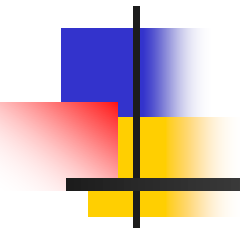
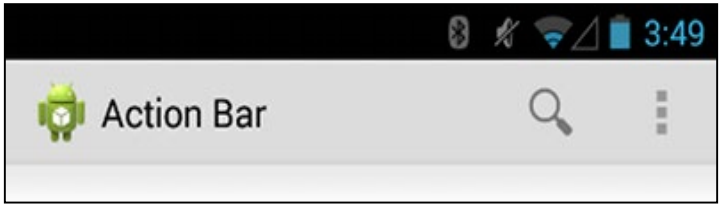


Menu(실습)

배 희호 교수
경복대학교
소프트웨어융합과

Android의 User Interface

- Action Bar – Toolbar
  - Action Bar는 상당히 중요한 구조적인 요소
  - Action Bar는 Action들 사이의 Navigation을 제공

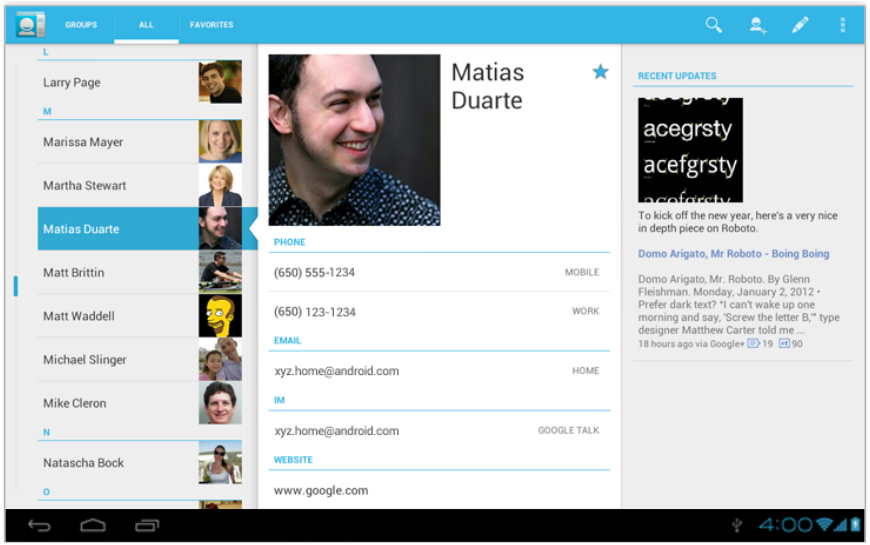


Title bar

- Navigation Bar
  - Android 4.0부터 전통적인 H/W Key를 대체하는 Navigation Bar 등장
  - Navigation Bar를 통하여 뒤로 가기, Home, 최근 App Button 제공

Android의 User Interface

- 다중 Panel Layout
  - http://klutzy.github.io/android-design-ko/patterns/multi-pane-layouts.html



- Gesture
  - Gesture는 제공된 Screen 객체를 사용자가 조작하여 App과 상호 작용하는 방법

핀치 오픈

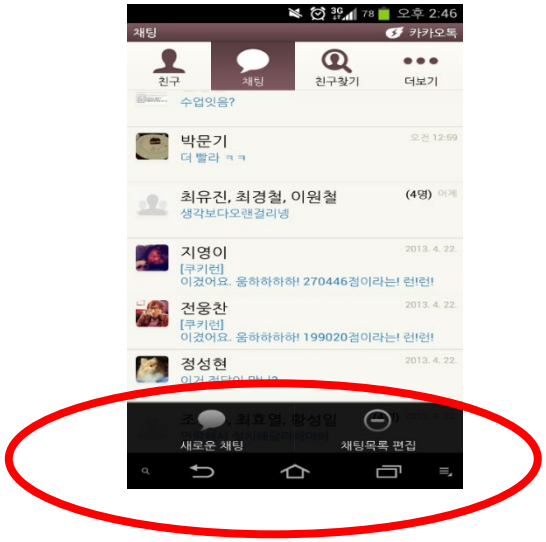
핀치 클로즈

Android의 User Interface

- Toast Message/SnackBar
- Dialog(대화상자)

Menu

- Menu는 많은 Application에서 사용자에게 전반적인 기능을 노출한 친숙한 User Interface 요소
- Menu는 PC Application에 많이 사용
  - Application에서 제공하는 모든 기능을 계층적으로 표시할 수 있기 때문임 (3단계 구분)
- Mobile Device에서도 Application에 대한 추가 Option을 표시하기 위하여 Menu를 사용



사용자의 명령을 받아들이는 Interface

Menu

- Menu 구성만 봐도 App이 어떤 기능들을 제공하고 있는지 쉽게 유추할 수 있음
- 그뿐만 아니라 Option Menu의 경우 Menu를 구동하는 별도의 Key가 존재하므로 접근성도 높음
- Menu가 사용하는 기본 Windows는 TYPE_APPLICATION_ATTACHED_DIALOG Type

Menu

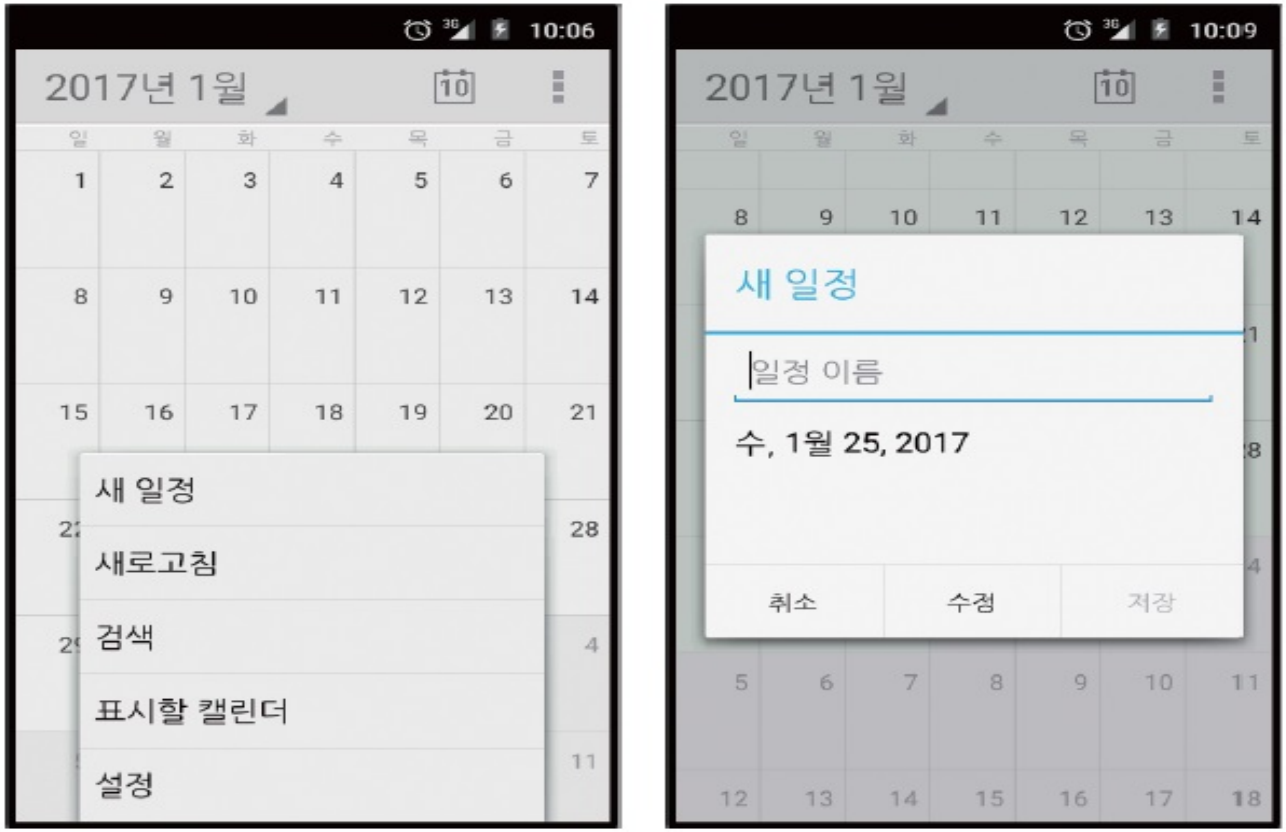
- Menu 3가지 Type
  - Option Menu
    - Device의 Hardware Menu key(일부 기기에서는 SoftKey로 되어 있음)를 눌렀을 때 아래로 쑥 내려가는 Menu
    - 허니콤(api 10)버전 이후부터 ActionBar의 개념이 생기면서 Option Menu의 위치가 ActionBar로 변경됨
  - Context Menu
    - 특정 View를 오래 누르고 있으면 나타나는 Menu
    - 그 뷰와 관련된 작업을 제공
      (일반 PC에서 Mouse 오른쪽 Button의 역할)
  - Popup Menu
    - 특정 명령에 따라 특정 위치에 Popup되어 Sub Menu 용도로 사용할 수 있는 Menu

