

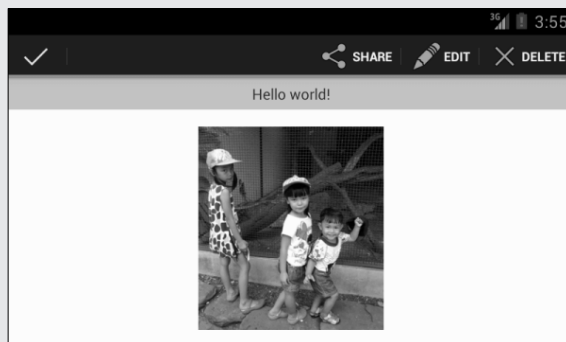
## CHAPTER

## 03

# Menu IIละ: Action Bar

## เนื้อหาในบทนี้

- ◆ การสร้างเมนู (Options Menu)
- ◆ การสร้างคอนเท็กซ์เมนู (Context Menu)
- ◆ การสร้างคอนเท็กซ์เมนูสำหรับ ListView
- ◆ การสร้างคอนเท็กซ์เมนูแบบ Contextual Action Bar



- ◆ การสร้างป๊อปอัพเมนู (Popup Menu)

## การสร้างเมนู (Options Menu)


เมนูในแอนดรอยด์แบ่งเป็น 3 ประเภทหลักๆ ได้แก่ Options Menu, Context Menu และ Popup Menu ในหัวข้อนี้จะพูดถึง Options Menu ก่อน

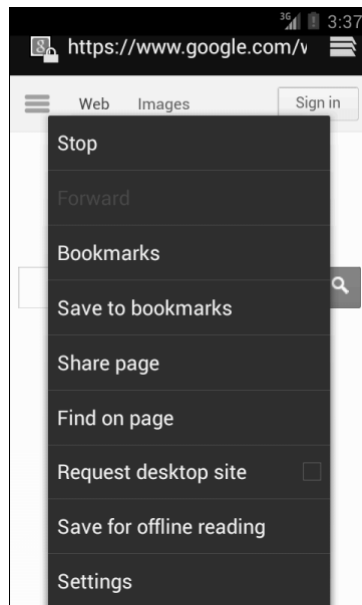
*Options Menu* คือเมนูที่รวบรวมตัวเลือก/คำสั่งที่เกี่ยวข้องกับหน้าจอ (แอกทิวิตี) ปัจจุบัน ผู้ใช้สามารถแสดง Options Menu ได้โดยกดหรือแตะปุ่ม Menu ที่ตัวเครื่อง (บางเครื่องเป็นปุ่มจริง บางเครื่องเป็นปุ่มสัมผัส) ซึ่งเมนูจะแสดงออกมาทางด้านล่างของหน้าจอ

รูปแบบของ Options Menu ที่แสดงออกมาจะขึ้นกับเวอร์ชันของแอนดรอยด์ด้วย โดยสำหรับแอนดรอยด์ 2.3 (API Level 10) หรือต่ำกว่า รายการตัวเลือกของ Options Menu จะแสดงด้านล่างหน้าจอ ดังรูป ซึ่งแสดงได้สูงสุด 6 ตัวเลือก กรณีมีมากกว่า 6 ตัวเลือก (เช่นในรูป) ผู้ใช้จะต้องเลือก More เพื่อเข้าถึงตัวเลือกอื่นๆที่เหลือ

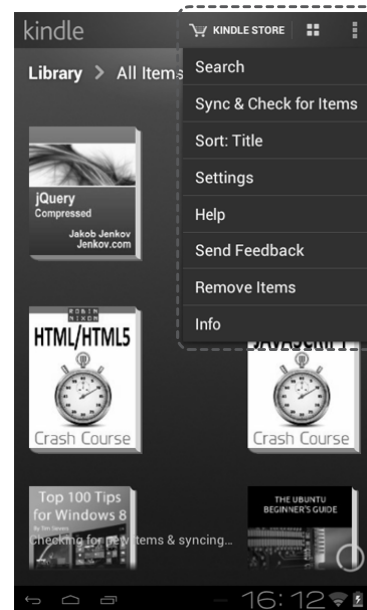


Options Menu  
ในแอนดรอยด์ 2.3 หรือต่ำกว่า

สำหรับแอนดรอยด์ 3.0 (API Level 11) ขึ้นไป รายการตัวเลือกของ Options Menu จะแสดงออกมาด้านล่างโดยเรียงเป็นแถวเดียวจากบนลงล่าง ดังรูปซ้าย นอกจากนี้เรายังกำหนดให้แสดงบน Action Bar ดังรูปขวา (หน้าถัดไป) ซึ่งจะช่วยให้ผู้ใช้เข้าถึงตัวเลือกต่างๆได้อย่างรวดเร็ว ทั้งนี้หากพื้นที่บน Action Bar ไม่พอแสดงตัวเลือกทั้งหมด จะปรากฏไอคอน  ทางขวาของ Action Bar สำหรับเข้าถึงตัวเลือกอื่นๆที่เหลือ หรือหากไม่ปรากฏไอคอนนี้ก็จะต้องกดหรือแตะปุ่ม Menu ที่ตัวเครื่องเพื่อเข้าถึงตัวเลือกอื่นๆที่เหลือ



Options Menu  
ในแอนดรอยด์ 3.0 ขึ้นไป



Options Menu  
ในแอนดรอยด์ 3.0 ขึ้นไป  
และให้แสดงตัวเลือกบน Action Bar

## ตัวอย่าง

ตัวอย่างนี้จะสร้างเมนูที่มี 6 ตัวเลือกหรือ 6 ไอเท็ม (Item) และให้แสดงตัวเลือกทั้งหมดบน Action Bar ถ้ามีที่ว่างพอ (มีผลกับแอนดรอยด์ 3.0 ขึ้นไป) ซึ่งเมื่อคลิกตัวเลือกในเมนูจะแสดงข้อความออกมาที่แท็กชีวในหน้าจอหลัก

- 1 สร้างแท็กชีวขึ้นมาใน Layout File โดยกำหนด ID ว่า text

โปรเจ็ค MenuDemo, ไฟล์ res\layout\activity\_\_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp" >

    <TextView
        android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello_world" />

</RelativeLayout>
```

- 2 สร้างไฟล์ XML กำหนดตัวเลือกของเมนู (ปกติ Eclipse จะสร้างไฟล์นี้มาให้อยู่แล้ว ให้คุณเพิ่มโค้ดเข้าไปตามนี้)

โปรเจ็ค MenuDemo, ไฟล์ res/menu/main.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >

    <item
        android:id="@+id/action_add"
        android:icon="@android:drawable/ic_menu_add"
        android:showAsAction="ifRoom|withText"
        android:title="Add"/>

    <item
        android:id="@+id/action_edit"
        android:icon="@android:drawable/ic_menu_edit"
        android:showAsAction="ifRoom"
        android:title="Edit"/>

    <item
        android:id="@+id/action_save"
        android:icon="@android:drawable/ic_menu_save"
        android:showAsAction="ifRoom"
        android:title="Save"/>

    <item
        android:id="@+id/action_close"
        android:icon="@android:drawable/ic_menu_close_clear_cancel"
        android:showAsAction="ifRoom"
        android:title="Close"/>

    <item
        android:id="@+id/action_share"
        android:icon="@android:drawable/ic_menu_share"
        android:showAsAction="ifRoom"
        android:title="Share"/>

    <item
        android:id="@+id/action_search"
        android:icon="@android:drawable/ic_menu_search"
        android:showAsAction="ifRoom"
        android:title="Search"/>

</menu>
```

