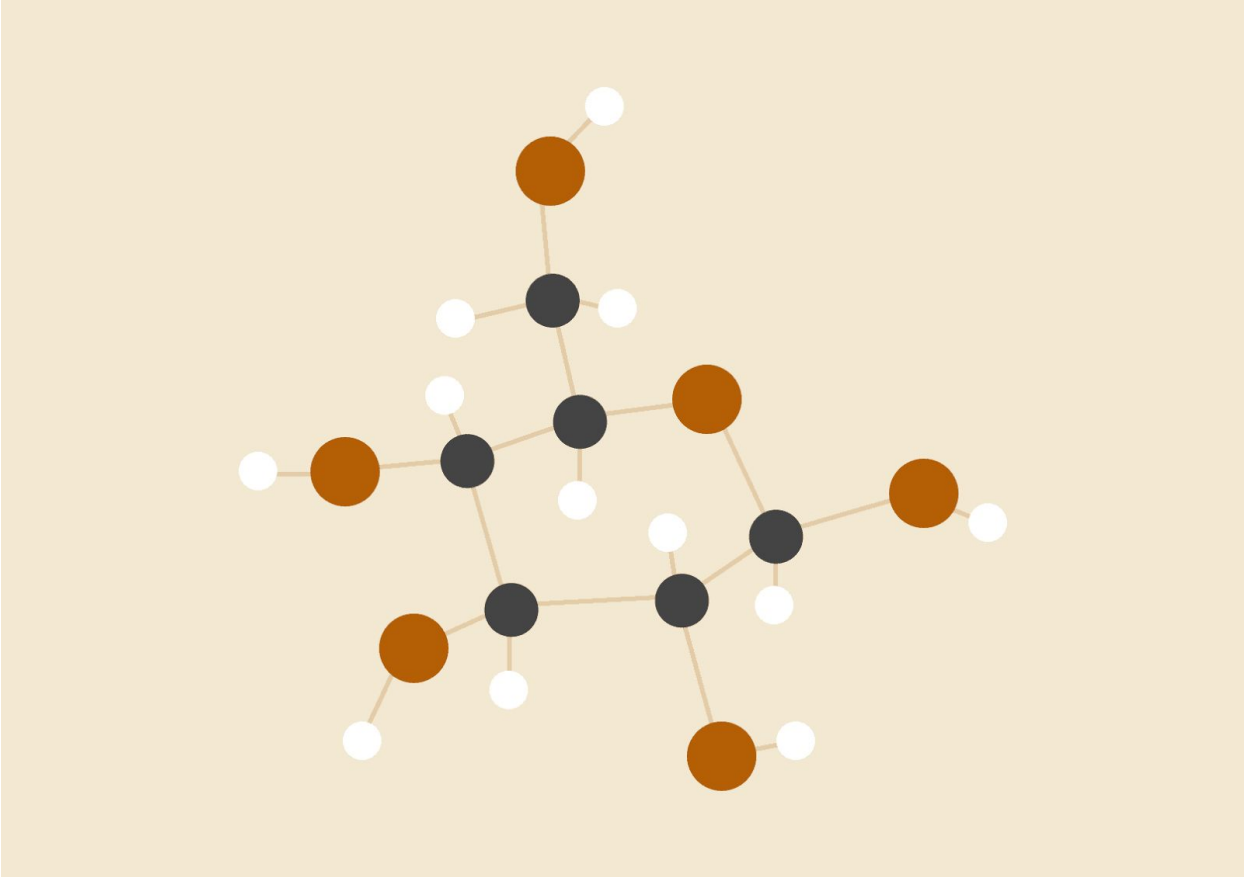
LBYCP12

Data Structures and Algorithm Analysis Laboratory



Laboratory Activity 8

Integrating Activity

By

Chan, Patrick Matthew J, LBYCP12-EQ1

INTRODUCTION

In this activity, students are expected to apply all their knowledge and learnings, and create a single grand all-in-one program that contains all the past activities. Here, the students are also given a sample program the communicates with an online database to use as the base application. In addition, the students are introduced to the concept of packages in Java, in order to separate the files from different activities in an organized manner.

OBJECTIVES

- To apply and review all past learnings
- To learn how to use packages in Java
- To unify data structure files into a single package, and similarly for interfaces and exceptions

MATERIALS

1. Java SE Development Kit (JDK) 8
2. NetBeans integrated development environment (IDE)
3. `acm.jar` by the ACM Java Task Force

PROCEDURE

1. Gather all the files needed (i.e. all past activities and the database base app)
2. For each of the past activities, create a package inside a new project.
3. Place a copy of the old files into their corresponding packages
4. Create a package for each of the ADTs used
5. Create a package for exceptions
6. Create a package for interfaces
7. Refactor each of the packages to update the codes.
8. Follow the naming convention:

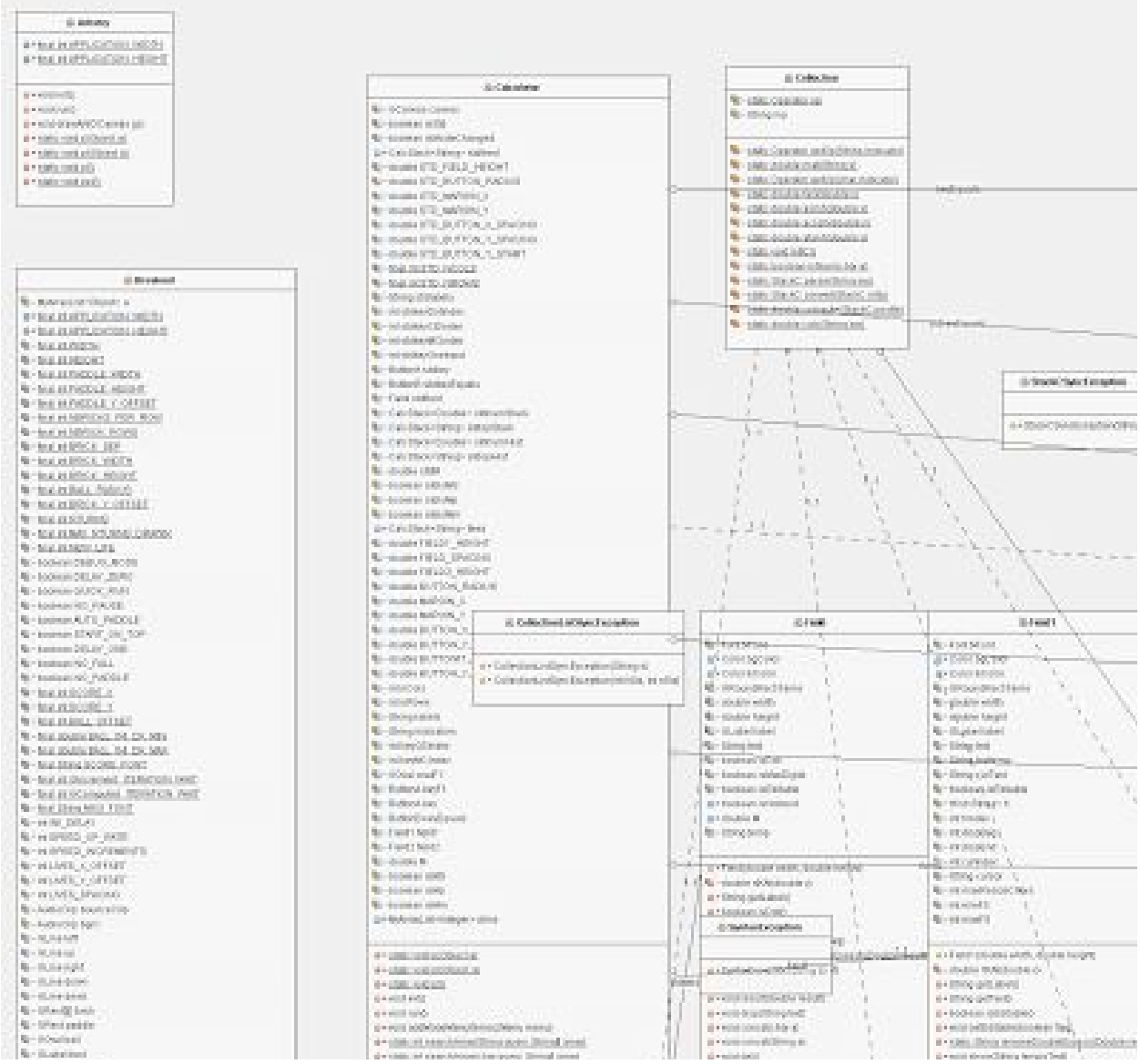
   ph.edu.dlsu.datasal.YOUR_SURNAME.name

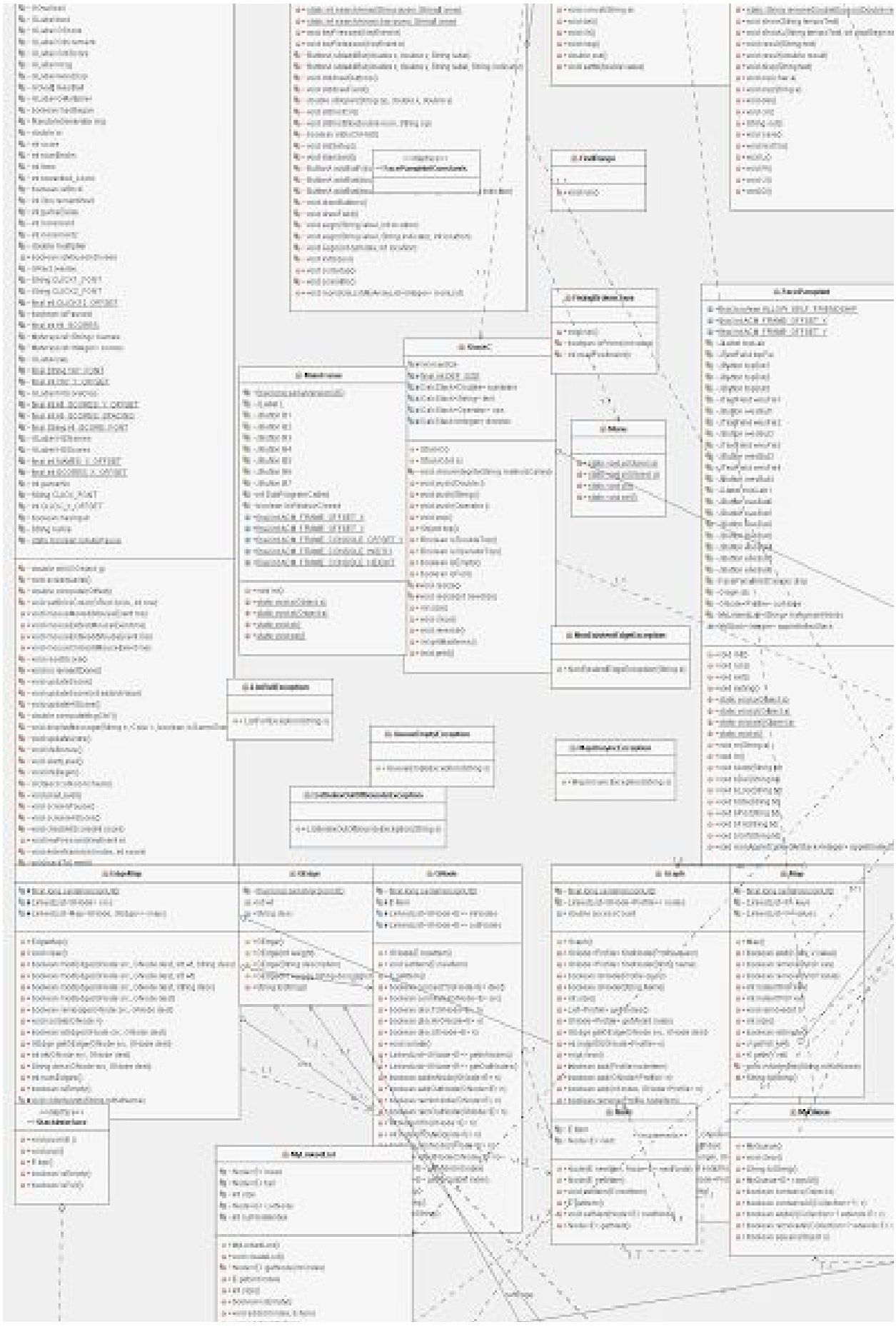
9. Remove the main method in each of the files, except the base app

