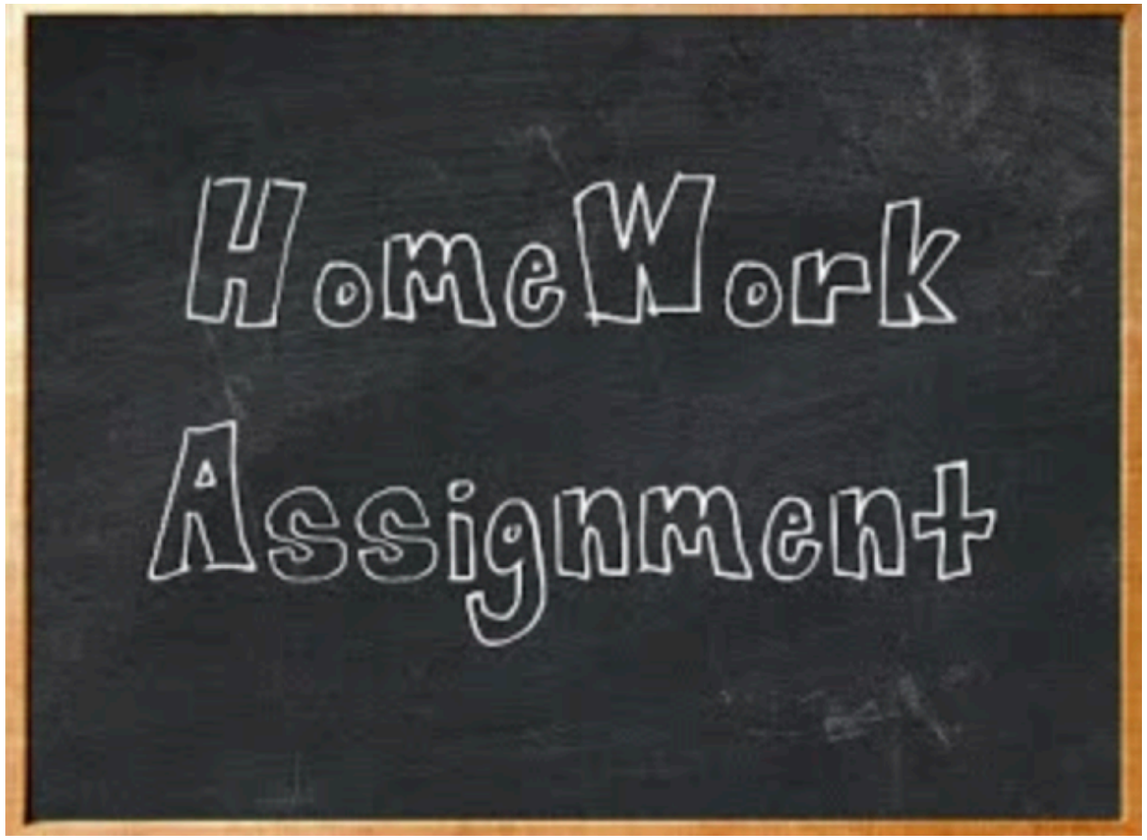
Lecture #2
Lists and Rest Resources

Mobile Applications 2020-2021

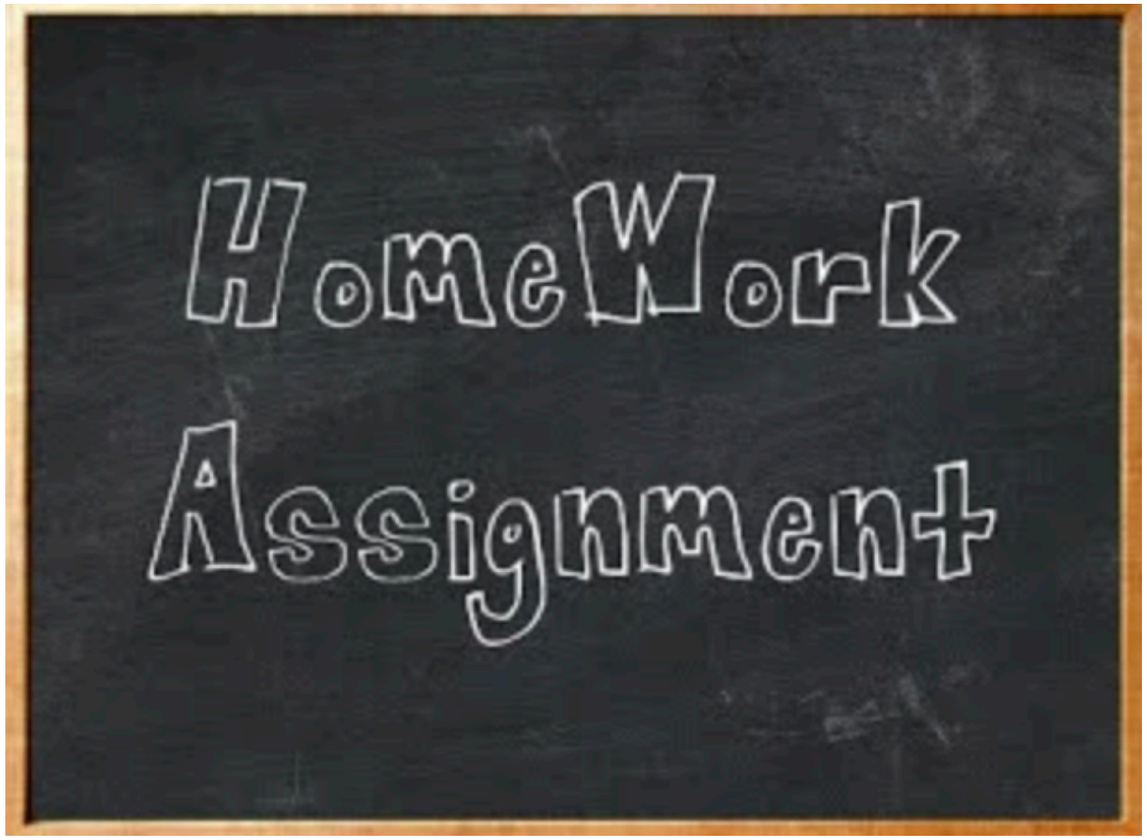
Homework assignments

- First assignment - Project details
- One CRUD project
  - UI Only
    - In Native and Non-Native
  - DB
  - NET



Homework assignments

- First assignment - Project details **Due: 2nd laboratory**
- One CRUD project
  - UI Only
    - In Native and Non-Native
      Due: 3rd and 4th laboratory
  - DB
    Due: 5th laboratory
  - NET
    Due: last laboratory



cs.ubbcluj.ro/~dan/ma/labPlan.html

CRUD Application

Native

Non-Native

CRUD Application

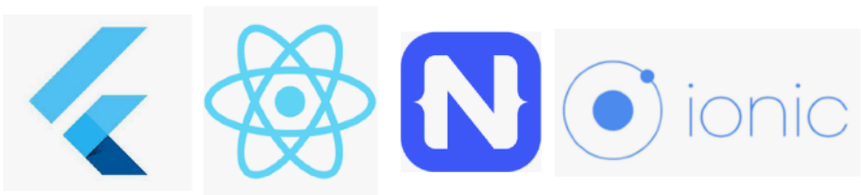
Native

Non-Native

CRUD Application

Native

Non-Native

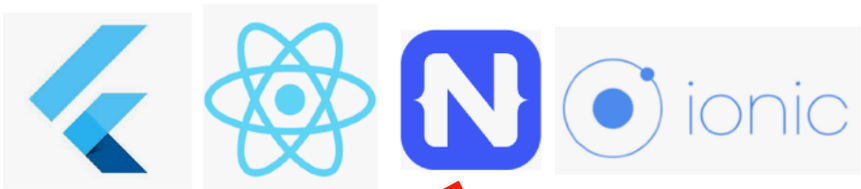


Other

CRUD Application

Native

Non-Native



Other

CRUD Application

Native

AND

Non-Native

Other

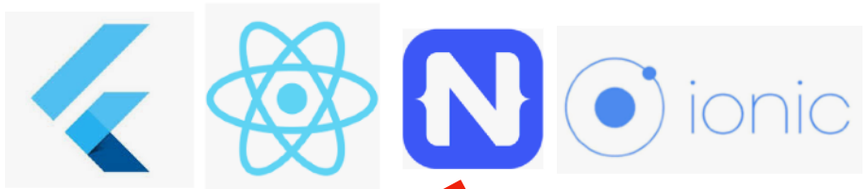
CRUD Application

Native

AND

UI Only

Non-Native

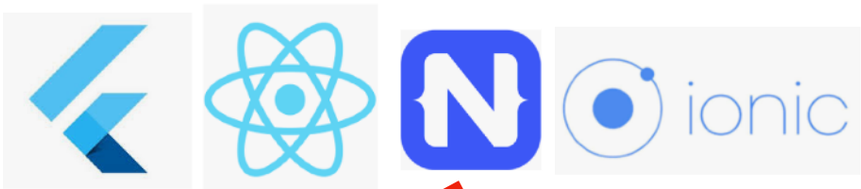


Other

CRUD Application
DB, and NET

Native

Non-Native



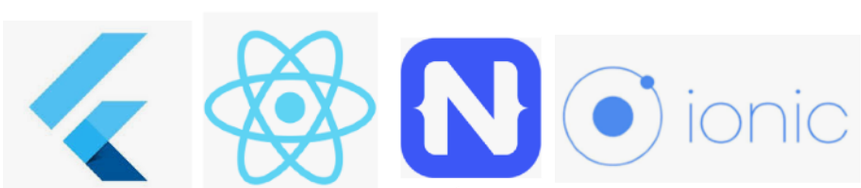
Other

CRUD Application
DB, and NET

Native

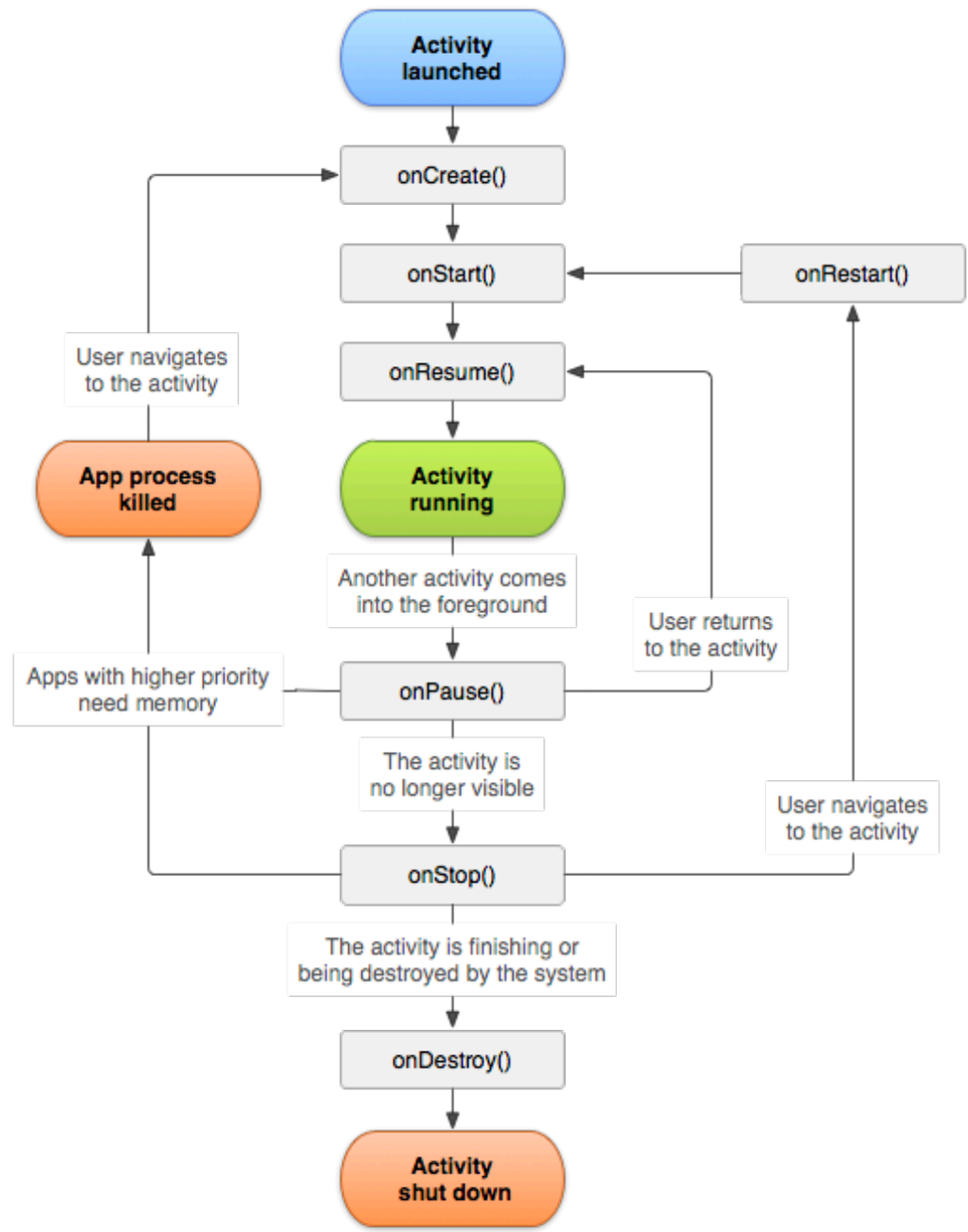
Only one

Non-Native

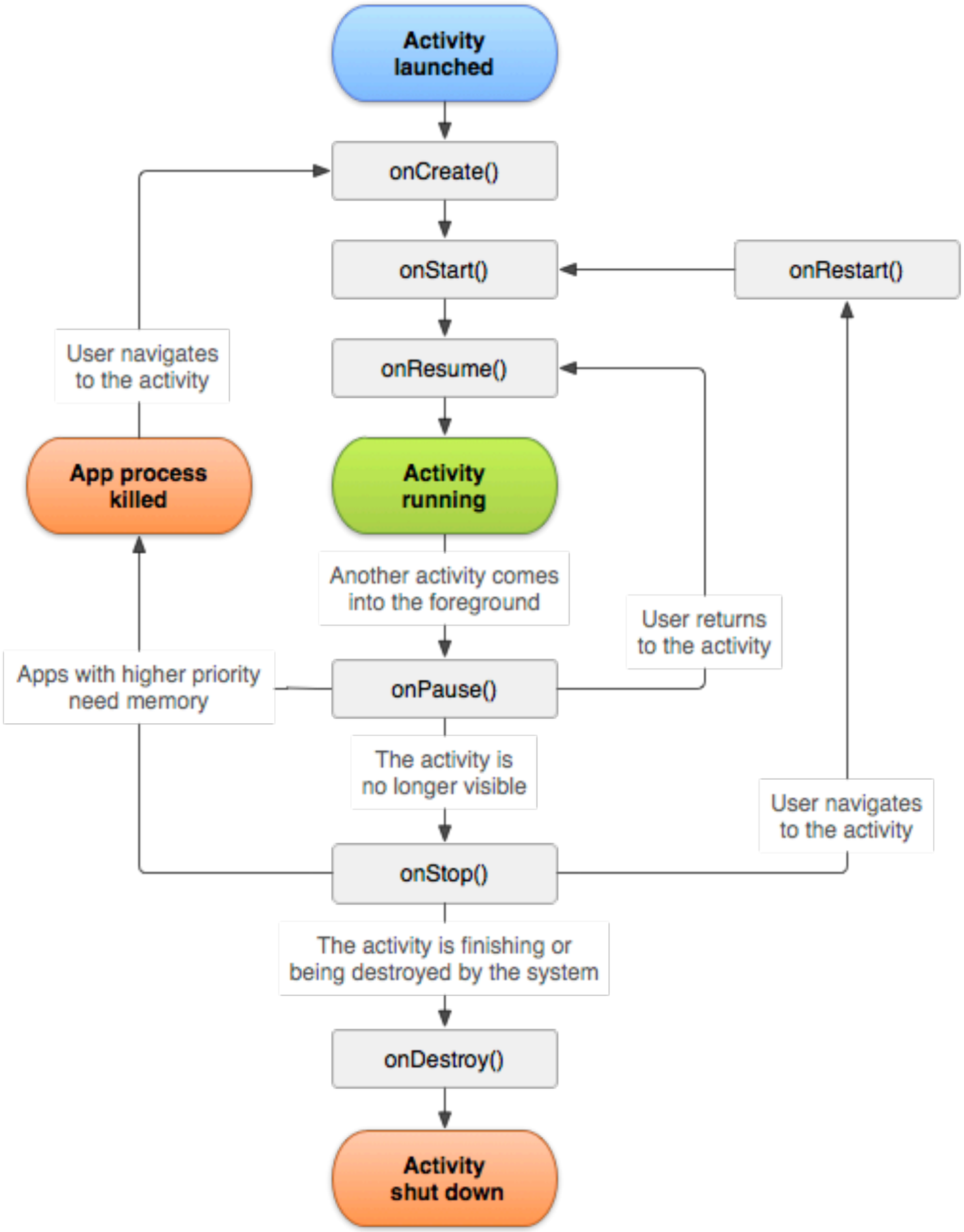


Other

Lifecycle



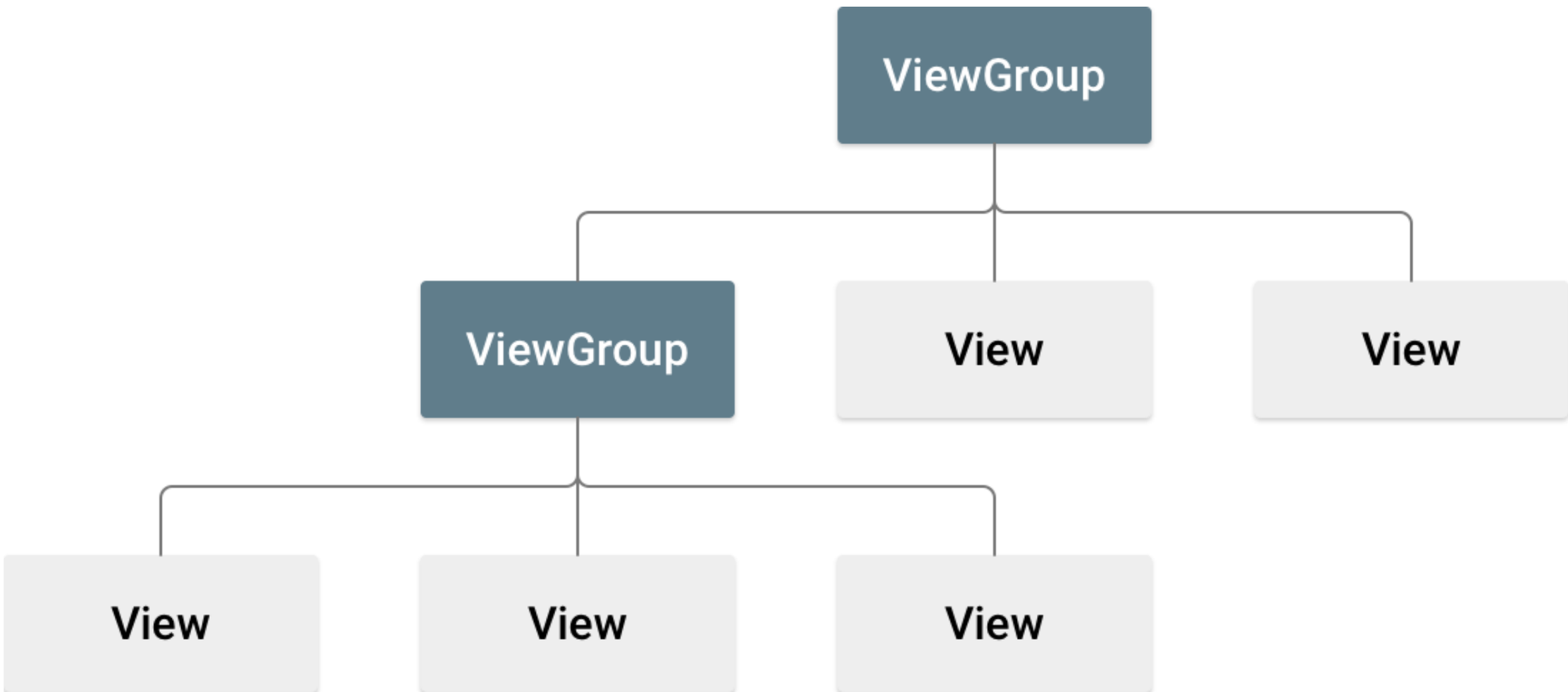
https://developer.android.com/guide/components/activities/activity-lifecycle


Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

DEMO

Layouts

https://developer.android.com/guide/topics/ui/declaring-layout



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="vertical">
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a
TextView"/>
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"/>
</LinearLayout>
```

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="vertical">
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a
TextView"/>
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"/>
</LinearLayout>
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
}
```

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="vertical">
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a
TextView"/>
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"/>
</LinearLayout>
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
}
```

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="vertical">
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a
TextView"/>
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"/>
</LinearLayout>
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
}
```

Accessing Assets

```
<?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="vertical">
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a TextView"/>
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"/>
</LinearLayout>
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
}
```

Accessing Assets

```
<?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="vertical">
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a TextView"/>
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"/>
</LinearLayout>
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    val myButton: Button = findViewById(R.id.button)
}
```

Accessing Assets

```
<?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="vertical">
    <TextView android:id="@+id/text"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:text="Hello, I am a TextView"/>
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"/>
</LinearLayout>
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    val myButton: Button = findViewById(R.id.button)
}
```

Add Event Handler

```
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    val myButton: Button = findViewById(R.id.button)
}
```

Add Event Handler

```
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
            android:onClick="sendMessage"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    val myButton: Button = findViewById(R.id.button)
}
```

Add Event Handler

```
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
            android:onClick="sendMessage"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    val myButton: Button = findViewById(R.id.button)
}
fun sendMessage(view: View) {
   logd("Ready!")
}
```

Add Event Handler

```
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
            android:onClick="sendMessage"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    val myButton: Button = findViewById(R.id.button)
}
fun sendMessage(view: View) {
   logd("Ready!")
}
```

Add Event Handler

```
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
            android:onClick="sendMessage"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    val myButton: Button = findViewById(R.id.button)
}
fun sendMessage(view: View) {
   logd("Ready!")
}
```

Using Android KTX

```
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
            android:onClick="sendMessage"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    val myButton: Button = findViewById(R.id.button)
}
fun sendMessage(view: View) {
   logd("Ready!")
}
```

Using Android KTX

```
import kotlinx.android.synthetic.main.activity_main.*
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    val myButton: Button = findViewById(R.id.button)
}
fun sendMessage(view: View) {
   logd("Ready!")
}
```

Using Android KTX

```
import kotlinx.android.synthetic.main.activity_main.*
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
}

fun sendMessage(view: View) {
   logd("Ready!")
}
```

Using Android KTX

```
import kotlinx.android.synthetic.main.activity_main.*
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
}
```

Using Android KTX

```
import kotlinx.android.synthetic.main.activity_main.*
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    button.setOnClickListener {
      text.text = "From editText: ${editText.text.toString()}"
      button.text = "Update"
    }
}
```

Using Android KTX

```
import kotlinx.android.synthetic.main.activity_main.*
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    button.setOnClickListener {
      text.text = "From editText: ${editText.text.toString()}"
      button.text = "Update"
    }
}
```

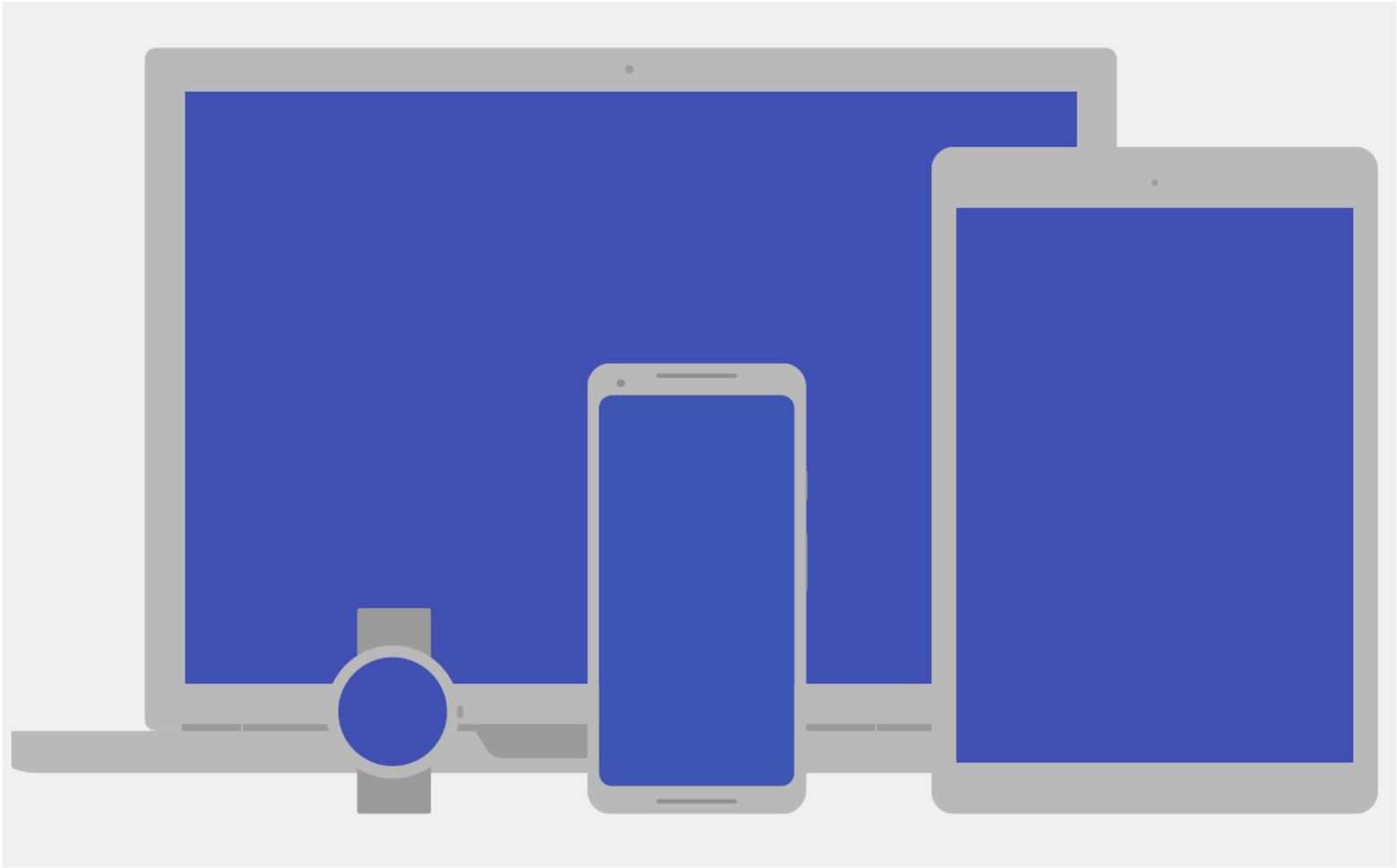
Using Android KTX

```
import kotlinx.android.synthetic.main.activity_main.*
...
    <Button android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Hello, I am a Button"
   />
...
```

```
fun onCreate(savedInstanceState: Bundle) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.main_layout)
    button.setOnClickListener {
      text.text = "From editText: ${editText.text.toString()}"
      button.text = "Update"
    }

}
```

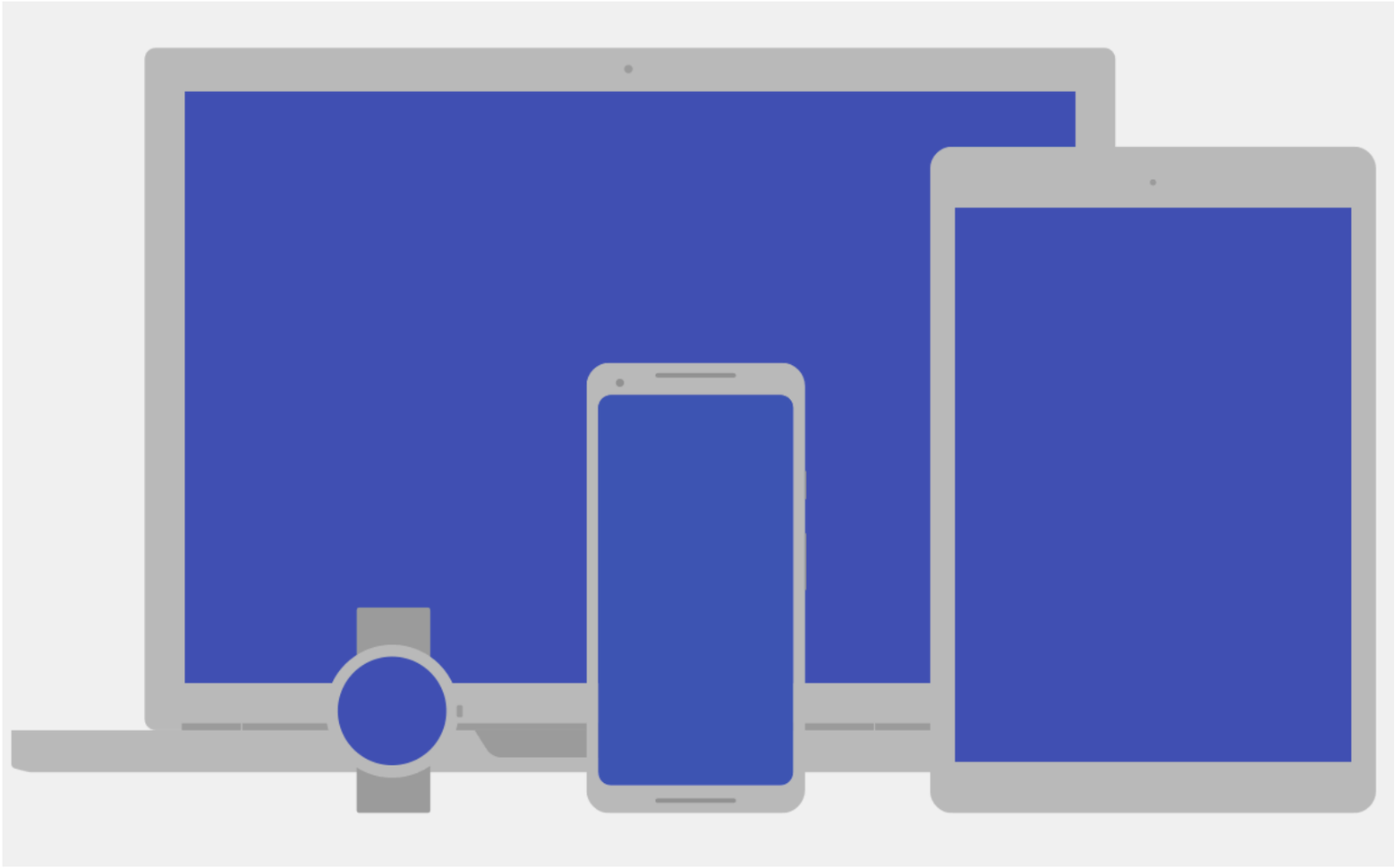
https://developer.android.com/kotlin/ktx

