2019ver

textview

<TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="textview 예제"  
 />

Match\_parent 는 부모 크기만큼

Wrap\_content 는 내용물의 크기만큼

Linear레이아웃 에는 orientation을 붙여주어야 한다.

Vertical은 세로 horizen은 가로이다.

Textcolor =”#ffffff” 를통해 색상을 지정할 수 있고 왼쪽의 박스를 눌러 원하는 색상을 지;정할 수 있다.

Textsize=”nsp”로 크지정이 가능하다.

Edittext 의 경우

Hint를 넣으면 입력 전에 가표시를 할 수 있다.

Button

Text로 버튼의 텍스트를 정할 수 있다.

Id 엔터 엔터 한 뒤 /뒤에 원하는 아이디 값을지정할 수 있다.

main에서 사용하기

et\_id = findViewById(R.id.*et\_id*);  
//아이디를 통해 불러옴  
btn\_test = findViewById(R.id.*btn\_test*);  
  
//버튼 클릭시의 처리  
btn\_test.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 //버튼의 텍스트를 edittext 의 글씨를 가져와서 버튼의 글자로 지정  
 btn\_test.setText(et\_id.getText());  
  
 //edittext창의 글자를 바꿈  
 et\_id.setText("입력됨");  
 }  
});

intent – 화면전환

activity = 화면전환

a라는 activity에서 b라는 activity로 이동.

mainactivity폴더에 subactivity를 만든다

버튼을 누르면 이동 하는 예제

btn\_move.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(MainActivity.this,SubActivity.class);  
 startActivity(intent);//액티비티 이동  
  
 }  
});

Intent 를 생성할 때 첫 인자로 현재 activity 두번째 인자로 이동할 activity를 선택한다.

다음 화면으로 데이터를 넘길 때

intent.putExtra("str",str);

key 와 value형태로 putExtra를 통해 넣는다

받는쪽

Intent intent = getIntent();  
String str = intent.getStringExtra("str");  
tv\_text = findViewById(R.id.*tv\_text*);  
tv\_text.setText(str);

getIntent로 intent를 통한 화면전환시 intent를 받을 수 있고,

getStringExtra등 해당 자료형을 key를 지정하여 받을 수 있다.

ImageView 와 Toast 메시지

<ImageView  
 android:layout\_width="100dp"  
 android:layout\_height="100dp"  
 android:src="@mipmap/ic\_launcher"  
 android:id="@+id/test"  
 />

이미지 뷰를 생성하고 src를 통해 이미지의 주소를 작성한다. Mipmap은 아이콘 등의 이미지가 저장되는 기본 위치이다.

이미지 클릭시 Toast 메시지 출력

test = findViewById(R.id.*test*);  
test.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Toast.*makeText*(getApplicationContext(), "배고파", Toast.*LENGTH\_SHORT*).show();  
 }  
});

첫 인자는 현재 activity화면을 의미하고 두번째는 표시할 텍스트, 세번째는 표시될 길이이다.

.show()를 통해 메시지를 출력한다.

정렬하기

gravity속성을 통해 정렬한다.

android:gravity는 위젯자체로의 정렬 값을 지정

android:layout\_gravity는 위젯자체가 아닌 위젯을 감싸주고있는 레이아웃의 정렬 값을 지정

리니어 레이아웃으로 감싼 뒤 해당 레이아웃을 정렬하는 방식사용.

<LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:gravity="center"  
 >  
  
 <ImageView  
 android:layout\_width="100dp"  
 android:layout\_height="100dp"  
 android:src="@mipmap/ic\_launcher"  
 android:id="@+id/test"  
 />  
</LinearLayout>

ListView

리스트 뷰 선언 자체는 간단하지만 사용할 activity에서 만져줘야함

//리스트 뷰를 가져온다  
list = findViewById(R.id.*list*);  
  
//리스트 뷰에 넣을 목록을 생성  
List<String> data = new ArrayList<>();  
  
//리스트 뷰와 리스트 를 연결해주는 어댑터  
//첫 인자로 해당 acitivity 두번째 인자로 ui, 세번째 인자로 들어갈 항목이다  
ArrayAdapter<String> adapter = new ArrayAdapter<>(this, android.R.layout.*simple\_list\_item\_1*,data);  
  
//리스트 뷰에 adapter를 설정한다.  
list.setAdapter(adapter);  
  
data.add("갸아아앙");  
data.add("햐아아악");  
data.add("먀아아아악");  
  
//adapter에 설정을 저장한다.  
adapter.notifyDataSetChanged();

String[] items = {"망고쥬스","토마토쥬스","포도쥬스"};  
ListAdapter adapter = new ArrayAdapter<String>(this, android.R.layout.*simple\_list\_item\_1*,items);  
ListView listView = (ListView)findViewById(R.id.*listview*);  
listView.setAdapter(adapter);

이와 같은 방식도 가능하다.

Navigation menu bar

프로젝트 생성시

Navigation drawer activity를 선택한다

버전의 차이로 예제처럼 하려면

implements NavigationView.OnNavigationItemSelectedListener

를 추가한뒤 메소드를 상속받고

@Override  
public boolean onOptionsItemSelected(@NonNull MenuItem item) {  
 return super.onOptionsItemSelected(item);  
}

를 추가한다.

