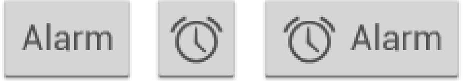
User Interaction

Input Controls

- User interaction in mobile apps typically involves tapping, typing, using gestures, or talking.
- When designing an interactive app:
  - Think about how users will use your app.
  - Minimize steps.
  - Use UI elements that are easy to access, understand, use.
  - Follow Android's best practices.
  - Meet user's expectations.
- A **button** is usually a rectangle or rounded rectangle with a descriptive caption or icon in its center. A button can have text or icon or both.



With text:

```
<Button
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="@string/button_text" />
```

With icon:

```
<ImageButton
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
    android:src="@drawable/button_icon" />
```

With text and icon:

```
<Button
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="@string/button_text"
   android:drawableLeft="@drawable/button_icon" />
```

- Each button has three (3) states: normal, disabled, and pressed.
- A **raised button** appears lifted from the screen—the shading around it indicates that it is possible to tap or click it.
- A **flat button** or borderless button is a text-only button that looks flat and doesn't have a shadow.

```
<Button
   android:id="@+id/button_send"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="@string/button_send"
   android:onClick="sendMessage"
   style="?android:attr/borderlessButtonStyle" />
```

- Clickable images include ImageView and floating action buttons (FAB).
- An **ImageView** is used to display image resources. It can be turned into a button by adding the `android:onClick` attribute in the XML layout.
- The image for the ImageView must already be stored in **app > src > main > res > drawable** folder of the project.

```
<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/icecream_circle"
    android:onClick="orderIcecream"/>
```

- A **floating action button** (FAB) is a circular button that appears to float above the layout.

```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom|end"
    android:layout_margin="@dimen/fab_margin"
    android:src="@drawable/ic_fab_chat_button_white" />
```

- A **touch gesture** occurs when a user places one or more fingers on the touchscreen, and the app interprets that pattern of touches as a particular gesture, such as a tap, touch & hold, double-tap, fling, or scroll.
- **Input controls** are interactive elements in the app's UI that accept data input.
- A text field **(EditText)** is used for entering and modifying text.

```
<EditText
    android:id="@+id/plain_text_input"
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:inputType="text"/>
```

- A **check box** is used for selecting one or more options.

```
<CheckBox
    android:id="@+id/checkbox1_chocolate"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/chocolate_syrup" />
```

- A **radio button** restricts the user to select only one (1) option from a set.

```
<RadioGroup
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:orientation="vertical">
       <RadioButton android:id="@+id/radio_pirates"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:text="@string/pirates"
          android:onClick="onRadioButtonClicked"/>
       <RadioButton android:id="@+id/radio_ninjas"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:text="@string/ninjas"
          android:onClick="onRadioButtonClicked"/>
</RadioGroup>
```

- A **spinner** is a drop-down menu for selecting only one (1) option.
res/values/strings.xml

```
<resources>
   <string-array name="language_array">
      <item>English</item>
      <item>Filipino</item>
      <item>Spanish</item>
      <item>Others</item>
   </string-array>
</resources>
```

Layout:

```
<Spinner
      android:id="@+id/language_spinner"
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      android:entries="@array/language_array" />
```

- Here is a sample statement that stores the value of the selected option in a String.
String text = mySpinner.getSelectedItem().toString();
- A **toggle button** lets the user change a setting between two (2) states.

```
<ToggleButton
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:id="@+id/my_toggle"
      androd:text="@string/turn_on_or_off"
       android:onClick="onToggleClick" />
```

- A type of toggle button called **switch** (for Android 4.0+) provides a slider for selection.

```
<Switch
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:id="@+id/my_switch"
      android:text="@string/turn_on_or_off"
      android:onClick="onSwitchClick"/>
```

- A **SeekBar** is dragged to the left or right to indicate a current progress level.

```
 <SeekBar
      android:id="@+id/seekBar1"
      android:layout_width="300dp"
      android:layout_height="wrap_content"
      android:max="100"
      android:progress="50" />
```

- Here is a sample statement that stores the progress of the SeekBar in an *int* variable:
int status = mySeekBar.getProgress();

Input Events

- An **event** occurs when the user interacts with an input control or item in the user interface.
- An **event listener** is an interface in the *View* class that contains a single callback method. These methods will be called when the *View* to which the listener has been registered is triggered by user interaction with the item in the UI.
- The most common callback methods are as follows:

Method	**Interface**	**Description**
onClick()	OnClickListener	When the user either touches or focuses upon the item
onLongClick()	OnLongClickListener	When the user either touches or focuses upon the item for a second
onKey()	OnKeyListener	When the user is focused on the item and presses or releases a hardware key on the device
onTouch()	OnTouchListener	When the user performs an action qualified as a touch event (a press, a release, or any movement gesture on the screen)

- Most input controls use onClick() and onTouch().
- Sample event for CheckBox:

```
public void onCheckBoxClicked(View view) {
  boolean checked = ((CheckBox) view).isChecked();
  switch(view.getId()) {
    case R.id.checkbox1:
      if (checked)
        // Actions here
      else
        // Actions here
      break;
    case R.id.checkbox2:
      if (checked)
        // Actions here
      else
        // Actions here
      break;
  }
}
```

Note: *This can also work with RadioButton.*

References:

DiMarzio, J. (2017). *Beginning Android programming with Android Studio*. Indiana: John Wiley & Sons, Inc.

Google Developers Training Team. (2018). *Android developer fundamentals (version 2)*. Retrieved from https://google-developer-training.github.io