FakeAlbumCoverGame

December 20, 2018

Make Fake Album Cover Game

0.1 Table of Contents

Our goal is to create randomly generated album covers with:
Learn how to use the function display_cover
Loading a random page from Wikipedia
Extracting the Title of the Article
Displaying the Album Cover
Estimated Time Needed: 60 min
Inspiration: Fake Album Covers

Import libraries

```
In [7]: from IPython.display import Image as IPythonImage
        from PIL import Image
        from PIL import ImageFont
        from PIL import ImageDraw
```

Helper function to superimpose text on image

```
In [8]: def display_cover(top,bottom ):
            """This fucntoin
            """
            import requests

            name='album_art_raw.png'
            # Now let's make get an album cover.
            # https://picsum.photos/ is a free service that offers random images.
            # Let's get a random image:
            album_art_raw = requests.get('https://picsum.photos/500/500/?random')
            # and save it as 'album_art_raw.png'
            with open(name,'wb') as album_art_raw_file:
               album_art_raw_file.write(album_art_raw.content)
            # Now that we have our raw image, let's open it
            # and write our band and album name on it
            img = Image.open("album_art_raw.png")
            draw = ImageDraw.Draw(img)
```

```
    # We'll choose a font for our band and album title,
    # run "% ls /usr/share/fonts/truetype/dejavu" in a cell to see what else is availabl
    # or download your own .ttf fonts!
    band_name_font = ImageFont.truetype("/usr/share/fonts/truetype/dejavu/DejaVuSans-Bol
    album_name_font = ImageFont.truetype("/usr/share/fonts/truetype/dejavu/DejaVuSansMon

    # the x,y coordinates for where our album name and band name text will start
    # counted from the top left of the picture (in pixels)
    band_x, band_y = 50, 50
    album_x, album_y = 50, 400

    # Our text should be visible on any image. A good way
    # of accomplishing that is to use white text with a
    # black border. We'll use the technique shown here to draw the border:
    # https://mail.python.org/pipermail/image-sig/2009-May/005681.html
    outline_color ="black"

    draw.text((band_x-1, band_y-1), top, font=band_name_font, fill=outline_color)
    draw.text((band_x+1, band_y-1), top, font=band_name_font, fill=outline_color)
    draw.text((band_x-1, band_y+1), top, font=band_name_font, fill=outline_color)
    draw.text((band_x+1, band_y+1), top, font=band_name_font, fill=outline_color)

    draw.text((album_x-1, album_y-1), bottom , font=album_name_font, fill=outline_color)
    draw.text((album_x+1, album_y-1), bottom , font=album_name_font, fill=outline_color)
    draw.text((album_x-1, album_y+1), bottom , font=album_name_font, fill=outline_color)
    draw.text((album_x+1, album_y+1), bottom , font=album_name_font, fill=outline_color)

    draw.text((band_x,band_y),top,(255,255,255),font=band_name_font)
    draw.text((album_x, album_y),bottom,(255,255,255),font=album_name_font)

    return img
```

```
In [9]: % ls /usr/share/fonts/truetype/dejavu
```

```
UsageError: Line magic function `%` not found.
```

0.2 1) Learn how to use the function display_cover

The function **display_cover** selects a random image from https://picsum.photos/ and will help us superimpose two strings over the image. The parameter **top** is the string we would like to superimpose on the top of an image. The parameter bottom is the string we would like to display on the bottom of the image. The function does not return the image but returns an object of type Image from the Pillow library; the object represents a PIL image.

```
In [10]: img=display_cover(top='top',bottom='bottom')
```

