TraCI/Change Vehicle State

目录

Command 0xc4: Change Vehicle State

ubyte	string	ubyte	<value_type>
Variable	**Vehicle ID**	**Type of the value**	**New Value**

Changes the state of a vehicle. Because it is possible to change different values of a vehicle, the number of parameter to supply and their types differ between commands. The following values can be changed, the parameter which must be given are also shown in the table. Furthermore it is possible to change all values mentioned in TraCI/Change_VehicleType_State. If you do so, the vehicle gets a new type (named "typeid@vehid") and won't be affected by further changes to the original type.

Overview Changeable Vehicle Variables

Variable	ValueType	Description	Python Method
stop (0x12)	compound (string, double, byte, double), see below	Lets the vehicle stop at the given edge, at the given position and lane. The vehicle will stop for the given duration. Re-issuing a stop command with the same lane and position allows changing the duration. Setting the duration to 0 cancels an existing stop.	setStop (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setStop) setBusStop (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setBusStop) setContainerStop (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setContainerStop) setChargingStationStop (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setChargingStationStop) setParkingAreaStop (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setParkingAreaStop)
			changeLane (https://sumo.

change lane (0x13)	compound (byte, double), see below	Forces a lane change to the lane with the given index; if successful, the lane will be chosen for the given amount of time (in seconds).	dlr.de/pydoc/traci._vehicle.html#VehicleDomain-changeLane)
change sublane (0x15)	double (lateral distance)	Forces a lateral change by the given amount (negative values indicate changing to the right, positive to the left). This will override any other lane change motivations but conform to safety-constraints as configured by laneChangeMode.	changeSublane (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-changeSublane)
slow down (0x14)	compound (double, double), see below	Changes the speed smoothly to the given value over the given amount of time in seconds (can also be used to increase speed).	slowDown (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-slowDown)
resume (0x19)	compound (), see below	Resumes from a stop	resume (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-resume)
change target (0x31)	string (destination edge id)	The vehicle's destination edge is set to the given. The route is rebuilt.	changeTarget (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-changeTarget)
speed (0x40)	double (new speed)	Sets the vehicle speed to the given value. The speed will be followed according to the current speed mode. By default the vehicle may drive slower than the set speed according to the safety rules of the car-follow model. When sending a value of -1 the vehicle will revert to its original behavior (using the *maxSpeed* of its vehicle type and following all safety rules).	setSpeed (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setSpeed)
color (0x45)	ubyte,ubyte,ubyte,ubyte (RGBA)	Sets the vehicle's color.	setColor (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setColor)
change route by id (0x53)	string (route id)	Assigns the named route to the vehicle, assuming a) the named route exists, and b) it starts on the edge the vehicle is currently at(1)(2).	setRouteID (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setRouteID)
change route (0x57)	stringList (ids of edges to pass)	Assigns the list of edges as the vehicle's new route assuming the first edge given is the one the vehicle is curently at(1)(2).	setRoute (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setRoute)
reroute parking area (0xc2)	string (parking area id)	Changes the next parking area in parkingAreaID, updates the vehicle route, and preserve consistency in case of passengers/containers on board.	rerouteParkingArea (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-rerouteParkingArea)
change edge travel time information (0x58)	compound (begin time, end time, edgeID, value), see below	Inserts the information about the travel time (in seconds) of edge "edgeID" valid from begin time to end time (in seconds) into the vehicle's internal edge weights container.	setAdaptedTraveltime (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setAdaptedTraveltime)
change edge effort information (0x59)	compound (begin time, end time, edgeID, value), see below	Inserts the information about the effort of edge "edgeID" valid from begin time to end time (in seconds) into the vehicle's internal edge weights container.	setEffort (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setEffort)
signal states (0x5b)	int	Sets a new state of signal. See TraCI/Vehicle Signalling for more information.	setSignals (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setSignals)
routing mode (0x89)	int	Sets the routing mode (0: default, 1: aggregated)	setRoutingMode (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setRoutingMode)
move to (0x5c)	compound (lane ID, position along lane)	Moves the vehicle to a new position along the current route (3).	moveTo (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-moveTo)
move to XY (0xb4)	compound (edgeID, laneIndex, x, y, angle, keepRoute) (see below)	Moves the vehicle to a new position after normal vehicle movements have taken place. Also forces the angle of the vehicle to the given value (navigational angle in degree). See below for additional details	moveToXY (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-moveToXY)
reroute (compute new route) by travel time (0x90)	compound (<empty>), see below	Computes a new route to the current destination that minimizes travel time. The assumed values for each edge in the network can be customized in various ways. See Simulation/Routing#Travel-time_values_for_routing. Replaces the current route by the found(2).	rerouteTraveltime (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-rerouteTraveltime)

