

## YDLIDAR SDK PACKAGE V1.3.7

SDK [test](#) application for YDLIDAR

Visit EAI Website for more details about [YDLIDAR](#) .

## How to build YDLIDAR SDK samples

```
$ git clone https://github.com/yangfuyuan/ydlidar_sdk
$ cd ydlidar_sdk
$ git checkout master
$ cd ..
```

### Linux:

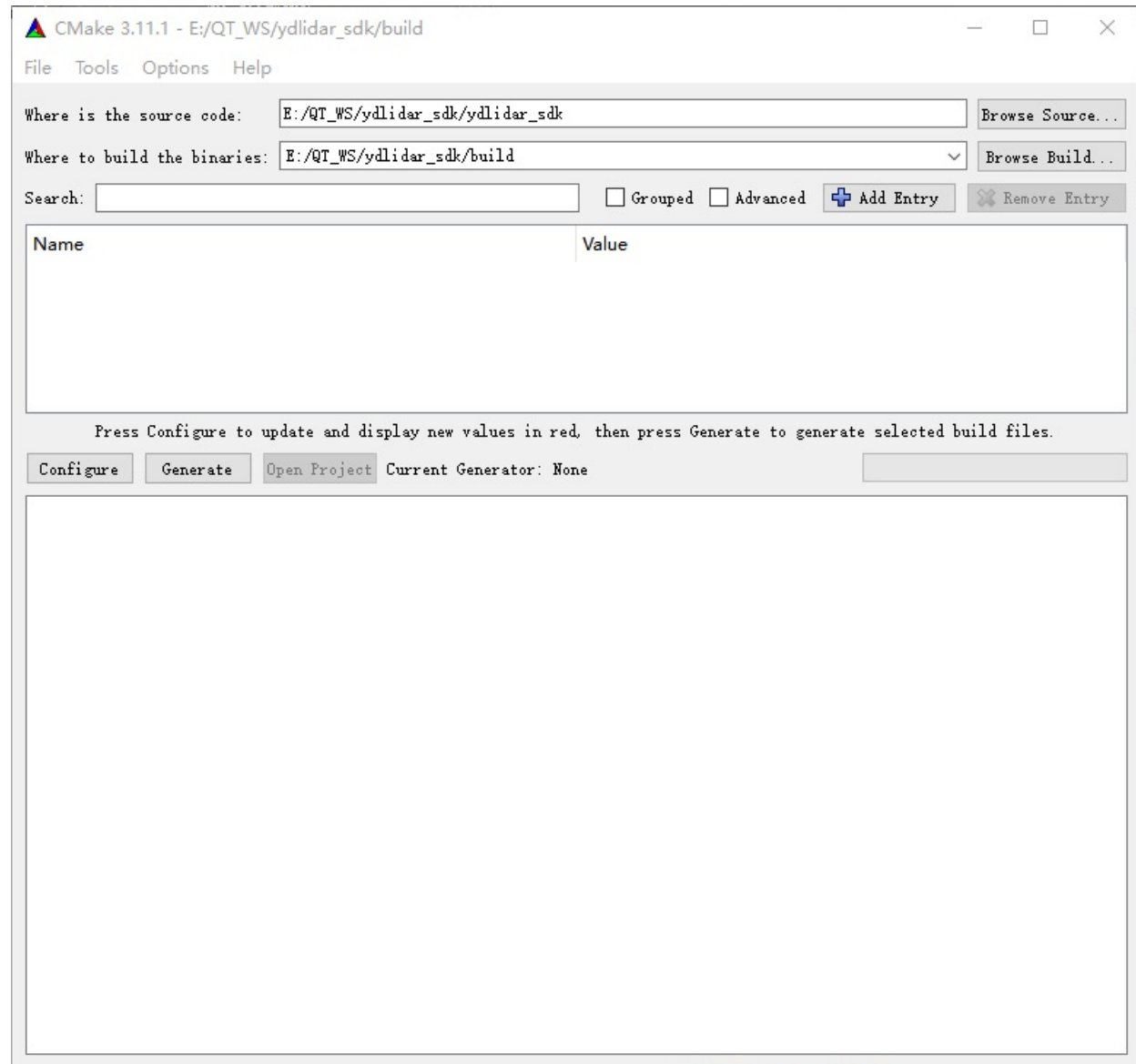
```
$ mkdir build
$ cd build
$ cmake ../ydlidar_sdk
$ make
```

