

# Mobile Wireless Printing with Pocket NiceLabel Version 3.2

## White Paper

Bar code labeling software for Windows® CE and Pocket PC

Version 20030206-05

# Index

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>General Introduction .....</b>                  | <b>3</b>  |
| <b>2</b> | <b>Pocket NiceLabel Overview.....</b>              | <b>4</b>  |
| <b>3</b> | <b>Using Pocket NiceLabel.....</b>                 | <b>5</b>  |
| 3.1      | General Overview.....                              | 5         |
| 3.2      | Software Installation .....                        | 5         |
| 3.3      | Label and Form Design on PC.....                   | 6         |
| 3.4      | Export to Pocket PC.....                           | 7         |
| 3.5      | Exchanging files with Windows CE.....              | 7         |
| 3.6      | Printing from Pocket NiceLabel software .....      | 7         |
| 3.6.1    | Selecting the printing method .....                | 7         |
| 3.6.2    | Direct printing from Windows CE device .....       | 8         |
| 3.6.3    | Distributed printing .....                         | 9         |
| 3.6.4    | Print Engine (ActiveX).....                        | 9         |
| <b>4</b> | <b>Pocket NiceLabel Programming Solutions.....</b> | <b>13</b> |

# 1 General Introduction

---

Extend label printing beyond the desktop and discover the small size and portability of Windows CE terminals for barcode label printing in the field and on demand!

Pocket NiceLabel™ is a software package for Windows® CE and Pocket PC, which brings the power of barcode label printing to portable Windows® CE compatible terminals. Pocket NiceLabel is an add-on module to the NiceLabel™ Suite of bar code label design and printing tools.

Pocket NiceLabel™ enables Windows® CE compatible computers and terminals to print barcode labels on any type of thermal printer supported by the NiceLabel printer drivers. Labels are designed in the easy-to-use and full-featured NiceLabel™ Pro software on a standard Windows® computer. There are no limitations to the label size and there are endless choices of label objects like text, graphics, barcodes, lines, etc. Designed labels are transferred to the Windows® CE terminal, where the user can print chosen number of labels. Labels do not need to have fixed content only, as new values for variable fields (text or barcode) can also be entered. Of course, you are not limited, in any way, to the number of stored label templates.

Pocket NiceLabel offers the industry's leading communications interface. Labels can be printed directly to the printer using a serial (COM port), Infrared (IR), Bluetooth or TCP/IP network (Wi-Fi) connection. Labels can also be printed via a distributed architecture. Pocket NiceLabel becomes a mobile client to the remote automated print server, NiceWatch; or from your custom application utilizing the ActiveX programming interface of Pocket NiceLabel directly on the device. The Pocket NiceLabel API provides a robust bolt-on label printing solution to mobile applications.

And that is not all! As it does for the desktop Windows® computers, NiceLabel™ brings the power of custom data entry forms to the Windows® CE terminal. On the basis of your requirements, number and complexity of label formats, you can design your own complete custom label printing solution using multiple forms to create an application that achieves your label printing goals. Forms can include label type selection, data selection, data input fields, menus, etc. All of this without the need to learn a custom programming language!

Pocket NiceLabel™ provides not just an efficient solution for every barcode labeling need, but also a reliable and user-friendly Windows® CE label printing solution based on specific label printing requirements.

## 2 Pocket NiceLabel Overview

---

Pocket NiceLabel is an application for Windows® CE, which brings the power of label printing to portable Windows® CE computers. Pocket NiceLabel is used in conjunction with NiceLabel Suite, a powerful labeling software package for desktop and enterprise-wide labeling applications.

Pocket NiceLabel can be used in three different ways:

**1** The first option is, that you design the label on a desktop computer, then download the label format to the Windows® CE terminal and use the device for direct label printing to the thermal printer. When you select the Print command, Pocket NiceLabel will prompt you to enter all variable data on that label format, then print the label. This solution is a one to one relationship where the output is designed for a single thermal printer family using a serial port, infrared, Bluetooth communication or a TCP/IP port.

