모바일프로그래밍기초

- 안드로이드(Android) -

- 안드로이드 기능/실습 - 11
  - 옵션 메뉴 구성 / 속성
  - XML을 이용한 구성 / 코드를 이용한 구성

- 안드로이드 기능/실습 - 12
  - 컨텍스트 메뉴 구성
  - XML을 이용한 구성
  - ConstraintLayout - chainStyle

목적

- 메뉴를 생성하는 과정 및 활용 방법 습득

메뉴 종류

- Option 메뉴
- Context 메뉴

메뉴를 구성하는 방법

- XML로 정의하는 방법 (기본)
- 코드에서 직접 정의하는 방법

❑ 안드로이드 기능/실습 – 11
옵션 메뉴 구성 / 속성

○목적

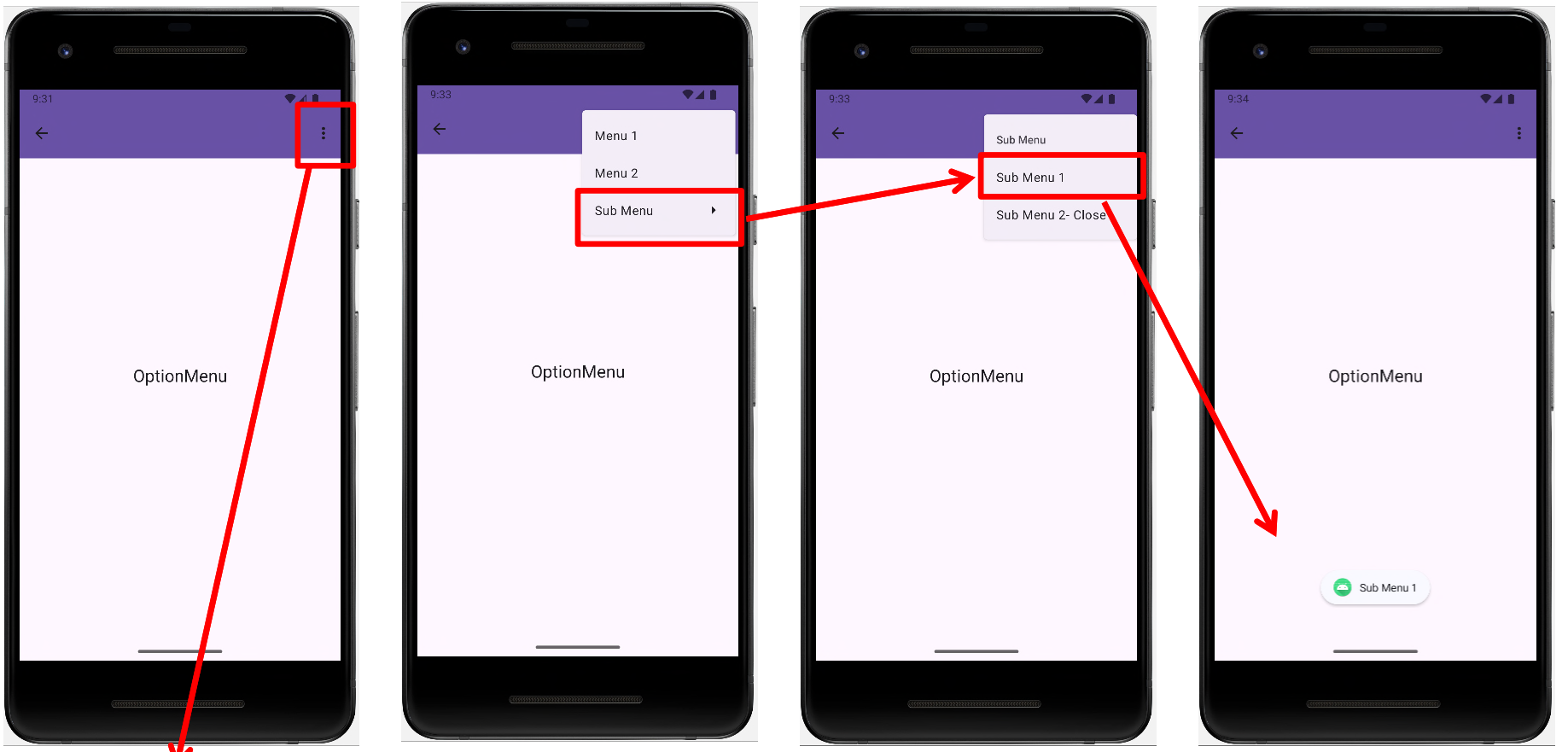
- ❑ 메뉴를 생성하는 방법 학습
- ❑ 메뉴를 생성하는 방법 중에서 XML을 이용하는 방법
  - ❖ Option Menu 생성 과정
  - ❖ 일반적인 형태의 메뉴 생성 방법
- ❑ 메뉴의 재사용성 등 …

