Contents

CHAPTER 1

เตรียมความพร้อมก่อนพัฒนา Android Apps

ในปัจจุบันคงไม่มีใครกล้าปฏิเสธได้ว่า การพัฒนา Android Apps มีบทบาทเพิ่มมากขึ้นเรื่อย ๆ อีกแพลตฟอร์มหนึ่ง หลายคนใช้อุปกรณ์มือถือ, แท็บแล็ตทำกิจกรรมต่าง ๆ ไม่ว่าจะเป็นการแชท, ส่งเมล, เล่นเกม ดูคลิปวิดีโอ เป็นต้น

บ่อยครั้งที่เราใช้งาน App ต่าง ๆ ซึ่งมีอยู่มากมายหลายประเภท แล้วเกิดความคิดที่ว่า เราอยากสร้าง App ที่มีการทำงานแบบนี้บ้าง, มีขั้นตอนที่เป็นมิตรต่อผู้ใช้งาน, มีส่วนแสดงผลที่รองรับความต้องการของเรา ณ เวลานั้น ๆ ได้อย่างลงตัว

หนังสือเล่มนี้เป็นการนำเสนอเนื้อหาการพัฒนา Android Apps ด้วย Eclipse เริ่มต้นตั้งแต่พื้นฐานการสร้าง App ง่าย ๆ เรียงลำดับไปจนกระทั่งถึงการนำเสนอเทคนิคต่าง ๆ ที่เห็นใน App มาตรฐานที่เราคุ้นเคย และใช้งานกันเป็นอย่างดี ไม่ว่าจะเป็น Google Map, Facebook, Instagram, Youtube เป็นต้น

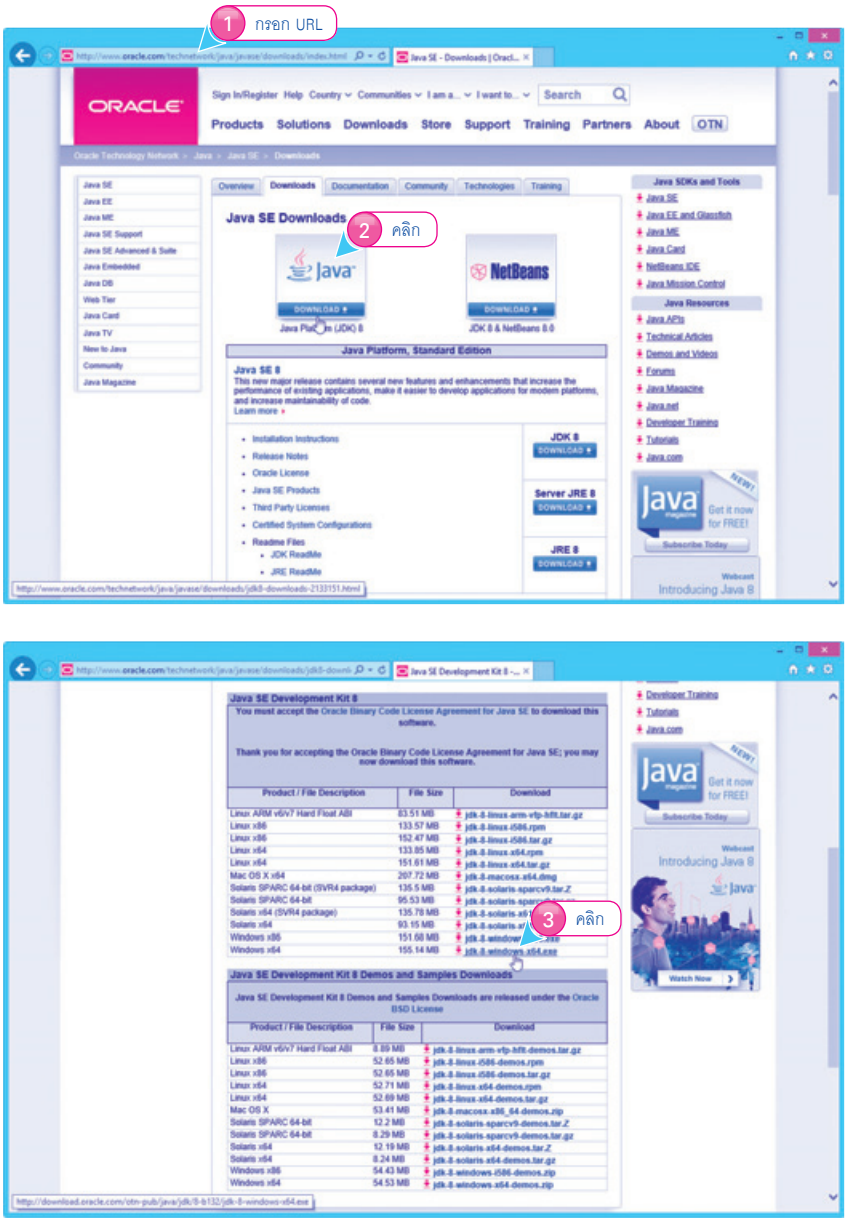
ผู้อ่านจะได้ศึกษาว่า เทคนิคและการทำงานบางส่วนของ App มาตรฐานเหล่านี้ มีขั้นตอน มีที่มาและมีวิธีการอย่างไร โดยที่ผู้เขียนตั้งโจทย์ของหนังสือเล่มนี้ไว้ว่า “เป็นเนื้อหาที่ใช้งานได้จริง”

รูปภาพแสดงผลการทำงานที่ลงในหนังสือเล่มนี้ โดยส่วนใหญ่ผู้เขียนทดสอบในเครื่องจริง เพื่อให้คุณเห็นส่วนแสดงผลที่แท้จริง

การดาวน์โหลดและติดตั้ง JAVA SDK

เริ่มต้นให้ผู้อ่านไปที่เว็บไซต์ http://www.oracle.com/technetwork/java/javase/downloads/index.html ดังรูปที่ 1-1

