Tables of content

Table of Contents

List of Figures

1.0. Introduction 2

1.1. Purpose 2

1.2. Scope of Project 3

1.3. Glossary 4

1.4. References 4

1.5. Overview of Document 5

2.0 **Graphical User Interface** 6

2.1 **Tkinter**

2.1.1 Fundamentals Of Tkinter6

2.1.1Tkinter Widgets 7

2.1.2 Button 8

2.1.3 Label 8

2.1.4 Message 9

2.1.5 Top Level9

2.1.6 Panned Window9

1. Metadata Management 10
   1. Existing metadata standards
   2. Exif - Exchangeable image file format
   3. Metadata formats within image files
   4. Handling a single metadata format
   5. Exif metadata description11
   6. Handling Exif
   7. More complex reconciliation in popular image formats
   8. Writing Exif 12
2. Image File Formats 13
   1. JPEG
   2. GIF
   3. PNG 14
3. PYTHON IMAGING LIBRARY 15
   1. Image Archives
   2. Image Display
   3. Using the Image Class
   4. Capabilities of Pil
4. HARDWARE AND SOFTWARE REQUIREMENT 16
   1. Operating System
   2. Hardware Interfaces
   3. Software Interfaces
5. CODE IMPLEMENATION
6. CONCLUSION

1. **Introduction** 
   1. **Purpose**

The purpose of this documents is to present a detailed metadata description of the Image Forensic. Image metadata is text information pertaining to an image file that is embedded into the file or contained in a separate file that is associated with it. It includes details relevant to the image itself as well as information about its production. Some metadata is generated automatically by the device capturing the image. In this case, metadata refers to additional information about the actual images, which is stored in the image files along with the images.

Metadata, often referred to as “data about data,” provides interesting information that supplements the primary content of digital documents. Metadata has become a powerful tool to organize and search through the growing libraries of image, audio and video content that users are producing and consuming. This is especially important in the area of digital photography where, despite the increased quality and quantity of sensor elements, it is not currently practical to organize and query images based only on the millions of image pixels. Instead, it is best to use metadata properties that describe what the photo represents and where, when and how the image was taken

Digital images are stored in a variety of common file formats such as, JPEG as well as proprietary formats such as RAW. Each file format has distinct rules regarding how metadata formats must be stored within the file.

* 1. **Scope of Project**

This software system can be used for finding the image metadata for forensic and the migration of the related issues that use of the device for crop the image, date and time of the image , last modification of the image date and time . Metadata is not visible when viewing data in a number of forms such as a word document or an image. It is, however, an important consideration in the discovery of information for use in digital forensic investigations. Different types of documents and files have a number of formats and types of metadata, which can be used to discover the properties of a file, document or network activity. Moreover, Metadata is useful in many circumstances, where it can provide collaboration evidence of between groups of people, because some of them are not aware of which type of information is stored within their document. Thus, the digital forensics investigator can access to this hidden document information. In legal cases, the identification of relevant digital evidence is crucial for supporting the case, verification and an examination existing legal argument forms. In this work, we show how to use the different formats and types of metadata in order to validate the legal argument for relevant evidence.

The intention of this document is to use and extend existing standards to address the key organizational metadata questions that most consumers have:

* Who is involved with this image? (who took it, who owns it, who’s in it?)
* What is interesting about this image?
* Where is this image from?
* When was this image created or modified?

The goal of this document is to provide best practices for solving interoperability issues in the consumer space.

When we look at the “four Ws” (who, what, where, when), it is clear that this data can range from highly precise (e.g. GPS latitude/longitude) to extremely vague or context-dependent (e.g. “In my back yard”). This document does not try to solve the difficult semantic issues around this problem; rather, it tries to ensure that semantically equivalent metadata is identified across standards, and if it exists, that best practice is followed to use semantically well-defined properties for that metadata. The key notions of “reconciliation” and “rationalization” for the consumer space define the scope of this initial work.