Menu

- 기타 메뉴
  - Navigation Drawer
    - Navigation Drawer는 화면의 왼쪽 가장자리에서 Slide 하여 나타나는 Menu
    - App 내의 주요 Section을 탐색할 수 있는 옵션을 제공
    - DrawerLayout과 NavigationView를 사용하여 Navigation Drawer를 구현할 수 있음
  - Bottom Navigation
    - Bottom Navigation은 화면 하단에 고정된 Menu
    - 사용자가 주요 Section을 탐색할 수 있는 옵션을 제공
    - BottomNavigationView를 사용하여 Bottom Navigation을 구현할 수 있음

Menu

- Option Menu와 Context Menu



(a) 옵션 메뉴 (b) 컨텍스트 메뉴(날짜를 롱클릭한 화면)

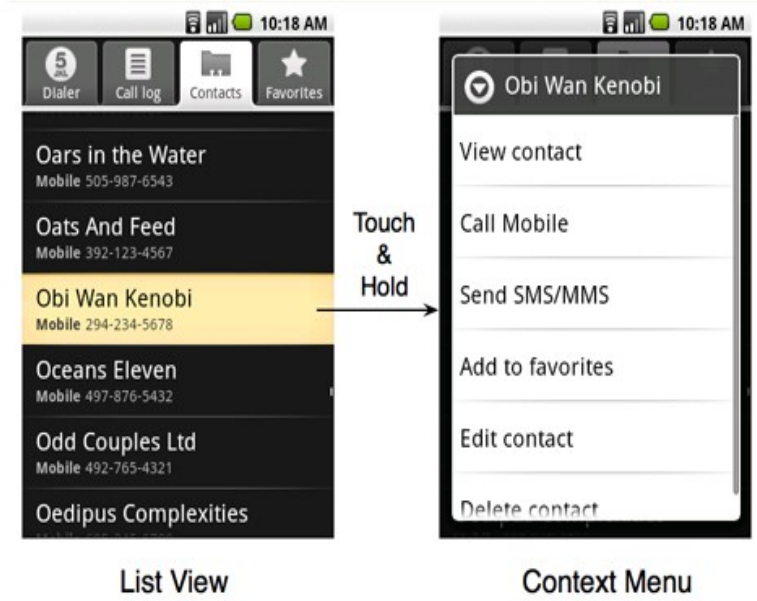
그림 7-1 옵션 메뉴와 컨텍스트 메뉴의 예

Menu

- Option Menu
  - 위치는 상단에 고정되며, 최대 6개까지 표시할 수 있으며 6개를 초과할 경우에는 More 항목이 생기면서 List Popup으로 추가 Menu를 볼 수 있도록 제공
  - 사용자가 자주 접근할 수 있는 주요 옵션을 제공
  - onCreateOptionsMenu() 메소드를 사용하여 옵션 메뉴를 정의

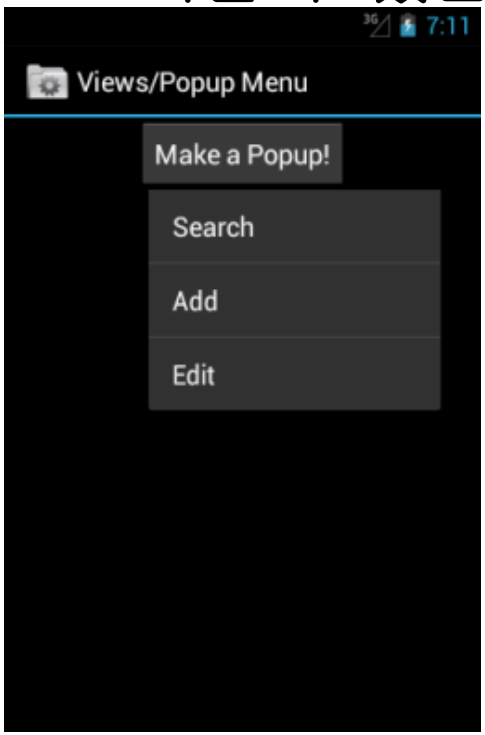
Menu

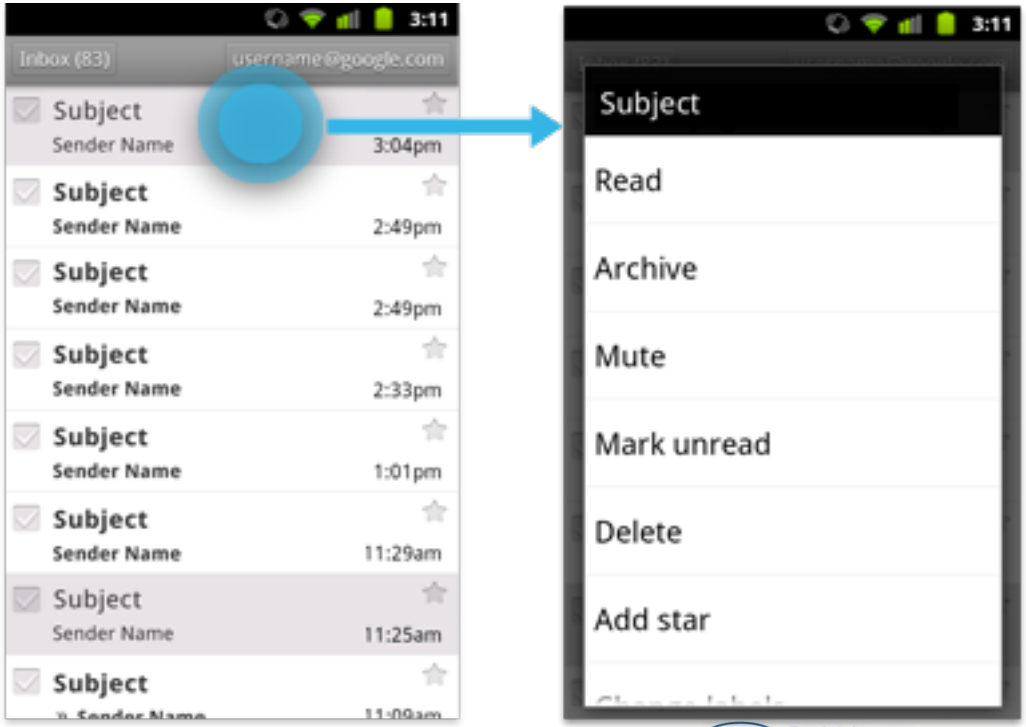
- Context Menu
  - 원하는 Widget을 길게 터치(long Touch)를 할 때 반응하는 Menu로 List Popup 형태로 Menu를 선택할 수 있음
  - registerForContextMenu() 메소드를 사용하여 View에 Context Menu를 등록하고, onCreateContextMenu() 메소드를 사용하여 Menu를 정의



Menu

- Popup Menu
  - Popup Menu는 특정 View를 클릭했을 때 나타나는 Menu
  - 일반적으로 간단한 옵션을 제공
  - PopupMenu 클래스를 사용하여 팝업 메뉴를 생성하고 표시할 수 있음





Menu

- Context Menu와 Popup Menu

	Context Menu	Popup Menu	비고
호출 시점	길게 누를 때	원하는 때에	여는 속도가 신속
열리는 위치	화면 중앙	앵커 뷰 근처	이동 거리를 최소화
열렸을 때 배경화면	흐려진다	변화가 없음	작업 대상을 명확히 볼 수 있음

Menu

- Menu를 생성하는 2가지 방법
  - Menu XML File 생성 후 JAVA에서 호출 (기본적인 방법)
    - Menu와 Menu Item은 XML Resource를 사용해서 정의
    - Activity나 별도로 추가한 View 클래스 등의 Application JAVA Code에서 이것을 사용해 Menu를 완성
  - JAVA Code만으로 Menu를 생성하는 방법

Menu의 내용은 Application Code(.java)에 직접 Coding 하는 것보다 XML Resource를 사용해서 따로 정의하는 것이 Menu의 구조를 이해하는데 더 좋음