○옵션 메뉴를 XML을 이용하여 생성

○컨텍스트 메뉴도 같은 방법으로 생성

❑ 실행 모습

❍ 메뉴를 xml을 이용하여 리소스 부분에 작성하여 사용

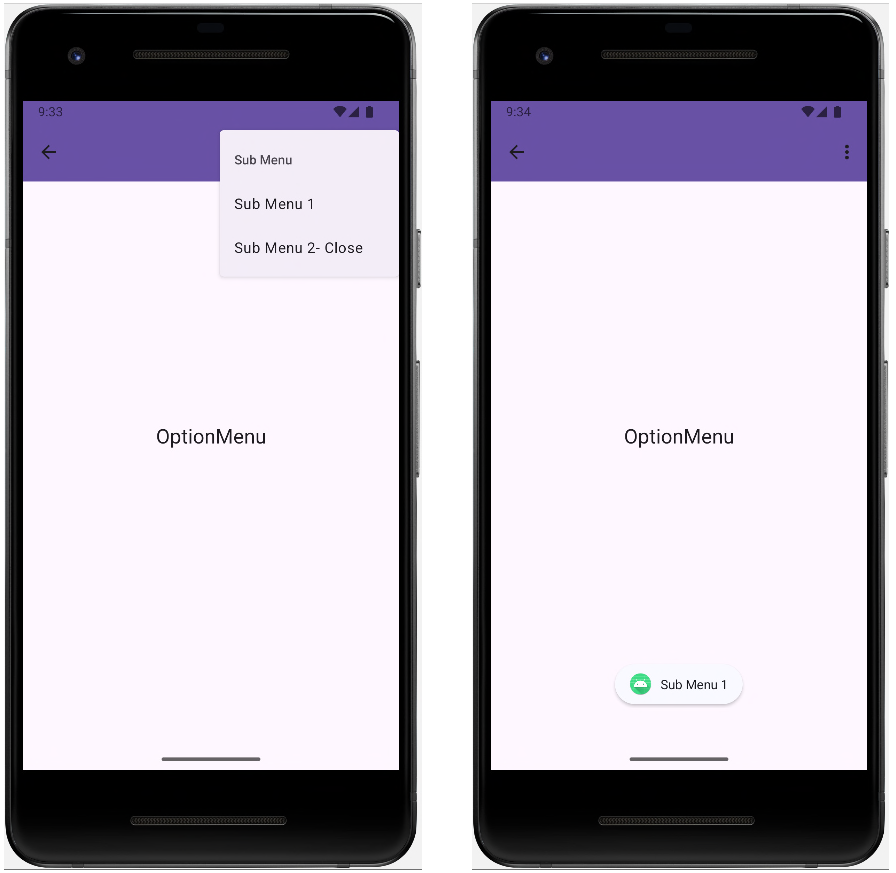


옵션메뉴를 선택

옵션메뉴를 선택하면
실행 또는
하위 메뉴 등장

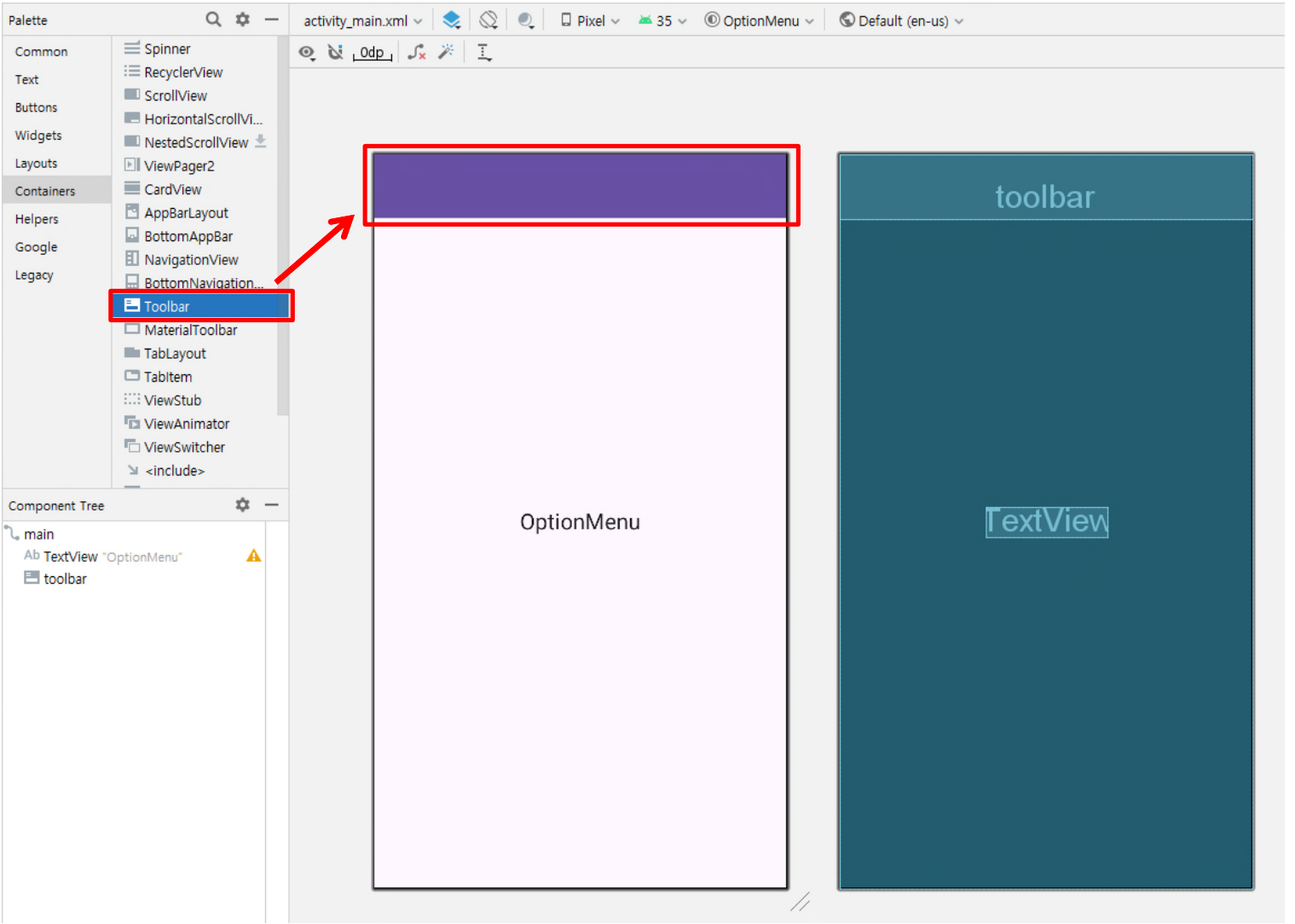
❑ 레이아웃 구성

- 종료 메뉴도 옵션 메뉴에 포함시켜서 옵션 메뉴만을 구성하는 과정을 보여주기 때문에 별도의 레이아웃 구성이 필요 없음
- 대신 옵션 메뉴에 대한 구성이 필요
  - 여기서는 Toolbar를 이용하여 구성
    - ❖ 테마를 변경하여 기본 ActionBar를 이용하는 것도 가능
  - 따라서 별도의 Toolbar 레이아웃 구성 및 테마 변경이 필요
- 옵션 메뉴에 대한 구성을 위한 XML 작업이 필요
  - 옵션 메뉴의 선택 여부는 토스트 기능으로 확인
  - 구성한 내용을 XML로 생성해야 함



❑ 레이아웃 작성

- 옵션 메뉴만을 구성하는 과정을 보여주기 때문에 옵션메뉴를 배치할 Toolbar 구성만 필요



❑ 레이아웃 작성

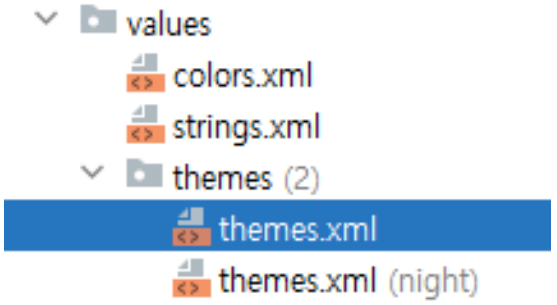
```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="OptionMenu"
        android:textAppearance="@style/TextAppearance.AppCompat.Large"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

❑ 레이아웃 작성

```
    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:background="?attr/colorPrimary"
        android:minHeight="?attr/actionBarSize"
        android:theme="?attr/actionBarTheme"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

❑ 테마 변경이 필요한 경우

❍ 테마 변경이 필요한 경우에는 Themes.xml 파일 수정



themes.xml

```xml
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Base.Theme.OptionMenu" parent="Theme.Material3.DayNight.NoActionBar">
