CSCI4061 Spring 2017 - Assignment 1
UNIX shell scripting

Due

February 5, 2017 by 11:59pm (Online submissions on moodle only - no hard copies accepted)

Purpose

The purpose of this assignment is to learn shell scripting with different commands, control constructs, conditional expressions and environment variables.

Scenario

After the winter holidays, you may have taken a lot of nice pictures. And you may want to show them on your personal web page (an html file). However, these pictures are of various format and resolution (e.g., some are from your smart phone and some are from your camera). Changing the format and resolution one by one is a laborious work. Fortunately, your knowledge of shell scripts saves you from the laborious work, helping you change the resolution and format of pictures automatically and merge them into an html file in a nice way.

Assignment

Your job is to write a shell script that searches through a directory tree looking for images of a specified type (**gif**, **png**, **tiff**, etc), convert all the images into **JPEG** format in a specified output directory, and create a web page called **pic_name_xx.html** that contains links to the converted images in the form of small "thumbnail" images. In other words, the website would basically be a presentation of all the thumbnail images of the images processed. The usage message [1] for your script will be:

Usage: create_images input_dir output_dir pattern_1 [pattern_2] ... [pattern_n]

where

create_images

The name of your script.

input_dir

This is the TOP/ROOT of a directory tree in which images are to be found. Images may occur in this directory, or in any subdirectory. There may also be other, non-image files in this directory. Any directory or file that is not readable, i.e., permission not allowed, should be skipped, after putting out an error message to standard output.

Attention:

1. Input directory may or may not exist.
2. Input directory may or may not have multiple subdirectories.
3. Input directory/subdirectories may or may not be readable.
4. Input file names in the input_dir will NOT have spaces. Input file names will have only ONE period (.) in the filename, e.g., hello.png

output_dir

Each image that satisfies the pattern requirement is to be converted to JPEG format (see the ImageMagick manual page) and stored in this directory. Thumbnail images must also be

generated from each image. These must be stored in a subdirectory of output_dir called "**thumbs**". The output directory may or may not already exist, and may or may not be empty initially.
Attention:

1. Output directory may or may not exist - create it if it does not exist
2. Output directory may or may not have multiple subdirectories (including the 'thumbs' subdirectory) - create thumbs if it does not exist.
3. Do not touch any files already present in the output directory or any of the subdirectories.

pattern

You will be given at least one pattern (the "[]" around the other patterns means they are optional). You need to finish 3 tasks as follows

1. Search the directory and subdirectory, find all the images satisfying the pattern(s).
2. Convert required images to *.JPG and according thumbnail images with specified resolution.
3. Merge the converted images into an html file.

Task 1 Find all files satisfying the patterns

One or more pattern arguments may be present. Each one is a search pattern (usable with find), that will identify image files to be converted. For example, the pattern *.png would identify images in the portable network graphics format. For this assignment, you need to handle conversion of ONLY **PNG**, **GIF** and **TIFF** images. Appropriate error messages must be written to standard output in case invalid patterns are used (e.g. *.pgn, *.tif) (Hint: Use regular expressions [2] to identify valid patterns)

Attention:

1. Valid regex patterns should be usable with the 'find' or 'grep' command, such as "*.png", "he*.gif".
2. If there are multiple patterns, deal with them one by one. For example, if you see both pattern "*.png" and "he*.gif", you first convert all images satisfying pattern "*.png", then deal with pattern "he*.gif". When you are converting images, follow the instructions below.

Task 2 Convert images to *.JPG format

The dimensions of the converted images should be the same dimensions as the original (in other words, do not scale the original images), and the thumbnail images must all be 200 pixels in the largest dimension (either width or height).

Converted images must be named as **<name-sans-ext-in-uppercase>.jpg**, and thumbnail images must be named as **<name-sans-ext-in-uppercase>_thumb.jpg**, where **<name-sans-ext-in-uppercase>** is the name of the original file without its original suffix in uppercase letters. In other words, if you convert an image called "**xyzzy.tiff**", the image must be named "**XYZZY.jpg**" and the thumbnail image must be named "**XYZZY_thumb.jpg**".

Each thumbnail image **you converted** should be embedded into the web page. At the same time, each of the thumbnail image should link to the original size version of the image.

Attention:

1. You must convert ONLY PNG, GIF & TIFF files; you must not attempt to convert any

other files.

2. The converted JPG image must be placed in the output directory.
3. All thumbnail images must be placed in the thumbs subdirectory.
4. Do not scale the images while converting (Maintain the same dimensions). Thumbnails should be generated and EITHER the width OR the height needs to be 200px.
5. Convert the images if and only if the output directory contains neither the original image nor the thumbnail image: If EITHER HELLO.jpg OR HELLO_thumb.jpg already exists in the output directory, you need to SKIP conversion of HELLO.png.
6. If multiple input files have the same **basename** (i.e., name without directory path, such as hello.png, hello.gif), since both will be converted to the same filename (HELLO.jpg & HELLO_thumb.jpg), only the file which matches the first pattern passed to the script must be converted.

Task 3 Merge the converted images into an html file

The HTML file is a presentation of the images you have converted. You should include ONLY those images which YOUR SCRIPT has converted. The HTML file contains links to the converted images in the form of small "thumbnail" images.

In addition, the HTML webpage should also display 1) the resolution (e.g., 200 x 300) of THUMBNAIL image, 2) the last MODIFY date of the image (hint, parse the "stat" command), 3) a user input theme message of the webpage (i.e., the script should prompt for user input, and use the input as the THEME), and 4) today's date and day (hint, parse the "date" command).

For the format of the HTML file, see the example **"mypic.html**". Generate your **pic_name_xx.html** similar to the example.

Attention:

1. The name of the HTML page: "pic_name_xx.html". You are allowed to overwrite the HTML file if present.
2. Date & day (no time) should be displayed only ONCE below the thumbnail images.
3. If the output directory previously contained images before your script was run, do not add those images to the HTML file.
4. Your script need to ask user to input the theme of the web page.

Error Handling

While the script runs, it must write to its standard output some messages that describe what it is doing. Lines should be printed at least when the script starts, when a file is converted, when a thumbnail image is created, and when the script finishes.

Errors must be handled gracefully, with informative error messages on standard output. Refer to all the "Attention" mentioned before and the grading part.

Platform

You may work on the platform of your choice: Linux, Solaris, MacOSX or Windows (using Cygwin). However, you need to ensure that your script works correctly on one of the CSELabs UNIX machines, where we will grade your assignments. All commands required in this project have already been tested on CSELabs UNIX (e.g., KH4240)

Your CSELabs UNIX account uses bash as its default shell. However, since the objective of

this assignment is to learn shell scripting (not to build a webpage or learn python or perl), this assignment must be done with the C shell [3] meaning the first line of your script should be

```
#!/bin/csh
```

Another way to go into c shell environment is to directly type "csh" in the command line.

Test Data

The compressed tar file provided along with this assignment handout contains a directory tree containing a few images, as well as a sample web page with thumbnails and converted images so you can see what the output should look like. (Note that the sample web page does not include all the images in the sample directory.)

You can extract a compressed tar file with the command

```
tar xzf tarfile
```

Grading

The general grading criteria are given in the course syllabus. For this exercise the "style" points will be based on readability. Use meaningful names if you create variables, and use consistent indentation to display any control structure in your script. For correctness of the program, you need to handle following cases:

Error handling: 25 pts (5 pt each)

1. Usage message from wrong #args
2. Non-existent input directory
3. Unreadable input directory
4. Unreadable input files
5. Existence of output directory

Correct handling of a single pattern: 30 pts (5 pt each)

1. Create output directory and a subdirectory for thumbnail images
2. No image files match the pattern
3. A single image file matches the pattern
4. Multiple image files match the pattern
5. Script is able to convert all files within a multi-level directory tree
6. Uppercase letters for converted image filenames.

Correct handling of multiple patterns: 15 pts (5 pt each)

1. Multiple patterns given, all patterns match
2. Multiple patterns given, some match
3. Multiple files with same base name, different type (e.g., sunset.gif and sunset.png)

Correctly build the HTML page as specified: 25 pts (5 pt each)

1. Show thumbnail images (200 pixels in the largest dimension) on the web page
2. Show the full sized image whenever a thumbnail image is clicked.
3. Display the resolution and the last modified date of the images
4. Display the day and date correctly on the webpage.

5. Display user input theme message

Comments and appropriate messages on standard output: 5 pts

Teamwork & Submission

This assignment has to be done in pairs. Each pair of students would need to submit only ONE copy of the assignment, include ONLY these files in a SINGLE archive (TAR) file: **README** & **the script**.

A template of script, **create_images.sh**, is provided. FILL remaining code in the template. Your code will be tested by another program. To make sure that your information is read correctly, please change the information to your own in the script template adhering to the format that is given below:

```
#!/bin/csh
# CSci4061 Spring 2017 Assignment 1
# Name: <Full name1>, <Full name2>
# Student ID: <ID1>, <ID2>
# CSELabs machine: <machine>
# Additional comments
```

In the README, you should explain, briefly, what your script does, how it works, how you use it, and how to interpret its output.

Resources & Hints

1. The format of usage message: https://en.wikipedia.org/wiki/Usage_message
2. A tutorial of regex patterns can be found in http://ryanstutorials.net/regular-expressions-tutorial/
3. In case c shell is not installed, in Linux, you can install csh by: sudo apt-get install csh
4. The class moodle page contains links to several shell-related documents.
5. You can parse the output of the 'date' command to obtain today's date and day.
6. You can parse the 'stat' of an image to find its modified date.
7. You might want to use following commands (not limited to) in your script. Refer to the man pages of these commands for more details.

   `echo set convert find shift mkdir chmod rm basename identify stat`