Table of Contents:

Using Android Studio

Welcome to the "emergency" chapter for readers of my two books --

Java Programming for Android Developers For Dummies
Android Application Development All-in-One For Dummies

I'm currently working on a second edition of the *All-in-One* book, but you probably don't want to wait for that edition to become available. In the meantime, note that the stewards of Android have changed their http://developer.android.com/sdk/index.html page. They now support Android Studio as the official development platform for Android apps.

TIP: You can still use Eclipse if you decide to do so, but it's clear to everyone that Android Studio is "in" and Eclipse with Android tooling is very "yesterday." If you want to use Eclipse, visit eclipse.org and get *Eclipse for Java Developers*. After installing *Eclipse for Java Developers*, follow the instructions at http://developer.android.com/sdk/installing/installing-adt.html to add the ADT (Android Developer Tools) to Eclipse.

Anyway, I've pasted together some instructions on running my book's examples with Android Studio. Please email any questions that you have to me at mailto:android@allmycode.com.

Installing Android Studio

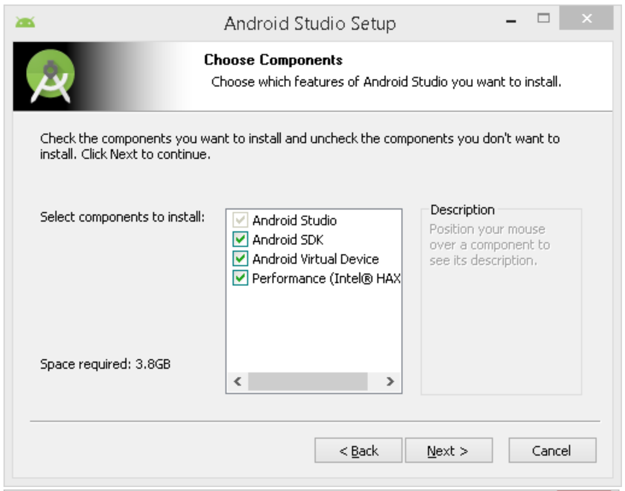
1. **Visit http://developer.android.com/sdk/index.html.**

2. **Click the download button for your operating system (Windows, Mac, or Linux).**

3. **(On a Mac :) Double-click the downloaded `.dmg` file's icon. In the resulting Finder window, drag `Android Studio` (or `Android Studio.app`) to the Applications folder.**

   (On Windows :) Double-clicking the `.exe` file's icon. And follow the steps in the resulting wizard.

   Here are some of the wizard steps that I see when I install Android Studio on Windows:


Android Studio Setup

Choose Components

Choose which features of Android Studio you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

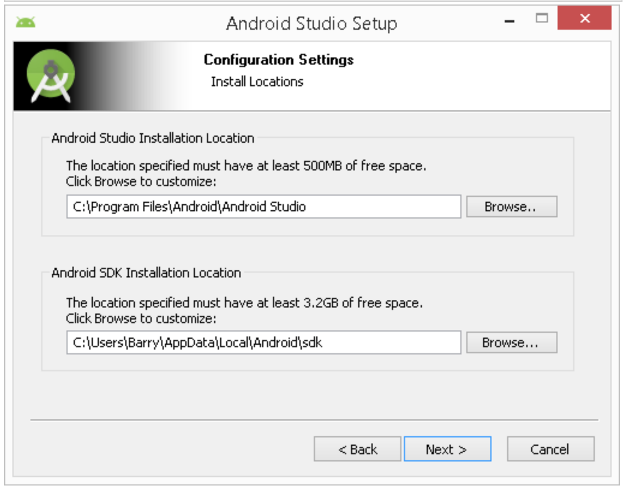
Select components to install:

- [x] Android Studio
- [x] Android SDK
- [x] Android Virtual Device
- [x] Performance (Intel® HAX

Description

Position your mouse over a component to see its description.

Space required: 3.8GB


Android Studio Setup

Configuration Settings

Install Locations

Android Studio Installation Location

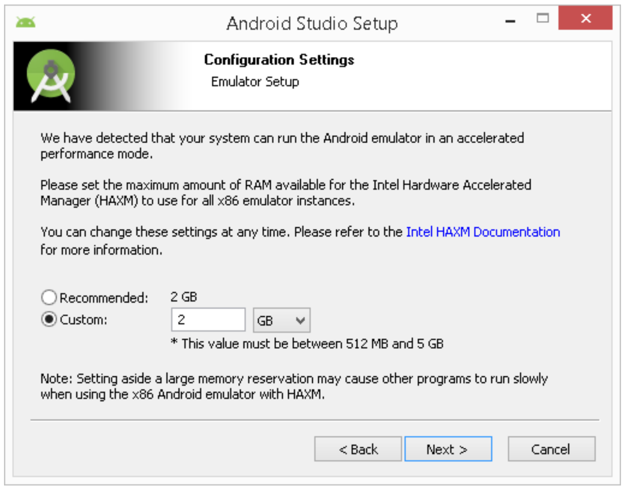
The location specified must have at least 500MB of free space.
Click Browse to customize:

C:\Program Files\Android\Android Studio

Android SDK Installation Location

The location specified must have at least 3.2GB of free space.
Click Browse to customize:

C:\Users\Barry\AppData\Local\Android\sdk


Android Studio Setup

Configuration Settings

Emulator Setup

We have detected that your system can run the Android emulator in an accelerated performance mode.

Please set the maximum amount of RAM available for the Intel Hardware Accelerated Manager (HAXM) to use for all x86 emulator instances.

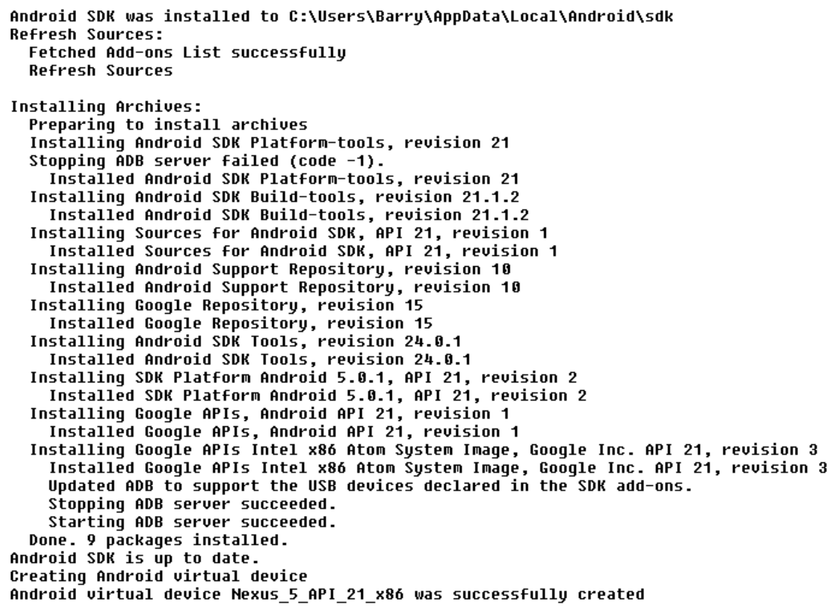
You can change these settings at any time. Please refer to the Intel HAXM Documentation for more information.

Recommended: 2 GB
[✓ Custom:] 2 GB

* This value must be between 512 MB and 5 GB

Note: Setting aside a large memory reservation may cause other programs to run slowly when using the x86 Android emulator with HAXM.

(I had no special reason for changing the memory value from Recommended to Custom. I was just goofing around with the settings.)



```
Android SDK was installed to C:\Users\Barry\AppData\Local\Android\sdk
Refresh Sources:
  Fetched Add-ons List successfully
  Refresh Sources

Installing Archives:
  Preparing to install archives
  Installing Android SDK Platform-tools, revision 21
  Stopping ADB server failed (code -1).
    Installed Android SDK Platform-tools, revision 21
  Installing Android SDK Build-tools, revision 21.1.2
    Installed Android SDK Build-tools, revision 21.1.2
  Installing Sources for Android SDK, API 21, revision 1
    Installed Sources for Android SDK, API 21, revision 1
  Installing Android Support Repository, revision 10
    Installed Android Support Repository, revision 10
  Installing Google Repository, revision 15
    Installed Google Repository, revision 15
  Installing Android SDK Tools, revision 24.0.1
    Installed Android SDK Tools, revision 24.0.1
  Installing SDK Platform Android 5.0.1, API 21, revision 2
    Installed SDK Platform Android 5.0.1, API 21, revision 2
  Installing Google APIs, Android API 21, revision 1
    Installed Google APIs, Android API 21, revision 1
  Installing Google APIs Intel x86 Atom System Image, Google Inc. API 21, revision 3
    Installed Google APIs Intel x86 Atom System Image, Google Inc. API 21, revision 3
    Updated ADB to support the USB devices declared in the SDK add-ons.
    Stopping ADB server succeeded.
    Starting ADB server succeeded.
  Done. 9 packages installed.
Android SDK is up to date.
Creating Android virtual device
Android virtual device Nexus_5_API_21_x86 was successfully created
```

By default, when you install Android Studio, you also get an Android virtual device (AVD), so you can skip any instructions in the book for adding an AVD.

Installing IntelliJ IDEA Community Edition

My *Java Programming for Android Developers* book contains both Android apps and plain old desktop Java programs. I've heard rumors that you can run desktop Java programs in Android Studio. But doing so means applying several workarounds, and that can be confusing. So for the desktop Java programs, I recommend installing Android Studio's parent IDE; namely, IntelliJ IDEA from JetBrains.

Remember: If you have the *All-in-One* book, you don't have to install IntelliJ IDEA.

