OSEMN PROJECT

Shweta Kumari

05 December 16

Are there more number of JavaScript or Java Repositories?

Email:skuma10@ilstu.edu

Illinois State University, Normal, Illinois

GitHub Introduction

GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere. Git, is strictly a command-line tool, GitHub provides a web-based graphical interface and desktop as well as mobile integration. It also provides access control and several collaboration features such as wikis, task management, and bug tracking and feature requests for every project.

GitHub offers both paid plans for private repositories and free accounts, which are usually used to host open-source software projects. As of April 2016, GitHub reports having more than 14 million users and more than 35 million repositories, making it the largest host of source code in the world. A user should create an account in order to contribute content to the site, but public repositories can be browsed and downloaded by anyone at any time. With a registered user account, users are able to discuss, manage, create repositories, submit contributions to others' repositories, and review changes to code.

Projects on GitHub can be accessed and manipulated using the standard git command-line interface and all of the standard git commands work with it. GitHub also allows registered and non-registered users to browse public repositories on the site. Multiple desktop clients and git plugins have also been created by GitHub and other third parties which integrate with the platform.

The site provides us social networking-like functions such as feeds, followers, wikis (using wiki software called Gollum) and a social network graph to display how developers work on their versions of a repository and which version is latest.

The software that runs GitHub was written using Ruby on Rails and Erlang by GitHub, Inc. developers Chris Wanstrath PJ Hyett, and Tom Preston-Werner.

GitHub's Terms of Service do not require public software projects hosted on GitHub to meet the Open Source Definition. Therefore it is advisable for users and developers intending to use a piece of software found on GitHub to read the software license in the repository (usually found in a top-level file called "LICENSE", "LICENSE.txt", or similar) to determine if it meets their needs.

[Source: http://en.wikipedia.org/wiki/GitHub]

Java Language

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture.

[https://en.wikipedia.org/wiki/Java]

Javascript

Javascript is a high-level, dynamic, untyped, and interpreted programming language. It has been standardized in the ECMAScript language specification. Alongside HTML and CSS, JavaScript is one of the three core technologies of World Wide Web content production; the majority of websites employ it, and all modern Web browsers support it without the need for plug-ins. JavaScript is prototype-based with first-class functions, making it a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles. It has an API for working with text, arrays, dates and regular expressions, but does not include any I/O, such as networking, storage, or graphics facilities, relying for these upon the host environment in which it is embedded.

[https://en.wikipedia.org/wiki/JavaScript]

Data Scrubbing

To gather data from github repositories, I used the Github API 'https://api.github.com'. Parameters have been passed to the queries. The following code describes how the data was derived from the api:

A little decription of the packages used is as follows:

jsonlite package

A fast JSON parser and generator optimized for statistical data and the web. Started out as a fork of 'RJSONIO', but has been completely rewritten in recent versions. The package offers flexible, robust, high performance tools for working with JSON in R and is particularly powerful for building pipelines and interacting with a web API.

plyr package

A set of tools that solves a common set of problems: you need to break a big problem down into manageable pieces, operate on each piece and then put all the pieces back together. For example, you might want to fit a model to each spatial location or time point in your study, summarise data by panels or collapse high-dimensional arrays to simpler summary statistics. The development of 'plyr' has been supported by 'Becton Dick inson'.