### 3 เพิ่มเมธอด onCreateOptionsMenu ในแอคทิวิตี้ (ปกติ Eclipse จะสร้างเมธอดนี้มาให้อยู่แล้ว)

โปรเจ็ค MenuDemo, ไฟล์ MainActivity.java

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.game_menu, menu);
    return true;
}
```

### 4 เพิ่มเมธอด onOptionsItemSelected ในแอคทิวิตี้

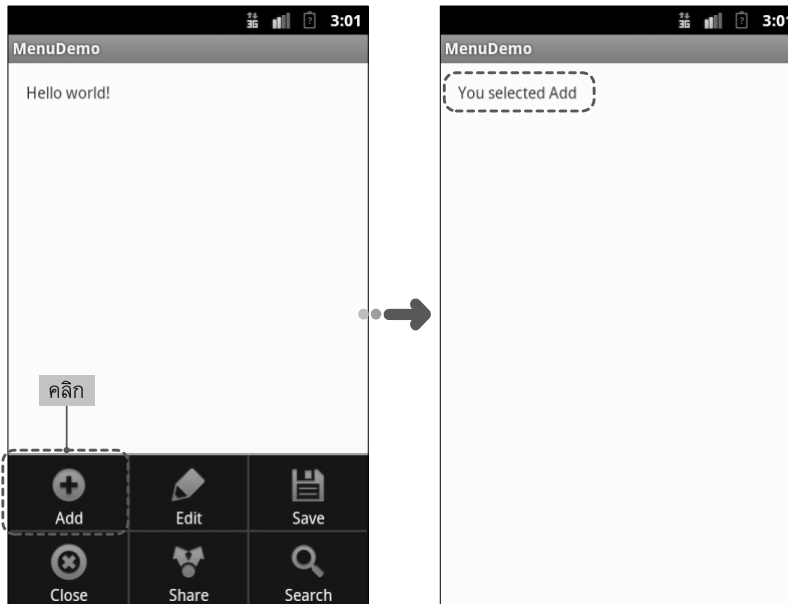
โปรเจ็ค MenuDemo, ไฟล์ MainActivity.java

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    TextView tv = (TextView) findViewById(R.id.text);

    switch (item.getItemId()) {
        case R.id.action_add:
            tv.setText("You selected Add");
            return true;
        case R.id.action_edit:
            tv.setText("You selected Edit");
            return true;
        case R.id.action_save:
            tv.setText("You selected Save");
            return true;
        case R.id.action_close:
            tv.setText("You selected Close");
            return true;
        case R.id.action_share:
            tv.setText("You selected Share");
            return true;
        case R.id.action_search:
            tv.setText("You selected Search");
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

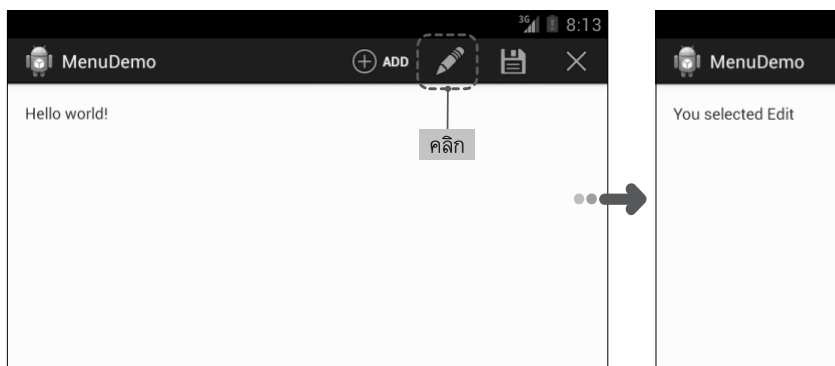
## ผลการรันบนแอนดรอยด์ 2.3

เมื่อคลิกปุ่ม Menu จะแสดงเมนูขึ้นมาด้านล่าง



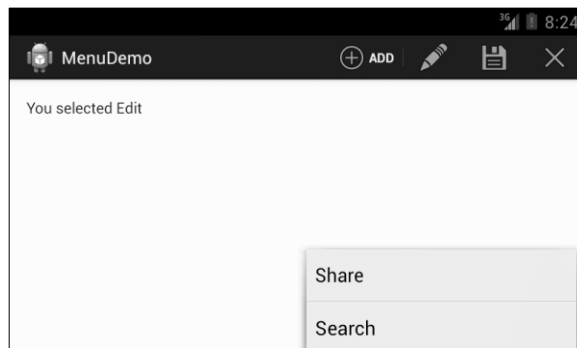
## ผลการรันบนแอนดรอยด์ 4.2

ในที่นี้จะแสดงหน้าจอแนวนอน (กด **Ctrl**+**F11** เพื่อหมุนหน้าจออิมูเลเตอร์) เพื่อให้ Action Bar มีพื้นที่แสดงตัวเลือกมากขึ้น จะได้เห็นผลอย่างชัดเจน



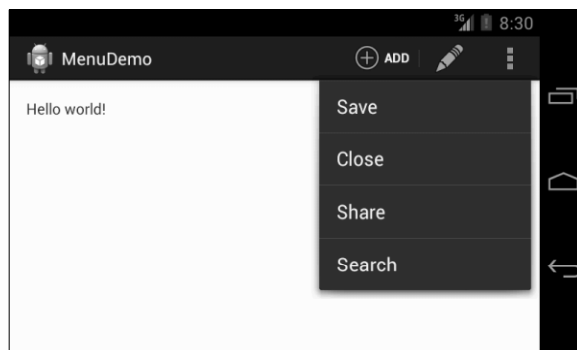
จะเห็นว่ามี 4 ตัวเลือกถูกแสดงบน Action Bar โดยที่ตัวเลือก Add มีข้อความกำกับไว้ด้วย ในขณะที่ตัวเลือกอื่นๆมีเฉพาะรูปไอคอน ไม่มีข้อความ

สำหรับอีก 2 ตัวเลือกจะเข้าถึงได้โดยการคลิกปุ่ม Menu ดังรูป



กรณีนี้แอนดรอยด์คงเห็นว่าตัวเครื่องมีปุ่ม Menu อยู่แล้ว จึงไม่แสดงไอคอน ที่ใช้เข้าถึงตัวเลือกอื่นๆที่ไม่ได้ปรากฏบน Action Bar โดยตรง

แต่หากตั้งค่าอิมูเลเตอร์ไม่ให้มีปุ่มควบคุมที่ตัวเครื่อง แอนดรอยด์ก็จะแสดงไอคอน บน Action Bar ซึ่งเมื่อคลิกปุ่มนี้จะสามารถเข้าถึงตัวเลือกอื่นๆที่เหลือได้ ดังรูป



#### NOTE»»

การตั้งค่าอิมูเลเตอร์ไม่ให้มีปุ่มควบคุมที่ตัวเครื่อง ทำได้โดยไปที่ไฟล์เดอร์

C:\Users\account\_name\.android\avd\avd\_name (account\_name คือชื่อบัญชีที่คุณใช้เข้าสู่ระบบ Windows, avd\_name คือชื่อของอิมูเลเตอร์) จากนั้นให้เปิดไฟล์ config.ini แล้วแก้ไขบรรทัด

```
hw.mainKeys=yes
```

เป็น

```
hw.mainKeys=no
```

จากนั้นให้บันทึกไฟล์ ปิดอิมูเลเตอร์ แล้วรันอิมูเลเตอร์ใหม่

## คำอธิบาย

### กำหนด Menu Resource

รายละเอียดของตัวเลือกในเมนูจะกำหนดด้วย Menu Resource ในไฟล์ XML ซึ่งเก็บอยู่ในโฟลเดอร์ res\menu โดยตัวเลือกหนึ่งๆจะกำหนดด้วยแท็ก <item> เช่น

