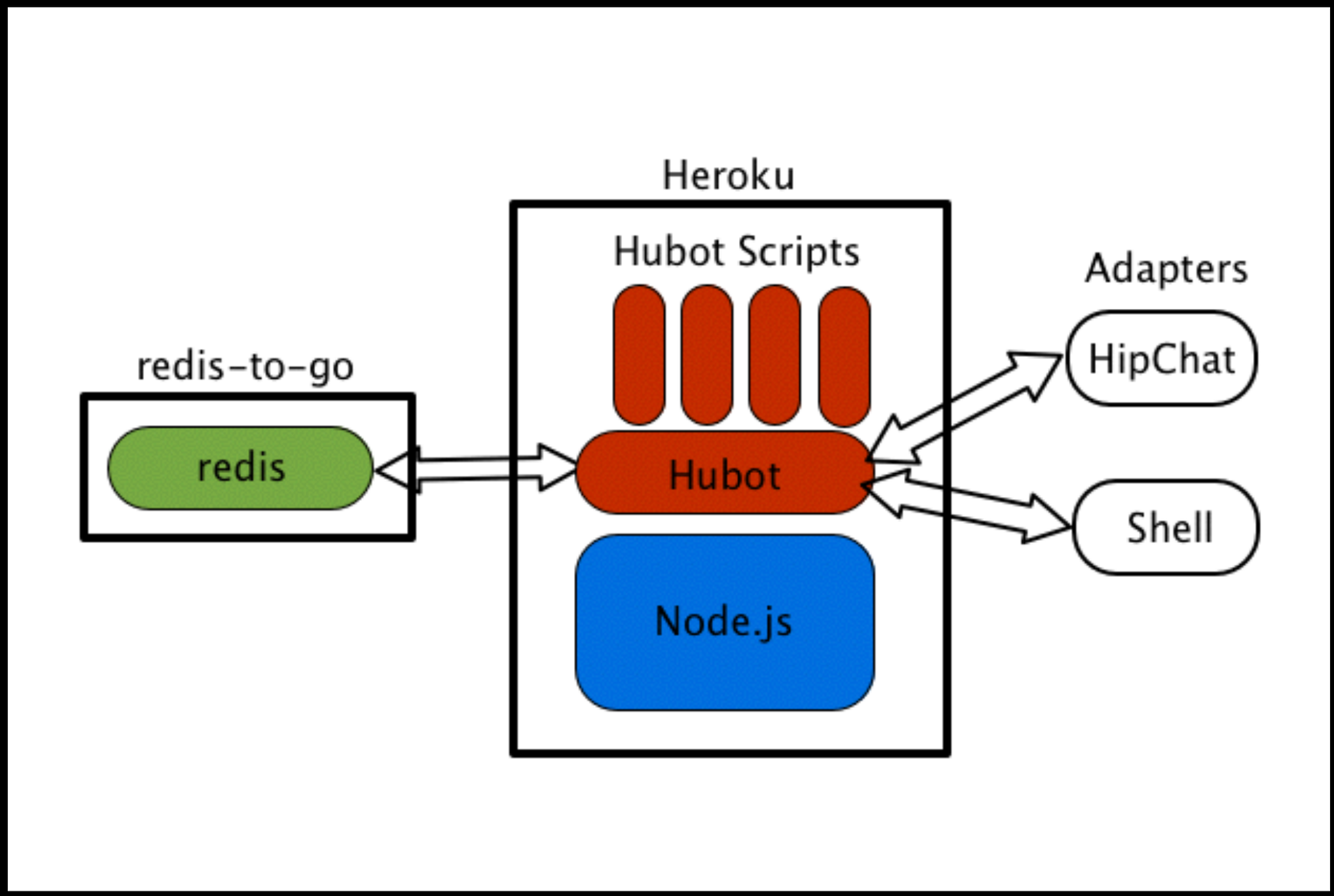
How do I Hubot?

Nate West

What is Hubot?

- Gif Summoner
- Reporter of information
- Annoying POS
- Whatever you want!


Heroku
Hubot Scripts
Adapters
HipChat
redis-to-go
redis
Hubot
Shell
Node.js

Node.js

- JavaScript runtime
- event-driven, non-blocking I/O
- Great for real-time applications across distributed devices (Chat!)