        <!-- Customize your light theme here. -->
        <!-- <item name="colorPrimary">@color/my_light_primary</item> -->
    </style>

    <style name="Theme.OptionMenu" parent="Base.Theme.OptionMenu" />
</resources>
```

❑ 테마 변경이 필요한 경우

❍ 테마 변경이 필요한 경우에는 Themes.xml 파일에 새로 적용할 테마를 추가하여 적용

- ❑ 여기서는 동일한 테마(NoActionBar)를 이름만 변경하여 적용함으로써 적용하는 방법 확인

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Base.Theme.OptionMenu" parent="Theme.Material3.DayNight.NoActionBar">
        <!-- Customize your light theme here. -->
        <!-- <item name="colorPrimary">@color/my_light_primary</item> -->
    </style>

    <style name="Custom.Theme.NoActionBar" parent="Theme.Material3.DayNight.NoActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_500</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>

    <style name="Custom.Theme.OptionMenu" parent="Custom.Theme.NoActionBar" />
<!--    <style name="Theme.OptionMenu" parent="Base.Theme.OptionMenu" />-->
</resources>
```

colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="black">#6851A5</color>
    <color name="white">#FFFFFFFF</color>
    <!--    색상 추가    -->
    <color name="tb_color">#FF6851A5</color>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <!--    여기까지    -->
</resources>
```