### Windows:

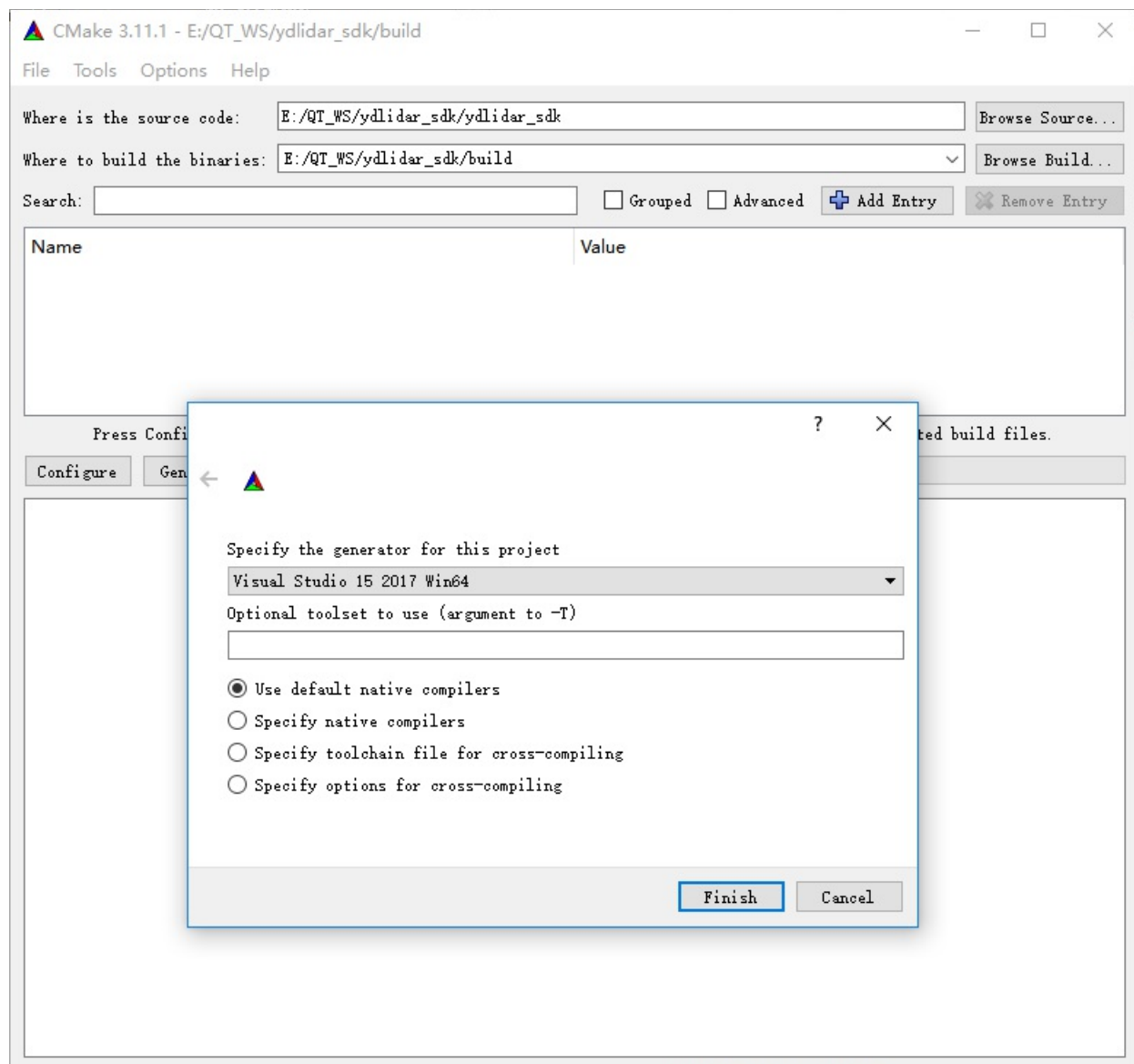
1. install [cmake](#)(if there is no cmake)