Instead of printing directly from Pocket NiceLabel, you can simplify the mobile label printing process by using the NiceForm application. In NiceForm, you can design data entry form files, which are in fact custom user interfaces for entering and selecting data. Many different objects are available for form design and you can build your own command buttons, menus, custom data entry prompts, etc. Form files must be downloaded to Pocket NiceLabel along with the Label files. When using forms, Pocket NiceLabel will not prompt you for each variable data field. It reads the values from the form, and then connects the form to the label automatically. The variables on the label are populated with the necessary data from the form interface. This solution is also a one to one relationship where the output is designed for a single thermal printer family using a serial port, infrared, Bluetooth communication or a TCP/IP port.

**2** The second option is distributed printing and is a more flexible label printing option. The difference from the first option, direct printing, exists in the use of the Pocket NiceLabel software for outputting a JOB file (contains NiceCommands, instructions that describe the label printing process) versus the printer specific code. Pocket NiceForm directs the output to a specified network location using standard \\server\share syntax to a file drop trigger in NiceWatch or to a specified TCP/IP socket trigger in NiceWatch. NiceWatch, running as a Windows service on the server, monitors the folder and socket for a print request from Pocket NiceLabel. When the generated JOB file is accepted in the monitored folder or through the TCP/IP socket, NiceWatch processes the structured file and delivers the label output to any printer on the network.

This solution allows the printer type and format to be selected at the time of printing versus at the time of design. As NiceLabel Pro running on the print server actually prints the labels, there are no design limitations, all functionality in the application can be used, including database lookup tables and global serial numbers.

**3** The third option is to use Pocket NiceLabel as an ActiveX server, which provides other applications on the Windows CE terminal the ability to control label-printing functions. Your Windows CE application can use all functions provided by NiceLabel's Application Programmable Interface (API). This option requires a programmable change in your custom application. Pocket NiceLabel becomes a bolt-on label printing module for mobile applications. Controlling label printing functions from a data collection application on Windows CE has never been easier. Pocket NiceLabel is the end of hand-written, embedded printer code.

## 3 Using Pocket NiceLabel

---

### 3.1 General Overview

Before you can install Pocket NiceLabel on your Windows CE computer, you should be familiar with some technical information regarding your Windows CE computer. You must know which processor type and which Windows CE platform your terminal is using.

The current version of Pocket NiceLabel supports ARM, MIPS, x86, SH3 and SH4 processors. If your Windows CE computer is using any other processor type, please contact our technical support for more information ([support@nicelabel.com](mailto:support@nicelabel.com)).

Pocket NiceLabel supports platforms:

- Pocket PC 2002, Pocket PC and Palm PC (PPC): small pocket computers, without a keyboard, Windows CE version 3.0 is usually used for this platform
- Handheld PC 2000 (H/PC 2000): advanced Windows CE computers, with keyboard, Windows CE version 3.0 and above are usually used for this platform
- Handheld PC Pro (H/PC Pro): advanced Windows CE computers, with keyboard, Windows CE version 2.11 and above are usually used for this platform

Please refer to your Windows CE terminal documentation to find information about your platform or check Windows CE About screen.

Some Windows CE computer model examples:

- Philips NINO: MIPS, PalmSize
- HP Jornada 420: SH3, PPC
- Husky: MIPS, H/PC Pro
- Compaq iPAQ: ARM, Pocket PC 2002
- Symbol PPT 2800 series: ARM, Pocket PC and Pocket PC 2002
- Intermec 700 Series Mobile Computer: ARM, Pocket PC and Pocket PC 2002
- Intermec 6651 Pen Tablet Computer: MIPS, HPC 2000

### 3.2 Software Installation

A software application called **Windows CE Services** or **ActiveSync** must already be installed on your PC before you can install Pocket NiceLabel. It is the software for your desktop PC that will allow you synchronize data with the Windows CE terminal. Please refer to the Windows CE device Owners' Guide. Step-by-step instructions for setting up the synchronize application should be available there.

