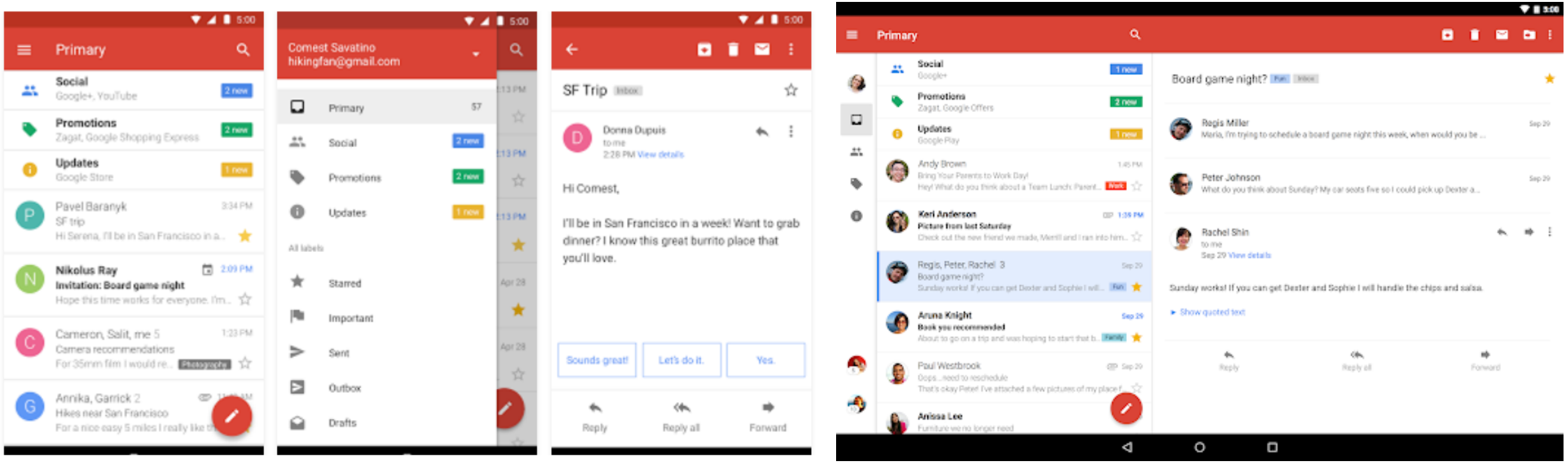
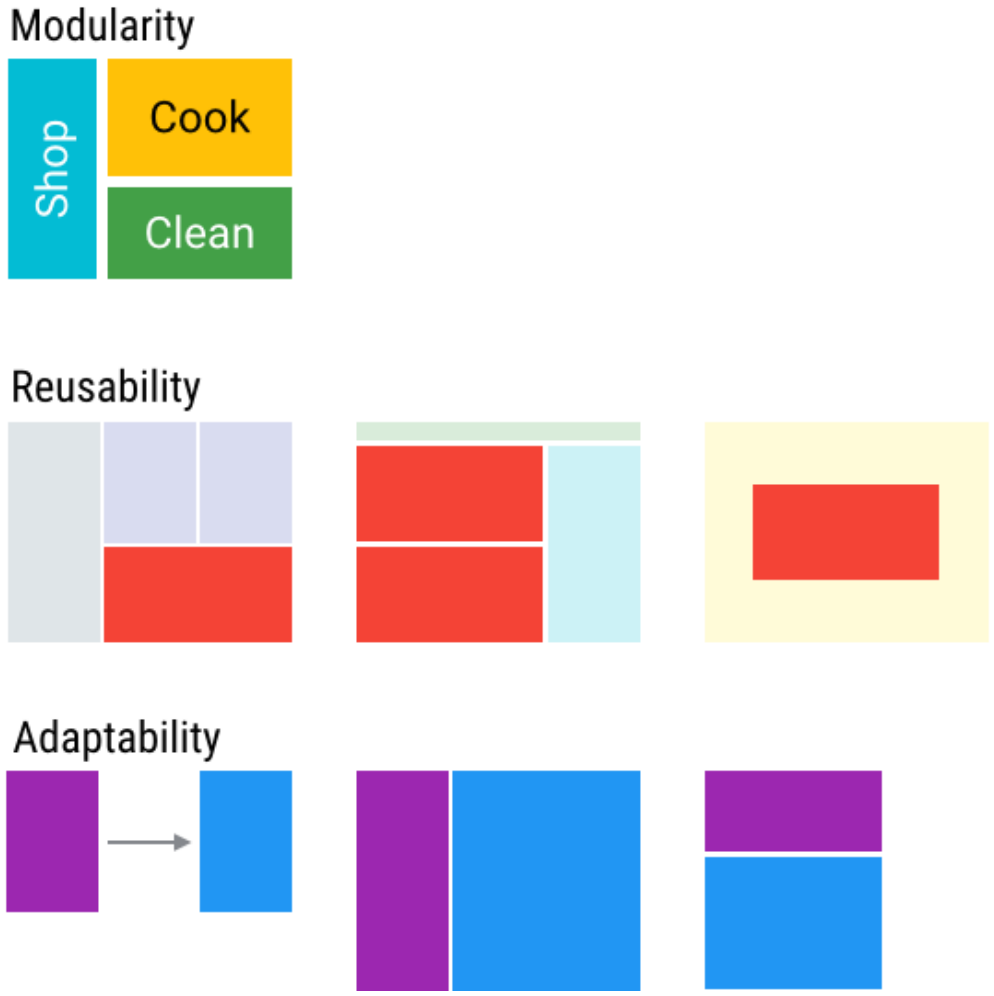
# Fragments

