모바일 프로그래밍

04 메뉴와 대화 상자 1

2017 2학기

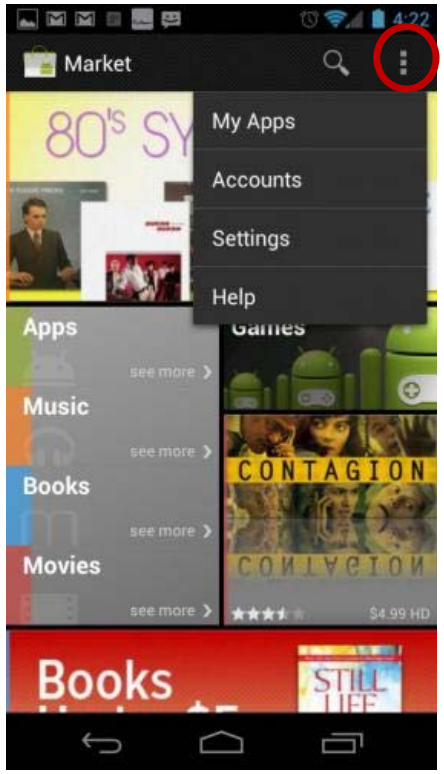
강승우

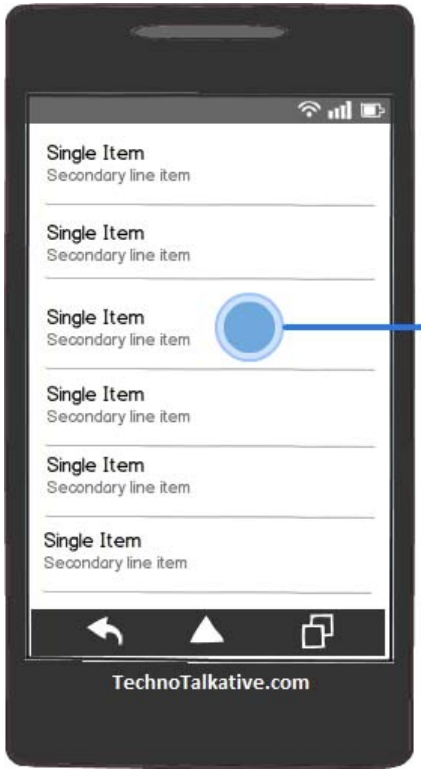
Computer Science and Engineering, KOREATECH

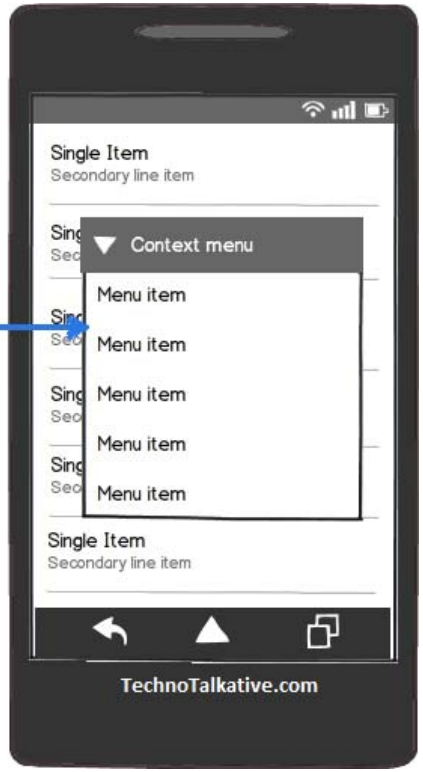
04-1 메뉴

메뉴

- 사용자에게 애플리케이션에서 제공하는 기능을 보여주는 인터페이스



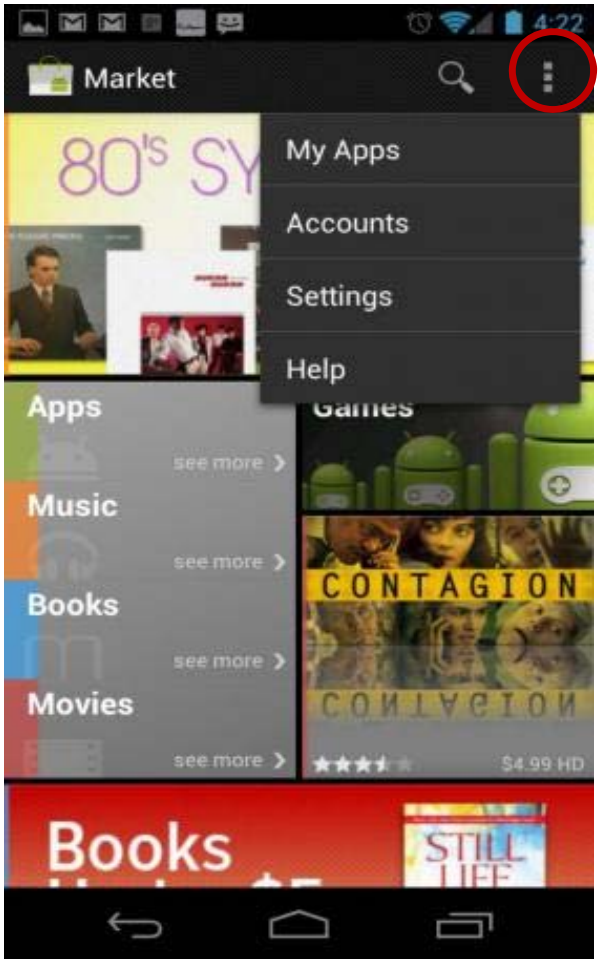




옵션 메뉴

옵션 메뉴 (options menu)

- Activity의 주된 메뉴
  - 앱 전체에서 사용되는 메뉴 항목을 포함
  - 한 Activity에 하나의 옵션 메뉴
  - Android 3.0 이상에서는 액션 바에 옵션 메뉴가 나타남
    - 2.3이나 그 이하 버전에서는 메뉴 버튼이 있었음



