Ebanistería León

Manual de Despliegue en Heroku

Back-End y Front-End

Presentado por:

Michael Valencia

Marlon Campo

Gohan Holguín

Alexander Correa-Diaz

SENA – Centro de Comercio y Turismo

Armenia – Quindío

Junio 15, 2022

Back-End

Despliegue en Heroku

Requerimientos previos

* [Cuenta en Heroku](https://signup.heroku.com/login).
* Tener instalado [Git](https://git-scm.com/).
* La aplicación tiene que compilar sin errores usando dmd (dub build).

Paso 1: configuración de la aplicación

*//Ejecutar comando* Python manage.py collectstatic --noinput

Además, hay que crear un archivo Procfile, que es un archivo de texto en el directorio raíz de la aplicación que explícitamente declara qué comando se tiene que ejecutar para arrancar la aplicación.

El archivo Procfile de este ejemplo es algo como:

web: ./hello-world

Paso 2: preparación de la aplicación

Antes de avanzar hay que hacer *login* en la línea de comandos de Heroku usando las herramientas de [Heroku Toolbelt](https://toolbelt.heroku.com/standalone)

Esto proporciona acceso a la interfaz de línea de comandos de Heroku que se puede usar para gestionar y escalar las aplicaciones y los *add-ons*.

Después de instalar las herramientas (*Heroku Toolbelt*) hay que ejecutar lo siguiente:

$ heroku login

Paso 3: creación de la aplicación

Para crear una aplicación hay que ir al [panel de control de Heroku](https://dashboard.heroku.com/) y seguir los pasos para crear dicha aplicación. Hay que recordar el nombre de la aplicación creada, ya que será útil más adelante.

También se puede usar la línea de comandos en lugar del panel de control para crear una aplicación:

$ heroku create

Creating app... done, ⬢ leon-ebanisteria

https: *https://github.com/Ebanisteria-Leon/Ebanisteria\_Django.git*

Aquí el nombre de la aplicación es leon-ebanisteria.

Despliegue usando Git

Para desplegar la aplicación se usan directamente comandos de git. Se debería añadir un nuevo servidor remoto de Git donde se publiquen las versiones finales. Este nuevo servidor remoto tiene el nombre de la aplicación indicado en el punto anterior (en este caso es *rocky-hamlet-67506*).

$ heroku git:remote -a leon-ebanisteria

El nuevo remoto se añade a la configuración de Git:

$ git remote -v

Heroku https:*https://github.com/Ebanisteria-Leon/Ebanisteria\_Django.git (fetch)*

heroku https:*https://github.com/Ebanisteria-Leon/Ebanisteria\_Django.git (push)*

Añadiendo el *buildpack*

Los *buildpacks* son los responsables de generar recursos y/o código compilado.

Para obtener más información sobre los *buildpacks* se puede buscar en la [documentación de Heroku](https://devcenter.heroku.com/articles/buildpacks).

Para desplegar el *[buildpack](https://github.com/MartinNowak/heroku-buildpack-d)*[de Vibe.d](https://github.com/MartinNowak/heroku-buildpack-d) se usa el comando:

$ heroku buildpacks:set https:*//github.com/Ebanisteria-Leon/Ebanisteria\_Django.git*

Este *buildpack* usa de forma predeterminada la última versión de compilador de D. También es posible usar GCD, LDC o una versión específica de cualquiera de los compiladores añadiendo el archivo .d-compiler al proyecto.

En este archivo se puede usar dmd, ldc o gdc para seleccionar la última versión de cada compilador o se puede usar dmd-2.0xxx, ldc-1.0xxx o gdc-4.9xxx para seleccionar versiones específicas de cada compilador.

Despliegue del código

Para lanzar una nueva versión después de seguir los hábitos normales de escritura de código usando Git, basta con hacer *push* de la última versión al remoto de Heroku añadido en puntos anteriores.

$ git add .

$ git commit -am "Primer Release"

$ git push heroku master

Counting objects: 9, done.

Delta compression using up to 2 threads.

Compressing objects: 100% (6/6), done.

Writing objects: 100% (9/9), 997 bytes, done.

Total 9 (delta 0), reused 0 (delta 0)

-----> Fetching custom git buildpack... done

-----> D (dub package manager) app detected

-----> Building libevent

-----> Building libev

-----> Downloading DMD

-----> Downloading dub package manager

-----> Setting PATH:

-----> Initializing toolchain

-----> Building app

Running dub build ...

Building configuration "application", build type release

Running dmd (compile)...

Compiling diet template 'index.dt' (compat)...

Linking...

Build was successful

-----> Discovering process types

Procfile declares types -> web

-----> Compiled slug size: 3.5MB

-----> Launching... done, v4

https:*//github.com/Ebanisteria-Leon/Ebanisteria\_Django.git* To git*@heroku*.com:rocky-hamlet-67506.git

\* [new branch] master -> master

Con el siguiente comando se abre la aplicación en el navegador:

$ heroku open

Archivo Procfile

Después del despliegue, se crea un archivo Procfile en el cual se pone un comando de ejecución automática, asi como se ejecuta el Python manage runserver

Ver los *logs*

Heroku trata los *logs* como conjuntos ordenadores por tiempo de eventos agregados de todas las posibles salidas de *log* que existan en la aplicación y en todos los componentes de Heroku, con lo que se proporciona un único canal para ver todos estos eventos.

$ heroku logs --tail

Despliegue Front-End

### [Create and Deploy a React App in Two Minutes](https://blog.heroku.com/deploying-react-with-zero-configuration" \l "create-and-deploy-a-react-app-in-two-minutes)

You can get started building React apps for free on Heroku.

npm install -g create-react-app

create-react-app my-app

cd my-app

git init

heroku create -b https://github.com/Ebanisteria-Leon/Ebanisteria\_react.git

git add .

git commit -m "Despliegue Front-End"

git push heroku master

heroku open

Bibliografía

* <https://tour.dlang.org/tour/es/vibed/deploy-on-heroku>, junio 14, 2022.
* <https://blog.heroku.com/deploying-react-with-zero-configuration>, junio 14, 2022.