## 2. build steps:

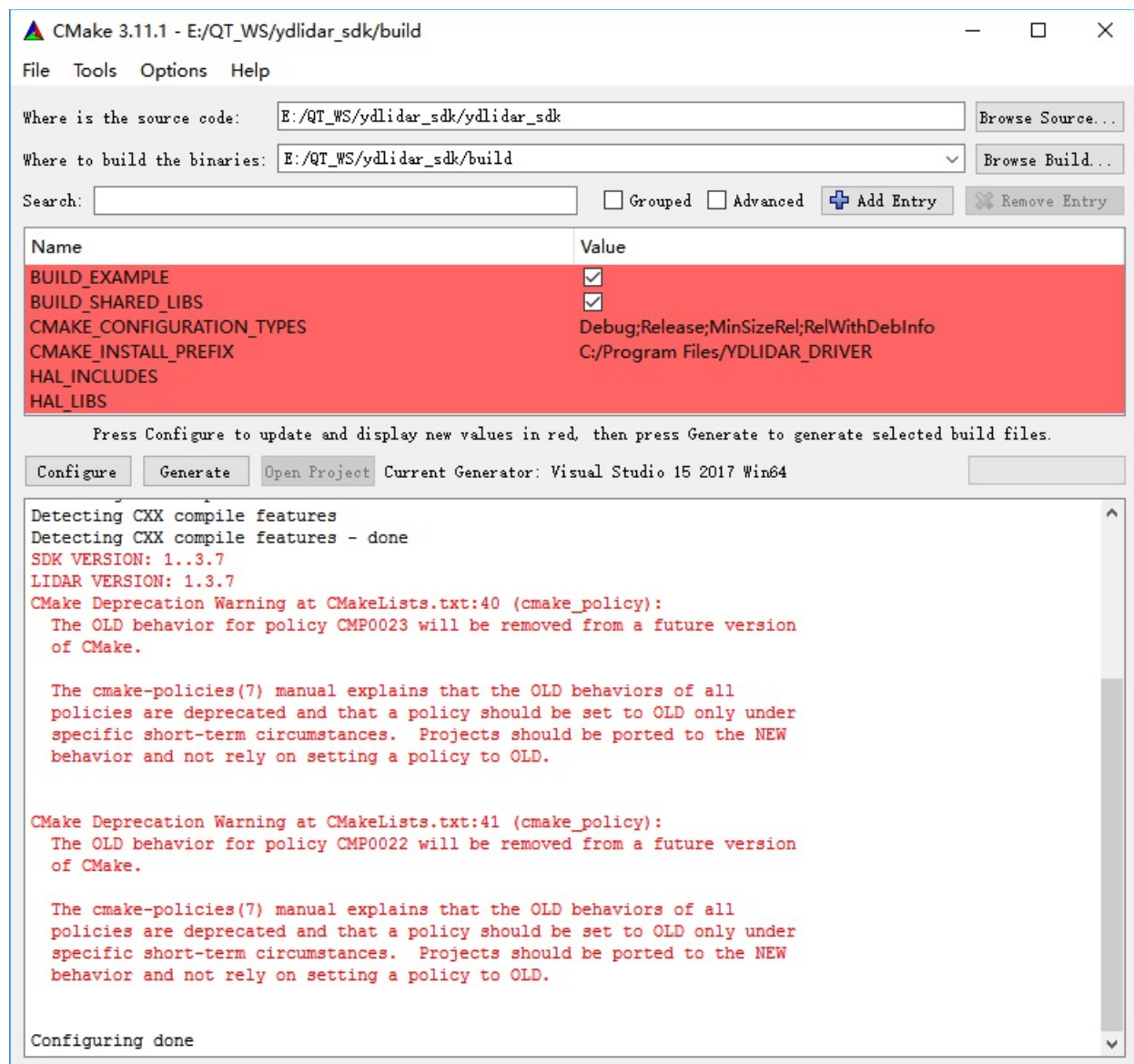
### Step1: open cmake-gui and select source code/binaries directory



