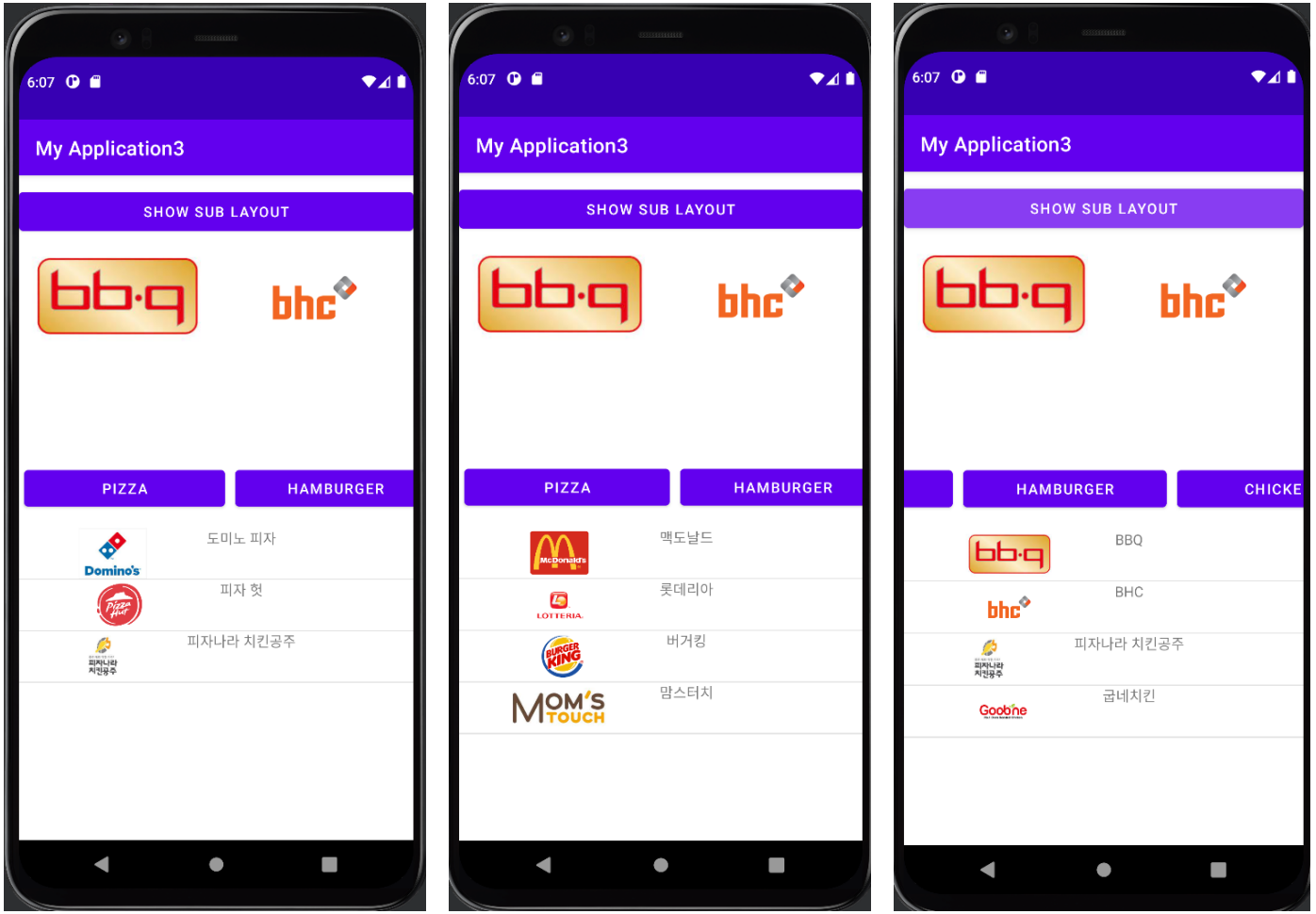
Advanced UI

Mobile App Programming
Fall, 2024

What we learn today?

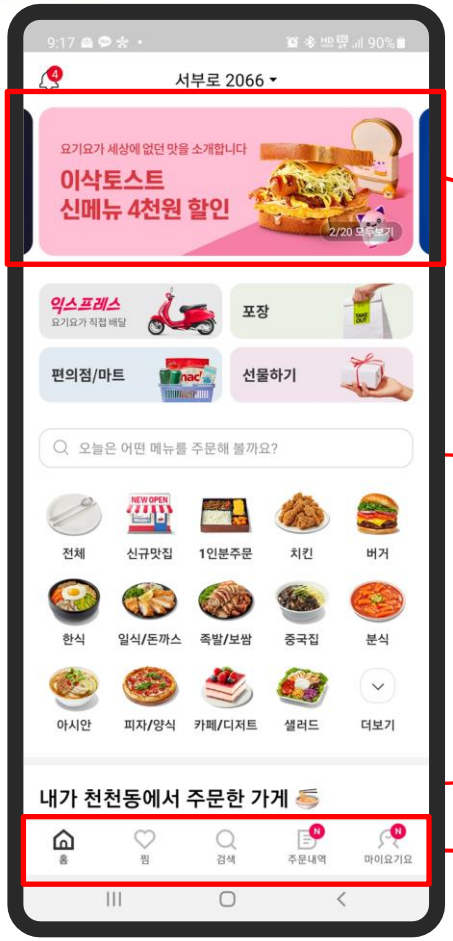
- Let's make more complicate, intelligent activity.
  - Split an **Activity** into multiple **Layouts**.
    - HorizontalScrollView
    - Inflater
  - Create and manage **ListView**.
    - Adapter

What we make today?



