## description for Parser.dll

### *File parser.h:*

#ifndef PARSER\_H

#define PARSER\_H

#include "parser\_global.h"

#include <QString>

class PARSERSHARED\_EXPORT Parser

{

public:

Parser();

enum Error{

NO\_ERRORS = 0,

FILE\_NOT\_FOUND = 1,

INVALID\_PACKET = 2

};

Error parse(QString const filename, QString &result);

};

#endif // PARSER\_H

### Discription:

* Parser(); It doesn’t require any input information;
* enum Error{

NO\_ERRORS = 0,

FILE\_NOT\_FOUND = 1,

INVALID\_PACKET = 2

};

* + description : list of all errors
  + parameters:
    - NO\_ERRORS = 0,( result of parsing without errors);
    - FILE\_NOT\_FOUND = 1,(error that file path is incorrect);
    - INVALID\_PACKET = 2 (error that packet is incorrect)
* Error parse(QString const filename, QString &result);
  + Description: parse packet (for example InfoFrame packet)
  + Retrun value: result
  + Parameters:
    - filename – the packet path (input data)
    - result – parsed packet info (output data)

### *File version.h:*

#ifndef PARSER\_H

#define PARSER\_H

#include "parser\_global.h"

#include <QString>

class PARSERSHARED\_EXPORT Version

{

public:

Version();

struct Info{

QString major;

QString minor;

QString specifications;

};

static Info getVersionInfo();

};

#endif // PARSER\_H

### Discription:

* Vectra(); It doesn’t require any input information;
* struct Info {

QString major;

QString minor;

QString specifications;

};

* + parameters:
    - QString major; ( major version of lib)
    - QString minor; ( minor version of lib)
    - QString specifications; (specifications, that was used in lib)
* static Info getVesionInfo();
  + Description: returns version of libulary
  + Retrun value: versionInfo

### eXample:

### *file mainwindow.cpp:*

...

QString packetInfo;

QString filename;

Parser parser;

Parser::Error res= parser.parse(filename, packetInfo);

switch (res)

{

case Parser::NO\_ERRORS: ui->result->setText(packetInfo); break;

case Parser::FILE\_NOT\_FOUND: ui->result->setText("FNF"); break;

case Parser::INVALID\_PACKET: ui->result->setText("Invalid packet");

break;

}

Version::Info versionInfo = Version::getVersionInfo();

ui->libVer->setText(versionInfo.major + versionInfo.minor);

ui->spec->setText(versionInfo.specifications);