How to Add a Copyright or Watermark to Photos Using Python

Protect your images and help spread your name

Robbert Brouwers Follow
Feb 9, 2020 · 8 min read ★

Photo by Jakob Owens on Unsplash

Your photos should have a watermark so you can protect your work, or you have to apply other people's watermarks in order to use their work. This tutorial shows you how to automate this process and be able to process any number of photos (within reason) using Python.

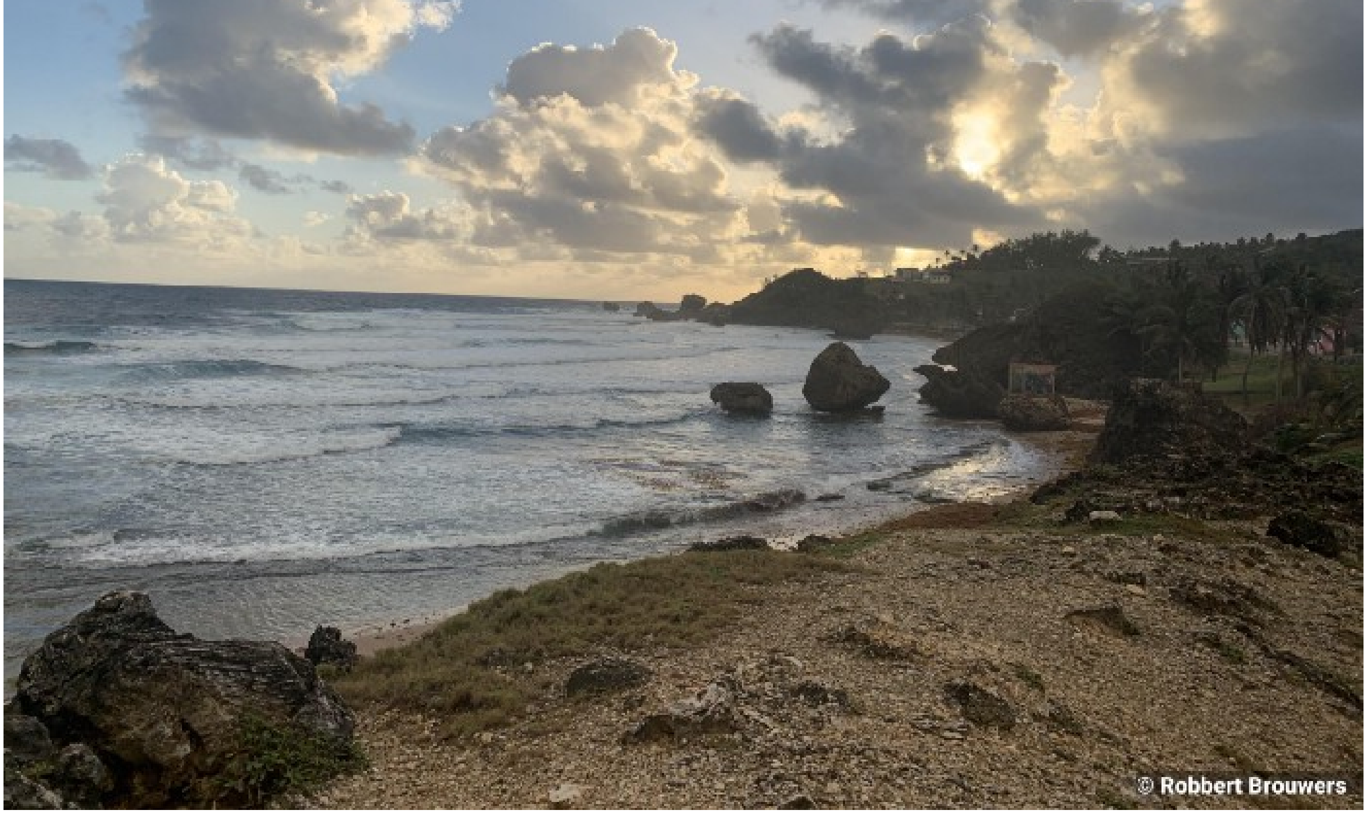
*This article assumes you have basic knowledge of Python. More specifically, you know how to install a **library** and are able to copy and paste the code examples. A complete version of the code can be found at the bottom of the article.*