- 옵션 메뉴 예제 프로젝트: Ch7OptionsMenu

XML로 메뉴 정의

- res/menu 폴더에 XML 파일 생성

- 메뉴 정의를 위한 요소
  - <menu>
    - 메뉴 항목을 저장하는 컨테이너
    - 안드로이드 자바 클래스인 Menu 클래스에 매핑
  - <item>
    - 하나의 메뉴 항목 정의
    - 안드로이드 자바 클래스인 MenuItem 클래스에 매핑
    - <menu> 요소를 내부에 포함할 수 있어 서브 메뉴 생성 가능
  - <group>
    - 여러 개의 <item> 요소들의 그룹을 정의

mymenu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android" >

    <item
        android:id="@+id/apple"
        android:icon="@drawable/ic_launcher"
        android:title="사과" />
    <item
        android:id="@+id/grape"
        android:icon="@drawable/ic_launcher"
        android:title="포도" />
    <item
        android:id="@+id/banana"
        android:icon="@drawable/ic_launcher"
        android:title="바나나" />

</menu>
```

<menu> 엘리먼트는 Menu를 생성하고 이것은 메뉴 항목들을 담는 컨테이너가 된다. 반드시 루트 노드이어야 하며 <item>이나 <group>을 하나 이상 포함한다.

<item> 엘리먼트는 MenuItem을 생성한다. 이것은 하나의 메뉴 항목을 나타낸다. 서브 메뉴를 작성하려면 <menu>를 포함할 수 있다.

옵션 메뉴 생성

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

Activity class에 정의된 메소드

- 3.0 버전 이상에서는 Activity가 시작될 때 액션 바에 메뉴를 표시하기 위해 호출됨

https://developer.android.com/reference/android/view/MenuInflater.html

옵션 메뉴 항목 선택 이벤트 처리

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.apple:
            Toast.makeText(getApplicationContext(), "Apple", Toast.LENGTH_SHORT).show();
            return true;
        case R.id.grape:
            Toast.makeText(getApplicationContext(), "Grape", Toast.LENGTH_SHORT).show();
            return true;
        case R.id.banana:
            Toast.makeText(getApplicationContext(), "Banana", Toast.LENGTH_SHORT).show();
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

코드로 옵션 메뉴 생성

```
public class OptionMenu1Activity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        super.onCreateOptionsMenu(menu);

        MenuItem item1 = menu.add(0, 1, 0, "사과");
        item1.setIcon(R.drawable.ic_launcher);
        item1.setAlphabeticShortcut('a');

        menu.add(0, 2, 0, "포도").setIcon(R.drawable.ic_launcher);
        menu.add(0, 3, 0, "바나나").setIcon(R.drawable.ic_launcher);

