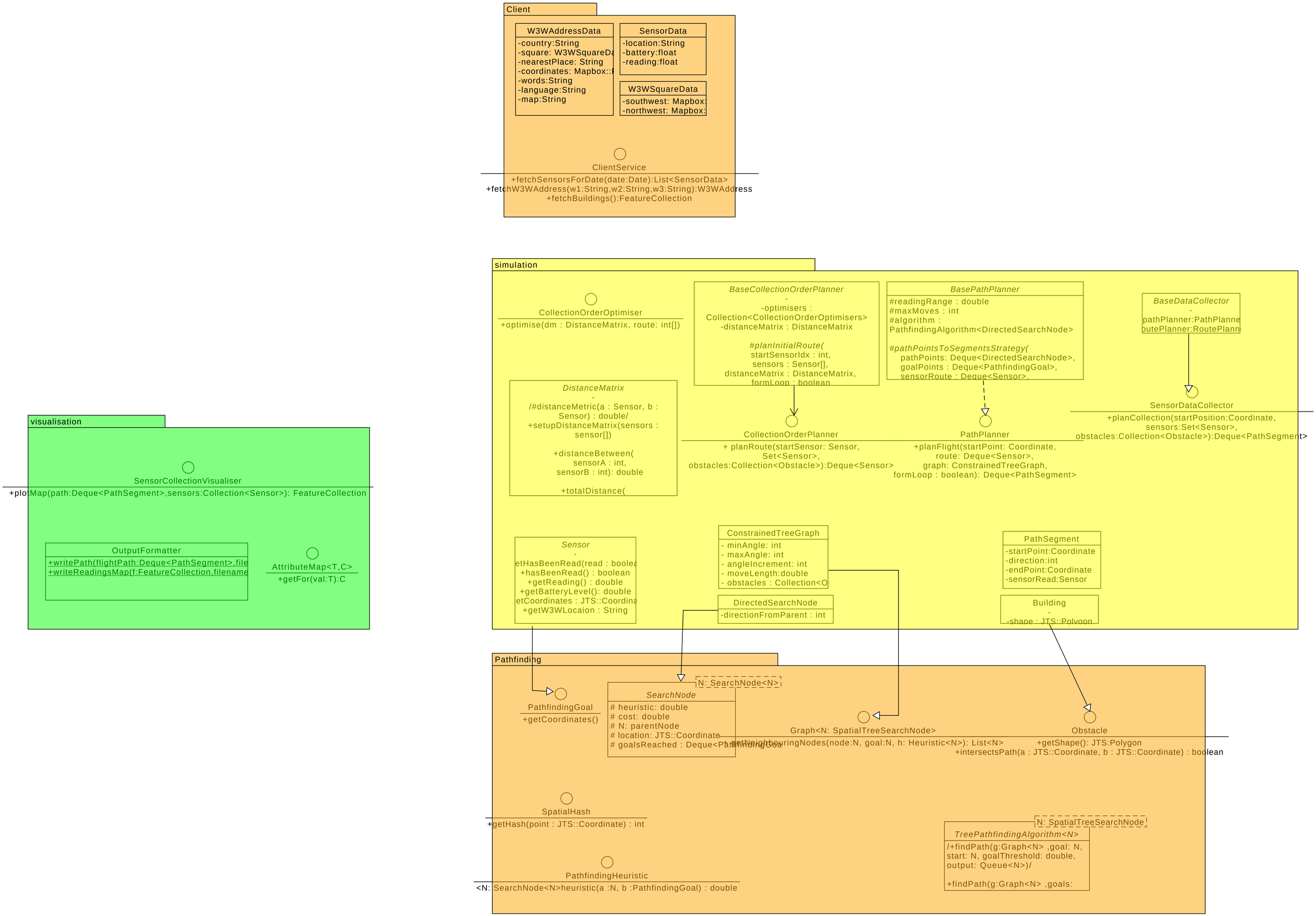
Client
W3WAddressData
-country:String
-square: W3WSquareDa
-nearestPlace: String
-coordinates: Mapbox::
-words:String
-language:String
-map:String
SensorData
-location:String
-battery:float
-reading:float
W3WSquareData
-southwest: Mapbox:
-northwest: Mapbox:
ClientService
+fetchSensorsForDate(date:Date):List<SensorData>
+fetchW3WAddress(w1:String,w2:String,w3:String):W3WAddress
+fetchBuildings():FeatureCollection
simulation
CollectionOrderOptimiser
+optimise(dm : DistanceMatrix, route: int[])
BaseCollectionOrderPlanner
-
-optimisers :
Collection<CollectionOrderOptimisers>
-distanceMatrix : DistanceMatrix
#planInitialRoute(
startSensorIdx : int,
sensors : Sensor[],
distanceMatrix : DistanceMatrix,
formLoop : boolean
BasePathPlanner
#readingRange : double
#maxMoves : int
#algorithm :
PathfindingAlgorithm<DirectedSearchNode>
#pathPointsToSegmentsStrategy(
pathPoints: Deque<DirectedSearchNode>,
goalPoints : Deque<PathfindingGoal>,
sensorRoute : Deque<Sensor>,
BaseDataCollector
-
pathPlanner:PathPlanne
outePlanner:RoutePlann
DistanceMatrix
-
/#distanceMetric(a : Sensor, b :
Sensor) : double/
+setupDistanceMatrix(sensors :
sensor[])
+distanceBetween(
sensorA : int,
sensorB : int): double
+totalDistance(
CollectionOrderPlanner
+ planRoute(startSensor: Sensor,
Set<Sensor>,
obstacles:Collection<Obstacle>):Deque<Sensor>
PathPlanner
+planFlight(startPoint: Coordinate,
route: Deque<Sensor>,
graph: ConstrainedTreeGraph,
formLoop : boolean): Deque<PathSegment>
SensorDataCollector