SharedPrefernces

데이터를 저장할 때 사용

앱이 삭제될 때 같이 삭제됨

EditText에 입력한 값을 저장하여 앱을 다시 실행할 때 마지막으로 입력되있던 값을 EditText에 설정하는 경우

@Override  
protected void onDestroy() {  
 super.onDestroy();  
 SharedPreferences sharedPreferences = getSharedPreferences(shared,0);  
 SharedPreferences.Editor editor = sharedPreferences.edit();  
 String value = et\_save.getText().toString();  
 editor.putString("val",value);  
 editor.commit();  
}

onCreate와 반대로 onDestroy는 앱의 종료시 동작

getSharedPreferences 첫번째 인자로 생성할 sharedpreference의 이름, 모드

에디터를 통해 값을 넣는다.

에디터 생성후 에디터에 키,값 의 형태로 값을 넣고

commit하여 상태를 저장한다.

꺼내는 경우

onCreate에서 실행될 때 설정할 수 있다.

SharedPreferences sharedPreferences = getSharedPreferences(shared,0);  
String value = sharedPreferences.getString("val","");  
et\_save.setText(value);

Sharedpreference를 불러온 뒤 key를 통해 값을 꺼낸다.

Webview

<WebView  
 android:id="@+id/webView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
</WebView>

웹뷰를 지정한 뒤 main에서 나머지를 설정한다

protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 webView = findViewById(R.id.*webView*);  
 //자바스크립트 허용  
 webView.getSettings().setJavaScriptEnabled(true);  
 //불러올 url설정  
 webView.loadUrl(url);  
 //크롬 환경에 맞춰주는 설정  
 webView.setWebChromeClient(new WebChromeClient());  
  
 //일반 웹뷰 설정  
 webView.setWebViewClient(new WebViewClientClass());  
}  
  
@Override  
public boolean onKeyDown(int keyCode, KeyEvent event) {  
 //뒤로가기가 가능한 경우 뒤로가기.  
 if((keyCode == KeyEvent.*KEYCODE\_BACK*)&&(webView.canGoBack())){  
 webView.goBack();  
 return true;  
 }  
 return super.onKeyDown(keyCode, event);  
}  
//해당 함수 설정  
private class WebViewClientClass extends WebViewClient {  
 @Override  
 public boolean shouldOverrideUrlLoading(WebView view, String url) {  
 //해당 뷰에 url을 넣는다.  
 view.loadUrl(url);  
 return true;  
 }  
}

이후 manifest파일을 수정한다

Manifest 태그와 application 태그사이에

<uses-permission android:name="android.permission.INTERNET"/>

를 추가하여 인터넷 사용을 허가하고

Application태그 안에

android:usesCleartextTraffic="true"

를 넣는다.

Navigation menu bar custom

생성시 empty로 생성

레이아웃을 Drawer로 수정한다.

아이디를 지정한 뒤 안에 LinearLayout을 만들어 그 안에 내용을 채운다.

LinearLayout과 DrawerLayout사이에

Include layout으로 이하에서 생성할 레이아웃의 파일명을 넣는다.

<androidx.drawerlayout.widget.DrawerLayout 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:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity"  
 android:id="@+id/drawer\_layout"  
 >  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <Button  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="열림"  
 android:id="@+id/btn\_open"/>  
 </LinearLayout>  
 <include layout = "@layout/activity\_drawer"/>  
  
</androidx.drawerlayout.widget.DrawerLayout>

Include는 지금 activity위에 해당 레이아웃을 포함시키겠다는 의미이다.

즉 해당 activity위에 다른 화면을 사용하겠다는 의미로 볼 수 있다.

이후 layout폴더에 레이아웃을 하나 추가한다.

전체 화면을 덮지 않도록 width를 조절한다.

왼쪽 정렬을 위해 latout\_gravity를 start로 지정하고, 아이디를 지정한다.

이후 내용을 채운다.

메인 파일에서 메인 화면인 drawer레이아웃의 아이디를 가져온다.

View 형태로 겹쳐질 레이아웃의 아이디를 가져온다.

private DrawerLayout drawerLayout;  
private View drawerview;

drawerLayout = findViewById(R.id.*drawer\_layout*);  
drawerview = findViewById(R.id.*drawer*);

버튼이 눌렸을 때 현재 레이아웃(drawerlayout)에 특정 레이아웃(drawerview)을 불러오기

btn\_open.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 drawerLayout.openDrawer(drawerview);  
 }  
});

버튼이 눌렸을 때 레이아웃 닫기

Button btn\_close = findViewById(R.id.*btn\_close*);  
btn\_close.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 drawerLayout.closeDrawers();  
 }  
});

drawerLayout.setDrawerListener(listner);  
drawerview.setOnTouchListener(new View.OnTouchListener() {  
 @Override  
 public boolean onTouch(View v, MotionEvent event) {  
 return false;  
 }  
});

레이아웃 상태값을 받아오는 리스너?와 오버레이된 레이아웃에 터치이벤트를 처리하는 리스너이다.

위의 setDrawerListtner는 따로 함수를 정의해야 하는데

DrawerLayout.DrawerListener listner = new DrawerLayout.DrawerListener() {  
 @Override  
 public void onDrawerSlide(@NonNull View drawerView, float slideOffset) {  
  
 }  
  
 @Override  
 public void onDrawerOpened(@NonNull View drawerView) {  
  
 }  
  
 @Override  
 public void onDrawerClosed(@NonNull View drawerView) {  
  
 }  
  
 @Override  
 public void onDrawerStateChanged(int newState) {  
  
 }  
};

각각의 함수의 이름과 같은 기능을한다.

이 두가지의 경우 현 예제에는 불필요 하나 응용이 가능하다.

카메라 예제

추후 따로 공부 필요