        return true;
    }
```

"사과"라는 메뉴 항목을 추가하고 아이콘과 단축키도 설정한다.

메뉴 항목의 아이디는 1

반환되는 값을 이용하여서 이렇게 하여도 된다.

코드로 옵션 메뉴 생성

- add(int groupId, int itemId, int order, CharSequence title);
  - groupId: 항목을 모아 그룹을 만들 때 사용. 그룹이 없으면 0
  - order: 메뉴 항목의 순서. 0으로 주면 항목이 추가된 순서대로 나타남

옵션 메뉴 관련 팁

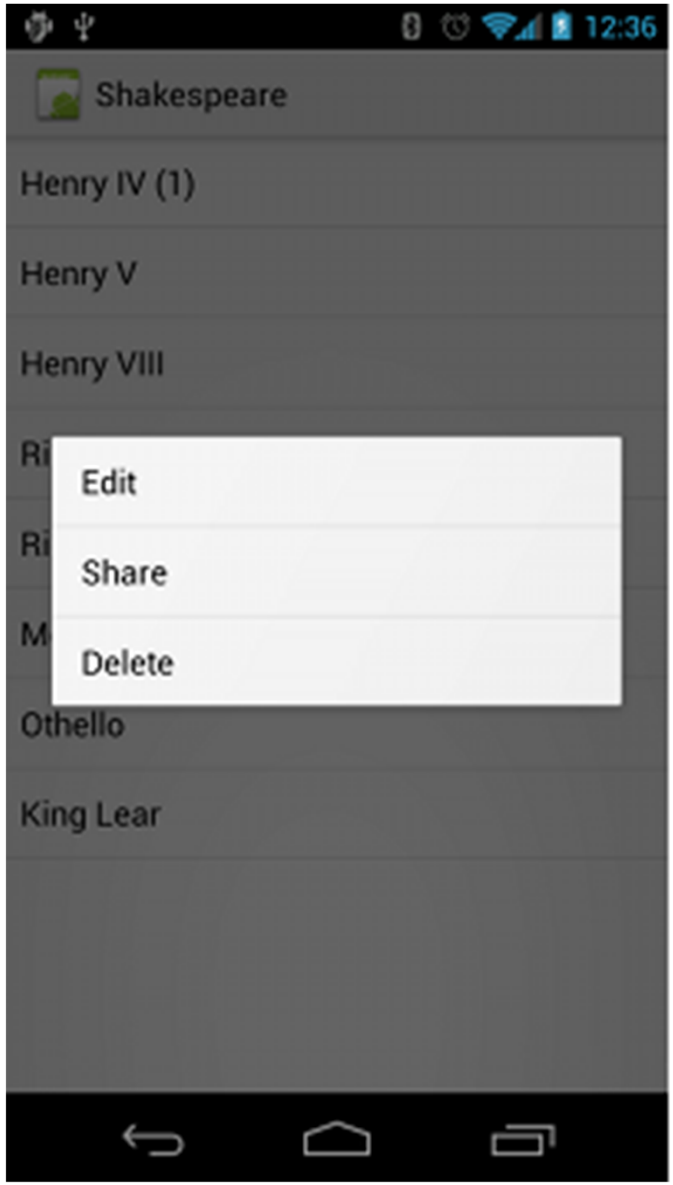
- 여러 개의 Activity로 이루어진 한 애플리케이션에서 모든 Activity가 동일한 옵션 메뉴를 갖는 경우
  - onCreateOptionsMenu(), onOptionsItemSelected() 메소드만 구현하는 Activity를 생성하고 애플리케이션을 구성하는 Activity는 이 Activity를 상속받아 생성
  - 만약 일부 Activity에서 새로운 메뉴 항목을 추가하고 싶다면, onCreateOptionsMenu() 메소드를 오버라이드하여 사용
    - super.onCreateOptionsMenu(menu)를 호출하고, menu.add() 메소드를 이용하여 항목 추가
- 메뉴 항목의 onClick 속성
  - 메뉴 항목이 선택되었을 때 호출할 함수를 onClick 속성으로 지정할 수 있다

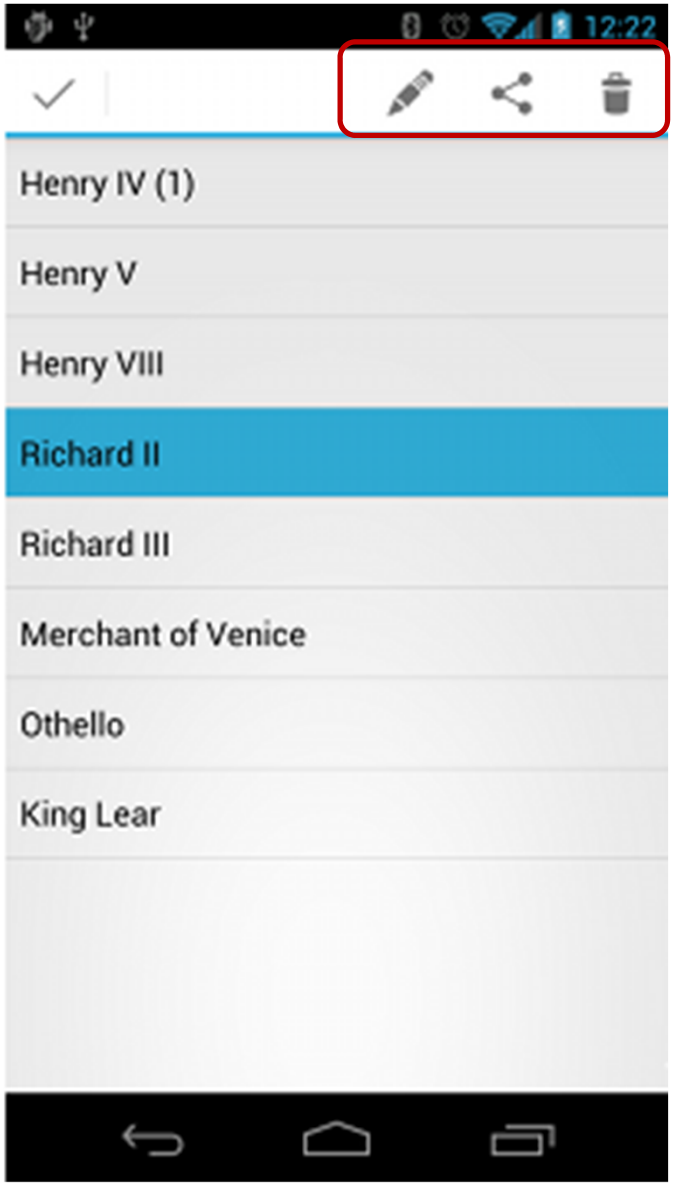
컨텍스트 메뉴

컨텍스트 메뉴 (contextual menu)

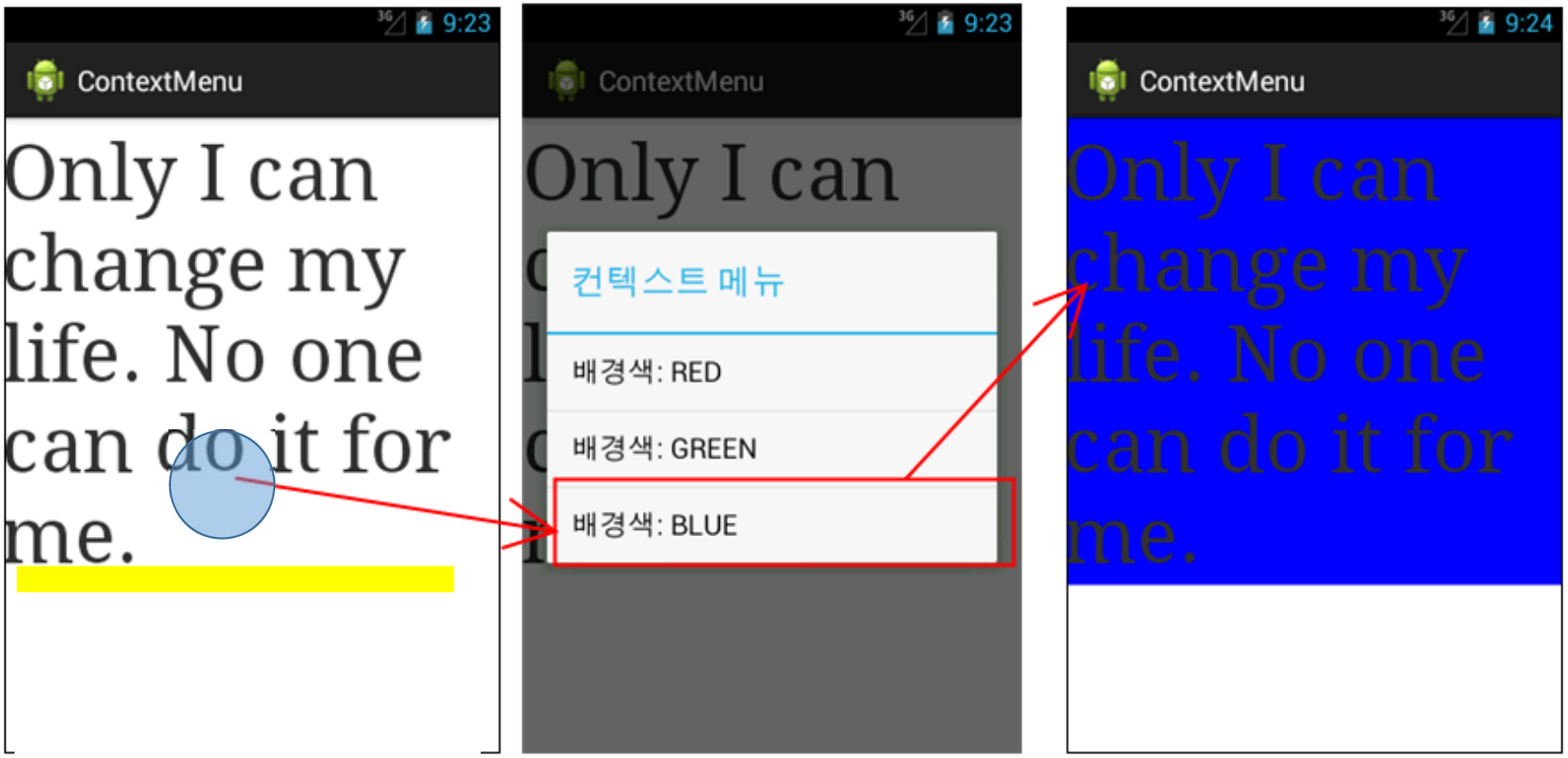
- UI에서 특정한 항목에 영향을 주는 동작을 선택할 수 있는 메뉴
- 보통 ListView나 GridView 항목에서 사용
  - 어떤 view에서도 컨텍스트 메뉴 제공 가능

- 2가지 종류
  - 플로팅 컨텍스트 메뉴 (floating context menu)
    - 항목을 long click 하면 메뉴가 표시
  - 컨텍스트 액션 모드 (contextual action mode)
    - 선택된 항목에 관련된 메뉴가 액션바에 표시


12:36
Shakespeare
Henry IV (1)
Henry V
Henry VIII
Edit
Share
Delete
Othello
King Lear


12:22
Henry IV (1)
Henry V
Henry VIII
Richard II
Richard III
Merchant of Venice
Othello
King Lear

플로팅 컨텍스트 메뉴 (floating context menu)



- 플로팅 컨텍스트 메뉴 예제 프로젝트: Ch7FloatingContextMenu

예제 - 레이아웃 XML

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/LinearLayout01"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <TextView
        android:id="@+id/TextView01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Only I can change my life. No one can do it for me."
        android:textSize="50px"
        android:typeface="serif" />

</LinearLayout>
```

