UDACITY

‹ Return to Classroom

Build an OpenStreetMap Route Planner

REVIEW

CODE REVIEW 3

HISTORY

▸ CppND-Route-Planning-Project/src/route_planner.cpp 2

▾ CppND-Route-Planning-Project/src_original/route_model.cpp 1

```
#include "route_model.h"
#include <iostream>

RouteModel::RouteModel(const std::vector<std::byte> &xml) : Model(xml) {
    // Create RouteModel nodes.
    int counter = 0;
    for (Model::Node node : this->Nodes()) {
        m_Nodes.emplace_back(Node(counter, this, node));
        counter++;
    }
    CreateNodeToRoadHashmap();
}


void RouteModel::CreateNodeToRoadHashmap() {
    for (const Model::Road &road : Roads()) {
        if (road.type != Model::Road::Type::Footway) {
            for (int node_idx : Ways()[road.way].nodes) {
                if (node_to_road.find(node_idx) == node_to_road.end()) {
                    node_to_road[node_idx] = std::vector<const Model::Road *> ();
                }
                node_to_road[node_idx].push_back(&road);
            }
        }
    }
}


RouteModel::Node *RouteModel::Node::FindNeighbor(std::vector<int> node_indices) {
    Node *closest_node = nullptr;
    Node node;

    for (int node_index : node_indices) {
        node = parent_model->SNodes()[node_index];
        if (this->distance(node) != 0 && !node.visited) {
            if (closest_node == nullptr || this->distance(node) < this->distance(*closest_node)) {
                closest_node = &parent_model->SNodes()[node_index];
            }
        }
    }
    return closest_node;
}


void RouteModel::Node::FindNeighbors() {
    for (auto & road : parent_model->node_to_road[this->index]) {
```

SUGGESTION

`road` could be a `const` here.

```
        RouteModel::Node *new_neighbor = this->FindNeighbor(parent_model->Ways()[road->way].nodes);
        if (new_neighbor) {
            this->neighbors.emplace_back(new_neighbor);
        }
    }
}


RouteModel::Node &RouteModel::FindClosestNode(float x, float y) {
    Node input;
    input.x = x;
    input.y = y;

    float min_dist = std::numeric_limits<float>::max();
    float dist;
    int closest_idx;

    for (const Model::Road &road : Roads()) {
        if (road.type != Model::Road::Type::Footway) {
            for (int node_idx : Ways()[road.way].nodes) {
                dist = input.distance(SNodes()[node_idx]);
                if (dist < min_dist) {
                    closest_idx = node_idx;
                    min_dist = dist;
                }
            }
        }
    }

    return SNodes()[closest_idx];
}
```

▸ **CppND-Route-Planning-Project/build/thirdparty/googletest/googlemock/gtest/CMakeFiles/gtest_main.dir/src/gtest_main.cc.o**

▸ **CppND-Route-Planning-Project/build/thirdparty/googletest/googlemock/CMakeFiles/gmock_main.dir/src/gmock_main.cc.o**

▸ **CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest_main.cc**

▸ **CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest.cc**

▸ **CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest-typed-test.cc**

▸ **CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest-test-part.cc**

▸ CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest-printers.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest-port.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest-matchers.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest-internal-inl.h

▸ CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest-filepath.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest-death-test.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googletest/src/gtest-all.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googlemock/src/gmock_main.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googlemock/src/gmock.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googlemock/src/gmock-spec-builders.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googlemock/src/gmock-matchers.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googlemock/src/gmock-internal-utils.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googlemock/src/gmock-cardinalities.cc

▸ CppND-Route-Planning-Project/thirdparty/googletest/googlemock/src/gmock-all.cc

▸ CppND-Route-Planning-Project/build/CMakeFiles/route_planner.dir/src/route_planner.cpp.o

▸ CppND-Route-Planning-Project/build/CMakeFiles/OSM_A_star_search.dir/src/route_planner.cpp.o

▸ CppND-Route-Planning-Project/build/CMakeFiles/OSM_A_star_search.dir/src/my_utility.cpp.o

▸ CppND-Route-Planning-Project/thirdparty/pugixml/src/pugixml.hpp

▸ CppND-Route-Planning-Project/thirdparty/pugixml/src/pugiconfig.hpp

▸ CppND-Route-Planning-Project/src_original/route_planner.h

▸ CppND-Route-Planning-Project/src_original/route_planner.cpp

▸ CppND-Route-Planning-Project/src_original/route_model.h

▸ CppND-Route-Planning-Project/src_original/render.h

▸ CppND-Route-Planning-Project/src_original/render.cpp

▸ CppND-Route-Planning-Project/src_original/model.h

▸ CppND-Route-Planning-Project/src_original/model.cpp

▸ CppND-Route-Planning-Project/src_original/main.cpp

▸ CppND-Route-Planning-Project/src/route_planner.h

▸ CppND-Route-Planning-Project/src/route_model.h

▸ CppND-Route-Planning-Project/src/route_model.cpp

▸ CppND-Route-Planning-Project/src/render.h

▸ CppND-Route-Planning-Project/src/render.cpp

▸ CppND-Route-Planning-Project/src/my_utility.h

▸ CppND-Route-Planning-Project/src/my_utility.cpp

▸ CppND-Route-Planning-Project/src/model.h

▸ CppND-Route-Planning-Project/src/model.cpp

▸ CppND-Route-Planning-Project/src/main.cpp

RETURN TO PATH