

---

# HelpButtons

Comunidad de HelpButtons

26 de abril de 2021

# Índice general

<b>1</b>	<b>Inicio</b>	<b>3</b>
<b>2</b>	<b>¿Qué es HelpButtons?</b>	<b>3</b>
<b>3</b>	<b>Instalaciones de Help Buttons</b>	<b>3</b>
3.1	En GNU/Linux . . . . .	3
3.1.1	Instalación de RVM (Ruby Version Manager) . . . . .	3
3.1.2	Instalación de la versión de ruby del proyecto . . . . .	4
3.1.3	Instalación y configuración de PostgreSQL . . . . .	5
3.1.4	Instalación de Nodejs y Yarn . . . . .	6
3.2	Ejecución del proyecto . . . . .	6
3.2.1	Hb-backend . . . . .	6
3.2.2	Hb-frontend . . . . .	8
3.3	Posibles errores durante la instalación . . . . .	9
3.3.1	Command not found . . . . .	9
3.3.2	Ruby version . . . . .	9

# 1 Inicio

## 2 ¿Qué es HelpButtons?

## 3 Instalaciones de Help Buttons

### 3.1 En GNU/Linux

Referencias seguidas:

- [Documentación oficial de la instalación de Ruby on Rails en castellano](#)
- [Documentación oficial de RVM](#)

Para realizar la instalación de Ruby on Rails (RoR) en Debian o derivados (Ubuntu) existen los siguientes requisitos:

- RVM
- PostgreSQL
- Nodejs
- Yarn

#### 3.1.1 Instalación de RVM (Ruby Version Manager)

Instalamos los paquetes del sistema operativo requeritos por RVM:

```
1 sudo apt install -y git-core subversion gnupg2 curl
```

Añadimos la firma del repositorio de RVM que se puede encontrar en su [página oficial](#):

```
1 gpg --keyserver hkp://pool.sks-keyservers.net --recv-keys 409
  B6B1796C275462A1703113804BB82D39DC0E3 7
  D2BAF1CF37B13E2069D6956105BD0E739499BDB
```

Instalamos la versión estable de RVM:

```
1 curl -sSL https://get.rvm.io | bash -s stable --ruby
```

Este comando instala las últimas versiones estables de **rvm**, **ruby** y **rails**. Al ejecutarlo se devuelve muchos mensajes por pantalla.

Una vez finalizado este proceso, RVM nos dice que ejecutemos el siguiente comando o que reiniciemos las shells abiertas. También ejecutaremos este comando en caso de obtener un error diciendo que no se ha encontrado el comando `bundle`: `bash: bundle: command not found`

```
1 source /home/$USER/.rvm/scripts/rvm
```

Se comprueba la configuración de rvm mediante la siguiente orden:

```
1 type rvm | head -n 1
```

La cual nos devolverá que `rvm is a function`. En caso de no ser así, rvm no está configurado adecuadamente. En tal caso habrá que consultar la documentación en <https://rvm.io/rvm/install> y solicitar ayuda.

A continuación, comprobamos los comandos: rvm, ruby, irb, gem.

```
1 rvm list
2 ruby -v
3 irb -v
4 gem -v
```

### 3.1.2 Instalación de la versión de ruby del proyecto

Bundle es una gema de ruby que lleva la cuenta e instala las diferentes versiones de las gemas que necesitamos. Ya tenemos el entorno instalado y disponible para trabajar con él.

Al intentar instalar el bundle con el comando a continuación, seguramente nos encontremos con un error `Your Ruby version is Y.Y.Y, but your Gemfile specified X.X.X`.

```
1 bundle install
```

Para resolverlo, se necesita instalar y usar la versión X.X.X. adecuada.

```
1 rvm install X.X.X
2 rvm use X.X.X
```

Si volvemos a ejecutar el comando `bundle install` nos aparecerá otro error indicando que falta instalar la base de datos PostgreSQL.

### 3.1.3 Instalación y configuración de PostgreSQL

Se instalan los paquetes necesarios para la instalación de PostgreSQL:

```
1 sudo apt install postgresql postgresql-contrib libpq-dev -y
```

Se levantan el servicio con:

```
1 systemctl start postgresql
2 systemctl enable postgresql
```

Se abre una consola en PostgreSQL para modificar la contraseña del usuario `postgres`:

```
1 sudo -i -u postgres psql
```

Se introduce el siguiente mandato para cambiar la contraseña:

```
1 \password postgres
```

A continuación, se crea un nuevo usuario de PostgreSQL llamado `rails_dev` con privilegios para crear una base de datos `with createdb` y una contraseña `aq1234567890` (se recomienda cambiarla por otra más robusta, a poder ser generada con KeePassXC o BitWarden)

```
1 create role hb_dev with createdb login password 'aq1234567890';
```

Se muestran los usuarios de PostgreSQL con:

```
1 \du
```

Escribe `exit` para salir de la consola de PostgreSQL.