Menu Inflation

- Menu Resource를 Inflate(팽창)하면 실제 Menu가 생성

Menu Inflation

- Inflator(인플레이터)
  - Inflator는 '부풀리는 장치' 또는 '자전거 등의 공기 펌프'라는 뜻
  - 풍선에 바람을 넣어서 실제 객체로 만들어 사용한다는 의미
  - Android에서 사용되는 Inflator를 이와 같이 생각하면, 정적으로 존재하는 XML File을 JAVA Code에서 접근하여 실제 객체로 만들어서 사용하는 것이라고 볼 수 있음
  - Menu Inflator 객체는 Menu XML File을 JAVA Code에서 가져와 사용하는 것이고, Layout Inflator는 Layout XML 파일을 JAVA Code에서 가져와 사용하는 것

Menu Inflation

- Option Menu Inflation 생성 과정
  - /res/menu 경로에 Menu Resource File 생성
  - Activity의 onCreateOptionMenu() Override
  - Activity의 onPrepareOptionMenu() Override
  - Activity의 onOptionsItemSelected() Override
  - Activity의 onOptionsMenuClosed() Override

Menu Inflation

- MenuInflater 클래스
  - Menu를 정의하는 Resource File을 이용하여 Menu 객체를 생성

```
OnCreateOptionsMenu(Menu menu)
 MenuInflater inflater = getMenuInflater();
 inflater.inflate(R.menu.menutest, menu);
```

Menu Resource File

- Menu Resource File 생성
  - Menu 구조를 Program이 아닌 외부 Resource File에서 정의
  - 필요할 때 Inflation시켜 Menu 객체 생성
  - Resource File로 Menu 구조를 정의 : menutest.xml

```
<menu> …
    <item>
        <menu> …
        </menu>
    </item> …
</menu>
```

Menu Resource File

- Menu Resource XML

요소	설명
<menu>	Menu 항목의 Container로 root node로 생성 < item>과 <group> 요소를 이용하여 Menu를 구성
<item>	하나의 Menu 항목을 나타냄 하위 Menu를 생성하기 위해 <menu> 요소를 포함할 수 있음
<group>	<item> 요소를 묶어 관리할 수 있는 투명 Container

Menu Resource File

- <item> 속성
  - Menu의 구현은 Text나 Image로 하는데 모두 item 요소를 이용해서 표현
  - 주요 속성

속성	설명
android:id	item의 고유한 리소스 ID Application에서 이 ID를 통해 해당 항목을 인식 하고 사용
android:icon	Menu에 Image를 사용하는 경우 @drawable의 참조
android:title	Menu에 사용할 문자열
android:showAsAction	App Bar에서 작업 항목으로 나타나는 시기와 방법을 지정

Menu Resource File

- android:showAsAction
  - item의 속성 값에서 보다 다양한 설정을 하기 위해서는 showAsAction 속성을 이용

속성값	설명
ifRoom	App Bar에 표시할 공간이 있으면 표시하고 아니면 더 보기 Menu에 포함
withText	Icon과 android:title에서 정의한 Text를 표시
never	App Bar에 공간이 있어도 배치하지 않고 더 보기 Menu에 포함
always	항상 App Bar에 배치 다른 UI와 겹쳐질 수 있음
collapseActionView	접기 기능을 사용할 수 있음 (API level 14)

