

Threads/spawning.py (Page 1 of 1)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # spawning.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class PrinterThread (threading.Thread):
13:
14:     def __init__(self, color):
15:         threading.Thread.__init__(self)
16:         self._color = color
17:
18:     def run(self):
19:         for i in range(5):
20:             print(self._color)
21:             print(self._color + ' thread terminated')
22:
23: #-----
24:
25: def main():
26:
27:     blueThread = PrinterThread('blue')
28:     redThread = PrinterThread('red')
29:
30:     blueThread.start()
31:     redThread.start()
32:
33:     print('main thread terminated')
34:
35: #-----
36:
37: if __name__ == '__main__':
38:     main()

```

Threads/Spawning1.java (Page 1 of 1)

```

1: //-----
2: // Spawning.java
3: // Author: Bob Dondero
4: //-----
5:
6: class PrinterThread extends Thread
7: {
8:     private String color;
9:
10:    public PrinterThread(String color) { this.color = color; }
11:
12:    public void run()
13:    {
14:        for (int i = 0; i < 5; i++)
15:            System.out.println(color);
16:        System.out.println(color + " thread terminated");
17:    }
18: }
19:
20: //-----
21:
22: public class Spawning1
23: {
24:     public static void main(String[] args)
25:     {
26:         PrinterThread blueThread = new PrinterThread("blue");
27:         PrinterThread redThread = new PrinterThread("red");
28:
29:         blueThread.start();
30:         redThread.start();
31:
32:         System.out.println("main thread terminated");
33:     }
34: }
35:

```

Threads/Spawning2.java (Page 1 of 1)

```

1: //-----
2: // Spawning2.java
3: // Author: Bob Dondero
4: //-----
5:
6: class PrinterRunnable implements Runnable
7: {
8:     private String color;
9:
10:    public PrinterRunnable(String color) { this.color = color; }
11:
12:    public void run()
13:    {
14:        for (int i = 0; i < 5; i++)
15:            System.out.println(color);
16:        System.out.println(color + " thread terminated");
17:    }
18: }
19:
20: //-----
21:
22: public class Spawning2
23: {
24:     public static void main(String[] args)
25:     {
26:         Runnable bluePrinterRunnable = new PrinterRunnable("blue");
27:         Runnable redPrinterRunnable = new PrinterRunnable("red");
28:
29:         Thread blueThread = new Thread(bluePrinterRunnable);
30:         Thread redThread = new Thread(redPrinterRunnable);
31:
32:         blueThread.start();
33:         redThread.start();
34:
35:         System.out.println("main thread terminated");
36:     }
37: }
38:

```

Threads/Spawning3.java (Page 1 of 1)

```

1: //-----
2: // Spawning3.java
3: // Author: Bob Dondero
4: //-----
5:
6: public class Spawning3
7: {
8:     public static void main(String[] args)
9:     {
10:         Thread blueThread = new Thread(() -> {
11:             for (int i = 0; i < 5; i++)
12:                 System.out.println("blue");
13:             System.out.println("blue thread terminated");
14:         });
15:
16:         Thread redThread = new Thread(() -> {
17:             for (int i = 0; i < 5; i++)
18:                 System.out.println("red");
19:             System.out.println("red thread terminated");
20:         });
21:
22:         blueThread.start();
23:         redThread.start();
24:
25:         System.out.println("main thread terminated");
26:     }
27: }
28:

```

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.

```

PennySwing1/runclient (Page 1 of 1)

```
1: #!/usr/bin/env bash
2:
3: #-----
4: # runclient
5: # Author: Bob Dondero
6: #-----
7:
8: if [ $# -ne 1 ]
9: then
10:     echo "usage: runserver serverURL"
11:     exit 1
12: fi
13:
14: mvn clean
15: mvn compile
16: mvn exec:java -Dexec.mainClass="edu.princeton.penny.PennyDesktop" \
17:     -Dexec.args="$@"
```

PennySwing1/runclient.bat (Page 1 of 1)

```
1: REM -----
2: REM runclient.bat
3: REM Author: Bob Dondero
4: REM -----
5:
6: mvn clean
7: mvn compile
8: mvn exec:java -Dexec.mainClass="edu.princeton.penny.PennyDesktop" \
9:     -Dexec.args="%1"
```

PennySwing1/pom.xml (Page 1 of 1)

```

1: <?xml version="1.0" encoding="UTF-8"?>
2:
3: <project xmlns="http://maven.apache.org/POM/4.0.0"
4:   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5:   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
6:     http://maven.apache.org/xsd/maven-4.0.0.xsd">
7:
8:   <modelVersion>4.0.0</modelVersion>
9:
10:  <groupId>edu.princeton.penny</groupId>
11:  <artifactId>PennyDesktop</artifactId>
12:  <version>1.0</version>
13:  <name>PennyDesktop</name>
14:
15:  <properties>
16:    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
17:    <maven.compiler.source>21</maven.compiler.source>
18:    <maven.compiler.target>21</maven.compiler.target>
19:  </properties>
20:
21:  <dependencies>
22:    <dependency>
23:      <groupId>org.json</groupId>
24:      <artifactId>json</artifactId>
25:      <version>20250107</version>
26:    </dependency>
27:  </dependencies>
28:
29: </project>

```

PennySwing1/src/main/java/edu/princeton/penny/PennyDesktop.java (Page 1

```

1: //-----
2: // PennyDesktop.java
3: // Author: Bob Dondero
4: //-----
5:
6: package edu.princeton.penny;
7:
8: import javax.swing.SwingUtilities;
9:
10: //-----
11:
12: public class PennyDesktop
13: {
14:     public static void main(String[] args)
15:     {
16:         String serverURL = args[0];
17:         SwingUtilities.invokeLater(new PennyDesktopRunnable(serverURL));
18:     }
19: }

