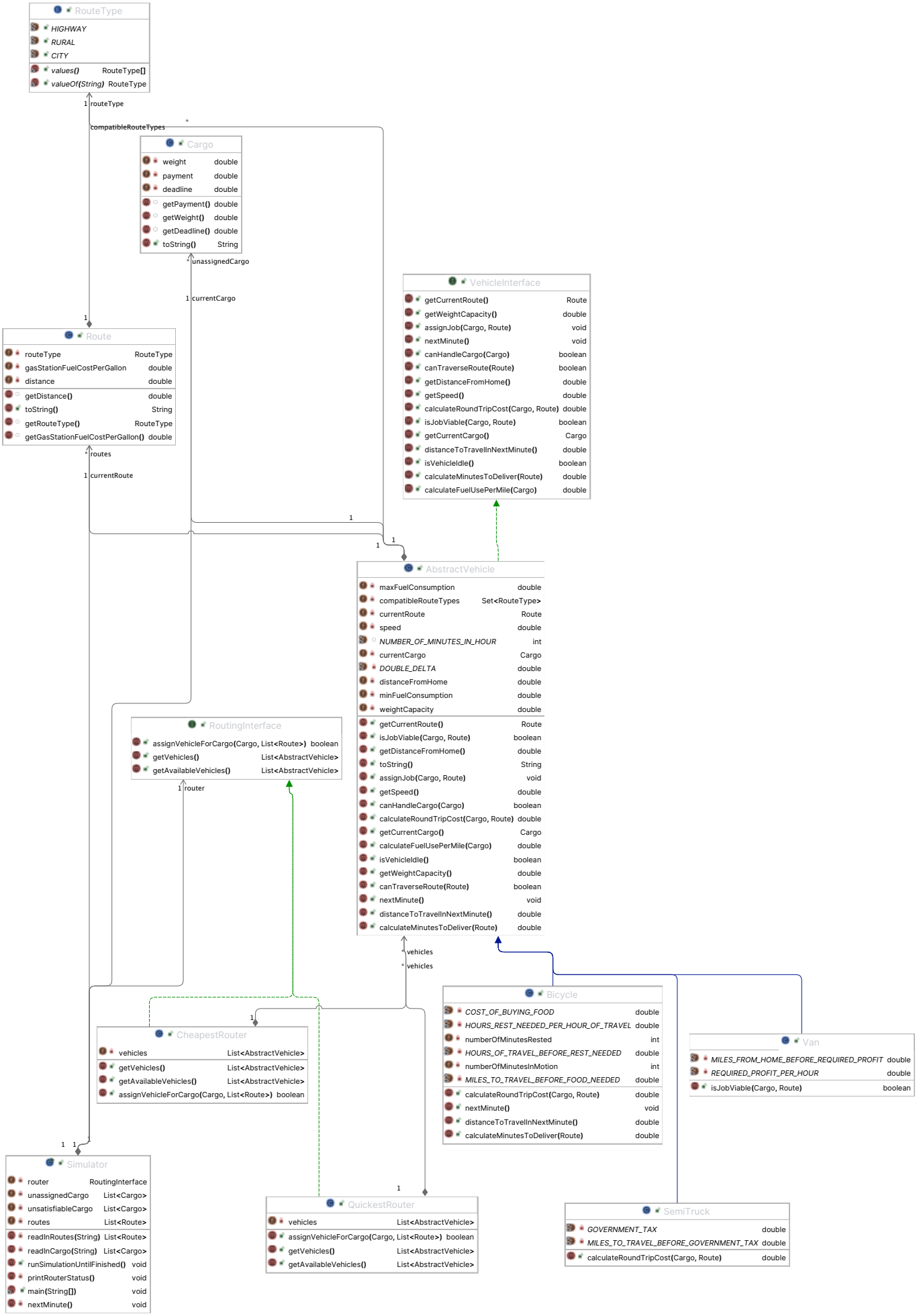
RouteType
HIGHWAY
RURAL
CITY
values() RouteType[]
valueOf(String) RouteType
1 routeType
compatibleRouteTypes
*
Cargo
weight double
payment double
deadline double
getPayment() double
getWeight() double
getDeadline() double
toString() String
* unassignedCargo
1 currentCargo
VehicleInterface
getCurrentRoute() Route
getWeightCapacity() double
assignJob(Cargo, Route) void
nextMinute() void
canHandleCargo(Cargo) boolean
canTraverseRoute(Route) boolean
getDistanceFromHome() double
getSpeed() double
calculateRoundTripCost(Cargo, Route) double
isJobViable(Cargo, Route) boolean
getCurrentCargo() Cargo
distanceToTravelInNextMinute() double
isVehicleIdle() boolean
calculateMinutesToDeliver(Route) double
calculateFuelUsePerMile(Cargo) double
1
Route
routeType RouteType
gasStationFuelCostPerGallon double
distance double
getDistance() double
toString() String
getRouteType() RouteType
getGasStationFuelCostPerGallon() double
* routes
1 currentRoute
1
1
1
AbstractVehicle
maxFuelConsumption double
compatibleRouteTypes Set<RouteType>
currentRoute Route
speed double
NUMBER_OF_MINUTES_IN_HOUR int
currentCargo Cargo
DOUBLE_DELTA double
distanceFromHome double
minFuelConsumption double
weightCapacity double
getCurrentRoute() Route
isJobViable(Cargo, Route) boolean
getDistanceFromHome() double
toString() String
assignJob(Cargo, Route) void
getSpeed() double
canHandleCargo(Cargo) boolean
calculateRoundTripCost(Cargo, Route) double
getCurrentCargo() Cargo
calculateFuelUsePerMile(Cargo) double
isVehicleIdle() boolean