Menu Resource File

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
  <group android:checkableBehavior="single"
      android:enabled="true">
      <item
          android:id="@+id/black"
          android:checked="true"
          android:title="검정" />
      <item
          android:id="@+id/red"
          android:title="빨강" />
      <item
          android:id="@+id/green"
          android:title="녹색" />
  </group>
</menu>
```

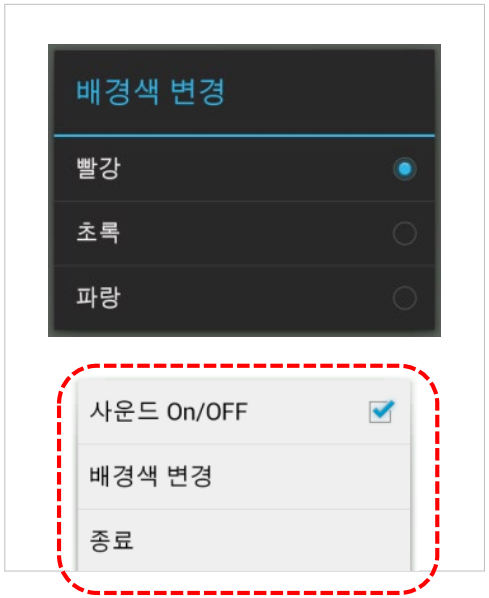
<menu> 엘리먼트는 Menu를 생성
이것은 Menu 항목을 담는 container
반드시 root node이어야 하며
<item>이나 <group>을 하나 이상 포함

<item> 엘리먼트는 MenuItem을 생성
이것은 Menu 항목을 나타냄
Sub Menu를 작성하려면
<menu>를 포함할 수 있음

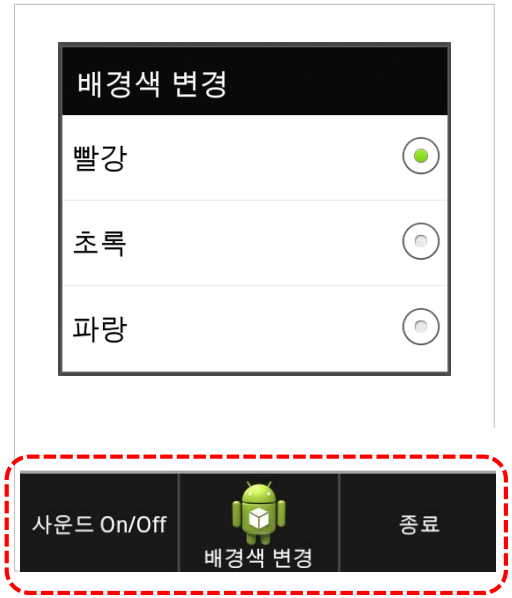
Menu 문제점

- Menu Theme와 Menu Key의 문제점
  - Menu 역시 아래 그림과 같이 Theme별로 외형의 차이가 큼

API 16 디바이스에서 실행(Holo 테마)



API 10 디바이스에서 실행(기본 테마)



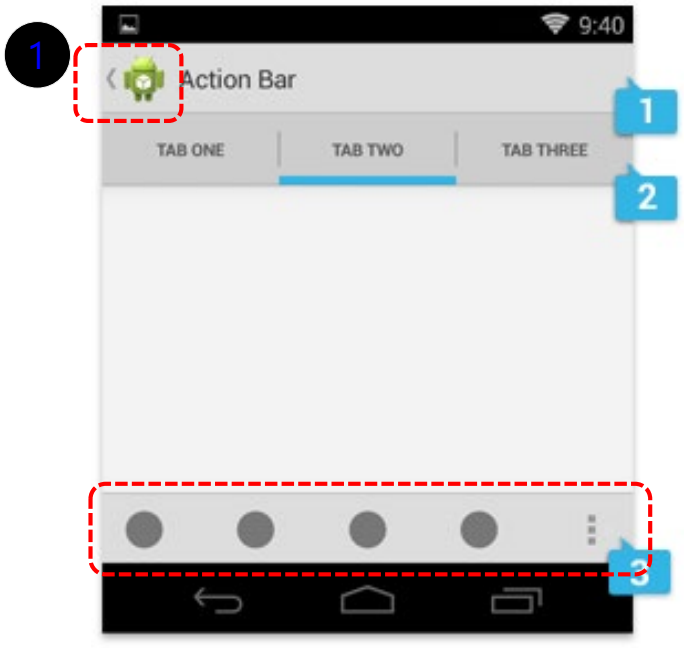
  - Option Menu는 Android API 10에서 Item들이 수평으로 배치되고, 심지어 Icon까지 지원

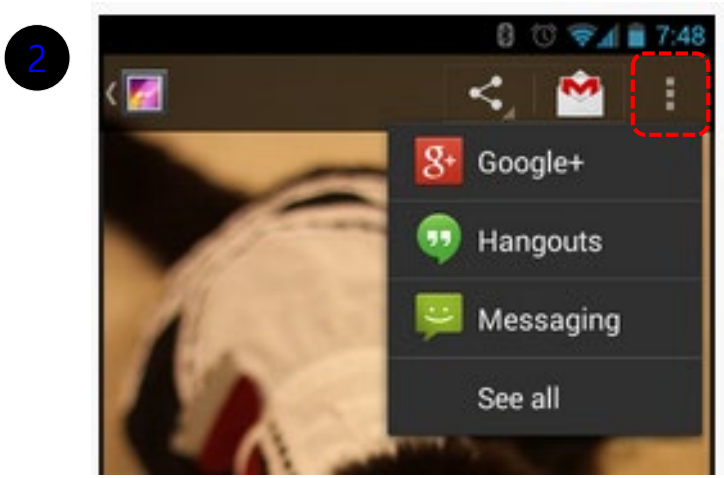
Menu 문제점

- Menu Theme와 Menu Key의 문제점
  - Option Menu와 Context Menu는 각각 사용자가 Menu Key를 누르거나, 특정 View를 길게 눌러야 활성화 됨
  - 외관으로 봐서는 Menu가 존재하는지 알 방법이 없음
  - 또한 모든 App들이 Menu를 제공하는 것은 아니므로 사용자는 Menu가 제공되는지 확인하기 위해 일일이 Menu Key 등을 눌러봐야 함
  - 감춰진 Menu는 직관적이지 않아 실용성이 떨어진다는 것
  - 이러한 문제들로 인해, 현재 많은 개발자들은 Option Menu나 Context Menu를 거의 사용하지 않는 추세임

Menu 문제점

- Android는 이 문제를 직시하고 API 11부터 ActionBar를 해결책으로 제시

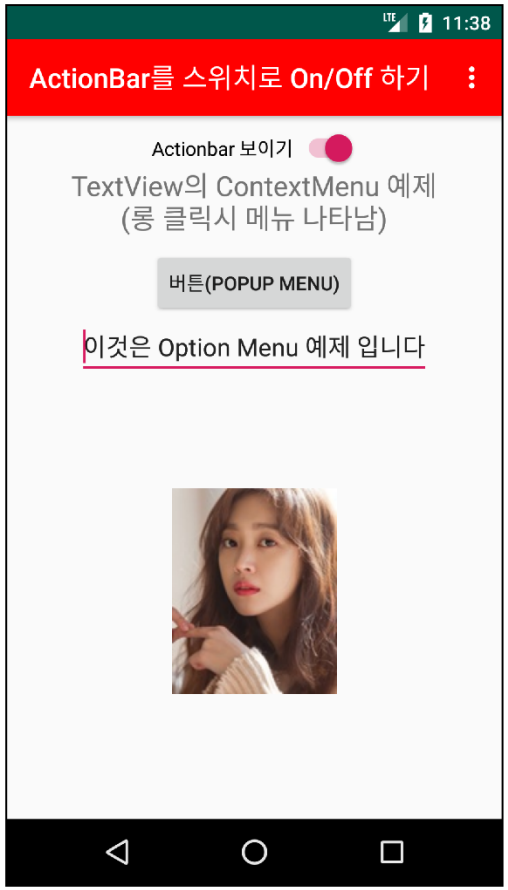


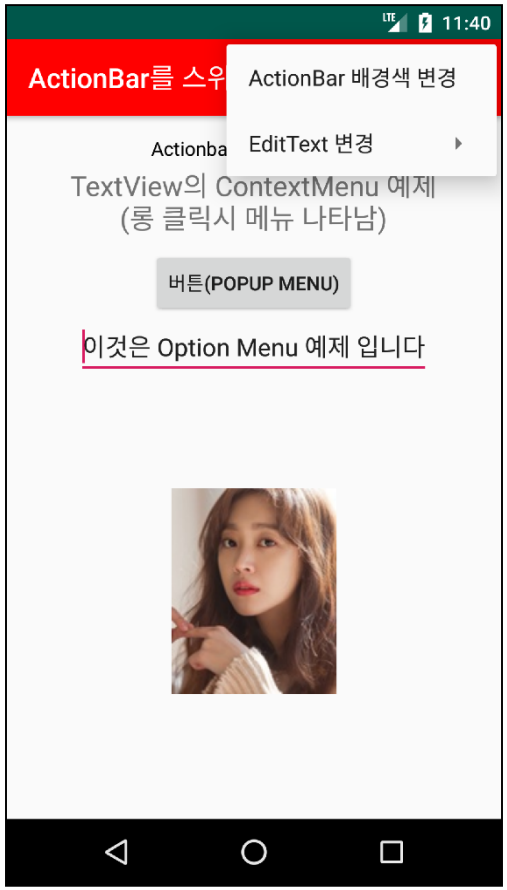


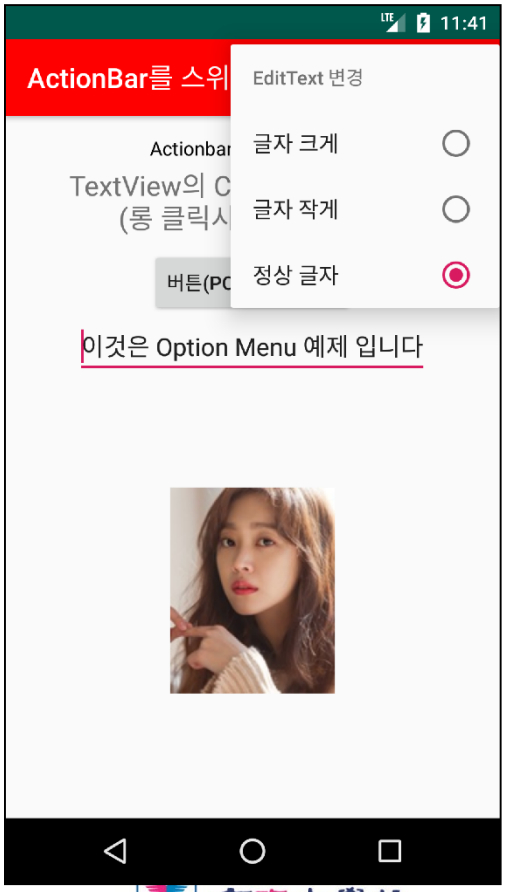
- Menu를 활성화하는 Button이 모두 화면에 표시
- ActionBar는 기능이 많고, Fragment와 함께 사용되기도 함

Menu

- Main Activity 상의 각종 View(TextView, Button, EditText, ImageView)를 롱 클릭 하면 각종 Menu가 나타나는 프로그램을 작성해보자

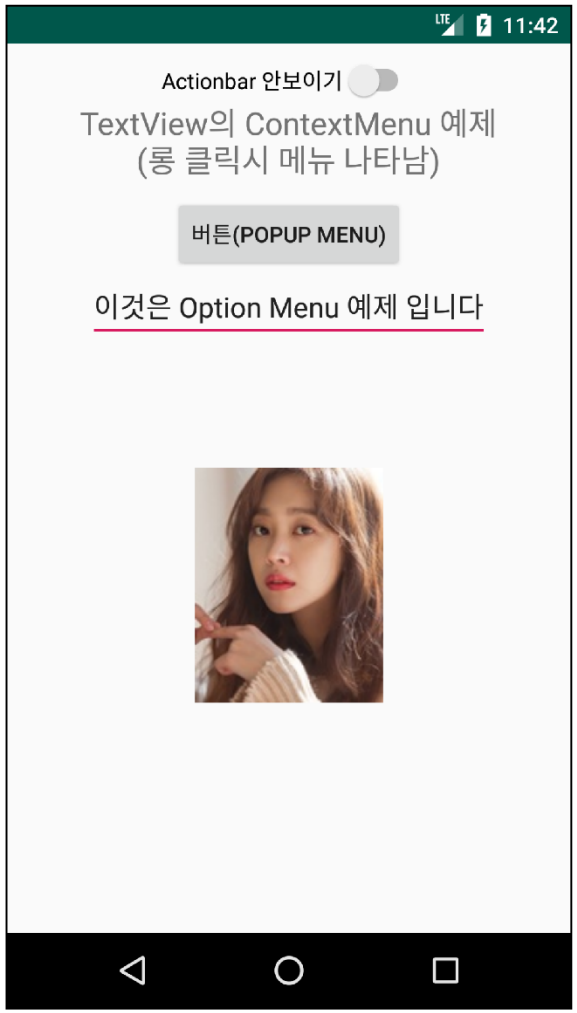


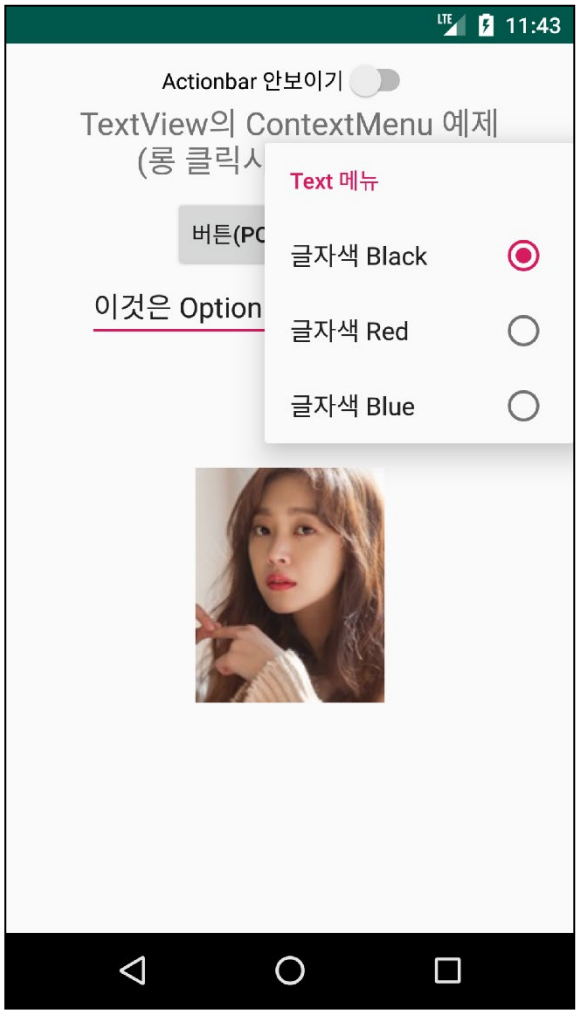


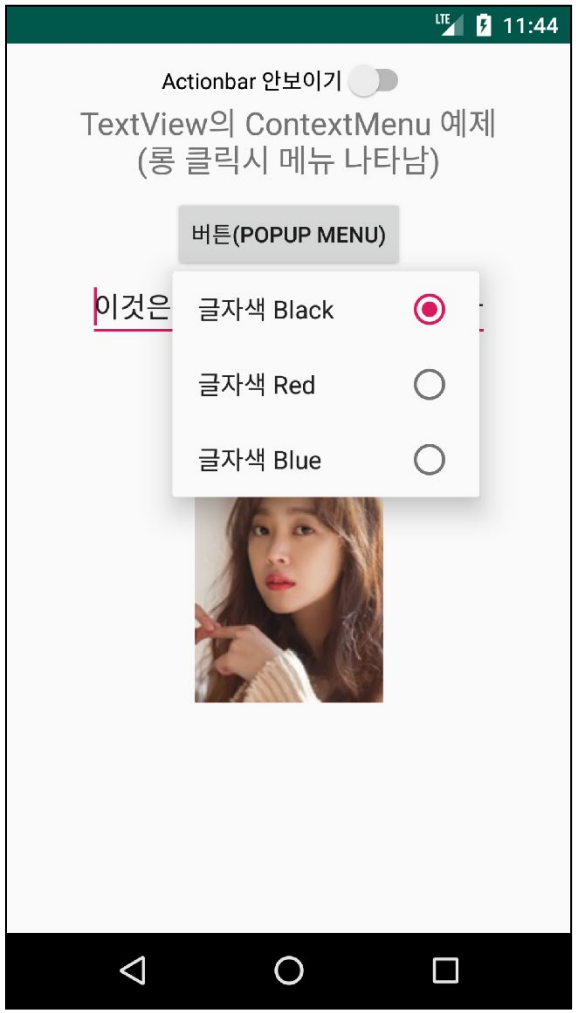


Menu

실행 결과

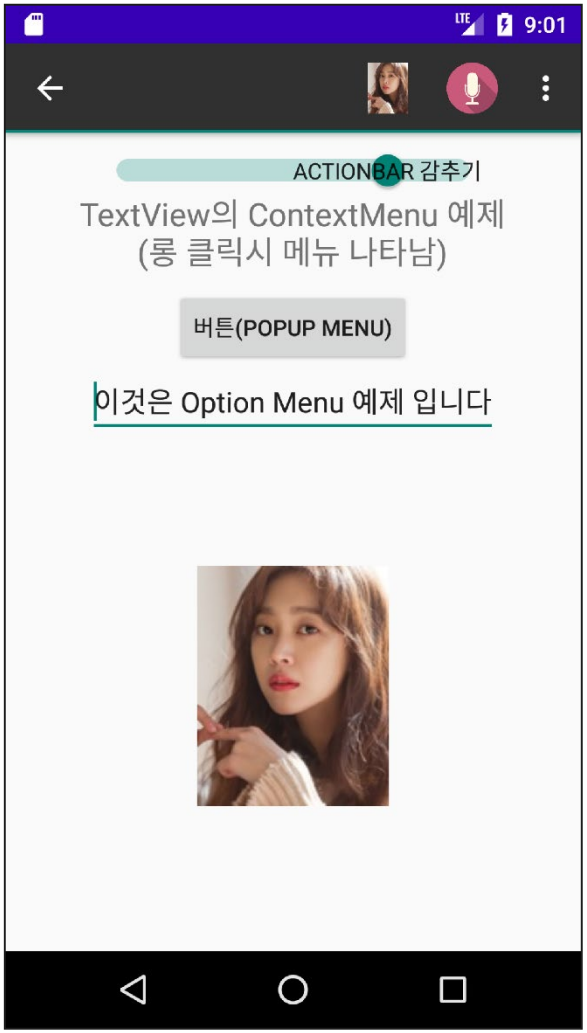


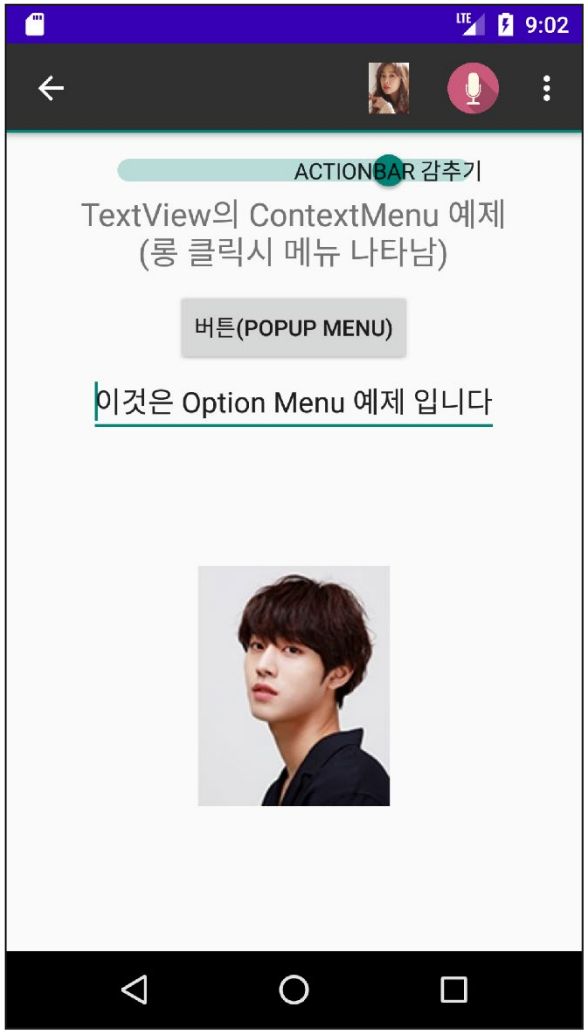