Introduction

# That is where fragments come handy



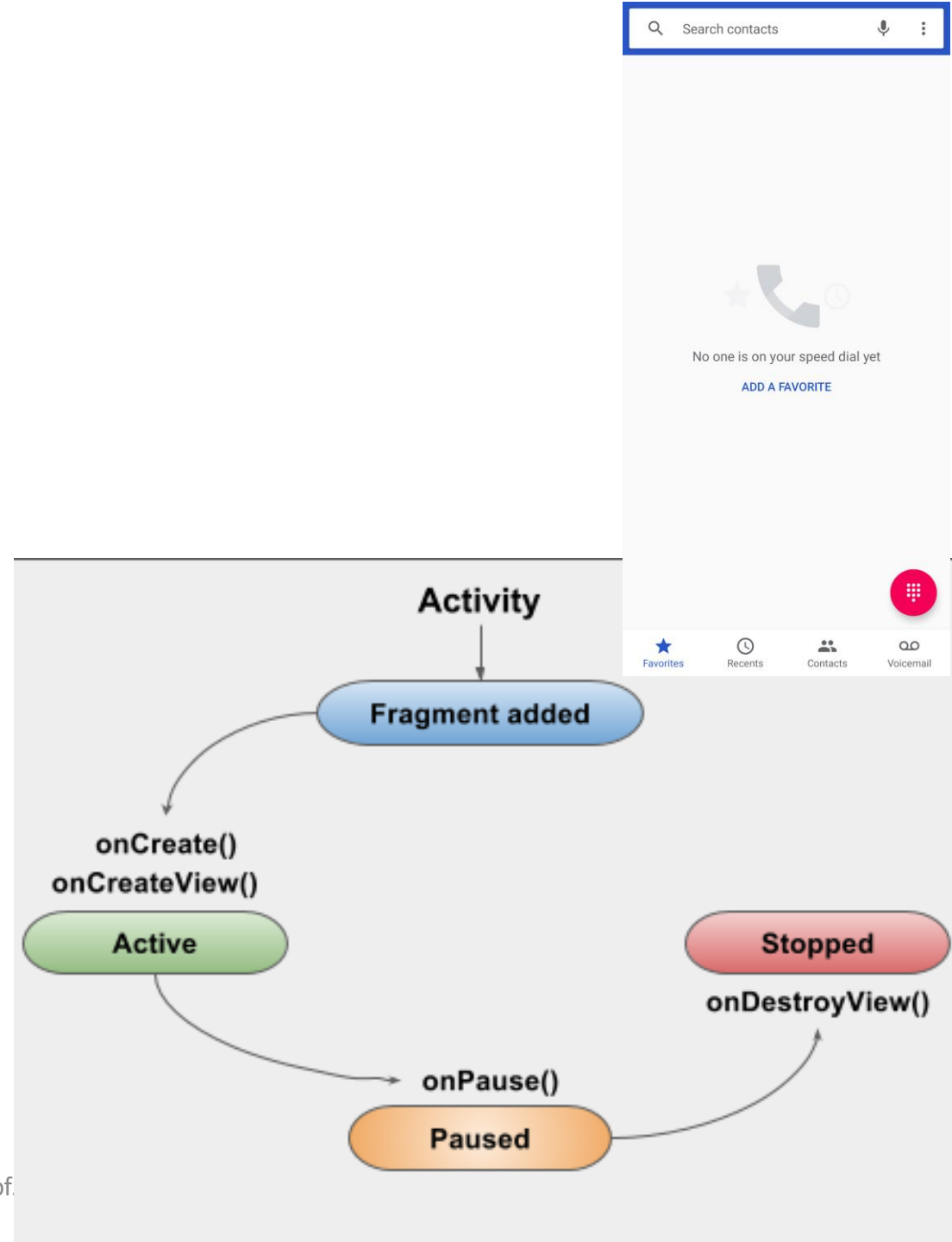
# Fragments can provide…




- ***Modularity***: Dividing complex activity code across fragments for better organization and maintenance.
- ***Reusability***: Placing behavior or UI parts into fragments that multiple activities can share.
- ***Adaptability***: Representing sections of a UI as different fragments and utilizing different layouts depending on screen orientation and size.

# Fragments

- A Fragment represents a behavior or a portion of user interface in an Activity.
- You can combine multiple fragments in a single activity to build a multi-pane UI and reuse a fragment in multiple activities.
- A fragment must always be embedded in an activity and its lifecycle is directly influenced by the host activity's lifecycle.