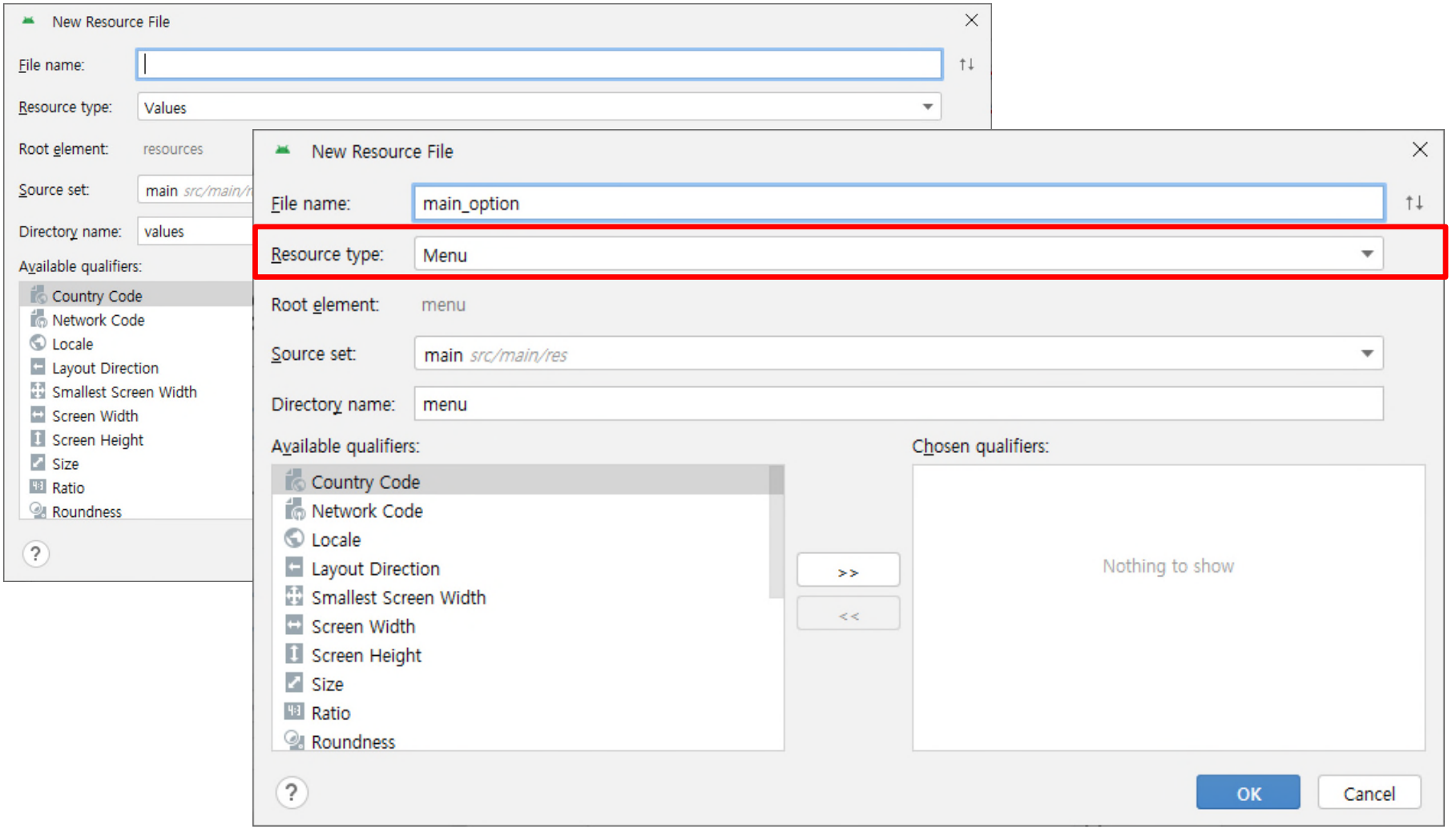
❑ 테마 변경이 필요한 경우

❍ AndroidManifest.xml에서 적용 테마를 변경

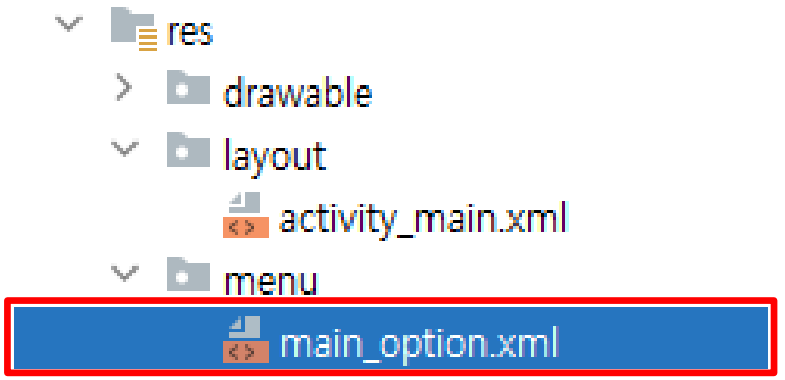
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="OptionMenu"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Custom.Theme.OptionMenu"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

❍메뉴 구성

❍res/menu 하단에 구성 (Resource type을 Menu로 설정)



- 메뉴 구성
- res/menu 하단에 구성
  - 메뉴 파일은 필요에 따라서 추가



main_option.xml

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

</menu>
```

❑ 메뉴 구성 - main_option.xml

```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="Menu 1"/>
    <item android:id="@+id/item2" android:title="Menu 2"/>
    <item android:id="@+id/item3" android:title="Sub Menu">
        <menu>
            <item android:id="@+id/item4" android:title="Sub Menu 1"/>
            <item android:id="@+id/item5" android:title="Sub Menu 2- Close"/>
        </menu>
    </item>
</menu>
```

새롭게 추가된 XML 메뉴 정보

❑ 코드 작성 / 설명

```
package com.practice.ex.optionmenu;

import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;

import androidx.activity.EdgeToEdge;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.content.ContextCompat;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

public class MainActivity extends AppCompatActivity {

    Toolbar toolbar; 3 usages

    int tbColor; 3 usages

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable( $this$enableEdgeToEdge: this);
        setContentView(R.layout.activity_main);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {...});
```

❑ 코드 작성 / 설명

```
        toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        getSupportActionBar().setDisplayShowTitleEnabled(false);

        tbColor = ContextCompat.getColor( context: this, R.color.tb_color);

        toolbar.setBackgroundColor(tbColor);
        getWindow().setStatusBarColor(tbColor);
    }

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

툴바 사용 설정

홈버튼 사용 설정
타이틀 사용 안함 설정

리소스에 정의된 색상 값을
정수형 값으로 얻음

- XML 메뉴를 생성할 때
  MenuInflater 객체의 inflate() 메소드 사용
- MenuInflater 객체는 Activity.getMenuInflater()
  메소드로 얻어올 수 있음
- 메뉴의 리소스 ID만 전달하면 XML 엘리먼트를
  읽어 항목을 추가할 수 있음