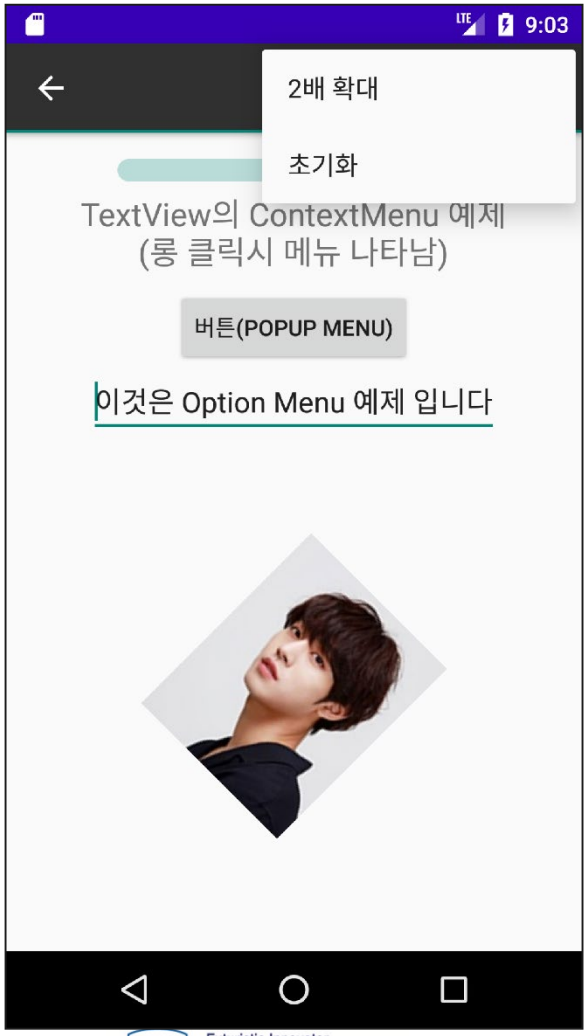


Menu

실행 결과







Menu

- Program 조건
  - ActionBar를 선택적으로 보여주기
    - Option Menu에서 배경 색상 변경
    - 배경 색상 변경을 위해서 Dialog를 사용할 것
  - TextView는 Long Click시에 Context Menu 실행
    - Floating 방식
  - Button은 Click시 Toast Message 출력
    - Button에 ListPopupWindow를 연결하여 배경색을 변경할 것
  - Button을 Long Click시에 Popup Menu 실행
  - EditText는 Option Menu로 글자색 변경 처리
  - ImageView는 Long Click시에 Context Menu 실행
    - Action 방식

Menu

- Options Menu
  - Windows 상단의 ActionBar나 Toolbar에 기본적으로 표시되는 Menu
  - App의 주요 기능들을 제공하며, Option Menu는 기기마다 설정 Button을 눌러 표시될 수도 있음
  - 구현 방법
    - onCreateOptionsMenu() 메소드를 Override하여 Menu 항목을 설정
    - Menu 항목을 XML File로 정의한 후 inflate하여 사용할 수 있음
  - 주요 기능
    - 기본적으로 icon이나 Text로 Menu 항목을 추가할 수 있음
    - 특정 Menu 항목을 AppBar에 고정해 자주 사용하는 기능을 강조할 수 있음

Menu

- Context Menu
  - 특정 항목이나 요소를 길게 누를 때 표시되는 Menu로, 요소와 관련된 추가 기능을 제공
  - 예) List item을 길게 누를 때 삭제, 수정 등의 Menu Option을 제공할 수 있음
  - 구현 방법
    - registerForContextMenu(View view) 메소드를 사용해 View와 연결
    - onCreateContextMenu() 메소드를 Override하여 Menu 항목을 정의
  - 주요 기능
    - Menu가 열리는 동안 특정 요소에 대한 추가 작업을 수행할 수 있음
    - 보통 목록에서 특정 Item과 관련된 작업을 제공하는 데 유용

Menu

- Context Menu 작성 방법
  - [res]-[menu] folder 생성
  - Menu XML File 생성 및 Coding
  - Activity(*.java)에 onCreate( ) 메소드 내부에서 registerForContextMenu() 메소드로 등록
    - Menu를 사용할 Widget을 등록하기 위함
    - Layout에 연결하면 모든 화면에 적용할 수 있음
  - Activity(*.java)에 onCreateContextMenu() 메소드 Overriding
    - Menu File 등록을 위함
  - Activity(*.java)에 onContextItemSelected( ) 메소드 Overriding
    - Menu 선택 시 동작을 위함

Menu

- Popup Menu
  - Windows의 특정 위치에 작은 Popup 형태로 표시되는 Menu
  - 보통 Button 또는 Icon Click 시 Popup Menu를 띄워 관련 Option을 제공하는 방식으로 사용
  - 구현 방법
    - PopupMenu 클래스를 사용해 Menu를 정의하고 표시
    - PopupMenu의 show() 메소드를 사용하여 Popup을 표시
  - 주요 기능
    - Menu를 특정 위치에 유연하게 표시할 수 있어 직관적인 UX 제공
    - 사용자가 특정 Action을 수행하기 전 확인 Message를 제공하는 데 유용

Menu

- Popup Menu 작성 방법
  - Popup Menu메뉴 생성