```
<item
    android:id="@+id/action_add"
    android:icon="@android:drawable/ic_menu_add"
    android:showAsAction="ifRoom|withText"
    android:title="Add"/>
```

ความหมายของแอตทริบิวต์ทั้งสี่ข้างต้น

android:id	กำหนด ID ของตัวเลือก
android:icon	กำหนดรูปภาพไอคอนของตัวเลือก ในที่นี่ใช้รูปภาพที่แอนดรอยด์เตรียมไว้ให้ (ถ้าหากเป็นรูปภาพที่เราเตรียมมาเองในโฟลเดอร์ res\drawable จะระบุว่า @drawable/ชื่อไฟล์รูปภาพ)
android:showAsAction	ค่า ifRoom หมายถึงให้แสดงตัวเลือกนี้เป็น Action Item บน Action Bar ถ้าหากมีที่ว่างพอ ส่วนค่า withText หมายถึงให้แสดงข้อความกำกับด้วย (แอตทริบิวต์นี้มีผลในแอนดรอยด์ 3.0 ขึ้นไป)
android:title	กำหนดข้อความของตัวเลือก

### สร้างเมนูขึ้นมาจาก Menu Resource

เมื่อผู้ใช้กดหรือแตะปุ่ม Menu ที่ตัวเครื่อง แอนดรอยด์จะเรียกมายังเมธอด onCreateOptionsMenu ของแอกทิวิตี ซึ่งเราต้องใส่โค้ดที่ใช้สร้างเมนูขึ้นมาจาก Menu Resource ไว้ในเมธอดนี้

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // เข้าถึง MenuInflater
    MenuInflater inflater = getMenuInflater(); ❶
    // เรียกเมธอด inflate ของ MenuInflater เพื่อสร้างเมนูขึ้นมาจาก Menu Resource
    inflater.inflate(R.menu.game_menu, menu); ❷
    return true;
}
```



หรืออาจยุบ ❶ กับ ❷ ให้เหลือบรรทัดเดียวก็ได้

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

เมธอด `inflate` จะอ่าน Menu Resource จากไฟล์ XML ที่ระบุด้วยพารามิเตอร์ตัวแรก แล้วสร้างตัวเลือกต่างๆของเมนูตาม Menu Resource นั้นขึ้นในออบเจ็กต์ Menu ที่ระบุด้วยพารามิเตอร์ตัวที่สอง (ตัวแปร `menu`) เพื่อที่แอนดรอยด์จะนำออบเจ็กต์ Menu นี้ไปแสดงเป็นเมนูบนหน้าจอต่อไป

### ระบูกำจวนเมื่อตัวเลือกถูกคลิก

เมื่อผู้ใช้คลิกตัวเลือกในเมนู แอนดรอยด์จะเรียกมายังเมธอด `onOptionsItemSelected` ของแอคทิวิตี พร้อมทั้งส่งตัวเลือกที่ถูกคลิกมาเป็นพารามิเตอร์ ซึ่งตัวเลือกต่างๆในเมนูคือออบเจ็กต์ `MenuItem`

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    TextView tv = (TextView) findViewById(R.id.text);

    switch (item.getItemId()) { ❶
    case R.id.action_add:
        tv.setText("You selected Add");
        return true; ❷
    case R.id.action_edit:
        tv.setText("You selected Edit");
        return true; ❷
    ...
    default:
        return super.onOptionsItemSelected(item); ❸
    }
}
```

จากโค้ด เราเรียกเมธอด `getItemId` บนออบเจ็กต์ `MenuItem` เพื่อหาว่าตัวเลือกใดที่ถูกคลิก ❶ แล้วจึงแสดงข้อความออกมาที่เท็กชีว

แอนดรอยด์กำหนดว่าเมื่อคุณจัดการอีเวนต์ของการคลิกตัวเลือกในเมนูเรียบร้อยแล้ว ให้ส่งคืนค่า `true` กลับออกไป ❷ แต่หากคุณไม่ได้จัดการอีเวนต์ ให้เรียกไปยัง `onOptionsItemSelected` ของคลาสแม่ (ซูเปอร์คลาส) ❸ ซึ่งในที่นี้ก็คือกรณี `default` ของประโยคคำสั่ง `switch`

## การสร้างคอนเท็กซ์เมนู (Context Menu)

คอนเท็กซ์เมนู (Context Menu) คือเมนูที่รวบรวมตัวเลือก/คำสั่งที่มีผลกับอิลิเมนต์หนึ่งๆบนหน้าจอ ซึ่งจะแสดงออกมาเมื่อผู้ใช้แตะค้างที่อิลิเมนต์นั้น (สำหรับฮาร์ดแวร์คือการคลิกเมาส์ค้าง) เทียบได้กับเมนูคลิกขวาของระบบ Windows

เราสามารถเตรียมคอนเท็กซ์เมนูให้กับวิวชนิดใดก็ได้ แต่โดยทั่วไปนิยมใช้กับ ListView และ GridView เพื่อให้ผู้ใช้ดำเนินการต่างๆกับไอเท็มใน ListView หรือ GridView นั้น

### ตัวอย่าง

ตัวอย่างนี้จะสร้างคอนเท็กซ์เมนูให้กับรูปภาพ (ImageView) และปุ่ม (Button) บนหน้าจอ ซึ่งอาจเป็นตัวอย่างที่ไม่มีประโยชน์ แต่จะทำให้คุณเข้าใจหลักการในการสร้างคอนเท็กซ์เมนูได้อย่างรวดเร็ว

#### 1 กำหนด Layout ของหน้าจอหลัก

โปรเจ็ค ContextMenuDemo, ไฟล์ res/layout/activity\_\_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp" >

    <TextView
        android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello_world" />

    <ImageView
        android:id="@+id/image"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/ic_launcher" />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Button" />

</LinearLayout>
```

## 2 สร้างไฟล์ XML กำหนดตัวเลือกของคอนเท็กซ์เมนูสำหรับรูปภาพ

โปรเจ็ค ContextMenuDemo, ไฟล์ res\menu\context\_menu\_image.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >

    <item
        android:id="@+id/action_image_item1"
        android:title="Item1"/>

    <item
        android:id="@+id/action_image_item2"
        android:title="Item2"/>

</menu>
```

## 3 สร้างไฟล์ XML กำหนดตัวเลือกของคอนเท็กซ์เมนูสำหรับปุ่ม

โปรเจ็ค ContextMenuDemo, ไฟล์ res\menu\context\_menu\_button.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >

    <item
        android:id="@+id/action_button_item1"
        android:title="Item1"/>

    <item
        android:id="@+id/action_button_item2"
        android:title="Item2"/>

    <item
        android:id="@+id/action_button_item3"
        android:title="Item3"/>

</menu>
```

## 4 เพิ่มโค้ดในเมธอด onCreate

โปรเจ็ค ContextMenuDemo, ไฟล์ MainActivity.java

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

    ImageView image = (ImageView) findViewById(R.id.image);
    Button button = (Button) findViewById(R.id.button);

    registerForContextMenu(image);
    registerForContextMenu(button);
}
```

## 5 เพิ่มเมธอด onCreateContextMenu ในแอคทิวิตี

โปรเจ็ค ContextMenuDemo, ไฟล์ MainActivity.java

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
                               ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);

    MenuInflater inflater = getMenuInflater();
    switch (v.getId()) {
        case R.id.image:
            inflater.inflate(R.menu.context_menu_image, menu);
            break;
        case R.id.button:
            inflater.inflate(R.menu.context_menu_button, menu);
            break;
    }
}
```