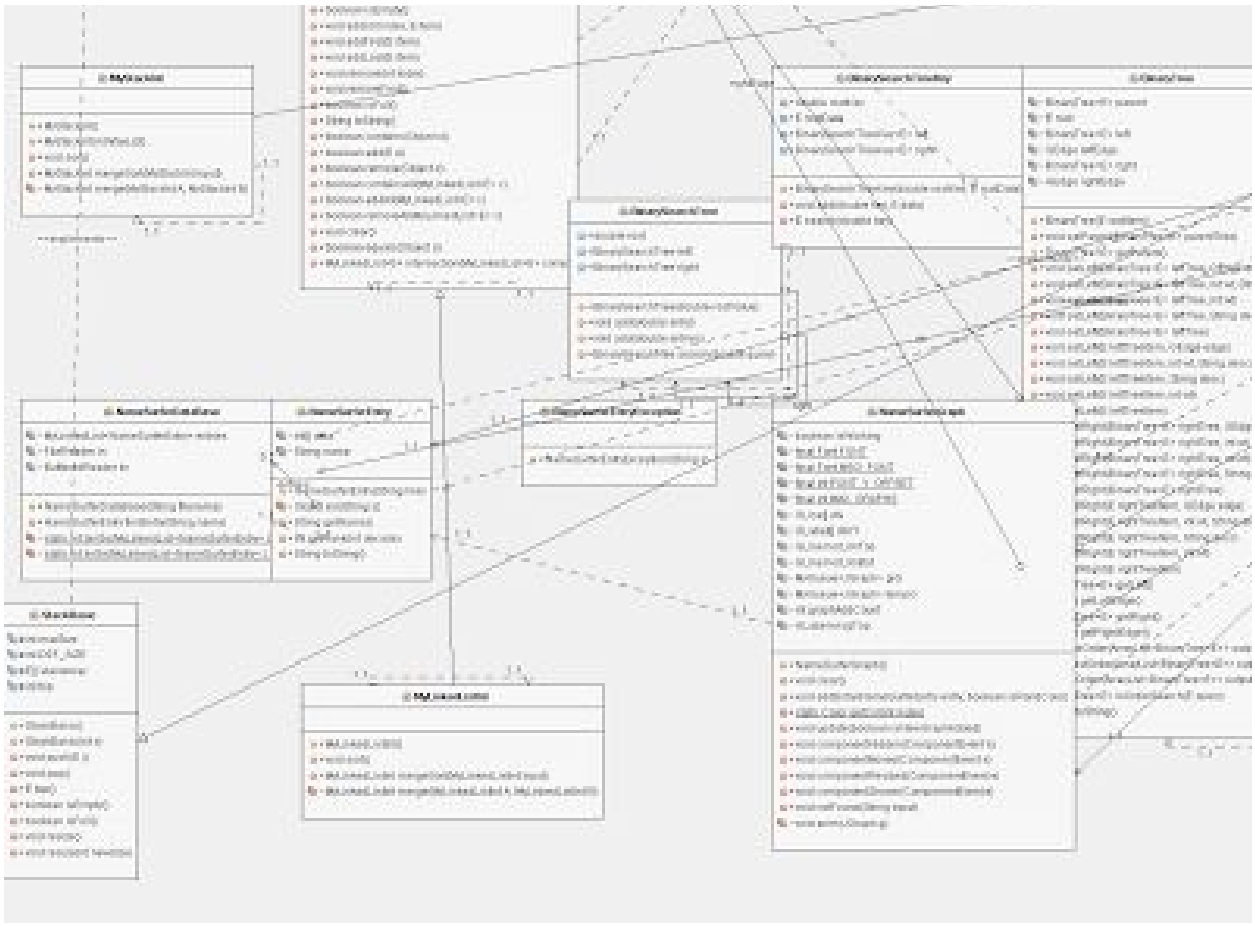
10. Add new buttons on the base app
11. Add action listeners to the base app
12. Make the buttons on the base app to call the different old applications' start(String) method.
13. Open the base app and test the code.

RESULTS AND DISCUSSION

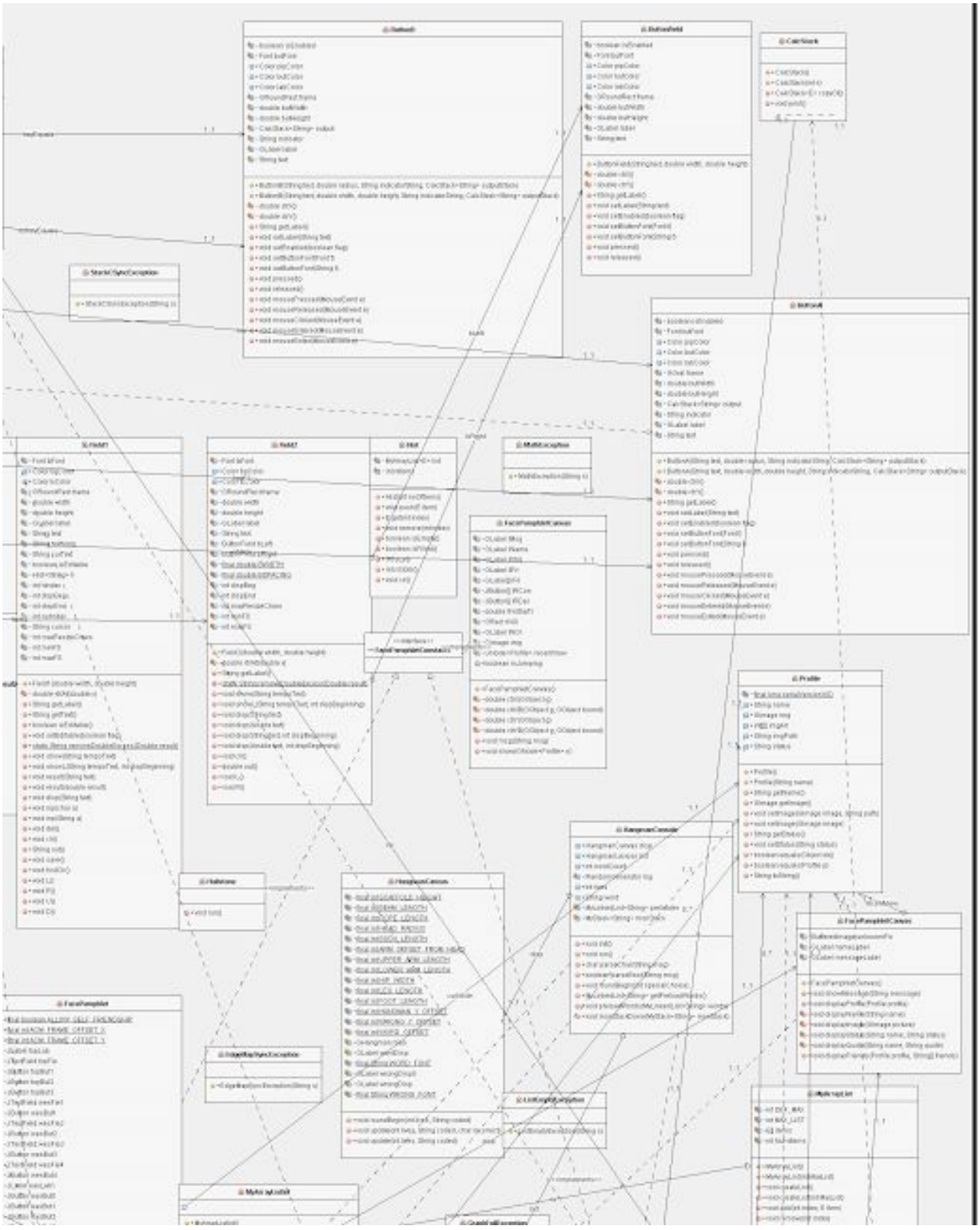
UML:

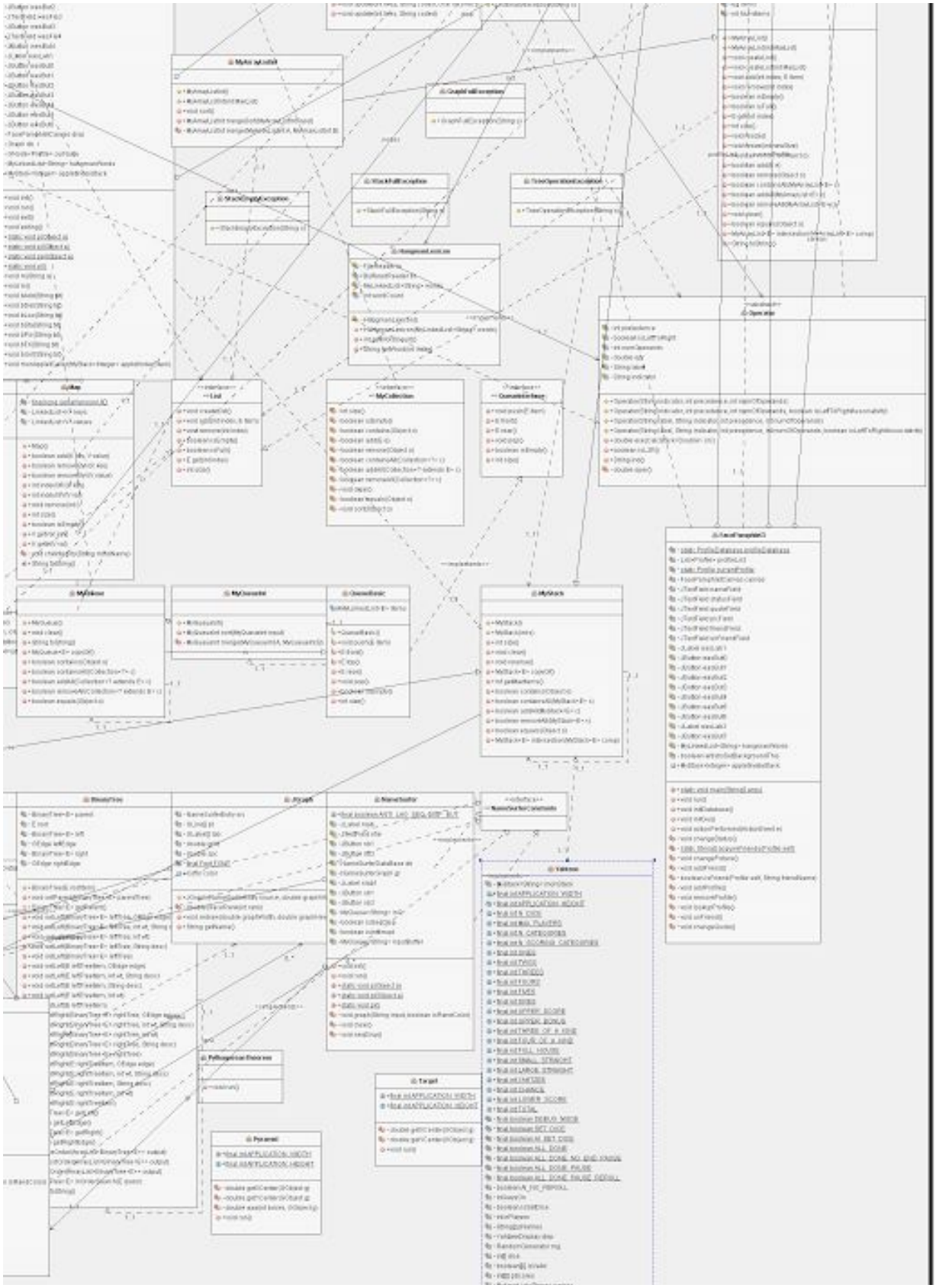


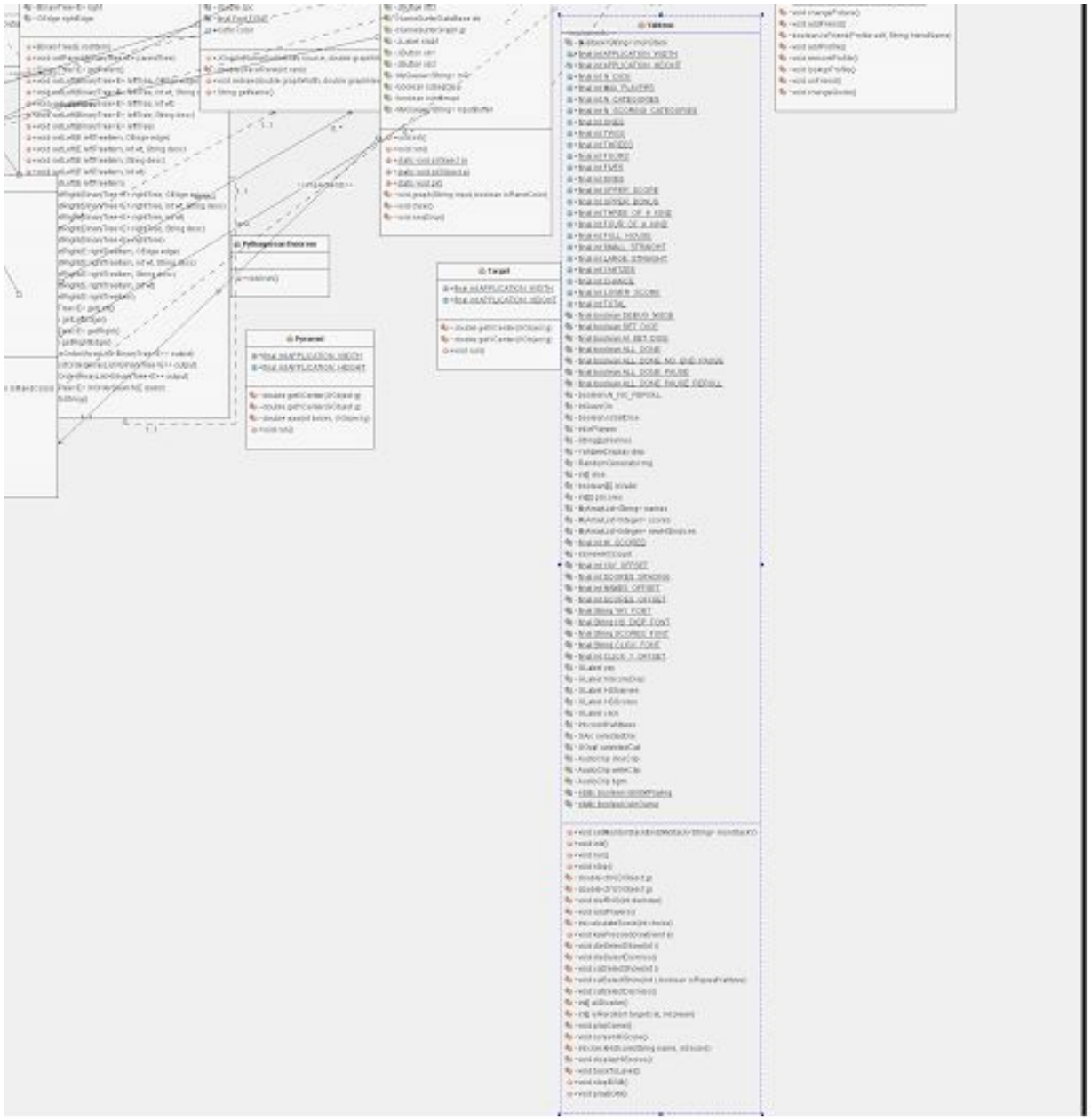




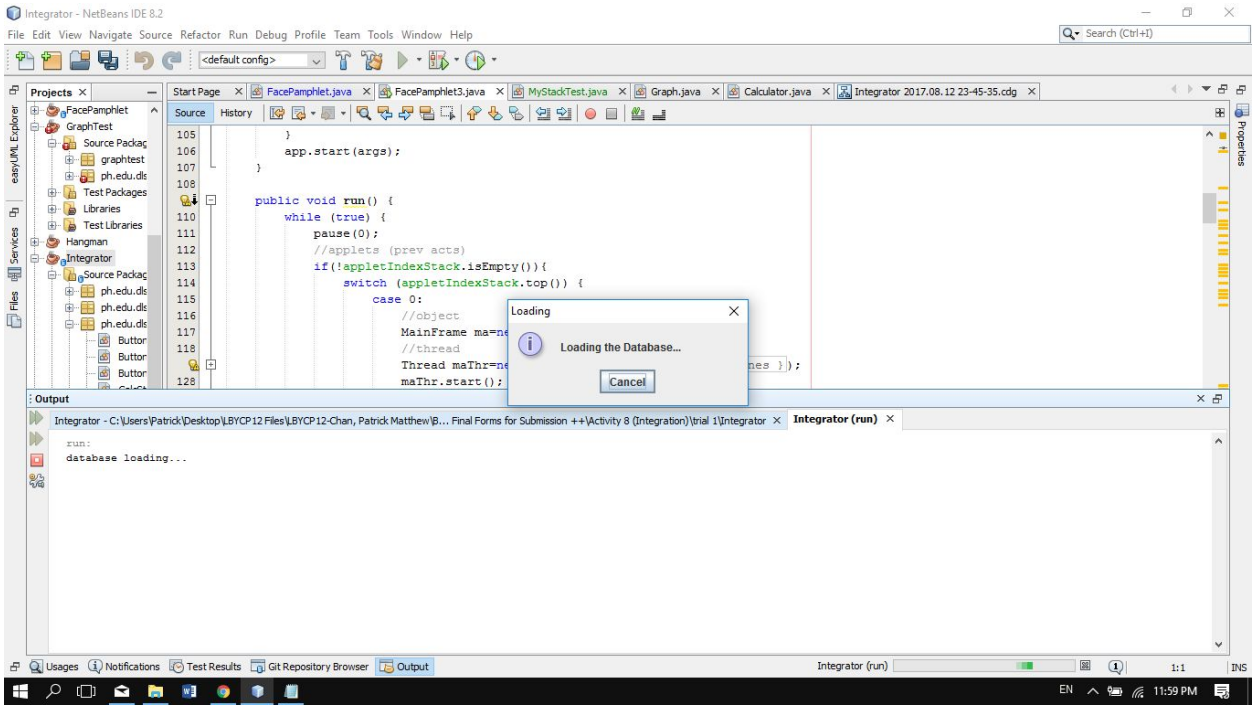
(other half:)