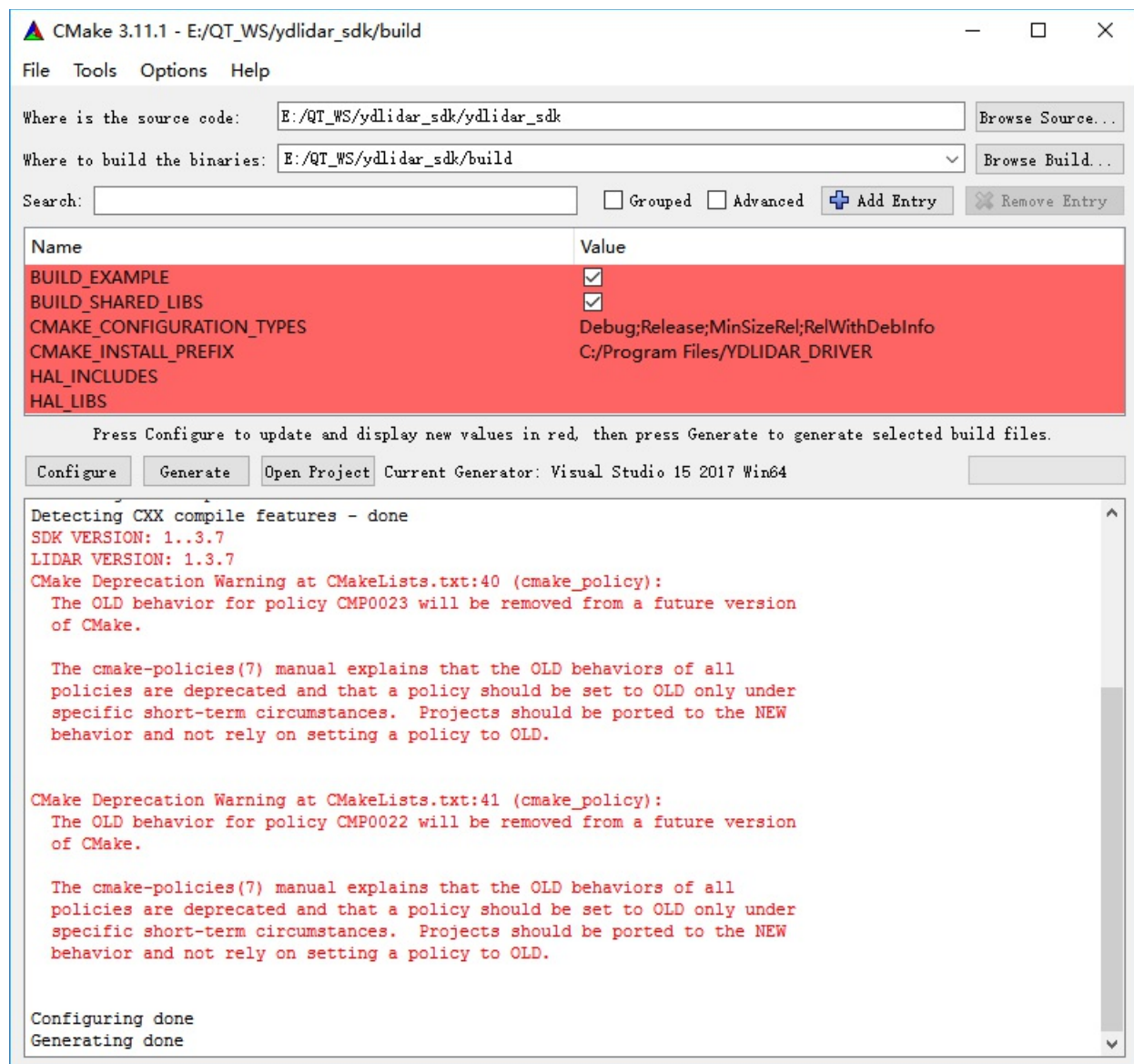
### Step2: Configure and select build toolchain(choose the VS version in your system)



