Chapter 20

Ten Libraries You Need to Know About

In This Chapter

- Securing your data using cryptology
- Working with databases
- Getting to where you're going and finding new locations
- Presenting the user with a GUI
- Creating tables that users will enjoy viewing
- Working with graphics
- Finding the information you need
- Allowing access to Java code from your Python application
- Obtaining access to local network resources
- Using resources found online

Python provides you with considerable power when it comes to creating average applications. However, most applications aren't average and require some sort of special processing to make them work. That's where libraries come into play. A good library will extend Python functionality so that it supports the special programming needs that you have. For example, you might need to plot statistics or interact with a scientific device. These sorts of tasks require the use of a library.

One of the best places to find a library listing online is the UsefulModules site at `https://wiki.python.org/moin/UsefulModules`. Of course, there are many other places to look for libraries as well. For example, the article entitled "7 Python Libraries you should know about" (`http://doda.co/7-python-libraries-you-should-know-about`) provides you with a relatively complete description of the seven libraries its title refers to. If you're working on a specific platform, such as Windows, you can find platform-specific sites, such as Unofficial Windows Binaries for Python Extension Packages (`http://www.lfd.uci.edu/~gohlke/pythonlibs/`). The point is that you can find lists of libraries everywhere.

The purpose of this chapter isn't to add to your already overflowing list of potential library candidates. Instead, it provides you with a list of ten libraries that work on every platform and provide basic services that just about everyone will need. Think of this chapter as a source for a core group of libraries to use for your next coding adventure.

Developing a Secure Environment Using PyCrypto

Data security is an essential part of any programming effort. The reason that applications are so valued is that they make it easy to manipulate and use data of all sorts. However, the application must protect the data or the efforts to work with it are lost. It's the data that is ultimately the valuable part of a business — the application is simply a tool. Part of protecting the data is to ensure that no one can steal it or use it in a manner that the originator didn't intend, which is where cryptographic libraries such as PyCrypto (`https://www.dlitz.net/software/pycrypto/`) come into play.

The main purpose of this library is to turn your data into something that others can't read while it sits in permanent storage. The purposeful modification of data in this manner is called *encryption.* However, when you read the data into memory, a *decryption* routine takes the mangled data and turns it back into its original form so that the application can manage it. At the center of all this is the *key,* which is used to encrypt and decrypt the data. Ensuring that the key remains safe is part of your application coding as well. You can read the data because you have the key; no others can because they lack the key.

Interacting with Databases Using SQLAlchemy

A *database* is essentially an organized manner of storing repetitive or structured data on disk. For example, customer *records* (individual entries in the database) are repetitive because each customer has the same sort of information requirements, such as name, address, and telephone number. The precise organization of the data determines the sort of database you're using. Some database products specialize in text organization, others in tabular information, and still others in random bits of data (such as readings taken from a scientific instrument). Databases can use a tree-like structure or a flat-file configuration to store data. You'll hear all sorts of odd terms when you start looking into DataBase Management System (DBMS) technology — most of which mean something only to a DataBase Administrator (DBA) and won't matter to you.

The most common type of database is called a Relational DataBase Management System (RDBMS), which uses tables that are organized into records and fields (just like a table you might draw on a sheet of paper). Each *field* is part of a column of the same kind of information, such as the customer's name. Tables are related to each other in various ways, so creating complex relationships is possible. For example, each customer may have one or more entries in a purchase order table, and the customer table and the purchase order table are therefore related to each other.

An RDBMS relies on a special language called the Structured Query Language (SQL) to access the individual records inside. Of course, you need some means of interacting with both the RDBMS and SQL, which is where SQLAlchemy (`http://www.sqlalchemy.org/`) comes into play. This product reduces the amount of work needed to ask the database to perform tasks such as returning a specific customer record, creating a new customer record, updating an existing customer record, and deleting an old customer record.

Seeing the World Using Google Maps

Geocoding (the finding of geographic coordinates, such as longitude and latitude from geographic data, such as address) has lots of uses in the world today. People use the information to do everything from finding a good restaurant to locating a lost hiker in the mountains. Getting from one place to another often revolves around geocoding today as well. Google Maps (`https://pypi.python.org/pypi/googlemaps/`) lets you add directional data to your applications.