```

PennySwing1/src/main/java/edu/princeton/penny/PennyDesktopRunnable.java (Page 6 of 21) PennySwing21/src/main/java/edu/princeton/penny/PennyDesktopRunnable.java

```

1: //-----
2: // PennyDesktopRunnable.java
3: // Author: Bob Dondero
4: //-----
5:
6: package edu.princeton.penny;
7:
8: import javax.swing.JLabel;
9: import javax.swing.JButton;
10: import javax.swing.JPanel;
11: import javax.swing.JFrame;
12: import javax.swing.JTextField;
13: import javax.swing.DefaultListModel;
14: import javax.swing.JList;
15: import java.awt.BorderLayout;
16: import java.awt.Toolkit;
17: import java.awt.Dimension;
18: import java.awt.event.ActionListener;
19: import java.awt.event.ActionEvent;
20: import java.io.BufferedReader;
21: import java.io.IOException;
22: import java.io.InputStreamReader;
23: import java.net.URL;
24: import java.net.HttpURLConnection;
25: import java.util.ArrayList;
26: import org.json.JSONObject;
27: import org.json.JSONArray;
28:
29: //-----
30:
31: public class PennyDesktopRunnable implements Runnable
32: {
33:     private String serverURL;
34:     private JTextField authorTextField = new JTextField("");
35:     private DefaultListModel bookListModel = new DefaultListModel();
36:
37:     public PennyDesktopRunnable(String serverURL)
38:     {
39:         this.serverURL = serverURL;
40:     }
41:
42:     //-----
43:
44:     private class AuthorActionListener implements ActionListener
45:     {
46:         public void actionPerformed(ActionEvent event)
47:         {
48:             String author = authorTextField.getText();
49:
50:             try
51:             {
52:                 URL url = new URL(serverURL +
53:                     "/searchresults?author=" + author);
54:                 HttpURLConnection con =
55:                     (HttpURLConnection) url.openConnection();
56:                 con.setRequestMethod("GET");
57:                 con.setDoInput(true);
58:                 con.connect();
59:
60:                 int responseCode = con.getResponseCode();
61:                 if (responseCode != HttpURLConnection.HTTP_OK)
62:                     throw new IOException(
63:                         "HTTP error code: " + responseCode);
64:                 BufferedReader in =
65:                     new BufferedReader(

```

```

66:                     new InputStreamReader(con.getInputStream()));
67:                 String jsonDoc = in.readLine();
68:                 in.close();
69:
70:                 ArrayList<String> bookList = new ArrayList<String>();
71:                 JSONArray jsonArray = new JSONArray(jsonDoc);
72:                 for (int i = 0; i < jsonArray.length(); i++)
73:                 {
74:                     JSONObject jsonObject = jsonArray.getJSONObject(i);
75:                     bookList.add(
76:                         jsonObject.getString("isbn")
77:                         + ": " + jsonObject.getString("author")
78:                         + ": " + jsonObject.getString("title")
79:                     );
80:                 }
81:
82:                 bookListModel.removeAllElements();
83:                 for (String book: bookList)
84:                     bookListModel.addElement(book);
85:             }
86:
87:             catch (Exception e)
88:             {
89:                 bookListModel.removeAllElements();
90:                 bookListModel.addElement(e.toString());
91:             }
92:         }
93:     }
94:
95:     //-----
96:
97:     public void run()
98:     {
99:         JLabel authorLabel = new JLabel("Author: ", JLabel.RIGHT);
100:         JButton submitButton = new JButton("Submit");
101:         JList bookList = new JList(bookListModel);
102:
103:         JPanel inputPanel = new JPanel();
104:         inputPanel.setLayout(new BorderLayout());
105:         inputPanel.add("West", authorLabel);
106:         inputPanel.add("Center", authorTextField);
107:         inputPanel.add("East", submitButton);
108:
109:         JFrame frame = new JFrame();
110:         frame.setTitle("Penny");
111:         frame.setLayout(new BorderLayout());
112:         frame.add("North", inputPanel);
113:         frame.add("Center", bookList);
114:         Toolkit kit = Toolkit.getDefaultToolkit();
115:         Dimension screenSize = kit.getScreenSize();
116:         int screenWidth = screenSize.width;
117:         int screenHeight = screenSize.height;
118:         frame.setSize(screenWidth/4, screenHeight/4);
119:
120:         AuthorActionListener authorActionListener =
121:             new AuthorActionListener();
122:         submitButton.addActionListener(authorActionListener);
123:
124:         frame.setLocationByPlatform(true);
125:         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
126:         frame.setVisible(true);
127:     }
128: }

```

PennySwing2/src/main/java/edu/princeton/penny/PennyDesktopRunnable.java (PennySwing2/src/main/java/edu/princeton/penny/PennyDesktopRunnable.java)

```

1: //-----
2: // PennyDesktopRunnable.java
3: // Author: Bob Dondero
4: //-----
5:
6: package edu.princeton.penny;
7:
8: import javax.swing.JLabel;
9: import javax.swing.JPanel;
10: import javax.swing.JFrame;
11: import javax.swing.JTextField;
12: import javax.swing.DefaultListModel;
13: import javax.swing.JList;
14: import javax.swing.event.DocumentListener;
15: import javax.swing.event.DocumentEvent;
16: import java.awt.BorderLayout;
17: import java.awt.Toolkit;
18: import java.awt.Dimension;
19: import java.io.BufferedReader;
20: import java.io.IOException;
21: import java.io.InputStreamReader;
22: import java.net.URL;
23: import java.net.HttpURLConnection;
24: import java.util.ArrayList;
25: import org.json.JSONObject;
26: import org.json.JSONArray;
27:
28: //-----
29:
30: public class PennyDesktopRunnable implements Runnable
31: {
32:     private String serverURL;
33:     private JTextField authorTextField = new JTextField("");
34:     private DefaultListModel bookListModel = new DefaultListModel();
35:
36:     //-----
37:     public PennyDesktopRunnable(String serverURL)
38:     {
39:         this.serverURL = serverURL;
40:     }
41:     //-----
42:
43:     private class AuthorDocumentListener implements DocumentListener
44:     {
45:         private void update()
46:         {
47:             String author = authorTextField.getText();
48:
49:             try
50:             {
51:                 URL url = new URL(serverURL +
52:                                 "/searchresults?author=" + author);
53:                 HttpURLConnection con =
54:                     (HttpURLConnection) url.openConnection();
55:
56:                 con.setRequestMethod("GET");
57:                 con.setDoInput(true);
58:                 con.connect();
59:                 int responseCode = con.getResponseCode();
60:                 if (responseCode != HttpURLConnection.HTTP_OK)
61:                     throw new IOException(
62:                         "HTTP error code: " + responseCode);
63:
64:                 BufferedReader in =
65:                     new BufferedReader(

