# PitedaBarcodes - dll API

**Date: 29.04.2020**
**Last Edit Date: 29.04.2020**
**Last Edit By: Omer Sade**
**Writter: Omer Sade**

**Index:**



## About:

In this document you will able to find all the relevant data about the **PitedaBarcode.dll** that you'll need to use this dll for the purpose of reading and extracting the data from the barcode.
This dll use the **IronBarcode** library to read the barcode itself.

**THIS LIB IS BUILD AS A X64 PROGRAM**

## class LOG

### public class LOG()

### About:

This class will use as a log system for the app returning the user some errors & messages about things in the barcode reading process.

### Constructor:

**default constructor:**

public LOG() {}

### Class use:

**public string PACK_SIZE_ERROR** - return an error about unmatch pack size.

**public string BARCODE_AS_BEEN_READ_MSG** - return a message that this barcode already readed.

## class ReadBarcodes

**public class ReadBarcodes()**

**About:**

This class is holding all the necessary parts for the barcode reading process.

**Constructor:**

**default constructor:**

public ReadBarcodes() { }

**Global Variables:**

**public int GetLastReadedBCode ->** return the index number of the last barcode that have been readed.

**public BarcodeState GetBarcodeState ->** return an enum of the state of the last barcode that have been readed (NOT_READED, VALID, NOT_VALID).

**public List<int> GetReadedBarcodesIndexes ->** return a list of all the indexes of the barcodes that have been read from the pack.

### Enum Structures:

**public enum BarcodeState {**NOT_READED = 0, VALID, NOT_VALID **};**

This enum structure is use to return the state of the barcodes.
By default the barcodes will be set as 0 (NOT_READED).
If the barcode as been read properly it will set it as 1 (VALID)
and in a case where the program was unable to read the barcode it will set it as 2 (NOT_VALID ).

You can get the state of the last barcode that have been read from the function: **ResetIndexesList()**.

### Public Functions:

**public void ResetIndexesList():**

**function explain:**

This function reset the list of all the barcodes that have been read.

**Input:**

None

**return:**

void

**public Tuple<int, string> ReadSingleBarcode(Bitmap src_img):**

**function explain:**

This function is used to read a single barcode, extract the barcode index & the barcode pack size from the barcode data.

**Input:**

**Bitmap src_img ->** A Bitmap map image with of the barcode.

**return:**

**Tuple<int, string> ->** This function will return a Tuple object of Item1 = int & Item2 = string. The int item of the tuple will hold the barcode the have been read index and the string item will hold the barcode data.

If an error will occur in the function it will return int=0 & string = error or message using the log class.

```
return new Tuple<int, string>(0, log.BARCODE_AS_BEEN_READ_MSG);
```

```
return new Tuple<int,string>(0, log.PACK_SIZE_ERROR);
```

**null ->** This function will return null if the it wasn't able the read the barcode and it will set the state of the barcode as NOT_VALID.

**public string SortBarcodeDataList(ref List<Tuple<int, string>> barcodes_list)**

**function explain:**

Use this function to check if you finish to read the pack and to sort the list.
If the pack of the readed barcode is not full, it will return null.
If the list is in the same size of the pack, it will sort the list and build the full sorted data string & return it to the user.

**Input:**

**ref List<Tuple<int, string>> barcodes_list ->** A reference to the list of tuple<int,string> that will hold all the barcodes that have been readed.

**return:**

**string full_data ->** the full data that build from all the barcodes pack data sort in the right order of the data.

**null ->** If the list is not the same size of the barcodes pack (not all the barcodes as been read and it aint possible to build the

full data).