Once ActiveSync is installed, you can proceed with the installation of Pocket NiceLabel. The recommended install procedure is as follows:

- Go to folder X:\setup\WindowsCE on the NiceLabel CD, substitute X: with the letter of your CD-ROM drive.
- Start the SETUP.EXE application and follow the on-screen instructions.
- The last step is to select the proper platform of your Pocket PC device. The installation will use ActiveSync to transfer the necessary files to the device and complete the installation process. Verify the screen of the Windows CE terminal if

any further action is required there. The application is stored in the \Program Files\PocketNiceLabel folder on the device. The Pocket NiceLabel program group will appear in the Start menu of your Windows CE terminal.

Alternatively, you can omit running the SETUP.EXE application and transfer the correct .CAB file to the Pocket PC device yourself:

- Go to folder, which is appropriate for your Windows CE Platform (HPC 2000, HPC Pro, PalmSize, Pocket PC or Pocket PC 2002).
- Locate the file for your processor type. Identify it by the file extension.
- Transfer the file to the Windows CE device using ActiveSync.
- Tap the .CAB file in the File Explorer on the device to perform the installation. The application is stored in the \Program Files\PocketNiceLabel folder on the device. The Pocket NiceLabel program group will appear in the Start menu of your Windows CE terminal.

### 3.3 Label and Form Design on PC

Labels must be designed using the NiceLabel Pro software installed on a Windows desktop computer. Any type of objects may be used in the label design, just be sure that all of the variable fields on the label are using the printer's internal elements (fonts, barcodes...). Pocket NiceLabel is a print-only solution and does not have all of the power of a desktop version of NiceLabel, but it can be successfully used to print labels with prompted variables.

When the label design is completed, select the Export submenu from the File menu and click Export to Pocket PC. If the label contains any setting that is not compatible with Pocket NiceLabel, NiceLabel Pro will stop exporting and show you an informational message with a detailed description of the error and a suggestion to what should be done to solve the export problem. Correct the label accordingly and repeat the export procedure.

If you want to use a custom data entry form with Pocket NiceLabel, you must use the NiceForm software in NiceLabel Suite to design the interface. Design any form you like, just remember to use the correct window size for your Windows CE Terminals. NiceForm supports different Windows CE Platforms (Pocket PC 2002, Pocket PC, Palm-Size, HPC 2000 and HPC/Pro) to automatically prepare the forms for the selected terminal.

There are a few general design limitations while preparing the label to be printed from a mobile Windows CE device:

- You cannot use any variable Content Provider (FACT, HIBC, Lookup table, Expression, UCC/EAN-128 function...) other than a prompted variable.
- Use plain prompted variables. Database variables are not available at this time (version 3.3.1). Of course, you can use printer internal variables for date/time stamps and internal printer serialization counters.
- You cannot use functions, as NiceLabel Pro is not available at print-time to process the calculations and logic.
- Variable text elements on the label must be selected in one of the printer internal fonts. TrueType fonts can only be used for static text elements as NiceLabel Pro is not available at print-time to process the variable truetype text elements.
- Variable bar codes must be printed as printer internal elements.
- UCC/EAN-128 bar codes cannot be printed automatically as NiceLabel Pro is not available at print-time to properly process the check digits and Application Identifiers. You must make sure to provide the proper contents yourself, including check digits, AIs and FNC1 characters.
- Some of the form objects available in NiceForm on a Windows desktop computer are currently not supported in Pocket NiceLabel (database table, graphics).

- Print the label elements as printer internal elements as much as possible. It will speed up label printing as less data is sent to the printer.

### 3.4 Export to Pocket PC

This command saves the label in a format that can be read by Pocket NiceLabel on hand-held computers running Windows CE. Before executing this command, the label must already be saved to disk. The export command then prepares the file format to be compatible with Pocket NiceLabel application.