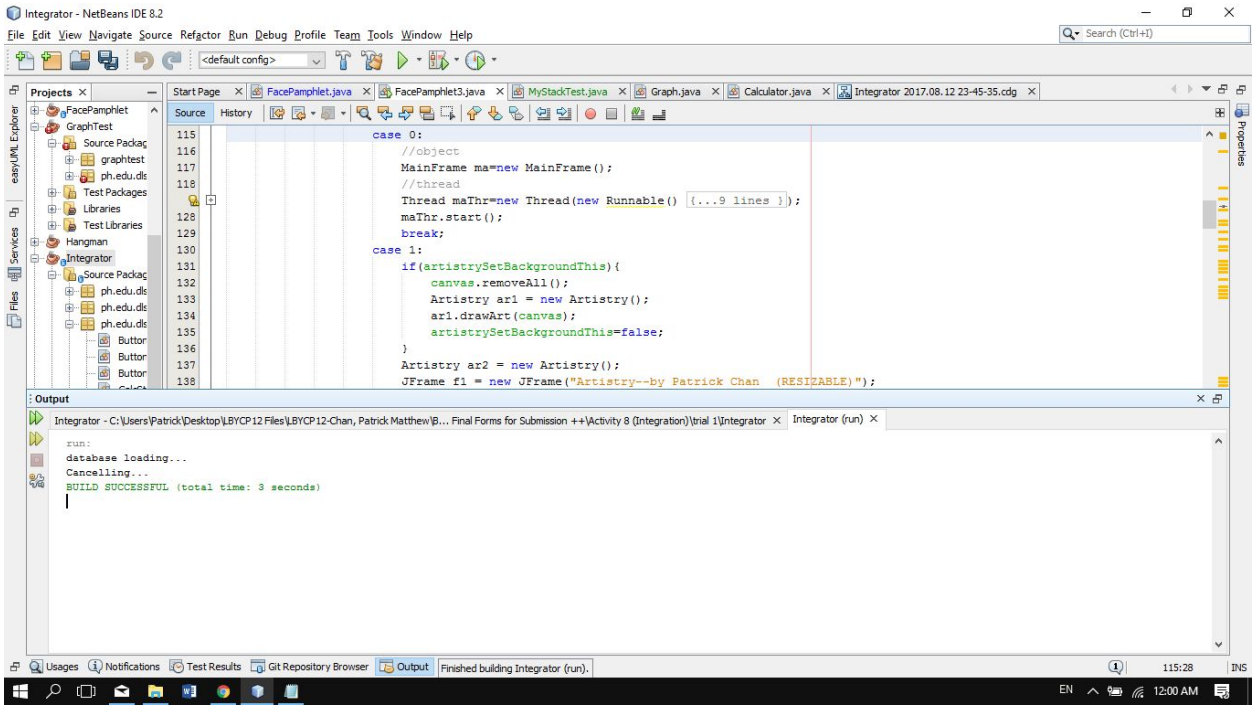




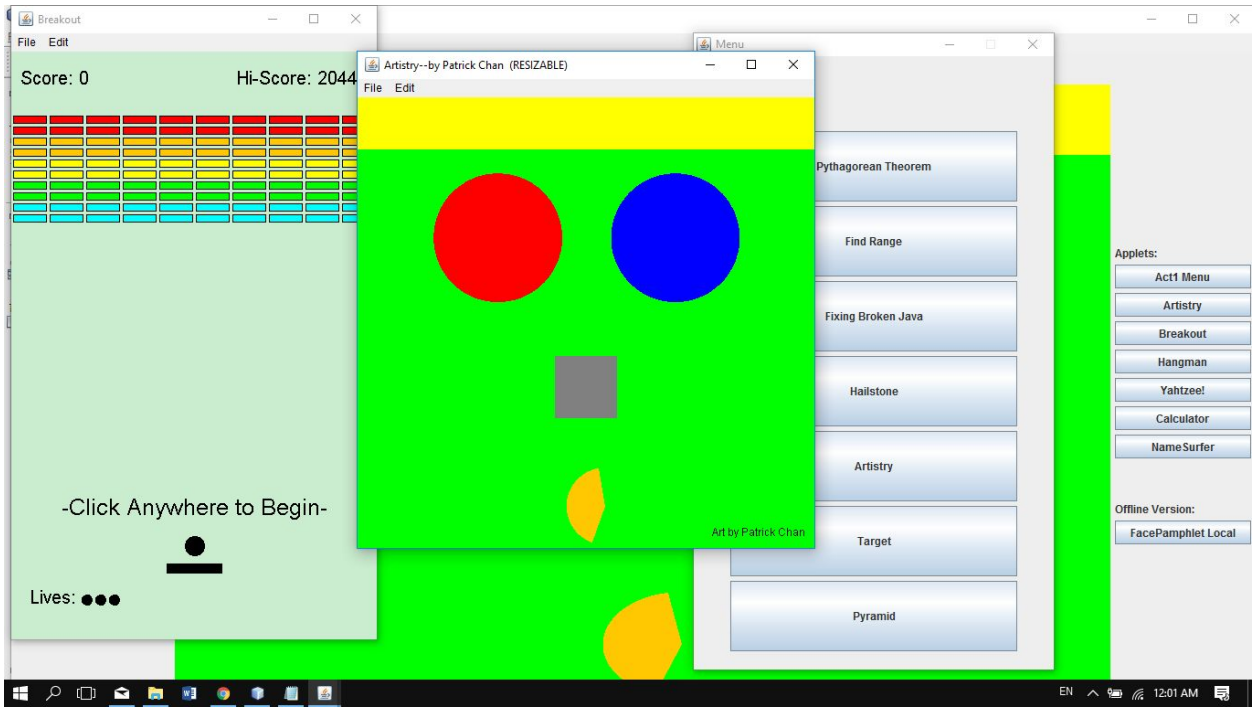
Screenshots:



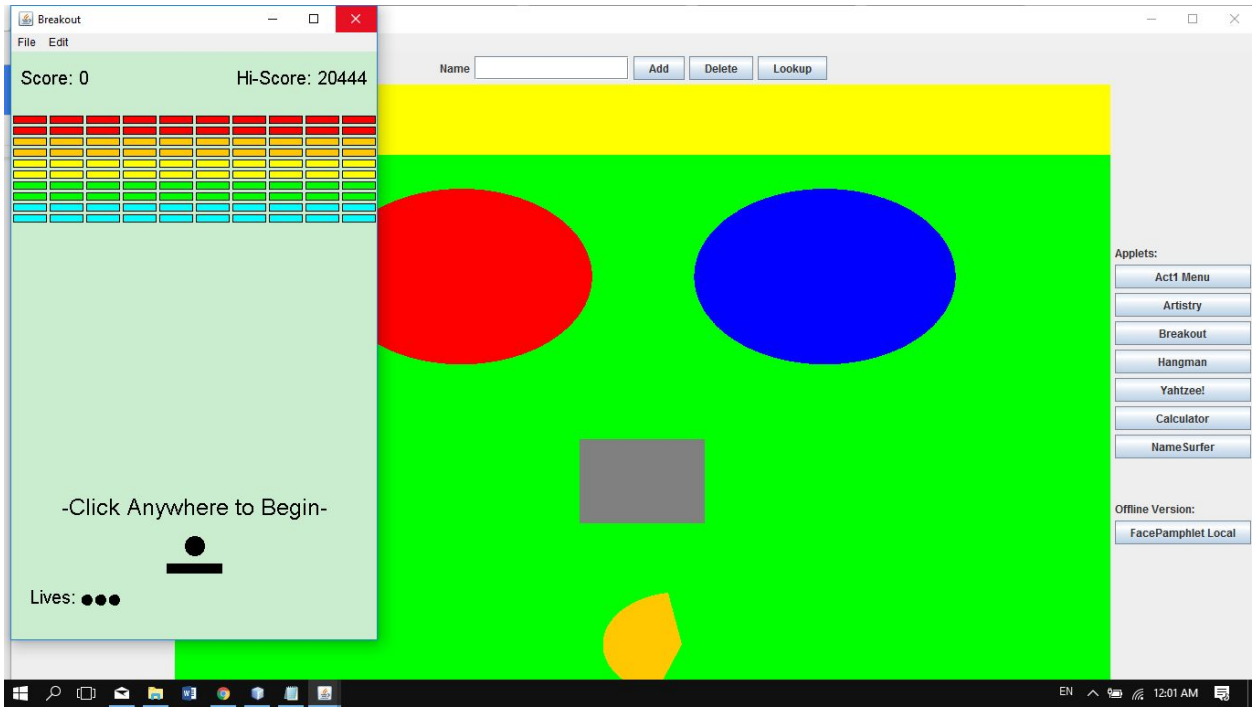
Loading the database takes a while, so a dialog box is shown, user can choose to cancel



