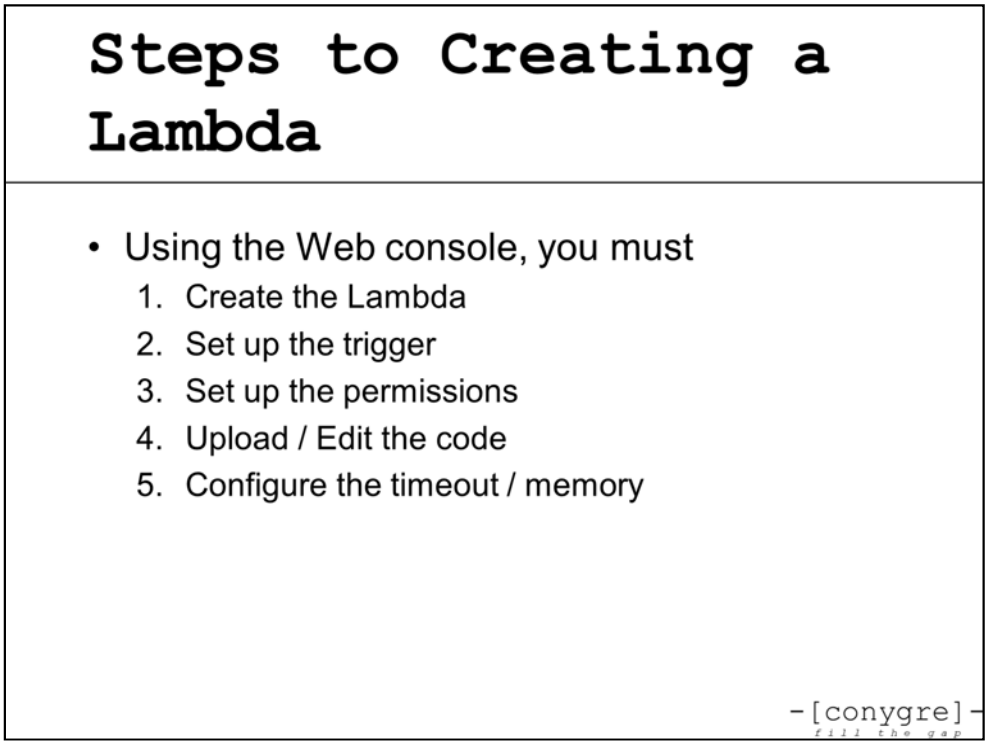
The Lambda Code

- Lambda code can be created
  - Using the inline editor in the browser or
  - In an IDE and then
    - A zip of your code can be uploaded
    - A zip of your code can be uploaded to an S3 bucket

- Note for Java and C# it cannot be edited inline since it must be compiled

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

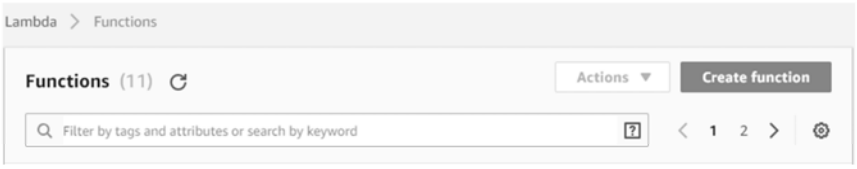


©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

Creating a 'HelloWorld' Lambda

- At the Lambda Console, click **Create function**



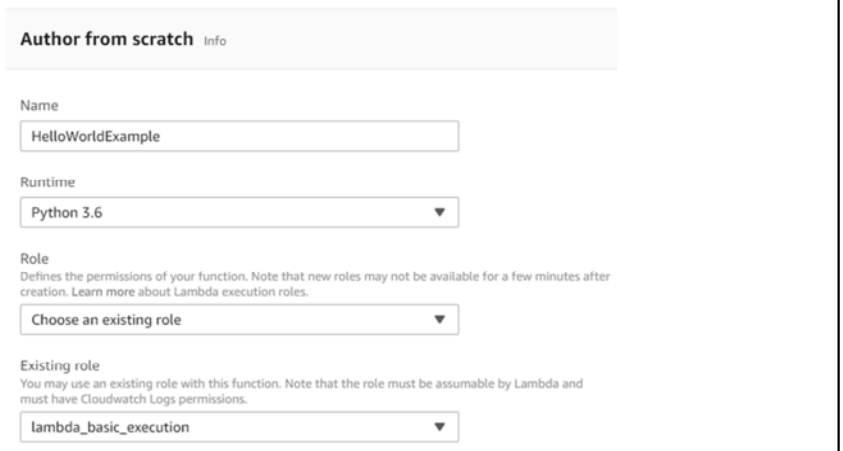
- There are many blueprints and there is also a marketplace of predefined lambdas, but to do your own, select **Author from Scratch**

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

Enter the Basic Details



- At a minimum, the role will need CloudWatch Logs

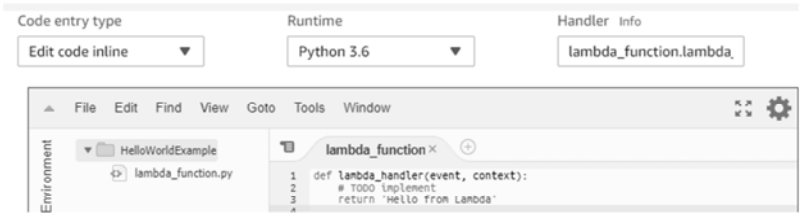
-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

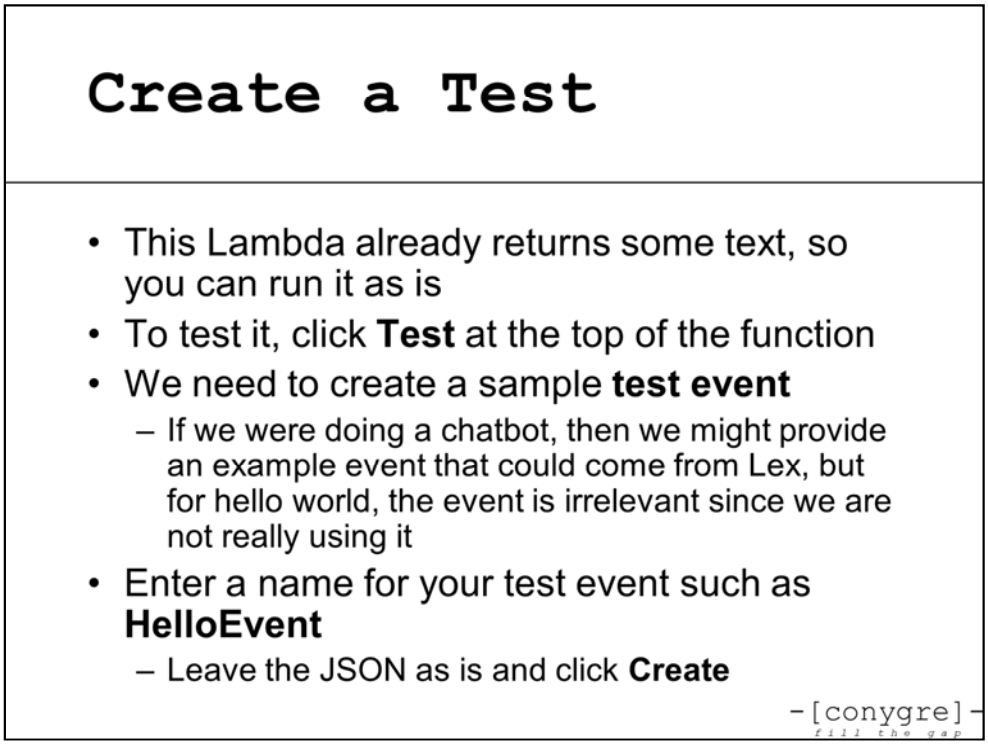
View the Code

- Click **Create Function**
- Switch the language to Python 3.x and note the code provided already has a function the name of which is in the Handler field



-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

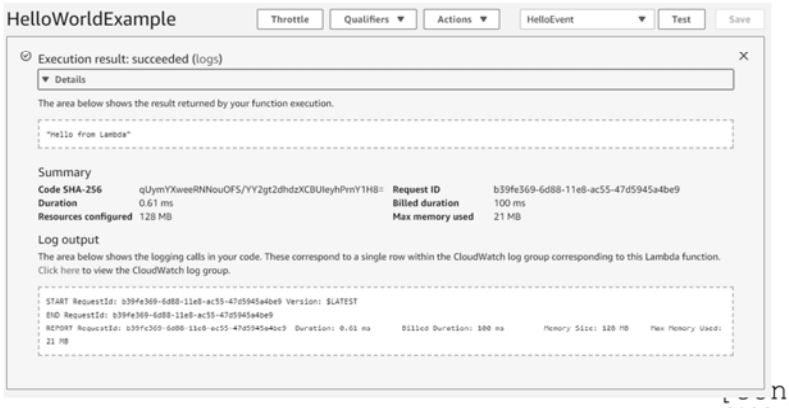


©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

Run the Test

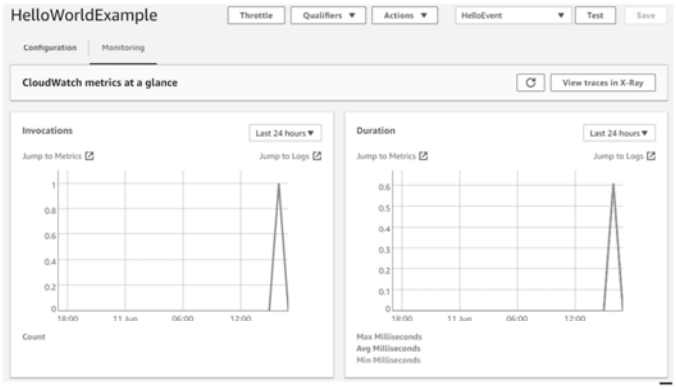
- To run the test, click **Test** again, and this time it will run the function



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

View the Monitoring

- You can easily see invocation metrics and logs by clicking on the **Monitoring** tab



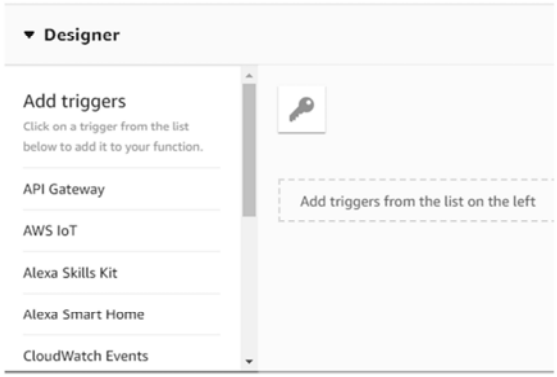
-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

Adding an Event Trigger

- Triggers can be added graphically from the list of possible triggers in the Web console



-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

A Scheduled Event Example

- Scheduled Events can be the basis of a trigger
  - Same time every day
  - Every x minutes or hours

- Scheduled events come from Cloudwatch
  - Schedules are specified using the **cron** standard

CloudWatch Events

OnHourAndFiveMinsPast
arn:aws:events:eu-west-1:963778699255:rule/OnHourAndFiveMinsPast
Schedule expression: **cron(0,5 * ? * MON-FRI *)** Description: **On the hour and 5 minutes past the hour**

