



Despliegue en Heroku de un Backend

Creado con Java y Maven

Creado por: Ing. Danie Duque

IMPORTANTE

Para hacer el despliegue de un proyecto con las características indicadas, es necesario garantizar que el proyecto se encuentre funcionando correctamente y sin errores en el localhost, los errores que allí se presenten van a replicarse en el servidor web y el aplicativo terminará por caer en una pantalla de error.

Primer Paso - Hosting db

Se debe tener en cuenta también que, si el proyecto cuenta con su base de datos, la conexión no puede ser establecida de forma local, es necesario, contar con un servicio de base de datos accesible desde internet (en la nube).

Algunos servicios gratuitos temporalmente para ubicar nuestra base de datos del proyecto en la nube son los siguientes. Considere uno de ellos para este ejercicio:

- <https://db4free.net/>
- [Remote MySQL](#)
- [Cloud SQL de Google](#)
- [Free MySQL Hosting](#)
- [Render Databases](#)

En nuestro caso, utilizaremos el servidor gratuito:

<https://db4free.net/>

A continuación, un vídeo inicial para configurar nuestro hosting MySQL, de la forma:

<https://www.youtube.com/watch?v=Cw4TgJIHaXI>

Segundo Paso – Instalación Maven

Seguidamente, es necesario contar con Maven instalado en el equipo desde el que realizaremos el despliegue del aplicativo, a continuación, algunas guías:

- [Instalación de Maven en Windows](#)
- [Instalación de Maven en MacOS con HomeBrew](#)
- [Instalación de Maven en Ubuntu](#)

Al finalizar la instalación correspondiente a su sistema operativo, compruebe desde la consola del sistema (CMD desde Windows) la correcta ejecución de Maven, de la forma:

```
mvn -version
```

Debería obtener una salida como la siguiente:

Windows

```
Microsoft Windows [Version 10.0.19044.1526]
(c) Microsoft Corporation. All rights reserved.

C:\Users\akova>mvn -version
Apache Maven 3.8.4 (9b656c72d54e5baced989b64718c159fe39b537)
Maven home: C:\Program Files\Maven\apache-maven-3.8.4
Java version: 1.8.0_321, vendor: Oracle Corporation, runtime: C:\Program Files (x86)\Java\jre1.8.0_321
Default locale: en_US, platform encoding: Cp1252
OS name: "windows 10", version: "10.0", arch: "x86", family: "windows"
```

MacOS

```
● usuarioutp@MAC331248-9 videojuegos_vscode % mvn -version
Apache Maven 3.8.6 (84538c9988a25aec085021c365c560670ad80f63)
Maven home: /usr/local/Cellar/maven/3.8.6/libexec
Java version: 18.0.2, vendor: Homebrew, runtime: /usr/local/Cellar/openjdk/18.0.2/libexec/openjdk.jdk
Default locale: es_419, platform encoding: UTF-8
OS name: "mac os x", version: "12.5.1", arch: "x86_64", family: "mac"
```

Ubuntu

```
phoenixnap@test-system:~$ mvn -version
Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.13, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "5.8.0-63-generic", arch: "amd64", family: "unix"
phoenixnap@test-system:~$
```

Tercer Paso – Repo en GitHub

Ahora, su proyecto debe contar con un repositorio inicializado de GIT, haber agregado los archivos del proyecto (al repositorio) y contar con al menos un commit, ya que lo que se encuentre en el último commit de la rama sobre la que estemos trabajando, será lo que vamos a enviar hacia la plataforma.

Para ello te sugiero, instalar [Git Extension Pack](#) para crear y mantener tus repos.

Con estos tres pasos completados, es posible iniciar con Heroku.

Cuarto Paso - Plataforma Heroku

Heroku es una plataforma que nos permite utilizar algunos servicios en la nube como bases de datos o ecosistemas más complejos principalmente para el despliegue de aplicaciones modernas. Para ello, es necesario crear una cuenta en su plataforma, eso lo logramos, accediendo a la siguiente página:

<https://signup.heroku.com/login>

IMPORTANTE

Las cuentas gratuitas, hasta la fecha, cuentan con hasta 5 aplicaciones con más de 400 horas de uso al mes por aplicación y, algunas funcionalidades adicionales están disponibles con solo agregar una tarjeta de crédito sin necesidad de incurrir en pagos.

Seguidamente, cuando se haya confirmado el alta de nuestra cuenta de Heroku, es momento de hacer login en nuestro computador.

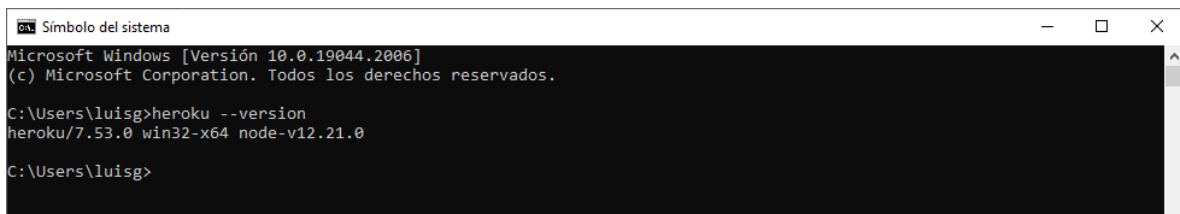
Para ello, inicialmente descargamos [La Interfaz de línea de comandos](https://devcenter.heroku.com/articles/heroku-cli) (Heroku CLI), en la siguiente dirección:

<https://devcenter.heroku.com/articles/heroku-cli>

Al finalizar la instalación correspondiente a su sistema operativo, compruebe desde la consola del sistema (CMD desde Windows) la correcta ejecución de Heroku CLI, de la forma:

```
heroku --version
```

Debería obtener una salida como la siguiente:

A screenshot of a Windows Command Prompt window titled 'Símbolo del sistema'. The window shows the following text: 'Microsoft Windows [Versión 10.0.19044.2006]', '(c) Microsoft Corporation. Todos los derechos reservados.', 'C:\Users\luisg>heroku --version', 'heroku/7.53.0 win32-x64 node-v12.21.0', and 'C:\Users\luisg>'. The text is displayed on a black background with white font.