## 6 เพิ่มเมธอด onContextItemSelected ในแอคทิวิตี

โปรเจ็ค ContextMenuDemo, ไฟล์ MainActivity.java

```
@Override
public boolean onContextItemSelected(MenuItem item) {
    TextView tv = (TextView) findViewById(R.id.text);

    switch (item.getItemId()) {
        case R.id.action_image_item1:
            tv.setText("You selected Item1 on Image");
            return true;
        case R.id.action_image_item2:
            tv.setText("You selected Item2 on Image");
            return true;

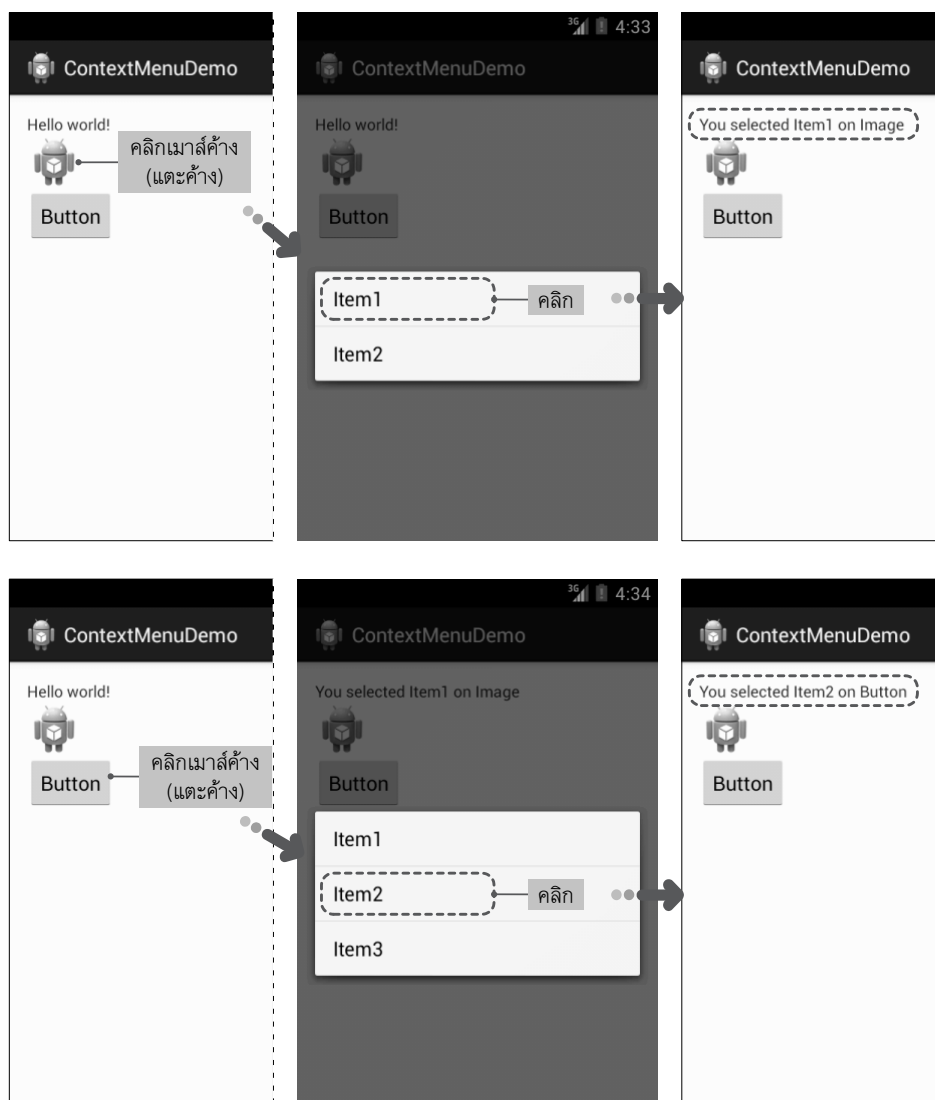
        case R.id.action_button_item1:
            tv.setText("You selected Item1 on Button");
            return true;
        case R.id.action_button_item2:
            tv.setText("You selected Item2 on Button");
            return true;
        case R.id.action_button_item3:
            tv.setText("You selected Item3 on Button");
            return true;
    }
}
```

```

default:
    return super.onContextItemSelected(item);
    }
}

```

## ผลการรัน



## คำอธิบาย

### กำหนด Menu Resource

การกำหนดตัวเลือกต่างๆของคอนเท็กซ์เมนูจะใช้ Menu Resource เช่นเดียวกับเมนูธรรมดา (Options Menu) ในตัวอย่างที่แล้ว ในที่นี้ต้องการแสดงคอนเท็กซ์เมนูสำหรับทั้งรูปภาพและปุ่ม เราจึงเตรียม Menu Resource ไว้ 2 ไฟล์ คือไฟล์ context\_menu\_image.xml สำหรับคอนเท็กซ์เมนูของรูปภาพ และไฟล์ context\_menu\_button.xml สำหรับคอนเท็กซ์เมนูของปุ่ม

### ลงทะเบียนวีธีที่ต้องการแสดงคอนเท็กซ์เมนู

แอนดรอยด์จะแสดงคอนเท็กซ์เมนูเมื่อผู้ใช้แตะค่างที่วิวหนึ่งๆบนหน้าจอ ดังนั้นเราต้องบอกให้แอนดรอยด์รู้ว่าวิวที่จะให้แสดงคอนเท็กซ์เมนูคือวิวใดบ้าง โดยเรียกเมธอด `registerForContextMenu` เพื่อลงทะเบียนวิวต่างๆเหล่านั้น

```
ImageView image = (ImageView) findViewById(R.id.image);
Button button = (Button) findViewById(R.id.button);

registerForContextMenu(image);
registerForContextMenu(button);
```

### สร้างคอนเท็กซ์เมนูขึ้นมาจาก Menu Resource

เมื่อผู้ใช้แตะค่างที่วิวซึ่งเราลงทะเบียนคอนเท็กซ์เมนูไว้ แอนดรอยด์จะเรียกมายังเมธอด `onCreateContextMenu` ของแอคทิวิตี ซึ่งเราจะใส่โค้ดที่ใช้สร้างคอนเท็กซ์เมนูขึ้นมาจาก Menu Resource ไว้ในเมธอดนี้

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
                                ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);

    MenuInflater inflater = getMenuInflater();
    switch (v.getId()) { ❶
    case R.id.image:
        inflater.inflate(R.menu.context_menu_image, menu); ❷
        break;
    case R.id.button:
        inflater.inflate(R.menu.context_menu_button, menu); ❸
        break;
    }
}
```

พารามิเตอร์ตัวที่สองของ `onCreateContextMenu` (ตัวแปร `v`) จะบอกให้รู้ว่าตอนนั้นผู้ใช้แตะค้างที่วิวใด เราเรียกเมธอด `getId` บนตัวแปรนี้เพื่อหา ID ของมัน ❶ ซึ่งถ้าเป็นรูปภาพจะสร้างเมนูจากไฟล์ `context_menu_image.xml` ❷ แต่ถ้าเป็นปุ่มจะสร้างเมนูจากไฟล์ `context_menu_button.xml` ❸

### สรุปการทำงานเมื่อตัวเลือกในคอนเท็กซ์เมนูถูกคลิก

เมื่อผู้ใช้คลิกตัวเลือกในเมนู แอนดรอยด์จะเรียกมายังเมธอด `onContextItemSelected` ของแอคทิวิตี พร้อมทั้งส่งตัวเลือกที่ถูกคลิกนั้นมาเป็นพารามิเตอร์