예제 – Activity View에 컨텍스트 메뉴 등록하기

```
...
public class MainActivity extends Activity {
    TextView text;
    @Override
    public void onCreate(BundlesavedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        text = (TextView) findViewById(R.id.TextView01);
        registerForContextMenu(text);
    }
```

이 Activity에 View.onCreateContextMenuListener를 설정한다는 의미

https://developer.android.com/reference/android/view/View.OnCreateContextMenuListener.html

예제 – 플로팅 컨텍스트 메뉴 생성하기

@Override

컨텍스트 메뉴를 보여줘야 하는 시점에 호출되는 메소드 (예: long click 시)

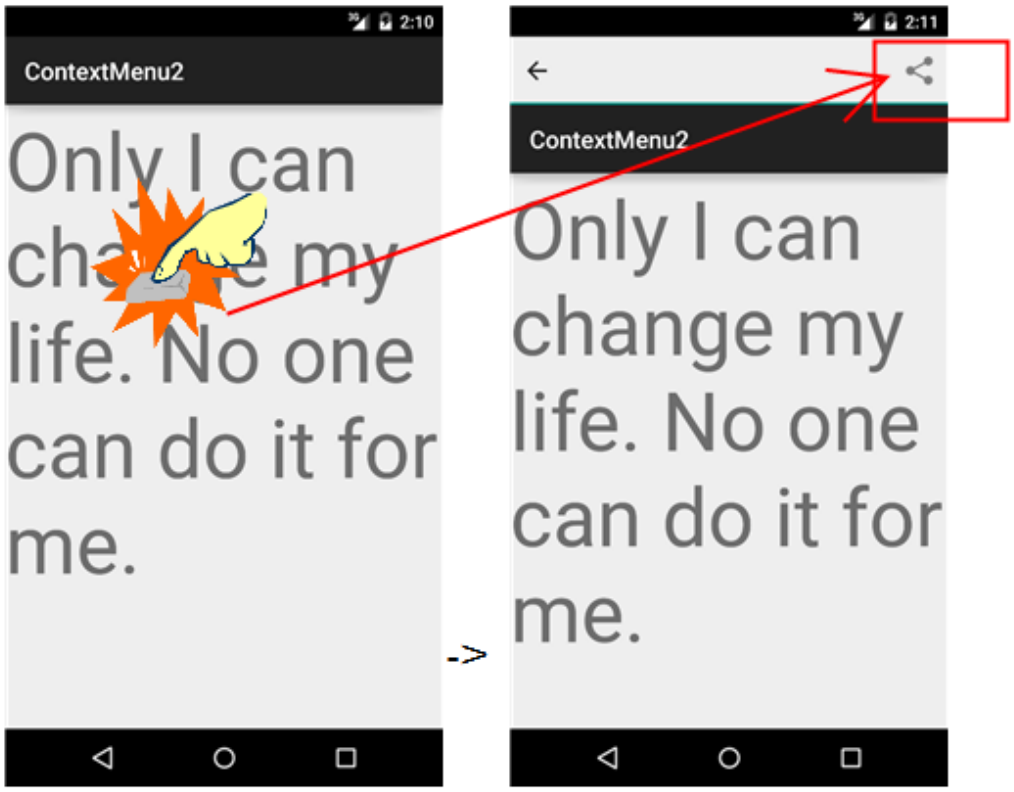
```
public void onCreateContextMenu(ContextMenu menu, View v,
        ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
    menu.setHeaderTitle("컨텍스트 메뉴");
    menu.add(0, 1, 0, "배경색: RED");
    menu.add(0, 2, 0, "배경색: GREEN");
    menu.add(0, 3, 0, "배경색: BLUE");
}
```