ggplot2

A system for declaratively creating graphics, based on "The Grammar of Graphics". You provide the data, and ggplot2 guides you how to map variables to aesthetics, what graphical primitives to use, and it takes care of the details as well.

```
install.packages("jsonlite")  ## Installing jsonlite package
```

```
## Installing package into '/home/ad.ilstu.edu/skuma10/R/x86_64-pc-linux-gnu-library/3.3'
## (as 'lib' is unspecified)
```

```
install.packages("plyr")  ## Installing Plyr package
```

```
## Installing package into '/home/ad.ilstu.edu/skuma10/R/x86_64-pc-linux-gnu-library/3.3'
## (as 'lib' is unspecified)
```

```
install.packages("ggplot2")  ## Installing ggplot2 package
```

```
## Installing package into '/home/ad.ilstu.edu/skuma10/R/x86_64-pc-linux-gnu-library/3.3'
## (as 'lib' is unspecified)
```

```
library(jsonlite)  ## Loading the jsonlite package
library(plyr)  ## Loading the plyr package
library(ggplot2)  ## Loading the ggplot2 package

## Creating the url to extract the data from the Git Hub api.

url <- "https://api.github.com/"  ## the GitHub API
path <- "search/repositories"  ## Repositories path
searchJava <- "?q=language:Java"  ## And language = Java
searchJavaScript <- "?q=language:JavaScript"  ## Language = JavaScript

URLjava <- paste0(url, path, searchJava)  ## Forming the URL for Java
URLjavascript <- paste0(url, path, searchJavaScript)  ## Forming the URL for JavaScript

data_json_Java <- jsonlite::fromJSON(URLjava)
data_json_Javascript <- jsonlite::fromJSON(URLjavascript)

## json data being converted as dataframe

data_dataframe_Java <- as.data.frame(data_json_Java)
data_dataframe_Javascript <- as.data.frame(data_json_Javascript)

njava <- data_dataframe_Java[1, 1]
njavascript <- data_dataframe_Javascript[1, 1]

# Creating a table with the values required for the problem:

language <- c("Java", "JavaScript")
repositories <- c(get("njava"), get("njavascript"))

graph <- cbind(language, repositories)
graph_data_frame <- as.data.frame(graph)
```

Data summarized below:

So, we have a table for repositories created in Java and in JavaScript. The total count of repositories for Java and JavaScript can be calculated as follows:

```
class(graph_data_frame)
```

```
## [1] "data.frame"
```

```
str(graph_data_frame)
```

```
## 'data.frame':    2 obs. of  2 variables:
##  $ language    : Factor w/ 2 levels "Java","JavaScript": 1 2
##  $ repositories: Factor w/ 2 levels "2584388","2905058": 1 2
```

```
summary(graph_data_frame)
```

```
##        language  repositories
##  Java      :1   2584388:1
##  JavaScript:1   2905058:1
```

```
graph_data_frame
```

```
##     language repositories
## 1       Java      2584388
## 2 JavaScript      2905058
```

```
write.csv(graph_data_frame,file="OSEMNassignment.csv")
```