Let's say you've shot some photos, like this one from a recent trip to Barbados:

Bathsheba, Barbados. © Robbert Brouwers

You want to make sure your hard work doesn't go unnoticed. Even though copying and reuse images without permission from the owner isn't allowed, there's still a risk of people doing so. This is why you can add your name, website or anything else to a photo to ensure you are being recognised as the owner.

How do we make sure our copyright claim is on the photo rather than added as a text below or above the photo?



No more manually typing the copyright claims!

Python allows us to do this in an automated manner. Meaning that whether we want to copyright one or one thousand photos, the effort is the same.

But before diving into some code, let’s have a look at the photo above again and discuss what specifically we want to achieve.

Requirements: What are we actually doing?

We have to decide a couple of things when it comes to adding watermarks or trademarks to photos:

- ***What*** should it say?
- ***Where*** do we want to place it?
- ***How*** should it look?

Where — In my example. I have decided to place the watermark in the right bottom of the photo, slightly above the bottom.

What — I’ve kept the trademark simple and decided to go for a “©” sign and my name, with a couple of empty spaces to offset it from the right edge of the photo.

How—The trademark has a white text on a slightly saturated black box. Furthermore, the box goes to the edge of the photo whereas the text doesn't. This design will ensure that the trademark is clearly visible regardless of photo colors.

Furthermore, we want to think about how we will use this Python program should we specify a single photo it needs to apply a trademark to? Should we be able to select a group of photos?

Keeping these requirements in mind will help us write the correct code.

In this tutorial, I'll show you how you can read a folder of images, but you can alter the code later to do what works best for you.

Preparations

If you haven't already, install Python. I'm running version 3.7:

```
Robberts-MacBook-Air:~ robbert$ python
Python 3.7.2 (v3.7.2:9a3ffc0492, Dec 24 2018, 02:44:43)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more
information.
>>>
```

We will use a handy Python library called `Pillow` . Install it by using the following command in your terminal (*Not sure how to do this? Read this first)*:

```
$ pip install pillow
```

All set! Let's edit some photos.

On To The Fun Stuff!

We'll be writing some code to get everything working. You can code along if you like, either in mac terminal, windows command prompt, or an IDE, but it's not required.

Our first step is to import a couple of modules from the PIL library. The PIL (Python Imaging Library) is what does the actual hard work for us when it comes to manipulating photos. We can then leverage these modules to add a watermark to our photos.

```
from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont
```

After that, we use the first module image to open an image. Opening an image makes it possible to perform operations on it.

```
photo = Image.open(‘yourimage/filename.jpg’)

#Store image width and height
w, h = photo.size
print(w,h)
```

We’ve just opened a photo, saved the width and the height of it and printed that. In my case it resulted in:

```
4032 3024
```

Next, we make the image drawable and set a font for our copyright text:

```
# make the image editable
drawing = ImageDraw.Draw(photo)
font = ImageFont.truetype(“RobotoBlack.ttf”, 68)
```

Note: I had some issues with my local fonts, so what I did is download the Roboto font from google fonts (here) and save the font file in the same directory as where I saved the Python script.

Great, we have made the image drawable and set the font to Roboto and size to 68px.

Next is to set the copyright text and at the same time store the width and height of that text (we will need the width and height of the text and of the image to calculate the correct position for the watermark):

```
#get text width and height

 text = “© Robbert Brouwers   “
 text_w, text_h = drawing.textsize(text, font)
```

Knowing the image width and height and the text width and height, I can calculate where the copyright text will be placed and store that in a variable called `pos` :

```
pos = w — text_w, (h — text_h) — 50
```

This positions the text against the right side of the image (since it starts exactly the width of the text from it) and 50 pixels from the bottom (hence the -50). Play around with these settings later to see how different positions look.