<코드로 메뉴를 생성한 예제>

예제 – 플로팅 컨텍스트 메뉴 이벤트 처리

```
public boolean onContextItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case 1:
            text.setBackgroundColor(Color.RED);
            return true;
        case 2:
            text.setBackgroundColor(Color.GREEN);
            return true;
        case 3:
            text.setBackgroundColor(Color.BLUE);
            return true;
        default:
            return super.onContextItemSelected(item);
    }
}
```

컨텍스트 액션 모드 (contextual action mode)



- 현재 선택된 항목에 대해 수행할 수 있는 액션들을 제공하는 컨텍스트 액션바가 표시됨
- 3.0 (API level 11) 이상에서는 플로팅 컨텍스트 메뉴 대신에 컨텍스트 액션 모드를 사용 권장
- 컨텍스트 액션 모드의 활성화
  - 뷰 위에서 long click
  - 뷰 내부의 체크박스 같은 UI 요소를 선택

컨텍스트 액션 모드 (contextual action mode)

- 특정 View를 선택했을 때 컨텍스트 액션 모드 실행을 위해서는

1. ActionMode.Callback interface를 구현해야 함

2. 컨텍스트 액션 바를 보여주고자 하는 시점에 startActionMode() 메소드를 호출
    - 사용자가 특정 view를 long click 했을 때 컨택스트 액션 모드를 실행하고 싶다면
      → 해당 view에서 발생한 long click 이벤트를 처리할 수 있도록 View.OnLongClickListener interfac를 구현 (onLongClick 메소드 구현)
      → onLongClick 메소드 내부에서 startActionMode() 메소드 호출

예제

- 컨텍스트 액션 모드 예제 프로젝트: Ch7ContextualActionMode

- Activity 클래스에서 ActionMode.Callback interface와View.OnLongClickListener interface 구현

```
public class MainActivity extends AppCompatActivity implements
                                View.OnLongClickListener, ActionMode.Callback {

    ActionMode mActionMode;

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

        TextView text = (TextView) findViewById(R.id.TextView01);
         text.setOnLongClickListener(this);
    }
```

ActionMode.Callback interface 구현할 때 필요한 public 메소드 4가지

https://developer.android.com/reference/android/support/v7/view/ActionMode.Callback.html

```
// startActionMode() 메소드가 호출될 때 호출되는 콜백 메소드
@Override
public boolean onCreateActionMode(ActionMode mode, Menu menu) {

    // MenuInflater 객체를 이용하여 컨텍스트 메뉴를 생성
    MenuInflater inflater = mode.getMenuInflater();
    inflater.inflate(R.menu.context_menu, menu);
    return true;
}

// onCreateActionMode()가 호출된 후에 호출.
// 액션 메뉴를 refresh하는 목적으로 여러 번 호출될 수 있음
@Override
public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
    return false; // 아무 것도 하지 않을 때 false 반환, 액션 메뉴가 업데이트 되면 true 반환
}
```

```
// 사용자가 액션 메뉴 항목을 클릭했을 때 호출
@Override
public boolean onActionItemClicked(ActionMode mode, MenuItem item) {
    switch (item.getItemId()) {
        case R.id.menu_share:
            // 필요한 작업을 수행하는 코드 작성
            mode.finish(); // 컨텍스트 액션 모드를 종료
            return true; // 이벤트를 처리하였으면 true 반환
        default:
            return false;
    }
}

// 사용자가 컨텍스트 액션 모드를 빠져나갈 때 호출
@Override
public void onDestroyActionMode(ActionMode mode) {
    mActionMode = null;
}
```

```
// 사용자가 액션 메뉴 항목을 클릭했을 때 호출
// View.OnLongClickListener를 구현하기 위해 필요한 콜백 메소드
public boolean onLongClick(View view) {
    if (mActionMode != null) {
        return false;
    }

    // 컨텍스트 액션 모드 시작
    mActionMode = this.startSupportActionMode(this);
    view.setSelected(true);
    return true;
}
```

- res/menu 디렉토리에 context_menu.xml 파일 생성

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/menu_share"
        android:icon="@drawable/ic_share_black_24dp"
        android:title="match_parent" />
</menu>
```

팝업 메뉴

팝업 메뉴 (popup menu)

- 특정 뷰에 고정된 모달 메뉴 (modal menu)
  - 참고

    Modal window: main window의 하위 window로 사용자로 하여금 main window로 다시 돌아가서 operation을 지속하기 위해서는 그 전에 인터랙션을 하도록 요구하는 window
- 뷰 아래 공간이 있으면 아래쪽에 나타나고 그렇지 않으면 위에 표시
- 용도
  - 특정 컨텐트와 관련된 액션을 overflow 스타일 메뉴로 제공
  - 서브 메뉴의 기능 제공
  - 드롭다운 메뉴 제공

1. 팝업 메뉴를 표시하고자 하는 뷰 요소에 onClick 속성값으로 콜백 메소드 이름 설정

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Popup"
    android:onClick="showPopup" />
```