```

```

66:                     new InputStreamReader(con.getInputStream()));
67:                 String jsonDoc = in.readLine();
68:                 in.close();
69:
70:                 ArrayList<String> bookList = new ArrayList<String>();
71:                 JSONArray jsonArray = new JSONArray(jsonDoc);
72:                 for (int i = 0; i < jsonArray.length(); i++)
73:                 {
74:                     JSONObject jsonObject = jsonArray.getJSONObject(i);
75:                     bookList.add(
76:                         jsonObject.getString("isbn")
77:                         + ": " + jsonObject.getString("author")
78:                         + ": " + jsonObject.getString("title")
79:                     );
80:                 }
81:
82:                 bookListModel.removeAllElements();
83:                 for (String book: bookList)
84:                     bookListModel.addElement(book);
85:             }
86:             catch (Exception e)
87:             {
88:                 bookListModel.removeAllElements();
89:                 bookListModel.addElement(e.toString());
90:             }
91:         }
92:
93:         public void changedUpdate(DocumentEvent event) { update(); }
94:         public void removeUpdate(DocumentEvent event) { update(); }
95:         public void insertUpdate(DocumentEvent event) { update(); }
96:     }
97:
98:     //-----
99:
100:     public void run()
101:     {
102:         JLabel authorLabel = new JLabel("Author: ", JLabel.RIGHT);
103:         JList bookList = new JList(bookListModel);
104:
105:         JPanel inputPanel = new JPanel();
106:         inputPanel.setLayout(new BorderLayout());
107:         inputPanel.add("West", authorLabel);
108:         inputPanel.add("Center", authorTextField);
109:
110:         JFrame frame = new JFrame();
111:         frame.setTitle("Penny");
112:         frame.setLayout(new BorderLayout());
113:         frame.add("North", inputPanel);
114:         frame.add("Center", bookList);
115:         Toolkit kit = Toolkit.getDefaultToolkit();
116:         Dimension screenSize = kit.getScreenSize();
117:         int screenWidth = screenSize.width;
118:         int screenHeight = screenSize.height;
119:         frame.setSize(screenWidth/4, screenHeight/4);
120:
121:         AuthorDocumentListener authorDocumentListener =
122:             new AuthorDocumentListener();
123:         authorTextField.getDocument().
124:             addDocumentListener(authorDocumentListener);
125:
126:         frame.setLocationByPlatform(true);
127:         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
128:         frame.setVisible(true);
129:     }
130: }

```

PennySwing3/src/main/java/edu/princeton/penny/PennyDesktopRunnable.java (PennySwing23/src/main/java/edu/princeton/penny/PennyDesktopRunnable.java)

```

1: //-----
2: // PennyDesktopRunnable.java
3: // Author: Bob Dondero
4: //-----
5:
6: package edu.princeton.penny;
7:
8: import javax.swing.JLabel;
9: import javax.swing.JPanel;
10: import javax.swing.JFrame;
11: import javax.swing.JTextField;
12: import javax.swing.DefaultListModel;
13: import javax.swing.JList;
14: import javax.swing.event.DocumentListener;
15: import javax.swing.event.DocumentEvent;
16: import java.awt.BorderLayout;
17: import java.awt.Toolkit;
18: import java.awt.Dimension;
19:
20: //-----
21:
22: class PennyDesktopRunnable implements Runnable
23: {
24:     private String serverURL;
25:     private JTextField authorTextField = new JTextField("");
26:     private DefaultListModel bookListModel = new DefaultListModel();
27:
28:     //-----
29:
30:     public PennyDesktopRunnable(String serverURL)
31:     {
32:         this.serverURL = serverURL;
33:     }
34:
35:     //-----
36:
37:     private class AuthorDocumentListener implements DocumentListener
38:     {
39:         private void update()
40:         {
41:             String author = authorTextField.getText();
42:
43:             WorkerThread workerThread =
44:                 new WorkerThread(serverURL, author, bookListModel);
45:             workerThread.start();
46:         }
47:
48:         public void changedUpdate(DocumentEvent event) { update(); }
49:
50:         public void removeUpdate(DocumentEvent event) { update(); }
51:
52:         public void insertUpdate(DocumentEvent event) { update(); }
53:     }
54:
55:     //-----
56:
57:     public void run()
58:     {
59:         JLabel authorLabel = new JLabel("Author: ", JLabel.RIGHT);
60:         JList bookList = new JList(bookListModel);
61:
62:         JPanel inputPanel = new JPanel();
63:         inputPanel.setLayout(new BorderLayout());
64:         inputPanel.add("West", authorLabel);
65:         inputPanel.add("Center", authorTextField);

```

```

66:
67:     JFrame frame = new JFrame();
68:     frame.setTitle("Penny");
69:     frame.setLayout(new BorderLayout());
70:     frame.add("North", inputPanel);
71:     frame.add("Center", bookList);
72:     Toolkit kit = Toolkit.getDefaultToolkit();
73:     Dimension screenSize = kit.getScreenSize();
74:     int screenWidth = screenSize.width;
75:     int screenHeight = screenSize.height;
76:     frame.setSize(screenWidth/4, screenHeight/4);
77:
78:     AuthorDocumentListener authorDocumentListener =
79:         new AuthorDocumentListener();
80:
81:     authorTextField.getDocument().
82:         addDocumentListener(authorDocumentListener);
83:
84:     frame.setLocationByPlatform(true);
85:     frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
86:     frame.setVisible(true);
87: }
88: }