1. **Visit https://www.jetbrains.com/.**

2. **Look for the download button for IntelliJ IDEA Community Edition.**

3. **Download and install the software.**

   The steps for doing the download should be similar to the steps given above for Android Studio.

After installing Android Studio and IntelliJ IDEA, follow these steps.

Downloading My Book's Examples

1. **Visit http://allmycode.com/android (for the *All-in-One* book) or http://allmycode.com/Java4Android (for the *Java Programming for Android* book).**

2. **On either page, click the link to download the code examples for Android Studio.**

3. **When the download is finished, double-click the downloaded file's icon to uncompress the file.**

   In case you're wondering, other words for "uncompressing" are "unzipping" and "expanding."

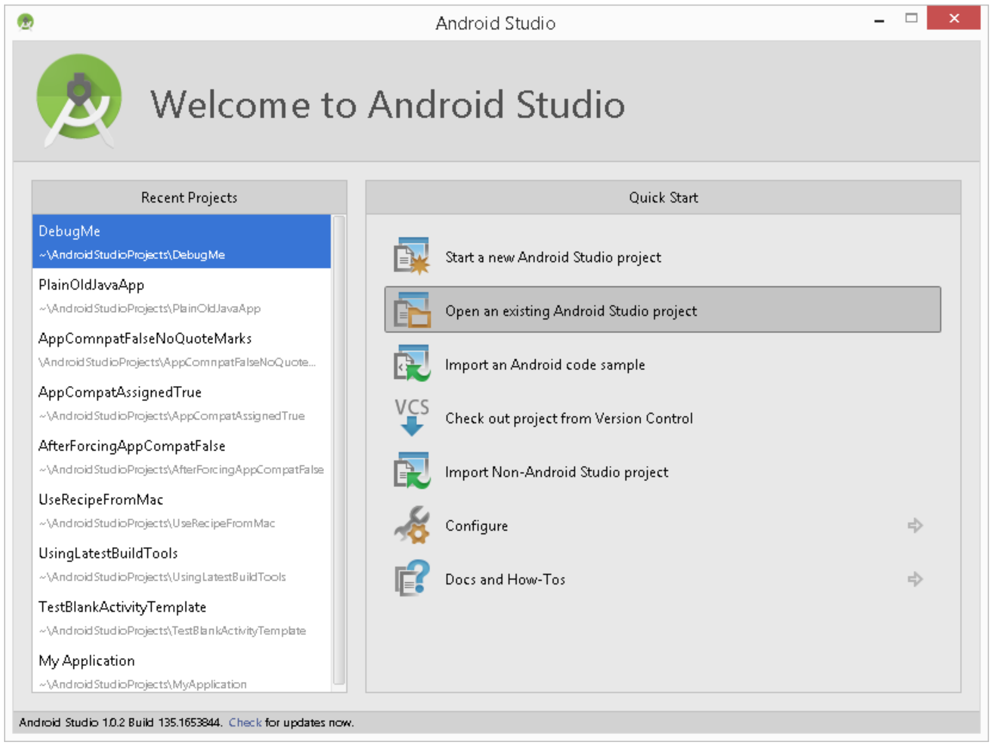
4. **Note the location of the uncompressed folder on your computer's hard drive.**

Launching Android Studio and Importing an Android Project

REMEMBER: For an app that involves two different projects (projects such as *03-01-05* and *03-01-05Other*), you must get Android Studio to run **both** projects (on the same emulator or the same Android device).

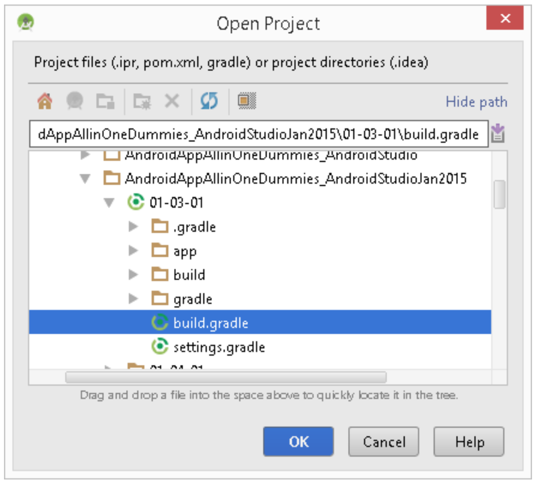
1. **Double-click the Android Studio app's icon.**

   A Welcome screen appears.


Android Studio
Welcome to Android Studio
Recent Projects
DebugMe
~\AndroidStudioProjects\DebugMe
PlainOldJavaApp
~\AndroidStudioProjects\PlainOldJavaApp
AppComnpatFalseNoQuoteMarks
\AndroidStudioProjects\AppComnpatFalseNoQuote...
AppCompatAssignedTrue
~\AndroidStudioProjects\AppCompatAssignedTrue
AfterForcingAppCompatFalse
~\AndroidStudioProjects\AfterForcingAppCompatFalse
UseRecipeFromMac
~\AndroidStudioProjects\UseRecipeFromMac
UsingLatestBuildTools
~\AndroidStudioProjects\UsingLatestBuildTools
TestBlankActivityTemplate
~\AndroidStudioProjects\TestBlankActivityTemplate
My Application
~\AndroidStudioProjects\MyApplication
Quick Start
Start a new Android Studio project
Open an existing Android Studio project
Import an Android code sample
VCS
Check out project from Version Control
Import Non-Android Studio project
Configure
Docs and How-Tos
Android Studio 1.0.2 Build 135.1653844. Check for updates now.

2. **In the Welcome screen, select *Open and Existing Android Studio Project*.**

   An Open Project dialog box appears.



Open Project
Project files (.ipr, pom.xml, gradle) or project directories (.idea)
Hide path
dAppAllinOneDummies_AndroidStudioJan2015\01-03-01\build.gradle
AndroidAppAllinOneDummies_AndroidStudio
AndroidAppAllinOneDummies_AndroidStudioJan2015
01-03-01
.gradle
app
build
gradle
build.gradle
settings.gradle
Drag and drop a file into the space above to quickly locate it in the tree.
OK
Cancel
Help

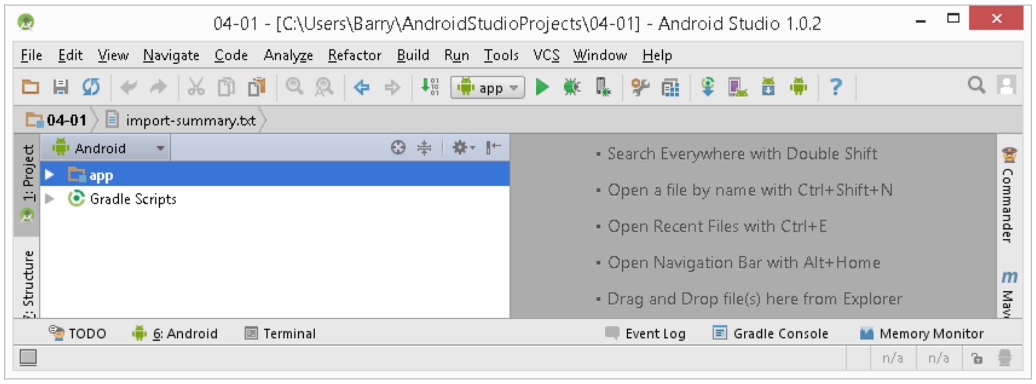
3. **Navigate to the folder containing one of my Android apps.**

   REMEMBER: In the All-in-One book, the first Android app is in Listing 3-1 in Book I. (So the folder's name is *01-03-01*.). In the *Java Programming for Android Developers* book, the first Android app is in Listing 4-1, *not* Listing 3-1. (So the folder's name is *04-01*.)

4. **Select the `build.gradle` file inside that folder.**

5. **Click OK.**

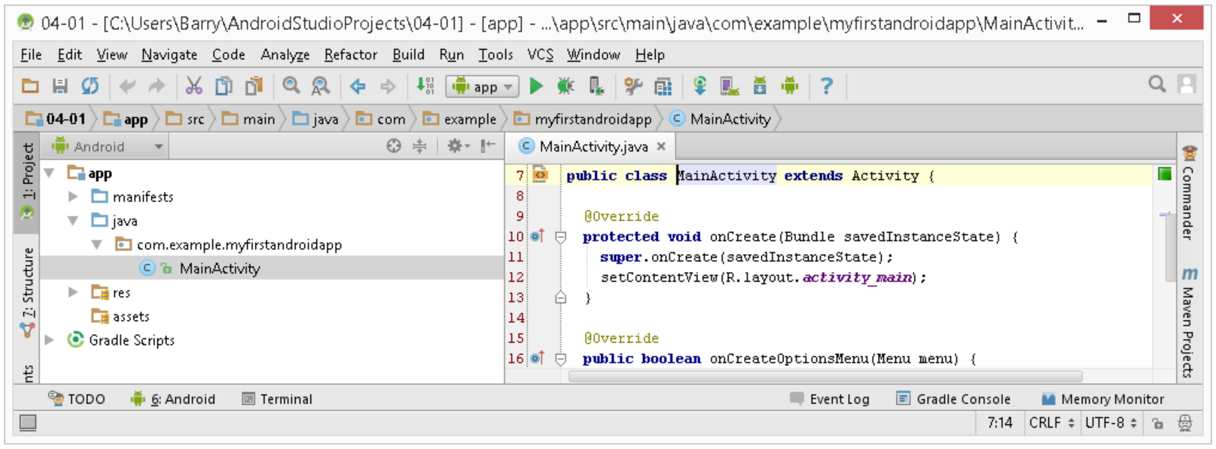
   As a result, Android Studio's main window opens. The project's files appear on the left side (in the Project tool window).