รูปที่ 1-1 แสดงการดาวน์โหลด Java SDK



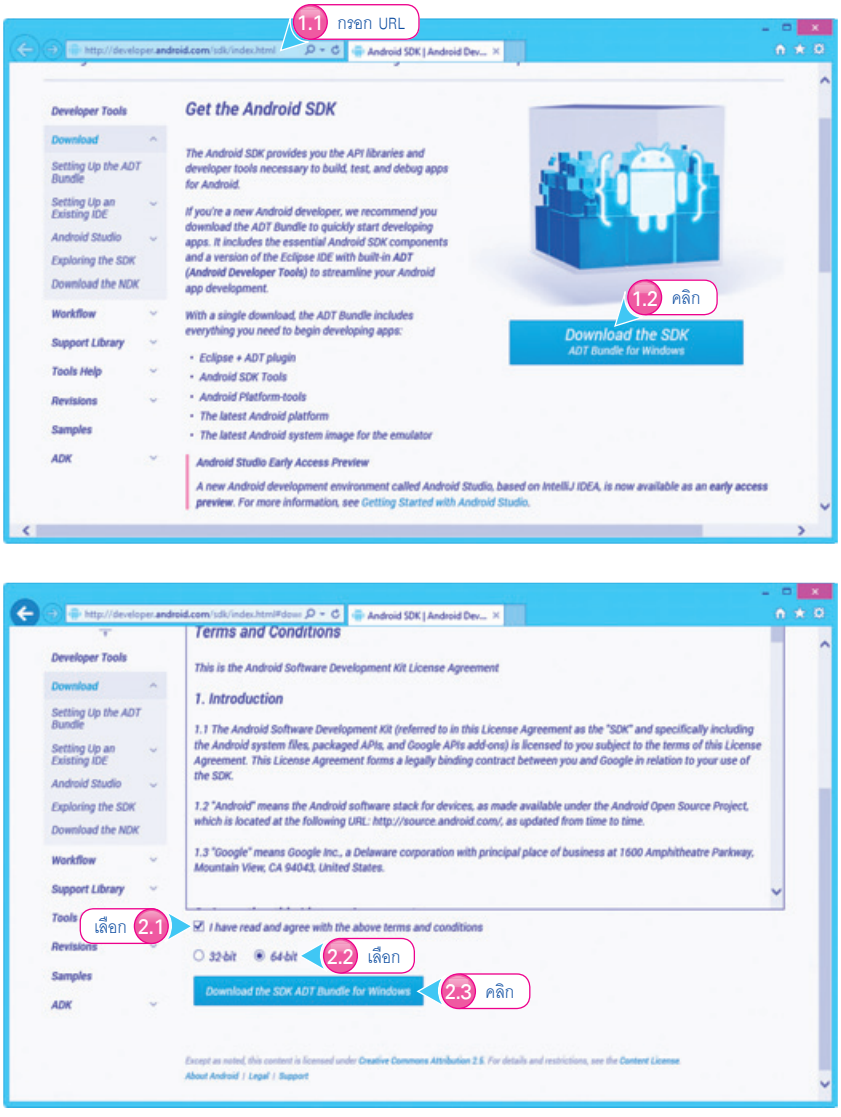
จากรูปที่ 1-1 ให้ผู้อ่านดาวน์โหลด Java SDK ตามระบบปฏิบัติการของผู้อ่าน ในกรณีนี้ผู้เขียนดาวน์โหลดแบบ Windows 64 บิต (x64) จากนั้นขอให้ผู้อ่านติดตั้ง Java SDK ตามขั้นตอนที่ปรากฏขึ้นมาจนเสร็จสมบูรณ์

การดาวน์โหลดและติดตั้งโปรแกรม Eclipse

การพัฒนา Android Apps ผู้เขียนเลือกใช้โปรแกรม Eclipse เป็นหลัก มีขั้นตอนการดาวน์โหลดและติดตั้ง ดังนี้

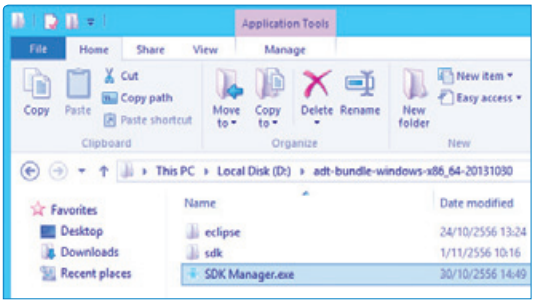
1. ให้ผู้อ่านไปที่เว็บไซต์ http://developer.android.com/sdk/index.html แล้วคลิกปุ่ม Download the SDK เพื่อดาวน์โหลดโปรแกรม Eclipse แบบ Bundle (กึ่งสำเร็จรูป)
2. ให้ผู้อ่านเลือกดาวน์โหลดโปรแกรม Eclipse ตามระบบปฏิบัติการที่ผู้อ่านใช้งาน ในกรณีนี้ผู้เขียนเลือกแบบ 64 บิต

รูปที่ 1-2 แสดงการดาวน์โหลดโปรแกรม Eclipse แบบ Bundle



3. ได้ไฟล์ดังรูป

รูปที่ 1-3 แสดงการดาวน์โหลด Eclipse แบบ 64 บิต

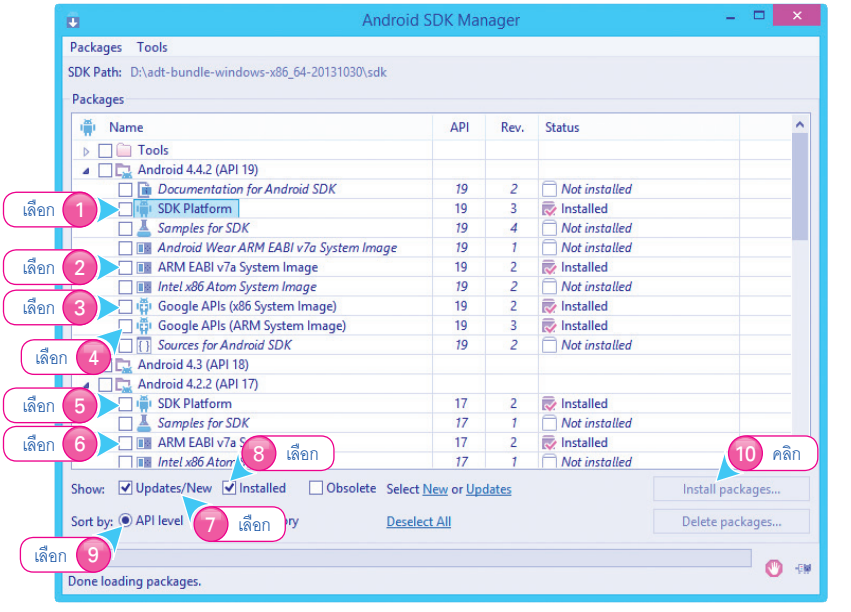


จากรูปที่ 1-3 หลังจากที่ผู้อ่านดาวน์โหลดเสร็จแล้วขอให้ขยายไฟล์ zip ที่ได้มา ในกรณีนี้ผู้เขียนเก็บไว้ที่พาธ D:\

การดาวน์โหลดและติดตั้ง Android SDK

ขั้นตอนต่อมาเป็นการดาวน์โหลดและติดตั้ง Android SDK โดยการดับเบิลคลิกที่ SDK Manager.exe จะปรากฏไดอะล็อกบ็อกซ์ Android SDK Manager และให้ผู้อ่านเลือกการติดตั้งดังรูปที่ 1-4

รูปที่ 1-4 แสดงการดาวน์โหลดและติดตั้ง Android SDK



จากรูปที่ 1-4 ผู้อ่านสามารถเลือกดาวน์โหลด Android SDK ตามเวอร์ชันที่ผู้อ่านสนใจได้อย่างอิสระ โดยการคลิกเลือกรายการให้มีเครื่องหมายถูก จากนั้นคลิกปุ่ม Install 11 packages... ส่วนรายการที่มีคำว่า Installed หมายถึง มีการติดตั้งในเครื่องของผู้อ่านแล้ว

