

# GUI programming

Graphical user interface-based  
programming

# Windchill

- Windchill

- There are several formulas for calculating the windchill temperature  $t_{wc}$
- The one provided by U.S. National Weather Service and is applicable for a windspeed greater than four miles per hour



$$t_{wc} = 0.081(t - 91.4)(3.71\sqrt{v} + 5.81 - 0.25v) + 91.4$$

- Where

- Variable  $t$  is the Fahrenheit temperature
- Variable  $v$  is the windspeed in miles per hour

# Console-based programming

## Console program

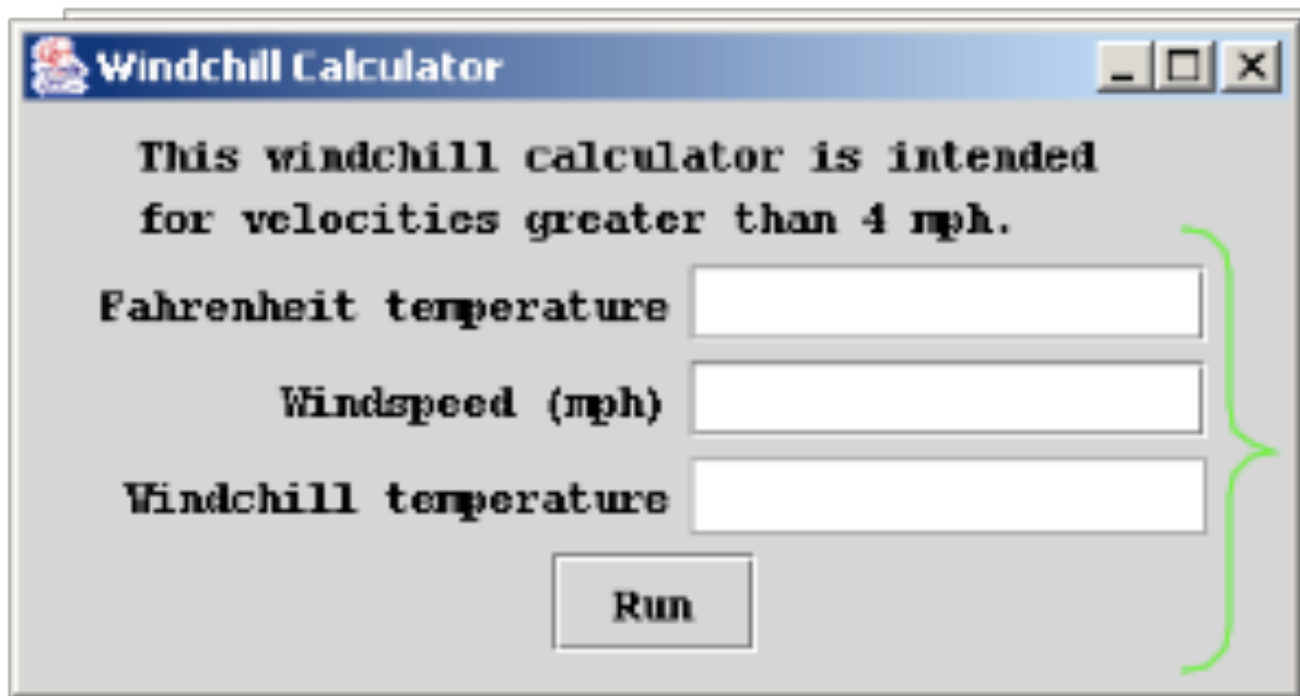
```
Method main() {  
    statement1;  
    statement2;  
    ...  
    statementm;  
}
```



Console programs  
begin and end in  
method main()

# In use

Program needs to respond whenever the run button is clicked



There needs to be an event loop that is looking for user interface events

# GUI-based programming

## GUI Program

```
main() {  
    GUI gui = new GUI();  
}  
  
GUI Constructor() {  
    constructor1;  
    constructor2;  
    ...  
    constructorn;  
}  
  
Action Performer() {  
    action1;  
    action2;  
    ...  
    actionk;  
}
```

GUI program begins in method main(). The method creates a new instance of the GUI by invoking the GUI constructor. On completion, the event dispatching loop is begun

Constructor configures the components of the GUI. It also registers the listener-performer for user interactions

## Event-dispatching loop

```
do  
    if an event occurs  
        then signal its  
        action listeners  
until program ends
```

The event-dispatching loop watches for user interactions with the GUI. When an event occurs, its listener-performers are notified

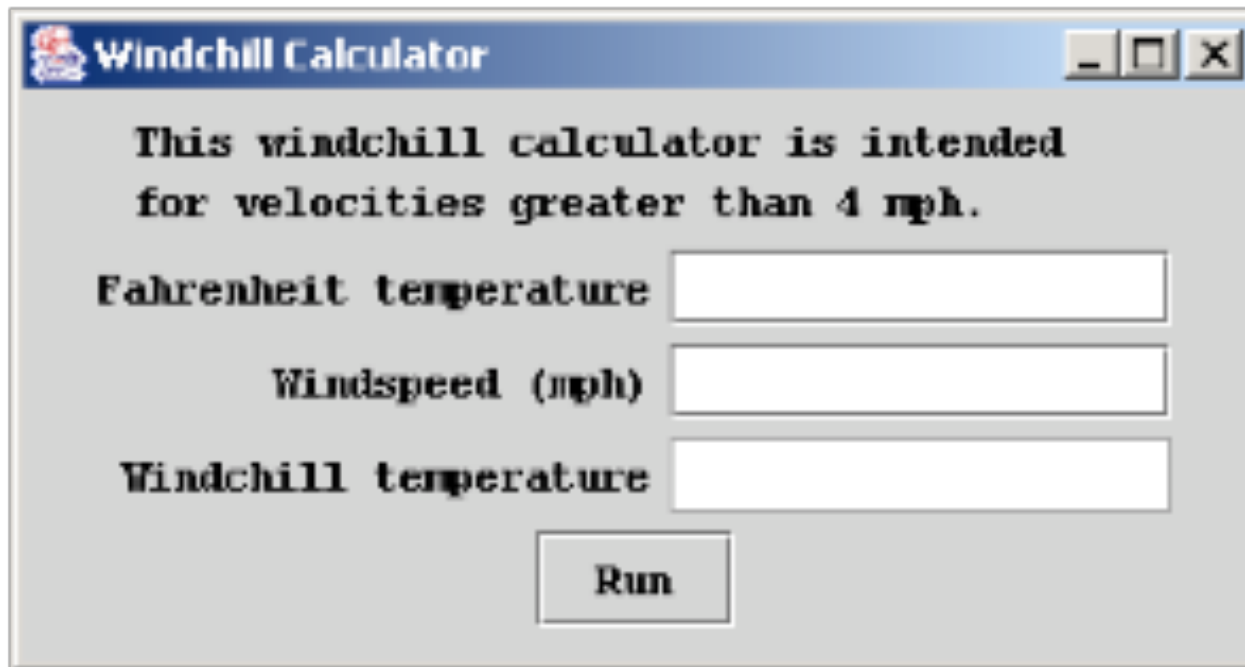
The action performer implements the task of the GUI. After it completes, the event-dispatching loop is restarted

# Java support

- JFrame
  - Represents a titled, bordered window
- JLabel
  - Represents a display area suitable for one or both of a single-line text or image.
- JTextField
  - Represents an editable single-line text entry component
- JButton
  - Represents a push button
- JTextArea
  - Represents an editable multiline text entry component

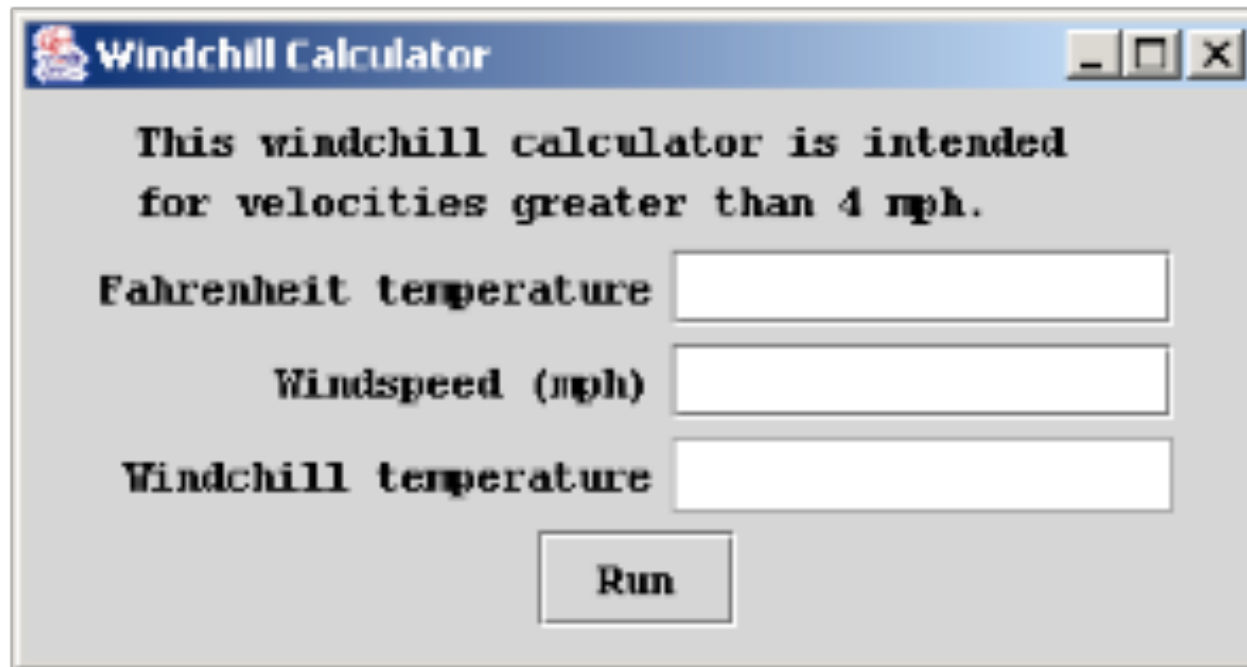
# Instance variables

- private JFrame window
  - References the window containing the other components of the GUI



