Lab 7 – GUI development and implementation II

The purpose of this lab is to get more familiar to GUI – Graphical User Interface. You will extend your knowledge and get familiar with different components.

1.1 GUI programming

- Some predefined **ViewGroups**
  - RadioGroup
  - TimePicker
  - DatePicker
  - WebView
  - MapView
  - Gallery
  - Spinner

- **RadioGroup**

  A View group containing a set of Radio Buttons, only one button can be selected at any one instant

```
<RadioGroup
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    >

    <RadioButton
        android:id="@+id/choice1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/choice_1_string"
        android:textSize="24sp"/>

    <RadioButton
        android:id="@+id/choice2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/choice_2_string"
        android:textSize="24sp"/>

    <RadioButton
        android:id="@+id/choice3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/choice_3_string"
        android:textSize="24sp" />
</RadioGroup>
```

1. In the **onCreate()** method in the **MainActivity.java** define a generic listener for all of the RadioButtons in the RadioGroup that was just formed.

2. Define the **TextView** so the text “Choice x chosen” will appear, don’t forget to write the code in the .**xml** as well
3. In the **onClick()** with argument **View v,** method define the **RadioButton** and set the text to the **TextView.**
4. Call the Listener
   Final RadioButton choice1 = (RadioButton) findViewById(R.id.choice1);
   Choice1.setOnClickListener(radioListener);
   Do this for all of the radioButtons accordingly.

1.2 Extra

- **TimePicker -** a ViewGroup that allows the user to select a time


```
<TextView
    android:id="@+id/timeDisplay"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=""
    android:textSize="24sp" />

<Button
    android:id="@+id/pickTime"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/change_the_time_string"
    android:textSize="24sp" />
```

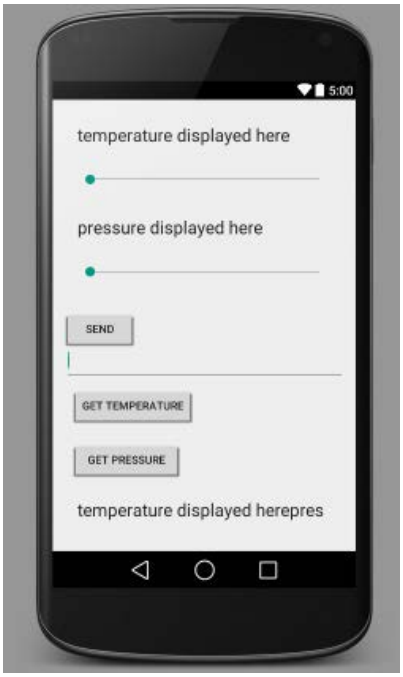
```
// The callback received when the user "sets" the time in the dialog
private TimePickerDialog.OnTimeSetListener mTimeSetListener = new TimePickerDialog.OnTimeSetListener() {
    public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
        mHour = hourOfDay;
        mMinute = minute;
        updateDisplay();
    }
};
```

```
@Override
protected Dialog onCreateDialog(int id) {
    switch (id) {
    case TIME_DIALOG_ID:
        return new TimePickerDialog(this, mTimeSetListener, mHour, mMinute,
                false);
    }
    return null;
}
```

- **UIDatePicker**

The DatePicker work in a very similar way as the TimePicker. All you need to do are small changes like instead of “Time” you should write “Date”

1.3 Water Tank Controller



New components are supposed to be added in the program.
Since the app is supposed to send the temperature and the pressure values using Wi-Fi, which is done using Sockets, it is user-friendly if the server informs the client when the message is received.
Think of another possible implementation and design.

The two buttons **Get Temperature** and **Get Pressure** will be used in order to get the Temperature and Pressure from the distance Controller.
Even though not visible on the figure, there is also a **TextView** under the sliders where a confirmation message from the server about the values of the temperature and pressure are registered there.
The two **Textviews** under the buttons are for the pressure and temperature accordingly.

References:

https://class.coursera.org/androidpart1-003/lecture