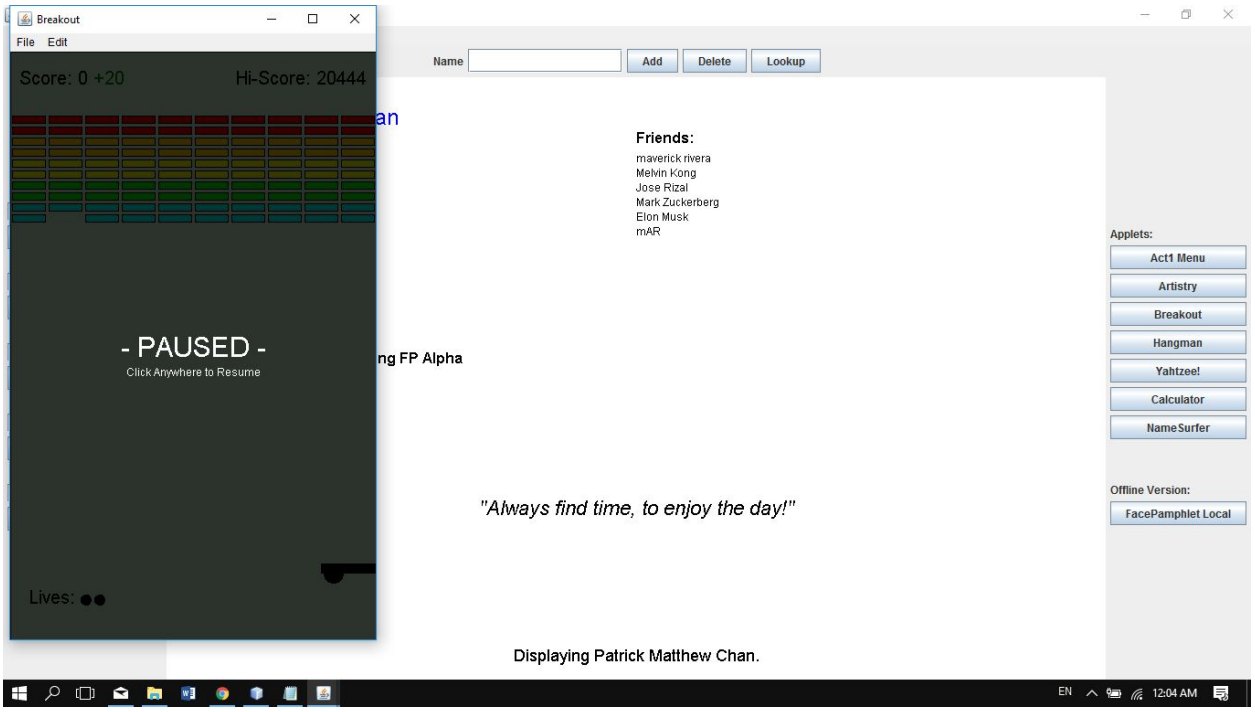
Terminated if cancelled



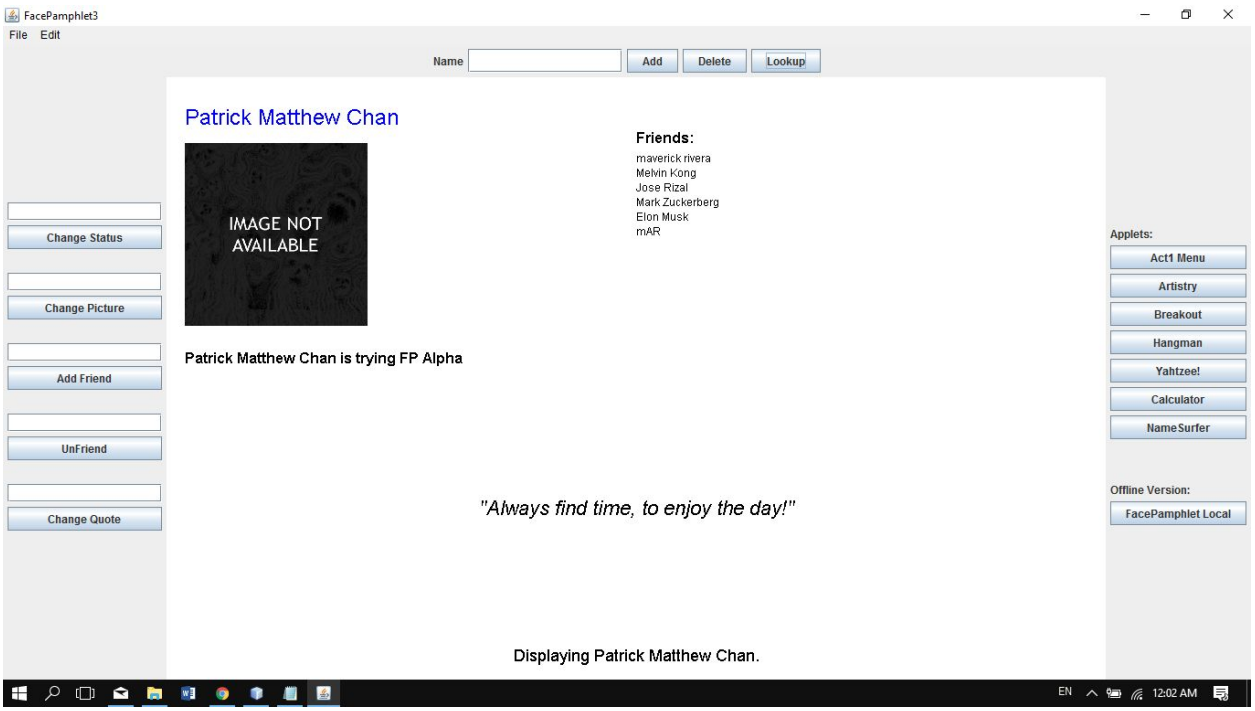
Programs can run at the same time



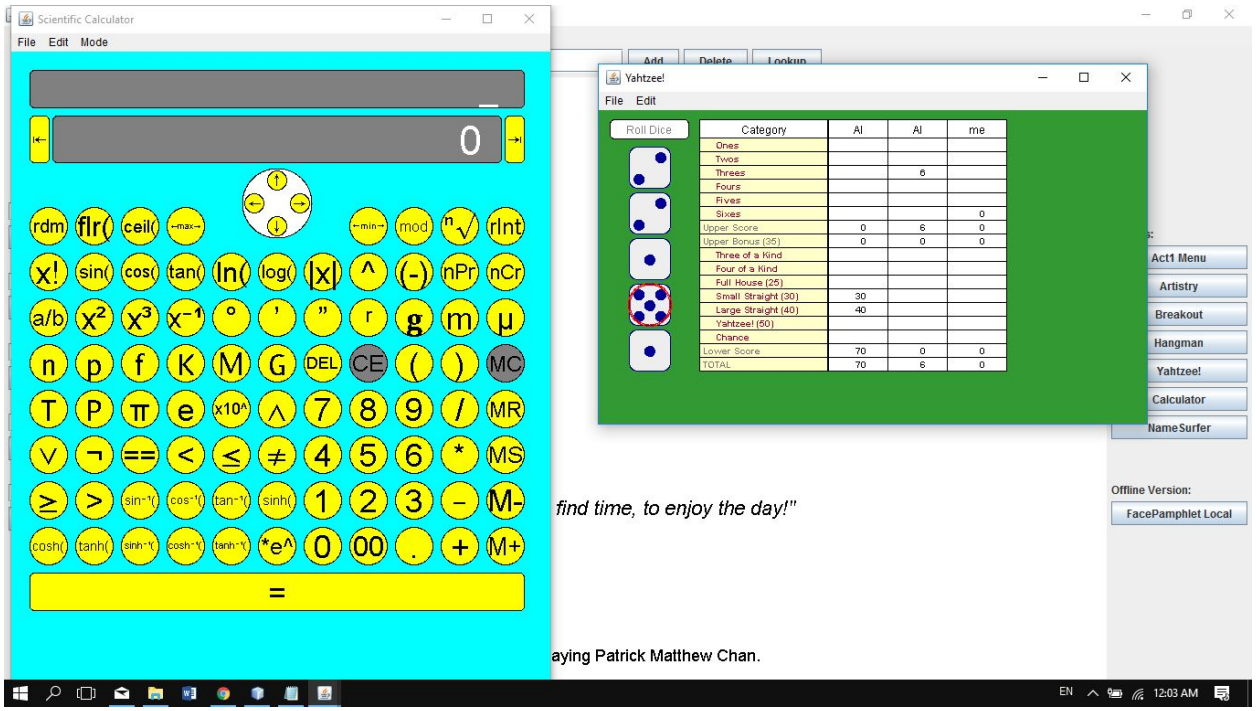
Main program does not close when “applets” are closed



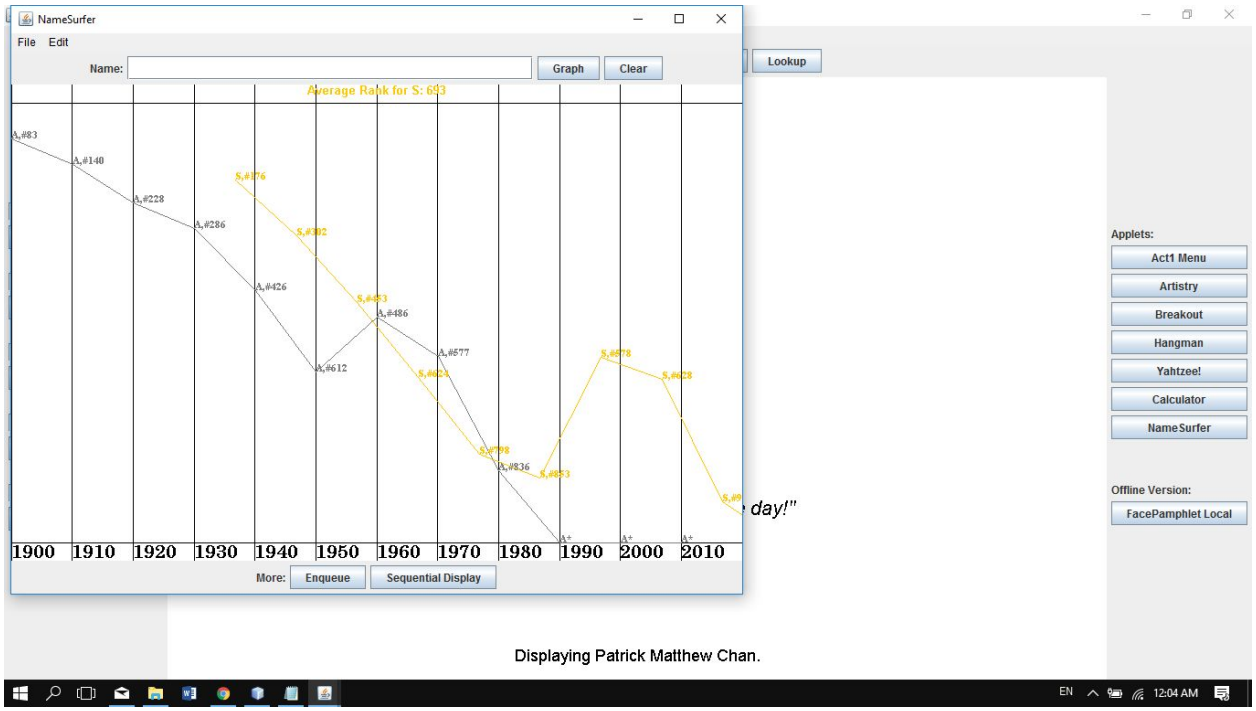
Breakout takes advantage of this window system and has a pause screen.