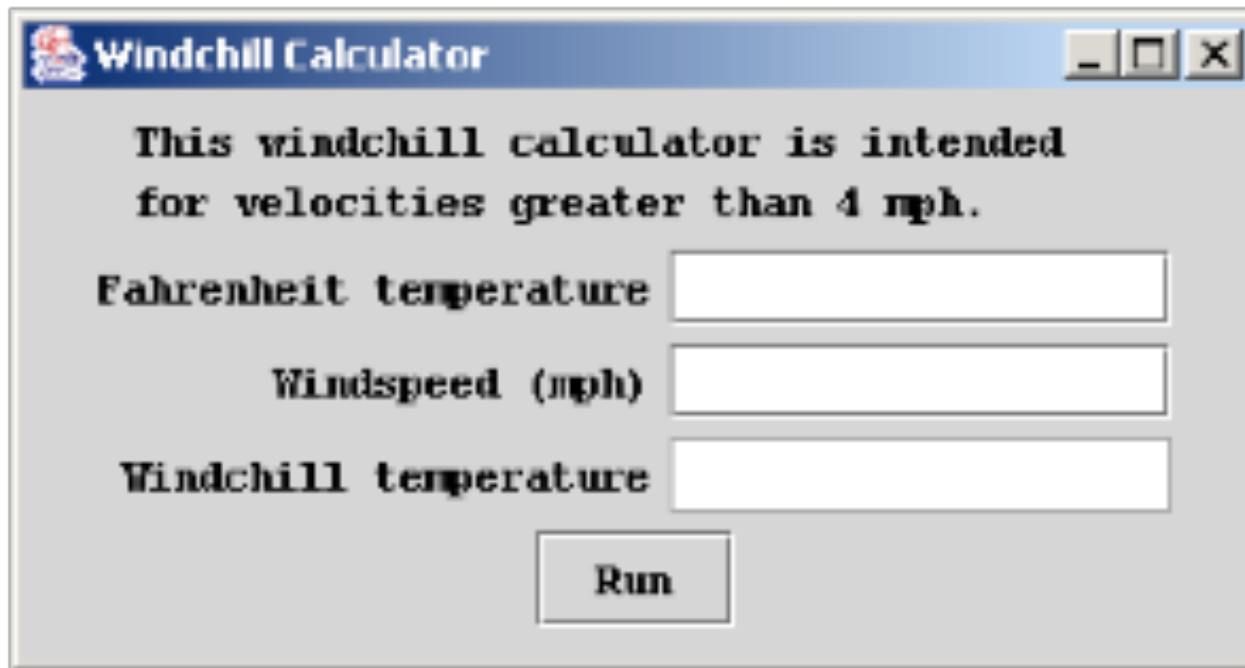
# Instance variables

- private JTextArea legendArea
  - References the text display for the multiline program legend



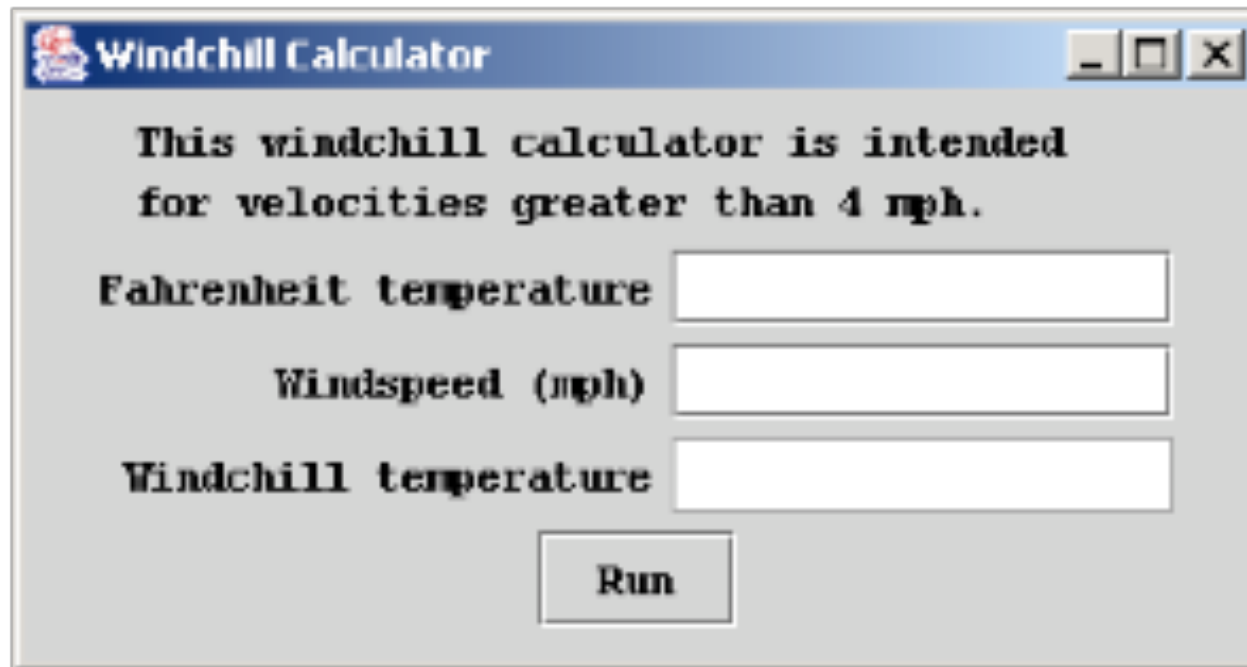
# Instance variables

- private JLabel fahrTag
  - References the label for the data entry area supplying the temperature



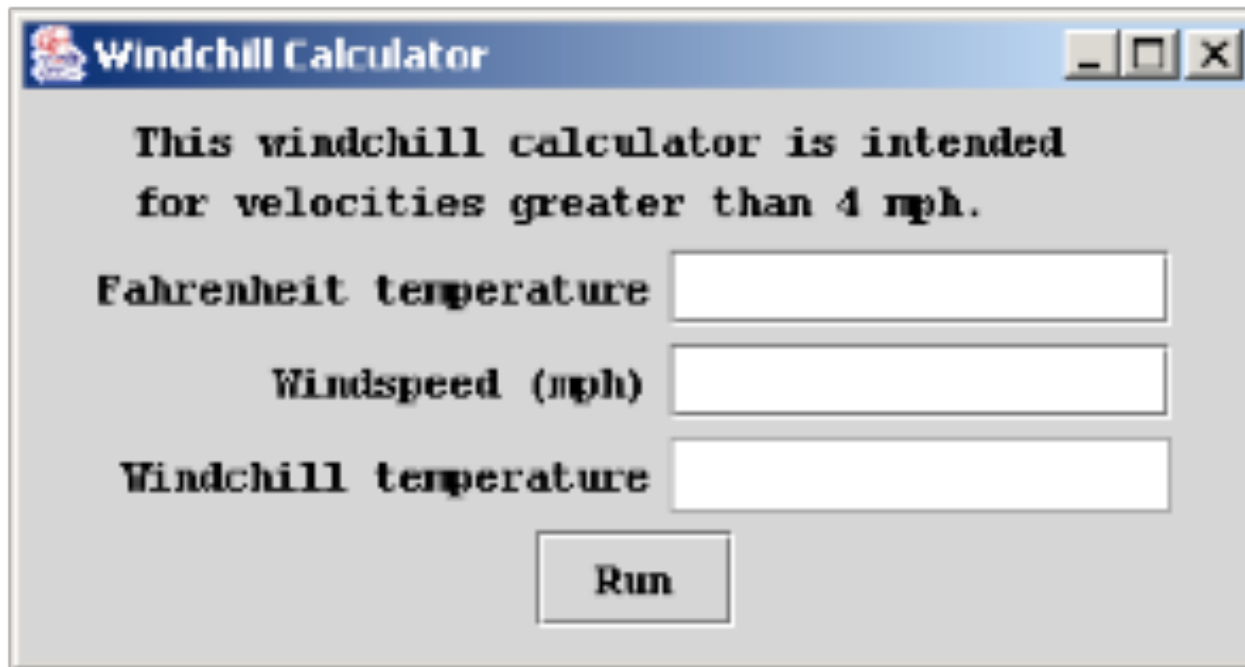
# Instance variables

- private JTextField fahrText
  - References the data area supplying the temperature