```

PennySwing3/src/main/java/edu/princeton/penny/WorkerThread.java (Page 1 of 2) PennySwing3/src/main/java/edu/princeton/penny/WorkerThread.java (Page 2

```

1: //-----
2: // WorkerThread.java
3: // Author: Bob Dondero
4: //-----
5:
6: package edu.princeton.penny;
7:
8: import javax.swing.DefaultListModel;
9: import java.io.BufferedReader;
10: import java.io.IOException;
11: import java.io.InputStreamReader;
12: import java.net.URL;
13: import java.net.HttpURLConnection;
14: import java.util.ArrayList;
15: import org.json.JSONObject;
16: import org.json.JSONArray;
17:
18: //-----
19:
20: public class WorkerThread extends Thread
21: {
22:     private String serverURL;
23:     private String author;
24:     private DefaultListModel bookListModel;
25:
26:     //-----
27:
28:     public WorkerThread(String serverURL, String author,
29:         DefaultListModel bookListModel)
30:     {
31:         this.serverURL = serverURL;
32:         this.author = author;
33:         this.bookListModel = bookListModel;
34:     }
35:
36:     //-----
37:
38:     public void run()
39:     {
40:         try
41:         {
42:             URL url = new URL(serverURL +
43:                 "/searchresults?author=" + author);
44:             HttpURLConnection con =
45:                 (HttpURLConnection) url.openConnection();
46:
47:             con.setRequestMethod("GET");
48:             con.setDoInput(true);
49:             con.connect();
50:             int responseCode = con.getResponseCode();
51:             if (responseCode != HttpURLConnection.HTTP_OK)
52:                 throw new IOException(
53:                     "HTTP error code: " + responseCode);
54:
55:             BufferedReader in =
56:                 new BufferedReader(
57:                     new InputStreamReader(con.getInputStream()));
58:             String jsonDoc = in.readLine();
59:             in.close();
60:
61:             ArrayList<String> bookList = new ArrayList<String>();
62:             JSONArray jsonArray = new JSONArray(jsonDoc);
63:             for (int i = 0; i < jsonArray.length(); i++)
64:             {
65:                 JSONObject jsonObject = jsonArray.getJSONObject(i);

```

```

66:                 bookList.add(
67:                     jsonObject.getString("isbn")
68:                     + ": " + jsonObject.getString("author")
69:                     + ": " + jsonObject.getString("title")
70:                 );
71:             }
72:
73:             bookListModel.removeAllElements();
74:             for (String book: bookList)
75:                 bookListModel.addElement(book);
76:         }
77:         catch (Exception e)
78:         {
79:             bookListModel.removeAllElements();
80:             bookListModel.addElement(e.toString());
81:         }
82:     }
83: }

```

PennySwing4/src/main/java/edu/princeton/penny/WorkerThread.java (Page 1 of 2) PennySwing4/src/main/java/edu/princeton/penny/WorkerThread.java (Page 2

```

1: //-----
2: // WorkerThread.java
3: // Author: Bob Dondero
4: //-----
5:
6: package edu.princeton.penny;
7:
8: import javax.swing.DefaultListModel;
9: import javax.swing.SwingUtilities;
10: import java.io.BufferedReader;
11: import java.io.IOException;
12: import java.io.InputStreamReader;
13: import java.net.URL;
14: import java.net.HttpURLConnection;
15: import java.util.ArrayList;
16: import org.json.JSONObject;
17: import org.json.JSONArray;
18:
19: //-----
20:
21: class WorkerThread extends Thread
22: {
23:     private String serverURL;
24:     private String author;
25:     private DefaultListModel bookListModel;
26:
27:     //-----
28:
29:     public WorkerThread(String serverURL, String author,
30:         DefaultListModel bookListModel)
31:     {
32:         this.serverURL = serverURL;
33:         this.author = author;
34:         this.bookListModel = bookListModel;
35:     }
36:
37:     //-----
38:
39:     public void run()
40:     {
41:         try
42:         {
43:             URL url = new URL(serverURL +
44:                 "/searchresults?author=" + author);
45:             HttpURLConnection con =
46:                 (HttpURLConnection) url.openConnection();
47:
48:             con.setRequestMethod("GET");
49:             con.setDoInput(true);
50:             con.connect();
51:             int responseCode = con.getResponseCode();
52:             if (responseCode != HttpURLConnection.HTTP_OK)
53:                 throw new IOException(
54:                     "HTTP error code: " + responseCode);
55:
56:             BufferedReader in =
57:                 new BufferedReader(
58:                     new InputStreamReader(con.getInputStream()));
59:             String jsonDoc = in.readLine();
60:             in.close();
61:
62:             ArrayList<String> bookList = new ArrayList<String>();
63:             JSONArray jsonArray = new JSONArray(jsonDoc);
64:             for (int i = 0; i < jsonArray.length(); i++)
65:             {

```

```

66:                 JSONObject jsonObject = jsonArray.getJSONObject(i);
67:                 bookList.add(
68:                     jsonObject.getString("isbn")
69:                     + ": " + jsonObject.getString("author")
70:                     + ": " + jsonObject.getString("title")
71:                 );
72:             }
73:
74:             SwingUtilities.invokeLater(
75:                 () -> {
76:                     bookListModel.removeAllElements();
77:                     for (String book: bookList)
78:                         bookListModel.addElement(book);
79:                 }
80:             );
81:         }
82:         catch (Exception e)
83:         {
84:             SwingUtilities.invokeLater(
85:                 () -> {
86:                     bookListModel.removeAllElements();
87:                     bookListModel.addElement(e.toString());
88:                 }
89:             );
90:         }
91:     }
92: }

```

PennySwing5/src/main/java/edu/princeton/penny/PennyDesktopRunnable.java (PennySwing25/src/main/java/edu/princeton/penny/PennyDesktopRunnable.java)

```

1: //-----
2: // PennyDesktopRunnable.java
3: // Author: Bob Dondero
4: //-----
5:
6: package edu.princeton.penny;
7:
8: import javax.swing.JLabel;
9: import javax.swing.JPanel;
10: import javax.swing.JFrame;
11: import javax.swing.JTextField;
12: import javax.swing.DefaultListModel;
13: import javax.swing.JList;
14: import javax.swing.event.DocumentListener;
15: import javax.swing.event.DocumentEvent;
16: import java.awt.BorderLayout;
17: import java.awt.Toolkit;
18: import java.awt.Dimension;
19:
20: //-----
21:
22: public class PennyDesktopRunnable implements Runnable
23: {
24:     private String serverURL;
25:     private JTextField authorTextField = new JTextField("");
26:     private DefaultListModel bookListModel = new DefaultListModel();
27:
28:     //-----
29:
30:     public PennyDesktopRunnable(String serverURL)
31:     {
32:         this.serverURL = serverURL;
33:     }
34:
35:     //-----
36:
37:     private class AuthorDocumentListener implements DocumentListener
38:     {
39:         private WorkerThread workerThread = null;
40:
41:         private void update()
42:         {
43:             String author = authorTextField.getText();
44:             if (workerThread != null)
45:                 workerThread.setShouldStop();
46:             workerThread =
47:                 new WorkerThread(serverURL, author, bookListModel);
48:             workerThread.start();
49:         }
50:
51:         public void changedUpdate(DocumentEvent event) { update(); }
52:
53:         public void removeUpdate(DocumentEvent event) { update(); }
54:
55:         public void insertUpdate(DocumentEvent event) { update(); }
56:     }
57:
58:     //-----
59:
60:     public void run()
61:     {
62:         JLabel authorLabel = new JLabel("Author: ", JLabel.RIGHT);
63:         JList bookList = new JList(bookListModel);
64:
65:         JPanel inputPanel = new JPanel();