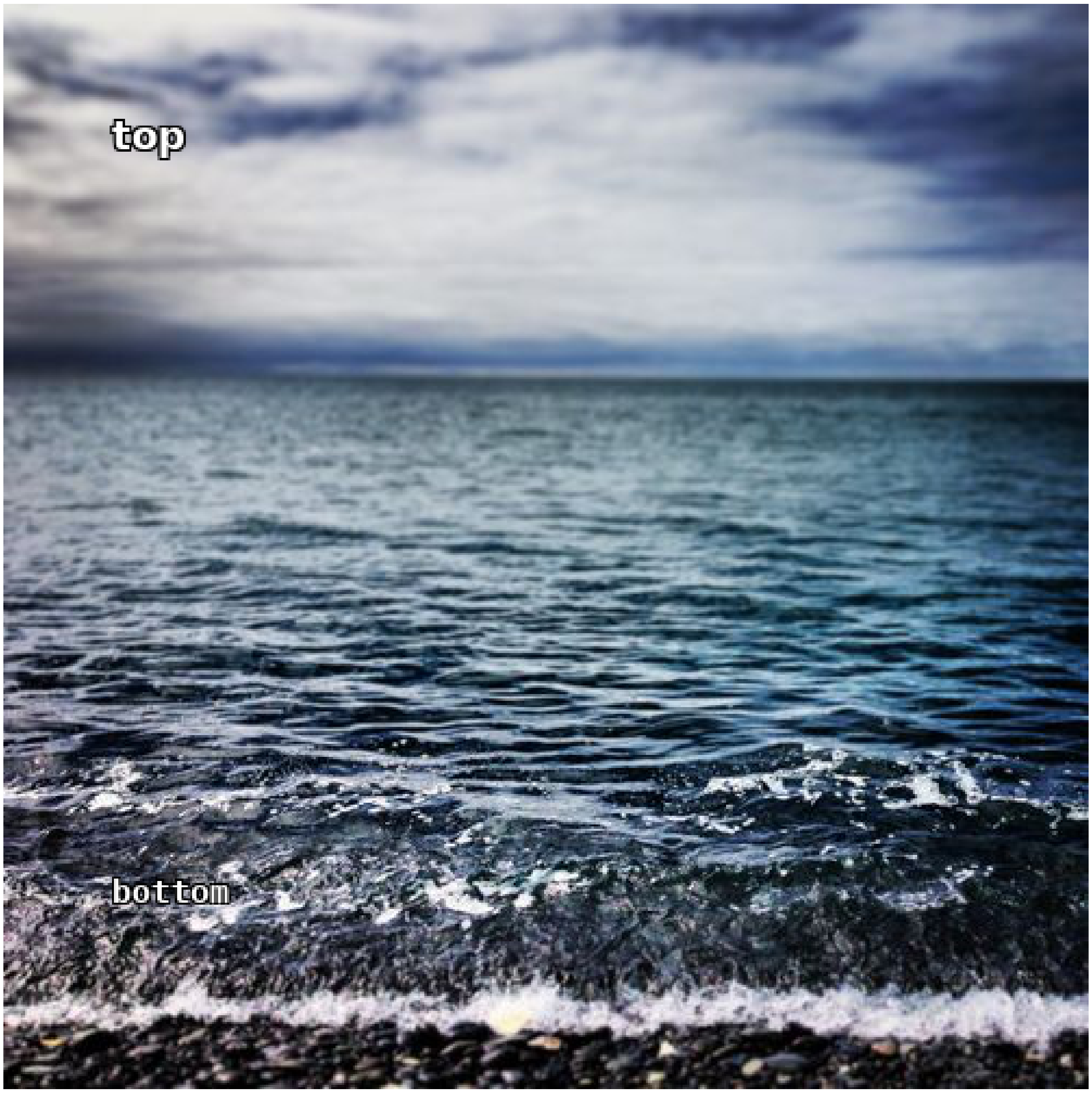
To save the image, we use the method **save** . The argument is the file name of the image we would like to save in this case 'sample-out.png'

```
In [11]: img.save('sample-out.png')
```

Finely we use **IPythonImage** to read the image file and display the results.

```
In [12]: IPythonImage(filename='sample-out.png')
```

```
Out[12]:
```



Question 1) Use the **display_cover** function to display the image with the name Python on the top and Data Science on the bottom. Save the image as **'sample-out.png'**.

```
In [13]: img=display_cover(top='Python',bottom='Data Science')
```

```
In [14]: img.save('Py-sample-out.png')
In [15]: IPythonImage(filename='Py-sample-out.png')
```

Out[15]:

