;; Auto-generated. Do not edit!

(when (boundp 'robotnik\_msgs\_new::alarmmonitor)

(if (not (find-package "ROBOTNIK\_MSGS\_NEW"))

(make-package "ROBOTNIK\_MSGS\_NEW"))

(shadow 'alarmmonitor (find-package "ROBOTNIK\_MSGS\_NEW")))

(unless (find-package "ROBOTNIK\_MSGS\_NEW::ALARMMONITOR")

(make-package "ROBOTNIK\_MSGS\_NEW::ALARMMONITOR"))

(in-package "ROS")

;;//! \htmlinclude alarmmonitor.msg.html

(defclass robotnik\_msgs\_new::alarmmonitor

:super ros::object

:slots (\_type \_display \_component \_hmi \_group \_text \_seconds\_active ))

(defmethod robotnik\_msgs\_new::alarmmonitor

(:init

(&key

((:type \_\_type) "")

((:display \_\_display) 0)

((:component \_\_component) "")

((:hmi \_\_hmi) nil)

((:group \_\_group) "")

((:text \_\_text) "")

((:seconds\_active \_\_seconds\_active) 0)

)

(send-super :init)

(setq \_type (string \_\_type))

(setq \_display (round \_\_display))

(setq \_component (string \_\_component))

(setq \_hmi \_\_hmi)

(setq \_group (string \_\_group))

(setq \_text (string \_\_text))

(setq \_seconds\_active (round \_\_seconds\_active))

self)

(:type

(&optional \_\_type)

(if \_\_type (setq \_type \_\_type)) \_type)

(:display

(&optional \_\_display)

(if \_\_display (setq \_display \_\_display)) \_display)

(:component

(&optional \_\_component)

(if \_\_component (setq \_component \_\_component)) \_component)

(:hmi

(&optional \_\_hmi)

(if \_\_hmi (setq \_hmi \_\_hmi)) \_hmi)

(:group

(&optional \_\_group)

(if \_\_group (setq \_group \_\_group)) \_group)

(:text

(&optional \_\_text)

(if \_\_text (setq \_text \_\_text)) \_text)

(:seconds\_active

(&optional \_\_seconds\_active)

(if \_\_seconds\_active (setq \_seconds\_active \_\_seconds\_active)) \_seconds\_active)

(:serialization-length

()

(+

;; string \_type

4 (length \_type)

;; int32 \_display

4

;; string \_component

4 (length \_component)

;; bool \_hmi

1

;; string \_group

4 (length \_group)

;; string \_text

4 (length \_text)

;; uint64 \_seconds\_active

8

))

(:serialize

(&optional strm)

(let ((s (if strm strm

(make-string-output-stream (send self :serialization-length)))))

;; string \_type

(write-long (length \_type) s) (princ \_type s)

;; int32 \_display

(write-long \_display s)

;; string \_component

(write-long (length \_component) s) (princ \_component s)

;; bool \_hmi

(if \_hmi (write-byte -1 s) (write-byte 0 s))

;; string \_group

(write-long (length \_group) s) (princ \_group s)

;; string \_text

(write-long (length \_text) s) (princ \_text s)

;; uint64 \_seconds\_active

#+(or :alpha :irix6 :x86\_64)

(progn (sys::poke \_seconds\_active (send s :buffer) (send s :count) :long) (incf (stream-count s) 8))

#-(or :alpha :irix6 :x86\_64)

(cond ((and (class \_seconds\_active) (= (length (\_seconds\_active . bv)) 2)) ;; bignum

(write-long (ash (elt (\_seconds\_active . bv) 0) 0) s)

(write-long (ash (elt (\_seconds\_active . bv) 1) -1) s))

((and (class \_seconds\_active) (= (length (\_seconds\_active . bv)) 1)) ;; big1

(write-long (elt (\_seconds\_active . bv) 0) s)

(write-long (if (>= \_seconds\_active 0) 0 #xffffffff) s))

(t ;; integer

(write-long \_seconds\_active s)(write-long (if (>= \_seconds\_active 0) 0 #xffffffff) s)))

;;

(if (null strm) (get-output-stream-string s))))

(:deserialize

(buf &optional (ptr- 0))

;; string \_type

(let (n) (setq n (sys::peek buf ptr- :integer)) (incf ptr- 4) (setq \_type (subseq buf ptr- (+ ptr- n))) (incf ptr- n))

;; int32 \_display

(setq \_display (sys::peek buf ptr- :integer)) (incf ptr- 4)

;; string \_component

(let (n) (setq n (sys::peek buf ptr- :integer)) (incf ptr- 4) (setq \_component (subseq buf ptr- (+ ptr- n))) (incf ptr- n))

;; bool \_hmi

(setq \_hmi (not (= 0 (sys::peek buf ptr- :char)))) (incf ptr- 1)

;; string \_group

(let (n) (setq n (sys::peek buf ptr- :integer)) (incf ptr- 4) (setq \_group (subseq buf ptr- (+ ptr- n))) (incf ptr- n))

;; string \_text

(let (n) (setq n (sys::peek buf ptr- :integer)) (incf ptr- 4) (setq \_text (subseq buf ptr- (+ ptr- n))) (incf ptr- n))

;; uint64 \_seconds\_active

#+(or :alpha :irix6 :x86\_64)

(setf \_seconds\_active (prog1 (sys::peek buf ptr- :long) (incf ptr- 8)))

#-(or :alpha :irix6 :x86\_64)

(setf \_seconds\_active (let ((b0 (prog1 (sys::peek buf ptr- :integer) (incf ptr- 4)))

(b1 (prog1 (sys::peek buf ptr- :integer) (incf ptr- 4))))

(cond ((= b1 -1) b0)

((and (= b1 0)

(<= lisp::most-negative-fixnum b0 lisp::most-positive-fixnum))

b0)

((= b1 0) (make-instance bignum :size 1 :bv (integer-vector b0)))

(t (make-instance bignum :size 2 :bv (integer-vector b0 (ash b1 1)))))))

;;

self)

)

(setf (get robotnik\_msgs\_new::alarmmonitor :md5sum-) "bb3d91213f2300b7fc9bb9063b1a7b1e")

(setf (get robotnik\_msgs\_new::alarmmonitor :datatype-) "robotnik\_msgs\_new/alarmmonitor")

(setf (get robotnik\_msgs\_new::alarmmonitor :definition-)

"# type: error, warning, event

string type

# displaing number for monitoring

int32 display

# component

string component

# To be shown in HMI

bool hmi

# group: alarm group all alarms of same group will be reset at same time

string group

# text: more explained information about alarm, should include group, time, conditions, etc

string text

# seconds active

uint64 seconds\_active

")

(provide :robotnik\_msgs\_new/alarmmonitor "bb3d91213f2300b7fc9bb9063b1a7b1e")