# Fragments, like Activities

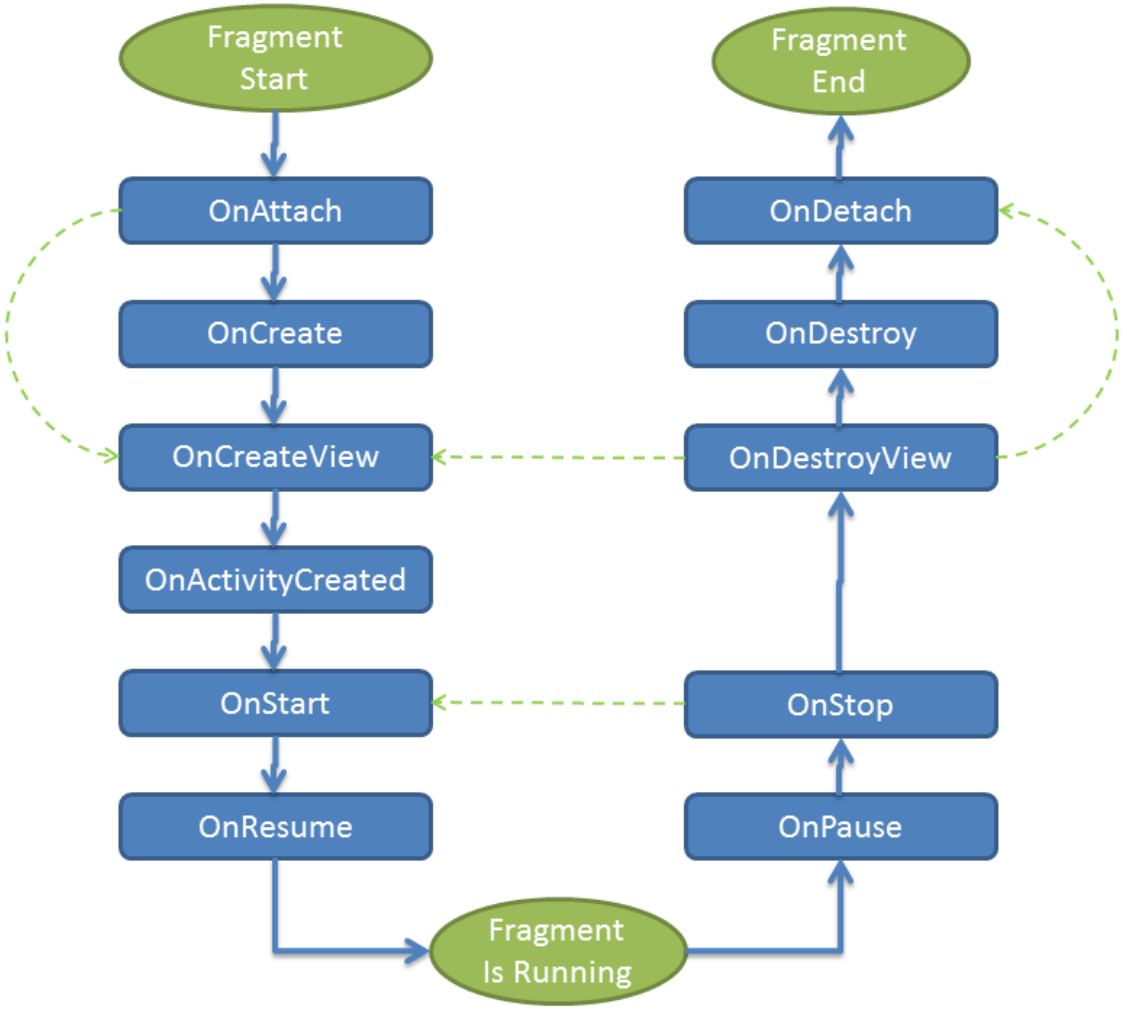
- A Fragment lives as part of the activity layout as a ViewGroup inside the activity's view hierarchy.
- Fragments define their **own** view **layout**.
- You declare fragments in the activity's layout file, as a <fragment> element, or from your application code by adding it to an existing ViewGroup.

```
<fragment
    android:id="@+id/fragmentbook
    android:name="net.workingdev.
    android:layout_width="match_p
    android:layout_height="0px"
    android:layout_weight="1" />

</LinearLayout>
```

```
<!-- res/layout/example_activity.xml -->
<androidx.fragment.app.FragmentContainerView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/fragment_container_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

# Fragments Lifecycle

- A fragment is created subclassing the **Fragment** class (or an existing subclass of it).
- The Fragment class has code that looks a lot like an Activity.
- It contains callback methods similar to an activity:
  - onCreate(),
  - onStart(),
  - onPause(), and
  - onStop().

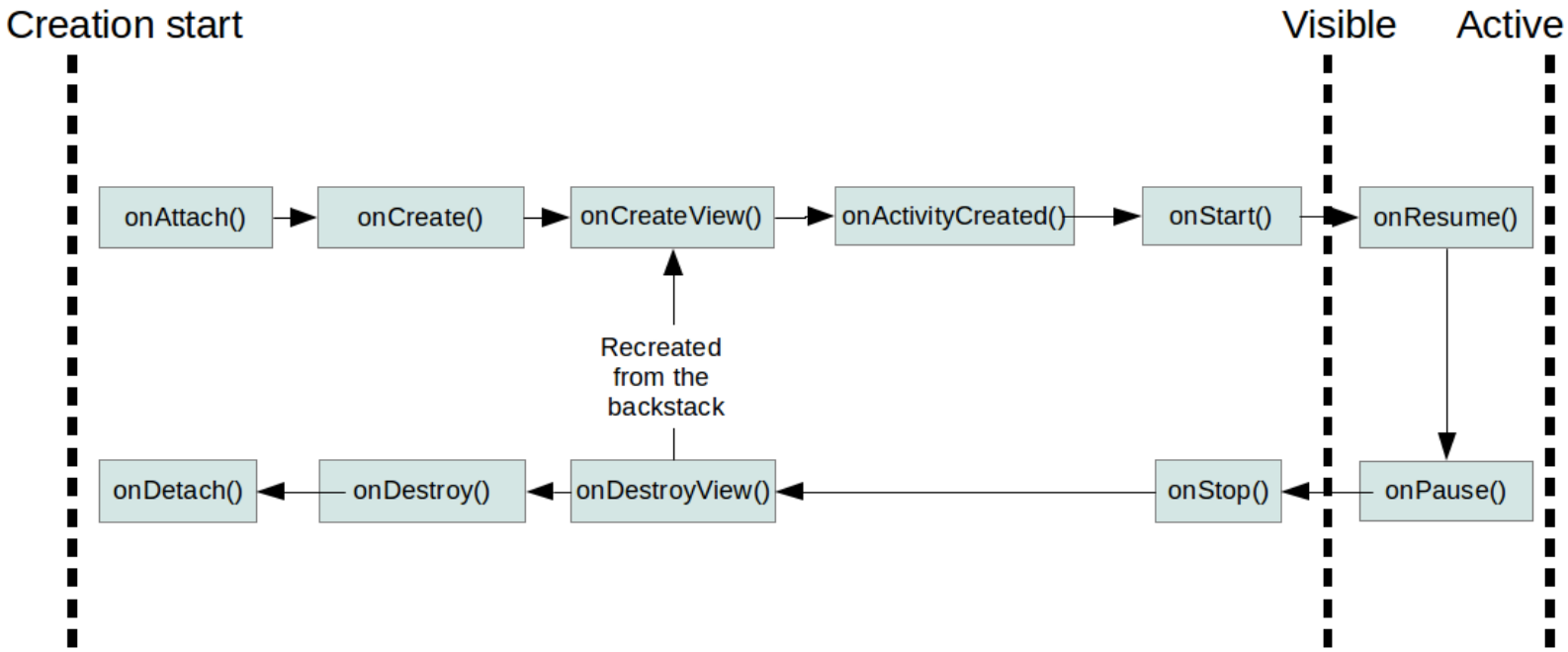


# Fragment lifecycle


| Method | Description |
|---|---|
| onAttach() | It is called when the fragment has been associated with an activity. |
| onCreate() | It is used to initialize the fragment. |
| onCreteView() | It is used to create a view hierarchy associated with the fragment. |
| onActivityCreated() | It is called when the fragment activity has been created and the fragment view hierarchy instantiated. |
| onStart() | It is used to make the fragment visible. |
| onResume() | It is used to make the fragment visible in an activity. |
| onPause() | It is called when fragment is no longer visible and it indicates that the user is leaving the fragment. |
| onStop() | It is called to stop the fragment using the onStop() method. |
| onDestoryView() | The view hierarchy associated with the fragment is being removed after executing this method. |
| onDestroy() | It is called to perform a final clean up of the fragments state. |
| onDetach() | It is called immediately after the fragment disassociated from the activity. |

# Fragment lifecycle

- The life-cycle of a fragment is connected to the life-cycle of its hosting activity.
- A fragment has its own life cycle. But it is always connected to the life cycle of the activity which uses the fragment.



# Creating Fragments

To create a Fragment, we generally do the following:

- Create an XML resource file
- Create the Fragment class
- Inflate the XML resource file in the onCreateView method of the Fragment class.
- Add the newly created Fragment.

res/layout/**fragment1.xml**

class Fragment1: **Fragment**() {...}

override fun **onCreateView**() ...
inflater.inflate(R.layout.fragment1, ...)

# Create a fragment class

- **extend** the AndroidX Fragment class