```

```

66:         inputPanel.setLayout(new BorderLayout());
67:         inputPanel.add("West", authorLabel);
68:         inputPanel.add("Center", authorTextField);
69:
70:         JFrame frame = new JFrame();
71:         frame.setTitle("Penny");
72:         frame.setLayout(new BorderLayout());
73:         frame.add("North", inputPanel);
74:         frame.add("Center", bookList);
75:         Toolkit kit = Toolkit.getDefaultToolkit();
76:         Dimension screenSize = kit.getScreenSize();
77:         int screenWidth = screenSize.width;
78:         int screenHeight = screenSize.height;
79:         frame.setSize(screenWidth/4, screenHeight/4);
80:
81:         AuthorDocumentListener authorDocumentListener =
82:             new AuthorDocumentListener();
83:
84:         authorTextField.getDocument().
85:             addDocumentListener(authorDocumentListener);
86:
87:         frame.setLocationByPlatform(true);
88:         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
89:         frame.setVisible(true);
90:     }
91: }

```

PennySwing5/src/main/java/edu/princeton/penny/WorkerThread.java (Page 1 of 2) PennySwing5/src/main/java/edu/princeton/penny/WorkerThread.java (Page 2

```

1: //-----
2: // WorkerThread.java
3: // Author: Bob Dondero
4: //-----
5:
6: package edu.princeton.penny;
7:
8: import javax.swing.DefaultListModel;
9: import javax.swing.SwingUtilities;
10: import java.io.BufferedReader;
11: import java.io.IOException;
12: import java.io.InputStreamReader;
13: import java.net.URL;
14: import java.net.HttpURLConnection;
15: import java.util.ArrayList;
16: import org.json.JSONObject;
17: import org.json.JSONArray;
18:
19: //-----
20:
21: public class WorkerThread extends Thread
22: {
23:     private String serverURL;
24:     private String author;
25:     private DefaultListModel bookListModel;
26:     private Boolean shouldStop = false;
27:
28:     //-----
29:
30:     public WorkerThread(String serverURL, String author,
31:         DefaultListModel bookListModel)
32:     {
33:         this.serverURL = serverURL;
34:         this.author = author;
35:         this.bookListModel = bookListModel;
36:     }
37:
38:     //-----
39:
40:     public void setShouldStop()
41:     {
42:         shouldStop = true;
43:     }
44:
45:     //-----
46:
47:     public void run()
48:     {
49:         try
50:         {
51:             URL url = new URL(serverURL +
52:                 "/searchresults?author=" + author);
53:             HttpURLConnection con =
54:                 (HttpURLConnection) url.openConnection();
55:
56:             con.setRequestMethod("GET");
57:             con.setDoInput(true);
58:             con.connect();
59:             int responseCode = con.getResponseCode();
60:             if (responseCode != HttpURLConnection.HTTP_OK)
61:                 throw new IOException(
62:                     "HTTP error code: " + responseCode);
63:
64:             BufferedReader in =
65:                 new BufferedReader(

```

```

66:                 new InputStreamReader(con.getInputStream()));
67:             String jsonDoc = in.readLine();
68:             in.close();
69:
70:             ArrayList<String> bookList = new ArrayList<String>();
71:             JSONArray jsonArray = new JSONArray(jsonDoc);
72:             for (int i = 0; i < jsonArray.length(); i++)
73:             {
74:                 JSONObject jsonObject = jsonArray.getJSONObject(i);
75:                 bookList.add(
76:                     jsonObject.getString("isbn")
77:                     + ": " + jsonObject.getString("author")
78:                     + ": " + jsonObject.getString("title")
79:                 );
80:             }
81:
82:             SwingUtilities.invokeLater(
83:                 () -> {
84:                     if (! shouldStop)
85:                     {
86:                         bookListModel.removeAllElements();
87:                         for (String book: bookList)
88:                             bookListModel.addElement(book);
89:                     }
90:                 }
91:             );
92:         }
93:         catch (Exception e)
94:         {
95:             SwingUtilities.invokeLater(
96:                 () -> {
97:                     if (! shouldStop)
98:                     {
99:                         bookListModel.removeAllElements();
100:                         bookListModel.addElement(e.toString());
101:                     }
102:                 }
103:             );
104:         }
105:     }
106: }

```

PennySwing6/src/main/java/edu/princeton/penny/PennyDesktopRunnable.java (PennySwing2) PennySwing2/src/main/java/edu/princeton/penny/PennyDesktopRunnable.java

```

1: //-----
2: // PennyDesktopRunnable.java
3: // Author: Bob Dondero
4: //-----
5:
6: package edu.princeton.penny;
7:
8: import javax.swing.JLabel;
9: import javax.swing.JPanel;
10: import javax.swing.JFrame;
11: import javax.swing.JTextField;
12: import javax.swing.DefaultListModel;
13: import javax.swing.JList;
14: import javax.swing.event.DocumentListener;
15: import javax.swing.event.DocumentEvent;
16: import java.awt.BorderLayout;
17: import java.awt.Toolkit;
18: import java.awt.Dimension;
19: import java.util.Timer;
20: import java.util.TimerTask;
21:
22: //-----
23:
24: public class PennyDesktopRunnable implements Runnable
25: {
26:     private String serverURL;
27:     private JTextField authorTextField = new JTextField("");
28:     private DefaultListModel bookListModel = new DefaultListModel();
29:
30:     //-----
31:
32:     public PennyDesktopRunnable(String serverURL)
33:     {
34:         this.serverURL = serverURL;
35:     }
36:
37:     //-----
38:
39:     private class AuthorDocumentListener implements DocumentListener
40:     {
41:         private WorkerThread workerThread = null;
42:         private Timer timer = null;
43:
44:         private class MyTimerTask extends TimerTask
45:         {
46:             public void run()
47:             {
48:                 String author = authorTextField.getText();
49:                 if (workerThread != null)
50:                     workerThread.setShouldStop();
51:                 workerThread =
52:                     new WorkerThread(serverURL, author, bookListModel);
53:                 workerThread.start();
54:             }
55:         }
56:
57:         private void update()
58:         {
59:             if (timer != null)
60:                 timer.cancel();
61:             timer = new Timer();
62:             timer.schedule(new MyTimerTask(), 500);
63:         }
64:
65:         public void changedUpdate(DocumentEvent event) { update(); }