Multiple Layouts

Multiple Layouts



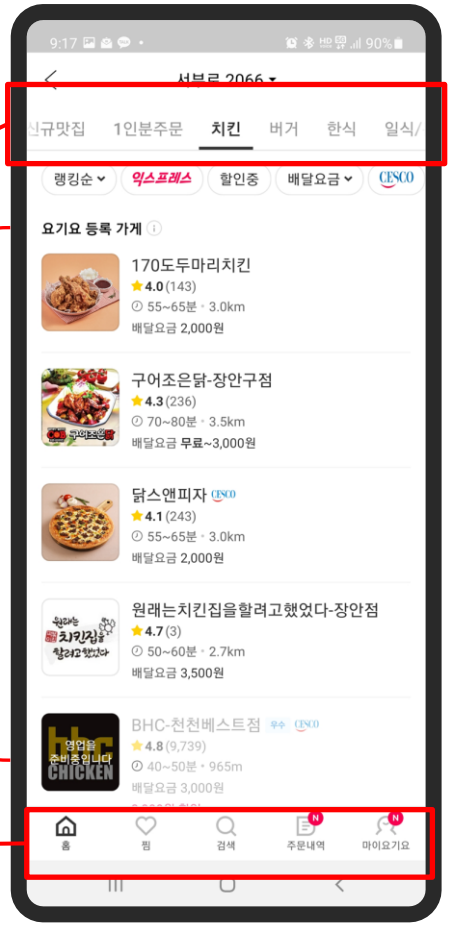
* Root element is maybe
Vertical arrangement

ScrollableView
(Maybe)

GridView

This part needs
Horizontal arrangement

ListView



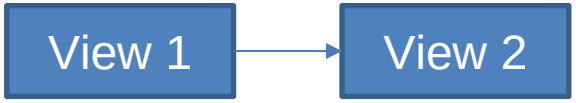
Multiple Layouts

- **Sometimes, we need more than one Layout.**
  - Sometimes, we need **partial arrangement**.
  - Some place needs linear arrangement, or some place needs scrollView.

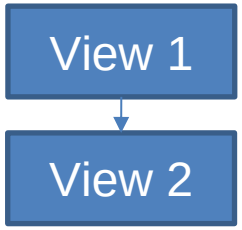
- **A layout is allowed to contain other Layouts.**
  - All Layouts are also have **constrains** and **attributes** for arrangement.
  - It actually **similar with inserting Views into a Layout**.

Constraint Layout (Default Layout)

- Similar with **Relative Layout**
- View(or ViewGroup) in constraint Layout need at least 1 constraints (x-axis, y-axis).
- Normally, we can add 4 constraints
  - Top, Bottom, Start, End
  - There are many attributes,
    - app:layout_constraint{**value(my)**}_to{**value(relative)**}Of = "{**id of relative**}"
    - Value: Bottom, Top, End, Start



app:layout_constraint**End**_to**Start**Of
="@id/view2"



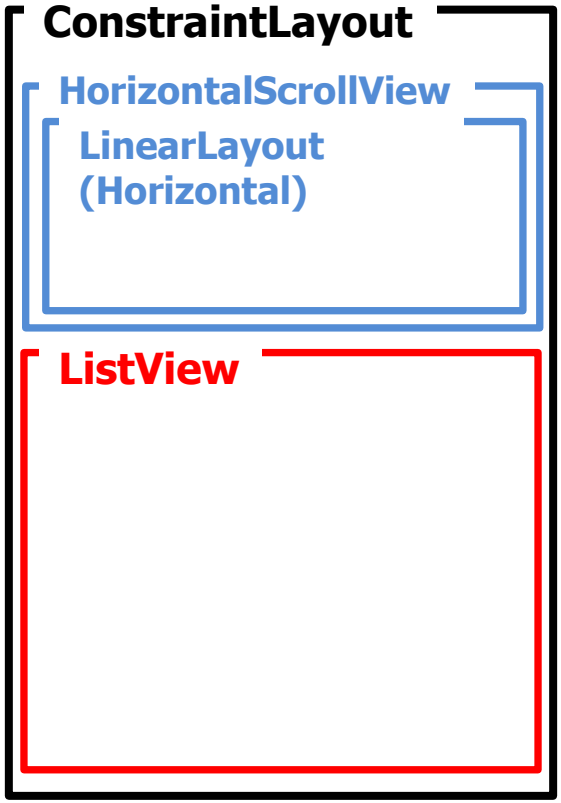
app:layout_constraint**Bottom**_to**Top**Of
="@id/view2"

HorizontalScrollView

- **Horizontally aligned** ScrollView.
- **Scrollable** but not **clickable** or **focusable**.
- **It is not a Layout!**
  - Only **1 View/Layout** can be located in this component.
  - Then, to put multiple Views on it, we should **insert Layout first**.

Ex1) Horizontal ScrollView

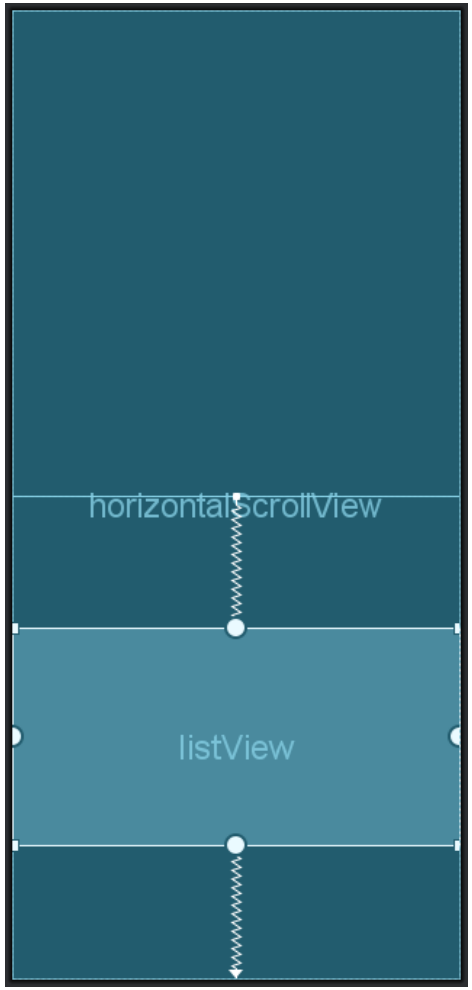
- Let's insert **HorizontalScrollView** and **LinearLayout** into **ConstraintLayout**.




```
<ConstraintLayout>
    <HorizontalScrollView>
        <LinearLayout>

            ...

        </LinearLayout>
    </HorizontalScrollView>

    <ListView>
            ...
    </ListView>
</ConstraintLayout>
```

