

## PennyWsgi/runserver.py (Page 1 of 1)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # runserver.py
5: # Author: Bob Dondero
6: #-----
7:
8: import sys
9: import wsgiref.simple_server
10: import penny
11:
12: def main():
13:
14:     if len(sys.argv) != 2:
15:         print('Usage: ' + sys.argv[0] + ' port', file=sys.stderr)
16:         sys.exit(1)
17:
18:     try:
19:         port = int(sys.argv[1])
20:     except Exception:
21:         print('Port must be an integer.', file=sys.stderr)
22:         sys.exit(1)
23:
24:     try:
25:         httpd = wsgiref.simple_server.make_server(
26:             '0.0.0.0', port, penny.app)
27:         print('Listening on port ' + str(port))
28:         httpd.serve_forever()
29:     except Exception as ex:
30:         print(ex, file=sys.stderr)
31:         sys.exit(1)
32:
33: if __name__ == '__main__':
34:     main()

```

## PennyWsgi/common.py (Page 1 of 1)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # common.py
5: # Author: Bob Dondero
6: #-----
7:
8: import time
9:
10: #-----
11:
12: def get_header():
13:     html_str = ''
14:     if time.strftime('%p') == "AM":
15:         greeting = 'morning'
16:     else:
17:         greeting = 'afternoon'
18:     html_str += '<hr>'
19:     html_str += 'Good ' + greeting + ' and welcome to '
20:     html_str += '<strong>Penny.com</strong>'
21:     html_str += '<hr>'
22:     return html_str
23:
24: #-----
25:
26: def get_footer():
27:     html_str = ''
28:     html_str += '<hr>'
29:     html_str += 'Today is ' + time.asctime(time.localtime()) + '.<br>'
30:     html_str += 'Created by '
31:     html_str += '<a href="https://www.cs.princeton.edu/~rdondero">'
32:     html_str += 'Bob Dondero</a>'
33:     html_str += '<hr>'
34:     return html_str
35:
36: #-----
37:
38: # For testing:
39:
40: def _test():
41:     print(get_header())
42:     print()
43:     print()
44:     print(get_footer())
45:
46: if __name__ == '__main__':
47:     _test()

```

## PennyWsgi/penny.py (Page 1 of 3)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # penny.py
5: # Author: Bob Dondero
6: #-----
7:
8: import http.cookies
9: import html
10: import common
11: import parseargs
12: import database
13:
14: def index(environ, start_response):
15:
16:     html_code = ''
17:     html_code += '<!DOCTYPE html>'
18:     html_code += '<html>'
19:     html_code += '<head>'
20:     html_code += '<title>Penny.com</title>'
21:     html_code += '</head>'
22:     html_code += '<body>'
23:     html_code += common.get_header()
24:     html_code += '<br>'
25:     html_code += 'Click here to <a href="/searchform">begin</a>.<br>'
26:     html_code += '<br>'
27:     html_code += common.get_footer()
28:     html_code += '</body>'
29:     html_code += '</html>'
30:
31:     content_header = ('content-type', 'text/html; charset=utf-8')
32:     headers = [content_header]
33:     start_response('200 OK', headers)
34:     return [html_code.encode('utf-8')]
35:
36: def search_form(environ, start_response):
37:
38:     prev_author = '(None)'
39:     if 'HTTP_COOKIE' in environ:
40:         cookie = http.cookies.SimpleCookie(environ['HTTP_COOKIE'])
41:         if 'prev_author' in cookie:
42:             prev_author_morsel = cookie['prev_author']
43:             if prev_author_morsel:
44:                 prev_author = prev_author_morsel.value
45:
46:     html_code = ''
47:     html_code += '<!DOCTYPE html>'
48:     html_code += '<html>'
49:     html_code += '<head>'
50:     html_code += '<title>Penny.com</title>'
51:     html_code += '</head>'
52:     html_code += '<body>'
53:     html_code += common.get_header()
54:     html_code += '<h1>Author Search</h1>'
55:     html_code += '<form action="/searchresults" method="get">'
56:     html_code += 'Please enter an author name:'
57:     html_code += '<input type="text" name="author" autofocus>'
58:     html_code += '<input type="submit" value="Go">'
59:     html_code += '</form>'
60:     html_code += '<br>'
61:     html_code += '<br>'
62:     html_code += '<strong>Previous author search:</strong>'
63:     html_code += html.escape(prev_author)
64:     html_code += '<br>'
65:     html_code += '<br>'

```

## PennyWsgi/penny.py (Page 2 of 3)

