Henry Post

ITMD 455

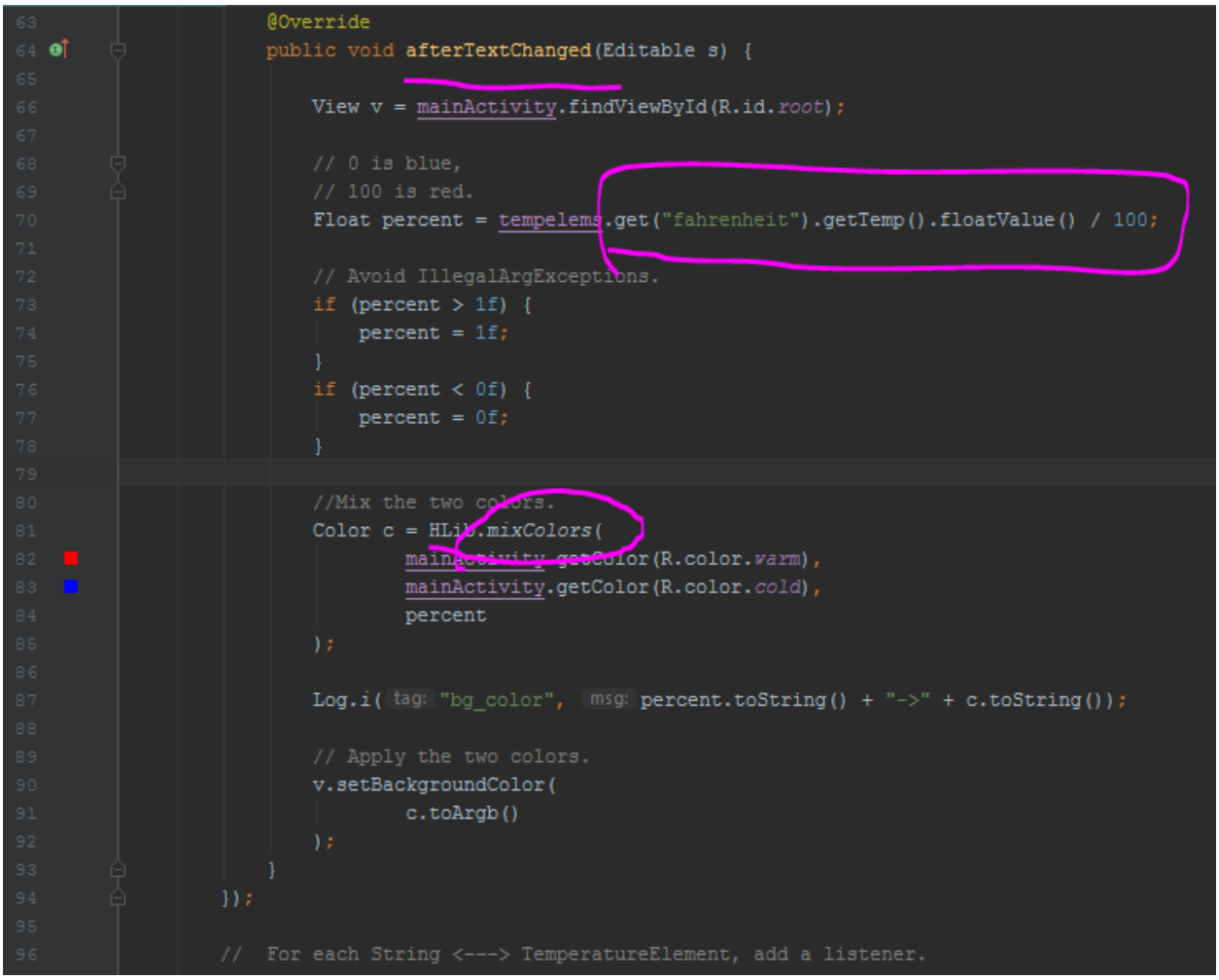
Lab 1: Temperature Converter

Contents

Color gradients and unit auto-conversion:

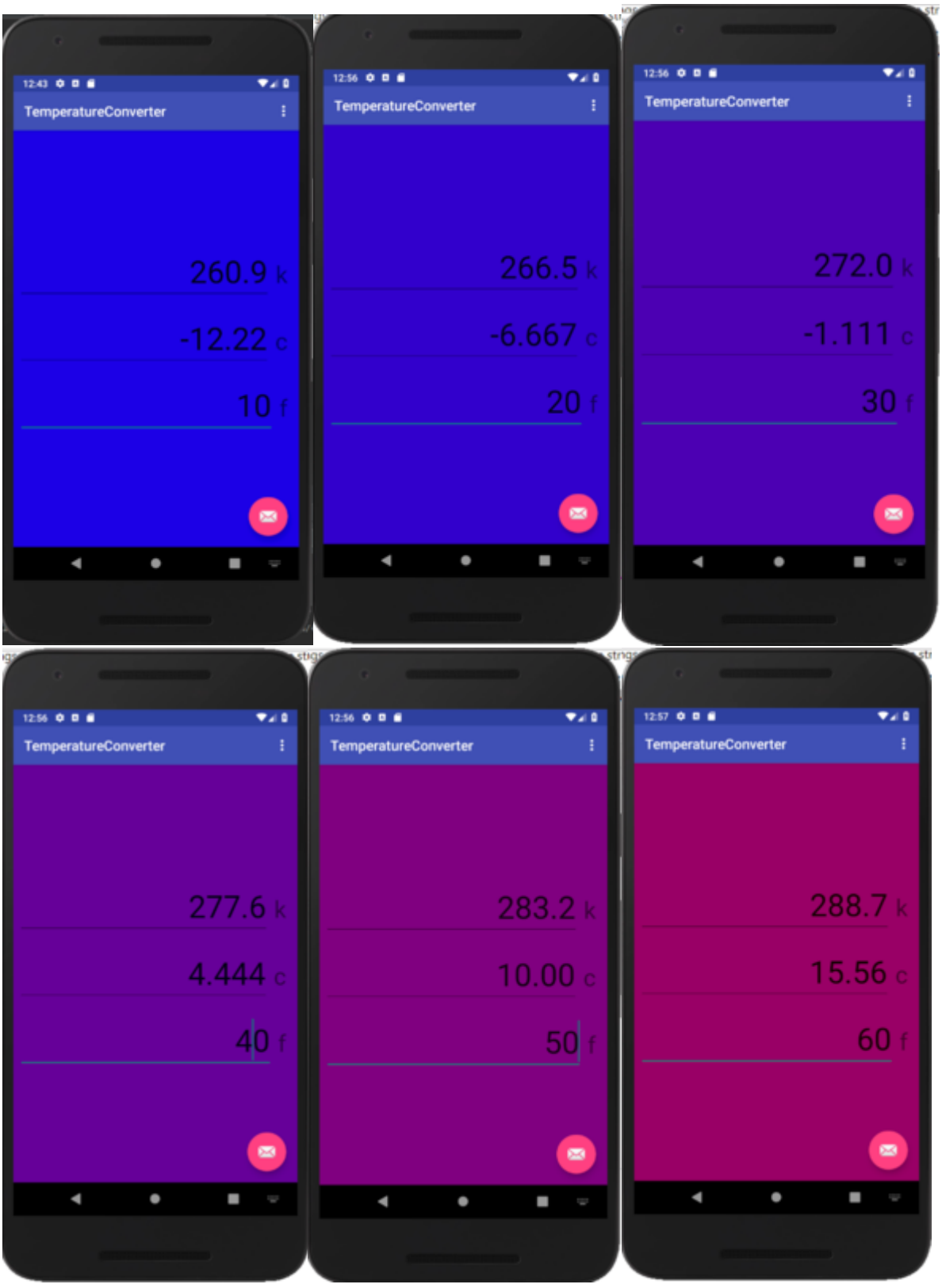
I designed my application to convert the units whenever any textbox changed, and It accomplishes this through an `OnTextChanged` listener.