```
@Override
public boolean onContextItemSelected(MenuItem item) {
    TextView tv = (TextView) findViewById(R.id.text);

    switch (item.getItemId()) {
        case R.id.action_image_item1:
            tv.setText("You selected Item1 on Image");
            return true;
        case R.id.action_image_item2:
            tv.setText("You selected Item2 on Image");
            return true;

        case R.id.action_button_item1:
            tv.setText("You selected Item1 on Button");
            return true;
        case R.id.action_button_item2:
            tv.setText("You selected Item2 on Button");
            return true;
        case R.id.action_button_item3:
            tv.setText("You selected Item3 on Button");
            return true;

        default:
            return super.onContextItemSelected(item);
    }
}
```

การทำงานของโค้ดข้างต้นจะมีหลักการเหมือนกับเมธอด `onOptionsItemSelected` ของเมนูธรรมดาในหัวข้อที่แล้ว

## การสร้างคอนเท็กซ์เมนูสำหรับ ListView

ดังที่อธิบายแล้วว่าเรามักเตรียมคอนเท็กซ์เมนูให้กับ ListView เพื่อแสดงตัวเลือก/คำสั่งให้ผู้ใช้ดำเนินการกับแต่ละไอเท็มในลิสต์ ในหัวข้อนี้จึงขอแสดงการสร้างคอนเท็กซ์เมนูสำหรับ ListView นอกจากนี้จะแสดงการสร้างตัวเมนูโดยใช้โค้ดจาวา แทนที่จะ inflate ขึ้นมาจาก Menu Resource ในไฟล์ XML

### ตัวอย่างและคำอธิบาย

เราจะสร้างแอปที่สมมติว่าเป็นรายชื่อผู้ติดต่อ (Contacts) ซึ่งผู้ใช้สามารถแตะค้างที่แต่ละรายชื่อเพื่อโทรออกไปยังบุคคลนั้น ส่งข้อความไปยังบุคคลนั้น แก้ไขชื่อ หรือลบชื่อได้

และเพื่อลดความยุ่งยาก สำหรับตัวอย่างนี้จะขออธิบายการทำงานของโค้ดในหัวข้อนี้เลย

#### 1 กำหนด Layout ของหน้าจอหลัก

```
โปรเจ็ค ContextMenuListViewDemo, ไฟล์ res\layout\activity__main.xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <TextView
        android:id="@+id/text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#66ff99"
        android:gravity="center"
        android:padding="5dp"
        android:text="@string/hello_world" />

    <ListView
        android:id="@+id/list"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1" />

</LinearLayout>
```

หน้าจอหลักประกอบด้วย TextView อยู่ด้านบน และ ListView อยู่ด้านล่าง โดย ListView จะแสดงรายชื่อผู้ติดต่อ ส่วน TextView จะแสดงข้อความหลังจากคลิกตัวเลือกในคอนเท็กซ์เมนูแล้ว การกำหนด layout\_weight ของ ListView เป็น 1 จะทำให้ ListView ใช้ความสูงที่เหลือทั้งหมดของหน้าจอ ไม่ว่าหน้าจอจะมีความสูงขนาดเท่าใดก็ตาม



## 2 กำหนด Layout ของแต่ละไอเท็มใน ListView

โปรเจ็ค ContextMenuListViewDemo, ไฟล์ res\layout\listitem.xml

```
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="5dp"
    android:textSize="24sp" />
```

แต่ละไอเท็มใน ListView จะสร้างขึ้นมาจาก TextView ที่มีขนาดตัวอักษรเป็น 24 sp และมี padding (ช่องว่างภายใน) ขนาด 5 dp

## 3 เพิ่มการประกาศอาร์เรย์ contacts, อาร์เรย์ actions และเพิ่มโค้ดในเมธอด onCreate

โปรเจ็ค ContextMenuListViewDemo, ไฟล์ MainActivity.java

```
// เก็บรายชื่อผู้ติดต่อ
private String[] contacts = { "Nooknet", "Numnahm", "Tonyod" };
// เก็บชื่อการดำเนินการที่จะแสดงเป็นไอเท็มในคอนเท็กซ์เมนู
private String[] actions = { "Call", "Send message", "Edit", "Delete" };

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

    ListView list = (ListView) findViewById(R.id.list);

    /* สร้าง ArrayAdapter สำหรับนำข้อมูลจากอาร์เรย์ contacts มาแสดงใน ListView
       และกำหนดให้แอนดรอยด์แสดงแต่ละไอเท็มใน ListView โดยใช้ Layout จากไฟล์
       listitem.xml */
    ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
        R.layout.listitem, contacts);
    list.setAdapter(adapter);

    // ลงทะเบียนคอนเท็กซ์เมนูให้กับ ListView
    registerContextMenu(list);
}
```

## 4 เพิ่มเมธอด onCreateContextMenu ในแอคทิวิตี เพื่อสร้างคอนเท็กซ์เมนูขึ้นมาเมื่อผู้ใช้แตะค้างที่ไอเท็มใน ListView

โปรเจ็ค ContextMenuListViewDemo, ไฟล์ MainActivity.java

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
    ContextMenuInfo menuInfo) {
```

```

if (v.getId() == R.id.list) { ❶
    AdapterView.AdapterContextMenuInfo info =
        (AdapterView.AdapterContextMenuInfo) menuInfo; ❷
    menu.setHeaderTitle(contacts[info.position]); ❸

    for (int i = 0; i < actions.length; i++) {
        menu.add(Menu.NONE, i, i, actions[i]);
    }
}
}

```

ก่อนอื่นเราตรวจสอบให้แน่ใจว่าไอเวนต์นี้มาจาก ListView ❶ (เพราะอาจมีวิวอื่นๆในหน้าจอที่ลงทะเบียนคอนเท็กซ์เมนูด้วย ถึงแม้ในตัวอย่างนี้จะไม่มีก็ตาม) สำหรับไอเท็มในลิสต์ที่ถูกแตะค่านั้นจะตรวจสอบได้จากพารามิเตอร์ตัวที่สาม (menuInfo) ซึ่งเราต้องแปลง (cast) เป็นชนิด AdapterView.AdapterContextMenuInfo ก่อน ❷ จากนั้นเราหาลำดับของไอเท็มนั้นใน ListView แล้วนำมาระบุเป็น Index ของอาร์เรย์ contacts เพื่ออ่านข้อความ (ชื่อผู้ติดต่อ) จากอาร์เรย์มาแสดงเป็นหัวเรื่องของคอนเท็กซ์เมนู ❸

ถัดไปเรียกเมธอด add ของออบเจ็ค ContextMenu เพื่อเพิ่มตัวเลือกลงในเมนูตามจำนวนข้อมูลในอาร์เรย์ actions โดยระบุว่าไม่ต้องการจัดกลุ่มตัวเลือก (Menu.NONE), ระบุ ID และลำดับของตัวเลือกโดยใช้ค่าของ i และสุดท้ายระบุข้อความของตัวเลือกโดยใช้ข้อมูลจากอาร์เรย์ actions

- 5 เพิ่มเมธอด onContextItemSelected ในแอคทิวิตี เพื่อระบุการทำงานเมื่อตัวเลือกในคอนเท็กซ์เมนูถูกคลิก