2. 팝업 메뉴 XML 파일 생성

res/menu 폴더 아래

3. Activity 클래스에 콜백 메소드 구현

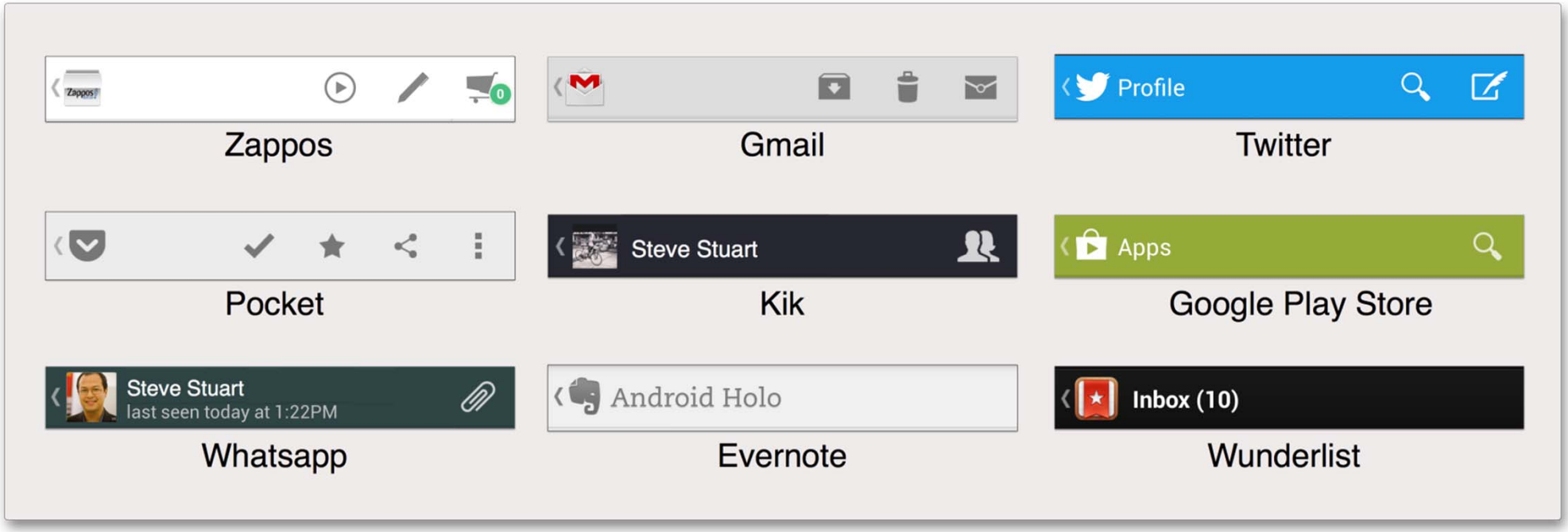
```
public void showPopup(View v) {
    PopupMenu popup = new PopupMenu(this, v);
    MenuInflater inflater = popup.getMenuInflater();
    inflater.inflate(R.menu.actions, popup.getMenu());
    popup.show();
}
```

4. 메뉴 항목 클릭 이벤트 처리를 위해서는 PopupMenu.OnMenuItemClickListener interface를 구현하고 PopupMenu의 setOnMenuItemClickListener() 메소드를 호출하여 이벤트 리스너를 등록한다

```
public void showPopup(View v) {
 PopupMenu popup = new PopupMenu(this, v);
 MenuInflater inflater = popup.getMenuInflater();
 inflater.inflate(R.menu.actions, popup.getMenu());
 popup.setOnMenuItemClickListener(
        new PopupMenu.OnMenuItemClickListener() {
            public boolean onMenuItemClick(MenuItem item) {
                // 이벤트 처리 작업
                        ……
                }
            }
 });
 popup.show();
}
```

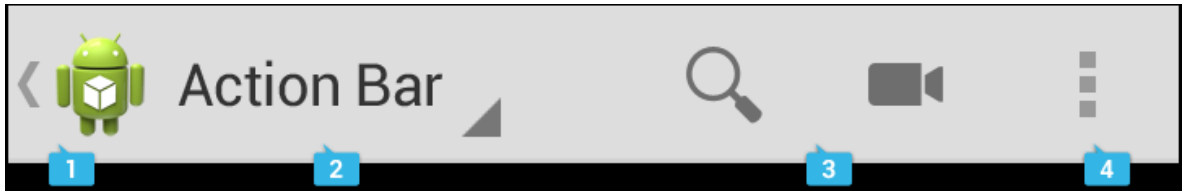
액션바

액션바 (action bar)



액션바 (action bar)

