```
 1: /*
 2:
 3: Jared Dyreson
 4: CWID: 889546529
 5: AboutPage.java ->
 6:
 7: */
 8:
 9: import javax.swing.*;
10: import java.text.MessageFormat;
11: import java.awt.*;
12: import javax.swing.JFrame;
13: import java.awt.event.*;
14: import java.lang.Math;
15:
16: import java.io.*;
17: import java.util.*;
18: import java.util.List;
19: import java.util.ArrayList;
20: import java.util.Arrays;
21:
22: public class AboutPage extends JFrame implements ActionListener{
23:         // Auto generated with caffine and ureadahead.service
24:
25:         private final int FRAME_HEIGHT = 424, FRAME_WIDTH = 373;
26:         private String about_message = "Zip Viewer\n1.0.0\nA zip manager written in pure Java.\nThis is a direc
27:         private JLabel center_label = new JLabel(about_message);
28:
29:         private JButton close = new JButton("Close");
30:         private JButton credits = new JButton("Credits");
31:
32:         private JPanel sub_panel = new JPanel();
33:
34:         public AboutPage(){
35:                 super("About Zip Viewer");
36:
37:                 this.setSize(FRAME_HEIGHT, FRAME_WIDTH);
38:                 this.setLocationRelativeTo(null);
39:                 this.setLayout(new FlowLayout());
40:                 this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
41:                 //this.add(sub_panel, BorderLayout.SOUTH);
42:
43:                 close.addActionListener(this);
44:                 credits.addActionListener(this);
45:
46:                 sub_panel.setLayout(new FlowLayout());
47:                 this.add(center_label, BorderLayout.CENTER);
48:                 this.add(credits, BorderLayout.WEST);
```

```
49:                 this.add(close, BorderLayout.EAST);
50:
51:                 //this.add(sub_panel, BorderLayout.SOUTH);
52:
53:
54:         }
55:         @Override
56:         public void actionPerformed(ActionEvent event){
57:                 Object source = event.getSource();
58:
59:                 if(source == close){
60:                         this.dispose();
61:                 }
62:                 else{
63:                         System.out.println("Credits...my mom");
64:                 }
65:
66:         }
67: }
```

```
 1: /*
 2:
 3: Jared Dyreson
 4: CWID: 889546529
 5: CreateNewArchive.java ->
 6:
 7: */
 8:
 9: import javax.swing.*;
10: import java.text.MessageFormat;
11: import java.awt.*;
12: import javax.swing.JFrame;
13: import java.awt.event.*;
14: import java.lang.Math;
15:
16: import java.io.*;
17: import java.util.*;
18: import java.util.List;
19: import java.util.ArrayList;
20: import java.util.Arrays;
21:
22:
23: public class CreateNewArchive extends JFrame implements ActionListener{
24:         // Auto generated with caffine and pppd-dns.service
25:
26:         private final int FRAME_HEIGHT = 522, FRAME_WIDTH = 193;
27:         JButton create = new JButton("Create");
28:         JButton close = new JButton("Close");
29:
30:         JLabel location_label = new JLabel("Location");
31:         JLabel archive_name_label = new JLabel("Archive Name:");
32:
33:         JTextField archive_name_field = new JTextField(20);
34:
35:         JPanel bottom_elements = new JPanel();
36:         JPanel center_elements = new JPanel();
37:
38:         ZipBackend zipper = new ZipBackend();
39:
40:         public CreateNewArchive(){
41:                 super("Create New Archive");
42:
43:                 this.setSize(FRAME_HEIGHT, FRAME_WIDTH);
44:                 this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
45:                 this.setLocationRelativeTo(null);
46:
47:                 create.addActionListener(this);
48:                 close.addActionListener(this);
```

```
49:                     bottom_elements.setLayout(new FlowLayout());
50:                     bottom_elements.add(create);
51:                     bottom_elements.add(close);
52:
53:                     center_elements.setLayout(new FlowLayout());
54:                     center_elements.add(location_label);
55:                     center_elements.add(archive_name_field);
56:
57:                     this.add(bottom_elements, BorderLayout.SOUTH);
58:                     this.add(center_elements, BorderLayout.CENTER);
59:             }
60:             @Override
61:             public void actionPerformed(ActionEvent event){
62:                     Object action_performed = event.getSource();
63:
64:                     if(action_performed == close){
65:                             this.dispose();
66:                     }
67:                     else if(action_performed == create){
68:                             String path_to_empty = archive_name_field.getText();
69:                             try{
70:
71:                                     zipper.create_empty_archive(path_to_empty+".zip");