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda

A Bucket Upload Example

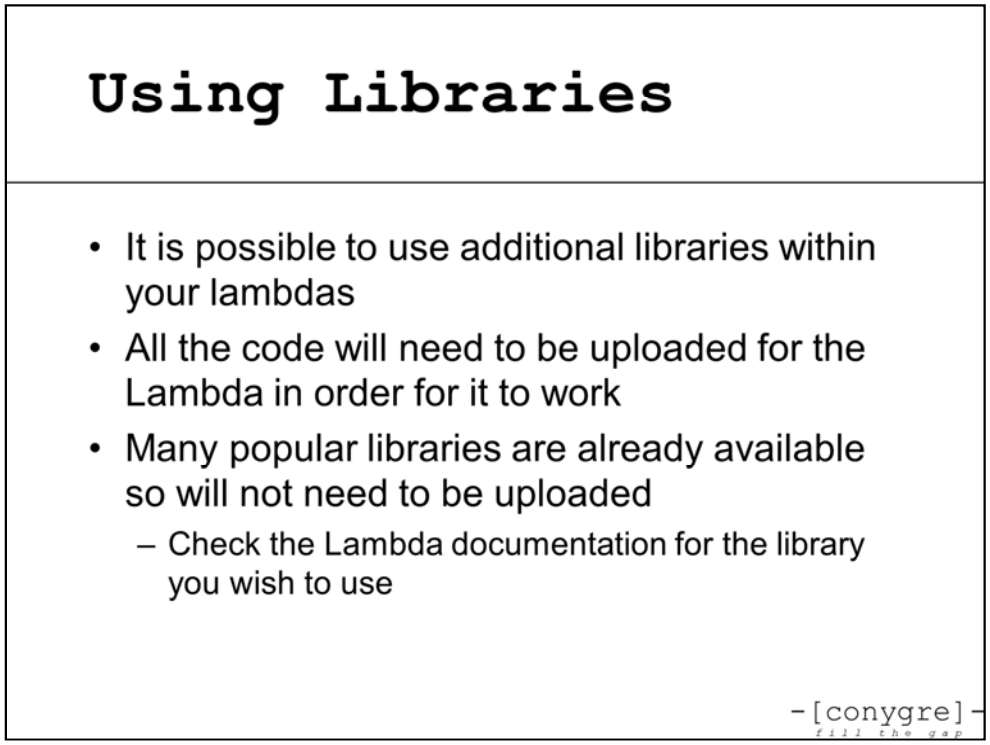
- Lambdas can be triggered by uploads to buckets
  - Notice the permissions on the right are S3 and the trigger on the left is S3

S3

lambda.conygre.com
arn:aws:s3:::lambda.conygre.com
Event type: **ObjectCreatedByPut** Notification name: **e82bea8e-5ceb-4f35-9aa5-43e69820b904**

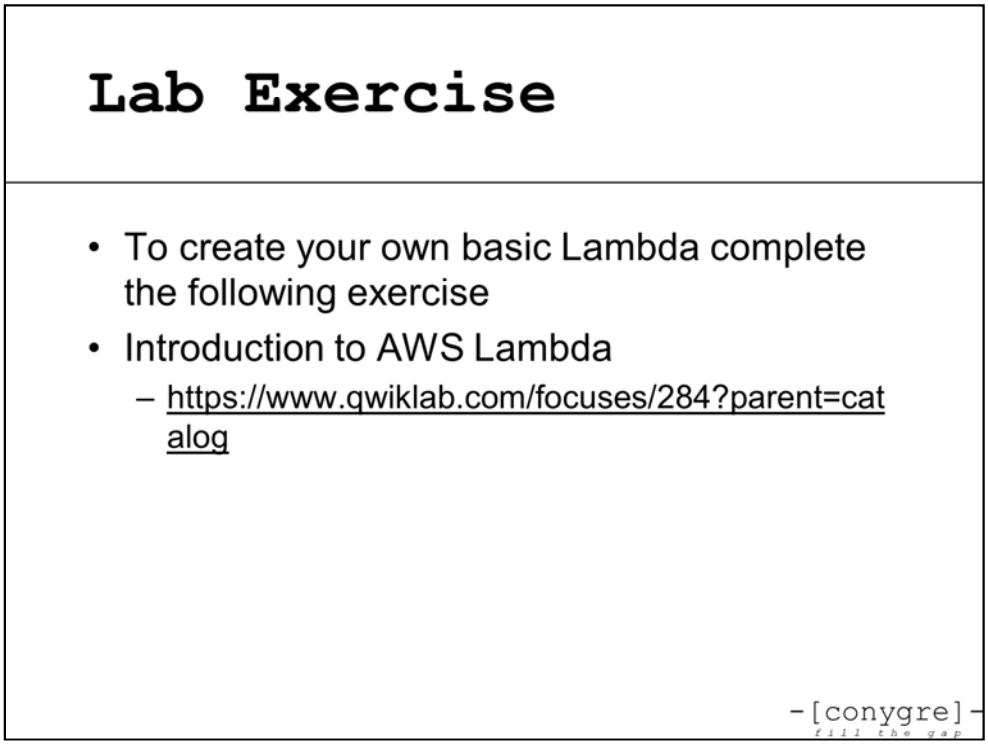
-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited



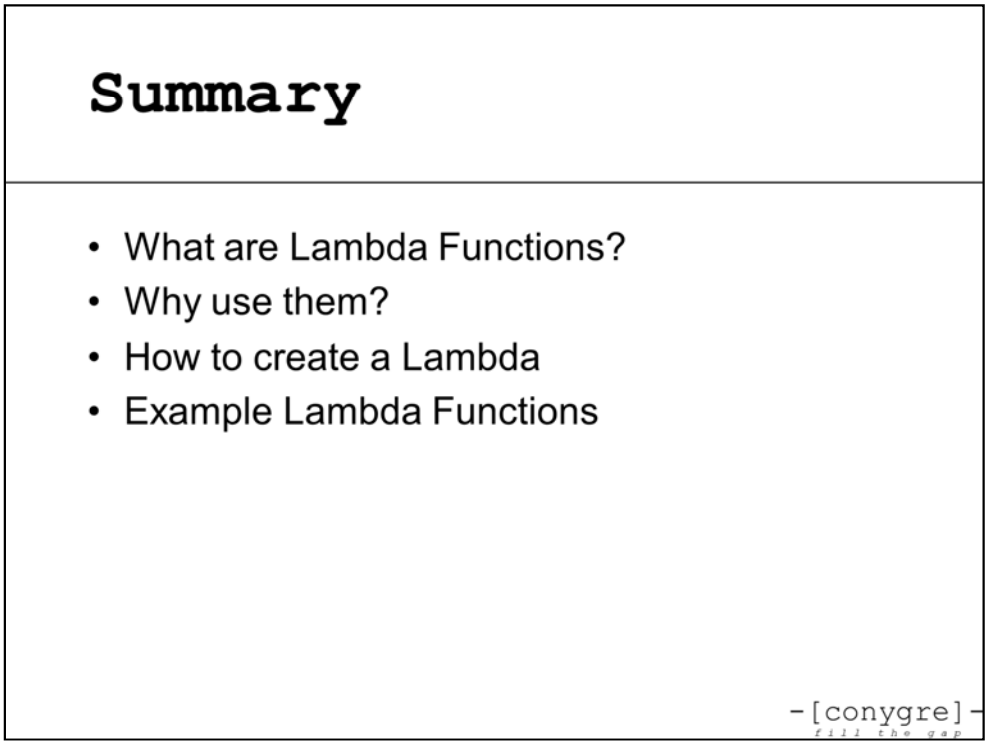
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introduction to AWS Lambda



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Objectives

- Where can Lex be deployed
- Deployment to a Web UI example

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

ChatBot Deployment Options

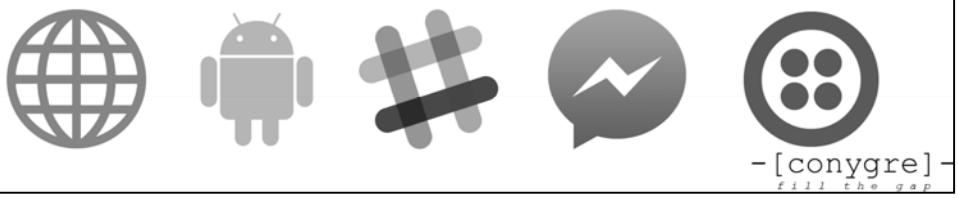
- ChatBots built using Lex have a variety of deployment options
- Alternative ChatBot technologies will have a similar range of choices

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Deployment Platforms

- Lex can be deployed to any of the Following Platforms
  - Web UI
  - Android
  - Slack
  - Facebook Messenger (Text Only)
  - Twilio (SMS api)



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

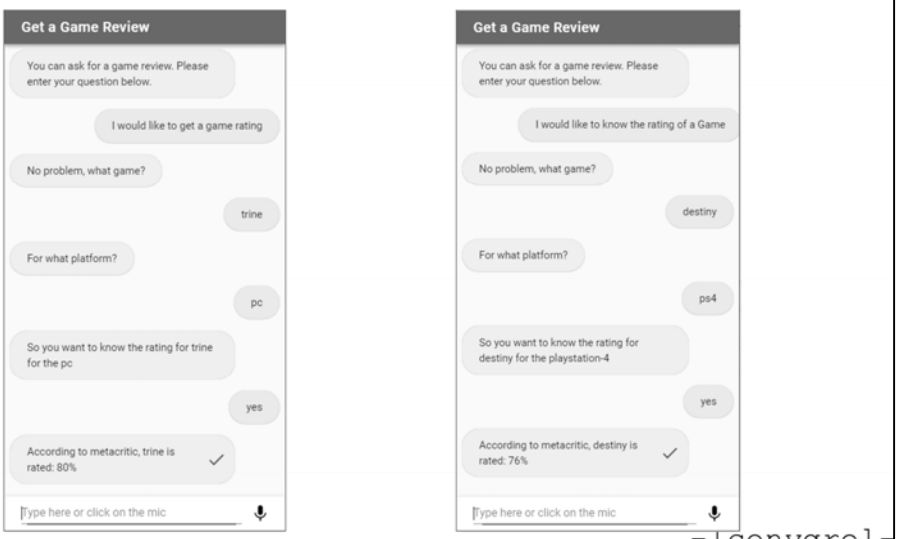
Basic Web Deployment

- A deployed Bot can be programmatically accessed and tested for Web sites
- The deployment process for Lex is very streamlined, so you can easily make updates to your bot and redeploy
  - This facilitates behaviour driven development – more later

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Web UI - Example



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Deployment Using CloudFormation

- CloudFormation is a service from Amazon to allow you to deploy entire infrastructures using a script
  - The details are beyond the scope of this course
- During the chapter you will see the fundamentals of how you can use it to successfully deploy your chatbot
- Full details can be found in this Amazon Blog post
  - https://aws.amazon.com/blogs/machine-learning/deploy-a-web-ui-for-your-chatbot/