+planCollection(startPosition:Coordinate,
sensors:Set<Sensor>,
obstacles:Collection<Obstacle>):Deque<PathSegment>
visualisation
SensorCollectionVisualiser
+plotMap(path:Deque<PathSegment>,sensors:Collection<Sensor>): FeatureCollection
OutputFormatter
+writePath(flightPath:Deque<PathSegment>,file
+writeReadingsMap(f:FeatureCollection,filename
AttributeMap<T,C>
+getFor(val:T):C
Sensor
-
etHasBeenRead(read : boolea
+hasBeenRead() : boolean
+getReading() : double
+getBatteryLevel(): double
etCoordinates : JTS::Coordina
+getW3WLocaion : String
ConstrainedTreeGraph
- minAngle: int
- maxAngle: int
- angleIncrement: int
- moveLength:double
- obstacles : Collection<O
DirectedSearchNode
-directionFromParent : int
PathSegment
-startPoint:Coordinate
-direction:int
-endPoint:Coordinate
-sensorRead:Sensor
Building
-
-shape : JTS::Polygon
Pathfinding
PathfindingGoal
+getCoordinates()
N: SearchNode<N>
SearchNode
heuristic: double
cost: double
N: parentNode
location: JTS::Coordinate
goalsReached : Deque<PathfindingGoa
Graph<N: SpatialTreeSearchNode>
+getNeighbouringNodes(node:N, goal:N, h: Heuristic<N>): List<N>
Obstacle
+getShape(): JTS:Polygon
+intersectsPath(a : JTS::Coordinate, b : JTS::Coordinate) : boolean
SpatialHash
+getHash(point : JTS::Coordinate) : int
PathfindingHeuristic
<N: SearchNode<N>heuristic(a :N, b :PathfindingGoal) : double
N: SpatialTreeSearchNode
TreePathfindingAlgorithm<N>
/+findPath(g:Graph<N> ,goal: N,
start: N, goalThreshold: double,
output: Queue<N>)/
+findPath(g:Graph<N> ,goals: