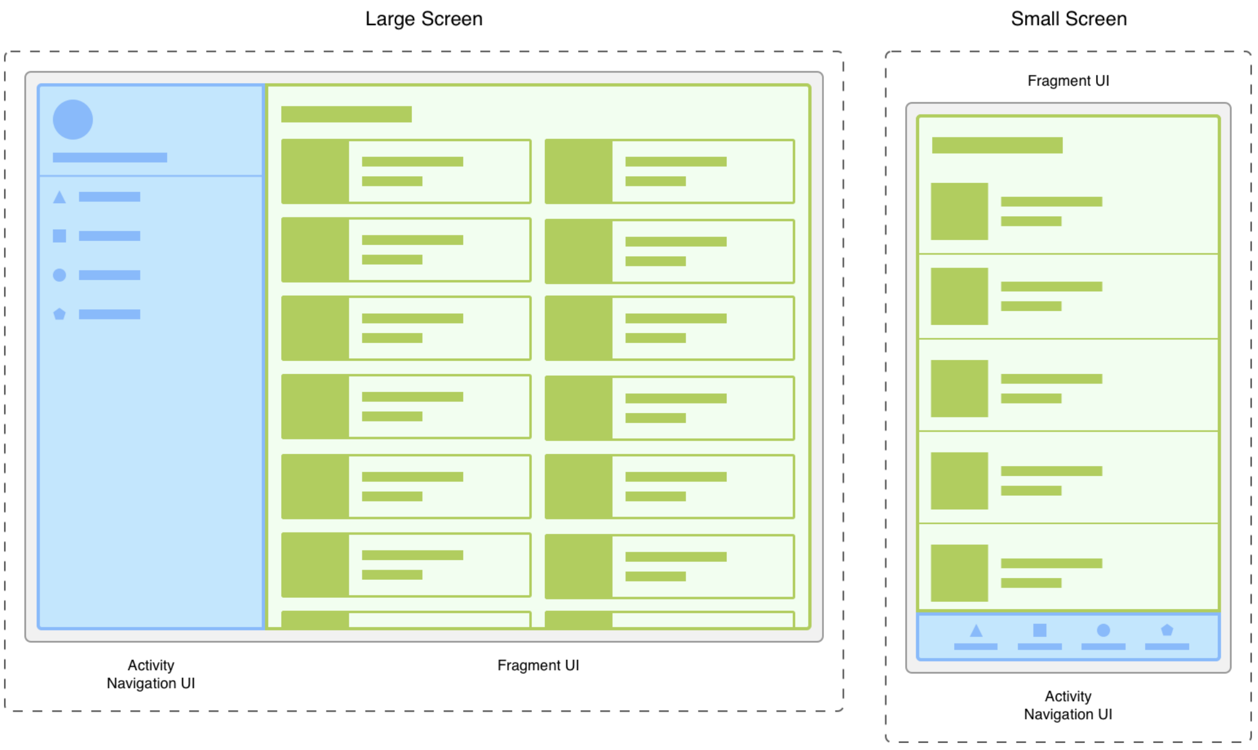
13. Develop a program to implement horizontal and vertical fragments.

A Fragment represents a reusable portion of an app's UI. A fragment defines and manages its own layout, has its own lifecycle, and can handle its own input events. Fragments cannot live on their own--they must be *hosted* by an activity or another fragment. The fragment's view hierarchy becomes part of, or *attaches to*, the host's view hierarchy. Fragment is a kind of sub-activity.

Fragments introduce modularity and reusability into your activity's UI by allowing you to divide the UI into discrete chunks. Activities are an ideal place to put global elements around an app's user interface, such as a navigation drawer. Conversely, fragments are better suited to define and manage the UI of a single screen or portion of a screen.

Consider an app that responds to various screen sizes. On larger screens, the app should display a static navigation drawer and a list in a grid layout. On smaller screens, the app should display a bottom navigation bar and a list in a linear layout. Managing all of these variations in the activity can be unwieldy. Separating the navigation elements from the content can make this process more manageable. The activity is then responsible for displaying the correct navigation UI while the fragment displays the list with the proper layout.



Two versions of the same screen on different screen sizes. On the left, a large screen contains a navigation drawer that is controlled by the activity and a grid list that is controlled by the fragment. On the right, a small screen contains a bottom navigation bar that is controlled by the activity and a linear list that is controlled by the fragment.

Dividing an UI into fragments makes it easier to modify an activity's appearance at runtime. While an activity is in the STARTED lifecyle state or higher, fragments can be added, replaced, or removed. Keep a record of these changes in a back stack that is managed by the activity, allowing the changes to be reversed.

Use multiple instances of the same fragment class within the same activity, in multiple activities, or even as a child of another fragment. Provide a fragment with the logic necessary to manage its own UI. Avoid depending on or manipulating one fragment from another.

Managing Fragments with FragmentManager

The FragmentManager class is part of the Activity. Use it to initialize a Fragment, add Fragments to the Activities layout, and to end a Fragment. An Activity has one FragmentManager, but it can take care of many fragments. This is just what is needed to have multiple behaviors and layouts within a single app.

FragmentManager also calls the various lifecycle methods of the fragments it is responsible for. This is distinct from the Activity lifecycle methods, which are called by Android, yet closely related because the FragmentManager calls many of the Fragment lifecycle methods in response to the Activity lifecycle methods being called.