My initial plan was to give the text a background color and make that a bit transparent. However, I that it’s not possible to make those changes to the text object in PIL. My workaround is to create a new image that has the size of the copyright text and give that a background color and opacity, then place the text in this image and place this complete image on top of the original photo.

```
c_text = Image.new('RGB', (text_w, (text_h)), color = '#000000')
drawing = ImageDraw.Draw(c_text)

drawing.text((0,0), text, fill="#ffffff", font=font)
c_text.putalpha(100)
```

What we’ve done above is create a new image with the width and height of the copyright text and the color black (#000000), then make this new image drawable by calling `ImageDraw.Draw` on it.

Then, starting at position 0,0 of our newly created image, we put the copyright text, make it white (#ffffff) and assign the font as defined earlier.

To finish the copyright image, we apply an alpha with the `putalpha()` function to make it slightly transparent. Again, play around with the input value of 100 to test different transparency settings.

Almost there. We now have our photo and a copyright image that we want to place on top of the photo on the position we've defined as `pos`.

```
photo.paste(c_text, pos, c_text)
photo.save('yourimage/filename_out.jpg')
```

The paste function takes three parameters: the image to paste on our photo, the position (coordinates) and a mask. The mask ensures that we apply the alpha from our image (more on that in the docs).

The last step saves our image to a desired location on our computer.

Excellent! We have our script that decides **what** the copyright should say, **how** it should look, and **where** it should be applied. Our final steps are to allow us to read in a directory of photos and output each one of them with the watermark.

But first, here's the complete script so far:

```
from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont
def copyright_apply(input_image_path,
                   output_image_path,
                   text):
    photo = Image.open(input_image_path)

    #Store image width and height
    w, h = photo.size


    # make the image editable
    drawing = ImageDraw.Draw(photo)
    font = ImageFont.truetype("RobotoBlack.ttf", 68)

    #get text width and height
    text = "© " + text + "   "
    text_w, text_h = drawing.textsize(text, font)

    pos = w - text_w, (h - text_h) - 50

    c_text = Image.new('RGB', (text_w, (text_h)), color = '#000000')
    drawing = ImageDraw.Draw(c_text)

    drawing.text((0,0), text, fill="#ffffff", font=font)
    c_text.putalpha(100)

    photo.paste(c_text, pos, c_text)
    photo.save(output_image_path)
```

I’ve changed a couple of things to make it easier to use it for a complete directory of photos.

First, I wrapped everything into a function called `copyright_apply()` . The idea of this is that we can call this function for each individual photo and loop over the list of photos in this way.

Second, I have changed the hardcoded variables for the input path, output path, and the copyright text. This way we can input the filename and automatically output it to another folder so we don’t overwrite original images. The text will take any string input and then prefix it with the ‘©’ sign and a couple of spaces after.

Getting Your List of Photos into Python

I have created a folder called `python_images` and in that folder I’ve created two more folders: `in` and `out` . In the `in` folder I’ve placed two photos that I would like to apply a watermark to.

```
import glob
list = glob.glob("python_images/in/*.*")
print(list)

>>>>['python_images/in/9C9595ED-BED3-4856-A74A-046DA4F54427.jpg',
'python_images/in/91F94C10-E1C4-471C-B89C-3E7FE78C2032.jpg']
```

`glob` is another handy library that allows us to get all filenames in a specific folder. Great! That means we can process these photos one by one and pass them into the function we’ve created above:

```
for photo in list:
 #create output path based off input path
 out = photo.replace(‘in’,’out’)
 copyright_apply(photo,
 out,
 ‘Image editing is awesome!’)
```

That’s it!

