;; Auto-generated. Do not edit!

(when (boundp 'gazebo\_msgs\_new::ContactsState)

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

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

(shadow 'ContactsState (find-package "GAZEBO\_MSGS\_NEW")))

(unless (find-package "GAZEBO\_MSGS\_NEW::CONTACTSSTATE")

(make-package "GAZEBO\_MSGS\_NEW::CONTACTSSTATE"))

(in-package "ROS")

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

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

(ros::roseus-add-msgs "std\_msgs"))

(defclass gazebo\_msgs\_new::ContactsState

:super ros::object

:slots (\_header \_states ))

(defmethod gazebo\_msgs\_new::ContactsState

(:init

(&key

((:header \_\_header) (instance std\_msgs::Header :init))

((:states \_\_states) (let (r) (dotimes (i 0) (push (instance gazebo\_msgs\_new::ContactState :init) r)) r))

)

(send-super :init)

(setq \_header \_\_header)

(setq \_states \_\_states)

self)

(:header

(&rest \_\_header)

(if (keywordp (car \_\_header))

(send\* \_header \_\_header)

(progn

(if \_\_header (setq \_header (car \_\_header)))

\_header)))

(:states

(&rest \_\_states)

(if (keywordp (car \_\_states))

(send\* \_states \_\_states)

(progn

(if \_\_states (setq \_states (car \_\_states)))

\_states)))

(:serialization-length

()

(+

;; std\_msgs/Header \_header

(send \_header :serialization-length)

;; gazebo\_msgs\_new/ContactState[] \_states

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

))

(:serialize

(&optional strm)

(let ((s (if strm strm

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

;; std\_msgs/Header \_header

(send \_header :serialize s)

;; gazebo\_msgs\_new/ContactState[] \_states

(write-long (length \_states) s)

(dolist (elem \_states)

(send elem :serialize s)

)

;;

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

(:deserialize

(buf &optional (ptr- 0))

;; std\_msgs/Header \_header

(send \_header :deserialize buf ptr-) (incf ptr- (send \_header :serialization-length))

;; gazebo\_msgs\_new/ContactState[] \_states

(let (n)

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

(setq \_states (let (r) (dotimes (i n) (push (instance gazebo\_msgs\_new::ContactState :init) r)) r))

(dolist (elem- \_states)

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

))

;;

self)

)

(setf (get gazebo\_msgs\_new::ContactsState :md5sum-) "acbcb1601a8e525bf72509f18e6f668d")

(setf (get gazebo\_msgs\_new::ContactsState :datatype-) "gazebo\_msgs\_new/ContactsState")

(setf (get gazebo\_msgs\_new::ContactsState :definition-)

"Header header # stamp

gazebo\_msgs\_new/ContactState[] states # array of geom pairs in contact

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

MSG: std\_msgs/Header

# Standard metadata for higher-level stamped data types.

# This is generally used to communicate timestamped data

# in a particular coordinate frame.

#

# sequence ID: consecutively increasing ID

uint32 seq

#Two-integer timestamp that is expressed as:

# \* stamp.sec: seconds (stamp\_secs) since epoch (in Python the variable is called 'secs')

# \* stamp.nsec: nanoseconds since stamp\_secs (in Python the variable is called 'nsecs')

# time-handling sugar is provided by the client library

time stamp

#Frame this data is associated with

string frame\_id

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

MSG: gazebo\_msgs\_new/ContactState

string info # text info on this contact

string collision1\_name # name of contact collision1

string collision2\_name # name of contact collision2

geometry\_msgs/Wrench[] wrenches # list of forces/torques

geometry\_msgs/Wrench total\_wrench # sum of forces/torques in every DOF

geometry\_msgs/Vector3[] contact\_positions # list of contact position

geometry\_msgs/Vector3[] contact\_normals # list of contact normals

float64[] depths # list of penetration depths

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

MSG: geometry\_msgs/Wrench

# This represents force in free space, separated into

# its linear and angular parts.

Vector3 force

Vector3 torque

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

MSG: geometry\_msgs/Vector3

# This represents a vector in free space.

# It is only meant to represent a direction. Therefore, it does not

# make sense to apply a translation to it (e.g., when applying a

# generic rigid transformation to a Vector3, tf2 will only apply the

# rotation). If you want your data to be translatable too, use the

# geometry\_msgs/Point message instead.

float64 x

float64 y

float64 z

")

(provide :gazebo\_msgs\_new/ContactsState "acbcb1601a8e525bf72509f18e6f668d")