72:                             }
73:                             catch(Exception error){
74:                                     System.out.println(error);
75:                             }
76:                     }
77:             }
78: }
```

```java
/*

Jared Dyreson
CWID: 889546529
Driver.java ->

*/

import java.io.File;
import javax.swing.JFileChooser;
import javax.swing.filechooser.FileSystemView;

public class Driver {
        public static void main(String[] args){
                // Auto generated with caffine and NetworkManager-dispatcher.service
                ZipWindow zipper = new ZipWindow();
                zipper.setVisible(true);
                //CreateNewArchive n = new CreateNewArchive();
                //n.setVisible(true);
        }
}
```

```
 1: /*
 2:
 3: Jared Dyreson
 4: CWID: 889546529
 5: FileHandler.java ->
 6:
 7: */
 8: import java.io.IOException;
 9: import java.nio.file.Files;
10: import java.nio.file.Path;
11: import java.nio.file.Paths;
12:
13: import java.io.File;
14: import java.text.SimpleDateFormat;
15:
16: public class FileHandler {
17:         // Auto generated with caffine and networking.service
18:
19:         private String path = "";
20:         private File file;
21:
22:         public FileHandler(String path_to_file){
23:                 this.path = path_to_file;
24:                 this.file = new File(this.path);
25:         }
26:
27:         public String get_path(){
28:                 return path;
29:         }
30:         public String get_absolute_path(){
31:                 if(this.file.exists()){
32:                         return this.file.getAbsolutePath();
33:                 }
34:                 return "";
35:         }
36:         public String get_last_modified(){
37:                 SimpleDateFormat formatter = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss");
38:                 return formatter.format(this.file.lastModified());
39:         }
40:         public String get_mime_type() throws IOException{
41:                 if(this.file.isDirectory()){ return "dir"; }
42:                 Path source = Paths.get(this.path);
43:                 return Files.probeContentType(source);
44:         }
45:
46:         public static void main(String args[]){
47:                 FileHandler f = new FileHandler("temp/");
48:                 try{
```

```
49:                         System.out.println(f.get_mime_type());
50:                         System.out.println(f.get_last_modified());
51:                         System.out.println(f.get_absolute_path());
52:                 }
53:                 catch(Exception error){
54:                         System.out.println(error);
55:                 }
56:
57:         }
58: }
```

```
 1: /*
 2:
 3: Jared Dyreson
 4: CWID: 889546529
 5: ZipBackend.java ->
 6:
 7: */
 8:
 9: import java.io.*;
10: import java.util.*;
11: import java.util.zip.*;
12:
13: public class ZipBackend {
14:         // Auto generated with caffine and accounts-daemon.service
15:
16:         private ZipOutputStream zip_stream_for_class;
17:
18:         public void zip_contents(List<String> file_contents, String path_to_file) throws IOException{
19:                 FileOutputStream file_stream = new FileOutputStream(path_to_file);
20:                 ZipOutputStream zip_out = new ZipOutputStream(file_stream);
21:                 File destination_zip = new File(path_to_file);
22:                 if(destination_zip.exists()){
23:                         System.out.println("File already exists");
24:                         return;
25:                 }
26:
27:                 for(String path : file_contents){
28:                         File zip_this_file = new File(path);
29:                         FileInputStream input_stream = new FileInputStream(zip_this_file);
30:                         ZipEntry zip_entry = new ZipEntry(zip_this_file.getName());
31:                         zip_out.putNextEntry(zip_entry);
32:
33:                         byte[] bytes = new byte[1024];
34:                         int len;
35:                         while((len = input_stream.read(bytes)) >= 0){
36:                                 zip_out.write(bytes, 0, len);
37:                         }
38:                         input_stream.close();
39:                 }
40:                 zip_out.close();
41:                 file_stream.close();
42:         }
43:
44:         public void create_empty_archive(String path_to_file) throws IOException{
45:                 File zip_file = new File(path_to_file);
46:                 FileOutputStream stream = new FileOutputStream(zip_file);
47:                 BufferedOutputStream buffered = new BufferedOutputStream(stream);
48:                 ZipOutputStream zip_stream = new ZipOutputStream(stream);