Supporting different screen sizes



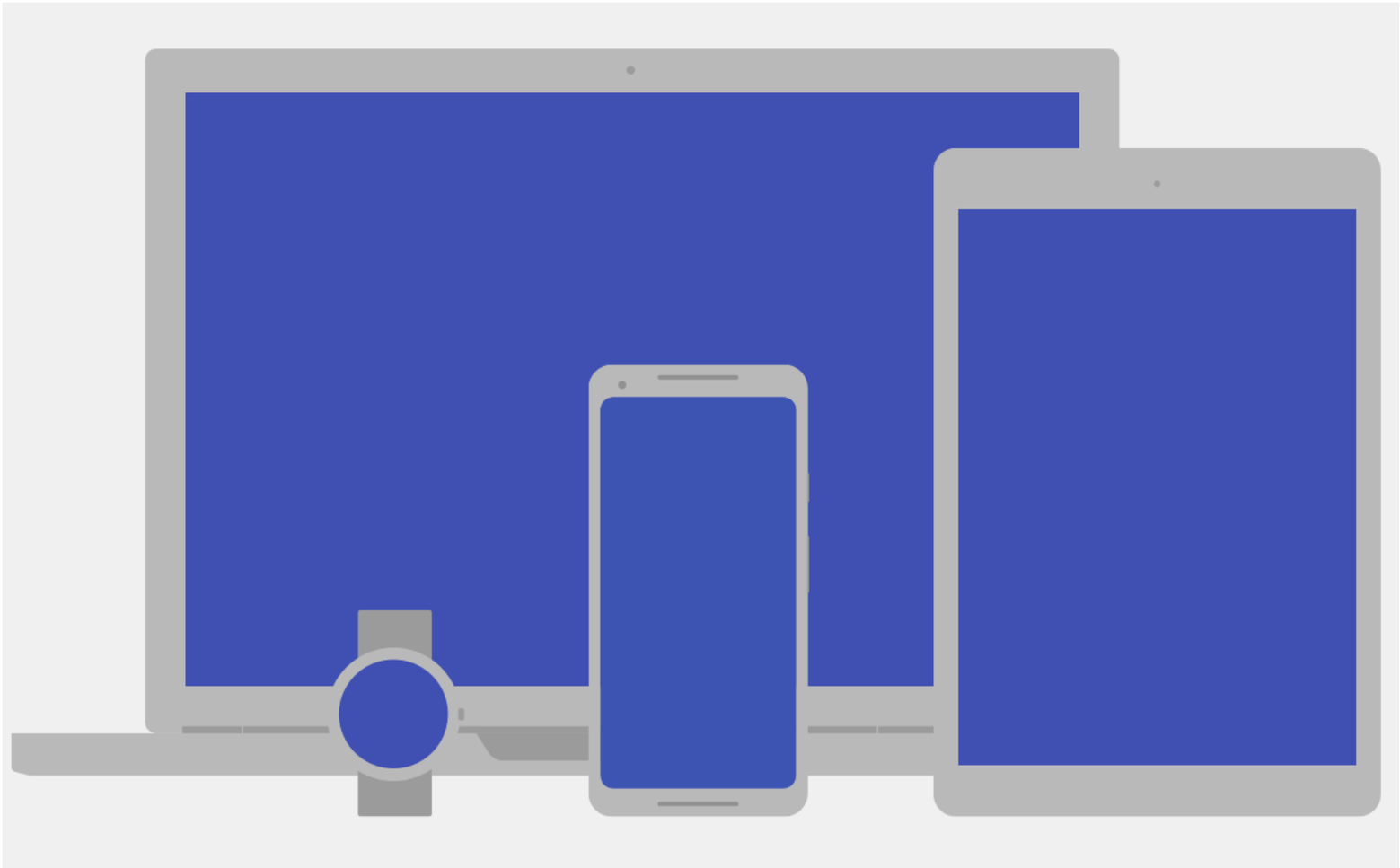
Supporting different screen sizes

- Flexible layouts



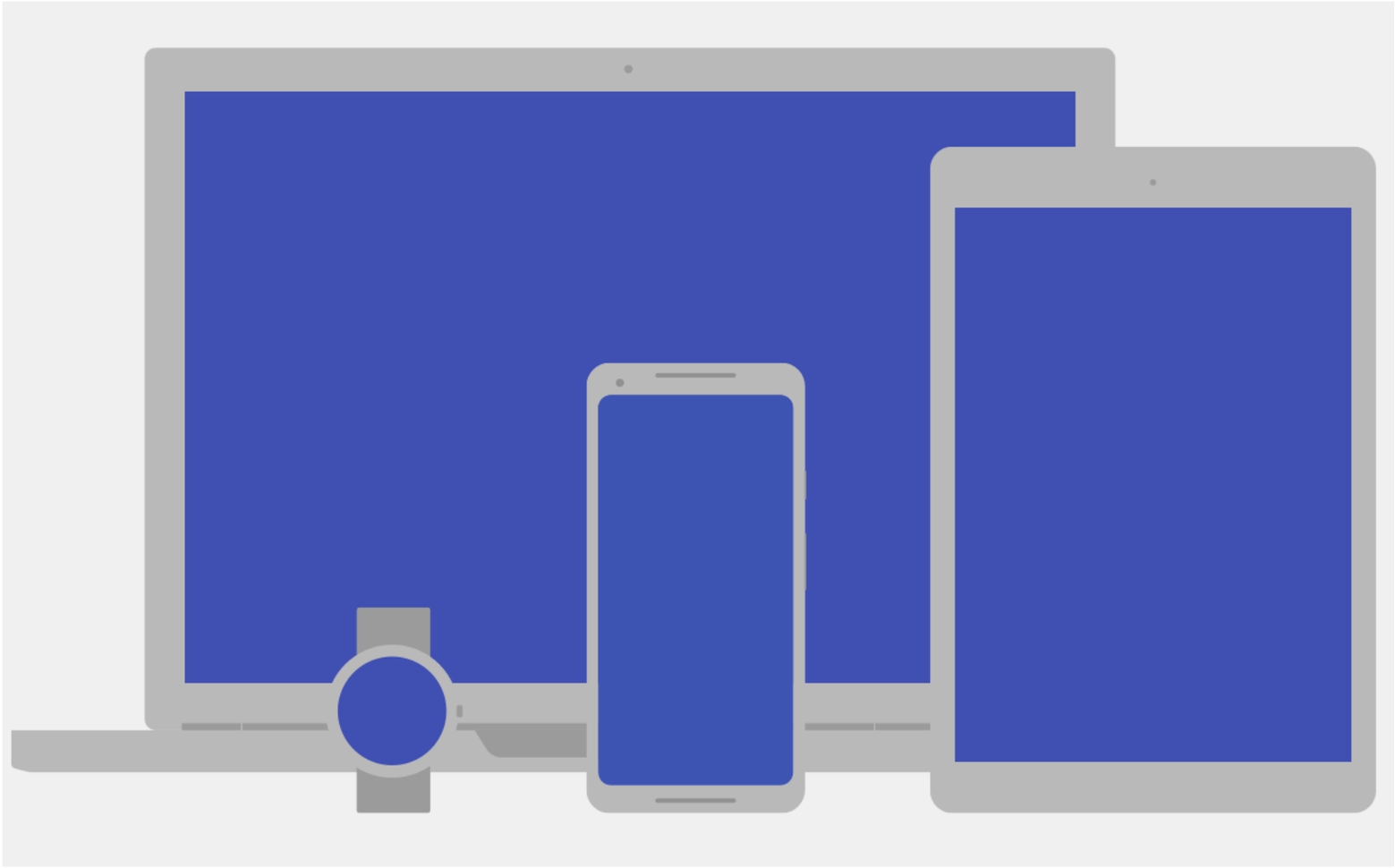
Supporting different screen sizes

- Flexible layouts
- Alternative layouts



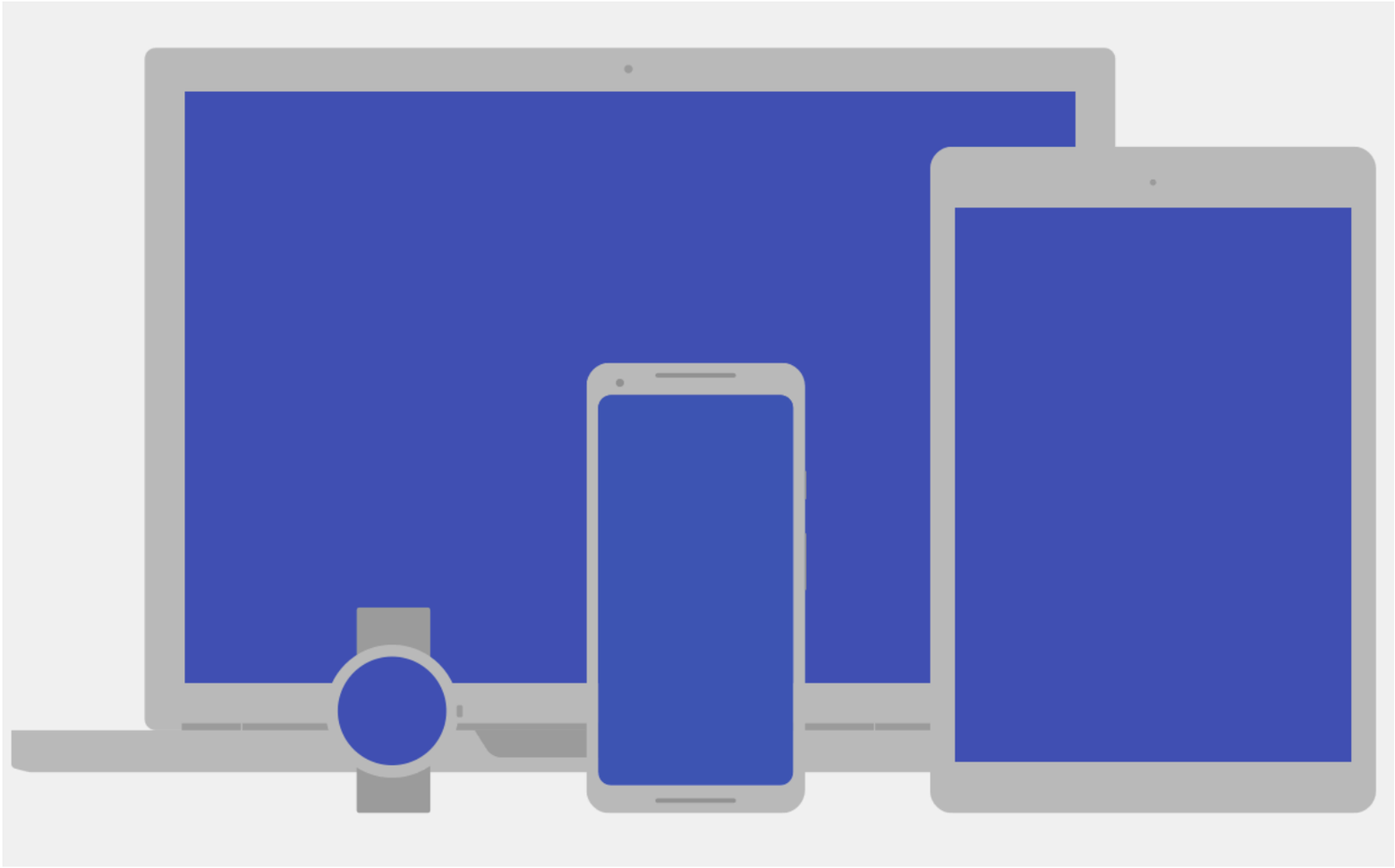
Supporting different screen sizes

- Flexible layouts
- Alternative layouts
- Stretchable images



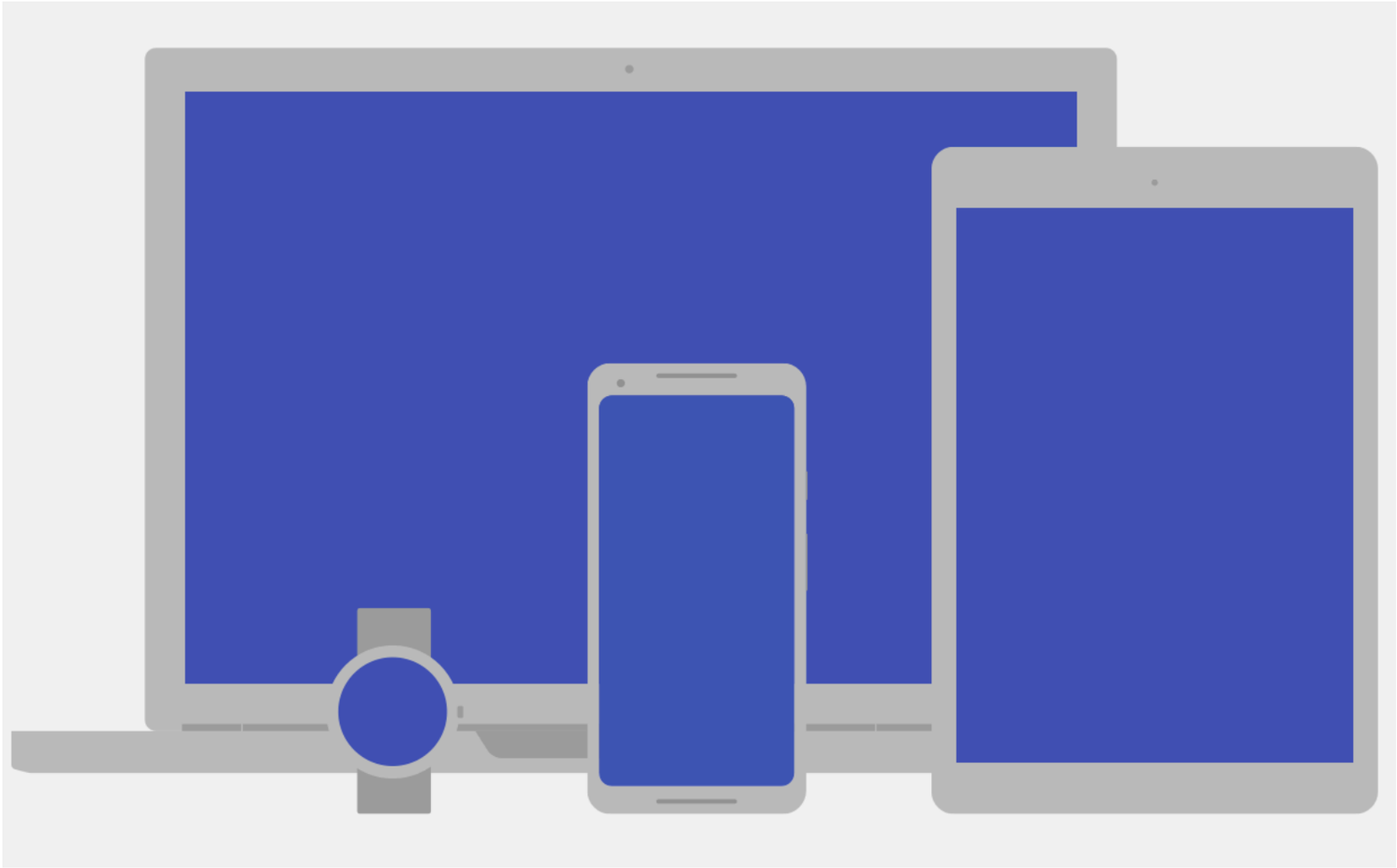
Supporting different screen sizes

- Flexible layouts
- Alternative layouts
- Stretchable images
- Alternative bitmaps



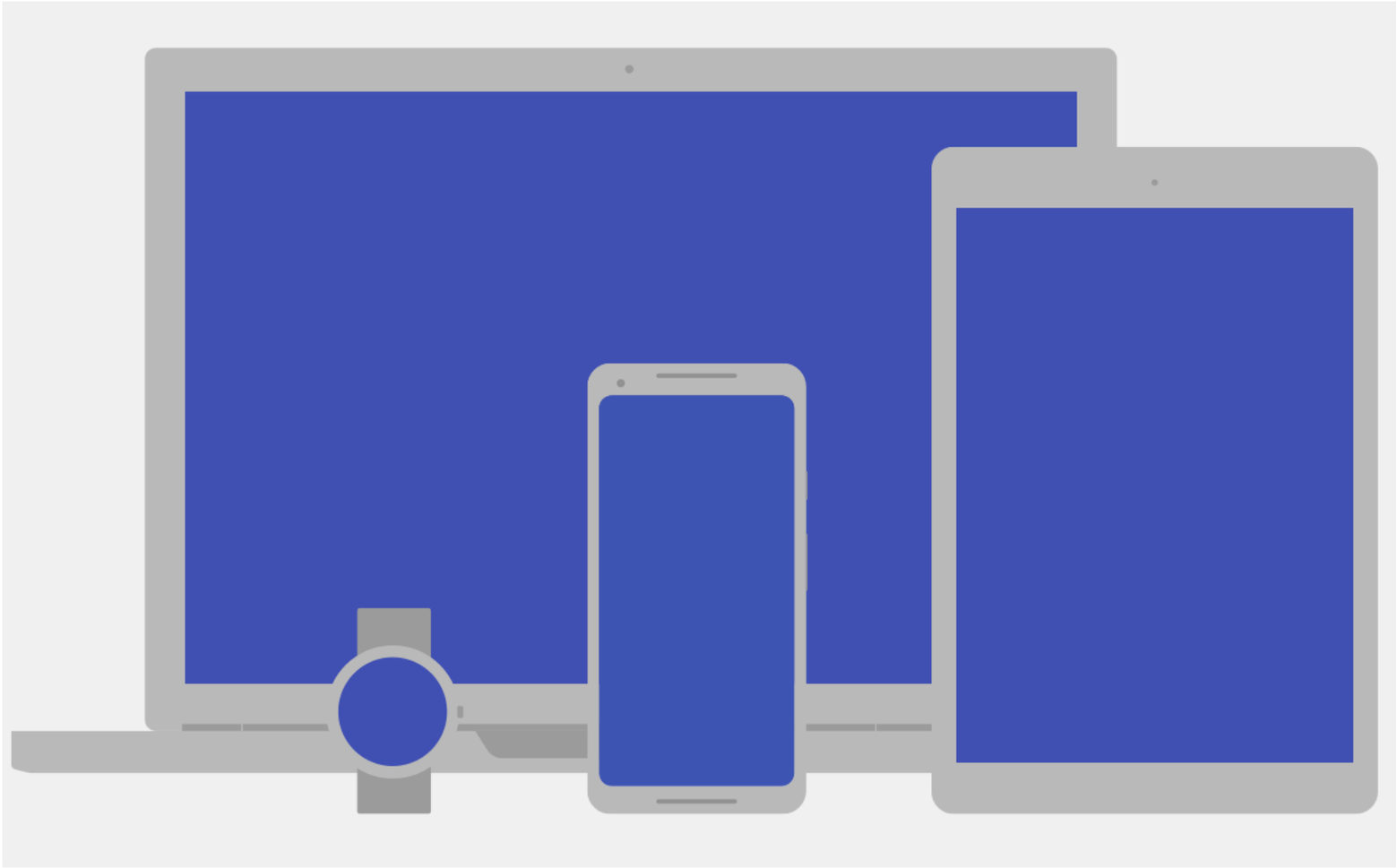
Supporting different screen sizes

- Flexible layouts
- Alternative layouts
- Stretchable images
- Alternative bitmaps
- Vector graphics



Supporting different screen sizes

- Flexible layouts
- Alternative layouts
- Stretchable images
- Alternative bitmaps
- Vector graphics



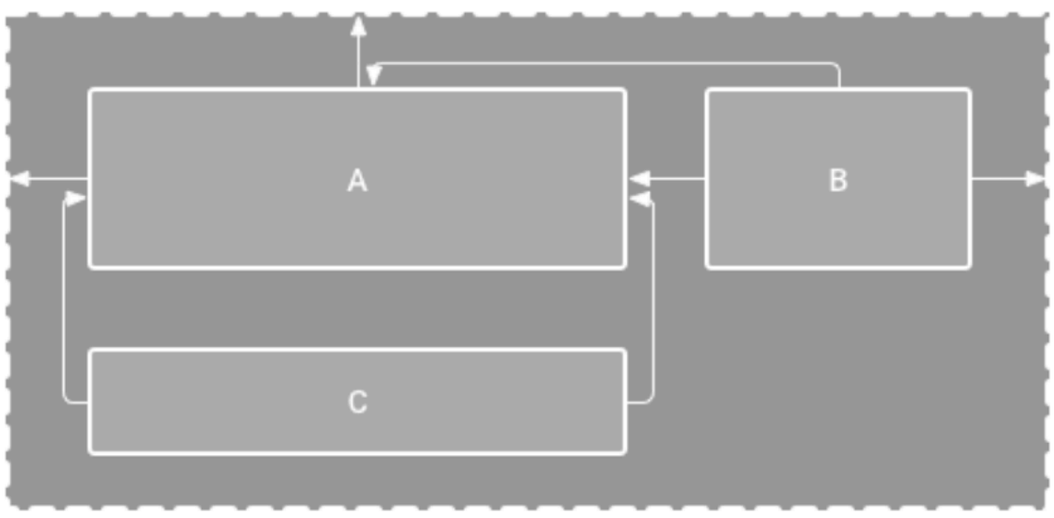
https://developer.android.com/guide/practices/screens_support

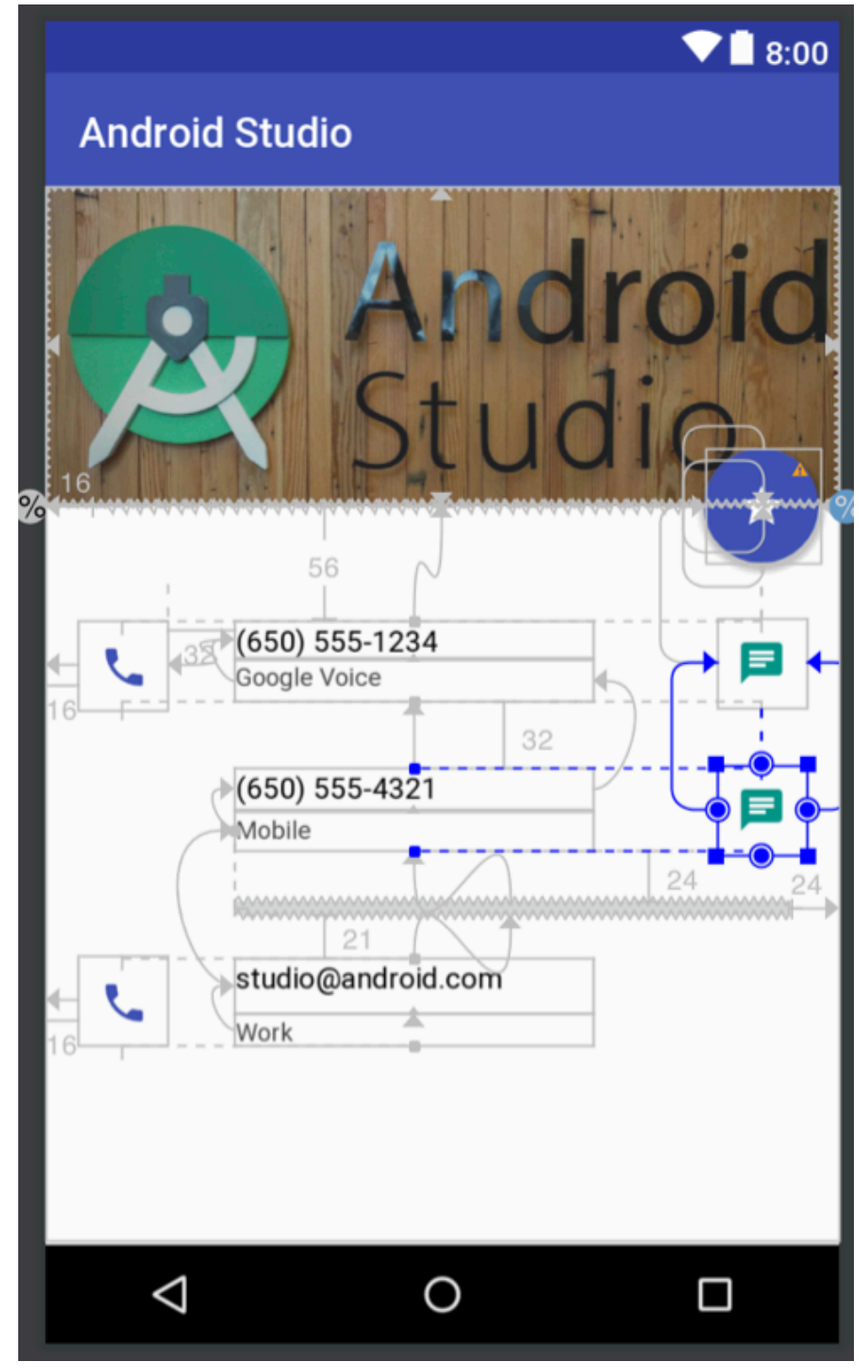
Flexible Layouts

ConstraintLayout

In module-level **gradle.build:**

```
repositories {
    google()
}

dependencies {
 implementation
 'com.android.support.constraint:constraint-layout:2.0.2'
}
```



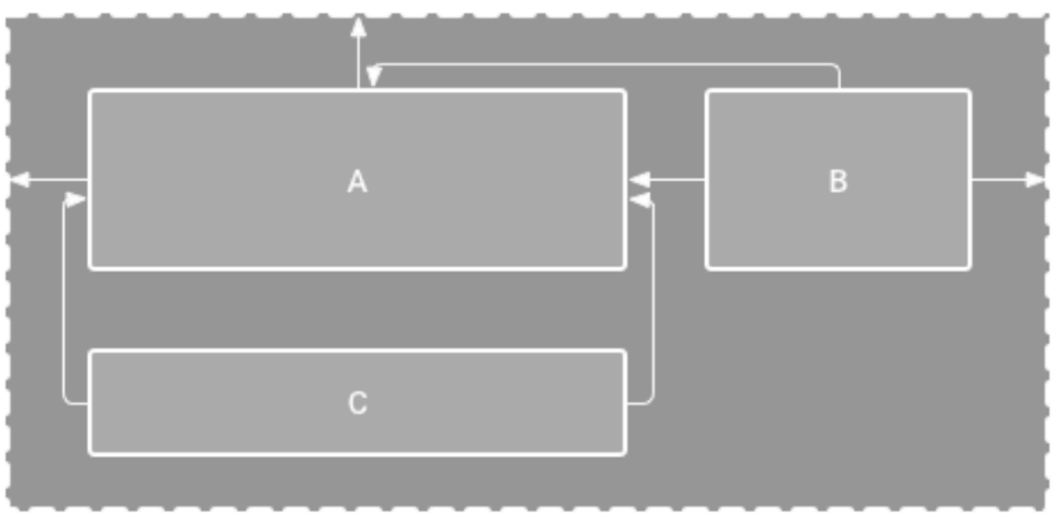


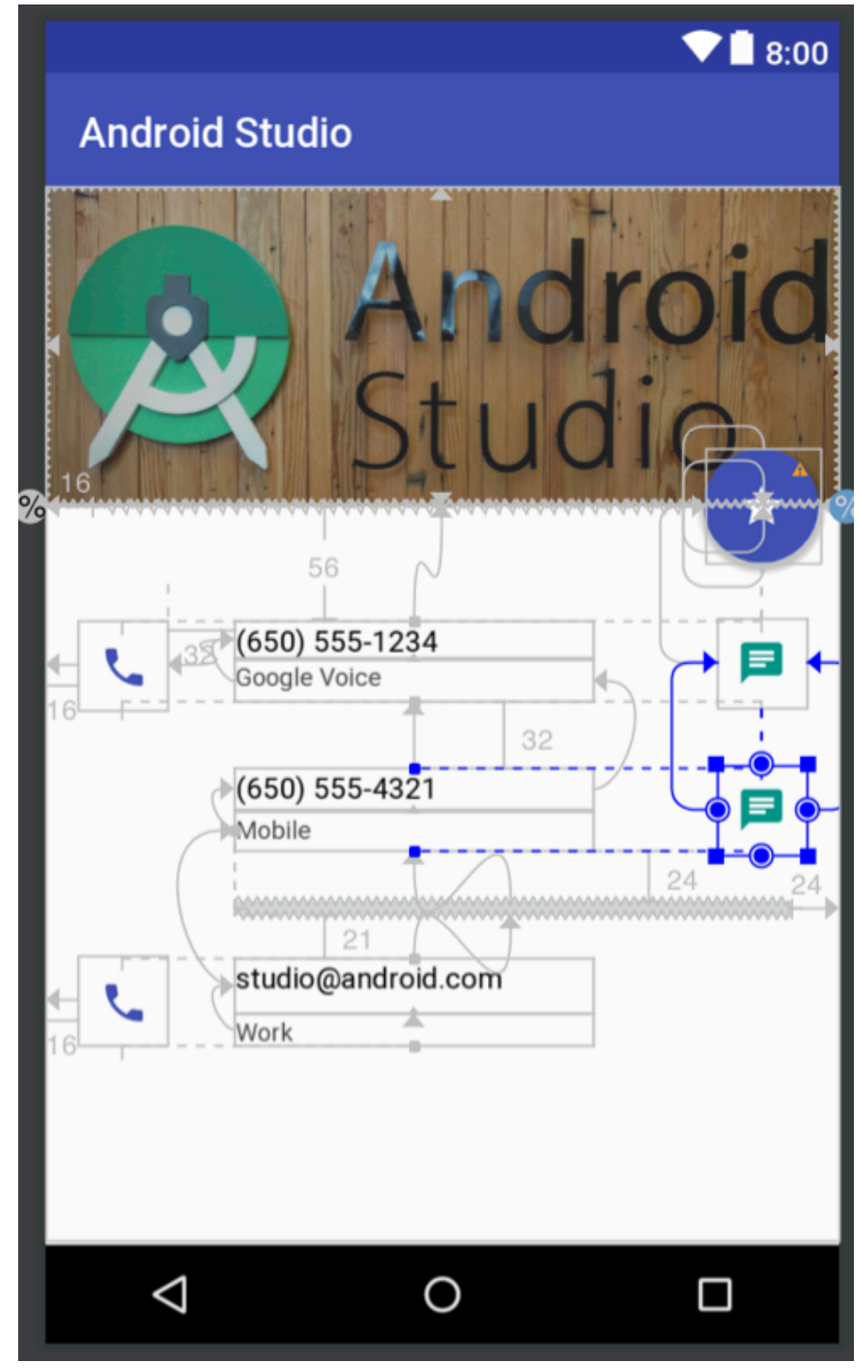
Flexible Layouts

ConstraintLayout

In module-level **gradle.build:**

```
repositories {
    google()
}

dependencies {
 implementation
 'com.android.support.constraint:constraint-layout:2.0.2'
}
```

