

# MiscStopwatch

**Inherits From:** MiscTime

**Declared In:** misckit/MiscStopwatch.h

## Class Description

The MiscStopwatch class is meant to be used to track the length of real time events. All you have to do to use it, after creating an instance, is to send `±startTiming:` and `±pauseTiming:` messages when appropriate. If you need to know the elapsed time while the MiscStopwatch is running, then send a `±calcElapsedTime:` message first. (In order to improve performance, elapsed time is updated only when you either pause the MiscStopWatch or when you explicitly ask it to update the time via the `±calcElapsedTime:` message.)

Because the MiscStopwatch inherits from MiscTime, you can use all the MiscTime methods to query and manipulate the time tracked by the MiscStopwatch. Also, by implication, the MiscStopwatch implements the NXTransport protocol, allowing it to be sent **bycopy** with Distributed Objects. Also note that it is possible to archive a running MiscStopWatch in a file, and it will continue measuring time. You can use this, for example, to see how long it has been since an application was last launched. The accuracy of the MiscStopWatch is only as good as the accuracy of the internal clock, however, since it is a wrapper around the standard UNIX `gettimeofday()` function.

If you wish to have the MiscStopwatch update it's time at regular intervals then you should use an Animator object to send it a `±calcElapsedTime:` message periodically. Future implementations may include a delegate object which is notified whenever the MiscStopwatch is updated, so that, for instance, a View subclass could

display the changing time dynamically.

## Instance Variables

```
BOOL paused;  
struct timeval startTime;
```

paused

Whether the MiscStopwatch is currently running or not.

startTime

The time when the MiscStopwatch was started, if it is running. Whenever the MiscStopwatch is paused or asked to update the elapsed time, this time is used to increment the elapsed time (available via MiscTime's methods) and is then set to the current system time in order to continue the timing process.

## Method Types

Initializing and freeing a MiscStopwatch ± init

Manipulating a MiscStopwatch ± calcElapsedTime:  
± clearTiming:  
± continueTiming:  
± pauseTiming:  
± startTiming:

Archiving

± read:  
± write:

## Instance Methods

**calcElapsedTime:**

- **calcElapsedTime:sender**

Recalculates the amount of time which has elapsed since the MiscStopwatch was started. If a MiscStopwatch is running, you should call this method before querying the time, since elapsed time is only updated when this method is invoked or when the MiscStopwatch is paused. (This is for performance reasons.) Returns *self*.

See also: **-pauseTiming:**

**clearTiming:**

- **clearTiming:sender**

Clears the MiscStopwatch's elapsed time. Returns *self*.

See also: **-continueTiming:**, **-pauseTiming:**, and **-startTiming:**

**continueTiming:**

- **continueTiming:sender**

Starts the MiscStopwatch's clock without clearing it first. Note that the time during which the MiscStopwatch was paused is not counted as part of the elapsed time. Returns *self*.

See also: **-clearTiming:**, **-pauseTiming:**, and **-startTiming:**

**init**

- **init**

Initializes and clear an instance of MiscStopwatch. Returns *self*.

**pauseTiming:**

- **pauseTiming:sender**

Stops the MiscStopwatch's clock and updates the time. Returns *self*.

See also: **-calcElapsedTime:**, **-clearTiming:**, **-continueTiming:**, and **-startTiming:**

**read:**

- **read:**(NXTypedStream \*)*stream*

Reads the MiscStopwatch from the typed stream *stream*. Returns **self**.

See also: - **write:**

**startTiming:**

- **startTiming:***sender*

Clears the MiscStopwatch and starts it's clock. Returns *self*.

See also: **-clearTiming:**, **-continueTiming:**, and **-pauseTiming:**

**write:**

- **write:**(NXTypedStream \*)*stream*

Writes the MiscStopwatch to the typed stream *stream*. Returns **self**.

See also: - **read:**