Development life cycle

- Discovery phase : idea ขอบเขต,ความสามารถ
- Design Phase : detail การทำงานต่างๆ, work-flow,model
- Development and testing phase
- Development phase : ใช้งานจริง,เอกสารเครื่องมือแผนการตลาด
- Maintenance and update phase

Activity life cycle (คำสั่งต่างๆ ควบคุม layout นั้นๆ)

- Lifecycle จัดการกับสภาวะต่างๆของแอปพลิเคชันได้โดยไม่ต้องเขียนโค้ดซ้ำหลายๆที่ เช่น onCreate, onStart, onResume, onPause เป็นต้น

Run project on emulator must create AVD or emulator first

ระบบ layout

- ConstraintLayout(แบบที่ทำอยู่คือลากมุม4มุม)
- LinearLayout

setContentView() เพื่อกำหนดว่าต้องการเขียนโปรแกรมเพื่อควบคุมหน้าจอใด โดยจะทำการกำหนดไว้ในส่วนของฟังก์ชัน onCreate() เนื่องจากฟังก์ชัน onCreate() จะสั่งให้ทำงานเป็นลำดับแรกโดยอัตโนมัติ

```
class MainActivity : AppCompatActivity() {
    var btnShow : Button? = null
    var txtShow : TextView? = null
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        init()
        btnShow!!.setOnClickListener{ view->
            txtShow!!.text="Hello Android-kotlin"
            Toast.makeText( context: this, text: "Hello", Toast.LENGTH_LONG).show()
        }
    }
    fun init(){
        btnShow = findViewById(R.id.btnShow)
        txtShow = findViewById(R.id.txtShow)
    }
}
```

- ประกาศตัวแปรที่จะใช้ในการเชื่อมกับ widget ที่หน้าจอเลย์เอาท์
- สร้างฟังก์ชัน init() เพื่อทำการเชื่อมตัวแปรกับ widget ในเลย์เอาท์
- แสดงข้อความเตือนแบบ Toast ด้วยข้อความปรากฏที่หน้าจอ

การเชื่อมหลาย ๆ เลย์เอาท์เข้าด้วยกัน : จะต้องทำการสร้างคลาสที่ควบคุมการทำงานแต่ละหน้าให้เรียบร้อยก่อน

- กดปุ่มแล้วไปยังหน้าถัดไป มี2คำสั่ง 1) Intent(หน้าต่างๆ) คือใช้เชื่อมไปหน้าต่างๆ อันนี้เชื่อมไป CalGrade.kotlin 2)startActivity กดปุ่มแล้วโหลดหน้านี้ขึ้นมาได้

```
buttonGrade!!.setOnClickListener {  it: View!
    var intent = Intent( packageContext: this, CalGrade::class.java)
    startActivity(intent)
}
```

การเพิ่มหน้าจอต่างๆเข้าสู้app :

- แก้ที่ manifests ปรับแก้ไฟล์ AndroidManifest.xml(จัดการขอสิทธิ์ต่างๆ,จัดการเรื่องการแนะนำให้รู้จักกับ resourceต่างๆ)
- เอาไฟล์ kotlin ที่ต้องการเป็นหน้าหลักให้ใส่ activity tag อันแรก หากต้องการมีหน้าเพิ่มอีกก็เพิ่ม activity tag อันใหม่เลย

```
<activity
    android:name=".MainMenu"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

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

    <meta-data
        android:name="android.app.lib_name"
        android:value="" />
</activity>
<activity android:name=".MainActivity"></activity>
```