```
import androidx.fragment.app.Fragment

class ExampleFragment : Fragment()
```

- **override** its methods to insert your app logic

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

# Add a fragment via XML

- To declaratively add a fragment to your activity layout's XML, use a FragmentContainerView element.

```
<!-- res/layout/example_activity.xml -->
<androidx.fragment.app.FragmentContainerView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/fragment_container_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:name="com.example.ExampleFragment" />
```

# Add a fragment programmatically

- the layout should include a FragmentContainerView

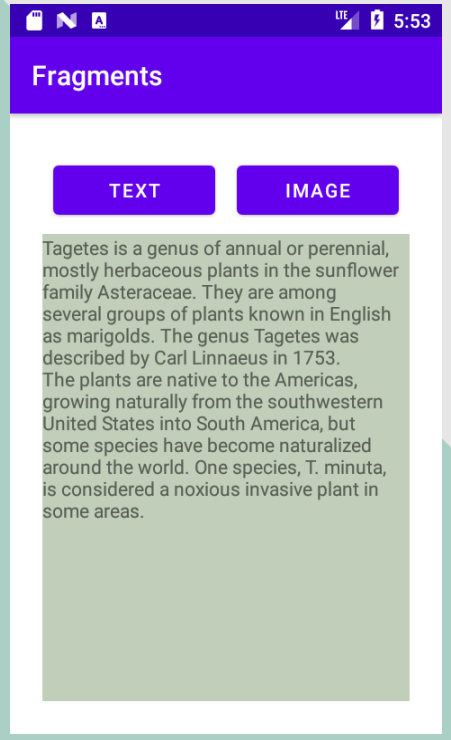
```xml
<!-- res/layout/example_activity.xml -->
<androidx.fragment.app.FragmentContainerView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/fragment_container_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