reroute (compute new route) by effort (0x91)	compound (<empty>), see below	Computes a new route using the vehicle's internal and the global edge effort information. Replaces the current route by the found[2].	rerouteEffort (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-rerouteEffort)
speed mode (0xb3)	int bitset (see below)	Sets how the values set by speed (0x40) and slowdown (0x14) shall be treated. Also allows to configure the behavior at junctions. See below.	setSpeedMode (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setSpeedMode)
speed factor (0x5e)	double	Sets the vehicle's speed factor to the given value	setSpeedFactor (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setSpeedFactor)
max speed (0x41)	double	Sets the vehicle's maximum speed to the given value	setMaxSpeed (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setMaxSpeed)
lane change mode (0xb6)	int bitset (see below)	Sets how lane changing in general and lane changing requests by TraCI are performed. See below.	setLaneChangeMode (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setLaneChangeMode)
update bestLanes (0x6a)		updates internal data structures for strategic lane choice. (e.g. after modifying access permissions). **Note:** This happens automatically when changing the route or moving to a new edge.	updateBestLanes (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-updateBestLanes)
add (0x80)	complex (see below)	Adds the defined vehicle. See below.	add (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-add)
add_full (0x85)	complex (see below)	Adds the defined vehicle. See below.	addFull (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-addFull)
remove (0x81)	complex (see below)	Removes the defined vehicle. See below.	remove (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-remove)
length (0x44)	double	Sets the vehicle's length to the given value	setLength (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setLength)
vehicle class (0x49)	string	Sets the vehicle's vehicle class to the given value	setVehicleClass (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setVehicleClass)
emission class (0x4a)	string	Sets the vehicle's emission class to the given value	setEmissionClass (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setEmissionClass)
width (0x4d)	double	Sets the vehicle's width to the given value	setWidth (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setWidth)
height (0xbc)	double	Sets the vehicle's height to the given value	setHeight (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setHeight)
min gap (0x4c)	double	Sets the vehicle's minimum headway gap to the given value	setMinGap (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setMinGap)
shape class (0x4b)	string	Sets the vehicle's shape class to the given value	setShapeClass (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setShapeClass)
acceleration (0x46)	double	Sets the vehicle's wished maximum acceleration to the given value	setAccel (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setAccel)
			setDecel (https://sumo.dlr.

deceleration (0x47)	double	Sets the vehicle's wished maximum deceleration to the given value	de/pydoc/traci._vehicle.html#VehicleDomain-setDecel)
imperfection (0x5d)	double	Sets the vehicle's driver imperfection (sigma) to the given value	setImperfection (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setImperfection)
tau (0x48)	double	Sets the vehicle's wished headway time to the given value.	setTau (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setTau)
type (0x4f)	string	Sets the id of the type for the named vehicle.	setType (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setType)
via (0xbe)	stringList	Changes the via edges to the given edges list (to be used during subsequent rerouting calls).	setVia (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setVia)
max lateral speed (0xba)	double	Sets the maximum lateral speed in m/s for this vehicle.	setMaxSpeedLat (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setMaxSpeedLat)
lateral gap (0xbb)	double	Sets the minimum lateral gap of the vehicle at 50km/h in m.	setMinGapLat (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setMinGapLat)
lateral alignment (0xb9)	string	Sets the preferred lateral alignment for this vehicle.	setLateralAlignment (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setLateralAlignment)
parameter (0x7e)	string, string	Sets the string value for the given string parameter	setParameter (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setParameter)
action step length (0x7d)	double (new action step length), boolean (reset action offset)	Sets the current action step length for the vehicle in s. If the boolean value resetActionOffset is true, an action step is scheduled immediately for the vehicle.	setActionStepLength (https://sumo.dlr.de/pydoc/traci._vehicle.html#VehicleDomain-setActionStepLength)
highlight (0xc7)	highlight specification, see below	Adds a highlight to the vehicle	highlight (http://sumo.sourceforge.net/pydoc/traci._vehicle.html#VehicleDomain-highlight)