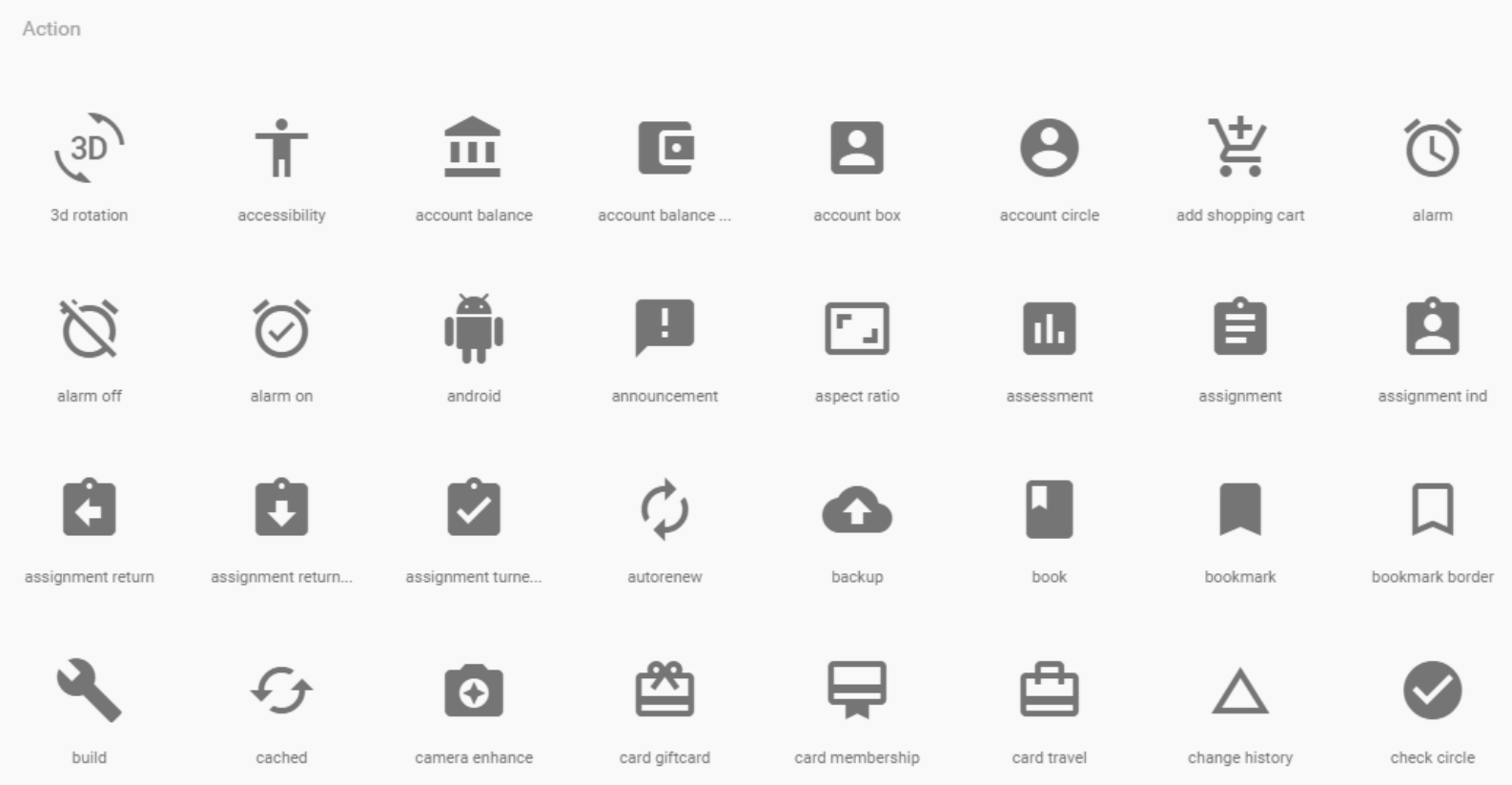
http://developer.android.com/guide/topics/ui/actionbar.html



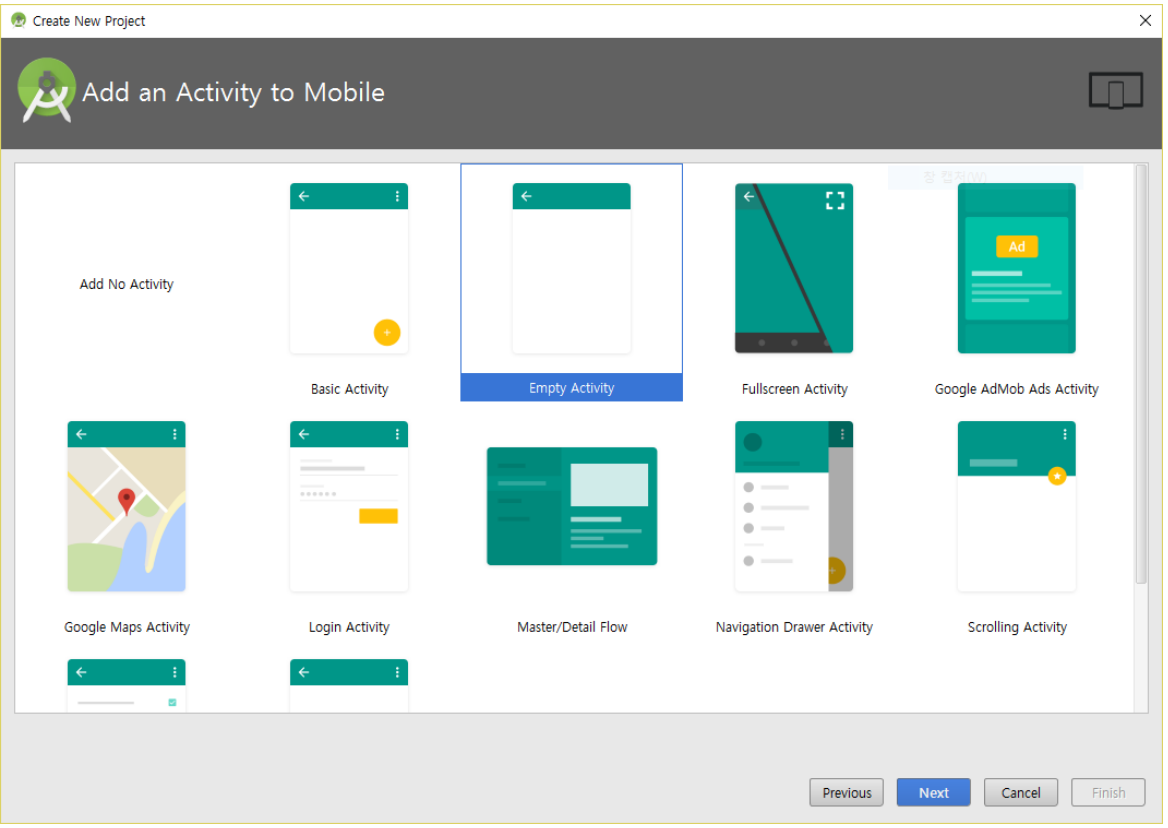
- 안드로이드 애플리케이션에서 중요한 디자인 요소
  - 1: 앱 아이콘, 2: 뷰 컨트롤, 3: 액션 버튼, 4: 액션 오버플로우
- 주요 용도
  - 앱의 아이덴티티를 부여하는 공간 제공 (앱 아이콘, 로고)
  - 검색과 같은 중요 기능을 눈에 띄게 함
  - 앱 내에서 일관된 내비게이션과 뷰 전환을 지원함
    - 탭, 드롭다운 메뉴
  - 별로 사용하지 않는 액션을 액션 오버플로우로 제공하여 산만함을 줄임