Two files will be generated:

- .LVR file, contains information about the variable fields on the label.
- .PNL file, contains the exported label layout with all of the elements in the printer specific code.

They will be saved to the same folder, where the original label file is stored.

### 3.5 Exchanging files with Windows CE

The Pocket NiceLabel label printing solution uses three types of files for printing. They all must be transferred to the Windows CE terminal. The required files are:

- .PNL & .LVR files (label files, created with the Export to PocketPC command in NiceLabel Pro)
- OFF files (Form files, created with NiceForm).  
Note! There is no special export function in NiceForm, because there is no need for that. Nice Form .OFF files can be used directly in Pocket NiceLabel. So there is no need for an Export command.

There are different methods of exchanging Label and Form files with Windows CE terminals (you can use the Windows CE Explorer, ActiveSync, etc). In all cases Windows CE Services must be installed on the PC computer before exchange can take place, but it is up to the user which method of transferring files will be used. ActiveSync allows for USB, serial, IR and TCP/IP wireless synchronization on a scheduled interval.

Refer to the Owner's Guide of your Windows CE computer for more information.

### 3.6 Printing from Pocket NiceLabel software

All the necessary files must be transferred to the Windows CE terminal, before you can start printing. The Pocket NiceLabel application is used to open a label file with extension .PNL.

When you want to print the label, you will get a chance to enter the values for all variable fields on the label in a similar dialog box as in NiceLabel Pro. The Pocket NiceForm application is used to open a file with extension .OFF and to print labels from your custom designed form file.

#### 3.6.1 Selecting the printing method

When you print the labels directly from the Pocket PC device, you have to select the port, where the label will be printed to. The procedure is the same for both, Pocket NiceLabel or

Pocket NiceForm software. The appropriate port can be set in Options|Preferences dialog box in both applications.

You can use the handheld to print the labels directly to the printer using different communication methods (COM port, IR port, TCP/IP port, Bluetooth). Or, you can enable distributed printing. Here the handheld is used as a mobile client to NiceWatch, the automated label-printing server running on a PC or server. Pocket NiceLabel will output a print request that describes the label layout and the printing process, encode this information to JOB file using NiceCommands and send the JOB file to a file trigger or a TCP/IP socket in NiceWatch. It will process the JOB file and print the labels to any available network printer.

|                    |   |
|--------------------|---|
| <b>COM port</b>    | Select this option if your printer is directly connected to the serial port on the Pocket PC device. Set the required connection parameters. The parameters should be the same on the handheld and on the printer.  |
| <b>LASP port</b>   | Select this option if you want to print over the built-in infrared port. Leave the IR Port setting to 0 for auto-detection. The printer IR receiver must be in line-of-sight range of the IR port on the handheld device.   |
| <b>TCP/IP port</b> | Select this option if you want to print to over the TCP/IP protocol directly to the printer. Please note, the printer must be equipped with an Ethernet or IEEE 802.11b wireless network card and be connected directly to the network. The printer must have its own IP address and the port must be known for the incoming connection to be accepted.   |
| <b>Bluetooth</b>   | Select this option if you want to print labels using the Bluetooth wireless protocol. In the Bluetooth properties on the Pocket PC device, you must define a mapping to a virtual COM port through the Bluetooth administration tool on the device. Then select the appropriate mapped COM port. The printer has to be in range of the Bluetooth printer to be effective. Check the operating manual of your Bluetooth devices for effective distances.   |
| <b>JOB file</b>    | Select this option if you do not want to print the label directly from the Pocket PC device. Instead a JOB file will be created and saved to the defined network drive location into a customized filename. The JOB file contains NiceCommands, describing the label to be printed, network printer to print to and data for the label. The file drop will be detected by NiceWatch running on a server then the JOB file will be processed and printed by the NiceWatch printing server. This is distributed printing. |
| <b>JOB TCP/IP</b>  | Select this option if you do not want to print the label directly from the Pocket PC device. Much like in the previous option, the JOB file is created and is then sent to NiceWatch using the TCP/IP protocol. NiceWatch accepts the incoming connection on a pre-defined TCP/IP socket and port and processes the NiceCommands from the file. Labels are printed by the NiceWatch label-printing server.  |