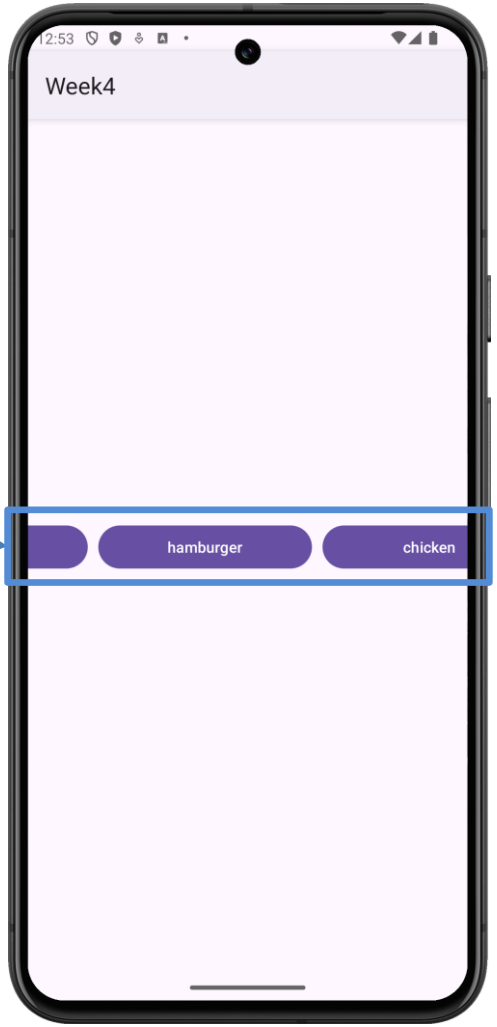
Ex1) Horizontal ScrollView

```
<?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">

    <HorizontalScrollView
        android:id="@+id/horizontalScrollView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">
    </HorizontalScrollView>

    <ListView
        android:id="@+id/listView"
        android:layout_width="match_parent"
        android:layout_height="200dp"
        app:layout_constraintTop_toBottomOf="@id/horizontalScrollView"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent">
    </ListView>

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



Ex1) Horizontal ScrollView

```
<HorizontalScrollView
    android:id="@+id/horizontalScrollView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent">

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:orientation="horizontal">

        <Button
            android:id="@+id/button1"
            android:layout_width="200dp"
            android:layout_height="wrap_content"
            android:layout_margin="5dp"
            android:text="pizza" />

        <Button
            android:id="@+id/button2"
            android:layout_width="200dp"
            android:layout_height="wrap_content"
            android:layout_margin="5dp"
            android:text="hamburger" />