-[conygre]-
fill the gap

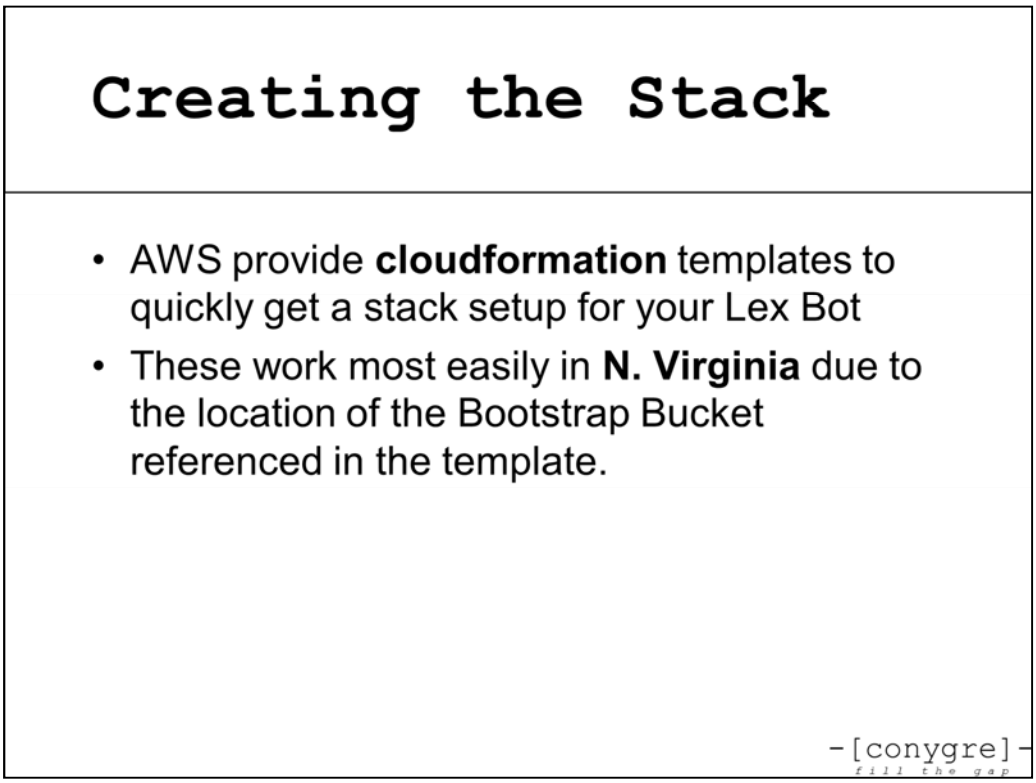
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

The Deployment Process

1. Cloudformation is used to create the Stack which the Bot will be hosted on
2. Permissions are set up in Cognito to allow the anonymous user to make requests
3. A set of pages and scripts are then created for you by the cloudformation template
4. A provided script snippet can then be pasted into your website

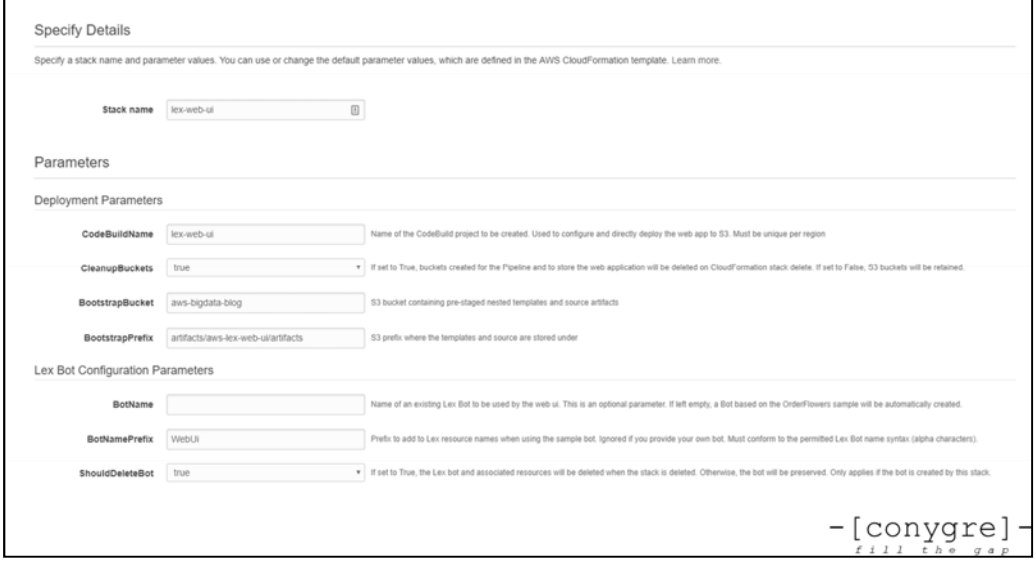
-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

The CloudFormation Parameters



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Managing Cognito

- Cognito is a service typically used on AWS for user authentication, often for mobile applications
- When a user interacts with your bot, it will need to have permissions specified
- Lex Bots run the the permissions of an **Unauthenticated Cognito User**
- In **IAM** we must ensure that a **Cognito Identity Pool** is available where the unauthenticated user has **Lex Post** permissions

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

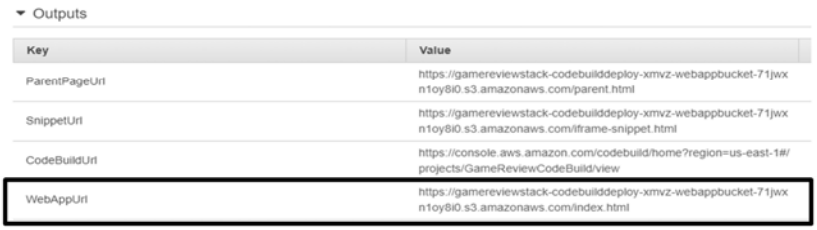
Managing Cognito

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Using the ChatBot from HTML

- Once the template has successfully run, a set of Web pages are created within an S3 bucket
- The **URLs** to these files are outputs from the cloudformation template



-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

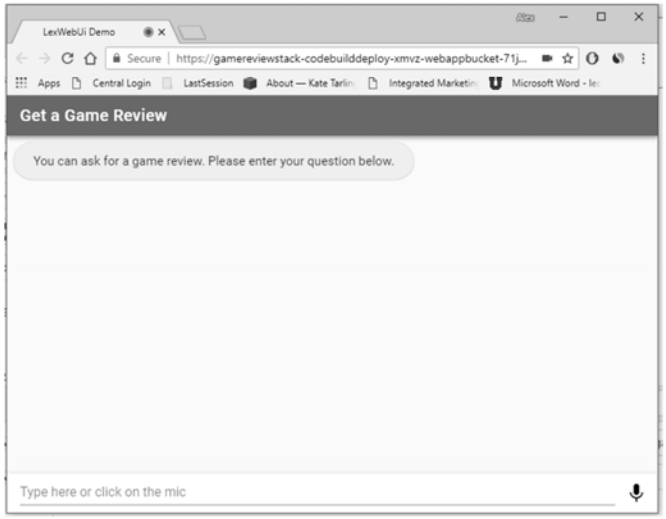
Using your ChatBot in a Web Site

- And the **code snippet** to implement it into your **own webpage** is found **here** too

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Accessing the ChatBot



-[conygre]-
fill the gap

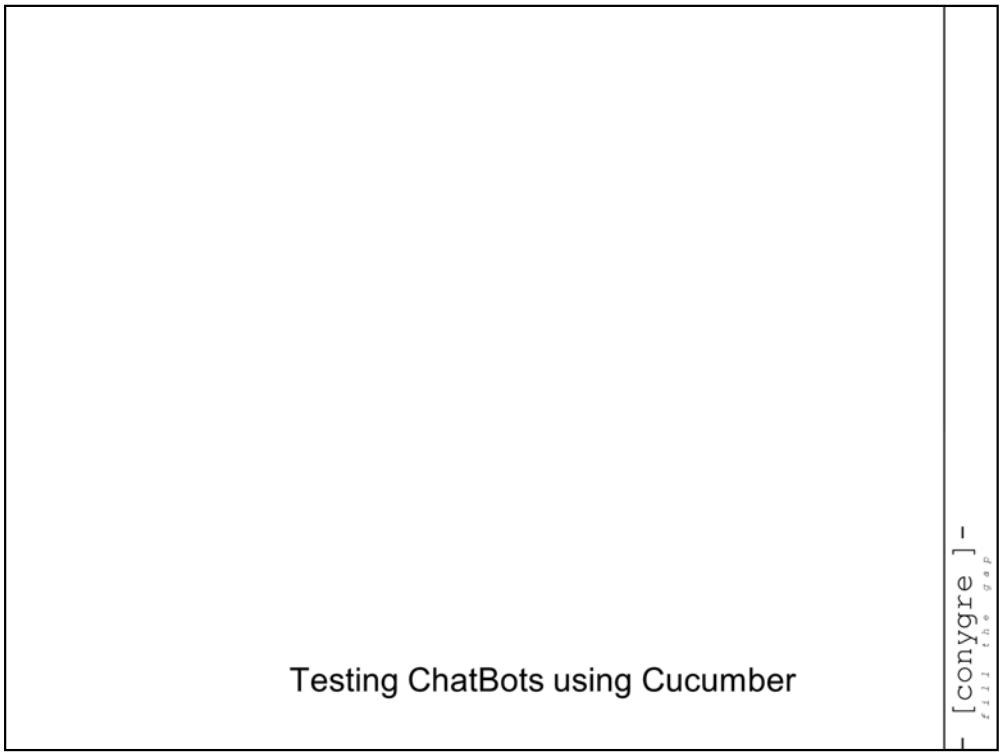
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Summary

- Where can Lex be deployed
- Deployment to a Web UI example

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Objectives

- What is Behaviour Driven Development (BDD)
- The Benefits of BDD
- The Relationship between BDD and Other Agile Practices
- What is Cucumber?
- Defining Features and Scenarios
- Linking Features and Scenarios to Tests
- Testing a ChatBot Example
- When to Run Tests

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

What is BDD?