ในขั้นต้นนี้ผู้เขียนแนะนำว่า การดาวน์โหลดและติดตั้ง Android SDK ในแต่ละเวอร์ชันควรที่จะประกอบด้วยรายการ SDK Platform, ARM EABI v** System Image และ Google APIs โดยพบว่า ณ เวลาที่เรียบเรียงหนังสือเล่มนี้สามารถพัฒนาได้ถึง Android เวอร์ชัน 4.4 (KitKat)

ในกรณีนี้ผู้เขียนเลือกดาวน์โหลดและติดตั้ง Android SDK เวอร์ชัน 4.4, 4.2 และเวอร์ชัน 2.2 เพื่อพัฒนา Android Apps ตั้งแต่เวอร์ชัน 2.2 ถึงเวอร์ชัน 4.4

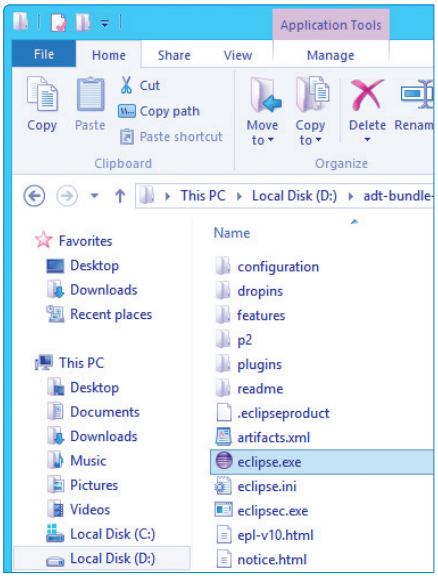
เมื่อใดก็ตามที่ Android มีเวอร์ชันใหม่ออกมา ผู้อ่านสามารถดาวน์โหลด Android SDK เวอร์ชันใหม่เพิ่มเติมได้ตามที่ต้องการได้

ในกรณีที่ต้องการพัฒนา Android Apps ให้ครอบคลุมเวอร์ชัน 4.4.x ขึ้นไป ผู้เขียนแยกอธิบายรายละเอียดในภาคผนวก

การเปิดโปรแกรม Eclipse

ในกรณีที่ผู้อ่านต้องการเปิดโปรแกรม Eclipse ให้ผู้อ่านเข้ามาที่โฟลเดอร์ eclipse ดับเบิลคลิกไฟล์ eclipse.exe ดังรูปที่ 1-5

รูปที่ 1-5 แสดงวิธีการเปิดโปรแกรม Eclipse



สรุปท้ายบท

เนื้อหาในบทนี้เป็นเพียงการเตรียมสภาพแวดล้อมให้พร้อมก่อนพัฒนา Android Apps ด้วย Eclipse โดยที่ผู้เขียนจะนำเสนอแยกตามหัวข้อในแต่ละบทต่อไป

CHAPTER 2

ความรู้เบื้องต้นในการพัฒนา Android Apps

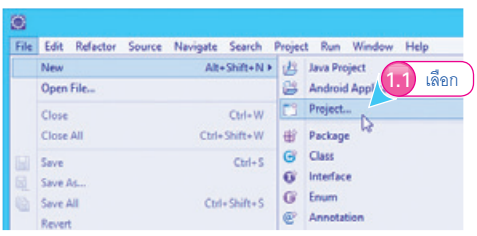
หลังจากที่ผู้อ่านเตรียมสภาพแวดล้อมในเครื่องให้พร้อมแล้วก็จะเข้าสู่เนื้อหาการพัฒนา Android App โดยที่ผู้เขียนจะเริ่มลำดับจากเนื้อหาที่ง่ายที่สุดก่อน เพื่อเป็นการปูพื้นฐานก่อนเข้าสู่การพัฒนา App ที่มีการเรียกใช้งานฟีเจอร์ต่าง ๆ ของ Android SDK เพิ่มมากขึ้น

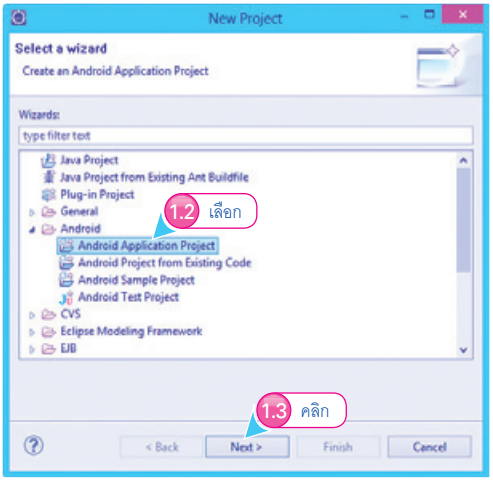
ขั้นตอนการสร้างโปรเจ็กต์ในโปรแกรม Eclipse

เริ่มต้นเราก็ต้องสร้างโปรเจ็กต์สำหรับใช้ในการพัฒนา Android App ก่อน โดยมีขั้นตอนง่าย ๆ ดังนี้

1. ให้ผู้อ่านเปิดโปรแกรม Eclipse ขึ้นมา แล้วคลิกเมนู File > Project ... > Android > Android Appslication Project เพื่อเริ่มต้นสร้างโปรเจ็กต์ว่าง ๆ ขึ้นมา ดังรูปที่ 2-1

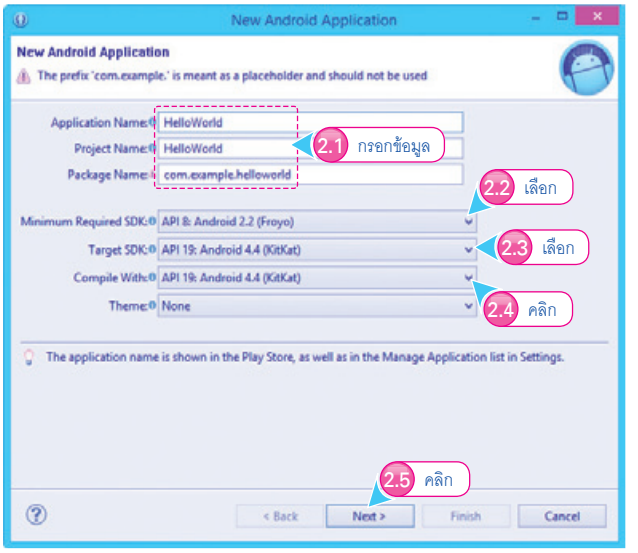
รูปที่ 2-1 แสดงการเริ่มต้นสร้าง Android Project แบบว่างๆ ขึ้นมา





2. เข้าสู่ขั้นตอนการกำหนดรายละเอียดของ Android Apps ในขั้นต้น โดยที่
   - **ช่อง Application Name** หมายถึง ชื่อ App ของเรา ในกรณีนี้คือ HelloWorld
   - **ช่อง Project Name** หมายถึง ชื่อ Android Project ปัจจุบันมีข้อแนะนำว่า ควรมีชื่อเดียวกับ App
   - **ช่อง Package Name** หมายถึง ชื่อ Package ของ Android Project ปัจจุบัน มีข้อแนะนำให้ใช้ชื่อที่ Eclipse แนะนำ