        <Button
            android:id="@+id/button3"
            android:layout_width="200dp"
            android:layout_height="wrap_content"
            android:layout_margin="5dp"
            android:text="chicken" />
    </LinearLayout>
</HorizontalScrollView>
```



Inflation & LayoutInflater

How can Kotlin class use XML resource?

- **Kotlin source only can use "objectified" resources.**
- **Somehow XML resources are converted into kotlin objects to enable Kotlin access those resources.**
  - Think about "**R.id.<someView>**"!
  - We already use **objectified resources** when we initialize **View instance**.
  - Then how? who's going to do those roles?
- **Think about what "setContentView" function does.**

What does setContentView() do?

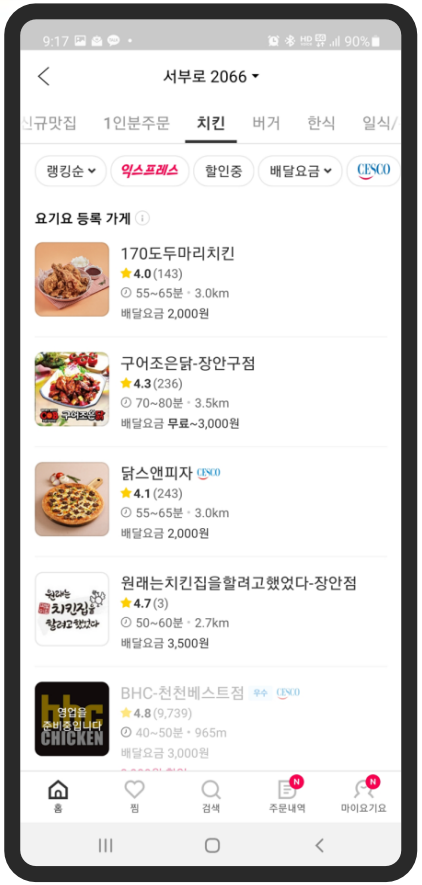
- **Android have some pairs between XML and Kotlin.**
  - To use XML contents in the Kotlin source codes, we need **objectified Views (or ViewGroup)**.
  - **setContentView()** function **convert contents** defined in XML to **Kotlin object**.

- **The process that objectify XML contents is called Inflation.**

LayoutInflater

- **Convert resources which declared in XML to view object.**
- **Commonly used for generating new View or ViewGroup in Kotlin source code, such as Adapter or Activity.**
  - We can design our custom views by using **LayoutInflater** and new .xml files.
  - Or we can use it when we **format our list** or something else.
- **LayoutInflater is almost same operation with "Inflation"**

Why we need Inflater?



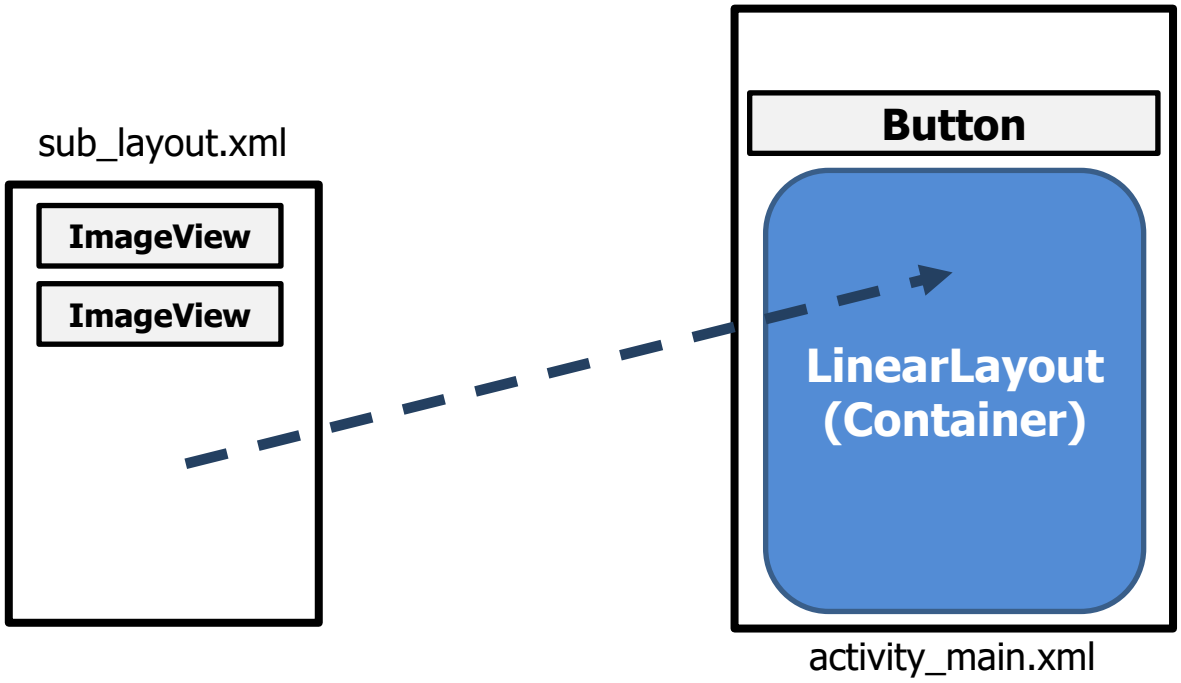
If you click category, below ListView will be changed properly.

Number of Restaurants will change according to your location.

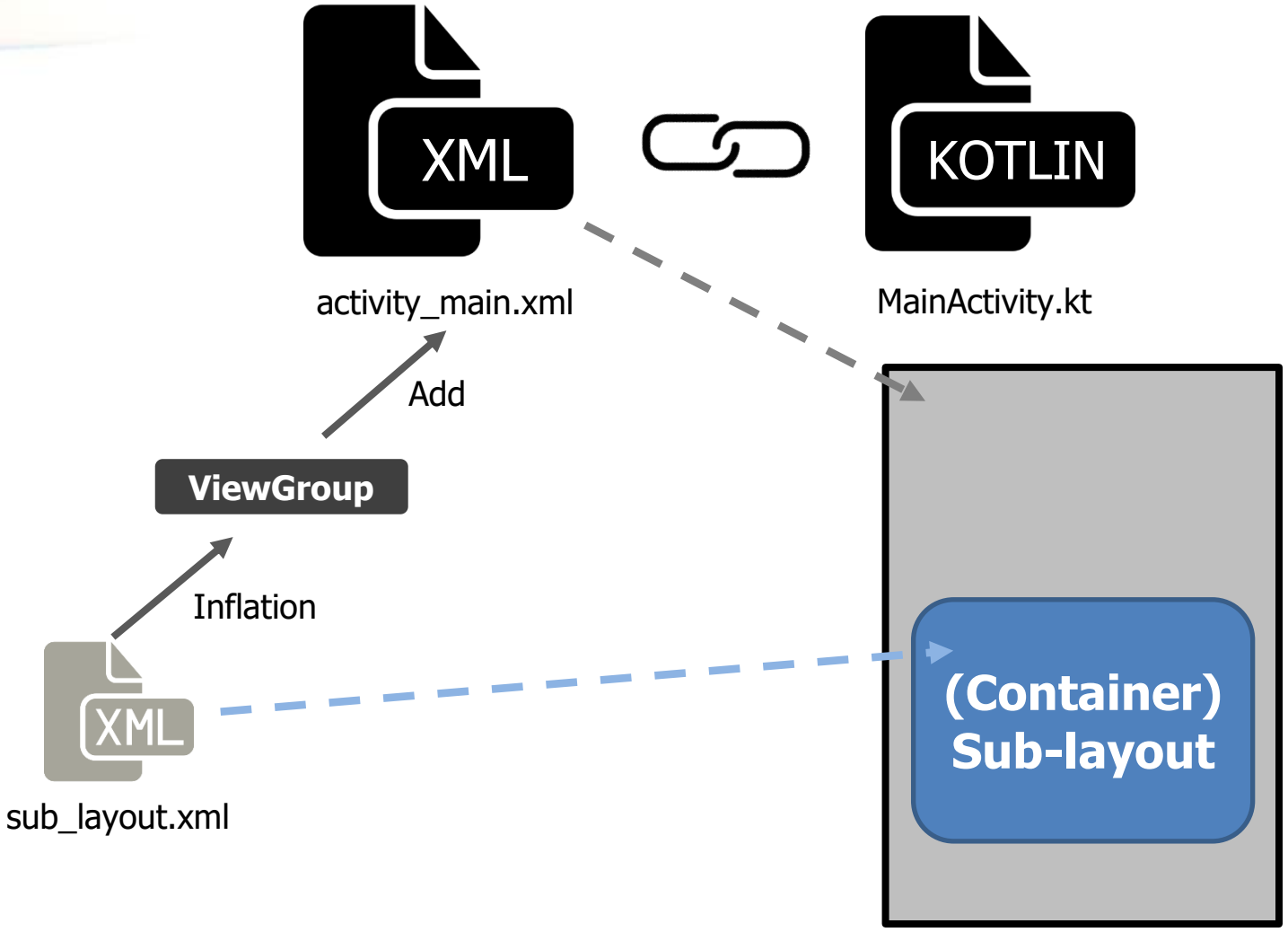
We need to change layout while runtime!

Ex2) SubLayout

- **Let's make a new view with LayoutInflater and XML.**
  - **activity_main.xml** declare your MainActivity's UI.
  - **sub_layout.xml** declare your own View.



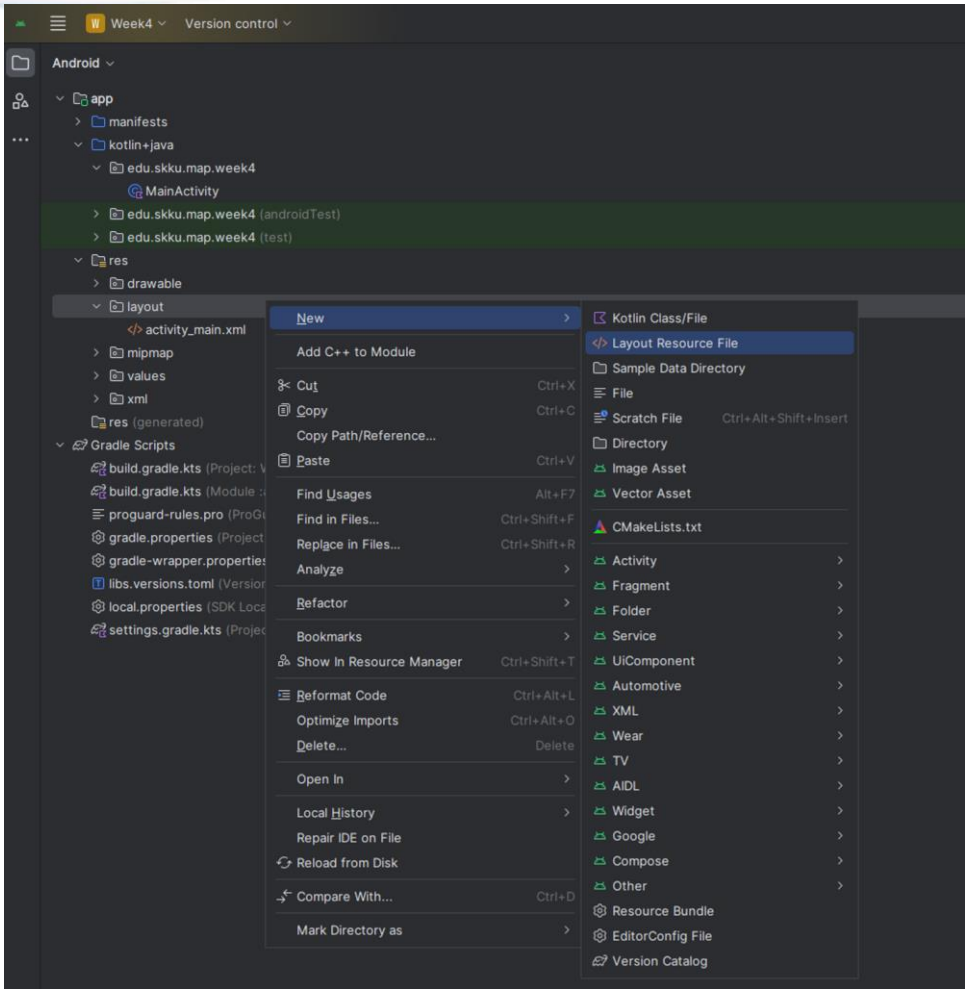
Ex2) SubLayout - LayoutInflater

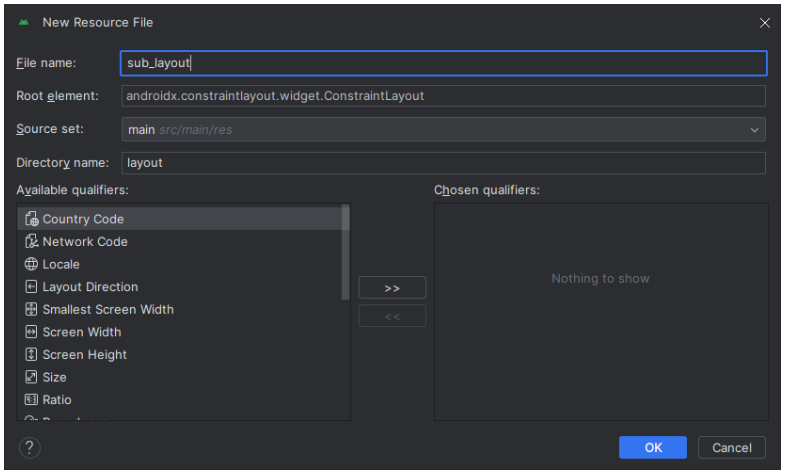


Ex2) SubLayout – activity_main.xml