04-01 - [C:\Users\Barry\AndroidStudioProjects\04-01] - Android Studio 1.0.2
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
app
04-01
import-summary.txt
1: Project
Android
app
Gradle Scripts
Structure
• Search Everywhere with Double Shift
• Open a file by name with Ctrl+Shift+N
• Open Recent Files with Ctrl+E
• Open Navigation Bar with Alt+Home
• Drag and Drop file(s) here from Explorer
Commander
Maven
TODO
6: Android
Terminal
Event Log
Gradle Console
Memory Monitor
n/a
n/a

6. **To see my app's Java code, go to the Project tool window, and double-click the** *app/Java/com.example.myfirstandroidapp/MainActivity* **branch.**

When you do, my Java code appears on the right side (in Android Studio's editor).


04-01 - [C:\Users\Barry\AndroidStudioProjects\04-01] - [app] - ...\app\src\main\java\com\example\myfirstandroidapp\MainActivit...
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
04-01 app src main java com example myfirstandroidapp MainActivity
Android
app
manifests
java
com.example.myfirstandroidapp
MainActivity
res
assets
Gradle Scripts
MainActivity.java
TODO
6: Android
Terminal
Event Log
Gradle Console
Memory Monitor
7:14 CRLF UTF-8

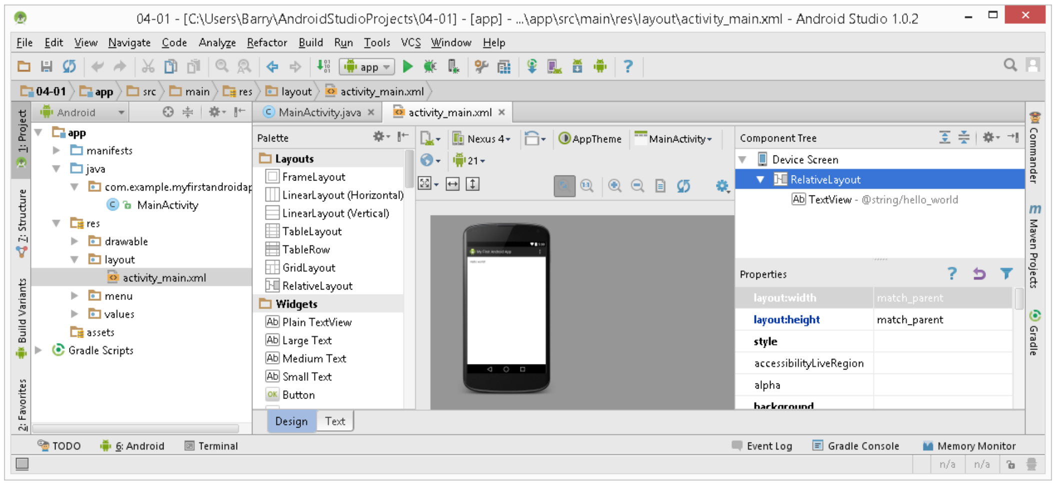
```
public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
```

(The name immediately under the *Java* branch might not be *com.example.myfirstandroidapp* , and the activity's name might not be *MainActivity*. One way or another, look for a branch with the name *Activity* in it.)

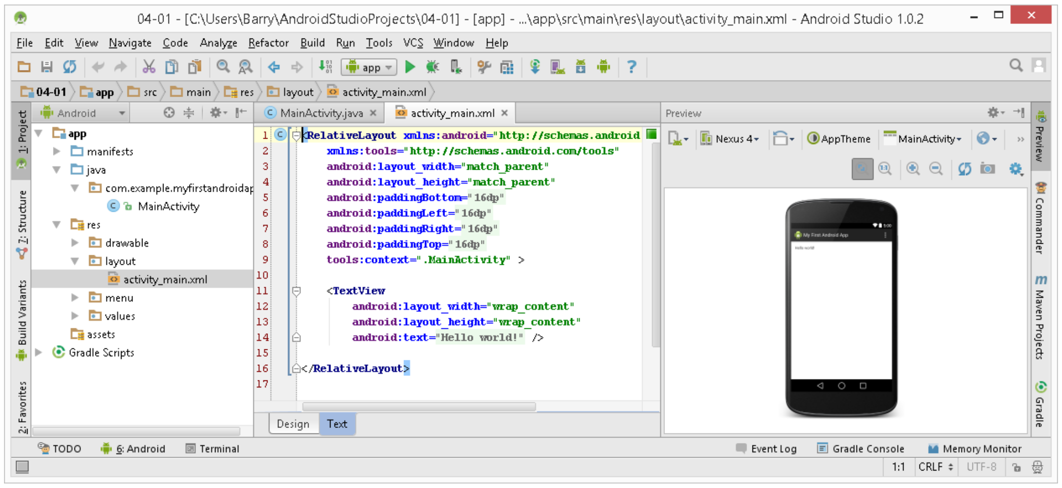
7. **To see my app's layout, go to the Project tool window, and double-click the** *app/res/layout/activity_main.xml* **(or maybe** *app/res/layout/main.xml***) branch.**

   When you do, the Designer tool appears on the right side of Android Studio's main window. The Designer tool is in one of two modes.

   - In Design mode, you see the Palette, a preview screen, the Component tree, and a Properties pane.



   - In Text mode, you see XML code and a preview screen.

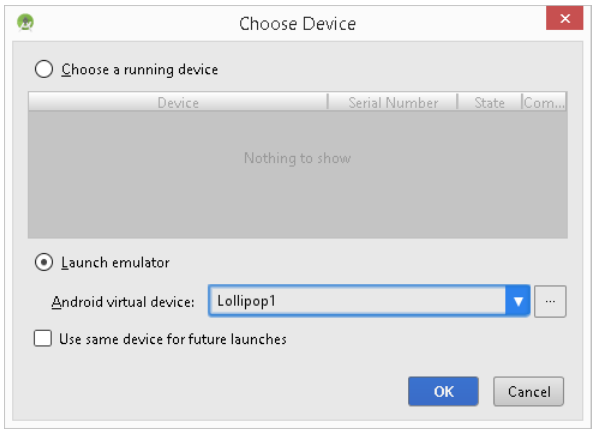

04-01 - [C:\Users\Barry\AndroidStudioProjects\04-01] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 1.0.2

```xml
<RelativeLayout xmlns:android="http://schemas.android
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context=".MainActivity" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello world!" />

</RelativeLayout>
```

Use the tabs in the lower left corner of the Designer tool to switch between Design mode and Text mode.

8. To run the project, go to Android Studio's main menu and click Run→Run 'app'.

As a result, you see a Choose Device dialog box.


Choose Device

Choose a running device

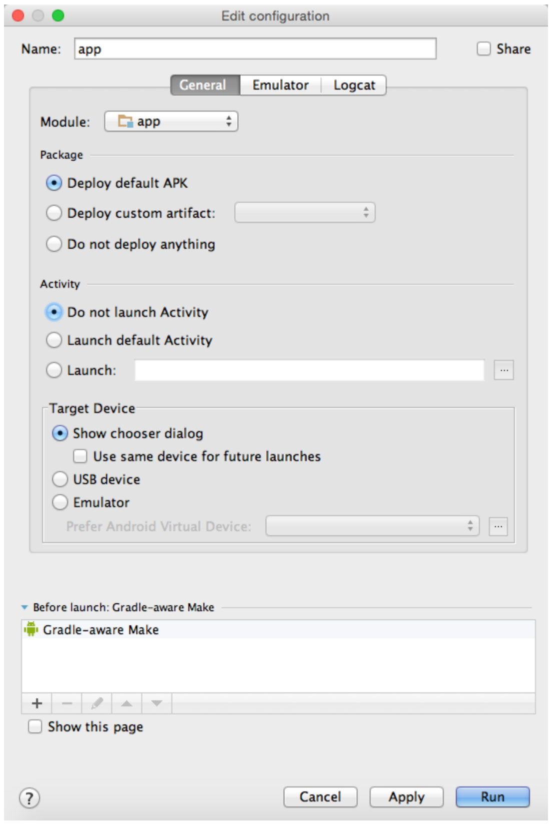
Device	Serial Number	State	Com...
Nothing to show			

[✓ Launch emulator]

Android virtual device: Lollipop1

Use same device for future launches

REMEMBER: For an app that involves two different projects (projects such as *03-01-05* and *03-01-05Other*), you must get Android Studio to run **both** projects (on the same emulator or the same Android device). One of the projects might not have any main activities. In that case, you might see an Edit Configuration dialog box. In that case, select the Do Not Launch Activity option, and then click the Run button.


Edit configuration

Name: app ☐ Share

General | Emulator | Logcat

Module: app

Package

- (●) Deploy default APK
- ( ) Deploy custom artifact:
- ( ) Do not deploy anything

Activity

- (●) Do not launch Activity
- ( ) Launch default Activity
- ( ) Launch:

Target Device

- (●) Show chooser dialog
  - ☐ Use same device for future launches
- ( ) USB device
- ( ) Emulator

Prefer Android Virtual Device:

Before launch: Gradle-aware Make

Gradle-aware Make

☐ Show this page

Cancel | Apply | Run

9. **If you've already started an emulator running, select Choose a Running Device. If you haven't already started an emulator running, select Launch Emulator.**

   In the Android Virtual Device dropdown list, select an AVD whose level number is at least as high is the app's minimum API level. (If you're not sure about this, just give any option in the dropdown list a try. You can read more about this issue in either book.)