### 3.6.2 Direct printing from Windows CE device

#### Printing Directly from Label Files

If you choose to print Label files directly, select the Print command in the File menu. If the label has no variable data, you will be prompted only for number of labels to be printed. If

there are variable fields on the label, Pocket NiceLabel will prompt you for the values of each field. Do not forget to enter the quantity of labels to be printed. When you are finished entering the values, the printing will start with the selection of the print button.

### **Printing From Form Files (.OFF)**

When printing labels in Pocket NiceLabel, it is much easier for an end-user to use a simplified custom user interface to scan or enter data. Pocket NiceForm is used for running forms designed in NiceForm. With NiceForm for Windows, the designer is able to create custom label printing applications and save them to OFF files. Pocket NiceForm reads OFF file and shows the NiceForm application printing interface from which pre-defined actions will be run. All possible values for variables are usually already entered in the form files and user only taps the buttons. This can greatly simplify the inputting of data and reduce errors in data collection and label printing.

A custom data entry interface can be designed using NiceForm. In this option, the form is tied directly to a label that is designed for a specific printer. The operator enters or scans data into the data entry fields on the form and initiates a print job by selecting a print button. The output is sent directly to a specific thermal printer via a serial, infrared, Bluetooth connection, or a TCP/IP port from a wired or wireless network.

To open the form file, select the Open Form command in File menu. Then select Run form the file menu, or select the Play button on the taskbar. The form will start and you will see the layout of the form exactly as it was designed on the PC. All initializations of variables are handled in the background. You can chain forms together to create a multi-form label printing application with a menu structure that allows the operator to easily select the label to be printed.

### **3.6.3 Distributed printing**

A custom data entry interface can be designed using NiceForm. In this option, the form is tied directly to a label that is residing on a print server running the NiceWatch centralized print server in NiceLabel Suite. The operator enters or scans data into the data entry fields on the form, selects the printer to print to (any thermal or standard Windows printer on the network) and initiates a print job by selecting a print button. A JOB file is created and sent to NiceWatch through a file drop on the server or through a TCP/IP socket connection to NiceWatch. The JOB file contains information about which label to print, which printer to print to, variable data for the label and the quantity of labels to print. The designer also has the ability to customize the name of the JOB files to ensure unique identification of all print requests.

This option requires that the Windows CE terminal be connected to a wired or wireless Local Area Network to connect to NiceWatch, but provides a truly distributed printing solution from mobile devices. Please see the NiceLabel Suite documentation for more information on the NiceWatch automated print server.

### **3.6.4 Print Engine (ActiveX)**

Print Engine is a COM object (ActiveX server), which gives other applications on the Windows CE terminal the ability to control label-printing functions. Your Windows CE application can print labels using NiceLabel as the print server. Print Engine will handle the opening of LVR, PNL and OFF files and the printing of labels to printers on the following ports: TCP/IP, IR, Bluetooth and COM (serial). Pocket NiceLabel becomes the perfect bolt-on label printing engine for mobile applications. Controlling label printing functions from a data collection application on Windows CE has never been easier.

Pocket NiceForm with Print Engine also has ability to create a JOB file instead of printing directly to a port with a printer specific output file. The JOB file will be saved on hard disk or network drives where the NiceWatch print server will execute the NiceCommands from the JOB file or it will be sent over TCP/IP to NiceWatch on a print server.

The Print Engine (ActiveX server) exports several functions to give clients the ability to print by using the interface (API). Almost all functions will return an integer value indicating whether an error occurred. If the function will return an error number, the error message can be captured with the functions listed in **Table 1**.

