

Tinker board

IoT를 위한 최선의 엣지 단말기





Day 3-Contents

- 키오스크 모드 시작 및 중지
- 부팅 시 APP 실행
- 키오스크 – 관리자 권한OS 설치
- Android UI 생성
- 과일 판매 키오스크 예제

*eMMC by SKU

키오스크 모드 시작 및 중지



키오스크 모드 시작 및 중지

<XML>

The screenshot shows the Android Studio interface with the XML tab selected. The code editor displays the XML configuration for an activity. The XML file defines a linear layout containing two buttons. The first button has an ID of @+id/start, a width of wrap_content, a height of wrap_content, and an onClick event named onclicked1. Its text is "Start Lock". The second button has an ID of @+id/end, a width of wrap_content, a height of wrap_content, and an onClick event named onclicked2. Its text is "End Lock". The XML code is as follows:

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     android:orientation="vertical"
8     tools:context=".MainActivity">
9
10    <Button
11        android:id="@+id/start"
12        android:layout_width="wrap_content"
13        android:layout_height="wrap_content"
14        android:onClick="onclicked1"
15        android:text="Start Lock" />
16
17    <Button
18        android:id="@+id/end"
19        android:layout_width="wrap_content"
20        android:layout_height="wrap_content"
21        android:onClick="onclicked2"
22        android:text="End Lock" />
23</LinearLayout>
```



키오스크 모드 시작 및 중지

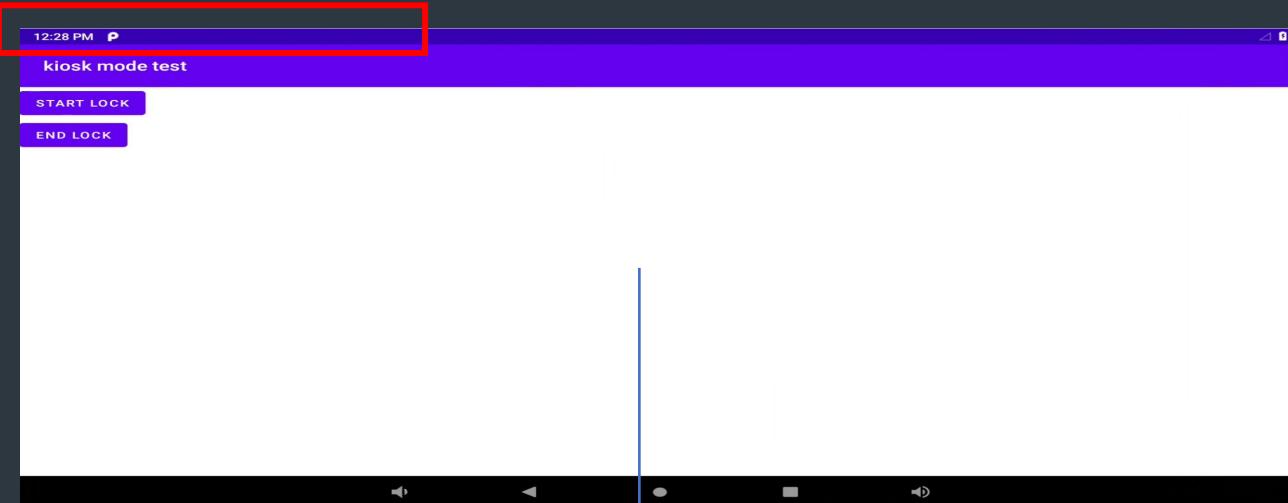
<java>

The screenshot shows the Android Studio interface with two tabs open: 'activity_main.xml' and 'MainActivity.java'. The code in 'MainActivity.java' is as follows:

```
1 package com.example.kioskmodeset;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity {
6
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        setContentView(R.layout.activity_main);
11    }
12
13    public void onclicked1(View v){
14        Toast.makeText( context: this, text: "키오스크 모드 ON.", Toast.LENGTH_SHORT).show();
15        startLockTask();
16    }
17
18    public void onclicked2(View v){
19        Toast.makeText( context: this, text: "키오스크 모드 OFF.", Toast.LENGTH_SHORT).show();
20        stopLockTask();
21    }
22
23 }
```



키오스크 모드 시작 및 중지 - Lock 모드 확인



1. Lock이 걸려 다른 화면으로 움직이지 않고 상태표시가 보이지 않는걸 확인 할 수 있습니다.

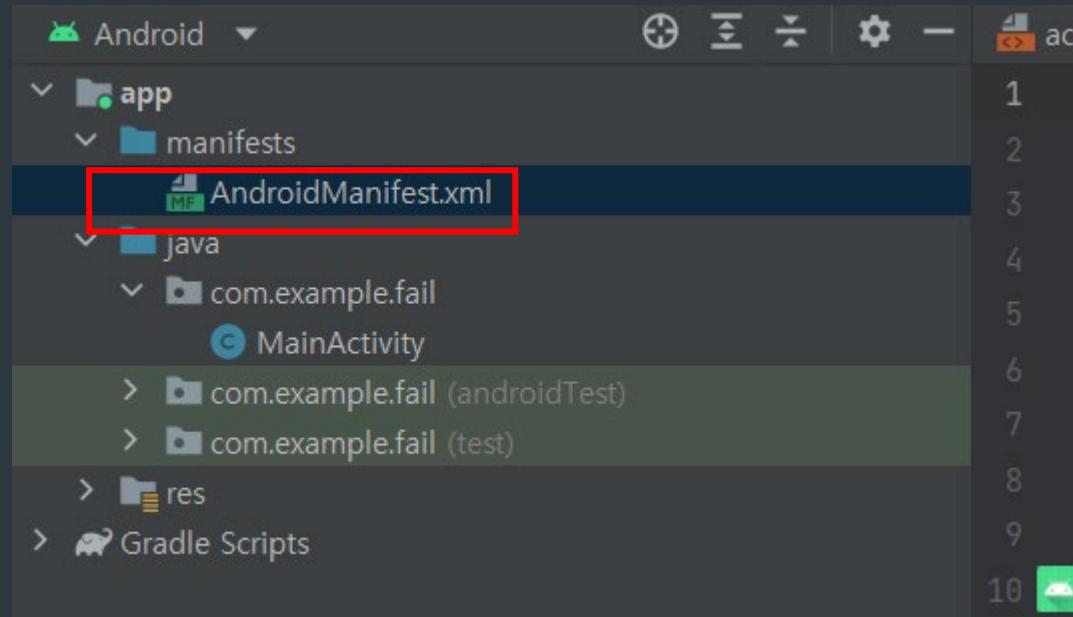


부팅시 APP 실행



부팅시 APP 실행 – Mainfest 추가

<Mainfest>



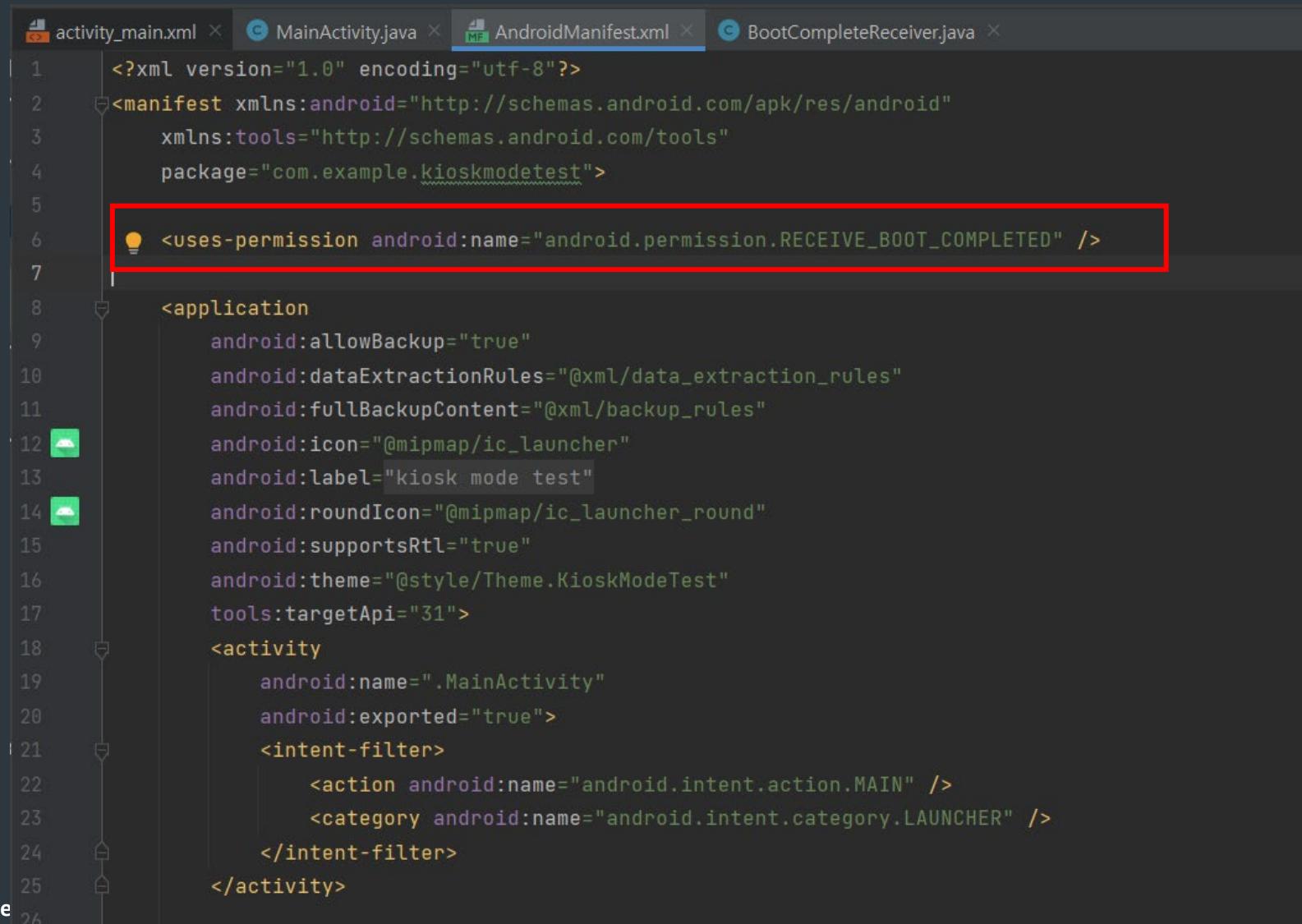
부팅 완료(BOOT_COMPLETED)
브로드 캐스트 추가

1. 키오스크는 부팅과 동시에 앱이 시작되어야 하기 때문에, BOOT_COMPLETED 브로드 캐스트를 추가 해야 합니다.



부팅시 APP 실행 – Mainfest 추가

<Mainfest-1>



```
activity_main.xml MainActivity.java AndroidManifest.xml BootCompleteReceiver.java
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools"
4      package="com.example.kioskmodetest">
5
6      <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
7
8      <application
9          android:allowBackup="true"
10         android:dataExtractionRules="@xml/data_extraction_rules"
11         android:fullBackupContent="@xml/backup_rules"
12         android:icon="@mipmap/ic_launcher"
13         android:label="kiosk mode test"
14         android:roundIcon="@mipmap/ic_launcher_round"
15         android:supportsRtl="true"
16         android:theme="@style/Theme.KioskModeTest"
17         tools:targetApi="31">
18         <activity
19             android:name=".MainActivity"
20             android:exported="true">
21             <intent-filter>
22                 <action android:name="android.intent.action.MAIN" />
23                 <category android:name="android.intent.category.LAUNCHER" />
24             </intent-filter>
25         </activity>
26     </application>

```

1. APP에 부팅시 자동으로 실행이 가능케 하는 유저 권한을 입력



부팅시 APP 실행 – Mainfest 추가

<Mainfest-2>

```
26  
27     <receiver  
28         android:name=".BootCompleteReceiver"  
29         android:enabled="true"  
30         android:exported="false"  
31         android:directBootAware="true"  
32         tools:targetapi="n">  
33             <intent-filter>  
34                 <action android:name="android.intent.action.BOOT_COMPLETED">  
35                 </action>  
36             </intent-filter>  
37         </receiver>  
38     </application>  
39 </manifest>
```

클릭시 자동
class 생성 가능

부팅시
키오스크 모드로 동작

부팅시 APP 실행 - BootCompleteReceiver Class 작성

<BootCompleteReceiver>

```
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Build;
import android.util.Log;

public class BootCompleteReceiver extends BroadcastReceiver {
    static final String TAG = "SIMPLE_KIOSK";

    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        Log.i(TAG, msg: "onReceive: " + action);
        if(Intent.ACTION_BOOT_COMPLETED.equals(action)) {
            startActivity(context);
        }
    }

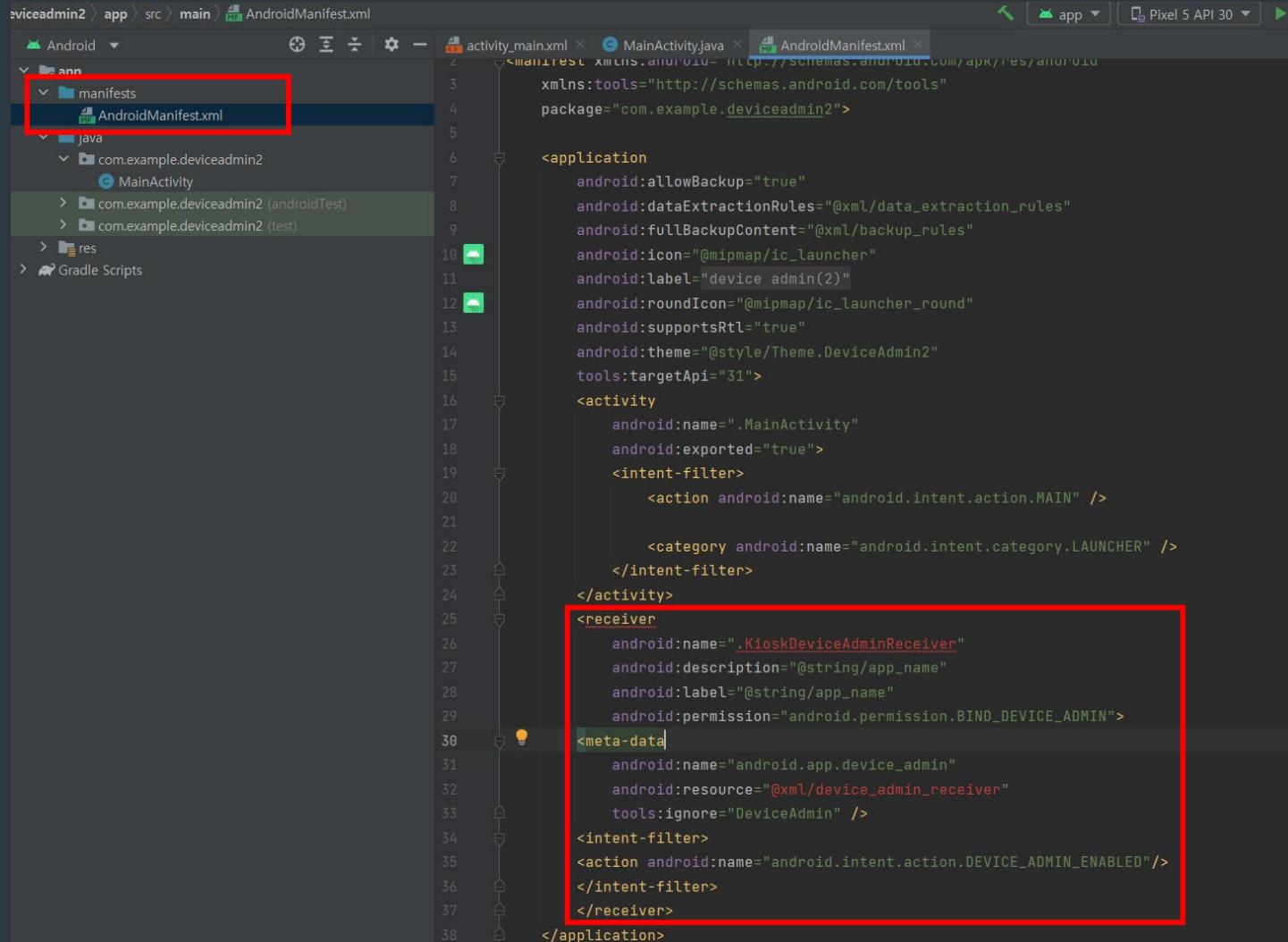
    static void startActivity(Context context) {
        Intent i = new Intent(context, MainActivity.class);
        // For Android 9 and below
        if(Build.VERSION.SDK_INT < Build.VERSION_CODES.Q) {
            i.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
            context.startActivity(i);
            return;
        }
    }
}
```

부팅시 앱을 자동으로
실행 시키는 코드

키오스크 – 관리자 권한

관리자 권한 - Mainfest 추가

<Mainfest>



```
deviceadmin2 app src main AndroidManifest.xml
Android app manifests MainActivity.java AndroidManifest.xml
activity_main.xml MainActivity.java AndroidManifest.xml
AndroidManifest.xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.deviceadmin2">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="device admin(2)"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.DeviceAdmin2"
        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>
        <receiver
            android:name=".KioskDeviceAdminReceiver"
            android:description="@string/app_name"
            android:label="@string/app_name"
            android:permission="android.permission.BIND_DEVICE_ADMIN">
            <meta-data
                android:name="android.app.device_admin"
                android:resource="@xml/device_admin_receiver"
                tools:ignore="DeviceAdmin" />
            <intent-filter>
                <action android:name="android.intent.action.DEVICE_ADMIN_ENABLED" />
            </intent-filter>
        </receiver>
    </application>

```

1. 안드로이드 앱에서 기기를 관리 할 수 있는 권한을 앱에 부여할 수 있다.
2. 권한을 얻으면, API를 사용해서 기기의 기능을 제한할 수 있다.



관리자 권한 - Mainfest 추가

<Mainfest>

