repostats

Command line Tool

Table of Contents

1. Introduction

repostats is a custom-made command which fetches data related with VCS repositories, parses them and performs statistical calculations. It also displays these aggregated results grouped by the option given as argument

2. Installation Guide

- Run gem install repostats-0.0.1.gem
- repostats command has been created and can be used

3. Command Syntax

Below is a detailed description of how the *repostat* command is used.

```
repostats [global options] command [command options]
```

where the implemented *global options* are:

--vcs=vcs_name	VCS host name	Optional (default value: 'github')
-f, --filename=file_name	Output file name	Optional (default: none)
--help	Display help message	-
-o, --org=org_name	Organization name	Required (default: none)
-t, --token=vcs_token	VCS user token	Optional (default: none)
--version	Display version	-
-c, --cons	Display result in console	Optional (default: false)

and the command options are:

allStats	Calculate all the statistics
avgPerLang	Calculate the average amount of bytes divided by #lang
avgPerRepo	Calculate the average amount of bytes divided by #repos
help	Display help regarding commands
langBytesPerc	Calculate bytes per language divided by the total repos bytes
langPerc	Calculate the percentage of each language frequency
mainLangPerc	Calculate the percentage of each main language frequency

Note that

- VCS user token is the token that vcs generates for an authorized user. It is optional since in case of GitHub a non-authorized is allowed to make 60 requests per hour while an authorized one has 5000 requests as a limit.

- The results of all kinds of calculation are either being displayed on the console, if user sets the appropriate switch (-c) or written in a file in order to save them for later. Although file name is optional, in case that user does not set one, the tool automatically generates a file name of the following format:

  <org>_<command option>_<timestamp>.txt

- When *allStats* command is executed and file option is active, only one single file is created and contains the results about all methods that have been calculated.

Execution Example

repostats -o skroutz langBytesPerc

Output:

```
|langBytesPercent|
{"CoffeeScript":16.11,"JavaScript":16.29,"CSS":4.73,"Java":15.27,"Ruby":21.04,"HTML":15.99
,"PHP":4.6,"Makefile":0.1,"Shell":0.01,"C":0.44,"Clojure":1.06,"Elixir":2.1,"Go":2.26}
```

4. Calculation Methods

This command line tool has been created to process data related with vcs repositories (in our case GitHub) and provide results about the volume of relevance between the programming languages that have been used in those repositories.

As a volume of languages relevance, we have assumed the bellow (which have been mentioned above as command options)

a) avgPerLang

The number of bytes per language divided by the number of distinct languages found in repositories

$$\frac{total\_lang\_bytes}{\#distinct\_langs}$$

b) avgPerRepo

The number of bytes per language divided by the number of public and non-forked repositories

$$\frac{total\_lang\_bytes}{\#repos}$$

c) langBytesPerc

Total percentage of bytes per programming language divided by the total bytes of all public and non-forked repositories

$$\frac{total\_lang\_bytes}{total\_repos\_bytes}\%$$

d) langPerc

The percentage of each language frequency compared with the total number of non-distinct languages

$$\frac{lang\_freq}{\#non\_distinct\_langs}\%$$

e) mainLangPerc

This calculation focuses on the main language of each repository and is computed by the division of the frequency of a language (as main) with the number of repositories

$$\frac{main\_lang\_freq}{\#repos}\%$$

5. Implementation Points

This section describes the main points of the implementation of repostat command.

- The command line tool has been implemented with Ruby

- ‘**Gli’** interface has been used to build the command line application (https://rubygems.org/gems/gli/versions/2.14.0)

- ‘**Faraday’** has been used as an HTTP/REST API client (https://rubygems.org/gems/faraday/versions/0.9.2)

- **‘Faraday Http Cache’** has been used for caching purposes. We want to cache the API responses due to the request limitation mentioned above. (https://rubygems.org/gems/faraday-http-cache/versions/1.2.2)

- ‘**ActiveSupport’** for JSON formatting (https://rubygems.org/gems/activesupport/versions/5.0.0)

- ‘**RSpec’** for Unit Testing (https://rubygems.org/gems/rspec/versions/3.5.0)

- Data Collection and Calculation Process:
  - Search organization: https://developer.github.com/v3/search/#search-users to confirm that the given argument is an organization in GitHub
  - Get repositories: https://developer.github.com/v3/repos/#list-organization-repositories using type parameter to keep the non-forked ones and then filter out the private ones.
  - Request for languages: https://developer.github.com/v3/repos/#list-languages to get the number of code bytes written in each language
  - Parse the repose and calculate statistics
  - Display the results in console / file

- This version of *repostat* command tool supports only GitHub as a VCS. However, it can be easily extended for other VCS only by implementing the methods in VCSApiCalls Class. For GitHub the implementations are included in GithubApiCalls Class which extends the VCSApiCalls.

- This tool can be used for different organizations as the organization name is set as an option of the repostat command.
- Logs have been added and are written in ‘logs_repostat.txt’