```

66:     html_code += common.get_footer()
67:     html_code += '</body>'
68:     html_code += '</html>'
69:
70:     content_header = ('content-type', 'text/html; charset=utf-8')
71:     headers = [content_header]
72:     start_response('200 OK', headers)
73:     return [html_code.encode('utf-8')]
74:
75: def search_results(environ, start_response):
76:
77:     args_str = environ.get('QUERY_STRING', '')
78:     args = parseargs.parse(args_str)
79:     author = args.get('author', '')
80:
81:     author = author.strip()
82:
83:     if author == '':
84:         prev_author = '(None)'
85:         books = []
86:     else:
87:         prev_author = author
88:         books = database.get_books(author) # Exception handling omitted
89:
90:     html_code = ''
91:     html_code += '<!DOCTYPE html>'
92:     html_code += '<html>'
93:     html_code += '<head>'
94:     html_code += '<title>Penny.com</title>'
95:     html_code += '</head>'
96:     html_code += '<body>'
97:     html_code += common.get_header()
98:     html_code += '<h1>Author Search Results</h1>'
99:     html_code += '<h2>Books by ' + html.escape(prev_author) + ':</h2>'
100:
101:     if len(books) == 0:
102:         html_code += '(None)<br>'
103:     else:
104:         pattern = '%s: <strong>%s</strong>: %s<br>'
105:         for book in books:
106:             html_code += pattern % (book['isbn'], book['author'],
107:                                     book['title'])
108:         html_code += '<br>'
109:         html_code += '<br>'
110:         html_code += 'Click here to do another '
111:         html_code += '<a href="/searchform">author search</a>.'
112:         html_code += '<br>'
113:         html_code += '<br>'
114:         html_code += '<br>'
115:         html_code += '<br>'
116:         html_code += common.get_footer()
117:         html_code += '</body>'
118:         html_code += '</html>'
119:
120:     content_header = ('content-type', 'text/html; charset=utf-8')
121:     cookie = http.cookies.SimpleCookie()
122:     cookie['prev_author'] = prev_author
123:     cookie_header = ('Set-Cookie', cookie['prev_author'].OutputString())
124:     headers = [content_header, cookie_header]
125:     start_response('200 OK', headers)
126:     return [html_code.encode('utf-8')]
127:
128: def not_found(environ, start_response):
129:
130:     # This is the response generated by Flask applications.

```

## PennyWsgi/penny.py (Page 3 of 3)

```
131:     html_code = ''
132:     html_code += '<!DOCTYPE html>'
133:     html_code += '<title>404 Not Found</title>'
134:     html_code += '<h1>Not Found</h1>'
135:     html_code += '<p>The requested URL was not found on the server. '
136:     html_code += 'If you entered the URL manually please check your '
137:     html_code += 'spelling and try again.</p>'
138:
139:     content_header = ('content-type', 'text/html; charset=utf-8')
140:     headers = [content_header]
141:     start_response('200 OK', headers)
142:     return [html_code.encode('utf-8')]
143:
144: def app(environ, start_response):
145:
146:     path = environ.get('PATH_INFO', '').strip('/')
147:
148:     if path in ('', 'index'):
149:         return index(environ, start_response)
150:     if path == 'searchform':
151:         return search_form(environ, start_response)
152:     if path == 'searchresults':
153:         return search_results(environ, start_response)
154:
155:     return not_found(environ, start_response)
```