- Behavior-driven development (BDD) is a software development methodology in which an application is specified and designed by describing how its behavior should appear to an outside observer.
  - *techtarget.com*

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Benefits of BDD

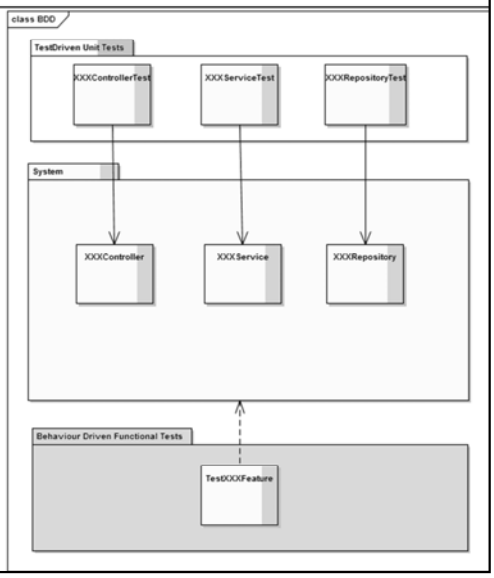
- Automated functional testing
- Completeness of functional testing
- Application only does what is required
- Behavioural requirements are captured in a format understood by technologists and business people alike
- Test cases are defined up front and the requirements are also the test cases

-[conygre]-
fill the gap

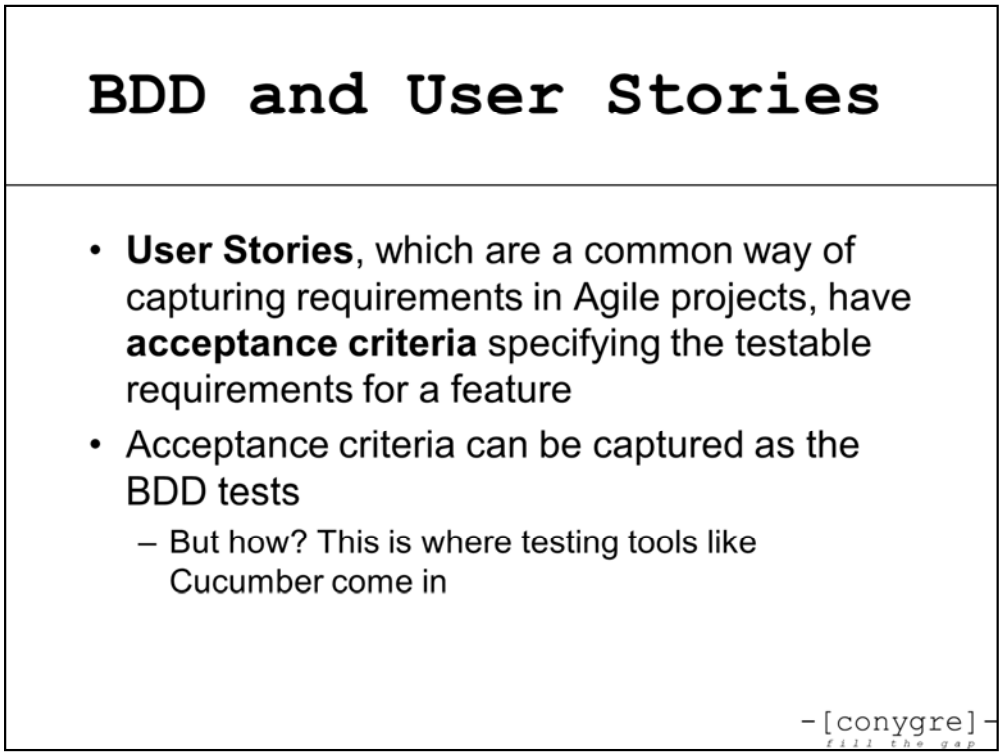
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Relationship with TDD

- Behavioural tests are written first (BDD)
- The system is then built using Test Driven Development
- At the end, you have two sets of **automated** tests
  - Unit tests (TDD)
  - Acceptance tests (BDD)



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Introducing Cucumber

- Cucumber is a tool that facilitates the creation and running of BDD tests
- Cucumber supports many different languages and platforms including
  - Java
  - C#
  - C++
  - Scala
  - Python

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Cucumber Feature Files

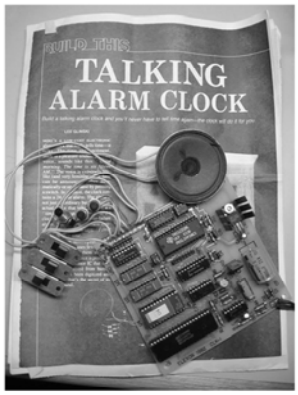
- The key artefact in Cucumber is the **Feature File**
- The Feature file specifies a **feature** as a series of **scenarios** which provide ***examples*** of how the system must work
  - This results in ***example based specifications***

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Worked Example – Time To Text

- Consider the example of a Speaking Clock system that provides a text value for a given time of day
  - 12pm returns the text ”midday”
  - 12am returns the text “midnight”
- This can be captured in the form of a feature file



-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Feature File

- Below is a Feature file for our system

```
Feature: SpeakingClock
In order to get the time as text
I want to be able to pass in a time
So that I can get a suitable text value back

Scenario: ConvertMidnightToText
Given the time is 0 hours and 0 minutes
When I request the time
Then the result should be midnight

Scenario: ConvertMiddayToText
Given the time is 12 hours and 0 minutes
When I request the time
Then the result should be midday
```

SpeakingClock.feature

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Feature File Explained

- The top level Feature describes the overall functionality to be implemented

```
Feature: SpeakingClock
In order to get the time as text
I want to be able to pass in a Datetime
So that I can get a suitable String back
```

- This is not read by a machine, but will appear in your automated test results

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

The Scenarios

- Each Scenario follows a standard convention

Keyword	Purpose	
Given	Arrange the test	Arrange
And	Additional aspects for arranging the test (if required)	Arrange
When	Specify the action that is to be done that requires testing	Act
Then	Assert what you expect the result to be	Assert

- For those of you familiar with unit testing, you might recognise the sections as equivalents of
  - ***Arrange, Act, Assert***

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Scenario Example

- Below is an example scenario for out text to time engine

Scenario: ConvertMidnightToText
Given the time is 0 hours and 0 minutes
When I request the time
Then the result should be midnight

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Creating Feature Files

- Each feature will have multiple scenarios that capture examples of how the system must work
- Scenarios can be created by business people and technical people
- Those involved in the process agree terminology for the domain in which they are working **(domain driven design)**
  - The agreed terms are then used consistently in the scenario text (important for automation)

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

What about the Code?

- To run as an automated test against the system, a code artefact needs to be created
- These code artefacts are called **Step Flow Definitions**
- These can be written in one of the supported languages

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Step Flow Definition

- Below is the step flow definition for our example written in C#

```
public class SpeakingClockSteps   {
   [Given(@"the time is (.*) hours and (.*) minutes")]
   public void GivenTheTimeIsHoursAndMinutes(int hours, int minutes)  {
              // set up a clock
   }
   [When(@"I request the time")]
   public void WhenIRequestTheTime() {
         //request the time from the clock
   }
   [Then(@"the result should be (.*)")]
   public void ThenTheResultShouldBeTimeAsText(string expectedTimeAsText) {
         // assert what you expect to happen
  }
}
```

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Steps Explained

- Each step is processed by Given / When / Then attributes
- Each attribute contains a regular expression with a pattern matching the text in the scenario
- Parameter values in the text are placed in parentheses which are then automatically available as method parameters

```
[Given(@"the time is (.*) hours and (.*) minutes")]
public void GivenTheTimeIsHoursAndMinutes(int hours, int minutes)
```

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Within the Methods

1. Arrange the system so it is ready for testing

```
[Given(@"the time is (.*) hours and (.*) minutes")]
public void GivenTheTimeIsHoursAndMinutes(int hours, int minutes) {
    dateTime = new DateTime(2016,10,27,hours,minutes,0);
    speakingClock = new SpeakingClock();
}
```

2. Exercise the system to see what happens

```
[When(@"I request the time")]
public void WhenIRequestTheTime() {
  actualTimeAsText = speakingClock.GetTimeAsText(dateTime);
}
```

3. Verify the result

```
[Then(@"the result should be (.*)")]
 public void ThenTheResultShouldBe(string expectedTimeAsText) {
    Assert.AreEqual(expectedTimeAsText, actualTimeAsText);
}
```

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Finally - What was Tested

- Below is the simple class we were testing

```
public class SpeakingClock : ISpeakingClock
{
      public string GetTimeAsText(DateTime dateTime  {
         if (dateTime.Hour == 0 && dateTime.Minute == 0)
            return "midnight";
         else if (dateTime.Hour == 12 && dateTime.Minute == 0)
            return "midday";
         return null;
      }
   }
}
```

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Setting up Cucumber in C#

- In Visual Studio you can install the SpecFlow Cucumber tools
- In the .NET project then add the following references
  - TechTalk.SpecFlow
  - TechTalk.SpecFlowRunner
- You will then have wizards to help create the Feature files
  - The StepFlow class stubs can be generated automatically from the feature file

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Setting up Cucumber in Java

- Both IntelliJ and Eclipse have plugins and extensions for Cucumber
- Projects built with Maven can automatically run your cucumber tests

```
<dependency>
  <groupId>info.cukes</groupId>
  <artifactId>cucumber-java</artifactId>
  <version>${cucumber.version}</version>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>info.cukes</groupId>
  <artifactId>cucumber-junit</artifactId>
  <version>${cucumber.version}</version>
  <scope>test</scope>
</dependency>
```

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Java Step Flow Definitions

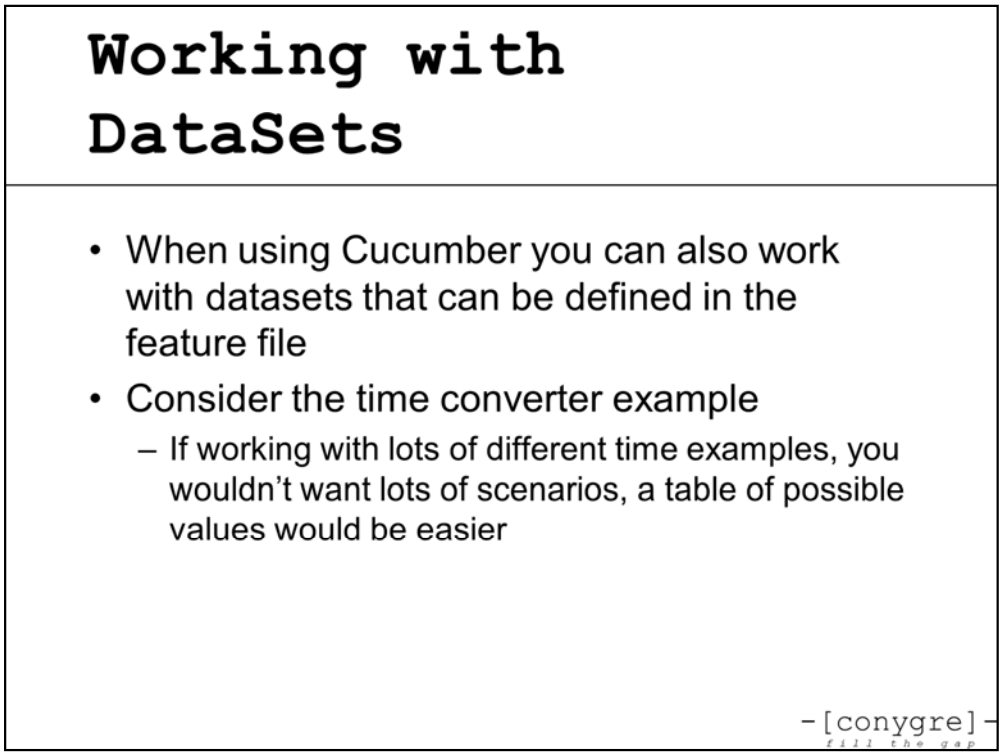
- Java step flows are almost identical to the C# ones except annotations are used instead of attributes

```
@When("^the customer searches for items published between (\\d+) and (\\d+)$")
public void setSearchParameters(@Format("yyyy") final Date from, @Format("yyyy") final Date to) {
  // act
}
```

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

DataSet Example C#

- Below is a scenario written with a set of data

```
Scenario: ConvertMultipleTimesToText
Given I have been provided this set of times and expected results

  | Hours | Minutes | ExpectedText |
  | 0     | 0       | midnight     |
  | 12    | 0       | midday       |

When I request all the times as text
Then all the text should match
```

