

## PennyJson/penny.py (Page 1 of 1)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # penny.py
5: # Author: Bob Dondero
6: #-----
7:
8: import os
9: import time
10: import json
11: import flask
12: import dotenv
13: import database
14:
15: dotenv.load_dotenv()
16: _IO_DELAY = int(os.getenv('IO_DELAY', '0'))
17:
18: #-----
19:
20: app = flask.Flask(__name__)
21:
22: #-----
23:
24: @app.route('/', methods=['GET'])
25: @app.route('/index', methods=['GET'])
26: def index():
27:
28:     return flask.send_file('index.html')
29:
30: #-----
31:
32: @app.route('/searchresults', methods=['GET'])
33: def search_results():
34:
35:     author = flask.request.args.get('author')
36:     if author is None:
37:         author = ''
38:     author = author.strip()
39:
40:     # Simulate a slow database.
41:     time.sleep(_IO_DELAY)
42:
43:     if author == '':
44:         books = []
45:     else:
46:         books = database.get_books(author) # Exception handling omitted
47:
48:     json_doc = json.dumps(books)
49:     response = flask.make_response(json_doc)
50:     response.headers['Content-Type'] = 'application/json'
51:     return response

```

## blank (Page 1 of 1)

1: This page is intentionally blank.

## PennyPyqt/pennydesktop1base.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # pennydesktop1base.py
5: # Author: Bob Dondero
6: #-----
7:
8: import sys
9: import urllib.parse
10: import urllib.request
11: import json
12: import PyQt5.QtWidgets
13:
14: #-----
15:
16: def create_widgets():
17:
18:     author_label = PyQt5.QtWidgets.QLabel('Author: ')
19:     author_lineedit = PyQt5.QtWidgets.QLineEdit()
20:     submit_button = PyQt5.QtWidgets.QPushButton('Submit')
21:     books_textedit = PyQt5.QtWidgets.QTextEdit()
22:     books_textedit.setReadOnly(True)
23:
24:     layout = PyQt5.QtWidgets.QGridLayout()
25:     layout.addWidget(author_label, 0, 0)
26:     layout.addWidget(author_lineedit, 0, 1)
27:     layout.addWidget(submit_button, 0, 2)
28:     layout.addWidget(books_textedit, 1, 0, 1, 3)
29:     layout.setRowStretch(0, 0)
30:     layout.setRowStretch(1, 1)
31:     layout.setColumnStretch(0, 0)
32:     layout.setColumnStretch(1, 1)
33:     layout.setColumnStretch(2, 0)
34:
35:     frame = PyQt5.QtWidgets.QFrame()
36:     frame.setLayout(layout)
37:
38:     window = PyQt5.QtWidgets.QMainWindow()
39:     window.setWindowTitle('Penny: Author Search')
40:     window.setCentralWidget(frame)
41:     screen_size = PyQt5.QtWidgets.QDesktopWidget().screenGeometry()
42:     window.resize(screen_size.width()/2, screen_size.height()/2)
43:
44:     return (window, author_lineedit, submit_button, books_textedit)
45:
46: #-----
47:
48: def author_slot_helper(server_url, author_lineedit, books_textedit):
49:
50:     author = author_lineedit.text()
51:     encoded_author = urllib.parse.quote_plus(author)
52:     url = server_url + '/searchresults?author=' + encoded_author
53:
54:     books_textedit.clear()
55:
56:     try:
57:         with urllib.request.urlopen(url) as in_flo:
58:             response = in_flo.read()
59:             json_doc = response.decode('utf-8')
60:             books = json.loads(json_doc)
61:
62:             if len(books) == 0:
63:                 books_textedit.insertPlainText('(None)')
64:             else:
65:                 pattern = '%s: <strong>%s</strong>: %s<br>'

