;; Auto-generated. Do not edit!

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

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

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

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

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

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

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

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

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

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

(in-package "ROS")

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

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

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

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

(defclass gazebo\_msgs\_new::SetJointTrajectoryRequest

:super ros::object

:slots (\_model\_name \_joint\_trajectory \_model\_pose \_set\_model\_pose \_disable\_physics\_updates ))

(defmethod gazebo\_msgs\_new::SetJointTrajectoryRequest

(:init

(&key

((:model\_name \_\_model\_name) "")

((:joint\_trajectory \_\_joint\_trajectory) (instance trajectory\_msgs::JointTrajectory :init))

((:model\_pose \_\_model\_pose) (instance geometry\_msgs::Pose :init))

((:set\_model\_pose \_\_set\_model\_pose) nil)

((:disable\_physics\_updates \_\_disable\_physics\_updates) nil)

)

(send-super :init)

(setq \_model\_name (string \_\_model\_name))

(setq \_joint\_trajectory \_\_joint\_trajectory)

(setq \_model\_pose \_\_model\_pose)

(setq \_set\_model\_pose \_\_set\_model\_pose)

(setq \_disable\_physics\_updates \_\_disable\_physics\_updates)

self)

(:model\_name

(&optional \_\_model\_name)

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

(:joint\_trajectory

(&rest \_\_joint\_trajectory)

(if (keywordp (car \_\_joint\_trajectory))

(send\* \_joint\_trajectory \_\_joint\_trajectory)

(progn

(if \_\_joint\_trajectory (setq \_joint\_trajectory (car \_\_joint\_trajectory)))

\_joint\_trajectory)))

(:model\_pose

(&rest \_\_model\_pose)

(if (keywordp (car \_\_model\_pose))

(send\* \_model\_pose \_\_model\_pose)

(progn

(if \_\_model\_pose (setq \_model\_pose (car \_\_model\_pose)))

\_model\_pose)))

(:set\_model\_pose

(&optional \_\_set\_model\_pose)

(if \_\_set\_model\_pose (setq \_set\_model\_pose \_\_set\_model\_pose)) \_set\_model\_pose)

(:disable\_physics\_updates

(&optional \_\_disable\_physics\_updates)

(if \_\_disable\_physics\_updates (setq \_disable\_physics\_updates \_\_disable\_physics\_updates)) \_disable\_physics\_updates)

(:serialization-length

()

(+

;; string \_model\_name

4 (length \_model\_name)

;; trajectory\_msgs/JointTrajectory \_joint\_trajectory

(send \_joint\_trajectory :serialization-length)

;; geometry\_msgs/Pose \_model\_pose

(send \_model\_pose :serialization-length)

;; bool \_set\_model\_pose

1

;; bool \_disable\_physics\_updates

1

))

(:serialize

(&optional strm)

(let ((s (if strm strm

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

;; string \_model\_name

(write-long (length \_model\_name) s) (princ \_model\_name s)

;; trajectory\_msgs/JointTrajectory \_joint\_trajectory

(send \_joint\_trajectory :serialize s)

;; geometry\_msgs/Pose \_model\_pose

(send \_model\_pose :serialize s)

;; bool \_set\_model\_pose

(if \_set\_model\_pose (write-byte -1 s) (write-byte 0 s))

;; bool \_disable\_physics\_updates

(if \_disable\_physics\_updates (write-byte -1 s) (write-byte 0 s))

;;

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

(:deserialize

(buf &optional (ptr- 0))

;; string \_model\_name

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

;; trajectory\_msgs/JointTrajectory \_joint\_trajectory

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

;; geometry\_msgs/Pose \_model\_pose

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

;; bool \_set\_model\_pose

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

;; bool \_disable\_physics\_updates

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

;;

self)

)

(defclass gazebo\_msgs\_new::SetJointTrajectoryResponse

:super ros::object

:slots (\_success \_status\_message ))

(defmethod gazebo\_msgs\_new::SetJointTrajectoryResponse

(:init

(&key

((:success \_\_success) nil)

((:status\_message \_\_status\_message) "")

)

(send-super :init)

(setq \_success \_\_success)

(setq \_status\_message (string \_\_status\_message))

self)

(:success

(&optional \_\_success)