* 1. **Glossary:**

|  |  |
| --- | --- |
| Term | Definition |
| Exif | Exchangeable image file format” – a standard for image file formats, jointly managed by Japan Electronics and Information Technology industries Association (JEITA) and Camera and Imaging Products Association (CIPA) |
| JPEG | A file format, widely used in image and photography workflows |
| Tkinter | Python GUI Library, to make windows in python programming language. |
| Time | Showing time and date on the window |
| UTF-8 | UTF-8 is a byte-oriented encoding form of Unicode |
| PIL | Image Library |
| PIL.ExifTags | Getting metadata |

1.4 **References**

* <https://en.wikipedia.org/wiki/Metadata>
* <https://developer.here.com/blog/getting-started-with-geocoding-exif-image-metadata-in-python3>
* <https://www.hackers-arise.com/post/2017/05/29/digital-forensics-part-9-extracting-exif-data-from-graphics-files>
* <https://www.researchgate.net/publication/329880328_Authentication_of_Digital_Image_using_Exif_Metadata_and_Decoding_Properties>

1.5 Overview of Document

The next chapter, the Overall Description section, of this document

1. **Graphical User Interface**

**GUI is**  nothing but a desktop application which helps you to interact with the computers. They are used to perform different tasks in the desktops, laptops and other electronic devices.

## **2.1 What is Tkinter?**

**Tkinter** is actually an inbuilt **Python** module used to create simple **GUI** apps. It is the most commonly used module for **GUI** apps in the **Python**.

**2.2.1** **Fundamentals Of Tkinter**

Consider the following diagram, it shows how an application actually executes in Tkinter:

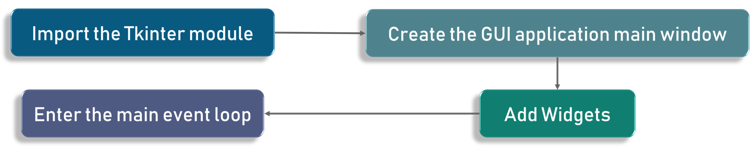


Figure 1

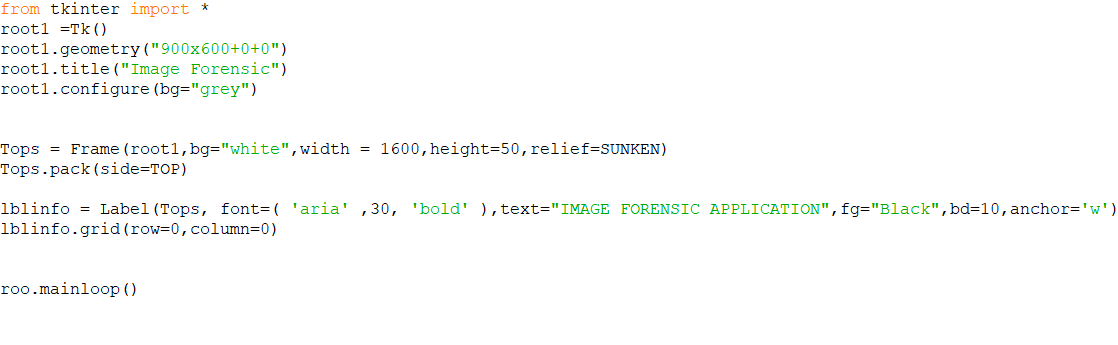
To start out with, we first import the Tkinter model. Followed by that, we create the main window. It is in this window that we are performing operations and displaying visuals and everything basically.

These are the 2 keywords:

* Widgets
* Main Event Loop

An event loop is basically telling the code to keep displaying the window until we manually close it. It runs in an infinite loop in the back-end.

Check out the following code for better clarity:



As you can see, we are importing the Tkinter package and defining a window. Followed by that, we are giving a image forensic title which is shown on the title tab whenever you open an application.

For example, Microsoft Word is shown on the title tab when you open a word application, correct? Similarly here we call it GUI. We can call it anything we want based on the requirement.

Lastly, we have a label. A label is nothing is but what output needs to be shown on the window. In this case as you can already see , it is image forensic application

Check out the output for the above code:



**2.1.1** **Tkinter Widgets**

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

* Button - The Button widget is used to display buttons in your application.
* Frame - The Frame widget is used as a container widget to organize other widgets.
* Label- The Label widget is used to provide a single-line caption for other widgets. It can also contain images.
* [Message](https://www.tutorialspoint.com/python/tk_message.htm)- The Message widget is used to display multiline text fields for accepting values from a user.
* [Toplevel](https://www.tutorialspoint.com/python/tk_toplevel.htm)- The Toplevel widget is used to provide a separate window container.
* [PanedWindow](https://www.tutorialspoint.com/python/tk_panedwindow.htm)- A PanedWindow is a container widget that may contain any number of panes, arranged horizontally or vertically.
* [LabelFrame](https://www.tutorialspoint.com/python/tk_labelframe.htm)- A labelframe is a simple container widget. Its primary purpose is to act as a spacer or container for complex window layouts.
* [tkMessageBox](https://www.tutorialspoint.com/python/tk_messagebox.htm)- This module is used to display message boxes in your applications.