- Create a C# class to match the table

```
public class TimeTable    {
   public int Hours { get; set; }
   public int Minutes { get; set; }
   public string ExpectedText { get; set; }
}
```

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Processing the Set

- To process the set, the method with the Given attribute is different

```
private List<TimeTable> timeTables;
[Given(@"I have been provided this set of times and expected results")]
public void GivenIHaveBeenProvidedThisSetOfTimesAndExpectedResults(Table tableOfTimes)
{
  timeTables = tableOfTimes.CreateSet<TimeTable>().ToList();
}
```

- SpecFlow provides a **Table** class that has methods to convert your table into a regular List<YourType> so long as the properties of the type match the table columns

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

DataSet Example Java

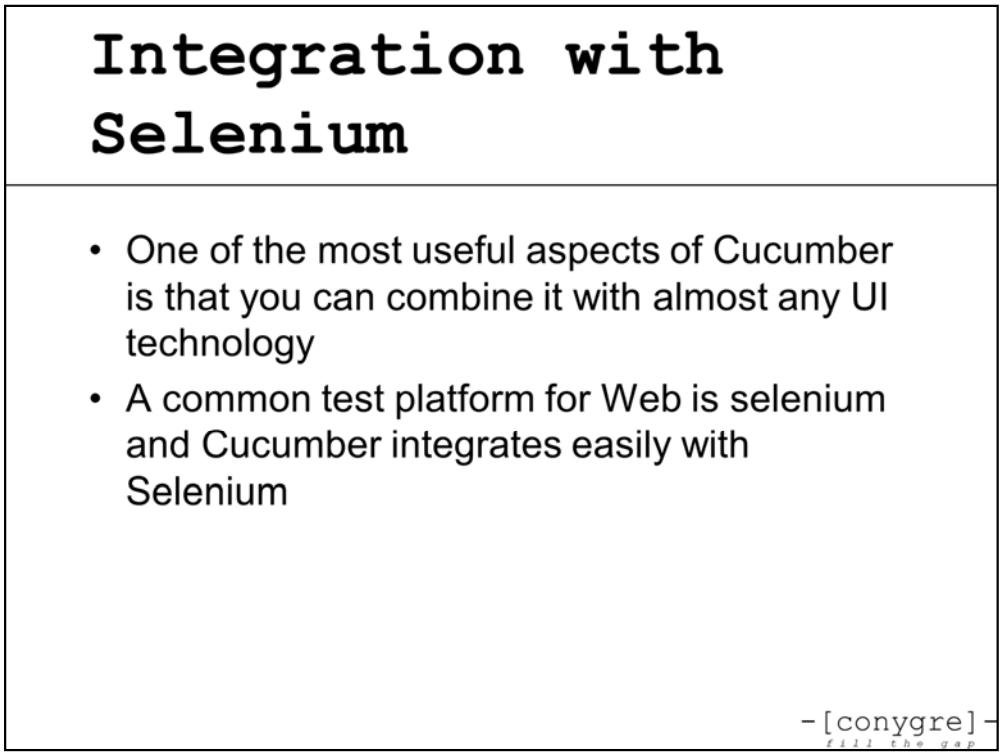
- Java is even easier than C#, just create a class matching your table with getters/setters
- Then have the @Given method to take in a parameter of a List<YourType>

```
@Given("^the system is initialized with the following data$")
public void initializeTheSystem(final List<TimeTable> timeTables) {
  // arrange
}
```

-[conygre]-
fill the gap

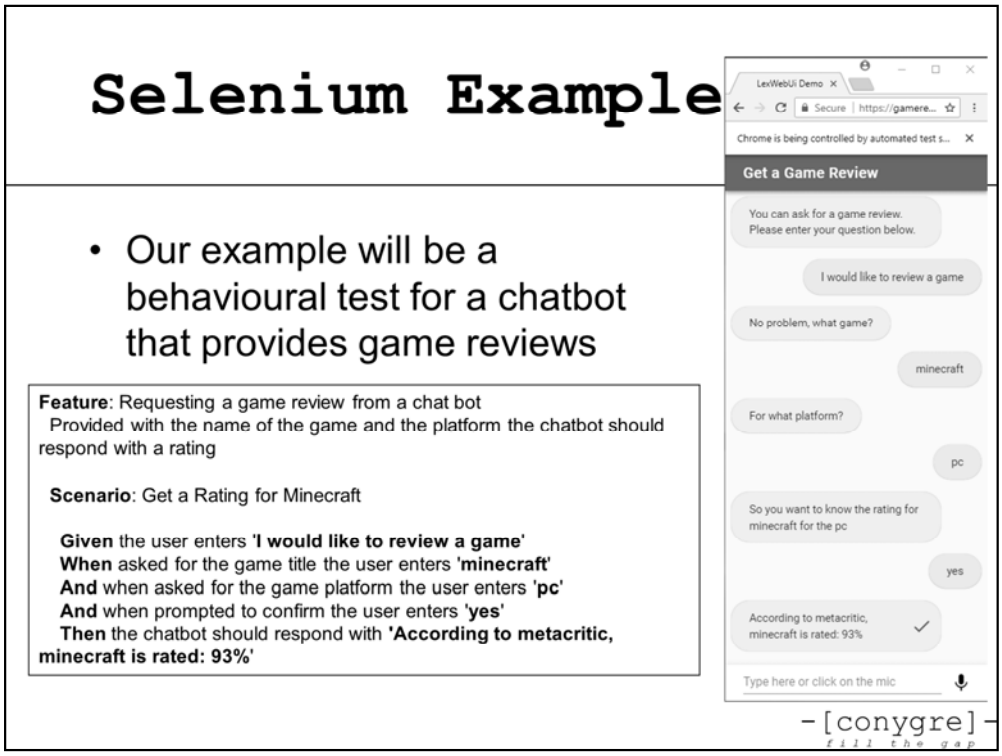
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber



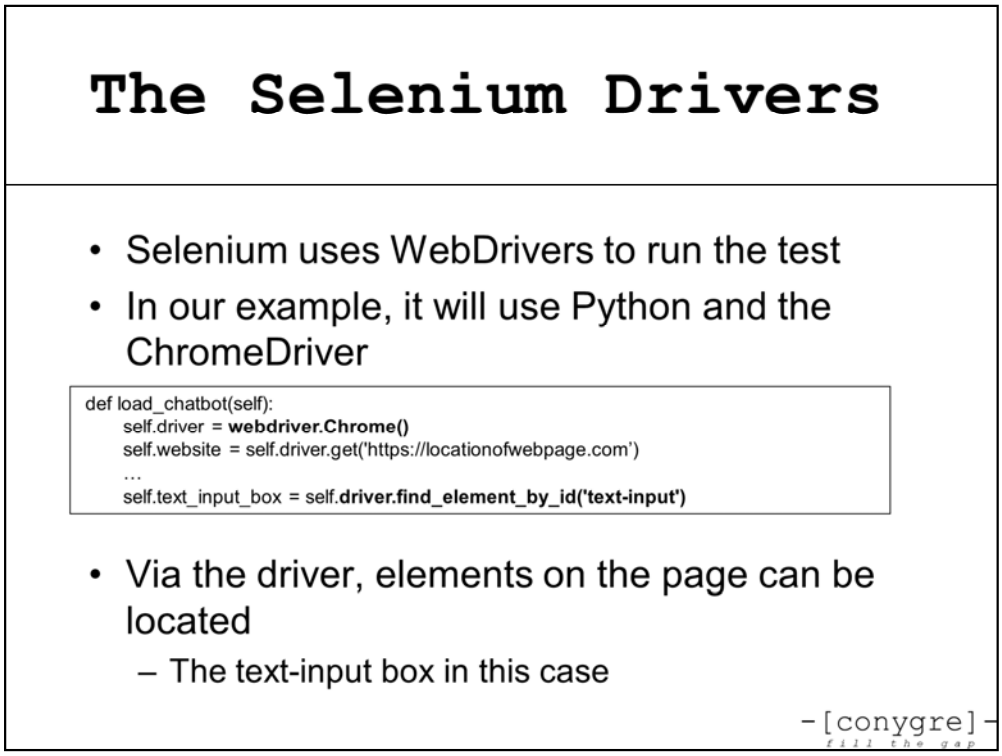
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber



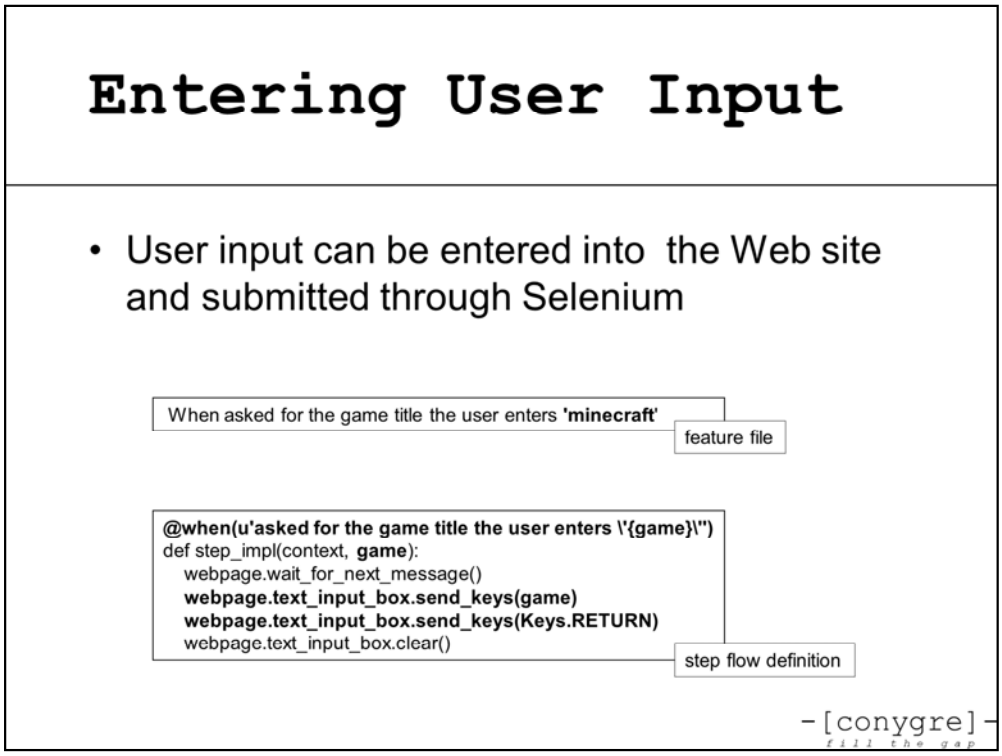
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber



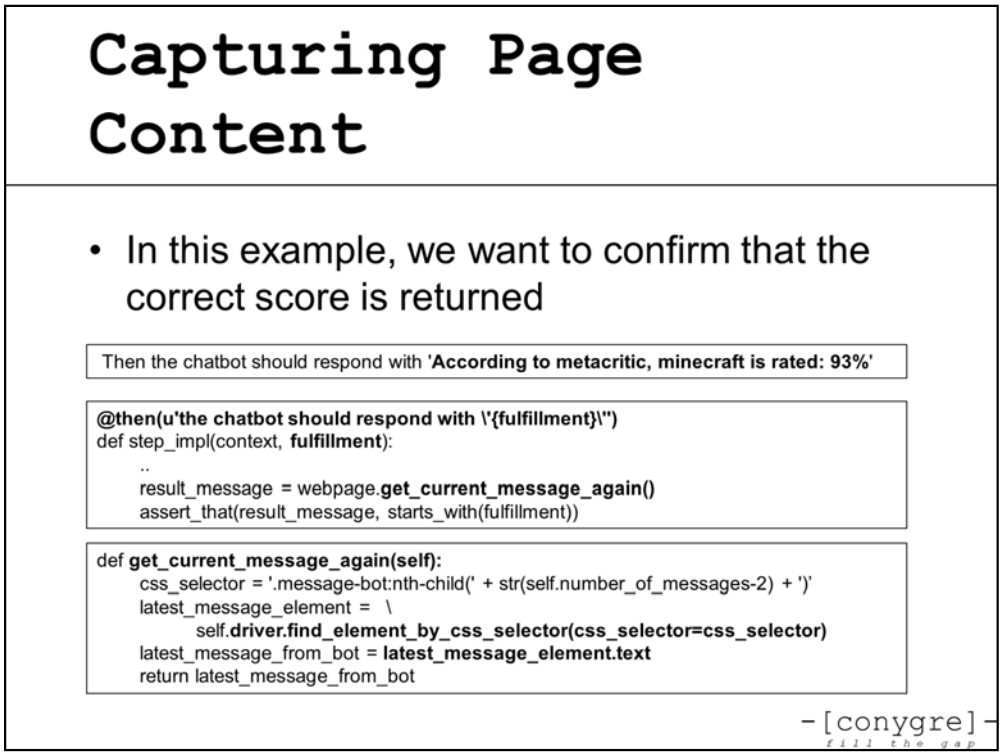
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber

ChatBot Example: Closer Look

- In the Python example we create a class as a singleton to represent the Web page hosting the chat bot

```
class WebPage:

    instance = None

    def __init__(self):
      pass

    @classmethod
    def get_instance(cls):
        if cls.instance is None:
            cls.instance = WebPage()
        return cls.instance
```

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber

Initialisation

- Our WebPage class has a function to initialise itself

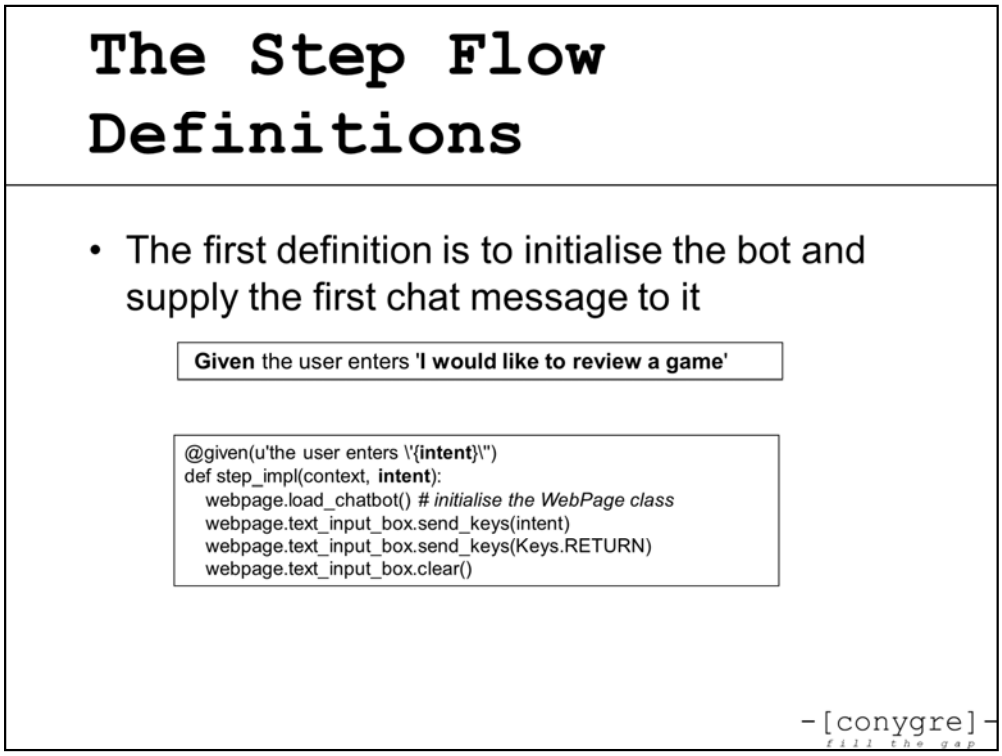
```
def load_chatbot(self):
    chrome_options = Options()
    chrome_options.add_argument("--use-fake-ui-for-media-stream") # options turn off prompt for mic
    self.driver = webdriver.Chrome(chrome_options=chrome_options)
    self.website = self.driver.get('https://gamereviewstack-codebuilddeploy-xmvz-webappbucket-
71jwxn1oy8i0.s3.amazonaws.com/index.html')
    self.number_of_messages = 1
    self.wait_for_next_message()
    self.text_input_box = self.driver.find_element_by_id('text-input')
```

  - Initialise the driver (turn off the mic prompt)
  - Set the Web page
  - Set the initial number of messages (1) this is increased as we use the chatbot
  - Wait for the initial message (see later)
  - Locate the input text box

-[conygre]-
fill the gap

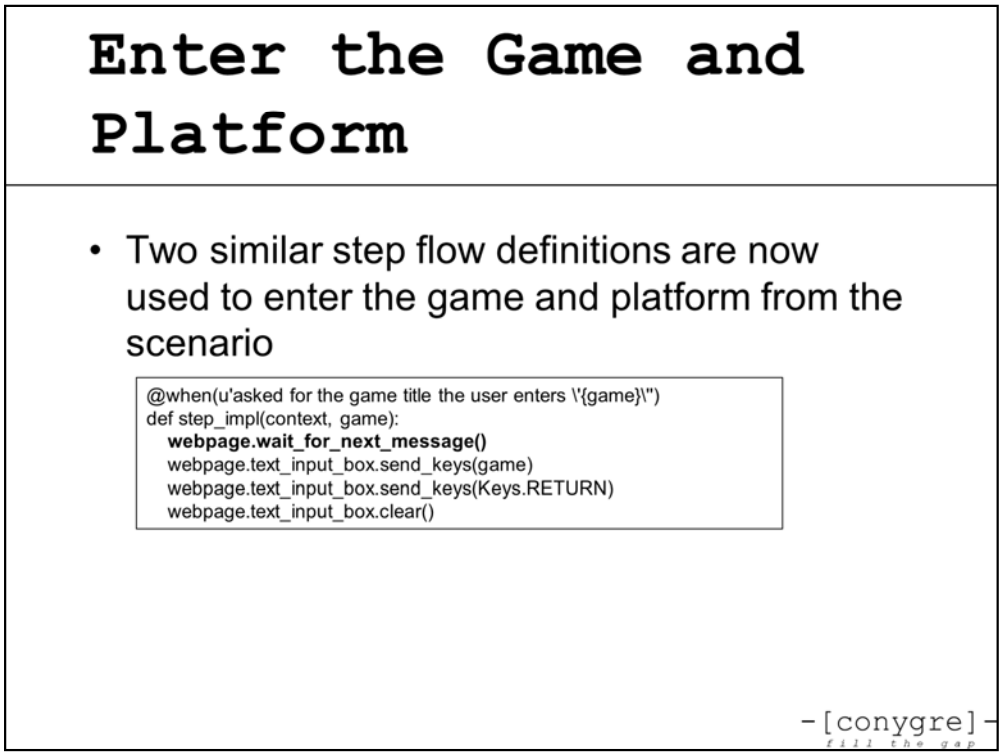
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber

Waiting for a Response

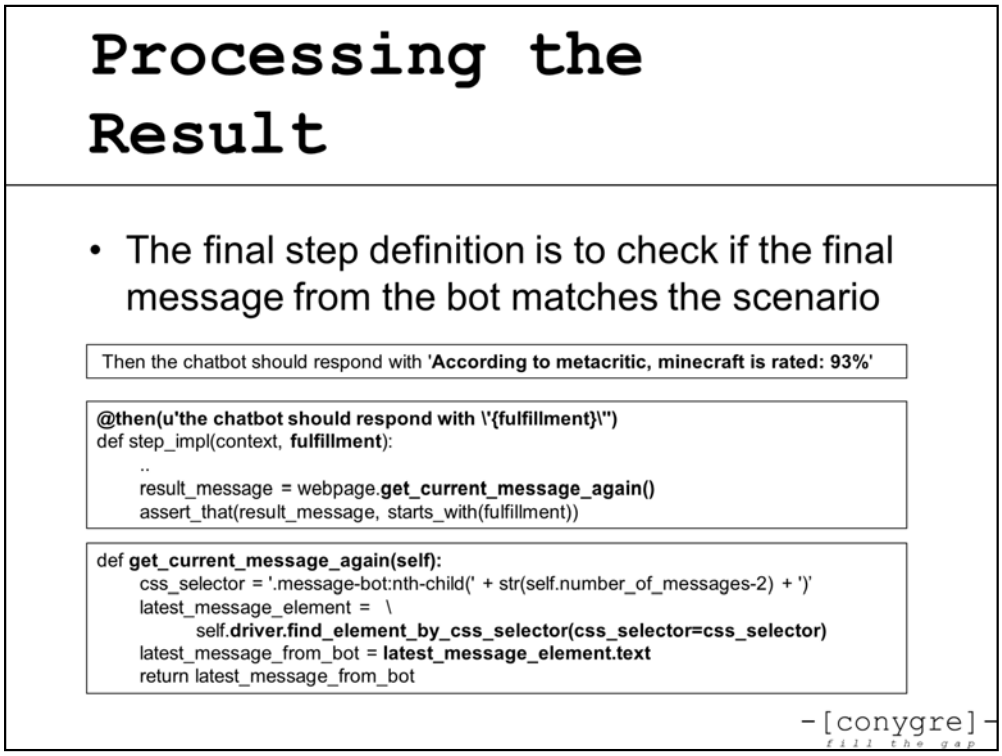
- A function has been added to get a response
  - The important bit is the CSS selector getting the next message from the bot

```
def wait_for_next_message(self):
    latest_message_from_bot = ''
    try:
        delay = 3
        css_selector = '.message-bot:nth-child(' + str(self.number_of_messages) + ')'
        WebDriverWait(self.driver,
delay).until(EC.presence_of_element_located((By.CSS_SELECTOR, css_selector)))
        time.sleep(2) # brief pause to allow the chatbot time to put some content in the field
        latest_message_element =
self.driver.find_element_by_css_selector(css_selector=css_selector)
        latest_message_from_bot = latest_message_element.text
        self.number_of_messages=self.number_of_messages+2 # 2 since every other is from human
    except TimeoutException:
        print("Loading took too much time!")
    return latest_message_from_bot
```

