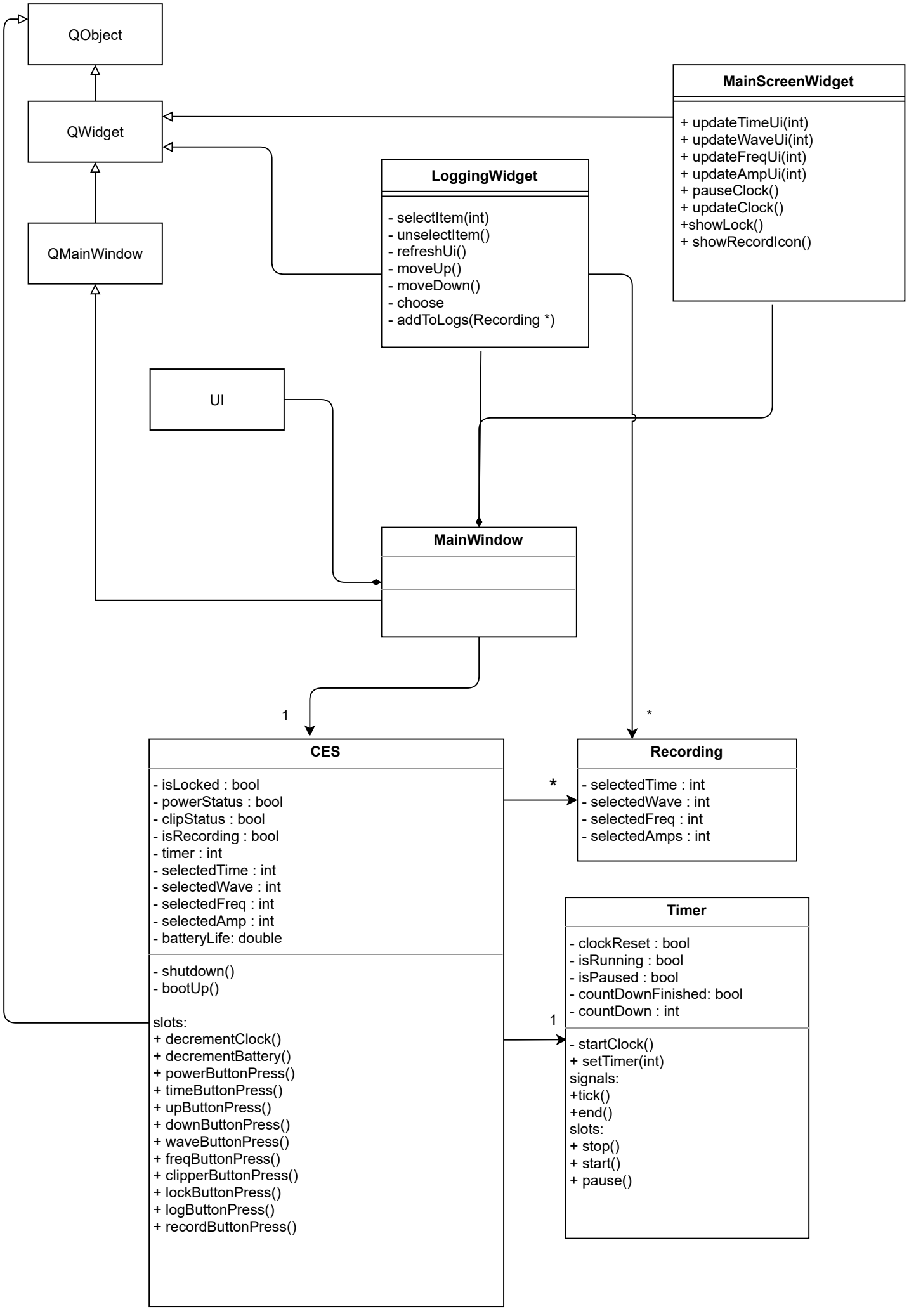
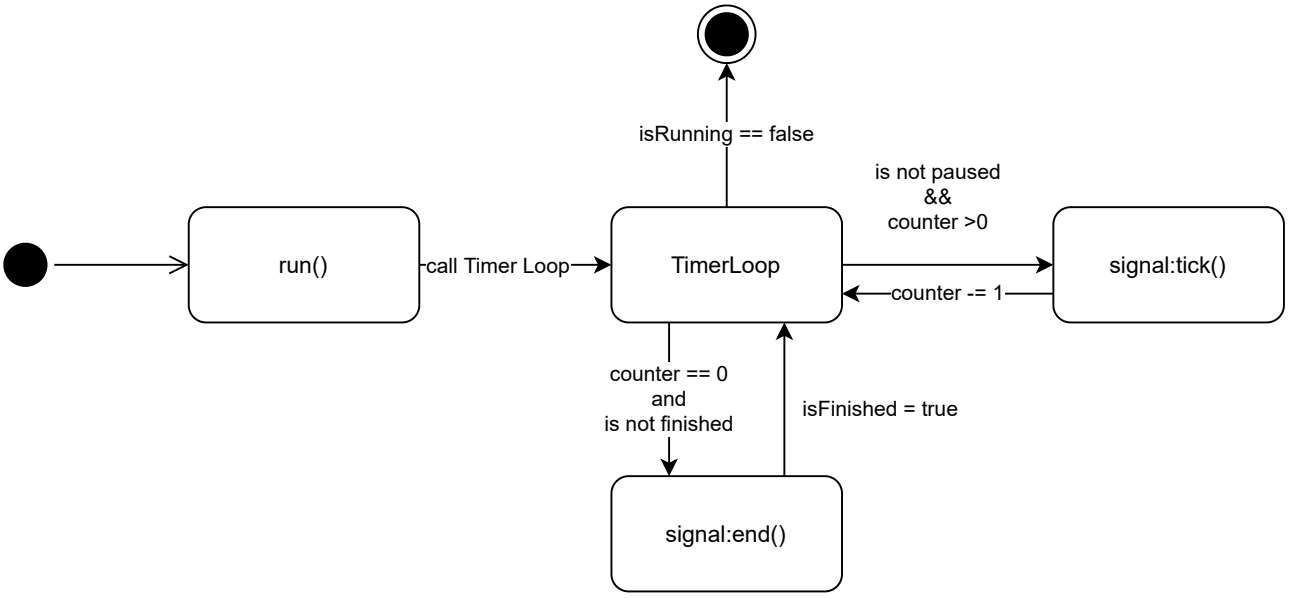