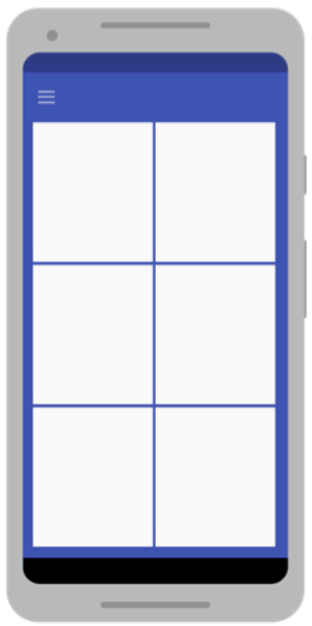


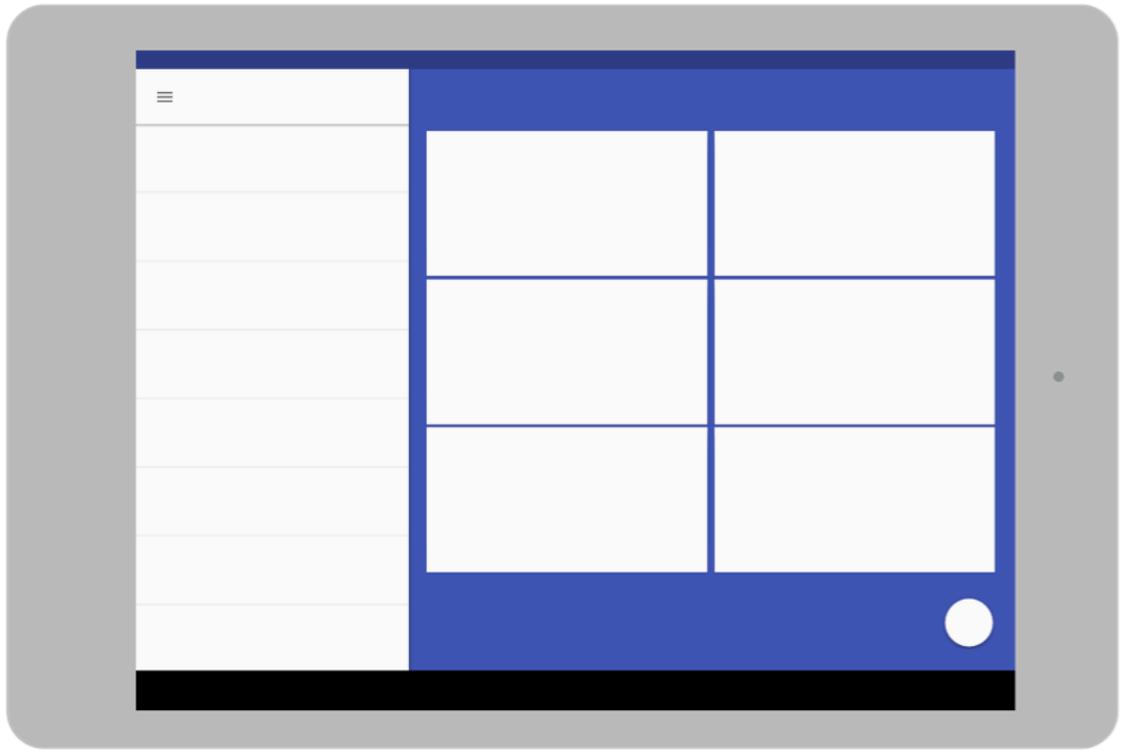


https://developer.android.com/training/constraint-layout/

Alternative layouts

```
res/layout/main_activity.xml                # Default layout
res/layout-land/main_activity.xml           # When in landscape mode
res/layout-sw600dp/main_activity.xml        # For 7” tablets
res/layout-sw600dp-land/main_activity.xml   # For 7” tablets in landscape
```

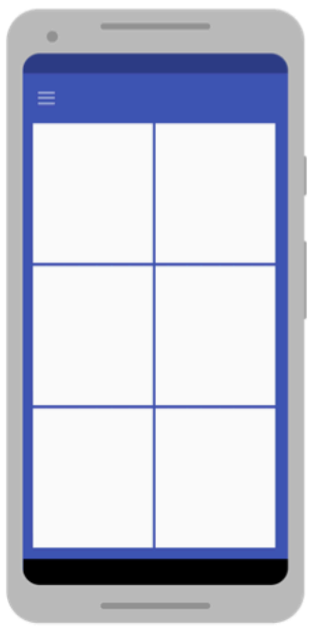


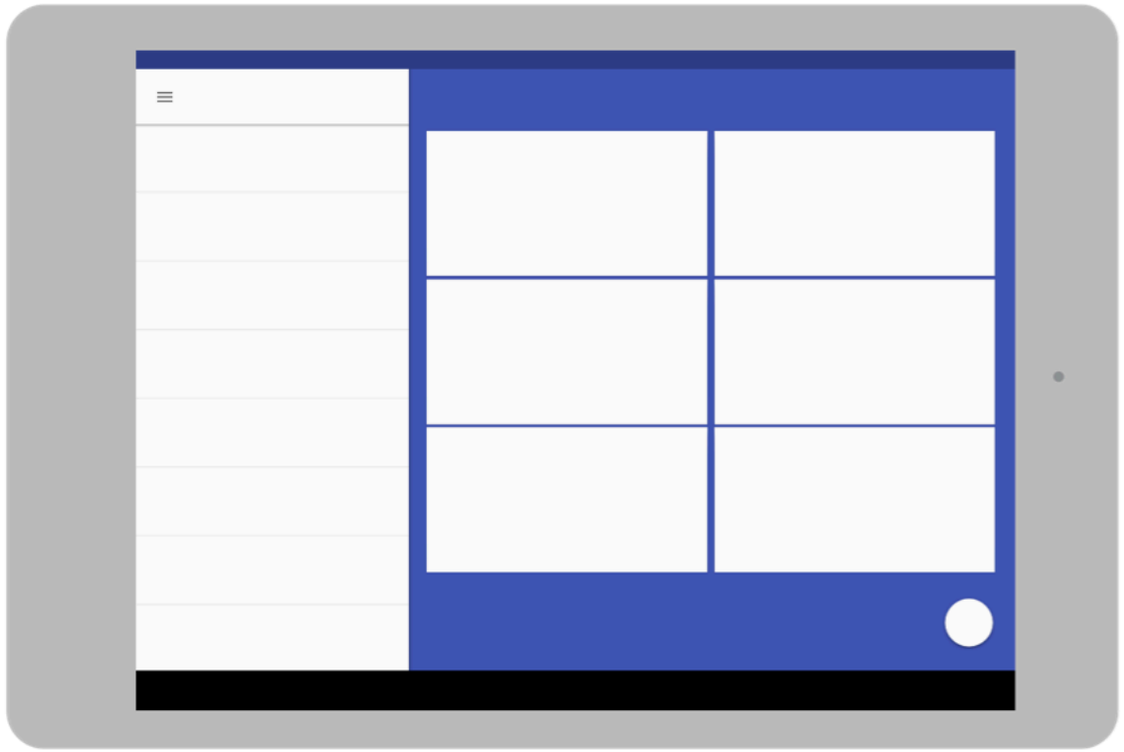


DEMO

Alternative layouts

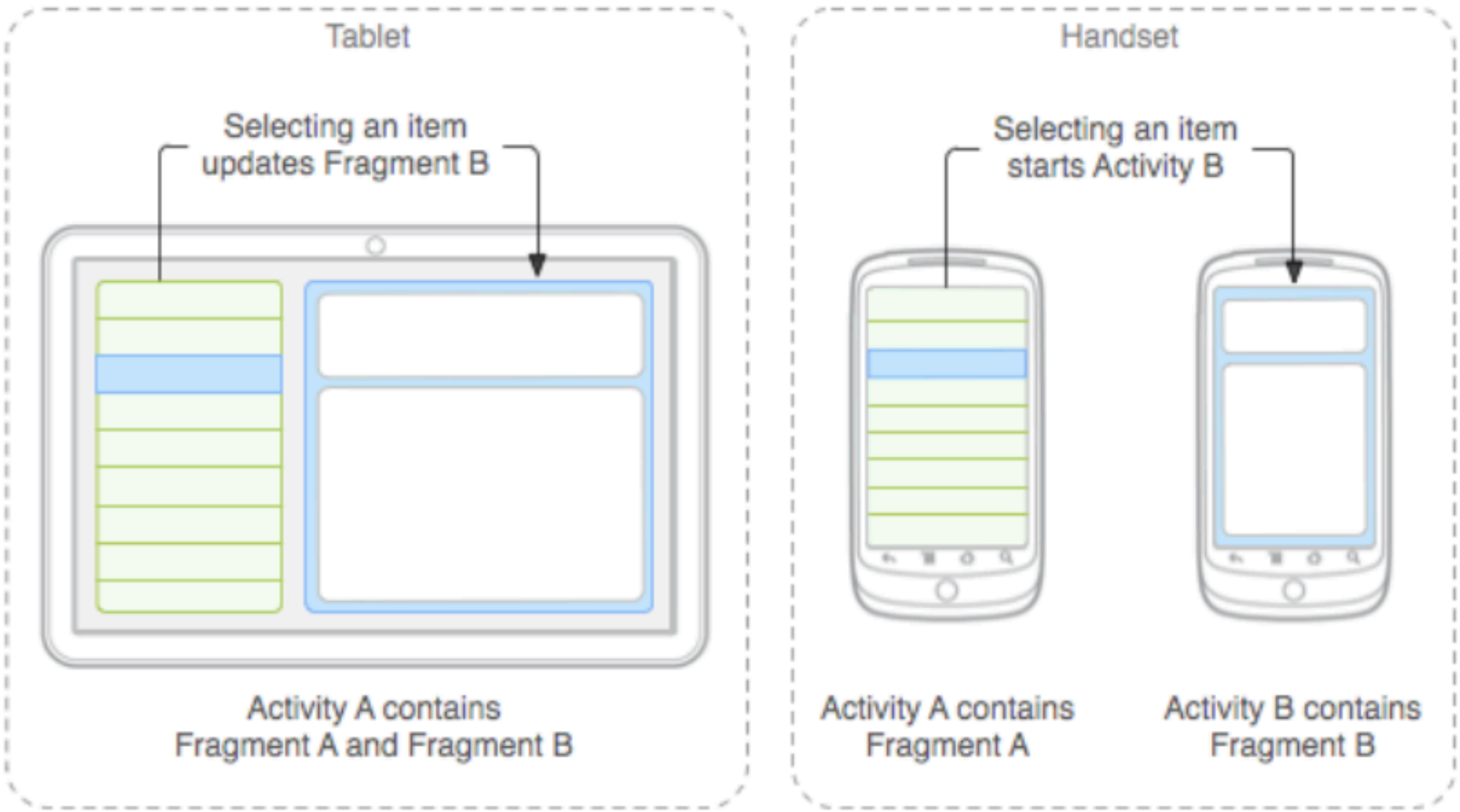
```
res/layout/main_activity.xml                # Default layout
res/layout-land/main_activity.xml           # When in landscape mode
res/layout-sw600dp/main_activity.xml        # For 7” tablets
res/layout-sw600dp-land/main_activity.xml   # For 7” tablets in landscape
```



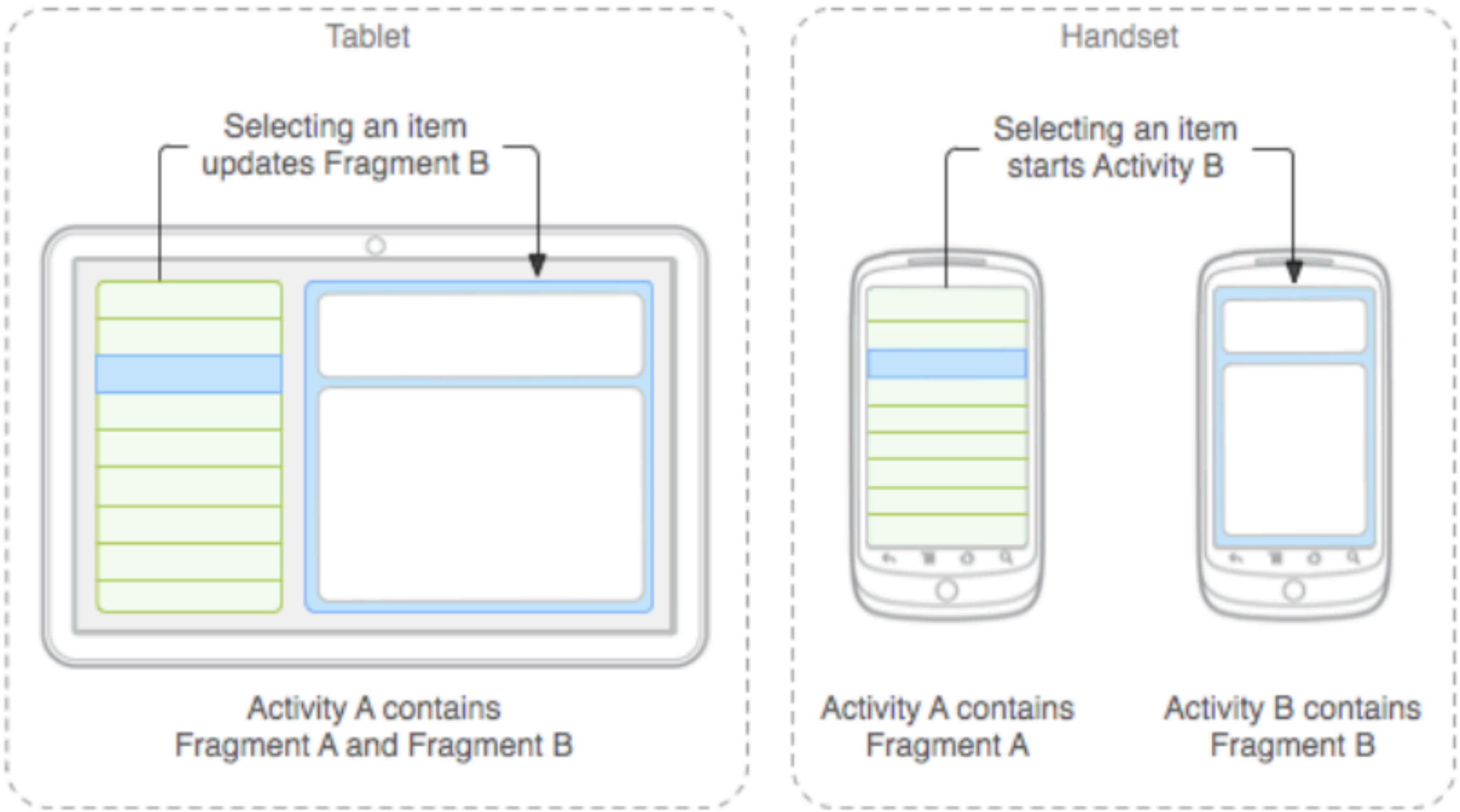


https://developer.android.com/training/multiscreen/screensizes#alternative-layouts