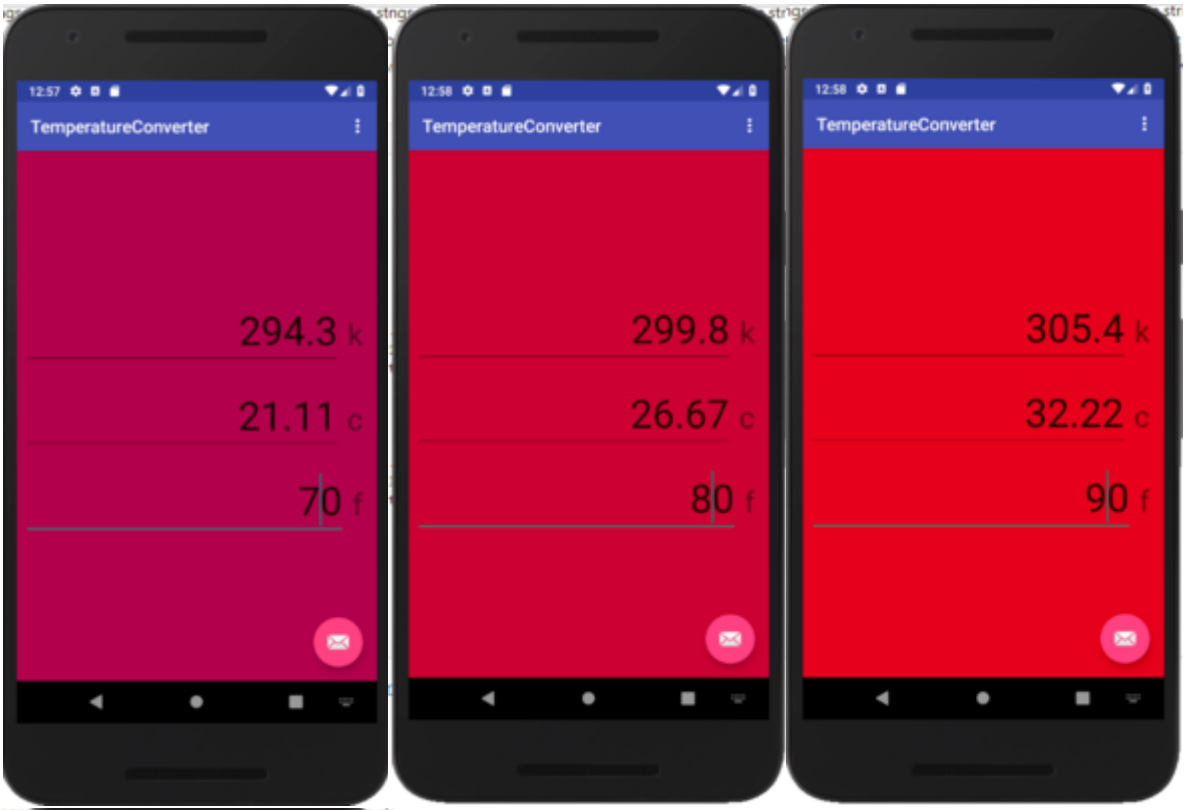
I also made the colors *interpolate* between two values, #FF0000 and #0000FF.

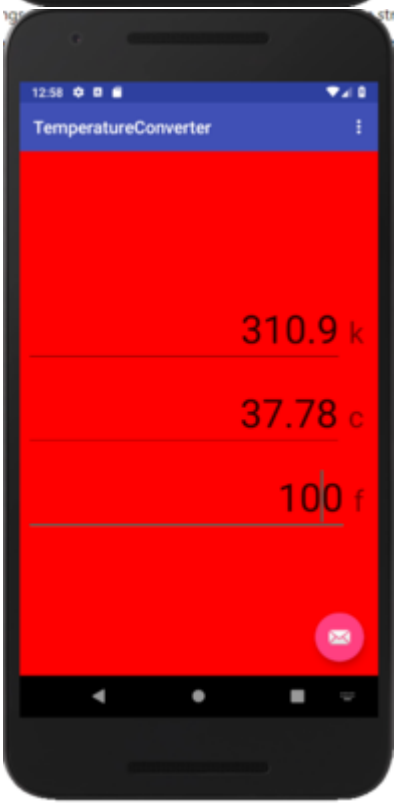


```
63          @Override
64          public void afterTextChanged(Editable s) {
65
66              View v = mainActivity.findViewById(R.id.root);
67
68              // 0 is blue,
69              // 100 is red.
70              Float percent = tempelems.get("fahrenheit").getTemp().floatValue() / 100;
71
72              // Avoid IllegalArgExceptions.
73              if (percent > 1f) {
74                  percent = 1f;
75              }
76              if (percent < 0f) {
77                  percent = 0f;
78              }
79
80              //Mix the two colors.
81              Color c = HLib.mixColors(
82                      mainActivity.getColor(R.color.warm),
83                      mainActivity.getColor(R.color.cold),
84                      percent
85              );
86
87              Log.i( tag: "bg_color",  msg: percent.toString() + "->" + c.toString());
88
89              // Apply the two colors.
90              v.setBackgroundColor(
91                      c.toArgb()
92              );
93          }
94      });
95
96      //  For each String <---> TemperatureElement, add a listener.
```

As you can see here, I convert the fahrenheit value to a float, divide it by 100, and then use it to scale how intensely I mix the ‘warm’ and ‘cold’ colors.