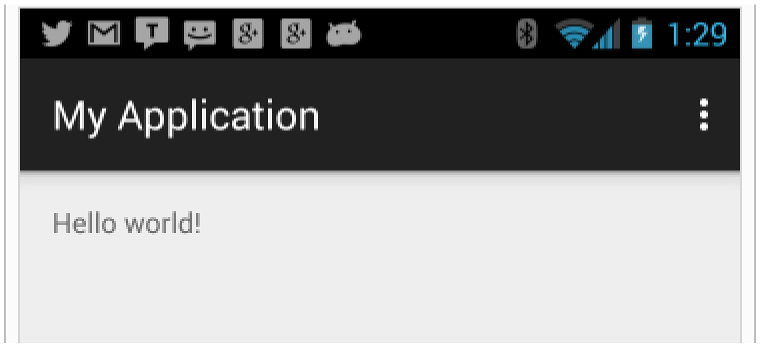
   TIP: For a better emulator experience, try the third-party Genymotion emulator. You can also try running apps on a real Android device (one that's connected to your development computer via USB.) For instructions on running apps on real devices, check out the appropriate sections in one of my books.

10. **Click OK.**

    After a painfully long wait, an emulator window appears on your development computer's screen. The emulator window's screen looks like an Android phone when you turn it on. Eventually, you see the phone's lock screen.

11. **With your mouse, do whatever you have to do in order to unlock the emulator screen. (Usually, a simple swipe does the trick.)**

Eventually, the app appears on your emulator's screen.

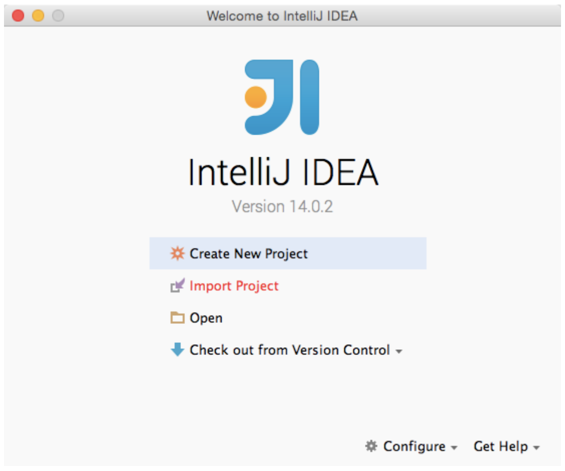


Launching IDEA and Importing a Desktop Java Project

Remember: If you have the *All-in-One* book, you can skip these Desktop Java instructions.

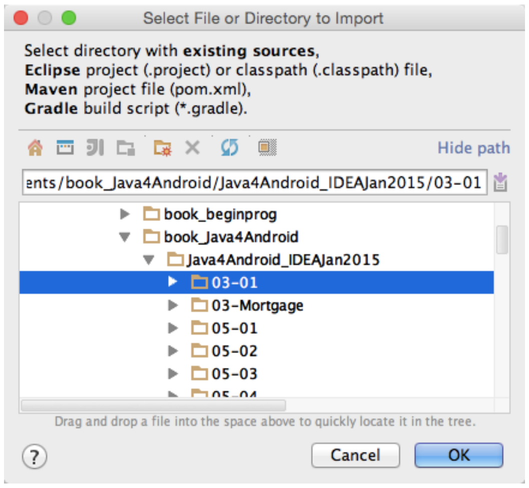
1. **Double-click the IntelliJ IDEA app's icon.**

   A Welcome screen appears.



2. **In the Welcome screen, select Import Project.**

   A *Select File or Directory to Import* dialog box appears.


Select File or Directory to Import

Select directory with **existing sources**,
Eclipse project (.project) or classpath (.classpath) file,
Maven project file (pom.xml),
Gradle build script (*.gradle).

ents/book_Java4Android/Java4Android_IDEAJan2015/03-01

```
▶ book_beginprog
▼ book_Java4Android
  ▼ Java4Android_IDEAJan2015
    ▶ 03-01
    ▶ 03-Mortgage
    ▶ 05-01
    ▶ 05-02
    ▶ 05-03
```

Drag and drop a file into the space above to quickly locate it in the tree.

3. **Navigate to the folder containing one of my desktop Java apps.**

   In the *Java Programming for Android Developers* book, the first desktop Java app is in Listing 3-1 (so the folder's name is *03-01*).

   I put desktop Java apps inside a folder named Java4Android_IDEAJan2014, and put Android apps inside a folder named *Java4Android_AndroidStudioJan2015*. If you're in doubt, you can usually tell which folders contain Android projects and which folders contain desktop Java projects by looking for a `res` subfolder.

   - **The folders that contain `res` subfolders are Android projects.**

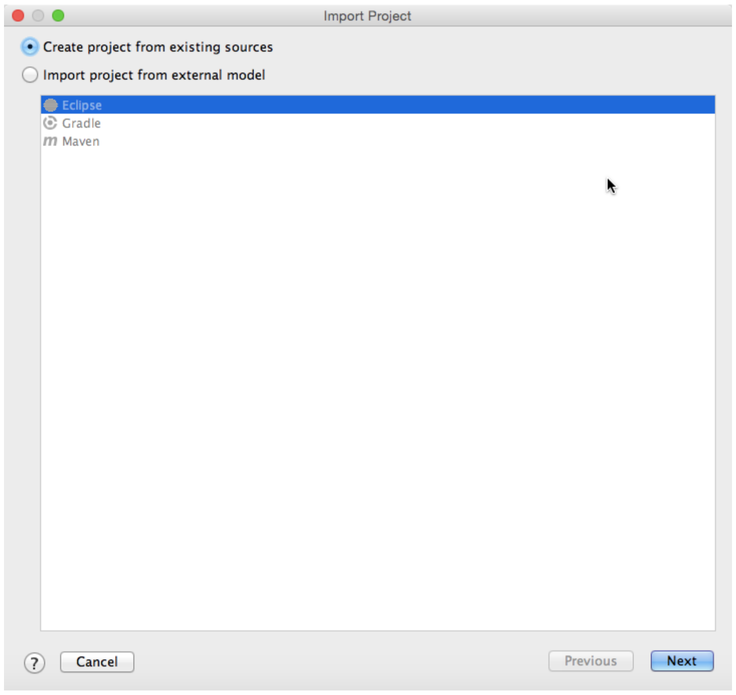
     Open these projects in Android Studio.

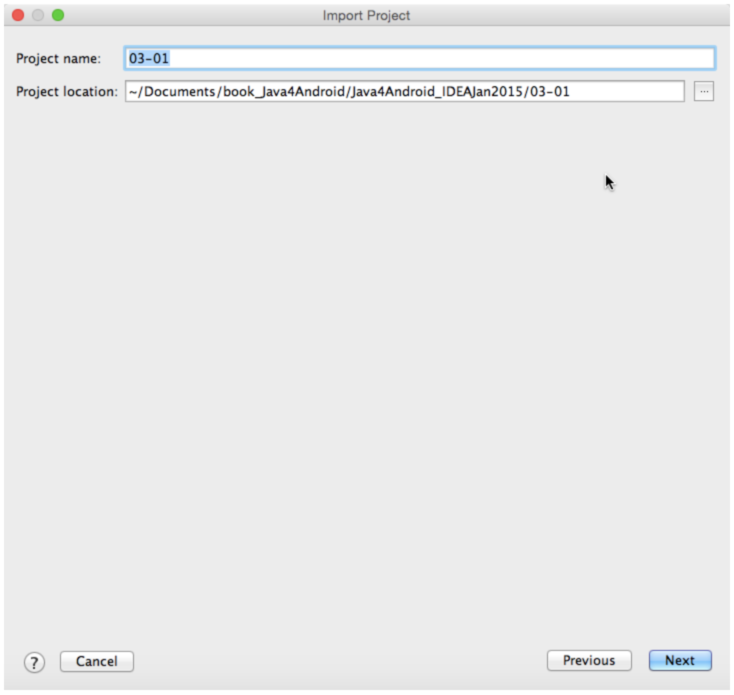
   - **The folders that don't contain `res` subfolders are desktop Java projects.**

     Open these projects in IntelliJ IDEA.

4. **Click OK.**

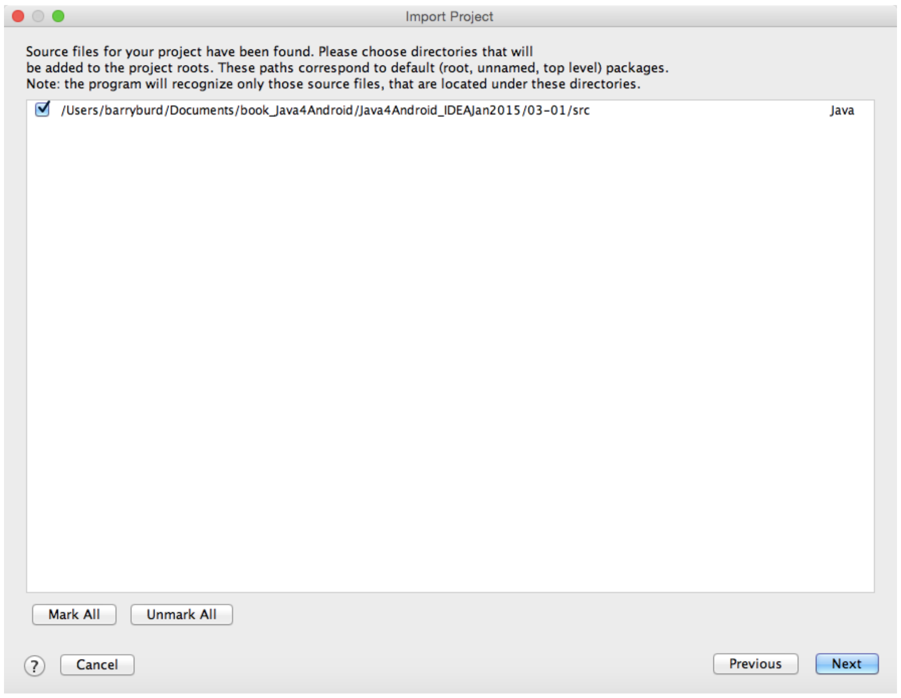
   As a result, you see several more dialog boxes. You can accept the defaults in each of these boxes.


