● 머티리얼 디자인(Material Design): 컴퓨터에서 수행되는 소프트웨어의 디자인도 실제 사물처럼 느껴지도록 해야 한다는 개념. 안드로이드에서는 머티리얼 디자인을 적용하여 개발할 수 있도록 라이브러리를 제공한다. 머티리얼 디자인이 제공되는 뷰를 이용하기 위해서는 material 라이브러리가 build.girdle 파일의 dependencies에 선되어 있어야 한다. 프로젝트를 만들면 build.gardle 파일에 material 라이브러리가 자동으로 선언된다.

```
implementation 'com.google.android.material:material:1.4.0'
```

● DrawerLayout: androidx에서 제공하는 클래스로 액티비티 화면의 구숑오소이지만 초기 화면에 출력되지 않다가 사용자 손가락 이동이나 툴바의 아이콘 클릭(ActionBarDrawerToggle)에 의해 왼쪽이나 오든쪽에서 밀려 들어오는 화면을 지칭한다. DrawerLayout의 내용은 다양하게 구성할 수 있지만, 대부분 메뉴를 제공하는 역할을 한다. 레이아웃 XML 파일의 구성이 중요하다. 전체 레이아웃 XML 파일을 DrawerLayout으로 감싸야 하며, 첫 번째로 추가 된 뷰가 액티비티 화면에 기본으로 보이는 뷰이고, 두 번째로 추가된 뷰가 DrawerLayout에 의해 화면에 나타난다. layout_gravity 속성은 화면에 나타나는 방향을 지정하는데 사용하며, 값을 left, right, start로 지정할 수 있다. start로 지정할 경우 시스템의 언어 설정에 따라 left, right가 알아서 선택되는데, 한국어처럼 왼쪽에서 오든쪽으로 쓰는 언어는 left, 아랍어는 right가 자동으로 선택된다.

```
<androidx.drawerlayout.widget.DrawerLayout>
    <LinearLayout>
    </LinearLayout>
    <TextView
        android:layout_gravity="start"/>
</androidx.drawerlayout.widget.DrawerLayout>
```

DrawerLayout을 열거나 닫기 위한 아이콘을 툴바 왼쪽에 제공하여 이를 통해서도 열리거나 닫히게 제공하고 싶으면, ActionBar DrawerToggle을 제공해야 한다. 이를 위해서는 자바 코드로 작업해야 한다.

```
ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, binding.drawerLayout, R.s
getSupportActionBar().setDisplayHomeAsUpEnabled(true);
toggle.syncState();
```

ActionBarDrawerToggle을 생성하면서 세네 번째 매개변수로 문자열 리소스를 지정해 주었는데, 이 문자열은 DrawerLayout이 열린 상태, 닫힌 상태를 표현하기 위한 문자열이며, 개발자 임의의 문자열을 지정할 수 있다.

또한 ActionBarDrawerToggle 생성 시 이벤트를 추가할 수 있다.

```
ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, binding.drawerLayout, R.s
    @Override
    public void onDrawerOpened(view drawerView) {
        super.onDrawerOpened(drawerView);
        //.....
    }

    @Override
    public void onDrawerClosed(View drawerView) {
        super.onDrawerClosed(drawerView);
        //....
    }
};
binding.drawerLayout.addDrawerListener(toggle);
```

ActionBarDrawerToggle을 상속받는 서브 클래스를 만들어 그 안에 onDrawerOpened() 함수와 onDrawerClosed() 함수를 정의해 놓으면, DrawerLayout이 열리거나 닫히는 순간 이 함수들이 호출되어 이벤트를 처리할 수 있다. 이렇게 만든 이벤트 핸들러를 DrawerLayout 객체에 addDrawerListener() 함수로 등록하여 사용한다.

위의 코드로 툴바에 ActionBarDrawerToggle 아이콘이 나오기는 하지만, 실제 아이콘 클릭 시 DrawerLayout이 열리거나 닫히지는 않는다. 메뉴 이벤트 함수를 정의하여 다음처럼 작성해야 열리고 닫힌다.

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    if (toggle.onOptionsItemSelected(item)) {
        return true;
    }
    return super.onOptionsItemSelected(item);
}
```

onOptionsItemSelected()는 Menu 부분에서 보았던 함수로 메뉴 클릭 이벤트를 처리한다. ActionBarDrawerToggle을 위해 메뉴 이벤트 함수가 사용되는 이유는 내부적으로 ActionBarDrawerToggle 아이콘 크릭이 메뉴 이벤트로 처리되기 때문이다. onOptionsItemSelected() 함수 내의 이벤트가 토글에서 발생한 거라면 반환해서 메뉴 이벤트 로직에서 벗어나게 해주어야 한다.

● NavigationView: DrawerLayout으로 구현하는 화면을 더 쉽게 구성할 수 있도록 하며, 메뉴 기법을 이용하여 화면을 쉽게 구성하고 사용자 이벤트를 처리할 수 있다. NavigationView의 화면 구성을 위한 속성은 두 개 추가되어야 하는데, app:headerLayout 속성은 NavigationView의 윗부분을 구성하기 위한 레이아웃 XML 파일이며, app:menu는 NavigationView의 메뉴 목록을 구성하기 위한 메뉴 XML 파일이다.