```
25 <receiver
26     android:name=".KioskDevicesAdminRecrieve"
27     android:description="@string/
28     android:label="@string/app_na
29     android:permission="android.p
30 <meta-data
31     android:name="android.app.dev
32     android:resource="@xml/device_admin_receiver"
33     tools:ignore="DeviceAdmin" />
34 <intent-filter>
35     <action android:name="android.intent.action.DEVICE_ADMIN_ENABLED"/>
36 </intent-filter>
37 </receiver>
```

The screenshot shows the AndroidManifest.xml file in an IDE. Line 26 contains the code `android:name=".KioskDevicesAdminRecrieve"`. A tooltip appears over this line with the message: "Class referenced in the manifest, com.example.deviceadmin2.KioskDevicesAdminRecrieve, was not found in the project or the libraries". Below the tooltip, another message says "Unresolved class 'KioskDevicesAdminRecrieve'". A red box highlights the "Create class 'KioskDevicesAdminRecrieve'" button in the tooltip, which is labeled with the keyboard shortcut "Alt+Shift+Enter".

클릭시 자동
class 생성 가능

관리자 권한 – KioskDeviceAdminReceiver Class 추가

<KioskDeviceAdminReceiver>



The screenshot shows the 'KioskDevicesAdminRecribee.java' file in the Android Studio code editor. The code is as follows:

```
1 package com.example.deviceadmin2;  
2  
3 import android.app.admin.DeviceAdminReceiver;  
4 import android.content.BroadcastReceiver;  
5  
6 public class KioskDevicesAdminRecribee extends DeviceAdminReceiver {  
7 }
```

A blue arrow points from the 'DeviceAdminReceiver' import statement in the code area down to the same import statement in the code completion dropdown shown in the previous screenshot.



관리자 권한 – KioskDeviceAdminReceiver Class 추가

<KioskDeviceAdminReceiver>

```
package com.example.deviceadmin2;

import android.app.admin.DeviceAdminReceiver;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;
import android.widget.Toast;

public class KioskDevicesAdminRecreivee extends DeviceAdminReceiver {
    static final String TAG = "SIMPLE_KIOSK";

    @Override
    public void onReceive(Context context, Intent intent){
        Log.i(TAG, msg: "onReceive: " + intent.getAction());
    }

    @Override
    public void onEnabled(Context context, Intent intent){
        Toast.makeText(context, text: "Device admin enabled", Toast.LENGTH_SHORT).show();
    }

    @Override
    public void onDisabled(Context context, Intent intent) {
        Toast.makeText(context, text: "device admin Disabled", Toast.LENGTH_SHORT).show();
    }

    @Override
    public void onLockTaskModeEntering(Context context, Intent intent, String pkg){
        Log.i(TAG, msg: "onLockTaskModeEntering: " + pkg);
    }
}
```



관리자 권한 - XML 작성

<Manifest>

```
<receiver
    android:name=".KioskDeviceAdminReceiver"
    android:description="kiosk mode test"
    android:label="kiosk mode test"
    android:permission="android.permission.BIND_DEVICE_ADMIN"
    android:exported="true">
    <meta-data
        android:name="android.app.device_admin"
        android:resource="@xml/device_admin_reeiver"
        tools:ignore="DeviceAdmin" />
    <intent-filter>
        <action android:name="android.int
    </intent-filter>
</receiver>
```

클릭시 자동
xml 생성 가능

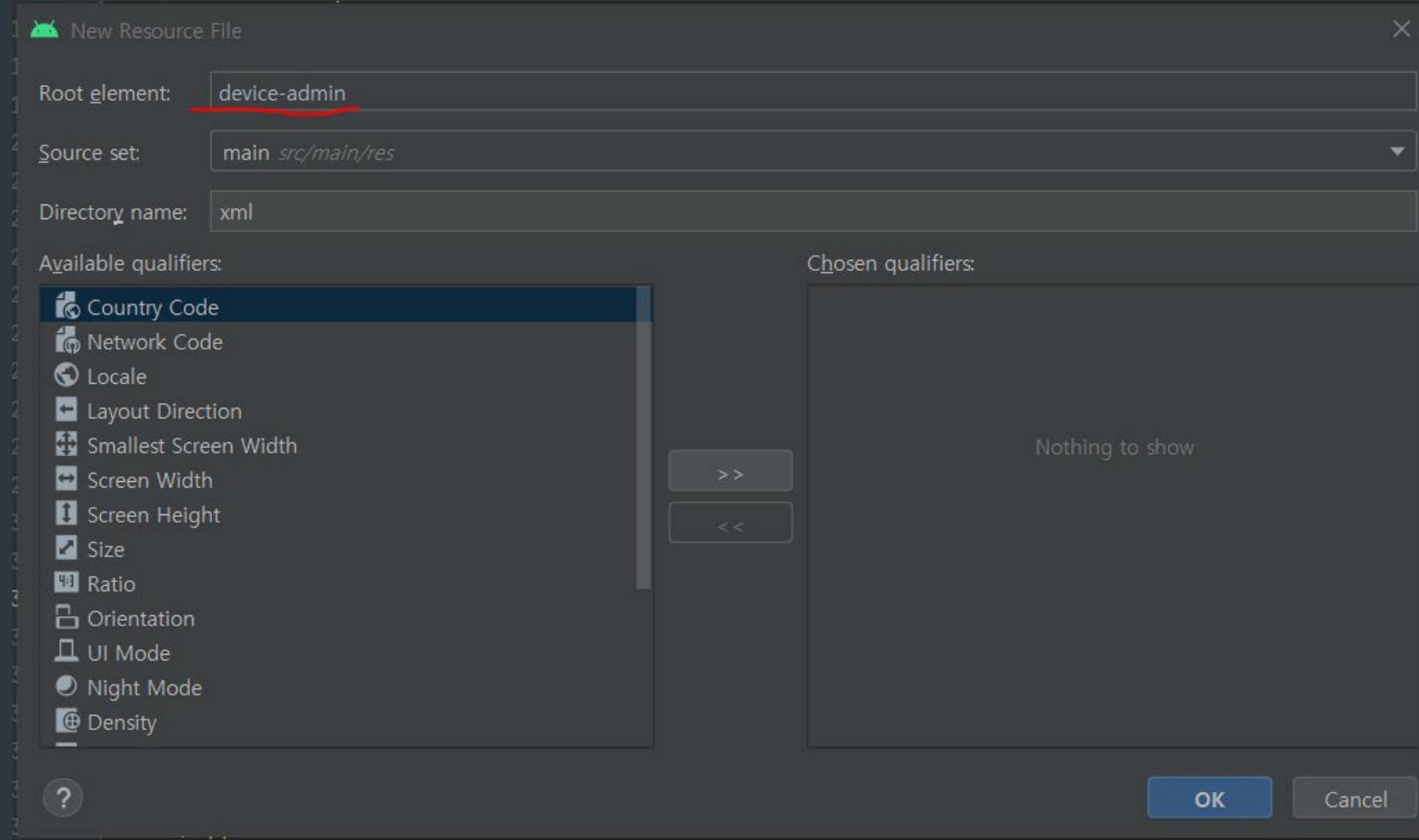
Cannot resolve symbol '@xml/device_admin_reeiver'

Create xml resource file 'device_admin_reeiver.xml' Alt+Shift+Enter



관리자 권한 - XML 작성

<device_admin_receiver.xml>



Device-admin으로
XML 형식 지정



관리자 권한 - XML 작성

<device_admin_receiver.xml>

```
<?xml version="1.0" encoding="utf-8"?>
<device-admin>
    <uses-policies>
        <limit-password />
        <watch-login />
        <reset-password /></reset-password />
        <force-lock />
        <wipe-data />
        <expire-password />
        <encrypted-storage />
        <disable-camera />
    </uses-policies>
</device-admin>
```

Limit-password: 사용자가 선택할 수 있는 암호를 제한하고, 비번 최소 길이 설정

Watch-login: 제한 회수 이상 오류시 로그인 불가능

Reset-password: 비밀번호 재설정 가능

Force-lock: 제한 오류 회수 이상시 화면 잠금 기능

Wipe-data: 장치 초기화 기능

Expire-password: 로그인 가능 제한 일 설정(예. 유효기간 1년시 자동 탈퇴)

Encrypted-storage: storage 암호화 가능

Disable-camera: 카메라 기능 비활성화



관리자 권한 – Mainactivity 수정

<Mainactivity.java>

```
// for device admin app  
static final int DEVICE_ADMIN_ADD_REQUEST = 1001;  
private ComponentName mAdminComponentName;  
private DevicePolicyManager mDevicePolicyManager;  
  
private boolean isDeviceAdminApp() {  
    return mDevicePolicyManager.isAdminActive(mAdminComponentName);  
}  
  
private boolean isDeviceOwnerApp() {  
    return mDevicePolicyManager.isDeviceOwnerApp(getPackageName());  
}  
  
private void enableAdmin() {  
    if (isDeviceAdminApp()) {  
        return;  
    }  
    Intent intent = new Intent(DevicePolicyManager.ACTION_ADD_DEVICE_ADMIN);  
    intent.putExtra(DevicePolicyManager.EXTRA_DEVICE_ADMIN, mAdminComponentName);  
    // Start the add device admin activity  
    startActivityForResult(intent, DEVICE_ADMIN_ADD_REQUEST);  
}
```

MainActivity에 기기 관리자 권한 요청 코드 추가



소유자 권한 부여

<Manifest>

```
<application
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@mipmap/ic_launcher"
    android:label="kiosk mode test"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:testOnly="true" // This line is highlighted with a red box
    android:theme="@style/Theme.KioskModeTest"
    tools:targetApi="31">
```

1. 소유자 앱으로 테스트 하다가 문제가 생겨서 기기를 사용할 수 없게 되거나 공장 초기화를 해야 하는 경우가 발생할 수 있다. 그래서 아래와 같이 Application 요소에 android:testOnly="true" 를 추가해서 기기의 잠금으로 인한 문제를 방지한다.



소유자 권한 부여 – ADB에서 APP 권한 수정

<cmd>

```
명령 프롬프트
Microsoft Windows [Version 10.0.19044.1826]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Del\>adb shell dpm set-device-owner net.sjava.examples.kiosk/.KioskDeviceAdminReceiver
C:\Users\Del\>adb shell dpm remove-active-admin net.sjava.examples.kiosk/.KioskDeviceAdminReceiver
```

1. 소유자 권한을 부여하고, 관리자 앱을 비활성화 하는 방법



기능 제한

<Mainactivity>

```
activity_main.xml × C MainActivity.java × MF AndroidManifest.xml × device_admin_receiver.xml × C KioskDevicesAdminRecrieve.java ×
package com.example.deviceadmin2;

import ...

public class MainActivity extends AppCompatActivity {

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

    private void enableAdmin() {
        if (isDeviceAdminApp()) {
            return;
        }
        Intent intent = new Intent(DevicePolicyManager.ACTION_ADD_DEVICE_ADMIN)
            Intent.putExtra(DevicePolicyManager.EXTRA_DEVICE_ADMIN, mAdminComponentName);

        startActivityForResult(intent, DEVICE_ADMIN_ADD_REQUEST);
    }

    private void disableCamera(boolean disabled) {
        mDevicePolicyManager.setCameraDisabled(mAdminComponentName, disabled);
    }
}
```

카메라 제어 기능



기능 제한

<Mainactivity>

```
ip > src > main > java > com > example > deviceadmin2 > MainActivity > mUserRestrictions
activity_main.xml > MainActivity.java > AndroidManifest.xml > device_admin_receiver.xml > KioskDevicesAdminReceivee.java > app >
16     setContentView(R.layout.activity_main);
17 }
18 private void enableAdmin() {
19     if (isDeviceAdminApp()) {
20         return;
21     }
22     Intent intent = new Intent(DevicePolicyManager.ACTION_ADD_DEVICE_ADMIN);
23     Intent.putExtra(DevicePolicyManager.EXTRA_DEVICE_ADMIN, mAdminComponentName);
24
25     startActivityForResult(intent, DEVICE_ADMIN_ADD_REQUEST);
26 }
27 private void disableCamera(boolean disabled) {
28     mDevicePolicyManager.setCameraDisabled(mAdminComponentName, disabled);
29 }
30
31 static final ArrayList mUserRestrictions = new ArrayList(
32     Arrays.asList(
33         UserManager.DISALLOW_SMS,
34         UserManager.DISALLOW_MOUNT_PHYSICAL_MEDIA,
35         //UserManager.DISALLOW_USB_FILE_TRANSFER,
36         UserManager.DISALLOW_BLUETOOTH
37     )
38 );
39
40 private void setUserRestrictions(boolean restricted) {
41     for (String restriction : mUserRestrictions) {
42         if (restricted) {
43             mDevicePolicyManager.addUserRestriction(mAdminComponentName, restriction);
44         } else {
45             mDevicePolicyManager.clearUserRestriction(mAdminComponentName, restriction);
46         }
47     }
48 }
49 }
```

사용자 제어 기능



기능 제한

<Mainactivity>

```
static final String[] mSuspendedPackageNames = {"com.twitter.android",
    "com.facebook.katana",
    "com.google.android.apps.nbu.files"
};

static final ArrayList<String> mUserRestrictions = new ArrayList<>(
    Arrays.asList(
        UserManager.DISALLOW_SMS,
        UserManager.DISALLOW_MOUNT_PHYSICAL_MEDIA,
        //UserManager.DISALLOW_USB_FILE_TRANSFER,
        UserManager.DISALLOW_BLUETOOTH
    )
);
```

앱 사용 제한

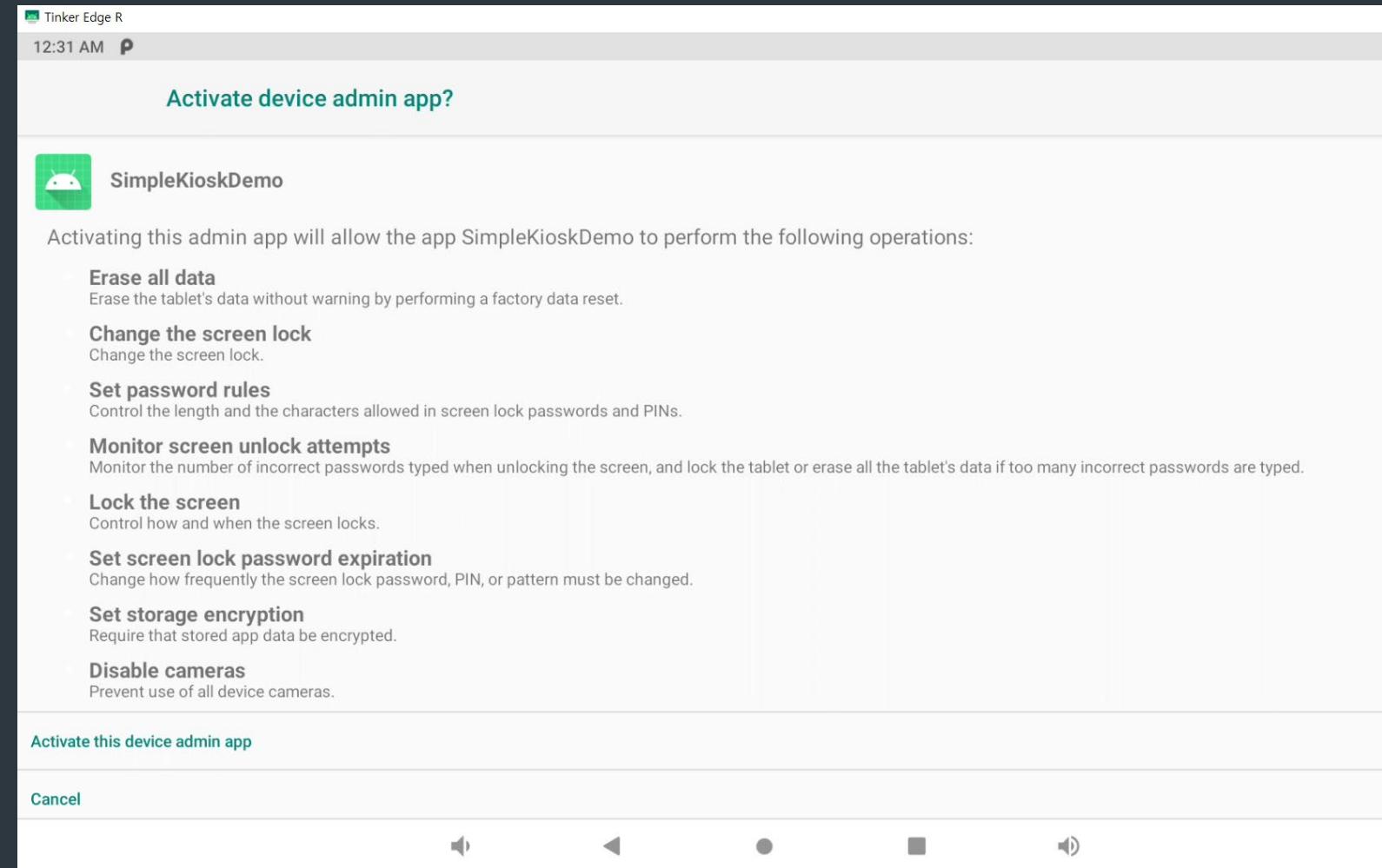


실제 APP 작동 모습

깃 허브 링크

완성본->

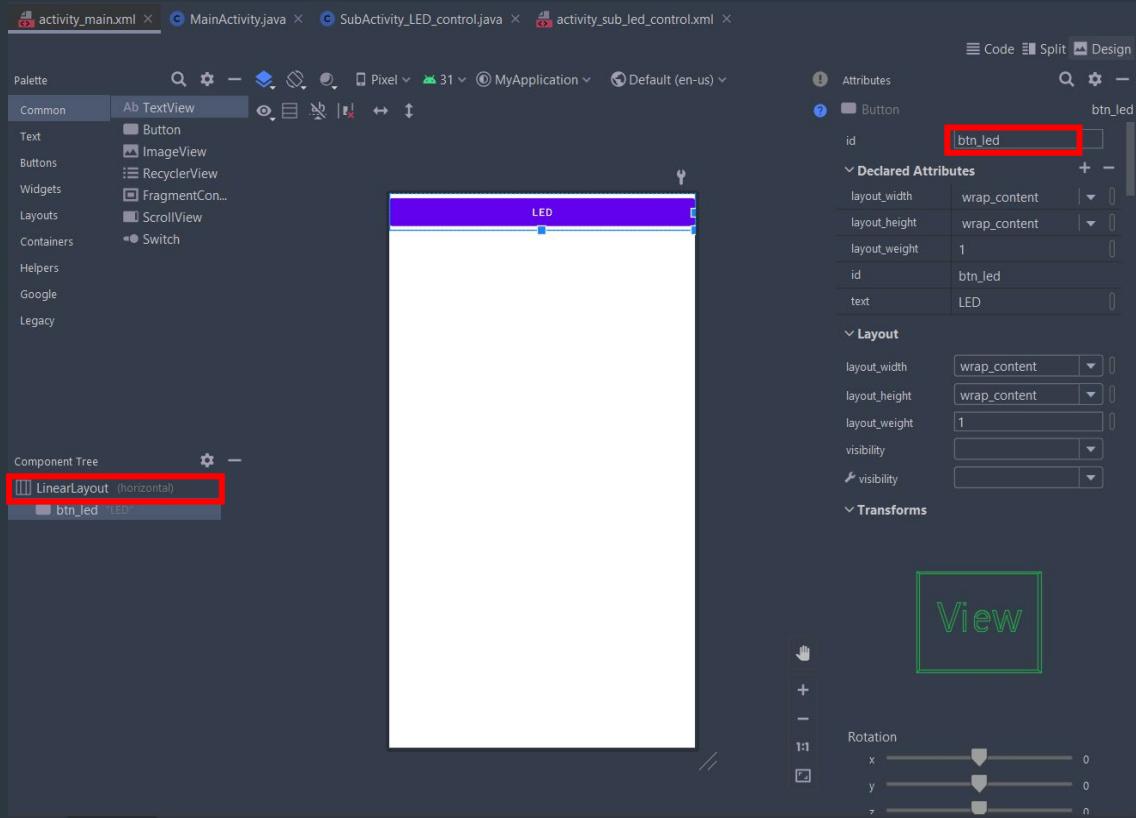
<device_admin_receiver.xml>



Android UI 생성

예제1: LED 버튼 구현 & 함수 연동

1. layout 파일에서 id 값 변경



2. Button 객체 선언후 onClick 메소드 정의

