Lab 3: Building User Interfaces for Activities

Introduction

In this lab, you'll learn how to create user interfaces for Android applications.

Objectives

- Get to know common layouts and views.
- Be able to create layouts in XML.
- Load the layouts for activities.
- Be able to define and use menus.
- Be able to create and show dialogs.
- Have an understanding and use of notifications.
- Understanding the concept of action bar.

Readings:

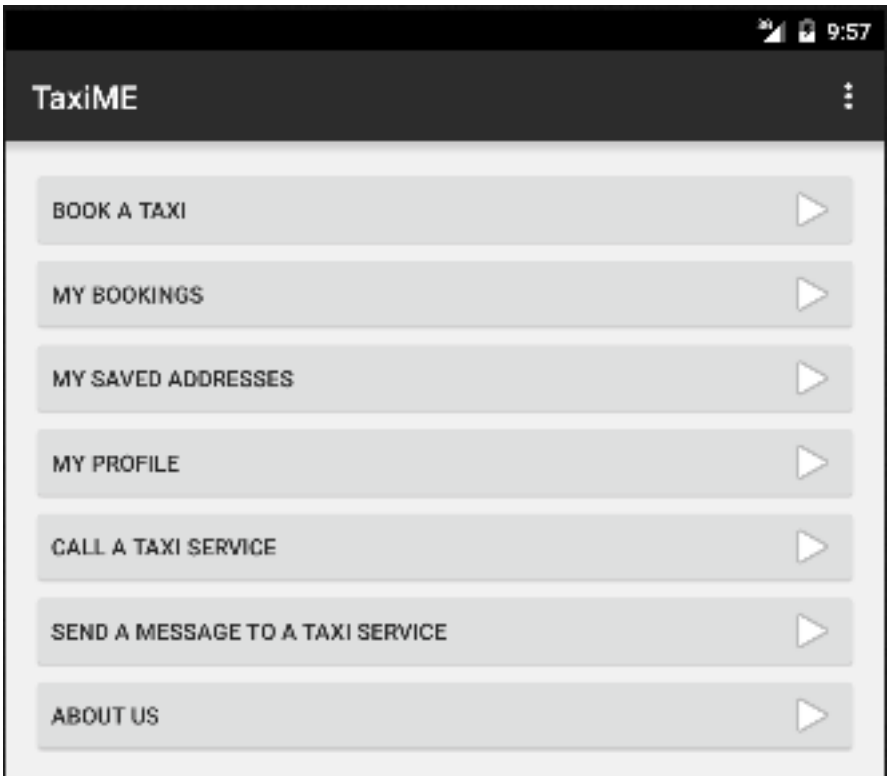
- The Power Point Slides: “Building User Interfaces for Activities”.
- http://developer.android.com/guide/index.html

Lab Exercises

In this lab, you will create a taxi booking app named "**TaxiME**" that allows the user to book a taxi.

Task 1:

Create the first activity named HomeActivity that has the following user interface:



Requirements:

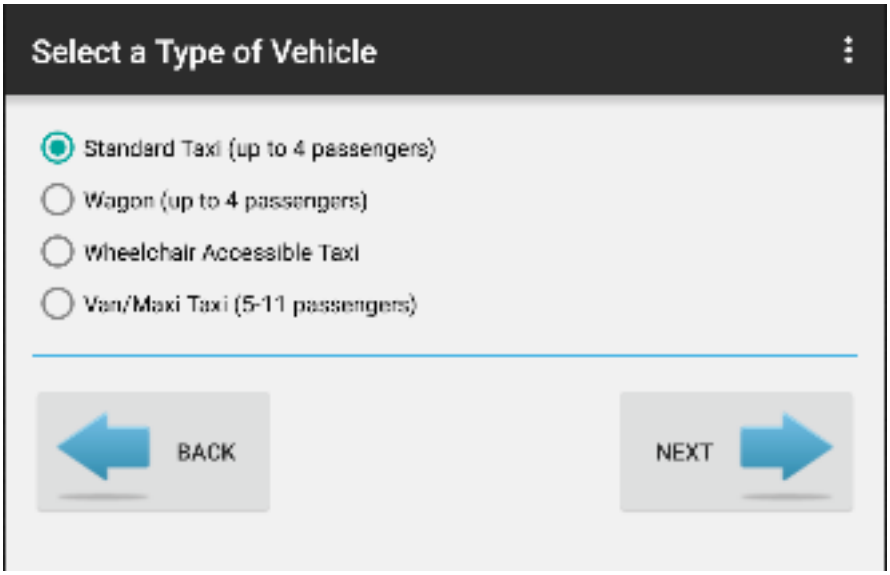
1. There is a padding space between the buttons and the borders of the screen. Hint: Use `android:padding="16dp"` for the parent layout.