```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">

    <Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toTopOf="@id/subLayout"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:text="SHOW SUB LAYOUT" />

    <LinearLayout
        android:id="@+id/subLayout"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:orientation="horizontal"
        app:layout_constraintTop_toBottomOf="@id/button"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />


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

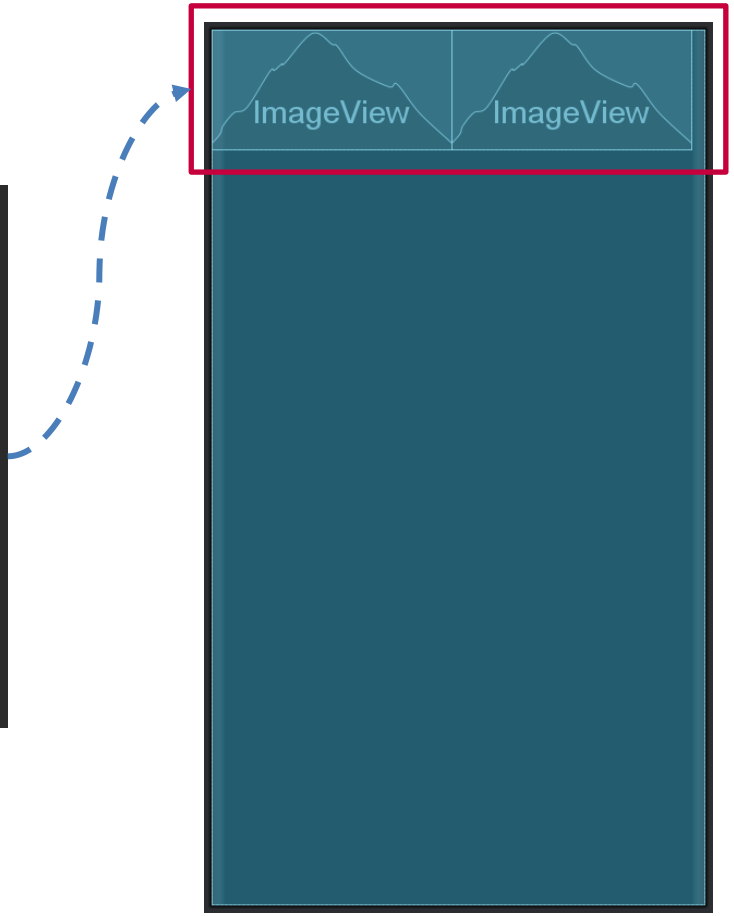
Ex2) SubLayout – Create sub_layout.xml





Ex2) SubLayout – sub_layout.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="200dp"
        android:layout_height="100dp" />