รูปที่ 2-2 แสดงการกำหนดรายละเอียดขั้นต้นให้กับ Android Project ปัจจุบัน



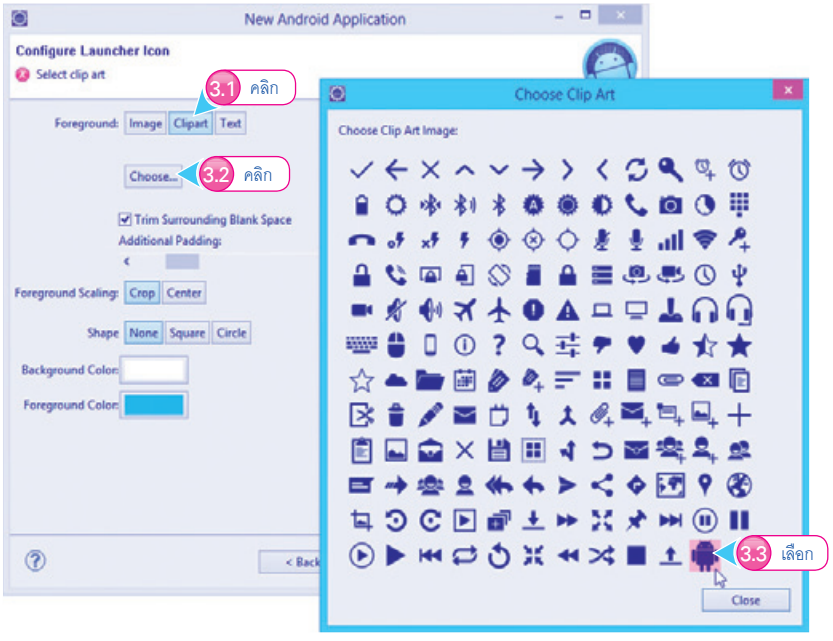
   - **ช่อง Minimum Required SDK** ในกรณีที่ผู้อ่านคาดว่า Android Apps ของเราครอบคลุมระบบปฏิบัติการ Android หลายเวอร์ชัน ขอให้ผู้อ่านระบุเวอร์ชันของระบบปฏิบัติ Android ที่ต่ำสุด ที่สามารถรัน Android Apps ของเราได้ก็จะทำให้ App ของเรารองรับหลายเวอร์ชันนั่นเอง
   - **ช่อง Target SDK** หมายถึง เวอร์ชันของระบบปฏิบัติการ Android ที่เป็นเป้าหมายหลักให้ App ของเรารัน
   - **ช่อง Compile With** หมายถึง ผู้อ่านต้องการทดสอบรันและคอมไพล์โปรเจ็กต์ปัจจุบันด้วยเวอร์ชันใด

NOTE

ผู้อ่านสามารถระบุเวอร์ชันที่รองรับได้ถึง Android 4.4 (KitKat) ในโลกของความเป็นจริง Android Apps ที่สร้างขึ้นมา ควรที่จะรองรับและสามารถติดตั้งบนระบบปฏิบัติการ Android หลายเวอร์ชันมากที่สุดเท่าที่เป็นไปได้ ขึ้นอยู่กับว่าในโปรเจ็กต์ปัจจุบันของเรา เรียกใช้ฟีเจอร์ใดบ้างของ Android SDK ก็จะมีข้อกำหนดแตกต่างกันไปตามแต่ละเวอร์ชัน

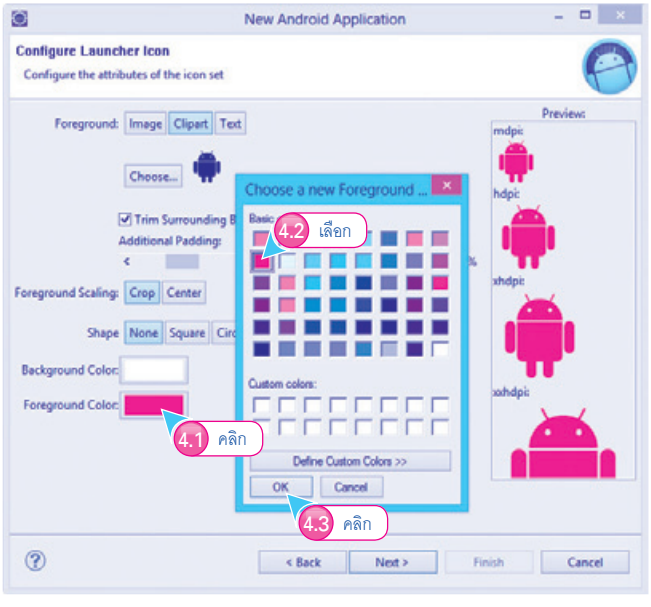
3. เข้าสู่ขั้นตอนการปรับแต่ง Android Apps ปัจจุบัน ให้ผู้อ่านคลิกแท็บ Clipart และคลิกปุ่ม Choose... เพื่อเลือกไอคอนสำเร็จรูปที่มากับ Eclipse ให้กับโปรเจ็กต์ปัจจุบัน ดังรูปที่ 2-3

รูปที่ 2-3 แสดงการเลือกไอคอนสำเร็จรูปให้กับ Android Apps ปัจจุบัน



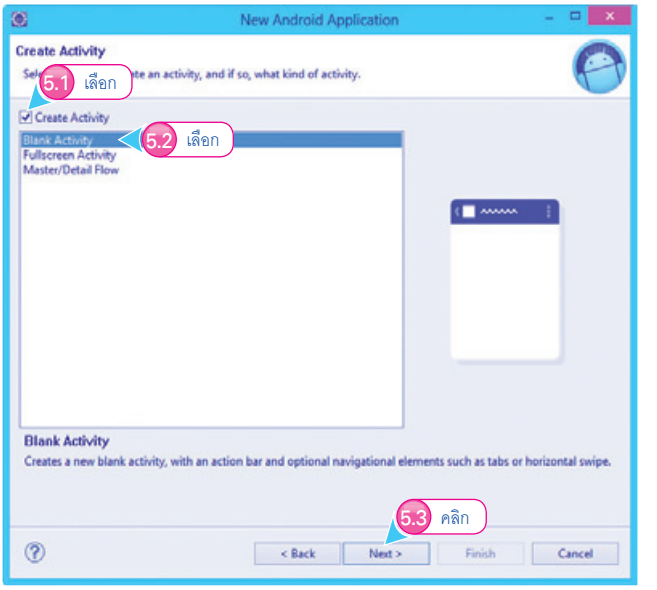
4. ผู้อ่านสามารถเลือกสีของไอคอนได้ โดยการเลือกสีที่ Foreground Color ดังรูปที่ 2-4