```
PopupMenu
    (현재 화면의 제어권자, 팝업을 뛰울 기준 좌표 위젯);
```

  - menu Resource에서 Menu 불러오기

```
getMenuInflater().inflate(R.menu.menu_main,
                                        p.getMenu())
```

  - PopupMenu.setOnMenuItemClickListener() 에서 onMenuItemClick() Overriding
  - PopupMenu.show()로 화면에 띄우기

Menu

- Popup Menu를 생성하는 순서
  - PopupMenu 클래스의 생성자로 Popup Menu 객체를 생성
  - 생성자는 현재 Application의 context와 Menu가 연결되는 Anchor View를 인수로 받음

```
PopupMenu popupMenu = new
                PopupMenu(getApplicationContext(), v);
```

  - MenuInflater 이용하여 XML로 정의된 Menu Resource를 popupMenu.getMenu()가 반환하는 Menu 객체에 추가

```
MenuInflater inflater = popupMenu.getMenuInflater();
Menu menu = popupMenu.getMenu();
inflater.inflate(R.menu.popup_menu, menu);
```

  - PopupMenu.show()를 호출

Menu

- 사용자 인터페이스

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center|top"
    android:orientation="vertical"
    android:padding="10dp"
    tools:context=".MainActivity">
    <Switch
        android:id="@+id/switch1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:showText="true"
        android:switchMinWidth="50dp"
        android:textOff="ActionBar 보이기"
        android:textOn="ActionBar 감추기"
        tools:ignore="UseSwitchCompatOrMaterialXml" />
```

Menu

- 사용자 인터페이스

```
<TextView
    android:id="@+id/text"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:text="TextView의 ContextMenu 예제₩n(롱 클릭시 메뉴 나타남)-XML"
    android:textSize="20dp" />

<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="10dp"
    android:text="버튼(PopUp menu)" />
```

Menu

- 사용자 인터페이스

```
    <EditText
        android:id="@+id/edittext"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="이것은 Option Menu 예제 입니다" />

    <ImageView
        android:id="@+id/image"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="80dp"
        android:src="@drawable/picture" />
</LinearLayout>
```

Menu(ActionBar)

- ActionBar 처리 내용

```
ActionBar actionBar = getSupportActionBar();
actionBar.setTitle("ActionBar를 스위치로 On/Off 하기");
actionBar.setBackgroundDrawable(new ColorDrawable(current));
Switch toggle = findViewById(R.id.switch1);
toggle.setOnCheckedChangeListener(
                                new CompoundButton.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(CompoundButton buttonView,
                                 boolean isChecked) {
        if (isChecked) {
            actionBar.show();
        } else {
            actionBar.hide();
        }
    }
});
toggle.performClick();
```

Menu(TextView의 ContextMenu)

- Context Menu는 Long Click 시 개별 View를 동작시키거나 Gallery와 E-mail App처럼 평소에는 보이지 않다가 Long Click 시 여러 개의 CheckBox가 나오면서 선택할 수 있게 하기 위해 사용

- Context Menu 2가지 방법으로 구현
  - Floating(플로팅) 방식
  - Action(액션) 방식

Menu(TextView의 ContextMenu)

- textmenu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <group android:checkableBehavior="single">
        <item
            android:id="@+id/itemBlack"
            android:title="글자색 Black"
            android:checked="true"/>
        <item
            android:id="@+id/itemRed"
            android:title="글자색 Red"/>
        <item
            android:id="@+id/itemBlue"
            android:title="글자색 Blue"/>
    </group>
</menu>
```

Menu(TextView의 ContextMenu)

- TextView의 ContextMenu 구현 내용
  - Floating 방식
    - Floating 방식은 Option Menu에서 구현했던 방법과 유사해서 간단함
    - Event Listener 부분이 거의 비슷해 메소드명만 다르고 onCreateContextMenu()에서 여러 View가 적용될 수 있기 때문에 View의 객체인 v를 이용해 현재 보고 있는 Resource Id를 가져와 inflater 객체를 만들어 줌
    - Floating 방식은 구현 후에 Activity에 적용하기 위해서 registerForContextMenu()를 통해 View에 Context Menu를 등록해야 함

```
textView = findViewById(R.id.text);
registerForContextMenu(textView);
```

Menu(TextView의 ContextMenu)

- TextView의 ContextMenu 구현 내용

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
                                ContextMenu.ContextMenuInfo menuInfo) {
    MenuInflater menuInflater = getMenuInflater();
    if (v == textView) {
        menu.setHeaderTitle("Text 메뉴");
        menu.setHeaderIcon(R.drawable.logo);
        if (codeflag) {
            menuInflater.inflate(R.menu.textmenu, menu);
        } else {
            menu.add(0, 1, 0, "글자색 Black").setChecked(true);
            menu.add(0, 2, 1, "글자색 Red");
            menu.add(0, 3, 2, "글자색 Blue");
            menu.setGroupCheckable(0, true, true);
        }
```