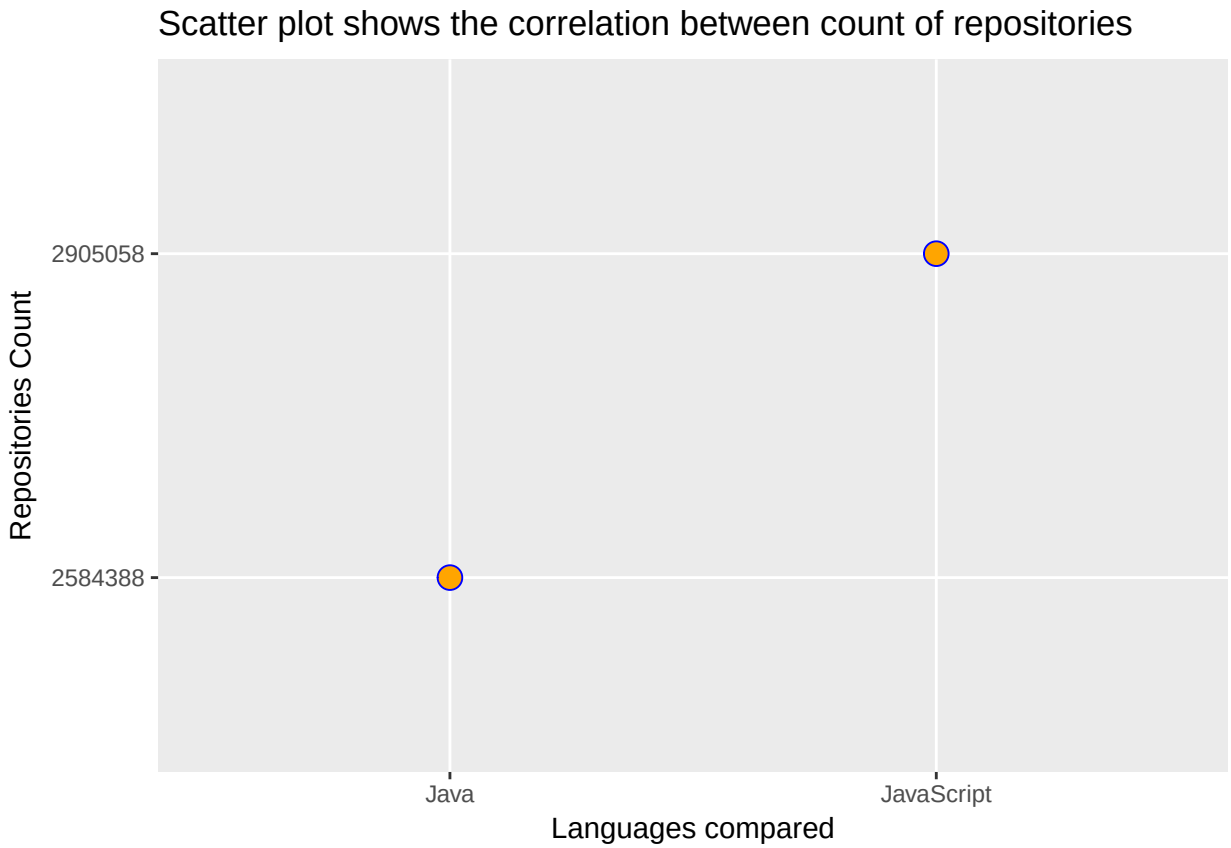
Data Analysis

Lets analyse the data via plotting them on the various graphs. I have plotted the language on x-axis and count of repositories in y-axis. I have demostrated three kinds of plots, first one being Scatter plot, second one bar graph and last one is line graph.

```
library(ggplot2)
library(scales)

## Scatter plot displaying the laaguage data along with repositories.

Scatter_plot<-ggplot2::ggplot(graph_data_frame, aes(x=language,y=repositories)) +
    geom_point(fill="orange", color="blue", shape=21, size=4)+
  xlab("Languages compared")+
  ylab("Repositories Count")+
  labs(title="Scatter plot shows the correlation between count of repositories")
Scatter_plot
```