QObject
QWidget
QMainWindow
MainScreenWidget
+ updateTimeUi(int)
+ updateWaveUi(int)
+ updateFreqUi(int)
+ updateAmpUi(int)
+ pauseClock()
+ updateClock()
+showLock()
+ showRecordIcon()
LoggingWidget
- selectItem(int)
- unselectItem()
- refreshUi()
- moveUp()
- moveDown()
- choose
- addToLogs(Recording *)
UI
MainWindow
1
*
CES
- isLocked : bool
- powerStatus : bool
- clipStatus : bool
- isRecording : bool
- timer : int
- selectedTime : int
- selectedWave : int
- selectedFreq : int
- selectedAmp : int
- batteryLife: double
- shutdown()
- bootUp()
slots:
+ decrementClock()
+ decrementBattery()
+ powerButtonPress()
+ timeButtonPress()
+ upButtonPress()
+ downButtonPress()
+ waveButtonPress()
+ freqButtonPress()
+ clipperButtonPress()
+ lockButtonPress()
+ logButtonPress()
+ recordButtonPress()
*
Recording
- selectedTime : int
- selectedWave : int
- selectedFreq : int
- selectedAmps : int
1
Timer
- clockReset : bool
- isRunning : bool
- isPaused : bool
- countDownFinished: bool
- countDown : int
- startClock()
+ setTimer(int)
signals:
+tick()
+end()
slots:
+ stop()
+ start()
+ pause()