In addition to getting from one point to another or finding a lost soul in the desert, Google Maps can also help in Geographic Information System (GIS) applications. The "Helping People Decide on Location" section of Chapter 18 describes this particular technology in more detail, but essentially, GIS is all about deciding on a location for something or determining why one location works better than another location for a particular task. In short, Google Maps presents your application with a look at the outside world that it can use to help your user make decisions.

Adding a Graphical User Interface Using TkInter

Users respond to the Graphical User Interface (GUI) because it's friendlier and requires less thought than using a command-line interface. Many products out there can give your Python application a GUI. However, the most

commonly used product is TkInter (`https://wiki.python.org/moin/TkInter`). Developers like it so much because TkInter keeps things simple. It's actually an interface for the Tool Command Language (Tcl)/Toolkit (Tk) found at `http://www.tcl.tk/`. A number of languages use Tcl/Tk as the basis for creating a GUI.

You might not relish the idea of adding a GUI to your application. Doing so tends to be time consuming and doesn't make the application any more functional (it also slows the application down in many cases). The point is that users like GUIs, and if you want your application to see strong use, you need to meet user requirements.

Providing a Nice Tabular Data Presentation Using PrettyTable

Displaying tabular data in a manner the user can understand is important. From the examples you've seen throughout the book, you know that Python stores this type of data in a form that works best for programming needs. However, users need something that is organized in a manner that humans understand and that is visually appealing. The PrettyTable library (`https://pypi.python.org/pypi/PrettyTable`) makes it easy to add an appealing tabular presentation to your command-line application.

Enhancing Your Application with Sound Using PyAudio

Sound is a useful way to convey certain types of information to the user. Of course, you have to be careful in using sound because special-needs users might not be able to hear it, and for those who can, using too much sound can interfere with normal business operations. However, sometimes audio is an important means of communicating supplementary information to users who can interact with it (or of simply adding a bit of pizzazz to make your application more interesting).

One of the better platform-independent libraries to make sound work with your Python application is PyAudio (`http://people.csail.mit.edu/hubert/pyaudio/`). This library makes it possible to record and play back sounds as needed (such as a user recording an audio note of tasks to perform later and then playing back the list of items as needed).

Classifying Python sound technologies

It's important to realize that sound comes in many forms in computers. The basic multimedia services provided by Python (see the documentation at `https://docs.python.org/3/library/mm.html`) provide essential playback functionality. You can also write certain types of audio files, but the selection of file formats is limited. In addition, some modules, such as winsound (`https://docs.python.org/3/library/winsound.html`), are platform dependent, so you can't use them in an application designed to work everywhere. The standard Python offerings are designed to provide basic multimedia support for playing back system sounds.

The middle ground, augmented audio functionality designed to improve application usability, is covered by libraries such as PyAudio. You can see a list of these libraries at `https://wiki.python.org/moin/Audio`. However, these libraries usually focus on business needs, such as recording notes and playing them back later. Hi-fidelity output isn't part of the plan for these libraries.

Gamers need special audio support to ensure that they can hear special effects, such as a monster walking behind them. These needs are addressed by libraries such as PyGame (`http://www.pygame.org/news.html`). When using these libraries, you need higher-end equipment and have to plan to spend considerable time working on just the audio features of your application. You can see a list of these libraries at `https://wiki.python.org/moin/PythonGameLibraries`.

Working with sound on a computer always involves trade-offs. For example, a platform-independent library can't take advantage of special features that a particular platform might possess. In addition, it might not support all the file formats that a particular platform uses. The reason to use a platform-independent library is to ensure that your application provides basic sound support on all systems that it might interact with.

Manipulating Images Using PyQtGraph

Humans are visually oriented. If you show someone a table of information and then show the same information as a graph, the graph is always the winner when it comes to conveying information. Graphs help people see trends and understand why the data has taken the course that it has. However, getting those pixels that represent the tabular information onscreen is difficult, which is why you need a library such as PyQtGraph (`http://www.pyqtgraph.org/`) to make things simpler.

Even though the library is designed around engineering, mathematical, and scientific requirements, you have no reason to avoid using it for other purposes. PyQtGraph supports both 2D and 3D displays, and you can use it to generate new graphics based on numeric input. The output is completely interactive, so a user can select image areas for enhancement or other sorts of manipulation. In addition, the library comes with a wealth of useful widgets (controls, such as buttons, that you can display onscreen) to make the coding process even easier.

