SE 3XA3: Module Interface Specification
Google Images Downloader

Team #201, CAS Dream Team
Sam Crawford, crawfs1, 400129435
Joshua Guinness, guinnesj, 400134735
Nicholas Mari, marin, 400132494

April 6, 2020

Introduction

This document shows the complete specification for implementing all modules of the Google Images Downloader program.

Date	Name	Version	Notes
1/21/2020	Joshua	1.0	Created document
3/11/2020	Sam	1.1	Set up format of document
3/11/2020	Sam	1.1.1	Set up skeleton of each module
3/11/2020	Joshua	1.1.2	Input and Navigate Page Module
3/12/2020	Nick	1.1.3	Search Query and Main Module
3/13/2020	Sam	1.1.4	Output Module
3/13/2020	Joshua, Sam, Nick	1.2	General Edits
4/6/2020	Sam	2.0	Updated for Rev 1

Table 1: **Revision History**

Input Format Module

Module

Input

Uses

None

Syntax

Exported Types

None

Exported Constants

None

Exported Access Programs

Routine name	In	Out	Exceptions
userInput	~~N/A~~user input from keyboard	dict	InvalidParam, MissingParam

Semantics

Environmental Variables

- Keyboard - the user will enter flags, usually followed by values, that will be parsed to provide the input arguments to the program. Valid parameters are given in Table 2.

State Variables

None

State Invariant

None

Assumptions & Design Decisions

- The `userInput` method is called by Main before any other method.

Access Routine Semantics

userInput():

- output: *out* := A dictionary (key:value pair) of parameters and their values. The keys are the parameters and the values are the user input. The following parameters are valid:

Parameter	Abbr.	Description	Example
`keyword`	`k`	The keyword to search Google Images with	`"software"`
`limit`	`l`	The maximum number of images to download	`10`
`safesearch`	`ss`	Whether or not to enable SafeSearch	N/A
`filetype`	`ft`	The file type of images to download	`"gif"`
`directory`	`d`	The directory to save images to	a path to a directory
`colour`	`c`	The colour of images to download	`"teal"`
`license`	`li`	The license of images to download	`"labeled-for-reuse"`
`imagetype`	`t`	The type of images to download	`"clipart"`
`imageage`	`a`	The age of images to download	`"past-year"`
`aspectratio`	`ar`	The aspect ratio of images to download	`"square"`
`imagesize`	`s`	The size of images to download	`">2MP"`
`serverhost`	`s`	The host of the server to download images to	`"moore.mcmaster.ca"`
`serverusername`	`u`	The username to log in to the given server	the user's MacID
`serverpassword`	`p`	The password to log in to the given server	the user's Moore password
`region`	`rg`	The geographical location of images to download	`"Canada"`
`whitelist`	`wl`	The website to download images from	`"mcmaster.ca"`
`blacklist`	`bl`	The website to exclude image download from	`"mcmaster.ca"`
`fromfile`	`ff`	The file to parse input arguments from	a path to a file

Table 2: **Input Parameters**

- exception: *exc* := (required parameter missing ⇒ *MissingParam* | improper format for a parameter ⇒ *InvalidParam*)

Local Functions

keywordFromFile: string → dict
keywordFromFile(s) ≡ Reads in file s of parameters and their associated values, populating fields with their default values if not found in the file.
exception: *exc* := ~~(can't find file ⇒ *InvalidFileName* | can't read file ⇒ *UnableToParseFile* | required parameter missing ⇒ *MissingParam* | improper format for a parameter ⇒ *InvalidParam*)~~ (error processing file ⇒ *Exception* | `keyword` argument not found ⇒ *ValueError*)

Search Query Module

Module

SearchQuery

Uses

None

Syntax

Exported Types

None

Exported Constants

None

Exported Access Programs

Routine name	In	Out	Exceptions
buildURL	dict of string	string	None

Semantics

Environmental Variables

None

State Variables

None

State Invariant

None

Assumptions & Design Decisions

- All input arguments stored in the dictionary of strings are correct and in the proper format

Access Routine Semantics

buildURL(dS):

