Language Specifications for Bhotoshop

Angela Gui

December 2022

1 Introduction

Bhotoshop is a language that allows users to modify an image easily. It implements several basic functions of a photo editor application, such as Adobe Photoshop. The main reason why I constructed my languge is that I wanted to provide an easier and more accessible way to modify an image from the command line. As Adobe products are rather costly for the average student who probably doesn't need its entire suite of functions, Bhotoshop could be a more convenient and cost-efficient way for users to perform basic photo editing.

2 Design Principles

The main guilding principles in Bhotoshop's language design are simplicity and usability. As I want my language to be easy to use even for users who are not tech savvy, I modeled my syntax after Python, which is one of the most accessible languages for scripting.

3 Examples

Example 1: Input("Filename")

Run the example using dotnet run "example-1.lang". This example loads in an image (jpg/jpeg/png file) with its name and path specified by the command line argument of Filename. If the image loading was successful, the command line output should be "Loaded successfully'.

Example 2: Output("Filename")

Run the example using dotnet run "example-2.lang". This example creates an jpg file with its name specified by the command line argument of Filename in the same directory of the code. If the image creation was successful, the command line output should be "Saved successfully'.

Example 3: Info("Filename")

Run the example using dotnet run "example-3.lang". This example loads in an image (jpg/jpeg/png file) with its name path specified by the command line argument of Filename, stores the image metadata in a string array, and

prints out its corresponding height and width in pixels. If the image loading was successful, the command line output should be "Image height: x pixels, Image width: y pixels".

4 Language Concepts

To write a program in Bhotoshop, users need to have basic familarity with primitives and combining forms.

Language primitives are ImageName and FunctionName, which are string variables that represent the name of the image and the name of the functions applied onto the image.

The two primitives combine to form Application. They are applied right to left, meaning a function of FunctionName will be applied to a variable of ImageName if they are written in the style of FunctionName(ImageName).

5 Formal Syntax

$\langle Application \rangle ::= \langle FunctionName \rangle + "(" \langle ImageName \rangle + ")"$

$\langle ImageName \rangle ::= String$
$\langle FunctionName \rangle ::= String$

6 Semantics

Application is an function application that applies the function specified by the FunctionName to the image specified by the ImageName. ImageName is a terminal of String type. FunctionName is a terminal of String type.

7 Remaining Work

There are still significant work left to do on implementing various type of function that alters the image, such as flip or recolor. There will probably be a variables that keeps track of pixel positions as well as pixel colors/