RecyclerView : ทำที่ Myadapter

- implementation 'androidx.recyclerview:recyclerview:1.2.0' and 'androidx.cardview:cardview:1.0.0'

```
//สร้าง RecyclerView.ViewHolder ก่อน
//สร้าง RecyclerView.Adapter ด้วยการ override 3 methods ข้างล่าง
class MyAdapter (val items: Array<String>, val imageId:Array<Int>): RecyclerView.Adapter<MyAdapter.ViewHolder>() {
    //อันนี้จะถูกเรียกเพื่อให้เราสร้าง view แต่ละตัวไว้ใน ViewHolder ซึ่งจะสร้างตามจำนวน view ที่ปรองเห็นบนหน้าจอ
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
        return ViewHolder(LayoutInflater.from(parent.context).inflate(R.layout.model,parent, attachToRoot: false))
    }
    //เพื่อบอกว่ามีข้อมูลกี่อัน
    override fun getItemCount(): Int {
        return items.size
    }
    //เอาไว้บอกว่าแสดงข้อมูลยังไง อันนี้จะถูกเรียกทุกครั้งที่ข้อมูลใหม่ scroll เข้าในจอ
    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
        holder.nameTextView.text = items.get(position)
        holder.imageIdView.setImageResource(imageId.get(position))
    }
    class ViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView) {
        internal var nameTextView : TextView = itemView.findViewById(R.id.nameTxt)
        internal var imageIdView : ImageView = itemView.findViewById(R.id.thumbnail)
    }
}
```

```
class GalleryView : AppCompatActivity() {
    var recyclerView : RecyclerV
    var gbtnMenu : Button? = nul

    var series = arrayOf(
        "Alchemy of Souls",
        "Who Rules the World",
        "Love Between Fairy and
        "SPY×FAMILY"
    )
    var arrSeries = arrayOf<Int>
        R.drawable.s4,
        R.drawable.s1,
        R.drawable.s2,
        R.drawable.s3
    )
```

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.gallery)
    recyclerView = findViewById(R.id.recyclerView)
    gbtnMenu = findViewById(R.id.btbn_menu)
    gbtnMenu!!.setOnClickListener {  it: View!
        var intent = Intent( packageContext: this,MainMenu::class.java)
        startActivity(intent)
    }
    recyclerView!!.layoutManager = LinearLayoutManager( context: this)
    var linearRadio = findViewById<RadioButton>(R.id.linear_layout_rb)
    //ตรวจสอบว่าถ้ามีการเลือก linear layout ก็ให้ทำการจัดรูปเป็น linear
    linearRadio.setOnClickListener {  it: View!
        recyclerView!!.layoutManager = LinearLayoutManager( context: this)
    }
    var gridRadio = findViewById<RadioButton>(R.id.grid_layout_rb)
    //ตรวจสอบว่าถ้ามีการเลือก grid layout ก็ให้ทำการจัดรูปเป็น grid โดยแบ่งเป็น 2 column
    gridRadio.setOnClickListener {  it: View!
        recyclerView!!.setLayoutManager(GridLayoutManager( context: this, spanCount: 2))
    }
    //ทำการนำข้อมูลที่ต้องการจัดรูปแบบใส่ลงใน adapter เพื่อจัดรูปแบบ
    val myAdapter = MyAdapter(series,arrSeries)
    //นำข้อมูลที่ผ่านการจัดรูปแบบเสร็จมาใส่ลงใน recyclerView
    recyclerView!!.adapter=myAdapter
}
```

การใช้ Fragment กับ การนำทางแบบ Bottom Navigation

- แก้ชื่อเมนูไปที่ string.xml
- Empty be 'this', botton be'**root**'

```
recyclerView = root.findViewById(R.id.recyclerView) as RecyclerView

recyclerView!!.layoutManager = LinearLayoutManager(root.context)

var linearRadio = root.findViewById(R.id.linear_layout_rb) as RadioButton

linearRadio.setOnClickListener {
    recyclerView!!.layoutManager = LinearLayoutManager(root.context) }

var gridRadio = root.findViewById<RadioButton>(R.id.grid_layout_rb)
gridRadio.setOnClickListener {
    recyclerView!!.setLayoutManager(GridLayoutManager(root.context, 2)) }

val myAdapter = MyAdapter(foods,arrImg)
recyclerView!!.adapter = myAdapter
```

การเพิ่มเมนูทดลองเพิ่มเมนู

1. เพิ่ม tag string ที่ string.xml
2. เพิ่ม layout
3. Create new packet
4. Create kotlin(NameFragment)
5. เพิ่ม tag fragment ใน mobile_navigation.xml
6. เพิ่ม tag item ใน bottom_nav_menu.xml
7. Create NewsViewModel.kt