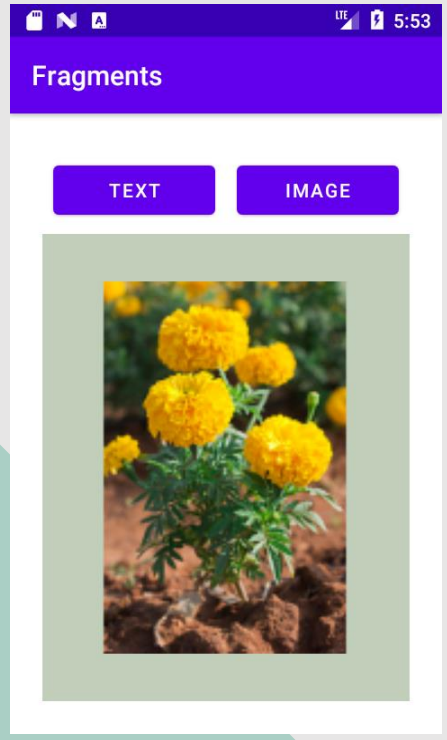
- a FragmentTransaction is used to instantiate a fragment and add it to the activity's layout.

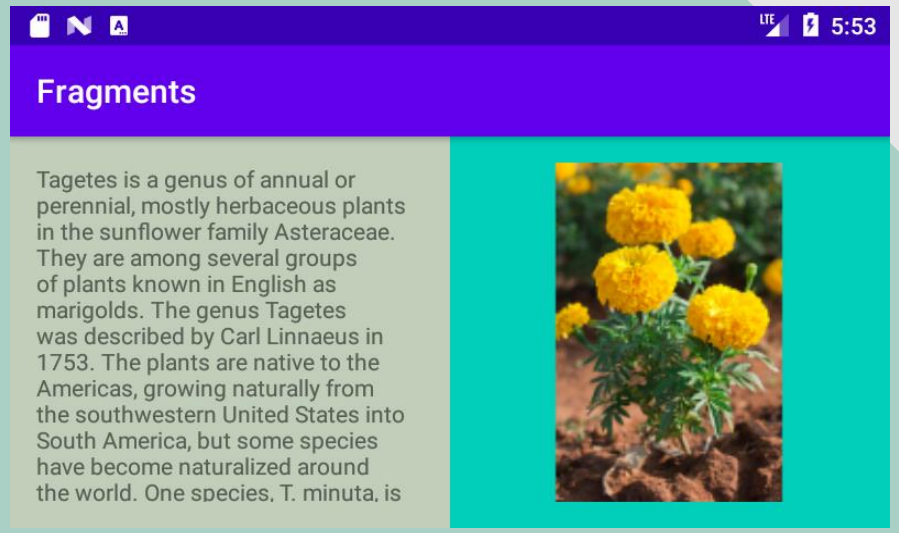
```
val transaction = supportFragmentManager.beginTransaction()
transaction.replace(R.id.fragmentContainerView, textFragment)
transaction.commit()
```

# Fragment Transaction

- At runtime, a **FragmentManager** can add, remove, replace, and perform other actions with fragments in response to user interaction.
- Use replace() to replace an existing fragment in a container with an instance of a new fragment class that you provide. Calling replace() is equivalent to calling remove() with a fragment in a container and adding a new fragment to that same container.