2. Each button has an icon (of your choice) on the right. For simplicity, you may use a pre-icon provided by Android such as `@android:drawable/ic_media_play`.

3. The text on each button must be aligned to the left and centered vertically. Hint: use `android:gravity="left|center_vertical"`.

Task 2:

Create the second activity named SelectVehicleActivity that has the following user interface:



Requirements:

1. The Back and Next buttons have icons (of your choice) (or download icons.zip).
2. There is a horizontal rule (line) between the radio group and the buttons. Hint:

```
<View android:layout_width="fill_parent"
      android:layout_height="1dp"
      android:layout_marginBottom="4dp"
      android:layout_marginTop="16dp"
      android:background="#33A1DE" />
```

3. The Back button is left aligned while the Next button is right aligned. Hint: Use a linear layout as a root layout, place a relative layout within this root layout which holds the two buttons like the following snippet:

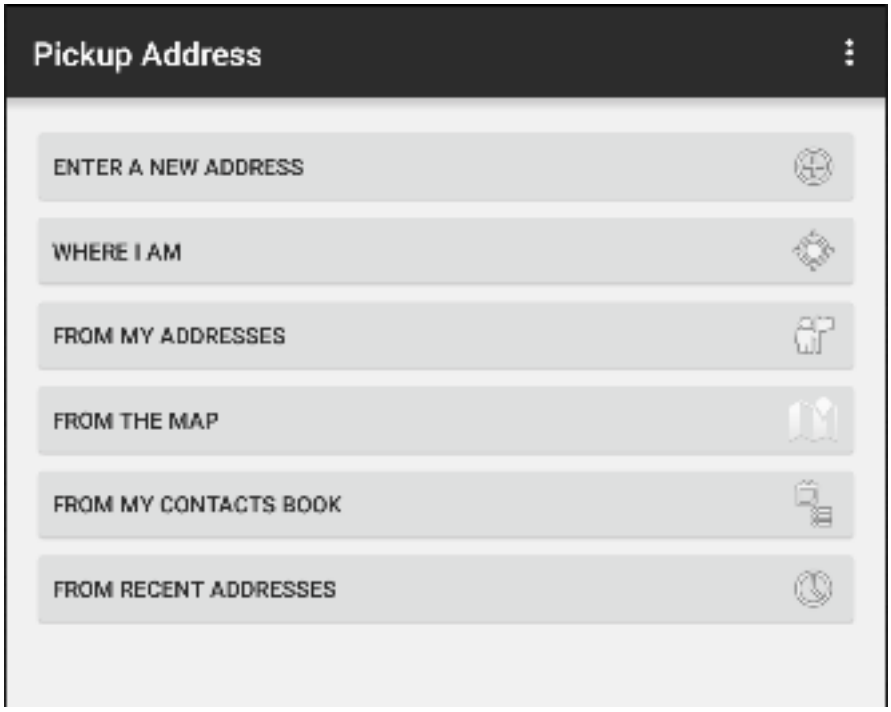
```
<RelativeLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal" >

    <Button
        android:id="@+id/btn_back"
        android:layout_width="110dp"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:drawableLeft="@drawable/back_icon"
        android:text="@string/back" />
    <Button
        android:id="@+id/btn_next"
        android:layout_width="110dp"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:drawableRight="@drawable/next_icon"
        android:text="@string/next" />
</RelativeLayout>
```

Task 3:

Make necessary changes in both the XML layout and the code of the HomeActivity activity to handle the click event of the "Book a Taxi" button. When the user clicks on the "Book a Taxi" button, start the SelectVehicleActivity activity.

Task 4:

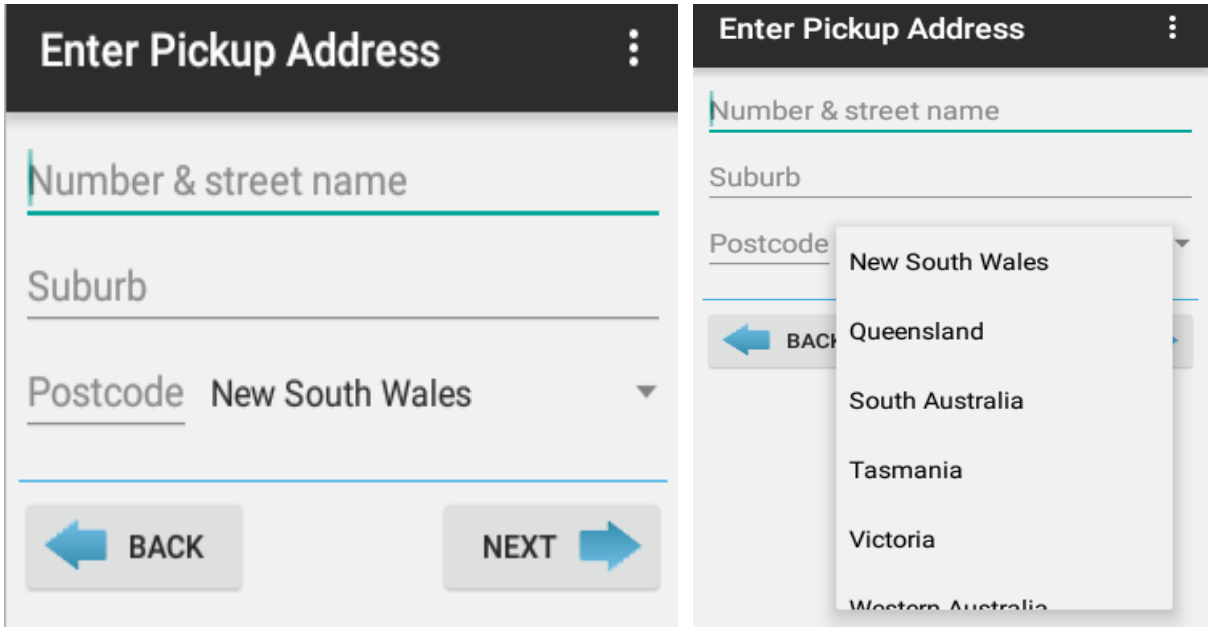
1. Create an activity named PickupAddressActivity that allows a user to choose how he/she'd like enter a pickup address.



2. Handle the click event of the "Next" button on the SelectVehicleActivity activity, which starts the PickupAddressActivity.

Task 5:

1. Create an activity named EnterPickupAddressActivity that allows a user to enter a pickup address.



Requirements:

- The edit text for postcode must only accept no more than 4 digits.
- Use a spinner for specifying state as can be seen on the figure above. Hint: Define a string array for all states in the strings.xml file, for example:

```
<string-array name="states_array">
        <item>New South Wales</item>
        <item>Queensland</item>
        <item>South Australia</item>
        <item>Tasmania</item>
        <item>Victoria </item>
        <item>Western Australia</item>
</string-array>
```

Then use this array for the spinner, for example:

```
<Spinner
        android:id="@+id/edit_state"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:entries="@array/states_array" />
```

- The edit text for postcode and the spinner for state must be on the same line. The size of the postcode field is "wrap_content", while the size of the spinner uses the remaining space. Hint:

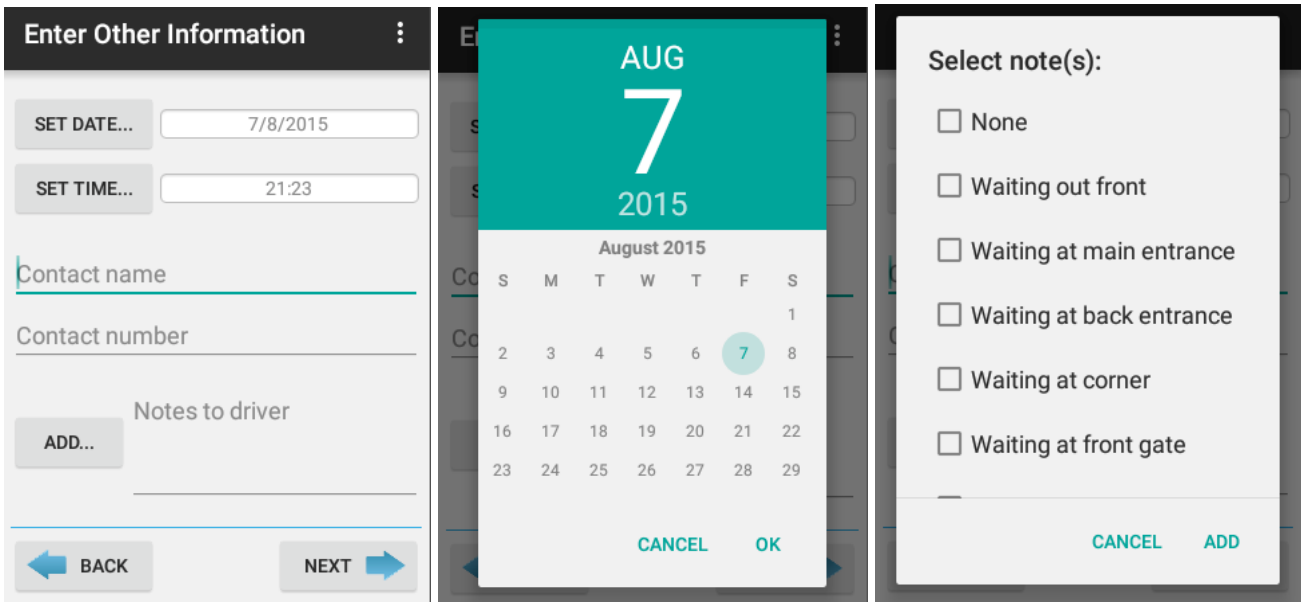
```
<LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal" >
```

```
        <EditText
            android:id="@+id/edit_postcode"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:maxLenght="4"
            android:hint="@string/postcode"
            android:inputType="number" />

        <Spinner
            android:id="@+id/edit_state"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:entries="@array/states_array" />
</LinearLayout>
```