```
class NewsViewModel : ViewModel()
```

8. เพิ่มคำสั่งใน NameFragment

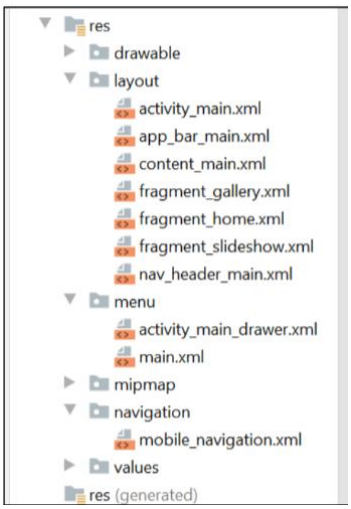
```
class NewsFragment : Fragment() {
    private var _binding: FragmentNewsBinding? = null
    // This property is only valid between onCreateView and
    // onDestroyView.
    private val binding get() = _binding!!
    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View {
        val newsViewModel =
            ViewModelProvider( owner: this).get(NewsViewModel::class.java)
        _binding = FragmentNewsBinding.inflate(inflater, container,  attachToParent: false)
        val root: View = binding.root
        /*val textView: TextView = binding.textHome
        homeViewModel.text.observe(viewLifecycleOwner) {
            textView.text = it
        }*/
        return root
    }
    override fun onDestroyView() {
        super.onDestroyView()
        _binding = null
    }
```

9. เพิ่มเมนูลงใน MainActivity

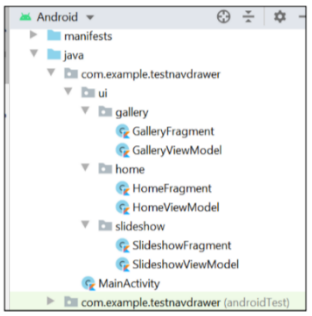
```
val appBarConfiguration = AppBarConfiguration(
    setOf(
        R.id.navigation_home, R.id.navigation_dashboard,
        R.id.navigation_news,R.id.navigation_notifications
    )
)
```

Navigation Drawer

ส่วนของ res



ส่วนที่ 2 คลาสต่าง ๆ ที่โปรแกรมสร้างขึ้น



-Fragment จะทำการเรียกใช้ ViewModel เพื่อกำหนดรายละเอียดส่วนต่าง ๆ ใน Fragment สำหรับไฟล์ MainActivity จะทำการเรียกใช้ทั้ง 3Fragment มาใช้งาน

การทำงานของแต่ละไฟล์

1. **Layout activity_main :**รวบรวมองค์ประกอบย่อยๆ

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

    <include
        layout="@layout/app_bar_main"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <com.google.android.material.navigation.NavigationView
        android:id="@+id/nav_view"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:fitsSystemWindows="true"
        app:headerLayout="@layout/nav_header_main"
        app:menu="@menu/activity_main_drawer" />
</androidx.drawerlayout.widget.DrawerLayout>
```

2. Layout nav_header_main : ทำหน้าที่กำหนดรูปแบบในส่วนของ header ในส่วนของ Navigation Drawer
3. Layout app_bar_main : นี้จะเป็นตัวกำหนดโครงสร้างของแต่ละหน้าจอ
4. Layout content_main : สร้างพื้นที่ fragment เพื่อใช้ในการวางข้อมูลในแต่ละหน้า โดยแต่ละ fragment มีคุณสมบัติอย่างไรจะอยู่ในส่วนของmobile_navigation
5. navigation/mobile navigation : หน้าเมนูต่างๆ

```
<fragment
    android:id="@+id/nav_home"
    android:name="com.example.testnavdrawer.ui.home.HomeFragment"
    android:label="Home"
    tools:layout="@layout/fragment_home" />
```

6. menu/activity_main_drawer สร้างเมนูในDrawer

```
<item
    android:id="@+id/nav_home"
    android:icon="@drawable/ic_menu_camera"
    android:title="Home" />
```

7. menu/main กำหนดจุดที่อยู่มุมขวาบน
8. MainActivity.kt เชื่อม layout activity_main and fragment

```
appBarConfiguration = AppBarConfiguration(setOf(
        R.id.nav_home, R.id.nav_gallery, R.id.nav_slideshow), drawerLayout)
```

ขั้นตอนการปรับแต่งเทมเพลต

1. เพิ่ม tag string ที่ string.xml
2. Create layout(fragment_name)
3. เพิ่ม tag item ใน activity_main_drawer (เพิ่มเมนูในหน้า)