Menu(TextView의 ContextMenu)

- TextView의 ContextMenu 구현 내용

```
        if (type1 == 1) {
            if (codeflag)
                menu.findItem(R.id.itemRed).setChecked(true);
            else
                menu.findItem(2).setChecked(true);
        } else if (type1 == 2) {
            if (codeflag)
                menu.findItem(R.id.itemBlue).setChecked(true);
            else
                menu.findItem(3).setChecked(true);
        }
    }
}
```

Menu(TextView의 ContextMenu)

- Floating은 Click Event Listener인 onContextItemSelected()가 Option Menu의 Click Event Listener와 비슷하게 구현
- Event Listener의 매개변수인 item으로 알맞는 item 객체를 switch로 찾아내 기능에 맞는 실행문을 작성해주면 됨

Menu(TextView의 ContextMenu)

- TextView의 ContextMenu 구현 내용

```
@Override
public boolean onContextItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId()) {
        case R.id.itemBlack:
        case 1:
            textView.setTextColor(Color.BLACK);
            type1 = 0;
            break;
        case R.id.itemRed:
        case 2:
            textView.setTextColor(Color.RED);
            type1 = 1;
            break;
        case R.id.itemBlue:
        case 3:
            textView.setTextColor(Color.BLUE);
            type1 = 2;
    }
    return true;
}
```

Menu(Button PopupMenu)

- PopupMenu
  - ContextMenu와 다르게 원하는 위치에서 Menu가 보이게 하고 싶을 때 사용하는 Menu
  - ContextMenu처럼 Button을 Long Click했을 때 Button에 Menu 보이기
  - 해당 View 아래에 공간이 있으면 아래에, 없다면 위에 Menu가 나타남
  - OptionMenu나 ContextMenu처럼 Activity에 create하는 메소드가 존재하지 않음
  - Menu 객체가 놓여지 길 원하는 위치에 PopupMenu 객체를 생성하여 붙이고(anchor view 설정) 그 PopupMenu 객체 안에 있는 Menu 객체에 MenuItem을 추가하여 보이도록 함

Menu(Button PopupMenu)

- PopupMenu
  - 다른 Menu들과 마찬가지로 XML로 Menu 항목들을 설계하고 JAVA 언어의 MenuItem 객체로 만들어서(부풀리다 inflate) Menu에 추가하는 방법을 사용함
  - res 폴더에 menu 폴더 생성
    - res 폴더에서 Mouse 오른쪽 Button Menu에서 'Android Resource Directory' 선택 후 위에서 2번째 항목의 'Resource Type'에서 'menu‘ 선택하면 menu 폴더 추가됨
  - menu 폴더에 있는 buttonmenu.xml 문서를 읽어와서 Menu 객체로 만들어주는(부풀려주는inflate) 객체인 MenuInflater사용

Menu(Button PopupMenu)

- buttonmenu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <group android:checkableBehavior="single">
        <item
            android:id="@+id/itemBlack"
            android:title="글자색 Black"
            android:checked="true"/>
        <item
            android:id="@+id/itemRed"
            android:title="글자색 Red"/>
        <item
            android:id="@+id/itemBlue"
            android:title="글자색 Blue"/>
    </group>
</menu>
```

Menu(Button PopupMenu)

Button 구현

```
Button button = findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(getBaseContext(), "버튼을 클릭하였습니다",
                                                    Toast.LENGTH_SHORT).show();
    }
});

button.setOnLongClickListener(new View.OnLongClickListener() {
    @Override
    public boolean onLongClick(View v) {
        popupMenu.show();
        return false;
    }
});
```

Menu(Button PopupMenu)

```
popupMenu = new PopupMenu(this, button);
MenuInflater inflater = popupMenu.getMenuInflater();
inflater.inflate(R.menu.buttonmenu, popupMenu.getMenu());
popupMenu.setOnMenuItemClickListener(
                                        new PopupMenu.OnMenuItemClickListener() {
    @Override
    public boolean onMenuItemClick(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.itemBlack1:
                button.setTextColor(Color.BLACK);
                break;
            case R.id.itemRed1:
                button.setTextColor(Color.RED);
                break;
            case R.id.itemBlue1:
                button.setTextColor(Color.BLUE);
        }
        item.setChecked(true);
        return true;
    }
});
```

Menu(ListPopupWindow)

- ListPopupWindow
  - 어떤 목록을 보여줄 때 Popup으로 보여주고 싶다면 ListPopupWindow Widget을 사용
  - PopupMenu와 유사하지만 목록을 Adapter로 받는 점이 특이함
- 폭과 높이는 픽셀 단위로 지정하는 메소드
  - void setWidth(int width)
  - void setHeight(int height)
  - void setContentWidth(int width)
- Adapter와 Ancher View 지정하는 메소드
  - void setAdapter(ListAdapter adapter)
  - void setAnchorView(View anchor)

Menu(ListPopupWindow)

- Popup 목록의 동작 방식을 지정하는 메소드
  - void setModal(boolean modal)

- 목록의 item 선택 변경이나 Click 동작에 대한 Event Listener를 등록
  - void setOnItemSelectedListener(AdapterView.OnItemSelectedListener selectedListener)

Menu(ListPopupWindow)

- ListPopupWindow 객체 생성

```
window = new ListPopupWindow(this);
window.setWidth(300);
window.setHeight(600);
window.setAnchorView(button);
window.setAdapter(new ArrayAdapter<>(this,
                                android.R.layout.simple_list_item_1, colors));
window.setModal(true);
window.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    public void onItemClick(AdapterView<?> parent, View view,
                                                        int position, long id) {
        switch (position) {
            case 0:
                button.setBackgroundColor(Color.RED);
                break;
            case 1:
                button.setBackgroundColor(Color.GREEN);
                break;
```

Menu(ListPopupWindow)

- ListPopupWindow 객체 생성

```
            case 2:
                button.setBackgroundColor(Color.BLUE);
                break;
            case 3:
                button.setBackgroundColor(Color.YELLOW);
                break;
            case 4:
                button.setBackgroundColor(Color.CYAN);
                break;
            case 5:
                button.setBackgroundColor(Color.MAGENTA);
        }
    }
});
```

Menu(ListPopupWindow)

- Button 에 ListPopupWindow 연결

```
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (window.isShowing()) {
            window.dismiss();
        } else {
            window.show();
        }
        Toast.makeText(getBaseContext(), "버튼을 클릭하였습니다",
                                                    Toast.LENGTH_SHORT).show();
    }
});
```

Menu(OptionMenu)

- EditText 등록

```
editText = findViewById(R.id.edittext);
editText.setTextSize(18.0f);
```

Menu(OptionMenu)

- optionmenu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/item1"
        android:title="ActionBar 배경색 변경" />
```

Menu(OptionMenu)

- optionmenu.xml

```
<item android:title="TextView 변경">
    <menu>
        <group android:checkableBehavior="single">
            <item
                android:id="@+id/code1"
                android:title="XML로 작성"
                android:checked="true"/>
            <item
                android:id="@+id/code2"
                android:title="JAVA Code로 작성" />
        </group>
    </menu>
</item>
```

Menu(OptionMenu)

- optionmenu.xml

```
<item android:title="EditText 변경">
    <menu>
        <group
            android:checkableBehavior="single">
            <item
                android:id="@+id/itemsize1"
                android:title="글자 크게" />
            <item
                android:id="@+id/itemsize2"
                android:title="글자 작게" />
            <item
                android:id="@+id/itemsize3"
                android:checked="true"
                android:title="정상 글자" />
        </group>
    </menu>
</item>
</menu>
```

Menu(OptionMenu)

OptionMenu Create

```
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.optionmenu, menu);
    return super.onCreateOptionsMenu(menu);
}
```

```
menu.add(0, 1, 0, "ActionBar 배경색 변경");
SubMenu subMenu = menu.addSubMenu(0, 2, 0, "EditText 변경");
subMenu.add(0, 3, 0, "글자 크게" );
subMenu.add(0, 4, 1, "글자 작게" );
subMenu.add(0, 5, 2, "정상 글자" ).setChecked(true);
subMenu.setGroupCheckable(0, true, true);
```

Menu(OptionMenu)

- OptionMenu 처리

```
public boolean onOptionsItemSelected(MenuItem item) {
    if (item.getItemId() == R.id.item1) {
        ColorSetting colorSetting = new ColorSetting(MainActivity.this, actionBar);
        colorSetting.setColor();
    } else if (item.getItemId() == R.id.itemsize3) {
        editText.setTextSize(18.0f);
    } else if (item.getItemId() == R.id.itemsize1) {
        editText.setTextSize(24.0f);
    } else if (item.getItemId() == R.id.itemsize2) {
        editText.setTextSize(14.0f);
```

