RFID Library- Scan.Dll

This document will describe the various methods that can be used to communicate with the RFID scanner. Remember to add the Scan.dll file to the project, call “using Scan”, and call RFIDReader “something” = new RFIDReader(); Now it is possible to set up the comport, create a communication, and read tags scanned.

Example:

RFIDReader rr = new RFIDReader();

Then the buadrate, the commport the scanner is connected to is set, the timer interval when the scanner will read tags is set, the card type is set (use EPC), the RF power (use 20), and finally the freqType, since we are in Europe just use Europe.

rr. CommPortSetUp(115200, "COM5", 500, "EPC", 20, "Europe");

Now it is possible to connect with the method call rr.OpenCompConnection(), it returns true if connection is established.

To read multi tags, use rr.MultiTagRead(), however if it is wanted to read from a specific antenna first call rr.ReadFromAntenna(true, false, false, false), here it is set to only read from antenna 1. If all is set to true, all antennas are used, however they will store tags in the same list. To test what antennas are currently used, you can use rr.getAntInUse(), which returns a string.

To get the tag IDs read from an antenna, activate some event and use rr.TagList(), which will return a list of Strings of all ID read from the currently used antenna. If it is just wanted to get the number of tags read, just use rr.GetNumberOfTagID(). It is a good idea to clear the tag list after it is read, so after the tags have been processed use rr.clearTagIDlist().

Furthermore you can stop the read timer with, rr.StopEPCTimers() and restart it with rr.RestartEPCRead(). When the connection to the scanner is no longer needed use rr.CloseCommPort().

TCP connection have not been tested yet, so the methods are not described.

Methods.

```
/// <summary>
///  This method sets up the communcation for using a commport.
/// </summary>
/// <param name="buadrate">Sets the baudrate</param>
/// <param name="commport">The commport used to connect to the
Scanner</param>
/// <param name="timerInterval">Timer interval for when the scanner will
read tags
///in ms</param>
/// <param name="cardType">Sets the card type, either EPC or ISO</param>
/// <param name="rfRange">RF power for the scanner</param>
/// <param name="freqType">Freq_type should be set according to the
frequencie ///regulation, i.e. bandwith, e.g. Europe for Europe</param>

public void CommPortSetUp(int buadrate, String commport, int
timerInterval, String cardType, byte rfRange, String freqType)

///<summary>
///This method is used to open a connection to the RFID scanner.
/// Buadrate and compport should be set before openning the connection,
///<see cref="CommPortSetUp"/>.
/// </summary>
/// <returns>The method will return true iff the connection is
established</returns>

public bool OpenCompConnection()


///<summary>
/// This method closes the connection to the commport
///</summary>

public void CloseConnection()



/// <summary>
/// This method reads mutliple tags according to what time interval and
and card type set ///in, <see cref="commPortSetUp"/> or <see
cref="TCPSetUp"/>.
/// </summary>

public void MultiTagRead()
```

```
/// <summary>
/// This method stops the EPC timer that is used to read the tags
/// </summary>

public void StopEPCTimers()



/// <summary>
/// This method restarst the EPC timer, and tags will be read again
/// </summary>

public void restartEPCRead()


///<summary>
///This method returns a string list of tagID
///</summary>
///<returns>Returns a list of strings</returns>

public List<String> TagList()


/// <summary>
/// This method clears the Tag ID list
/// </summary>’

public void clearTagIDlist()


/// <summary>
/// Get the number of how many tags just read
/// </summary>
/// <returns>Number of tags</returns>

public int GetNumberOfTagID()

/// <summary>
/// This method returns a string of the currently set antennas
/// </summary>
/// <returns>A string of antennas</returns>

public String getAntInUse()
```

```
/// <summary>

/// This method sets what antenna the scanner shall read from. To
activate an antenna set it /// to true,
/// if more than one antenna is needed set all antennas need to true,
e.g. if ant1 ant3 is /// need  ReadFromAntenna(true,flase,true,flase)
// </summary>
///<param name="ant1">Set to true, to read from antenna 1</param>
///<param name="ant2">Set to true, to read from antenna 2</param>
///<param name="ant3">Set to true, to read from antenna 3</param>
///<param name="ant4">Set to true, to read from antenna 4</param>
///<returns>The method will return true iff the antenna is set</returns>

public bool ReadFromAntenna(bool ant1, bool ant2, bool ant3,
bool ant4)


/// <summary>
/// This method gets the current set RF-power, frequenzy type and prints
it in the console
/// </summary>
/// <returns>A string containing the RF-power and frequenzy</returns>

public String getRangeInfo()
```