2. Handle the click event of the "Enter a new address" button on the PickupAddressActivity activity, which starts the EnterPickupAddressActivity.

Task 6:

1. Create an activity named EnterOtherInfoActivity that allows a user to enter other information such as the pickup date and time, contact name, contact number, and note to the driver.

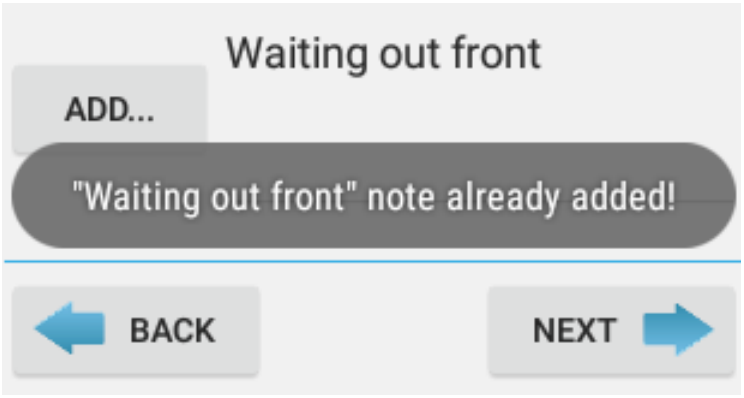


Requirements:

- The date and time fields must be *readable* only their texts are centered horizontally. Hint: Use TextView instead of EditText and set its background to @android:drawable/editbox_background in xml file:

```
android:background="@android:drawable/editbox_background"
android:gravity="center_horizontal"
```

- Click on the "Set Date…" button will open a DatePickerDialog. Similarly, click on the "Set Time…" will open a TimePickerDialog.
- The inputType of the "Contact name" field must be "textPersonName".
- The inputType of the "Contact number" field must be "phone".
- Click on the "Add…" button will open a dialog where the user can select one or more pre-defined notes, and add them into the "Notes to driver" field.
- [Advanced] If a type of notes has been already added, don't add it and just show a toast notification let the user know about this like the following figure.



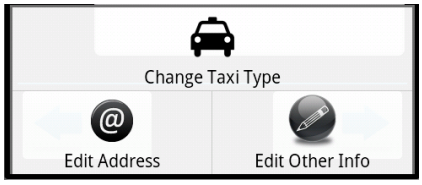
2. Handle the click event of the "Next" button on the EnterPickupAddressActivity activity, which starts the EnterOtherInfoActivity.

Task 7:

1. Create an activity named ReviewBookingActivity that allows a user to review their booking details (which he/she has entered previously) before he/she submits them to the taxi service provider. You are free to creatively design this user interface.

2. Handle the click event of the "Next" button on the EnterOtherInfoActivity activity, which starts the ReviewBookingActivity.

3. Add a menu for this ReviewBookingActivity activity that allows the user to quickly edit the booking details. The menu items are "Change Taxi Type", "Edit Address", and "Edit Other Info". Clicking on each of these menu time, start the corresponding activities which you have design in the previous tasks.



Task 8:

Handle the click events of the "Back" button on the activities, which come back to the previous corresponding activities.

Prepared by Chuong C. Vo, 2012, modified by S.W. Loke, 2014, modified by A.A.D.V.S Abeysinghe 2015.