    <ImageView
        android:id="@+id/iamgeView2"
        android:layout_width="200dp"
        android:layout_height="100dp" />

</LinearLayout>
```

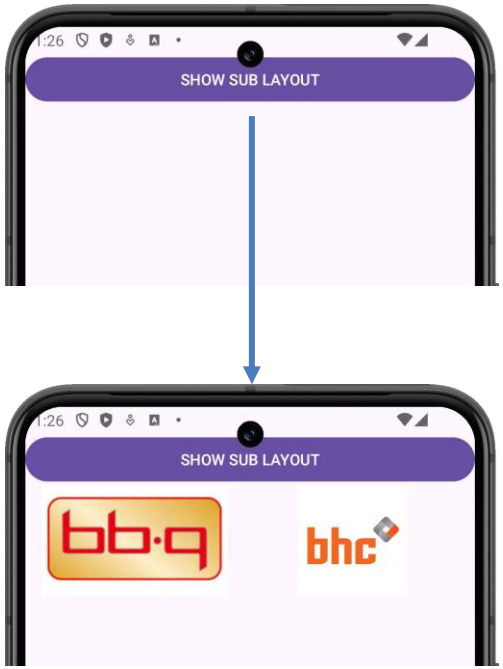


Ex2) SubLayout – MainActivity.kt

```kotlin
package edu.skku.map.week4

import android.content.Context
import android.os.Bundle
import android.view.LayoutInflater
import android.widget.Button
import android.widget.ImageView
import android.widget.LinearLayout
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v, insets ->
            val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)
            insets
        }

        var linearLayout = findViewById<LinearLayout>(R.id.subLayout)
        var button = findViewById<Button>(R.id.button)

        button.setOnClickListener {
            val layoutInflater: LayoutInflater =
                applicationContext.getSystemService(Context.LAYOUT_INFLATER_SERVICE) as LayoutInflater
            layoutInflater.inflate(R.layout.sub_layout, linearLayout, true)

            var img1 = findViewById<ImageView>(R.id.imageView)
            var img2 = findViewById<ImageView>(R.id.imageView2)

            img1.setImageResource(R.drawable.bbq)
            img2.setImageResource(R.drawable.bhc)
        }
    }
}
```

Convert container's view to sub_layout

```kotlin
// Other ways to declare constant layoutInflater instance
val layoutInflater: LayoutInflater = LayoutInflater.from(applicationContext)
```



ListView (Custom)

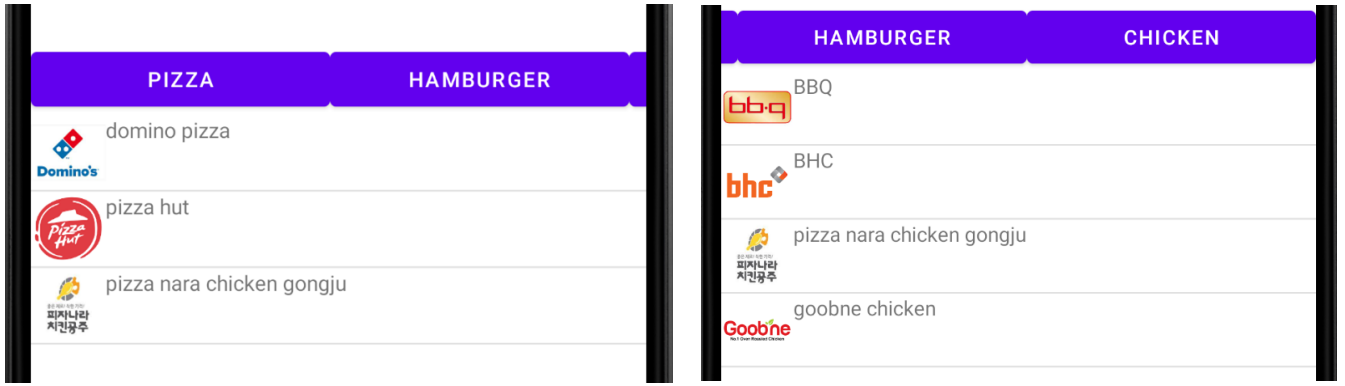
ListView

- **A ViewGroup which groups several items and display them in vertical scrollable list.**

- **The list items are automatically inserted to the ListView using Adapter.**
  - Adapter pull contents from a source (e.g. DB, ArrayList etc.)

- To make ListView, you have to make
  1) Custom Adapter → .kt file
  2) Custom List Layout → .xml file
  3) Set custom adapter to listview

Ex3) ListView (Custom)

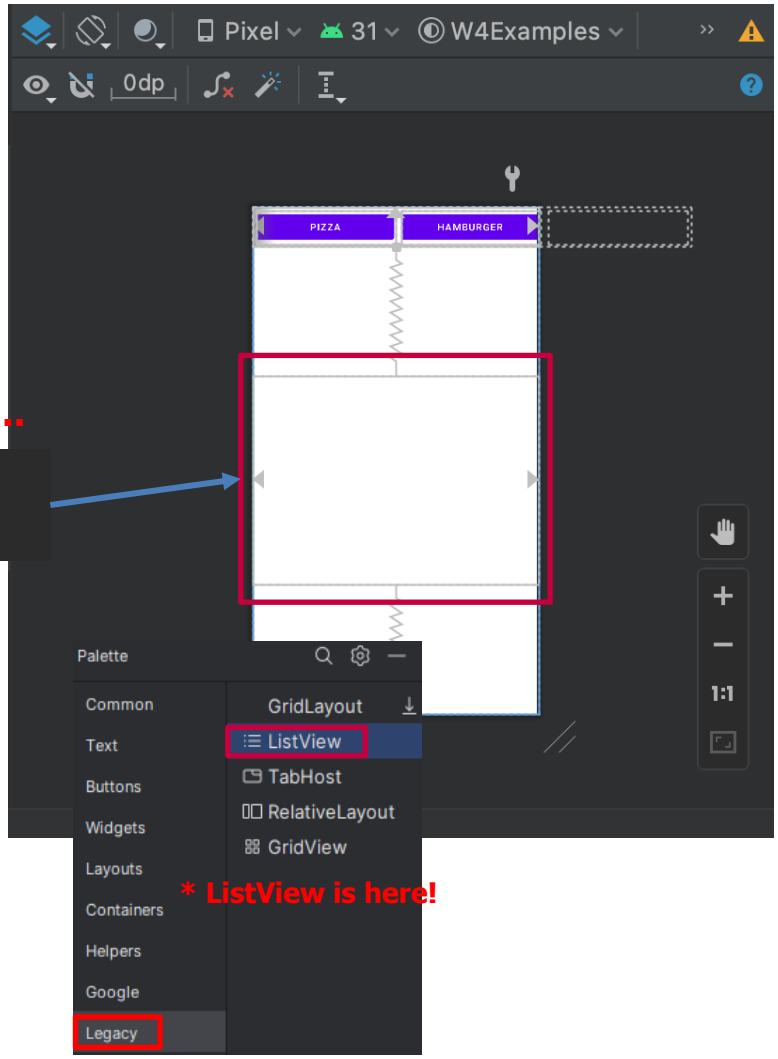


Ex3) ListView (Custom) – activity_main.xml