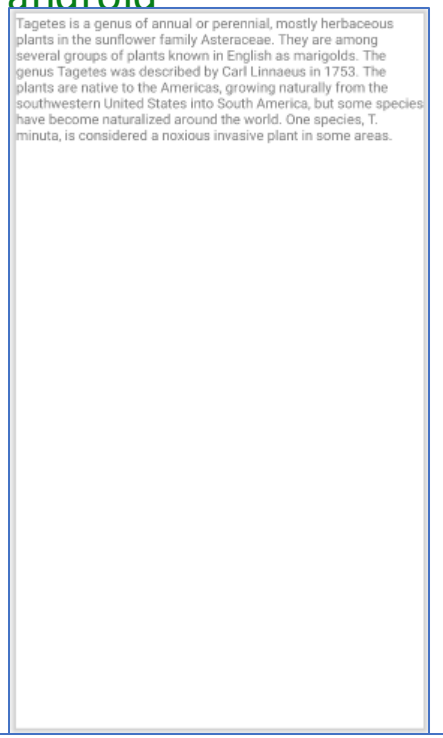




# Example

Using two Fragments

# Create Fragment for text visualization

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".TextInfoFragment">

<TextView
android:id="@+id/textView"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:text="@string/tagete_description" />

</FrameLayout>
```




```
import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.fragment.app.Fragment

class TextInfoFragment : Fragment() {
  override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle? ): View? {
    // Get the custom view for this fragment layout
    return inflater!!.inflate(R.layout.fragment_text_info,container,false)
  }
}
```

# Create Fragment for Html visualization

```
override fun onStart() {
  super.onStart()
  val tagete = "<h1>Tagete</h1>" +
  "Sono piante erbacee che hanno mediamente un'altezza che varia dai 25 " +
  "agli 80 cm, ma ci sono specie che possono raggiungere anche " +
  "i 3 metri[senza fonte]; il fusto è cespuglioso e ramificato, " +
  "le foglie sono pennate, lucide e di colore verde scuro; " +
  "i fiori sono dei capolini semplici o doppi, di colore giallo, " +
  "arancio o rosso."
  tvDesc.text = Html.fromHtml(tagete)
}
```

# Create Fragment for image visualization

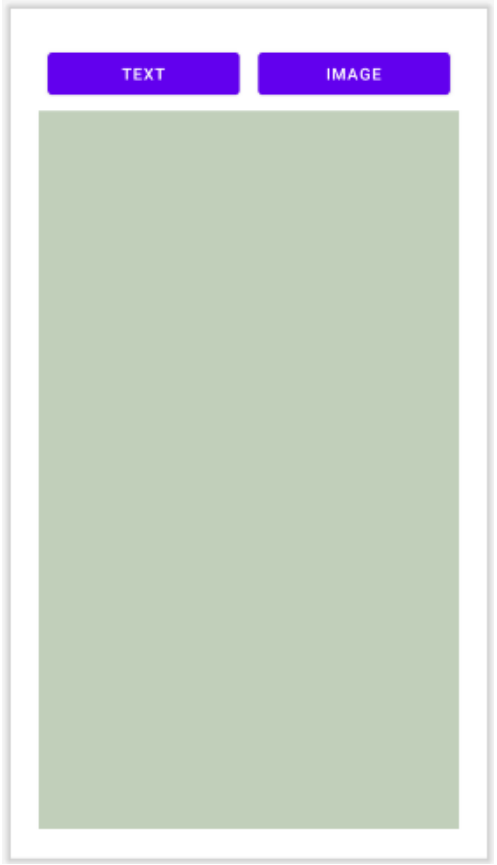
```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".ImageInfoFragment">

<ImageView
android:id="@+id/imageView"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:src="@mipmap/ic_tagete_foreground" />

</FrameLayout>
```

```
import android.os.Bundle
import androidx.fragment.app.Fragment
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup

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

```
class FragmentActivity : AppCompatActivity() {
    private val imageFragment = ImageFragment()
    private val textFragment = TextFragment()
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_fragment)

        btnText.setOnClickListener{ it: View!
            val transaction = supportFragmentManager.beginTransaction()
            transaction.replace(R.id.fragmentContainerView, textFragment)
            transaction.commit()
        }

        btnImage.setOnClickListener { it: View!
            val transaction = supportFragmentManager.beginTransaction()
            transaction.replace(R.id.fragmentContainerView, imageFragment)
            transaction.commit()
        }
    }
}
```