getWeightCapacity() double
canTraverseRoute(Route) boolean
nextMinute() void
distanceToTravelInNextMinute() double
calculateMinutesToDeliver(Route) double
RoutingInterface
assignVehicleForCargo(Cargo, List<Route>) boolean
getVehicles() List<AbstractVehicle>
getAvailableVehicles() List<AbstractVehicle>
1 router
* vehicles
* vehicles
Bicycle
COST_OF_BUYING_FOOD double
HOURS_REST_NEEDED_PER_HOUR_OF_TRAVEL double
numberOfMinutesRested int
HOURS_OF_TRAVEL_BEFORE_REST_NEEDED double
numberOfMinutesInMotion int
MILES_TO_TRAVEL_BEFORE_FOOD_NEEDED double
calculateRoundTripCost(Cargo, Route) double
nextMinute() void
distanceToTravelInNextMinute() double
calculateMinutesToDeliver(Route) double
Van
MILES_FROM_HOME_BEFORE_REQUIRED_PROFIT double
REQUIRED_PROFIT_PER_HOUR double
isJobViable(Cargo, Route) boolean
1
CheapestRouter
vehicles List<AbstractVehicle>
getVehicles() List<AbstractVehicle>
getAvailableVehicles() List<AbstractVehicle>
assignVehicleForCargo(Cargo, List<Route>) boolean
1 1
Simulator
router RoutingInterface
unassignedCargo List<Cargo>
unsatisfiableCargo List<Cargo>
routes List<Route>
readInRoutes(String) List<Route>
readInCargo(String) List<Cargo>
runSimulationUntilFinished() void
printRouterStatus() void
main(String[]) void
nextMinute() void
1
QuickestRouter
vehicles List<AbstractVehicle>
assignVehicleForCargo(Cargo, List<Route>) boolean
getVehicles() List<AbstractVehicle>
getAvailableVehicles() List<AbstractVehicle>
SemiTruck
GOVERNMENT_TAX double
MILES_TO_TRAVEL_BEFORE_GOVERNMENT_TAX double
calculateRoundTripCost(Cargo, Route) double