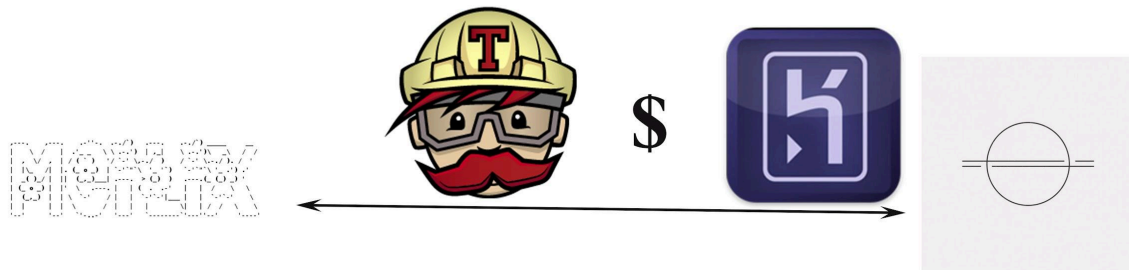


Setting Environment Variables up in Travis and Heroku for Meilix and Meilix-Generator



[Meilix Generator](#) is a webapp whose task is to take input as a configuration and start the [Meilix](#) build. But an anonymous person cannot start the Meilix build of any user and deploy the release in the repository. There are ways which are used as authentication passes through environment variable to start the build. In this article, I show the way I used to trigger Meilix by setting up environment variables in Meilix Generator.

Environment variables are of great use when one has to supply personal token in an open-source project for accessing the repository. So down there, we will have ways to configure the variables in Heroku and Travis. There are so many wikis out there but this one is the **blend of both Heroku and Travis**.

Heroku

There are several ways to setup variables in Heroku. The way I'm going to describe below is used to access travis build using Heroku.

Using the Heroku variable generated using Travis will help to trigger the build on Travis.

How:

Idea:

We will use Travis CLI to generate token (unique and keep it secret). Then provide the token as a variable name to the Heroku.

Backdoor:

This Travis token will give access to the Heroku to trigger the build on that particular Travis account. We use variable to provide the token, since in the script we will use this variable as environment variable to fetch the token in the place of token like as \$token.

Implementation:

Open your terminal and type the following:

```
sudo apt install ruby ruby-dev
sudo gem install travis                # install Travis CLI
travis login --org                      # login into Travis
travis token --org                      # generate your secret personal access token
```








You will get a token, copy and paste it into the your Heroku app's settings config vars token. You have to use the `KEY` as the variable which is used in the script for triggering the build. Save it and you are done setting of the token in the Heroku.

Config Variables

Config vars change the way your app behaves. In addition to creating your own, some add-ons come with their own.

Config Vars

Hide Config Vars

email		 
KEY	hidden	 
TRAVIS_TAG		 
KEY	VALUE	

Travis

Now it's time for Travis token.

It is used to deploy the build in that repository only.

We can use the token in two ways either paste it in the setting of that repository on Travis or pasting the encrypted form of that in the .travis.yml file in that repository. Both will work. But one thing to remember that you must have the write access of that repository.

How

Idea:

It is used in .travis.yml file as a environment variable to successfully build and deploy the application as a Github release.

Backdoor:

The token gives the permission to Travis to deploy the build application in the github release of the repo(if one using to deploy it there only).

Implementation:

Head up to Github and generate a personal access token with scope repo. Copy the generated token in a safe place.

Way 1:

Paste the token in the setting of the repo in Travis in Environment Variable option. Now it will access the Github repository since it has got the permission from the personal token generated from Github.

Environment Variables

Notice that the values are not escaped when your builds are executed. Special characters (for bash) should be escaped accordingly.

KEY	Value	Display value in build log
<input type="text" value="Name"/>	<input type="text" value="Value"/>	<input type="button" value="OFF"/> <small>Display value in build log</small>

Way 2:

Open terminal:

```
cd repo_name # cd into the cloned repo
travis encrypt secret_token #replace secret_token with the token
generated
```

or

```
travis encrypt secret_token -r user/repo #if you are not in the repo
```

Copy that encrypted token and paste it in proper format in the .travis.yml file. Now you enabled Travis giving permission to release the build.

How can this idea be helpful to a developer

A developer can use this to build the Github Release in its repository. One can secure its token using this technique. One can use it to trigger its personal project in Travis using Heroku.

Useful repositories and link which uses this:

I have used the same idea in my project. Do have a look

<https://github.com/fossasia/meilix-generator>

[about environment variable](#)

[encryption keys](#)

[triggering build](#)

Tags: Heroku, Travis, Environment Variable, meilix, meilix-generator, GSoC, FOSSASIA