Please note:

1. The first occurrence of the edge is currently at is searched within the new route; the vehicle continues the route from this point in the route from. If the edge the vehicle is currently does not exist within the new route, an error is generated.
2. changing the route works only if the vehicle is not within an intersection.
3. The specified lane must lie ahead of the vehicle on its current route. The next occurence is used. Intermediate edges will count the vehicle when entering and exiting.

compound message details

The message contents are as following:

stop (0x12)

byte	integer	byte	string	byte	double	byte	byte	byte	double	byte	int	byte	double	byte	double
value type *compound*	**item number (4 to 7)**	**value type *string***	**Edge ID**	**value type *double***	**end position**	**value type *byte***	**Lane Index**	**value type *double***	**Duration in seconds**	**value type *byte* (optional)**	**stop flags: (see below)**	**value type *double* (optional)**	**start position**	**value type *double* (optional)**	**Until in seconds**

The stop flags are a bitset with the following additive components

- 1 : parking
- 2 : triggered
- 4 : containerTriggered
- 8 : busStop (Edge ID is re-purposed as busStop ID)
- 16 : containerStop (Edge ID is re-purposed as containerStop ID)
- 32 : chargingStation (Edge ID is re-purposed as chargingStation ID)
- 64 : parkingArea (Edge ID is re-purposed as parkingArea ID)

change lane (0x13)

byte	integer	byte	byte	byte	double	byte	byte
value type *compound*	**item number (2 or 3)**	**value type *byte***	**Lane Index**	**value type *double***	**Duration in seconds**	**value type *byte* (optional)**	**bool for relative lane change**

slow down (0x14)

byte	integer	byte	double	byte	double
value type *compound*	**item number (always 2)**	**value type *double***	**Speed**	**value type *double***	**Duration in seconds**

open gap (0x16)

byte	integer	byte	double	byte	double	byte	double	byte	double	byte	double	byte	string
value type *compound*	**item number (4 or 5)**	**value type *double***	**Adapted time headway in seconds**	**value type *double***	**Adapted space headway in seconds**	**value type *double***	**Duration in seconds**	**value type *double***	**changeRate**	**value type *double***	**maxDecel**	**value type *string* (optional)**	**reference vehicle ID**

This command induces a temporary increase of the vehicles desired time headway (car-following parameter tau) and specifies a minimal space headway to keep, as well. The execution is seperated into an adaptation phase, where the headways are gradually altered at the specified rate. As soon as the desired headways are established they are kept for the specified duration. Afterwards they are reset to the original value. Deactivation of the control can be achieved by sending newHeadway=duration=maxDecel=-1. The desired time headway of the vehicle is left unchanged if a value of -1 is send at the corresponding position. If -1 is sent for duration, the largest possible time is set. If -1 is sent for the maxDecel, no limitation of the induced braking rate is applied. If the optional reference vehicle ID is specified, the method does not use the current leader as a reference for the gap creation but the specified vehicle.

Note:
When using the Euler integration method, the time headway control does not work properly for low speeds.

move to (0x5c)

byte	integer	byte	string	byte	double
value type *compound*	**item number (always 2)**	**value type *string***	**Lane ID**	**value type *double***	**Position along lane**

The vehicle will be removed from its lane and moved to the given position on the given lane. No collision checks are done, this means that moving the vehicle may cause a collisions or a situations leading to collision. The vehicle keeps its speed - in the next time step it is at given position + speed. Note that the lane must be a part of the following route, this means it must be either a part of the edge the vehicle is currently on or a part of an edge the vehicle will pass in future; setting a new route before moving the vehicle if needed should work.

Note:
This can also be used to force a vehicle into the network that has been loaded but could not depart due to having it's departure lane blocked.

move to XY (0xb4)

The vehicle (the center of it's front bumper) is moved to the network position that best matches the given x,y network coordinates. The edgeID and laneIndex are compared against the original OpenDRIVE lane id when possible to resolve ambiguities. The optional keepRoute flag influences mapping as follows

- keepRoute = **1**: The vehicle is mapped to the closest edge within it's existing route. If no suitable position is found within 100m mapping fails with an error.
- keepRoute = **0**: The vehicle is mapped to the closest edge within the network. If that edge does not belong to the original route, the current route is replaced by a new route which consists of that edge only. If no suitable position is found within 100m mapping fails with an error. When using the sublane model the best lateral position that is fully within the lane will be used. Otherwise, the vehicle will drive in the center of the closest lane.
- keepRoute = **2**: The vehicle is mapped to the exact position in the network (including the exact lateral position). If that position lies outside the road network, the vehicle stops moving on it's own accord until it is placed back into the network with another TraCI command.

