;; Auto-generated. Do not edit!

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

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

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

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

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

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

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

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

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

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

(in-package "ROS")

(defclass gazebo\_msgs\_new::DeleteLightRequest

:super ros::object

:slots (\_light\_name ))

(defmethod gazebo\_msgs\_new::DeleteLightRequest

(:init

(&key

((:light\_name \_\_light\_name) "")

)

(send-super :init)

(setq \_light\_name (string \_\_light\_name))

self)

(:light\_name

(&optional \_\_light\_name)

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

(:serialization-length

()

(+

;; string \_light\_name

4 (length \_light\_name)

))

(:serialize

(&optional strm)

(let ((s (if strm strm

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

;; string \_light\_name

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

;;

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

(:deserialize

(buf &optional (ptr- 0))

;; string \_light\_name

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

;;

self)

)

(defclass gazebo\_msgs\_new::DeleteLightResponse

:super ros::object

:slots (\_success \_status\_message ))

(defmethod gazebo\_msgs\_new::DeleteLightResponse

(: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::DeleteLight

:super ros::object

:slots ())

(setf (get gazebo\_msgs\_new::DeleteLight :md5sum-) "ae11ea8100d536d5b92c2b5917abf912")

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

(setf (get gazebo\_msgs\_new::DeleteLight :request) gazebo\_msgs\_new::DeleteLightRequest)

(setf (get gazebo\_msgs\_new::DeleteLight :response) gazebo\_msgs\_new::DeleteLightResponse)

(defmethod gazebo\_msgs\_new::DeleteLightRequest

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

(setf (get gazebo\_msgs\_new::DeleteLightRequest :md5sum-) "ae11ea8100d536d5b92c2b5917abf912")

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

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

"string light\_name # name of the light to be deleted

---

bool success # return true if deletion is successful

string status\_message # comments if available

")

(setf (get gazebo\_msgs\_new::DeleteLightResponse :md5sum-) "ae11ea8100d536d5b92c2b5917abf912")

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

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

"string light\_name # name of the light to be deleted

---

bool success # return true if deletion is successful

string status\_message # comments if available

")

(provide :gazebo\_msgs\_new/DeleteLight "ae11ea8100d536d5b92c2b5917abf912")