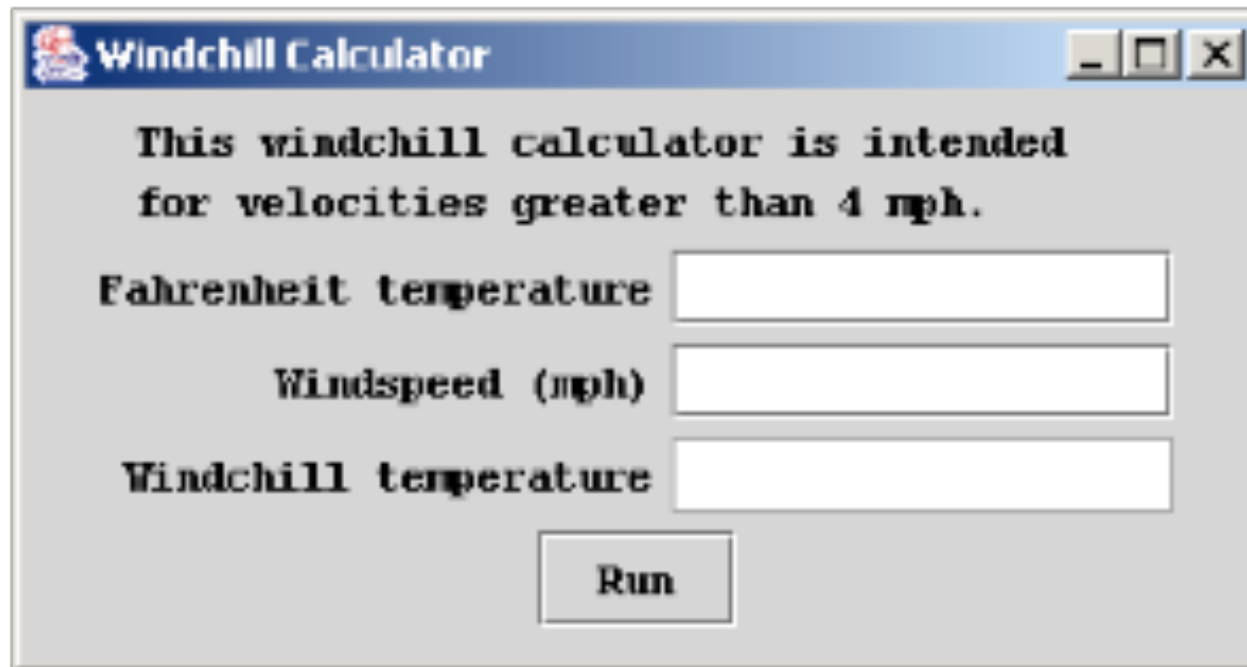
# Instance variables

- private JLabel windTag
  - References the label for the data entry area supplying the windspeed



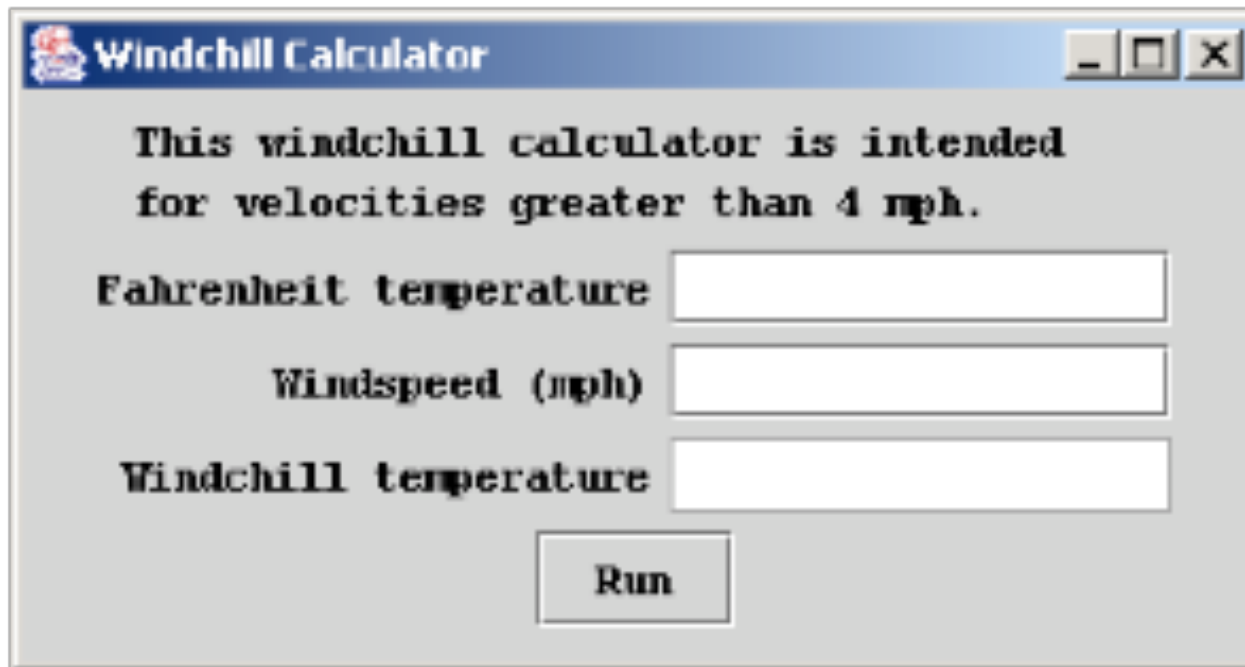
# Instance variables

- private JTextField windText
  - References the data area supplying the windspeed



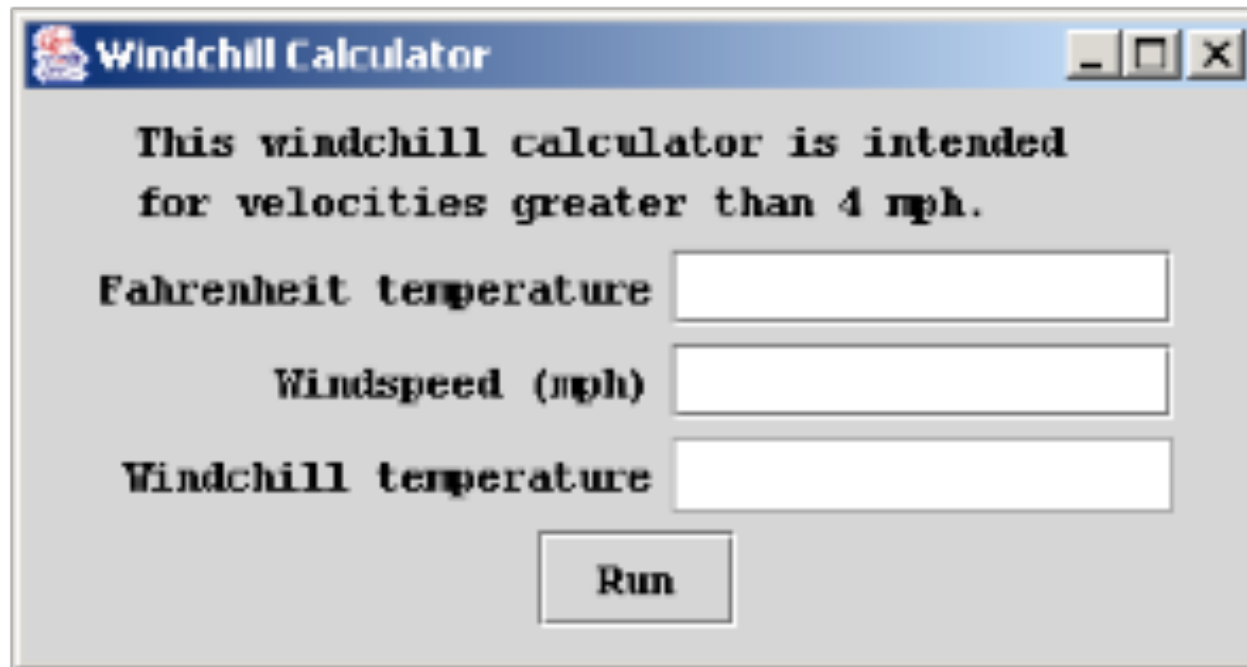
# Instance variables

- private JLabel chillTag
  - References the label for the data area giving the windchill