```

## PennyPyqt/pennydesktop1base.py (Page 2 of 2)

```

66:                 for book in books:
67:                     books_textedit.insertHtml(pattern %
68:                                                 (book['isbn'], book['author'], book['title']))
69:
70:             except Exception as ex:
71:                 books_textedit.insertPlainText(str(ex))
72:
73:     books_textedit.repaint() # Required on Mac.
74:
75: #-----
76:
77: def main():
78:
79:     if len(sys.argv) != 2:
80:         print('Usage: penny serverURL', file=sys.stderr)
81:         sys.exit(1)
82:
83:     server_url = sys.argv[1]
84:
85:     # Create and lay out the widgets.
86:
87:     app = PyQt5.QtWidgets.QApplication(sys.argv)
88:     window, author_lineedit, submit_button, books_textedit = (
89:         create_widgets())
90:
91:     # Handle signals.
92:
93:     def author_slot():
94:         author_slot_helper(server_url, author_lineedit, books_textedit)
95:         submit_button.clicked.connect(author_slot)
96:
97:     # Start up.
98:
99:     window.show()
100:    author_slot() # Populate books_textedit initially.
101:    sys.exit(app.exec_())
102:
103: if __name__ == '__main__':
104:     main()

```

## PennyPyqt/pennydesktop2seq.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # pennydesktop1seq.py
5: # Author: Bob Dondero
6: #-----
7:
8: import sys
9: import urllib.parse
10: import urllib.request
11: import json
12: import PyQt5.QtWidgets
13:
14: #-----
15:
16: def create_widgets():
17:
18:     author_label = PyQt5.QtWidgets.QLabel('Author: ')
19:     author_lineedit = PyQt5.QtWidgets.QLineEdit()
20:     books_textedit = PyQt5.QtWidgets.QTextEdit()
21:     books_textedit.setReadOnly(True)
22:
23:     layout = PyQt5.QtWidgets.QGridLayout()
24:     layout.addWidget(author_label, 0, 0)
25:     layout.addWidget(author_lineedit, 0, 1)
26:     layout.addWidget(books_textedit, 1, 0, 1, 2)
27:     layout.setRowStretch(0, 0)
28:     layout.setRowStretch(1, 1)
29:     layout.setColumnStretch(0, 0)
30:     layout.setColumnStretch(1, 1)
31:     layout.setColumnStretch(2, 0)
32:
33:     frame = PyQt5.QtWidgets.QFrame()
34:     frame.setLayout(layout)
35:
36:     window = PyQt5.QtWidgets.QMainWindow()
37:     window.setWindowTitle('Penny: Author Search')
38:     window.setCentralWidget(frame)
39:     screen_size = PyQt5.QtWidgets.QDesktopWidget().screenGeometry()
40:     window.resize(screen_size.width()/2, screen_size.height()/2)
41:
42:     return (window, author_lineedit, books_textedit)
43:
44: #-----
45:
46: def author_slot_helper(server_url, author_lineedit, books_textedit):
47:
48:     author = author_lineedit.text()
49:     encoded_author = urllib.parse.quote_plus(author)
50:     url = server_url + '/searchresults?author=' + encoded_author
51:
52:     books_textedit.clear()
53:
54:     try:
55:         with urllib.request.urlopen(url) as in_flo:
56:             response = in_flo.read()
57:             json_doc = response.decode('utf-8')
58:             books = json.loads(json_doc)
59:
60:             if len(books) == 0:
61:                 books_textedit.insertPlainText('(None)')
62:             else:
63:                 pattern = '%s: <strong>%s</strong>: %s<br>'
64:                 for book in books:
65:                     books_textedit.insertHtml(pattern %

```

## PennyPyqt/pennydesktop2seq.py (Page 2 of 2)