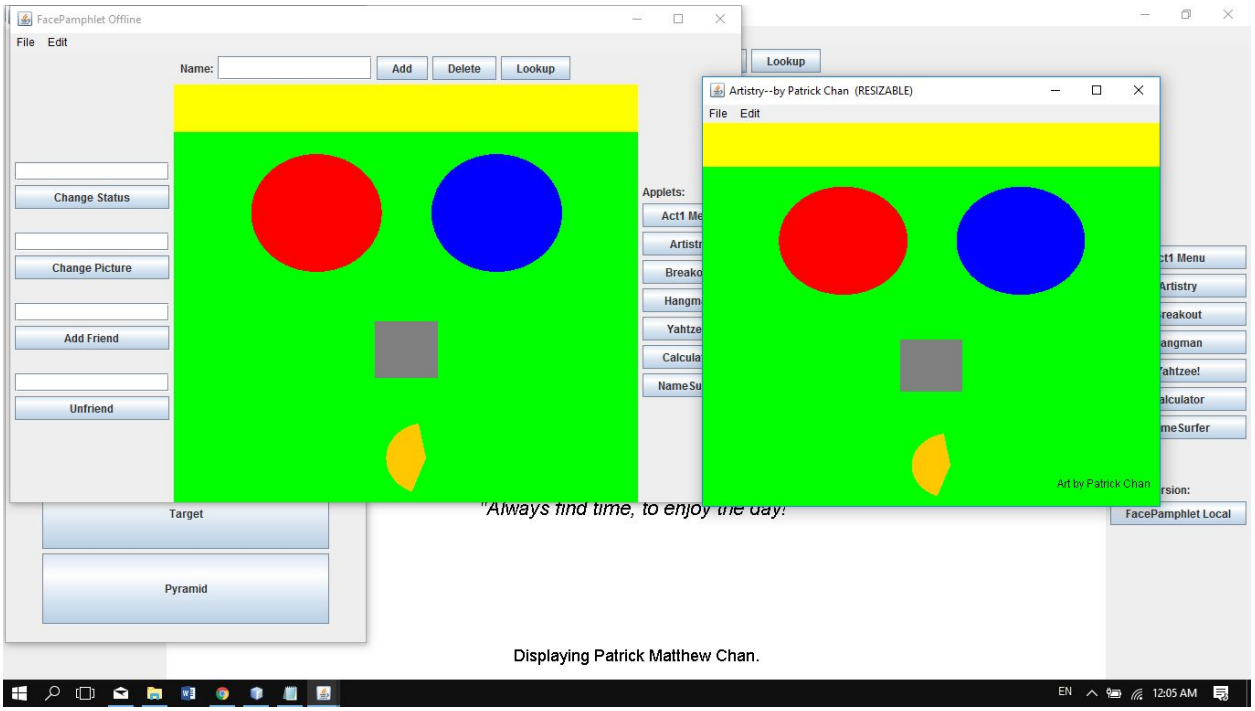
Main Functionality is still there



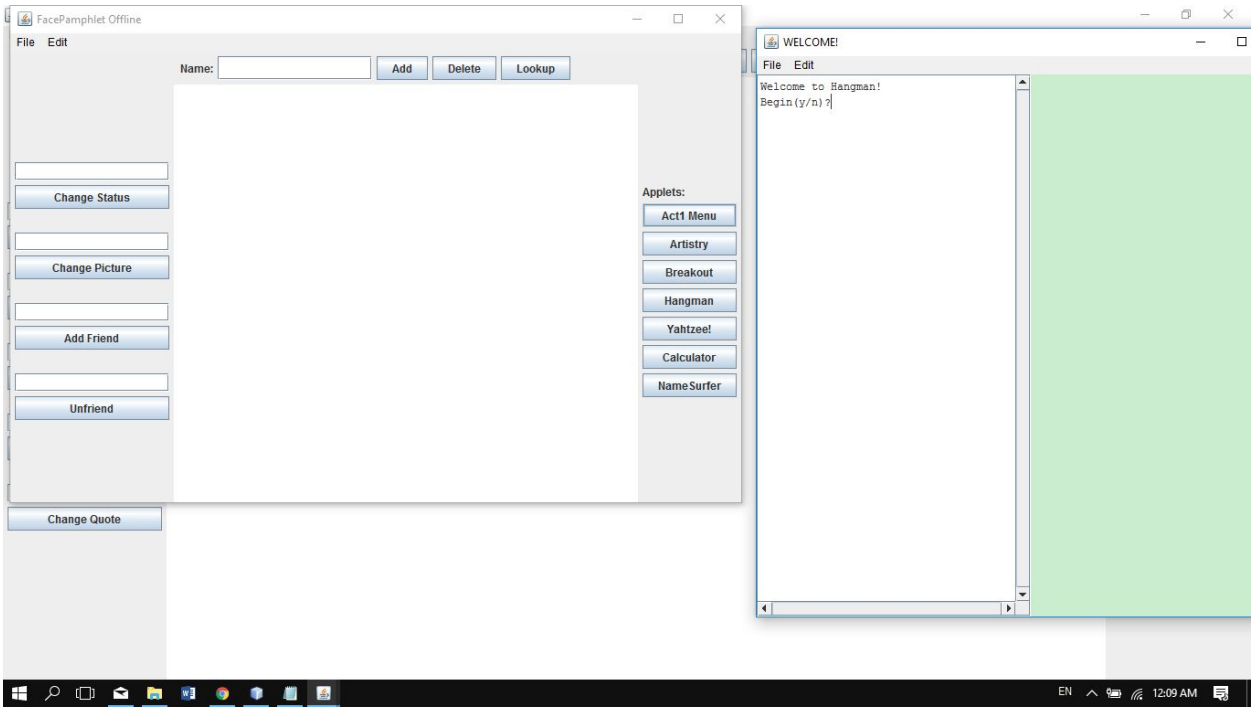
More Programs



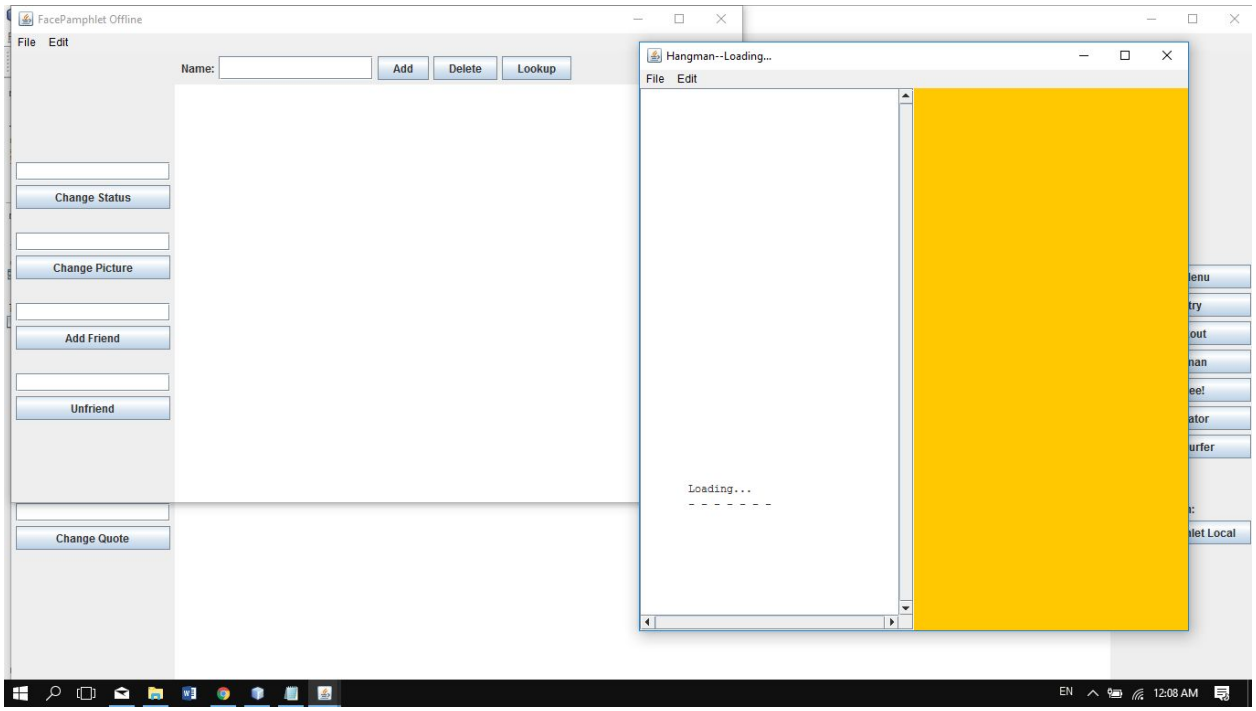
More programs



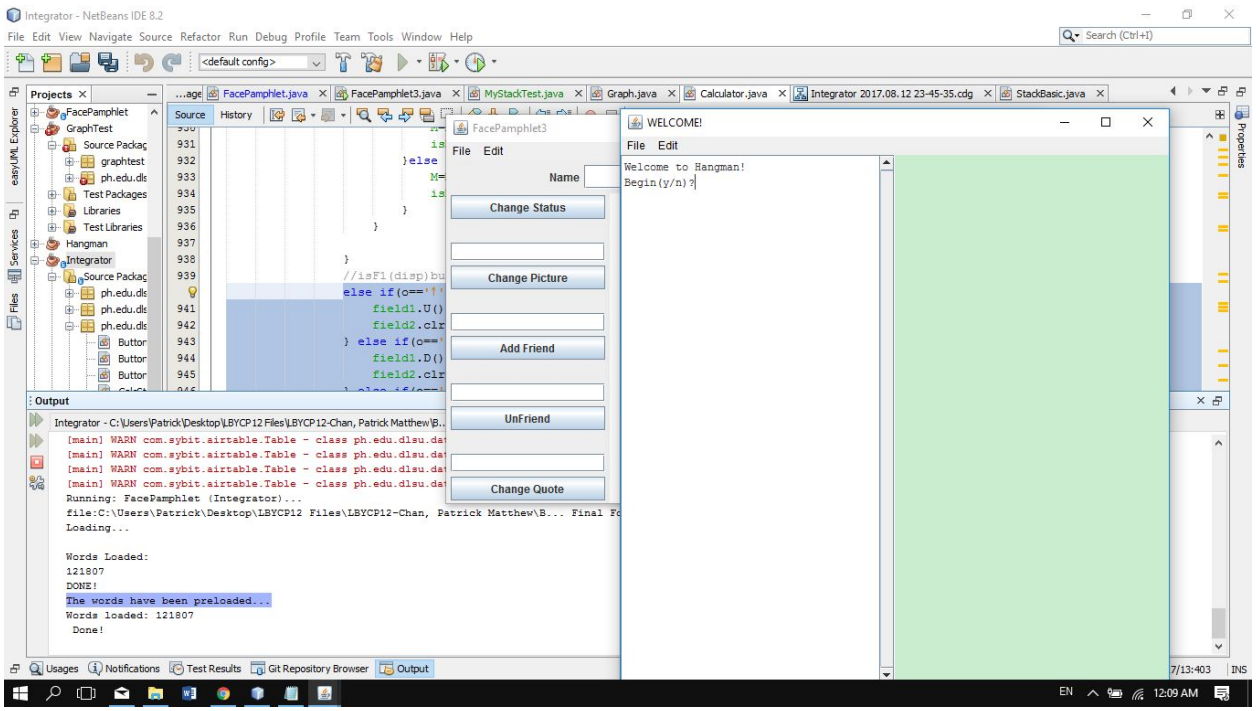
Local version also includes buttons, and its artistry changes its background instead.



Offline facepamphlet calls “applets” also, like the online version



Hangman has a loading screen, and words loaded don't need to be loaded again when hangman is called again.



Hangman doesn't need to load twice

SUMMARY

Before, I used to have a hard time trying to organize my class files in a project other than the default alphabetical order shown in the NetBeans pane. I even encountered times wherein it was quite hassle to keep renaming every call to a certain object as having the same names would cause the compiler to be confuse about the two objects and treat them as one. Here, I learned that it was actually just simple to solve this problem by creating packages inside java which act like folders, and importing these packages into my code instead.

REFERENCES

1. E Roberts. *Art and Science of Java*. Pearson; 2013.
2. E Roberts, M Sahami, and M Stepp, *CS 106A: Programming Methodology (Java) Handouts*, Stanford University.

APPENDIX

Main Class:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package ph.edu.dlsu.datasal.chan.facepamphlet3;

import acm.io.IODialog;
import com.sybit.airtable.exception.AirtableException;
import java.io.IOException;
import java.lang.reflect.InvocationTargetException;
import java.util.logging.Level;
import java.util.logging.Logger;
import org.apache.http.client.HttpResponseException;
```

```
import ph.edu.dlsu.datasal.facepamphlet.Profile;
import ph.edu.dlsu.datasal.facepamphlet.ProfileDatabase;

import acm.program.*;
import static acm.program.Program.EAST;
import acm.util.JTFTools;
import java.awt.Dimension;
import java.util.*;
import java.awt.event.*;
import javax.swing.*;
import ph.edu.dlsu.datasal.chan.hangman.HangmanConsole;
import ph.edu.dlsu.datasal.chan.namesurfer.NameSurfer;
import ph.edu.dlsu.datasal.chan.yahtzee.Yahtzee;
import ph.edu.dlsu.datasal.chan.artistry.Artistry;
import ph.edu.dlsu.datasal.chan.breakout.Breakout;
import ph.edu.dlsu.datasal.chan.calculator.Calculator;
import ph.edu.dlsu.datasal.chan.facepamphlet.FacePamphlet;
import static
ph.edu.dlsu.datasal.chan.facepamphlet.FacePamphlet.ACM_FRAME_OFFSET_X;
import static
ph.edu.dlsu.datasal.chan.facepamphlet.FacePamphlet.ACM_FRAME_OFFSET_Y;
import static
ph.edu.dlsu.datasal.chan.facepamphlet.FacePamphletConstants.EMPTY_LABEL_TEXT;
import ph.edu.dlsu.datasal.chan.menu.MainFrame;
import ph.edu.dlsu.datasal.chan.myarraylist.MyArrayList;
import ph.edu.dlsu.datasal.chan.mylinkedlist.MyLinkedList;
import ph.edu.dlsu.datasal.chan.mystack.MyStack;

/**
 *
 * @author cobalt mkc 2017
 */
public class FacePamphlet3 extends Program implements FacePamphletConstants {

    // Data
    private static ProfileDatabase profileDatabase;
    private List<Profile> profileList;
    private static Profile currentProfile;

    // GUI
    private FacePamphletCanvas canvas;
    private JTextField nameField;
    private JTextField statusField;
    private JTextField quoteField;
    private JTextField picField;
    private JTextField friendField;
    private JTextField unFriendField;
```

```
    JLabel easLab1 = new JLabel("Applets:");
    JButton easBut0 = new JButton("Act1 Menu");
    JButton easBut1 = new JButton("Artistry");
    JButton easBut2 = new JButton("Breakout");
    JButton easBut3 = new JButton("Hangman");
    JButton easBut4 = new JButton("Yahtzee!");
    JButton easBut5 = new JButton("Calculator");
    JButton easBut6 = new JButton("NameSurfer");
    JLabel easLab2 = new JLabel("Offline Version:");
    JButton easBut7 = new JButton("FacePamphlet Local");

    MyLinkedList<String> hangmanWords = null;
    boolean artistrySetBackgroundThis=false;
    public MyStack<Integer> appletIndexStack=new MyStack<>(1);

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        FacePamphlet3 app = new FacePamphlet3();
        JFrame fd=new JFrame();
        Thread info=new Thread(new Runnable() {
            @Override
            public void run() {
                String a[]=new String[1];
                a[0]=new String("Cancel");
                int option=JOptionPane.showOptionDialog(fd, "Loading the Database...",
"Loading",
                    JOptionPane.DEFAULT_OPTION, JOptionPane.INFORMATION_MESSAGE,
                    null,a,a[0]);//option is array index of choice, -1 if closed.
                if(option==0){
                    System.out.println("Cancelling...");
                    System.exit(0);
                }//System.out.println("closed");
                //thread auto stops when either "choice" is chosen
            }
        });
        info.start();
        System.out.println("database loading...");
        app.initDatabase();
        //will only reach here if database has finished loading
        fd.dispatchEvent(new WindowEvent(fd,WindowEvent.WINDOW_CLOSING));
        try {
        app.initGui();
        } catch (IOException ex) {
        Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null, ex);
```