โปรเจ็ค ContextMenuListViewDemo, ไฟล์ MainActivity.java

```

@Override
public boolean onContextItemSelected(MenuItem item) {
    // หาข้อความของตัวเลือกในคอนเท็กซ์เมนูที่ถูกเลือก (ชื่อคำสั่ง)
    int actionIndex = item.getItemId();
    String actionName = actions[actionIndex];

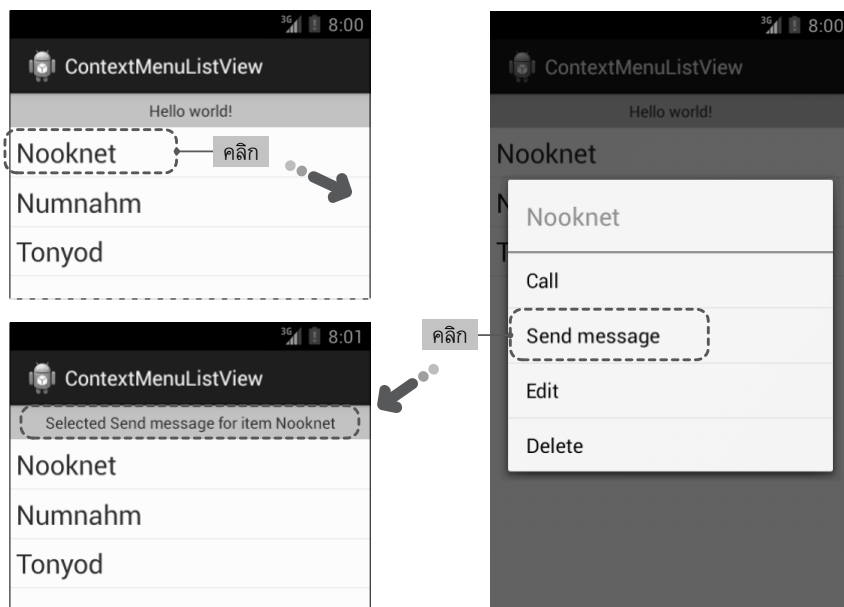
    // หาข้อความของไอเท็มใน ListView ที่ถูกแตะค้าง (ชื่อผู้ติดต่อ)
    AdapterView.AdapterContextMenuInfo info =
        (AdapterView.AdapterContextMenuInfo) item getMenuInfo();
    String contactName = contacts[info.position];

    // แสดงข้อความออกมาที่ TextView
    TextView text = (TextView) findViewById(R.id.text);
    text.setText(String.format("Selected %s for item %s", actionName,
                               contactName));

    return true;
}

```

## ผลการรัน



## การสร้างคอนเท็กซ์เมนูแบบ Contextual Action Bar

คอนเท็กซ์เมนูที่ยกตัวอย่างมาทั้งหมดก่อนหน้านี้เป็นรูปแบบที่เรียกว่า Floating Context Menu ซึ่งมีลักษณะคล้ายไดอะล็อกที่ลอยอยู่บนหน้าจอปัจจุบัน แต่ในแอนดรอยด์ 3.0 (API Level 11) ขึ้นไป เรายังสามารถแสดงคอนเท็กซ์เมนูได้อีกรูปแบบหนึ่ง เรียกว่า *Contextual Action Bar* หรือพูดง่ายๆ ก็คือการแสดงคอนเท็กซ์เมนูบน Action Bar นั้นเอง (แต่จริงๆ แล้ว Contextual Action Bar กับ Action Bar เป็นคนละส่วนกัน เพียงแต่มีหน้าตาคล้ายกัน และแสดงอยู่ด้านบนของหน้าจอเหมือนกัน)

### NOTE>>>

Contextual Action Bar จะใช้ได้ตั้งแต่แอนดรอยด์ 3.0 ขึ้นไป ดังนั้นต้องกำหนด `minSDKVersion` ในไฟล์ `AndroidManifest.xml` เป็น 11 หรือสูงกว่า

## ตัวอย่างและคำอธิบาย

ตัวอย่างนี้จะมีรูปภาพในหน้าจอหลัก ซึ่งเมื่อแตะค้างที่รูปภาพจะแสดง Contextual Action Bar ที่มี 3 ตัวเลือกคือ Share, Edit และ Delete ออกมา

## 1 กำหนด Layout ของหน้าจอหลัก

โปรเจ็ค ContextualActionBarDemo, ไฟล์ res\layout\activity\_\_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <TextView
        android:id="@+id/text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#66ff99"
        android:gravity="center"
        android:padding="5dp"
        android:text="@string/hello_world" />

    <ImageView
        android:id="@+id/image"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:padding="16dp"
        android:src="@drawable/kids" />

</LinearLayout>
```

## 2 เพิ่มการประกาศอาร์เรย์ actionTitles, อาร์เรย์ actionIcons และตัวแปร mActionMode ในแอสกทิตวี

โปรเจ็ค ContextualActionBarDemo, ไฟล์ MainActivity.java

```
// ข้อความของตัวเลือก
private String[] actionTitles = { "Share", "Edit", "Delete" };
// รูปภาพไอคอนของตัวเลือก
private int[] actionIcons = {
    android.R.drawable.ic_menu_share,
    android.R.drawable.ic_menu_edit,
    android.R.drawable.ic_menu_close_clear_cancel };
// ออบเจ็ค ActionMode
private ActionMode mActionMode;
```

## 3 สร้างออบเจ็คซึ่งทำการ Implement อินเทอร์เฟซ ActionMode.Callback แล้วเก็บออบเจ็คนี้ในตัวแปร mActionModeCallback

โปรเจ็ค ContextualActionBarDemo, ไฟล์ MainActivity.java

```
private ActionMode.Callback mActionModeCallback =
    new ActionMode.Callback() {
```

```
// ถูกเรียกเมื่อเข้าสู่ Action Mode
@Override
public boolean onCreateActionMode(ActionMode mode, Menu menu) {
    MenuItem item;

    for (int i = 0; i < actionTitles.length; i++) {
        item = menu.add(Menu.NONE, i, i, actionTitles[i]); ❶
        item.setIcon(actionIcons[i]);
        item.setShowAsAction(MenuItem.SHOW_AS_ACTION_IF_ROOM |
                             MenuItem.SHOW_AS_ACTION_WITH_TEXT);
    }
    return true;
}

// ถูกเรียกหลังจาก onCreateActionMode
@Override
public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
    return false; // ให้ return ค่า false ถ้าไม่ได้ทำอะไร
}

// ถูกเรียกเมื่อผู้ใช้คลิกตัวเลือกบน Contextual Action Bar
@Override
public boolean onOptionsItemSelected(ActionMode mode, MenuItem item) {
    int actionIndex = item.getItemId();
    String actionName = actionTitles[actionIndex];