Unlike many of the offerings in this chapter, PyQtGraph isn't a free-standing library, which means that you must have other products installed to use it. This isn't unexpected because PyQtGraph is doing quite a lot of work. You need these items installed on your system to use it:

- Python version 2.7 or above
- PyQt version 4.8 or above (`https://wiki.python.org/moin/PyQt`) or PySide (`https://wiki.python.org/moin/PySide`)
- numpy (`http://www.numpy.org/`)
- scipy (`http://www.scipy.org/`)
- PyOpenGL (`http://pyopengl.sourceforge.net/`)

Locating Your Information Using IRLib

Finding your information can be difficult when the information grows to a certain size. Consider your hard drive as a large, free-form, tree-based database that lacks a useful index. Any time such a structure becomes large enough, data simply gets lost. (Just try to find those pictures you took last summer and you'll get the idea.) As a result, having some type of search capability built into your application is important so that users can find that lost file or other information.

A number of search libraries are available for Python. The problem with most of them is that they are hard to install or don't provide consistent platform support. In fact, some of them work on only one or two platforms. However, IRLib (`https://github.com/gr33ndata/irlib`) is written in pure Python, which ensures that it works on every platform. If you find that IRLib doesn't meet your needs, make sure the product you do get will provide the required search functionality on all the platforms you select and that the installation requirements are within reason.

IRLab works by creating a search index of whatever information you want to work with. You can then save this index to disk for later use. The search mechanism works through the use of metrics — you locate one or more entries that provide a best fit for the search criteria.

Creating an Interoperable Java Environment Using JPype

Python does provide access to a huge array of libraries, and you're really unlikely to use them all. However, you might be in a situation in which you find a Java library that is a perfect fit but can't use it from your Python application unless you're willing to jump through a whole bunch of hoops. The JPype library (`http://jpype.sourceforge.net/`) makes it possible to access most (but not all) of the Java libraries out there directly from Python. The library works by creating a bridge between the two languages at the byte-code level. Consequently, you don't have to do anything weird to get your Python application to work with Java.

Converting your Python application to Java

There are many different ways to achieve interoperability between two languages. Creating a bridge between them, as JPype does, is one way. Another alternative is to convert the code created for one language into code for the other language. This is the approach used by Jython (`https://wiki.python.org/jython/`). This utility converts your Python code into Java code so that you can make full use of Java functionality in your application while maintaining the features that you like about Python.

You'll encounter trade-offs in language interoperability no matter which solution you use. In the case of JPype, you won't have access to some Java libraries. In addition, there is a speed penalty in using this approach because the JPype bridge is constantly converting calls and data. The problem with Jython is that you lose the ability to modify your code after conversion. Any changes that you make will create an incompatibility between the original Python code and its Java counterpart. In short, no perfect solutions exist for the problem of getting the best features of two languages into one application.

Accessing Local Network Resources Using Twisted Matrix

Depending on your network setup, you may need access to files and other resources that you can't reach using the platform's native capabilities. In this case, you need a library that makes such access possible, such as Twisted Matrix (`https://twistedmatrix.com/trac/`). The basic idea behind this library is to provide you with the calls needed to establish a connection, no matter what sort of protocol is in use.

The feature that makes this library so useful is its event-driven nature. This means that your application need not get hung up while waiting for the network to respond. In addition, the use of an event-driven setup makes asynchronous communication (in which a request is sent by one routine and then handled by a completely separate routine) easy to implement.

Accessing Internet Resources Using Libraries

Although products such as Twisted Matrix can handle online communication, getting a dedicated HTTP protocol library is often a better option when working with the Internet because a dedicated library is both faster and more feature complete. When you specifically need HTTP or HTTPS support, using a library such as httplib2 (`https://github.com/jcgregorio/httplib2`) is a good idea. This library is written in pure Python and makes handling HTTP-specific needs, such as setting a Keep-Alive value, relatively easy. (A Keep-Alive is a value that determines how long a port stays open waiting for a response so that the application doesn't have to continuously re-create the connection, wasting resources and time as a result.)

You can use httplib2 for any Internet-specific methodology — it provides full support for both the `GET` and `POST` request methods. This library also includes routines for standard Internet compression methods, such as deflate and gzip. It also supports a level of automation. For example, httplib2 adds ETags back into `PUT` requests when resources are already cached.