0.3 Part 2: Loading a random page from Wikipedia

In this project, we will use the request library, we used it in the function **display_cover**, but you should import the library in the next cell.

```
In [16]: import requests
```

The following is the URL to the page

```
In [17]: wikipedia_link='https://en.wikipedia.org/wiki/Special:Random'
```

Question 2) Get Wikipedia page is converted to a string

Use the function **get** from the **requests** library to download the Wikipedia page using the **wikipedia_link** as an argument. Assign the object to the variable **raw_random_wikipedia_page**.

```
In [ ]: #hint: requests.get()
```

```
In [18]: raw_random_wikipedia_page = requests.get(wikipedia_link)
```

```
In [19]: box = []
         for items in raw_random_wikipedia_page:
             box.append(items)
             page = box
```

Use the data attribute **text** to extract the XML as a text file a string and assign the result variable **page**:

```
In [20]: print(page)
```

```
[illegible]
```

1 Part 3: Extracting the Title of the Article

Question 3 (part 1) Use the title of the Wikipedia article as the title of the band. The title of the article is surrounded by the XML node title as follows: **<title>title - Wikipedia</title>** . For example, if the title of the article was Python we would see the following: **<title>Python - Wikipedia</title>**. Consider the example where the title of the article is Teenage Mutant Ninja Turtles the result would be: **<title>Teenage Mutant Ninja Turtles - Wikipedia</title>**. The first step is to find the XML node **<title>** and **</title>**indicating the start and end of the title. The string function **find** maybe helpful, you can also use libraries like **xlxml**.

```
In [21]: import re

         pattern = r'>.*<'
         found = re.findall(pattern, str(page))
         string = found[0:1]

         for thing in found:
             if 'title' in thing:
                 target = thing[50:200]
                 find = re.search(r'<title.*?>(.+)<.*title>',str(target)).group(0)
                 print(find)
```

```
<title>Glycoside hydrolase family ', b'92 - Wikipedia</title>
```

Question 3 (part 2) Next get rid of the term ** - Wikipedia** from the title and assign the result to the **band_title** For example you can use the function or method **strip** or **replace**.

```
In [22]: if "Wikip', b'edia" in find:
             band_title = find.replace("Wikip', b'edia", "")
             print('band_title')
             print(band_title)
         else:
             if 'Wikipedia' in find:
                 band_title = find.replace('Wikipedia', '')
                 print('band_title')
                 print(band_title)
```

```
band_title
<title>Glycoside hydrolase family ', b'92 - </title>
```

Question 4) Repeat the second and third step, to extract the title of a second Wikipedia article but use the result to **album_title**

```
In [23]: if "Wikip', b'edia" in find:
             album_title = find.replace("Wikip', b'edia", "")
             print('album_title')
             print(album_title)
         else:
             if 'Wikipedia' in find:
                 album_title = find.replace('Wikipedia', '')
                 print('album_title')
                 print(album_title)
```

```
album_title
<title>Glycoside hydrolase family ', b'92 - </title>
```

If you did everything correct the following cell should display the album and band name:

```
In [24]: print("Your band: ", band_title)
         print("Your album: ", album_title)
```

```
Your band:  <title>Glycoside hydrolase family ', b'92 - </title>
Your album:  <title>Glycoside hydrolase family ', b'92 - </title>
```

1.1 Part 4: Displaying the Album Cover

Use the function **display_cover** to superimpose the band and album title over a random image, assign the result to the variable **album_cover** .

Question 5) use the function display_cover to display the album cover with two random article titles representing the name of the band and the title of the album.

```
In [25]: band = band_title
         album = album_title
         img=display_cover(top=band,bottom=album)
```

