```java
package ca.camosun.lab6converter;

import java.util.Hashtable;

// A Conversion that can be performed.  Contains two buttons to provide conversions between two
// different units of measurement in both directions
public class Conversion {
    // the name to display for the conversion
    private String name;
    // the left button
    private ConversionButton leftButton;
    // the right button
    private ConversionButton rightButton;

    // Initializes the Conversion with its name and its two buttons to perfom unit conversions.
    public Conversion(String name, ConversionButton leftButton, ConversionButton rightButton){
        this.name = name;
        this.leftButton = leftButton;
        this.rightButton = rightButton;
    }

    // Public accessor for name
    public String getName(){
        return this.name;
    }

    // Public accessor for LeftButton
    public ConversionButton getLeftButton(){
        return this.leftButton;
    }

    // Public accessor for RightButton
    public ConversionButton getRightButton(){
        return this.rightButton;
    }

    // Helper method to generate all available conversions.  To add a conversion, simply instantiate
    // a new Conversion object and append to the Hashtable.
    public static Hashtable<String, Conversion> generateConversions(){

        // Initialize all the conversions
        Conversion areaConversion = new Conversion("Area",
                new ConversionButton("ha to ac", (Double value) -> { return value * 2.471; }),
                new ConversionButton("ac to ha", (Double value) -> { return value * 0.405; }));

        Conversion tempConversion = new Conversion("Temperature",
                new ConversionButton("C to F", (Double value) -> { return value * 9.0 / 5.0 + 32.0; }),
                new ConversionButton("F to C", (Double value) -> { return (value - 32.0) * 5.0 / 9.0; }));

        Conversion lengthConversion = new Conversion("Length",
                new ConversionButton("ft to m", (Double value) -> { return value * 0.305; }),
                new ConversionButton("m to ft", (Double value) -> {
```

```
52 return value * 3.281; }));
53
54         Conversion weightConversion = new Conversion("Weight",
55                 new ConversionButton("lbs to kg", (Double value) -> {
    return value * 0.454; }),
56                 new ConversionButton("kg to lbs", (Double value) -> {
    return value * 2.205; }));
57
58         // Store all conversions together
59         Hashtable<String, Conversion> conversions = new Hashtable<>()
   ;
60         conversions.put(areaConversion.name, areaConversion);
61         conversions.put(tempConversion.name, tempConversion);
62         conversions.put(lengthConversion.name, lengthConversion);
63         conversions.put(weightConversion.name, weightConversion);
64
65         return conversions;
66     }
67 }
```

```java
package ca.camosun.lab6converter;

import android.app.Activity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;

import java.util.ArrayList;
import java.util.Hashtable;
import java.util.List;

public class MainActivity extends Activity implements AdapterView.OnItemSelectedListener {
    // Holds all the available conversions
    private Hashtable<String, Conversion> conversions;
    // the left button
    private Button leftButton;
    // the right button
    private Button rightButton;
    // the current conversion
    private Conversion selectedConversion;

    // An item was selected in the Spinner.  Detect choice and update UI and instance variables.
    public void onItemSelected(AdapterView<?> parent, View view,
                               int pos, long id) {
        if(conversions == null){
            // conversions was not instantiated properly
            // log and abort
            Log.e("onItemSelected", "Conversions is null.");
            return;
        }

        if(conversions.isEmpty()){
            // there are no conversions to use
            // log and abort
            Log.e("onItemSelected", "Conversions is empty.");
            return;
        }

        // Get selected item and update the current conversion instance
        String selectedConversionName = parent.getItemAtPosition(pos).toString();
        selectedConversion = conversions.get(selectedConversionName);

        if(selectedConversion == null){
            // selectedConversion was not set up properly
            // log and abort
            Log.e("onItemSelected", "Desired conversion was not found.");
            return;
        }

        if(leftButton == null){
            // left button was not set up properly
```

```
 57             // log and abort
 58             Log.e("onItemSelected", "Left button was not set");
 59             return;
 60         }
 61
 62         if(rightButton == null){
 63             // right button was not set up properly
 64             // log and abort
 65             Log.e("onItemSelected", "Right button was not set");
 66             return;
 67         }
 68
 69         // update each button's text
 70         leftButton.setText(selectedConversion.getLeftButton().getName
    ());
 71         rightButton.setText(selectedConversion.getRightButton().
    getName());
 72
 73     }
 74
 75     // When nothing is selected.  OnItemSelectedListener requires us
    to implement this
 76     public void onNothingSelected(AdapterView<?> parent) {
 77         // Do nothing
 78         return;
 79     }
 80
 81
 82     // Generates the conversions instance, sets up the spinner and
    grabs references to the buttons.
