2

Welcome App

Objectives

In this chapter you'll:

- Learn the basics of the Android Studio IDE, which you'll use to write, test and debug your Android apps.
- Use the IDE to create a new app project.
- Design a graphical user interface (GUI) visually (without programming) using the IDE's layout editor.
- Display text and an image in a GUI.
- Edit the properties of views (GUI components).
- Build and launch an app in the Android emulator.
- Make the app more accessible to visually impaired people by specifying strings for use with Android's TalkBack and Explore-by-Touch features.
- Support internationalization so your app can display strings localized in different languages.

Self-Review Exercises

2.1 Fill in the blanks in each of the following statements:

a) Layout files are considered app resources and are stored in the project's _____ folder. GUI layouts are placed within that folder's `layout` subfolder.

ANS: `res`.

b) When designing an Android GUI, you typically want it to be _____ so that it displays properly on various devices.

ANS: scalable.

c) You can easily _____ your app by creating additional XML resource files for string resources in other languages.

ANS: localize.

d) The two measurement units for density independent pixels are _____ and _____.

ANS: `dp` and `dip`.

e) _____ enables the user to hear TalkBack speak what's on the screen where the user touches.

ANS: Explore by Touch.

f) Android uses a special folder-naming scheme to automatically choose the correct localized resources—for example, the folder _____ would contain a `strings.xml` file for French and the folder _____ would contain a `strings.xml` file for Spanish.

ANS: `values-fr`, `values-es`.

2.2 State whether each of the following is *true* or *false*. If *false*, explain why.

a) Android Studio is used to create and test Android apps.

ANS: True.

b) A `RelativeLayout` arranges views relative to one another or relative to their parent container.

ANS: True.

c) A `LinearLayout` arranges views horizontally.

ANS: False. A `LinearLayout` arranges views horizontally or vertically.

d) To center the text in the `TextView`, set its `alignment` property to center.

ANS: False. To center the text in the `TextView`, set its `gravity` property to center.

e) Android's accessibility features help people with various disabilities use their devices.

ANS: True.

f) For people with visual disabilities, Android's SpeakBack can speak screen text or text that you provide to help the user understand the purpose of a GUI component.

ANS: False. The feature is named TalkBack.

g) It's considered a best practice in Android to ensure that every GUI component can be used with TalkBack by providing text for the `contentDescription` property of any component that does not display text.

ANS: True.

Exercises

2.3 Fill in the blanks in each of the following statements:

a) Android Studio's _____ allows you to build GUIs using drag-and-drop techniques.

ANS: layout editor.

b) For an app based on the **Empty Activity** template, the GUI layout is stored in an XML file called _____, by default.

ANS: `activity_main.xml`.

c) The default GUI for an app based on the **Empty Activity** template consists of a(n) _____ (layout) and a `TextView` containing `"Hello world!"`.

ANS: `RelativeLayout`.

d) The documentation for supporting multiple screen sizes recommends that you use density-independent pixels for the dimensions of GUI components and other screen elements and _____ for font sizes.

ANS: scale-independent pixels

e) One density-independent pixel is equivalent to one pixel on a screen with 160 dpi (dots per inch). On a screen with 240 dpi, each density-independent pixel will be scaled by a factor of _____.

ANS: 240/160 (i.e., 1.5).

f) On a screen with 120 dpi, each density-independent pixel is scaled by a factor of _____. So, the same component that's 100 density-independent pixels wide will be 75 actual pixels wide.

ANS: 120/160 (i.e., .75).

2.4 State whether each of the following is *true* or *false*. If *false*, explain why.

a) For images to render nicely, a high-pixel-density device needs lower-resolution images than a low-pixel-density device.

ANS: False. For images to render nicely, a high-pixel-density device needs *higher*-resolution images than a low-pixel-density device.

b) It's considered a good practice to "externalize" strings, string arrays, images, colors, font sizes, dimensions and other app resources so that you, or someone else on your team, can manage them separately from your application's code.

ANS: True.

c) You can use the **Layout** editor to create a working app without writing any Java code.

ANS: True.

2.5 ***(Scrapbooking App)*** Find three open source images of famous landmarks using websites such as Flickr. Create an app in which you arrange the images in a collage. Add text that identifies each landmark. Recall that image file names must use all lowercase letters.

ANS: This is nearly identical to the **Welcome** app, but consists of three `ImageViews` and three `TextViews`. Using the layout editor, place an `ImageView` then `TextView` onto a `LinearLayout`, then repeat this process two more times. Use smaller font sizes for the captions, so that more space can be used to display the images.

2.6 ***(Scrapbooking App with Accessibility)*** Using the techniques you learned in Section 2.7, enhance your solution to Exercise 2.5 to provide strings that can be used with Android's TalkBack accessibility feature. If you have an Android device available to you, test the app on the device with TalkBack enabled.

ANS: This requires the same steps we demonstrated in Section 2.7 for the **Welcome** app's `TextView` and `ImageViews`. For this exercise, apply the steps to all three `TextViews` and all three `ImageViews`.

2.7 ***(Scrapbooking App with Internationalization)*** Using the techniques you learned in Section 2.8, enhance your solution to Exercise 2.6 to define a set of strings for another spoken language. Use an online translator service, such as `translate.google.com` to translate the strings and place them in the appropriate `strings.xml` resource file. Use the instructions in Section 2.8 to test the app on an AVD (or a device if you have one available to you).

ANS: This requires the same steps we demonstrated in Section 2.8 for the **Welcome** app's `Strings`. To translate the app's `Strings`, students can use an online translation service such as `http://translate.google.com` or `http://www.bing.com/translator/`. When localizing a for-sale app, `Strings` should be translated by someone with locale-specific expertise to ensure that the text makes sense in each spoken language and dialect.