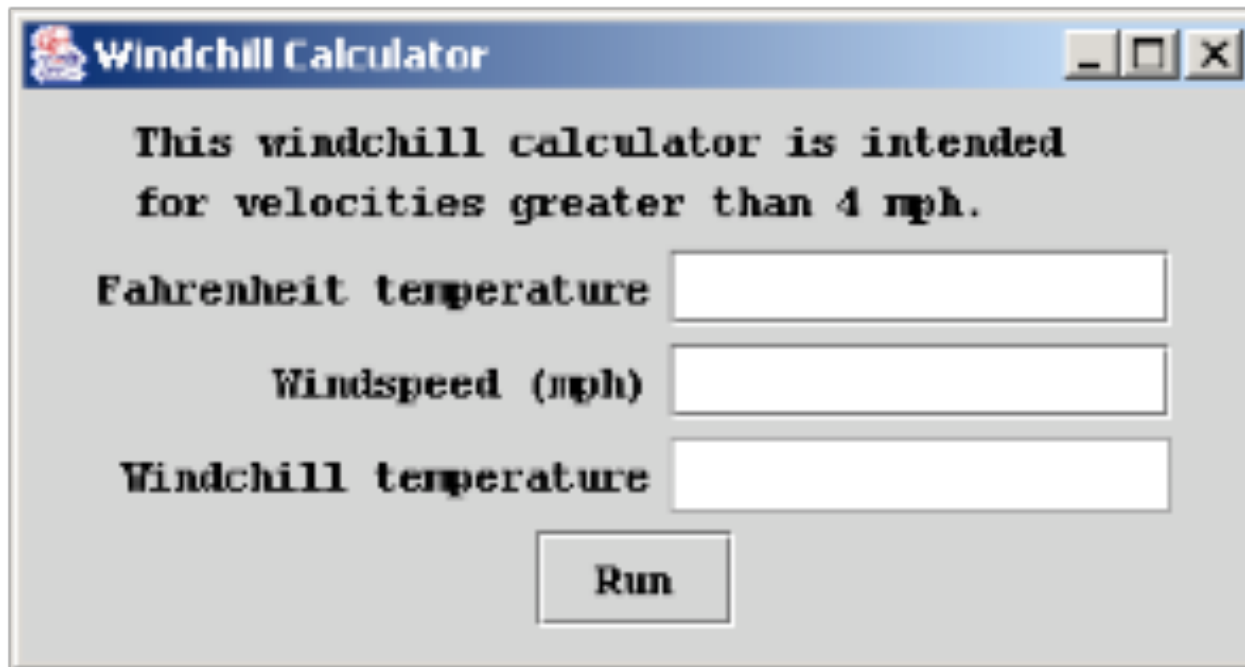
# Instance variables

- private JTextField chillText
  - References the data area giving the windchill



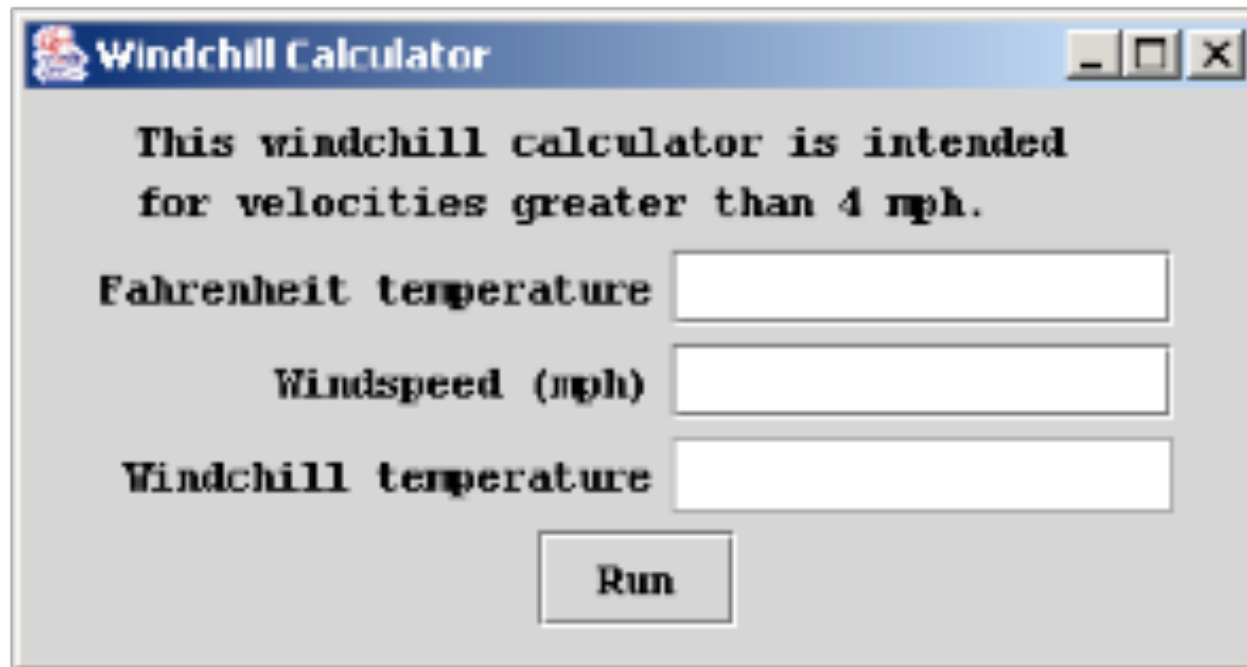
# Class constants

- private static final String LEGEND = "This windchill calculator"  
+ "is intended for velocities greater than 4 mph."
  - Program legend text