```
    }
    app.start(args);
}

public void run() {
    while (true) {
        pause(0);
        //applets (prev acts)
        if(!appletIndexStack.isEmpty()){
            switch (appletIndexStack.top()) {
                case 0:
                    //object
                    MainFrame ma=new MainFrame();
                    //thread
                    Thread maThr=new Thread(new Runnable() {
                        @Override
                        public void run() {
                            ma.ini();
                            //thread automatically stops when menu window is closed.
                            //child threads are independent of this menu thread,
                            //and stop on their own when they are closed.
                        }
                    });
                    maThr.start();
                    break;
                case 1:
                    if(artistrySetBackgroundThis){
                        canvas.removeAll();
                        Artistry ar1 = new Artistry();
                        ar1.drawArt(canvas);
                        artistrySetBackgroundThis=false;
                    }
                    Artistry ar2 = new Artistry();
                    JFrame f1 = new JFrame("Artistry--by Patrick Chan  (RESIZABLE)");
                    //app thread
                    Thread arThr = new Thread(new Runnable() {
                        @Override
                        public void run() {

                            ar2.start(new String[0]);
                        }
                    });

                    //frame
                    f1.setVisible(true);
                    f1.setSize(new Dimension(ar2.APPLICATION_WIDTH
                            + ACM_FRAME_OFFSET_X, ar2.APPLICATION_HEIGHT
```

```
                    + ACM_FRAME_OFFSET_Y));
                f1.add(ar2);
                f1.setMenuBar(ar2.getMenuBar().createOldStyleMenuBar());
                f1.addWindowListener(new WindowAdapter() {
                    @Override
                    public void windowClosing(WindowEvent e) {
                        super.windowClosing(e);
                        ar2.stop();
                        f1.dispose();
                    }
                });      //f.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
                f1.validate();
                arThr.start();
                break;
            case 2:
                Runnable boRun = new Runnable() {
                    @Override
                    public void run() {
                        Breakout bo = new Breakout();
                        //bo.start(new String[0]);
                        JFrame f = new JFrame("Breakout");
                        f.setVisible(true);
                        f.setSize(new Dimension(bo.APPLICATION_WIDTH +
ACM_FRAME_OFFSET_X, bo.APPLICATION_HEIGHT + ACM_FRAME_OFFSET_Y));
                        f.add(bo);
                        f.setMenuBar(bo.getMenuBar().createOldStyleMenuBar());
                        f.addWindowListener(new WindowAdapter() {
                            @Override
                            public void windowClosing(WindowEvent e) {
                                super.windowClosing(e);
                                bo.stop();
                                f.dispose();
                            }

                            @Override
                            public void windowDeactivated(WindowEvent e) {
                                super.windowDeactivated(e);
                                bo.pauseGame();
                            }

                        });      //f.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
                        f.validate();
                        bo.setAutoPause(false);
                        bo.start(new String[0]);
                    }
                };
                Thread boThr = new Thread(boRun);
```

```
                boThr.start();
                break;
            case 3:
                HangmanConsole ha = new HangmanConsole();
                MyStack<String> monStackH
                    = new MyStack<>(1);
                JFrame f3 = new JFrame("Hangman");
                Runnable haRun = new Runnable() {
                  @Override
                  public void run() {
                    if (hangmanWords != null) {
                      ha.preloadWords(hangmanWords);
                    }
                    ha.monStackDone(monStackH);
                    ha.start(new String[0]);
                  }
                };
                Thread haThr = new Thread(haRun);
                //frame
                f3.setVisible(true);
                f3.setSize(new Dimension(600 + ACM_FRAME_OFFSET_X, 600 +
ACM_FRAME_OFFSET_Y));
                f3.add(ha);
                f3.setMenuBar(ha.getMenuBar().createOldStyleMenuBar());
                f3.addWindowListener(new WindowAdapter() {
                  @Override
                  public void windowClosing(WindowEvent e) {
                    super.windowClosing(e);
                    hangmanWords = ha.getPreloadWords();
                    ha.stop();
                    haThr.stop();
                    f3.dispose();
                  }
                });     //f.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
                f3.validate();//f.pack();

                Runnable monStRun = new Runnable() {
                  @Override
                  public void run() {
                    while (haThr.isAlive()) {
                      pause(500);
                      if(!f3.getTitle().equals(ha.getTitle())){
                        f3.setTitle(ha.getTitle());
                      }
                      if (!monStackH.isEmpty()) {
                        f3.dispatchEvent(new WindowEvent(f3,
WindowEvent.WINDOW_CLOSING));
```

```
                        }
                    }
                }
            };
            Thread monSThr = new Thread(monStRun);

            haThr.start();
            monSThr.start();
            monSThr.setPriority(Thread.MIN_PRIORITY);
            break;
        case 4:
            Yahtzee ya = new Yahtzee();
            JFrame f4 = new JFrame("Yahtzee!");
            MyStack<String> monStackY
                = new MyStack<String>(1);
            Runnable yaRun = new Runnable() {
                @Override
                public void run() {
                    ya.setMonitorStackEnd(monStackY);
                    ya.start(new String[0]);
                }
            };
            Thread yaThr = new Thread(yaRun);
            //frame
            f4.setVisible(true);
            f4.setSize(new Dimension(ya.APPLICATION_WIDTH +
ACM_FRAME_OFFSET_X, ya.APPLICATION_HEIGHT + ACM_FRAME_OFFSET_Y));
            f4.add(ya);
            f4.setMenuBar(ya.getMenuBar().createOldStyleMenuBar());
            f4.addWindowListener(new WindowAdapter() {
                @Override
                public void windowClosing(WindowEvent e) {
                    super.windowClosing(e);
                    ya.stop();
                    yaThr.stop();
                    f4.dispose();
                }

                @Override
                public void windowDeactivated(WindowEvent e) {
                    ya.stopBGM();
                }

                @Override
                public void windowActivated(WindowEvent e) {
                    ya.playBGM();
                }
```

```
                });      //f.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
                f4.validate();

                //monitor
                Thread monSThrY = new Thread(new Runnable() {
                   @Override
                   public void run() {
                      while (yaThr.isAlive()) {
                         pause(500);
                         if (!monStackY.isEmpty()) {
                            f4.dispatchEvent(new WindowEvent(f4,
WindowEvent.WINDOW_CLOSING));
                         }
                      }
                   }
                });
                //start threads
                yaThr.start();
                monSThrY.start();
                break;
             case 5:
                MyArrayList<Integer> monList= new MyArrayList<>(2);
                Calculator ca = new Calculator();
                Runnable caRun = new Runnable() {
                   @Override
                   public void run() {
                      ca.start(new String[0]);
                   }
                };
                Thread caThr = new Thread(caRun);

                //bo.start(new String[0]);
                JFrame f5 = new JFrame("Calculator");
                f5.setVisible(true);
                f5.setSize(new Dimension(400 + ACM_FRAME_OFFSET_X, 400 +
ACM_FRAME_OFFSET_Y));
                f5.add(ca);
                //f.setMenuBar(ca.getMenuBar().createOldStyleMenuBar());
                f5.addWindowListener(new WindowAdapter() {
                   @Override
                   public void windowClosing(WindowEvent e) {
                      super.windowClosing(e);
                      ca.stop();
                      f5.dispose();
                      caThr.stop();
                   }
                });      //f.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
```

```
                f5.validate();
                ca.monSizeList(monList);
                Thread monThr = new Thread(new Runnable() {
                   @Override
                   public void run() {
                      int a = 400;
                      int b = 400;
                      pause(500);
                      f5.setMenuBar(ca.getMenuBar().createOldStyleMenuBar());
                      while (caThr.isAlive()) {
                         pause(100);
                         if(!f5.getTitle().equals(ca.getTitle())){
                            f5.setTitle(ca.getTitle());
                         }
                         if (monList.size() == 2 && !(monList.get(1).equals(a))
                               && !(monList.get(2).equals(b))) {
                            a = monList.get(1);
                            b = monList.get(2);
                            System.out.println("a = " + a + ",b=" + b);
                            f5.setSize(a + ACM_FRAME_OFFSET_X, b +
ACM_FRAME_OFFSET_Y);
                         }
                      }
                   }
                });
                caThr.start();
                monThr.start();
                monThr.setPriority(Thread.MIN_PRIORITY);
                break;
             case 6:
                Runnable naRun = new Runnable() {
                   @Override
                   public void run() {
                      NameSurfer na = new NameSurfer();
                      //bo.start(new String[0]);
                      JFrame f = new JFrame("NameSurfer");
                      f.setVisible(true);
                      f.setSize(new Dimension(na.APPLICATION_WIDTH +
ACM_FRAME_OFFSET_X, na.APPLICATION_HEIGHT + ACM_FRAME_OFFSET_Y));
                      f.add(na);
                      f.setMenuBar(na.getMenuBar().createOldStyleMenuBar());
                      f.addWindowListener(new WindowAdapter() {
                         @Override
                         public void windowClosing(WindowEvent e) {
                            super.windowClosing(e);
                            na.stop();
                            f.dispose();
```

```
                        }
                    });      //f.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
                    f.validate();
                    na.start(new String[0]);
                }
            };
            Thread naThr = new Thread(naRun);
            naThr.start();
            break;
        case 7:
            Runnable fpRun = new Runnable() {
                @Override
                public void run() {
                    FacePamphlet fp = new FacePamphlet();
                    JFrame f7 = new JFrame("FacePamphlet Offline");
                    f7.setVisible(true);
                    f7.setSize(new Dimension(fp.APPLICATION_WIDTH +
ACM_FRAME_OFFSET_X, fp.APPLICATION_HEIGHT + ACM_FRAME_OFFSET_Y));
                    f7.add(fp);
                    f7.setMenuBar(fp.getMenuBar().createOldStyleMenuBar());
                    f7.addWindowListener(new WindowAdapter() {
                        @Override
                        public void windowClosing(WindowEvent e) {
                            super.windowClosing(e);
                            fp.stop();
                            fp.exiting();
                            f7.dispose();
                        }
                    });      //f.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
                    f7.validate();

                    //start
                    fp.monAppletCaller(appletIndexStack);
                    fp.start(new String[0]);
                }
            };
            Thread fpThr = new Thread(fpRun);
            fpThr.start();
            break;
        default:
            System.out.println("Unsupported AppletCalledIndex:"+
                    appletIndexStack.top());
            break;
      }
      appletIndexStack.pop();
    }
  }
```

```
  }


  public void initDatabase() {
     try {
        profileDatabase = new ProfileDatabase();
        profileList = profileDatabase.getProfileList();
     } catch (HttpResponseException | AirtableException ex) {
        Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null, ex);
     }
  }

  /**
   * This method has the responsibility for initializing the interactors in
   * the application, and taking care of any other initialization that needs
   * to be performed.
   *
   * @throws java.io.IOException
   */
  public void initGui() throws IOException {

