Bot Telegram
Moto GP

Proyecto infraestuctura virtual

Realizado por Miguel Moral Llamas

GitHub: https://github.com/Miguelmoral/IV

Indice

* Como se hizo.
* Funcionalidad del Bot.
* Integración continua.
* Despliegue PaaS.
* Entorno de pruebas.
* Despliegue en un IaaS.

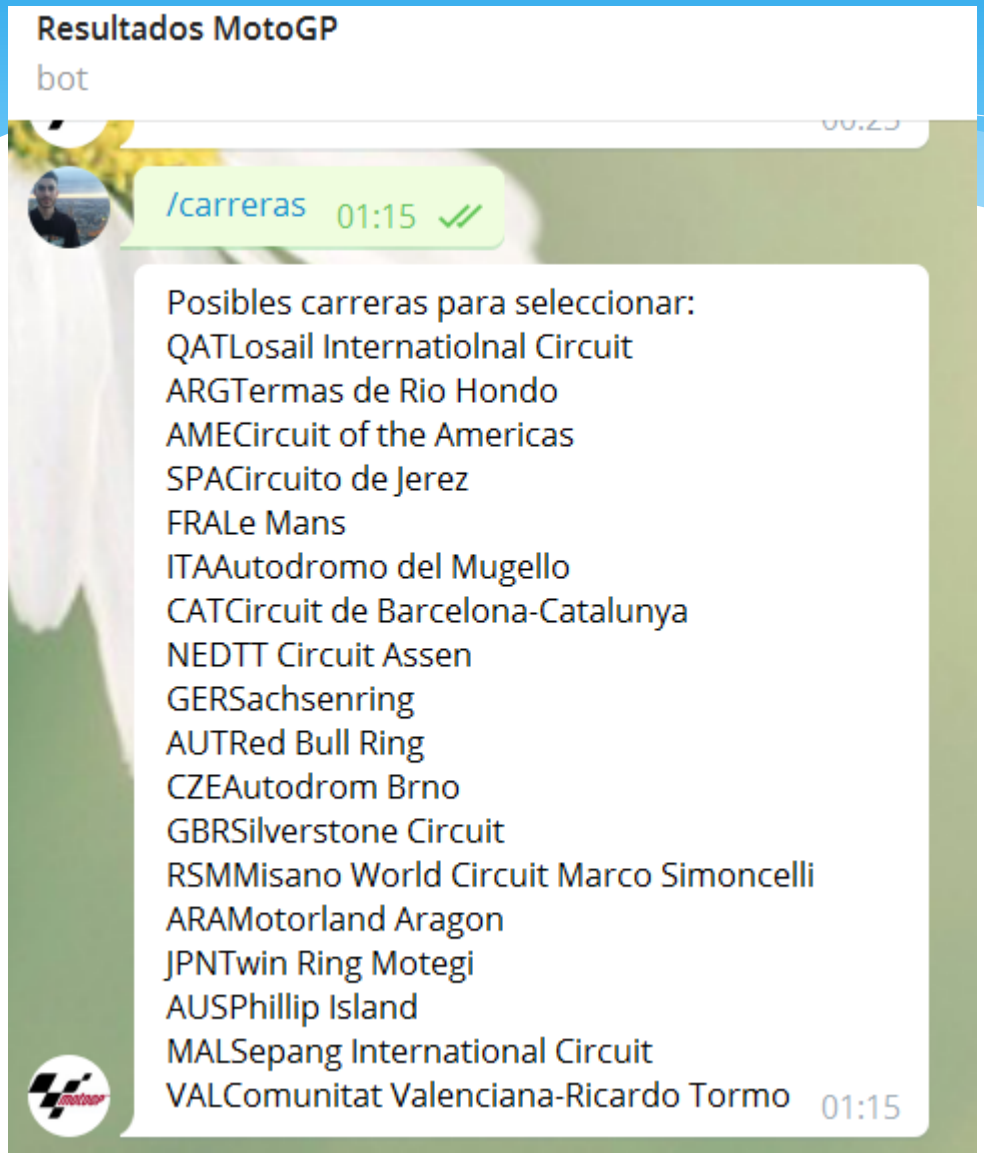
Como se hizo

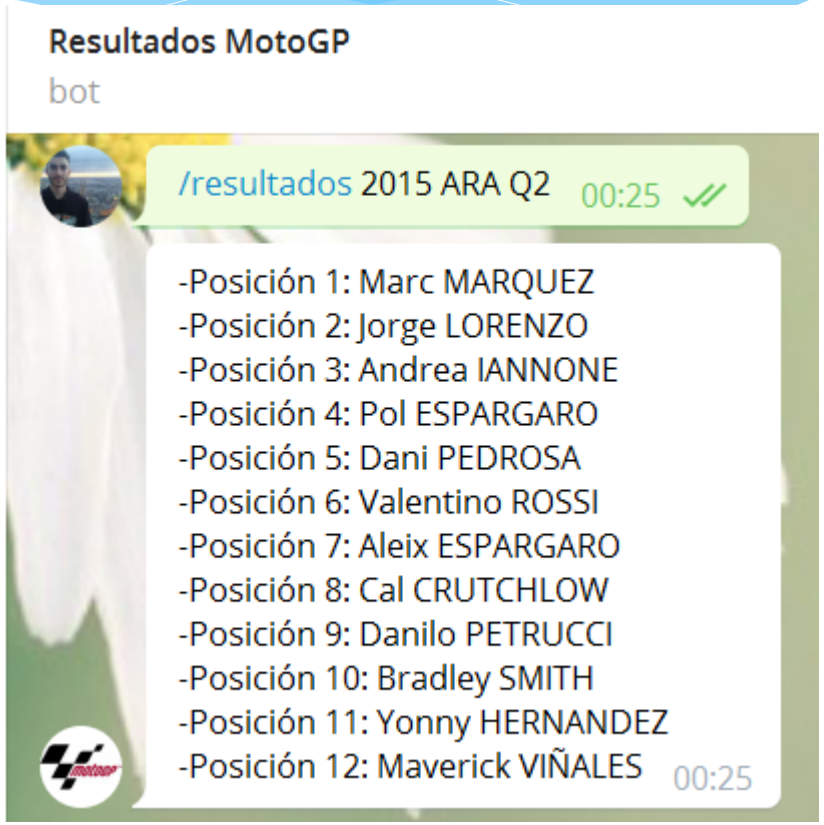
* Beautifulsoup (herramienta scraping).
* Python.
* TravisCI(Integración continua).
* Heroku(PaaS, BD postgresql).
* Docker(Entorno de pruebas).
* Azure(IaaS).

Funcionalidad Bot

* Comandos:
  * **/start** -> Ayuda con los posibles comandos del Bot
  * **/carreras** -> Listado con los códigos de las carreras
  * **/resultados año codigo_carrera sesion** -> Muestra los resultados de dicha carrera en concreto

Funcionalidad Bot





Integración continua

* **Travis CI.**
  * Conectar repositorio GitHub, regenera proyecto tras cada push build passing

Current Branches Build History Pull Requests

✓ master Añadiendo enlace bot

Commit 93c68d4

Compare 5e0b5a2..93c68d4

Branch master

Miguel Moral authored GitH

```
cd bot_motoGP && python tests.py
...
----------------------------------------------------------------------
Ran 3 tests in 0.586s

OK


The command "make test" exited with 0.

Done. Your build exited with 0.
```

Integración continua

* Archivo .travis.yml

```
branches:
  except:
    - gh-pages

language: python
python:
  - "2.7"

# command to install dependencies
install: make install

# command to run tests
script: make test
```

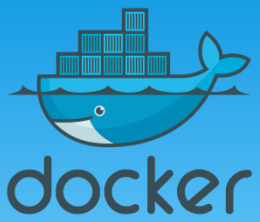
* Makefile

```
install:
        pip install -r requirements.txt

test:
        cd bot_motoGP && python tests.py

ejecutar:
        cd bot_motoGP && python bot.py
```

Despliegue PaaS

* **Heroku.**
* Base de datos creada en postgresql en Heroku a la que accedera nuestro Bot.
* Conectado al repositorio GitHub. (Necesita archivo procfile).

Entorno de pruebas



* **Docker Hub.**
* Necesario archivo Dockerfile.