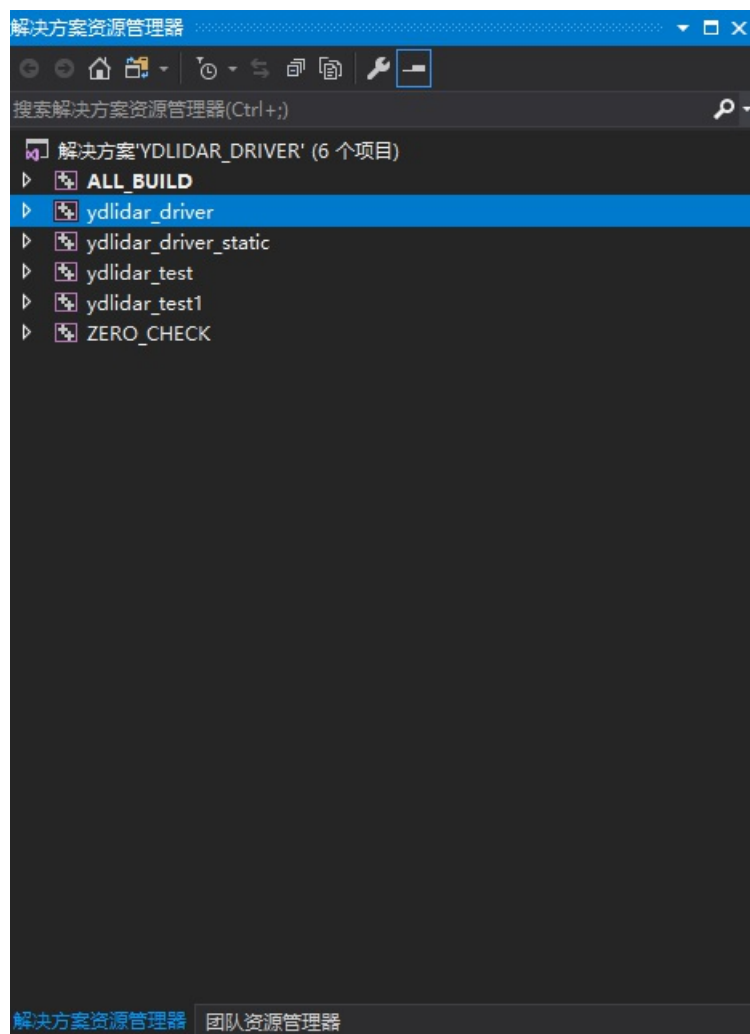
**Step3: configuring done(click “Configure” button)**



**Step4: generating done(click “Generate” button)**



## Step5: open vs Project in binaries directory



**Step6: build finished and run test:**

```
E:\QT_WS\ydlidar_sdk\build\Debug\ydlidar_test.exe
YDLIDAR C++ TEST
Radar[ydlidar7] detected, whether to select current radar(yes/no)? :yes
0. ydlidar7
Please select the lidar port:0
0. 115200
1. 128000
2. 153600
3. 230400
Please select the lidar baud rate:3
0. false
1. true
Please select the lidar intensity:0
SDK Version: 1.3.7
LIDAR Version: 1.3.7
firmware: 521
[YDLIDAR] Connection established in [COM3]:
Firmware version: 2.0.9
Hardware version: 2
Model: G4
Serial: 2018042100000023
[YDLIDAR INFO] Current Sampling Rate : 9K
[YDLIDAR INFO] Current Scan Frequency : 7.000000Hz
start scanning.....
```

## How to run YDLIDAR SDK samples

linux:

```
$ ./ydlidar_test

YDLIDAR C++ TEST

Radar[ydlidar7] detected, whether to select current radar(yes/no)? :yes

0. ydlidar7

$ Please select the lidar port:0

0. 115200
1. 128000
2. 153600
3. 230400

$ Please select the lidar baud rate:3

0. false
1. true

$ Please select the lidar intensity:0
```

windows:

```
$ ydlidar_test.exe

YDLIDAR C++ TEST

Radar[ydlidar7] detected, whether to select current radar(yes/no?):yes

0. ydlidar7

$ Please select the lidar port:0

0. 115200
1. 128000
2. 153600
3. 230400

$ Please select the lidar baud rate:3

0. false
1. true

$ Please select the lidar intensity:0
```

## Console Display

You should see YDLIDAR's scan result in the console:

```
YDLIDAR C++ TEST

Radar[ydlidar7] detected, whether to select current radar(yes/no?):yes

0. ydlidar7

Please select the lidar port:0

0. 115200
1. 128000
2. 153600
3. 230400

Please select the lidar baud rate:3

0. false
1. true

Please select the lidar intensity:0

SDK Version: 1..3.7

LIDAR Version: 1.3.7
```



```
fhs_lock: creating lockfile:      18341

firmware: 521

[YDLIDAR] Connection established in [/dev/ttyUSB0]:

Firmware version: 2.0.9

Hardware version: 2

Model: G4

Serial: 2018042100000023

[YDLIDAR INFO] Current Sampling Rate : 9K

[YDLIDAR INFO] Current Scan Frequency : 7.000000Hz

received scan size: 1039

scan   system time: 1534400129245291000

scan   self time: 1534400129103710800

scan   frequency: 8.67053HZ

received scan size: 1231

scan   system time: 1534400129379541000

scan   self time: 1534400129232496800

scan   frequency: 7.31708HZ

received scan size: 1272

scan   system time: 1534400129530262000

scan   self time: 1534400129378863800

scan   frequency: 7.08105HZ

received scan size: 1295

scan   system time: 1534400129671749000

scan   self time: 1534400129519748800

scan   frequency: 6.95518HZ

^Csignal_handler(2)

received scan size: 1341

scan   system time: 1534400129839365000

scan   self time: 1534400129671106800

scan   frequency: 6.71642HZ
```