❑ 코드 작성 / 설명

```
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    int id = item.getItemId();
    if(id == android.R.id.home) {
        Toast.makeText( context: this,  text: "Home", Toast.LENGTH_LONG).show();
    }
    if(id == R.id.item1) {
        Toast.makeText( context: this,  text: "Menu 1", Toast.LENGTH_LONG).show();
    }
    if(id == R.id.item2) {
        Toast.makeText( context: this,  text: "Menu 2", Toast.LENGTH_LONG).show();
    }
    if(id == R.id.item3) {
        Toast.makeText( context: this,  text: "Sub Menu", Toast.LENGTH_LONG).show();
    }
    if(id == R.id.item4) {
        Toast.makeText( context: this,  text: "Sub Menu 1", Toast.LENGTH_LONG).show();
    }
    if(id == R.id.item5) {
        Toast.makeText( context: this,  text: "Sub Menu 2 - Close", Toast.LENGTH_LONG).show();
        finish();
    }
    return super.onOptionsItemSelected(item);
}
}
```

선택된 메뉴를 구분 처리

❑ 메뉴 표현 속성

❍옵션 속성

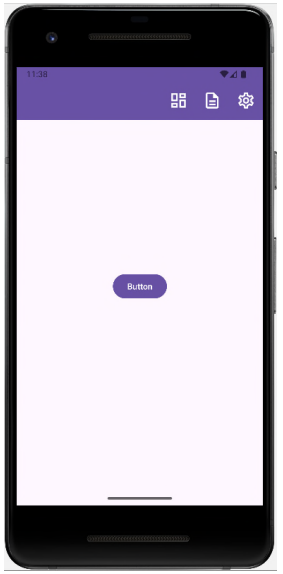
❑ android:showAsAction

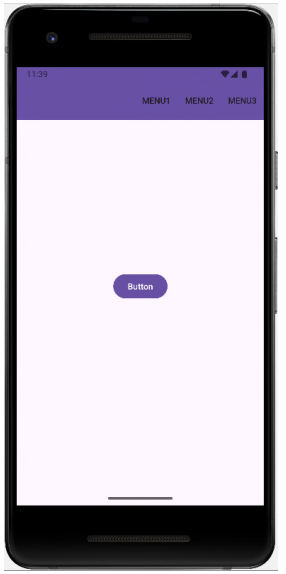
❖ 코드

- setShowAsAction(MenuItem.SHOW_AS_ACTION_ALWAYS) …

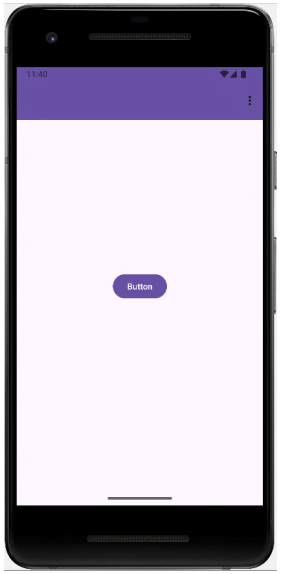
❖ 종류

- never : 항상 액션 아이템으로 표시하지 않음 (기본값)
- ifRoom : 액션 아이템을 표시할 수 있는 공간이 있다면 액션 아이템을 표시함
- withText : 메뉴 항목의 텍스트를 함께 액션 아이템으로 표시
- always : 항상 액션 아이템으로 표시