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber

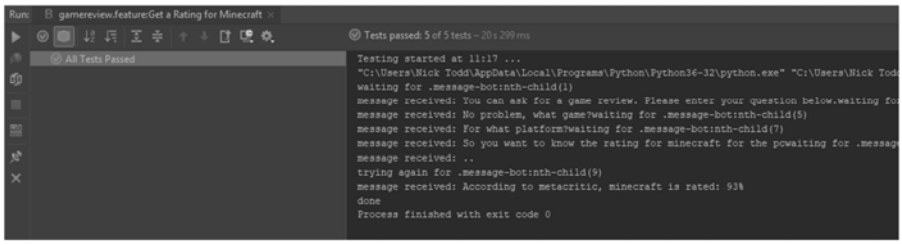


©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber

Running the Test

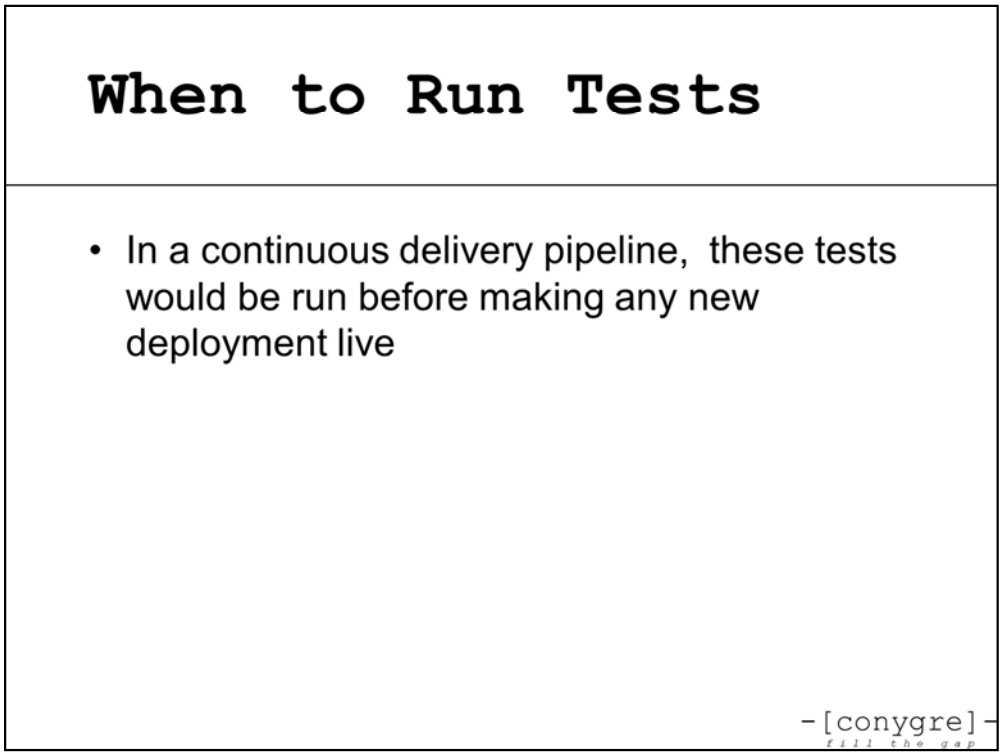
- Below is the test being run from IntelliJ



-[conygre]-
fill the gap

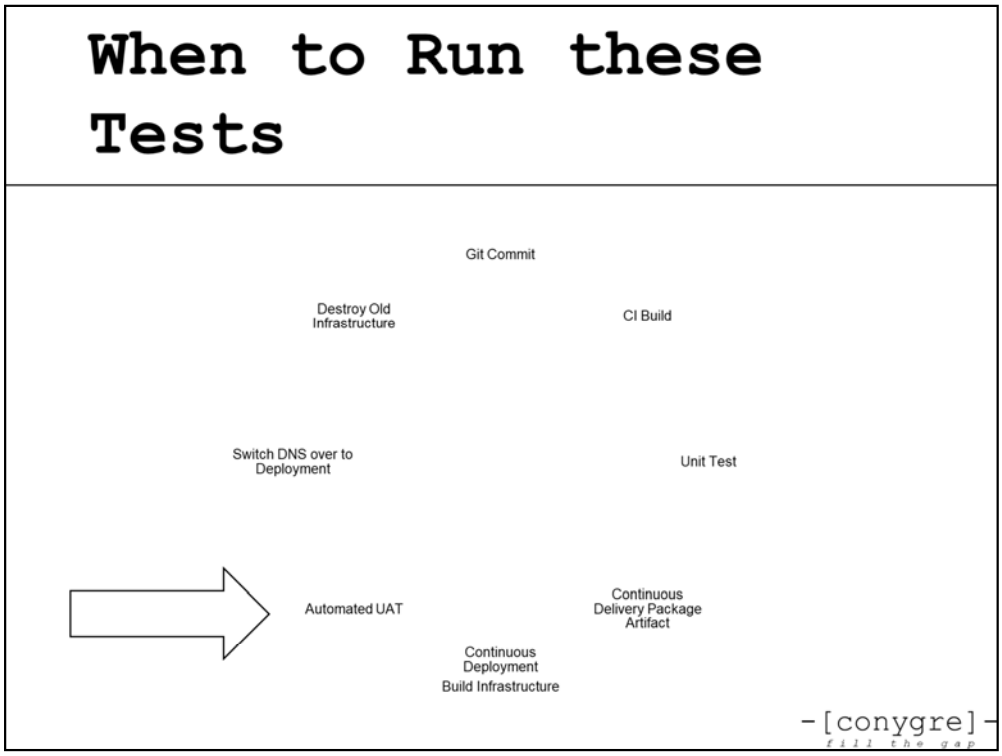
©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Testing Chatbots using Cucumber



©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Summary

- What is Behaviour Driven Development (BDD)
- The Benefits of BDD
- The Relationship between BDD and Other Agile Practices
- What is Cucumber?
- Defining Features and Scenarios
- Linking Features and Scenarios to Tests
- Testing a ChatBot Example
- When to Run Tests

-[conygre]-
fill the gap

©Copyright Conygre IT Limited Jun-18
Copying or reproduction of this material is prohibited without written authorisation from Conygre IT Limited

Amazon Lex

Course Labs Contents

Building a Basic Lex Bot

Aims

In this exercise you will create a basic Amazon Lex bot to handle a simple request from a user. The context for this lab will be getting a quote for car insurance.

Part 1 Logging in to Amazon Web Services

1. Go to http://aws.amazon.com
2. Click the '**Create an aws account'** button in the top right corner
3. Then click '**Sign in to an existing AWS account**'
4. Sign into AWS using the account credentials provided by the instructor

Part 2 Creating the Lex Bot

1. From the **AWS console** search for '**Lex**' in the **services** tab
2. In the top right corner, ensure that your Region is set to **N. Virginia**.
3. Click the blue **Create** button to create your first Lex bot
4. Select **Custom Bot** and give your bot a name. Use your first and last name in the Bot name so it does not get confused with other peoples' bots. (e.g. **`JohnSmithInsuranceBot`**)
5. You can also select an output voice if you would like your bot to use text-to-speech.
6. In the **Session Timeout** field, enter 5 minutes which will be plenty of time.
7. At the **COPPA** radio buttons select **No**.

Part 3 Creating your first Intent

1. Click the **Create Intent** Button
2. Remember that an **Intent** is the specific goal the user will want to achieve so name your intent **`GetQuote`**.
3. Add some **utterances** to the Intent in order to trigger it. E.g. 'I would like to get a quote'. (Remember you can have multiple **utterances** which trigger the same intent.)

Part 4 Creating the Slot Types

We need to get some information from the user in order to fulfil the GetQuote Intent. Assuming we are talking about car insurance, we will need to know:

a) The make of car
b) The model of car

c) The Engine size
d) The Date of Birth of the User
e) The year the user passed their test

Lex calls these pieces of information **Slots** which we need to fill.

We will first need to tell Lex the different types of information that the user could enter. We will need to create a **slot type** for each piece of information a-e.

For some of the slot types in this section, it will be easier to use the python script provided in order to generate files to upload to Lex to create the slot types for you. You don't *have* to use this script as it is possible to enter the information in manually but getting familiar with it will prove a time saver later on.

1. In the slot types tab on the left-hand side, click the blue + sign to add a type.
2. **If** you have used the python script, then you can click 'upload' and upload the zip file created by the script.
3. **Otherwise:**
    a. Click **Create Slot Type** and give the slot type a name. The first of these will be **`CarMakes`**.
    b. Add a description that briefly describes the slot. (e.g. **`available car manufacturers`**)
    c. It is best to select this field to **Restrict to slot values and synonyms** in this case. This is to ensure that the user is only able to select from the car manufacturers that our insurance company is able to deal with.
    d. Using a list from the internet or your own knowledge, go ahead an enter in all of the car manufacturers that you have patience to.

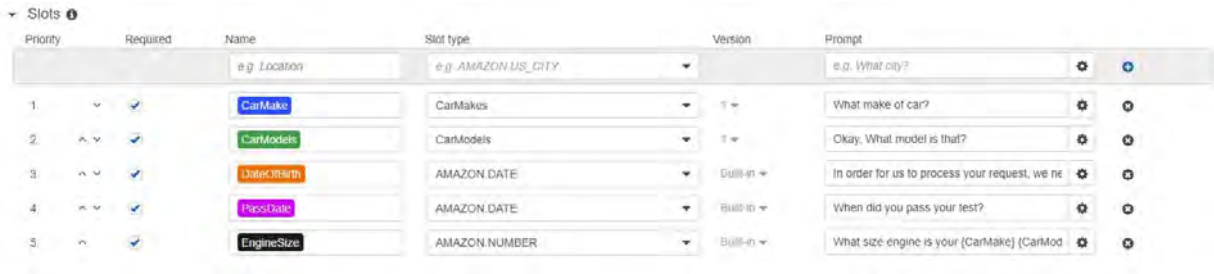
Note that for some of the Slot Types, amazon has built-in types. For example, amazon has a built in AMAZON.DATE type which we can use for the user's pass date and birthday.

4. Once you have created the **CarMakes** slot type, repeat the process for **CarModels**, **DOB**, **PassDate** and **EngineSize**.

Part 5 Creating the Slots

For each of the slots we defined we tell Lex how to prompt the user for that piece of information as well as what that information might look like.

1. In the Name section, type the name of the slot (This is case sensitive so try to stick to a convention) e.g. **`CarMake`**
2. In the slot type section, search for the slot type that fits the slot the best. For the **`CarMake`** slot this will be the **`CarMakes`** slot type you have created above.
3. In the Prompt section, type the question that Lex will ask the user, in order to get this piece of information. E.g. **`What make of car do you drive?`**
4. Do this for all of the pieces of information that you need for your bot and you should have something similar to below.



Part 6 Confirmation

It is good practice to ensure that the user knows that the bot has acquired all the information correctly. We can use the **confirmation prompt** tab to do this.

1. In the **Confirmation Prompt** tab, check the box to enable the confirmation prompt
2. In the confirm section we can write the confirmation message that will tell the client all of the information that the bot has acquired. We can link to the slots by using the following syntax: **`{SLOT_NAME}`** in the confirmation message.