```
public class MainActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        Button btn_led = findViewById(R.id.btn_led);  
        btn_led.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View view) {  
                Intent intent = new Intent(getApplicationContext(), SubActivity_LED_control.class);  
                startActivity(intent);  
            }  
        });  
    }  
}
```

The screenshot shows the MainActivity.java code. It defines a button named "btn_led" and sets its onClickListener to a lambda expression that starts an activity. The entire onClick section is highlighted with a red box.



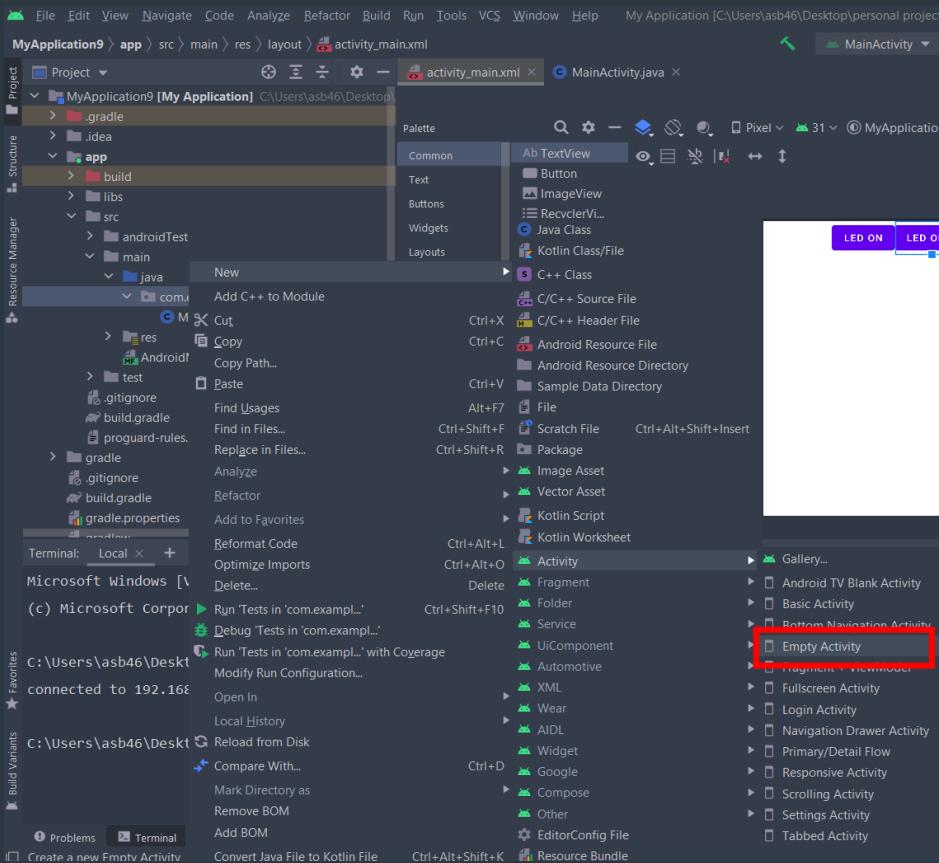
Intent 객체

- 다른 액티비티를 띄우거나 기능을 동작하기 위한 객체
- startActivityForResult 메소드를 호출하여 다른 액티비티로 전환
- 다른 화면으로 이동하거나 웹페이지를 열거나 다이얼 화면으로 전환하는데 사용 가능

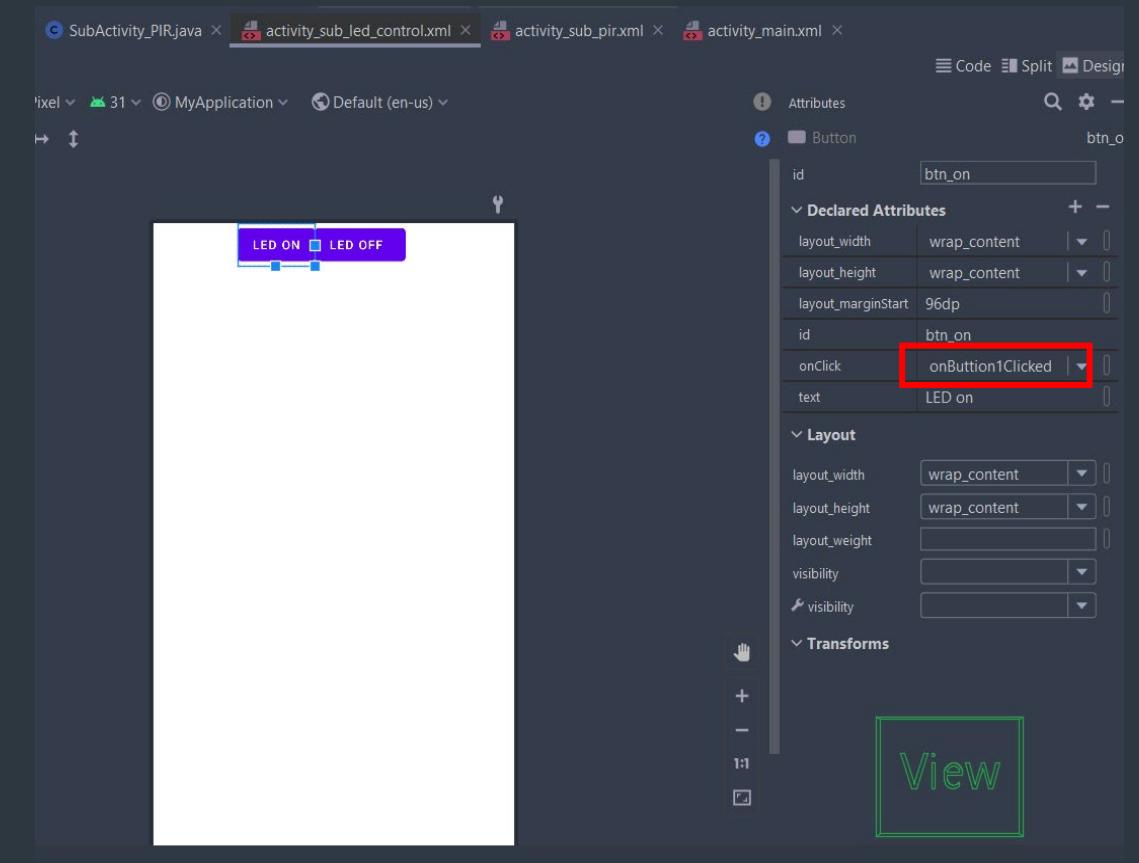
```
Button btn_led = findViewById(R.id.btn_led); layout 파일에서 id가 btn_led라는 이름을 가진 오브젝트를 find
btn_led.setOnClickListener(new View.OnClickListener() { View 클래스의 OnClickListener 인터페이스 (추상 클래스)의 onClick 함수를 재정의
    @Override
    public void onClick(View view) {
        Intent intent = new Intent(packageName: MainActivity.this, SubActivity_LED_control.class); intent 객체 초기화
        startActivityForResult(intent);
    }
});
```

예제1: SubActivity 생성

3. Empty Activity 생성



4. onClick 함수 사용



예제1: SubActivity 작성

The screenshot shows the Android Studio interface with the SubActivity_LED_control.java file open. The code implements a SubActivity that controls an LED via GPIO 3. It imports mraa.* and defines two button click listeners: onButton1Clicked and onButton2Clicked. Each listener initializes a Gpio object for TINKERBOARD_PIN3, sets it to OUT direction, and then writes 1 or 0 to it respectively. A Toast message is displayed to indicate the LED state change.

```
import mraa.*;

public class SubActivity_LED_control extends AppCompatActivity {

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

    public void onButton1Clicked(View v) {
        Gpio gpio3 = new Gpio(TinkerBoard.TINKERBOARD_PIN3.swigValue());
        gpio3.dir(Dir.DIR_OUT);
        gpio3.write( i: 1);
        Toast.makeText( context: this,  text: "LED가 켜졌습니다.", Toast.LENGTH_LONG).show();
    }

    public void onButton2Clicked(View v) {
        Gpio gpio3 = new Gpio(TinkerBoard.TINKERBOARD_PIN3.swigValue());
        gpio3.dir(Dir.DIR_OUT);
        gpio3.write( i: 0);
        Toast.makeText( context: this,  text: "LED가 끄졌습니다.", Toast.LENGTH_LONG).show();
    }
}
```



onClick 함수를 구현하는 2가지 방법

MainActivity에서 setOnTouchListener를 통해 onClick 재정의

```
Button btn_led = findViewById(R.id.btn_led);
btn_led.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent intent = new Intent(packageContext: MainActivity.this, SubActivity_LED_control.class);
        startActivity(intent);
    }
});
```

MainActivity.java의 onCreate 안에서 정의

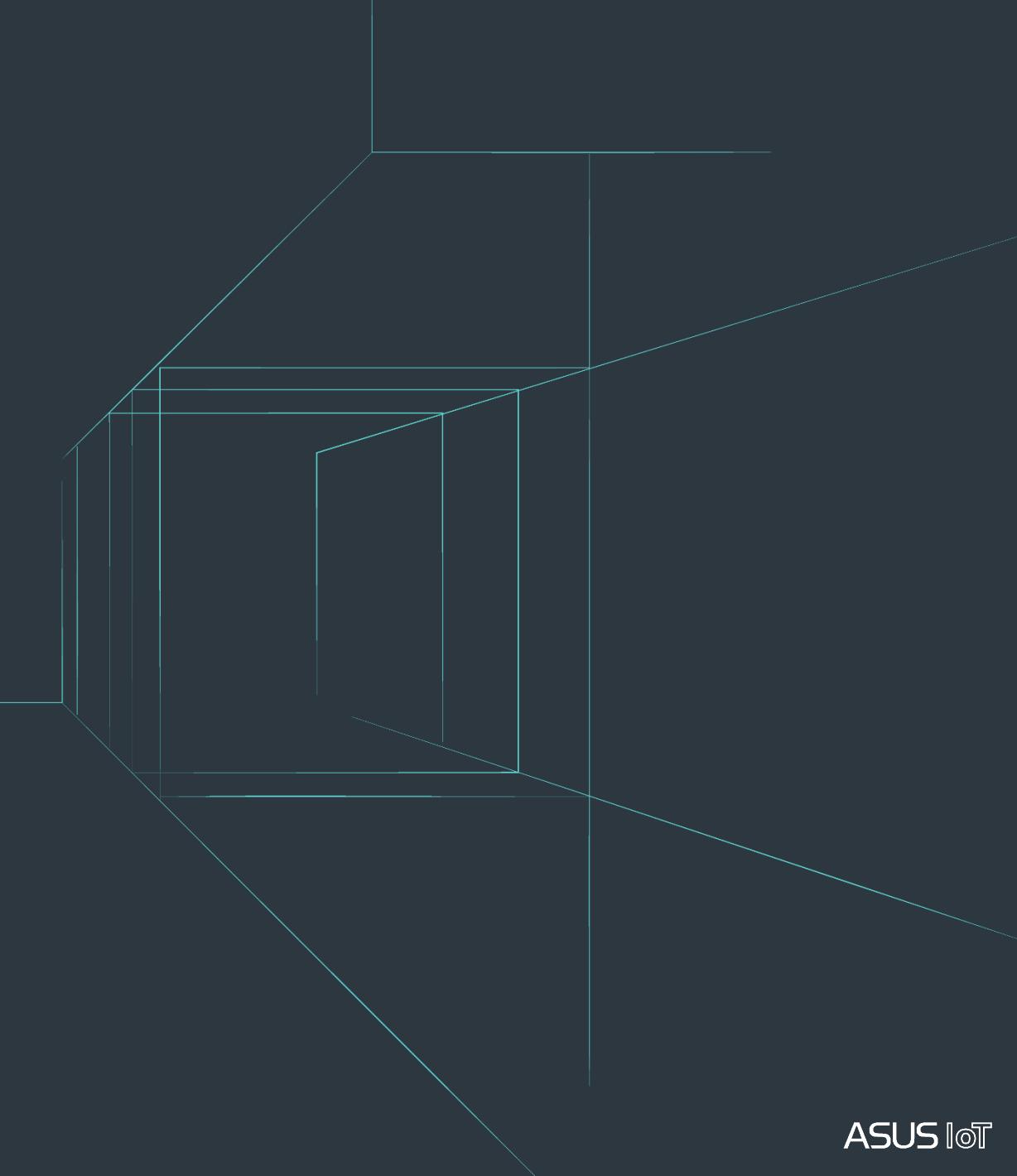
XML에서 Button의 onClick 속성에 함수 선언

```
<Button
    android:id="@+id	btn_on"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="96dp"
    android:onClick="onButton1Clicked"
    android:text="LED on" />

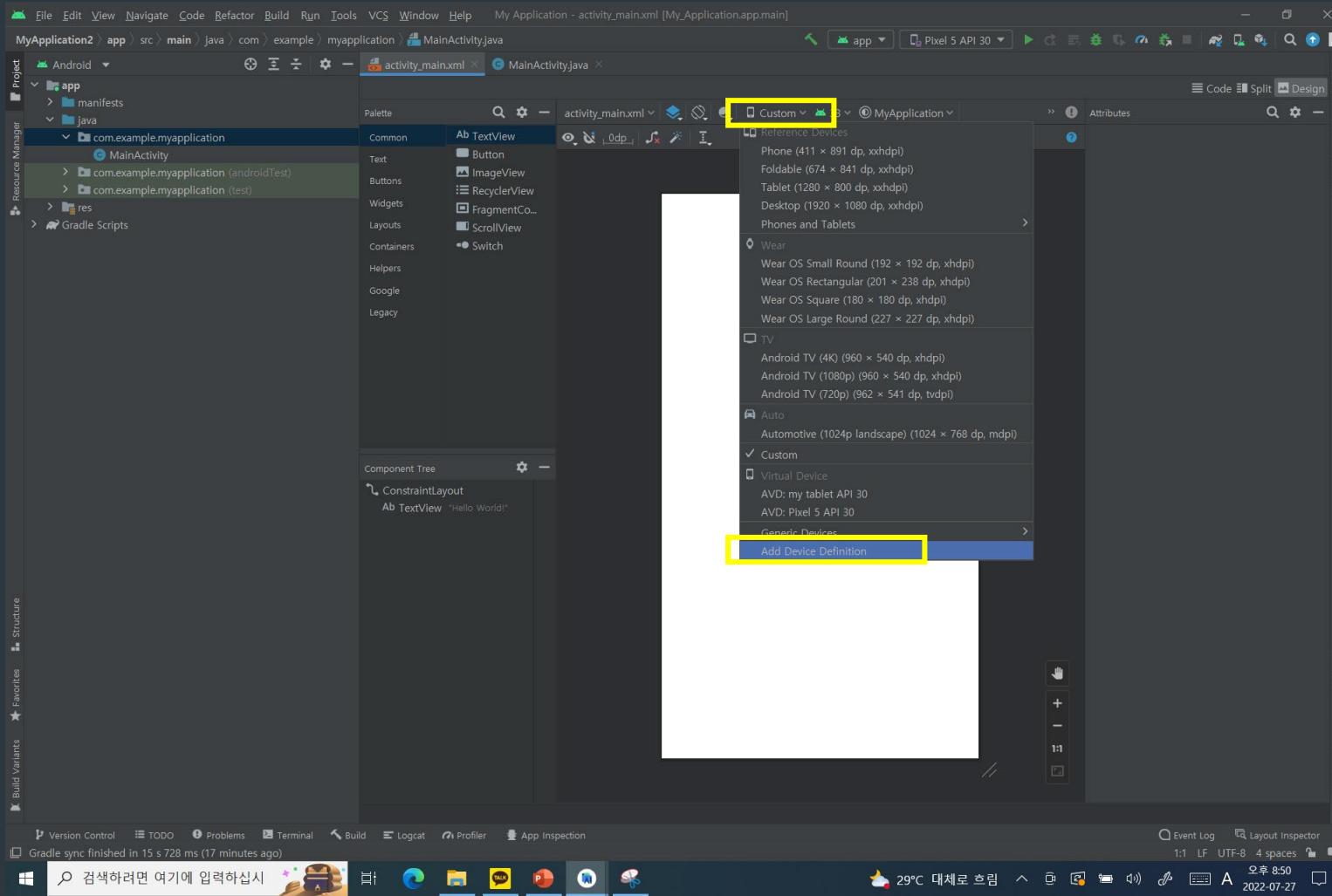
public void onButton1Clicked(View v) {
    Gpio gpio3 = new Gpio(TinkerBoard.TINKERBOARD_PIN3.swigValue());
    gpio3.dir(Dir.DIR_OUT);
    gpio3.write( i: 1);
    Toast.makeText(context: this, text: "LED가 켜졌습니다.", Toast.LENGTH_LONG).show();
}
```

MainActivity.java의 onCreate 밖에서 정의

과일 판매 키오스크 예제



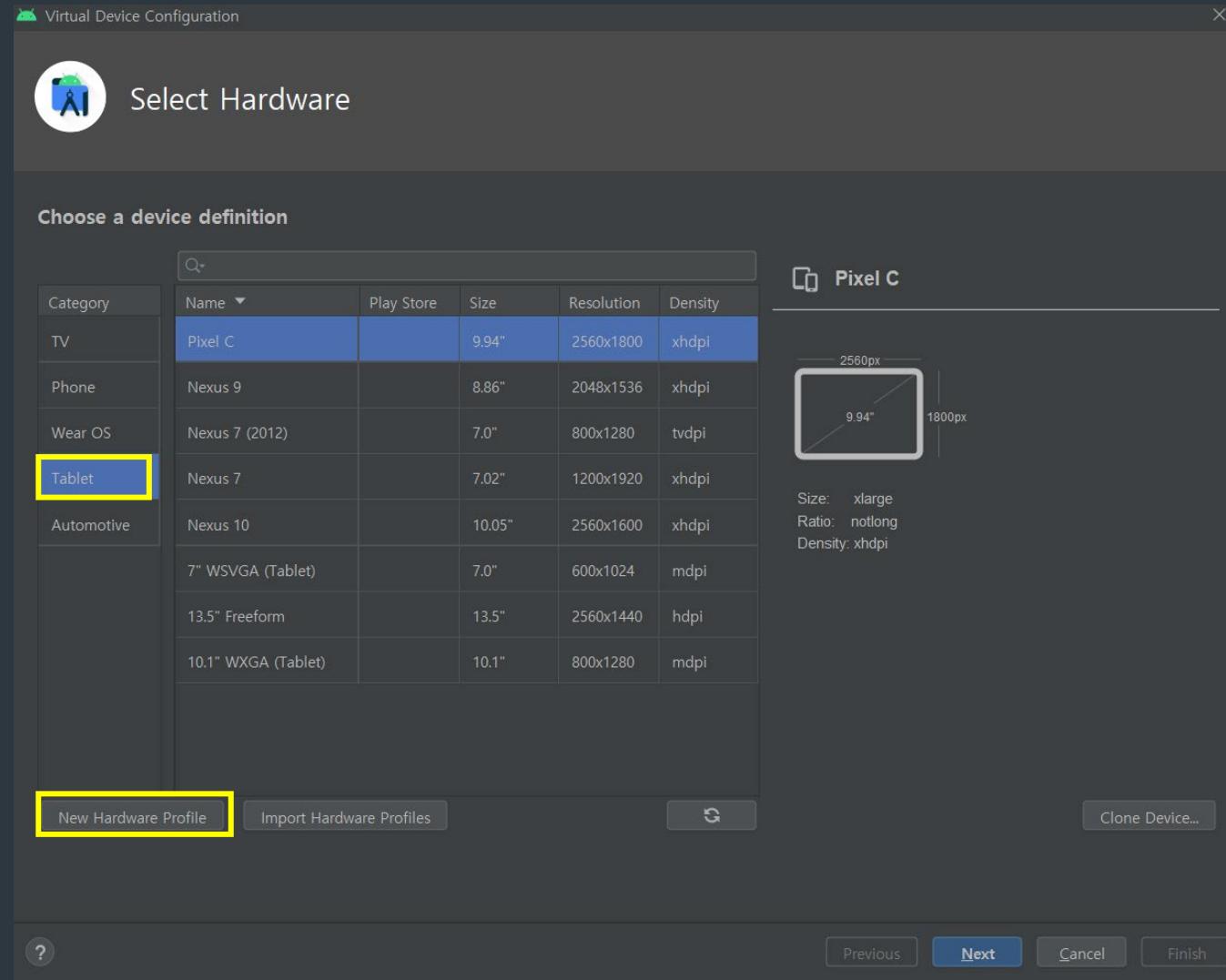
Custom layout 설정하기



1. Custom layout을 만들기 위해 Add device defftion을 눌러줍니다.

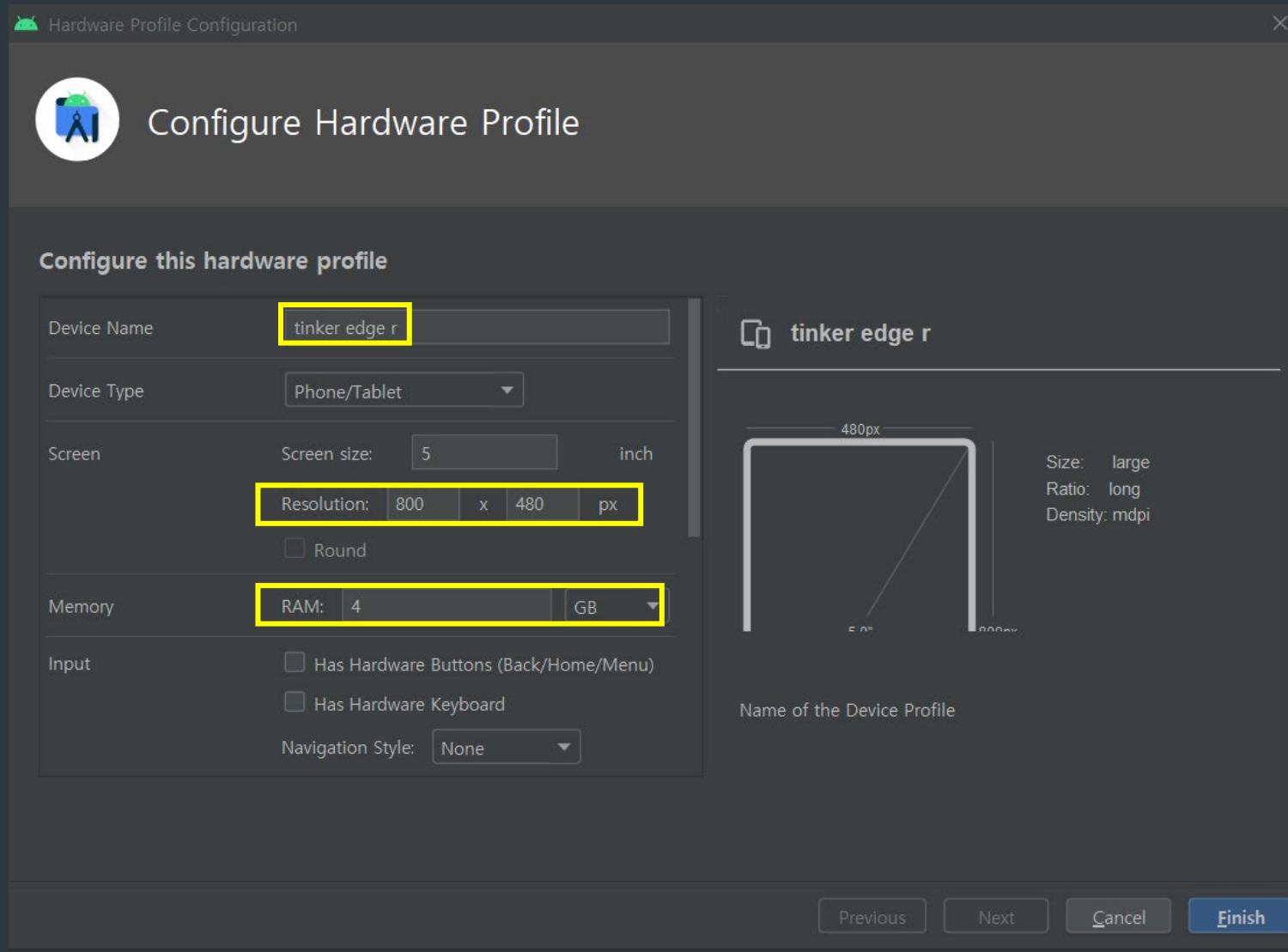


Custom layout 설정하기



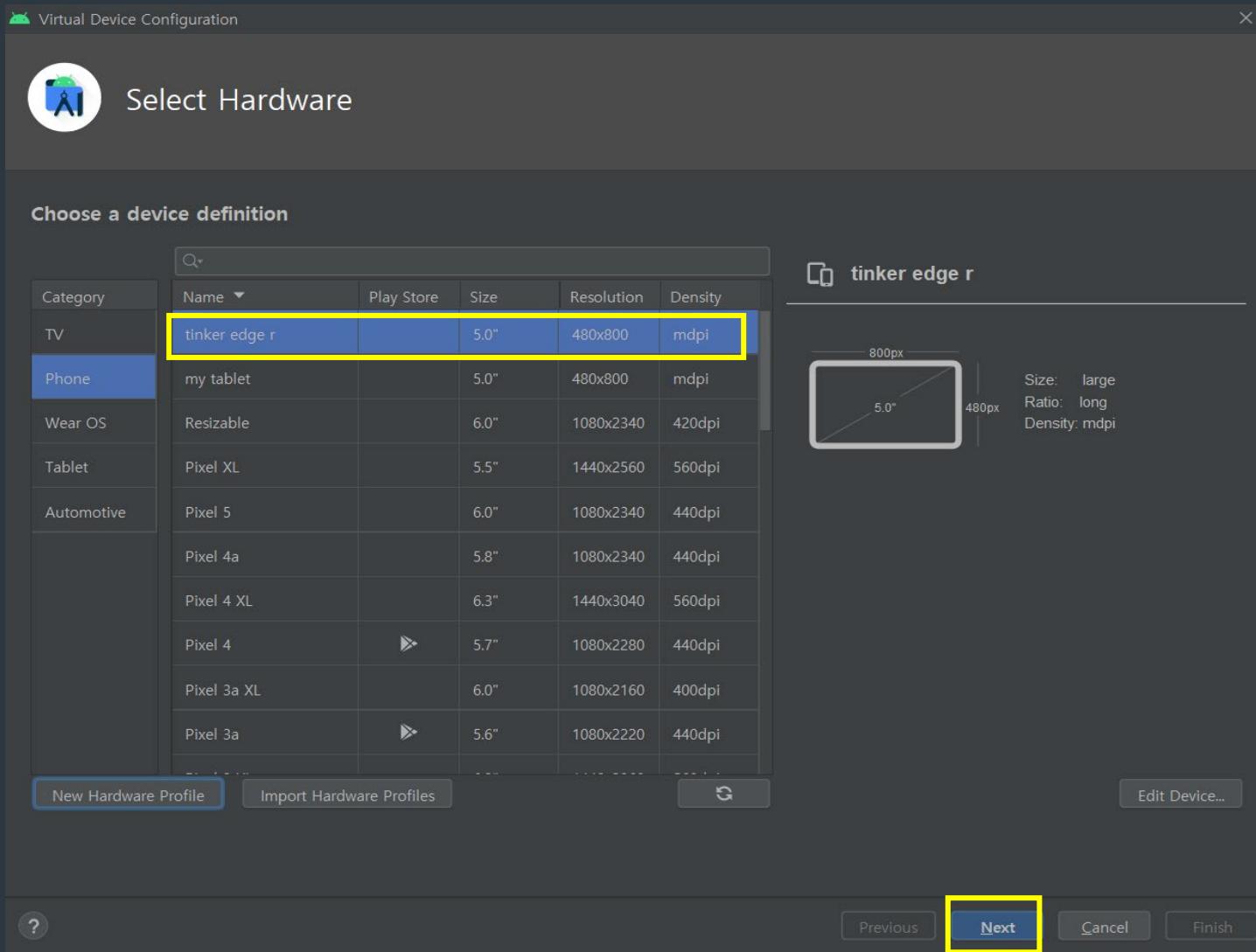
1. Tablet을 클릭하여 평평한 모니터 사이즈에 맞게 사용합니다.
2. New Hardware Profile을 클릭하여 새로운 layout을 만들어줍니다.

Custom layout 설정하기



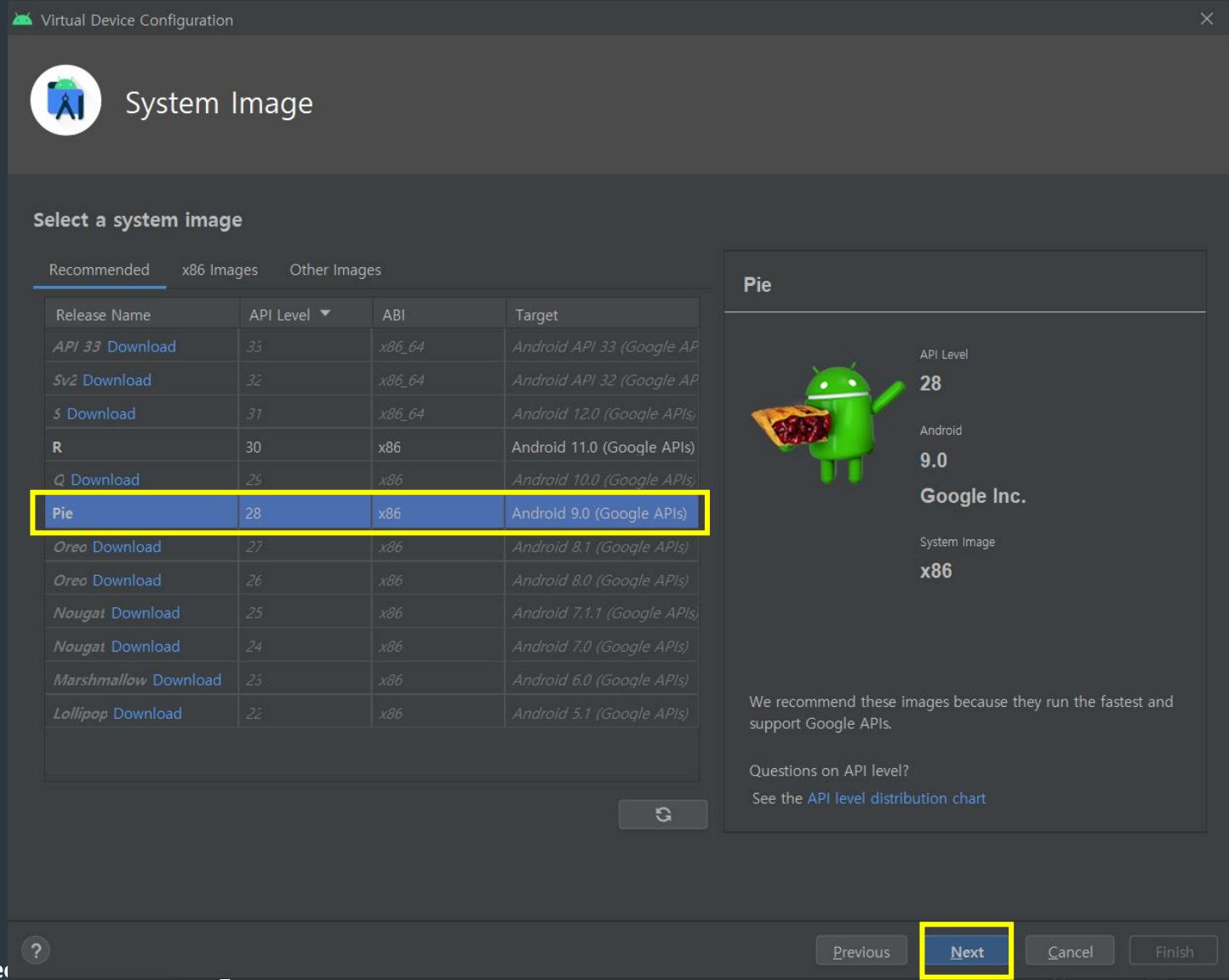
1. Device Name: 원하는 tablet layout 이름을 설정합니다.
2. Resolution: 우리의 화면 모니터 사이즈 800 * 480을 설정하여 줍니다.
3. Ram: GB로 설정 변경 후 4를 입력하여 줍니다.

Custom layout 설정하기



1. 우리가 만든 tinker edge r을 선택하여 줍니다.

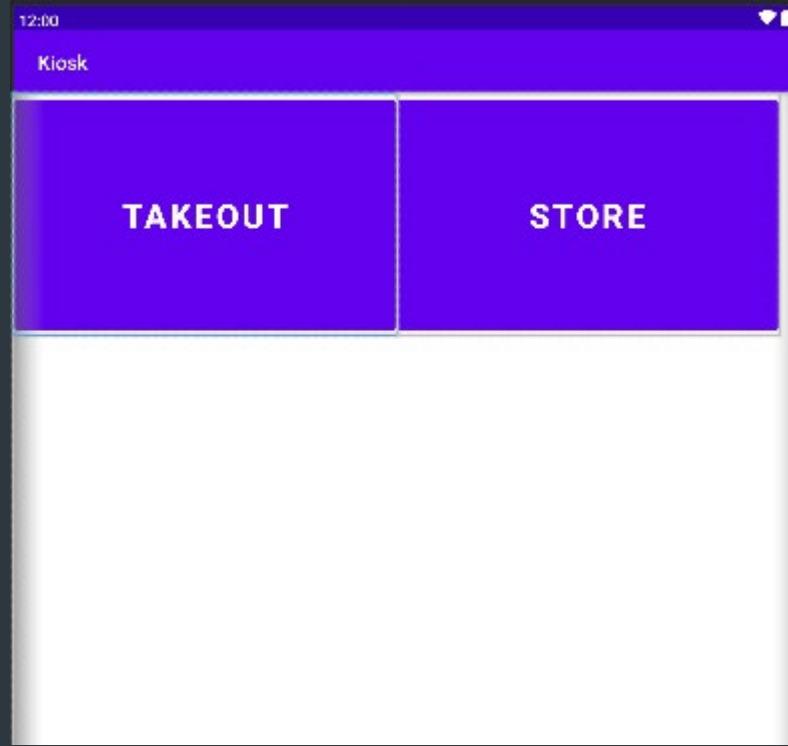
Custom layout 설정하기



1. Android 9.0으로 설정하여 줍니다.

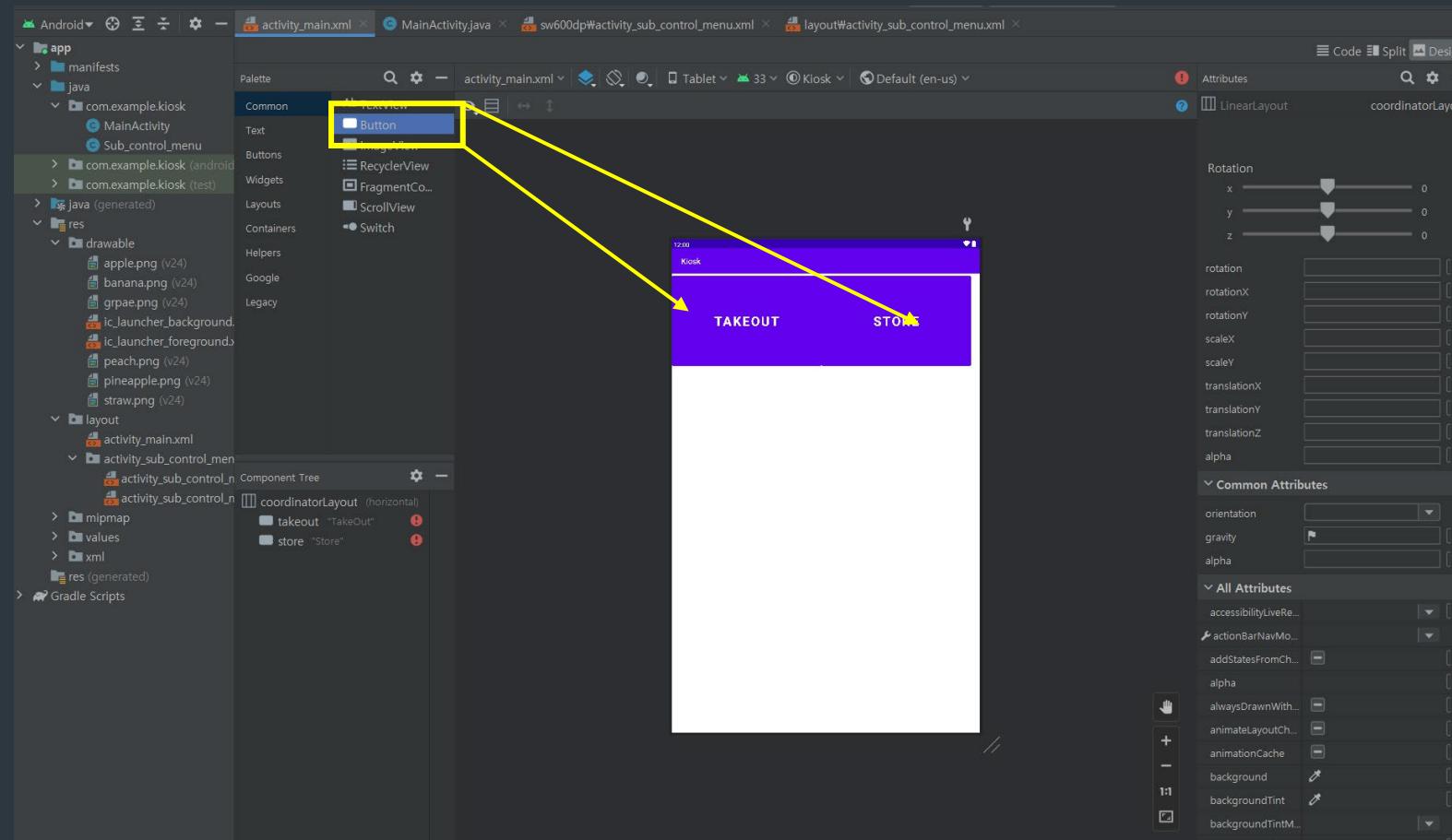


키오스크 예제



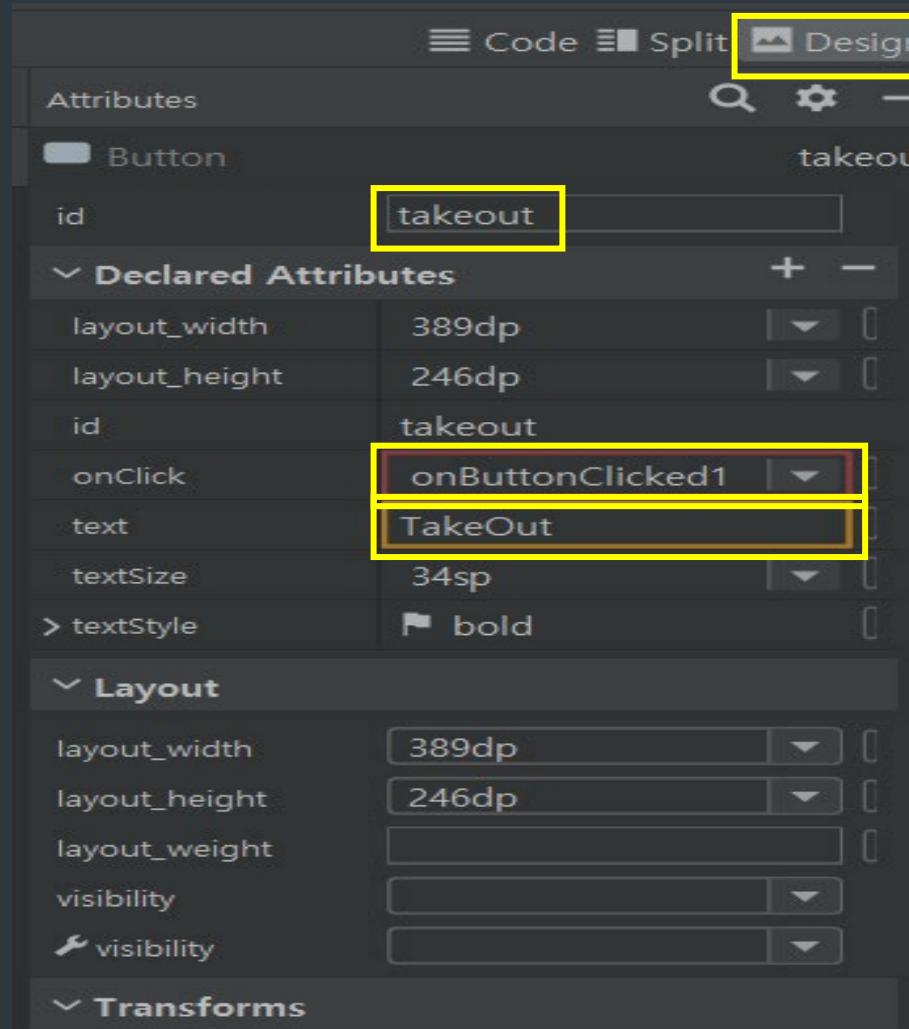
포장과 매장 중 선택을 할 수 있는 창을
만들 예제를 만듭니다.

키오스크 예제



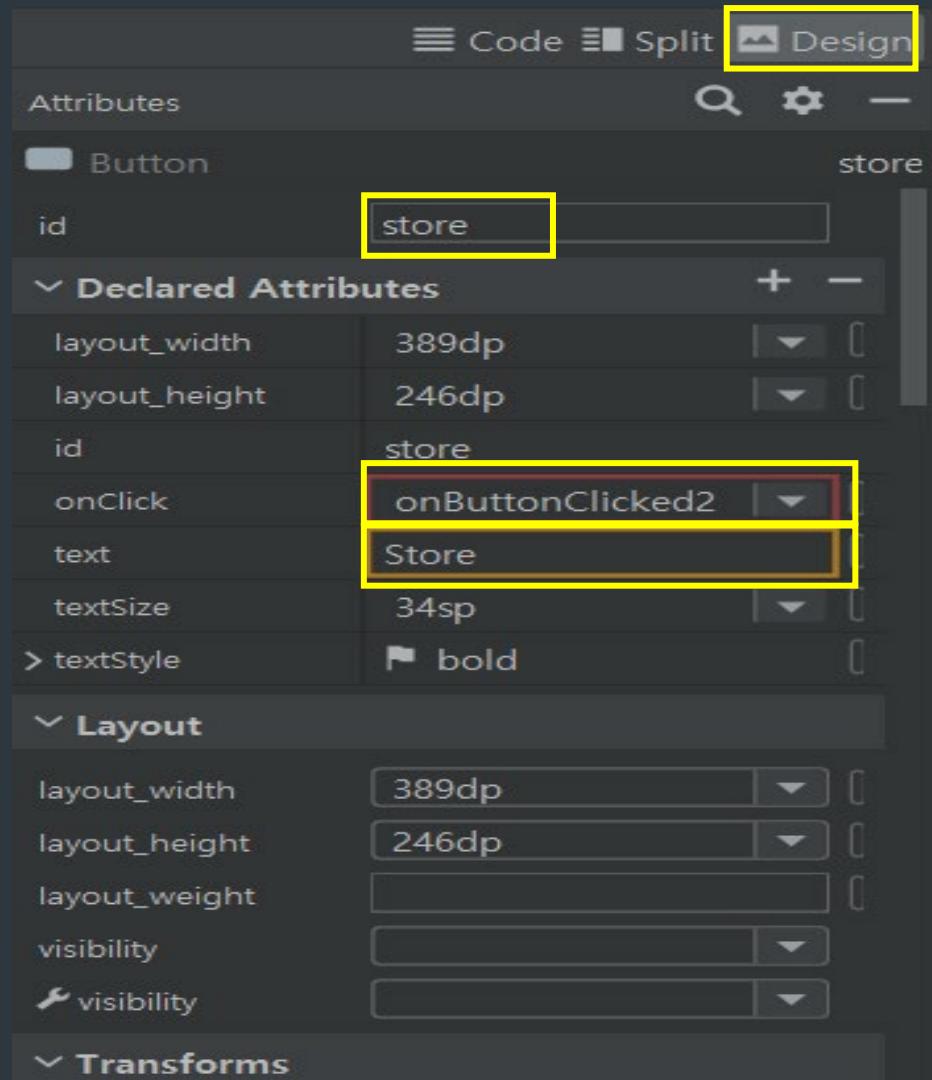
1. 왼쪽 상단에 있는 Button을 2개 끌어 당겨 줍니다.

키오스크 예제-첫번째 take out 버튼 만드는 방법