참고 - 아이콘 이미지 다운로드

- https://developer.android.com/design/downloads/index.html
- https://github.com/google/material-design-icons



액션바를 포함하는 Activity 생성



- 새 프로젝트를 생성할 때 이미 액션바가 포함되는 Activity를 만들었음
  - AppCompatActivity(ActionBarActivity)를 상속받는 Activity 생성했음
  - Activity의 theme 설정을 보면 DarkActionBar라는 것을 볼 수 있음

AndroidManifest.xml

```
<android:theme="@style/AppTheme" >
```

styles.xml

```
<style name="AppTheme"
parent="Theme.AppCompat.Light.DarkActionBar">
```

액션바에 표시할 액션 메뉴 XML 작성

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

    <item
        android:id="@+id/action_refresh"
        android:icon="@drawable/ic_refresh_white_24dp"
        app:showAsAction="always|withText"
        android:title="refresh"
        />
    <item
        android:id="@+id/action_search"
        android:icon="@drawable/ic_search_white_24dp"
        app:showAsAction="ifRoom"
        android:title="search"
        />
    <item
        android:id="@+id/action_settings"
        android:title="@string/action_settings"
        android:orderInCategory="100"
        app:showAsAction="never"
        />
</menu>
```

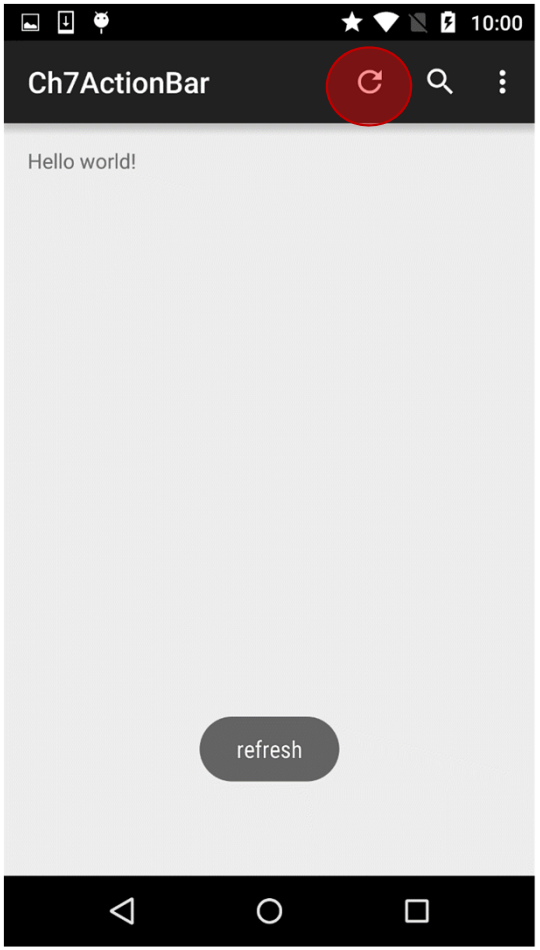
- 이 예제에서는 res/menu 디렉토리에 action_bar.xml 파일 생성

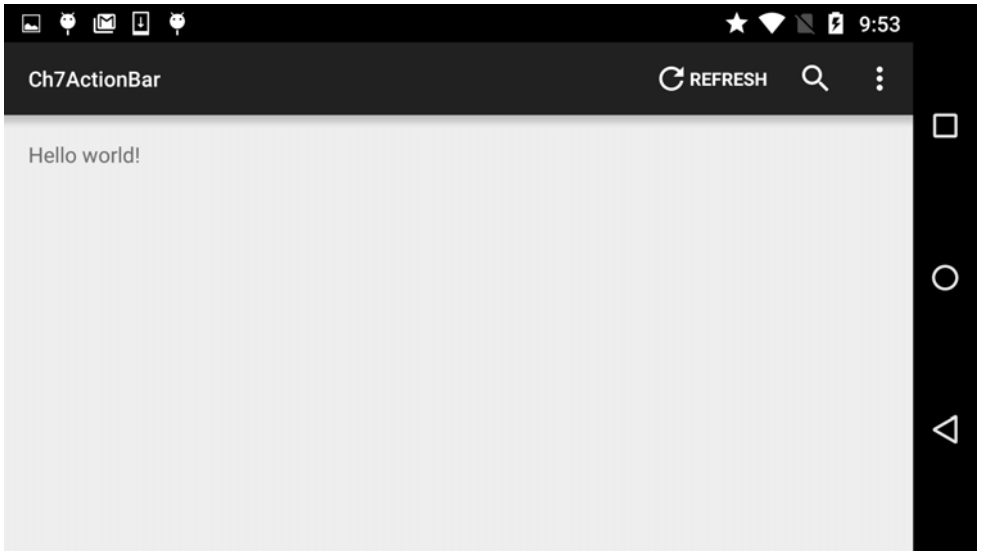
코드 작성

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

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
        switch(item.getItemId()) {
         case R.id.action_refresh:
            Log.v("ActionBar", "refresh button");
            Toast.makeText(getApplicationContext(),
"refresh", Toast.LENGTH_SHORT).show();
            return true;
         case R.id.action_search:
            Log.v("ActionBar", "search button");
            Toast.makeText(getApplicationContext(), "search",
Toast.LENGTH_SHORT).show();
            return true;
         case R.id.action_settings:
            Log.v("ActionBar", "setting button");
            Toast.makeText(getApplicationContext(),
"settings", Toast.LENGTH_SHORT).show();
            return true;
         default:
            return super.onOptionsItemSelected(item);
      }
}
```

실행 화면





- 액션바 예제 프로젝트: Ch7ActionBar