CoffeeScript

- Compiles down to JavaScript
- Mind the indentation!
- Similar idea to HAML/SCSS
- Can also use plain ol' JavaScript

Local setup

- `brew install node redis heroku-toolbelt`
- `git clone git@github.com:detroit-labs/hubot-badger.git`
- `npm install`
- `redis-server #in a separate tab!`
- `bin/hubot`

npm

```
{
  "dependencies": {
    "hubot": "2.11.0",
    "hubot-scripts": "2.5.16",
    "hubot-hipchat": “2.7.5",
    "github": "0.2.3",
    "promise": "6.1.0"
  },
  "engines": {
    "node": "0.10.30",
    "npm": "2.1.8"
  }
}
```

Hubot scripts

- Hubot provides a nice DSL
- respond or hear paired with a regular expression
- send messages back in text
- written in CoffeeScript

```coffee
module.exports = (robot) ->
  robot.hear /hello world/i, (msg) ->
    msg.send "hello world”

  robot.respond /botsnack/i, (msg) ->
    msg.send "Aw thanks!"
```

```
module.exports = (robot) ->
  robot.respond /say hi to (.*)/i, (msg) ->
    name = msg.match[1]
    msg.send "hello #{name}”
```

```
noises = [
  “Beep beep!”
  “BOOP!”
]
```

```
module.exports = (robot) ->
  robot.respond /poke/i, (msg) ->
    msg.send msg.random noises
```

```
# Description:
#   GitHub Project Status
#
# Dependencies:
#   underscore, github, promise
#
# Configuration:
#   GITHUB_API_KEY
#   GITHUB_ORG
#
# Commands:
#   hubot project status - Show github status of the project
#   hubot project repos - list repos of the project
#   hubot project repos add <repo(s)> - add repos to the project
#
# Notes:
#
# Author:
#   nwest
```

ENV

- Use environment variables to store API keys, secrets
- `bin/hubot` looks for a .env file for local development

```
githubAPI = require 'github'
github = new githubAPI(version: “3.0.0")

if process.env.GITHUB_API_KEY
  github.authenticate(
    type: "oauth",
    token: process.env.GITHUB_API_KEY
  )
```

redis

- Key/Value persistence
- No relational schemas
- Store whatever you want, make up a key
- hubot provides a getter and a setter
- could be room specific!

```
module.exports = (robot) ->
  robot.respond /taco champion/i, (msg) ->
    oldChampion = robot.brain.get(“taco-champion”)
    msg.send oldChampion

robot.respond /set champion (.*)/i, (msg) ->
  newChampion = msg.match[1]
  robot.brain.set(“taco-champion”, newChampion)
  msg.send “Congrats!”
```

```coffee
reposKey = (room) ->
  "repos-#{room}"

module.exports = (robot) ->
  robot.respond /repos/i, (msg) ->
    key = reposKey(msg.envelope.room)
    repos = robot.brain.get(key)
```

How?

- Hubot reads every message in the room
- Compares against every registered regular expression
- Fires all matches (Not first!)
- `.img that` vs `.imgthat`

External scripts

- Might already be written
  - old way: https://github.com/github/hubot-scripts
  - new way: https://github.com/hubot-scripts
  - http://hubot-script-catalog.herokuapp.com/

```
» cat hubot-scripts.json
[
  "redis-brain.coffee",
  "shipit.coffee",
  "applause.coffee",
  "bees.coffee",
  "github-status.coffee",
  "corgime.coffee",
  "ruby.coffee",
  "rdio.coffee",
  "haskell.coffee",
  "gosling.coffee",
  "kittens.coffee",
  "jenkins.coffee",
  "darksky.coffee",
  "giphy.coffee",
  "disassemble.coffee",
  "eight-ball.coffee",
  "lmgtfy.coffee",
  "emoji.coffee"
]
```

Deploying

- `git remote add heroku git@heroku.com:your-dyno.git`
- `git push heroku master`
- Need collaborator access!

```
» cat Procfile
web: bin/hubot -a hipchat
```

ENV on Heroku

- use heroku-toolbelt
- `heroku config`
- `heroku config:set foo=bar`

Questions?