```

66:                                     (book['isbn'], book['author'], book['title']))
67:
68:     except Exception as ex:
69:         books_textedit.insertPlainText(str(ex))
70:
71:     books_textedit.repaint() # Required on Mac.
72:
73: #-----
74:
75: def main():
76:
77:     if len(sys.argv) != 2:
78:         print('Usage: penny serverURL', file=sys.stderr)
79:         sys.exit(1)
80:
81:     server_url = sys.argv[1]
82:
83:     # Create and lay out the widgets.
84:
85:     app = PyQt5.QtWidgets.QApplication(sys.argv)
86:     window, author_lineedit, books_textedit = create_widgets()
87:
88:     # Handle signals.
89:
90:     def author_slot():
91:         author_slot_helper(server_url, author_lineedit, books_textedit)
92:     author_lineedit.textChanged.connect(author_slot)
93:
94:     # Start up.
95:
96:     window.show()
97:     author_slot() # Populate books_textedit initially.
98:     sys.exit(app.exec_())
99:
100: if __name__ == '__main__':
101:     main()

```

## PennyPyqt/pennydesktop3bad.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # pennydesktop3bad.py
5: # Author: Bob Dondero
6: #-----
7:
8: import sys
9: import threading
10: import urllib.parse
11: import urllib.request
12: import json
13: import PyQt5.QtWidgets
14:
15: #-----
16:
17: def create_widgets():
18:
19:     author_label = PyQt5.QtWidgets.QLabel('Author: ')
20:     author_lineedit = PyQt5.QtWidgets.QLineEdit()
21:     books_textedit = PyQt5.QtWidgets.QTextEdit()
22:     books_textedit.setReadOnly(True)
23:
24:     layout = PyQt5.QtWidgets.QGridLayout()
25:     layout.addWidget(author_label, 0, 0)
26:     layout.addWidget(author_lineedit, 0, 1)
27:     layout.addWidget(books_textedit, 1, 0, 1, 2)
28:     layout.setRowStretch(0, 0)
29:     layout.setRowStretch(1, 1)
30:     layout.setColumnStretch(0, 0)
31:     layout.setColumnStretch(1, 1)
32:     layout.setColumnStretch(2, 0)
33:
34:     frame = PyQt5.QtWidgets.QFrame()
35:     frame.setLayout(layout)
36:
37:     window = PyQt5.QtWidgets.QMainWindow()
38:     window.setWindowTitle('Penny: Author Search')
39:     window.setCentralWidget(frame)
40:     screen_size = PyQt5.QtWidgets.QDesktopWidget().screenGeometry()
41:     window.resize(screen_size.width()//2, screen_size.height()//2)
42:
43:     return (window, author_lineedit, books_textedit)
44:
45: #-----
46:
47: class WorkerThread (threading.Thread):
48:
49:     def __init__(self, server_url, author, books_textedit):
50:         threading.Thread.__init__(self)
51:         self._server_url = server_url
52:         self._author = author
53:         self._books_textedit = books_textedit
54:
55:     def run(self):
56:         encoded_author = urllib.parse.quote_plus(self._author)
57:         url = self._server_url
58:         url += '/searchresults?author=' + encoded_author
59:
60:         self._books_textedit.clear()
61:
62:         try:
63:             with urllib.request.urlopen(url) as in_flo:
64:                 response = in_flo.read()
65:                 json_doc = response.decode('utf-8')

```

## PennyPyqt/pennydesktop3bad.py (Page 2 of 2)

```

66:         books = json.loads(json_doc)
67:
68:         if len(books) == 0:
69:             self._books_textedit.insertPlainText('(None)')
70:         else:
71:             pattern = '%s: <strong>%s</strong>: %s<br>'
72:             for book in books:
73:                 self._books_textedit.insertHtml(pattern %
74:                                                    (book['isbn'], book['author'],
75:                                                     book['title']))
76:
77:         except Exception as ex:
78:             self._books_textedit.insertPlainText(str(ex))
79:
80:         self._books_textedit.repaint() # Required on Mac.
81:
82: #-----
83:
84: def main():
85:
86:     if len(sys.argv) != 2:
87:         print('Usage: penny serverURL', file=sys.stderr)
88:         sys.exit(1)
89:
90:     server_url = sys.argv[1]
91:
92:     # Create and lay out the widgets.
93:
94:     app = PyQt5.QtWidgets.QApplication(sys.argv)
95:     window, author_lineedit, books_textedit = create_widgets()
96:
97:     # Handle signals.
98:
99:     def author_slot():
100:         author = author_lineedit.text()
101:         worker_thread = WorkerThread(server_url, author, books_textedit)
102:         worker_thread.start()
103:         author_lineedit.textChanged.connect(author_slot)
104:
105:     # Start up.
106:
107:     window.show()
108:     author_slot() # Populate books_textedit initially.
109:     sys.exit(app.exec_())
110:
111: if __name__ == '__main__':
112:     main()