Volvemos a ejecutar `bundle install` para continuar con la instalación.

```
1 bundle install
```

Termina la ejecución sin errores. El siguiente paso será instalar Nodejs y Yarn.

### 3.1.4 Instalación de Nodejs y Yarn

Primero instalamos las herramientas de desarrollo para compilar C.

```
1 sudo apt install gcc g++ make
```

Se instalan los repositorios de Nodejs 14 a través del siguiente comando:

```
1 curl -sL https://deb.nodesource.com/setup_14.x | sudo bash -
```

Se instala nodejs mediante la siguiente orden:

```
1 sudo apt-get install -y nodejs
```

Se comprueba la instalación de nodejs y npm con:

```
1 node -v
2 npm -v
```

Se añade la firma y los repositorios de yarn mediante:

```
1 curl -sL https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add
-
2 echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee /
  etc/apt/sources.list.d/yarn.list
```

Instalamos yarn:

```
1 sudo apt update && sudo apt install yarn
```

## 3.2 Ejecución del proyecto

### 3.2.1 Hb-backend

Creamos el archivo de configuración de la base de datos `config/database.yml` con vim o nano (se recomienda nano en caso de no saber utilizar vim):

```
1 vim config/database.yml
```

Por defecto, no existirá este fichero porque se incluye en el `.gitignore`. Por lo tanto, se puede coger una plantilla y añadir las siguientes secciones o

añadir el documento entero como se expone un poco más abajo: - En la sección **development**, añade la configuración de PostgreSQL:

```
1 development:
2   <<: *default
3   database: test_project_development
4   username: hb_dev
5   password: aq1234567890
6   host: localhost
7   port: 5432
```

- En la sección **testing**, añade la configuración de PostgreSQL:

```
1 test:
2   <<: *default
3   database: test_project_test
4   host: localhost
5   port: 5432
6   username: hb_dev
7   password: aq1234567890
```

Un ejemplo del archivo final de `config/database.yml` será el siguiente:

```
1 default: &default
2   adapter: postgresql
3   encoding: unicode
4   pool: <%= ENV.fetch("RAILS_MAX_THREADS") { 5 } %>
5
6 development:
7   <<: *default
8   database: test_project_development
9   username: hb_dev
10  password: aq1234567890
11  host: localhost
12  port: 5432
13
14 test:
15   <<: *default
16   database: test_project_test
17   username: hb_dev
18   password: aq1234567890
19   host: localhost
20   port: 5432
21
22 production:
```

```
23 <<: *default
24 database: test_project_production
25 username: test_project
26 password: <%= ENV['TEST_PROJECT_DATABASE_PASSWORD'] %>
```

Guarda los cambios y sal del editor.

Lo siguiente será generar el esquema de la base de datos PostgreSQL ejecutando:

```
1 rake db:setup
```

O alternativamente:

```
1 rails db:setup
2 rails db:migrate
```

Por último, levanta el servidor en local y accede a la dirección <http://localhost:3000/>.

```
1 rails s
```

### 3.2.2 Hb-frontend

Instalamos ember glabalmente:

```
1 npm install -g ember-cli
```

En caso de obtener un error relaciondo con permisos de acceso `npm ERR! Error: EACCES,,` cambiamos el propietario de la carpeta `node_modules` de root al usuario actual para poder acceder al directorio:

```
1 sudo chown -R $USER /usr/lib/node_modules
```

Instalamos las dependencias del front:

```
1 npm install
```

Se sirve el frontend y se accede desde el navegador a la url <http://localhost:4200/>:

```
1 ember s
```



## 3.3 Posibles errores durante la instalación

### 3.3.1 Command not found

- **Bundle:** Si al ejecutar `bundle install` nos devuelve el error `bash: bundle: command not found` será necesario volver a ejecutar:

```
1 source /home/$USER/.rvm/scripts/rvm
```

- **Rails:** Si al ejecutar `rails db:setup` nos devuelve el error `bash: rails: command not found` será necesario volver a ejecutar:

```
1 source /home/$USER/.rvm/scripts/rvm
```

- **ember:** Si al ejecutar `ember -v` nos devuelve el error `bash: ember: command not found` será necesario volver a ejecutar:

```
1 export PATH=node_modules/.bin:$PATH
```

Añadirlo al PATH para tener una solución permanente. Por ejemplo, si se está utilizando bash como consola, se podrá incluir los comandos como últimas líneas en el archivo `.bashrc` y posteriormente aplicar los cambios con `source .bashrc`.

### 3.3.2 Ruby version

Al intentar instalar el bundle con el comando a continuación, seguramente nos encontremos con un error `Your Ruby version is Y.Y.Y, but your Gemfile specified X.X.X`.

```
1 bundle install
```

Para resolverlo, se necesita instalar y usar la versión X.X.X. adecuada.

```
1 rvm install X.X.X
2 rvm use X.X.X
```