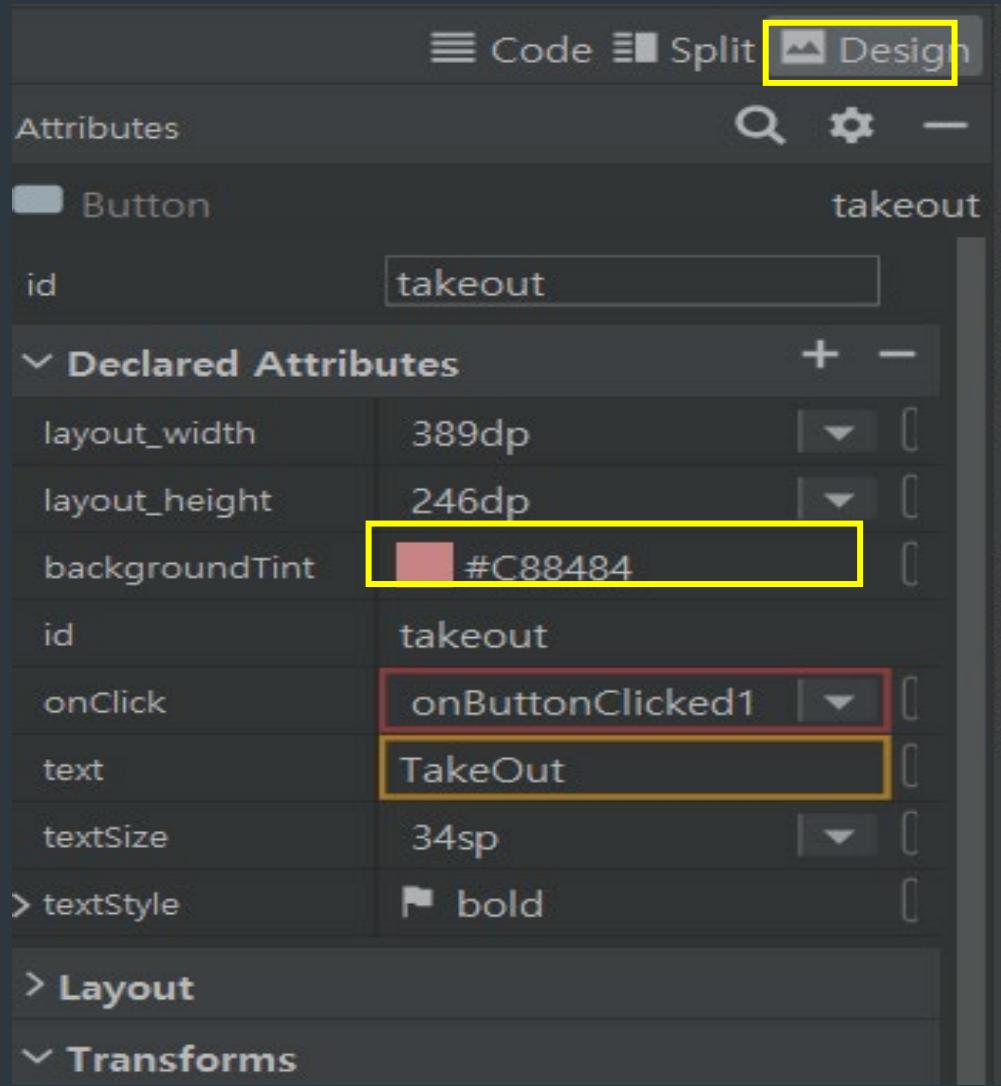
1. Design 버튼 클릭시: 왼쪽의 창을 확인 할 수 있습니다.
2. Id: 원하는 id로 설정하여 줍니다.
3. Onclick: onButtonClicked1을 적어줍니다.
4. Text: Button 안에 들어가는 문구를 작성하여 줍니다.

키오스크 예제-두번째 store 버튼 만드는 방법

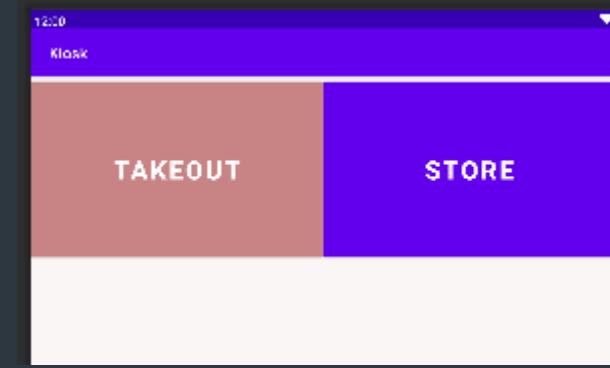


1. Design 버튼 클릭시: 왼쪽의 창을 확인 할 수 있습니다.
2. Id: 원하는 id로 설정하여 줍니다.
3. Onclick: onButtonClicked2를 적어줍니다.
4. Text: Button 안에 들어가는 문구를 작성하여 줍니다.

키오스크 예제-두번째 take out 색 변화 방법



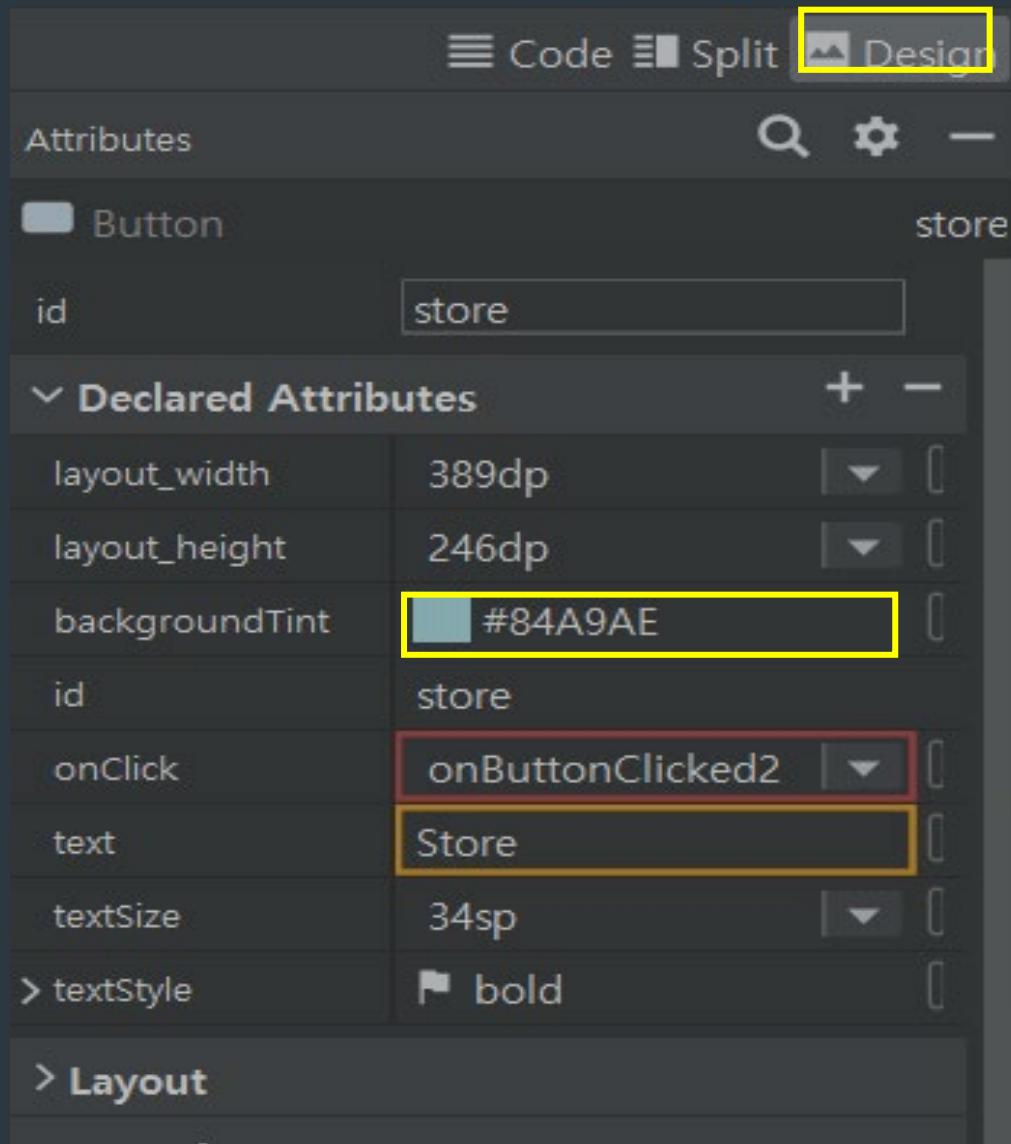
1. backgroundTint: button의 색을 바꿀수 있습니다.



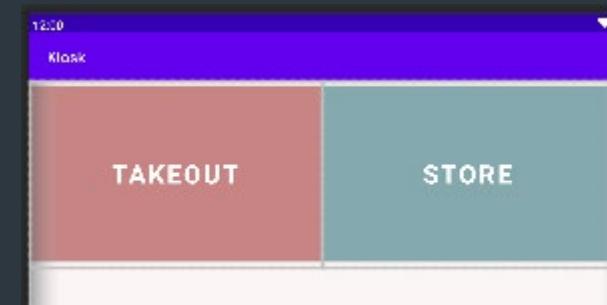
```
<Button  
    android:id="@+id/takeout"  
    android:layout_width="389dp"  
    android:layout_height="246dp"  
    android:backgroundTint="#C88484"  
    android:onClick="onButtonClicked1"  
    android:text="TakeOut"  
    android:textSize="34sp"  
    android:textStyle="bold" />
```

A screenshot of the Android Studio code editor showing the XML configuration for the 'takeout' button. The 'backgroundTint' attribute is highlighted with a yellow box and set to the value '#C88484'. Other attributes like id, layout_width, layout_height, onClick, text, textSize, and textStyle are also listed.

키오스크 예제-두번째 store 색 변화 방법



1. backgroundTint: button의 색을 바꿀수 있습니다.