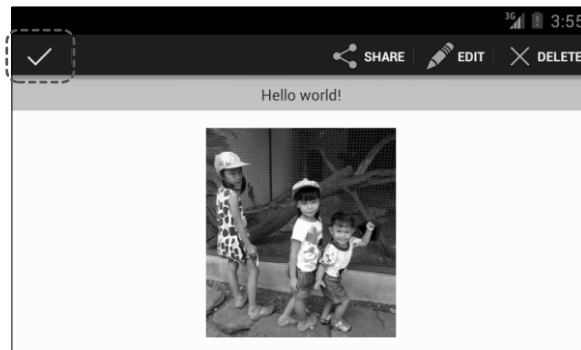
    TextView text = (TextView) findViewById(R.id.text);
    text.setText("You selected " + actionName); ❷

    mode.finish(); ❸
    return true;
}

// ถูกเรียกเมื่อออกจาก Action Mode
@Override
public void onDestroyActionMode(ActionMode mode) {
    mActionMode = null; ❹
}
};
```

ภายในอินเทอร์เฟซ `ActionMode.Callback` เราต้อง Override เมธอด 4 เมธอด ได้แก่

- ◆ `onCreateActionMode` เมธอดนี้จะถูกเรียกเมื่อเข้าสู่ Action Mode หรือก็คือเมื่อผู้ใช้แตะค้างที่วิวหนึ่งๆ และแอนดรอยด์เตรียมจะแสดงตัวเลือกออกมาบน Contextual Action Bar ซึ่งภายในเมธอดนี้เราใช้เมธอด `add` เพิ่มตัวเลือกต่างๆ เข้าไปในออบเจ็ค `Menu` ❶ แต่คุณอาจใช้วิธี `inflate` เมนูจาก Menu Resource ในไฟล์ XML ก็ได้เช่นกัน
- ◆ `onPrepareActionMode` เมธอดนี้จะถูกเรียกเมื่อเข้าสู่ Action Mode แต่จะถูกเรียกหลังจากเมธอด `onCreateActionMode` และอาจถูกเรียกมากกว่า 1 ครั้ง (รายละเอียดเพิ่มเติมดูได้จากเอกสารของแอนดรอยด์)
- ◆ `onActionItemClicked`  
เมธอดนี้จะถูกเรียกเมื่อตัวเลือกบน Contextual Action Bar ถูกคลิก ในที่นี้เราอ่านข้อความของตัวเลือกที่ถูกคลิกมาแสดงใน `TextView` ❷ แล้วเรียก `mode.finish()` ❸ เพื่อออกจาก Action Mode ซึ่งจะ ทำให้ Contextual Action Bar ถูกปิดไป  
(ถ้าไม่เรียก `mode.finish()` ผู้ใช้จะต้องปิด Contextual Action Bar เองโดยคลิกไอคอน  ทางซ้าย)
- ◆ `onDestroyActionMode` เมธอดนี้จะถูกเรียกเมื่อออกจาก Action Mode ในที่นี้เรากำหนดค่า `null` ให้ตัวแปร `mActionMode` เพื่อทำลายออบเจ็ค `ActionMode` ทั้งหมด ❹ (ออบเจ็ค `ActionMode` จะถูกสร้างในเมธอด `onCreate` ที่จะกล่าวต่อไป)



#### 4 เพิ่มโค้ดในเมธอด `onCreate`

โปรเจ็ค ContextualActionBarDemo, ไฟล์ MainActivity.java

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

    ImageView image = (ImageView) findViewById(R.id.image);

    // ระบุการทำงานเมื่อผู้ใช้แตะค้างที่รูปภาพ
    image.setOnLongClickListener(new View.OnLongClickListener() {
```

```

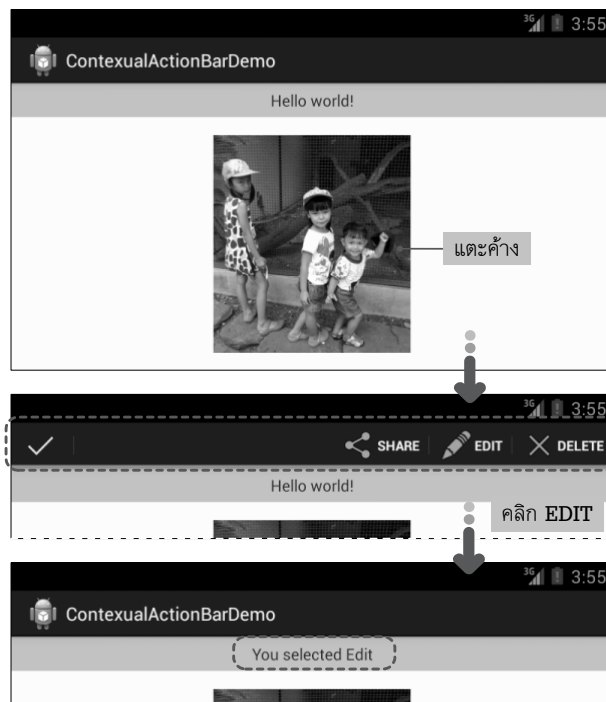
public boolean onLongClick(View view) {
    if (mActionMode != null) {
        return false;
    }

    /* เริ่มต้น Action Mode โดยใช้ ActionMode.Callback ที่ Implement ไว้ใน
       ตัวแปร mActionModeCallback */
    mActionMode =
        MainActivity.this.startActionMode(mActionModeCallback); ❶
    return true;
}
});
}

```

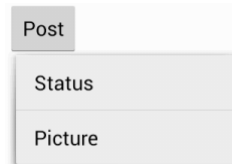
เมื่อผู้ใช้แตะค้างที่รูปภาพ เราจะเริ่มต้น Action Mode โดยเรียกเมธอด `startActionMode` และ  
 ระบุพารามิเตอร์เป็น `ActionMode.Callback` ที่เราได้ Implement ไว้ในตัวแปร  
`mActionModeCallback` ❶ โค้ดส่วนนี้จะเทียบได้กับการลงทะเบียนคอนเท็กซ์เมนูด้วยเมธอด  
`registerForContextMenu` ในสองตัวอย่างที่ผ่านมา

## ผลการรัน



## การสร้างป๊อปอัพเมนู (Popup Menu)

ป๊อปอัพเมนู (Popup Menu) คือเมนูที่ติดอยู่กับวิวหนึ่งๆ โดยจะปรากฏออกมาด้านล่างของวิวนั้น (หรือด้านบน ถ้าด้านล่างมีที่ว่างไม่พอ) และใช้แสดงตัวเลือก/คำสั่งที่เป็นรายละเอียดการทำงานเพิ่มเติมของวิวนั้น เช่น คุณอาจมีปุ่ม Post บนหน้าจอ ซึ่งเมื่อคลิกจะแสดงตัวเลือก Status และ Picture ออกมาในป๊อปอัพเมนู เพื่อให้ผู้ใช้เลือกโพสต์ข้อความสดหรือรูปภาพไปยังเฟสบุ๊ก ดังรูป



### NOTE >>>

- ป๊อปอัพเมนูกับคอนเท็กซ์เมนูจะมีหน้าที่แตกต่างกัน เช่น ถ้าเป็นคอนเท็กซ์เมนูของปุ่มก็อาจมีตัวเลือกสำหรับเปลี่ยนแปลงขนาดและตำแหน่งของปุ่ม ในขณะที่ป๊อปอัพเมนูของปุ่มจะแสดงทางเลือกการทำงานเพิ่มเติม เช่นปุ่ม Post ที่ยกตัวอย่างข้างต้น
- ป๊อปอัพเมนูจะใช้ได้ในแอนดรอยด์ 3.0 ขึ้นไป ดังนั้นต้องกำหนด minSDKVersion ในไฟล์ AndroidManifest.xml เป็น 11 หรือสูงกว่า

## ตัวอย่าง

เราจะสร้างปุ่ม Post ที่มีตัวเลือก Status และ Picture ตามที่บอกเมื่อครู่นี้ แต่เมื่อคลิกตัวเลือกทั้งสองจะแสดงข้อความใน TextView แทนการโพสต์ไปยังเฟสบุ๊กจริงๆ

### 1 กำหนด Layout ของหน้าจอหลัก

โปรเจกต์ PopupMenuDemo, ไฟล์ res/layout/activity\_\_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp" >