**2.1.2 Button**:

To add a button in your application, this widget is used.  
The general syntax is:

w=Button(master, option=value)

master is the parameter used to represent the parent window.  
There are number of options which are used to change the format of the Buttons. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **activebackground**: to set the background color when button is under the cursor.
* **activeforeground**: to set the foreground color when button is under the cursor.
* **bg**: to set he normal background color.
* **command**: to call a function.
* **font**: to set the font on the button label.
* **image**: to set the image on the button.
* **width**: to set the width of the button.
* **height**: to set the height of the button.

**2.1.3 Label**:

It refers to the display box where you can put any text or image which can be updated any time as per the code.  
The general syntax is:

w=Label(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **bg**: to set he normal background color.
* **bg** to set he normal background color.
* **command**: to call a function.
* **font**: to set the font on the button label.
* **image**: to set the image on the button.
* **width**: to set the width of the button.
* **height**” to set the height of the button.

**2.1.4 Message**:

It refers to the multi-line and non-editable text. It works same as that of Label.  
The general syntax is:

w = Message(master, option=value)

master is the parameter used to represent the parent window.

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **bd**: to set the border around the indicator.
* **bg**: to set he normal background color.
* **font**: to set the font on the button label.
* **image**: to set the image on the widget.
* **width**: to set the width of the widget.
* **height**: to set the height of the widget.

**2.1.5 TopLevel:**

This widget is directly controlled by the window manager. It don’t need any parent window to work on.The general syntax is:

w = TopLevel(master, option=value)

There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **bg**: to set he normal background color.
* **bd**: to set the size of border around the indicator.
* **cursor**: To appear the cursor when the mouse over the menubutton.
* **width**: to set the width of the widget.
* **height**: to set the height of the widget.

**2.1.6 Panned Window :**

It is a container widget which is used to handle number of panes arranged in it. The general syntax is:

w = PannedWindow(master, option=value)

master is the parameter used to represent the parent window.  
There are number of options which are used to change the format of the widget. Number of options can be passed as parameters separated by commas. Some of them are listed below.

* **bg**: to set he normal background color.
* **bd**: to set the size of border around the indicator.
* **cursor**: To appear the cursor when the mouse over the menubutton.
* **width**: to set the width of the widget.
* **height**: to set the height of the widget.

1. **Metadata Management**

Metadata is an essential part of image and photography based workflows. Cameras capture device metadata while taking pictures. Operating systems and other software subsequently read metadata to build up catalogs and offer effective search capabilities. In addition to this, the user is able to enhance this workflow with their own metadata that may be stored either inside the file or within caching or database systems. In the context of consumer image-based workflows, the existence of different metadata standards leads to interoperability issues when using various devices, operating systems and software tools. Although the majority of metadata properties are unique, there are a number of properties which overlap across several metadata standards and cause interoperability issues as a consequence. The goal of this section is to identify those overlapping properties and provide guidance on how to handle them correctly across the different metadata formats. After a brief overview of existing metadata standards, this chapter introduces the most common metadata properties in the context of consumer workflows. To ensure best interoperability across software and hardware systems, a general reconciliation mechanism is then discussed. Finally, the chapter will close with a detailed analysis of each focus area and discuss specific technical issues and obstacles.

* 1. **Existing metadata standards**

This section gives an overview of the existing metadata formats. As described in the introduction, this document will focus on photography workflows in the context of the consumer, so the choice of discussed metadata formats covers Exif.

* 1. **Exif - Exchangeable image file format**

The Exif standard has been jointly managed by the Japan Electronics and Information Technology industries Association (JEITA1) and Camera and Imaging Products Association (CIPA). In particular, the Exif image interchange format defines a set of TIFF tags that describe photographic images, and is widely used by digital cameras. Exif metadata can be found in TIFF, JPEG, and PSD files.

* 1. **Metadata formats within image files**

There are three metadata formats widely used in the industry:

* Exif
* IPTC-IIM
* XMP
  1. **Handling a single metadata format**

In the simplest scenario, a given metadata property is only defined in a single metadata format. This is, for example, true for the rating property - this value should always be read and written into the corresponding XMP (xmp:Rating) field. No further reconciliation is necessary. Also, there are a variety of properties defined in Exif (device properties) or in IPTC-IIM (workflow properties) that are unique to the container and won't be reconciled amongst the other formats.