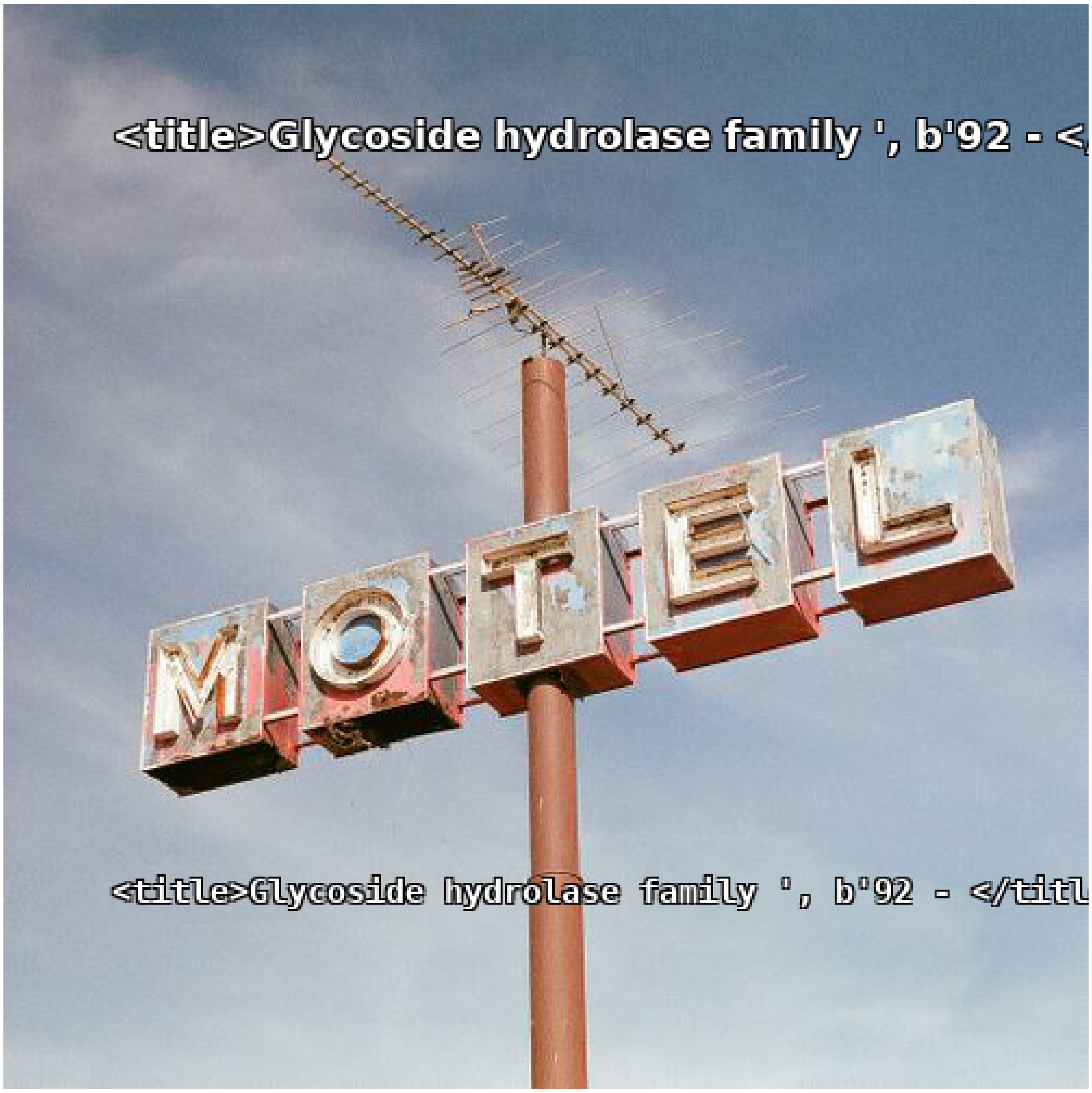
Use the method save to save the image as **sample-out.png**:

```
In [26]: img.save('sample-out.png')
```

Use the function **IPythonImage** to display the image

```
In [27]: IPythonImage(filename='sample-out.png')
```

```
Out[27]:
```


<title>Glycoside hydrolase family ', b'92 - <
MOTEL
<title>Glycoside hydrolase family ', b'92 - </titl

1.1.1 About the Authors:

James Reeve James Reeves is a Software Engineering intern at IBM.

Joseph Santarcangelo has a PhD in Electrical Engineering, his research focused on using machine learning, signal processing, and computer vision to determine how videos impact human cognition. Joseph has been working for IBM since he completed his PhD.