Building a Dynamic UI with Fragments



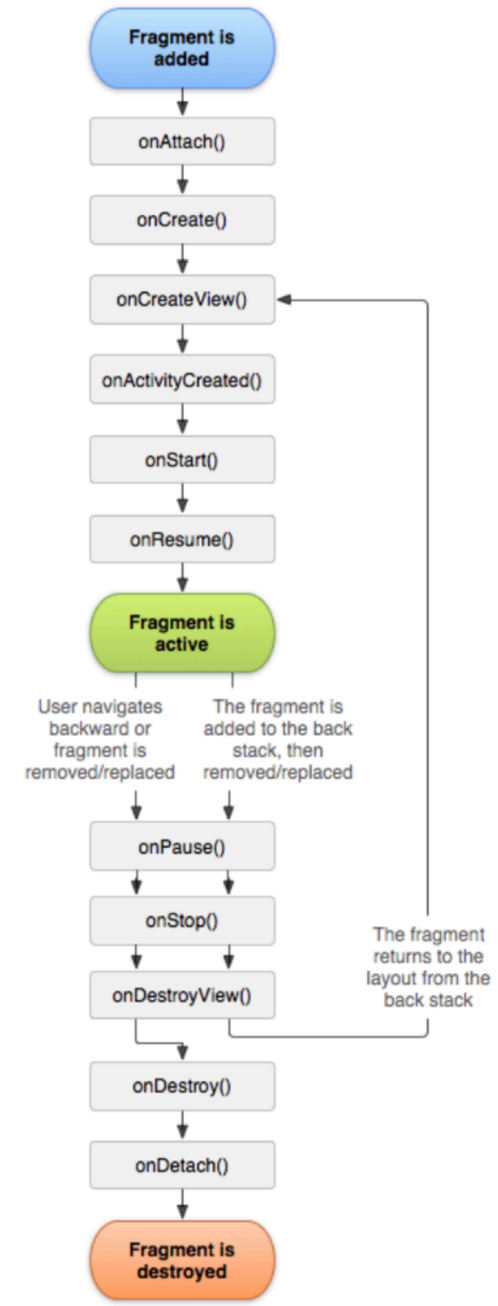
Building a Dynamic UI with Fragments



https://developer.android.com/guide/components/fragments

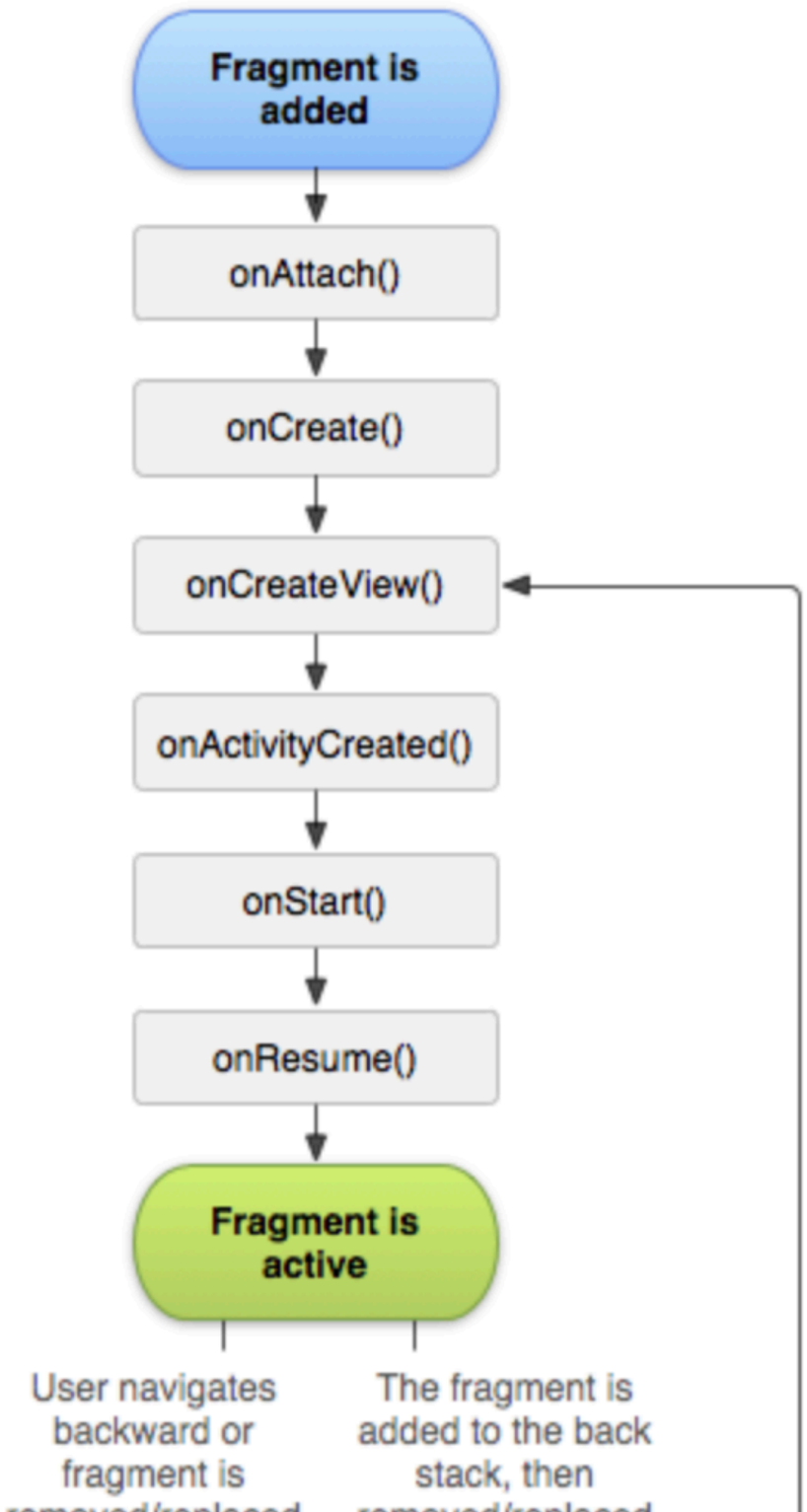
Creating a Fragment

- New callbacks
  - onAttach
  - onCreateView
  - onActivityCreated
  - onDestroyView
  - onDetach



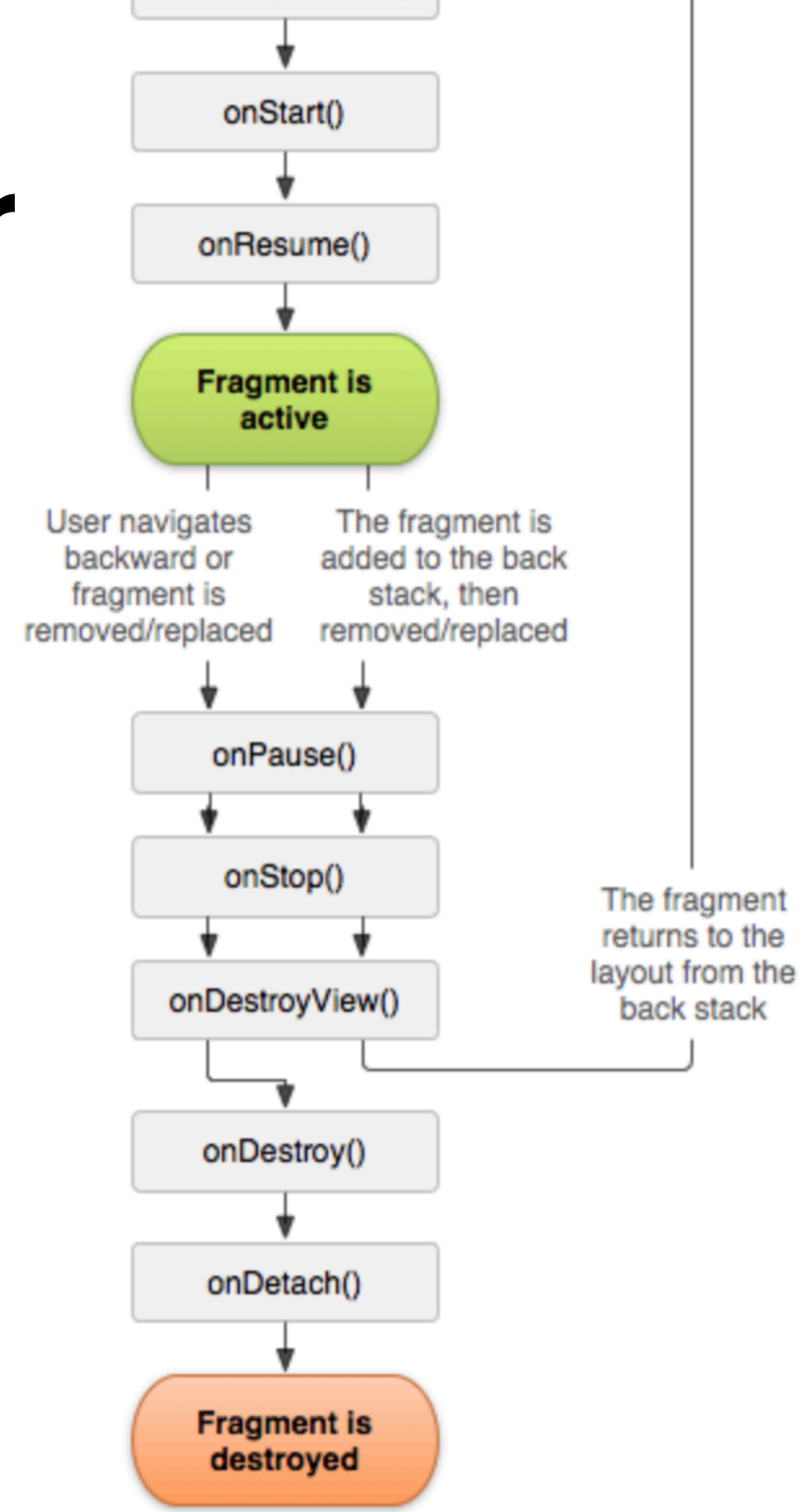
Creating a Fragment

- New callbacks
  - onAttach
  - onCreateView
  - onActivityCreated
  - onDestroyView
  - onDetach



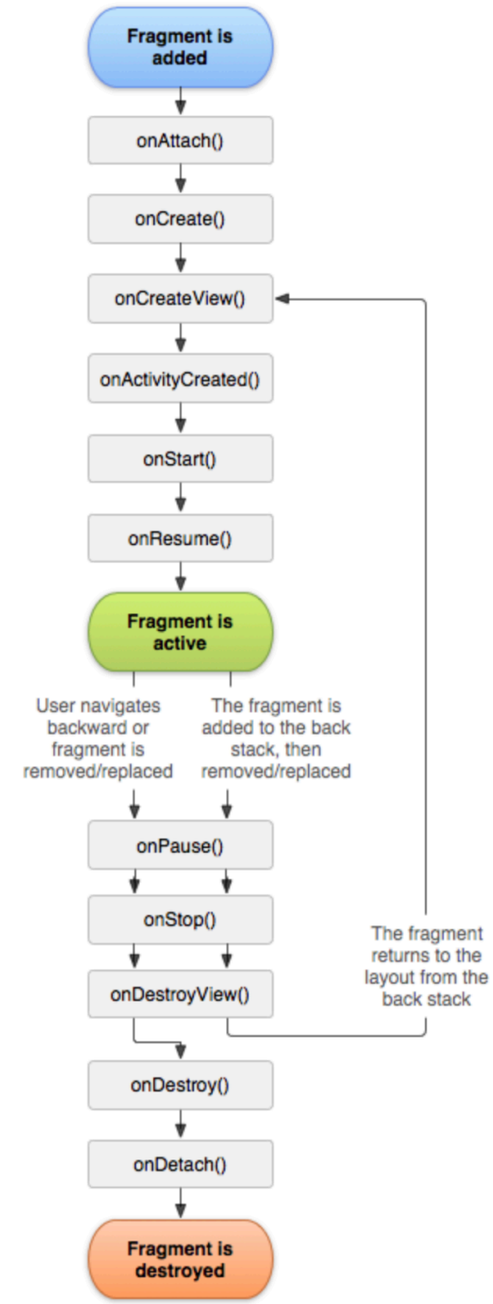
Creating a Fr

- New callbacks
  - onAttach
  - onCreateView
  - onActivityCreated
  - onDestroyView
  - onDetach



Creating a Fragment

- New callbacks
  - onAttach
  - onCreateView
  - onActivityCreated
  - onDestroyView
  - onDetach



https://developer.android.com/reference/android/support/v4/app/Fragment

Creating a Fragment

```
class ArticleListFragment : Fragment() {

    override fun onCreateView(
            inflater: LayoutInflater,
            container: ViewGroup?,
            savedInstanceState: Bundle?
    ): View {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.example_fragment, container, false)
    }
}
```

Creating a Fragment

```
class ArticleListFragment : Fragment() {

    override fun onCreateView(
            inflater: LayoutInflater,
            container: ViewGroup?,
            savedInstanceState: Bundle?
    ): View {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.example_fragment, container, false)
    }
}
```

https://developer.android.com/reference/android/support/v4/app/Fragment

Creating a Fragment

```
class ArticleListFragment : Fragment() {

    override fun onCreateView(
            inflater: LayoutInflater,
            container: ViewGroup?,
            savedInstanceState: Bundle?
    ): View {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.example_fragment, container, false)
    }
}
```

Declare the fragment inside the activity's layout file.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <fragment android:name="com.example.news.ArticleListFragment"
            android:id="@+id/list"
            android:layout_weight="1"
            android:layout_width="0dp"
            android:layout_height="match_parent" />
    <fragment android:name="com.example.news.ArticleReaderFragment"
            android:id="@+id/viewer"
            android:layout_weight="2"
            android:layout_width="0dp"
            android:layout_height="match_parent" />
</LinearLayout>
```

https://developer.android.com/reference/android/support/v4/app/Fragment

Creating a Fragment

```
class ArticleListFragment : Fragment() {

    override fun onCreateView(
            inflater: LayoutInflater,
            container: ViewGroup?,
            savedInstanceState: Bundle?
    ): View {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.example_fragment, container, false)
    }
}
```

Or, programmatically add the fragment to an existing ViewGroup

```
val fragmentManager = supportFragmentManager
val fragmentTransaction = fragmentManager.beginTransaction()
```

https://developer.android.com/reference/android/support/v4/app/Fragment

Creating a Fragment

```
class ArticleListFragment : Fragment() {

    override fun onCreateView(
            inflater: LayoutInflater,
            container: ViewGroup?,
            savedInstanceState: Bundle?
    ): View {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.example_fragment, container, false)
    }
}
```

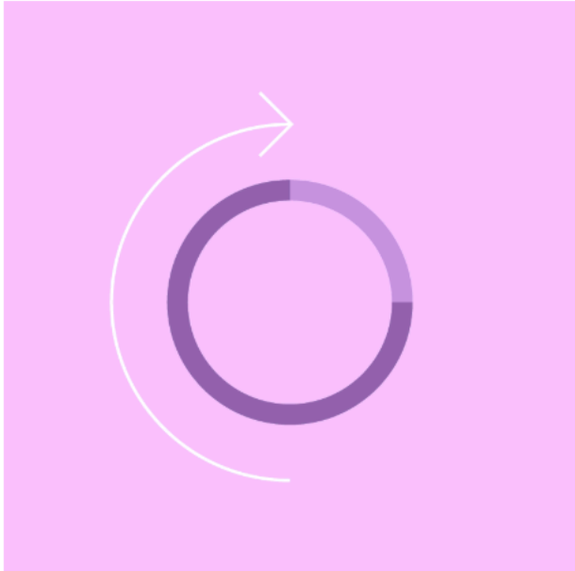
Or, programmatically add the fragment to an existing ViewGroup

```
val fragmentManager = supportFragmentManager
val fragmentTransaction = fragmentManager.beginTransaction()

val fragment = ArticleListFragment()
fragmentTransaction.add(R.id.fragment_container, fragment)
fragmentTransaction.commit()
```

https://developer.android.com/reference/android/support/v4/app/Fragment

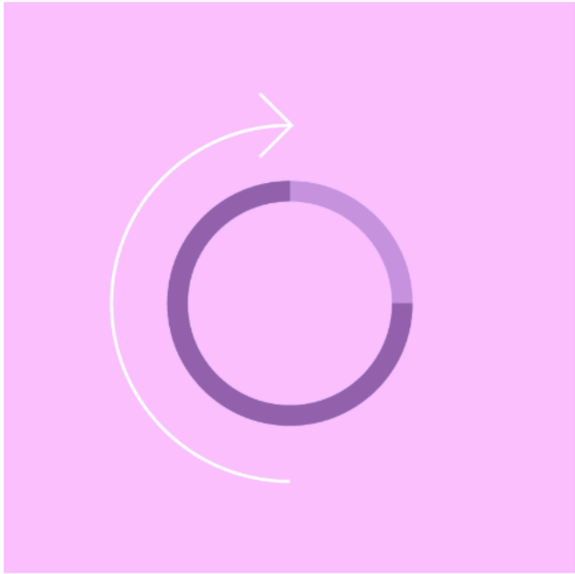
Progress Indicators

- ProgressBar
- RatingBar
- SeekBar



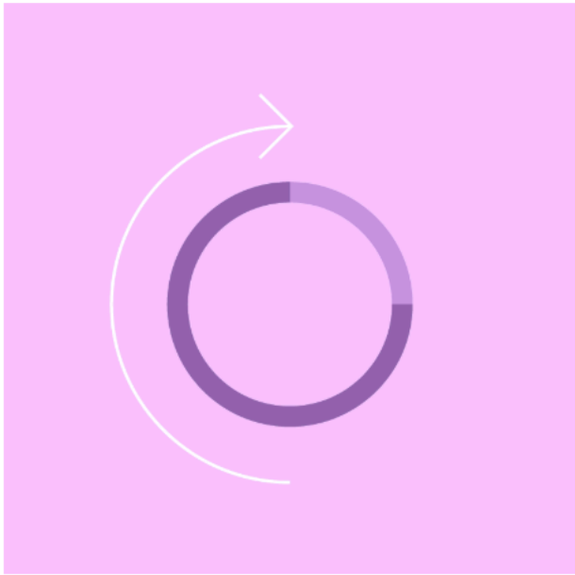
Progress Indicators

```
<ProgressBar
      android:id="@+id/indicator"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:visibility="gone"
      />
```



Progress Indicators

```
<ProgressBar
      android:id="@+id/indicator"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:visibility="gone"
      />