(if \_\_success (setq \_success \_\_success)) \_success)

(:status\_message

(&optional \_\_status\_message)

(if \_\_status\_message (setq \_status\_message \_\_status\_message)) \_status\_message)

(:serialization-length

()

(+

;; bool \_success

1

;; string \_status\_message

4 (length \_status\_message)

))

(:serialize

(&optional strm)

(let ((s (if strm strm

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

;; bool \_success

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

;; string \_status\_message

(write-long (length \_status\_message) s) (princ \_status\_message s)

;;

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

(:deserialize

(buf &optional (ptr- 0))

;; bool \_success

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

;; string \_status\_message

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

;;

self)

)

(defclass gazebo\_msgs\_new::SetJointTrajectory

:super ros::object

:slots ())

(setf (get gazebo\_msgs\_new::SetJointTrajectory :md5sum-) "88f5c10979e3f9649d5ae87a3b12aa65")

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

(setf (get gazebo\_msgs\_new::SetJointTrajectory :request) gazebo\_msgs\_new::SetJointTrajectoryRequest)

(setf (get gazebo\_msgs\_new::SetJointTrajectory :response) gazebo\_msgs\_new::SetJointTrajectoryResponse)

(defmethod gazebo\_msgs\_new::SetJointTrajectoryRequest

(:response () (instance gazebo\_msgs\_new::SetJointTrajectoryResponse :init)))

(setf (get gazebo\_msgs\_new::SetJointTrajectoryRequest :md5sum-) "88f5c10979e3f9649d5ae87a3b12aa65")

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

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

"string model\_name

trajectory\_msgs/JointTrajectory joint\_trajectory

geometry\_msgs/Pose model\_pose

bool set\_model\_pose

bool disable\_physics\_updates # defaults to false

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

MSG: trajectory\_msgs/JointTrajectory

Header header

string[] joint\_names

JointTrajectoryPoint[] points

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

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: trajectory\_msgs/JointTrajectoryPoint

# Each trajectory point specifies either positions[, velocities[, accelerations]]

# or positions[, effort] for the trajectory to be executed.

# All specified values are in the same order as the joint names in JointTrajectory.msg

float64[] positions

float64[] velocities

float64[] accelerations

float64[] effort

duration time\_from\_start

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

MSG: geometry\_msgs/Pose

# A representation of pose in free space, composed of position and orientation.

Point position

Quaternion orientation

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

MSG: geometry\_msgs/Point

# This contains the position of a point in free space

float64 x

float64 y

float64 z

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

MSG: geometry\_msgs/Quaternion

# This represents an orientation in free space in quaternion form.

float64 x

float64 y

float64 z

float64 w

---

bool success # return true if set wrench successful

string status\_message # comments if available

")

(setf (get gazebo\_msgs\_new::SetJointTrajectoryResponse :md5sum-) "88f5c10979e3f9649d5ae87a3b12aa65")

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

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

"string model\_name

trajectory\_msgs/JointTrajectory joint\_trajectory

geometry\_msgs/Pose model\_pose

bool set\_model\_pose

bool disable\_physics\_updates # defaults to false

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

MSG: trajectory\_msgs/JointTrajectory

Header header

string[] joint\_names

JointTrajectoryPoint[] points

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

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: trajectory\_msgs/JointTrajectoryPoint

# Each trajectory point specifies either positions[, velocities[, accelerations]]

# or positions[, effort] for the trajectory to be executed.

# All specified values are in the same order as the joint names in JointTrajectory.msg

float64[] positions

float64[] velocities

float64[] accelerations

float64[] effort

duration time\_from\_start

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

MSG: geometry\_msgs/Pose

# A representation of pose in free space, composed of position and orientation.

Point position

Quaternion orientation

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

MSG: geometry\_msgs/Point

# This contains the position of a point in free space

float64 x

float64 y

float64 z

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

MSG: geometry\_msgs/Quaternion

# This represents an orientation in free space in quaternion form.

float64 x

float64 y

float64 z

float64 w

---

bool success # return true if set wrench successful

string status\_message # comments if available

")

(provide :gazebo\_msgs\_new/SetJointTrajectory "88f5c10979e3f9649d5ae87a3b12aa65")