;; Auto-generated. Do not edit!

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

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

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

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

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

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

(in-package "ROS")

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

(defclass robotnik\_msgs\_new::RobotnikMotorsStatus

:super ros::object

:slots (\_name \_can\_id \_motor\_status ))

(defmethod robotnik\_msgs\_new::RobotnikMotorsStatus

(:init

(&key

((:name \_\_name) (let (r) (dotimes (i 0) (push "" r)) r))

((:can\_id \_\_can\_id) (make-array 0 :initial-element 0 :element-type :integer))

((:motor\_status \_\_motor\_status) (let (r) (dotimes (i 0) (push (instance robotnik\_msgs\_new::MotorStatus :init) r)) r))

)

(send-super :init)

(setq \_name \_\_name)

(setq \_can\_id \_\_can\_id)

(setq \_motor\_status \_\_motor\_status)

self)

(:name

(&optional \_\_name)

(if \_\_name (setq \_name \_\_name)) \_name)

(:can\_id

(&optional \_\_can\_id)

(if \_\_can\_id (setq \_can\_id \_\_can\_id)) \_can\_id)

(:motor\_status

(&rest \_\_motor\_status)

(if (keywordp (car \_\_motor\_status))

(send\* \_motor\_status \_\_motor\_status)

(progn

(if \_\_motor\_status (setq \_motor\_status (car \_\_motor\_status)))

\_motor\_status)))

(:serialization-length

()

(+

;; string[] \_name

(apply #'+ (mapcar #'(lambda (x) (+ 4 (length x))) \_name)) 4

;; int32[] \_can\_id

(\* 4 (length \_can\_id)) 4

;; robotnik\_msgs\_new/MotorStatus[] \_motor\_status

(apply #'+ (send-all \_motor\_status :serialization-length)) 4

))

(:serialize

(&optional strm)

(let ((s (if strm strm

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

;; string[] \_name

(write-long (length \_name) s)

(dolist (elem \_name)

(write-long (length elem) s) (princ elem s)

)

;; int32[] \_can\_id

(write-long (length \_can\_id) s)

(dotimes (i (length \_can\_id))

(write-long (elt \_can\_id i) s)

)

;; robotnik\_msgs\_new/MotorStatus[] \_motor\_status

(write-long (length \_motor\_status) s)

(dolist (elem \_motor\_status)

(send elem :serialize s)

)

;;

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

(:deserialize

(buf &optional (ptr- 0))

;; string[] \_name

(let (n)

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

(setq \_name (make-list n))

(dotimes (i n)

(let (n) (setq n (sys::peek buf ptr- :integer)) (incf ptr- 4) (setf (elt \_name i) (subseq buf ptr- (+ ptr- n))) (incf ptr- n))

))

;; int32[] \_can\_id

(let (n)

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

(setq \_can\_id (instantiate integer-vector n))

(dotimes (i n)

(setf (elt \_can\_id i) (sys::peek buf ptr- :integer)) (incf ptr- 4)

))

;; robotnik\_msgs\_new/MotorStatus[] \_motor\_status

(let (n)

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

(setq \_motor\_status (let (r) (dotimes (i n) (push (instance robotnik\_msgs\_new::MotorStatus :init) r)) r))

(dolist (elem- \_motor\_status)

(send elem- :deserialize buf ptr-) (incf ptr- (send elem- :serialization-length))

))

;;

self)

)

(setf (get robotnik\_msgs\_new::RobotnikMotorsStatus :md5sum-) "40f5305ac2c33d213d54d21bf9ba7fc9")

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

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

"# Name of each motor drive

string[] name

# CAN ID of each motor drive

int32[] can\_id

# Status of the motor drive

robotnik\_msgs\_new/MotorStatus[] motor\_status

================================================================================

MSG: robotnik\_msgs\_new/MotorStatus

string state

string status

string communicationstatus

string statusword

string driveflags

string[] activestatusword

string[] activedriveflags

int32 digitaloutputs

int32 digitalinputs

float32 averagecurrent

float32[] analoginputs

")

(provide :robotnik\_msgs\_new/RobotnikMotorsStatus "40f5305ac2c33d213d54d21bf9ba7fc9")