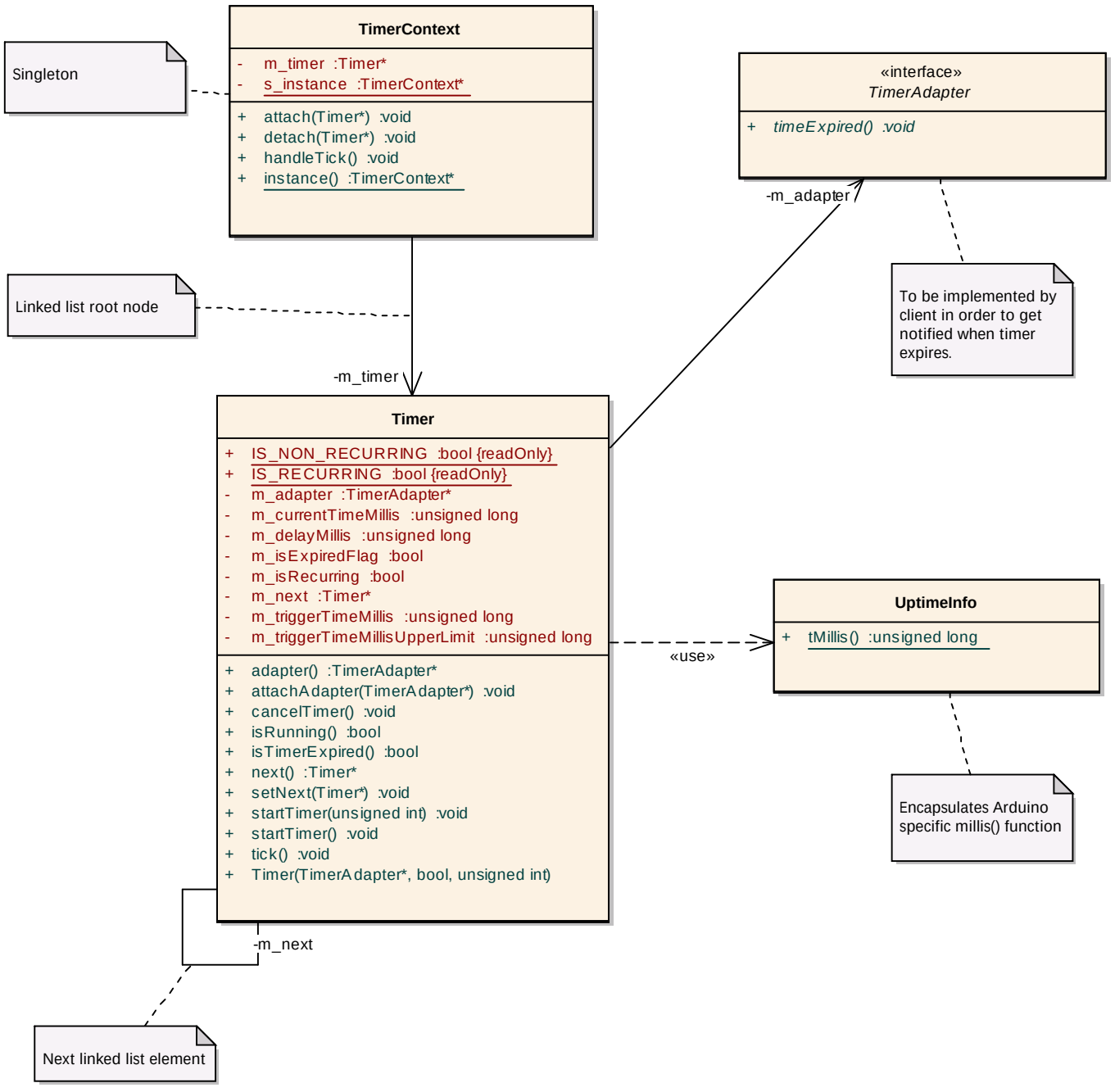
TimerContext
- m_timer :Timer*
- s_instance :TimerContext*
+ attach(Timer*) :void
+ detach(Timer*) :void
+ handleTick() :void
+ instance() :TimerContext*
Singleton
Linked list root node
-m_timer
«interface»
TimerAdapter
+ timeExpired() :void
-m_adapter
To be implemented by client in order to get notified when timer expires.
Timer
+ IS_NON_RECURRING :bool {readOnly}
+ IS_RECURRING :bool {readOnly}
- m_adapter :TimerAdapter*
- m_currentTimeMillis :unsigned long
- m_delayMillis :unsigned long
- m_isExpiredFlag :bool
- m_isRecurring :bool
- m_next :Timer*
- m_triggerTimeMillis :unsigned long
- m_triggerTimeMillisUpperLimit :unsigned long
+ adapter() :TimerAdapter*
+ attachAdapter(TimerAdapter*) :void
+ cancelTimer() :void
+ isRunning() :bool
+ isTimerExpired() :bool
+ next() :Timer*
+ setNext(Timer*) :void
+ startTimer(unsigned int) :void
+ startTimer() :void
+ tick() :void
+ Timer(TimerAdapter*, bool, unsigned int)
«use»
UptimeInfo
+ tMillis() :unsigned long
Encapsulates Arduino specific millis() function
-m_next
Next linked list element