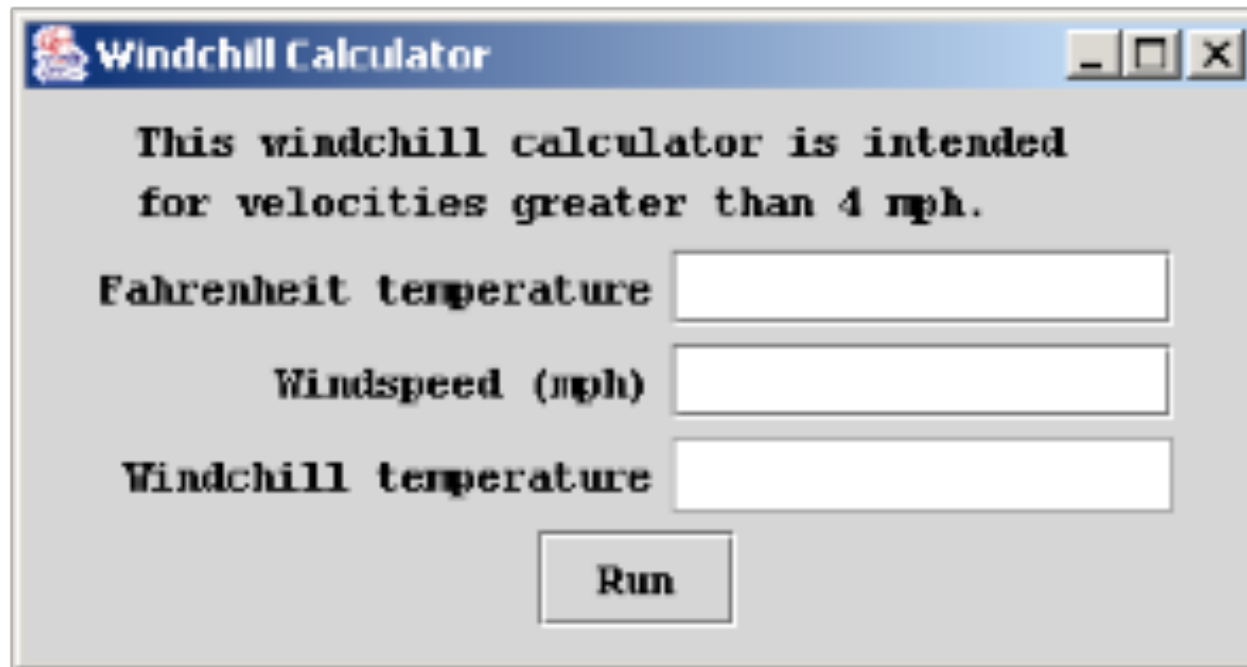
# Class constants

- private static final int WINDOW\_WIDTH = 350
  - Initial width of the GUI



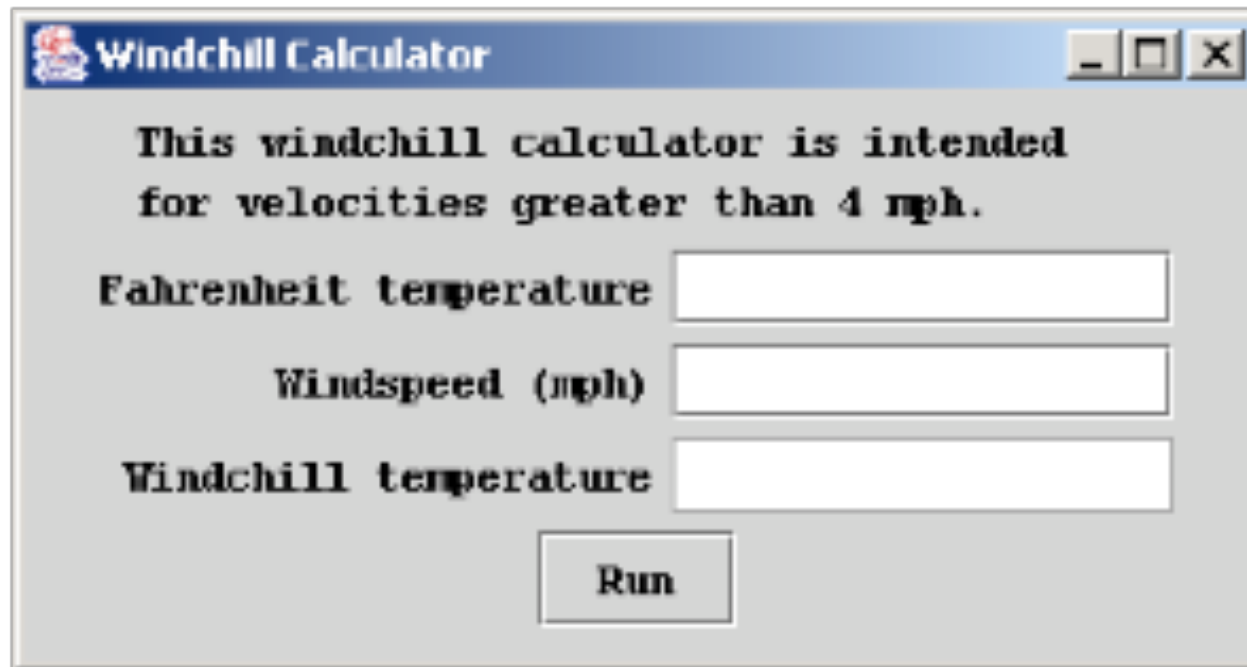
# Class constants

- `private static final int WINDOW_HEIGHT = 185`
  - Initial height of the GUI



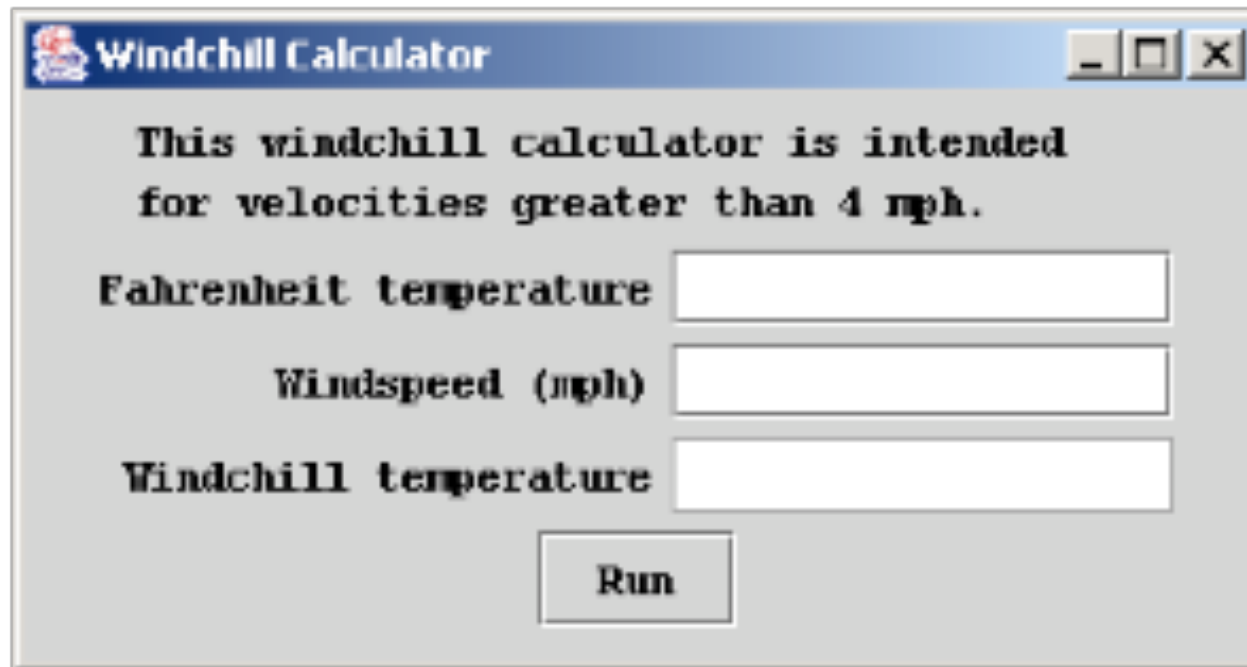
# Class constants

- private static final int AREA\_WIDTH = 40
  - Width of the program legend in characters



# Class constants

- private static final int FIELD\_WIDTH = 40
  - Number of characters per data entry area



Windchill Calculator

This windchill calculator is intended  
for velocities greater than 4 mph.

Fahrenheit temperature

Windspeed (mph)

Windchill temperature

Run

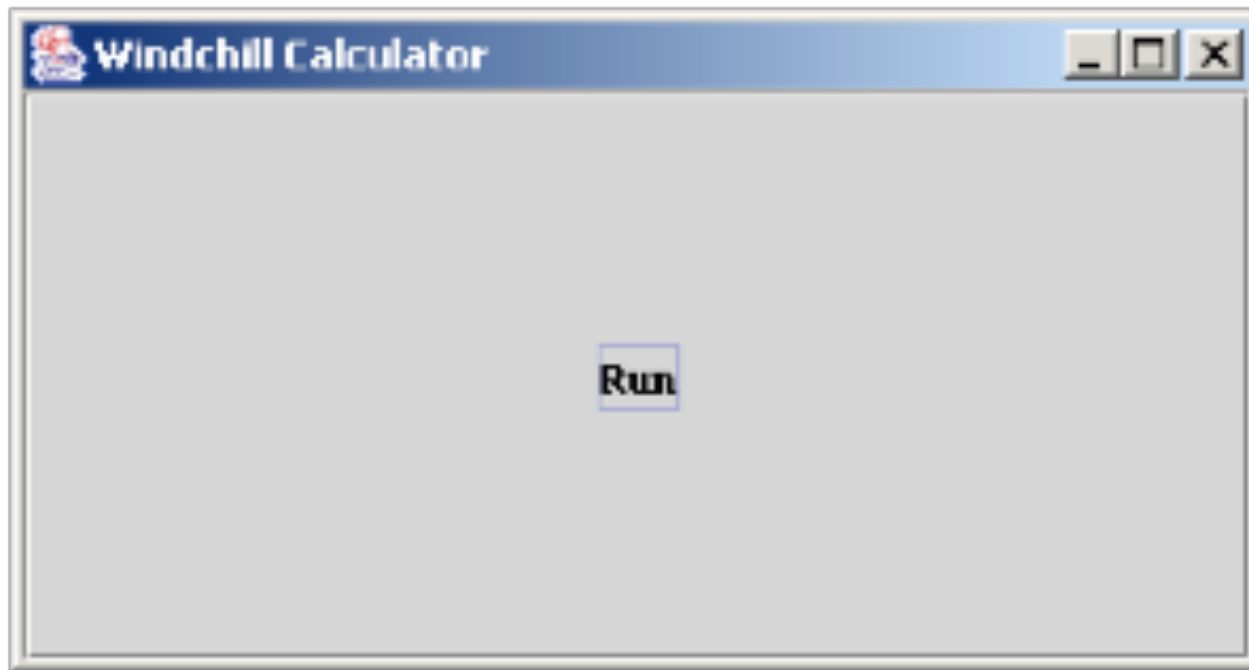
# Class constants

- private static final `FlowLayout LAYOUT_STYLE = new FlowLayout()`
  - References manager that lays out GUI components in a top-to-bottom, left-to-right manner



