OrientationChanges.md 7/13/2020

Orientation Changes

- Recreates Activity and Fragment
 - Android does this so that your application can reload resources based on the new configuration.
- What Not To Do
 - android:screenOrientation
 - Prevents orientation changes from happening
 - android:configChanges="orientation|screenSize|keyboardHidden"
 - Activity and Fragment: onConfigurationChanged()
 - You won't be able to reload resources that may need to be refreshed
 - Manual re-inflate: make your code difficult to work with in the future
 - Fragment.setRetainInstance(true)
 - If Activity that is NOT using the configChanges flag and a Fragment that IS being retained.
 - Android does not call onCreate, onDestroy and the constructor. But it will call all of the other callbacks because the Fragment's parent Activity IS being destroyed and recreated
 - If Activity HAS the configChanges flag set and your Fragment IS retained.
 - onConfigurationChanged()
- · Saving State
 - In most cases this involves implementing the onSaveInstanceState() method (this could be in your Activity, Fragment or both). It gets called before onStop().
 - View
 - For a View's state to be saved it MUST have an android:id attribute
 - Call super on Save Instance State () for View
 - Any custom Views, they should contain an implementation of onSaveInstanceState()
 - ListView
 - For Restoring scroll position, save the contents of the adapter (eg: List -> Serializable ArrayList)
- Restoring State
 - Activity
 - onCreate(savedInstanceState)
 - onRestoreInstanceState(savedInstanceState) gets call after onStart() and before onResume()
 - If you want to wait for all of your onCreate initialization to be done before restoring state
 - If you want to allow subclasses to specifically handle restoring state.
 - Fragment
 - onCreate(savedInstanceState)
 - onCreateView(savedInstanceState)
 - onActivityCreated(savedInstanceState)

OrientationChanges.md 7/13/2020

onViewStateRestored(savedInstanceState)

- View
 - For a View's state to be restore it MUST have an android:id attribute
 - Call super.onRestoreInstanceState() for View
 - Any custom Views, they should contain an implementation of onRestoreInstanceState()
 - ListView
 - For Restoring scroll position, restore adapter before it gets its state restored
- Handling AsyncTasks
 - Memory leaks can occur if your AsyncTask holds on to a reference to an Activity or a Fragment.
 - If an AsyncTask has a reference to a now-destroyed Activity or Fragment, the garbage collector won't be able to collect that Activity or Fragment even though it should never be used again.
 - Memory leaks if you have an AsyncTask declared as a non-static inner class of an Activity or Fragment. (AsyncTask will implicitly hold a reference to its parent class)
 - o Creash: View...not attached to window manager
 - Cancelling AsyncTasks in onDestroy()
 - Save the state of your task if needed
 - It is not always a good option to cancel and restart your AsyncTasks whenever an orientation change occurs.
 - Unpleasant user experience
 - Retained Fragment
 - If you use a retained Fragment to host your AsyncTask, you can avoid ever having to restart your tasks.
 - No UI
 - Declare an interface to tell Activity to update UI or anything else.
 - Cast Activity to the interface in onAttach()
 - Reset interface in onDetach() (Won't leak an Activity reference)