```

```

66:
67:         public void removeUpdate(DocumentEvent event) { update(); }
68:
69:         public void insertUpdate(DocumentEvent event) { update(); }
70:     }
71:
72:     //-----
73:
74:     public void run()
75:     {
76:         JLabel authorLabel = new JLabel("Author: ", JLabel.RIGHT);
77:         JList bookList = new JList(bookListModel);
78:
79:         JPanel inputPanel = new JPanel();
80:         inputPanel.setLayout(new BorderLayout());
81:         inputPanel.add("West", authorLabel);
82:         inputPanel.add("Center", authorTextField);
83:
84:         JFrame frame = new JFrame();
85:         frame.setTitle("Penny");
86:         frame.setLayout(new BorderLayout());
87:         frame.add("North", inputPanel);
88:         frame.add("Center", bookList);
89:         Toolkit kit = Toolkit.getDefaultToolkit();
90:         Dimension screenSize = kit.getScreenSize();
91:         int screenWidth = screenSize.width;
92:         int screenHeight = screenSize.height;
93:         frame.setSize(screenWidth/4, screenHeight/4);
94:
95:         AuthorDocumentListener authorDocumentListener =
96:             new AuthorDocumentListener();
97:
98:         authorTextField.getDocument().
99:             addDocumentListener(authorDocumentListener);
100:
101:         frame.setLocationByPlatform(true);
102:         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
103:         frame.setVisible(true);
104:     }
105: }

```

Threads/race.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # race.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class BankAcct:
13:
14:     def __init__(self):
15:         self._balance = 0
16:
17:     def get_balance(self):
18:         return self._balance
19:
20:     def deposit(self, amount):
21:         temp = self._balance
22:         temp += amount
23:         print(temp)
24:         self._balance = temp
25:
26:     def withdraw(self, amount):
27:         temp = self._balance
28:         temp -= amount
29:         print(temp)
30:         self._balance = temp
31:
32: #-----
33:
34: class DepositThread (threading.Thread):
35:
36:     def __init__(self, bank_acct):
37:         threading.Thread.__init__(self)
38:         self._bank_acct = bank_acct
39:
40:     def run(self):
41:         for _ in range(10):
42:             self._bank_acct.deposit(1)
43:
44: #-----
45:
46: class WithdrawThread (threading.Thread):
47:
48:     def __init__(self, bank_acct):
49:         threading.Thread.__init__(self)
50:         self._bank_acct = bank_acct
51:
52:     def run(self):
53:         for _ in range(5):
54:             self._bank_acct.withdraw(2)
55:
56: #-----
57:
58: def main():
59:
60:     bank_acct = BankAcct()
61:
62:     deposit_thread = DepositThread(bank_acct)
63:     withdraw_thread = WithdrawThread(bank_acct)
64:
65:     deposit_thread.start()

```

Threads/race.py (Page 2 of 2)

```

66:     withdraw_thread.start()
67:
68:     deposit_thread.join()
69:     withdraw_thread.join()
70:
71:     print('Final balance:', bank_acct.get_balance())
72:
73: #-----
74:
75: if __name__ == '__main__':
76:     main()

```

Threads/Race.java (Page 1 of 2)

```

1: //-----
2: // Race.java
3: // Author: Bob Dondero
4: //-----
5:
6: class Bank
7: {
8:     private int balance = 0;
9:
10:    public int getBalance()
11:    {
12:        return balance;
13:    }
14:
15:    public void deposit(int amount)
16:    {
17:        int temp = balance;
18:        temp += amount;
19:        System.out.println(temp);
20:        balance = temp;
21:    }
22:
23:    public void withdraw(int amount)
24:    {
25:        int temp = balance;
26:        temp -= amount;
27:        System.out.println(balance);
28:        balance = temp;
29:    }
30: }
31:
32: //-----
33:
34: class DepositThread extends Thread
35: {
36:     private Bank bank;
37:
38:     public DepositThread(Bank bank)
39:     {
40:         this.bank = bank;
41:     }
42:
43:     public void run()
44:     {
45:         for (int i = 0; i < 10; i++)
46:             bank.deposit(1);
47:     }
48: }
49:
50: //-----
51:
52: class WithdrawThread extends Thread
53: {
54:     private Bank bank;
55:
56:     public WithdrawThread(Bank bank)
57:     {
58:         this.bank = bank;
59:     }
60:
61:     public void run()
62:     {
63:         for (int i = 0; i < 5; i++)
64:             bank.withdraw(2);
65:     }

```

Threads/Race.java (Page 2 of 2)