fhs\_unlock: Removing LockFile

## Lidar point data structure

data structure:

```
/// A struct for returning configuration from the YDLIDAR
struct LaserConfig {

    /// Start angle for the laser scan [rad]. 0 is forward and angles are measured clockwise when viewing YDLIDAR from the top.
    float min_angle;

    /// Stop angle for the laser scan [rad]. 0 is forward and angles are measured clockwise when viewing YDLIDAR from the top.
    float max_angle;

    /// Scan resolution [rad].
    float ang_increment;

    /// Scan resolution [ns]
    float time_increment;

    /// Time between scans
    float scan_time;

    /// Minimum range [m]
    float min_range;

    /// Maximum range [m]
    float max_range;

    /// Range Resolution [m]
    float range_res;

};

struct LaserScan {

    /// Array of ranges
    std::vector<float> ranges;

    /// Array of intensities
    std::vector<float> intensities;

    /// Self reported time stamp in nanoseconds
    uint64_t self_time_stamp;

    /// System time when first range was measured in nanoseconds
    uint64_t system_time_stamp;

    /// Configuration of scan
    LaserConfig config;

};
```

example:

```
for(size_t i =0; i < scan.ranges.size(); i++) {  
  
    // current angle  
    double angle = scan.config.min_angle + i*scan.config.ang_increment;  
  
    //current distance  
    double distance = scan.ranges[i];  
  
    //current intensity  
    int intensity = scan.intensities[i];  
  
}
```

code:

```
void LaserScanCallback(const LaserScan& scan) {  
  
    std::cout<< "received scan size: "<< scan.ranges.size()<<std::endl;  
  
    std::cout<< "scan    system time: "<< scan.system_time_stamp<<std::endl;  
  
    std::cout<< "scan    self time: "<< scan.self_time_stamp<<std::endl;  
  
    std::cout<< "scan    frequency: "<< 1000000000.0/scan.config.scan_time << "HZ"<<std::endl;  
  
    for(size_t i =0; i < scan.ranges.size(); i++) {  
  
        // current angle  
        double angle = scan.config.min_angle + i*scan.config.ang_increment;  
  
        //current distance  
        double distance = scan.ranges[i];  
  
        //current intensity  
        int intensity = scan.intensities[i];  
  
    }  
  
}
```

## Examples

samples in the file test.cpp/test1.cpp.

## SIMPLE USAGE

```
try {  
  
    LIDAR ydlidar;  
  
    LaserParamCfg cfg;
```

```

ydlidar.RegisterLIDARDataCallback(&LaserScanCallback);

ydlidar.UpdateLidarParamCfg(cfg);

while(ydlidar::ok()){

    try {

        ydlidar.spinOnce();

    }catch(TimeoutException& e) {

        std::cout<< e.what()<<std::endl;

    }catch(CorruptedDataException& e) {

        std::cout<< e.what()<<std::endl;

    }catch(DeviceException& e) {

        std::cout<< e.what()<<std::endl;

        break;
    }

}

}catch(TimeoutException& e) {

    std::cout<< e.what()<<std::endl;

}catch(CorruptedDataException& e) {

    std::cout<< e.what()<<std::endl;

}catch(DeviceException& e) {

    std::cout<< e.what()<<std::endl;

}

}

