src/sensor/include/CLidar.h

```cpp
1   // This header file is for class CLidar which has the capability to
2   // read lidar data and publish it to TB3Drive class.
3   // The purpose of this is to isolate capability of reading in Lidar
4   // values from LaserScan msg so that any error in reading in lidar
5   // can be identified easily -- Follow OOP design
6
7   #ifndef CLIDAR_H_
8   #define CLIDAR_H_
9
10  #include <ros/ros.h>
11
12  #include <sensor_msgs/LaserScan.h>
13  #include <geometry_msgs/Twist.h>
14  #include <nav_msgs/Odometry.h>
15  #include <vector>
16  #include <std_msgs/Float64MultiArray.h>
17
18  const char TOPIC_NAME[] = "LIDAR";
19  const int LIDAR_DATA_SIZE = 3;
20  const int SCAN_ANGLE[] = {0, 45,315};
21
22  // Set queue size big to prevent loss if any
23  // delay occurs
24  //https://stackoverflow.com/questions/56444248/reason-to-set-queue-size-of-ros-
    publisher-or-subscriver-to-a-large-value
25  const int QSize = 1000;
26
27  /// @brief---
28  // CLidar interface-------------------------------------------------------
29  // This class is for storing the lidar data and publish to its own topic
30  // for drive class to listen to and get the data. It serves as a class
31  // that subscribes to the internal lidar scan topic and store it in a
32  // vector.
33  class CLidar{
34    public:
35      CLidar();
36      ~CLidar();
37      void LidarScanMsgCallBack(const sensor_msgs::LaserScan::ConstPtr &msg);
38      void FillPublishData();
39
40      private:
41          // ROS NodeHandle
42      ros::NodeHandle nh_;
43      ros::NodeHandle nh_priv_;
44
45      // ROS Subscriber to listen in lidar
46      ros::Subscriber laserScanSub;
47
48      // ROS publisher to publish to a new topic
49      ros::Publisher lidarPub;
50
51          // Data for publishing
52      std::vector<double> ScanData;
53      std_msgs::Float64MultiArray msg2Pub;
54  };
55
```

```
56   #endif
```