```
<?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">

    <HorizontalScrollView
        ...
        <LinearLayout
            ...
        </LinearLayout>
    </HorizontalScrollView>

    <ListView
        android:id="@+id/listView"
        android:layout_width="match_parent"
        android:layout_height="200dp"
        app:layout_constraintTop_toBottomOf="@id/horizontalScrollView"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />

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

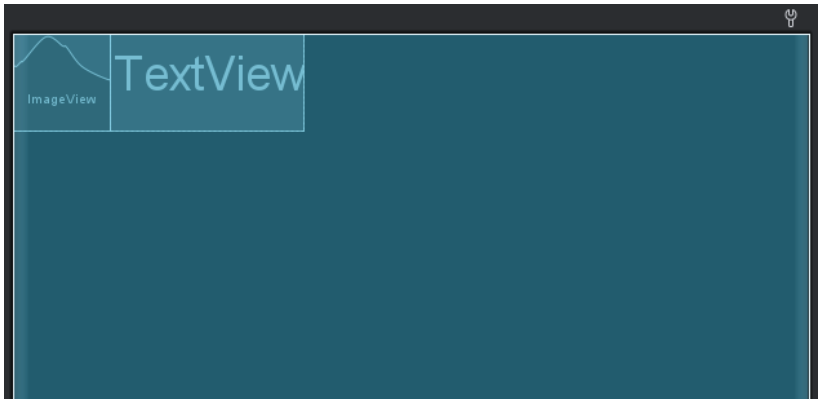
* To arrange element like..

```
app:layout_constraintTop_toTopOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent">
```



Ex3) ListView (Custom) – Create item.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">

    <ImageView
        android:id="@+id/itemImageView"
        android:layout_width="50dp"
        android:layout_height="50dp" />

    <TextView
        android:id="@+id/itemTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TextView" />

</LinearLayout>
```

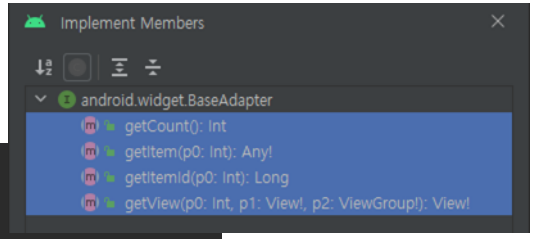


Just put ImageView and Textview in horizontal LinearLayout!

Ex3) ListView (Custom) – ListViewAdapter.kt

1) Create New Class "Restaurant"

3) Extend "BaseAdapter"




```kotlin
class Restaurant (val id: Int, val name: String)

class CustomAdapter (val context: Context, val items: ArrayList<Restaurant>): BaseAdapter(){
    override fun getCount(): Int {
        return items.size
    }

    override fun getItem(position: Int): Any {
        return items[position]
    }

    override fun getItemId(position: Int): Long {
        return 0
    }

    override fun getView(position: Int, convertView: View?, parent: ViewGroup?): View {
        // LayoutInflater is used!
        val inflater: LayoutInflater = LayoutInflater.from(context)
        val view: View = inflater.inflate(R.layout.item, null)

        var imageView = view.findViewById<ImageView>(R.id.itemImageView)
        var textView = view.findViewById<TextView>(R.id.itemTextView)

        textView.text = items[position].name
        imageView.setImageResource(items[position].id)

        return view
    }
}
```

2) Make ArrayList to save items

4) Define 3 override functions

5) Override "getView" function <IMPORTANT!!>

This part change ViewGroup to item.xml

This part generate each list element
(image, string)

Ex3) ListView (Custom) – MainActivity.kt

```
package edu.skku.map.week4

import android.os.Bundle
import android.widget.ListView
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v, insets ->
            val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)
            insets
        }

        val items = arrayListOf<Restaurant>()
        items.add(Restaurant(R.drawable.bbq, "bbq"))
        items.add(Restaurant(R.drawable.bhc, "bhc"))

        var mainList = findViewById<ListView>(R.id.listView)
        val listAdapter = CustomAdapter(this, items)