* 1. **Exif metadata description**

Exchangeable image file format (Exif) was created by the Japan Electronic Industries Development Association (JEIDA) [38], and is a format to describe metadata. Images can have more or less metadata attached to them. Due to the fact that Exif information stores information about the image, this information can pose a privacy issue, especially concerning information describing time and date a picture was taken, the location the picture was taken and the original unedited thumbnail of the picture.

The above picture shows Exif data for a typical photo taken with a mobile device, and is meant to illustrate the type of information which can be derived from the metadata associated with the photo.

**3.6 Handling Exif :**

This chapter discusses reconciliation guidance in Exif .. Upon writing Exif metadata, a Changer MUST update all formats that were originally present in the file. If not all of the formats were originally present, a Changer MAY choose to write the complete set.

* 1. **More complex reconciliation in popular image formats :**

Finding and interpreting the metadata embedded in JPEG, PNG files is complicated by the fact that all three file formats contain the same kinds of metadata (XMP, Exif/TIFF, and IPTC-IIM), but store it slightly differently. For example, all of the kinds of metadata can be contained in Photoshop Image Resources (PSIRs), and all three file formats (JPEG, TIFF, and PSD) can contain PSIRs. However, the specific contents of the PSIRs are different when contained in different image file formats. Each type of metadata is stored inside the PSIR for some file formats, and separately for others. However, the recursive embedding of metadata formats is more a theoretical possibility, so this document will simplify this process by identifying the three most relevant places to find Exif.

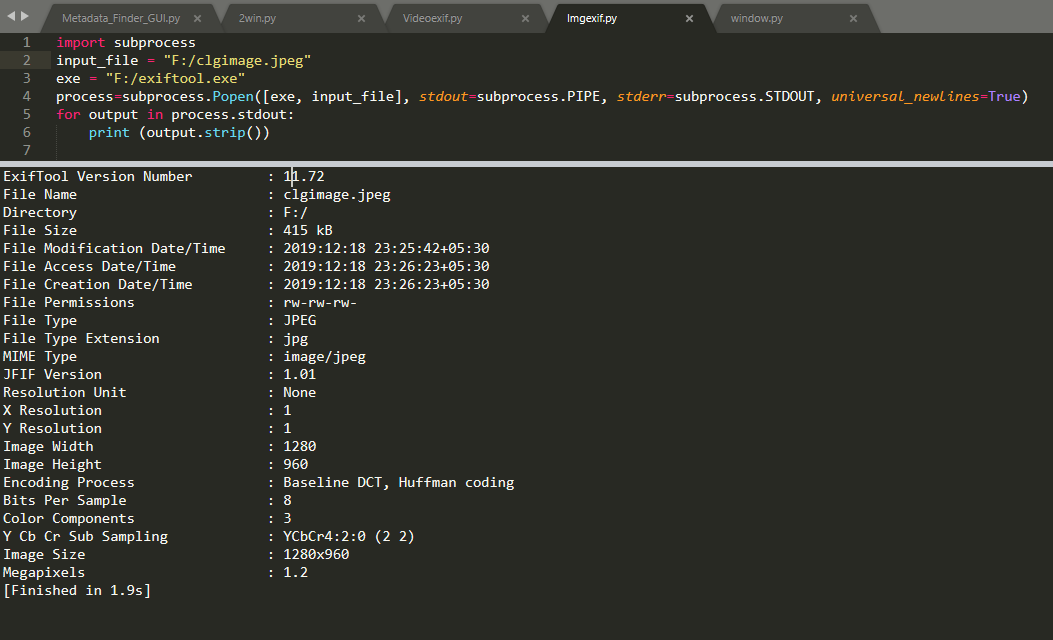
**3.8** **Writing Exif .**

In the context of the actor definitions, the following rules describe the guidance on how to write Exif:

**Creator**

Exif metadata MUST be created.

**Changer**

* Exif metadata SHOULD be consumed according the reconciliation guidance .
* In the case the file format does support Exif natively, Exif and TIFF device properties (e.g. XResolution, YResolution, WhitePoint, etc.)
* Exif metadata is formatted as a TIFF stream, even in JPEG files. TIFF streams have an explicit indication of being big endian or little endian. A **Changer** SHOULD preserve the existing byte-order.
* Exif string values SHOULD be written as UTF-8. However, clients MAY write ASCII to allow broader interoperability2.

