Shawn Fenner

INF 473 – Problems in Informatics

Progress Report

June 29, 2018

Prepared for

David Gray

Fort Hays State University

Summer Semester 2018

Python Bottle Framework

Since the last progress report, I have changed my focus from programming my game to learning how a Python-based web framework can be used to build websites. In this instance, I decided to use *Bottle*. Bottle provides a means to quickly create websites without much of the setup required that other Python web frameworks such as *Flask* and *Django* require, hence why it is actually referred to as a *micro web framework*. However, this also means that some things, such as form validation, are not provided and must be implemented by the programmer or provided via plugins.

Below lists some fundamental concepts of Bottle that I have learned while reading its documentation and building a website that uses it.

* Bottle’s entire source code is contained within one file, which makes importing and using it fast and simple.
* A built-in single-threaded server is provided to quickly test websites.
* Bottle provides a debug mode that helps the programmer to find errors by providing traceback information and not caching templates (template pages are therefore refreshed on each page load). For these reasons, debug mode should be disabled on production servers.
* Bottle uses function decorators to implement what are called *routes*, where URLs are linked to callback functions – going to a particular URL will call the specified function. Bottle in general requires heavy use of function decorators.
* Templates provide a way to generate web pages and send variables to them to create content dynamically. Template files have the .tpl extension. Templates are cached in memory to speed up page loading except while in debug mode.
* HTTP errors, such as the common 404 page-not-found error, can be handled by the programmer by specifying a custom error page to redirect the user to that hides error information while providing a user-friendly experience.
* Static files, such as CSS, PNG, and ICO files are not served automatically as one would assume they would be by simply specifying them in an HTML document. Rather, they must be served via routing and use of the *static\_file* function, which not only determines the mime-type of the file but also restricts file paths to the root directory for security purposes.
* HTTP request methods, such as GET, POST, and PUT can be responded to and handled by specifying one of several decorators, such as @get(‘/’), @post(‘/’), and @put(‘/’), respectively. For example, in Bottle, a GET request could generate the page with the form while a POST request would handle the validation of the form.

About the Website

The website I built using Bottle is about the game I am currently building in Python – Canvas Knight (it was originally an HTML Canvas game, hence the name). The website currently has four pages – a default index page (homepage), spells page, monsters page, and an error page. Besides using Bottle, the website relies on two other technologies.

It relies on the Bootstrap framework (v4.1) for various features, such as ensuring the pages have a responsive page design as well as providing an image slideshow for the homepage. While my experience with Bootstrap is limited, I’ve found using it simple and appreciate the way it speeds up the design process while at the same time allowing flexibility and customization in the way pages are presented. Building this website has helped me improve my knowledge of Bootstrap.

The website also relies on the sqlite3 module in order to read data from database files. An interesting feature of this website occurs in the spells and monsters pages, where data is pulled from the corresponding database and sent in to a template file which generates responsive, formatted Bootstrap rows and columns that are populated with the data. This means that as more spells and monsters are added to the game, the database files simply need updating and the pages will then display this new data.

Finally, by using the Bottle template feature, page header and footer content that is identical across pages can instead be placed in their own files and simply included at the top and bottom of each page, respectively, using include statements. This feature makes reading source files easier and updating header and footer information much faster.

Important: To view the website, open the file named app.py in IDLE and press F5 to run the application and start the Bottle server. Then, open a web browser and paste in the URL: http://localhost:8080/.

Continued Learning Objectives

For the remainder of the semester, I will focus my efforts on implementing the remaining features of my game which I specified in the learning objectives document. This includes the primary feature – the two-player mode. As time permits, I will also add bosses, additional spells and monsters, and a simple C extension. By completing these tasks, I hope to learn additional features of the Python programming language and become more proficient in its use.