The angle value is assumed to be in navigational degrees (between 0 and 360 with 0 at the top, going clockwise). The angle is used when scoring candidate mapping lanes to resolve ambiguities. Upon mapping the vehicle is angle set to the given value for purposes of drawing and all outputs. If the angle is set to the special TraCI constant INVALID_DOUBLE_VALUE, the angle will not be used for scoring and the vehicle will assume the angle of the best found lane. When mapping the vehicle outside the road network, the angle will be computed from the previous and the new position instead.

Note:
This function can also be used to force a vehicle into the network that has been loaded but could not depart due to having it's departure lane blocked.

Caution:
When mapping a vehicle to an edge that is not currently on it's route, the new route will consist of that edge only. Once the vehicle reaches the end of that edge it disappears unless receiving another moveToXY command in that simulation step. This means, vehicles may disappear when calling *traci.simulationStep* with arguments that cause SUMO to perform multiple steps.

byte	integer	byte	string	byte	double								
value type	**item number**	**value type**	**edge ID (to resolve ambiguities,**	**value type**	**lane index (to resolve ambiguities,**	**value type**	**x Position (network**	**value type**	**y Position (network**	**value type**	**angle**	**value type**	**keepRoute**

compound	**(5 or 6)**	***string***	**may be arbitrary)**	***int***	**may be arbitrary)**	***double***	**coordinates)**	***double***	**coordinates)**	***double***		***byte***	**(0 - 2)**

resume (0x19)

byte	integer
value type *compound*	**item number (always 0)**

change target (0x31)

byte	string
value type *string*	**Edge ID**

change route by id (0x54)

byte	string
value type *string*	**Route ID**

change route (0x57)

byte	int	string
value type *string list*	**number of elements**	**Edge ID**

change edge travel time information (0x58)

This command can be sent using 4, 2, or 1 parameters. In the first form, it adds a new override of edge information to the list of existing overrides for the given vehicle, valid from begin time to end time. In the second form, it adds a new override that is valid throughout the simulation, replacing an earlier one for the same edge. In the third form, it removes as earlier override for the given edge, valid throughout the simulation.

byte	int	byte	double	byte	double	byte	string	byte	double
value type *compound*	**number of elements (always=4)**	**value type *double***	**begin time (in seconds)**	**value type *double***	**end time (in seconds)**	**value type *string***	**edge id**	**value type *double***	**travel time value (in s)**

byte	int	byte	string	byte	double
value type *compound*	**number of elements (always=2)**	**value type *string***	**edge id**	**value type *double***	**travel time value (in s)**

byte	int	byte	string
value type *compound*	**number of elements (always=1)**	**value type *string***	**edge id**

change edge effort information (0x59)

This command can be sent using 4, 2, or 1 parameters. In the first form, it adds a new override of edge information to the list of existing overrides for the given vehicle, valid from begin time to end time. In the second form, it adds a new override that is valid throughout the simulation, replacing an earlier one for the same edge. In the third form, it removes as earlier override for the given edge, valid throughout the simulation.

byte	int	byte	double	byte	double	byte	string	byte	double
value type *compound*	**number of elements (always=4)**	**value type *double***	**begin time (in seconds)**	**value type *double***	**end time (in seconds)**	**value type *string***	**edge id**	**value type *double***	**effort value (abstract)**

byte	int	byte	string	byte	double
value type *compound*	**number of elements (always=2)**	**value type *string***	**edge id**	**value type *double***	**effort value (abstract)**

byte	int	byte	string
value type *compound*	**number of elements (always=1)**	**value type *string***	**edge id**

reroute (compute new route) by travel time (0x90)

byte	int
value type *compound*	**number of elements (always=0)**

reroute (compute new route) by effort (0x91)

byte	int
value type *compound*	**number of elements (always=0)**

speed mode (0xb3)

This command controls how speeds set with the command *setSpeed (0x40)* and *slowDown (0x14)* are used. Per default, the vehicle may only drive slower than the speed that is deemed safe by the car following model and it may not exceed the bounds on acceleration and deceleration. Furthermore, vehicles follow the right-of-way rules when approaching an intersection and if necessary they brake hard to avoid driving across a red light. One can control this behavior using the speed mode (0xb3) command. The given integer is a bitset (bit0 is the least significant bit) with the following fields:

- bit0: Regard safe speed
- bit1: Regard maximum acceleration
- bit2: Regard maximum deceleration
- bit3: Regard right of way at intersections
- bit4: Brake hard to avoid passing a red light

Setting the bit enables the check (the according value is regarded), keeping the bit==zero disables the check.