```

## Get Lidar List

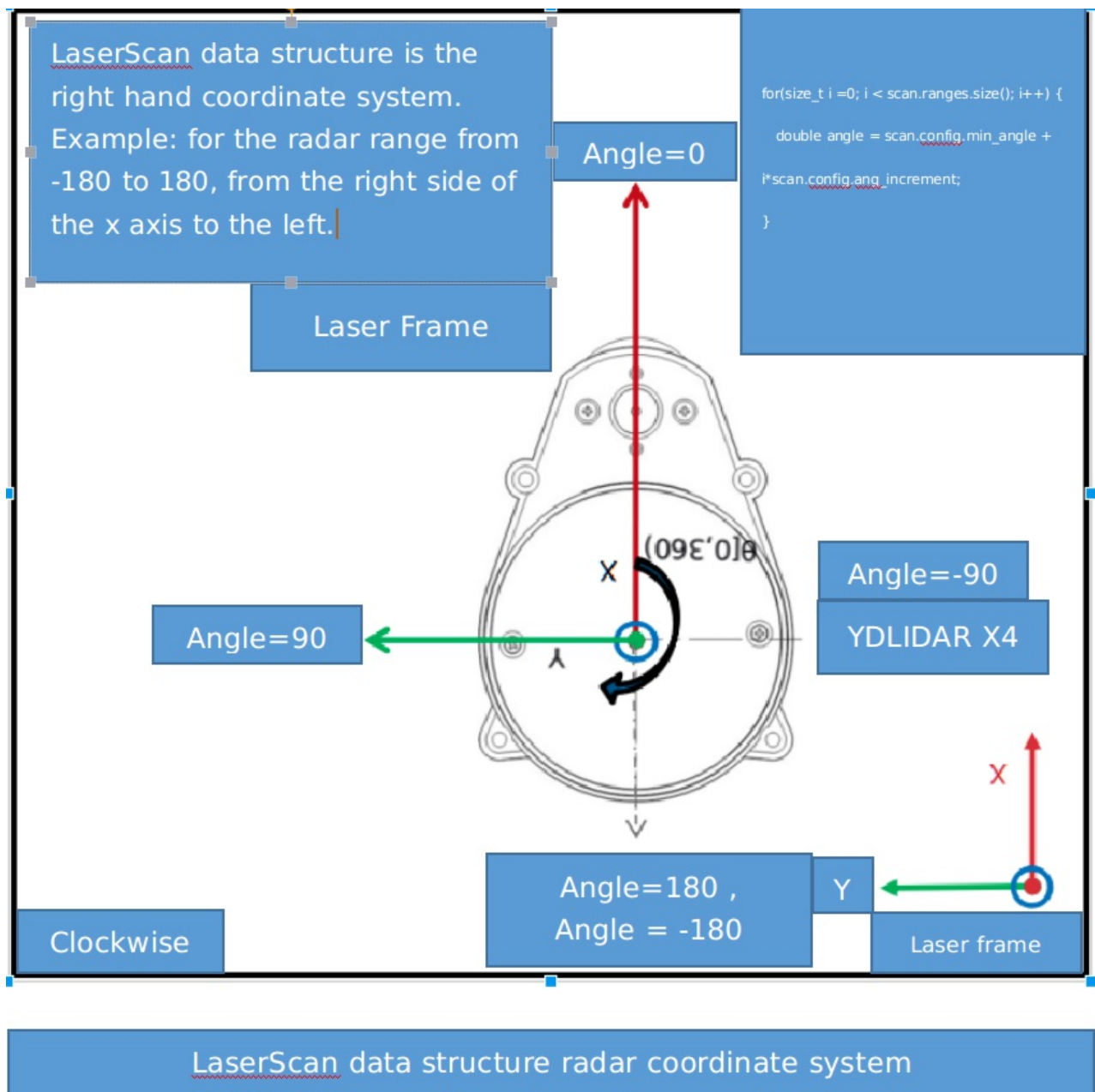
```

std::vector<string> ports = YDlidarDriver::lidarPortList();
for(std::vector<string>::iterator it = ports.begin(); it != ports.end(); it++) {
    printf("%s\n", (*it).c_str());
}

```

# Coordinate System

---



**The relationship between the angle value and the data structure in the above figure:**

```
double Angle = scan.config.min_angle + index*scan.config.ang_increment;
```

## Upgrade Log

2018-08-14 version:1.3.7

1. update sdk interface function.
2. add get lidar port list.

3. support mutil-lidar binding port.
4. support **for** configuring radar parameters through ini file.
5. the currend interface **is not** compatible **with** the old sdk.