Import Project
(●) Create project from existing sources
( ) Import project from external model
Eclipse
Gradle
Maven
?
Cancel
Previous
Next



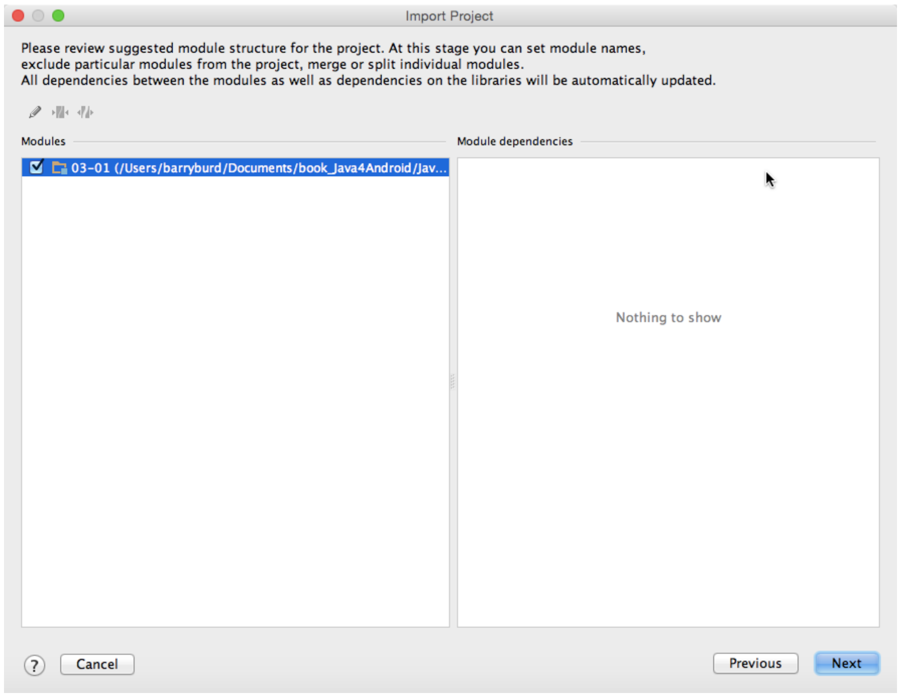
Project name: 03-01

Project location: ~/Documents/book_Java4Android/Java4Android_IDEAJan2015/03-01



Source files for your project have been found. Please choose directories that will
be added to the project roots. These paths correspond to default (root, unnamed, top level) packages.
Note: the program will recognize only those source files, that are located under these directories.

☑ /Users/barryburd/Documents/book_Java4Android/Java4Android_IDEAJan2015/03-01/src Java



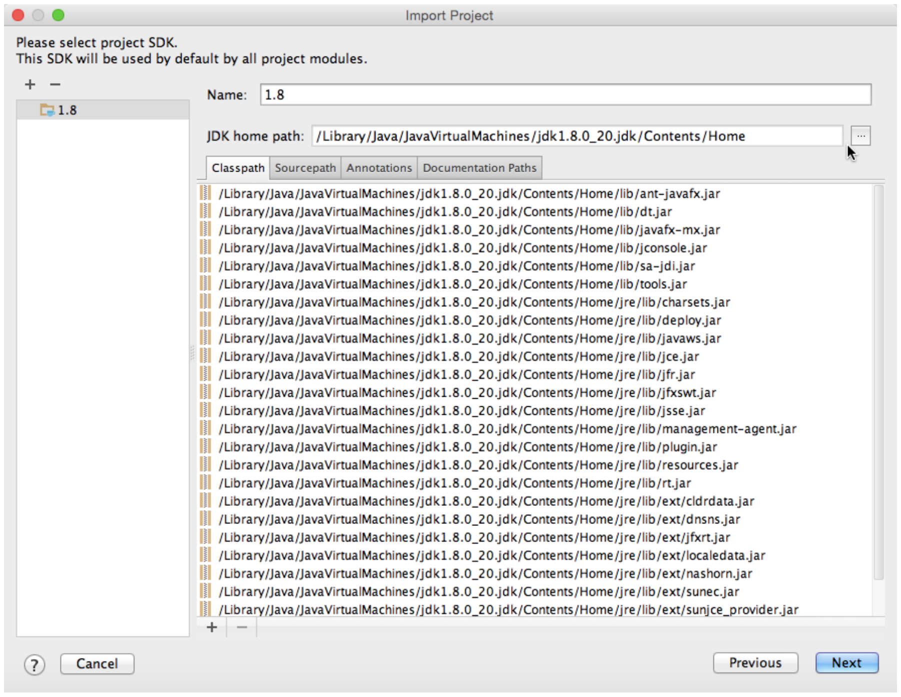
Please review suggested module structure for the project. At this stage you can set module names,
exclude particular modules from the project, merge or split individual modules.
All dependencies between the modules as well as dependencies on the libraries will be automatically updated.

Modules

☑ 03-01 (/Users/barryburd/Documents/book_Java4Android/Jav...

Module dependencies

Nothing to show



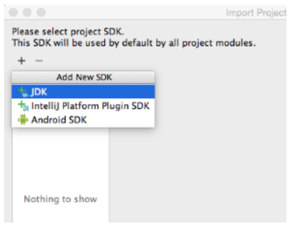
Please select project SDK.
This SDK will be used by default by all project modules.

Name: 1.8

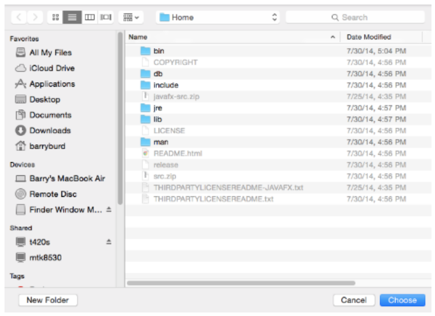
JDK home path: /Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/lib/ant-javafx.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/lib/dt.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/lib/javafx-mx.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/lib/jconsole.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/lib/sa-jdi.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/lib/tools.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/charsets.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/deploy.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/javaws.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/jce.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/jfr.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/jfxswt.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/jsse.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/management-agent.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/plugin.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/resources.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/rt.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/ext/cldrdata.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/ext/dnsns.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/ext/jfxrt.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/ext/localedata.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/ext/nashorn.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/ext/sunec.jar
/Library/Java/JavaVirtualMachines/jdk1.8.0_20.jdk/Contents/Home/jre/lib/ext/sunjce_provider.jar
```

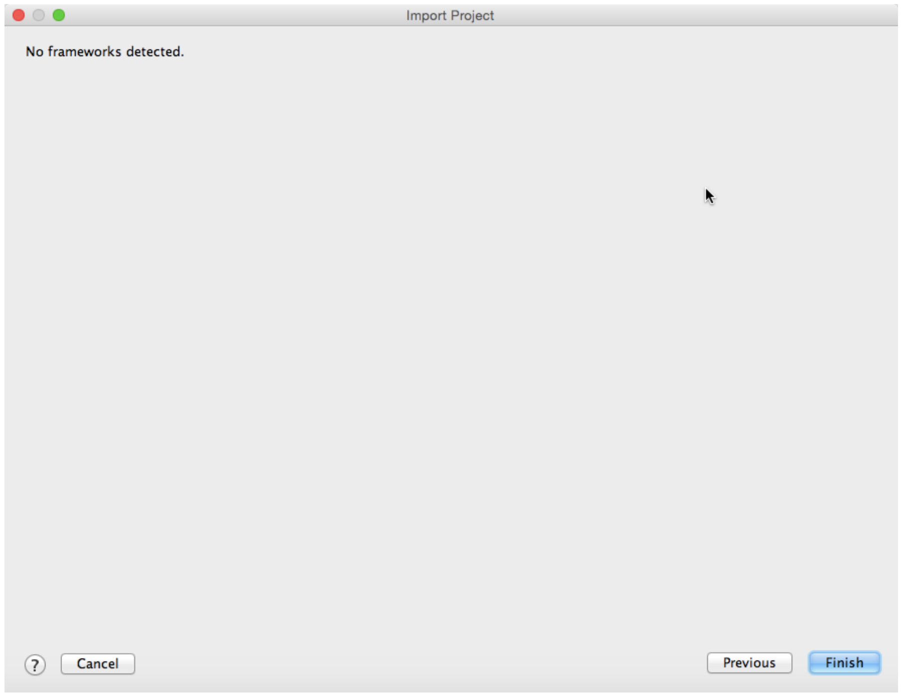
If you see *Nothing to Show* on the left side…



Please select project SDK.
This SDK will be used by default by all project modules.

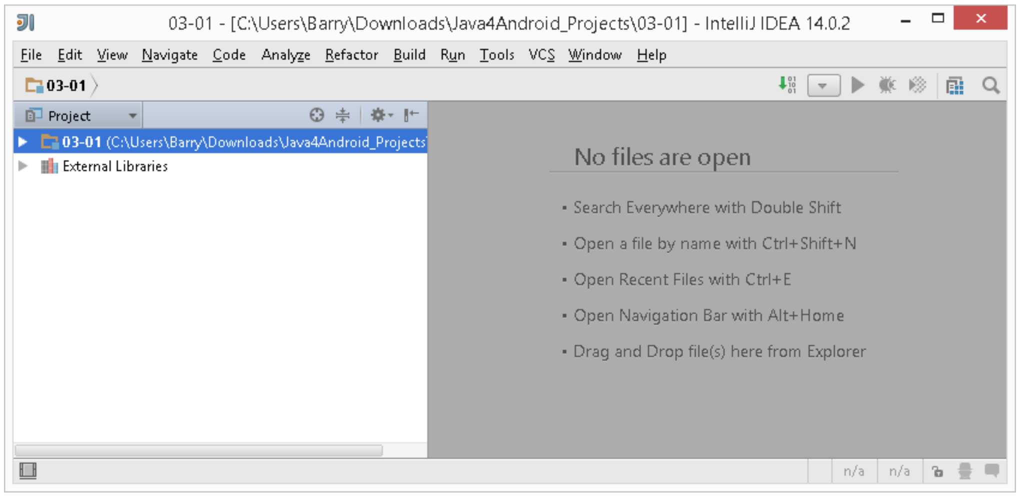


Name	Date Modified
bin	7/30/14, 5:04 PM
COPYRIGHT	7/30/14, 4:56 PM
db	7/30/14, 4:56 PM
include	7/30/14, 4:56 PM
javafx-src.zip	7/25/14, 4:35 PM
jre	7/30/14, 4:57 PM
lib	7/30/14, 4:57 PM
LICENSE	7/30/14, 4:56 PM
man	7/30/14, 4:56 PM
README.html	7/30/14, 4:56 PM
release	7/30/14, 4:56 PM
src.zip	7/30/14, 4:56 PM
THIRDPARTYLICENSEREADME-JAVAFX.txt	7/25/14, 4:35 PM
THIRDPARTYLICENSEREADME.txt	7/30/14, 4:56 PM



No frameworks detected.

Eventually, IDEA's main window opens. The project's files appear on the left side (in the Project tool window).


03-01 - [C:\Users\Barry\Downloads\Java4Android_Projects\03-01] - IntelliJ IDEA 14.0.2

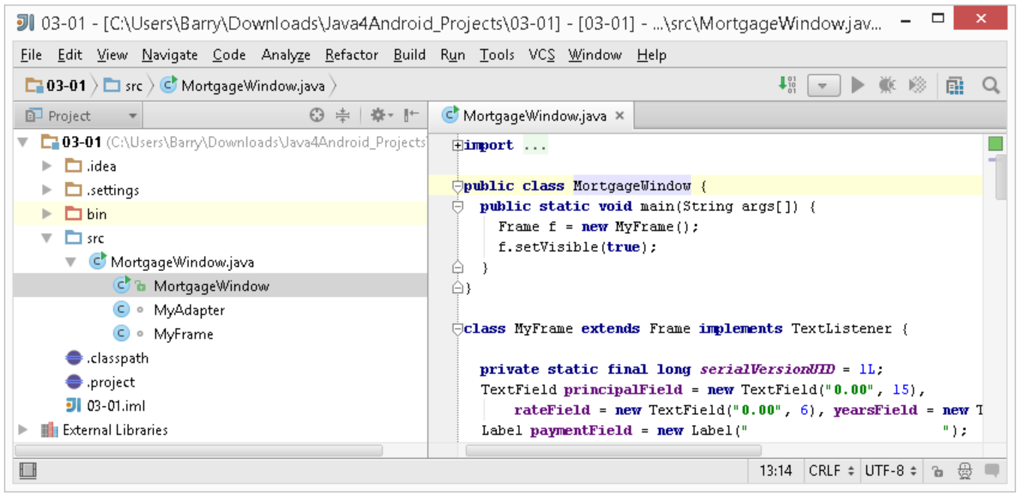
No files are open

- Search Everywhere with Double Shift
- Open a file by name with Ctrl+Shift+N
- Open Recent Files with Ctrl+E
- Open Navigation Bar with Alt+Home
- Drag and Drop file(s) here from Explorer

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
03-01
Project
03-01 (C:\Users\Barry\Downloads\Java4Android_Projects
External Libraries
n/a
n/a

5. To see my program's Java code, go to the Project tool window, and double-click the ***03-01/src/MortgageWindow.java/MortgageWindow*** **branch.**

When you do, my Java code appears on the right side (in IDEA's editor).

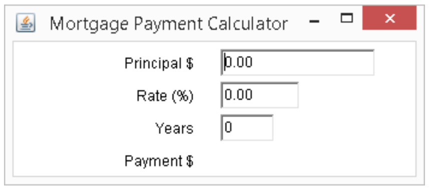

03-01 - [C:\Users\Barry\Downloads\Java4Android_Projects\03-01] - [03-01] - ...\src\MortgageWindow.jav...

