Hubot

Sima Sharma

Introduction to Hubot

- Software tool and chat bot developed by Ryan Tomayko, a Github engineer, in 2011

- An open-source chat bot that helps automate DevOps workflows

- Performs actions to respond to users' commands

- Integrates into all popular chat platforms (e.g. Slack, Skype, Twitter)

Buzzwords

- **ChatOps**

Chatbots used by DevOps teams and developers who wish to create custom

tooling instead of consumer focused chatbots.

- **CoffeeScript**

A programming language that compiles to JavaScript. It adds syntactic sugar to

JavaScript to make the language more readable and user-friendly.

Goals of Hubot

- Originally designed so that anyone in GitHub could write new scripts for this automated system

- Performing tasks in an automated manner to make work more efficient

- Solving the problem of relying heavily on the availability of certain people to do specific tasks

- Solving gaps in communication by self-documenting

How Hubot works

- Is a repository and collection of executable scripts that responds specific strings in its chat clients

- Written in CoffeeScript on Node.js

- Automates tasks through CoffeeScript modules which are extendable by users and can be customized

- Hubot uses the adapter pattern to support multiple chat-backends

How Hubot works II

- Event driven, scripts define callbacks that should happen when an event occurs

Possible events	Callbacks that can occur
- Message in the chatroom - Private message to Hubot - Text pattern detected in a message - HTTP request	- Message in the chatroom - Reply to a message - HTTP response (if trigger was HTTP request) - New HTTP request - Executing a shell command - Executing something on a remote server

Hubot scripts

```
# Description:
#   Test script
#
# Dependencies:
#   None
#
# Commands:
#   hubot wiki search <query> - Returns the first Wikipedia article matching the search <query>
#
# Author:
#   Sima Sharma
```

Commands in the header show up when running **<hubot's name> help** to see the full list of commands

```
hubotadvice = ['Go for it!', 'You should think about it', 'Hmm... not too sure',
'You should call someone to discuss this with', 'Tell me more', ]

module.exports = (robot) ->

  robot.hear /(hi|hello|hey)/i, (res) ->
    res.send "Hi, my name is Hubot"

  robot.hear /(bye|see you|see ya)/i, (res) ->
    res.send "Goodbye, nice meeting you!"

  robot.respond /advice/i, (res) ->
    res.send res.random hubotadvice
```

An array of possible responses for Hubot's callback function

takes two arguments, a regex and a callback function which will be triggered if a match is found

robot.respond requires a message to begin with Hubot's name, while **robot.hear** reacts on any part of message

What you need to use Hubot

- Need to install Node.js, npm, hubot generator

- Can be deployed to Heroku (officially supported method)

- Can be directly configured with a chat client (most commonly used

  method in the workplace)

Use Cases

- Deploying releases (e.g. Github)

- Sending reports

- Search for information and images

- Obtaining a graph of desired information
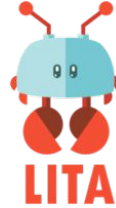
Strengths and Limitations

Strengths	Limitations
- Can integrate with existing IT workflows - Open-source - Platform-neutral, can integrate with many chat applications - Easily customizable - Reduces reliance on people - Reduces communication gap	- Installation process is not very straightforward - Bugs in user-created scripts can hinder productivity - Minimal documentation - Does not perform many tasks purely on its own - Security concerns due to the shared knowledge

Opportunities for Hubot

- There has been an increase in the adoption of Chatbots, particularly of Hubot, in the tech industry

- GitHub uses Hubot and other ChatOps bots internally to operate website and infrastructure, and this use is expanding across other companies

- Hubot can keep employees informed of internal operations and aid collaboration across cross-functional teams

Hubot and Competing Chatbots

Hubot	Lita	Cog
• Most widely used Chatbot • Commands are written in Coffeescript on node.js • Users can easily create their own scripts and run it on various chat platforms	• Commands are written in Ruby • Can be used to build operational task automations • Can be easily integrated into various chat platforms	• Commands are language and Chat platform agnostic • Uses a Unix-style pipeline to activate complex functionality • Well-documented installation guides and tutorials

Demonstration

Thank you