```

66: }
67:
68: //-----
69:
70: public class Race
71: {
72:     public static void main(String[] args)
73:     {
74:         Bank bank = new Bank();
75:
76:         Thread depositThread = new DepositThread(bank);
77:         Thread withdrawThread = new WithdrawThread(bank);
78:
79:         depositThread.start();
80:         withdrawThread.start();
81:
82:         try
83:         {
84:             depositThread.join();
85:             withdrawThread.join();
86:         }
87:         catch (InterruptedException e)
88:         {
89:             Thread.currentThread().interrupt();
90:         }
91:
92:         System.out.println("Final balance: " + bank.getBalance());
93:     }
94: }

```

Threads/lockinuser.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # lockinuser.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class BankAcct:
13:
14:     def __init__(self):
15:         self._balance = 0
16:
17:     def get_balance(self):
18:         return self._balance
19:
20:     def deposit(self, amount):
21:         temp = self._balance
22:         temp += amount
23:         print(temp)
24:         self._balance = temp
25:
26:     def withdraw(self, amount):
27:         temp = self._balance
28:         temp -= amount
29:         print(temp)
30:         self._balance = temp
31:
32: #-----
33:
34: class DepositThread (threading.Thread):
35:
36:     def __init__(self, bank_acct, bank_acct_lock):
37:         threading.Thread.__init__(self)
38:         self._bank_acct = bank_acct
39:         self._bank_acct_lock = bank_acct_lock
40:
41:     def run(self):
42:         for _ in range(10):
43:             self._bank_acct_lock.acquire()
44:             self._bank_acct.deposit(1)
45:             self._bank_acct_lock.release()
46:
47: #-----
48:
49: class WithdrawThread (threading.Thread):
50:
51:     def __init__(self, bank_acct, bank_acct_lock):
52:         threading.Thread.__init__(self)
53:         self._bank_acct = bank_acct
54:         self._bank_acct_lock = bank_acct_lock
55:
56:     def run(self):
57:         for _ in range(5):
58:             self._bank_acct_lock.acquire()
59:             self._bank_acct.withdraw(2)
60:             self._bank_acct_lock.release()
61:
62: #-----
63:
64: def main():
65:

```

Threads/lockinuser.py (Page 2 of 2)

```

66:     bank_acct = BankAcct()
67:     bank_acct_lock = threading.RLock()
68:
69:     deposit_thread = DepositThread(bank_acct, bank_acct_lock)
70:     withdraw_thread = WithdrawThread(bank_acct, bank_acct_lock)
71:
72:     deposit_thread.start()
73:     withdraw_thread.start()
74:
75:     deposit_thread.join()
76:     withdraw_thread.join()
77:
78:     print('Final balance:', bank_acct.get_balance())
79:
80: #-----
81:
82: if __name__ == '__main__':
83:     main()

```

Threads/LockInUser.java (Page 1 of 2)

```

1: //-----
2: // LockInUser.java
3: // Author: Bob Dondero
4: //-----
5:
6: class Bank
7: {
8:     private int balance = 0;
9:
10:    public int getBalance()
11:    {
12:        return balance;
13:    }
14:
15:    public void deposit(int amount)
16:    {
17:        int temp = balance;
18:        temp += amount;
19:        System.out.println(temp);
20:        balance = temp;
21:    }
22:
23:    public void withdraw(int amount)
24:    {
25:        int temp = balance;
26:        temp -= amount;
27:        System.out.println(temp);
28:        balance = temp;
29:    }
30: }
31:
32: //-----
33:
34: class DepositThread extends Thread
35: {
36:     private Bank bank;
37:
38:     public DepositThread(Bank bank)
39:     {
40:         this.bank = bank;
41:     }
42:
43:     public void run()
44:     {
45:         for (int i = 0; i < 10; i++)
46:             synchronized (bank)
47:             {
48:                 bank.deposit(1);
49:             }
50:     }
51: }
52:
53: //-----
54:
55: class WithdrawThread extends Thread
56: {
57:     private Bank bank;
58:
59:     public WithdrawThread(Bank bank)
60:     {
61:         this.bank = bank;
62:     }
63:
64:     public void run()
65:     {

```

Threads/LockInUser.java (Page 2 of 2)

```

66:         for (int i = 0; i < 5; i++)
67:             synchronized (bank)
68:             {
69:                 bank.withdraw(2);
70:             }
71:     }
72: }
73:
74: //-----
75:
76: public class LockInUser
77: {
78:     public static void main(String[] args)
79:     {
80:         Bank bank = new Bank();
81:
82:         Thread depositThread = new DepositThread(bank);
83:         Thread withdrawThread = new WithdrawThread(bank);
84:
85:         depositThread.start();
86:         withdrawThread.start();
87:
88:         try
89:         {
90:             depositThread.join();
91:             withdrawThread.join();
92:         }
93:         catch (InterruptedException e)
94:         {
95:             Thread.currentThread().interrupt();
96:         }
97:
98:         System.out.println("Final balance: " + bank.getBalance());
99:     }
100: }

```

Threads/lockinresource.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # lockinresource.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class BankAcct:
13:
14:     def __init__(self):
15:         self._balance = 0
16:         self._lock = threading.RLock()
17:
18:     def get_balance(self):
19:         self._lock.acquire()
20:         try:
21:             return self._balance
22:         finally:
23:             self._lock.release()
24:
25:     def deposit(self, amount):
26:         self._lock.acquire()
27:         temp = self._balance
28:         temp += amount
29:         print(temp)
30:         self._balance = temp
31:         self._lock.release()
32:
33:     def withdraw(self, amount):
34:         self._lock.acquire()
35:         temp = self._balance
36:         temp -= amount
37:         print(temp)
38:         self._balance = temp
39:         self._lock.release()
40:
41: #-----
42:
43: class DepositThread (threading.Thread):
44:
45:     def __init__(self, bank_acct):
46:         threading.Thread.__init__(self)
47:         self._bank_acct = bank_acct
48:
49:     def run(self):
50:         for _ in range(10):
51:             self._bank_acct.deposit(1)
52:
53: #-----
54:
55: class WithdrawThread (threading.Thread):
56:
57:     def __init__(self, bank_acct):
58:         threading.Thread.__init__(self)
59:         self._bank_acct = bank_acct
60:
61:     def run(self):
62:         for _ in range(5):
63:             self._bank_acct.withdraw(2)
64:
65: #-----

```

Threads/lockinresource.py (Page 2 of 2)

```

66:
67: def main():
68:
69:     bank_acct = BankAcct()
70:
71:     deposit_thread = DepositThread(bank_acct)
72:     withdraw_thread = WithdrawThread(bank_acct)
73:
74:     deposit_thread.start()
75:     withdraw_thread.start()
76:
77:     deposit_thread.join()
78:     withdraw_thread.join()
79:
80:     print('Final balance:', bank_acct.get_balance())
81:
82: #-----
83:
84: if __name__ == '__main__':
85:     main()

```

Threads/LockInResource.java (Page 1 of 2)

