❖ Android Fragment

- ✓ Fragment class in Android is used to build dynamic User Interfaces.
- ✓ **Fragment** should be used within the Activity.
- ✓ A greatest advantage of fragments is that it simplifies the task of creating UI for multiple screen sizes.
- ✓ A activity can contain any number of fragments.
- ✓ An Android fragment is not by itself a subclass of View which most other UI components are. Instead, a fragment has a view inside it.
- ✓ It is this view which is eventually displayed inside the activity in which the fragment lives.
- ✓ Because an android fragment is not a view, adding it to an activity looks somewhat different than adding a view (e.g. TextView).
- ✓ A fragment is added to a **ViewGroup** inside the activity.
- ✓ The fragment's view is displayed inside this ViewGroup.

❖ Fragment Lifecycle

Below are the methods of fragment lifecycle.

1. onAttach() :This method will be called first, even before onCreate(), letting us know that your fragment has been attached to an activity. You are passed the Activity that will host your fragment

2. onCreateView() : The system calls this callback when it's time for the fragment to draw its UI for the first time. To draw a UI for the fragment, a View component must be returned from this method which is the root of the fragment's layout. We can return null if the fragment does not provide a UI

3. onViewCreated() : This will be called after onCreateView(). This is particularly useful when inheriting the onCreateView() implementation but we need to configure the resulting views, such as with a ListFragment and when to set up an adapter

4. onActivityCreated() :This will be called after onCreate() and onCreateView(), to indicate that the activity's onCreate() has completed. If there is something that's needed to be initialised in the fragment that depends upon the activity's onCreate() having completed its work then onActivityCreated() can be used for that initialisation work

5. onStart() : The onStart() method is called once the fragment gets visible

6. onPause() : The system calls this method as the first indication that the user is leaving the fragment. This is usually where you should commit any changes that should be persisted beyond the current user session

7. onStop() : Fragment going to be stopped by calling onStop()

8. onDestroyView() : It's called before onDestroy(). This is the counterpart to onCreateView() where we set up the UI. If there are things that are needed to be cleaned up specific to the UI, then that logic can be put up in onDestroyView()

9. onDestroy() : onDestroy() called to do final clean up of the fragment's state but Not guaranteed to be called by the Android platform.

10. onDetach() : It's called after onDestroy(), to notify that the fragment has been disassociated from its hosting activity

Android fragment lifecycle is illustrated in below image.

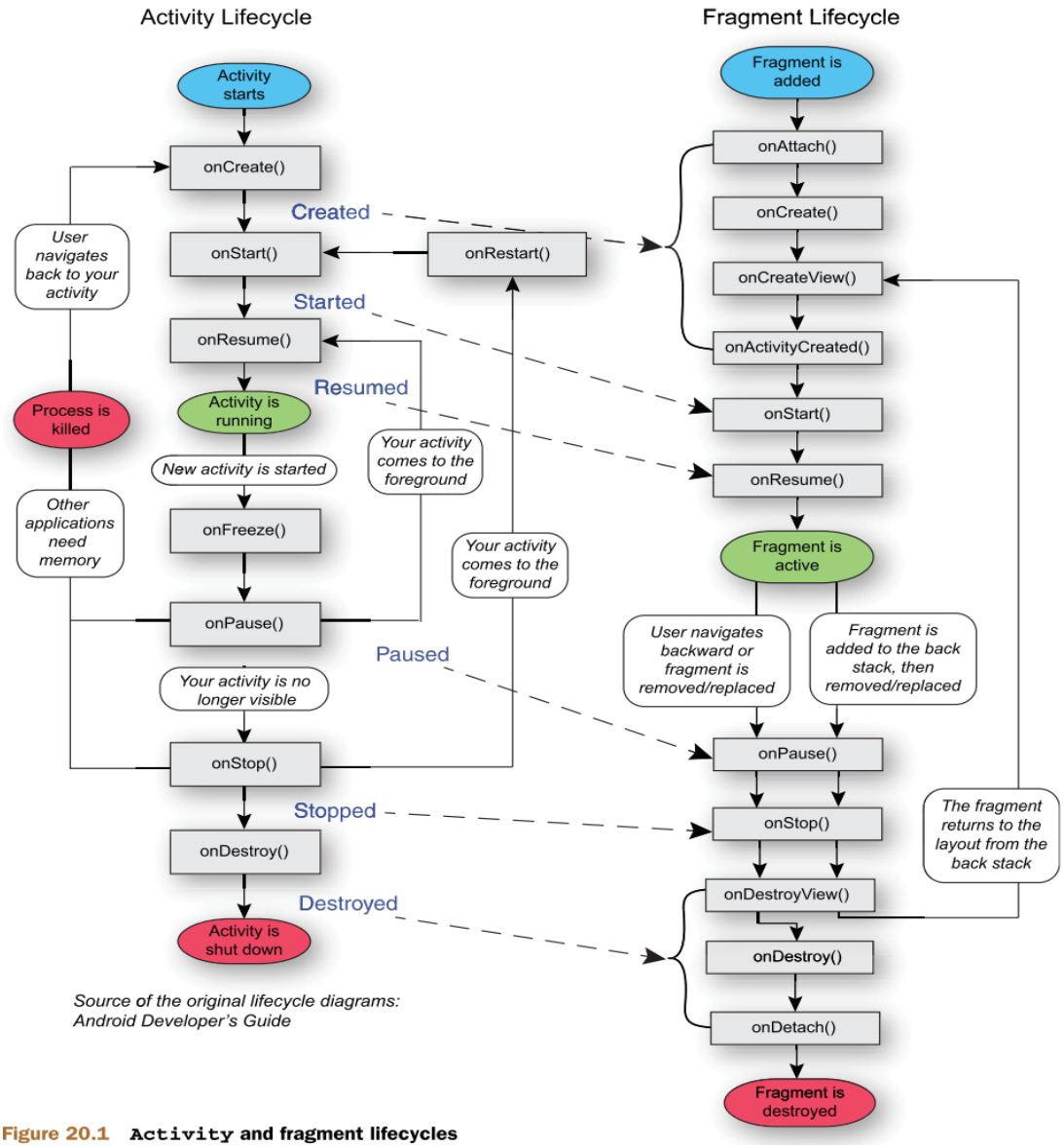

Figure 20.1 `Activity` and fragment lifecycles