รูปที่ 2-4 แสดงการเลือกสีของไอคอน



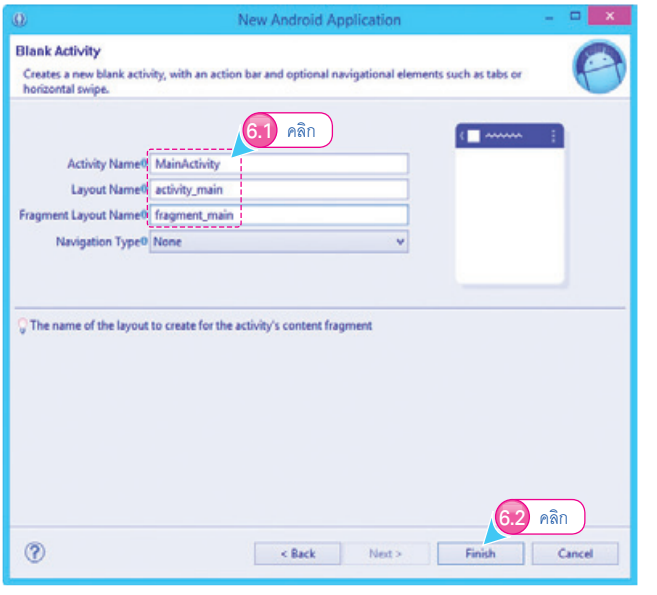
5. การเลือกประเภทของ App ที่ต้องการสร้างขึ้นมา ในขั้นต้นนี้เราต้องการ App ที่มีส่วนแสดงผลแบบหน้าจอเดียว จึงเลือกแบบ **Blank Activity**

รูปที่ 2-5 แสดงการเลือกประเภท App ที่ต้องการสร้างขึ้นมา

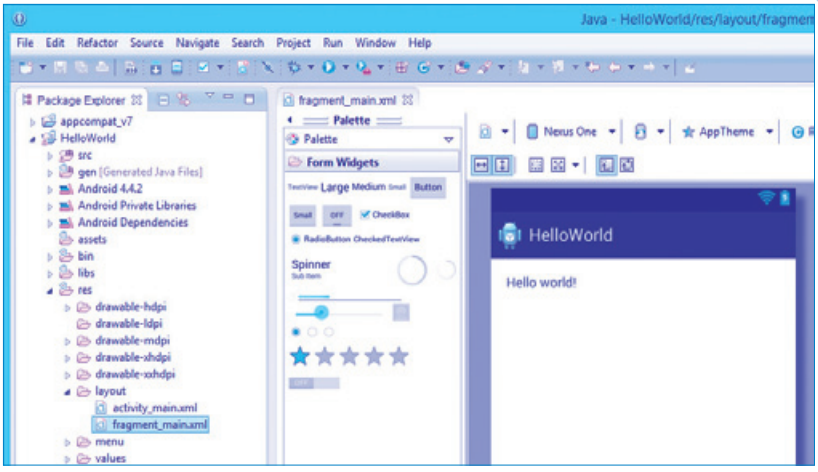


6. กำหนดรายละเอียดขั้นต้นของประเภท App ปัจจุบันที่น่าสนใจคือ
   - **ช่อง Activity Name** หมายถึง ชื่อส่วนแสดงผลที่จะปรากฏขึ้นมาเป็นลำดับแรก ในกรณีนี้ชื่อว่า MainActivity
   - **ช่อง Layout Name** หมายถึง Layout ของส่วนแสดงผลปัจจุบันคือ ไฟล์ที่ชื่อว่า activity_main (.xml) ที่เก็บอยู่ในโฟลเดอร์ layout ของโปรเจ็กต์ปัจจุบัน

รูปที่ 2-6 แสดงการกำหนดชื่อ App ปัจจุบันว่า Hello World



รูปที่ 2-6 (ต่อ) แสดงการกำหนดชื่อ App ปัจจุบันว่า Hello World



ณ จุดนี้โปรแกรม Eclipse จะสร้าง Android Project ว่างๆ ขึ้นมาให้เรียบร้อยแล้ว ส่วนแสดงผลหลักที่ปรากฏขึ้นมาเป็นลำดับแรก เกิดจากไฟล์ที่ชื่อว่า fragment_main.xml เก็บอยู่ในโฟลเดอร์ \res\layout

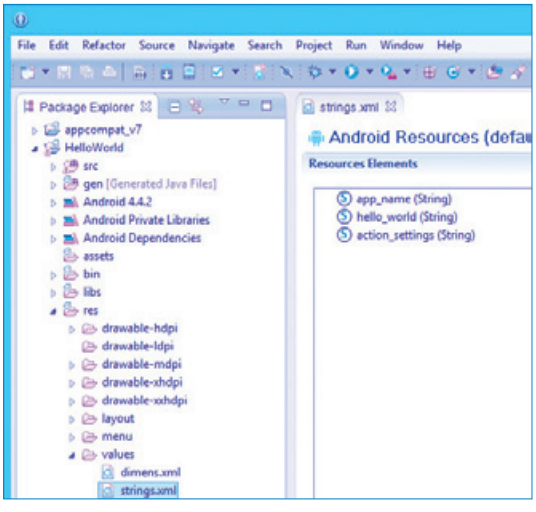
ในปัจจุบัน Android Apps กำหนดให้ใช้ Layout แบบ **fragment** หมายถึง การสร้างส่วนแสดงผลที่สามารถแสดงผลได้ทั้งมือถือและแท็บเล็ตในคราวเดียวกันอยู่ในความรับผิดชอบของไฟล์ fragment_main.xml (\res\layout\fragment_main.xml)

ตัวอย่างการออกแบบส่วนแสดงผลอย่างง่าย

เมื่อสร้างโปรเจ็กต์เรียบร้อยแล้ว ต่อไปก็เป็นขั้นตอนการออกแบบส่วนแสดงผลของ App โดยแต่ละไฟล์มีการทำงานแตกต่างกันไป ซึ่งมีรายละเอียดดังนี้

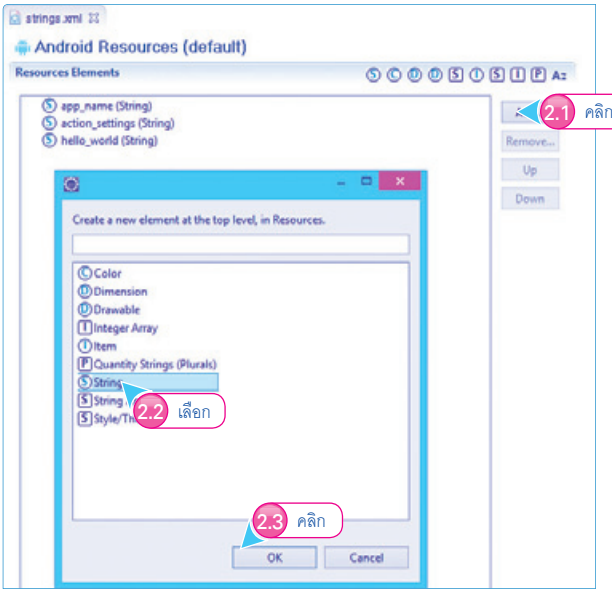
1. เริ่มต้นให้ผู้อ่านลบข้อความ Hello world ที่มากับโปรเจ็กต์ออก จากนั้นเปิดไฟล์ strings.xml ที่พาธ \res\values ของโปรเจ็กต์ปัจจุบัน เพื่อกำหนดข้อความที่ต้องการใช้งานก่อน เพราะว่า Android Project ถือว่า ข้อความ (string) เป็นทรัพยากรประเภทหนึ่งของโปรเจ็กต์ปัจจุบัน ดังรูปที่ 2-7