4. Create new packet
5. Create kotlin

1. NameFragment

```
class ContactFragment : Fragment() {
    private var _binding: FragmentContactBinding? = null
    // This property is only valid between onCreateView and
    // onDestroyView.
    private val binding get() = _binding!!
    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View {
        val contactViewModel =
            ViewModelProvider( owner: this).get(ContacrViewModel::class.java)

        _binding = FragmentContactBinding.inflate(inflater, container, attachToParent: false)
        val root: View = binding.root
        val textView: TextView = binding.txtContact
        contactViewModel.text.observe(viewLifecycleOwner) { it: String!
            textView.text = it
        }
        return root
    }
    override fun onDestroyView() {
        super.onDestroyView()
        _binding = null
    }
```

2. NameViewModel

```
class ContacrViewModel: ViewModel() {
    private val _text = MutableLiveData<String>().apply{ this: MutableLiveData<String>
        value = "This is cintact Fragment"
    }
    val text: LiveData<String> = _text
}
```

6. เพิ่ม tag fragment ที่ mobile_navigation(เชื่อม layout ในหน้าเมนู)
7. เพิ่มเมนูลงใน MainActivity

การนำค่าจาก EditText มาใช้

```
class HomeFragment : Fragment() {

    private var _binding: FragmentHomeBinding? = null
    // This property is only valid between onCreateView and
    // onDestroyView.
    private val binding get() = _binding!!
    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View {
        val homeViewModel =
            ViewModelProvider( owner: this).get(HomeViewModel::class.java)

        _binding = FragmentHomeBinding.inflate(inflater, container, attachToParent: false)
        val root: View = binding.root
        /*val textView: TextView = binding.textHome
        homeViewModel.text.observe(viewLifecycleOwner) {
            textView.text = it
        }*/
```

กำหนดขนาด widget

1. wrap_content มีขนาดเท่ากับเนื้อหาที่แสดงอยู่
2. match_parent มีขนาดเท่ากับตัวบรรจุที่ตัวเองอาศัยอยู่
3. แบบกำหนดขนาดในหน่วย dp

ระบบเลย์เอาท์ของแอนดรอยด์

1. ConstraintLayout : top automatic
2. LinearLayout
   a. แนวตั้ง (LinearLayout:vertical)
   b. แนวนอน (LinearLayout:horizontal)

```
// ประกาศตัวแปรที่ต้องการใช้
var uName : EditText? = null
var bYear : EditText? = null
var mess : TextView? = null
var compute : Button? = null
var clear : Button? = null
// ทำการเชื่อมตัวแปรกับเลย์เอาท์
uName = root.findViewById(R.id.edname)
bYear = root.findViewById(R.id.edyear)
mess = root.findViewById(R.id.txtAns)
compute = root.findViewById(R.id.btnCompute)
clear = root.findViewById(R.id.btnClear)

    compute.setOnClickListener { it: View!
        //นำค่าจาก EditText มาใส่ตัวแปร String
        var youName = uName.text.toString()
        //นำค่าจาก EditText มาใส่ตัวแปร Int เพื่อให้สามารถคำนวณได้
        var age : Int = bYear.text.toString().toInt()
        age = 2566-age
        // กำหนดค่าที่จะแสดงใน TextView
        mess.text="Hello "+ youName + "\nYour age is " + age
    }
    clear.setOnClickListener { it: View!

        uName.setText("") //กำหนดค่าที่จะแสดงใน EditText
        bYear.setText("")
        mess.text =""
        uName.isCursorVisible=true // กำหนดตำแหน่ง Cursor
    }

    return root
}

override fun onDestroyView() {
    super.onDestroyView()
    _binding = null
}
}
```

Model layout

```
<LinearLayout xmlns:android="http://schemas.ar
    xmlns:app="http://schemas.android.com/apk/
    android:layout_width="170dp"
    android:layout_height="190dp"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="10dp"
    android:orientation="vertical"
    android:padding="5dp">
    <ImageView
        android:id="@+id/thumbnail"
        android:layout_width="match_parent"
        android:layout_height="133dp"
        android:scaleType="centerCrop"
        app:srcCompat="@drawable/me1" />
    <TextView
        android:id="@+id/nameTxt"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="my pic"
        android:textAlignment="center"
        android:textSize="16sp" />
</LinearLayout>
```