        mainList.adapter = listAdapter
    }
}
```

```
// Or… you can put elements in the declaration of the list
var items = arrayListOf(
    Restaurant(R.drawable.bbq, "bbq"),
    Restaurant(R.drawable.bhc, "bhc")
    // ...
)
```

1) Define and update ArrayList

2) Connect listview to variable

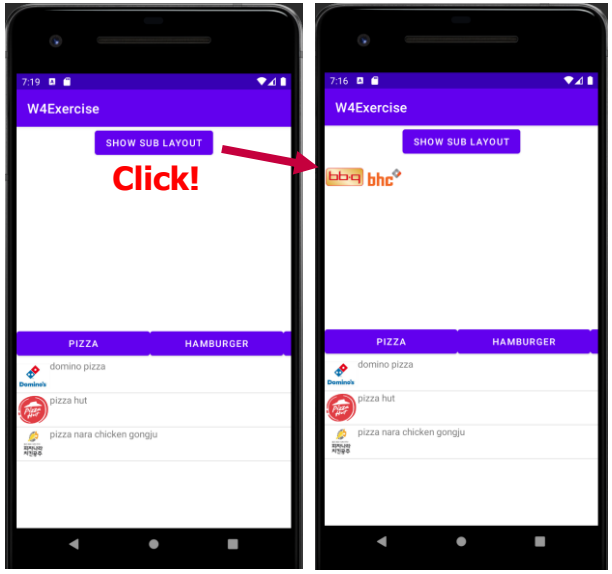
3) Generate new CustomAdaptor instance

4) Set Adapter to listview

Tip) If you call setAdapter with another adapter one more, list view will change based on new adapter (and new items).

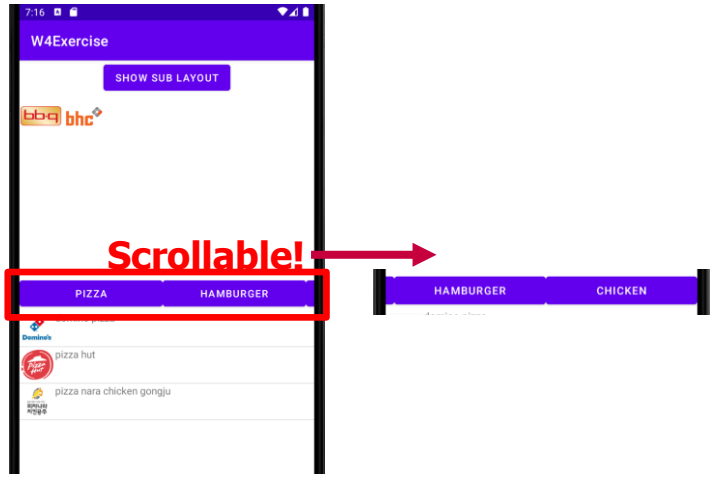
[Lab-Practice #4]

- **Let's make an application satisfying the belows:**
  - 3 Contents which we learned today.
    - **SubLayout**
    - HorizontalScrollView
    - ListView
  - SubLayout
    - Make a button with "SHOW SUB LAYOUT".
    - Make 1 LinearLayout (for Inflating Views)
    - Make sublayout.xml file
    - There are two ImageViews in sublayout.xml
    - If you click the button, LinearLayout will be changed to sublayout.xml, with 2 logos (bbq, bhc)



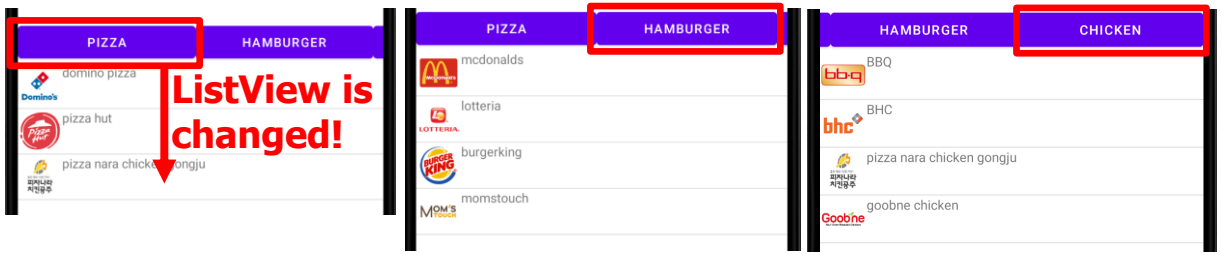
[Lab-Practice #4]

- **Let's make an application satisfying the belows:**
  - 3 Contents which we learned today.
    - Sub Layout
    - **HorizontalScrollView**
    - ListView
  - HorizontalScrollView
    - Make an HorizontalScrollView.
    - Put three Buttons "pizza", "hamburger", "chicken".
    - Call(and Set) *setOnClickListener()* functions to each button.



[Lab-Practice #4]

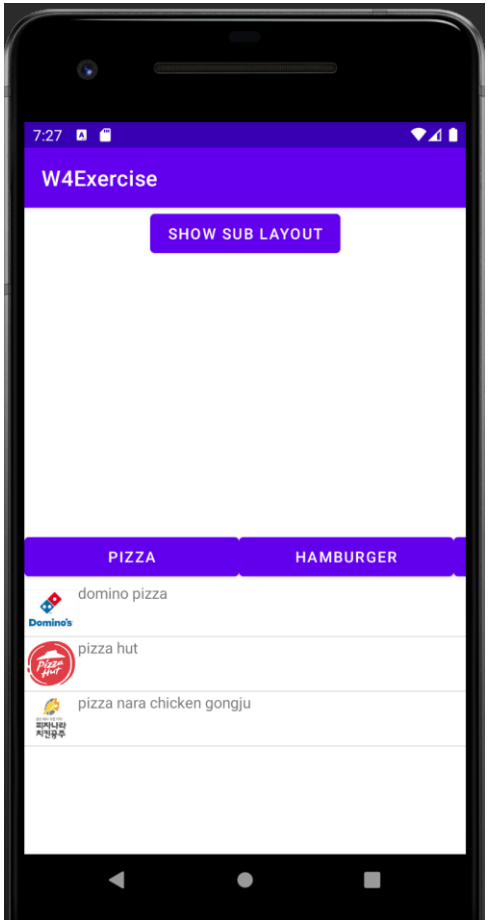
- **Let's make an application satisfying the belows:**
  - 3 Contents which we learned today.
    - Sub Layout
    - Horizontal Scroll View
    - **ListView**



  - ListView
    - **Make one ListView in activity_main.xml**
    - Make three ListViewAdapters and ListArrays.
    - When you click the button in HorizontalScrollView, change below listview.
      - Chicken – BBQ, BHC, pizza nara chicken gongju, goobne
      - Pizza – domino, pizza hut, pizza nara chicken gongju
      - Hamburger – mcdonalds, lotteria, burgerking, momstouch
    - Each element composed to its name(TextView) and logo image(ImageView) → **Use 04_images.zip**

[Lab-Practice #4]

- **Submit to ICAMPUS**
  - Extract your project to Week4_studentID.zip file.
    - (File → Export → Extract to Zip)
  - Design Issue
    - If we can check each function works well, **design is not important** in this assignment.
    - **Just make similar** to right pictures. (Size of Buttons, Images are **not important!)**



Note

- Default setting can be different by IDE version (e.g. Make New Project, onCreate function)
- Make active use of the auto-complete feature → It can help you import the package!
- Name of variables, files, classes are up to you