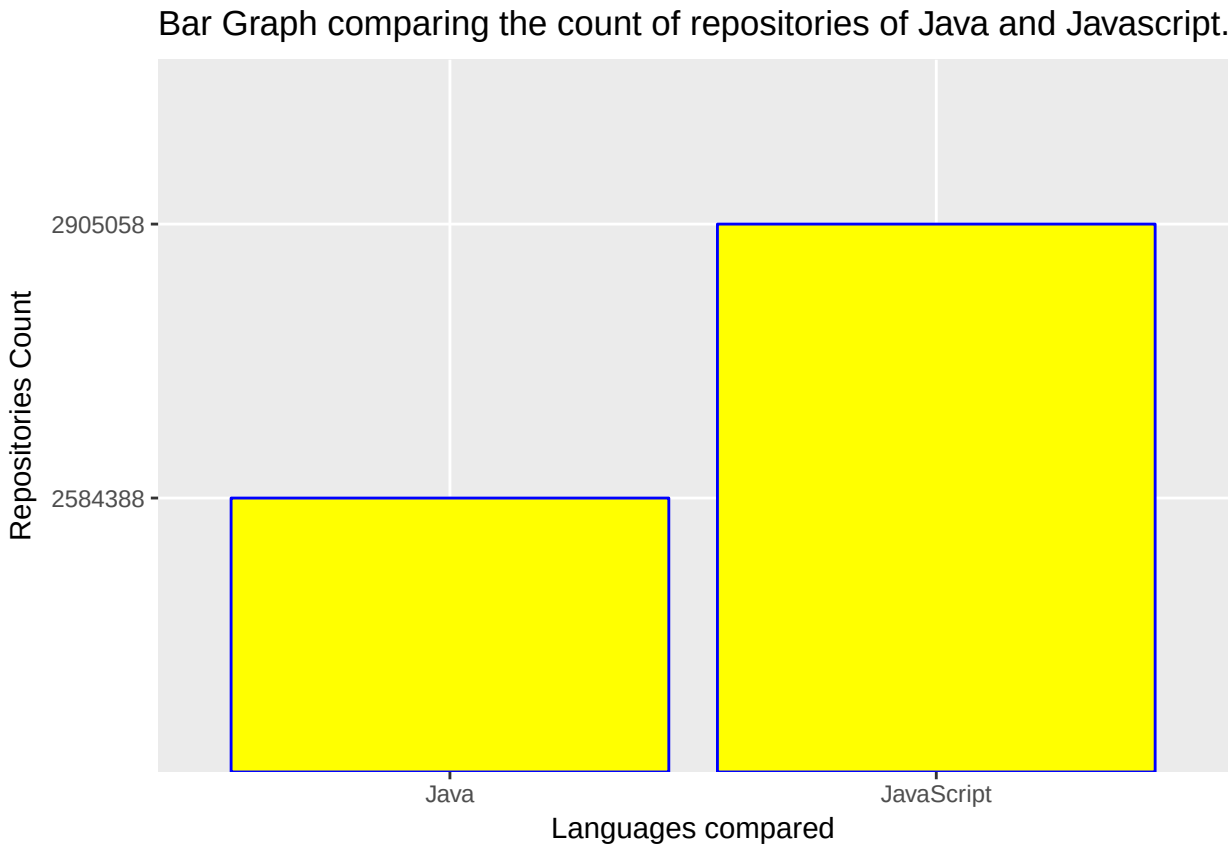


Scatter Plot

Scatter plot is a set of data plotted on horizontal and vertical axes. It is an important statistical tool for measuring correlation between sets of data.If there is a correlation the point are concentrated over a line otherwise in case of no correlation between the variables, the points appear randomly scattered on the coordinate plane. I have plotted the count of repositories on the y-axis and languages on the x-axis. As we can see the graph above we find no correlation, and we conclude that there are more javascript repositories in github.

```
library(ggplot2)
library(scales)

## Bar Graph demonstrating the data.
Bar_Graph<-ggplot2::ggplot(graph_data_frame,aes(x=language,y=repositories,fill=repositories))+
        geom_bar(stat="identity",fill="yellow",color="blue")+
  xlab("Languages compared")+
  ylab("Repositories Count")+
  labs(title="Bar Graph comparing the count of repositories of Java and Javascript.")
Bar_Graph
```