```
import ...

public class MortgageWindow {
  public static void main(String args[]) {
    Frame f = new MyFrame();
    f.setVisible(true);
  }
}

class MyFrame extends Frame implements TextListener {

  private static final long serialVersionUID = 1L;
  TextField principalField = new TextField("0.00", 15),
      rateField = new TextField("0.00", 6), yearsField = new T
  Label paymentField = new Label("                    ");
```

(If the project that you've opened isn't *03-01*, the name immediately under the *src* branch might not be *MortgageWindow.java*, and the name immediately under the whatever.Java branch won't be *MortgageWindow*. Adjust your mouse clicks accordingly.)

6. To run the project, go to IDEA's main menu and click Run→Run 'MortgageWindow'.

As a result, the app starts running on your computer's screen.



Principal $ 0.00
Rate (%) 0.00
Years 0
Payment $

Creating Your Own App

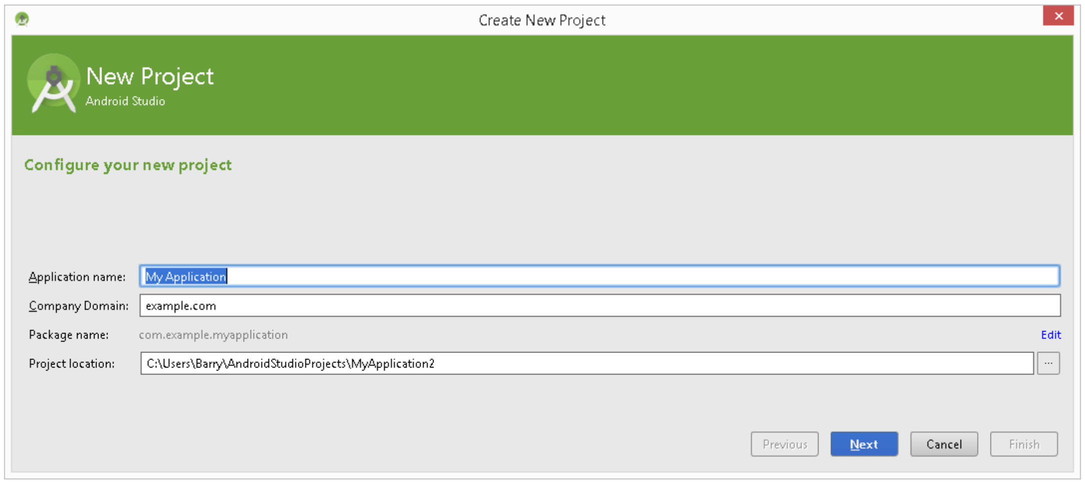
Here's how you create an app:

1. **Launch Android Studio.**

   When you do, you'll see either Android Studio's main window or the Welcome screen.

2. **If you see the main window, click File→New Project. If you see the Welcome screen, click Start a New Android Studio Project.**

   A New Project dialog box appears.



Application name: My Application
Company Domain: example.com
Package name: com.example.myapplication
Project location: C:\Users\Barry\AndroidStudioProjects\MyApplication2

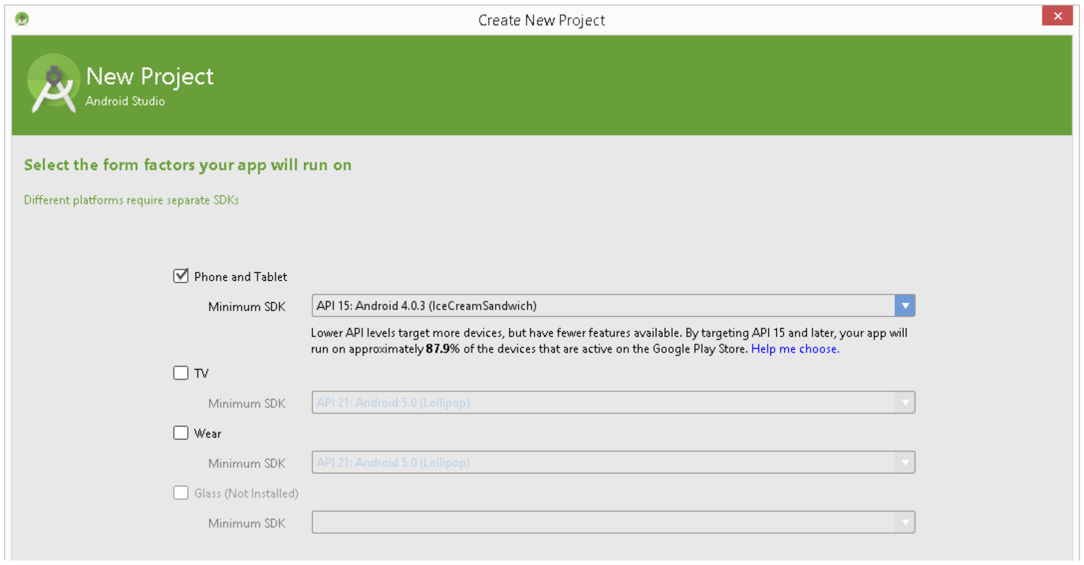
3. **Fill in the fields in the New Project dialog box, and then click Next.**

   If you're just practicing (that is, if you're not creating an app for distribution to the public), you can accept all the defaults.

   For public distribution, you want a catchy Application Name.

   You also want a package name that uniquely identifies you and this app. If you have a domain name, start by reversing the domain name and then adding a word to identify this particular app. In the figure in Step 2, the domain name is *example.com*, and the app's identifying info is *myapplication*.

When you press Next, the next page of the New Project dialog box appears.


Create New Project

New Project
Android Studio

Select the form factors your app will run on

Different platforms require separate SDKs

☑ Phone and Tablet

Minimum SDK: API 15: Android 4.0.3 (IceCreamSandwich)

Lower API levels target more devices, but have fewer features available. By targeting API 15 and later, your app will run on approximately **87.9%** of the devices that are active on the Google Play Store. Help me choose.

☐ TV

Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ Wear

Minimum SDK: API 21: Android 5.0 (Lollipop)

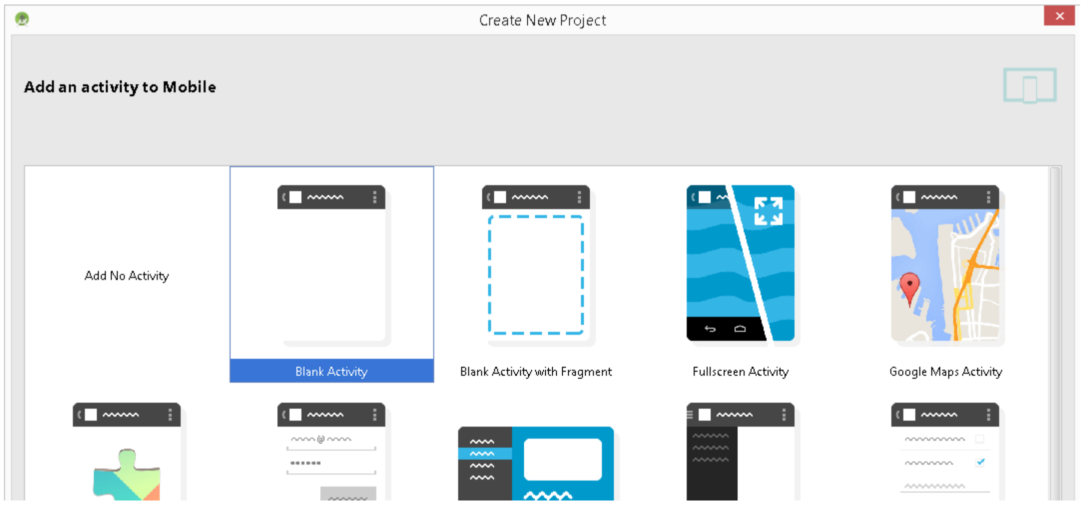
☐ Glass (Not Installed)

Minimum SDK:

4. **For a practice app, accept the defaults (Phone and Tablet with Minimum SDK API 15), and then click Next.**

   For advice on Minimum SDKs and such things, see the appropriate sections of either book.

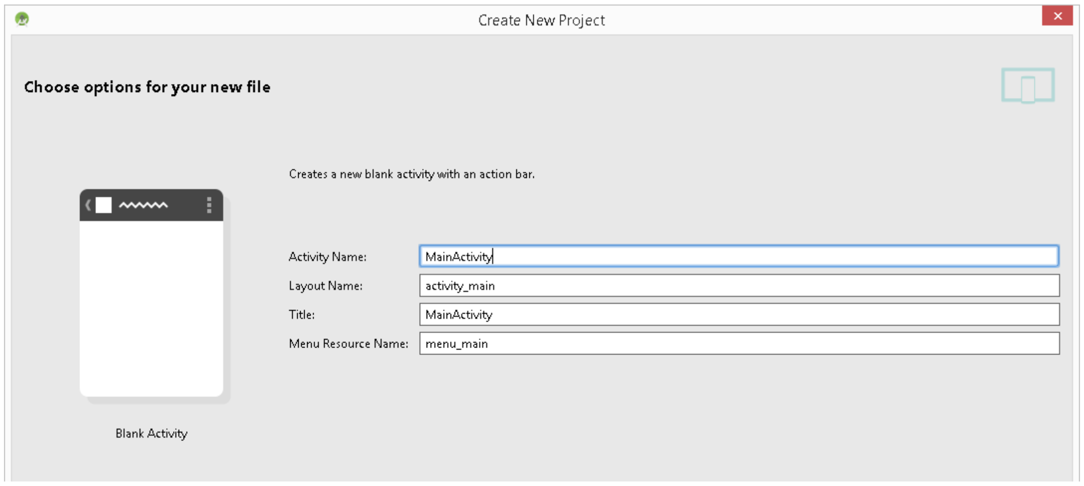
   When you press Next, yet another page of the New Project dialog box appears.


Create New Project

Add an activity to Mobile

5. **For a practice app, accept the default (Blank Activity), and then click Next.**

   You guessed it! When you press Next, another page of the New Project dialog box appears.


Create New Project

Choose options for your new file

Creates a new blank activity with an action bar.

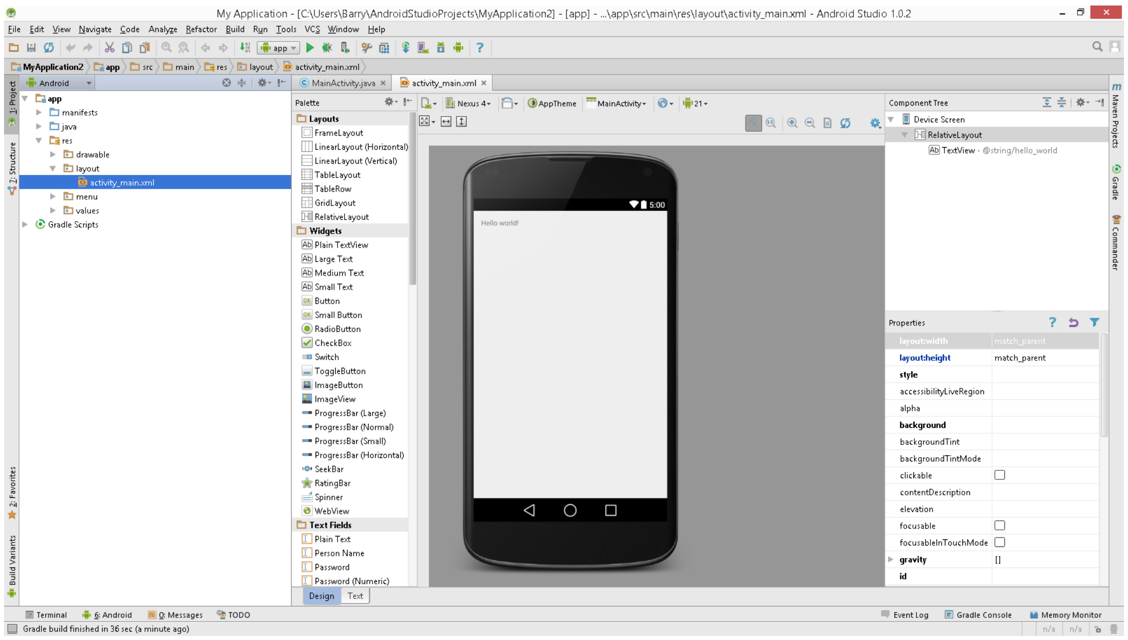
Activity Name: MainActivity

Layout Name: activity_main

Title: MainActivity

Menu Resource Name: menu_main

Accept the defaults and click Finish! As a result, the new application appears in Android Studio's main window.


My Application - [C:\Users\Barry\AndroidStudioProjects\MyApplication2] - [app] - ...\app\src\main\res\layout\activity_main.xml - Android Studio 1.0.2