 83     @Override
 84     protected void onCreate(Bundle savedInstanceState) {
 85         super.onCreate(savedInstanceState);
 86         setContentView(R.layout.activity_main);
 87
 88         // Call helper method to generate all conversions
 89         // Store all these conversions for later reference
 90         conversions = Conversion.generateConversions();
 91
 92         // Get a list of all conversion names for the spinner to use
 93         List<String> conversionNames = new ArrayList<>();
 94         for(String key : conversions.keySet()){
 95             conversionNames.add(key);
 96         }
 97
 98         // Setup the spinner with conversion names
 99         Spinner conversionsSpinner = (Spinner) findViewById(R.id.
    conversionsSpinner);
100         ArrayAdapter<String> adapter = new ArrayAdapter<String>(
101                 this,
102                 R.layout.support_simple_spinner_dropdown_item,
103                 conversionNames);
104
105         conversionsSpinner.setAdapter(adapter);
106
107         // Ensure spinner calls us back on item selection
108         conversionsSpinner.setOnItemSelectedListener(this);
109
110         // Grab references for later use by the spinner
111         leftButton = (Button) findViewById(R.id.leftButton);
112         rightButton = (Button) findViewById(R.id.rightButton);
```

```
113     }
114
115     // Called when the left button was clicked.  Converts using the
    currently selected conversion
116     // from the conversion spinner.
117     public void leftButton(View view){
118         if(selectedConversion == null){
119             // selected conversion was never set properly
120             // log and abort
121             Log.e("leftButton", "selectedConversion was not set
    properly");
122             return;
123         }
124
125         // Call helper method
126         convertValue(selectedConversion.getLeftButton().getAction());
127     }
128
129     // Called when the right button was clicked.  Converts using the
    currently selected conversion
130     // from the conversion spinner.
131     public void rightButton(View view){
132         if(selectedConversion == null){
133             // selected conversion was never set properly
134             // log and abort
135             Log.e("rightButton", "selectedConversion was not set
    properly");
136             return;
137         }
138
139         // Call helper method
140         convertValue(selectedConversion.getRightButton().getAction())
    ;
141     }
142
143     // Converts the user value using the passed action.  The action
    is a lambda expression as per
144     // the PerformsConversion interface.
145     private void convertValue(PerformsConversion action){
146         // Grab the user variable
147         EditText converterField = (EditText) findViewById(R.id.
    userVariable);
148
149         try {
150             // Throws if Null or no Double found
151             double temp = Double.parseDouble(converterField.getText()
    .toString());
152
153             // Convert the user variable and output the result
154             double convertedTemp = action.convert(temp);
155             converterField.setText(Double.toString(convertedTemp));
156         } catch (NullPointerException|NumberFormatException ex){
157             // Failed to convert to a double - the value either
    contained no double or was empty
158             converterField.setText("N/A");
159         }
160     }
161 }
162
```

```java
package ca.camosun.lab6converter;

// A button that belongs to a Conversion
public class ConversionButton{
    // the text label to display for the button
    private String name;
    // the lambda expression to call on button click
    private PerformsConversion action;

    // Initializes the ConversionButton with the name and action.  The action must conform to
    // the PerformsConversion interface.  The action is a lambda expression for use later.
    public ConversionButton(String buttonName, PerformsConversion action){
        this.name = buttonName;
        this.action = action;
    }

    // Public accessor for name
    public String getName(){
        return this.name;
    }

    // Public accessor for action
    public PerformsConversion getAction(){
        return this.action;
    }
}

```

```java
package ca.camosun.lab6converter;

// Interface for all conversion actions to conform to
// All conversion actions must convert an input Double to an output Double
public interface PerformsConversion {
    Double convert(Double value);
}
```