```
<Button  
    android:id="@+id/store"  
    android:layout_width="389dp"  
    android:layout_height="246dp"  
    android:backgroundTint="#84A9AE"  
    android:onClick="onButtonClicked2"  
    android:text="Store"  
    android:textSize="34sp"  
    android:textStyle="bold" />
```

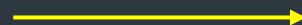
A screenshot of the XML code for the button. The line `android:backgroundTint="#84A9AE"` is highlighted with a yellow box.

키오스크 예제-xml편

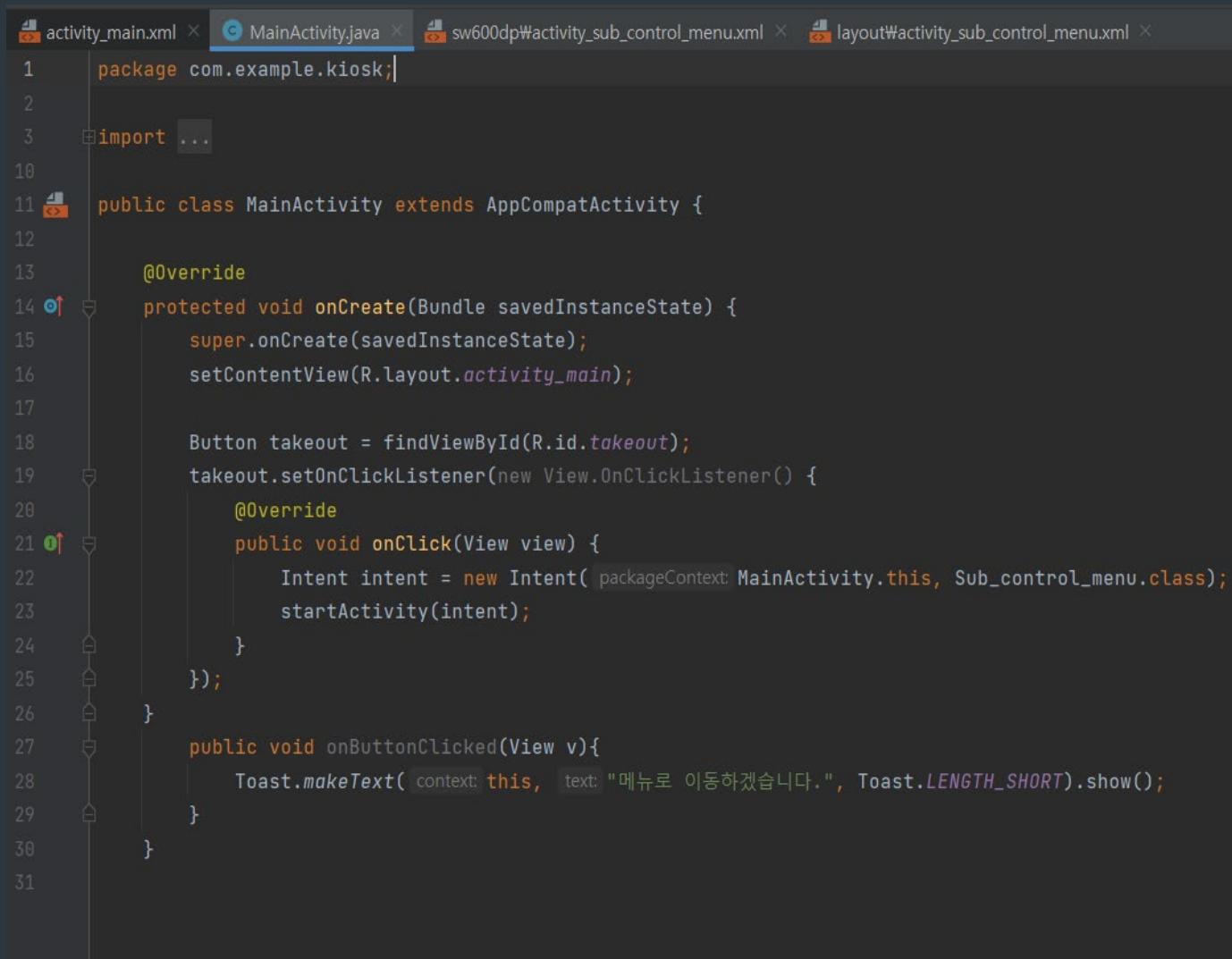
```
activity_main.xml x MainActivity.java x sw600dp\activity_sub_control_menu.xml x layout\acti
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:id="@+id/coordinatorLayout"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     android:background="#FBF6F6"
9     app:layoutDescription="@xml/activity_main_scene"
10    tools:context=".MainActivity">
11
12    <Button
13        android:id="@+id/takeout"
14        android:layout_width="389dp"
15        android:layout_height="246dp"
16        android:backgroundTint="#C88484"
17        android:onClick="onButtonClicked1"
18        android:text="TakeOut"
19        android:textSize="34sp"
20        android:textStyle="bold" />
21
22    <Button
23        android:id="@+id/store"
24        android:layout_width="389dp"
25        android:layout_height="246dp"
26        android:backgroundTint="#84A9AE"
27        android:onClick="onButtonClicked2"
28        android:text="Store"
29        android:textSize="34sp"
30        android:textStyle="bold" />
31
32 </LinearLayout>
```

1. Activity_main.xml은 버튼 등의 Front End를 정의하는 항목입니다.
2. Design으로 버튼 두개를 생성 후 Code란으로 가면 각 버튼을 정의하는 XML 파일로 보입니다.

완성본



키오스크 예제-java편



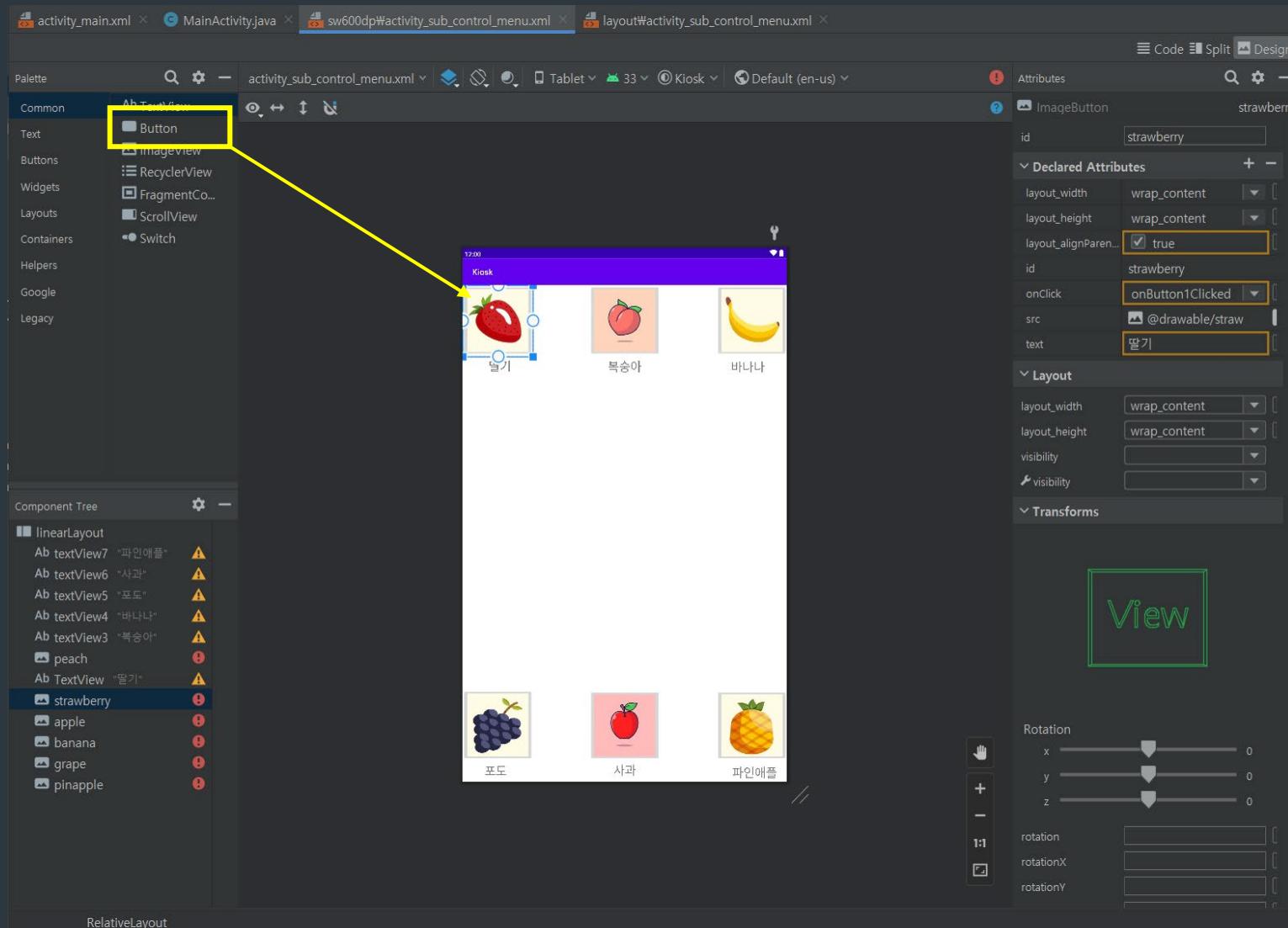
```
activity_main.xml x MainActivity.java x sw600dp\activity_sub_control_menu.xml x layout\activity_sub_control_menu.xml x
1 package com.example.kiosk;
2
3 import ...
10
11 public class MainActivity extends AppCompatActivity {
12
13     @Override
14     protected void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         setContentView(R.layout.activity_main);
17
18         Button takeout = findViewById(R.id.takeout);
19         takeout.setOnClickListener(new View.OnClickListener() {
20             @Override
21             public void onClick(View view) {
22                 Intent intent = new Intent(getApplicationContext(), Sub_control_menu.class);
23                 startActivity(intent);
24             }
25         });
26     }
27
28     public void onButtonClicked(View v){
29         Toast.makeText(this, "메뉴로 이동하겠습니다.", Toast.LENGTH_SHORT).show();
30     }
31 }
```

1. MainActivity.java는 Activity_main.xml 버튼의 행동을 정의하는 Backend
2. Java로 각 버튼 클릭시의 행동을 제어 할 수 있습니다.

완성본

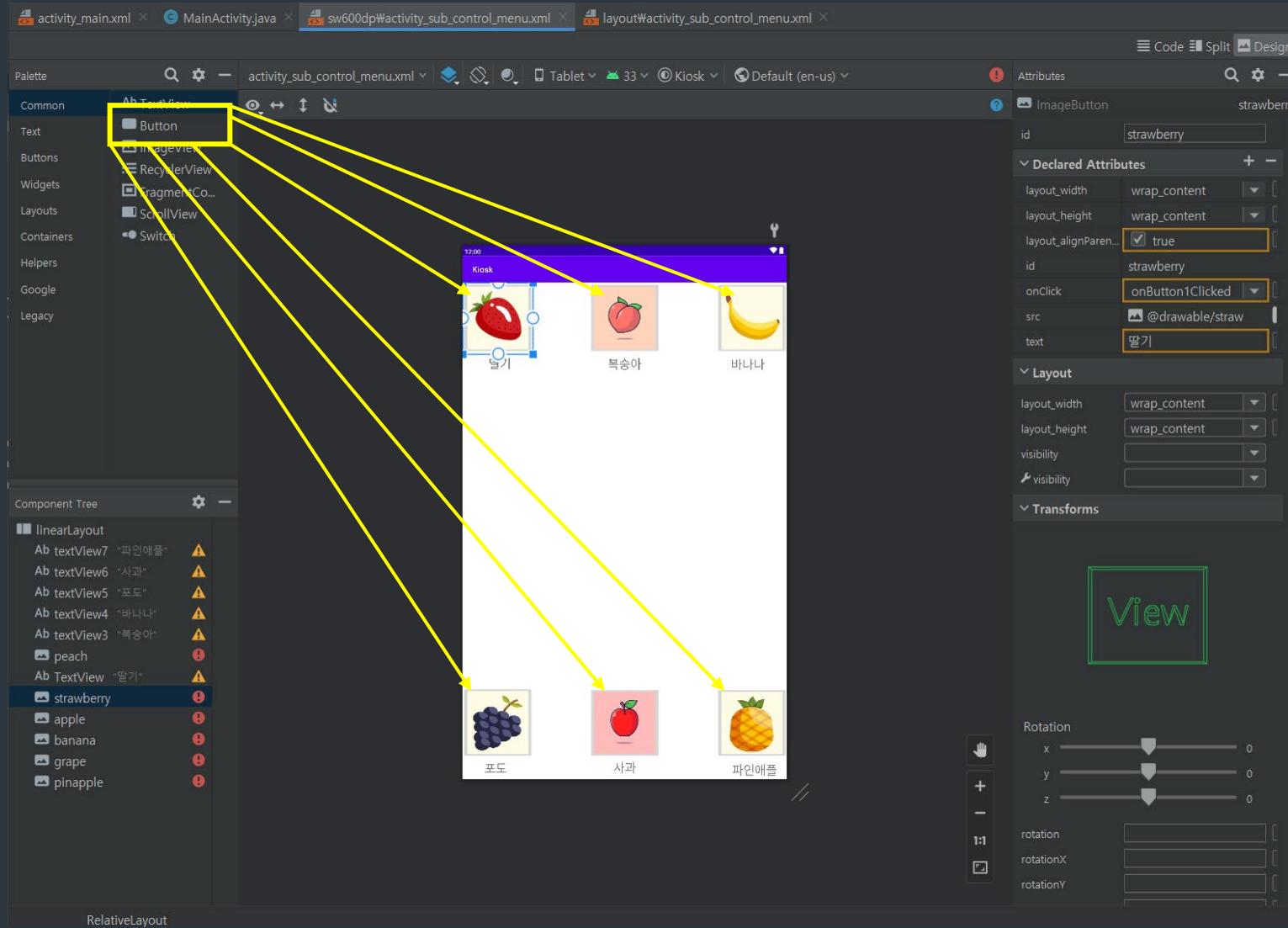


키오스크 예제-딸기



1. 왼쪽 상단에 있는 Button을 6개를 끌어 당겨 줍니다.

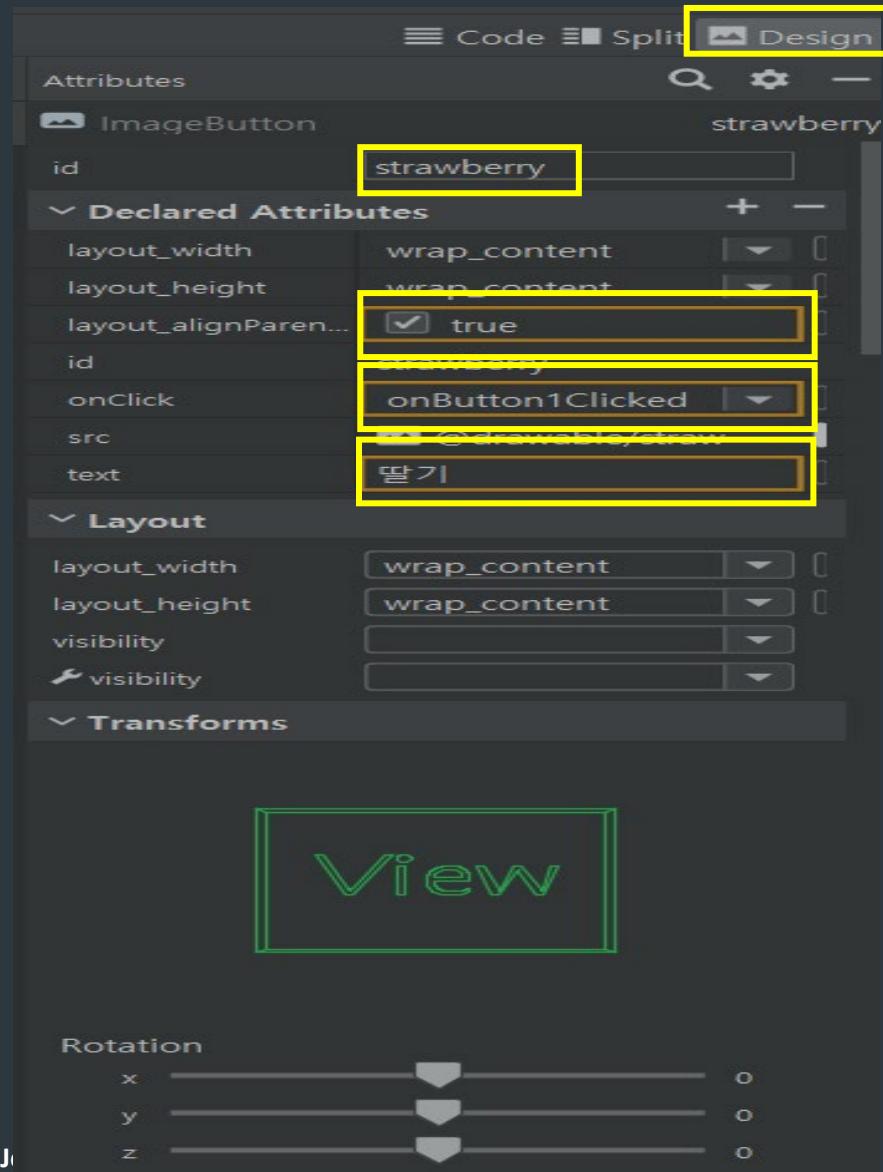
키오스크 예제



1. 왼쪽 상단에 있는 Button을 6개를 끌어 당겨 줍니다.



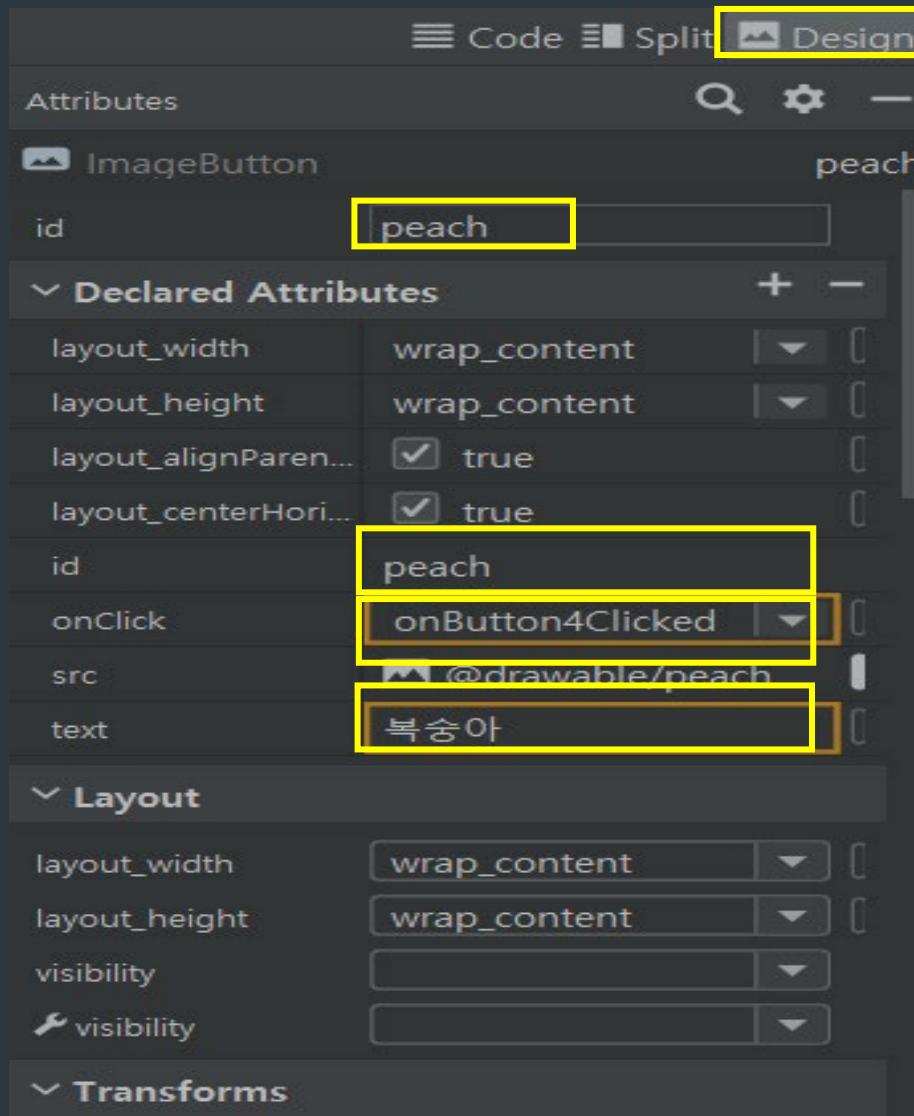
키오스크 예제-딸기



1. Design 버튼 클릭시: 왼쪽의 창을 확인 할 수 있습니다.
2. Id: 원하는 id로 설정하여 줍니다.
3. Onclick: onButton1Clicked을 적어줍니다.
4. Text: Button 안에 들어가는 문구를 작성하여 줍니다.



키오스크 예제-복숭아



1. Design 버튼 클릭시: 왼쪽의 창을 확인 할 수 있습니다.
2. Id: 원하는 id로 설정하여 줍니다.
3. Onclick: onButton4Clicked을 적어줍니다.
4. Text: Button 안에 들어가는 문구를 작성하여 줍니다.

→ 이와 같은 방법으로 총 6개의 Button을 완성 시켜 줍니다.





키오스크 layout

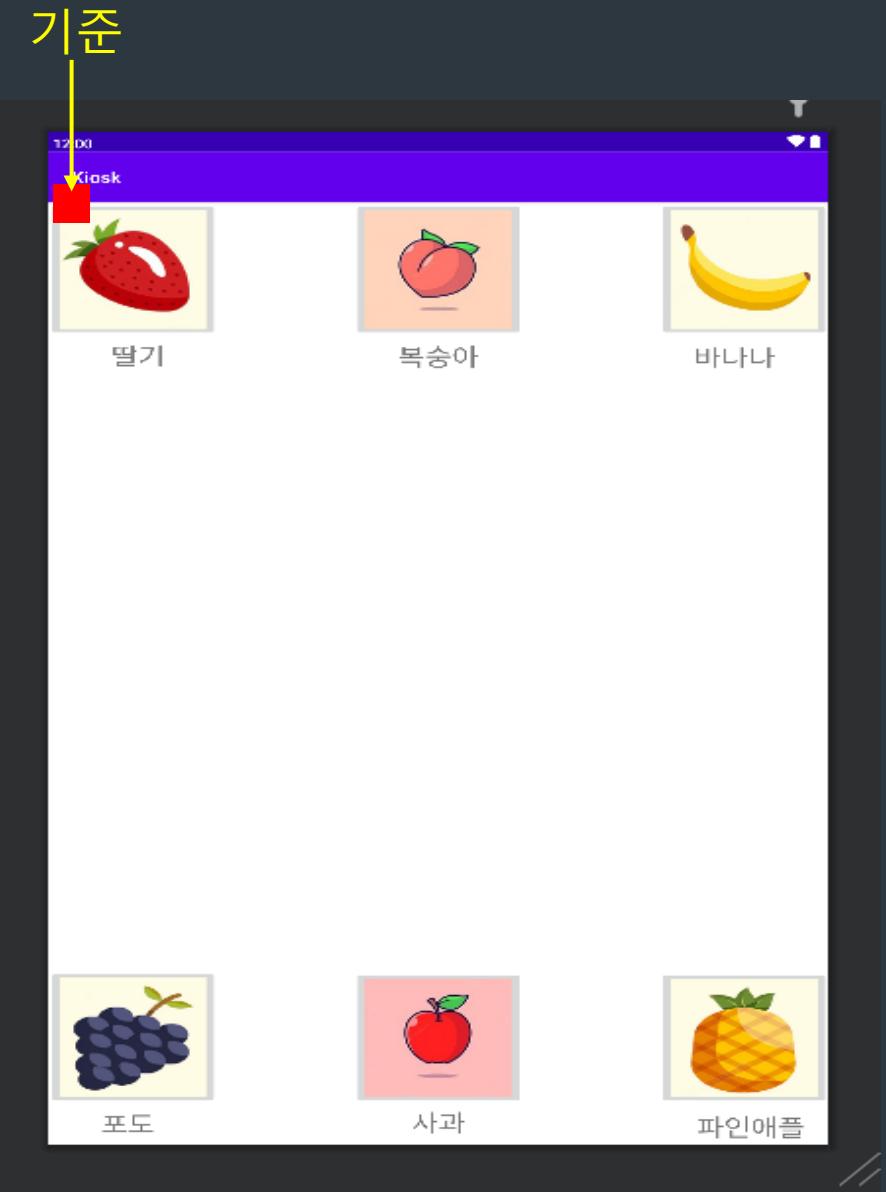
1. layout_alignParentLeft: 뷰(View)를 부모(Parent) 영역 내에서 왼쪽(Left)에 배치.
2. layout_alignParentTop: 뷰(View)를 부모(Parent) 영역 내에서 위쪽(Top)에 배치.
3. layout_alignParentRight: 뷰(View)를 부모(Parent) 영역 내에서 오른쪽(Right)에 배치.
4. layout_alignParentBottom: 뷰(View)를 부모(Parent) 영역 내에서 아래쪽(Bottom)에 배치.
5. layout_centerHorizontal: 뷰(View)를 부모(Parent) 영역의 가로 방향 가운데 배치.
6. layout_centerVertical: 뷰(View)를 부모(Parent) 영역의 세로 방향 가운데 배치.
7. layout_centerInParent: 뷰(View)를 부모(Parent) 영역의 정 중앙(center)에 배치.





키오스크 layout-딸기