Gradle build finished in 36 sec (a minute ago)

Embellishing Your App

In this section, you add functionality to the basic Android app:

1. **Follow the steps in the previous section (entitled "Creating Your Own App").**

   As a result, you see the new app's Designer tool with the Palette and the big preview screen. (Refer to the final figure in the "Creating Your Own App" section.)

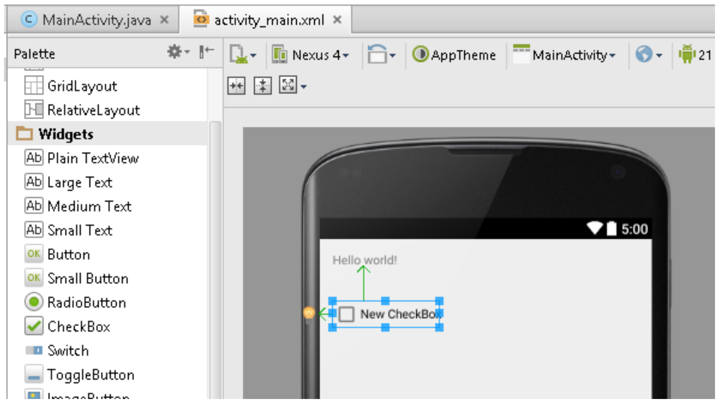
2. **If you don't see the Designer tool, navigate to the *app/res/layout* branch on the left side of Android Studio's main window, and double-click the *activity_main.xml* item.**

   Remember: The Designer tool has two modes: the Design mode and the Text mode. The mode that you want is the Design mode. To switch between modes click the tabs in the lower left corner of the Designer tool.

3. **In the Widgets group of the Palette, click the CheckBox item.**

4. **Click anywhere inside the preview screen.**

   As a result, a CheckBox item appears in your preview screen.



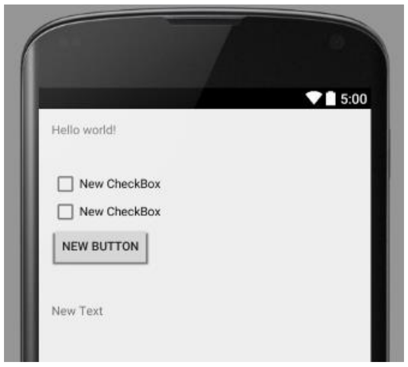
   TIP: You don't have to do two clicks, as in Steps 3 and 4. Instead, you can drag directly from the CheckBox item in the palette to the preview screen.

5. **Repeat Steps 3 and 4 for another CheckBox item.**

6. **Repeat Steps 3 and 4 for Button item.**

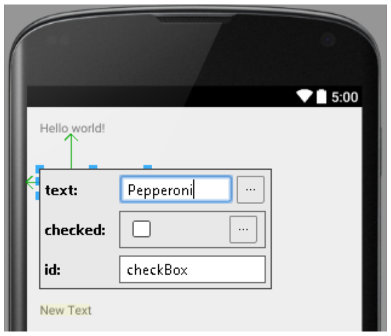
7. **Repeat Steps 3 and 4 for Plain TextView item.**

   Here's what the preview screen looks like now.



8. **In the preview screen, double-click the first CheckBox.**

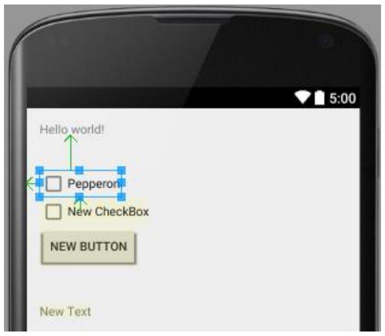
   A Properties popup appears.



text: Pepperoni
checked: ☐
id: checkBox

9. **In the Properties popup's Text field, type `Pepperoni`. Then press Enter.**

   As a result, the checkbox's label changes to `Pepperoni`.



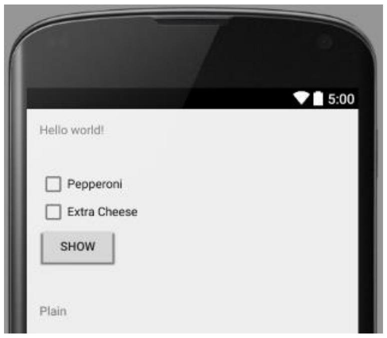
10. **Repeat Steps 8 and 9 for the second checkbox. Change its label to `Extra Cheese`.**

11. **Repeat Steps 8 and 9 for the button. Change its label to `Show`.**

12. **Repeat Steps 8 and 9 for the Plain TextView. Change its text to the word `Plain`.**

    (In case you're wondering, the fact that this TextView item is a *Plain* TextView has nothing to do with typing the word `Plain` in this step. A Plain TextVIew isn't necessarily associated with a plain pizza.)

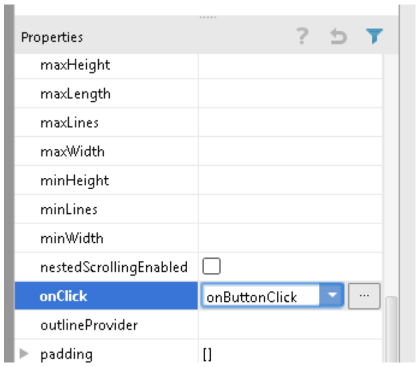
    When you've finished Step 12, the preview screen looks like this:



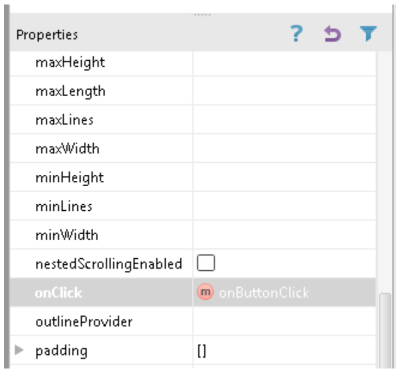
13. **In the preview screen, select the button.**

14. **In the Properties pane (in the lower-right part of the main window) look for the *onClick* item.**

15. **Type `onButtonClick`, and then click your mouse in a neutral spot outside of the Properties pane.**



Properties	
maxHeight	
maxLength	
maxLines	
maxWidth	
minHeight	
minLines	
minWidth	
nestedScrollingEnabled	☐
onClick	onButtonClick
outlineProvider	
padding	[]



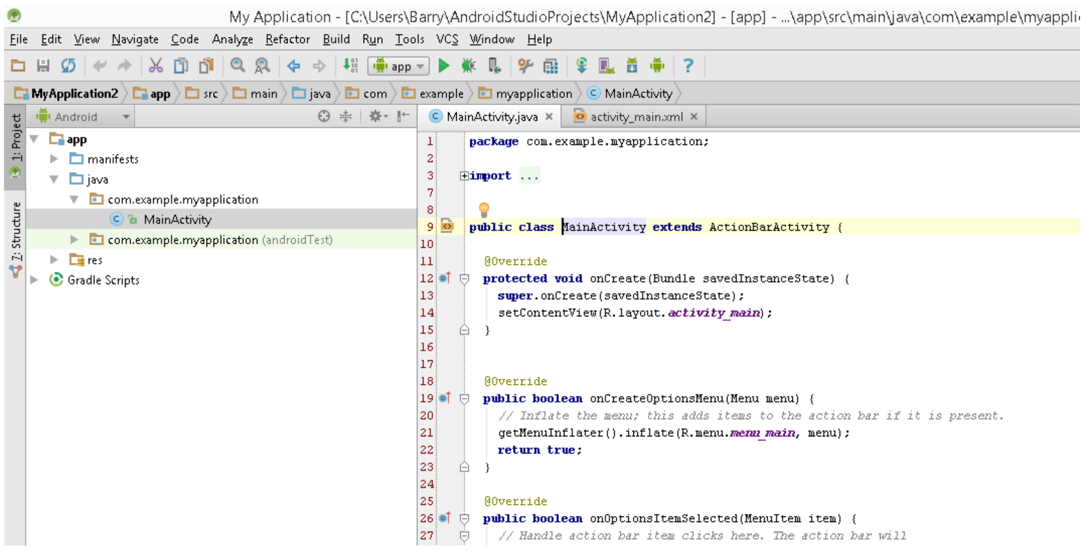
Properties	
maxHeight	
maxLength	
maxLines	
maxWidth	
minHeight	
minLines	
minWidth	
nestedScrollingEnabled	☐
onClick	m onButtonClick
outlineProvider	
padding	[]

16. **In Android Studio's main menu, select File→Save All.**

    Better Save than sorry!

17. **In the Project tool window, double-click the *app/java/com.example.myapplication/MainActivity* branch.**

The `MainActivity` class's code appears in Android Studio's editor.


My Application - [C:\Users\Barry\AndroidStudioProjects\MyApplication2] - [app] - ...\app\src\main\java\com\example\myappli
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication2 app src main java com example myapplication MainActivity
Android
app
manifests
java
com.example.myapplication
MainActivity
com.example.myapplication (androidTest)
res
Gradle Scripts
MainActivity.java
activity_main.xml