```

## PennyPyqt/pennydesktop4threads.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # pennydesktop4threads.py
5: # Author: Bob Dondero
6: #-----
7:
8: import sys
9: import urllib.parse
10: import urllib.request
11: import json
12: import PyQt5.QtWidgets
13: import PyQt5.QtCore
14:
15: #-----
16:
17: def create_widgets():
18:
19:     author_label = PyQt5.QtWidgets.QLabel('Author: ')
20:     author_lineedit = PyQt5.QtWidgets.QLineEdit()
21:     books_textedit = PyQt5.QtWidgets.QTextEdit()
22:     books_textedit.setReadOnly(True)
23:
24:     layout = PyQt5.QtWidgets.QGridLayout()
25:     layout.addWidget(author_label, 0, 0)
26:     layout.addWidget(author_lineedit, 0, 1)
27:     layout.addWidget(books_textedit, 1, 0, 1, 2)
28:     layout.setRowStretch(0, 0)
29:     layout.setRowStretch(1, 1)
30:     layout.setColumnStretch(0, 0)
31:     layout.setColumnStretch(1, 1)
32:     layout.setColumnStretch(2, 0)
33:
34:     frame = PyQt5.QtWidgets.QFrame()
35:     frame.setLayout(layout)
36:
37:     window = PyQt5.QtWidgets.QMainWindow()
38:     window.setWindowTitle('Penny: Author Search')
39:     window.setCentralWidget(frame)
40:     screen_size = PyQt5.QtWidgets.QDesktopWidget().screenGeometry()
41:     window.resize(screen_size.width()//2, screen_size.height()//2)
42:
43:     return (window, author_lineedit, books_textedit)
44:
45: #-----
46:
47: class WorkerThread (PyQt5.QtCore.QThread):
48:
49:     _response_signal = PyQt5.QtCore.pyqtSignal(bool, object)
50:
51:     def __init__(self, server_url, author, handle_response):
52:         super().__init__()
53:         self._server_url = server_url
54:         self._author = author
55:         self._response_signal.connect(handle_response)
56:
57:     def run(self):
58:         encoded_author = urllib.parse.quote_plus(self._author)
59:         url = self._server_url
60:         url += '/searchresults?author=' + encoded_author
61:         try:
62:             with urllib.request.urlopen(url) as in_flo:
63:                 response = in_flo.read()
64:                 json_doc = response.decode('utf-8')
65:                 books = json.loads(json_doc)

```

## PennyPyqt/pennydesktop4threads.py (Page 2 of 2)

```

66:             self._response_signal.emit(True, books)
67:         except Exception as ex:
68:             self._response_signal.emit(False, str(ex))
69:
70: #-----
71:
72: def main():
73:
74:     if len(sys.argv) != 2:
75:         print('Usage: penny serverURL', file=sys.stderr)
76:         sys.exit(1)
77:
78:     server_url = sys.argv[1]
79:
80:     # Create and lay out the widgets.
81:
82:     app = PyQt5.QtWidgets.QApplication(sys.argv)
83:     window, author_lineedit, books_textedit = create_widgets()
84:
85:     # Handle signals.
86:
87:     def handle_response(successful, data):
88:         books_textedit.clear()
89:         if successful:
90:             books = data
91:             if len(books) == 0:
92:                 books_textedit.insertPlainText('(None)')
93:             else:
94:                 pattern = '%s: <strong>%s</strong>: %s<br>'
95:                 for book in books:
96:                     books_textedit.insertHtml(pattern %
97:                                                (book['isbn'], book['author'], book['title']))
98:             else:
99:                 ex = data
100:                 books_textedit.insertPlainText(ex)
101:                 books_textedit.repaint()
102:
103:     worker_thread = None
104:     def author_slot():
105:         nonlocal worker_thread
106:         author = author_lineedit.text()
107:         worker_thread = WorkerThread(server_url, author,
108:                                     handle_response)
109:         worker_thread.start()
110:
111:     author_lineedit.textChanged.connect(author_slot)
112:
113:     # Start up.
114:
115:     window.show()
116:     author_slot() # Populate books_textedit initially.
117:     sys.exit(app.exec_())
118:
119: if __name__ == '__main__':
120:     main()

```

## PennyPyqt/pennydesktop5stop.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # pennydesktop5stop.py
5: # Author: Bob Dondero
6: #-----
7:
8: import sys
9: import threading
10: import urllib.parse
11: import urllib.request
12: import json
13: import queue
14: import PyQt5.QtWidgets
15: import PyQt5.QtCore
16:
17: #-----
18:
19: def create_widgets():
20:
21:     author_label = PyQt5.QtWidgets.QLabel('Author: ')
22:     author_lineedit = PyQt5.QtWidgets.QLineEdit()
23:     books_textedit = PyQt5.QtWidgets.QTextEdit()
24:     books_textedit.setReadOnly(True)
25:
26:     layout = PyQt5.QtWidgets.QGridLayout()
27:     layout.addWidget(author_label, 0, 0)
28:     layout.addWidget(author_lineedit, 0, 1)
29:     layout.addWidget(books_textedit, 1, 0, 1, 2)
30:     layout.setRowStretch(0, 0)
31:     layout.setRowStretch(1, 1)
32:     layout.setColumnStretch(0, 0)
33:     layout.setColumnStretch(1, 1)
34:     layout.setColumnStretch(2, 0)
35:
36:     frame = PyQt5.QtWidgets.QFrame()
37:     frame.setLayout(layout)
38:
39:     window = PyQt5.QtWidgets.QMainWindow()
40:     window.setWindowTitle('Penny: Author Search')
41:     window.setCentralWidget(frame)
42:     screen_size = PyQt5.QtWidgets.QDesktopWidget().screenGeometry()
43:     window.resize(screen_size.width()/2, screen_size.height()/2)
44:
45:     return (window, author_lineedit, books_textedit)
46:
47: #-----
48:
49: class WorkerThread (PyQt5.QtCore.QThread):
50:
51:     _response_signal = PyQt5.QtCore.pyqtSignal(bool, object)
52:
53:     def __init__(self, server_url, author, handle_response):
54:         super().__init__()
55:         self._server_url = server_url
56:         self._author = author
57:         self._response_signal.connect(handle_response)
58:         self._should_stop = False
59:
60:     def stop(self):
61:         self._should_stop = True
62:
63:     def run(self):
64:         encoded_author = urllib.parse.quote_plus(self._author)
65:         url = self._server_url