     /*       GUI NORTH : Name Add Delete Lookup     */
/* Name */
     add(new JLabel("Name"), NORTH);
     nameField = new JTextField(TEXT_FIELD_SIZE);
     add(nameField, NORTH);
     nameField.addActionListener(this);

     add(new JButton("Add"), NORTH);
     add(new JButton("Delete"), NORTH);
     add(new JButton("Lookup"), NORTH);

     /*       GUI WEST : Status Picture AddFriend Unfriend    */
/* Change Status */
     statusField = new JTextField(TEXT_FIELD_SIZE);
     add(statusField, WEST);
     statusField.addActionListener(this);
     add(new JButton("Change Status"), WEST);
     add(new JLabel(EMPTY_LABEL_TEXT), WEST);
     /* Change Picture */
     picField = new JTextField(TEXT_FIELD_SIZE);
     add(picField, WEST);
     picField.addActionListener(this);
     add(new JButton("Change Picture"), WEST);
     add(new JLabel(EMPTY_LABEL_TEXT), WEST);
     /* Add Friend */
     friendField = new JTextField(TEXT_FIELD_SIZE);
```

```
add(friendField, WEST);
friendField.addActionListener(this);
add(new JButton("Add Friend"), WEST);
/* Un Friend */
unFriendField = new JTextField(TEXT_FIELD_SIZE);
add(new JLabel(EMPTY_LABEL_TEXT), WEST);
add(unFriendField, WEST);
unFriendField.addActionListener(this);
add(new JButton("UnFriend"), WEST);
/* Change Quote */
quoteField = new JTextField(TEXT_FIELD_SIZE);
add(new JLabel(EMPTY_LABEL_TEXT), WEST);
add(quoteField, WEST);
quoteField.addActionListener(this);
add(new JButton("Change Quote"), WEST);
add(new JLabel(EMPTY_LABEL_TEXT), WEST);

addActionListeners();

//east
add(easLab1, EAST);
add(easBut0, EAST);
add(easBut1, EAST);
add(easBut2, EAST);
add(easBut3, EAST);
add(easBut4, EAST);
add(easBut5, EAST);
add(easBut6, EAST);
add(new JLabel(EMPTY_LABEL_TEXT), EAST);
add(new JLabel(EMPTY_LABEL_TEXT), EAST);
add(easLab2, EAST);
add(easBut7, EAST);

//east
easBut0.addActionListener(new ActionListener() {
   @Override
   public void actionPerformed(ActionEvent e) {
      appletIndexStack.push(0);
   }
});
easBut1.addActionListener(new ActionListener() {
   @Override
   public void actionPerformed(ActionEvent e) {
      artistrySetBackgroundThis=true;
      appletIndexStack.push(1);
   }
});
```

```
    easBut2.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            appletIndexStack.push(2);
        }
    });
    easBut3.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            appletIndexStack.push(3);
        }
    });
    easBut4.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            appletIndexStack.push(4);
        }
    });
    easBut5.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            appletIndexStack.push(5);
        }
    });
    easBut6.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            appletIndexStack.push(6);
        }
    });
    easBut7.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            appletIndexStack.push(7);
        }
    });

    canvas = new FacePamphletCanvas();
    add(canvas);
}

/**
 * This method is responsible for detecting when the buttons are clicked or
 * interactors are used, so you will have to add code to respond to these
 * actions.
 *
 * @param e
```

```
    */
   @Override
   public void actionPerformed(ActionEvent e) {
      if (e.getActionCommand().equals("Change Status") || e.getSource() == statusField) {
         try {
            changeStatus();
         } catch (AirtableException ex) {
            Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null, ex);
         }
         statusField.setText("");
      } else if (e.getActionCommand().equals("Change Picture") || e.getSource() ==
picField) {
         try {
            changePicture();
         } catch (AirtableException ex) {
            Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null, ex);
         }
         picField.setText("");
      } else if (e.getActionCommand().equals("Add Friend") || e.getSource() ==
friendField) {
         try {
            try {
               addFriend();
            } catch (HttpResponseException ex) {
               Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null,
ex);
            }
         } catch (AirtableException ex) {
            Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null, ex);
         }
         friendField.setText("");
      } else if (e.getActionCommand().equals("UnFriend") || e.getSource() ==
unFriendField) {
         try {
            unFriend();
         } catch (AirtableException | HttpResponseException ex) {
            Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null, ex);
         }
         unFriendField.setText("");
      } else if (e.getActionCommand().equals("Add") && !nameField.getText().equals(""))
{
         try {
            addProfile();
         } catch (AirtableException ex) {
            Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null, ex);
         }
         nameField.setText("");
```

```
      } else if (e.getActionCommand().equals("Delete")) {
         removeProfile();
         nameField.setText("");
      } else if (e.getActionCommand().equals("Lookup")) {
         try {
            lookupProfile();
         } catch (AirtableException ex) {
            Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null, ex);
         }
         nameField.setText("");
      } else if (e.getActionCommand().equals("Change Quote") || e.getSource() ==
quoteField) {
         try {
            changeQuote();
         } catch (AirtableException ex) {
            Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null, ex);
         }
         quoteField.setText("");
      }
   }

   private void changeStatus() throws AirtableException {
      String status = statusField.getText();
      if (!(status.equals(""))) {
         if (currentProfile != null) {
            profileDatabase.updateProfileStatus(currentProfile, status);
            canvas.displayProfile(currentProfile);
            canvas.displayFriends(currentProfile, acquireFriends(currentProfile));
            canvas.showMessage("Status updated to " + status);
         } else {
            canvas.showMessage("Please select a profile to change status.");
         }
      }
   }

   private static String[] acquireFriends(Profile self) throws AirtableException {
      String[] friends = null;
      String[] friendIds = self.getFriends();
      if (friendIds != null) {
         friends = new String[friendIds.length];
         for (int i = 0; i < friends.length; ++i) {
            friends[i] = (profileDatabase.getProfileById(friendIds[i])).getName();
         }
      }
      return friends;
   }
```

```
   private void changePicture() throws AirtableException {
      String picName = picField.getText();
      if (!picName.equals("")) {
         if (currentProfile != null) {
            try {
               currentProfile = profileDatabase.updateProfilePhoto(currentProfile,
picName);
            } catch (IOException ex) {
               Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null,
ex);
            }
            if (currentProfile != null) {
               canvas.displayProfile(currentProfile);
               canvas.displayFriends(currentProfile, acquireFriends(currentProfile));
               canvas.showMessage("Picture updated.");
            }
         }
      } else {
         canvas.showMessage("Please select a profile to change picture.");
      }
   }

   private void addFriend() throws AirtableException, HttpResponseException {
      String friendName = friendField.getText();
      if (!friendName.equals("")) {
         if (currentProfile != null) {
            if (profileDatabase.containsProfile(friendName)) {
               if (!isFriend(currentProfile, friendName)) {
                  Profile friend = profileDatabase.getProfileByName(friendName);
                  currentProfile = profileDatabase.addFriend(currentProfile, friend);
                  canvas.displayProfile(currentProfile);
                  try {
                     canvas.displayFriends(currentProfile, acquireFriends(currentProfile));
                  } catch (AirtableException ex) {
                     Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE,
null, ex);
                  }
                  canvas.showMessage(friendName + " added as a friend.");
               } else {
                  canvas.showMessage(friendName + " is already a friend.");
               }
            } else {
               canvas.showMessage(friendName + " does not exist.");
            }
         } else {
            canvas.showMessage("Please select a profile to add a friend to.");
         }
```

```
    }
  }

  private boolean isFriend(Profile self, String friendName) throws AirtableException {
    String[] friendList = acquireFriends(self);
    if (friendList != null && friendList.length > 0) {
      for (String element : friendList) {
        if (element.equals(friendName)) {
          return true;
        }
      }
    }
    return false;
  }

  private void addProfile() throws AirtableException {
    String name = nameField.getText();
    if (!profileDatabase.containsProfile(name)) {
      Profile newProfile = new Profile();
      newProfile.setName(name);
      try {
        currentProfile = profileDatabase.addProfile(newProfile);
      } catch (AirtableException | IllegalAccessException | InvocationTargetException
| NoSuchMethodException | HttpResponseException ex) {
        Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null, ex);
      }
      canvas.displayProfile(currentProfile);
      canvas.displayFriends(currentProfile, acquireFriends(currentProfile));
      canvas.showMessage("New profile created");
    } else {
      currentProfile = profileDatabase.getProfileByName(name);
      canvas.displayProfile(currentProfile);
      canvas.displayFriends(currentProfile, acquireFriends(currentProfile));
      canvas.showMessage("A profile with the name " + name + " already exists.");
    }
  }

  private void removeProfile() {
    String name = nameField.getText();
    if (!name.equals("")) {
      currentProfile = null;
      canvas.removeAll();
      if (profileDatabase.containsProfile(name)) {
        try {
          profileDatabase.deleteProfile(name);
        } catch (AirtableException | HttpResponseException ex) {
          Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE, null,
```

```
ex);
            }
            canvas.showMessage("Profile of " + name + " deleted.");
        } else {
            canvas.showMessage("A profile with the name " + name + " does not exist.");
        }
    }
  }

  private void lookupProfile() throws AirtableException {
    String name = nameField.getText();
    if (!name.equals("")) {
        if (profileDatabase.containsProfile(name)) {
            currentProfile = profileDatabase.getProfileByName(name);
            canvas.displayProfile(currentProfile);
            canvas.displayFriends(currentProfile, acquireFriends(currentProfile));
            canvas.showMessage("Displaying " + name + ".");
        } else {
            currentProfile = null;
            canvas.removeAll();
            canvas.showMessage("A profile with the name " + name + " does not exist.");
        }
    }
  }

  private void unFriend() throws AirtableException, HttpResponseException {
    String friendName = unFriendField.getText();
    if (!friendName.equals("")) {
        if (currentProfile != null) {
            if (profileDatabase.containsProfile(friendName)) {
                if (isFriend(currentProfile, friendName)) {
                    Profile friend = profileDatabase.getProfileByName(friendName);
                    currentProfile = profileDatabase.unFriend(currentProfile, friend);
                    canvas.displayProfile(currentProfile);
                    try {
                        canvas.displayFriends(currentProfile, acquireFriends(currentProfile));
                    } catch (AirtableException ex) {
                        Logger.getLogger(FacePamphlet3.class.getName()).log(Level.SEVERE,
null, ex);
                    }
                    canvas.showMessage(friendName + " removed as a friend.");
                } else {
                    canvas.showMessage(friendName + " and " + currentProfile.getName() + "
are not friends.");
                }

            } else {
```

```
                canvas.showMessage(friendName + " does not exist.");
            }
        } else {
            canvas.showMessage("Please select a profile to remove a friend to.");
        }
    }
}

private void changeQuote() throws AirtableException {
    String quote = quoteField.getText();
    if (!(quote.equals(""))) {
        if (currentProfile != null) {
            profileDatabase.updateProfileQuote(currentProfile, quote);
            canvas.displayProfile(currentProfile);
            canvas.displayFriends(currentProfile, acquireFriends(currentProfile));
            canvas.showMessage("Quote updated to " + quote);
        } else {
            canvas.showMessage("Please select a profile to change quote.");
        }
    }
}

}
```

The project consists of a lot of packages of classes, and the rest of the files can be found at https://drive.google.com/drive/u/1/folders/0B_ngwdgBk09tOFRmd3BSY3R1T2s.