Menu(OptionMenu)

OptionMenu 처리

```
} else if (item.getItemId() == R.id.code1) {
    textView.setText("TextView의 ContextMenu 예제₩n(롱 클릭시 메뉴 나타남)-XML");
    codeflag = true;
} else {
    textView.setText("TextView의 ContextMenu 예제₩n(롱 클릭시 메뉴 나타남)-JAVA");
    codeflag = false;
}
item.setChecked(true);
return true;
```

Menu(OptionMenu)

- custom_dialog.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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/redText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Red : "
        android:textColor="#ff0000"
        android:textSize="20dp" />
    <SeekBar
        android:id="@+id/redBar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:max="255"
        android:progress="0" />
```

Menu(OptionMenu)

- custom_dialog.xml

```
<TextView
    android:id="@+id/greenText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Green : "
    android:textColor="#00FF00"
    android:textSize="20dp" />

<SeekBar
    android:id="@+id/greenBar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:max="255"
    android:progress="0" />
```

Menu(OptionMenu)

- custom_dialog.xml

```
<TextView
    android:id="@+id/blueText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Blue : "
    android:textColor="#0000FF"
    android:textSize="20dp" />

<SeekBar
    android:id="@+id/blueBar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:max="255"
    android:progress="0" />
```

Menu(OptionMenu)

- custom_dialog.xml

```
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:background="#FFFFFF"
        android:padding="10dp"
        android:text="변경 색상"
        android:textColor="#000000" />
</LinearLayout>
```

Menu(OptionMenu)

- ColorSetting.JAVA

```
public class ColorSetting {
    Context context;
    ActionBar actionBar;

    TextView textView1, textView2, textView3;
    Button button;
    SeekBar seekBar1, seekBar2, seekBar3;

    public ColorSetting(Context context, ActionBar actionBar) {
        this.context = context;
        this.actionBar = actionBar;
    }
```

Menu(OptionMenu)

- ColorSetting.JAVA

```
public void setColor() {
    LayoutInflater inflater = LayoutInflater.from(context);
    final View customView = inflater.inflate(R.layout.custom_dialog, null);

    textView1 = customView.findViewById(R.id.redText);
    textView2 = customView.findViewById(R.id.greenText);
    textView3 = customView.findViewById(R.id.blueText);
    button = customView.findViewById(R.id.button);
    button.setBackgroundColor(MainActivity.current);
```

Menu(OptionMenu)

- ColorSetting.JAVA

```
seekBar1 =customView.findViewById(R.id.redBar);
seekBar1.setOnSeekBarChangeListener(
                                new SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress,
                                                    boolean fromUser) {
        changeColor();
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {
    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
    }
});
```

Menu(OptionMenu)

- ColorSetting.JAVA

```
seekBar2 = customView.findViewById(R.id.greenBar);
seekBar2.setOnSeekBarChangeListener(
                                new SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress,
                                                    boolean fromUser) {
        changeColor();
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {
    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
    }
});
```

Menu(OptionMenu)

- ColorSetting.JAVA

```
seekBar3 = customView.findViewById(R.id.blueBar);
seekBar3.setOnSeekBarChangeListener(
                        new SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress,
                                                    boolean fromUser) {
        changeColor();
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {
    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
    }
});
```

Menu(OptionMenu)

- ColorSetting.JAVA

```
AlertDialog.Builder dialog = new AlertDialog.Builder(context);
dialog.setIcon(R.drawable.icon_mic1);
dialog.setTitle("ActionBar 배경색을 선택해주세요");
dialog.setView(customView);
dialog.setPositiveButton("확인", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        actionBar.setBackgroundDrawable(new ColorDrawable(MainActivity.current));
    }
});
dialog.setNegativeButton("취소", new DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int which) {
        Toast.makeText(context, "취소 클릭", Toast.LENGTH_SHORT).show();
    }
});
dialog.setCancelable(false);
dialog.show();
}
```

Menu(OptionMenu)

- ColorSetting.JAVA

```
    public void changeColor() {
        int red = seekBar1.getProgress();
        int green = seekBar2.getProgress();
        int blue = seekBar3.getProgress();
        MainActivity.current = Color.rgb(red, green, blue);

        textView1.setText("Red : " + red + "(" +
                (red == 0 ? "00" : String.format("%2H", red)) + ")");
        textView2.setText("Green : " + green + "(" +
                (green == 0 ? "00" : String.format("%2H", green)) + ")");
        textView3.setText("Blue : " + blue + "(" +
                (blue == 0 ? "00" : String.format("%2H", blue)) + ")");

        button.setBackgroundColor(MainActivity.current);
    }
}
```

Menu(ImageView ContextMenu)

- Action 방식
  - Action 방식은 콜백 메소드가 필요하기 때문에 ActionMode 클래스를 이용해야 함
  - Resource에서 id로 찾은 View 객채에 LongClick Listener를 추가하고 ActionMode 객체가 null이면 Action Mode를 ON 상태로 변경
  - Action Mode를 작동시키기 위해서는 콜백 메소드가 구현된 객체의 Data를 받아야 하기 때문에 콜백 메소드가 필요
  - ActionMode.Callback 객체에서 onCreateActionMode()는 inflater 객체를 생성하고 onPrepareActionMode()에서는 상태가 Update되면 실행할 동작문을 작성하며, onActionItemClicked()는 Click Event Listener를 구현
  - 마지막으로 onDestoryActionMode()는 Action Mode가 종료될 때 실행할 작동문을 구현

Menu(ImageView ContextMenu)

- imagemenu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/change"
        android:title="이미지 변경"/>
    <item
        android:id="@+id/rotate"
        android:title="45도 회전"/>
    <item
        android:id="@+id/scale"
        android:title="2배 확대"/>
    <item
        android:id="@+id/normal"
        android:title="초기화" />
</menu>
```

Menu(ImageView ContextMenu)

- ImageView 등록
  - ActionMode는 콜백 메소드 객체로 활성화시키기 때문에 onLongClickListener()를 통해 LongClick 시 actionMode가 null이 아니라면 MainActivity에서 Action Mode를 활성화 시킴

```
imageView = findViewById(R.id.image);
imageView.setOnLongClickListener(new View.OnLongClickListener() {
    @Override
    public boolean onLongClick(View v) {
        if (actionMode != null) {
            return false;
        }
        actionMode = startActionMode(callback);
        v.setSelected(true);
        return true;
    }
});
```

Menu(ImageView ContextMenu)

- ActionMode Callback 객체 추가

```
private ActionMode.Callback callback = new ActionMode.Callback() {
    @Override
    public boolean onCreateActionMode(ActionMode mode, Menu menu) {
        MenuInflater menuInflater = mode.getMenuInflater();
        menuInflater.inflate(R.menu.imagemenu, menu);

        return true;
    }

    @Override
    public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
        return false;
    }
```

Menu(ImageView ContextMenu)

- ActionMode Callback 객체 추가

```
@Override
public boolean onActionItemClicked(ActionMode mode, MenuItem item) {
    Bitmap bitmapFactory;
    switch (item.getItemId()) {
        case R.id.change:
            if (flag) {
                bitmapFactory = BitmapFactory.decodeResource(getResources(),
                        R.drawable.an1);
                flag = false;
            } else {
                bitmapFactory = BitmapFactory.decodeResource(getResources(),
                        R.drawable.picture);
                flag = true;
            }
            imageView.setImageBitmap(Bitmap.createBitmap(bitmapFactory));
            break;
```

Menu(ImageView ContextMenu)

- ActionMode Callback 객체 추가

```
            case R.id.rotate:
                imageView.setRotation(imageView.getRotation() + 45.0f);
                break;
            case R.id.scale:
                imageView.setScaleX(2.0f);
                imageView.setScaleY(2.0f);
                break;
            case R.id.normal:
                imageView.setScaleX(1.0f);
                imageView.setScaleY(1.0f);
                imageView.setRotation(0.0f);
        }
        return true;
    }
    @Override
    public void onDestroyActionMode(ActionMode mode) {
        actionMode = null;
    }
};
```

Menu(ImageView ContextMenu)

- onCreateActionMode()는 상태 활성화 시 처리 메소드인데 inflater를 만들어 inflate()에 활성화 할 Context Menu Resource Id와 menu를 넘겨주면 ActionBar에 Context Menu가 활성화 됨
- 그리고 onActionItemClicked()로 Context Menu가 Click되면 Click된 item을 찾아 알맞게 작동