```
package com.example.myapplication;

import ...

public class MainActivity extends ActionBarActivity {

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
  }


  @Override
  public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
```

WARNING: Instead of

```
    public class MainActivity extends ActionBarActivity
```

in the editor's code, you might see

```
    public class MainActivity extends Activity
```

You might also see something like

```
    import android.support.v7.app.ActionBarActivity;
```

near the top of the code. There are bound to be some other differences between what your computer shows you and what I describe in these notes and in the book. For now, you can ignore these differences.

18. **Modify the code by adding several lines, as is shown below.**

The lines that you add are set in **green boldface type**. (If see `extends ActionBarActivity` instead of `extends Activity`, you can change it to `extends Activity` or leave it as it is. Either way, the app will work.)

```
package com.example.myapplication;

import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.CheckBox;
import android.widget.TextView;


public class MainActivity extends Activity {
  TextView textView;
  CheckBox pepBox, cheeseBox;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    pepBox = (CheckBox) findViewById(R.id.checkBox);
    cheeseBox = (CheckBox) findViewById(R.id.checkBox2);
    textView = (TextView) findViewById(R.id.textView2);
  }

  public void onButtonClick(View view) {
    StringBuilder str = new StringBuilder("");
    if (pepBox.isChecked()) {
      str.append("Pepperoni" + " ");
    }
    if (cheeseBox.isChecked()) {
      str.append("Extra cheese");
    }
    if (str.length() == 0) {
      str.append("Plain");
    }
    textView.setText(str);
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem item) {
     int id = item.getItemId();

     if (id == R.id.action_settings) {
      return true;
    }
    return super.onOptionsItemSelected(item);
  }
}
```

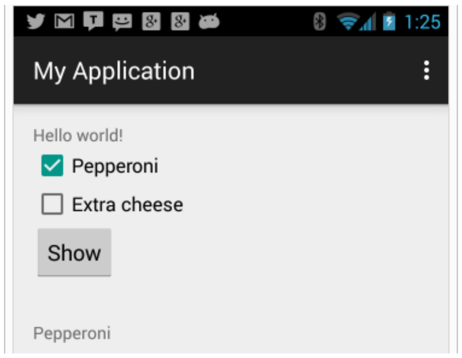
TIP: When you type the word `TextView` or the word `CheckBox`, the word's typeface might be red in Android Studio's editor. If so, it might mean that you haven't yet typed in the appropriate `import` declaration near the top of the editor. If you want to avoid typing the `import` declaration, click your mouse on the red word, and then press Alt+Enter. With any luck, Android Studio will add the `import` declaration automatically for you.

19. **In Android Studio's main menu, select File→Save All.**

20. **In Android Studio's main menu, select Run→Run 'app'.**

    Heed all the advice about running an app that I provided in the "Launching Android Studio and Importing an Android Project" section.

    A successful run of the app looks like this:



That's it. If you have any comments about this document, or if you have any other questions about Android Studio or about any of the examples in my book, email me at android@allmycode.com. You can also tweet me at @allmycode or send a Facebook message to /allmycode.

Happy coding!