```
<ImageButton  
    android:id="@+id/strawberry"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentLeft="true"  
    android:onClick="onButton1Clicked"  
    android:src="@drawable/straw"  
    android:text="딸기" />
```



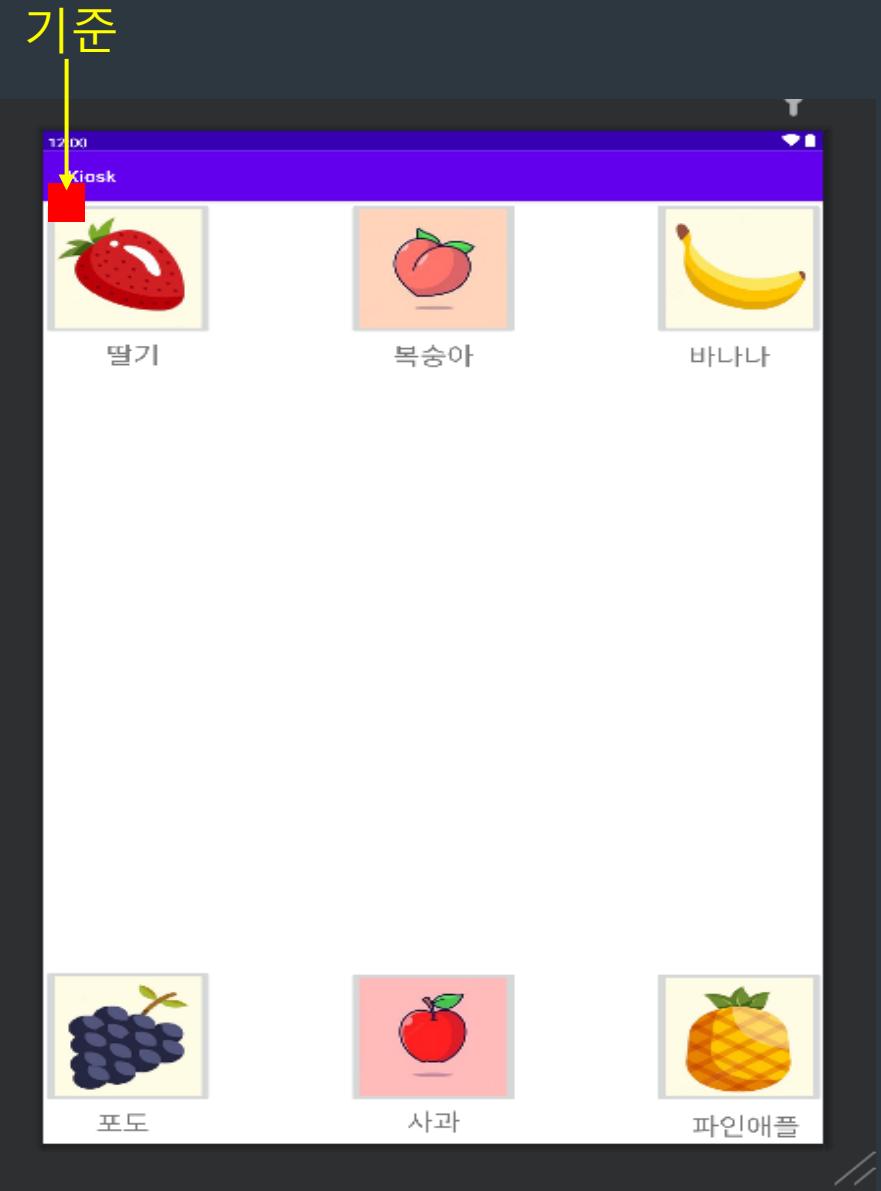
키오스크 layout-복숭아