//before starting the action
indicator.visibility = View.VISIBLE

//when the action is done
indicator.visibility = View.GONE
```

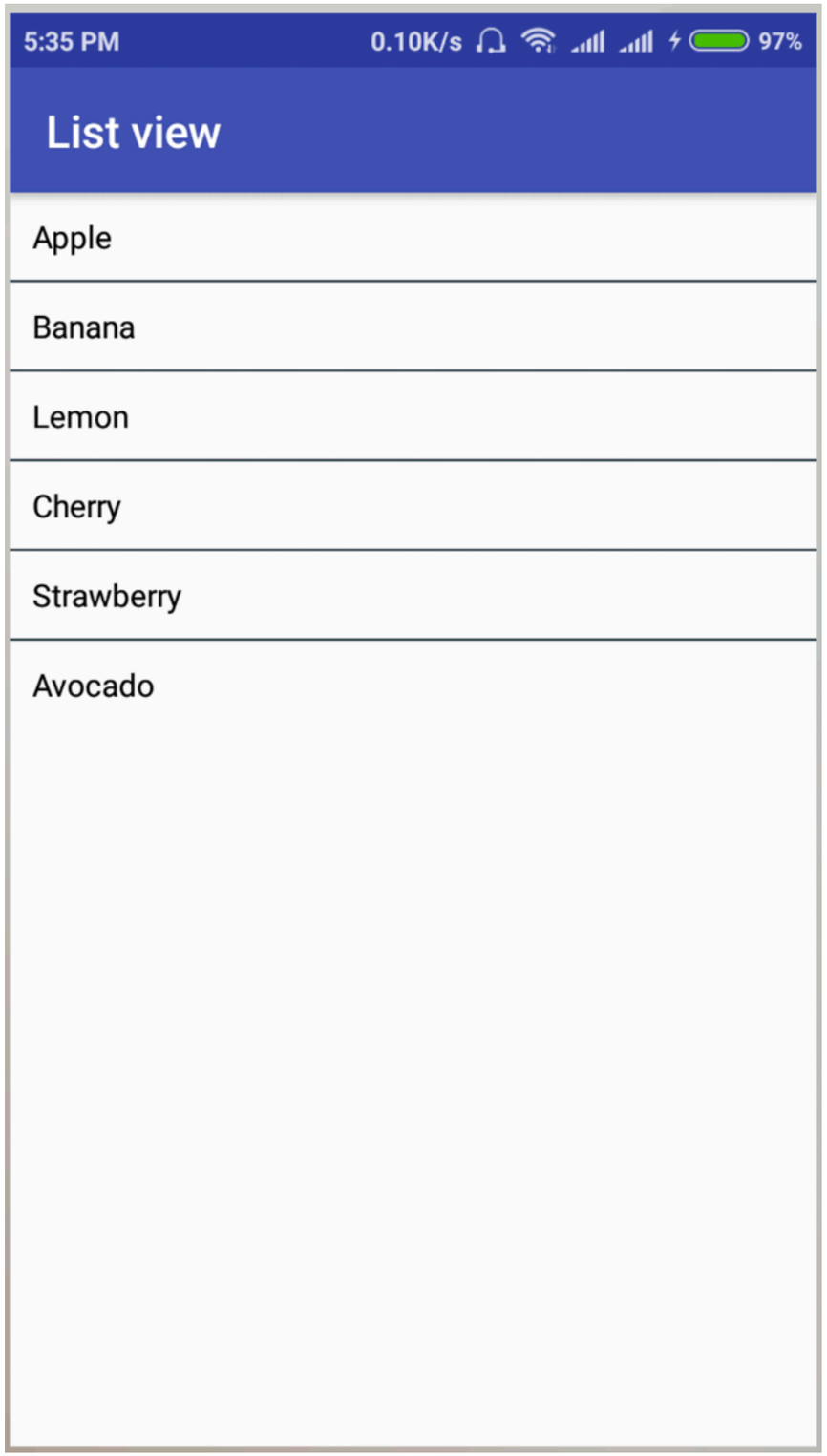


https://developer.android.com/reference/android/widget/ProgressBar

Lists

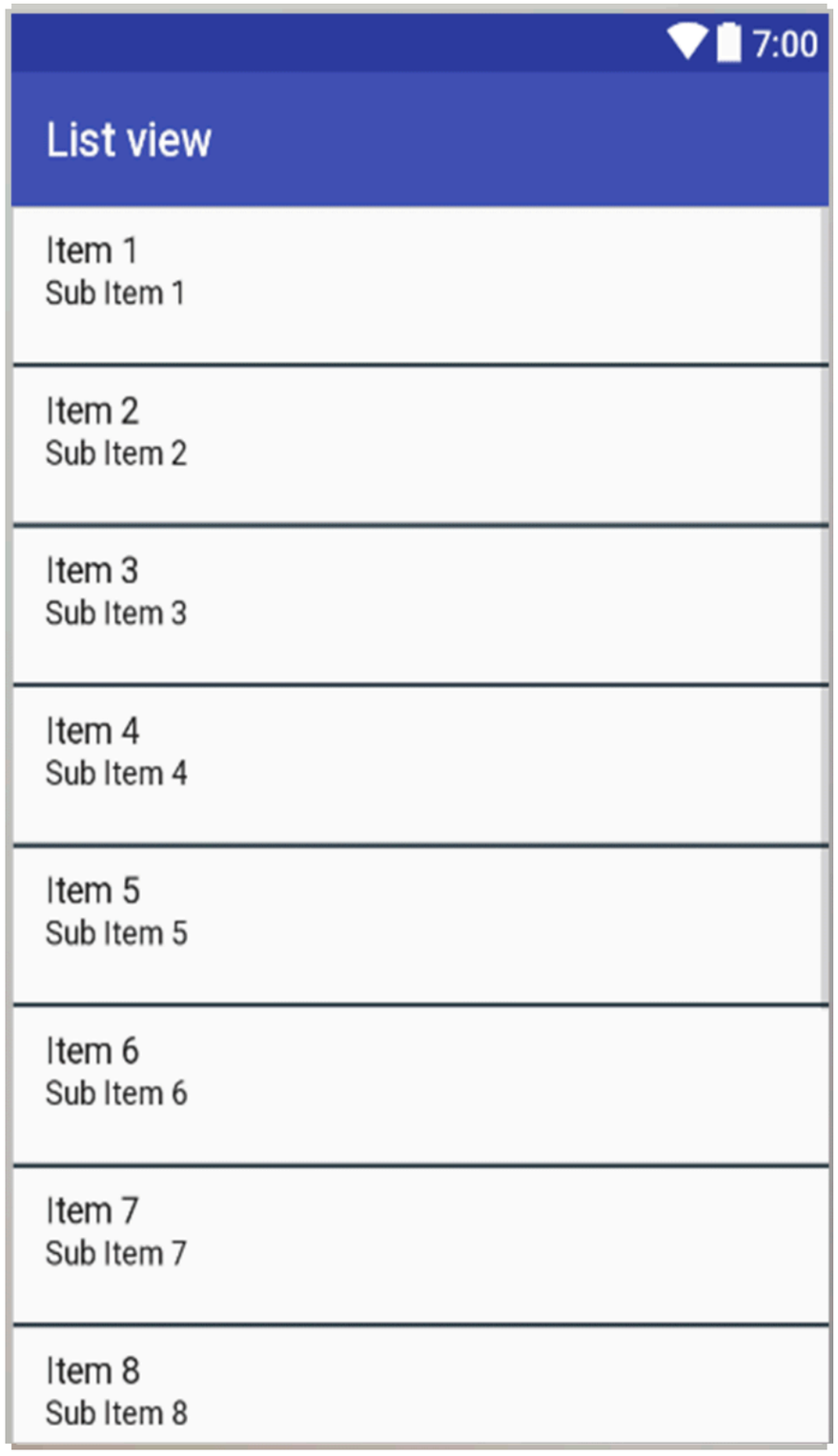
Lists

- ListView



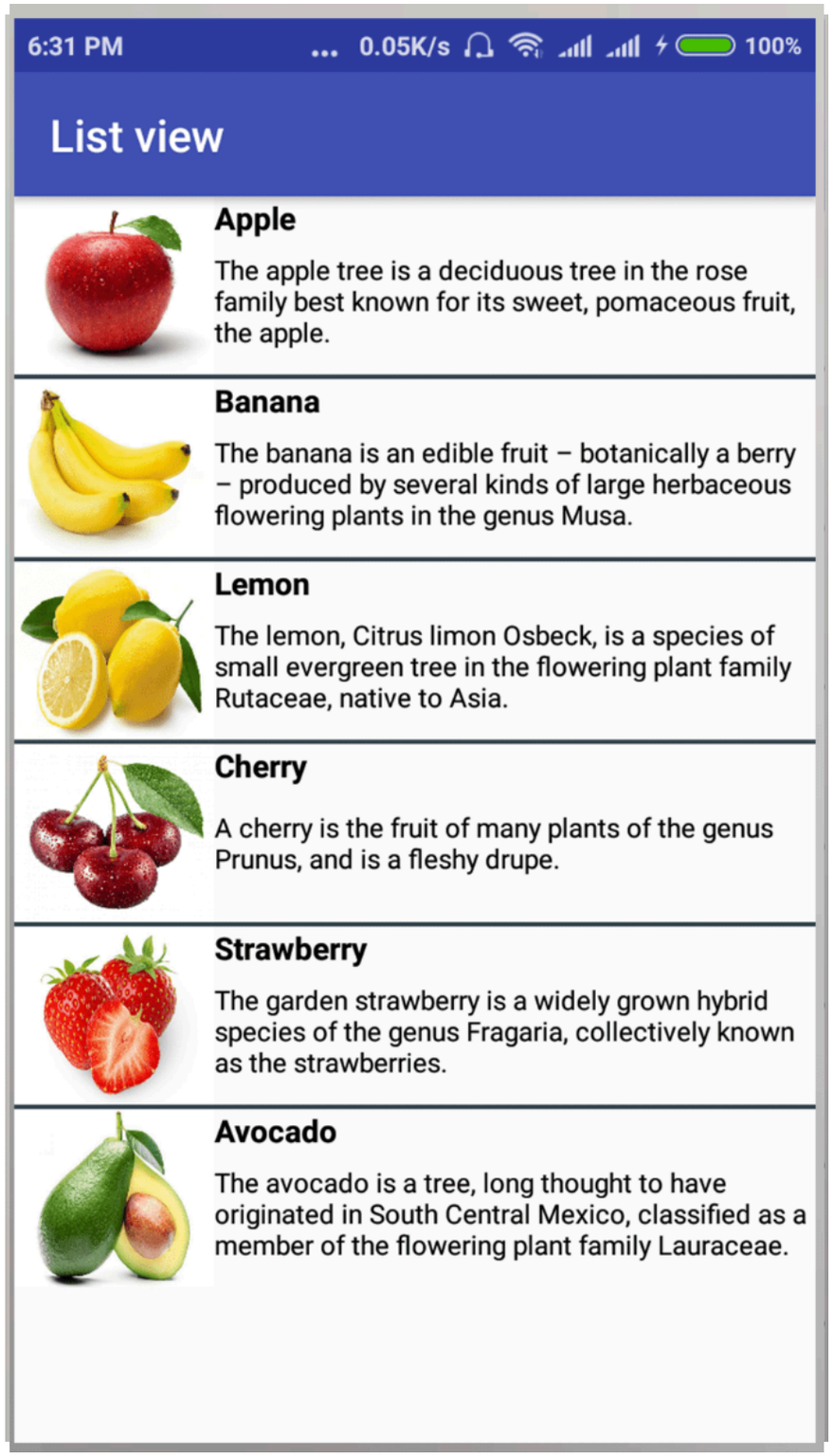
Lists

- ListView
- RecyclerView



Lists

- ListView
- RecyclerView



ListView

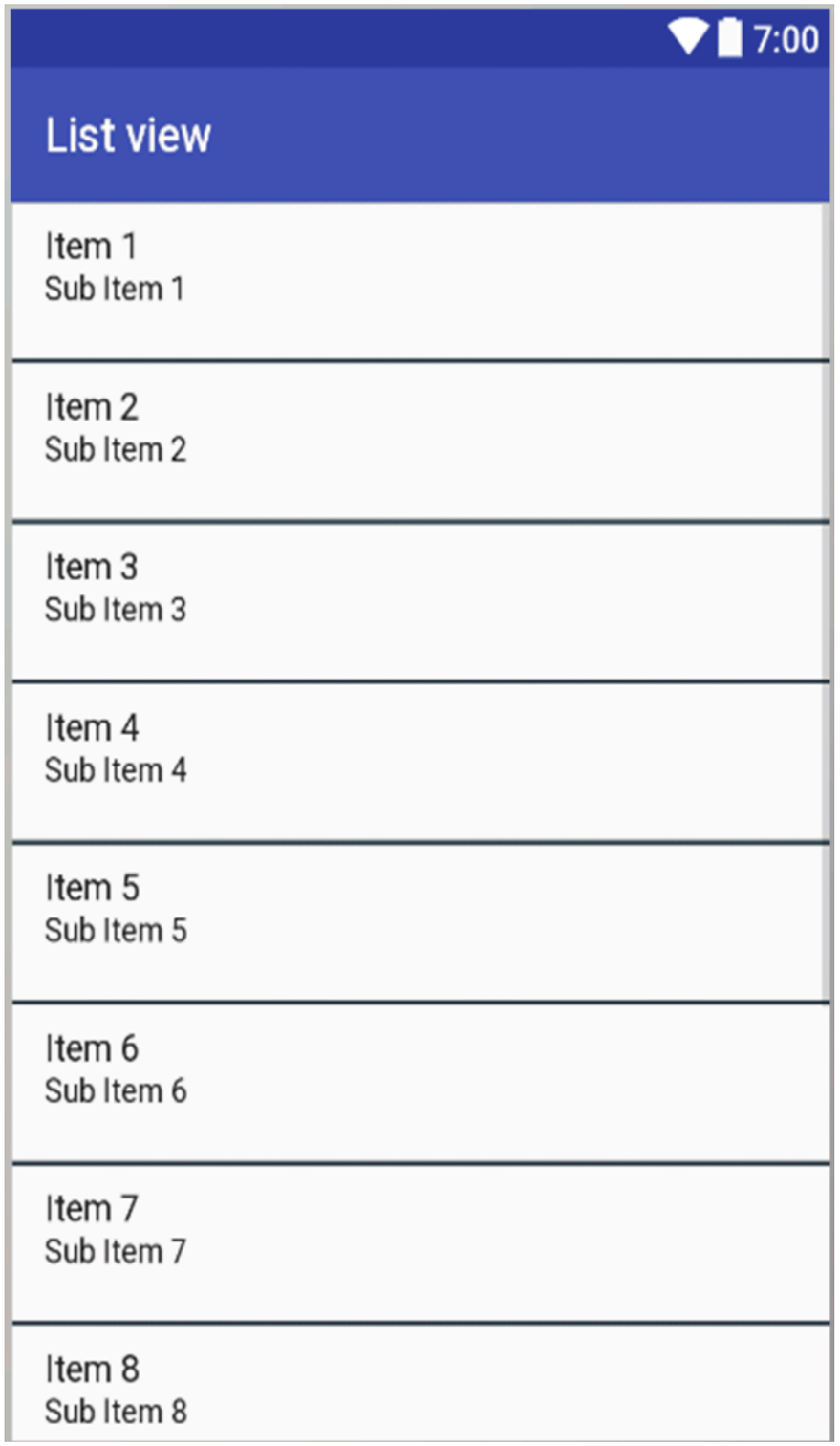
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingLeft="8dp"
        android:paddingRight="8dp">
    <ListView android:id="@android:id/list"
              android:layout_width="match_parent"
              android:layout_height="match_parent"/>
 </LinearLayout>
```

```
<?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="wrap_content"
     android:orientation="vertical">
     <TextView android:id="@+id/text"
         android:textSize="16sp"
         android:textStyle="bold"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"/>
     <TextView android:id="@+id/subText"
       ...
 </LinearLayout>
```

ListView

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingLeft="8dp"
        android:paddingRight="8dp">
    <ListView android:id="@android:id/list"
              android:layout_width="match_parent"
              android:layout_height="match_parent"/>
 </LinearLayout>
```

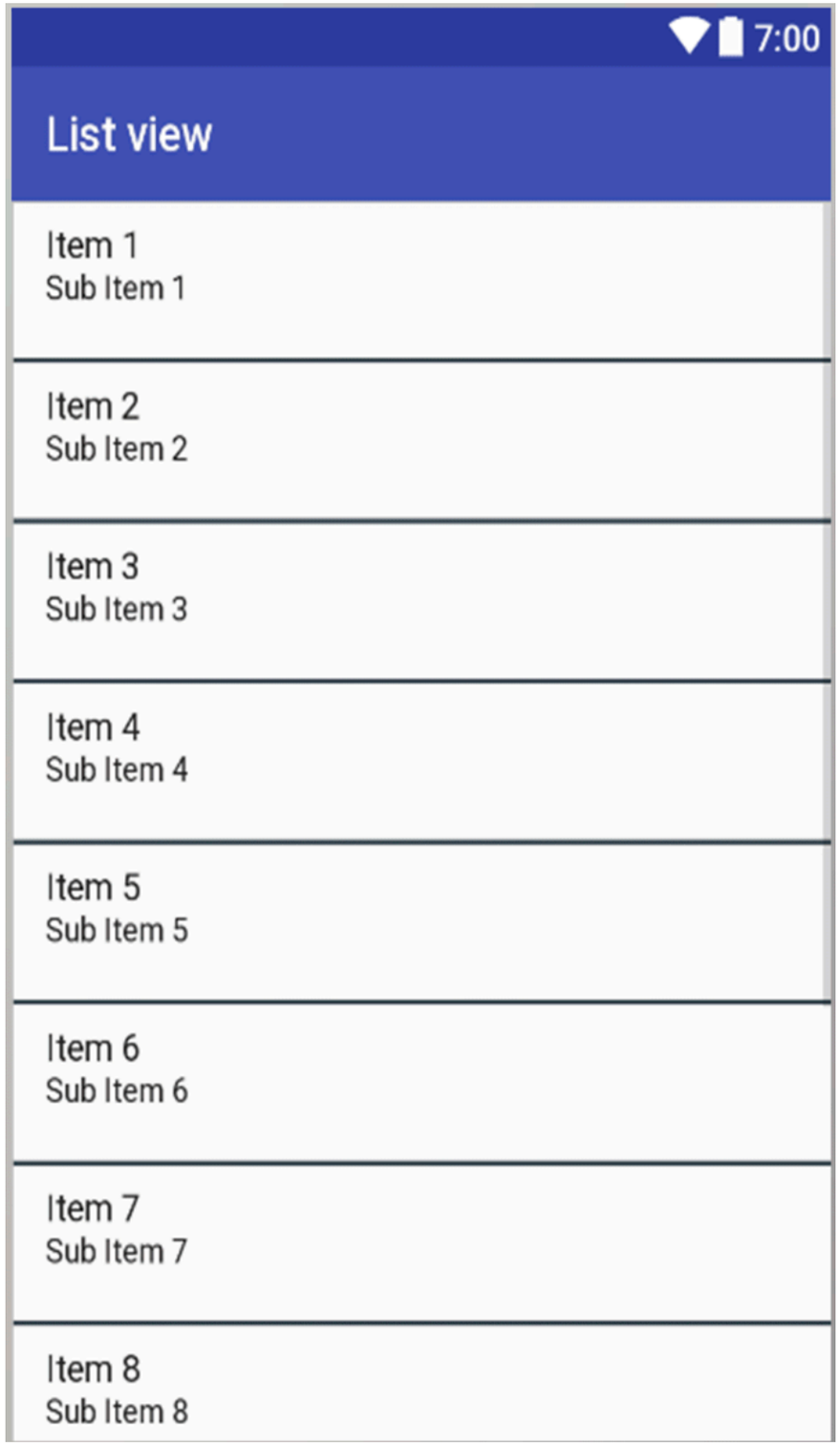
```
<?xml version="1.0" encoding="utf-8"?>
 <LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:orientation="vertical">
     <TextView android:id="@+id/text"
         android:textSize="16sp"
         android:textStyle="bold"
         android:layout_width="match_parent"
         android:layout_height="wrap_content"/>
     <TextView android:id="@+id/subText"
       ...
 </LinearLayout>
```



ListView

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="..."
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingLeft="8dp"
        android:paddingRight="8dp">
    <ListView android:id="@android:id/myList"
              android:layout_width="match_parent"
              android:layout_height="match_parent"/>
</LinearLayout>
```

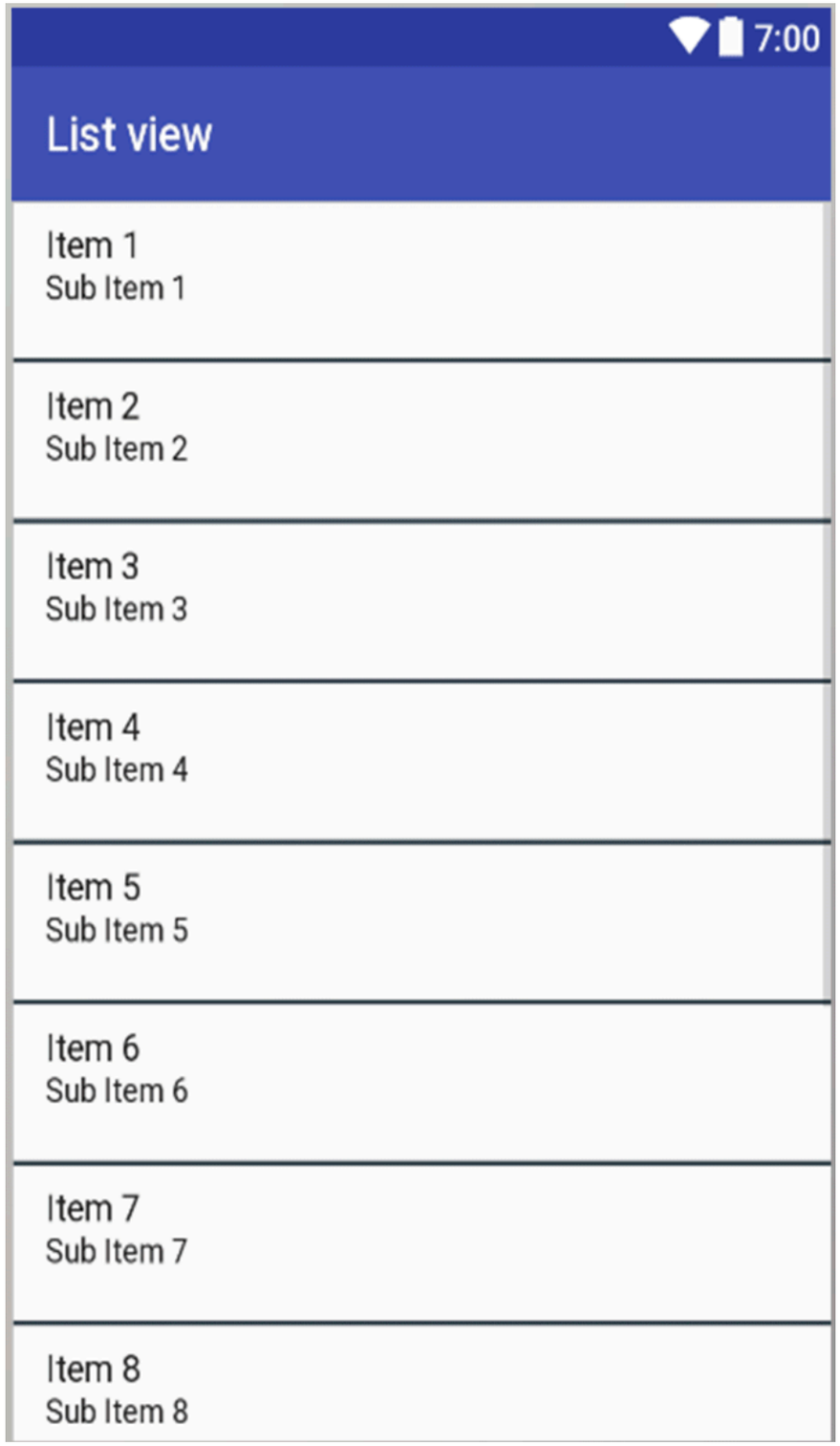
```
val arrayAdapter = ArrayAdapter<String>(this,
  android.R.layout.simple_list_item_1, arrayList)

myList.adapter = arrayAdapter
```