State Diagram: Life of a Timer Object

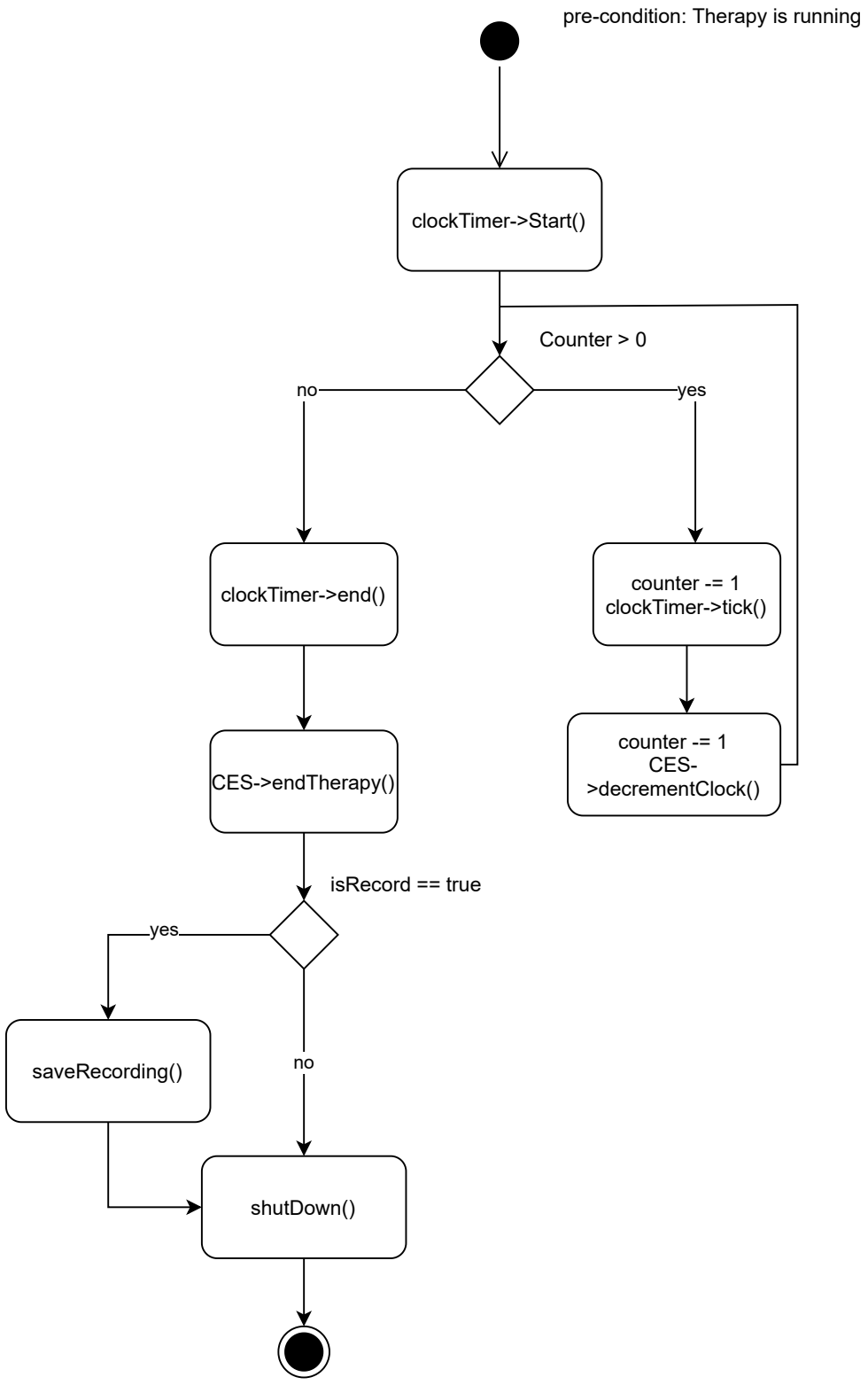


This state diagram shows the life cycle of a Timer Object. When the object is created and set to run with QThreadPool::globalinstance()->start(), it first calls the timerLoop(). At first its default for isNotPaused is true, meaning that the timer remain on a state of waiting for commands.

When the CES calls Timer::start(), it changes isNotPaused to false which begins the flow of sending a tick() signal and decreasing the counter value. If the counter value reaches 0, it sends a end() signal to the CES which the CES can interpret how ever it wants. Because the counter is now 0, the timer goes back to a state of waiting. The counter can be reset with calling Timer::setTime(int) . Timer::pause() will also set the timer to a wating state.

The object is destroyed when the CES call Timer::stop()

Activity Diagram: Therapy Clock



This activity diagram details a timer sequence when a therapy is currently running. At first when the user starts a therapy, it triggers the clock to start. While the counter is over 0, it goes into a loop of tick() which is a signal that calls decrementClock() in the CES. decrementClock() decrements the CES timer by 1 and updates the clock UI accordingly.

Once the counter reaches 0, the clockTimer sends the end() signal which is caught by the endTherapy() function in the CES. The CES then checks if it needs to record the session and then shuts down the device.