```
<ImageButton  
    android:id="@+id/peach"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentTop="true"  
    android:layout_centerHorizontal="true"  
    android:onClick="onButton4Clicked"  
    android:src="@drawable/peach"  
    android:text="복숭아" />
```



키오스크 layout-바나나

```
<ImageButton  
    android:id="@+id/banana"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentRight="true"  
    android:layout_marginRight="0dp"  
    android:onClick="onButton5Clicked"  
    android:src="@drawable/banana"  
    android:text="바나나" />
```



키오스크 layout-포도

```
<ImageButton  
    android:id="@+id/grape"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentBottom="true"  
    android:layout_marginBottom="51dp"  
    android:onClick="onButton2Clicked"  
    android:src="@drawable/grape"  
    android:text="포도" />
```



키오스크 layout-사과

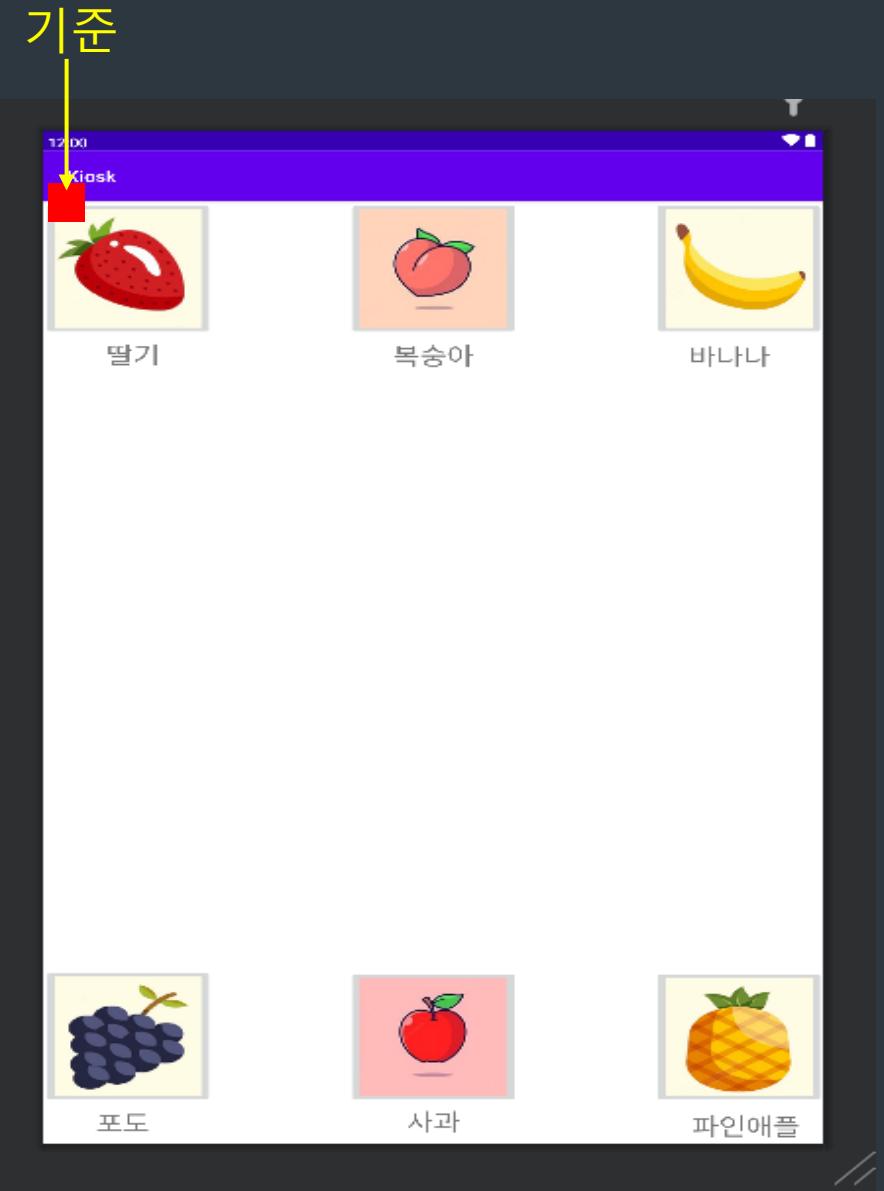
```
<ImageButton  
    android:id="@+id/apple"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentBottom="true"  
    android:layout_centerInParent="true"  
    android:layout_marginBottom="50dp"  
    android:onClick="onButton6Clicked"  
    android:src="@drawable/apple"  
    android:text="사과" />
```





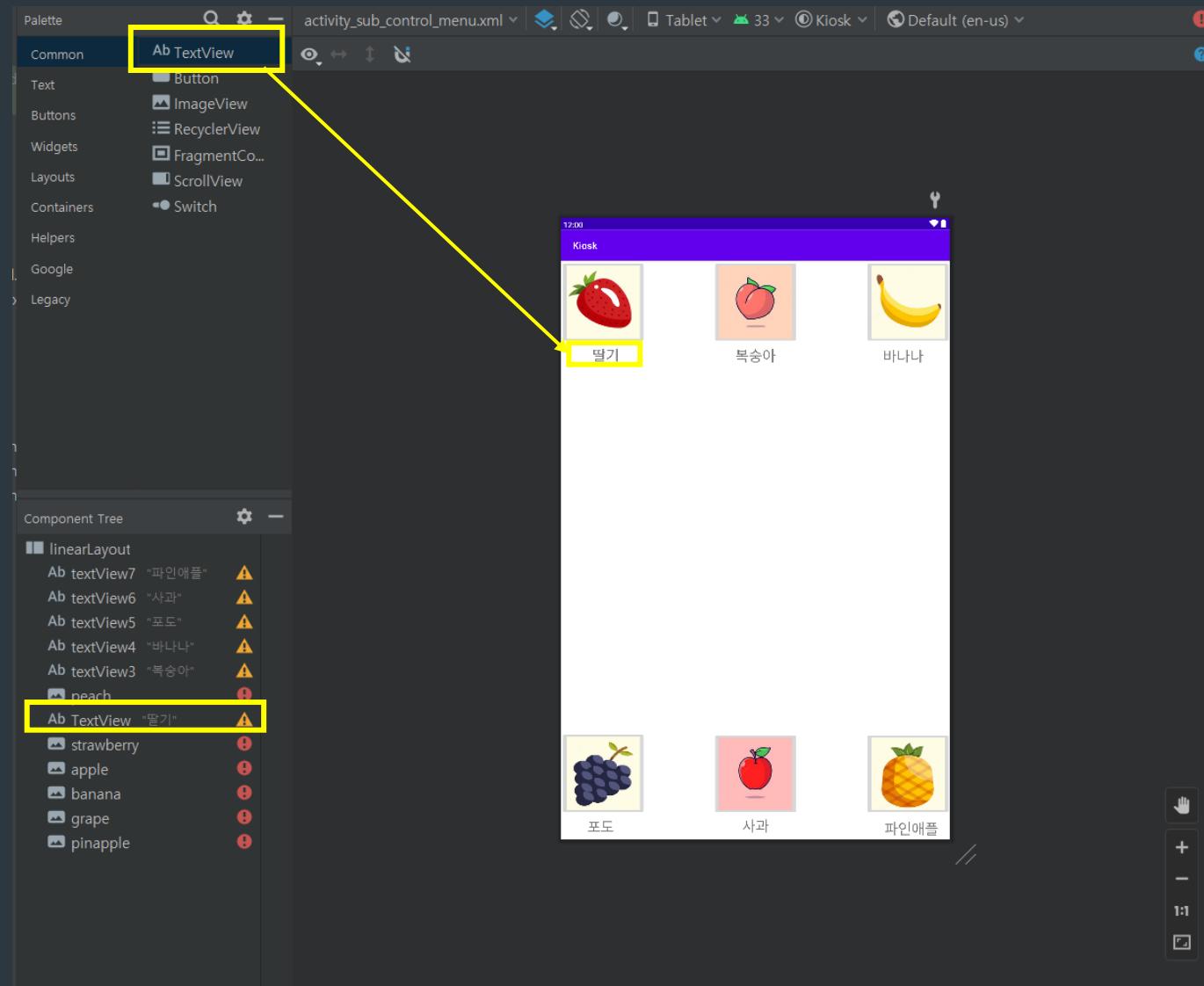
키오스크 layout-파인애플

```
<ImageButton  
    android:id="@+id/pinapple"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_alignParentRight="true"  
    android:layout_alignParentBottom="true"  
    android:layout_marginBottom="50dp"  
    android:onClick="onButton3Clicked"  
    android:src="@drawable/pineapple"  
    android:text="파인애플" />
```





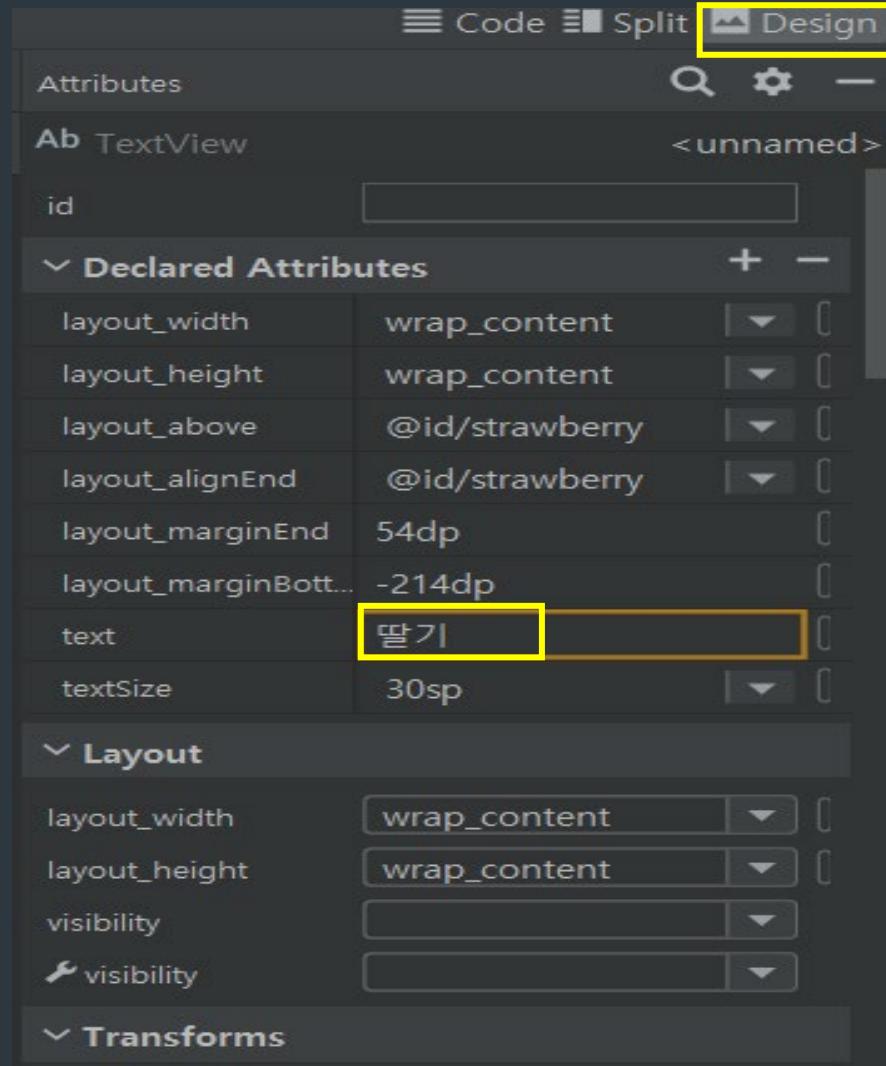
키오스크 예제-text 나타내기



1. 왼쪽 상단에 있는 TextView를 6개를 끌어 당겨 줍니다.

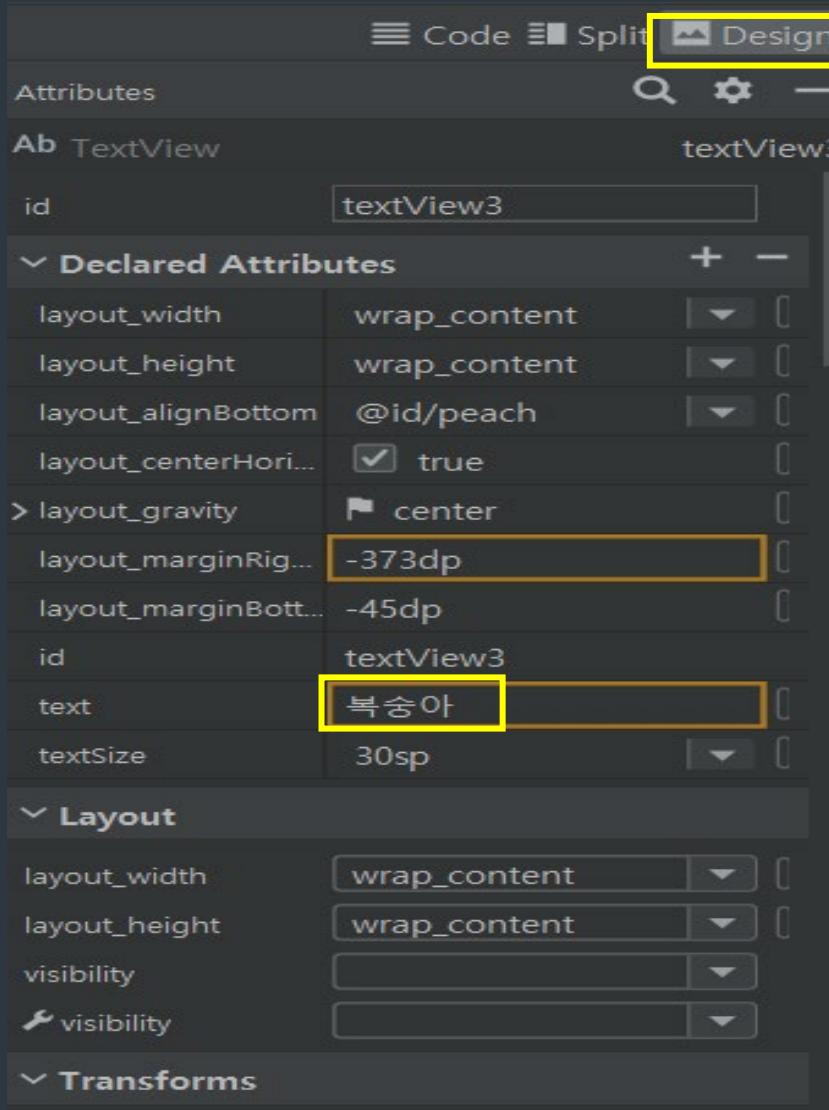


키오스크 예제-딸기



1. Design 버튼 클릭시: 왼쪽의 창을 확인 할 수 있습니다.
2. Text: TextView 창 안에 들어가는 문구를 작성하여 줍니다.

키오스크 예제-복숭아



1. Design 버튼 클릭시: 왼쪽의 창을 확인 할 수 있습니다.
2. Text: TextView 창 안에 들어가는 문구를 작성하여 줍니다.

→ 이와 같은 방법으로 총 6개의 TextView를 완성 시켜 줍니다.

키오스크 예제-사진 삽입(딸기)

The screenshot shows the Android Studio interface with three tabs at the top: activity_main.xml, MainActivity.java, and sw600dp\activity_sub_control_menu.xml. The sw600dp\activity_sub_control_menu.xml tab is selected and highlighted with a yellow box. The XML code in the editor is as follows:

```
94 <ImageButton  
95     android:id="@+id/strawberry"  
96     android:layout_width="wrap_content"  
97     android:layout_height="wrap_content"  
98     android:layout_alignParentLeft="true"  
99     android:onClick="onButton1Clicked"  
100    android:src="@drawable/straw"  
101    android:text="딸기" />  
102
```

The line `android:src="@drawable/straw"` is also highlighted with a yellow box.

1. Xml 파일로 들어가 코드를 확인 합니다.
2. ImageButton: 일반 버튼이 아닌 이미지 삽입을 하므로 ImageButton으로 설정하여 줍니다.
3. Android:src="@drawable/사진에 설정한 이름"으로 만들면 Button에 사진 삽입이 가능 합니다.

▷ 사진 저장시 **.jpg**가 아닌 **.png** 파일로 저장을 해주셔야 합니다.

▷ 사진 크기 **150 * 150**으로 설정해주세요.

A file properties dialog is shown with the following settings:

이미지	사진 크기	150 x 150
너비	150픽셀	
높이	150픽셀	
비트 수준	32	
파일		

An arrow points from the text "▷ 사진 크기 150 * 150으로 설정해주세요." to the "사진 크기" row in the table.



키오스크 예제-사진 삽입(복숭아)

```
activity_main.xml MainActivity.java sw600dp\activity_sub_control_menu.xml
73
74 < MageButton
75     android:id="@+id/peach"
76     android:layout_width="wrap_content"
77     android:layout_height="wrap_content"
78     android:layout_alignParentTop="true"
79     android:layout_centerHorizontal="true"
80     android:onClick="onButton4Clicked"
81     android:src="@drawable/peach"
82     android:text="복숭아" />
```

1. Xml 파일로 들어가 코드를 확인 합니다.
2. ImageButton: 일반 버튼이 아닌 이미지 삽입을 하므로 ImageButton으로 설정하여 줍니다.
3. Android:src="@drawable/사진에 설정한 이름"으로 만들면 Button에 사진 삽입이 가능 합니다.

→ 이와 같은 방법으로 총 6개의 image를 생성시켜 완성 시켜 줍니다.

키오스크 예제-xml 코딩(1)

The screenshot shows the Android Studio interface with four tabs open:

- activity_main.xml
- MainActivity.java
- sw600dp#activity_sub_control_menu.xml
- layout#activity_sub_control_menu.xml

The layout#activity_sub_control_menu.xml file is the active tab, displaying the following XML code:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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/linearLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layoutDescription="@xml/activity_sub_control_menu_scene"
    tools:context=".Sub_control_menu">

    <TextView
        android:id="@+id/textView7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/pineapple"
        android:layout_alignBottom="@+id/pineapple"
        android:layout_centerHorizontal="true"
        android:layout_gravity="center"
        android:layout_marginLeft="39dp"
        android:layout_marginBottom="-48dp"
        android:text="파인애플"
        android:textSize="30sp"></TextView>

    <TextView
        android:id="@+id/textView6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/apple"
        android:layout_centerHorizontal="true"
        android:layout_gravity="center"
        android:layout_marginBottom="-42dp"
        android:text="사과"
        android:textSize="30sp"></TextView>

    <TextView
        android:id="@+id/textView5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/grape"
        android:layout_alignBottom="@+id/grape"
        android:layout_centerHorizontal="true"
        android:layout_gravity="center"
        android:layout_marginLeft="54dp"
        android:layout_marginBottom="-45dp"
        android:text="포도"
        android:textSize="30sp"></TextView>

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/banana"
        android:layout_alignBottom="@+id/banana"
        android:layout_centerHorizontal="true"
        android:layout_gravity="center"
        android:layout_marginLeft="36dp"
        android:layout_marginRight="-828dp"
        android:layout_marginBottom="-46dp"
        android:text="바나나"
        android:textSize="30sp"></TextView>
```

키오스크 예제-xml 코딩(2)

```
61
62     <TextView
63         android:id="@+id/textView3"
64         android:layout_width="wrap_content"
65         android:layout_height="wrap_content"
66         android:layout_alignBottom="@+id/peach"
67         android:layout_centerHorizontal="true"
68         android:layout_gravity="center"
69         android:layout_marginRight="-373dp"
70         android:layout_marginBottom="-45dp"
71         android:text="복숭아"
72         android:textSize="30sp"></TextView>
73
74     <ImageButton
75         android:id="@+id/peach"
76         android:layout_width="wrap_content"
77         android:layout_height="wrap_content"
78         android:layout_alignParentTop="true"
79         android:layout_centerHorizontal="true"
80         android:onClick="onButton4Clicked"
81         android:src="@drawable/peach"
82         android:text="복숭아" />
83
84     <TextView
85         android:layout_width="wrap_content"
86         android:layout_height="wrap_content"
87         android:layout_above="@+id/strawberry"
88         android:layout_alignEnd="@+id/strawberry"
89         android:layout_marginEnd="54dp"
90         android:layout_marginBottom="-214dp"
91         android:text="딸기"
92         android:textSize="30sp"></TextView>
93
```

```
94
95     <ImageButton
96         android:id="@+id/strawberry"
97         android:layout_width="wrap_content"
98         android:layout_height="wrap_content"
99         android:layout_alignParentLeft="true"
100        android:onClick="onButton1Clicked"
101        android:src="@drawable/straw"
102        android:text="딸기" />
103
104     <ImageButton
105         android:id="@+id/apple"
106         android:layout_width="wrap_content"
107         android:layout_height="wrap_content"
108         android:layout_alignParentBottom="true"
109         android:layout_centerInParent="true"
110         android:layout_marginBottom="50dp"
111        android:onClick="onButton6Clicked"
112        android:src="@drawable/apple"
113        android:text="사과" />
114
115     <ImageButton
116         android:id="@+id/banana"
117         android:layout_width="wrap_content"
118         android:layout_height="wrap_content"
119         android:layout_alignParentRight="true"
120         android:layout_marginRight="0dp"
121        android:onClick="onButton5Clicked"
122        android:src="@drawable/banana"
123        android:text="바나나" />
```

키오스크 예제-xml 코딩(3)

```
124 <ImageButton  
125     android:id="@+id/grape"  
126     android:layout_width="wrap_content"  
127     android:layout_height="wrap_content"  
128     android:layout_alignParentBottom="true"  
129     android:layout_marginBottom="51dp"  
130     android:onClick="onButton2Clicked"  
131     android:src="@drawable/grape"  
132     android:text="포도" />  
  
134 <ImageButton  
135     android:id="@+id/pineapple"  
136     android:layout_width="wrap_content"  
137     android:layout_height="wrap_content"  
138     android:layout_alignParentRight="true"  
139     android:layout_alignParentBottom="true"  
140     android:layout_marginBottom="50dp"  
141     android:onClick="onButton3Clicked"  
142     android:src="@drawable/pineapple"  
143     android:text="파인애플" />  
  
144 </RelativeLayout>
```

완성 후 디자인

