CSCI4061 Spring 2015 - Assignment 1
UNIX shell scripting

Due

February 8, 2015 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.

Description

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 thumbnails of the images processed. Here's what the command line for your script will look like:

```
create_images input_dir  output_dir  'pattern_1' 'pattern_2' … 'pattern_n'
```

where
create_images is 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 should be skipped, after putting out an error message to standard output.

output_dir
Each image 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.

pattern
Multiple 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 **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 to identify valid patterns)

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

image.

The HTML webpage should also display the current day and date below the thumbnails. (Parse the output of the '**date**' command to obtain this information).

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. You cannot count on your user always giving you the right number of arguments, or of giving you files or directories that actually exist or are readable. If the wrong number of arguments are given, then your script should print a "usage" message and exit with a status of 1. It's also possible that you would find multiple images in the input directory tree that would produce output files with the same name (because they have same base name e.g., sunset.jpg and sunset.png), or that you would find pre-existing files in the output directory that would conflict with an image you intend to convert. In all these cases, the correct behavior of your script should print an error message and skip the conflicting input file.

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 (csh)**, meaning the first line of your script should be

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

Here is an example **pic_name_xx.html** page:

```
<html>
    <head>
        <title>Image 01</title>
    </head>
    <body>
        <a href="images/sunset.jpg">
        <img src="images/thumbs/sunset_thumb.jpg"/></a>
    </body>
</html>
```

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
```

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, since we will be evaluating and grading your assignments on these machines.

Teamwork

This assignment has to be done in pairs. Each pair of students would need to submit only one copy of the assignment.

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: 15 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 day and date correctly on the webpage

Comments and appropriate messages on standard output: 15 pts

Deliverables

Your script must be contained within a single file. Your code will be tested by another program. To make sure that your information is read correctly, please adhere to the format that is given below:

```
#!/bin/csh
# CSci4061 Spring 2015 Assignment 1
# Name: <Full name1>, <Full name2>
# Student ID: <ID1>, <ID2>
# CSELabs machine: <machine>
# Additional comments
```

The comments should explain, briefly, what your script does, how it works, how you use it, and how to interpret its output.

Resources

The class moodle page contains links to several shell-related documents.

Hints

You might want to use following commands (not limited to) in your script:

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

Refer to the man pages of these commands for more details.