**Fragment Lifecycle:**

In Android development, a fragment is a modular section of an activity that has its own lifecycle, similar to an activity. Understanding the fragment lifecycle is crucial for managing the UI components and handling various events. Here's an overview of the Android fragment lifecycle:

1. \*\*onAttach():\*\*

- Called when the fragment is attached to its parent activity.

- The `onAttach()` method receives the context of the activity as a parameter.

2. \*\*onCreate():\*\*

- Called to initialize the fragment. This is where you should initialize essential components.

- This is also the place to retrieve arguments passed to the fragment using `getArguments()`.

3. \*\*onCreateView():\*\*

- Called to create the view hierarchy associated with the fragment.

- In this method, you inflate the fragment's layout and initialize UI components.

4. \*\*onActivityCreated():\*\*

- Called when the parent activity's `onCreate()` method has completed.

- At this point, the fragment's view hierarchy is guaranteed to be initialized.

5. \*\*onViewStateRestored():\*\*

- Called after `onActivityCreated()` when the fragment's state has been restored.

6. \*\*onStart():\*\*

- Called when the fragment becomes visible to the user.

- This is a good place to perform tasks that should be done every time the fragment becomes visible.

7. \*\*onResume():\*\*

- Called when the fragment gains focus and is actively running.

- This is where you should register any broadcast receivers or perform other tasks that require an active state.

8. \*\*onPause():\*\*

- Called when the fragment is no longer in the foreground or is about to be replaced.

- Perform operations like saving data or cleaning up resources that should be done when the fragment is not actively running.

9. \*\*onStop():\*\*

- Called when the fragment is no longer visible to the user.

- This is a good place to save persistent data or release resources.

10. \*\*onDestroyView():\*\*

- Called when the view hierarchy associated with the fragment is being removed.

- This is where you should clean up resources associated with the fragment's UI.

11. \*\*onDestroy():\*\*

- Called when the fragment is no longer in use.

- Perform final cleanup, such as releasing resources.

12. \*\*onDetach():\*\*

- Called when the fragment is detached from its parent activity.

- This is the last method called in the fragment lifecycle.

It's essential to override these methods in your fragment to handle lifecycle events appropriately and ensure proper resource management. Understanding the fragment lifecycle is crucial for building robust and responsive Android applications.

fragment lifecycle states and their relation both the fragment's
            lifecycle callbacks and the fragment's view lifecycle