**4.0 Image File Formats**

**4.1 JPEG**

JPEG is short for Joint Photographic Experts Group, and is the most popular among the image formats used on the web. JPEG files are very ‘lossy’, meaning so much information is lost from the original image when you save it in a JPEG file.

This is because JPEG discards most of the information to keep the image file size small; which means some degree of quality is also lost.

**Pros of JPEG:**

* 24-bit color, with up to 16 million colors
* Rich colors, great for photographs that need fine attention to color detail
* Most used and most widely accepted image format
* Compatible in most OS (Mac, PC, Linux)

**Cons of JPEG:**

* They tend to discard a lot of data
* After compression, JPEG tends to create artifacts
* Cannot be animated
* Does not support transparency

**4.2 GIF**

GIF, short for Graphics Interchange Format, is limited to the 8 bit palette with only 256 colors. GIF is still a popular image format on the internet because image size is relatively small compared to other image compression types.

Compared to JPEG, it is lossless and thus more effective with compressing images with a single color, but pales in detailed or dithered pictures. In other words, GIF is lossless for images with 256 colors and below. So for a full color image, it may lose up to 99.998% of its colors.

One edge of the GIF image format is the interlacing feature, giving the illusion of fast loading graphics. When it loads in a browser, the GIF first appears to be blurry and fuzzy, but as soon

as more data is downloaded, the image becomes more defined until all the date has been downloaded.

**Pros of GIF:**

* Can support transparency
* Can do small animation effects
* ‘Lossless’ quality–they contain the same amount of quality as the original, except of course it now only has 256 colors
* Great for images with limited colors, or with flat regions of color

**Cons of GIF:**

* Only supports 256 colors
* It’s the oldest format in the web, having existed since 1989. It hasn’t been updated since, and sometimes, the file size is larger than PNG.

**4.3 PNG**

PNG or (Portable Network Graphics) is a recently introduced format, so not everyone is familiar with it. But PNG has been approved as a standard since 1996. It is an image format specifically designed for the web. PNG is, in all aspects, the superior version of the GIF.

Just like the GIF format, the PNG is saved with 256 colors maximum but it saves the color information more efficiently. It also supports an 8 bit transparency.

PNG was actually created for the intent to replace the GIF as an image format that doesn’t require a patent license. PNG can support 24 bit RGB color images, grayscale images, both with and without alpha channels. RGB cannot support CMYK color spaces, and is not designed for print graphics.

**Pros of PNG:**

* Lossless, so it does not lose quality and detail after image compression
* In a lot ways better then GIF. To start, PNG often creates smaller file sizes than GIF
* Supports transparency better than GIF

**Cons of PNG:**

* Not good for large images because they tend to generate a very large file, sometimes creating larger files than JPEG.
* Unlike GIF however, it cannot be animated.

**5.0 PYTHON IMAGING LIBRARY**

The Python Imaging Library adds image processing capabilities to your Python interpreter. This library provides extensive file format support, an efficient internal representation, and fairly powerful image processing capabilities.

The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool. Let’s look at a few possible uses of this library:

**5.1 Image Archives**

The Python Imaging Library is ideal for for image archival and batch processing applications. You can use the library to create thumbnails, convert between file formats, print images, etc.

The current version identifies and reads a large number of formats. Write support is intentionally restricted to the most commonly used interchange and presentation formats.

**5.2 Image Display**

The current release includes Tk Photo Image and Bitmap Image interfaces, as well as a Windows DIB interface that can be used with Python and other Windows-based toolkits. Many other GUI toolkits come with some kind of PIL support. For debugging, there’s also a show method which saves an image to disk, and calls an external display utility. 1.1.4 Image Processing

The library contains basic image processing functionality, including point operations, filtering with a set of built-in convolution kernels, and colour space conversions. The library also supports image resizing, rotation and arbitrary affine transforms.

There’s a histogram method allowing you to pull some statistics out of an image. This can be used for automatic contrast enhancement, and for global statistical analysis.

**5.3 Using the Image Class**

The most important class in the Python Imaging Library is the Image class, defined in the module with the same name. You can create instances of this class in several ways; either by loading images from files, processing other images, or creating images from scratch.