Examples:

- all checks off -> [0 0 0 0 0] -> Speed Mode = 0
- disable right of way check -> [1 0 1 1 1] -> Speed Mode = 23
- all checks on -> [1 1 1 1 1] -> Speed Mode = 31
- run a red light [0 0 1 1 1] = 7 (also requires setSpeed or slowDown)

lane change mode (0xb6)

The laneChangeModel discriminates four reasons to change lanes:

- strategic (change lanes to continue the route)
- cooperative (change in order to allow others to change)
- speed gain (the other lane allows for faster driving)
- obligation to drive on the right

During each simulation step the laneChangeModel computes an *internal* request to change the lane or to stay on the current lane.

If an *external* change lane command (0x13) command is in conflict with the *internal* request this is resolved by the current value of the vehicles *lane change mode*. The given integer is interpreted as a bitset (bit0 is the least significant bit) with the following fields:

- bit1, bit0: 00 = do no strategic changes; 01 = do strategic changes if not in conflict with a TraCI request; 10 = do strategic change even if overriding TraCI request
- bit3, bit2: 00 = do no cooperative changes; 01 = do cooperative changes if not in conflict with a TraCI request; 10 = do cooperative change even if overriding TraCI request
- bit5, bit4: 00 = do no changes; 01 = do speed gain changes if not in conflict with a TraCI request; 10 = do speed gain change even if overriding TraCI request
- bit7, bit6: 00 = do no right drive changes; 01 = do right drive changes if not in conflict with a TraCI request; 10 = do right drive change even if overriding TraCI request
- bit9, bit8:
  - 00 = do not respect other drivers when following TraCI requests, adapt speed to fulfill request
  - 01 = avoid immediate collisions when following a TraCI request, adapt speed to fulfill request
  - 10 = respect the speed / brake gaps of others when changing lanes, adapt speed to fulfill request
  - 11 = respect the speed / brake gaps of others when changing lanes, no speed adaption
- bit11, bit10: 00 = do no sublane changes; 01 = do sublane changes if not in conflict with a TraCI request; 10 = do sublane change even if overriding TraCI request

The default lane change mode is 0b011001010101 = **1621** which means that the laneChangeModel may execute all changes unless in conflict with TraCI. Requests from TraCI are handled urgently (with cooperative speed adaptations by the ego vehicle and surrounding traffic) but with full consideration for safety constraints.

To disable all autonomous changing but still handle safety checks in the simulation, either one of the modes **256** (collision avoidance) or **512** (collision avoidance and safety-gap enforcement) may be used.

add (0x80)

byte	int	byte	string	byte	string	byte	int	byte	double	byte	double	byte	byte
value type *compound*	**number of elements (always=6)**	**value type *string***	**vehicle type ID (must exist)**	**value type *string***	**route ID (must exist)**	**value type *integer***	**depart time (ms)**	**value type *double***	**depart position**	**value type *double***	**depart speed**	**value type *byte***	**depart lane**

Note:
Please note that the values are not checked in a very elaborated way. Make sure they are correct before sending.

If a negative departure time is set, one of the following fixed time settings will be used:

- -1: "triggered"
- -2: "containerTriggered"

If a negative departure speed is set, one of the following fixed speed settings will be used:

- -2: "random"
- -3: "max"

If a negative departure position is set, one of the following position settings will be used:

- -2: "random"
- -3: "free"
- -4: "base"
- -5: "last"
- -6: "random_free"

Please note that giving 0 as depart position will result in the vehicle starting with its front at the begin of lane (unlike the simulation default which is the vehicle placed completely on the lane "base")

If a negative departure lane is set, one of the following lane settings will be used:

- -2: "random"
- -3: "free"
- -4: "allowed"
- -5: "best"
- -6: "first"

If an empty routeID is given, the vehicle will be placed on an route that consists of a single arbitrary edge (with suitalbe vClass permissions). This can be used to simply the initialization of remote controlled vehicle (moveToXY).