**Table 1 Export Com Object Functions:**

| Function  | Description   |
|---|---|
| <p>GetLastError (integer ErrorID, string *szErrorStr, integer *MaxLen)</p>    | <p><b>GetLastError:</b> Function will give clients the ability to retrieve error description. ErrorID parameter will have to be set with error with failed function error number. Parameter szErrorStr will return error message description. Parameter MaxLen specifies the allocated size of szErrorStr. Parameter will have to be set with maximum length which error message can contain. Then it will return length of error message. Function will return 0 if everything is OK or number of message errors if error occurs. Possible error values are:</p> <p>1 - Buffer is too small to hold error description.<br/>                 2 - Resource string from ID not found.<br/>                 99 - Internal error.</p> |
| <p>GetErrorStack (string *szErrorStackStr, integer *MaxLen)</p>               | <p><b>GetErrorStack:</b> Function will also give clients the ability to retrieve error description. szErrorStackStr will receive CR/LF separated error descriptions from Print Engine. MaxLen will specify the allocated size of szErrorStackStr. szErrorStr parameter and function return value will act as parameters in GetLastError function.</p>   |
| <p>Init (integer LangID)</p>  | <p><b>Init:</b> The first function which has to be called will be an initialize function. Print Engine will allocate its needed memory and determine what language to use for returning strings. Default language and for now only supported language will be English (LangID=0). Function will return 1 if everything is OK or 0 if error occurs.</p>  |
| <p>IsDemo (String User, String Company, String NumOfUsers, String Serial)</p> | <p><b>IsDemo:</b> The function returns 1 if application runs in demo mode. If application is registered, the functions returns 0 and all parameters (User, Company, NumOfUsers and Serial) will have required data values.</p>  |
| <p>SetInputFile (String szFileName)</p>                                       | <p><b>SetInputFile:</b> The following function will check if LVR and PNL files exist. Then Print Engine will open the label. In szFileName parameter has to be written path and filename without an extension. Function will return 0 if everything is OK or error number if error occurs. If functions GetLastError or GetErrorStack with error number parameter are called, functions will return last error description or error stack descriptions.</p>   |

|  |  |
|--|--|
| <p>GetVariableCount( )</p>   | <p><b>GetVariableCount:</b> Function will return the number of variables from the selected label. If a label doesn't contain variables, function will return 0. This function will not return error number as other functions.</p>   |
| <p>GetVariable(integer Id, String* VariableName, String* PromptText, integer* LengthLimit, integer* bRequired)</p>       | <p><b>GetVariable:</b> Function will retrieve variable data. This function will have to be called for each variable. Id variable number will have to be entered in first parameter. Other parameters return variable data. Function will return 0 if everything is OK or error number if error occurs.</p>   |
| <p>SetVariable(integer Id, String VariableValue)</p>   | <p><b>SetVariable:</b> This function will have to be called to set the value of variable. Id parameter represents number of variable on selected label. Function will return 0 if everything is OK or error number if error occurs.</p>  |
| <p>SetComPort(integer PortNum, integer BPS, integer DataBits, integer Parity, integer StopBits, integer FlowControl)</p> | <p><b>SetComPort:</b> Print engine must get the information about the device to print on. This function must be called in order to use a device connected to COM port. Parameter values, which will have to be set, are: PortNum defines number of COM ports. Valid values are from 1 to 8; BPS describes the speed (baud rate) of the data port; DataBits defines the data bits which port should use. Valid values are from 4 to 8; Parity defines the parity check. Valid values are:</p> <p style="padding-left: 40px;">even = 1<br/>odd = 2<br/>none = 3<br/>mark = 4<br/>space = 5</p> <p>StopBits defines stop bits of the port. Valid values are 1, 1.5 and 2</p> <p>FlowControl defines the type of flow control, which port uses. Valid values are:</p> <p style="padding-left: 40px;">Xon/Xoff = 1<br/>Hardware = 2<br/>none = 3</p> <p>Function will return 0 if everything is OK or error number if error occurs.</p> |
| <p>SetIrPort(integer Lasp)</p>   | <p><b>SetIrPort:</b> Print engine must get the information about the device to print on. This function has to be called in order to use a device connected to IR port. LSAP will define the Logical Service Access Point value. If value is 0, auto detect will be performed, otherwise value will have to be from 1 to 127.</p> <p>More information about LSAP can be accessed on MSDN article titled: "<i>Windows CE 2.0</i>"</p>  |