always



never

❑ 안드로이드 기능/실습 - 12
컨텍스트 메뉴

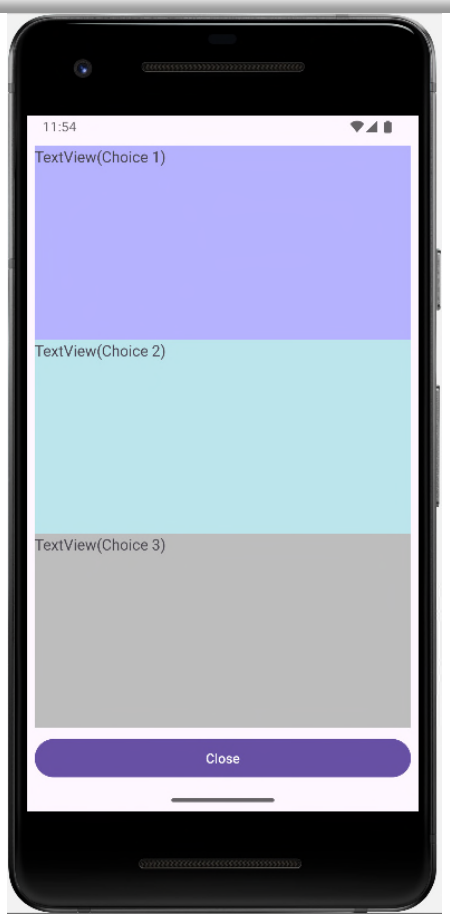
○목적

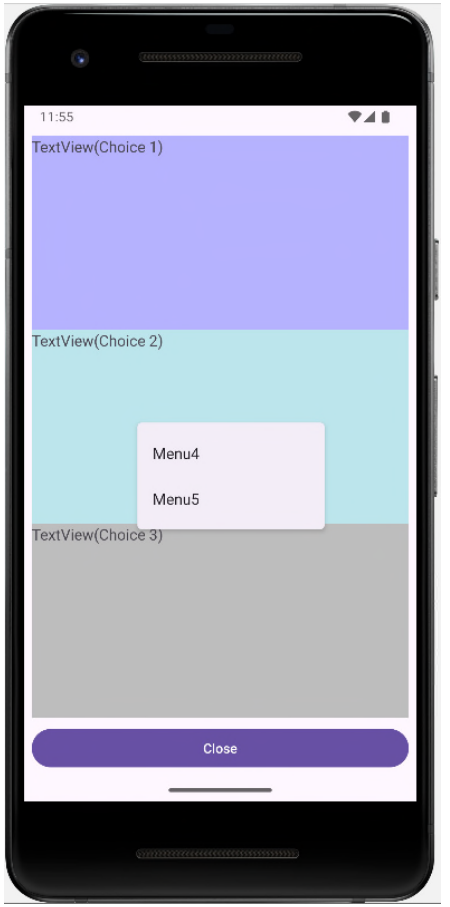
- ❑ 메뉴를 생성하는 방법 중에서 XML을 이용하여 컨텍스트 메뉴 구성
  - ❖ 컨텍스트 메뉴 생성 과정 습득
  - ❖ 메뉴 생성
    - ▪ 메뉴를 생성하는 방법 중에서 XML을 이용하는 방법을 사용(기본)

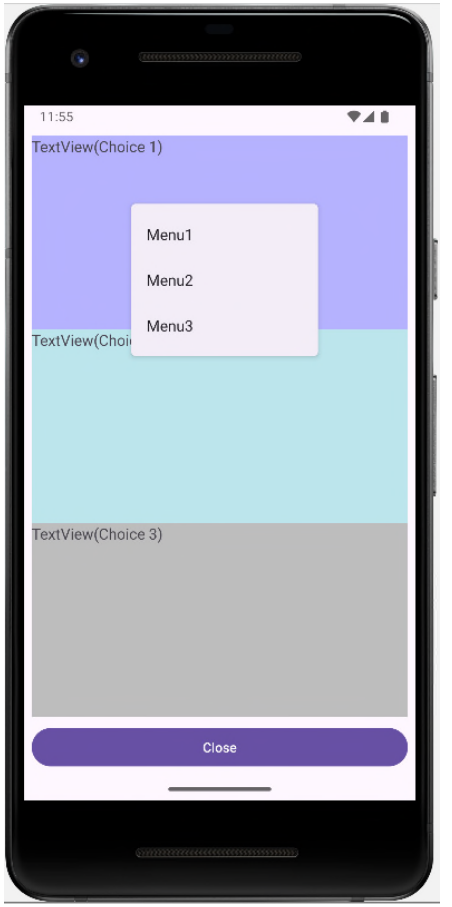
○컨텍스트 메뉴

- ❑ 특정 뷰 또는 항목에 필요한 명령들만 모아 놓은 메뉴
- ❑ 옵션 메뉴 생성 과정과 상당부분이 일치
  - ❖ 메뉴 구성하는 순서를 기억하세요.