To load an image from a file, use the open function in the Image module.:

**5.4 Capabilities of PIL**

Pillow offers several standard procedures for image manipulation. These include:

* per-pixel manipulations,
* masking and transparency handling,
* image filtering, such as blurring, contouring, smoothing, or edge finding,
* image enhancing, such as sharpening, adjusting brightness, contrast or color,

**6.0 HARDWARE AND SOFTWARE REQUIREMENTS:**

**6.1 Operating System:**

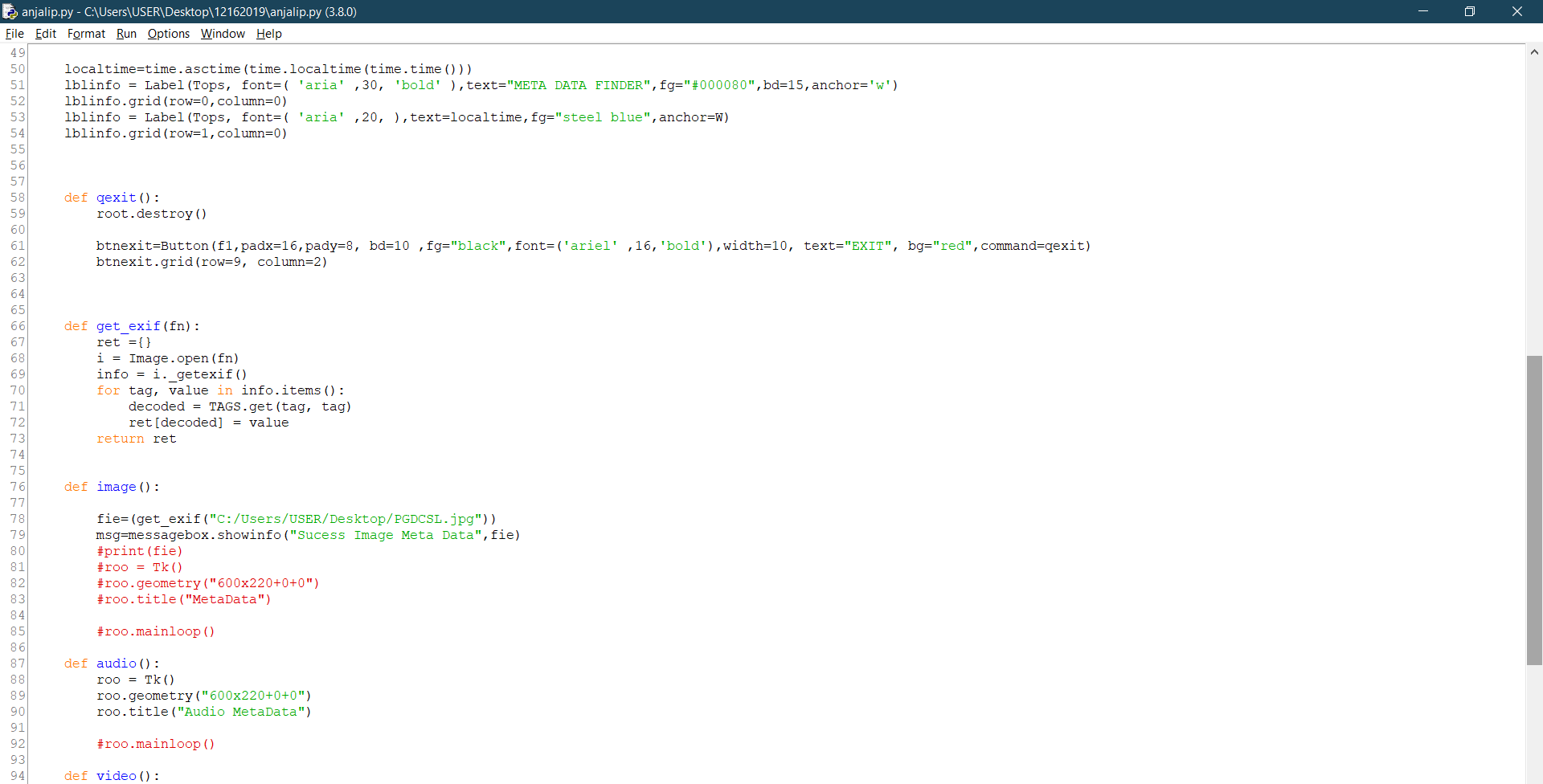