12:43
TemperatureConverter
260.9 k
-12.22 c
10 f
12:56
TemperatureConverter
266.5 k
-6.667 c
20 f
12:56
TemperatureConverter
272.0 k
-1.111 c
30 f
12:56
TemperatureConverter
277.6 k
4.444 c
40 f
12:56
TemperatureConverter
283.2 k
10.00 c
50 f
12:57
TemperatureConverter
288.7 k
15.56 c
60 f

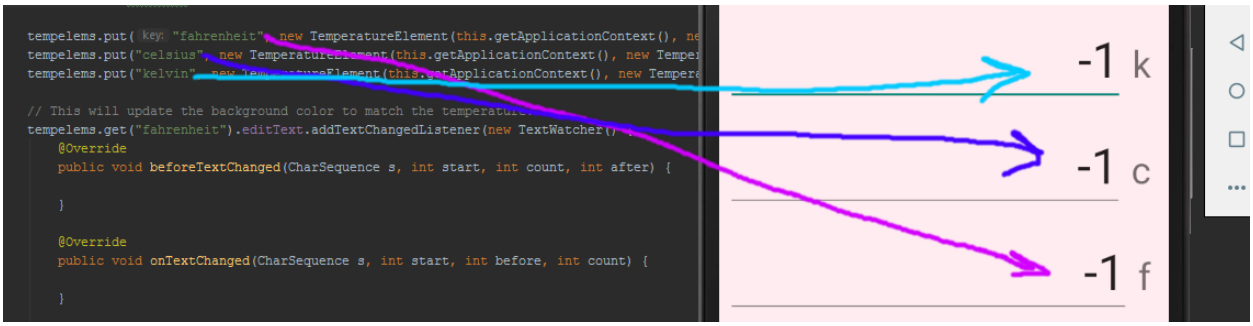

12:57
TemperatureConverter
294.3 k
21.11 c
70 f
12:58
TemperatureConverter
299.8 k
26.67 c
80 f
12:58
TemperatureConverter
305.4 k
32.22 c
90 f


12:58
TemperatureConverter
310.9 k
37.78 c
100 f

Custom LinearLayout subclass, ‘TemperatureElement’

I wrote a custom LinearLayout subclass called a TemperatureElement that stores extra metadata like a removable TextWatcher, a TemperatureUnit, and has some convenience text-to-BigDecimal functions.

It also allows inflating layouts to GUI elements with a SINGLE LINE OF CODE!



As you can see, it takes little effort to populate the screen.

They draw their style from a single, 20-line XML file.

HashMap-backed formula solving system

You may notice that I have absolutely no hard-coded logic for turning one unit into another.

That is because I define them all inside of `TemperatureSolver` and `TemperatureSolverSingle`!

I use the `MXParser` library to turn strings into algebraic expressions.

```
/***
 * Solve any temperature.
 */
public class TemperatureSolver {

    public Map<ImmutableList, TemperatureSolverSingle> map = new HashMap<>();

    public TemperatureSolver() { //TODO: Use linear algebra to automatically fill-in
missing formulas.
        this.addTemp(
      new TemperatureSolverSingle("celsius", "fahrenheit", "F(x) = ((9/5) * x) + 32"),
      new TemperatureSolverSingle("celsius", "kelvin", "F(x) = x + 273.15"),
      new TemperatureSolverSingle("fahrenheit", "kelvin", "F(x) = (x + 459.67) *
(5/9)"),
      new TemperatureSolverSingle("fahrenheit", "celsius", "F(x) = (x - 32) * (5/9)"),
      new TemperatureSolverSingle("kelvin", "celsius", "F(x) = x - 273.15"),
      new TemperatureSolverSingle("kelvin", "fahrenheit", "F(x) = ((9/5) * x) -
459.67")
        );
    (…)}
}
```