❑ 실행 모습



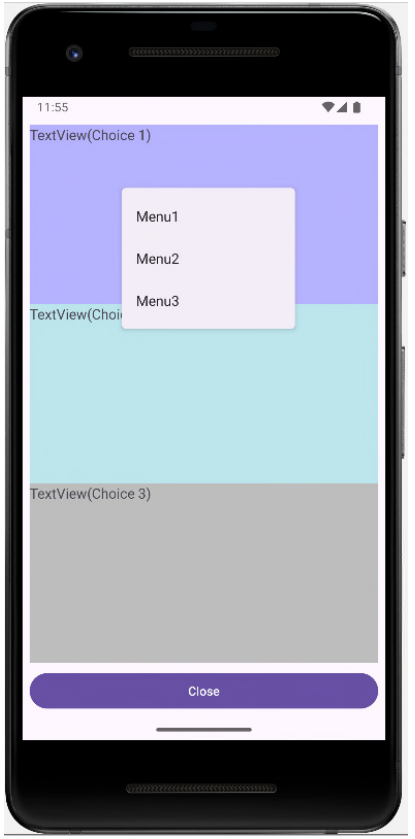


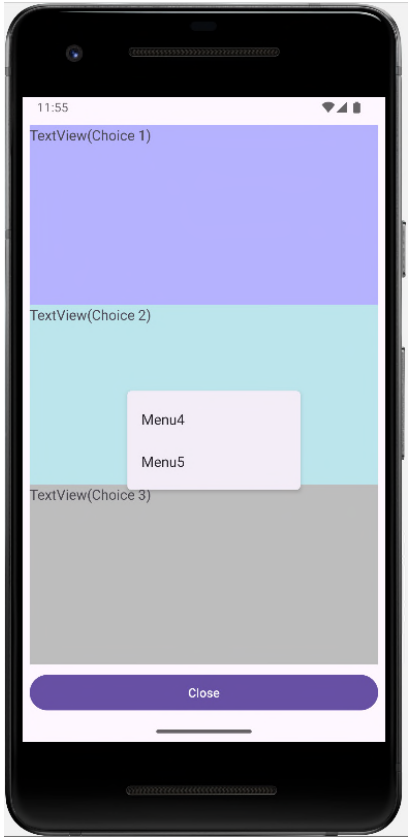


TextView(Choice 1)을 선택하는 경우와 TextView(Choice 2)를 선택하는 경우에 따라 정의한 컨텍스트 메뉴를 각각 호출

❑ 레이아웃 구성

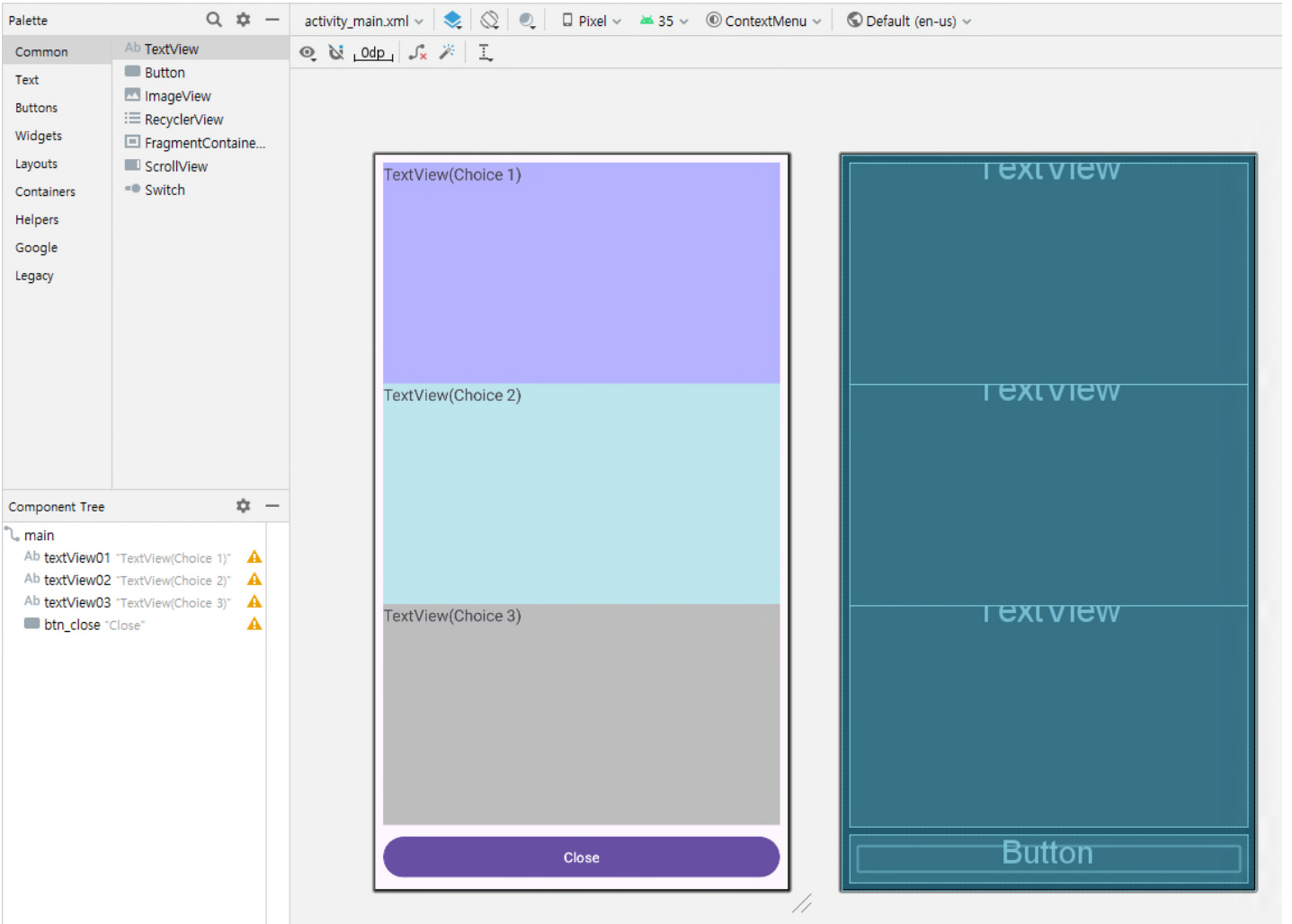
- 메뉴 구성
  - 컨텍스트 메뉴에 대한 XML 파일 작성이 필요
- XML 파일
  - 구성한 내용을 XML로 생성해야 함
- 옵션 메뉴와 달리 컨텍스트 메뉴는 원하는 위치에서 각각 호출





❍컨텍스트 메뉴를 호출할 공간을 텍스트뷰로 구분

- ❑ chainStyle을 이용하여 버튼을 제외한 텍스트뷰 3개만 비율로 공간을 할당하도록 지정



❑ 레이아웃 작성

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView01"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:background="#B5B2FE"
        android:text="TextView(Choice 1)"
        android:textSize="16sp"
        app:layout_constraintBottom_toTopOf="@+id/textView02"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_chainStyle="packed"
        app:layout_constraintVertical_weight="1" />
```

- chainStyle 정보는 A에 표기
  - packed, spread, spread_inside
- weight가 필요한 경우
- 각 요소에 기록

여기서는 TextView 3개가
1:1:1 비율로 동일하기 때문에
비율 표기 없이 가능하지만
코드 설명을 위해 기술

❑ 레이아웃 작성

```
<TextView
    android:id="@+id/textView02"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginStart="8dp"
    android:layout_marginEnd="8dp"
    android:background="#BCE5EC"
    android:text="TextView(Choice 2)"
    android:textSize="16sp"
    app:layout_constraintBottom_toTopOf="@+id/textView03"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView01"
    app:layout_constraintVertical_weight="1" />