รูปที่ 2-7 แสดงไฟล์ strings.xml ในโฟลเดอร์ \res\values

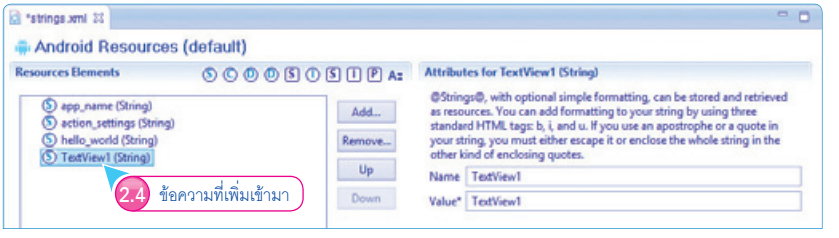


2. โปรเจ็กต์ปัจจุบันต้องการใช้ 2 ข้อความ ให้ผู้อ่านคลิกปุ่ม Add... เพื่อเพิ่มข้อความใหม่เข้ามาในโปรเจ็กต์ปัจจุบัน

รูปที่ 2-8 แสดงการเพิ่มข้อความที่ชื่อว่า TextView1



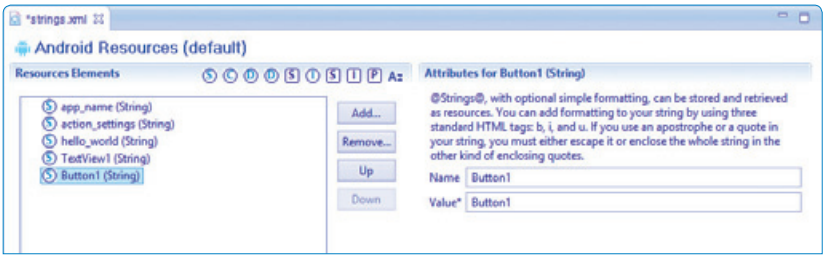
รูปที่ 2-8 (ต่อ) แสดงการเพิ่มข้อความที่ชื่อว่า TextView1



ข้อความที่ชื่อว่า TextView1 (ช่อง Name) ทำหน้าที่เก็บข้อความ “TextView1” (ช่อง Value*)

3. ให้ผู้อ่านเพิ่มอีก 1 ข้อความตั้งชื่อว่า Button1 ดังรูปที่ 2-9

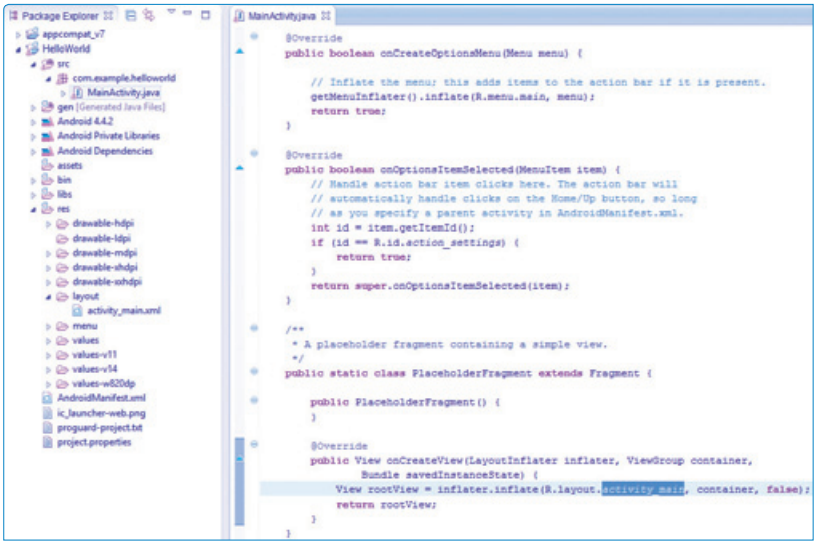
รูปที่ 2-9 แสดงการเพิ่มข้อความที่ชื่อว่า Button1



จากรูปที่ 2-9 ข้อความที่ชื่อว่า Button1 (ช่อง Name) ทำหน้าที่เก็บข้อความ “Button1” (ช่อง Value*) ณ จุดนี้ในโปรเจ็กต์ปัจจุบันของเรามีข้อความใหม่ 2 ข้อความพร้อมให้เรียกใช้งานแล้ว

4. ตัวอย่างที่นำเสนอในหนังสือเล่มนี้ ไม่ได้ใช้ฟีเจอร์ fragment ดังนั้น ขอให้ผู้อ่านลบไฟล์ fragment_main.xml ออกจากโฟลเดอร์ \res\layout จากนั้นแก้ไขโค้ดในไฟล์ MainActivity.java

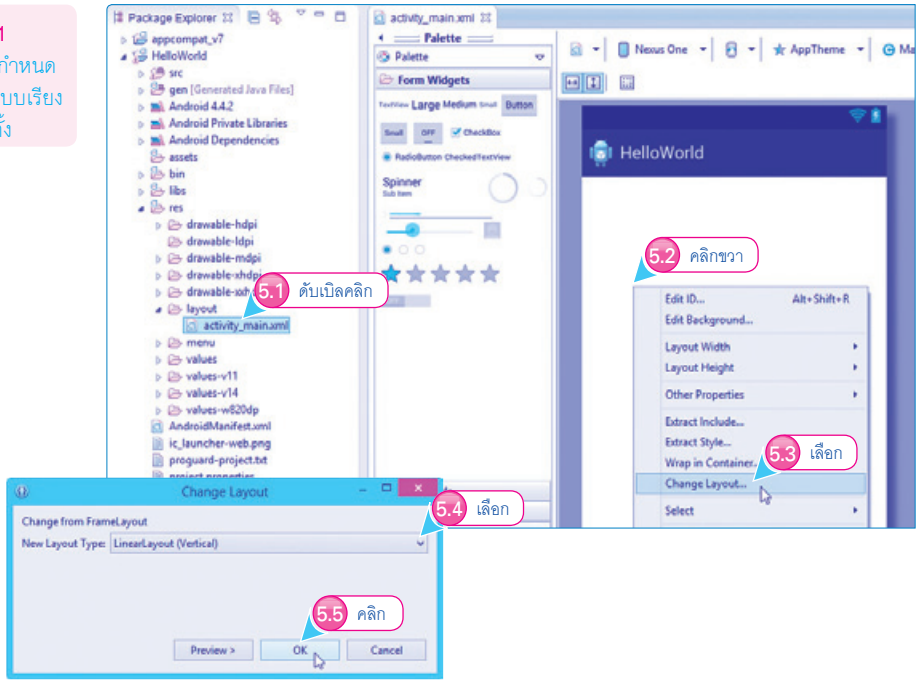
รูปที่ 2-10 แสดงการแก้ไขโค้ดในไฟล์ MainActivity.java