```

```
49:
50:                 //ZipEntry fake_entry = new ZipEntry("fake_meta");
51:                 //zip_stream.putNextEntry(fake_entry);
52:                 //zip_stream.closeEntry();
53:                 zip_stream.close();
54:         }
55: }
```

```
 1: /*
 2:
 3: Jared Dyreson
 4: CWID: 889546529
 5: ZipWindow.java ->
 6:
 7: */
 8: import javax.swing.JFileChooser;
 9: import javax.swing.filechooser.FileSystemView;
10:
11: import javax.swing.*;
12: import java.text.MessageFormat;
13: import java.awt.*;
14: import javax.swing.JFrame;
15: import java.awt.event.*;
16: import java.lang.Math;
17:
18: import java.io.*;
19: import java.util.*;
20: import java.util.List;
21: import java.util.ArrayList;
22: import java.util.Arrays;
23:
24: public class ZipWindow extends JFrame implements ActionListener{
25:         // Auto generated with caffine and cups.service
26:         final int FRAME_WIDTH = 959, FRAME_HEIGHT = 1052;
27:
28:         JPanel file_manipulation_buttons = new JPanel();
29:         JPanel menu_panel = new JPanel();
30:
31:         JButton extract_button = new JButton("Extract");
32:         JButton add_files = new JButton("+");
33:
34:         JButton next = new JButton("<--");
35:         JButton previous = new JButton("-->");
36:         JButton home_button = new JButton("Root folder");
37:         JButton about = new JButton("about");
38:
39:         JLabel location_label = new JLabel("Location:");
40:
41:         JTextField path_in_archive = new JTextField();
42:
43:         JToolBar toolbar = new JToolBar();
44:
45:         String[] menu_options = {"File", "Edit", "View", "Help"};
46:         String[] file_menu_options = {"New Archive", "Open", "Extract Files", "Close"};
47:         List<String> file_manifest = new ArrayList<>();
48:
```

```
49:         JMenuBar menu_bar = new JMenuBar();
50:         ZipBackend zipper = new ZipBackend();
51:
52:         public ZipWindow(){
53:                 super("Zip Viewer");
54:
55:                 setSize(FRAME_WIDTH, FRAME_HEIGHT);
56:                 setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
57:                 setLocationRelativeTo(null);
58:                 add_files.addActionListener(this);
59:                 extract_button.addActionListener(this);
60:                 about.addActionListener(this);
61:                 //setLayout(new BorderLayout(2, 1));
62:
63:                 toolbar.setRollover(true);
64:                 for(int i = 0; i < menu_options.length; ++i){
65:                         menu_bar.add(new JMenu(menu_options[i]));
66:                 }
67:
68:                 toolbar.add(menu_bar);
69:
70:                 file_manipulation_buttons.setLayout(new FlowLayout());
71:                 file_manipulation_buttons.add(extract_button);
72:                 file_manipulation_buttons.add(add_files);
73:                 file_manipulation_buttons.add(about);
74:
75:                 menu_panel.setLayout(new BorderLayout(2, 1));
76:                 menu_panel.add(toolbar, BorderLayout.WEST);
77:                 menu_panel.add(file_manipulation_buttons, BorderLayout.EAST);
78:                 //menu_panel.add(extract_button, BorderLayout.EAST);
79:                 this.add(menu_panel, BorderLayout.NORTH);
80:
81:         }
82:
83:         public String get_file(){
84:                 String file_path = "";
85:                 JFileChooser jfc = new JFileChooser(FileSystemView.getFileSystemView().getHomeDirectory());
86:                 int returnValue = jfc.showOpenDialog(null);
87:                 if (returnValue == JFileChooser.APPROVE_OPTION) {
88:                         File selectedFile = jfc.getSelectedFile();
89:                         file_path = selectedFile.getAbsolutePath();
90:                 }
91:                 return file_path;
92:
93:         }
94:         @Override
95:         public void actionPerformed(ActionEvent event){
96:                 Object source = event.getSource();
```

```
 97:                 if(source == add_files){
 98:                         String path_to_file = get_file();
 99:                         file_manifest.add(path_to_file);
100:                 }
101:                 else if(source == extract_button){
102:                         try{
103:                                 zipper.zip_contents(file_manifest, "example.zip");
104:                         }
105:                         catch(Exception error){
106:                                 System.out.println(error);
107:                         }
108:                 }
109:                 else if(source == about){
110:                         AboutPage about_page = new AboutPage();
111:                         about_page.setVisible(true);
112:                 }
113:         }
114:
115: }
```