* Estará enlazado con repo GitHub.

Docker Pull Command	Status	Actions	Tag
`docker pull miguelmoral/iv`	✔ Success		latest

Despliegue en un IaaS

* **Azure.**
* ¿Qué utilizamos para hacer funcionar el Bot en una máquina virtual en la nube?
  * Vagrant. (Crea MV)
  * Ansible. (Aprovisionamiento de la MV)
  * Fabric. (Conexión a la MV)

Despliegue en un IaaS

* **Vagrant.**
  * **Creación de la máquina virtual a medida gracias a un Vagrantfile.**
  * **Contenido más significativo Vagrantfile:**
    * **Credenciales Azure.**
    * **Playbook Ansible.**

Despliegue en un IaaS

* **Ansible.**
  * **Aprovisionamiento de la máquina creada con Vagrantfile con todo lo necesario para que el Bot se ejecute en la MV.**

```
    apt: name=libpq-dev state=present
- name: Clone
    shell: rm -rf IV && git clone https://github.com/Miguelmoral/IV
- name: Actualizar pip
    command: pip install -y -U pip
    command: sudo apt-get install -y python-dev
- name: Instalar requirements
    command: sudo pip install -r IV/requirements.txt
```

Despliegue en un IaaS

* **Fabric.**
  * Nos permite ejecutar comandos en la máquina que está alojada en Azure.
  * Comandos especificados en Fabfile.py (iniciar, demoniohub, descargar, detener, borrar, testear e instalar)
  * Utilizamos nohub para mantener el Bot en ejecución aún cerrando la terminal.

Despliegue en un IaaS

* Automatización del proceso de despliegue y ejecución del Bot con un sencillo script.

```
#!/bin/bash

sudo apt-get update

# Instalación de vagrant
wget https://releases.hashicorp.com/vagrant/1.8.7/vagrant_1.8.7_x86_64.debsudo dpkg -i vagrant_1.8.7_x86_64.deb
# Instalar plugin para azure
vagrant plugin install vagrant-azure

# Instalación Ansible
sudo apt-get install ansible


# Despliegue en Azure
sudo vagrant up --provider=azure

# Despliegue de la aplicación con Fabric
sudo pip install fabric
# Actualiza el supervisor
fab -p '0123456789Contrasenia!' -H miguel@botmotogp.cloudapp.net demoniohup
```

FIN