MSPaintr: Bottle Report

Github: https://github.com/bottlepy/bottle/blob/master/bottle.py
Documentation: https://bottlepy.org/docs/dev/bottle-docs.pdf

1. What does this technology (library/framework/service) accomplish for you?

Usage: https://github.com/hhuang36/MSPaintr/blob/master/backend/server.py

Bottle is a web-framework for Python which has a built-in HTTP Server. Bottle helps us serve our frontend (images, HTML and CSS files), specifically through the following functions:

Get

We use the get method to serve the proper file(s) which correspond to a certain path. For example, if the path is "/login", we make sure to serve the Login.html file for the login page.

Run

Run sets the host and port for our server in the main function.

Open

Open retrieves the file specified in our readFile function so we can read the lines of that file.

Redirect

For our site, "/" and "/home" both bring users to the news feed. In order to link both paths to the news feed, we use redirect to bring users who go to the "/home" path to the "/" path, so we can serve the Home.html file.

Route

Route helps us redirect users who go to the "/home" path to the "/" path.

Response

Response, as the name implies, helps us send responses to the client. It stores all metadata including headers and the status codes. It also makes it easy to set the content type and character set of the files we are trying to serve (html, css, png, etc.).

2. How does this technology accomplish what it does?

Get

Link: https://github.com/bottlepy/bottle/blob/master/bottle.py#L942

Get calls route, which is explained later. Get takes in a path and request type (e.g. GET and POST). These parameters are then passed in to route.

Run

Link: https://github.com/bottlepy/bottle/blob/master/bottle.py#L3637

The run method takes in the host and port number as parameters. If no host or port is specified, the default for them are '127.0.0.1' and '8080', respectively. Run starts a new server instance with the specified parameters, which blocks until the server is terminated (Control-c).

Open

Link: https://github.com/bottlepy/bottle/blob/master/bottle.py#L2735

Open finds the file corresponding to the filename specified as a parameter. It first looks up the file using the filename. If there isn't a file found with the filename, it returns a IOError. Otherwise, it will create a new file object from the file found and return it.

Redirect

Link: https://github.com/bottlepy/bottle/blob/master/bottle.py#L2818

Redirect calls response to set the location header and either update the status code to a user defined one, or 302 (if the server protocol is HTTP/1.1) else 303. This then will raise an error while the server is generating the response. This error is handled by setting the response equal to the error (which contains the header information) and is then sent on to the user.

Route

Link: https://github.com/bottlepy/bottle/blob/master/bottle.py#L892

Route is a decorator which binds code to a specified path which is passed in as a parameter to the function. Upon a request to a path, the corresponding function is called. The return value of that function is then sent to the client. As mentioned previously, it creates a list of plugins needed and then any plugins which are not installed.

Request

Links: https://github.com/bottlepy/bottle/blob/master/bottle.py#L4351

https://github.com/bottlepy/bottle/blob/master/bottle.py#L1931

Request is set to a new instance of the LocalRequest class. In LocalRequest, a BaseRequest is initialized. As explained in the comments, the BaseRequest class is a “wrapper for WSGI environment dictionaries that adds a lot of convenient access methods and properties.” After this initialization, the global ‘environ’ variable is set to ‘_local_property()’, which is a function that returns the request context/properties as a new property object. Whenever an attribute is added to a request, those attributes are actually added into the ‘environ’ variable. Essentially, this is a way to save and retrieve request-specific data.

Response

Links: https://github.com/bottlepy/bottle/blob/master/bottle.py#L4355

https://github.com/bottlepy/bottle/blob/master/bottle.py#L1941

Response is used to store all metadata including headers and the status codes that will be included in the response. This allows for dictionary-like storing of headers, as well as updating of the headers.

3. What license(s) or terms of service apply to this technology?

The license for Bottle can be found at:
https://github.com/bottlepy/bottle/blob/master/LICENSE

Bottle is under the MIT License and is copyrighted by Marcel Hellkamp, 2009-2018. The MIT License allows commercial use, modification, distribution, and private use of the provided software & documentation. It does not have any warranty or liability assurance.

This license states that anyone is free to use, copy, modify, merge, publish, distribute, sublicense and/or sell copies of the software, under the condition that the copyright and permission notices be included in all copies or important portions of the software and documentation. There is also no warranty of any kind for Bottle, which means its creators and copyright holders are not liable for any damages for those who wish to use the software. For our project, this means that we are free to use any library that Bottle has available online with no charge. However, if anything happens to our project as a result of using Bottle, the authors/developers who created the software are not held responsible. It is the group's decision to use Bottle as part of the project.