I have saved it as `copyright.py` and anytime I want to apply watermarks I make sure the correct folders are assigned in the script and run (in mac terminal, windows command prompt or an IDE/Jupyter Notebook):

```
python copyright.py
```

This will neatly output the photos with watermark into the out folder!

Full Code (remember to adjust any filepaths in case you use different folders/names:

```
#!/usr/bin/env python
# coding: utf-8

# In[29]:

from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont
import glob
def copyright_apply(input_image_path,
 output_image_path,
 text):
 photo = Image.open(input_image_path)

 #Store image width and heigth
 w, h = photo.size
 #print(w, h )

# make the image editable
 drawing = ImageDraw.Draw(photo)
 black = (3, 8, 12)
 font = ImageFont.truetype(“RobotoBlack.ttf”, 68)

 #get text width and heigth

 text = “© “ + text + “ “
 text_w, text_h = drawing.textsize(text, font)

 pos = w — text_w, (h — text_h) — 50
```

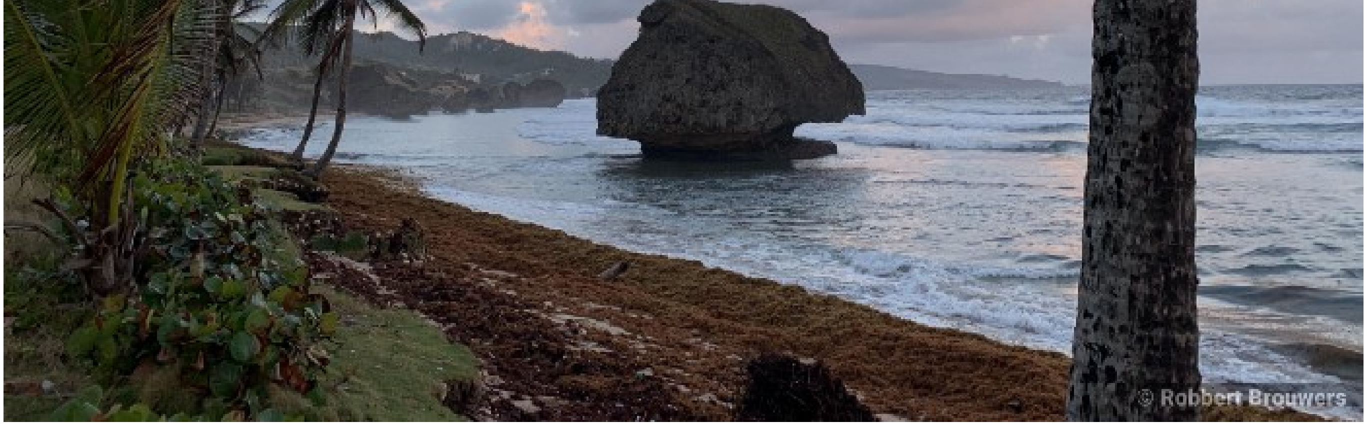
```
   c_text = Image.new(‘RGB’, (text_w, (text_h)), color = ‘#000000’)
   drawing = ImageDraw.Draw(c_text)

   drawing.text((0,0), text, fill=”#ffffff”, font=font)
   c_text.putalpha(100)


   photo.paste(c_text, pos, c_text)
   photo.save(output_image_path)

  list = glob.glob(“python_images/in/*.*”)

  for photo in list:

   out = photo.replace(‘in’,’out’)
   copyright_apply(photo,
   out,
   ‘Image editing is awesome!’)
```

What Next?

From here you can do a couple of things to improve the script:

- Set a dynamic font size, based on the image size.
- Ask the user for input folder and copyright text.
- Create an interface where a user can drop an image in and the script will output the watermarked image.

Look at my second picture that now has copyright on it:



Let me know any cool things you have created with the PIL library or any other library, for that matter!

Photography Python Programming Photos Image Processing