Finalmente, es necesario realizar el login en nuestro computador, para realizar el login puede ejecutar uno de los dos comandos y seguidamente, introducir sus datos de

```
heroku login
```

ó

```
heroku login -i
```

Con la autenticación exitosa, ya es posible pasar a crear el proyecto en Heroku.

Quinto Paso – Actualización de los archivos: pom.xml y application.properties

Para lograr subir con éxito el proyecto a Heroku, es necesario hacer algunas adiciones en primer lugar (configuraciones del plugin de Maven). Así entonces, actualizamos:

El archivo “pom.xml”

Agreguemos el siguiente código en el Plug-in que se encuentra dentro de la etiqueta <build> </build> (ubicada al final del archivo “pom.xml”), por el siguiente:

```
<plugin>
  <groupId>com.heroku.sdk</groupId>
  <artifactId>heroku-maven-plugin</artifactId>
  <version>3.0.4</version>
</plugin>
```

Ahora, actualicemos el archivo “application.properties”

Es necesario indicar un puerto que la plataforma tenga disponible, ya que, por defecto, el 8080 no debería estar disponible. Para ello, es posible dejar que la aplicación decida o nosotros indicar uno directamente.

Entre todas las opciones posibles, para el presente proyecto se va a configurar el puerto 8090 en el archivo “application.properties” pero, siéntase libre de utilizar cualquier otra combinación disponible.

El archivo se ubica en la ruta /src/main/resources/ o puede crearlo allí.

Ahora, al inicio del archivo application.properties debe agregar la siguiente línea

```
# port
server.port=${PORT:8090}
```

Sexto Paso – Creación de un proyecto en Heroku

Una vez desarrollado efectivamente los pasos anteriores, procedemos a abrir una consola en vscode y ejecutar los siguientes comandos:

```
git add .
```

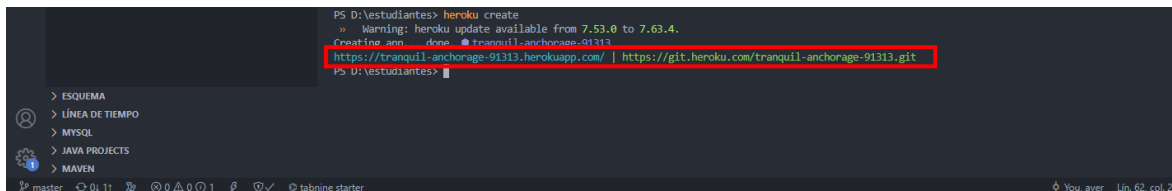
Seguidamente:

```
git commit -m "initial commit"
```

Ahora, ejecutamos el siguiente comando:

```
heroku create
```

Si todo se ejecuta correctamente, deberá ver un mensaje indicando que la aplicación se ha creado junto con un enlace, de la forma:



```
PS D:\estudiantes> heroku create
» Warning: heroku update available from 7.53.0 to 7.63.4.
Creating app... done ⬢ tranquil-anchorage-91313
https://tranquil-anchorage-91313.herokuapp.com/ | https://git.heroku.com/tranquil-anchorage-91313.git
PS D:\estudiantes>
```

Ahora, se actualiza nuevamente el Plug-in del archivo “pom.xml”, de la forma:

```
<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
    </plugin>
    <plugin>
      <groupId>com.heroku.sdk</groupId>
      <artifactId>heroku-maven-plugin</artifactId>
      <version>3.0.4</version>
      <configuration>
        <appName>tranquil-anchorage-91313</appName>
        <includeTarget>false</includeTarget>
        <includes>
          <include>${project.build.directory}/${project.build.finalName}.jar</include>
        </includes>
        <jdkVersion>${java.version}</jdkVersion>
        <processTypes>
          <web>java $JAVA_OPTS -jar target/${project.build.finalName}.jar</web>
        </processTypes>
      </configuration>
    </plugin>
  </plugins>
</build>
```

A continuación, se explica en que consiste el cambio, a saber:

- versión: la versión del plugin para realizar el despliegue

```
<version>3.0.4</version>
```

- appName: debe coincidir con el nombre de la aplicación creada en Heroku

```
<appName>tranquil-anchorage-91313</appName>
```

Séptimo Paso – Despliegue de la aplicación en Heroku

Si todo lo hemos hecho como en la guía y cruzamos los dedos para que el último paso nos funcione correctamente, estamos a dos comandos de visualizar nuestra aplicación en la web.

Antes de ejecutarlos verificamos que existen cambios no confirmados o pendientes por cargar al repositorio de GIT, la secuencia sería la siguiente:

<code>git status</code>	-> para verificar el estado
<code>git add .</code>	-> para agregar los cambios a la pila
<code>git commit -m "commit message"</code>	-> para confirmar los cambios

Y listo, ahora ejecutamos el comando que realizará el push de nuestro proyecto a la aplicación creada en heroku, el comando puede tardar unos minutos dependiendo de nuestro aplicativo:

```
mvn clean heroku:deploy
```

Una vez finalizado el proceso, ejecutamos:

```
heroku open
```

para que abra la aplicación en el navegador.

En caso de errores ejecutar:

```
heroku logs
```

ó

```
heroku logs -tail
```

que puede ayudarnos con la depuración.