Chapter 2. Basic Application Structure

In this chapter, you will learn about the different parts of a Flask application. You will also write and run your first Flask web application.

Initialization

All Flask applications must create an *application instance*. The web server passes all requests it receives from clients to this object for handling, using a protocol called Web Server Gateway Interface (WSGI, pronounced "wiz-ghee"). The application instance is an object of class `Flask`, usually created as follows:

```
from flask import Flask
app = Flask(__name__)
```

The only required argument to the `Flask` class constructor is the name of the main module or package of the application. For most applications, Python's `__name__` variable is the correct value for this argument.

> **TIP**
>
> The `__name__` argument that is passed to the Flask application constructor is a source of confusion among new Flask developers. Flask uses this argument to determine the location of the application, which in turn allows it to locate other files that are part of the application, such as images and templates.

Later you will learn more complex ways to initialize an application, but for simple applications this is all that is needed.

Routes and View Functions

Clients such as web browsers send *requests* to the web server, which in turn sends them to the Flask application instance. The Flask application instance needs to know what code it needs to run for each URL requested, so it keeps a mapping of URLs to Python functions. The association between a URL and the function that handles it is called a *route*.

The most convenient way to define a route in a Flask application is through the `app.route` decorator exposed by the application instance. The following example shows how a route is declared using this decorator: