;; Auto-generated. Do not edit!

(when (boundp 'ourSensor\_msgs::Accel)

(if (not (find-package "OURSENSOR\_MSGS"))

(make-package "OURSENSOR\_MSGS"))

(shadow 'Accel (find-package "OURSENSOR\_MSGS")))

(unless (find-package "OURSENSOR\_MSGS::ACCEL")

(make-package "OURSENSOR\_MSGS::ACCEL"))

(in-package "ROS")

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

(defclass ourSensor\_msgs::Accel

:super ros::object

:slots (\_accel1\_x \_accel1\_y ))

(defmethod ourSensor\_msgs::Accel

(:init

(&key

((:accel1\_x \_\_accel1\_x) 0.0)

((:accel1\_y \_\_accel1\_y) 0.0)

)

(send-super :init)

(setq \_accel1\_x (float \_\_accel1\_x))

(setq \_accel1\_y (float \_\_accel1\_y))

self)

(:accel1\_x

(&optional \_\_accel1\_x)

(if \_\_accel1\_x (setq \_accel1\_x \_\_accel1\_x)) \_accel1\_x)

(:accel1\_y

(&optional \_\_accel1\_y)

(if \_\_accel1\_y (setq \_accel1\_y \_\_accel1\_y)) \_accel1\_y)

(:serialization-length

()

(+

;; float64 \_accel1\_x

8

;; float64 \_accel1\_y

8

))

(:serialize

(&optional strm)

(let ((s (if strm strm

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

;; float64 \_accel1\_x

(sys::poke \_accel1\_x (send s :buffer) (send s :count) :double) (incf (stream-count s) 8)

;; float64 \_accel1\_y

(sys::poke \_accel1\_y (send s :buffer) (send s :count) :double) (incf (stream-count s) 8)

;;

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

(:deserialize

(buf &optional (ptr- 0))

;; float64 \_accel1\_x

(setq \_accel1\_x (sys::peek buf ptr- :double)) (incf ptr- 8)

;; float64 \_accel1\_y

(setq \_accel1\_y (sys::peek buf ptr- :double)) (incf ptr- 8)

;;

self)

)

(setf (get ourSensor\_msgs::Accel :md5sum-) "136544c7034858b5ba440b7169a0f268")

(setf (get ourSensor\_msgs::Accel :datatype-) "ourSensor\_msgs/Accel")

(setf (get ourSensor\_msgs::Accel :definition-)

"float64 accel1\_x

float64 accel1\_y

")

(provide :ourSensor\_msgs/Accel "136544c7034858b5ba440b7169a0f268")