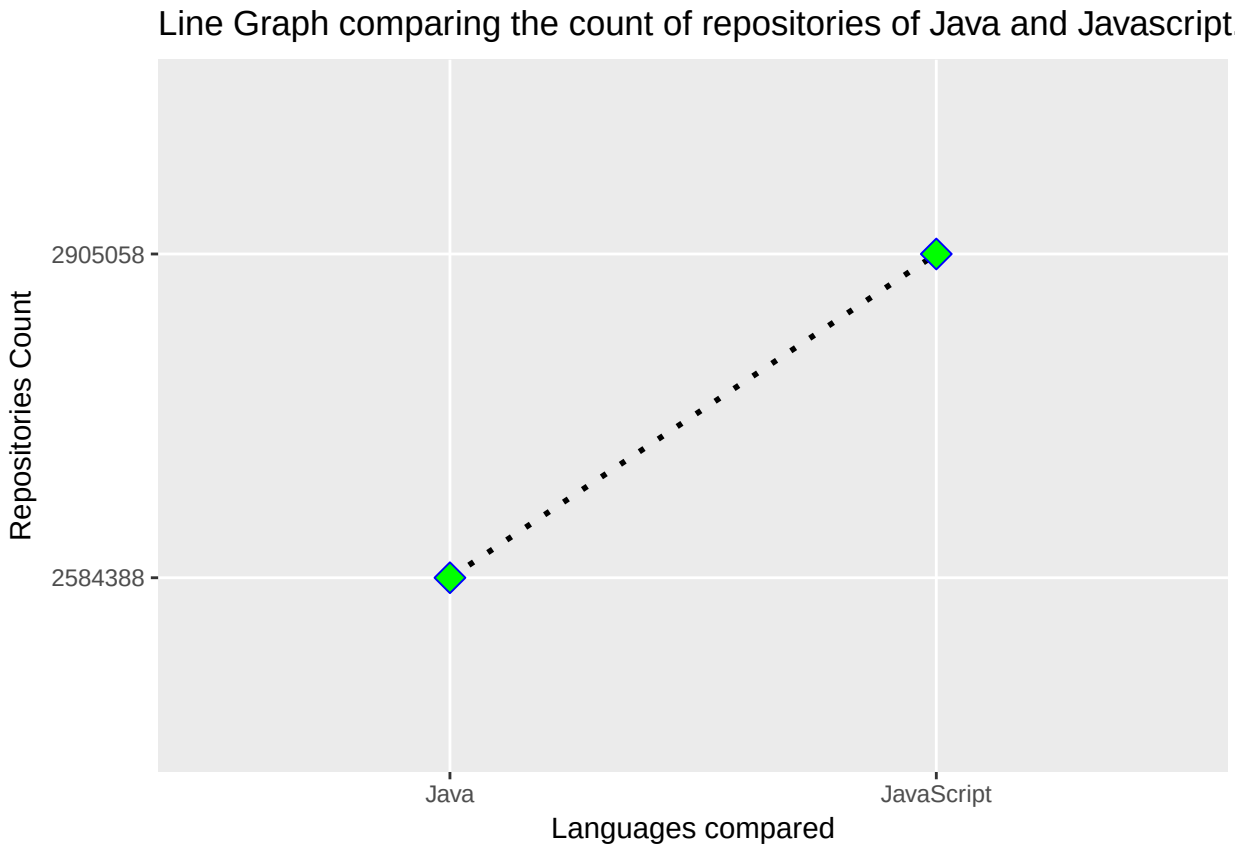


Bar Graph

Bar graphs are good when you're plotting data that spans many years or days, weeks, has really big changes from year to year or day to day, or when you are comparing things. Like in our case we are comparing whether there are more Java repositories or Javascript repositories. I have plotted the count of repositories on the y-axis and languages on the x-axis. It is so clearly evident which language has more repositories in github. Javascript repositories are more in number when compared to java repositories.

```
library(ggplot2)
library(scales)

# Line curve demonstrating the data.
line<-ggplot2::ggplot(graph_data_frame,aes(x=language,y=repositories,group=1))+
       geom_line(color="black", linetype="dotted", size=1)+
        geom_point(fill="green", color="blue", shape=23, size=4)+
  xlab("Languages compared")+
  ylab("Repositories Count")+
  labs(title="Line Graph comparing the count of repositories of Java and Javascript.")
line
```



Line Graph

A line graph, also known as a line chart, is a type of chart used to visualize the value of something over time. The line graph consists of a horizontal x-axis and a vertical y-axis. Most line graphs only deal with positive number values, so these axes typically intersect near the bottom of the y-axis and the left end of the x-axis. The point at which the axes intersect is always (0, 0). Each axis is labeled with a data type. I have plotted the count of repositories on the y-axis and languages on the x-axis. It is so clearly evident which language has more repositories in github. Javascript repositories are more in number when compared to java repositories.

```
# 3D Exploded Pie Chart
install.packages("plotrix")
```

```
## Installing package into '/home/ad.ilstu.edu/skuma10/R/x86_64-pc-linux-gnu-library/3.3'
## (as 'lib' is unspecified)
```

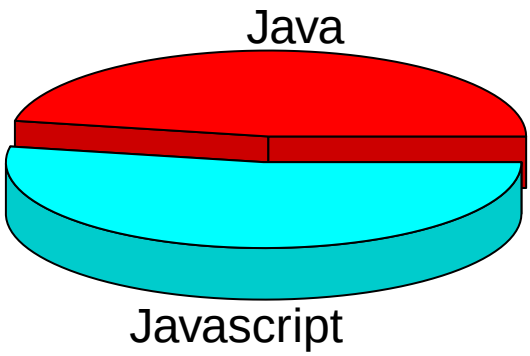
```
library(plotrix)
```

```
##
## Attaching package: 'plotrix'
```

```
## The following object is masked from 'package:scales':
##
##     rescale
```

```
slices <- c(repositories)
count <- c("Java", "Javascript")
pie3D(slices,labels=count,explode=0.1,
    main="Pie Chart of total number of Java and Javascript repositories ")
```

Pie Chart of total number of Java and Javascript repositories



3D Pie Chart

A pie chart (or a circle chart) is a circular statistical graphic, which is divided into slices to illustrate numerical proportion. In a pie chart, the arc length of each slice (and consequently its central angle and area), is proportional to the quantity it represents. While it is named for its resemblance to a pie which has been sliced, there are variations on the way it can be presented.

Result: The answer to the problem was accomplished using Github API. I wrote the queries available on the api and passed language parameter as 'java'and 'javascript' respectively. The data had to be cleaned up and converted to dataframe for further analysis.The various statistical tools were used to get a graphical representation of data and I conclude that there are more no of Javascript repositories in github.

Getting the count of Java and Javascript repositories gives the solution of the problem and it is evident that we have more number of javascript repositories in the Github.