    <TextView
        android:id="@+id/text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#66ff99"
        android:gravity="center"
        android:padding="5dp"
```



```

        android:text="@string/hello_world" />

        <Button
            android:id="@+id/post_button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Post" />

    </LinearLayout>

```

## 2 สร้างไฟล์ XML กำหนดตัวเลือกของป๊อปอัพเมนู

โปรเจ็ค PopupMenuDemo, ไฟล์ res\menu\popup\_\_menu.xml

```

<menu xmlns:android="http://schemas.android.com/apk/res/android" >

    <item
        android:id="@+id/action_post_status"
        android:title="Status"/>

    <item
        android:id="@+id/action_post_picture"
        android:title="Picture"/>

</menu>

```

## 3 เพิ่มโค้ดในเมธอด onCreate เพื่อให้แสดงป๊อปอัพเมนูเมื่อปุ่มถูกคลิก

โปรเจ็ค PopupMenuDemo, ไฟล์ MainActivity.java

```

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

    Button btnPost = (Button) findViewById(R.id.post_button);
    btnPost.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View v) {
            PopupMenu popup = new PopupMenu(MainActivity.this, v);
            popup.setOnMenuItemClickListener(new MyPopupHandler());

            MenuInflater inflater = popup.getMenuInflater();
            inflater.inflate(R.menu.popup_menu, popup.getMenu());

            popup.show();
        }
    });
}

```

4. เพิ่มคลาส MyPopupHandler ภายในแอคทิวิตี (MyPopupHandler จะเป็น Inner Class ของ MainActivity) เพื่อระบุการทำงานเมื่อตัวเลือกในป๊อปอัพเมนูถูกคลิก

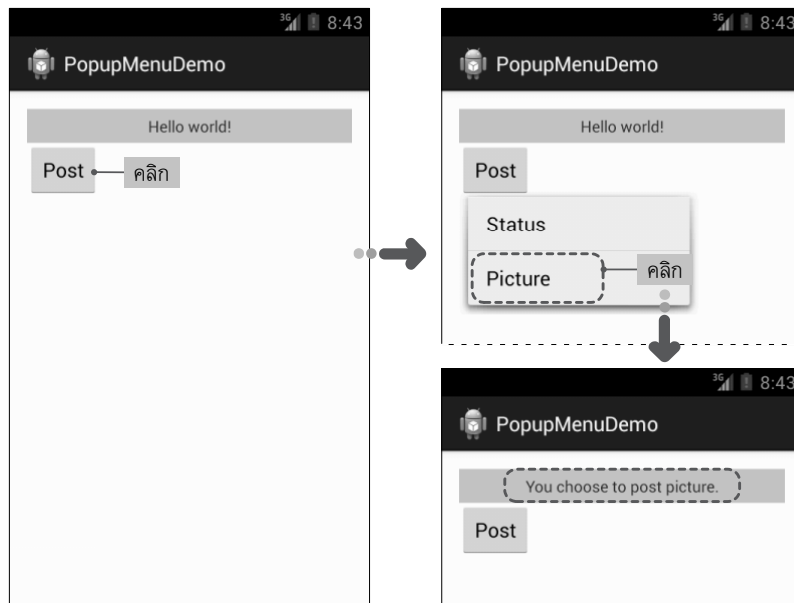
โปรเจ็ค PopupMenuDemo, ไฟล์ MainActivity.java

```
private class MyPopupHandler implements PopupMenu.OnMenuItemClickListener {

    @Override
    public boolean onMenuItemClick(MenuItem item) {
        TextView tv = (TextView) findViewById(R.id.text);

        switch (item.getItemId()) {
            case R.id.action_post_status:
                tv.setText("You choose to post status.");
                return true;
            case R.id.action_post_picture:
                tv.setText("You choose to post picture.");
                return true;
            default:
                return false;
        }
    }
}
```

## ผลการรับ



## คำอธิบาย

### สร้างป๊อปอัพเมนูขึ้นมาจาก Menu Resource

เช่นเดียวกับเมนูแบบอื่นๆ เราสามารถสร้างตัวเลือกของป๊อปอัพเมนูโดยการ inflate ขึ้นมาจาก Menu Resource หรือใช้เมธอด add เพิ่มตัวเลือกเข้าไปเองทีละตัวเลือกก็ได้ ในที่นี่ใช้วิธีแรก โดยจะแสดงป๊อปอัพเมนูเมื่อปุ่มถูกคลิก

```
Button btnPost = (Button) findViewById(R.id.post_button);
btnPost.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {
        PopupMenu popup = new PopupMenu(MainActivity.this, v); ❶

        MenuInflater inflater = popup.getMenuInflater();
        inflater.inflate(R.menu.popup_menu, popup.getMenu()); ❷

        popup.show(); ❸
    }
});
```

การแสดงผลป๊อปอัพเมนูจะเริ่มจากการสร้างออบเจ็คของคลาส **PopupMenu** ❶ โดยพารามิเตอร์ตัวแรกคือคอนเท็กซ์ (**MainActivity.this**) พารามิเตอร์ตัวที่สองคือวิวที่จะนำป๊อปอัพเมนูนี้ไปยึดติด (**v** ซึ่งก็คือปุ่มที่ถูกคลิก) จากนั้นจึง inflate เมนูขึ้นมาจาก Menu Resource ❷ แล้วแสดงป๊อปอัพเมนูออกมา ❸

### ระบุการก่อกำเนิดเมื่อตัวเลือกในป๊อปอัพเมนูถูกคลิก

ขั้นตอนนี้จะต้อง Implement อินเทอร์เฟซ **PopupMenu.OnMenuItemClickListener** และระบุการทำงานที่ต้องการไว้ในเมธอด **onMenuItemClick** ในที่นี่เราตรวจสอบว่าตัวเลือกใดถูกคลิก แล้วจึงแสดงข้อความที่ **TextView**

```
private class MyPopupHandler implements PopupMenu.OnMenuItemClickListener {

    @Override
    public boolean onMenuItemClick(MenuItem item) {
        TextView tv = (TextView) findViewById(R.id.text);

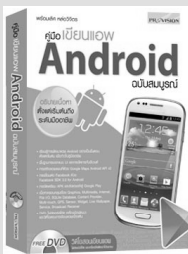
        switch (item.getItemId()) {
            case R.id.action_post_status:
                tv.setText("You choose to post status.");
                return true;
            case R.id.action_post_picture:
                tv.setText("You choose to post picture.");
        }
    }
}
```

```
        return true;
    default:
        return false;
    }
}
```

จากนั้นผู้การทำงานข้างต้นเข้ากับป๊อปอัพเมนู โดยใช้เมธอด `setOnMenuItemClickListener` ของป๊อปอัพเมนู ซึ่งโค้ดบรรทัดนี้อยู่ในเมธอด `onClick` ของปุ่ม

```
popup.setOnMenuItemClickListener(new MyPopupHandler());
```

#### NOTE >>>



สำหรับการโพสต์ข้อความหรือรูปภาพไปยังเฟสบุ๊คจริง ๆ นั้น คุณสามารถศึกษาได้จากหนังสือเล่มหนึ่งของผู้เขียนคือ “คู่มือเขียนแอป Android ฉบับสมบูรณ์” ซึ่งมีบทหนึ่งที่ผู้เขียนอธิบายการเขียนแอป Android เพื่อเชื่อมต่อกับเฟสบุ๊คไว้อย่างละเอียด

ดูข้อมูลเพิ่มเติมเกี่ยวกับหนังสือเล่มนี้และสั่งซื้อได้ที่ [bit.ly/14IK6hm](http://bit.ly/14IK6hm)