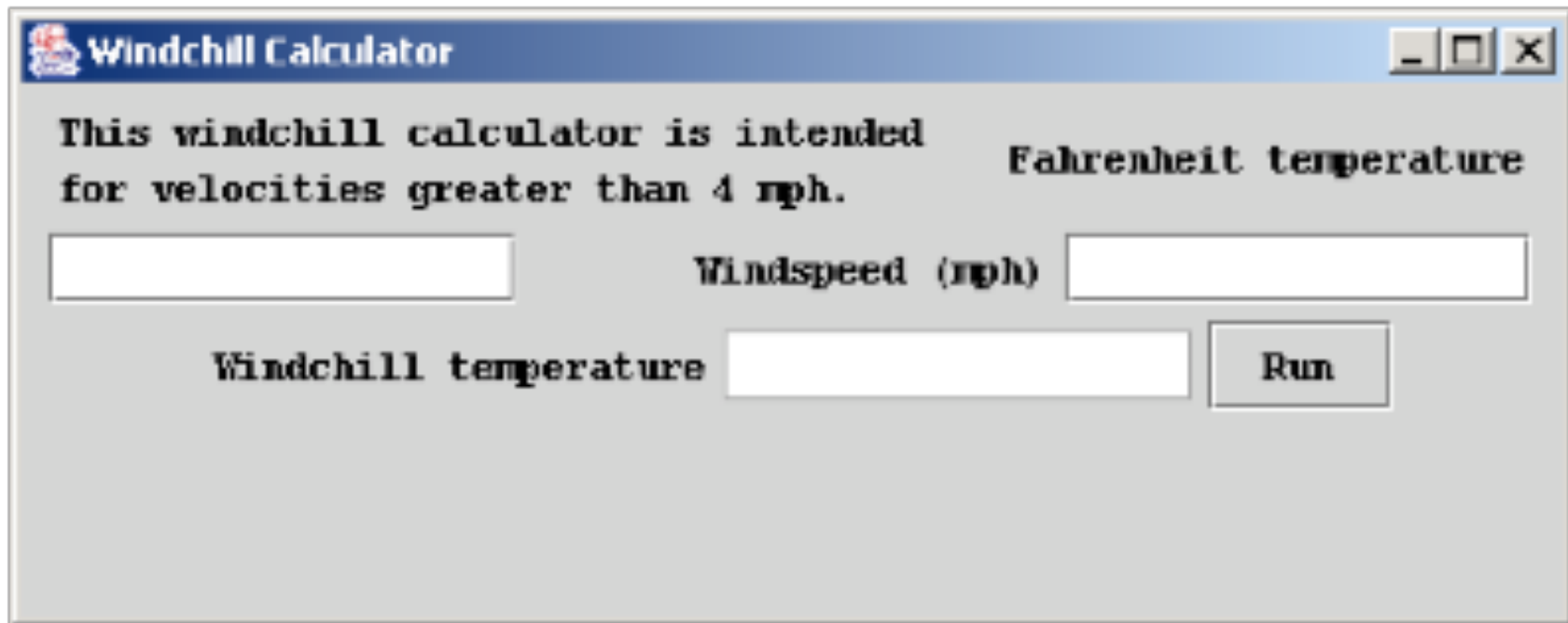
# Class constants

- `private static BorderLayout LAYOUT_STYLE =  
new BorderLayout()`
  - References manager that lays out GUI components in a top-to-bottom, left-to-right manner



# Class constants

- private static `FlowLayout LAYOUT_STYLE = new FlowLayout()`
  - References manager that lays out GUI components in a top-to-bottom, left-to-right manner



# Program Windchill.java

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Windchill implements ActionListener {
    // class constants

    // instance variables with initialization

    // Windchill(): default constructor

    // actionPerformed(): run button action event handler

    // main(): application entry point

}
```

# Program Windchill.java – class constants

```
private static final int WINDOW_WIDTH = 350; // pixels
private static final int WINDOW_HEIGHT = 185; // pixels
private static final int FIELD_WIDTH = 20; // characters
private static final int AREA_WIDTH = 40; // characters
private static final FlowLayout LAYOUT_STYLE =
    new FlowLayout();
private static final String LEGEND = "This windchill "
    + "calculator is intended for velocities greater than 4
    mph.";
```

# Program Windchill.java – instance variables

```
// window for GUI
private JFrame window =
    new JFrame("Windchill Calculator");

// legend
private JTextArea legendArea = new JTextArea(LEGEND, 2,
    AREA_WIDTH);

// user entry area for temperature
private JLabel fahrTag = new JLabel("Fahrenheit temperature");
private JTextField fahrText = new JTextField(FIELD_WIDTH);
```

# Program Windchill.java – instance variables

```
// user entry area for windspeed
private JLabel windTag = new JLabel("    Windspeed (mph)");
private JTextField windText = new JTextField(FIELD_WIDTH);

// entry area for windchill result
private JLabel chillTag =
    new JLabel(" Windchill temperature");

private JTextField chillText = new JTextField(FIELD_WIDTH);

// run button
private JButton runButton = new JButton("Run");
```

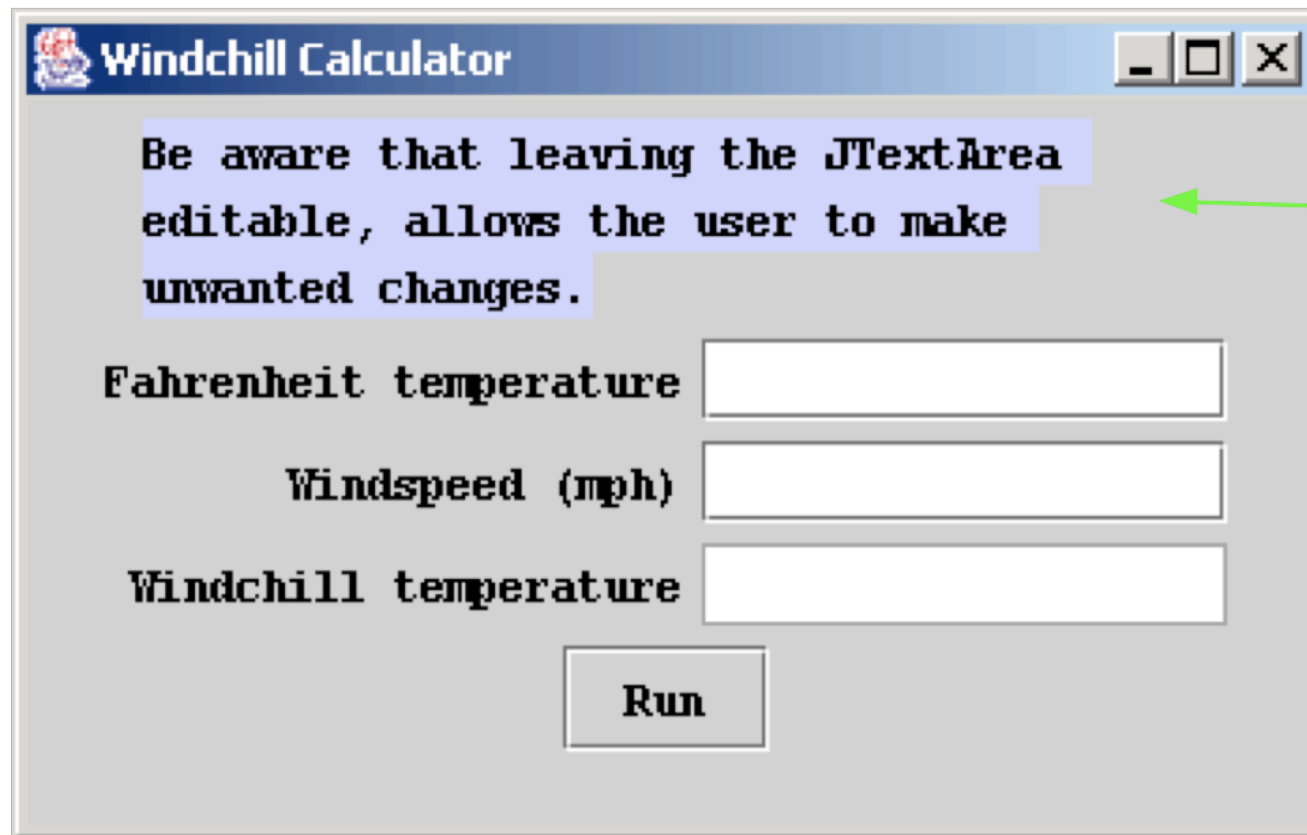
# Program Windchill.java – constructor

```
public Windchill() {  
    // configure GUI  
  
    // register event listener  
  
    // add components to container  
  
    // display GUI  
  
}
```

# Program Windchill.java – constructor

```
public Windchill() {  
    // configure GUI  
    window.setSize(WINDOW_WIDTH, WINDOW_HEIGHT);  
    window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
  
    legendArea.setEditable(false);  
    legendArea.setLineWrap(true);  
    legendArea.setWrapStyleWord(true);  
    legendArea.setBackground(window.getBackground());  
  