|   |  |
|---|--|
|   | <p><i>Networking Offers Exciting Mobile Computing Possibilities</i>". Function will return 0 if everything is OK or error number if error occurs.</p>  |
| <p>SetTcpPort(String szIp, integer Port)</p>                            | <p><b>SetTcpPort:</b> Print engine must get the information about the device to print on. This function has to be called in order to use a device connected to TCP/IP port. szIP will represent the IP address of device where the Print Engine should print to. Port represents the port of the device where it will be waiting. Function will return 0 if everything is OK or error number if error occurs.</p>            |
| <p>SetJobFile (String Path, String FileName)</p>                        | <p><b>SetJobFile:</b> This function will set Print Engine to start building job file and it will remember the location and the name of the file. In Path parameter user will specify the location of the saved file. FileName will represent JOB file name. Function will return 0 if everything is OK or error number if error occurs.</p>  |
| <p>SetJobTCP (String Host, integer Port, String ValidationMessage )</p> | <p><b>SetJobTCP:</b> This function will connect Print Engine to server and set Print Engine to start building JOB file. Host will represent IP address of the device or host name. Port will represent the port where device will be waiting. ValidationMessage will be a string to authenticate if engine is connected to the right server. Function will return 0 if everything is OK or error number if error occurs.</p> |
| <p>StartPrint()</p>   | <p><b>StartPrint:</b> This function will print the constant part of label or labels (like header) without variable part. Function will return 0 if everything is OK or error number if error occurs.</p>   |
| <p>Print(integer Quantity)</p>  | <p><b>Print:</b> This function will print repeatable part of label. This function will also have to be called always if label is printed more times. Function will return 0 if everything is OK or error number if error occurs.</p>   |
| <p>EndPrint()</p>   | <p><b>EndPrint:</b> This function will print last part of label or labels (like footer). Function will return 0 if everything is OK or error number if error occurs.</p>   |
| <p>SetVirtualComPort(long VirtualPortNum)</p>                           | <p><b>SetVirtualComPort:</b> This function is used when printing to a Bluetooth device via Virtual COM port. VirtualPortNum is the number of the COM port to use. Note that some devices are bidirectional and use two ports (Inbound/Outbound). You must specify the outbound port for printing from Pocket NiceEngine.</p>   |

# 4 Pocket NiceLabel Programming Solutions

---

NiceLabel for Windows CE is composed of Pocket NiceLabel, Pocket NiceForm and Print Engine. The names of these applications are:

|   |
|---|
| Pocket NiceLabel - PNice.exe.             |
| Pocket NiceForm - PNForm.exe.             |
| Print Engine for Windows CE - PNPrint.dll |

The Print engine does not display any message boxes and does not have its own user interface. It is implemented as a component that works in the background. ActiveX clients receive the results via the exported functions return values.background.

Pocket NiceForm with Print Engine supports the printing of labels on printers using different ports: TCP/IP, IR, Bluetooth and COM (serial). In addition to this it will support the creation of a JOB file, which will be saved to a network folder or will be sent to a TCP/IP port in which the print requests will be executed with the NiceWatch automated print server. Information about JOB file settings are saved into OFF file, which is the form interface designed with the NiceForm application on a Windows desktop computer. Information about the variables and printers are saved in OFF file also. Because of this feature, LVR and PNL files will not be necessary. JOB file information saved in the OFF file contains the following parameters:

- If JOB file is written to a network folder, the parameters are:  
**Directory path and JOB file name.**

Directory path specifies the location of the saved file. JOB file name contains name, which can be created from fixed name, computer name, user name and (or) date/time stamp. These options will give user the ability to create a file name, which is unique in the network folder. This eliminates the risk of duplicate file names written to the same network folder, which would cause label requests to be missed.