- output: *out* := ~~"https://www.google.com/search?q=&espv=2&sxsrf=ACYBGNSwqBUElVjmEWOTu3-mXPnReqFoLw:1581376760401&source=lnms" + buildURLParam(dS) + "isch&sa=X&ved=2ahUKEwiY7bzAj8jnAhUQjq0KHbXwBEYQ_AUoAXoECBMQAw&biw=838&bih=880"~~ A valid Google Images search URL with the `keyword`, `region`, `whitelist`, and the rest of the parameters formatted and encoded as necessary.
- exception: none

Local Functions

buildURLParam: dict of string → string
buildURLParam(dS) ≡ ~~s such that:~~
    ~~s = “”~~
    ~~for all i in dS:~~
        ~~$s = s + i$~~
    A string consisting of the correct Google parameters for `aspectratio`, `colour`, `filetype`, `imageage`, `imagesize`, `imagetype`, and `license` as a comma-separated string.

Navigate Page Module

Module

NavigatePage

Uses

None

Syntax

Exported Types

None

Exported Constants

None

Exported Access Programs

Routine name	In	Out	Exceptions
getImageURL	string, $\mathbb{Z}$, string	list of string	None

Semantics

Environmental Variables

State Variables

None

State Invariant

None

Assumptions & Design Decisions

- The link passed into the `getImageURL` method is correct and corresponds to a real link.
- The user has setup the program correctly (correct chromedriver in right location, installed selenium, put correct location of Google Chrome binary).

Access Routine Semantics

getImageURL(s, n, b):

- output: out := a list ($urls$) of image URLs from s where $|urls| = n$ and none of the URLs are from the blacklisted site b.
- exception: ~~None~~ ($n \leq 0 \Rightarrow$ *ValueError* | OS not supported $\Rightarrow$ *Exception*)

Output Format Module

Module

Output

Uses

None

Syntax

Exported Types

None

Exported Constants

None

Exported Access Programs

Routine name	In	Out	Exceptions
downloadImages	list of string, string, string	None	None
moveToServer	string, string, string, string, string	None	Exception

Semantics

Environmental Variables

- Screen

State Variables

None

State Invariant

None

Assumptions & Design Decisions

- ~~The first string parameter (the download location) in the `downloadImages` method is optional and has a default value ("./Images").~~
- The `createDirectory` method creates a directory called *key* in the directory *loc* and raises an *OSError* if the directory can't be created. ~~with the file path *loc* + "/" + *key*.~~

Access Routine Semantics

downloadImages(*lst*, *key*, *loc*):

- output: save each image represented by a string in *lst* and save it with the name *key* followed by an appropriate number to the folder created by calling the `createDirectory` method.
- exception: none

moveToServer(*key*, *direc*, *shost*, *suser*, *spass*):

- output: Puts the directory at src/Images/*key* on *suser*@*shost*:*key*/
- exception: If problem with putting directory on server → Exception

Local Functions

createSSH(*key*, *direc*, *shost*, *suser*, *spass*): string x string x string x string x string → SSHClient
createSSH(*key*, *direc*, *shost*, *suser*, *spass*): An SSHClient object that can be used to transport files to and from a server

deleteLocalImageFolder(*direc*, *key*): string x string → ϵ
createSSH(*direc*, *key*): Deletes local copy of images downloaded from src/Images/*key*

createDirectory(~~*loc, key*~~ *dir*): ϵ
createDirectory((~~*loc, key*~~ *dir*) ≡ Null

getRequest(*img*): byte string
getRequest(*img*) ≡ The data of the image denoted by *img*.

Main Module

Module

Main

Uses

Input, SearchQuery, NavigatePage, Output

Syntax

Exported Types

None

Exported Constants

None

Exported Access Programs

None

Semantics

Environmental Variables

None

State Variables

None

State Invariant

None

Assumptions & Design Decisions

- The main function in the Main module is used to call all the other modules in order to run them in the right order. The main function will call the modules in the following order: Input, SearchQuery, NavigatePage, Output.

Access Routine Semantics

None

Local Functions

main: $\epsilon \rightarrow \epsilon$
main() $\equiv$ Null