Stateful life cycle of stateful widget

#	METHOD	DESCRIPTION
1	createState()	When the Framework is instructed to build a StatefulWidget, it immediately calls createState()
2	mounted is true	When createState creates your state class, a buildContext is assigned to that state. buildContext is the place in the widget tree in which this widget is placed. All widgets have a bool this.mounted property. It is turned true when the buildContext is assigned. It is an error to call setState when a widget is unmounted.
3	initState()	This is the first method called when the widget is created (after the class constructor, of course.) initState is called once and only once. It must call super.initState().
4	didChangeDependencies()	This method is called immediately after initState on the first time the widget is built.
5	build()	This method is called often. It is required, and it must return a Widget.
6	didUpdateWidget(Widget oldWidget)	If the parent widget changes and has to rebuild this widget (because it needs to give it different data), but it's being rebuilt with the same runtimeType, then this method is called. This is because Flutter is re-using the state, which is long-lived. In this case, you may want to initialize some data again, as you would in initState.
7	setState()	This method is called often from the framework itself and from the developer. It's used to notify the framework that data has changed.
8	deactivate()	deactivate() is called when the State object is removed from the tree, but it might be reinserted before the current frame change is finished. This method exists because State objects can be moved from one point in a tree to another.
9	dispose()	dispose() is called when the State object is removed, which is permanent. This method is where you should unsubscribe and cancel all animations, streams, etc.
10	mounted is false	The state object can never remount, and an error will be thrown if setState is called.