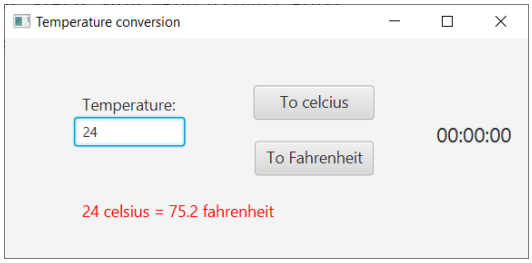
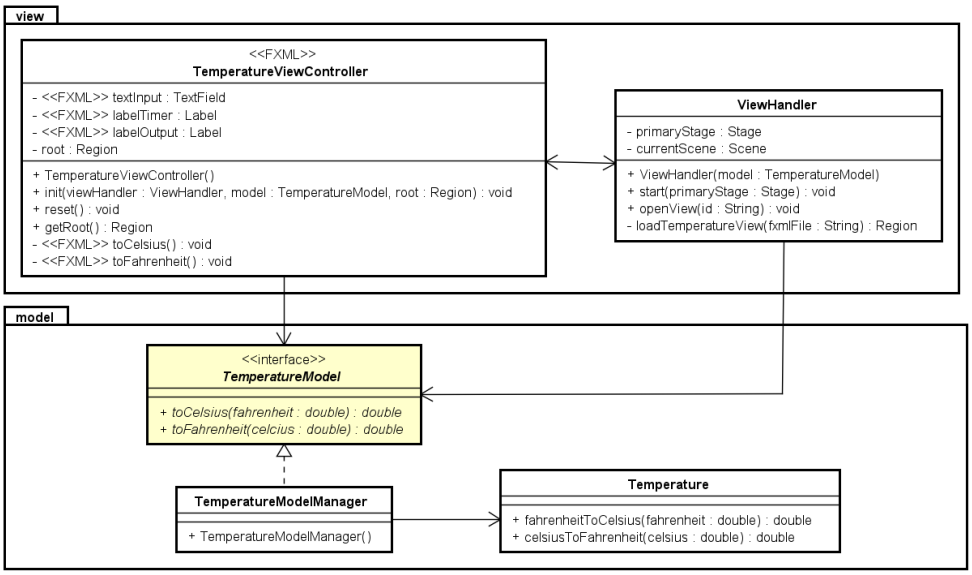
A JavaFX temperature Converter with a Clock (see below)

The uploaded file consists of a Java application including a GUI implemented in JavaFX.

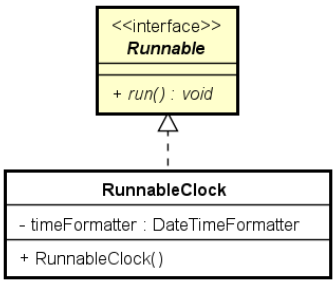


When you run the program, the follow window is shown (Note: the label shows 00:00:00 and not the current time):



Part 1 – A Clock thread

Implement a class `RunnableClock` simulating a clock (in a package `external`). In the `run` method print out the current time every second. Make an infinite loop in which you print out the current time in the format: HH:mm:ss and pause 1000 milliseconds.



Hint: The current time can be found this way

```
LocalTime time = LocalTime.now();
```

and converted to format HH:mm:ss this way:

```
DateTimeFormatter timeFormatter =
DateTimeFormatter.ofPattern("HH:mm:ss");
String timeString = time.format(timeFormatter);
```

Alternatively, you may use the `Clock` class you implemented last semester - instead of a `LocalTime`.

Make a test class with a `main` method in which you create a `Thread` with a `RunnableClock` object reference as argument, and start the thread.

Part 2 on the next page…

Part 2 – Combining the GUI and the Clock

In this part of the exercise, you update the GUI such that the timer label always show the current time (and is updated every second)

Step 1: Update class `TemperatureViewController`

- Add a method to be called from the `RunnableClock` object

  ```
  public void showTime(String timeString)
  ```

  In this method, you update the timer label (set the text of the label). Note that you have to wrap this statement in a JavaFX-thread (a `Platform.runLater`) because an extern thread are not allowed to update a JavaFX component. The syntax may be like:

  ```
  Platform.runLater(() -> labelTimer.setText(timeString));
  ```

Step 2: Update class `RunnableClock`

- Create an instance variable of the type `TemperatureViewController` and initialize it to a parameter to the constructor.
- After printing out to the console (in the `run` method), you call the method `showTime` in the `TemperatureViewController` (do not delete the print-out yet)

Step 3: Update class `TemperatureViewController`

- In the `init` method, create the `RunnableClock` (with `this` as the argument), create a `Thread` with the `RunnableClock` object as argument, and start the thread.

Step 4: Run the `main` method in class `Main` and observe the result.

Step 5 (Update to step 3): You may have noticed that the time is still printed to the console after the GUI is terminated, i.e. the clock thread is not terminated. Therefore, in the `init` method, mark the thread as a daemon thread (calling `Thread` method `setDaemon(true)` before you start the thread). Run again and observe the result.