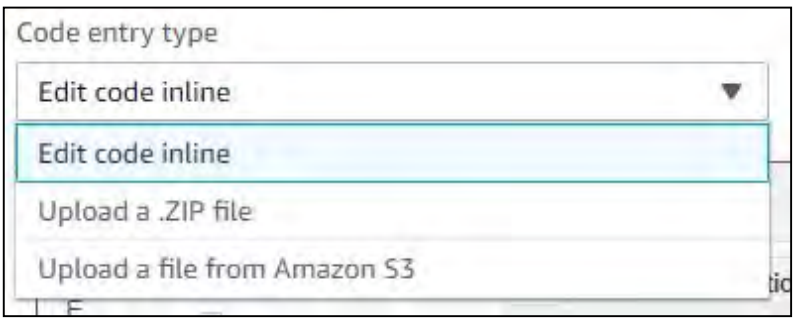
   Example: `'So you would like a quote for a {CarMake} {CarModel} {EngineSize} litre? And you were born on {DateOfBirth} and passed your test on {PassDate}?'`

3. In the cancel section, we can put a simple message to confirm to the user that their request will not be processed. E.g. **`Okay, no problem. See you again soon.`**

Part 7 Fulfilment with Lambda

Once Lex has elicited all the information from the user, we are now in a position to **do something with it.** This is where lambda comes in.

1. Ensure you've saved your intent in Lex
2. In the **AWS Services**, Search for **`Lambda`**
3. Ensure you're in the same region as your Lex Bot (N. Virginia)
4. At the Lambda homepage, click **Create** and then select **Author from scratch**.
5. Enter the name **`FirstNameLastNameInsuranceFunction`** and select **Python 3** as the runtime.
6. Assign it a role of **`lambda_basic_execution`**.
7. In the **Function Code** section, from the dropdown **Code entry type**, select **Upload a .ZIP file** and go ahead and upload the **lambda.zip** file from the Lex Labs directory.



8. Go back to the **Lex** console and in the **Fulfillment** teb, select **AWS Lambda Function**.
9. Find **your lambda** in the drop-down menu and make sure it's set to the **latest** version.

10. You won’t need to worry about the response tab in this lab as the response is handled by your lambda function.

Part 8 Finishing off

1. At the bottom of the screen, click **Save** to save your intent.
2. Click **Build** to build your Bot.
3. **Test it** using the testing panel on the right-hand side of the screen.

Deployment to Web UI

Aims

By the end of this exercise, you will have a fully-working chatbot deployed within a Web site. This will facilitate testing your chatbot programmatically as well as provide a way to implement your chatbot into your website of choice.

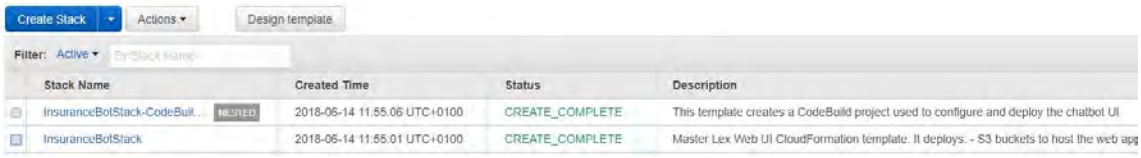
Part 1 Launching the ChatBot

Note: Additional documentation can be found here:
https://aws.amazon.com/blogs/machine-learning/deploy-a-web-ui-for-your-chatbot/

1. Follow the **link** to go to **cloudformation** with the correct template already populated and click **Next**
2. Enter a name for your stack **InsuranceBotStack**
3. Enter a code build name **InsuranceBotCodeBuild**

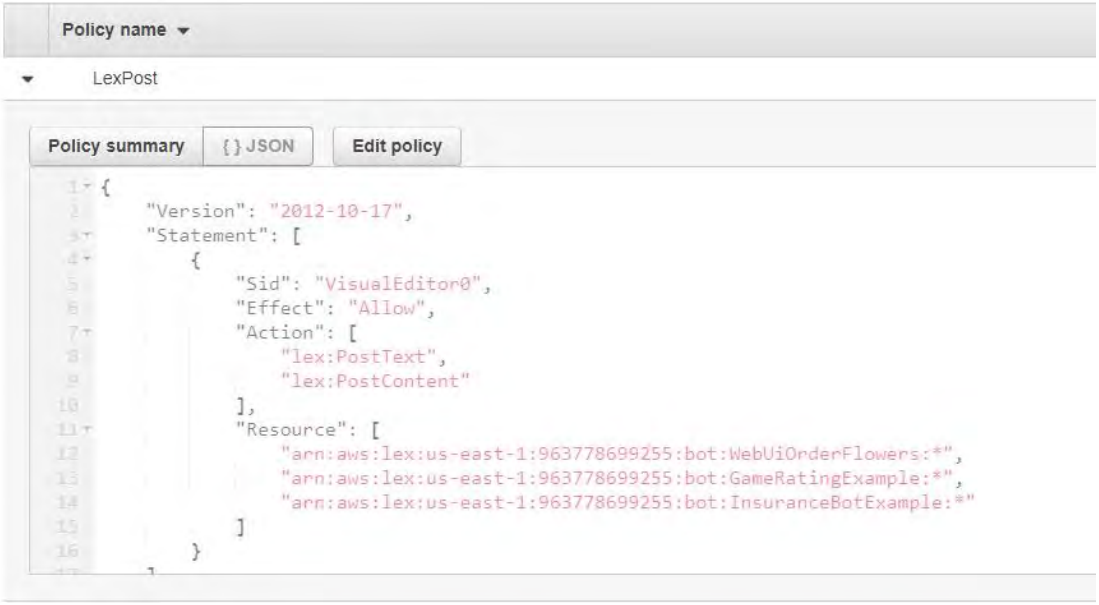
   IMPORTANT: *DO NOT CHANGE THE BOOTSTRAP BUCKET OR BOOTSTRAP PREFIX as these refer to standard locations where boilerplate code is going to be copied from*

5. Enter the **Bot Name** exactly as it appears in your Lex configuration.
6. Enter the **Cognito Identity Pool ID** if one is already set up.
   (You can get this from Cognito / Manage Pools / Click on the graph of the pool already there and copy its id)
   1. Set the WebAppConfBotInitialText to something appropriate. For Example: *'You can ask for an insurance Quote. Please enter your question below'*
7. Set the **WebAppConfBotInitialSpeech** to something like: *"say 'I would like to get a quote' to get started"*.
8. Set the **WebAppConfToolbarTitle** to 'Insurance Quote'.
9. Click **Next**.
10. At the **Options** dialog, click **Next**.
11. At the Review dialog, scroll to the end and tick the **I acknowledge that AWS CloudFormation might create IAM resources with custom names** option.
12. Click **Create**.
14. If successful you will see the following:



Part 2 Configuring the IAM roles for the servers

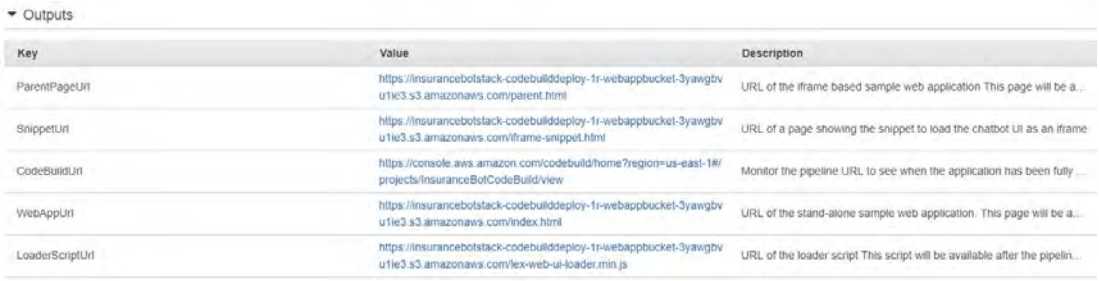
1. To ensure that the role that the **Cognito unauthenticated user** is assigned allows access to your chatbot, click **Services**, and then type **IAM**, right click on the IAM link and choose **Open in a new tab**.
2. In the IAM console, click Roles, and then find and then select the role with a name like:
   lex-web-ui-CognitoIdentityPool-1-CognitoUnauthRole-xxxxxxxxxxxx.
3. Expand the LexPost policy and then check that your LexBot is listed in the Allow section of the JSON



4. If it is not present, then click **Edit Policy**, and add into the list of resources, your Bots Amazon Resource Name (arn). It will be identical to the other present but with the different name on the end. Don't forget the :* which means any version of it.
5. **Save** the changes.

Part 3 Testing the Web UI

1. You are now ready to test your chatbot, go back to CloudFormation and click on the **InsuranceBotStack**, and then click **Outputs**



2. To do a basic test, click on the **WebAppUri** link to see your chatbot in action. You can try it out from here.
3. Finally, to test in your own Web page, click on the **SnippetUrl** link above which will show you the instructions for how to create your own Web page containing the chat bot.

4. Create a simple HTML file and add in the relevant snippet as per the instructions in the Web page.
5. Upload the page into the same S3 bucket as all the rest of your chat bot links are from and ensure that is set to **public visibility**.
6. Visit the page in a browser.

Appendix Writing a Lambda Function for Lex

Aims:

By the end of this lab you will have written a lambda function that will interface with the Lex bot you built in Chapter 1.

Part 1 Creating the function

This is the same as above:

2. Open Amazon Lambda from the AWS Console
3. Ensure that you're in the same region as your Lex Bot (N. Virginia)
4. Create a new lambda Function and choose 'Author from Scratch'
5. Give your Lambda a **name** and select the **Python 2.7/3** runtime (whichever you are most comfortable with)
6. Assign a suitable role- in this case probably (lambda_basic_execution) and click **create**

Part 2 Writing the code

This section will mostly be your **own code** written in **Python,** with only a small amount of guidance. You are free to either write your code in an **IDE of your choice** and upload it as a **.ZIP** file, or alternatively you can edit it from within the **built-in IDE** in amazon Lambda. Consider the following steps more as *hints* than as a step-by-step guide.

1. The function that will be called within your code is the **lambda_handler** function which takes in two parameters: an **event** and a **context.** Make sure you have this function.
2. The event that the **lambda_handler** is passes is in the following **JSON** format

```
{
  "currentIntent": {
    "slots": {
      "CarMake": "bmw",
      "CarModel": "m5",
      "DateOfBirth": "1975-5-16"
       "PassDate": "1997-8-12"
       "EngineSize": "5.0"
    },
    "name": "GetQuote",
    "confirmationStatus": "None"
  },
  "bot": {
    "alias": null,
    "version": "$LATEST",
    "name": "JohnSmithInsuranceBot"
  },
  "userId": "bijt6rovckwecnzesbthrr1d7lv3ja3n",
  "invocationSource": "DialogCodeHook",
  "outputDialogMode": "Text",
  "messageVersion": "1.0",
  "sessionAttributes": {}
}
```

3. Lex will receive back the response from the **Lambda_handler** function so you must ensure that it has the appropriate **JSON** format in order to **fulfill** the **intent**.
4. This format is best achieved using python **dict** notation (see below)

```
response = {
            "dialogAction":
               {
                "fulfillmentState":fulfillmentType,
                "type":"Close","message":
                  {
                    "contentType":"PlainText",
                    "content": result
                  }
               }
            }
```

Conygre IT Limited - Amazon Lex