```

## PennyPyqt/pennydesktop5stop.py (Page 2 of 2)

```

66:         url += '/searchresults?author=' + encoded_author
67:     try:
68:         with urllib.request.urlopen(url) as in_flo:
69:             response = in_flo.read()
70:             json_doc = response.decode('utf-8')
71:             books = json.loads(json_doc)
72:             if not self._should_stop:
73:                 self._response_signal.emit(True, books)
74:     except Exception as ex:
75:         if not self._should_stop:
76:             self._response_signal.emit(False, str(ex))
77:
78: #-----
79:
80: def main():
81:
82:     if len(sys.argv) != 2:
83:         print('Usage: penny serverURL', file=sys.stderr)
84:         sys.exit(1)
85:
86:     server_url = sys.argv[1]
87:
88:     # Create and lay out the widgets.
89:
90:     app = PyQt5.QtWidgets.QApplication(sys.argv)
91:     window, author_lineedit, books_textedit = create_widgets()
92:
93:     # Handle signals.
94:
95:     def handle_response(successful, data):
96:         books_textedit.clear()
97:         if successful:
98:             books = data
99:             if len(books) == 0:
100:                 books_textedit.insertPlainText(' (None)')
101:             else:
102:                 pattern = '%s: <strong>%s</strong>: %s<br>'
103:                 for book in books:
104:                     books_textedit.insertHtml(pattern %
105:                                                 (book['isbn'], book['author'], book['title']))
106:             else:
107:                 ex = data
108:                 books_textedit.insertPlainText(ex)
109:                 books_textedit.repaint()
110:
111:     worker_thread = None
112:     def author_slot():
113:         nonlocal worker_thread
114:         author = author_lineedit.text()
115:         if worker_thread is not None:
116:             worker_thread.stop()
117:         worker_thread = WorkerThread(server_url, author,
118:                                     handle_response)
119:         worker_thread.start()
120:
121:     author_lineedit.textChanged.connect(author_slot)
122:
123:     # Start up.
124:
125:     window.show()
126:     author_slot() # Populate books_textedit initially.
127:     sys.exit(app.exec_())
128:
129: if __name__ == '__main__':
130:     main()