<TextView
    android:id="@+id/textView03"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginStart="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="8dp"
    android:background="#BDBDBD"
    android:text="TextView(Choice 3)"
    android:textSize="16sp"
    app:layout_constraintBottom_toTopOf="@+id/btn_close"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView02"
    app:layout_constraintVertical_weight="1" />
```

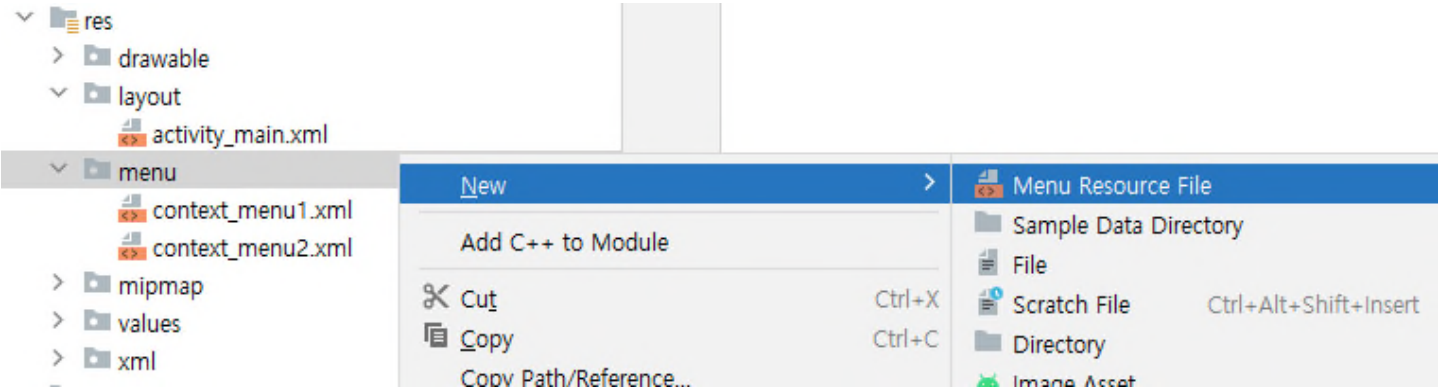
- chainStyle은 서로 양방향으로 연결되는 과정을 진행해야 함
- 연결점 정보 확인

❑ 레이아웃 작성

```
    <Button
        android:id="@+id/btn_close"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:text="Close"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

❍ 메뉴 구성

- ❑ 각각 호출할 메뉴에 대한 파일을 별도 작성
- ❑ 우클릭을 한 후에 Menu resource file 메뉴를 선택
- ❑ 생성할 메뉴관련 XML 파일 이름을 입력 후 생성



❑ 메뉴 - context_menu1.xml

context_menu1.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <menu xmlns:android="http://schemas.android.com/apk/res/android">
3      <item android:id="@+id/item1" android:title="Menu1" />
4      <item android:id="@+id/item2" android:title="Menu2" />
5      <item android:id="@+id/item3" android:title="Menu3" />
6  </menu>
```

작성한 부분

❑ 메뉴 - context_menu2.xml

context_menu2.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/item4" android:title="Menu4" />
    <item android:id="@+id/item5" android:title="Menu5" />
</menu>
```

작성한 부분

❑ 코드 분석 - MainActivity

```
package com.practice.ex.contextmenu;

import android.os.Bundle;
import android.view.ContextMenu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import androidx.activity.EdgeToEdge;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {

    TextView text01, text02, text03;  2 usages
    Button btn01;  2 usages

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable( $this$enableEdgeToEdge: this);
        setContentView(R.layout.activity_main);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {...});
```

❑ 코드 분석 - MainActivity

```
        text01 = (TextView)findViewById(R.id.textView01);
        registerForContextMenu(text01);

        text02 = (TextView)findViewById(R.id.textView02);
        registerForContextMenu(text02);

        text03 = (TextView)findViewById(R.id.textView03);

        btn01 = (Button) findViewById(R.id.btn_close);
        btn01.setOnClickListener(this);
    }

    @Override
    public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo menuInfo) {
        super.onCreateContextMenu(menu, v, menuInfo);
        if (v.getId() == R.id.textView01)
            getMenuInflater().inflate(R.menu.context_menu1, menu);
        if (v.getId() == R.id.textView02)
            getMenuInflater().inflate(R.menu.context_menu2, menu);
    }
```

컨텍스트 메뉴를 호출할 수 있는 뷰를 설정

-컨텍스트 메뉴를 호출할 수 있게 등록 registerForContextMenu(호출할 뷰 지정)

컨텍스트 메뉴가 호출되면 필요한 컨텍스트 메뉴 생성

일반적인 옵션메뉴와 달리 설정된 뷰에서 각각 필요한 컨텍스트 메뉴를 호출함으로 그에 따른 구분이 필요

❑ 코드 분석 - MainActivity

```
    @Override
    public boolean onContextItemSelected(@NonNull MenuItem item) {
        int itemId = item.getItemId();
        if (itemId == R.id.item1) {
            Toast.makeText( context: this,  text: "Menu1", Toast.LENGTH_LONG).show();
        } else if (itemId == R.id.item2) {
            Toast.makeText( context: this,  text: "Menu2", Toast.LENGTH_LONG).show();
        } else if (itemId == R.id.item3) {
            Toast.makeText( context: this,  text: "Menu3", Toast.LENGTH_LONG).show();
        } else if (itemId == R.id.item4) {
            Toast.makeText( context: this,  text: "Menu4", Toast.LENGTH_LONG).show();
        } else if (itemId == R.id.item5) {
            Toast.makeText( context: this,  text: "Menu5", Toast.LENGTH_LONG).show();
        }
        return super.onContextItemSelected(item);
    }

    @Override
    public void onClick(View v) {
        finish();
    }
}
```

메뉴 아이템이 선택되었을 때 처리하는 부분은 컨텍스트 메뉴를 호출하는 구간에 관계없이 한곳에서 모두 구분하여 필요한 내용을 처리

onContextItemSelected(…)
– 컨텍스트 메뉴에서 아이템이 선택되었을 때 호출