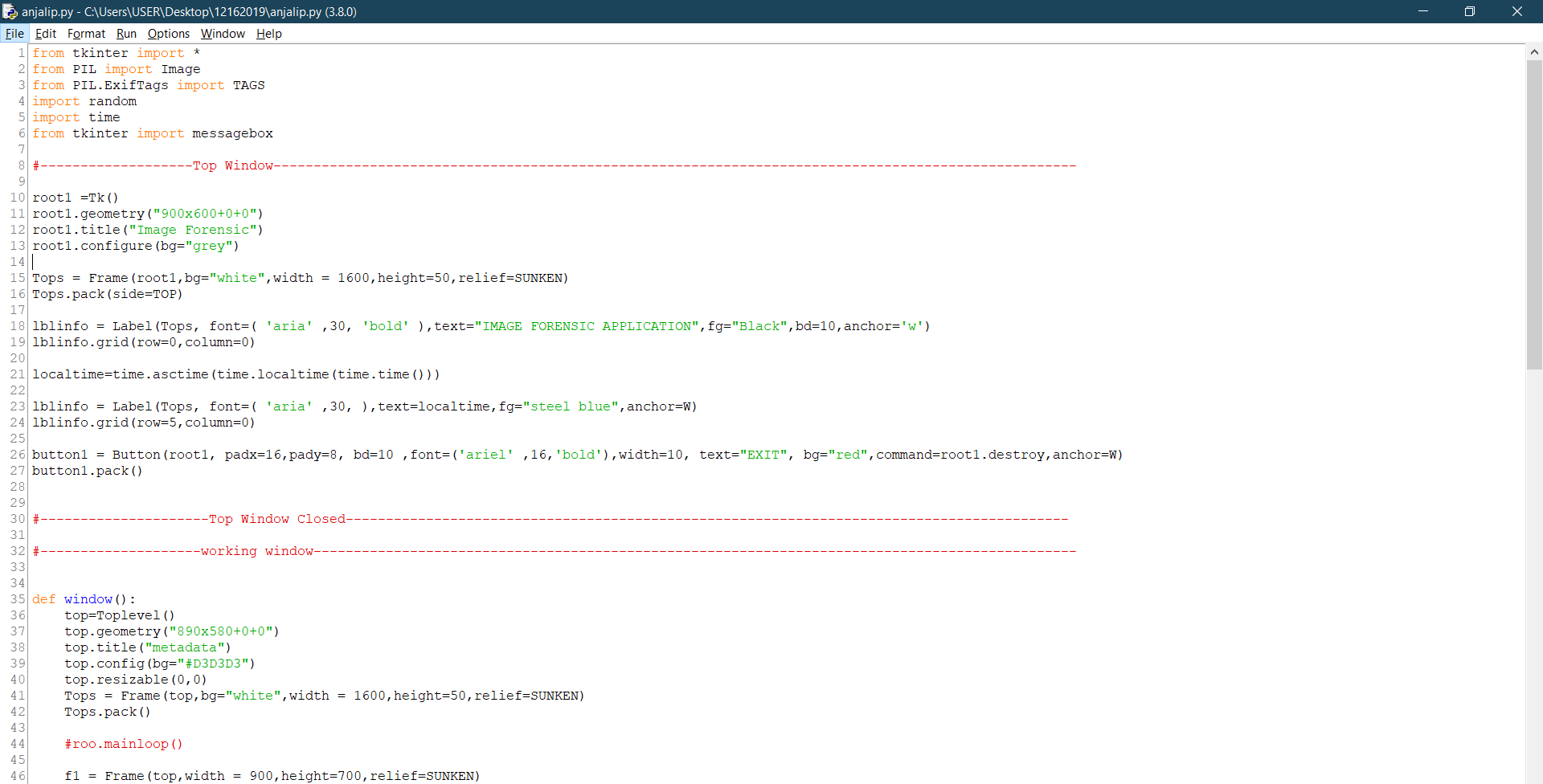
* Windows 7, Windows 8 or Windows 10
* Mac OSX 10.8, 10.9, 10.10 or 10.11
* Linux

**6.2 Hardware Interfaces**

* Processor (CPU) with 2 gigahertz (GHz) frequency or above
* A minimum of 2 GB of RAM
* Monitor Resolution 1024 X 768 or higher
* A minimum of 20 GB of available space on the hard disk

**6.3 Software:**

* Python 2.7 or above
* Sublime text 3
  + - 1. **CODE IMPLEMENATION :**

****

CONCLUSION :