```

## PennyPyqt/pennydesktop6debounce.py (Page 1 of 3)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # pennydesktop6debounce.py
5: # Author: Bob Dondero
6: #-----
7:
8: import sys
9: import urllib.parse
10: import urllib.request
11: import json
12: import PyQt5.QtWidgets
13: import PyQt5.QtCore
14:
15: #-----
16:
17: def create_widgets():
18:
19:     author_label = PyQt5.QtWidgets.QLabel('Author: ')
20:     author_lineedit = PyQt5.QtWidgets.QLineEdit()
21:     books_textedit = PyQt5.QtWidgets.QTextEdit()
22:     books_textedit.setReadOnly(True)
23:
24:     layout = PyQt5.QtWidgets.QGridLayout()
25:     layout.addWidget(author_label, 0, 0)
26:     layout.addWidget(author_lineedit, 0, 1)
27:     layout.addWidget(books_textedit, 1, 0, 1, 2)
28:     layout.setRowStretch(0, 0)
29:     layout.setRowStretch(1, 1)
30:     layout.setColumnStretch(0, 0)
31:     layout.setColumnStretch(1, 1)
32:     layout.setColumnStretch(2, 0)
33:
34:     frame = PyQt5.QtWidgets.QFrame()
35:     frame.setLayout(layout)
36:
37:     window = PyQt5.QtWidgets.QMainWindow()
38:     window.setWindowTitle('Penny: Author Search')
39:     window.setCentralWidget(frame)
40:     screen_size = PyQt5.QtWidgets.QDesktopWidget().screenGeometry()
41:     window.resize(screen_size.width()//2, screen_size.height()//2)
42:
43:     return (window, author_lineedit, books_textedit)
44:
45: #-----
46:
47: class WorkerThread (PyQt5.QtCore.QThread):
48:
49:     _response_signal = PyQt5.QtCore.pyqtSignal(bool, object)
50:
51:     def __init__(self, server_url, author, handle_response):
52:         super().__init__()
53:         self._server_url = server_url
54:         self._author = author
55:         self._response_signal.connect(handle_response)
56:         self._should_stop = False
57:
58:     def stop(self):
59:         self._should_stop = True
60:
61:     def run(self):
62:         encoded_author = urllib.parse.quote_plus(self._author)
63:         url = self._server_url
64:         url += '/searchresults?author=' + encoded_author
65:         try:

```

## PennyPyqt/pennydesktop6debounce.py (Page 2 of 3)

```

66:             with urllib.request.urlopen(url) as in_flo:
67:                 response = in_flo.read()
68:                 json_doc = response.decode('utf-8')
69:                 books = json.loads(json_doc)
70:                 if not self._should_stop:
71:                     self._response_signal.emit(True, books)
72:             except Exception as ex:
73:                 if not self._should_stop:
74:                     self._response_signal.emit(False, str(ex))
75:
76: #-----
77:
78: def main():
79:
80:     if len(sys.argv) != 2:
81:         print('Usage: penny serverURL', file=sys.stderr)
82:         sys.exit(1)
83:
84:     server_url = sys.argv[1]
85:
86:     # Create and lay out the widgets.
87:
88:     app = PyQt5.QtWidgets.QApplication(sys.argv)
89:     window, author_lineedit, books_textedit = create_widgets()
90:
91:     # Handle signals.
92:
93:     def handle_response(successful, data):
94:         books_textedit.clear()
95:         if successful:
96:             books = data
97:             if len(books) == 0:
98:                 books_textedit.insertPlainText('(None)')
99:             else:
100:                 pattern = '%s: <strong>%s</strong>: %s<br>'
101:                 for book in books:
102:                     books_textedit.insertHtml(pattern %
103:                                                 (book['isbn'], book['author'], book['title']))
104:         else:
105:             ex = data
106:             books_textedit.insertPlainText(ex)
107:             books_textedit.repaint()
108:
109:     worker_thread = None
110:     def author_slot():
111:         nonlocal worker_thread
112:         author = author_lineedit.text()
113:         if worker_thread is not None:
114:             worker_thread.stop()
115:         worker_thread = WorkerThread(server_url, author,
116:                                     handle_response)
117:         worker_thread.start()
118:
119:     debounce_timer = None
120:     def debounced_author_slot():
121:         nonlocal debounce_timer
122:         debounce_timer = PyQt5.QtCore.QTimer()
123:         debounce_timer.timeout.connect(author_slot)
124:         debounce_timer.setSingleShot(True)
125:         debounce_timer.setInterval(500) # milliseconds
126:         debounce_timer.start()
127:
128:     author_lineedit.textChanged.connect(debounced_author_slot)
129:
130:     # Start up.

```

**PennyPyqt/pennydesktop6debounce.py (Page 3 of 3)**

```
131:
132:     window.show()
133:     author_slot() # Populate books_textedit initially.
134:     sys.exit(app.exec_())
135:
136: if __name__ == '__main__':
137:     main()
```