add_full (0x85)

byte	int	byte	string	byte	string	byte	string	byte	string	byte	string	byte	string	byte	string	byte	string	b
value type *compound*	number of elements (always=14)	value type *string*	route ID (must exist)	value type *string*	vehicle type ID (must exist)	value type *string*	depart time	value type *string*	depart lane	value type *string*	depart position	value type *string*	depart speed	value type *string*	arrival lane	value type *string*	arrival position	va t *s*

If an empty routeID is given, the vehicle will be placed on an route that consists of a single arbitrary edge (with suitalbe vClass permissions). This can be used to simply the initialization of remote controlled vehicle (moveToXY).

Note:
Please note that the values are not checked in a very elaborated way. Make sure they are correct before sending.

remove (0x81)

byte	byte
value type *byte*	reason

The following reasons may be given:

- 0: NOTIFICATION_TELEPORT
- 1: NOTIFICATION_PARKING
- 2: NOTIFICATION_ARRIVED
- 3: NOTIFICATION_VAPORIZED
- 4: NOTIFICATION_TELEPORT_ARRIVED

highlight (0xc7)

ubyte	int	ubyte	color	ubyte	double	ubyte	ubyte	ubyte	double	ubyte	ubyte
value type *compound*	number of items in {0,1,2,3,4,5}	value type *color*	color	value type *double*	size (radius in m.)	value type *ubyte*	maximal alpha value	value type *double*	duration (in secs.)	value type *ubyte*	type ID (allows different highlight categories)

The highlight method adds a circle of the specified size and color centered at the vehicle. The circles alpha value is animated over the specified duration (fade in and out) if a positive value is given, maxAlpha <= 0 implies persistent highlighting without animation.

Setting Device and LaneChangeModel Parameters (0x7e)

Vehicles supports setting of device parameters using the generic parameter setting call.

Supported Device Parameters

- device.battery.actualBatteryCapacity (double literal)
- device.battery.maximumBatteryCapacity (double literal)
- device.battery.vehicleMass (double literal)
- device.rerouting.period (double literal, set rerouting period in seconds)
- device.rerouting.edge:EDGE_ID (double literal, set assumed travel time for rerouting for **all** vehicles (where EDGE_ID is the id if a network edge). This value is overwritten at the next update interval (**--device.rerouting.adaptation-interval**).
- device.example.customValue1 (double literal)
- has.rerouting.device ("true"): can be used to dynamically enable automatic rerouting

Caution:
If the vehicles does not carry the respective device an error is returned.

Supported LaneChangeModel Parameters

- laneChangeModel.<ATTRNAME> , (double literal): <ATTRNAME> must be one of the parameters supported by the laneChangeModel of the vehicle. i.e. *lcStrategic*)

Application order of traci commands and simulation step

Step **n**:

1. *traci.vehicle.getPosition* and *traci.vehicle.getSpeed* show the values from step **n-1**
2. when calling *traci.vehicle.moveTo* the vehicle is instantly placed (and still moves in this step)
3. *vNext* (the speed for this step) is computed based on the values of the previous step (*traci.vehicle.setSpeed* overrides this)
4. vehicle position is updated based on *vNext* (either by adding *vNext* or by adding *(vPrev + vNext) / 2* depending on the value of option **--step-method.ballistic**)
5. the value given to *traci.vehicle.moveToXY* overrides the previously computed position
6. GUI is updated to show final state of step **n**, outputs like *fcd-output* also show this state for step **n**

Step **n+1**

This means, that the position given to *moveToXY* in step **n** will be visible in the outputs for step **n** whereas *traci.vehicle.getPosition* will always show the value from the previous step.

Vehicle insertion

When a vehicle is added using method *add* or *addFull* it is not immediately inserted into the network. Only after the next call to *simulationStep* does the simulation try to insert it (and this may fail when in conflict with other traffic). The result of *getIDList* only contains vehicles that have been inserted into the network which means the vehicle will not be listed immediately. You can force a vehicle to be inserted instantly by calling the function *vehicle.moveTo* or *vehicle.moveToXY* after adding it to the simulation.

取自“http://sumo.dlr.de/w/index.php?title=TraCI/Change_Vehicle_State&oldid=12284”

本页面最后编辑于2019年6月3日 (星期一) 14:30。