ListView

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="..."
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingLeft="8dp"
        android:paddingRight="8dp">
    <ListView android:id="@android:id/myList"
              android:layout_width="match_parent"
              android:layout_height="match_parent"/>
</LinearLayout>
```

```
val arrayAdapter = ArrayAdapter<String>(this,
  android.R.layout.simple_list_item_1, arrayList)

myList.adapter = arrayAdapter
```

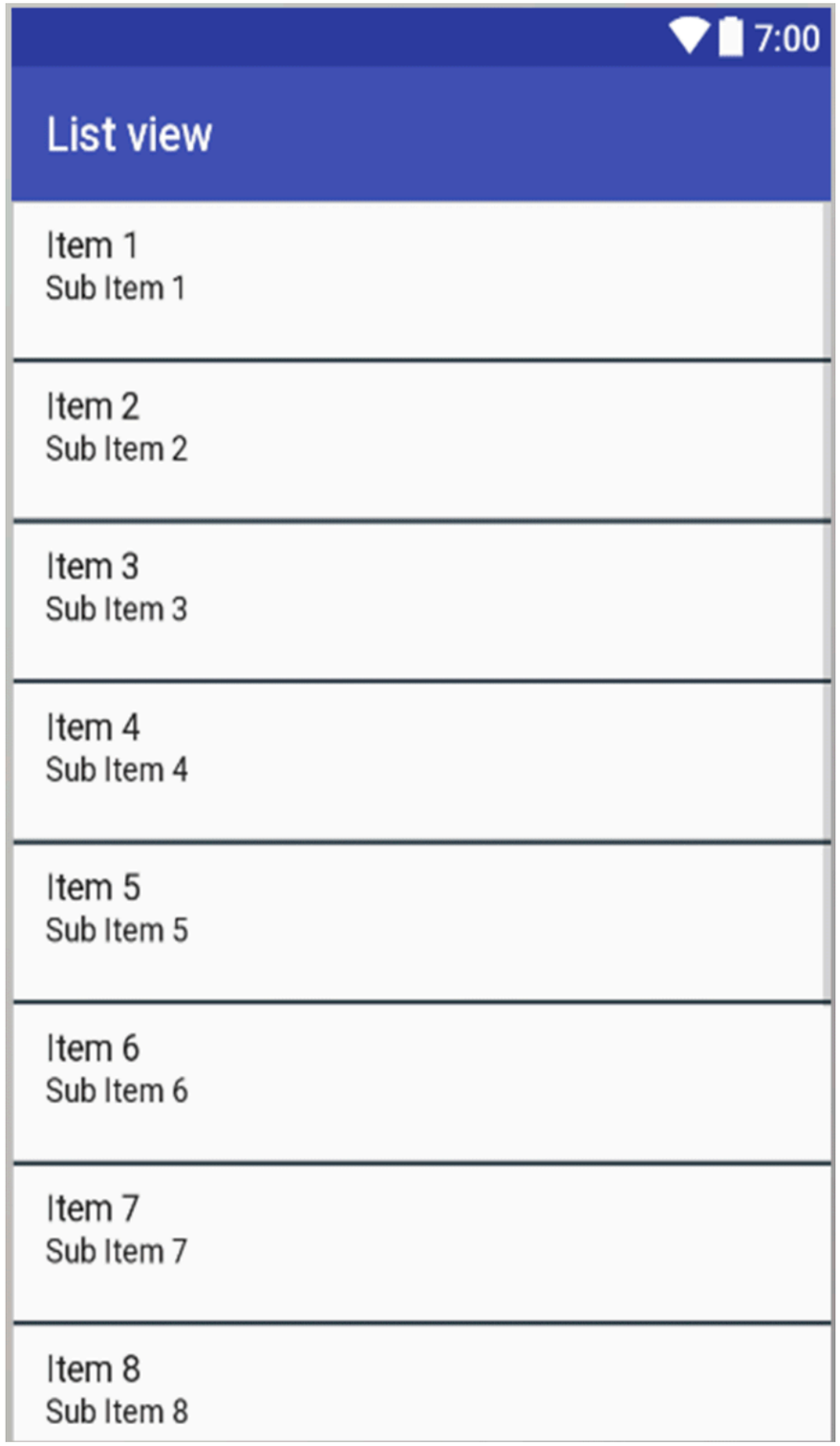


DEMO

ListView

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="..."
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingLeft="8dp"
        android:paddingRight="8dp">
    <ListView android:id="@android:id/myList"
              android:layout_width="match_parent"
              android:layout_height="match_parent"/>
</LinearLayout>
```

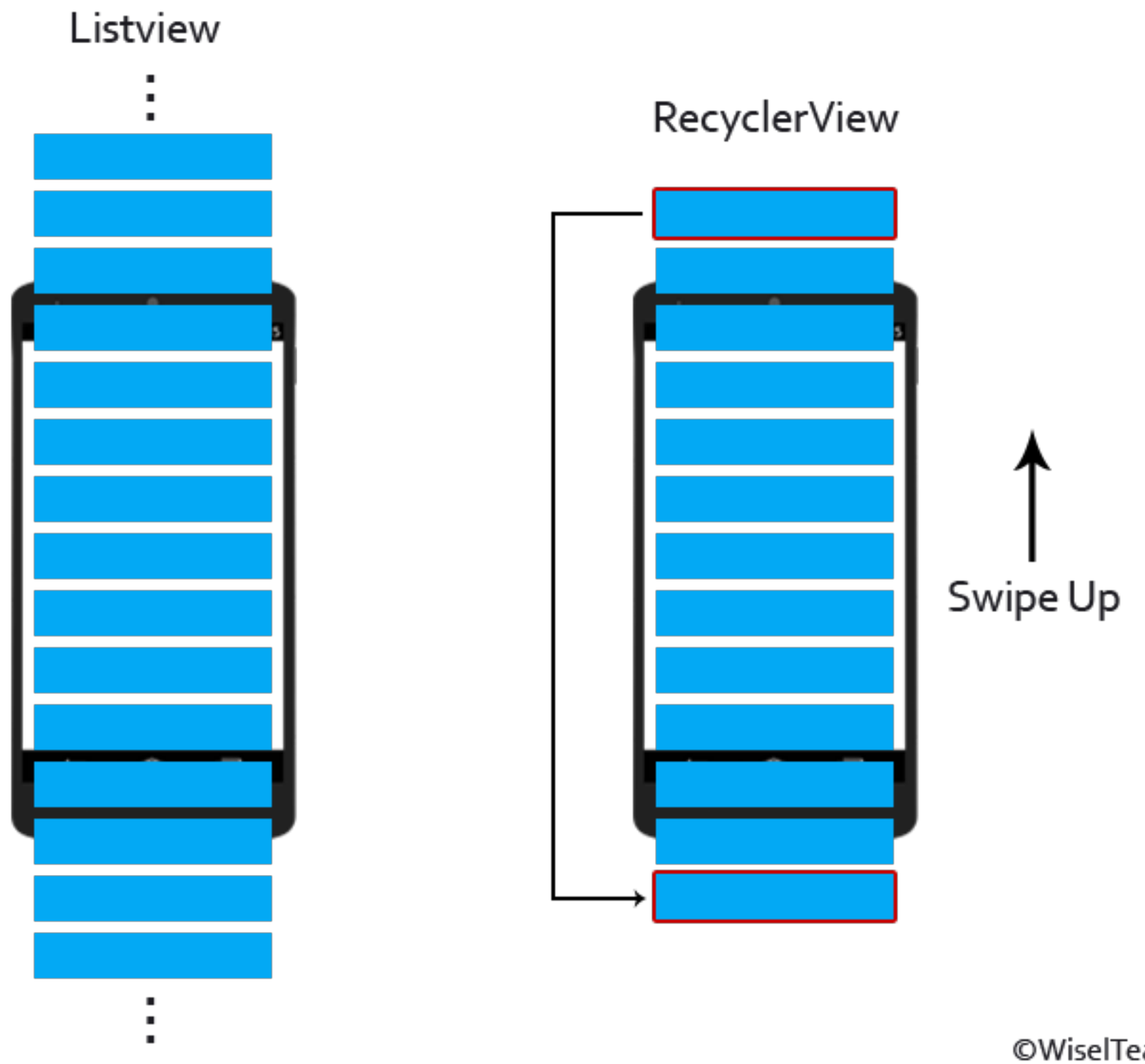
```
val arrayAdapter = ArrayAdapter<String>(this,
  android.R.layout.simple_list_item_1, arrayList)

myList.adapter = arrayAdapter
```



https://developer.android.com/guide/topics/ui/layout/listview

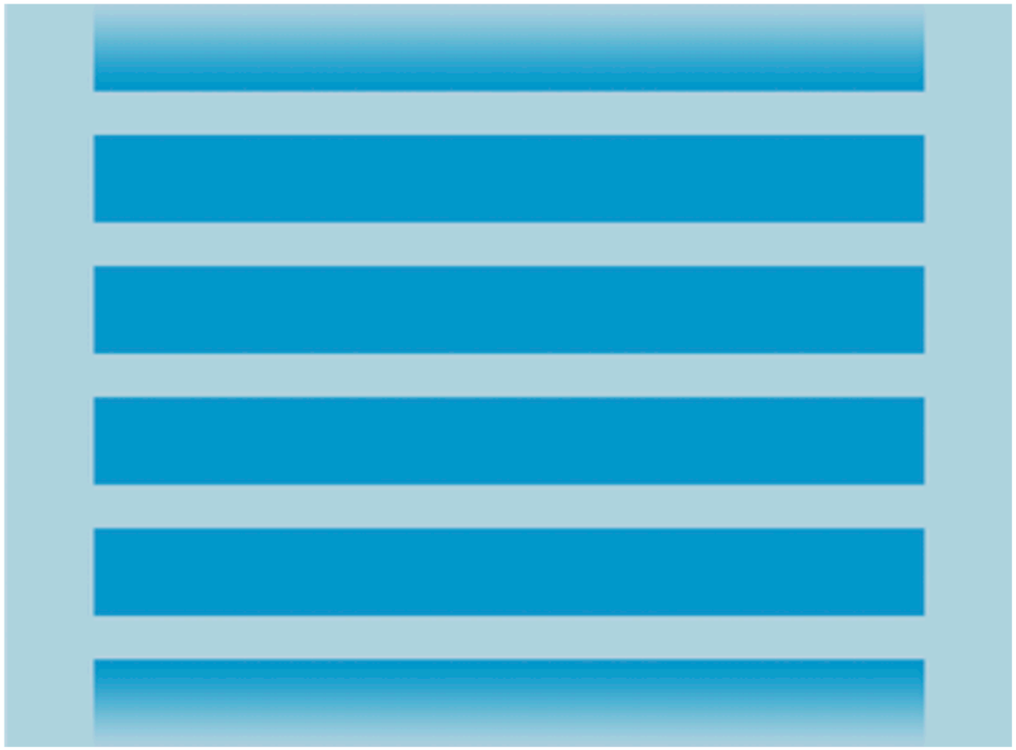
RecyclerView

- ListView
- RecyclerView



RecyclerView

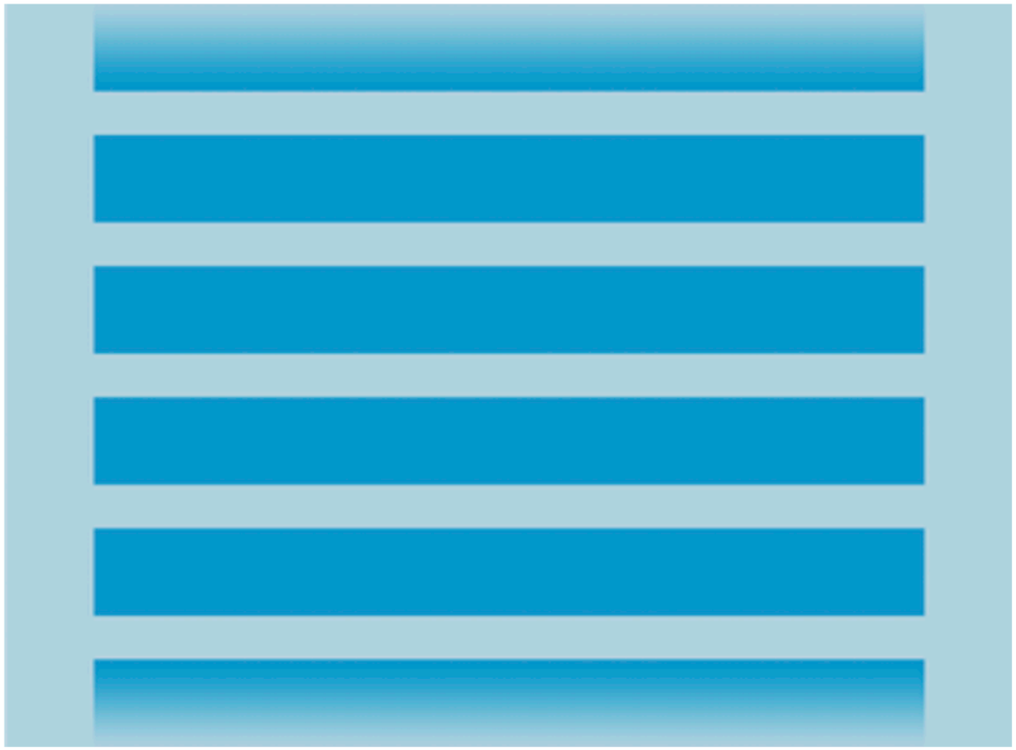
```xml
<?xml version="1.0" encoding="utf-8"?>
<!-- A RecyclerView with some commonly used attributes -->
<android.support.v7.widget.RecyclerView
    android:id="@+id/my_recycler_view"
    android:scrollbars="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
```



```fun
class MyActivity : Activity() {
    private lateinit var recyclerView: RecyclerView
    private lateinit var viewAdapter: RecyclerView.Adapter<*>
    private lateinit var viewManager: RecyclerView.LayoutManager
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.my_activity)
        viewManager = LinearLayoutManager(this)
        viewAdapter = MyAdapter(myDataset)
        recyclerView = findViewById<RecyclerView>(R.id.my_recycler_view).apply {
            setHasFixedSize(true)
            layoutManager = viewManager
            adapter = viewAdapter
        }
    }
    // ...
}
```

RecyclerView

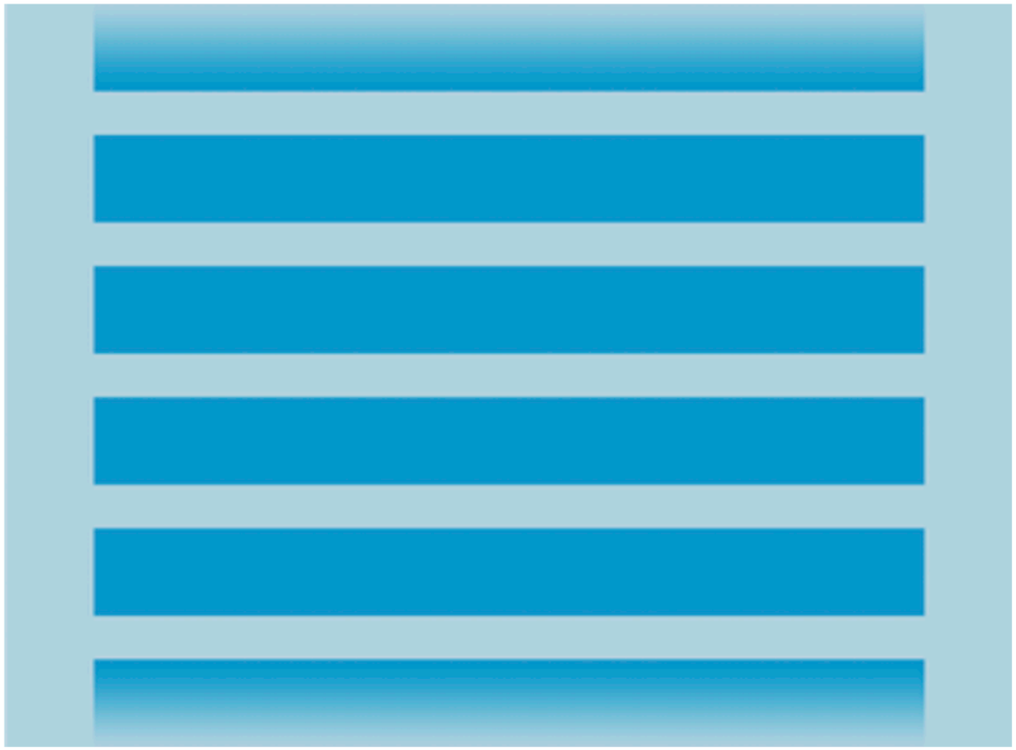
```xml
<?xml version="1.0" encoding="utf-8"?>
<!-- A RecyclerView with some commonly used attributes -->
<android.support.v7.widget.RecyclerView
    android:id="@+id/my_recycler_view"
    android:scrollbars="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
```



```fun
class MyActivity : Activity() {
    private lateinit var recyclerView: RecyclerView
    private lateinit var viewAdapter: RecyclerView.Adapter<*>
    private lateinit var viewManager: RecyclerView.LayoutManager
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.my_activity)
        viewManager = LinearLayoutManager(this)
        viewAdapter = MyAdapter(myDataset)
        recyclerView = findViewById<RecyclerView>(R.id.my_recycler_view).apply {
            setHasFixedSize(true)
            layoutManager = viewManager
            adapter = viewAdapter
        }
    }
    // ...
}
```

RecyclerView

```
<?xml version="1.0" encoding="utf-8"?>
<!-- A RecyclerView with some commonly used attributes -->
<android.support.v7.widget.RecyclerView
    android:id="@+id/my_recycler_view"
    android:scrollbars="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"/>
```



```
class MyActivity : Activity() {
    private lateinit var recyclerView: RecyclerView
    private lateinit var viewAdapter: RecyclerView.Adapter<*>
    private lateinit var viewManager: RecyclerView.LayoutManager
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.my_activity)
        viewManager = LinearLayoutManager(this)
        viewAdapter = MyAdapter(myDataset)
        recyclerView = findViewById<RecyclerView>(R.id.my_recycler_view).apply {
            setHasFixedSize(true)
            layoutManager = viewManager
            adapter = viewAdapter
        }
    }
    // ...
}
```

https://developer.android.com/guide/topics/ui/layout/recyclerview

RecyclerView.Adapter

```
class MyAdapter(private val myDataset: Array<String>) :
        RecyclerView.Adapter<MyAdapter.MyViewHolder>() {

    class MyViewHolder(val textView: TextView) : RecyclerView.ViewHolder(textView)

    override fun onCreateViewHolder(parent: ViewGroup,
                                    viewType: Int): MyAdapter.MyViewHolder {
        val textView = LayoutInflater.from(parent.context)
                .inflate(R.layout.my_text_view, parent, false) as TextView
        ...
        return MyViewHolder(textView)
    }

    // Replace the contents of a view (invoked by the layout manager)
    override fun onBindViewHolder(holder: MyViewHolder, position: Int) {
        // - get element from your dataset at this position
        // - replace the contents of the view with that element
        holder.textView.text = myDataset[position]
    }

    override fun getItemCount() = myDataset.size
}
```