จากรูปที่ 2-10 ผู้เขียนกำหนดให้โปรเจ็กต์กลับมาอ่านโครงสร้างส่วนแสดงผลจากไฟล์ activity_main

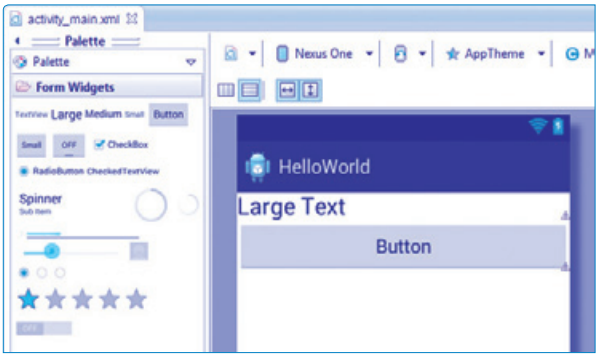
5. กำหนด Layout ให้กับส่วนแสดงผล activity_main.xml (\res\layout\activity_main.xml) ในขั้นต้นนี้กำหนดให้ใช้ Layout แบบเรียงตามแนวตั้ง

รูปที่ 2-11 แสดงการกำหนด Layout แบบเรียงตามแนวตั้ง



6. ให้ผู้อ่านออกแบบดังรูปที่ 2-12

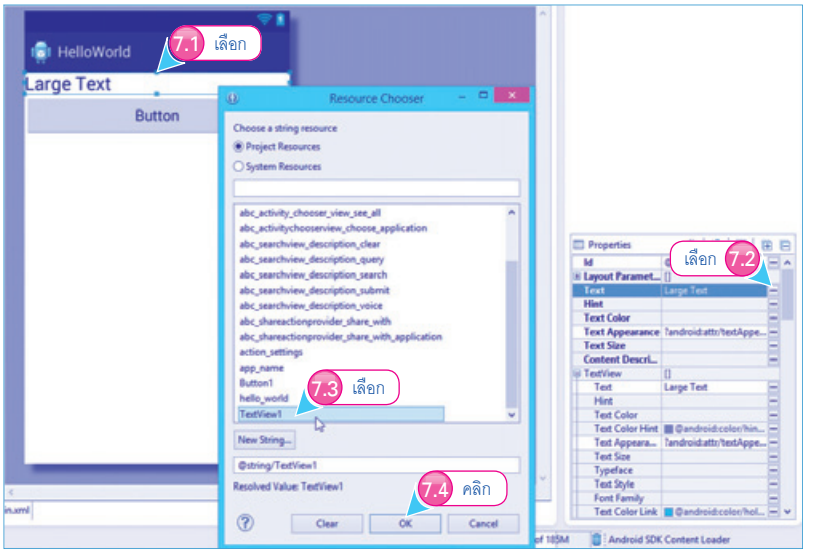
รูปที่ 2-12 ส่วนแสดงผลในขณะออกแบบ



จากรูปที่ 2-12 เป็นการใช้งาน widget ประเภท TextView แบบ Large (ทำหน้าที่รับหรือแสดงข้อความที่มีขนาดตัวอักษรใหญ่กว่าปกติ) และ widget ประเภท Button (ทำหน้าที่เป็นปุ่มกดจากการสัมผัสหน้าจอ)

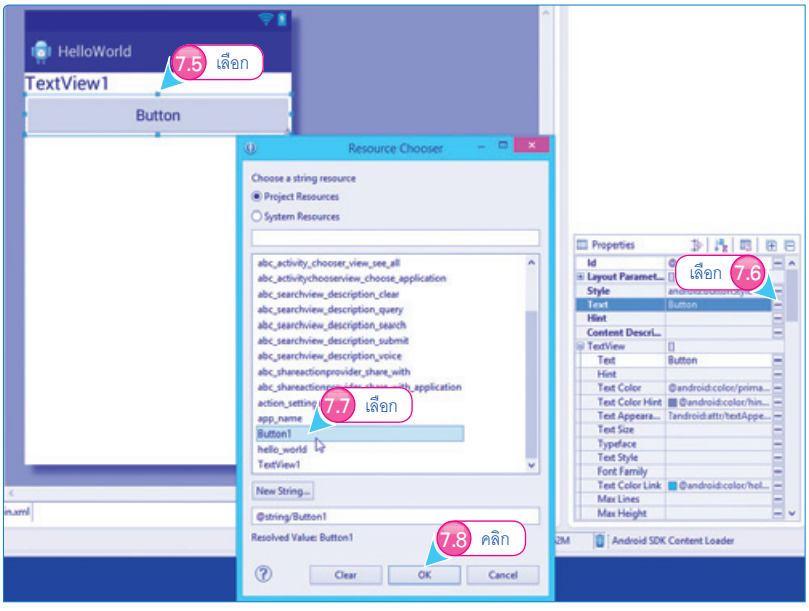
7. เราจะกำหนดให้ widget ประเภท TextView ใช้ข้อความที่ชื่อว่า TextView1 และ widget ประเภท Button ใช้ข้อความที่ชื่อว่า Button1 ที่เราสร้างไว้ในขั้นตอนที่แล้ว ดังรูป

รูปที่ 2-13 แสดงการกำหนดให้ widget ประเภท TextView ใช้ข้อความที่ชื่อว่า TextView1



จากรูปที่ 2-13 ให้กำหนดที่คุณสมบัติ Text ในหน้าต่าง Properties

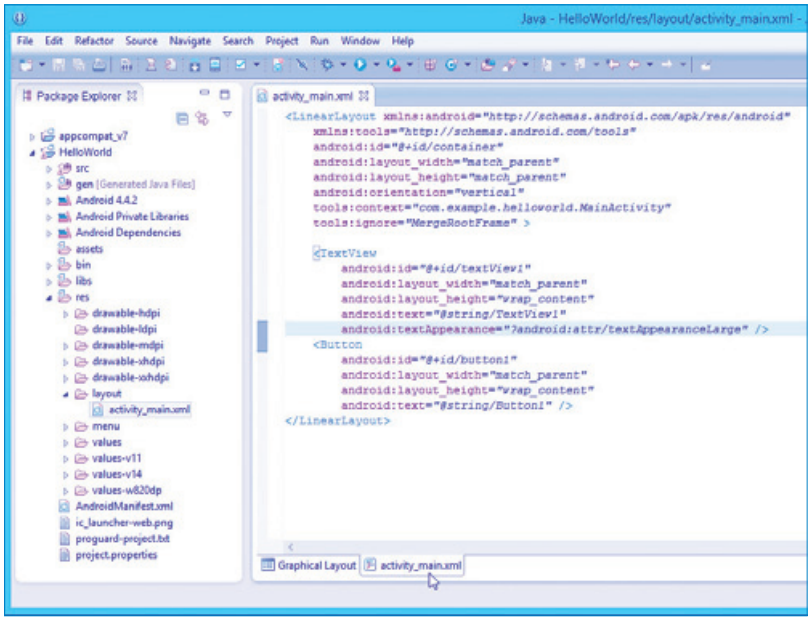
รูปที่ 2-14 แสดงการกำหนดให้ widget ประเภท Button ใช้ข้อความที่ชื่อว่า Button1