    chillText.setEditable(false);  
    chillText.setBackground(Color.WHITE);  
}
```

# Bad line wrapping

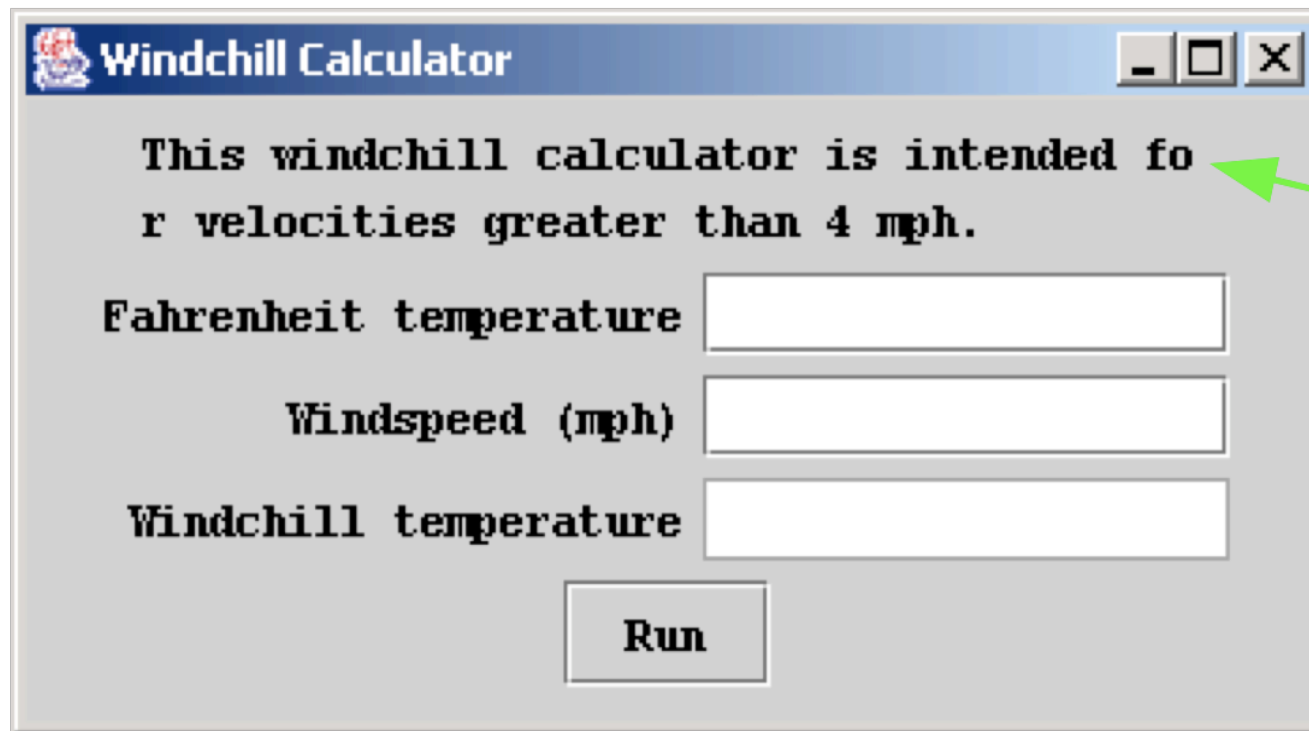


It is important to make program legends uneditable

# Program Windchill.java – constructor

```
public Windchill() {  
    // configure GUI  
    window.setSize(WINDOW_WIDTH, WINDOW_HEIGHT);  
    window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
  
    legendArea.setEditable(false);  
    legendArea.setLineWrap(true);  
    legendArea.setWrapStyleWord(true);  
    legendArea.setBackground(window.getBackground());  
  
    chillText.setEditable(false);  
    chillText.setBackground(Color.WHITE);  
}
```

# Bad Line Wrapping



The screenshot shows a window titled "Windchill Calculator" with a blue title bar and standard window controls. The main content area has a monospaced font. The text "This windchill calculator is intended for velocities greater than 4 mph." is displayed on two lines. The second line starts with "r velocities greater than 4 mph.", where the "r" is the second character of the word "for" from the line above. Below the text are three input fields: "Fahrenheit temperature", "Windspeed (mph)", and "Windchill temperature". A "Run" button is located at the bottom center.

Line wrapping in the middle of a word

# Program Windchill.java – constructor

```
public Windchill() {  
    // configure GUI  
    window.setSize(WINDOW_WIDTH, WINDOW_HEIGHT);  
    window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
  
    legendArea.setEditable(false);  
    legendArea.setLineWrap(true);  
    legendArea.setWrapStyleWord(true);  
    legendArea.setBackground(window.getBackground());  
  
    chillText.setEditable(false);  
    chillText.setBackground(Color.WHITE);  
}
```

Fahrenheit temperature

A JLabel is  
noneditable  
by the user

By default the text field  
of a JTextField is  
editable by the user

# Program Windchill.java – constructor

```
public Windchill() {  
    // configure GUI ...  
  
    // register event listener  
    runButton.addActionListener(this);  
}
```

# Run button action-event handling

Action events are sent to registered action listeners



Action Event

When the run button is clicked, it dispatches an action event

An ActionListener has an actionPerformed() method that handles the class-specific activity

GUI : Action Listener

actionPerformer() Method

- Get data entries from temperature and windspeed data areas
- Compute windchill according to the Weather Service formula
- Display result to windchill data area

# Program Windchill.java – constructor

```
public Windchill() {  
    // configure GUI ...  
  
    // register event listener ...  
  
    // add components to container  
    Container c = window.getContentPane();  
    c.getContentPane().setLayout(LAYOUT_STYLE);  
  
    c.getContentPane().add(legendArea);  
    c.getContentPane().add(fahrTag);  
    c.getContentPane().add(fahrText);  
    c.getContentPane().add(windTag);  
    c.getContentPane().add(windText);  
    c.getContentPane().add(chillTag);  
    c.getContentPane().add(chillText);  
    c.getContentPane().add(runButton);  
}
```

# Program Windchill.java – constructor

```
public Windchill() {  
    // configure GUI ...  
  
    // register event listener ...  
  
    // add components to container ...  
  
    // make GUI visible  
    window.setVisible(true);  
}
```

# Laying out the GUI components

Windchill Calculator

This windchill calculator is intended for velocities greater than 4 mph.

Fahrenheit temperature

Windspeed (mph)

Windchill temperature

Run

Top to  
bottom,  
left to  
right

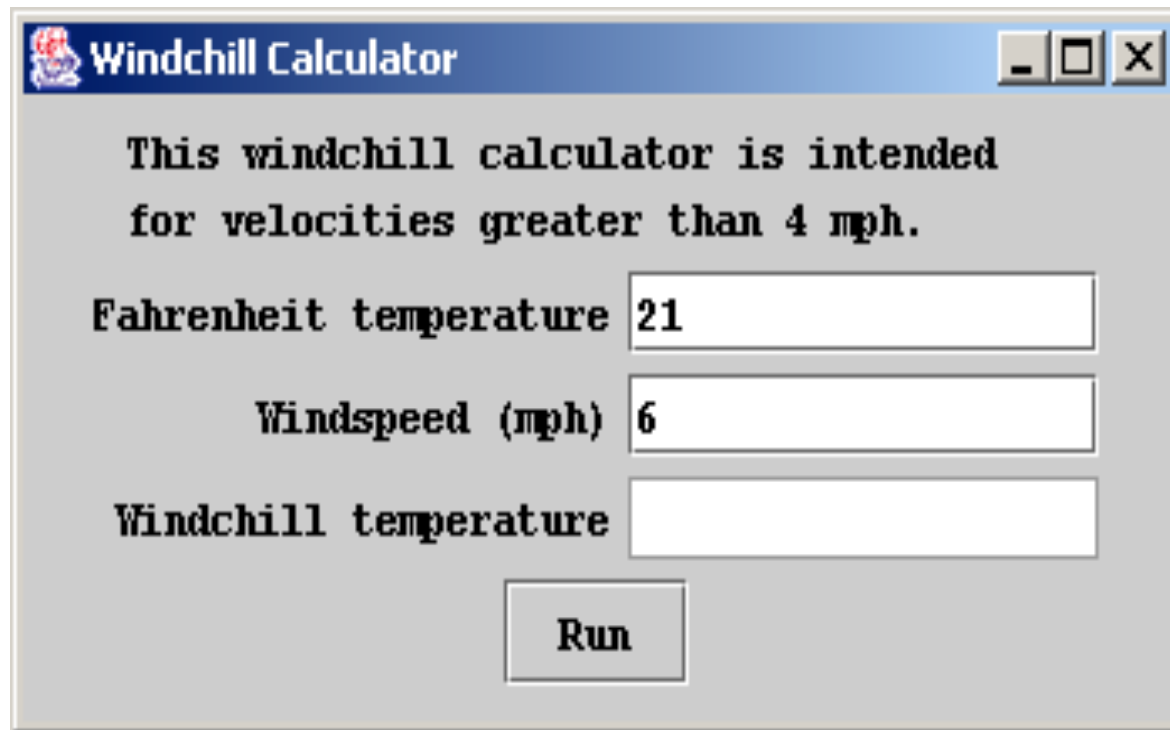
# Program Windchill.java – action performer

```
public void actionPerformed(ActionEvent e) {  
    // get user's responses  
  
    // compute windchill  
  
    // display windchill  
}
```

# Program Windchill.java – action performer

```
public void actionPerformed(ActionEvent e) {  
    // get user's responses  
    String response1 = fahrText.getText();  
    double t = Double.parseDouble(response1);  
    String response2 = windText.getText();  
    double v = Double.parseDouble(response2);  
  
    // compute windchill  
  
    // display windchill  
}
```

# Program Windchill.java – action performer



The screenshot shows a Java Swing window titled "Windchill Calculator". The window has a blue title bar with a small icon on the left and standard window control buttons (minimize, maximize, close) on the right. The main content area is light gray and contains the following text and input fields:

This windchill calculator is intended  
for velocities greater than 4 mph.

Fahrenheit temperature

Windspeed (mph)

Windchill temperature

Run

# Program Windchill.java – action performer