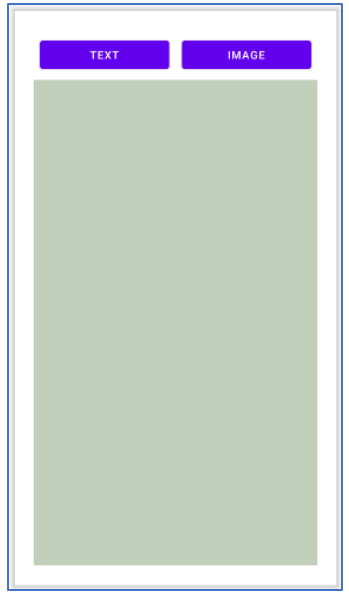
transaction.addToBackStack(null)

# Main Activity



# activity_main.xml

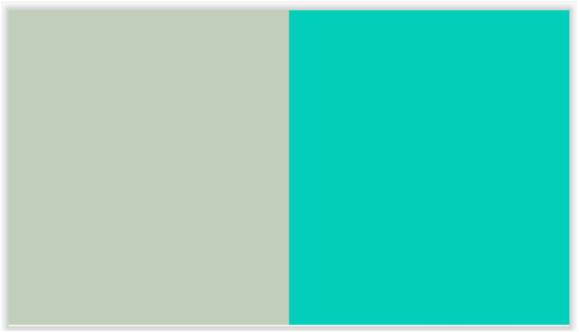
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity"
android:orientation="vertical"
android:layout_margin="24dp">

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

<Button
android:id="@+id/button1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:layout_margin="8dp"
android:text="Text" />

<Button
android:id="@+id/button2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:layout_margin="8dp"
android:text="Image" />
</LinearLayout>

<androidx.fragment.app.FragmentContainerView
android:id="@+id/fragmentContainerView"
android:name="it.uninsubria.pdm.fragtmp.TextInfoFragment"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:background="#c1cfba" />

</LinearLayout>
```



# activity_main.xml landscape

```
<?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/linearLayout"
android:layout_width="match_parent"
android:layout_height="match_parent">

<androidx.fragment.app.FragmentContainerView
android:id="@+id/fragment_text_container"
android:name="it.uninsubria.pdm.fragtmp.TextInfoFragment"
android:layout_width="0dp"
android:layout_height="0dp"
android:background="#c1cfba"
android:padding="16dp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toStartOf="@+id/fragment_image_container"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="@+id/fragment_image_container" />
...
</androidx.constraintlayout.widget.ConstraintLayout>
```

```
<androidx.fragment.app.FragmentContainerView
android:id="@+id/fragment_image_container"
android:name="it.uninsubria.pdm.fragtmp.ImageInfoFragment"
android:layout_width="0dp"
android:layout_height="match_parent"
android:background="#01cfba"
android:padding="16dp"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toEndOf="@+id/fragment_text_container"
/>
```



# Swipe Fragments Left/Right

```
override fun onTouch(view: View, motionEvent: MotionEvent): Boolean{
    when(motionEvent.action) {
        MotionEvent.ACTION_DOWN -> {
            xs = motionEvent.x
            ys = motionEvent.y
            return true
        }
        MotionEvent.ACTION_UP -> {
            val xdiff = xs - motionEvent.x
            val ydiff = ys - motionEvent.y
            if (Math.abs(xdiff) > Math.abs(ydiff)) { // horizontal
                if (xdiff > SWIPE_THRESHOLD) {// right
                    rightFragment()
                } else if (xdiff < -SWIPE_THRESHOLD)  { // left
                    leftFragment()
                }
                return true
            }
        }
        return false
    }
```

```
private fun leftFragment() {
    val transaction = supportFragmentManager.beginTransaction()
    transaction.replace(R.id.fragmentContainerView, imageFragment)
    // transaction.addToBackStack(null)
    transaction.commit()
}
private fun rightFragment() {
    val transaction = supportFragmentManager.beginTransaction()
    transaction.replace(R.id.fragmentContainerView, textFragment)
    transaction.commit()
}
```

```
private fun isOrientationPortrait() : Boolean{
    val orientation = resources.configuration.orientation
    if (orientation == Configuration.ORIENTATION_LANDSCAPE) return false
    return true // In portrait
}
```

```
if (isOrientationPortrait()) {
    btnText.setOnClickListener {  it: View!
        rightFragment()
    }
    btnImage.setOnClickListener {  it: View!
        leftFragment()
    }
    clFragments.setOnTouchListener(this)
}
```