เป็นการกำหนดให้ widget ทั้ง 2 ตัวใช้ข้อความที่มาจากทรัพยากรของโปรเจ็กต์ปัจจุบัน

8. ผู้อ่านสามารถคลิกที่แท็บ activity_main.xml เพื่อแสดงส่วนแสดงผลแบบโครงสร้าง XML ดังรูปที่ 2-15

รูปที่ 2-15 แสดงโครงสร้าง XML ของส่วนแสดงผลปัจจุบัน



ในขณะที่ออกแบบส่วนแสดงผล โปรแกรม Eclipse จะสร้างสคริปต์ XML เหล่านี้ให้โดยอัตโนมัติ หมายความว่า ส่วนแสดงผลใน Android Apps ใช้โครงสร้างของ XML เข้ามาทำหน้าที่รับผิดชอบส่วนแสดงผลให้นั่นเอง

ในทางกลับกันเราสามารถแก้ไขสคริปต์ XML เหล่านี้เพื่อแก้ไขส่วนแสดงผลได้เช่นกัน

สคริปต์ XML ที่ 2-1 Hello World Apps (\res\layout\activity_main.xml)

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context="com.example.helloworld.MainActivity"
    tools:ignore="MergeRootFrame" >

    <TextView
        android:id="@+id/textView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/TextView1"
        android:textAppearance="?android:attr/textAppearanceLarge" />
    <Button
        android:id="@+id/button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/Button1" />
</LinearLayout>
```

9. เขียนโค้ด JAVA ในไฟล์ MainActivity.java ดังต่อไปนี้

โค้ด JAVA ที่ 2-1 Hello World Apps (\src\com.example.hello.android\MainActivity.java)

```
package com.example.helloworld;

import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends ActionBarActivity implements android.view.View.OnClickLis-
tener  {

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

                 Button cmd = (Button) findViewById(R.id.button1);
                 cmd.setOnClickListener(this);
       }
```

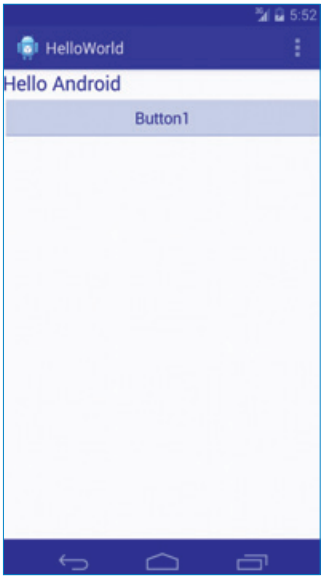
```
        public void onClick(View v) {
                TextView tv =(TextView) findViewById(R.id.textView1);
                tv.setText("Hello Android");
        }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.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);
    }
}
```

10. ทดสอบการทำงานได้ผลลัพธ์ดังรูป

รูปที่ 2-16
ผลการรัน
ตัวอย่างที่ 2-1



จากรูปที่ 2-16 เมื่อคลิกปุ่ม Button1 ช่องรับข้อความจะแสดงข้อความ “Hello Android” ตามที่เราต้องการแล้ว

อธิบายการทำงานของโค้ด

1. เริ่มต้นแพ็คเกจปัจจุบันมีชื่อว่า com.example.hello.android เกิดขึ้นในขณะที่สร้างโปรเจ็กต์ขึ้นมาในโปรแกรม Eclipse จากนั้นให้ระบุไลบรารีต่อไปนี้ให้ครบถ้วน

\src\com.example.hello.android\MainActivity.java

```
package com.example.helloworld;

import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
```

2. โปรเจ็กต์ปัจจุบันถูกกำหนดให้มีส่วนแสดงผลแบบหน้าจอเดียว มี Activity เดียวชื่อว่าคลาส MainActivity.java ให้ผู้อ่านกำหนดให้มีรองรับ (implements) เหตุการณ์คลิกด้วย (View.OnClickListener)

\src\com.example.hello.android\MainActivity.java

```
public class MainActivity extends ActionBarActivity implements android.view.View.OnClickLis-
tener  {
       @Override
       protected void onCreate(Bundle savedInstanceState) {
                 super.onCreate(savedInstanceState);
                 setContentView(R.layout.activity_main);
```

ใน MainActivity เหตุการณ์ onCreate() จะเกิดขึ้นมาเป็นลำดับแรก (เป็นเหตุการณ์ที่เกิดขึ้นก่อนแสดงผลให้ผู้ใช้เห็น) ผู้อ่านต้องสร้างส่วนแสดงผลใน App ให้พร้อมใช้งานเสียก่อนในเหตุการณ์นี้

โดยที่โปรแกรม Eclipse จะกำหนดให้ใช้ส่วนแสดงผลที่ชื่อว่า activity_main (อยู่ในไฟล์ activity_main.xml) ผ่านทางเมธอดที่ชื่อว่า setContentView() ผ่านทางคลาส R

คลาส R ถูกสร้างขึ้นมาโดยอัตโนมัติทำหน้าที่อ้างอิงสิ่งต่าง ๆ ที่อยู่ในโปรเจ็กต์ปัจจุบัน ในกรณีนี้เป็นการอ้างอิงไฟล์ที่ทำหน้าที่สร้างส่วนแสดงผลจึงระบุว่า R.layout.activity_main

3. ให้สร้างเหตุการณ์ Click() ขึ้นมารองรับการกดปุ่ม Button ที่ชื่อว่า button1 โดยที่ใช้เมธอดที่ชื่อว่า findViewById() ทำหน้าที่ค้นหาออบเจ็กต์ที่ชื่อว่า button1 ผ่านทางคลาส R.id ที่อยู่ในส่วนแสดงผล ผลการค้นหาที่ได้ให้เก็บไว้ในตัวแปรออบเจ็กต์ Button ที่ชื่อว่า cmd ก่อน

จากนั้นสร้างเหตุการณ์ Click() ให้กับตัวแปรออบเจ็กต์ cmd ผ่านทางเมธอด setOnClickListener() ให้กับส่วนแสดงผลปัจจุบัน (this)

\src\com.example.hello.android\MainActivity.java

```
        Button cmd = (Button) findViewById(R.id.button1);
        cmd.setOnClickListener(this);
    }
```

4. ที่เหตุการณ์ onClick() สร้างตัวแปรออบเจ็กต์ TextView ที่ชื่อว่า tv ขึ้นมารับค่าจากการค้นหาออบเจ็กต์จากส่วนแสดงผลที่ชื่อว่า textView1 กำหนดข้อความผ่านทางเมธอด setText()