```
public void actionPerformed(ActionEvent e) {  
    // get user's responses  
    String response1 = fahrText.getText();  
    double t = Double.parseDouble(response1);  
    String response2 = windText.getText();  
    double v = Double.parseDouble(response2);  
  
    // compute windchill  
    double windchillTemperature = 0.081 * (t - 91.4)  
        * (3.71*Math.sqrt(v) + 5.81 - 0.25*v) + 91.4;  
  
    int perceivedTemperature =  
        (int) Math.round(windchillTemperature);  
  
    // display windchill  
}
```

# Program Windchill.java – action performer

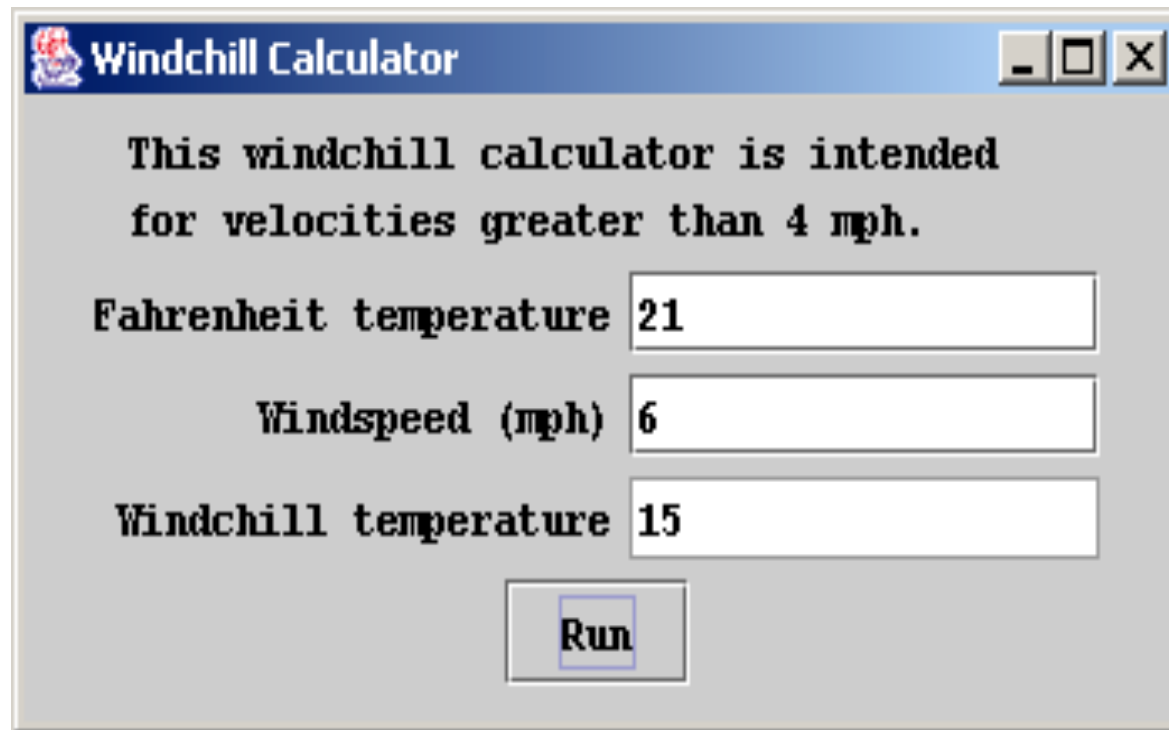
```
public void actionPerformed(ActionEvent e) {
    // get user's responses
    String response1 = fahrText.getText();
    double t = Double.parseDouble(response1);
    String response2 = windText.getText();
    double v = Double.parseDouble(response2);

    // compute windchill
    double windchillTemperature = 0.081 * (t - 91.4)
        * (3.71*Math.sqrt(v) + 5.81 - 0.25*v) + 91.4;

    int perceivedTemperature =
        (int) Math.round(windchillTemperature);

    // display windchill
    String output = String.valueOf(perceivedTemperature);
    chillText.setText(output);
}
```

# Program Windchill.java – action performer



**Windchill Calculator**

This windchill calculator is intended  
for velocities greater than 4 mph.

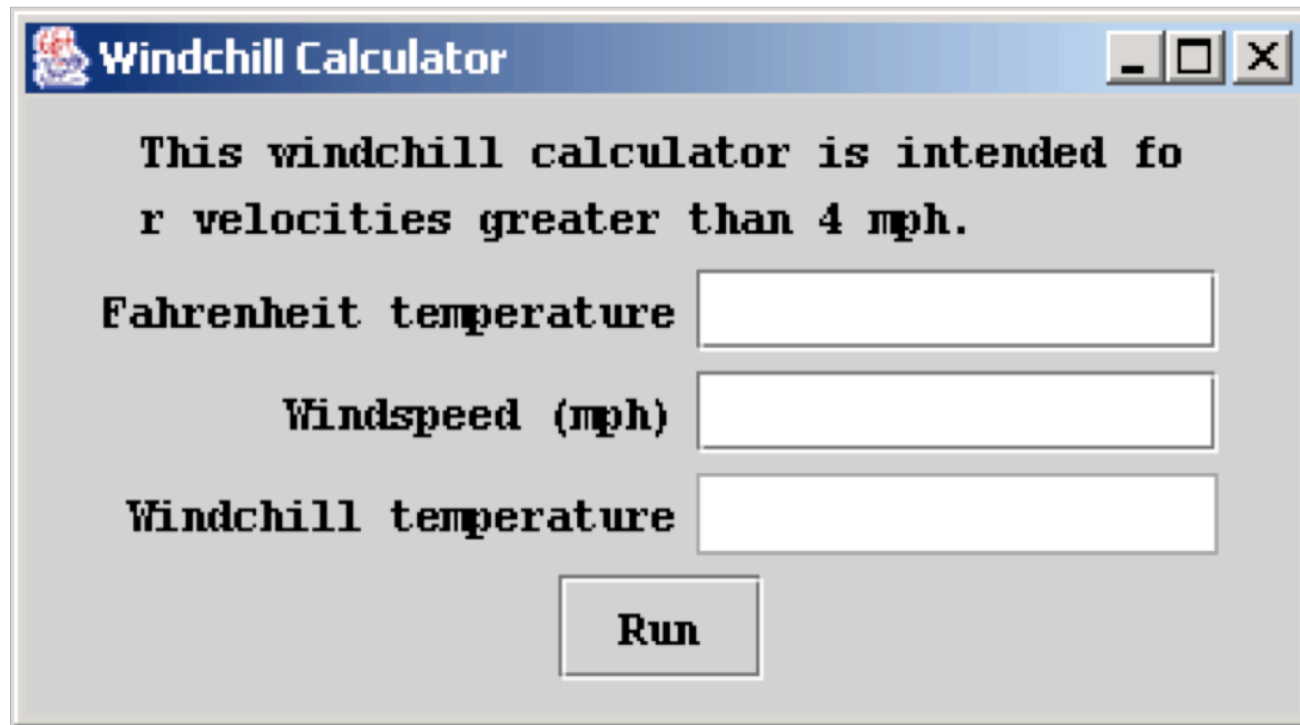
Fahrenheit temperature

Windspeed (mph)

Windchill temperature

# Method main()

```
public static void main(String[] args) {  
    Windchill gui = new Windchill();  
}
```



# Another method main()

```
public static void main(String[] args) {  
    Windchill gui1 = new Windchill();  
    Windchill gui2 = new Windchill();  
}
```



**Windchill Calculator**

This windchill calculator is intended  
for velocities greater than 4 mph.

Fahrenheit temperature

Windspeed (mph)

Windchill temperature

Run

**Windchill Calculator**

This windchill calculator is intended  
for velocities greater than 4 mph.

Fahrenheit temperature

Windspeed (mph)

Windchill temperature

Run

# Applets

- Java program run within a browser
  - Implies an applet is run from a web page
- Applets may not access or modify the file system running the applet
- A modern applet has JApplet as its superclass
  - JApplet is part of the swing package
- An applet uses JFrame
  - A JApplet has a content pane

# Applets

- Important inherited methods
  - init()
    - Run when browser loads applet
  - start()
    - Run by browser to start applet execution
  - stop()
    - Run by browser to stop its execution
  - destroy()
    - Run by browser immediately before it its ended
  - paint(Graphics g)
    - Run by browser to refresh its GUI
- By default the inherited methods do nothing

# A simple applet

```
import java.awt.*;
import javax.swing.*;
public class DisplayQuote extends JApplet {
    public void paint(Graphics g) {
        g.drawString("Anyone who spends their life on a "
            + " computer is pretty unusual.", 20, 30);
        g.drawString("Bill Gates, Chairman of Microsoft", 25, 45);
    }
}
```

# Web page – quote.htm

```
<html>  
  <title> Quotation </title>  
  <applet code="DisplayQuote.class" width=400 height=300>  
  </applet>  
</html>
```