DEMO

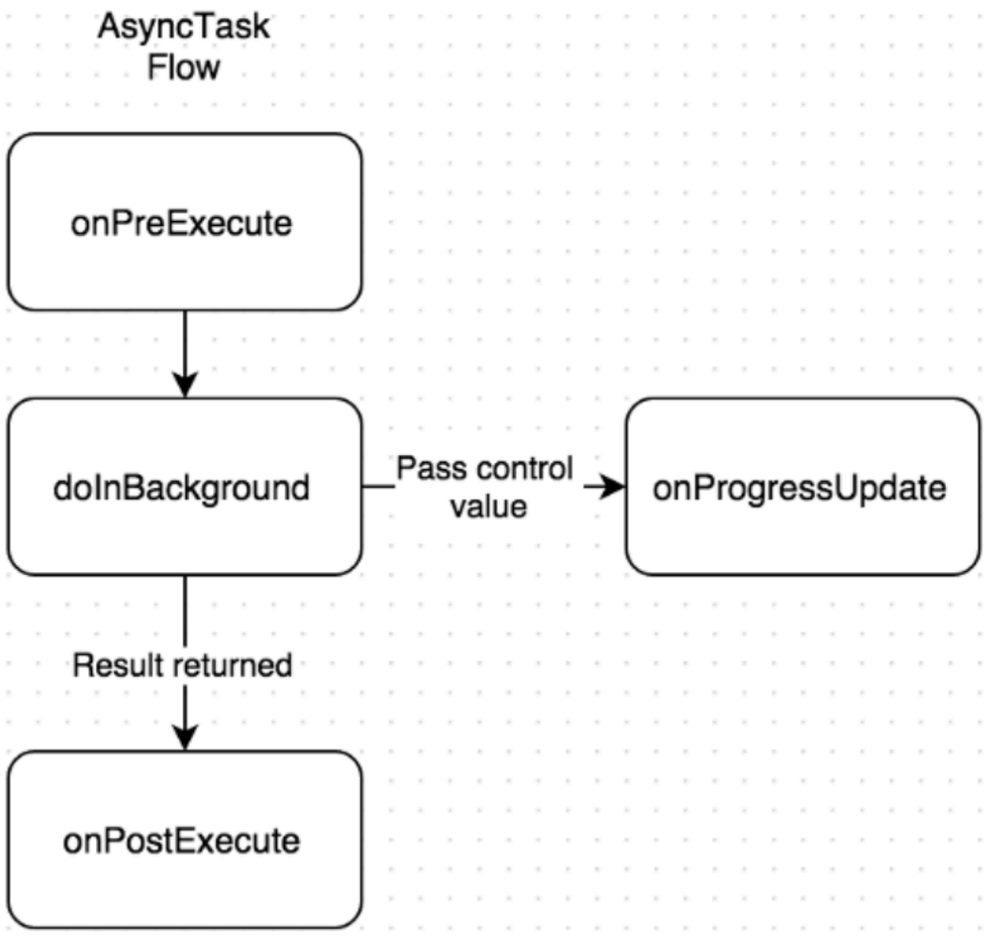
RecyclerView.Adapter

```
class MyAdapter(private val myDataset: Array<String>) :
        RecyclerView.Adapter<MyAdapter.MyViewHolder>() {

    class MyViewHolder(val textView: TextView) : RecyclerView.ViewHolder(textView)

    override fun onCreateViewHolder(parent: ViewGroup,
                                    viewType: Int): MyAdapter.MyViewHolder {
        val textView = LayoutInflater.from(parent.context)
                .inflate(R.layout.my_text_view, parent, false) as TextView
        ...
        return MyViewHolder(textView)
    }

    // Replace the contents of a view (invoked by the layout manager)
    override fun onBindViewHolder(holder: MyViewHolder, position: Int) {
        // - get element from your dataset at this position
        // - replace the contents of the view with that element
        holder.textView.text = myDataset[position]
    }

    override fun getItemCount() = myDataset.size
}
```

https://developer.android.com/guide/topics/ui/layout/recyclerview

AsyncTask

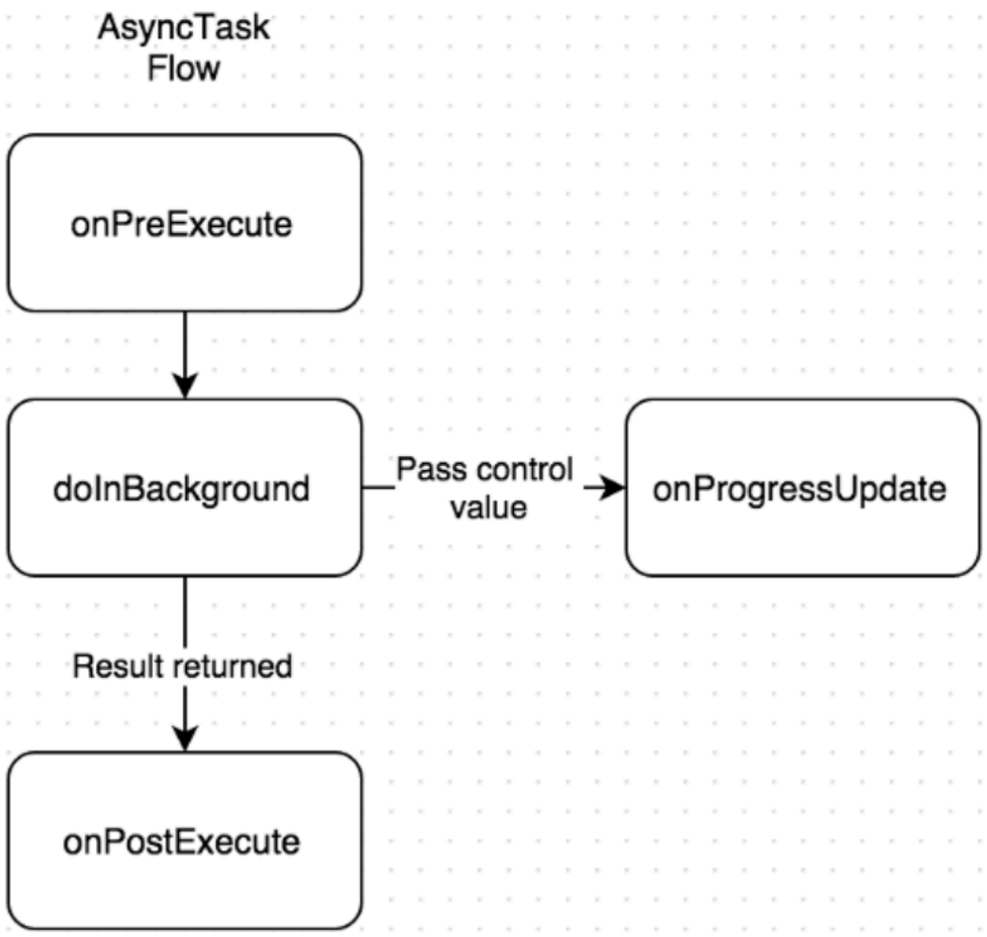
```
class SomeTask():
    AsyncTask<Void, Int, String>() {
 override fun doInBackground(
    vararg params: Void?): String? {
   // ...
 }
 override fun onPreExecute() {
   super.onPreExecute()
   // ...
 }
 override fun onPostExecute(
    result: String?) {
   super.onPostExecute(result)
   // ...
 }
 override fun onProgressUpdate(
    vararg values: Int){
   super.onProgressUpdate(result)
   // ...
 }
}
```



AsyncTask

Deprecated

```
class SomeTask():
    AsyncTask<Void, Int, String>() {
 override fun doInBackground(
    vararg params: Void?): String? {
   // ...
 }
 override fun onPreExecute() {
   super.onPreExecute()
   // ...
 }
 override fun onPostExecute(
    result: String?) {
   super.onPostExecute(result)
   // ...
 }
 override fun onProgressUpdate(
    vararg values: Int){
   super.onProgressUpdate(result)
   // ...
 }
}
```

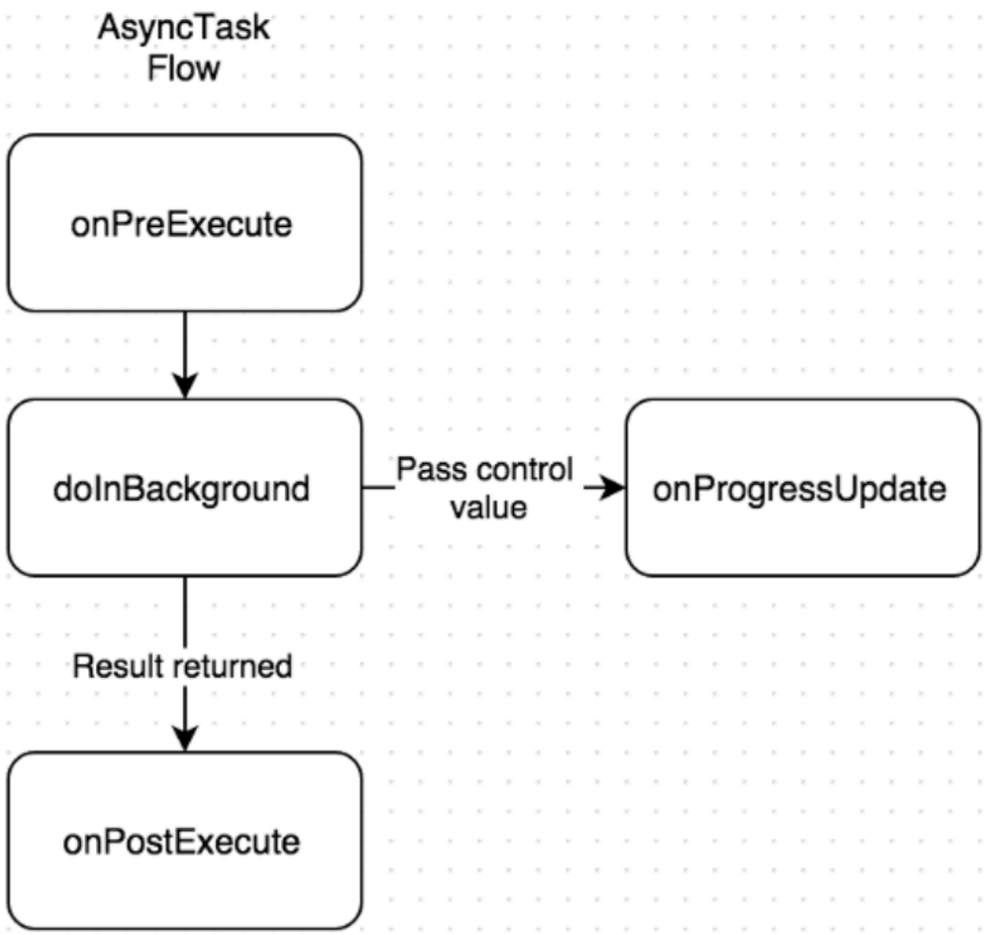


AsyncTask

Deprecated

BackgroundThread

```
class SomeTask():
    AsyncTask<Void, Int, String>() {
 override fun doInBackground(
    vararg params: Void?): String? {
   // ...
 }
 override fun onPreExecute() {
   super.onPreExecute()
   // ...
 }
 override fun onPostExecute(
    result: String?) {
   super.onPostExecute(result)
   // ...
 }
 override fun onProgressUpdate(
    vararg values: Int){
   super.onProgressUpdate(result)
   // ...
 }
}
```

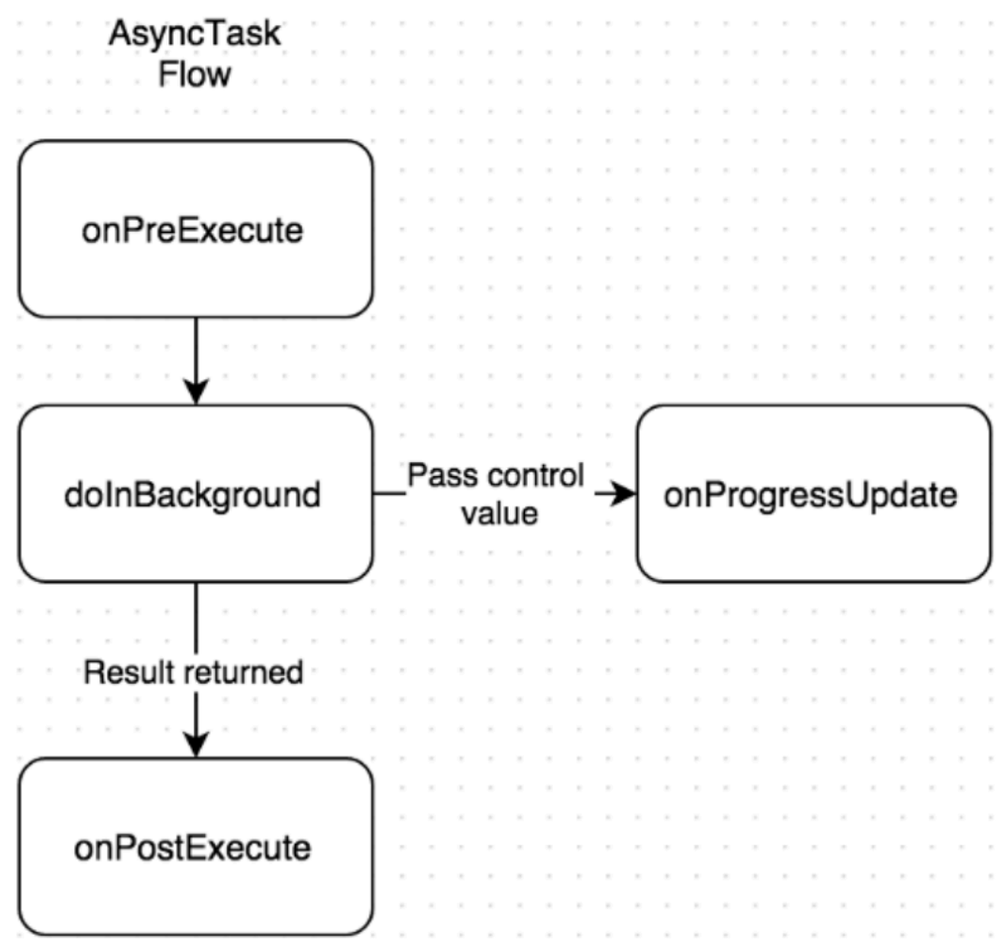


AsyncTask

Deprecated

BackgroundThread

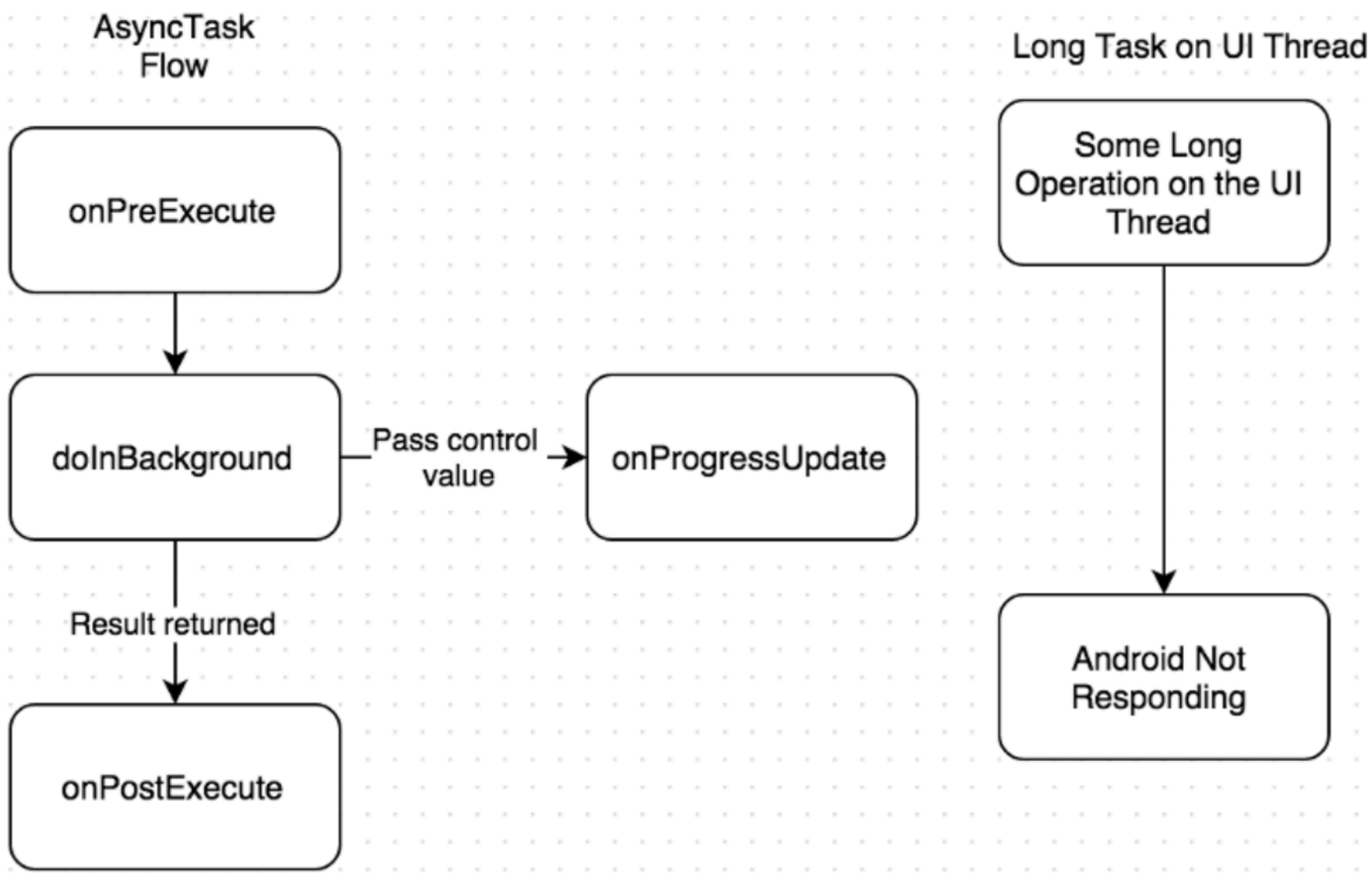
```
class SomeTask():
    AsyncTask<Void, Int, String>() {
 override fun doInBackground(
    vararg params: Void?): String? {
   // ...
 }
 override fun onPreExecute() {
   super.onPreExecute()
   // ...
 }
 override fun onPostExecute(
    result: String?) {
   super.onPostExecute(result)
   // ...
 }
 override fun onProgressUpdate(
    vararg values: Int){
   super.onProgressUpdate(result)
   // ...
 }
}
```



UiThread

AsyncTask

Deprecated



https://developer.android.com/reference/android/os/AsyncTask

Anko AsyncTask Alternative

```
doAsync {
  //Execute all the long running
  // tasks here
  val s: String = makeNetworkCall()
  uiThread {
    //Update the UI thread here
   alert("Downloaded data is $s",
      "Hi I'm an alert") {
    yesButton { toast("Yay !") }
    noButton { toast(":( !") }
   }.show()
  }
}
```

Anko AsyncTask Alternative

BackgroundThread

```
doAsync {
  //Execute all the long running
  // tasks here
  val s: String = makeNetworkCall()
  uiThread {
    //Update the UI thread here
   alert("Downloaded data is $s",
      "Hi I'm an alert") {
    yesButton { toast("Yay !") }
    noButton { toast(":( !") }
   }.show()
  }
}
```

Anko AsyncTask Alternative

BackgroundThread

```
doAsync {
  //Execute all the long running
  // tasks here
  val s: String = makeNetworkCall()
  uiThread {
    //Update the UI thread here
   alert("Downloaded data is $s",
      "Hi I'm an alert") {
    yesButton { toast("Yay !") }
    noButton { toast(":( !") }
   }.show()
  }
}
```

UiThread

DEMO

Anko AsyncTask Alternative

BackgroundThread

```
doAsync {
  //Execute all the long running
  // tasks here
  val s: String = makeNetworkCall()
  uiThread {
    //Update the UI thread here
   alert("Downloaded data is $s",
      "Hi I'm an alert") {
    yesButton { toast("Yay !") }
    noButton { toast(":( !") }
   }.show()
  }
}
```

UiThread

https://github.com/Kotlin/anko/wiki/Anko-Coroutines

Lecture outcomes

- Support different screen, using layouts and fragments
- ListView, RecyclerView, Progress Indicators
- Retrieve data on background threads