- If JOB file is sent over a TCP/IP port, the parameters are:  
**TCP/IP Port or host, Port number, Validation message.**

First two parameters contain the NiceWatch TCP/IP server location (IP address). The third parameter, Validation message, is a string which must be identical to the Welcome message set in the NiceWatch TCP/IP socket trigger. Print engine checks this parameter to be sure that it is connected to the correct NiceWatch label printing server. This is an optional parameter and should only be defined if it is enabled in NiceWatch.

Print Engine (ActiveX server) will have exported functions settled into classes listed in **Table 2**.

**Table 2 Error Handler Class Functions**

| Functions     | Description   |
|---------------|---|
| GetLastError  | long GetLastError(long ErrorId, BSTR* szErrorStr, long* MaxLen) |
| GetErrorStack | long GetErrorStack(BSTR* szErrorStackStr, long* MaxLen)         |

**Table 3 Parser Class Functions**

| Functions        | Description   |
|------------------|---|
| SetInputFile     | long SetInputFile(BSTR szFileName)  |
| GetVariableCount | long GetVariableCount( )  |
| GetVariable      | long GetVariable(long Id, BSTR* VariableName, BSTR* PromptText, long* LengthLimit, long* bRequired) |
| SetVariable      | long SetVariable(long Id, BSTR VariableValue)   |

**Table 4 Printer Class Functions**

| Printer Class     | Description  |
|-------------------|--|
| SetComPort        | long SetComPort(long PortNum, long BPS, long DataBits, long Parity, long StopBits, long FlowControl) |
| SetLrPort         | long SetLrPort(long Lasp)  |
| SetTcpPort        | long SetTcpPort(BSTR szIp, Long Port)  |
| SetJobFile        | long SetJobFile (BSTR Path, BSTR FileName)   |
| SetJobTCP         | long SetJobTCP (BSTR Host, Long Port, BSTR ValidationMessage)  |
| StartPrint        | long StartPrint()  |
| Print             | long Print(long Quantity   |
| EndPrint          | long EndPrint()  |
| SetVirtualComPort | long SetVirtualComPort(long VirtualPortNum)  |

The Print Engine (ActiveX server) must be correctly registered as a COM object. Functions, which register and unregister the COM object are DllRegisterServer and DllUnregisterServer.

After the object is registered, the actual communication between the client and COM server application takes place. The application first creates an interface to our automated class IPocketNiceLabel.

The first function, which will need to be called, is Init. The Init function will create TParser, TPrinter and TErrorHandler classes and it will prepare the Print engine for integration.

The next function which needs to be called, is one of the following functions: SetComPort, SetLrPort, SetTcpPort, SetJobFile or SetJobTcp. These called functions will set the device that will be used for label printing.

The next function that needs to be called is SetInputFile. It will open the selected label to be printed.

With the function, GetVariableCount, the number of variables on an opened label will be returned.

If the number of variables is greater than 0 for each variable, the functions GetVariable and SetVariable will need to be used. The function, GetVariable, will return detailed information about the variable. Then the SetVariable variable value will need to be set.

After setting the variables, the StartPrint function needs to be called. This function will prepare the device for printing.

With the Print function labels will start to print. Function parameter Quantity represents how many equal labels are to be printed. Print function has to be called more times. Another possible call is setting the variable with GetVariable and SetVariable cycle before the print function is called.

With the EndPrint function, printing on device is ended.

If any of the functions returns an error, the GetLastError or GetErrorStack functions must be called. With these two functions, information about error or errors which occur are returned.