```

1: //-----
2: // LockInResource.java
3: // Author: Bob Dondero
4: //-----
5:
6: class Bank
7: {
8:     private int balance = 0;
9:
10:    public synchronized int getBalance()
11:    {
12:        return balance;
13:    }
14:
15:    public synchronized void deposit(int amount)
16:    {
17:        int temp = balance;
18:        temp += amount;
19:        System.out.println(temp);
20:        balance = temp;
21:    }
22:
23:    public synchronized void withdraw(int amount)
24:    {
25:        int temp = balance;
26:        temp -= amount;
27:        System.out.println(temp);
28:        balance = temp;
29:    }
30: }
31:
32: //-----
33:
34: class DepositThread extends Thread
35: {
36:     private Bank bank;
37:
38:     public DepositThread(Bank bank)
39:     {
40:         this.bank = bank;
41:     }
42:
43:     public void run()
44:     {
45:         for (int i = 0; i < 10; i++)
46:             bank.deposit(1);
47:     }
48: }
49:
50: //-----
51:
52: class WithdrawThread extends Thread
53: {
54:     private Bank bank;
55:
56:     public WithdrawThread(Bank bank)
57:     {
58:         this.bank = bank;
59:     }
60:
61:     public void run()
62:     {
63:         for (int i = 0; i < 5; i++)
64:             bank.withdraw(2);
65:     }

```

Threads/LockInResource.java (Page 2 of 2)

```

66: }
67:
68: //-----
69:
70: public class LockInResource
71: {
72:     public static void main(String[] args)
73:     {
74:         Bank bank = new Bank();
75:
76:         Thread depositThread = new DepositThread(bank);
77:         Thread withdrawThread = new WithdrawThread(bank);
78:
79:         depositThread.start();
80:         withdrawThread.start();
81:
82:         try
83:         {
84:             depositThread.join();
85:             withdrawThread.join();
86:         }
87:         catch (InterruptedException e)
88:         {
89:             Thread.currentThread().interrupt();
90:         }
91:
92:         System.out.println("Final balance: " + bank.getBalance());
93:     }
94: }

```

Threads/conditions.py (Page 1 of 2)

```

1: #!/usr/bin/env python
2:
3: #-----
4: # conditions.py
5: # Author: Bob Dondero
6: #-----
7:
8: import threading
9:
10: #-----
11:
12: class BankAcct:
13:
14:     def __init__(self):
15:         self._balance = 0
16:         self._lock = threading.RLock()
17:         self._condition = threading.Condition(self._lock)
18:
19:     def get_balance(self):
20:         self._lock.acquire()
21:         try:
22:             return self._balance
23:         finally:
24:             self._lock.release()
25:
26:     def deposit(self, amount):
27:         self._condition.acquire()
28:         self._balance += amount
29:         print(self._balance)
30:         self._condition.notify_all()
31:         self._condition.release()
32:
33:     def withdraw(self, amount):
34:         self._condition.acquire()
35:         while self._balance < amount:
36:             self._condition.wait()
37:         self._balance -= amount
38:         print(self._balance)
39:         self._condition.release()
40:
41: #-----
42:
43: class DepositThread (threading.Thread):
44:
45:     def __init__(self, bank_acct):
46:         threading.Thread.__init__(self)
47:         self._bank_acct = bank_acct
48:
49:     def run(self):
50:         for _ in range(10):
51:             self._bank_acct.deposit(1)
52:
53: #-----
54:
55: class WithdrawThread (threading.Thread):
56:
57:     def __init__(self, bank_acct):
58:         threading.Thread.__init__(self)
59:         self._bank_acct = bank_acct
60:
61:     def run(self):
62:         for _ in range(5):
63:             self._bank_acct.withdraw(2)
64:
65: #-----

```

Threads/conditions.py (Page 2 of 2)

```

66:
67: def main():
68:     bank_acct = BankAcct()
69:
70:     deposit_thread = DepositThread(bank_acct)
71:     withdraw_thread = WithdrawThread(bank_acct)
72:
73:     deposit_thread.start()
74:     withdraw_thread.start()
75:
76:     deposit_thread.join()
77:     withdraw_thread.join()
78:
79:     print('Final balance:', bank_acct.get_balance())
80:
81: #-----
82:
83: if __name__ == '__main__':
84:     main()

```

Threads/Conditions.java (Page 1 of 2)

```

1: //-----
2: // Conditions.java
3: // Author: Bob Dondero
4: //-----
5:
6: class Bank
7: {
8:     private int balance = 0;
9:
10:    public synchronized int getBalance()
11:    {
12:        return balance;
13:    }
14:
15:    public synchronized void deposit(int amount)
16:    {
17:        balance += amount;
18:        System.out.println(balance);
19:        notifyAll();
20:    }
21:
22:    public synchronized void withdraw(int amount)
23:    {
24:        try
25:        {
26:            while (balance < amount)
27:                wait();
28:            balance -= amount;
29:            System.out.println(balance);
30:        }
31:        catch (InterruptedException e)
32:        {
33:            Thread.currentThread().interrupt();
34:        }
35:    }
36: }
37:
38: //-----
39:
40: class DepositThread extends Thread
41: {
42:     private Bank bank;
43:
44:     public DepositThread(Bank bank)
45:     {
46:         this.bank = bank;
47:     }
48:
49:     public void run()
50:     {
51:         for (int i = 0; i < 10; i++)
52:             bank.deposit(1);
53:     }
54: }
55:
56: //-----
57:
58: class WithdrawThread extends Thread
59: {
60:     private Bank bank;
61:
62:     public WithdrawThread(Bank bank)
63:     {
64:         this.bank = bank;
65:     }

```

Threads/Conditions.java (Page 2 of 2)

```

66:
67:     public void run()
68:     {
69:         for (int i = 0; i < 5; i++)
70:             bank.withdraw(2);
71:     }
72: }
73:
74: //-----
75:
76: public class Conditions
77: {
78:     public static void main(String[] args)
79:     {
80:         Bank bank = new Bank();
81:
82:         Thread depositThread = new DepositThread(bank);
83:         Thread withdrawThread = new WithdrawThread(bank);
84:
85:         depositThread.start();
86:         withdrawThread.start();
87:
88:         try
89:         {
90:             depositThread.join();
91:             withdrawThread.join();
92:         }
93:         catch (InterruptedException e)
94:         {
95:             Thread.currentThread().interrupt();
96:         }
97:
98:         System.out.println("Final balance: " + bank.getBalance());
99:     }
100: }

```