```
<androidx.drawerlayout.widget.DrawerLayout>

    <LinearLayout>
        <!--- 중략 --->
    </LinearLayout>

    <com.google.android.material.navigation.NavigationView
         android:layout_width="match_parent"
         android:layout_height="match_parent"
         android:layout_gravity="start"
         app:headerLayout="@layout/navigation_header"
         app:menu="@menu/menu_drawer"/>

</androidx.drawerlayout.widget.DrawerLayout>
```

NavigationView에서의 사용자 항목 선택 이벤트 처리는 OnNavigationItemSelectedListener을 구현한 이벤트 핸들러를 등록하여 처리할 수 있다.

```
binding.navigationView.setNavigationItemSelectedListener (menuItem -> {
    int id = menuItem.getItemId();
    //....
    return true;
});
```

```
// navigation_header.xml

<?xml version="1.0" encoding="utf-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="150dp"
    android:background="@android:color/holo_orange_light"
    android:text="Hello kkang"
    android:textSize="20dp"
    android:textStyle="bold"
    android:gravity="bottom"
    android:padding="20dp"
    android:textColor="@android:color/white"/>
```

```
// res/layout/menu - menu_drawer.xml

<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/menu_drawer_home"
        android:title="Home"
        android:icon="@android:drawable/ic_menu_help"/>
    <item
        android:id="@+id/menu_drawer_manage"
        android:title="Manage"
        android:icon="@android:drawable/ic_menu_manage"/>
    <item
        android:id="@+id/menu_drawer_add"
        android:title="Add"
        android:icon="@android:drawable/ic_menu_add"/>
</menu>
```

```
// activity_main.xml

<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout xmlns:android="http://schemas.android.com/apk
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/drawerLayout">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:gravity="center"
            android:text="HelloWorld"
            android:textSize="20dp"
            android:textStyle="bold"/>
    </LinearLayout>

</androidx.drawerlayout.widget.DrawerLayout>
```

```
// string.xml
// ActionBarDrawerToggle 생성 시 생성자에 명시할 문자열 리소스를
// res/values/string.xml 파일에 다음처럼 추가해 주기

<resources>
    <string name="app_name">PracticeNavigationView</string>
    <string name="drawer_open">Open navigation drawer</string>
    <string name="drawer_close">Close navigation drawer</string>
</resources>
```

```
// MainActivity.java
```

```java
package com.android.practicenavigationview;

import androidx.appcompat.app.ActionBarDrawerToggle;
import androidx.appcompat.app.AppCompatActivity;

import android.app.Notification;
import android.os.Bundle;
import android.view.MenuItem;
import android.widget.Toast;

import com.android.practicenavigationview.databinding.ActivityMainBinding;

public class MainActivity extends AppCompatActivity {
    ActionBarDrawerToggle toggle;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        ActivityMainBinding binding = ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        // ActionBar DrawerToggle - DrawerLayout을 열고 닫기 위한 아이콘을 툴바 왼쪽에 제공
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        toggle = new ActionBarDrawerToggle(this, binding.drawerLayout, R.string.drawer_c
        toggle.syncState();

        // 이벤트 처리
        binding.navigationView.setNavigationItemSelectedListener(menuItem -> {
            int id = menuItem.getItemId();
            if (id == R.id.menu_drawer_home) {
                showToast("NavigationDrawer...home...");
            } else if (id == R.id.menu_drawer_manage) {
                showToast("NavigationDrawer...message...");
            } else if (id == R.id.menu_drawer_add) {
                showToast("NavigationDrawer...add..");
            }
            return false;
        });

    }

    private void showToast(String message) {
        Toast toast = Toast.makeText(this, message, Toast.LENGTH_SHORT);
        toast.show();
    }

    // ActionBarDrawerToggle 아이콘이 나오기는 하지만, 실제 아이콘 클릭 시
    // DrawerLayout이 열리거나 닫히지는 않는다. 메뉴 이벤트 함수를 정의하여,
    // 다음처럼 작성해야 실제 열리거나 닫히게 된다.
    // onOptionsItemSelected() 함수는 메뉴 클릭 이벤트를 처리한다.
    // 내부적으로 ActionBar DrawaerToggle이 메뉴 이벤트로 처리되기 때문이다.
    // onOptionsitemSelected() 함수 내의 이벤트가 토글에서 발생한 거라면,
    // 반환해서 메뉴 이벤트 로직에서 벗어나게 해주어야 한다.
    // 그래야 토글 버튼에 의해서 drawer가 제어가 된다.
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        if (toggle.onOptionsItemSelected(item)) {
            return true;
        }
```

```
        return super.onOptionsItemSelected(item);
    }

}
```

Resources: 깡쌤의 안드로이드 프로그래밍